The 21st International Conference
on Computer Graphics and Vision

GraphiCon'2011

September 26–30, 2011
Moscow, Russia

Conference Proceedings

21-я Международная Конференция
по Компьютерной Графике и Зрению

ГрафиКон'2011

26–30 сентября, 2011
Москва, Россия

Труды Конференции

Московский государственный университет имени М.В. Ломоносова

МАКС ПРЕСС

МОСКВА – 2011

Organizing Committee:

Co-Chairs: Evgeny Moiseev (Lomonosov Moscow State University)
Andrey Krylov (Lomonosov Moscow State University)
Yuri Bayakovsky (Lomonosov Moscow State University)

Conference Organizing Committee

Chair:	Maxim Mizotin (Lomonosov Moscow State University)
Reviewing Coordinator:	Anton Konushin (Lomonosov Moscow State University)
Web site	Artem Semenov (Lomonosov Moscow State University)
Publishing/Printed Materials	Andrey Nasonov (Lomonosov Moscow State University)
Participants support	Roman Shapovalov (Lomonosov Moscow State University)
General Administration	Maxim Mizotin (Lomonosov Moscow State University)
Financial Manager	Maxim Fedyukov (Lomonosov Moscow State University)
Officials	Andrey Krylov (Lomonosov Moscow State University)

Advisory Board

Yuri Bayakovski (Lomonosov Moscow State University)
Valeriy Bobkov (Institute of Automation and Control Processes FEB RAS)
Victor Debelov (Novosibirsk State University)
Vladimir Galaktionov (Keldysh Institute of Applied Mathematics RAS)
Denis Ivanov (Corporation "Russian Systems")
Stanislav Klimenko (Institute of Computing for Physics and Technology)
Anton Konushin (Lomonosov Moscow State University)
Andrey Krylov (Lomonosov Moscow State University)
Timour Paltashev (National Research University ITMO)
Vadim Turlapov (Lobachevsky State University of Nizhni Novgorod)
Dmitriy Vatolin (Lomonosov Moscow State University)

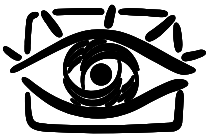
International Program Committee:

Chair: Anton Konushin (Lomonosov Moscow State University)

Members:

Valery Adzhiev (UK)
Boris Alpatov (Russia)
Pavel Babayan (Russia)
Mark Bannatyne (USA)
Olga Barinova (Russia)
Boris Barladyan (Russia)
Yuri Bayakovsky (Russia)
Alexander Belyaev (UK)
Sergey Belyaev (Russia)
Sergey Berezin (Russia)
Valery Bobkov (Russia)
Alexander Bondarev (Russia)
Alex Bovyrin (Russia)
Aleksey Chernavskiy (Russia)
Vladimir Chernov (Russia)
Dmitry Chetverikov (Hungary)
Sabine Coquillart (France)
Marc Daniel (France)
Victor Debelov (Russia)
Roman Durikovic (Slovakia)
Viktor Eruhimov (Russia)
Atanas Gotchev (Finland)
Vladimir Grishin (Russia)
Mikhail Iakobovski (Russia)
Andres Iglesias (Spain)
Denis Ivanov (Russia)
Vladimir Kaleyda (Russia)
Konstantin Kiy (Russia)
Nikolay Kim (Russia)
Stanislav Klimenko (Russia)
Anton Konushin (Russia)
Andrey V. Kopylov (Russia)
Vassili Kovalev (Belarus)
Alexander Krainov (Russia)
Andrey Krylov (Russia)
Victor Lempitsky (UK)
Alexey Lukin (Russia)
Tom Malzbender (USA)
Leonid Mestetskiy (Russia)
Vadim V. Mottl (Russia)
Karol Myszkowski (Germany)
Timour Paltashev (Russia)
Alexander Pasko (UK)
Denis Perevalov (Russia)
Alexander Reshetov (Russia)
Michael Rychagov (Russia)
Ilya Safonov (Russia)
Vladimir Savchenko (Japan)
Bogdan Savchynskyy (Ukraine)
Philipp Slusallek (Germany)
Sergey Sokolov (Russia)
Alexei Sourin (Singapore)
Daniel Thalmann (Switzerland)
Vadim Turlapov (Russia)
Oleg Ushmaev (Russia)
Natalia Vassilieva (Russia)
Dmitriy Vatolin (Russia)
Dmitriy Vetrov (Russia)
Yuri Vizilter (Russia)
Aleksey Voloboy (Russia)
Konstantin Vostryakov (Russia)
Dmitriy Yurin (Russia)

Address: GraphiCon-2011, Graphics&Media Lab, 7th floor, 701 room, Graphics&Media Lab VMK (2-Gum corpus), Lomonosov Moscow State University, Leninskie Gory, Moscow, Russia, 119991
Tel: +7 (495) 939-0190 E-mail: gc2011@graphicon.ru Web Site: http://gc2011.graphicon.ru/


The conference is organized by
Lomonosov Moscow State University,
Dept. of Computational Mathematics and Cybernetics,
Graphics & Media Lab,
Laboratory of Mathematical Methods in Image Processing

Sponsored by Russian Foundation for Basic Research

Supported by Russian Academy of Science

In cooperation with M.V.Keldysh Institute for Applied Mathematics of Russian Academy of Science

Sponsored by LLC IETP, the distributor for Xenics, The Imaging Source and Allied Vision Technologies

Supported by Nizhny Novgorod Foundation for Education and Research Assistance

The conference is devoted to the 300-year anniversary of M.V.Lomonosov.

Конференция посвящена 300-летию со дня рождения М.В.Ломоносова.

Contents

S4: Rendering

S5: Medical image processing

S6: Visualization

S7: Geometry processing

S8: Image enhancement

S9: Image segmentation and recognition

S10: Tracking, acquisition, face recognition

S11: Image matching

SR1: Biometry (In Russian)

Young Scientists School

Preface

Dear Participant, We would like to welcome you at GraphiCon'2011, a major international conference on Computer Graphics, Computer Vision, Image and Video processing in Russia. Following well established traditions, the 21st event will be hosted by Lomonosov Moscow State University on September 26–30, 2011.

This year we have a great program consisting of scientific papers, carefully selected by International Program Committee for oral and poster presentations and young scientist school. The International Program Committee was formed of 64 members representing 11 countries all over the world. Being top experts in the respective areas, all of them have done a tremendous job reviewing on average 4 papers out of 81 submitted. We express thanks to the committee members, who served at considerable personal sacrifice and with impressive collective wisdom. The final decision was based on at least two reviews of each manuscript and ended up with 46 works selected for oral presentation. Also we have decided to organize a young scientist school, where 17 papers will be presented as posters.

Keeping traditions, this year GraphiCon has not only scientific, but also extensive educational program consisting of 2 full-day tutorials and workshops for both undergraduate and PhD students, researchers and engineers of various companies that are interested in computer graphics topics.

We would like to thank our sponsors, volunteers organized by CS MSU Graphics and Media Lab, and everyone who made this event happen.

We do hope you will enjoy the conference,
Maxim Mizotin, on behalf of GraphiCon'2011 Organizing Committee

Technical section

GraphiCon'2011

September 26–30, 2011
Moscow, Russia

Single-Image Depth Map Estimation Using Blur Information

Dmitry Akimov
Department of Computational Mathematics and Cybernetics
Moscow State University, Moscow, Russia
dakimov@graphics.cs.msu.ru

Dmitry Vatolin
Department of Computational Mathematics and Cybernetics
Moscow State University, Moscow, Russia
dmitry@graphics.cs.msu.ru

Maxim Smirnov
YUVsoft Corp.
ms@yuvsoft.com

Abstract

This paper presents a novel approach for depth map estimation from a single image using information about edge blur. The blur amount at the edge is calculated from the gradient magnitude ratio of the input and re-blurred images. The final depth map can be obtained by propagating estimated information from the edges to the entire image using cross-bilateral filtering. Experimental results for real images and video sequences demonstrate the effectiveness of this method in providing a plausible depth map for 2D-to-3D conversion that generates comfortable stereo viewing.

Keywords: *Image Processing, Depth Map, Defocus Blur, Cross-bilateral Filtering.*

1. INTRODUCTION

3D images and videos have become more and more popular in everyday life, but the process of making 3D content is still very complicated and expensive. Many techniques for automatic 2D-to-3D conversion, and depth map estimation from 2D image sources plays an important role in this process.

Depth can be recovered either from binocular cues, such as stereo correspondence, or monocular cues, such as shading, perspective distortion, motion and texture. Conventional methods of depth map estimation have relied on multiple images, video sequences, calibrated cameras or specific scenes, or a huge training database. Stereo vision [3] measures disparities between a pair of images of the same scene taken from two different viewpoints, and it uses these disparities to recover the depth map. Shape from shading [4] offers another method for monocular depth reconstruction, but it is difficult to apply to scenes that do not have fairly uniform color and texture. Saxena et al. [2] presented an algorithm for estimating the depth map from numerous monocular image features using MRF modeling and a training database. The proposed algorithm in this paper focuses on a more challenging problem: recovering depth from a single defocused image captured by an uncalibrated camera without any prior information.

2. DEPTH ESTIMATION

The main part of the proposed algorithm is blur estimation at the edges and proper depth propagation from the edges to the entire image.

2.1 Defocus blur model

To estimate a blur value, the defocus blur model and special blur estimation method were introduced in [5].

Assume that the ideal edge is the step function $f(x)$ (Fig. 1(a)) with the jump at $x = 0$:

$$u(x) = \begin{cases} -1, x \le 0 \\ 1, x \ge 0 \end{cases}$$

$$f(x) = Au(x) + B$$

Here, A and B is the amplitude and offset of the edge, respectively.

The blurred edge is the function $i(x)$ (Fig. 1(b)), which is the convolution of $f(x)$ and $g(x,\sigma)$ where the latter is a Gaussian function with an unknown parameter σ.

$$i(x) = f(x) \otimes g(x,\sigma) = \int_{-\infty}^{+\infty} f(\tau) g(t-\tau) d\tau$$

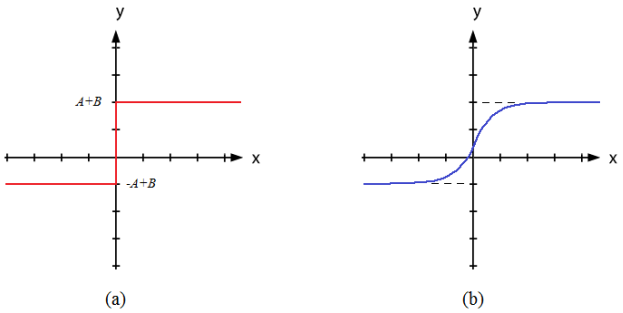


Figure 1: (a) Ideal edge function $f(x)$.
(b) Blurred edge function $i(x)$.

2.2 Blur value estimation

A blurred edge with unknown parameter σ is re-blurred using a Gaussian function with a known parameter σ_1.

$$i_1(x) = i(x) \otimes g(x,\sigma_1)$$

Consider the gradient magnitude of $i_1(x)$:

$$\nabla i_1(x) = \nabla\big(i(x) \otimes g(x,\sigma_1)\big) = \\ = \nabla\Big(\big((Au(x)+B) \otimes g(x,\sigma)\big) \otimes g(x,\sigma_1)\Big) = \\ = \frac{A}{\sqrt{2\pi(\sigma^2+\sigma_1^2)}} e^{-\frac{x^2}{2(\sigma^2+\sigma_1^2)}}$$

The gradient magnitude ratio of the re-blurred edge and the original edge is the function $B_i(x)$.

$$B_i(x) = \frac{\nabla i_1(x)}{\nabla i(x)} = \sqrt{\frac{\sigma^2}{\sigma^2+\sigma_1^2}} e^{\frac{x^2}{2\sigma^2} - \frac{x^2}{2(\sigma^2+\sigma_1^2)}}$$

The maximum of the function is at the edge location ($x = 0$):

$$B_i(0) = \frac{\nabla i_1(0)}{\nabla i(0)} = \sqrt{\frac{\sigma^2}{\sigma^2+\sigma_1^2}}$$

This function depends only on blur parameters, and the unknown σ can be estimated from the gradient magnitude ratio:

$$\sigma = \frac{\sigma_1}{\sqrt{B_i^2(0)-1}}.$$

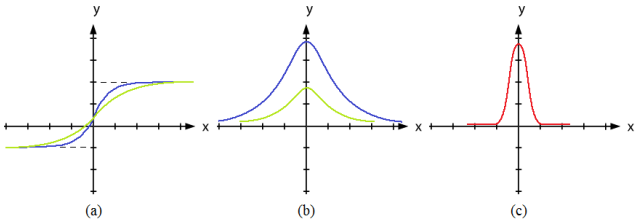


Figure 2: (a) Blurred edge $i(x)$ and re-blurred edge $i_1(x)$.
(b) Gradient magnitudes of $i(x)$ and $i_1(x)$.
(c) Gradient magnitude ratio function $B_i(x)$.

2.3 Blur value filtering

Blur values are estimated for each point of the image's edge mask. Assuming the entire scene in the image is beyond the camera's focal plane, the depth at the edges can be measured by the estimated parameter σ, forming a sparse map $D_0(x,y)$ and the mask of points with known depth values $Mask_0$. Because of noise in the image and inaccuracies in edge detection, however, the estimate of the sparse depth map $D_0(x,y)$ may contain some errors. To avoid such estimation errors, several prefiltering techniques are proposed.

The first step of filtration is outlier removal. Points with rare depth values relative to the depth value histogram are excluded from D_0 and $Mask_0$ (Fig. 3, row 1 and 2).

The second step is median filtering of D_0 according to mask and color difference. Figure 3 illustrates the result of median filtering; following noise reduction, two main depth profiles appeared.

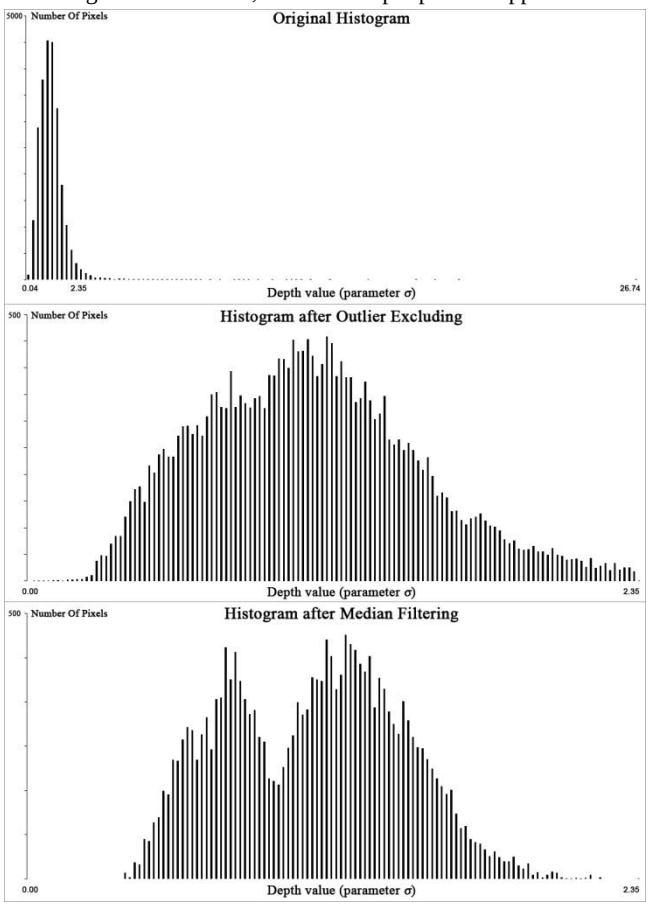


Figure 3: Histograms of depth values in sparse depth map.

2.4 Depth Propagation

The next important step in depth estimation is correct propagation of the blur information from the edges to the entire image. To achieve this, iterative cross-bilateral mask-oriented filtering was used:

$$D_i(x,y) = \\ = \frac{1}{2\pi\sigma_s\sigma_c\mathcal{N}} \iint\limits_{\Omega\cap Mask_{i-1}} D_{i-1}(\tau,\delta)e^{-\frac{(x-\tau)^2+(y-\delta)^2}{2\sigma_s^2}-\frac{(I(x,y)-I(\tau,\delta)^2}{2\sigma_c^2}} d\tau d\delta$$

where $D_i(x,y)$ is the depth map estimated at the iteration i of the propagation process, $Mask_i$ is the propagated region mask, σ_s and σ_c are spatial and color weights respectively and $\mathcal{N}$ is the normalization factor.

3. FINAL DEPTH MAP

The final depth map of the image is obtained from the propagated depth map. Following the sky detection process, the sky region is defined as the most distant area of the scene. The refined depth map is then filtered using cross-bilateral filtering.

3.1 Sky detection

To achieve more-accurate depth maps for outdoor scenes, a sky detection algorithm is introduced (Fig. 4). This algorithm is based on color segmentation of potential sky areas in HSV color space.

Figure 4: Original image and detected sky region.

The sky region is assumed to be in the upper area of the image, so the method recursively propagates the sky region from the image's upper edge. If a pixel color corresponds to a comparison criterion, the method marks it as a sky pixel and then checks the neighboring pixels. The algorithm stops if the current pixel doesn't correspond to the color criterion or if a color difference between the current and previous pixels exceeds a constant threshold.

The color criterion used here was chosen empirically and consists of three parts:

Figure 6: Comparison of proposed method (column 3) and [5] (column 2).

1. $(V(x,y) > 0.75) \cap (S(x,y) < 0.2)$
   to detect light tones in the sky

2. $(V(x,y) > 0.75) \cap (H(x,y) < 0.01)$
   to detect grey tones in the sky

3. $(V(x,y) > 0.3) \cap (H(x,y) > 0.4) \cap (H(x,y) < 0.8)$
   to detect blue, orange and yellow tones in the sky

4. RESULTS

In this work the Canny edge detection algorithm [1] was used and was tuned for uniform estimation of edges with different magnitudes.

The parameter σ_1 for re-blurring was set to $\sigma_1 = 0.5$. The estimated blur value at the edge location is then filtered using median filtering with a radius of 4 and using outlier exclusion according to histogram values.

For bilateral propagation, the parameters were set to $\sigma_s = 10$ and $\sigma_c = 7$, and Ω was defined as a circle of radius 30.

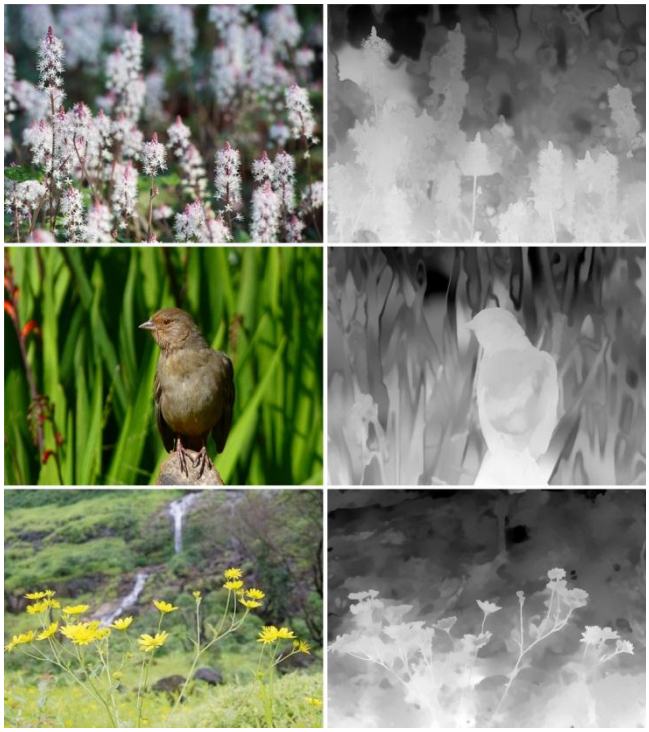


Figure 5: Source images (column 1) and estimated depth maps (column 2) using proposed method.

Figure 5 shows the results of the algorithm for real images. Rows 1 and 3 illustrates that the proposed method preserves all thin details of the source image, providing a correct depth map.

In figure 6 the proposed algorithm was compared with that of [5], which produced a coarse layered depth map with some layers not corresponding well to edges in the source image. By contrast, the results of the proposed algorithm are more accurate, continuous and detailed, all while preserving object boundaries.

5. CONCLUSION

This paper presents a novel approach for depth map estimation from a single image. The blur amount is calculated from the gradient magnitude ratio of the re-blurred and original images. A final depth map is then obtained using iterative cross-bilateral mask-oriented filtering. The proposed technique generates plausible depth maps that can be more accurate than those produced by existing methods.

6. ACKNOWLEDGMENTS

This work was partially supported by grant 10-01-00697-a from the Russian Foundation for Basic Research.

7. REFERENCES

[1] Canny J., "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Mach. Intell. 8(6) (1986) 679-698.*

[2] Saxena A., Chung S. H., and Ng A. Y., "Learning depth from single monocular images," *in Neural Information Processing System (NIPS) 18, 2005.*

[3] Scharstein D. and Szeliski R., "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," *International Journal of Computer Vision (IJCV), vol. 47, 2002.*

[4] Shao M., Simchony T., and Chellappa R., "New algorithms for reconstruction of a 3-d depth map from one or more images", *In Proc IEEE CVPR, 1988.*

[5] Zhuo S. and Sim T., "On the Recovery of Depth from a Single Defocused Image," *In Computer Analysis of Images and Patterns. Springer, 2009.*

About the author

Dmitriy Vatolin (M'06) received his M.S. degree in 1996 and his Ph.D. in 2000, both from Moscow State University. Currently he is head of the Video Group at the CS MSU Graphics & Media Lab. He is author of the book Methods of Image Compression (Moscow, VMK MGU, 1999), co-author of the book Methods of Data Compression (Moscow, Dialog-MIFI, 2002) and co-founder of www.compression.ru and www.compression-links.info. His research interests include compression techniques, video processing and 3D video technics: optical flow, depth estimation - from motion, focus, cues, video matting, background restoration, high quality stereo generation. His contact email is dmitry@graphics.cs.msu.ru.

Maxim Smirnov is the chief technology officer at YUVsoft IT company. Maxim has graduated from Saint Petersburg State University of Aerospace Instrumention in 2001 and received a Ph.D. in data compression from the same university in 2005. His research interests include forecasting of processes in technical and economic systems, video and universal data compression, stereo video. His contact email is ms@yuvsoft.com.

Dmitry Akimov is a student at Moscow State University's Department of Computational Mathematics and Cybernetics. His contact email is dakimov@graphics.cs.msu.ru.

Viewpoint Information

Xavier Bonaventura, Miquel Feixas, Mateu Sbert*
Graphics and Imaging Laboratory
University of Girona, Spain
{xavierb, feixas, mateu}@ima.udg.edu

Abstract

In this paper, we present a new perspective to quantify the information associated with a viewpoint. The starting point is twofold: a visibility channel between a set of viewpoints and the polygons of an object, and two specific information measures introduced respectively by DeWeese and Meister (1999) and Butts (2003) to evaluate the significance of stimuli and responses in the neural code. In our approach, these information measures are applied to the visibility channel in order to quantify the information associated with each viewpoint and are compared with both viewpoint entropy and viewpoint mutual information. A number of experiments show the behavior of the proposed measures in best view selection.

Keywords: *Viewpoint selection, Mutual information, Specific information.*

1. INTRODUCTION

In computer graphics, several viewpoint quality measures, such as viewpoint entropy and viewpoint mutual information, have been applied in areas such as best view selection for polygonal models [1, 2], scene exploration [3], and volume visualization [4, 5]. Best view selection is also a fundamental task in object recognition. Many works have demonstrated that the recognition process is view-dependent [6, 7, 8]. On the one hand, Tarr et al. [7] found that "visual recognition may be explained by a view-based theory in which viewpoint-specific representations encode both quantitative and qualitative features". On the other hand, Palmer et al. [6] and Blanz et al. [8] have presented different experiments demonstrating that observers prefer views (called canonical views) that avoid occlusions and that are off-axis (such as a three-quarter viewpoint), salient (the most significant characteristics of an object are visible), stable, and with a large number of visible surfaces.

In this paper, we propose two new viewpoint quality measures that are respectively derived from two different decompositions of mutual information proposed by DeWeese and Meister [9] and Butts [10] in the field of neural systems to quantify the information associated with stimuli and responses. First, we set an information channel between a set of viewpoints and the polygons of an object, and, then, we use those information measures to calculate the information associated with a viewpoint. Experimental results show the performance of these information measures to evaluate the quality of a viewpoint. This paper is organized as follows. In Section 2, we present the most basic information-theoretic measures and different decompositions of mutual information that are applied to quantifying the information associated with stimuli and responses. In Section 3, two new viewpoint information measures are presented. In Section 4, experimental results show the behavior of the proposed measures to select the best views. Finally, in Section 5, our conclusions and future work are presented.

*This work has been funded in part by grant number TIN2010-21089-C03-01 of Spanish Government and grant number 2009-SGR-643 of Generalitat de Catalunya (Catalan Government).

2. INFORMATION THEORY TOOLS

In this section, we present the most basic information measures and also three different ways of decomposing the mutual information between two random variables.

2.1 Basic Information Measures

Let X be a random variable with alphabet $\mathscr{X}$ and probability distribution $\{p(x)\}$, where $p(x) = Pr\{X = x\}$ and $x \in \mathscr{X}$. Likewise, let Y be a random variable taking values y in $\mathscr{Y}$. A communication channel $X \rightarrow Y$ between two random variables (input X and output Y) is characterized by a *probability transition matrix* (composed of conditional probabilities) which determines the output distribution given the input distribution [11].

The *Shannon entropy* $H(X)$ of a random variable X is defined by

$$H(X) = -\sum_{x \in \mathscr{X}} p(x) \log p(x). \quad (1)$$

Entropy measures the average *uncertainty* of a random variable X. All logarithms are base 2 and entropy is expressed in bits. The convention that $0 \log 0 = 0$ is used. The *conditional entropy* $H(Y|X)$ is defined by

$$H(Y|X) = \sum_{x \in \mathscr{X}} p(x) H(Y|x), \quad (2)$$

where $p(y|x) = Pr[Y = y|X = x]$ is the conditional probability and $H(Y|x) = -\sum_{y \in \mathscr{Y}} p(y|x) \log p(y|x)$ expresses the uncertainty of Y given x. $H(Y|X)$ measures the average uncertainty associated with Y if we know the outcome of X, and $H(X) \geq H(X|Y) \geq 0$.

The *mutual information* $I(X;Y)$ between X and Y is defined by

$$I(X;Y) = H(Y) - H(Y|X) = \sum_{x \in \mathscr{X}} p(x) \sum_{y \in \mathscr{Y}} p(y|x) \log \frac{p(y|x)}{p(y)}. \quad (3)$$

Mutual information expresses the *shared information* or dependence between X and Y. That is, mutual information expresses how much the knowledge of Y decreases the uncertainty of X, or vice versa. It can be seen that $I(X;Y) = I(Y;X) \geq 0$. If X and Y are independent, then $I(X;Y) = 0$.

2.2 Decomposition of Mutual Information

Given a communication channel $X \rightarrow Y$, mutual information can be decomposed in different ways to obtain the information associated with a state in $\mathscr{X}$ or $\mathscr{Y}$. Next, we present different definitions of information that have been analyzed in the field of neural systems to investigate the significance associated to stimuli and responses [9, 10].

For random variables S and R, representing an ensemble of stimuli $\mathscr{S}$ and a set of responses $\mathscr{R}$, respectively, mutual information (see Equation 3) is given by

$$I(S;R) = H(R) - H(R|S) \quad (4)$$

$$= \sum_{s \in \mathscr{S}} p(s) \sum_{r \in \mathscr{R}} p(r|s) \log \frac{p(r|s)}{p(r)}, \quad (5)$$

where $p(r|s)$ is the conditional probability of value r known value s, and $p(S) = \{p(s)\}$ and $p(R) = \{p(r)\}$ are the marginal probability distributions of the input and output variables of the channel, respectively. Note that capital letters S and R as arguments of $p(.)$ are used to denote probability distributions.

To quantify the information associated to each stimulus or response, $I(S;R)$ can be decomposed as

$$I(S;R) = \sum_{s \in \mathscr{S}} p(s) I(s;R) \quad (6)$$

$$= \sum_{r \in \mathscr{R}} p(r) I(S;r), \quad (7)$$

where $I(s;R)$ and $I(r;S)$ represent, respectively, the information associated to stimulus s and response r. Thus, $I(S;R)$ can be seen as a weighted average over individual contributions from particular stimuli or particular responses. The definition of the contribution $I(s;R)$ or $I(S;r)$ can be performed in multiple ways, but we present here the three most basic definitions denoted by I_1, I_2 [9], and I_3 [10].

Given a stimulus s, three information measures that fulfill (6) are:

- The *surprise* I_1 can be directly derived from (5), taking the contribution of a single stimulus to $I(S;R)$:

$$I_1(s;R) = \sum_{r \in \mathscr{R}} p(r|s) \log \frac{p(r|s)}{p(r)}. \quad (8)$$

  This measure expresses the surprise about R from observing s. It can be shown that I_1 is the only positive decomposition of $I(S;R)$ [9]. This positivity can be proven by the fact that $I_1(s;R)$ is the Kullback-Leibler distance [11] between $p(R|s)$ and $p(R)$.

- The *specific information* I_2 [9] can be derived from (4), taking the contribution of a single stimulus to $I(S;R)$:

$$I_2(s;R) = H(R) - H(R|s) \quad (9)$$

$$= -\sum_{r \in \mathscr{R}} p(r) \log p(r) + \sum_{r \in \mathscr{R}} p(r|s) \log p(r|s).$$

  This measure expresses the change in uncertainty about R when s is observed. Note that I_2 can take negative values. This means that certain observations s do increase our uncertainty about the state of the variable R.

- The *stimulus-specific information* I_3 (see [10] for a proof):

$$I_3(s;R) = \sum_{r \in \mathscr{R}} p(r|s) I_2(S;r). \quad (10)$$

  A large value of $I_3(s;R)$ means that the states of R associated with s are very informative in the sense of $I_2(S;r)$. That is, the most informative input values s are those that are related to the most informative output values r.

Similar to the above definitions for a stimulus s, the information associated to a response r could be defined. In the next sections, these information measures will be studied with more detail in the context of a communication channel between viewpoints and polygons.

The properties of positivity and additivity of these measures have been studied in [9, 10]. A measure is additive when the information obtained about X from two observations, $y \in \mathscr{Y}$ and $z \in \mathscr{Z}$, is equal to that obtained from y plus that obtained from z when y is known. While I_1 is always positive and non-additive, I_2 can take negative values but is additive, and I_3 can take negative values and is non additive. Because of the additivity property, DeWeese and Meister [9] prefer I_2 against I_1 since they consider that additivity is a fundamental property of any information measure.

3. VIEWPOINT QUALITY MEASURES

In this section, we present the main elements of the communication channel between viewpoints and polygons, and then we define the viewpoint information measures derived from the measures presented in Section 2.2.

3.1 Visibility Channel

In this section, we review the elements of an information channel between a set of viewpoints and the set of polygons of an object.

In [2], a viewpoint selection framework was proposed from an information channel $V \rightarrow Z$ between the random variables V (input) and Z (output), which represent, respectively, a set of viewpoints $\mathscr{V}$ and the set of polygons $\mathscr{Z}$ of an object. This channel is defined by a conditional probability matrix obtained from the projected areas of polygons at each viewpoint and can be interpreted as a visibility channel where the conditional probabilities represent the probability of seeing a determined polygon from a given viewpoint. Viewpoints are indexed by v and polygons by z. The three basic elements of the visibility channel are:

- Conditional probability matrix $p(Z|V)$, where each element $p(z|v) = \frac{a_z(v)}{a_t}$ is defined by the normalized projected area of polygon z over the sphere of directions centered at viewpoint v, $a_z(v)$ is the projected area of polygon z at viewpoint v, and a_t is the total projected area of all polygons over the sphere of directions. Conditional probabilities fulfil $\sum_{z \in \mathscr{Z}} p(z|v) = 1$. The background is not taken into account.

- Input distribution $p(V)$, which represents the probability of selecting each viewpoint, is obtained from the normalization of the projected area of the object at each viewpoint. The input distribution can be interpreted as the importance assigned to each viewpoint v.

- Output distribution $p(Z)$, given by $p(z) = \sum_{v \in \mathscr{V}} p(v) p(z|v)$, which represents the average projected area of polygon z.

From this visibility channel, different measures of viewpoint quality, such as viewpoint entropy [1] and viewpoint mutual information [2], have been defined in the past.

3.2 Viewpoint Information Measures

In this section, the information measures I_1, I_2 and I_3 presented in Section 2.2. are applied to the above visibility channel. Although this perspective of analyzing the viewpoint quality is new, it is important to note that I_1 is equivalent to viewpoint mutual information [2] and I_2 has a close relationship with viewpoint entropy [1].

Given the visibility channel $V \rightarrow Z$, the *viewpoint information* is defined in the following three alternative ways:

- From (8), the *viewpoint information* I_1 of a viewpoint v is defined as

$$I_1(v;Z) = \sum_{z \in \mathscr{Z}} p(z|v) \log \frac{p(z|v)}{p(z)}. \quad (11)$$

  Observe that I_1 coincides with the viewpoint mutual information defined in [2]. The lowest value of I_1 (i.e., $I_1(v;Z) = 0$) would be obtained when $p(Z|v) = p(Z)$. This means that the distribution of projected areas at a given viewpoint ($p(Z|v)$) would coincide with the average distribution of projected areas from all viewpoints ($p(Z)$). In this case, the view is considered maximally *representative*. Thus, while the most surprising views correspond to the highest I_1 values, the most

representative ones correspond to the lowest I_1 values. The best viewpoint is defined as the one that has the lowest value of I_1 (i.e., maximum representativeness).

- From (9), the *viewpoint information* I_2 of a viewpoint v is defined as

$$\begin{aligned} I_2(v;Z) &= H(Z) - H(Z|v) \\ &= -\sum_{z\in\mathscr{Z}} p(z)\log p(z) + \sum_{z\in\mathscr{Z}} p(z|v)\log p(z|v). \end{aligned} \tag{12}$$

  While the highest value of I_2 would correspond to a viewpoint that could only see one polygon, the lowest value of I_2 would be obtained if a viewpoint could see all polygons with the same projected area. In this case, the view is maximally *diverse*. The best viewpoint is defined as the one that has the lowest value of I_2 (i.e., maximum diversity).

  Specific information $I_2(v;Z)$ is closely related to viewpoint entropy, defined as $H(Z|v)$ [1, 2], since $I_2(v;Z) = H(Z) - H(Z|v)$. As $H(Z)$ is constant for a given mesh resolution, $I_2(v;Z)$ and viewpoint entropy will essentially have the same behavior in viewpoint selection because the highest value of $I_2(v;Z)$ corresponds to the lowest value of viewpoint entropy, and vice versa. An important drawback of viewpoint entropy is that it goes to infinity for finer and finer resolutions of the mesh (see [2]), while I_2 presents a more stable behavior due to the normalizing effect of $H(Z)$ in (12). The advantage of I_2 against viewpoint entropy could be appreciated in areas such as object recognition and mesh simplification. In the first case, the stable behavior of I_2 would enable us to compare the obtained values for objects with different mesh resolutions and, in the second case, I_2 would take into account the variation of $H(Z)$ in the simplification process.

- From (10), the *viewpoint information* I_3 of a viewpoint v is defined as

$$I_3(v;Z) = \sum_{z\in\mathscr{Z}} p(z|v) I_2(V;z), \tag{13}$$

  where $I_2(V;z)$ is the *specific information of polygon* z given by

$$\begin{aligned} I_2(V;z) &= H(V) - H(V|z) \\ &= -\sum_{v\in\mathscr{V}} p(v)\log p(v) + \sum_{v\in\mathscr{V}} p(v|z)\log p(v|z). \end{aligned} \tag{14}$$

  A high value of $I_3(v;Z)$ means that the polygons seen by v are very informative in the sense of $I_2(V;z)$. The most *informative* viewpoints are considered as the best views and correspond to the viewpoints that see the highest number of maximally informative polygons.

As we have seen above, $I_1(x;Y)$, $I_2(x;Y)$, and $I_3(x;Y)$ represent three different ways of quantifying the information associated with a viewpoint v. Observe that we consider that the best views correspond to the lowest values of I_1 and I_2, and the highest values of I_3; and the contrary for the worst views. That is, the goodness of a viewpoint is associated with its representativeness (minimum I_1), diversity (minimum I_2), and informativeness (maximum I_3). The word 'informativeness' is used here to express the capability of I_3 to capture information from the polygons of the object. Another aspect to take into account is that the concept of 'best' or 'worst' is relative to the objective we pursue. Thus, for instance, the 'worst' view in the sense of I_2 could be used to select the view with the lowest diversity, such as the one that better shows the structure of a molecule (see [12]).

	Number of polygons	**Computational cost**
Coffee cup	10732	3526 ms
Horse	43571	3650 ms
Ship	48811	3822 ms
Lady of Elche	51978	3946 ms

Table 1: Number of polygons of the models used and computational cost of the preprocess step for each model in miliseconds.

4. RESULTS

In this section, the behavior of I_1, I_2, and I_3 is analyzed. To calculate these measures, we need to obtain the projected area of every poligon for every viewpoint, and these areas will enable us to obtain the probabilities of the visibility channel ($p(V)$, $p(Z|V)$, and $p(Z)$). In this paper, all measures have been computed without taking into account the background, and using a projection resolution of 640×480. In our experiments, all the objects are centered in a sphere of 642 viewpoints built from the recursive discretisation of an icosahedron and the camera is looking at the center of this sphere. To obtain the viewpoint sphere, the smallest bounding sphere of the model is obtained and, then, the viewpoint sphere adopts the same center as the bounding sphere and a radius three times the radius of the bounding sphere.

In Table 1 we show the number of polygons of the models used in this section and the cost of the preprocess step, i.e., the cost of computing the projected areas $a_z(v)$ and a_t. To show the behavior of the measures, the sphere of viewpoints is represented by a color map, where red and blue colors correspond respectively to the best and worst views. Remember that a good viewpoint corresponds to a low value of I_1 and I_2, and to high value of I_3. Our tests were run on a Intel© Core™ i5 430M 2.27GHz machine with 4 GB RAM and an ATI Mobility Radeon™ HD 5470 with 512 MB.

To evaluate the performance of the viewpoint quality measures, four models have been used: a coffee cup, a horse, the Lady of Elche, and a ship. Figure 1 has been organized as follows. Rows (i), (iii) and (v) show, respectively, the best (columns (a-d)) and worst (columns (e-h)) views corresponding to I_1, I_2 and I_3, and rows (ii), (iv), and (vi) show the viewpoint spheres corresponding to the views shown in rows (i), (iii) and (v), respectively.

While the best views selected by I_1 show a global view of the object, the best views obtained by I_2 capture the maximum number of polygons in a balanced way (i.e., with a similar projected area). This means that I_2 has a high dependence of the resolution of the mesh, trying to see the areas with a finer discretization. On the contrary, it has been shown in [2] that I_1 is very robust with respect to the variation of the mesh resolution. The behavior of I_3 is very different of the one of I_1 and I_2 because the view with maximum I_3 tries to see the most informative polygons, that in general are placed in the most occluded, salient, and complex areas of the object. To better appreciate the behavior of I_3, the best and worst views (see row(v)) show the degree of informativeness of each polygon using a thermal scale, from blue (minimum information) to red (maximum information). Thus, it can be easily seen how I_3 selects the views with the highest informativeness. It is also important to note that a similar view can be considered as the best for one measure and the worst for another. See for instance the best and worst view of the coffee cup for I_2 and I_1, respectively (Figures 1(iii.b) and 1(i.f)), and the best and worst view of the horse for I_3 and I_1, respectively (Figures 1(v.c) and 1(i.g)).

5. CONCLUSIONS

In this paper, we have presented a new perspective based on the decomposition of mutual information to study the quality of a viewpoint. Two measures of specific information introduced in the field

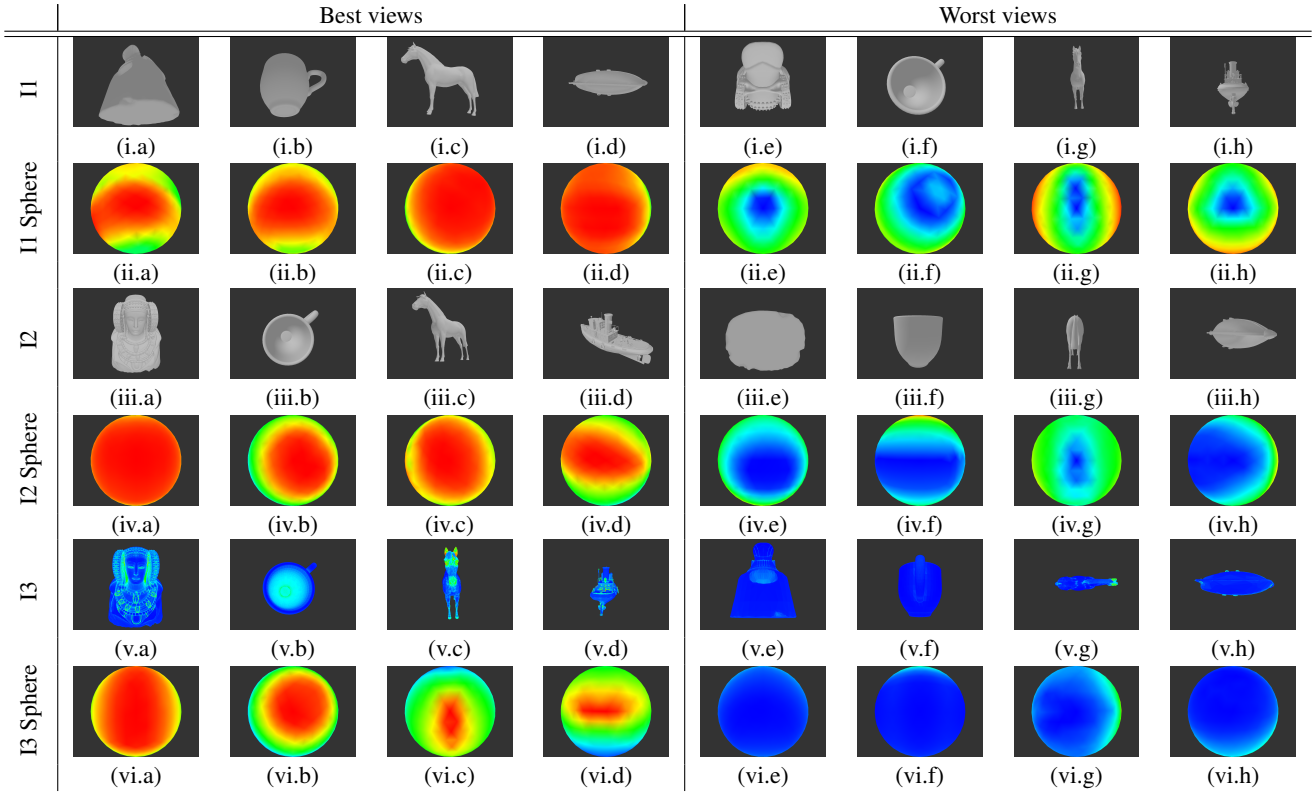


Figure 1: Rows (i), (iii), and (v) show, respectively, the best (a-d) and the worst (e-h) views of four models, obtained with I_1, I_2, and I_3. Rows (ii), (iv), and (vi) show, respectively, the viewpoint spheres corresponding to the views shown in rows (i), (iii), and (v).

of neural systems have been adapted to quantify the information associated with a viewpoint. These measures have been compared with viewpoint entropy and viewpoint mutual information, and different experiments have shown their performance in best view selection. The concepts of surprise, diversity, and informativeness associated with a viewpoint have been also discussed. Further research will be done to analyze the use of the new measures to select N best views, to explore a scene, and to compute the information associated with the polygons of an object.

6. REFERENCES

[1] Pere P. Vázquez, Miquel Feixas, Mateu Sbert, and Wolfgang Heidrich, "Viewpoint selection using viewpoint entropy," in *Proceedings of Vision, Modeling, and Visualization 2001*, 2001, pp. 273–280.

[2] Miquel Feixas, Mateu Sbert, and Francisco González, "A unified information-theoretic framework for viewpoint selection and mesh saliency," *ACM Transactions on Applied Perception*, vol. 6, no. 1, pp. 1–23, 2009.

[3] Dmitry Sokolov, Dimitri Plemenos, and Karim Tamine, "Methods and data structures for virtual world exploration.," *The Visual Computer*, vol. 22, no. 7, pp. 506–516, 2006.

[4] Udeepta D. Bordoloi and Han-Wei Shen, "Viewpoint evaluation for volume rendering," in *IEEE Visualization 2005*, 2005, pp. 487–494.

[5] Ivan Viola, Miquel Feixas, Mateu Sbert, and M. Eduard Gröller, "Importance-driven focus of attention," *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 5, pp. 933–940, 2006.

[6] S.E. Palmer, E. Rosch, and P. Chase, "Canonical perspective and the perception of objects.," *Attention and Performance IX*, pp. 135–151, 1981.

[7] M.J. Tarr, H.H. Bülthoff, M. Zabinski, and V. Blanz, "To what extent do unique parts influence recognition across changes in viewpoint?," *Psychological Science*, vol. 8, no. 4, pp. 282–289, 1997.

[8] V. Blanz, M.J. Tarr, and H.H. Bülthoff, "What object attributes determine canonical views?," *Perception*, vol. 28, pp. 575–599, 1999.

[9] Michael R. Deweese and Markus Meister, "How to measure the information gained from one symbol," *Network: Computation in Neural Systems*, vol. 10, no. 4, pp. 325–340, November 1999.

[10] Daniel A Butts, "How much information is associated with a particular stimulus?," *Network: Computation in Neural Systems*, vol. 14, pp. 177–187, 2003.

[11] Thomas M. Cover and Joy A. Thomas, *Elements of Information Theory*, Wiley Series in Telecommunications, 1991.

[12] Pere P. Vázquez, Miquel Feixas, Mateu Sbert, and Antoni Llobet, "Realtime automatic selection of good molecular views," *Computers & Graphics*, vol. 30, no. 1, pp. 98–110, 2006.

Interactive flower modeling with 3Gmap L-systems

Olga Petrenko, Olivier Terraz, Mateu Sbert, Djamchid Ghazanfarpour*
Graphics and Imaging Laboratory
University of Girona, Spain
University of Limoges, France
lelya.fleur@gmail.com, olivier.terraz@unilim.fr, mateu@ima.udg.edu, djamchid.ghazanfarpour@unilim.fr

Abstract

Flowers have quite an intricate structure consisting of numerous components which, in turn, have an enormous variety of shapes. Therefore, it is not an easy task for computer graphics to simulate such kind of natural phenomena. We propose in this paper an application of the 3Gmap L-systems to flower modeling by growth simulation. Our approach combines L-systems grammar writing with interactive control of parameter settings. The L-systems are used for creating the entire model, with stems,stamens, petals, leaves, etc., by simply operating with 3Gmap volumes. The presented contributions will make the task of a user more obvious and intuitive enabling her/him to create more accurate models. Moreover the way the model is built allows us to take into account its internal structure. As the flower tissue is non-homogeneous, the possibility of obtaining its internal composition could be quite useful for rendering, allowing for instance to render more accurate subsurface scattering.

Keywords: *Geometric volume modeling, L-Systems, Natural phenomena, Flower modeling.*

1. INTRODUCTION

Men's habitat is a green carpet of plants covering our earth. And the most impressive among them are flowers. But besides its enchanted beauty they have quite intricate structure consisting of numerous components which, in its turn, have an enormous variety of shapes. Therefore, it is not an easy task for computer graphics to simulate such kind of natural phenomena.

We can distinguish two different approaches to flower simulation. The first one is aiming at getting a plausible model while the botanical correctness is usually disregarded. Here the task of modeling is undertaken mainly by the user describing a plant structure and its components and defining the required parameters. The degree of realism depends on the users skills. This approach is quite intuitive for a common user, but has the inconvenience of creating each sample from scratch in case of generating a variation of slightly different flowers [10], [11].

The other approach could be referred to as procedural modeling, which tries to provide biologically faithful and visually realistic models [22], [5], [15]. Most of these approaches are based on L-systems, which can generate complicated multicellular structures from a small number of rules [22], [5]. They are able to get a lot of flower samples based on a single grammar by simply changing the parameter values. Although these methods can provide impressive results, the underlying algorithms are not so intuitive for common users.

An analysis of previous work points to look for some kind of symbiosis between these two groups of approaches in order to simplify the task of the user and at the same time to retain the realism of the models. Pursuing this goal we propose in this paper an application of the 3Gmap L-systems: flower modeling by growth simulation. Our approach combines L-systems grammar writing with interactive control of parameter settings. Here the L-systems, in contrast with previously proposed models, are used for creating the entire model, with all its components by simply operating with subdivision of volumes, namely 3Gmaps [12]. The user can control the final result by interactively setting the parameters of the grammar. The used L-systems grammars have a nested structure allowing to combine several grammars which represent the different flower organs. These contributions make the task of a user more obvious and intuitive which in turn enables to create more accurate models. In addition, the way the model is built allows us to take into account its internal structure. As the flower tissue is non-homogeneous, this can be quite useful to render more accurate subsurface scattering.

2. PREVIOUS WORK

Flower modeling methods are not numerous and belong to a bigger research field such as plant modeling, which origins are traced back to L-systems introduced in 1968 by Aristid Lindenmayer. He proposed a formal description of plant development as a string rewriting mechanism which has a recursive nature and leads to a self-similarity in plants. Later on L-systems were extended to several geometric interpretations which were used by computer graphics as a diversified tool for plant modeling [22], [5], [15].

Plant modeling tools such as AMAP [23] and LIGNUM [16] provide a wide range of models and take into account knowledge about plant architecture. Lintermann and Deussen [3] proposed a modelling method that allows easy generation of many types of objects that have branching structures. In this approach, components encapsulate data and algorithms to generate plant elements. Although the L-systems are not explicitly used, still the architectural models described above follow the basic principles of plant development.

In [22], [19], [21] different types of L-systems were elucidated at length and accompanied with the modeling software L-studio [19] and the Virtual Laboratory [5]. These tools enabled to model a wide range of structures and developmental processes in plants by operating with the L-system-based languages (cpfg and lpfg). Yet the shapes of plant organs, represented mostly as predefined surfaces or generalized cylinders [9], are specified by the user and then are incorporated into a plant model. In [7], [8] flowers were described as configurations of modules in space. In [22], [6] such examples as sun flower head, zinnias, water lily and roses were performed using phyllotaxis. Peiyu [15] proposed a flower model using L-systems which represented the topologic information of plant flower and Bezier surfaces for depicting its geometric information. String L-systems are applied to model a wide variety of plants [20], [21]. However they are of one-topological dimension, even if 3D geometrical features are incorporated into a model. Many shapes in nature can only be described by two or three topological dimensions. Thus, Prusinkiewicz and Lindenmayer described in [22] map L-systems and cellwork L-systems which were mainly used for mod-

*This work has been funded in part by grant number TIN2010-21089-C03-01 of Spanish Government and grant number 2009-SGR-643 of Generalitat de Catalunya (Catalan Government).

eling of cellular layers. These methods provide quite realistic results. However, it is quite difficult to specify L-system grammars and there is not enough control over the generated topologies. In [17], [24] 2Gmap and 3Gmap L-systems address some of the limitations of the previously described methods. These approaches are applied to model realistic leaves and wood. 2Gmap and 3Gmap L-systems are based on two and three-dimensional generalized maps [12], which could be controlled by the operations associated with production rules. The direct use of high level operations on surfaces and volumes simplifies model specification and the use of adjacency relations between volumes allows context-dependent behaviors.

Other methods have also found its niche in the plant modeling area, since the L-systems specification is not so intuitive for the common user. Sketch-based modeling techniques allow a user to easily create a rough model from several strokes. The work of Igiri [10], [11] is an interactive modelling system for flower composition. Here the task of the user is quite easier and takes less time, but still we cannot reckon on creating the models of complicated structures with botanical correctness, neither cannot consider the obtained model as a sample for creating the huge diversity of individuals.

Taking into account the benefits of the previous approaches some methods were proposed, such as [14], [18], [13]. Here the L-systems are mixed with interactive methods, such as Sketch-based or 3D gesture modeling or simply interactive control of parameter values performing bending and pruning branches and arranging and clipping leaves and flowers. Anastacio and Prusinkiewicz [1] propose a combination between sketch-based modeling and L-system, where construction lines are employed to parameterize global features of L-system models.

3. BIOLOGICAL ASPECTS OF FLOWERS

From the biological point of view, a flower can be pictured as a short stem (the receptacle) which holds the components in sequence. At the very tip of this stem are the female organs (carpels), which are surrounded by the male organs (stamens). The outside part of the flower is enveloped by petals and sepals. If we go further into analysis of flower, we can find that the internal structure of its organs is not homogeneous, which plays an important role in the way they absorb and reflect light. Leaves and petals tissues consist of different layers of cells, which have their own properties defining the variety of surfaces and colorations (Figure 1b, 1c). As we can see

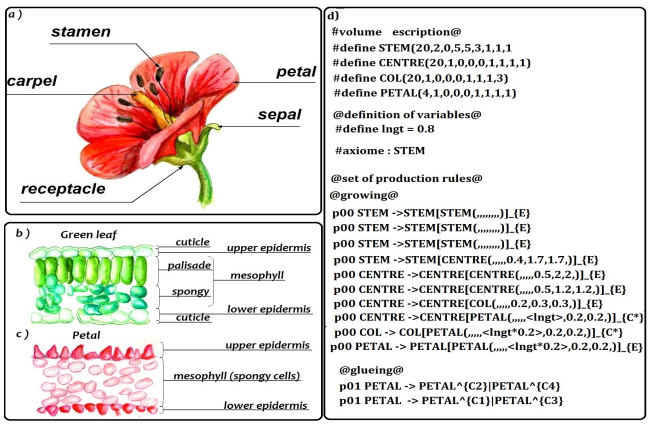


Figure 1: a) Description of flower structure. b) Internal structure of green leaf. c) Internal structure of petal. d) Grammar of a simple flower.

from the flower structure in Figure 1, the shapes of flower components are not only flat but also volumetric, so we have to use a 3 topological dimension structure to represent them. In this paper we propose a flower modeling based on 3Gmap L-systems, which allows to construct a topology of any three-dimensional subdivision. In our method we use a formal L-system grammar to describe a flower structure. The grammar is provided with different tools which make the creation process easier and allows to generate a various flowers with the same grammar. The obtained model then can be interactively adjusted by the user to get more accurate results.

4. GENERATION OF FLOWERS WITH 3GMAP L-SYSTEMS

We propose a flower modeling based on 3Gmap L-systems, a model based on three-dimensional generalized maps, which could be controlled by the operations associated with production rules. 3Gmap is an ordered topological model that allows to represent the topology of subdivisions of orientable or non-orientable 3D spaces, with or without boundary. It is close to facet-edge data structure [2] or cell_tuple [4].

A subdivision of a topological space is a partition of this space into cells with dimensions 0, 1, 2, 3, i.e. into vertices, edges, faces, volumes. This model is based on the use of an unique basic element on which four operators act. These operators are used to represent adjacency relations between edges, faces and volumes. A combination of these basic elements allows to represent the topology of an object, which corresponds to an unlimited number of embeddings of this structure in the three-dimensional space.

3Gmap L-systems are operated with volumes, which are mostly regular prisms. In order to distinguish each volume we use a label, associated to it, which is the word in capital letters. A prism V of order n is denoted as V(n). Each face of a volume also has a label which is defined as V_O, V_E and $V_{C1}, V_{C2}, \ldots, V_{Cn}$ for the base, the end and the side faces of the prism V respectively. A flower

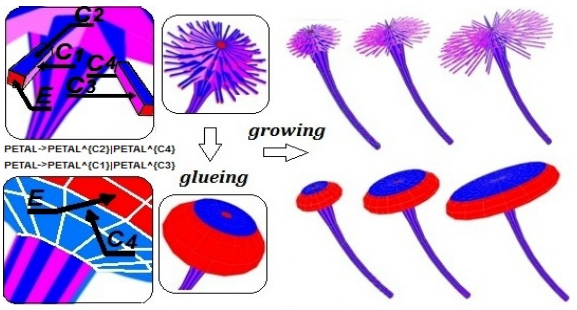


Figure 2: Generated model of grammar in Figure 1d, where growing and glueing operations are used. The model is generated with steps 1, 3 and 6.

shape is created by building volumes using a formal L-systems grammar which consists of a volume description, a variable definition, an axiom and a set of production rules (volume growing, splitting and joining) (Figure 1d, 2). Using 3Gmap L-systems, we can create different flower shapes (Figure 2). By writing a grammar we are operating recursively with different volumes and thus forming an appropriate shape. Once written a grammar we can change its parameter values and get a diversity of flowers.

5. 3GMAP L-SYSTEM ENHANCEMENTS

3Gmap L-systems allow getting quite complex models, however the more complicated the model is the more difficult it is to handle the grammar and the less intuitive it becomes for the user. In order to facilitate the task we propose to add new functionalities such as materials, interactive change of parameter values and modules.

5.1 Materials

Flower structures consist of various organs, which have distinct shapes as well as distinct materials. Different groups of volumes of the grammar represent different components of the flower, such as petals, leaves, stamens etc, which are characterized by diverse colors and illumination properties. Thereby we decided to add a new parameter in a volume definition of grammar. It represents a material of a flower organ and is denoted as integer, which is a number of materials listed in *material.mtl* file, containing definition of its various properties. This function is also useful for the models having an internal structure. Flower tissue is not homogeneous and consists of several layers: upper epidermis, lower epidermis, veins, mesophyll. If we create a model of a petal with all these layers, we assign to each one of them its proper material. Using this contribution the rendered results look much more realistic, as each component of a flower has its own color, illumination properties, etc.

5.2 Interactivity

3Gmap L-system is a parametric model, which allows us to adjust the shape of the flower, by interactively changing parameter values of the grammar. We do not have to recreate the model, but, reusing its topology, reload the embedding of the model (Figure 3a). This constitutes a great advantage because separate management from the topology and from its embedding simplifies the algorithms allowing us to easily create lots of model variations.

After the first loading of a grammar, the model may have a lot of imperfections because the parameter values of the volumes might have not been fitted properly. In order to fix it we should have to come back to the grammar again, change the values of parameters and load a model from the very beginning. This work occupies enough time to make the process of the model creation quite cumbersome. We added a new functionality which makes it possible for a user to change parameter values interactively. This way of changing parameter values is quite faster as it only takes into account the embedding part, leaving the topology part of the program untouched. In Figure 3b we can see how different components of the flower are changing. A user just selects an appropriate volume and, by clicking on a box of parameter values, changes a model shape on the fly, observing the results at the same time. Groups of different volumes can represent flower organs. In order to control them we used variables, which represent different flower component measures. The variable values can also be interactively changed, thus making the flower shape managing even more intuitive and faster. We can also change the topology of volumes interactively which simplifies the creation of models with different number of petals, leaves, etc. In this case the order of the prism is changed and we have to rerun the grammar from the very beginning which is a much slower process than the previous one. Nevertheless, this makes the process of modeling more intuitive for the user.

5.3 Grammar modules

The more complicated the flower is the more intricate and cumbersome the grammar could be. As the flower consists of different organs we decided to introduce modules which can substitute them. Modules are files containing grammars which can represent flower components. They are included into initial grammar, representing the whole model. The module itself can include another module, therefore the grammar has a folded structure, which simplifies its construction and allows creating quite intricate models. In order to construct a grammar with modules which can substitute the flower component, or a flower itself, we just add a needed module like a usual volume, with the only difference that its name must have an "&" in front of it.

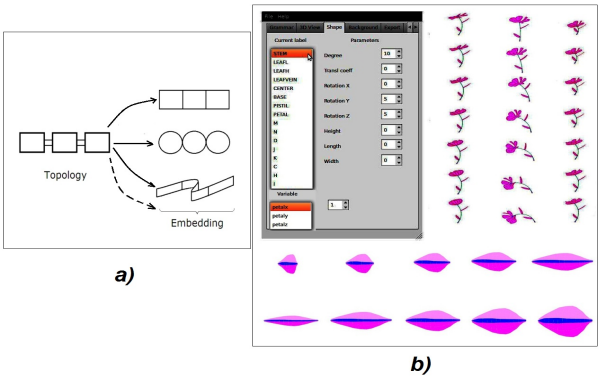


Figure 3: a) Principle of separate management. b) Interactive changing of parameter and variable values in order to modify rotation angle and shape of flower petals and stems.

In order to facilitate a task of grammar creation we added a Flower Database (Figure 5a) with different flower organs, where the user can have a look at already existing modules.

6. RESULTS

Our method allows to get a mesh of huge variety of flowers, as well as flower compositions and terrains. The software was developed under Linux Ubuntu 9, in c++ using cross platform library Qt, version 4.2. The output is exported to obj format file with the information of material assigned to each volume composing a flower geometry. The geometric models are rendered using Blender version 2.49. In Figure 4 several species of flowers are presented. Figure 5b shows a model of a petal which consists of three layers, each one of which has its proper material. Using subsurface scattering these layers will improve the realistic effect of the rendering results. All the models were created using modules and adjusted interactively.

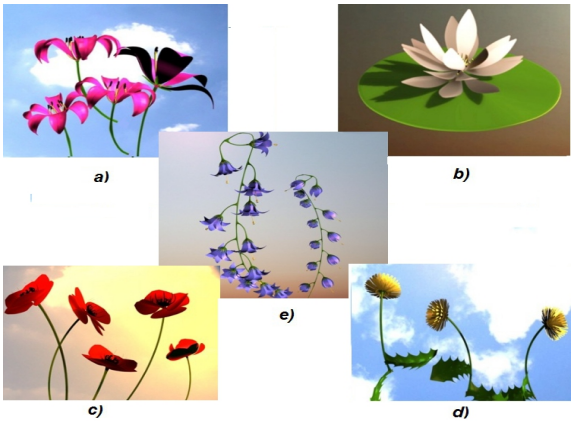


Figure 4: Models of : a) lilies; b) lotus; c) poppies; d) dandelions; e) bluebells.

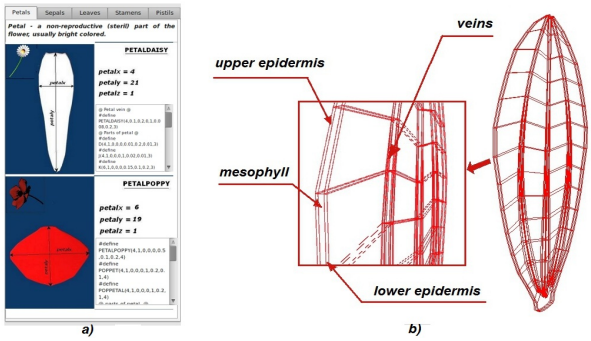


Figure 5: a) Flower database; b) Petal consisting of three layers, each one of it has its own material.

Moreover, our method allows to create realistic terrains of flowers

Figure 6: Flower terrains with models of a) wildflowers; b) daisies.

just using one grammar. We construct a terrain containing volumes and add modules which represent different flower species. In Figure 6 we can see several terrains of flowers, where each flower is different due to passing random parameter values of the modules. The time of generation of models, represented in Figures 6a and 6b is 42.4 s and 27.5 s respectively.

7. CONCLUSION

We introduced a method of flower generation based on 3Gmap L-system, allowing to create a grammar in order to construct flower models. Once written one grammar we can get a great variety of flowers by simply changing the values of its parameters. In that way we have a possibility to obtain complicated scenes with a large number of different flowers with a minimum amount of work on the grammar. We also took into account the needs of the user to have an intuitive modeling tool. Thus the shape of the flower can be modified interactively and the grammar has an intuitive structure allowing to use the modules. Due to volumetric model we can construct the internal structure of flower tissue, which consists of several layers. Assigning a material with special properties to each layer, we would be able to get more realistic results while rendering. Using subsurface scattering for rendering by taking into account the internal structure of the tissue is a future task yet to be implemented. Another improvement to be attained is texture generation according to the flower organs.

8. REFERENCES

[1] F. Anastcio, P. Prusinkiewicz, and M.C. Sousa. Sketch-based parameterization of l-systems using illustration-inspired construction lines. *University of Calgary, Canada*, 2008.

[2] E. Brisson. Representing geometric structures in d dimensions : Topology and order. *5th ACM Symposium on Computational Geometry*, pages 218–227, 1989.

[3] O. Deussen and B. Lintermann. Interactive modeling of plants. *IEEE Computer graphics and Applications*, 1999.

[4] D. Dobkin and M. Laszlo. Primitives for the manipulation of three-dimensional subdivisions. *3rd ACM Symosium on Computational Geometry*, 1987.

[5] P. Federl and P. Prusinkiewicz. Virtual laboratory. an interactive software environment for compuer graphics. 1999.

[6] D. Fowler, P. Prusinkiewicz, and J. Battjes. A collision-based model of spiral phyllotaxis. *Proceedings of SIGGRAPH : Computer Graphics*, pages 361–368, 1992.

[7] D. Frijters and A. Lindenmayer. Art and science for life: Designing and growing virtual plants with l-systems. *Springer-Verlag*, pages 24–52, 1974.

[8] D. Frijters and A. Lindenmayer. Developmental descriptions of branching patterns with paracladial relationships. *Automata, Languages and Development*, 1976.

[9] M. Fuhrer, H. Jensen, and P. Prusinkiewicz. Modeling hairy plants. *Computer Graphics and Applications. 12th Pacific Conference*, pages 217–225, 2004.

[10] T. Ijiri, S. Owada, and T. Igarashi. Floral diagrams and inflorescences: Interactive flower modeling using botanical structural constraints. *ACM Trans. On Graph*, 2005.

[11] T. Ijiri, S. Owada, and T. Igarashi. Seamless integration of initial sketching and subsequent detail editing in flower modeling. *Computer Graphics Forum*, 2006.

[12] P. Lienhardt. N-dimensional generalized combinatorial maps and cellular quasi-manifolds. *Int. J. Comput. Geom. Appl*, 1994.

[13] J. McCormack. Interactive evolution of l-system grammars for computer graphics modeling. *Clayton, Australia*, 1993.

[14] K. Onishi, N. Murakami, Y. Kitamura, and F. Kishino. Modeling of trees with interactive l-system and 3d gestures. *Proc. of BioADIT*, pages 222–235, 2006.

[15] Q. Peiyu, C. Chuanbo, and L. Zehua. Simulation model of flower using the interaction of l-systems with bezier surfaces. *Computer Engineering and application*, (16):6–8, 2006.

[16] J. Perttunen, R. Sievanen, E. Nikinmaa, H. Salminen, H. Saarenmaa, and J. Vakeva. Lignum: A tree model based on simple structural units. *Annals of Botany*, pages 87–98, 1996.

[17] A. Peyrat, O. Terraz, S. Merillou, and E. Galin. Generating vast varieties of realistic leaves with parametric 2gmap l-systems. *Springer-Verlag*, 2008.

[18] J. L. Power, A. J. Bernheim Brush, P. Prusinkiewicz, and D. H. Salesin. Interactive arrangement of botanical l-system models. *University of Washington, University of Calgary*, 1999.

[19] P. Prusinkiewicz. Art and science for life: Designing and growing virtual plants with l-systems. *Acta Horticulturae*, pages 15–28, 2004.

[20] P. Prusinkiewicz. Modeling plant growth and development. current opinion in plant biology. *Current Opinion in Plant Biology*, pages 79–83, 2004.

[21] P. Prusinkiewicz and M. Hammel. The artificial life of plants. *Artificial life for graphics, animation, and virtual reality*, 1995.

[22] P. Prusinkiewicz and A. Lindenmayer. *The Algorithmic Beauty of Plants*. Springer-Verlag, 1990.

[23] P. Reffye, T. Fourcaud, F. Blaise, D. Barthelemy, and F. Houllier. A functional model of tree growth and tree architecture. *Silva Fennica*, 1997.

[24] O. Terraz, G. Guimberteau, S. Merillou, D. Plemenos, and D. Ghazanfarpour. 3gmap l-systems. an application to the modeling of wood. *Springer-Verlag*, 2008.

Background mosaic reconstruction

Anton Zachesov
Department of Computational Mathematics and Cybernetics
Moscow State University, Moscow, Russia
azachesov@graphics.cs.msu.ru

Dmitry Vatolin
Department of Computational Mathematics and Cybernetics
Moscow State University, Moscow, Russia
dmitry@graphics.cs.msu.ru

Maxim Smirnov
YUVsoft Corp.
ms@yuvsoft.com

Abstract

In this paper, we present a new fast approach for generation of background panorama from a video sequence based on motion estimation algorithm. The proposed method provides correct matching of background areas in video sequences with complex camera movement. It also performs rough preliminary segmentation which allows removing foreground objects from mosaic on inserting stage. Result sprites have appropriate quality and level of detail to use them in other applications.

Keywords: *panorama, motion-based segmentation, background mosaicing, background restoration.*

1. INTRODUCTION

Nowadays video processing algorithms, for example multiview generation or diminished reality construction, require high-quality background restoration technique. Different approaches such as inpainting, motion-based background completion and mosaicking have been proposed for the solution of this task.

Multi-layer texture inpainting have been used in [2] for objects' removing from the original video. Method provides a good quality of restored area, but has been tested for sequences with static camera and does not provide extrapolation of background outside the frame borders. Real-time background inpainting method has been proposed by [3]. It works fine for uniform background areas, but its work on complex textured background with larger foreground objects is unknown.

Motion based background completion have been proposed in [6]. Method uses tracking window for foreground objects marking. The main disadvantage of this method is the dependence of computational complexity on foreground objects' size. Also method has not been tested on sequences, where foreground occupies a large area of frame. And it does not provide the opportunity for expanding background outside the frame borders.

Mosaicking and background sprites construction is one of widespread approaches. The method of super resolution sprite generation was described in [5]. It provides single image representing background of a video sequence. The main disadvantage of this method is geometrical distortions which may appear on sprite borders. Another approach presented in [1] uses optical flow for frame segmentation and hierarchical insertion coordinates calculation which makes it not fast enough for use as a part of other algorithms.

The main task of the current approach was to create a method for fast background sprites construction, which quality and level of detail is the same as the quality of source frames. Most of existing approaches [1][5] are complicated for use at high resolutions video sequences and as part of other algorithms.

The main problems for panorama construction are connected with accuracy of frame matching during the insertion process and, in particular, with the choice of points for the transformation matrix calculation between background areas of frames. In this approach the coordinates of points in neighbor frames are obtained from motion estimation algorithm and the 8-parameter affine transformation is used to align two frames.

This paper is organized as follows. Section 2 describes motion estimation algorithm and points choosing algorithm. Section 3 describes frames' merging algorithm. Section 4 provides experimental results and Section 5 summarizes the paper.

2. MOTION ANALYSIS

This section describes the algorithm of motion analysis used for background transformation obtaining, background area evaluation and algorithm of choosing points for transformation calculation. Motion estimation algorithm [4] is used for correspondence search between points of analyzed frames. It provides motion vectors between square blocks in two frames with quarter-pixel precision and information about estimation error for every vector.

2.1 Background segmentation

For background area evaluation a simple motion-based segmentation algorithm is proposed. It uses two-dimensional histogram of motion vectors.

The histogram has size $MV_{max} \times MV_{max}$ for all possible motion vectors' coordinates, where MV_{max} is an algorithm parameter, which specifies the maximum length of a motion vector.

Each element with position (x, y) in the histogram corresponds to a motion vector with coordinates:

$(x - center\_x,\ y - center\_y)$,

where $(center\_x, center\_y)$ – coordinates of the center of the histogram.

Each element of the histogram stores:

- list of points in a frame corresponding to this motion vector
- top left and bottom right points from this list
- number of points corresponding to this motion vector
- label – in a clusterized histogram, number of a cluster containing this point

The main idea of histogram clusterization is to distinguish a rough mask of background area for the current frame. This segmentation will be used afterwards in transform matrix calculation.

First of all, the histogram represents all possible motion vectors for a pair of frames. If some point in an image has a motion vector with coordinates (x,y), histogram element (x,y) contains this point. Segmentation of a frame is based on analysis of histogram elements that contain at least two points. These elements will be called "valuable elements".

The proposed method considers all connected regions in the histogram as clusters. It means the algorithm marks neighbor valuable elements of the histogram and all valuable elements with the same label from one cluster. A cluster, in which the histogram element with maximum of points is situated (points from the current frame), is assumed to be background.

Figure 1 illustrates example of a clusterized histogram. Colored pixels correspond to valuable elements of histogram. White color marks the background area, blue – foreground objects. Three-dimensional plot represents the number of points in current frame, contained in each histogram element. Red peak corresponds to the background area.

2.2 Sky detection

For outdoor scenes background area often contains sky. The proposed method considers sky in calculations to improve segmentation accuracy. It is assumed, that sky area always belongs to background, so the points' choice is modified if sky detection gives positive result.

The sky detection algorithm is based on a color segmentation of potential sky areas, using pixel colors in HSV color space.

First of all, it is supposed that sky is in the upper region of a frame, so the method searches for it there. If a pixel color corresponds to a comparison criterion, the method checks its neighbor pixels. The algorithm stops if the next pixel doesn't correspond to the color criterion or a difference between neighbor pixels colors is bigger than a constant threshold.

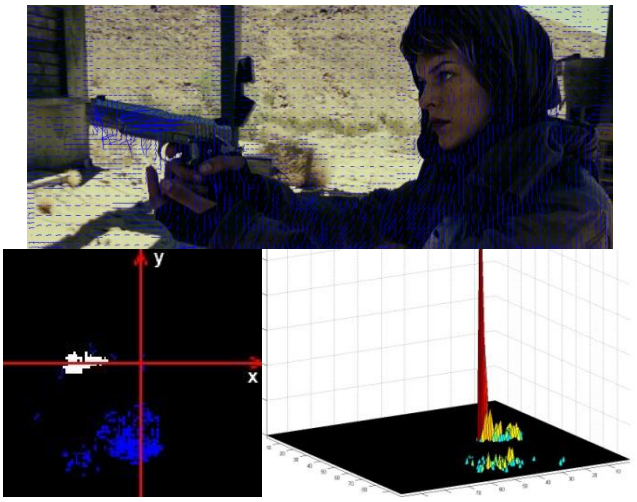


Figure 1: Example of source frame with motion vectors (blue lines) and clusterized histogram. Orthogonal projection (left) and 3D-visualization (right)

The used color criterion has been chosen empirically based on set of ~20 outdoor images containing sky. It consists of 3 inequalities:

- $\begin{cases} hsv.v > 0.7 \\ hsv.h < 0.001 \end{cases}$
- $\begin{cases} hsv.v > 0.3 \\ hsv.h > 0.4 \\ hsv.s < 0.8 \end{cases}$
- $\begin{cases} hsv.v > 0.75 \\ hsv.s < 0.2 \end{cases}$

Figure 2 shows the example of obtained sky mask.

Figure 2: Result of sky detection algorithm

2.3 Equation points choice

Transform matrix is calculated from corresponding points' coordinates. Motion estimation provides large number of points with different coordinate precision, so there is need in choosing appropriate small set of points' pairs for further calculations. This set will be called equation points

The following algorithm is used for selecting equation points. First, points chosen for the previous pair of frames are checked with the current comparison criterion. It means that method checks several properties of a point. Two methods are implemented.

The first is fully based on the result of segmentation results and also considers the variance of the block containing analyzed point (point does not suit if the variance is smaller than a threshold which depends on the average variance of the frame and is calculated for every frame). The second considers the sky mask and segmentation results. It takes only points from the background area which motion vectors are similar to the average motion vector of the sky area.

$$P = P(x, y, variance) \cdot label \cdot sky\_label$$

$$label = \begin{cases} 1, & pixel \in background \\ 0, & otherwise \end{cases}$$

$P(x, y, variance)$ – probability for point with coordinates (x, y) situated in block with $variance$ variance

If a point suits, information about it is updated and used in further calculations. If the number of suitable points from the previous step is less than predefined threshold (algorithm parameter, differs from 10 to 30), the method chooses other points from the current frame, using 32x32 pixels' grid. From each grid cell the candidate point with the smallest motion vector error is chosen. Then array of suitable points is sorted using points' motion vector error as a criterion. ¼ of points with the largest errors are deleted from the list. Then remaining points are decimated by removing every second point from the points list until the list contains ~25 points for the transformation calculation.

3. FRAME WARPING

After obtaining the set of equation points with appropriate coordinates, frame is inserted into the mosaic. 8-parametric affine transform is used to warp frame to his actual coordinates in mosaic. This section describes transformation matrix obtaining and inserting process. Such matrix is calculated for all neighbor frames in video sequence.

3.1 Transformation calculation

The used model can be represented by the following matrix:

$$H = \begin{pmatrix} h00 & h01 & h02 \\ h10 & h11 & h12 \\ h20 & h21 & 1 \end{pmatrix}.$$

This model provides accurate background matching in cases of plane-parallel, rotational camera motion, zoom and more complex combinations of such motion. The transformation is applied to every pixel of inserted frame. This means the matrix is multiplied by a pixel coordinate vector (z coordinate always equals to 1).

$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} h00 & h01 & h02 \\ h10 & h11 & h12 \\ h20 & h21 & 1 \end{pmatrix} * \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}; \quad \begin{pmatrix} x'' \\ y'' \end{pmatrix} = \frac{1}{z'} * \begin{pmatrix} x' \\ y' \end{pmatrix},$$

Here (x, y) are pixel coordinates in the current frame, (x'', y'') are pixel coordinates in the reference frame.

After equation points have been chosen, system of equations for the projective transformation matrix calculation is configured. From every point 2 equations are obtained:

$$X * h00 + Y * h01 + 1 * h02 + 0 * h10 + 0 * h11 + 0 * h12 - X * X' * h20 - Y * X' * h21 - X' = 0$$

$$0 * h00 + 0 * h01 + 0 * h02 + X * h10 + Y * h11 + 1 * h12 - X * Y' * h20 - Y * Y' * h21 - Y' = 0$$

where (X, Y) are coordinates of a pixel in the current frame, (X', Y') – in a reference frame, hxx – the corresponding element of 3x3 transformation matrix. Then obtained system is reduced to the square form by applying least squares and solved.

3.2 Frame insertion

Before frame insertion into mosaic the final transformation matrix between frame and mosaic is obtained. It equals to product of all consecutive transformation matrices between frames, starting from frame, which was the initial for the current mosaic. Good precision of consecutive transformation matrix calculation minimizes the result error of matrix multiplication for a long sequence.

If camera rotation angle for final transformation matrix is large, new sprite generation starts. Otherwise frame distortion on sprite boundaries will be very notable and spoil the quality of result panorama. This will make a sprite inapplicable for further use.

If frame transform increases frame area or its dimensions comparing to source dimensions frame is upscaled before the insertion into the mosaic by bicubic interpolation. Every frame changes only unfilled regions of mosaic except of frame boundaries. Linear blending of pixels is performed for boundary pixels in mosaic. It is performed to make these boundaries not notable in final sprite. Blending is based on bilateral filtering.

4. EXPERIMENTAL RESULTS

The main criterion for result quality measure is to avoid discontinuities on boundaries of inserted frames and to preserve precise level of detail.

The algorithm has been tested on fragments from movies such as “James Bond” and “Resident evil”. Scenes with both simple and complex motion have been used. The number of used test sequences is 44. The example of algorithm result is shown in Figure 3 and 4.

Use of frame segmentation during insertion process provides a mosaic with removed foreground objects, as shown in Figure 4. It is shown that the proposed method allows correct inpainting of massive foreground objects. Scaled fragment shows that the proposed method allows correct matching for complex background with large number of small details. This makes method useful in tasks where quality of restored background is critical.

The speed of background mosaic construction depends on algorithm mode. It is possible to create a set of mosaic sprites for analyzed video sequence or to create panorama for every frame in both directions. For the first mode algorithm speed exceeds 4 fps for 960×400 resolution sequence. Second mode has much more computational complexity and requires huge amount of memory operations. It also depends on number of frames added to mosaic for each frame. Creating panorama for one frame (960×400) in this mode takes approximately 40-45 seconds with 25 frames range in both directions. Speed tests were made on Intel Core2 Quad Q9450 2.66 GHz.

5. CONCLUSION

This work presents a fast approach of creating background mosaic for input videos. The proposed method is suitable for fully automatic background restoration and provides sprites of good level of detail. Used foreground-background pre-segmentation allows full or partial removing of foreground objects from the result sprite. Method can also be used for background generation for stereo and multiview generation.

6. ACKNOWLEDGMENTS

This work was supported by RFFI grant 10-01-00697-a.

7. REFERENCES

[1] A. Krutz, A. Glantz, T. Borgmann, M. Frater, and T. Sikora, “*Motionbased object segmentation using local background sprites*,” in IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Taipei, Taiwan, April 2009.

[2] I. Cheng Chang and Chia-We Hsu, “*Video Inpainting Based on Multi-Layer Approach*,” Proceedings of APSIPA Annual Summit and Conference, Sapporo, Japan, Oct. 2009

[3] Jan Herling and Wolfgang Broll, http://www.tu-ilmenau.de/journalisten/pressemeldungen/einzelnachricht/newsbeitrag/5784/

[4] K. Simonyan, S. Grishin, D. Vatolin, D. Popov, “*Fast video super-resolution via classification*,” in Proc. IEEE ICIP, pp. 349-352, San Diego, Oct. 2008.

[5] M. Kunter, J. Kim, and T. Sikora, “*Super-resolution mosaicking using embedded hybrid recursive flow-based segmentation*,” in IEEE Int. Conf. on Information, Communication and Signal Processing (ICICS’05), Bangkok, Thailand, Dec. 2005.

[6] Soon-Yong Park1, Chang-Joon Park2, and Inho Lee, “*Moving Object Removal and Background Completion in a Video Sequence*,” 2008

About the authors

Dmitriy Vatolin (M’06) received his M.S. degree in 1996 and his Ph.D. in 2000, both from Moscow State University. Currently he is Head of the Video Group at the CS MSU Graphics & Media Lab. He is author of the book Methods of Image Compression (Moscow, VMK MGU, 1999), co-author of the book Methods of Data Compression (Moscow, Dialog-MIFI, 2002) and co-founder of www.compression.ru and www.compression-links.info. His research interests include compression techniques, video processing and 3D video technics: optical flow, depth estimation - from motion, focus, cues, video matting, background restoration, high quality stereo generation. His contact email is dmitriy@graphics.cs.msu.ru.

Maxim Smirnov is the chief technology officer at YUVsoft IT company. Maxim has graduated from Saint Petersburg State University of Aerospace Instrumention in 2001 and received a Ph.D. in data compression from the same university in 2005. His research interests include forecasting of processes in technical and economic systems, video and universal data compression, stereo video. His contact email is ms@yuvsoft.com.

Anton Zachesov is a student at Moscow State University, Department of Computational Mathematics and Cybernetics. His contact email is azachesov@graphics.cs.msu.ru.

Figure 3: Frames 28, 93, 135 of the original sequence and panorama based on frame 93

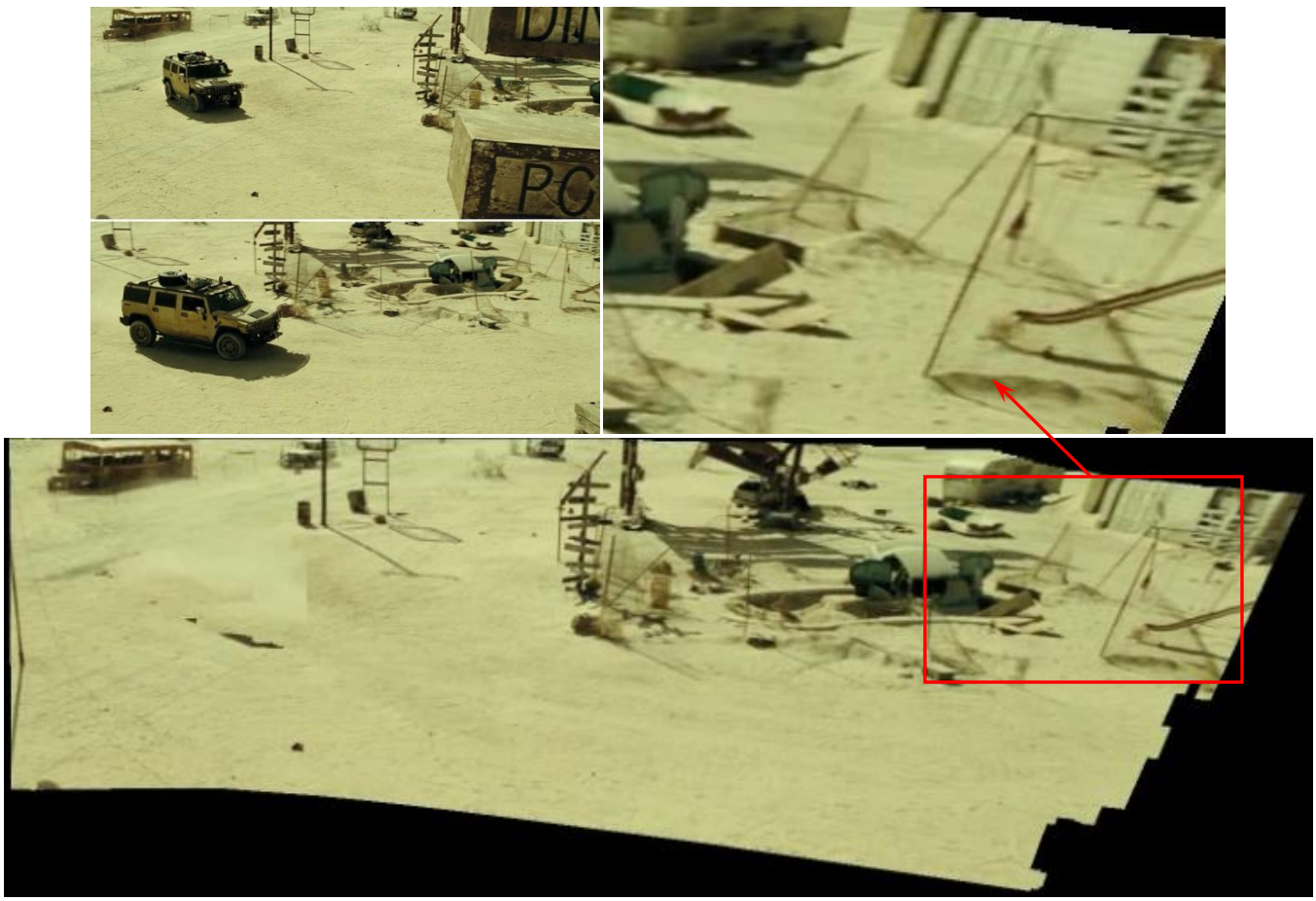


Figure 4: Frames 4, 54 of original sequence and panorama based on frame 4

High-speed OCR algorithm for portable passport readers

Victor Bessmeltsev, Evgeny Bulushev, Nikolay Goloshevsky
Institute of Automation and Electrometry SB RAS
Novosibirsk, Russia
e.d.bulushev@gmail.com

Abstract

The modern portable passport readers are able to acquire high quality images (up to 450 dpi), but have poor performance embedded computational facilities, because of the requirement on long battery life. For the extraction and recognition of machine-readable document codes the high-speed and accurate projection-based OCR algorithm is developed. Profile projection is calculated for middle columns of document image and is smoothed using the moving average filter. The slant compensation is conducted, using the angle between the horizontal line and the line connecting centers of the extreme individual codes. For the embedded processor (AMD Geode 500 MHz 128 RAM) the algorithm shows 100% accuracy and processing time less than 1 second for 300 and 450 dpi images.

Keywords: *MRZ, recognition of passports, OCR, high-speed segmentation, slant compensation, portable passport readers, profile projection*

1. INTRODUCTION

A biometric passport and visa contains picture and user information areas for visual inspection and the machine-readable zone (MRZ) with individual codes, specified in the standard ICAO 9303, which can be captured and processed using optical character recognition (OCR) systems. Currently the relevance of the development of portable passport readers for rapid examination of documents on board ships, road and railway transport, border crossings increases. Basic requirements for such devices – light weight (less than 3 kg), several hours of battery life, short time of document processing (less than 1 second) and wireless data transmission to the server for additional inspection. Such devices include embedded computing facilities, sensor module to acquire document image and modules for reading a contactless chip and data transmission. A document image is processed using both OCR methods and face recognition methods. In order to improve the reliability of the recognition and the determination of forged documents images are acquired in high-resolution (up to 450 dpi) and in several spectral ranges (UV, visible, IR).

Portable passport readers exploit relatively poor performance processors in order to remain low-power and have long battery life. According to our experience the commercial solutions for mobile passport readers recognize MRZ in more than 3 seconds. The efficiency of standard solutions for recognition of individual codes was estimated, as with the specialized software, based on the algorithm [7], and the general-purpose software – CuneiForm OCR version 12. The average recognition time of high resolution (450 dpi) document image is respectively 7 and 3 seconds at the portable passport reader with the embedded processor AMD Geode 500MHz and 128 MB RAM. Such performance is unacceptable, and therefore the development of algorithm with the accuracy higher than 99% and the processing time less than 1 second is required.

2. OVERVIEW

The document image processing consists of the following steps: detection of MRZ and code sequence blocks, extraction and slant compensation of document codes, individual codes recognition. In the specialized algorithm [7] code sequence blocks are extracted from the document image using Sobel masking, horizontal smearing and contour tracking. Template matching, morphological methods [8], fuzzy logic, artificial neural networks [9] and crosscheck with the visual inspection area [9] are used for the individual code recognition. The implementation [6] of these algorithms is efficient (accuracy above 98%), robust to noisy and low detail images (resolution below 300 dpi, skew angle up to 10 degrees), but demanding to the computational resources. Generally such high robustness is excess for modern mobile passport readers, due to their ability to acquire high resolution images (up to 450 dpi) with high signal to noise ratio [2] and small maximum image skew (less than 1 degree).

Printed and hand-written documents are segmented using projection profiles method [11]. The vertical projection profile is obtained by summing intensity of pixels along the horizontal axis for each image row. The profile can be used to segment text lines as average intensity of background and symbols differs. The projection-based method is applied to the black-and-white image to extract MRZ and picture area [10]. The projection-based method is less resource-intensive than method, based on horizontal smearing, due to applying simple algorithms of image transformation and analyzes. Meanwhile, the smearing of vertical profile, caused by image skewing and noise, makes this technique less reliable and accurate.

The accuracy of individual codes recognition can be improved for skewed document images using slant compensation. The Hough transform is performed to determine skew angle [16]. The accuracy of 1-2 degrees was achieved for the low resolution images (150 dpi) with irregular illumination, and maximum skew angle up to 10 degrees.

In Karateev et al. [5] is argued that individual codes recognition is the most resource-intensive step of the document image processing, because of the numerous template matching operations. To accelerate this step the rapid template-based matching is performed using font image, generated with special software. According to our experience the speed of individual code recognition can be boosted, using parallelization, both at the CPU and at the hardware level (e.g. using FPGA). Meanwhile for the high resolution images MRZ detection and code sequence blocks extraction takes longer time and is not easily parallelized.

So, methods, based on the horizontal smearing, and the profile projection exist for MRZ detection and individual codes extraction. The first is highly robust but relatively slow, the second is efficient but less robust in case of noise and document slanting. In this paper the high-speed and accurate algorithm is developed for individual codes extraction from high resolution (more than 300 dpi) document images with small slant angle (less

than 1 degree). The algorithm is based on the transformation of vertical profile projection and skew angle detection, using the individual codes position.

3. EXTRACTION AND SLANT COMPENSATION OF DOCUMENT CODES

The MRZ is detected using the vertical projection method. Unlike the standard methods, which project the intensity of source image or black-and-white image, we use the projection of red channel of source image. In Figure 1 the lower part of the document image is depicted. The vertical projection profile of red channel is shown in Figure 2. The right extreme sharp gap corresponds to the bottom border of the document; the two next recessions match the centers of the code sequence blocks. The bottom part of gap corresponds to the center of text line.

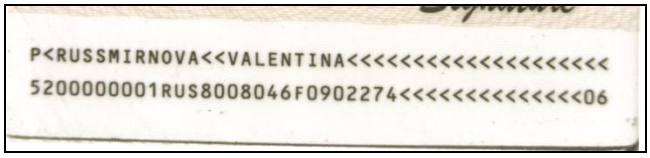


Figure 1: MRZ and individual codes.

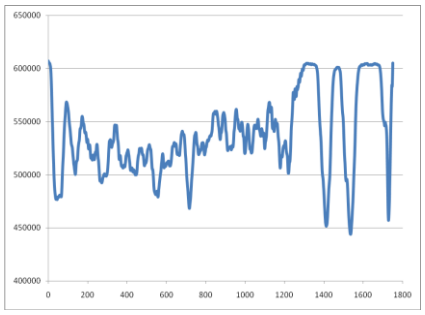


Figure 2: Projection of red channel. In horizontal direction - row number, in vertical direction – the total sum of red color intensity for row. Three rightmost drops correspond to code sequence blocks and border of the document.

Two gaps, corresponding to code sequence blocks, intersect a little for small skew angles (less than 1-2 degree) of document image and can be detected independently. The complexity of automatic detection of the required minima is caused by plenty of local extrema, caused by skew or moderate fluctuations of intensity of text lines. The criterion of the predefined depth of profile gap is not adaptive, because of the various intensity distributions and features of different countries documents. The statistical criterion, calculated based on the histogram of distance between adjacent local minima, is used in [1] to determine potential segmentation points in printed documents.

In order to increase robustness of segmentation an image may be divided into vertical strips and profiles sought inside each strip [18]. The Gaussian filtering can be used to reduce the number of local minima of profile [12]. We applied both approaches with small time-saving modifications. In order to increase computational speed the moving average filter (MAF) is applied instead of the Gaussian smoothing. The vertical projection is computed only for middle strip, that reduces the volume of processed information and increases the speed of document analyze. Generally the duration of data analyze can be reduced up to one order.

The resulting profile has less local extrema (Figure 3). Code sequence blocks are extracted using the criterion of monotonicity. First two profile points, for which the previous values monotonically decrease in more than N consecutive image rows, starting from the bottom of image, are the required extrema. For the demonstrated profile the following coefficients were used (N = 20, the size of the smoothing window = 41, coefficients of constraints of strip are 0.4 and 0.6). Experiments show that the developed algorithm for code sequence blocks extraction is adaptive to small image slanting, change of illumination and diversity of document features.

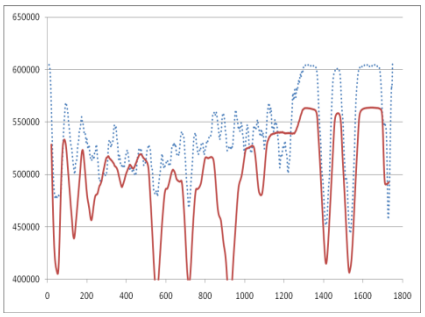


Figure 3: The dotted blue line - the projection of the red channel. The solid red line - the projection of the red channel, calculated for the middle columns, smoothed by MAF, and scaled by "y"

The MRZ image is formed, after the code sequence blocks position is determined. In order to simplify the individual codes extraction the threshold conversion of the MRZ image is performed, using the threshold value, computed automatically using the Otsu method [15]. This method is robust to the average intensity change of document image. Figure 4 shows the result of image conversion.

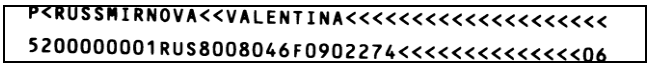


Figure 4: The result of MRZ image binarization

The horizontal position of individual passport codes is determined using horizontal projection of black-and-white image. The vertical position of codes is refined using vertical projection in the vicinity of codes location. The MRZ image and zoom image with red rectangles, allocating the result of individual codes extraction, are depicted on Figure 6.

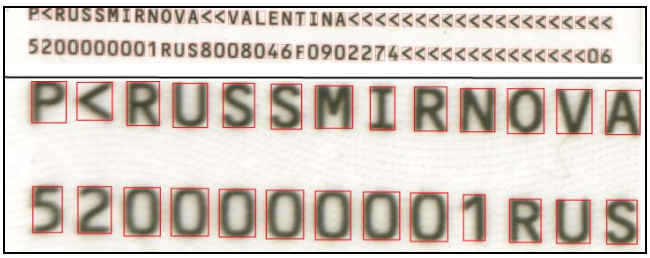


Figure 5: Individual codes extraction. Red rectangles correspond to document code positions.

Black-and-white MRZ image is adjusted, using slant compensation, in order to improve the efficiency of recognition of individual codes. The Hough transform [4] or image moments

[17] are usually used to estimate a skew angle. However these methods are inefficient in case of geometric distortions of image, which can be significant and caused by imperfection of optical registration system. Even in the case of correction of geometrical distortions the presented algorithms are relatively slow, because a lot of statistical data is analyzed (more than a thousand of points). For this reason the efficient and fast slant compensation method is proposed. The slant of each code sequence block is determined as the angle between the horizontal line and the line between centers of the extreme rectangles of individual codes. Image skew is calculated as the average of two values.

For small skew angles (angle is less than 0.1 degree), image is not adjusted since the recognition accuracy does not change. In this case the algorithm demonstrates high productivity, as a skew angle is defined based on the statistical data reduced in several orders. If skew angle exceeds the preset threshold a document image is adjusted and positions of individual codes are refined using projection method in the vicinity of previous locations. The result of image slant compensation is shown on Figure 6 for the MRZ image and zoom image.

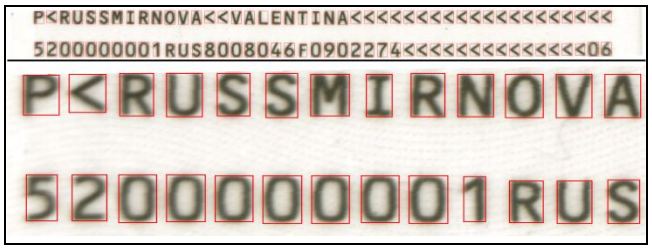


Figure 6: Image slant compensation. Red rectangles correspond to document code positions.

4. RECOGNITION

The recognition step is performed after document codes are extracted, that includes feature extraction and classification. The template matching and line intersections methods can be used to get features of the individual codes [3]. In [14] the general case of feature extraction from the monochrome characters image is considered. We use template-based matching, which is efficient in case of high resolution and low noise document images. The template was generated on the basis of the standard font OCR-B (Figure 6).

Figure 6: Font Template

5. RESULTS

The algorithm is tested on the portable passport reader [2], consisting of image sensor module, single-board industry computer and storage device. Image sensor module is intended to preprocess data acquired by Foveon F13 (14 million elementary photodetectors, organized in matrix 2688 x 1792 x 3 layers with up to 50MHz data output frequency) [13]. Field-programmable gate array (FPGA) controls the matrix and carries out real-time preprocessing of data stream. The main preprocessing stages are linearization, dark frame subtraction, column filtering, bad pixel replacement and color space transformation. Module has external memory connected to FPGA to store intermediate data, calibrating tables and images. Converted data is transferred to processing module connected to image sensor module by parallel 32-bit interface. Image recognition, visualization and storing are performed by industry computer (AMD Geode based). In order to output and transfer information, system is completed with display and wireless LAN.

The algorithm is implemented in C++ and its code is optimized using the static analyzer PVS-Studio and VS 2010 profiler. For the performance evaluation we experimented on the CPU Intel Pentium Dual-Core 2.50 GHz 1.96 GB of RAM and the portable passport reader with embedded computational facilities AMD Geode low power LX800 500 MHz 128 MB RAM. The results of the testing are shown in Table 1.

	PC. Time (ms)	Portable passport reader. Time (ms)
450 dpi	31	460
Image skew. 450 dpi	47	801
300 dpi	25	360
Image skew. 300 dpi	37	530

Table 1: Testing the performance of the algorithm

The first row shows the duration of recognition of high quality images (450 dpi) from the test database of 30 documents. The performance higher than one document in half of second and 100% accuracy was achieved. The second row shows the time span of processing of document images rotated by 1 degree. The required performance and the 100% accuracy were obtained. However the necessity of recalculating of profile projections significantly reduces the calculation speed. The testing has shown that the main time-saving algorithmic improvement is the computation of the vertical projection for the middle strip. It saves about 50 ms or 10% of calculating time. For the described portable passport reader the algorithm is robust to image skew angle less than 2 degree. For bigger skew angles the criterion of monotonicity is not effective, because of the smearing of vertical profile.

For downsampled to 300 dpi images 100% accuracy was achieved and calculation speed increased. So the reduction of resolution of document image may be applied, in order to increase the productivity. For 200 dpi images the accuracy of recognition dropped, because of using template-based approach with general font image. Black-and-white individual codes, extracted by the proposed algorithm on utilized portable passport reader, may be used to specify the template in order to increase the accuracy of recognition.

6. CONCLUSION

This paper proposes the high-speed and accurate algorithm for individual codes extraction from document image. The algorithm is based on segmentation of vertical projection profile, supplemented by MAF smoothing and data reduction to middle strip. The vertical projection profile is computed for the red channel of source image. In addition, the efficient slant compensation method is proposed, using the angle between the horizontal line and the line, connecting centers of the extreme individual codes. The algorithm is 100% accurate for high quality document images (300 dpi and 450 dpi, high signal to noise ratio, skew angle less than 2 degree). For low resolution images (less

than 200 dpi), high skew angles (more than 2 degree) or low signal to noise ratio algorithm is inaccurate.

For the embedded processor (AMD Geode 500 MHz 128 RAM) the implementation of the algorithm processes document images in less than 1 second. It revealed at least double performance increase over the algorithm implementation, based on the horizontal smearing. The boost of the document analyze is caused by the algorithmic reduction of the processing data volume and by applying general OCR approaches for the specified task of extraction of individual codes. The performance of the developed software can be improved by at least twice in the case of implementing the algorithm on the hardware level (e.g., FPGA) and recognition of the individual codes in parallel.

7. REFERENCES

[1] A. Antonacopoulos, D. Karatzas, "Document image analysis for World War II personal records", *First International Workshop on Document Image Analysis for Libraries, DIAL'04*, Palo Alto, pp. 336–341.

[2] V. Bessmeltsev and N. Goloshevsky, "System of high-resolution machine vision with stacked photodiodes structure of photosensitive sensor," *Proceedings of the IASTED International Conference, ACIT*, Novosibirsk: 2010.

[3] A.V. Bondarenko, V.I. Galaktionov, V.A. Goremychkin, A.V. Yermakov, and S.Y. Zgelto, *Research of the Approaches to Construction of Systems of Automatic Reading of the Symbolical Information Preprint, Inst. Appl. Math., the Russian Academy of Science*, Moscow: 2003.

[4] G. Kapogiannopoulos, "A fast high precision algorithm for the estimation of skew angle using moments," *Proceeding of IASTED*, 2002.

[5] S.L. Karateev, I.V. Beketova, M.V. Ososkov, V.A. Knyaz, Y.V. Vizilter, A.V. Bondarenko, and S.Y. Zheltov, "Software-Hardware System for Digital Face Imagery Acquisition and Testing for Biometric Documents," *Вестник компьютерных и информационных технологий*, vol. 2, 2008, pp. 9-14.

[6] K. Kim, "Intelligent immigration control system by using passport recognition and face verification," *Advances in Neural Networks–ISNN 2005*, 2005, pp. 147–156.

[7] K. Kim, "Passport Recognition Using Enhanced ART2-based RBF Neural Networks," *IJCSNS*, vol. 6, 2006, pp. 12-17.

[8] K. Kim, "A passport recognition and face verification using enhanced fuzzy ART based RBF network and PCA algorithm," *Neurocomputing*, vol. 71, 2008, pp. 3202–3210.

[9] K.B. Kim, J.H. Cho, and C.K. Kim, "Recognition of passports using FCM-based RBF network," *AI 2005: Advances in Artificial Intelligence*, 2005, pp. 1241–1245.

[10] T. Kim and Y. Kwon, "Crosscheck of Passport Information for Personal Identification," *Graphics Recognition. Ten Years Review and Future Perspectives*, 2006, pp. 162–172.

[11] L. Likforman-Sulem, A. Zahour, and B. Taconet, "Text line segmentation of historical documents: a survey," *International Journal of Document Analysis and Recognition (IJDAR)*, vol. 9, Sep. 2006, pp. 123-138.

[12] R. Manmatha and N. Srimal, "Scale space technique for word segmentation in handwritten documents," *Scale-Space Theories in Computer Vision*, 1999, pp. 22–33.

[13] R. Merrill, "Color separation in an active pixel cell imaging array using a triple-well structure," *U.S. Patent 5,965,875*, 1999

[14] I.A. Mikhaylov, "Image recognition using a radial neighborhood method," *Computer Optics*, vol. 34, 2010, pp. 399-407.

[15] N. Otsu, "A threshold selection method from gray-level histograms," *Automatica*, vol. 11, 1975, p. 285–296.

[16] Y.V. Visilter, S.Y. Zheltov, and A.A. Lukin, "Development of OCR system for portable passport and visa reader" *Proceedings of SPIE*, 1999, pp. 194-199.

[17] B. Yu and A. Jain, "A robust and fast skew detection algorithm for generic documents," *Pattern Recognition*, vol. 29, 1996, pp. 1599-1629.

[18] A. Zahour, B. Taconet, P. Mercy, and S. Ramdane, "Arabic hand-written text-line extraction," *Proceedings of Sixth International Conference on Document Analysis and Recognition*, 2001, pp. 281–285.

A Two-stage and Parameter-free Binarization Method for Degraded Document Images

Yung-Hsiang Chiu[1], Kuo-Liang Chung[1], Yong-Huai Huang[2], Wei-Ning Yang[3], Chi-Huang Liao[4]
[1]Department of Computer Science and Information Engineering,
National Taiwan University of Science and Technology, Taipei, Taiwan, R. O. C.
[2]Institute of Computer and Communication Engineering,
Jinwen University of Science and Technology, Taipei, Taiwan, R. O. C.
[3]Department of Information Management,
National Taiwan University of Science and Technology, Taipei, Taiwan, R. O. C.
[4]System Online Co. Ltd., Taiwan, R. O. C.

Abstract

Binarization plays an important role in document image processing, especially in degraded documents. For degraded document images, adaptive binarization methods often incorporate local information to determine the binarization threshold for each individual pixel in the document image. We propose a two-stage parameter-free window-based method to binarize the degraded document images. In the first stage, a proposed scheme is used to determine a proper window size beyond which no substantial increase in the local variation of pixel intensities is observed. In the second stage, based on the determined window size, a noise-suppressing scheme delivers the final binarized image by contrasting two binarized images which are produced by two adaptive thresholding schemes depending on the change rate of the number of binarized foreground pixels. Empirical results demonstrate that the proposed method is competitive when compared to the existing adaptive binarization methods and achieves better performance in F-measure.

Keywords: *Adaptive binarization method, Degraded document image, Document image processing.*

1. INTRODUCTION

Document image processing is necessary for storing, transmitting, and managing digital documents. Among different types of document image processing, binarization is a preliminary process and the resultant binary images usually affect the performance of the succeeding processes, such as the document image segmentation, the optical character recognition, and so on. For binarization, each pixel in a document image can be classified as a foreground or a background pixel. Pixels inside characters, lines, and curves in a document image are foreground pixels and should be binarized as black pixels and the remaining background pixels should be binarized as white pixels.

For maximizing the between-class variance of foreground and background pixels, Otsu [1] proposed an automatic thresholding scheme to determine a global threshold for the input image. It usually yield good resultant binary images. However, the determined global threshold may not be applicable for degraded document images since intensities of foreground and background pixels are contaminated at different positions of the images. To alleviate the problem caused by the degraded document images, adaptive binarization schemes [2, 3, 4, 5, 7, 8] which incorporate the information from local statistics of an image are proposed to improve the global thresholding method. Niblack [2] presented a window-based method to determine the threshold for each pixel by incorporating the information of the mean and the standard deviation of gray levels within each window. Sauvola and Pietikainen [3] modified Niblack's method by proposing different weights on the mean and the standard deviation of gray levels within each window. For blueprint images, Zhao *et al.* [4] utilized geometric features and proposed an efficient window-based thresholding method. Gatos *et al.* [5] binarize the document image by contrasting the document image to the background surface which is constructed by interpolating the background pixels after removing the binarized foreground pixels via Sauvola and Pietikainen's method. Based on the edge map detected by the Canny edge-detector [6], Chen *et al.* [7] binarized the input document image using a pair of high and low thresholds. Moghaddam and Cheriet [8] proposed a multi-scale window-based thresholding scheme which first generates several binarized images based on different window sizes and then iteratively combines the binarized images to yield the final binarized image.

In this paper, we presented a two-stage parameter-free window-based method to binarize the degraded documents. In the first stage, a proposed scheme is used to determine a proper window size beyond which no substantial increase in the local variation of gray levels is observed. In the second stage, given the determined window size, a noise-suppressing scheme delivers the final binary image by contrasting two binarized images which are produced by two adaptive thresholding schemes depending on the change rate of the number of foreground pixels. Empirical results demonstrate that the proposed method is competitive when compared to the existing adaptive binarization methods and achieves better performance in F-measure.

2. CHALLENGES IN ADAPTIVE BINARIZATION

The adaptive binarization scheme needs to deal with two challenges: (a) the determination of a proper window size used to collect the local information and (b) the trade-off between detail preservation and noise suppression. These two challenges motivate the research of this paper and are addressed in this section.

The quality of the resultant binary document images produced by the existing adaptive binarization methods often are very sensitive to the window size used [2, 3, 4, 5]. Proper window size usually depends on the scale of objects in the document images. The document images with large objects require large window size in the adaptive binarization scheme.

Binarizing an image as shown in Figure 1 (a) with large objects using smaller than necessary window size may erroneously binarize foreground pixels to background pixels as shown in Figure 1 (b). Figure 1 (c) illustrates a better binarized result of Figure 1 (a) when a large windows size is used. However, adaptive binarization scheme with larger than necessary window size will not significantly increase the quality of the binarized images, as shown in Figure 1 (e) and (f), but incurs higher computational cost.

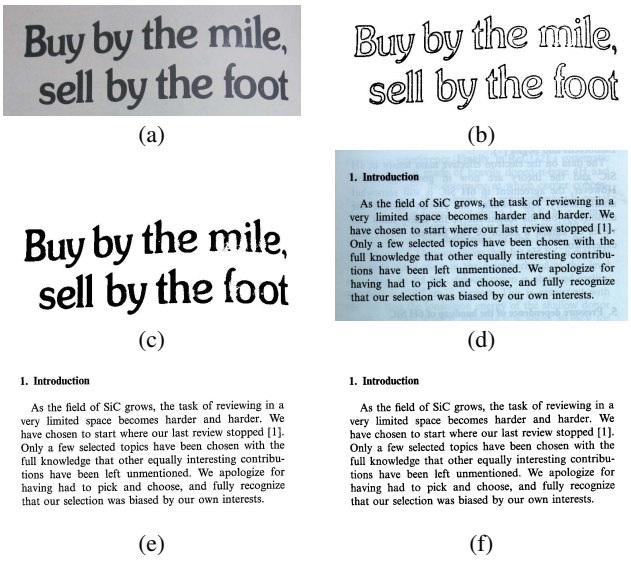


Figure 1: The effect of window size when using Sauvola and Pietikainen's method. (a) Document image with large-scale characters. (b) Binarized image of (a) using a 9×9 window. (c) Binarized image of (a) using a 33×33 window. (d) Document image with small-scale characters. (e) Binarized image of (d) using a 9×9 window. (f) Binarized image of (d) using a 33×33 window.

For proper window size, Gatos *et al.* [5] suggest that window size should cover at least 1 to 2 characters. However, detecting character size usually requires image segmentation and is difficult for degraded documents. Chen *et al.* [7] apply a 3×3 window and determine two thresholds based on the edge pixels detected by the Canny edge detector. The quality of the binarized image heavily depends on the correctness of the edge map which is usually poor for degraded documents. Moghaddam and Cheriet [8] propose a scheme which starts with a large window size determined by the average line height of the input document image and iteratively reduces to a proper window size. Since the average line height is usually determined by the image segmentation process and the proposed scheme suffers from the same problem as Gatos *et al.*'s method.

Free from other image pre-processings, we first apply Otsu's method to obtain a rough foreground image and then determine a proper window size based on the change rate of the variation of the foreground intensities within each window. In addition to determining the proper window size, the trade-off between the preservation of detailed contents and noise suppressing should be addressed in the adaptive binarization scheme.

Let f be the input document image and the intensity value of the pixel at position (x, y) is denoted by $f(x, y)$, $0 \leq f(x, y) \leq 1$. Given a specific window of size $w \times w$ with $w = 2r + 1$, the threshold used for binarization in Niblack's method is expressed as

$$T_{Nib,w}(x, y) = \mu_w(f, x, y) + k\sigma_w(f, x, y) \tag{1}$$

where k is a user-defined parameter and $\mu_w(f, x, y)$ and $\sigma_w(f, x, y)$ represent respectively the mean and standard deviation of intensities of the pixels within the window centered at (x, y) and can be expressed as

$$\mu_w(f, x, y) = \frac{1}{w^2} \sum_{i=-r}^{r} \sum_{j=-r}^{r} f(x + i, y + j), \tag{2}$$

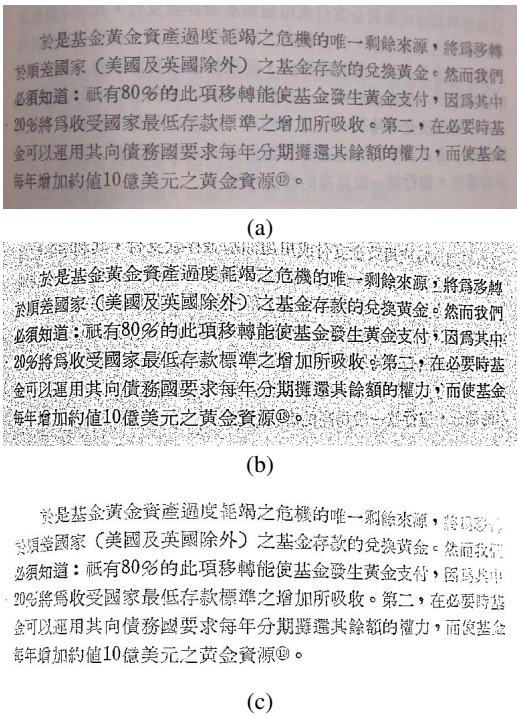


Figure 2: The effect of k' when using Sauvola and Pietikainen's method. (a) Degraded document image. (b) Binarized image using $k' = 0.01$. (c) Binarized image using $k' = 0.2$.

$$\sigma_w(f, x, y) = \sqrt{\frac{1}{w^2} \sum_{i=-r}^{r} \sum_{j=-r}^{r} (f(x + i, y + j) - \mu_w(f, x, y))^2}. \tag{3}$$

To improve Niblack's method, Sauvola and Pietikainen [3] proposed a modified threshold $T_{Sau,w}(x, y)$ which is expressed as

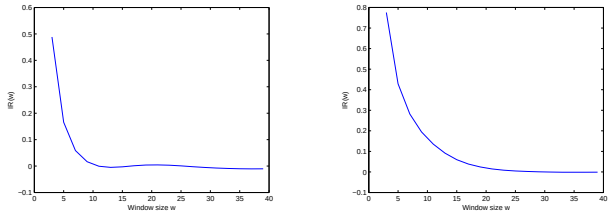
$$T_{Sau,w}(x, y) = \mu_w(f, x, y) \times \left(1 - k' \left(1 - \frac{\sigma_w(f, x, y)}{R}\right)\right) \tag{4}$$

where both R and k' are set to 0.5 in [3].

Parameters k and k' used in Eq. (1) and Eq. (4), respectively, are sensitive to the contents of the input document images and may not be applicable for degraded document images. For example, for a degraded document image in Figure 2 (a), Figure 2 (b) and (c) are binarized images obtained by Sauvola and Pietikainen's method with $k' = 0.01$ and $k' = 0.2$, respectively. The binarized image with smaller k' preserves more detailed contents but suffers from more noises. This observation motivates using two thresholding schemes to produce two binarized images from which the final binarized image is delivered.

3. THE PROPOSED TWO-STAGE AND PARAMETER-FREE BINARIZATION METHOD

In this section, we present a two-stage and parameter-free binarization scheme for degraded document images. The first stage determines a proper window size by considering the variation of foreground pixel intensities within windows. In the second stage, based on the window size determined in stage 1, a final binarized image is delivered by contrasting two binarized images produced by two adaptive thresholding schemes which consider the content preser-



(a) (b)

Figure 3: $IR(w)$ for documents in Figure 1(a) and (d)

vation and noise suppressing.

3.1 Determine the proper window size

To start the two-stage binarization scheme, we first apply the Gaussian low-pass filter to obtain the smoothed image and then the Otsu's method is used to determine the set of the rough foreground pixels, denoted by RFG. The variation of foreground pixel intensities within each window usually increases as the window size increases. Large window size usually delivers binarized images with better quality but suffers from larger computational cost, indicating that the window size larger than necessary for acceptable quality should not be adopted.

Since binarizing with small window size may erroneously binarize foreground pixels to background pixels and using large window size increases the computational cost without significantly increasing the quality, we start with a small window size and keep increasing the window size until no substantial increase in the variation of the pixel intensities within each window is observed.

Starting with a window of size 3×3, we compute the standard deviation of the foreground pixel intensities within each window and use the average of the standard deviations as the indicator to search for the proper window size.

Let $IR(w)$ denote the increasing rate of the average standard deviation when enlarging the window from $w \times w$ to $(w+2) \times (w+2)$ and is expressed as

$$IR(w) = \frac{\overline{\sigma}_{w+2} - \overline{\sigma}_w}{\overline{\sigma}_w} \tag{5}$$

with

$$\overline{\sigma}_w = \frac{1}{|RFG|} \sum_{(x,y) \in RFG} \sigma_w(f, x, y), \tag{6}$$

where $|RFG|$ is the cardinality of the set of rough foreground pixels RFG and $\sigma_w(f, x, y)$ is the standard deviation of pixel intensities within the $w \times w$ window centered at (x, y). The increasing rate $IR(w)$ decreases as the window size w increases as shown in Figure 3. The proper window size w^* is the smallest window size such that $IR(w)$ is less than or equal to 0.01; that is, $w^* = \min\{w : IR(w) \leq 0.01\}$.

3.2 Proposed noise-suppressing thresholding scheme

For low contrasting documents, information contained in the neighborhood of a specific pixel can be helpful in determining the binarization threshold. Let $g(x, y)$ denote the gradient magnitude, proposed by Sobel operator [9], formed from the pixels in the neighborhood of pixel (x, y). Large values of $g(x, y)$ indicate that pixel (x, y) is around the boundary between foreground and background pixels. Based on the window size $w^* = 2r^* + 1$ determined in stage 1, compute

$$\mu_{w^*}(g, x, y) = \frac{1}{w^{*2}} \sum_{i=-r^*}^{r^*} \sum_{j=-r^*}^{r^*} g(x+i, y+j), \tag{7}$$

and

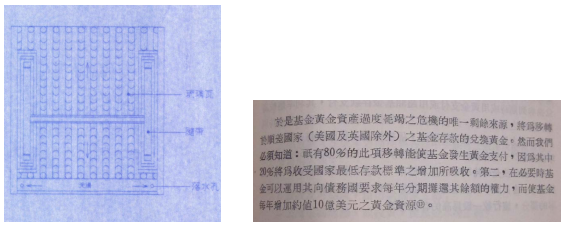
$$\sigma_{w^*}(g, x, y) = \frac{1}{w^*} \sqrt{\sum_{i=-r^*}^{r^*} \sum_{j=-r^*}^{r^*} (g(x+i, y+j) - \mu_{w^*}(g, x, y))^2} \tag{8}$$

When incorporating the information contained in the mean $\mu_{w^*}(g, x, y)$ and the standard deviation $\sigma_{w^*}(g, x, y)$ of local gradients around pixel (x, y), we propose a binarization threshold

$$T(x, y) = \mu_{w^*}(f, x, y)\big(1 - k'' e^{-\left((\mu_{w^*}(g,x,y) + \sigma_{w^*}(g,x,y))/M\right)}\big), \tag{9}$$

where $M = \max_{(x,y) \in D}\{\mu_{w^*}(g, x, y) + \sigma_{w^*}(g, x, y)\}$ with D denoting the input document. Decreasing parameter k'' increases the threshold $T(x, y)$ and the number of pixels identified as foreground pixels increases. Thus when decreasing parameter k'' from some large initial value, the number of identified foreground pixels increases sharply and becomes saturated as most of the true foreground pixels are correctly identified as foreground pixels. If keep decreasing the parameter k'', the number of identified foreground pixels may increase sharply again since the present noises are erroneously identified as foreground pixels. Let $|FG|(k'')$ denote the number of identified foreground pixels using threshold $T(x, y)$ on pixel $f(x, y)$. Starting with $k''_0 = 0.2$, iteratively decrease the threshold by modifying the parameter k'' according to $k''_{i+1} = (0.9)k''_i$. Denote by k''_{1*} and k''_{2*}, with $k''_{1*} > k''_{2*}$, the reflection points of the function $|FG|(k'')$; that is, $\frac{d^2|FG|(k'')}{dk''^2}\big|_{k''=k''_{1*}} = \frac{d^2|FG|(k'')}{dk''^2}\big|_{k''=k''_{2*}} = 0$. Two thresholds $T_1(x, y)$ and $T_2(x, y)$ for pixel (x, y) are determined by Eq. 9.

Let B_i denote the binarized image produced by the threshold $T_i(x, y)$, $i = 1, 2$. Since $T_1(x, y) < T_2(x, y)$, in image B_1 noises are suppressed but some true foreground pixels are not correctly identified. On the other hand, in image B_2 almost all the true foreground pixels are identified but in the meantime the noises are about to be included. Two binarized images B_1 and B_2 are then contrasted to deliver the final binarized image. Since $T_1(x, y) < T_2(x, y)$, if (x, y) is a background pixel in B_2, then (x, y) must be a background pixel in B_1 and is very likely to be a true background pixel in the document. Thus the pixel appeared to be a background pixel in B_2 will be identified as a background pixel in the final binarized image. Similarly, if (x, y) is a foreground pixel in B_1, then (x, y) must be a foreground pixel in B_2 and is very likely to be a true foreground pixel in the document. Thus the pixel appeared to be a foreground pixel in B_1 will be identified as a foreground pixel in the final binarized image. If (x, y) is a background pixel in B_1 and a foreground pixel in B_2, then it can be a foreground or a noise in the document. To tackle such pixels, a region-growing scheme, based on the pixels which are identified as foreground pixels in both B_1 and B_2, is proposed. For each pixel (x, y) identified as a foreground pixel in both B_1 and B_2 images, a 3×3 window centered at (x, y) is considered. Within the window, for each of the eight pixels surrounding (x, y), if it is identified as a background in B_1 and a foreground in B_2, then it is identified as a foreground pixel in the final binarized image. Furthermore, the region-growing scheme will be applied to the newly identified foreground pixel by the region-growing scheme. This proposed region-growing scheme mends some true foreground pixels that are suppressed in B_1 when suppressing the noises.



(a) Test image 1 ($\#FG/N = 11.03\%$)

(b) Test image 2 ($\#FG/N = 13.12\%$)

Figure 4: The test images

Table 1: The performance comparison of test image 1

Method	Precision	Recall	Accuracy	F-measure
[3]	69.6861	78.4149	93.8572	73.7932
[4]	74.8738	57.3418	92.8991	64.9454
[5]	75.6668	65.1826	93.8480	70.0345
[7]	58.9203	87.3272	91.8872	70.3649
[8]	62.6726	83.4845	92.6945	71.5968
Proposed	83.0143	79.3807	95.9345	81.1568

4. EXPERIMENTAL RESULTS

In this section, we empirically compare the proposed method with six existing methods — Sauvola and Pietikainen's method [3], Zhao *et al.*'s method [4], Gatos *et al.*'s method [5], Chen *et al.*'s method [7], and Moghaddam and Cheriet's method [8]. All methods are implemented by Borland C++ Builder 6.0 and run on a standard PC with AMD Athlon 64X2 4800+ CPU(2.5 GHz) and 1.87 GB of RAM. The test images include scanned machine-printed image and blueprint image for which we create the true binary image by human eyes. Test images 1 in Figure 4 is the blueprint image of architectures with proportion $\#FG/N$ of foreground pixels, where $\#FG$ is the number of true foreground pixels in the document with N pixels. Test images 2 in Figure 4 is a textual image with non-uniform illumination. The performance evaluations are based on four accuracy measures: (a) recall, (b) precision, (c) accuracy, and (d) F-measure. Recall is the proportion of correctly binarized foreground pixels within the true foreground pixels. Precision is the proportion of true foreground pixels within the binarized foreground pixels. Accuracy is the weighted average of the proportions of correctly binarized foreground and background pixels within the true corresponding pixels with weights proportional to the numbers of true foreground and background pixels. The F-measure is the harmonic mean of recallz and precision. Let TP and TN denote respectively the number of pixels that are correctly binarized as foreground and background pixels. And denote respectively by FP and FN the number of pixels that are erroneously binarized as foreground and background pixels. Then we have recall $= TP/(TP + FN)$, precision $= TP/(TP+FP)$, accuracy $= (TP+TN)/(N)$, and F-measure $= 2 \times$ recall $\times$ precision$/$(recall $+$ precision). Empirical results are listed in Tables 1 and 2 respectively.

Based on the empirical results, the following general conclusions are obvious:

1. The proposed method has significantly higher F-measure than the existing methods, indicating that the proposed method achieves higher accuracy in both recall and precision.

2. In terms of accuracy, the proposed method is competitive when compared to the existing methods.

Table 2: The performance comparison of test image 3

Method	Precision	Recall	Accuracy	F-measure
[3]	96.5006	72.2408	96.0135	82.6268
[4]	85.1910	60.3617	93.4216	70.6586
[5]	90.8324	85.6041	96.9771	88.1408
[7]	43.8361	89.6292	83.5699	58.8767
[8]	74.9070	93.0555	94.9981	83.0008
Proposed	89.1313	88.9508	97.1267	89.0409

3. For highly degraded documents such as test blueprint image 1, results in Tables 1 show that the proposed method achieves higher precision with acceptable recall compared to the existing methods.

5. CONCLUSION

In this paper, a two-stage parameter-free window-based binarization method is proposed. In general, the proposed binarization scheme is competitive when compared with the existing methods. Specifically, the proposed method has good performance in both recall and precision measures, resulting a higher F-measure.

6. ACKNOWLEDGEMENTS

K.-L. Chung, Y.-H. Huang, and W.-N. Yang are supported by the National Science Council of R.O.C. under Contract NSC98-2923-E-011-001-MY3, NSC99-2221-E-228-006, and NSC100-2218-E-011-006 respectively.

7. REFERENCES

[1] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Transcations on Systems, Man and Cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.

[2] W. Niblack, "An introduction to digital image processing," Prentice Hall, Englewood Cliffs, NJ, pp. 115–116, 1986.

[3] J. Sauvola and M. Pietikainen, "Adaptive document image binarization," *Pattern Recognition*, vol. 33, no. 2, pp. 225–236, 2000.

[4] M. Zhao, Y. Yang and H. Yan, "An adaptive thresholding method for binarization of blueprint images," *Pattern Recognition Letter*, vol. 21, no. 10, pp. 927–943, 2000.

[5] B. Gatos, I. Pratikakis and S. J. Perantonis, "Adaptive degraded document image binarization," *Pattern Recognition*, vol. 39, no. 3, pp. 317–327, 2006.

[6] J. Canny, "A computational approach to edge detection," *IEEE Transcations on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, no. 6, pp. 679–698, 1986.

[7] Q. Chen, Q. S. Sun, P. A. Heng and D. S. Xia, "A double-threshold image binarization method based on edge detector," *Pattern Recognition*, vol. 41, no. 4, pp. 1254–1267, 2008.

[8] R. F. Moghaddam and M. Cheriet, "A multi-scale framework for adaptive binarization of degraded document images," *Pattern Recognition*, vol. 43, no. 6, pp. 2186–2198, 2010.

[9] R. C. Gonzalez and R. E. Woods, "Digital Image Processing," Section 7:1.3: Edge Detection, Addison Wesley, 1992.

ABOUT THE AUTHORS

Yung-Hsiang Chiu is now a Ph.D. student in the Department of Computer Science and Information Engineering of National Taiwan University of Science and Technology, Taiwan. His research interests include document image processing and advance video coding. His contact email is D9915013@mail.ntust.edu.tw.

Kuo-Liang Chung is a Chair Professor in the Department of Computer Science and Information Engineering of the National Taiwan University of Science and Technology, Taiwan. His current research interests include image processing, video coding, and data hiding. His contact email is k.l.chung@mail.ntust.edu.tw.

Yong-Huai Huang is an assistant professor in the Institute of Computer and Communication Engineering at Jinwen University of Science and Technology, Taiwan. His research interests include image processing and compression, and algorithms. His contact email is yonghuai@ms28.hinet.net.

Wei-Ning Yang is now an associate professor in the Department of Information Management, National Taiwan University of Science and Technology, Taiwan. His research interests include statistical analysis, stochastic simulation, and image processing. His contact email is yang@cs.ntust.edu.tw.

Chi-Huang Liao is now the manager at System Online Co. Ltd., Taiwan. His research interests include geographic information system, system integration, and image processing. His contact email is chin.laiw@msa.hinet.net.

Intellectual Two-sided Card Copy

Ilia Safonov[a], Hokeun Lee[b], Sangho Kim[b], Donchul Choi[b]
[a]Samsung Research Center, Moscow, Russia
[b]Samsung Electronics, Suwon, Korea
ilia dot safonov at samsung dot com

Abstract

The paper is devoted to intellectual algorithm for processing of scanned images of various two-sided cards such as ID, bank, business cards, etc. Card images significantly differ from common document images and conventional approaches developed for text document image processing do not operate well on card images. Our technique provides a sophisticated arrangement of both images of a card, front and back, for single-page and duplex printing. The method detects location, skew angle and orientation of card image. Orientation detection is based on Arabic numerals segmentation and recognition. Recognition of numerals on card images is a challenging problem, since often cards have complex color background; digits can have arbitrary color and typeface. For digits segmentation on complex background we apply labeling of connected edge regions, whereas edge pixels are labeled for dark numerals on light background and *vice versa* independently from each other. We construct a decision tree with a great generalization capability for recognition of each numeral; it improves recognition for various typefaces. We propose a final decision rule for image orientation detection based on recognition outcomes.

Our algorithm operates very fast and outperforms OCR applications ReadIRIS and FineReader in numerals recognition and orientation detection for card images. Also our algorithm works well enough for majority of originals.

Keywords: *orientation detection, segmentation, numerals recognition.*

1. INTRODUCTION

Frequently consumers copy various two-sided card-size originals such as identity cards, passports, bank cards, certificates, charge cards, business cards, etc. Samsung Electronics has implemented a special "ID Copy" function in MFP devices since 2004 year. User places one card side on scanner glass that corresponds to top half of A4 page and presses copy button. Image of card side is scanned and stored to MFP memory. Further user places another card side on scanner glass and presses copy button again. Image of another card side is scanned. Page is rendered from the two images of both card sides. The function operates well enough for all types of originals.

However existing "ID Copy" function has some limitations. For example, images of card sides are copied without any correction of skew and orientation. Each side can have a skew, and copy looks unattractive. Moreover it can appear in opposite orientations or one side in landscape and other side in portrait orientation. So, our aim is creation of an "Intellectual ID Copy" algorithm providing a sophisticated arrangement of images of both sides for single-page and duplex printing. The method should perform parallel translation and deskew depending on orientation of card image. Fig. 1 illustrates intellectual two-sided card copy. Two scanning passes provide images of front and back sides of card in arbitrary orientation. On a copy both sides are placed upright according to predefined layout.

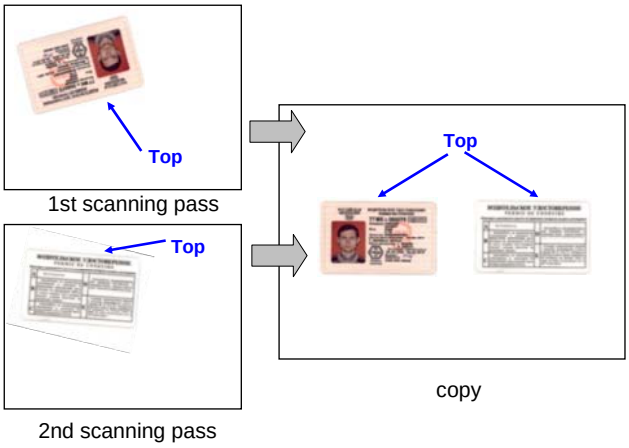


Figure 1: Illustration of *Intellectual ID Copy* function.

There are strict limitations on computational complexity of an algorithm and memory size because it is intended for embedded systems on ARM and MIPS-based SoC. Several hundreds kilobytes of RAM are available only; and all image transformation procedures should be performed *in situ*.

2. WAYS FOR ORIENTATION DETECTION

Key problem of intellectual copy is orientation detection for 4 possible orientations. There are a lot of publications devoted to this problem for various types of images. Initially we collected test set from about 300 card images and tried various approaches for orientation detection.

Some existing techniques on orientation detection, for example, [9, 10], are aimed at photographic images and strongly depend on background of the card, therefore in general case they are unfeasible for card images Another approach could use facial images for orientation detection, since almost all ID cards have face photo. However, face detector from OpenCV produced a lot of false positives in our ID cards images in various orientations. Additional conditions on skin tones and face size as in [3] allowed to decrease significantly number of FP, but several faces were detected in upright and upturned orientation identically. Fig. 2 demonstrates example of face detection for ID card rotated by 180 degrees.

Paper [1] describes computationally effective algorithm for orientation detection of text document images. The algorithm exploits an up/down asymmetry of text composed of roman letters

and Arabic numerals. However, majority of card images has complex color background; therefore detection quality of the algorithm is low. In addition language origin significantly influences its detection quality even on uniform background.

Figure 2: Example of face detection for upturned card.

OCR applications such as ReadIRIS and FineReader can detect image orientation based on results of characters recognition for all possible orientations. But presently embedding of full-functional OCR for many languages in devices for low- and middle-end markets is unfeasible. However, implementation of some reductive character recognition procedure makes sense, since, as it can be noticed, absolute majority of cards contains several Arabic numerals.

We can recognize the following numerals which are rotation noninvariant: '1', '2', '3', '4', '5', '7'. Of course, such approach has some limitations. For example, sometimes cards do not contain any numerals or have rotation invariant numerals only. Also, cards can have numerals in mutually perpendicular directions. Fortunately, these are very rare cases: in our test set only about 1% of images belong to these cases. So, we selected Arabic numerals as the main attribute for card image orientation recognition.

3. PROCESSING OF CARD IMAGES

3.1 General workflow

Front and back sides of card are processed identically. Fig. 3 demonstrates general processing pipeline for image of one side. First of all, bounding box is detected. We slightly modified traditional robust approach for RoI segmentation that employs analysis of projections on horizontal and vertical axes [6]. The next step is detection of skew-angle. Straightforward techniques for skew-angle detection is to use projection profile analysis or Hough transform. Comprehensive survey of skew-angle detection problem was presented in [4]. Unfortunately, well-known approaches do not work well enough for card images which have complex color background. So, we proposed our own skew-angle detection algorithm that is discussed in separate paper [7]. Third step is deskew and translation by means of in place affine transformations [2]. Orientation detection comprises numerals segmentation, recognition for four directions and making decision basing on recognition results. Finally, image is rotated if it is necessary.

3.2 Bounding Box detection

Projections on horizontal and vertical axes are computed according to the following formula:

$$Ph(c) = \sum_{r=1}^{Nr} BW(r,c); \quad Pv(r) = \sum_{c=1}^{Nc} BW(r,c),$$

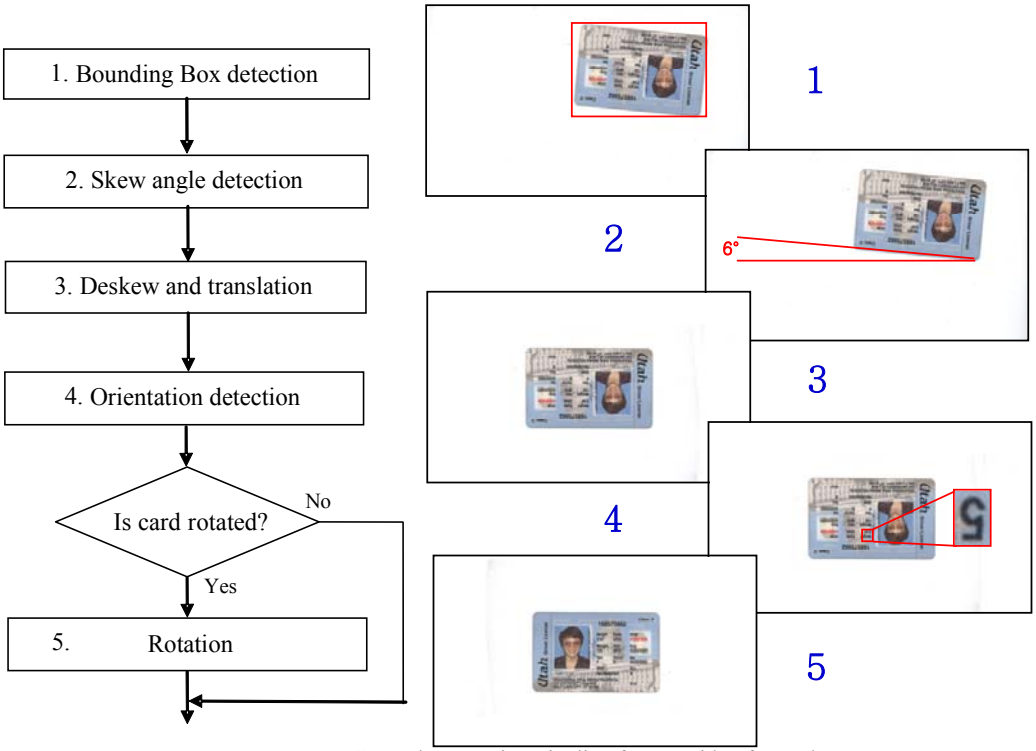


Figure 3. General processing pipeline for one side of a card.

where (r,c) are pixel coordinates, Nr is number of rows, Nc is number of columns, BW is binary image that is the result of scanned image thresholding. On this stage we use fixed threshold because we precisely know scanner characteristics. In order to suppress some noise peaks we apply 1D erosion filter to both projections:

$$Phf(c) = \min(Ph(c-2), Ph(c-1), Ph(c), Ph(c+1), Ph(c+2)),$$
$$Pvf(r) = \min(Pv(r-2), Pv(r-1), Pv(r), Pv(r+1), Pv(r+2)),$$

Bounding box coordinates are calculated as follows:

$$r_{\min} = \min_{\forall r}(Nr - h, r \,|\, Pvf(r) > Nc/k),$$

$$r_{\max} = \max_{\forall r}(h, r \,|\, Pvf(r) > Nc/k),$$

$$c_{\min} = \min_{\forall c}(Nc - h, c \,|\, Phf(c) > Nr/k),$$

$$c_{\max} = \max_{\forall c}(h, c \,|\, Phf(r) > Nr/k),$$

where coefficient k depends on scanning resolution, $h>0$ is introduced to prevent detection of possible shadow areas near platen boundaries.

We implemented two additional functions based on bounding box coordinates: blank page detection and image clipping detection: If $r_{\min} > r_{\max}$ or $c_{\min} > c_{\max}$ then image is blank page; If $r_{\max} == Nr$ then it is possible that card image is clipped in central part of scanner platen.

3.3 Numerals segmentation

There are several challenges for numerals segmentations: often cards have complex color background; numerals can be dark on light background or they can be light on dark background; typefaces can differ significantly. To segment digits on complex background we apply two labeling procedures to grayscale image that is a result of high-pass filtering of brightness channel of deskewed image.

Fig. 4 shows flow-chart of our numerals segmentation procedure. At the beginning we process image by FIR high-pass filter. Convolution kernel here looks like Laplacian with negative central element and depends on scanning resolution. Next step is labeling of connected regions for pixels of filtered image which are greater then threshold T1. These pixels correspond to inner edges of dark symbols on light background. Third step is labeling of connected regions for pixels which are less then predefined threshold -T1. These pixels correspond to inner edges of light symbols on dark background. For each connected region from both labeled images bounding box is calculated; and regions with appropriate bounding box size and aspect ratio are selected. Thresholds for bounding boxes selection criteria are chosen keeping in mind typical size of numerals on card images from 5 to 20 pt.

Fig. 5 illustrates numerals segmentation steps. To minimize memory requirements and processing time we implement two-pass labeling scheme. First pass is HPF and calculation of bounding boxes of connected regions for dark and light symbols separately. Second pass is labeling of selected appropriate regions. Such approach is different from conventional ways, which use global or local tresholding of brightness channel as, for example, in [5]. As a result of our segmentation bodies of numerals contain a lot of holes and boundary breaks. However following recognition procedure is robust to such possible defects. Great advantage of our technique is successful segmentation both dark and light symbols on complex color background.

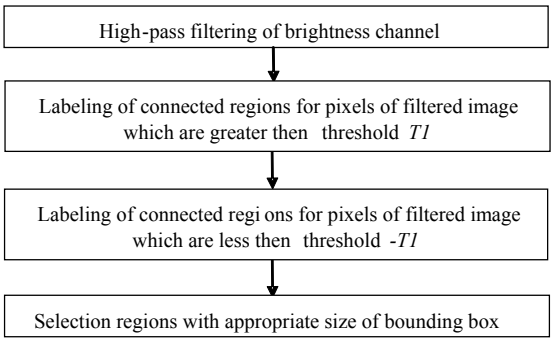


Figure 4: Flow-chart of numerals segmentation.

Figure 5: Illustration of numerals segmentation steps.

3.4 Arabic numerals recognition

Initially we stored about 50000 images with connected regions from our test card images by means of segmentation method as described above. Further images with numerals were selected manually. We tried several machine learning techniques for classifiers construction. However, those classifiers tend to overfitting: quite frequently symbols with typefaces absent in training set triggered false negative errors. So, for recognition of each numeral we constructed a decision tree with a great generalization capability. For some digits we created multiple trees, and it significantly improved recognition capability, allowing correct answers for numerals of completely different typefaces and even handwriting numerals.

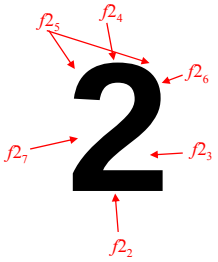


Figure 6: Features for '2' numeral.

For recognition of digit '2' we calculate the following features, which are shown on fig. 6:

$$f2_1 = Mr / Mc,\quad f2_7 = \sum_{r=Mr/2}^{2Mr/3} \sum_{c=1}^{2Mr/3-r+1} F(r,c),$$

$$f2_2 = \max(\sum_{r=Mr-1}^{Mr} \sum_{c=1}^{Mc} F(r,c); \sum_{r=Mr-2}^{Mr-1} \sum_{c=1}^{Mc} F(r,c)) / (2Mc),$$

$$f2_3 = \sum_{c=3Mr/5}^{3Mr/4} \sum_{r=0}^{c-3Mr/5} F(Mr-r,c),$$

$$f2_4 = \sum_{r=1}^{2} \sum_{c=1}^{2} F(r,c) + \sum_{r=1}^{2} \sum_{c=Mc-1}^{Mc} F(r,c),$$

$$f2_5 = \max(\sum_{r=1}^{2} \sum_{c=Mc/3}^{2Mc/3} F(r,c); \sum_{r=2}^{3} \sum_{c=Mc/3}^{2Mc/3} F(r,c)) / (2(Mc/3+1)),$$

$$f2_6 = \max(\sum_{r=Mr/5}^{2Mr/5} \sum_{c=Mc-1}^{Mc} F(r,c); \sum_{r=Mr/5}^{2Mr/5} \sum_{c=Mc-2}^{Mc-1} F(r,c)) / (2(Mr/5+1)),$$

where F is binary image of connected region, Mr x Mc is size of the image F.

If the following rule is true then connected region F corresponds to '2': $f2_1$>=1.1 and $f2_1$<=2.2 and $f2_2$>=0.65 and $f2_3$==0 and $f2_4$==0 and $f2_5$>0.65 and $f2_6$>0.55 and $f2_7$==0. Similar feature sets and decision trees were created for other numerals [8].

3.5 Orientation detection

We recognize numerals for each connected region for four orientations 0°, 90°, 180°, 270° and calculate number of recognized regions. Finally we make decision about card image orientation Ω based on numbers of recognized regions for each orientation according to the formula:

$$\Omega = \arg\max_{i \in \{0,90,180,270\}} (i\ (w_1 N_1(i) + w_2 N_2(i) + w_3 N_3(i) + w_4 N_4(i) +$$
$$+ w_5 N_5(i) + w_7 N_7(i)) \times (sign(N_1(i)) + sign(N_2(i)) + sign(N_3(i)) +$$
$$+ sign(N_4(i)) + sign(N_5(i)) + sign(N_7(i)))),$$

where $N_1(i)$, $N_2(i)$, $N_3(i)$, $N_4(i)$, $N_5(i)$ and $N_7(i)$ are numbers of recognized regions as numerals '1', '2', '3', '4', '5', '7' accordingly for orientation I; $w_{1\text{-}7}$ are weights, which are defined during testing on our set; w_1 and w_7 are smaller than other weights because probability of false positive error is higher for '1' and '7' in comparison with other digits. Detection of several various digits indicates orientation with high probability because sum of recognized regions for each digit is multiplied by quantity of recognized various numerals.

Obviously, the technique can be extended easily by means of recognition of '0', '6', '8' and '9' numerals for 90° orientation. Also there is no problem to add recognition of several characters.

4. RESULTS AND CONCLUSION

Our algorithm operates very fast. Processing time for 6.2 MPix (2500 x 2500) RGB image is in range 0.2 - 0.3 s on PC with single-core P4 3.2 GHz. Application requires about 165 Kb of additional memory only.

99% of card images with numerals from our test set are processed correctly. We compared our digits recognition, orientation detection for upturned images and deskew outcomes for 75 images with the results of well-known OCR applications ReadIRIS 11 and FineReader 10. Totally 1137 concerned digits are on the test images. In addition OCR applications recognized several English characters. Our approach detected numerals only. Table 1 contains comparison of operation quality for our algorithm, FineReader and ReadIRIS. Our method outperforms respected OCR applications. It is clear that those OCR packages have a lot of features and are not specifically intended for processing of card images. However, proposed algorithm demonstrates the same advantages in processing any originals with complex color background.

Our balanced solution allows to create intellectual two-side card copy function for modern MFP devices. Such functionality is attractive for consumers.

TABLE 1 COMPARISON OF NUMERALS DETECTION QUALITY AS WELL AS ORIENTATION DETECTION AND DESKEW

	Numerals detection, %	*Orientation detection, %*	*Deskew, %*
Proposed	87	99	99
ReadIRIS	74	72	69
FineReader	63	69	68

5. REFERENCES

[1] Caprari, R.S., *Algorithm for text page up/down orientation determination*, Pattern Recognition Letters 21, pp. 311-317, 2000.

[2] Catmull, E., Smith, A., *3-D transformations of images in scanline order*, Proc. of SIGGRAPH'80, pp. 279–285, 1980.

[3] Egorova, M.A., Murynin, A.B., Safonov, I.V., *An Improvement of face detection algorithm for color photos*, Pattern Recognition and Image Analysis, vol. 19, No. 4, pp. 634–640, 2009.

[4] Hull, J., Document image skew detection: survey and annotated bibliography, Document Analysis Systems II, pp. 40-64, 1998.

[5] Llados, J., Lumbreras, F., Chapaprieta, V., Queralt, J., ICAR: Identity Card Automatic Reader, Proc. of ICDAR, pp. 470-474, 2001.

[6] Pavlidis, T., Algorithms for graphics and image processing, Springer, 1982.

[7] Safonov, I., Kurilin, I., *Deskew for Card Image Scanning*, Graphicon - 2011.

[8] Safonov, I., Lee, H.K., Kim, S.H., US2011/0141534.

[9] Tolstaya E. *Content-based image orientation recognition,* Graphicon – 2007, pp.158-161.

[10] Wang, Y., Zhang, H. *Content-based image orientation detection*, Proc. of IEEE Workshop on content-based access of image and video libraries, pp. 17-23, 2001.

About the author

Ilia V. Safonov received his MS degree in automatic and electronic engineering from Moscow Engineering Physics Institute/University (MEPhI), Russia in 1994 and his PhD degree in computer science from MEPhI in 1997. Since 2004, Dr. I. Safonov has joint Samsung Research Center, Moscow, Russia where he is engaged in image and video processing projects.

Deskew for Card Image Scanning

Ilia Safonov, Ilya Kurilin
Samsung Research Center, Moscow, Russia
ilia dot safonov at samsung dot com

Abstract

Skew is a common defect of scanned images; therefore, deskew is a mandatory function of any scanning solution. Frequently users scan cards such as identity, bank, charge and business, etc. Often images of cards differ from common document images; and conventional deskew techniques do not perform well enough, especially for cards with light background. We propose a new robust skew-angle estimation algorithm based on iterative analysis of distance arrays for each column from top and bottom of extended bounding box to the nearest foreground pixel. Also rotation in situ is discussed that has minimal requirements to memory and is suitable for implementation in embedded system. Performance of proposed method corresponds to the best solutions in OCR and scanning software.

Keywords: *Deskew, Rotation in place.*

1. INTRODUCTION

Frequently consumers scan and copy various card-size documents such as postage cards and stamps, identity cards, passports, bank cards, certificates, charge cards, business cards, etc. The card image may be skewed by some angle due to inexact placing on the platen of a device. Skewed images should be corrected i.e. *deskew* has to be done. That is why deskew is mandatory function of any scanning solution and sometimes it is embedded in copying devices. Fig. 1 demonstrates deskew of a card image.

Card images have rectangular form like majority of document images, but usually card images significantly differ from conventional text document images: they have complex color background, number of text symbols is relatively small, and size of symbols can vary significantly, a lot of additional graphic elements present. That is why very often well-known deskewing methods fail.

We propose new fast and robust deskew algorithm intended for card images, but it operates well for any types of documents. Our technique is suitable for implementation in embedded systems because it has low computational complexity and requires small amount of memory due to *in situ* rotation approach.

2. RELATED WORK

There are plenty publication devoted to deskew algorithms. The key problem is reliable skew-angle estimation. A survey [4] classifies about 50 techniques in four groups: projection profile analysis, Hough or Radon transforms analysis, feature point distribution and orientation-sensitive feature analysis. All algorithms assume that an input document contains some amount of text. Recent publications [1, 3, 5, 6, 8] contain some improvements and specifications, but, in general, they use similar assumptions and approaches. A lot of modern techniques rely on text areas detection as first step and make skew-angle estimation for those areas only.

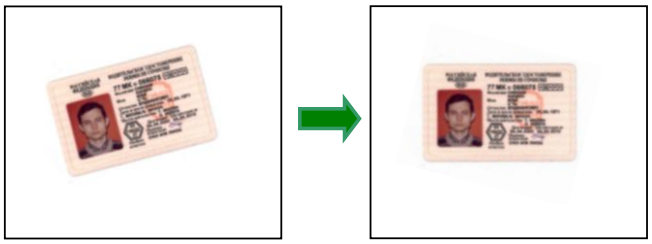


Figure 1: Example of deskew of card image.

Initially we collected test set from about 300 skewed card images and tried to apply those four general approaches to the test images. We did not do formal measurement of outcomes, but we evaluated applicability of each approach for our task only.

A straightforward solution for determining the skew-angle of a binary document image is to use a projection profile. Each element in the vertical projection profile is a count of the number of black pixels in the corresponding row of the image. This profile has the maximum energy when the text lines lie horizontally. In order to find skew-angle, the image or projection axis is rotated by a small step. Usually card image contains a lot of various objects on complex color background. It leads to noisy objects in binary image around and between text lines. Frequently maximum of projection profile energy does not correspond to zero-angle. Only about for 50% of card images skew-angle can be estimated with high precision by means of projection profile analysis.

Preliminary detection of text blocks makes outcomes a little bit better. For document segmentation we used approach described in [7]. Its performance corresponds to well-known OCR applications FineReader and ReadIRIS, but for images with complex background it is too hard to segment text areas only. In this way skew-angle is estimated well enough for about 65% of card images; and this result is not acceptable. Moreover, projection profile calculation for a great number of angles requires big computational resources. So, projection profile analysis is not applicable for our task.

Another class of technique for skew-angle detection reduces the number of operations that are performed in a projection analysis by first extracting the x-y coordinates of connected regions in an image. All subsequent computations are performed on those coordinates. Nearest objects are connected in chains or graphs. Skew-angle is calculated by means of approximation, for example by application of LSM. This technique is faster in comparison with profile analysis, but nevertheless computational complexity is high because labeling, chains and approximations are complex procedures. Skew-angle estimation outcomes are not so good too due to the same complexities.

The Hough transform is a well-known technique. In particular, it is used to detect straight lines. Techniques based on Hough transform operate well when edge detection is a preprocessing step to obtain binary image. Outer edges of cards have maximum magnitude in Hough space. Unfortunately, for cards with white background often outer edges are not detected at all. In this case inner noisy objects can lead to wrong skew-angle estimation. Fig. 2 shows two examples of improper skew-angle estimation by lines detection via Hough transform. Detected lines are marked by green. Only part of the lines is applicable for skew-angle estimation. Nevertheless for our test set the method based on Hough-transform allows to estimate properly about 95% of card images. Computational complexity of the algorithm for high-resolution image is high enough. Usually downsampled image is carried out to decrease processing time.

Figure 2: Examples of improper skew-angle estimation by lines detection via Hough transform.

Another type of approach for determining the skew angle detects the presence of local orientation-sensitive features in an image and uses their angles to vote for the skew-angle estimation. Such approach combines aspects of the feature extraction techniques and methods based on Hough transform. Several our modifications of orientation-sensitive feature analysis provided correct skew-angle estimation above 90%. Such way is promising; and we generated our own approach based on main concepts of orientation-sensitive feature analysis.

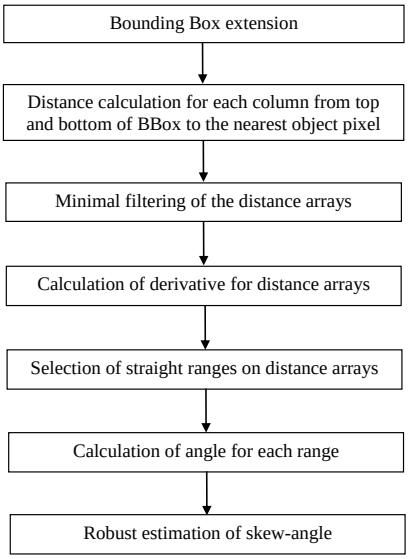


Figure 3: General scheme of skew-angle estimation.

3. SKEW-ANGLE ESTIMATION

3.1 General workflow

Let's we have card binary image BW. This image is computed by thresholding the brightness channel or edge detection. Usually combination of both techniques provides a better result. Bounding box is calculated via projection profile analysis. Fig. 3 demonstrates general scheme of skew-angle estimation.

In order to guarantee presence of image of entire card side inside bounding box we first extend bounding box:

$$ce_{\min} = \max(1, c_{\min} - (c_{\max} - c_{\min})/4),$$

$$ce_{\max} = \min(Nc, c_{\max} + (c_{\max} - c_{\min})/4),$$

$$re_{\min} = \max(1, r_{\min} - (r_{\max} - r_{\min})/4),$$

$$re_{\max} = \min(Nr, r_{\max} + (r_{\max} - r_{\min})/4),$$

where c is column index, r is row index. Nr – number of rows in image; Nc – number of columns in image, c_{min}, r_{min} are coordinates of left-top corner of initial bounding box, c_{max}, r_{max} are coordinates of right-bottom corner.

We calculate skew-angle by means of analysis of distances from top and bottom of extended bounding box to the nearest foreground pixel (see fig. 4). Such approach detects angle of outer edges when they are present. If background of card image is white then distance arrays reflects angle of inner objects. Iterative statistical estimation of angles of some number of objects allows to find skew-angle with high precision or to make decision about absence of skew or non-rectangular form of the image.

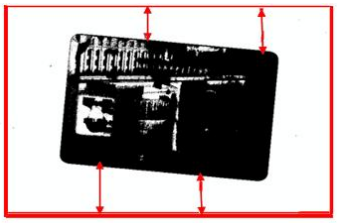


Figure 4: Distances from top and bottom of extended bounding box to the nearest foreground pixel.

3.2 Distances calculation

Arrays of distance from top and bottom of bounding box correspondingly are calculated by the formulae:

$$Dt(c) = \min((re_{\max} - re_{\min}), \min_{r \in [re_{\min}, re_{\max}]}(r \mid BW(r,c) = 1)),$$

$$Db(c) = \min((re_{\max} - re_{\min}), \max_{r \in [re_{\min}, re_{\max}]}(r \mid BW(r, ce_{\max} - c) = 1)),$$

where $c \in [ce_{\min}, ce_{\max}]$.

In order to suppress some noisy peaks we apply 1D erosion filter to both arrays:

$$Dtf(c) = \min(Dt(c-h), ..., Dt(c), ..., Dt(c+h)),$$

$$Dbf(c) = \min(Db(c-h), ..., Db(c), ..., Db(c+h)),$$

where aperture depends on scanning resolution, for 300 dpi $h = 4$.

3.3 Selection of straight line ranges

In order to detect straight ranges on distance arrays we calculate derivatives for these arrays:

$$dDtf(c) = Dtf(c+h) - Dtf(c-h),$$

$$dDbf(c) = Dbf(c+h) - Dbf(c-h).$$

Strictly speaking, dDtf and dDbf are not derivatives but finite differences. However, similarly to a lot of publications in image processing we will reference them as derivatives.

Straight line ranges are selected according to the following statements:

$$\{St(i)\} = \{\forall \underset{c\in[st1(i),st2(i)]}{|dDtf(c)| \le k1}, (st2(i) - st1(i)) \ge k2,$$
$$Dtf(st2(i)) < 3(re_{\max} - re_{\min})/4\}, i = 1..Nt,$$
$$\{Sb(j)\} = \{\forall \underset{c\in[sb1(j),sb2(j)]}{|dDbf(c)| \le k1}, (sb2(j) - sb1(j)) \ge k2,$$
$$Dbf(sb2(j)) < 3(re_{\max} - re_{\min})/4\}, j = 1..Nb,$$

where *{st1}* and *{sb1}* are coordinates of beginning of straight line ranges on distance arrays from top and bottom of bounding box correspondingly, *{st2}* and *{sb2}* are coordinates of ending of straight line ranges on distance arrays from top and bottom of bounding box, correspondingly, *Nt* and *Nb* are numbers of straight ranges on distance arrays from top and bottom of bounding box, correspondingly; constants *k1* and k2 depend on scanning resolution, for 300 dpi *k1* = 10, *k2* = 50.

Fig. 5 shows plots of distance from top array and its derivative as well as two ranges of straight lines.

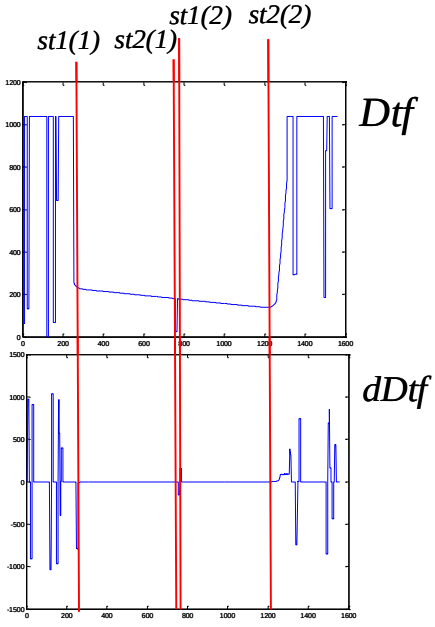


Figure 5: Distance and derivative of distance arrays.

3.4 Iterative estimation of skew-angle

For each straight line range we compute adjacent cathetus *dx*, opposite cathetus *dy* and angle α:

$$dx(i) = st2(i) - st1(i) - 2k1,$$

$$dx(j+Nt) = sb2(j) - sb1(j) - 2k1,$$

$$dy(i) = \frac{1}{2z+1}\sum_{k=-z}^{z} Dtf(st2(i) - k1 + k) -$$
$$-\frac{1}{2z+1}\sum_{k=-z}^{z} Dtf(st1(i) + k1 + k),$$

$$dy(j+Nt) = \frac{1}{2z+1}\sum_{k=-z}^{z} Dbf(sb2(i) - k1 + k)$$
$$-\frac{1}{2z+1}\sum_{k=-z}^{z} Dbf(sb1(i) + k1 + k),$$

$$\alpha(n) = arctg(dy(n)/dx(n)),\ N = Nb + Nt,\ n = 1..N,$$

where *N* is total number of straight line ranges for both arrays of distances from top and bottom of bounding box; *z* is number of points for averaging, *z* depends on scanning resolution, for 300 dpi *z* = 2.

We propose rule for reliable skew-angle φ estimation: φ is weighted average of angles which differ from φ by variance less than 1. For appropriate angles selection iterative procedure is used. Initial estimation is calculated as:

$$\varphi = \sum_{n=1}^{N}\alpha(n)\times dx(n) / \sum_{n=1}^{N} dx(n),\ \delta = \frac{1}{N-1}\sqrt{\sum_{n=1}^{N}(\varphi - \alpha(n))^2}.$$

If $\delta > 25$ then special processing for positive and negative angles which are close to 45° and -45° is performed:

$$bp(x) = \begin{cases} 1, x > 38^\circ \\ 0 \end{cases},\ bn(x) = \begin{cases} 0, x < -38^\circ \\ 0 \end{cases},$$

$$qap = \sum_{n=1}^{N} bp(\alpha(n)),\ qan = \sum_{n=1}^{N} bn(\alpha(n)),$$

$$sap = \sum_{n=1}^{N}\alpha(n)\times bp(\alpha(n)),\ san = \sum_{n=1}^{N}\alpha(n)\times bn(\alpha(n)),$$

if *qap* > *0* and *qan* > *0*, then δ=*0*, if *sap* > *san*, then $\varphi = sap/qap$, otherwise $\varphi = san/qan$, where *bp* and *bn* are functions for indication of big positive and negative angles, correspondingly; *qap* and *qan* are numbers of big positive and negative angles; *sap* and *san* are sum of big positive and negative angles.

Iterations are continued while variance $\delta > 1$ and array $\alpha(n)$ is not empty: if $(\alpha(n) < \varphi - \delta)$ or $(\alpha(n) > \varphi + \delta)$ then $\alpha(n)$ and *dx(n)* are excluded from corresponding arrays; further we make new estimation of weighted average of angles φ and variance δ as above. In case when angles significantly differ from each other, all angles are excluded from $\alpha(n)$ on some iteration. It corresponds to a situation, when we cannot make decision about skew-angle.

4. ROTATION IN PLACE

Image has to be rotated by minus skew-angle. Usually for rotation by angle φ the following transformation matrix is used:

$$[x' \quad y'] = [x \quad y]\begin{bmatrix} \cos\varphi & -\sin\varphi \\ \sin\varphi & \cos\varphi \end{bmatrix}.$$

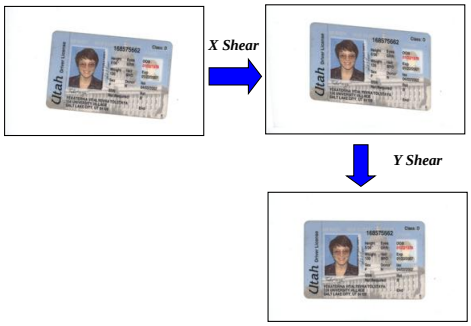


Figure 6: Example of rotation by two shears.

However in this case additional memory buffer for at least a part of rotated image is necessary. It can be impractical for copying devices where memory size is limited and/or operations with memory are relatively slow. It is preferable to use the same memory buffer where initial image is stored. Such algorithms are named 'in place' or *in situ*. Several papers describe decomposition of transformation matrix on two or three shears. Practical approach for rotation via two shears for rows and columns separately is proposed in [2] (see fig. 6 for example). The approach applies the following decomposition:

$$[x' \quad y'] = [x \quad y] \begin{bmatrix} \cos\varphi & 0 \\ \sin\varphi & 1 \end{bmatrix} \begin{bmatrix} 1 & -\tan\varphi \\ 0 & \sec\varphi \end{bmatrix}.$$

Various interpolation algorithms can be applied during rotation. Bilinear interpolation is good trade-off between image quality and processing speed. Application of this technique allowed us to meet strong limitations of embedding systems.

5. RESULTS AND CONCLUSION

Proposed algorithm is extremely fast. Processing time on ARMv7 800 MHz for 1920x1080 RGB 24 bpp image is about 0.3 s without application of SIMD instructions. For 300 dpi scanning resolution our method requires about 150 Kb of additional memory only. So, it is applicable not only in software but also in firmware of embedded systems.

For conventional A4 and Letter-size document images paper [4] claims that a skew by as a little as 0.1° is noticeable for observer. Size of card images is smaller; and according to our investigation skew-angle of 0.1° is almost unnoticeable for naked eye. Survey among 14 observers has shown that skew-angle in 0.4° of card images is critical for visual perception.

For evaluation of deskew quality we randomly selected 10 cards from our test set and processed these cards by scanning software of the scanners HP ScanJet G4010, Epson Perfection V300 Photo and Canon CanoScan 8800F. Also we checked deskew function in well-known OCR applications ReadIRIS 11 and FineReader 10. We calculated percentage of deskewed images, average angle after deskew and percentage of cases, when angle after deskew is less than 0.4°. Table 1 contains results of our investigation. Our solution shares the first place with HP software and ReadIRIS.

We tested proposed algorithm for various types of document images different from card images. Proposed technique works well for all types of documents that we tested. Moreover, we applied this deskew technique for deskewing several rectangular objects placed on scanner platen and obtained good results. At the same time proposed algorithm is simple enough. Its implementation on C programming language contains about 400 lines only. So, it is an effective approach for plenty of scanning/copying applications and several other image correction tasks.

TABLE 1 DESKEW QUALITY

	Averaged angle after deskew, °	*Deskewed images, %*	*Angle after deskew is less than 0.4°, %*
HP G4010	**0.2**	**100**	**90**
Epson V300	3.5	40	30
Canon 8800	1.3	90	50
ReadIRIS	**0.2**	**100**	**90**
FineReader	2.2	60	60
Proposed	**0.2**	**100**	**90**

6. REFERENCES

[1] Beusekom, J., Shafait, F. & Breuel, T., *Resolution Independent Skew and Orientation Detection for Document Images*, Proc.of SPIE: Document Recognition and Retrieval XVI, 2009.

[2] Catmull, E., Smith, A., *3-D transformations of images in scanline order*, Proc. of SIGGRAPH'80, pp. 279–285, 1980.

[3] Chou, C.H., Chu, S.Y., Chang, F., Estimation of skewangles for scanned documents based on piecewise covering by parallelograms, Pattern Recognition 40, pp. 443 – 455, 2007.

[4] Hull, J., Document image skew detection: survey and annotated bibliography, Document Analysis Systems II, pp. 40-64, 1998.

[5] Konya, I., Eickeler, S., Seibert, C., *Fast Seamless Skew and Orientation Detection in Document Images*, Proc. of ICPR, pp. 1924-1928, 2010.

[6] Meng, G., Pan, C., Zheng, N., Sun, C,. *Skew Estimation of Document Images Using Bagging*, IEEE Transactions on image processing, vol. 19, No. 7, 2010.

[7] Vil'kin, A.M., Safonov, I.V., Egorova, M.A., *Bottom-up page segmentation based on texture features*, Proc. of PRIA, vol. 2, pp. 373-376, 2010.

[8] Yuan, B., Tan, C.L, *Fiducial line based skew estimation*, Pattern Recognition 38, pp. 2333 – 2350, 2005

About the author

Ilia V. Safonov received his MS degree in automatic and electronic engineering from Moscow Engineering Physics Institute/University (MEPhI), Russia in 1994 and his PhD degree in computer science from MEPhI in 1997. Since 2004, Dr. I. Safonov is with Samsung Research Center in Moscow, where he is engaged in image and video processing projects.

Ilya V. Kurilin received his MS degree in radio engineering from Novosibirsk State Technical University (NSTU), Russia in 1999 and his PhD degree in computer science from NSTU in 2006. At present time Dr. I. Kurilin is in Samsung Research Center, where he is engaged in photo and documents image enhancement projects.

Ray Maps technique for effective interrogation of results of MCRT simulation

B.Kh. Barladyan, L.Z. Shapiro, A.G. Voloboy
Keldysh Institute of Applied Mathematics RAS, Moscow

Abstract

The new lighting simulation result interrogation technique called Ray Maps is described. These maps can be effectively used in physically accurate optic simulation system based on forward and backward Monte-Carlo ray tracing. In particular the Ray Maps can increase efficiency for photo-realistic images creation, fast analyze of light characteristics on the radiation detector with variable parameters, studying the fine details of the light propagation in the complex scenes and so on. The example of Ray Maps usage is Ray Visualization intended for the complex optical system design. This example is considered in details in the paper.

Keywords: *Monte Carlo ray tracing, Lighting simulation, Global illumination, Photo-realistic images, Interactive scene analysis, Ray Maps*

1. INTRODUCTION

Monte-Carlo ray tracing, both forward and backward ones, is widely used in various systems for physically accurate simulation of light propagation in complex scenes [Pharr 2004]. These methods are used for simulation of global illumination, photo realistic images generation and simulation of complex optical devices. Optical simulation of light transportation using Monte Carlo ray tracing usually requires considerable time. Long calculation time is especially critical if Monte Carlo ray tracing is used in rendering like described in [Hashisuka 2009]. In such a case the additional techniques should be elaborated to improve the effectiveness of the user interaction with such system.

The main idea of our approach is to store the set of rays traced by the Monte Carlo method together with the necessary supplementary data. After that the other subsystems of the optical simulation can be used for quick interrogation of calculated results as well as the ways of light propagation in a scene. This stored data can be called Ray Maps. It should be pointed that the set of the supplementary data to be stored together with the array of traced rays depends on the goal for which these data will be used. The volume of the stored data can be very large and the efficiency of the following processing strongly depends on this volume. The storing format of this data also is critical in the terms of further processing efficiency and therefore should be elaborated taking further processing into account. The present paper considers in detail only Ray Maps creation during forward Monte Carlo ray tracing and their post processing for the purpose of ray visualization system. Although the proposed approach can be used for other goals as well. Certainly the data set and the storage format depends on the certain purpose. More precisely it is determined by the post processing effectiveness. In the case when the stored data are used frequently (for example, the realistic images creation for several camera positions is the typical task) the access to the data should be accelerated by using special accelerated structures like, for example, BSP tree [Havran 2000].

2. DATA FOR RAY VISUALIZATION SYSTEM

The ray set in our Ray Visualization system [Voloboy at al 2009] is stored as segments set and contains the following data:

1. Segment color. The Monte-Carlo simulation in our system can be done in two color spaces – RGB and spectral ones. Even if the spectral color mode is used during simulation the user can request to store Ray Maps in the RGB format. This allows decreasing the stored data size. In the same time for ray visualization purpose the RGB format is sufficient. The spectral mode of segment representation which is also possible may be needed for more detailed investigation.
2. The light source emitted the given ray.
3. The geometry object from which the ray segment starts (if exists).
4. Triangle index for the geometry object from which the ray segment starts.
5. 3D point coordinates for the segments start and end points.
6. Normal to the surface in the ray/surface intersection points.
7. Optical surface properties for intersected surfaces.
8. Ray transformation type on the given surface – diffuse refraction/reflection, specular refraction/reflection, Fresnel refraction/reflection, BRDF/BTDF diffuse and specular refraction/reflection, absorption on the surface.
9. Transformation type in the medium – absorption or ray scattering.
10. Optical properties of the medium along the ray propagation.
11. Virtual radiation detectors intersected by the given ray segment and parameters of the given intersection.

This data is used for the two purposes:

1. For checking ray visualization criterion – only rays satisfied to the criterion defined by user should be visualized.
2. For Ray Tracing Report creation. User can investigate the history of the ray path in details in the report where information about all visualized rays satisfied to the given criterion is collected. In particularly the user can be interested in light source which emit the given ray, in transformation which occurred on the surface or in the media, in ray segment color (RGB and spectral), in surface/medium optical attributes and so on.

3. DATA STRUCTURES AND STORING

The number of ray segments created during Monte Carlo ray tracing can reach millions or even billions in some cases. So the Ray Maps created by Monte Carlo ray tracing have to be stored on the hard disk during simulation and then loaded for analyzing as needed. It should be pointed that while the most part of the segment data of the traced ray has fixed size, the data described the intersections with the radiation detector has variable length – it is possible that the given ray segment does not intersect the radiation detectors at all or does intersect several such detectors. The data describing the ray segment color has variable length also because the spectral color description as well as RGB color description is possible according user request. Moreover in common case the user can switch color mode storing in any moment during simulation. In the result one part of ray segments will contain the spectral data while another one – RGB data. Due to variable length of additional data each portion of traced rays is stored in the three parts. The first part contains basic ray segment descriptions of fixed length and links to variable additional data. The structure of basic description contains the following data:

1. Segment color in RGB format. The RGB color is needed to visualize the ray trajectory on display.
2. Index of light source emitted the ray.
3. The index of object from which the ray segment starts.
4. The index of the triangle for an object from which the ray segment starts.
5. 3D coordinates of the beginning of the ray segment.
6. Normal to the surface in the ray segment beginning point, if it corresponds to ray/surface intersection.
7. Ray transformation type on the surface or in the medium which cause the given ray segment (in particular, it can be absorbed on the surface, in the medium or the ray can left the scene).
8. Index of the first descriptor of the ray intersections with the radiation detectors.
9. The number of detectors intersected by the ray segment.

The second portion part contains a description of ray (segment) intersections with radiation detectors. It is an array of structures of fixed length. This structure contains the following data:

1. Index of radiation detector
2. Radiation detector cell index
3. The angle between the ray and the normal to the plane of the radiation detector.
4. The 2D coordinate of intersection of ray segment with radiation detector plane (local coordinates of the intersection point in the rectangle of the radiation detector coordinate system)
5. Was the intersection registered by the radiation detector or not.

This structure is used for each ray intersection with the each radiation detector. The array of these structures together with fields 8 and 9 in the previous structure provides information about ray/detector intersections in relatively compact form.

The last third part of the recorded data for each portion contains a description of a spectral color of ray segments in case of the spectral simulation was performed and storing of spectral data have been requested by the user.

The described above data provides storing the results of Monte Carlo ray tracing in compact form. In the same time the data contains the complete information necessary for the post processing. During ray visualization and analysis user can set different criterion for the ray visualization. The criterion consists of the following set of elementary events:

1. Ray was emitted by the specified light source.
2. Ray intersected the specified face of the optical device.
3. The ray intersected the pointed part of a geometric object of optical system, and the specified optical ray transformation took place.
4. The ray intersected the specified radiation detector or some its area.
5. The ray intersected any surface which have the specified set of optical properties, and the specified optical ray transformation took place.
6. The ray undertook the specified optical ray transformation (diffuse or specular reflection, absorption, etc.) on the one of the objects of the optical system.

User can apply for these events, their logical negation, intersection ($\cap$) and union ($\cup$) with an unlimited number of operands. In the result only the rays that satisfy to final logical expression will be visualized.

The user can also interactively change the conditions of registration of radiation at the radiation detector, for example, the resolution of a radiation detector, the angle of registration, show results only for the given light source. Also it is possible to obtain the different statistical data for rectangular and elliptical region of the radiation detector.

The changing of the ray visualization criterion or parameters of the radiation detector does not require new simulation, because the all necessary data are already presented in the saved data (Ray Maps) and can be processed relatively quickly to obtain the requested information.

There is an important aspect concerned to the effective simulation result saving in the form of Ray Maps – the size of file with stored data. For the 1 million rays (in average ~5 million segments) it requires ~220Mb if simulation was done in RGB space, and ~540Mb in spectral simulation mode with 33 wavelengths. Here the main problem is not in the required disk space, but in the time (speed) of the recording itself. It should be also taken into account that the Monte Carlo ray tracing typically uses multi-threading and distributed computing on multiple computers (the number of threads, used in Monte Carlo ray tracing is the total number of processors on all computers involved in simulation) while recording (and reading) typically uses the single thread. So in some cases storing the data to the disk becomes a bottleneck that limits the speed of simulation.

To overcome this problem in our system the data compression is used before writing it to disk. Directly Monte Carlo ray tracing is performed in our system by portions of a number of rays. The portion size is tuned during simulation depending on the different parameters such as the required accuracy, the ordered number of rays, number of available processors etc. Each portion is calculated in a separate thread (in general case on different

computers) almost independently. In this situation we applied the compression for each portion of the traced rays in calculating threads. Already compressed data is transferred to the main thread for storing to the disk. As a result the storing the data to disk became shorter and does not de-accelerate lighting simulation, at least for the typical case of four calculations threads in one computer. After compressing the file size in the case of simulation in RGB space was decreased from 220Mb to 50Mb and in spectral case from 540Mb to 51Mb. The total simulation time practically was not affected in RGB case and accelerated in spectral case. The time spend on the compression performed in parallel in multiple threads were compensated by the acceleration of the data writing on the disk performed by the single thread. To compress the data the free cross-platform data compression library, created by Jean-Loup Gailly and Mark Adler, zlib [Zlib 2010] was used.

4. RESULTS

Ray Maps creation and further its processing in Ray Visualization module was implemented in our lighting simulation system [Voloboy, Galaktionov 2006].

During testing on many scenes it was confirmed that the time spent for the Ray Maps writing to the disk performed in a separate thread is almost negligible for the forward Monte Carlo rays tracing executed in parallel threads. For example if four threads (Intel Core2 Quad Q9550 2.83GHz) are used for Monte Carlo ray tracing and one more thread for data storing in uncompressed state then we often get situation when four calculation threads have to wait until storing thread has completed writing of previous data portion. As the compression reduces the size of recorded data in ~4.5 times for RGB mode and in ~10 times in spectral mode the data storing thread provides writing all necessary data to disk without suspending the work of simulation threads. The situation is explained by the fact that the processor spent small fraction of time for writing data to a disk.

It should be noted that the computational threads spend some time on compression. For very simple scenes and simulation in RGB mode compression time can be up to 50% of simulation time. But time spent for compression is determined mainly by number of segments and wavelengths for a ray portion while simulation time is determined by scene complexity. For example for simulation in spectral mode only 25% of CPU time is spent for compression. For complex real scenes this per cent decreases more. These costs are justified if four processors are used which is typical for modern computers. According to our estimates a recording thread will provide effective work till 6 - 8 simulation threads especially when the spectral simulation will be done with a large number of wavelengths.

The following results were obtained for typical scenes for simulation in RGB space (on 1000000 rays, ~5000000 segments):

1. Monte Carlo tracing time is and ~5-10 seconds in case of simulation of typical scene with surfaces described by complex optical attributes (BRDF).
2. Post processing. Loading time is 1.6 sec without compression. Compression (decompression is needed during loading) increases the whole loading time to ~2.3-2.4 seconds.
3. Post processing. Criterion analyzing time is ~1.2-1.3 seconds if a single elementary event is used. Each additional elementary event used in criterion increases processing time on ~0.5-0.6 seconds. This time is required if all 1000000 rays will be processed. Typically only small part of rays should be processed to obtain requested number of output rays satisfying to the specified criterion.

Therefore the proposed Ray Maps technique is an effective tool for interrogation of simulation results calculated by Monte Carlo ray tracing. This is important for design of complex optical devices. The biggest benefit of the proposed approach takes place in the spectral simulation and if optical properties of surfaces are described by the complex bidirectional reflectance distribution functions (BRDF) of large size. Another case when advantage of our approach is significant is volume scattering in a complex multi component media. In all cases when the time of simulation by the Monte Carlo ray tracing is large and the results should be used multiple times the usage of Ray Maps increases efficiency of the simulation system.

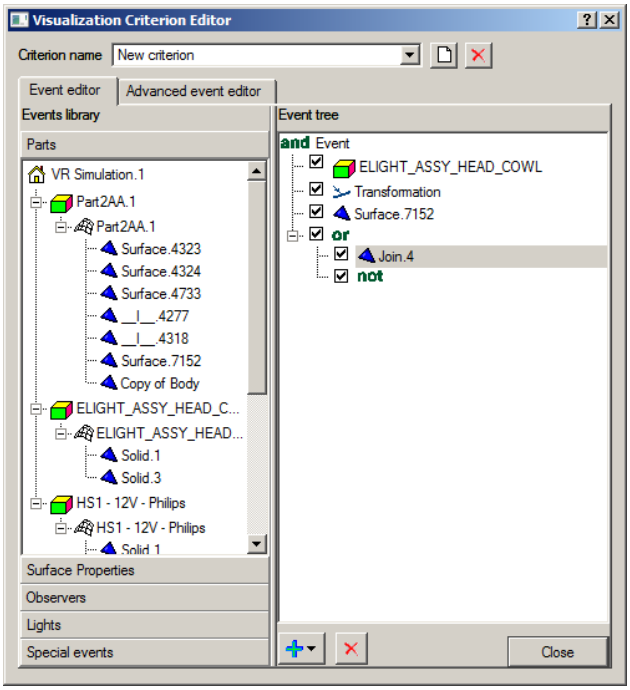


Figure 1: Visualization Criterion Editor.

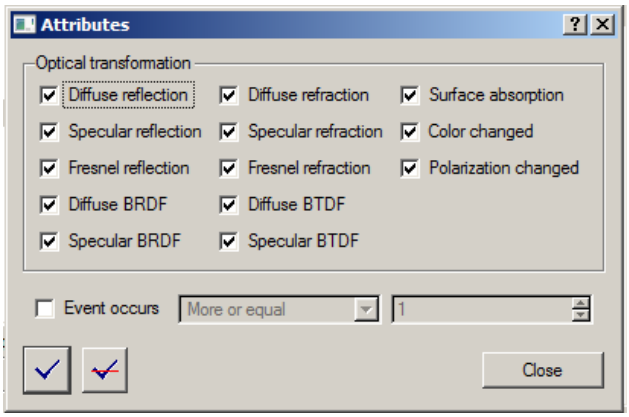


Figure 2: Optical Transformation Attributes.

The **Figure 1** and **Figure 2** illustrates the user possibilities to control the set of visualized rays by using the **Visualization Criterion Editor.** The user can set different criteria for ray visualization. The following set of elementary events can be selected in the dialog on **Figure 1:**

- light source emitted ray,
- intersection with specified geometry part,
- intersection with the surface with specified optical properties,
- special events along the ray path (killed ray or transformation).

For events it is possible to select special type of transformations by using the dialog presented on the **Figure 2**.

Also it is possible to construct logical expression from elementary event as it is shown in right part of dialog on **Figure 1**.

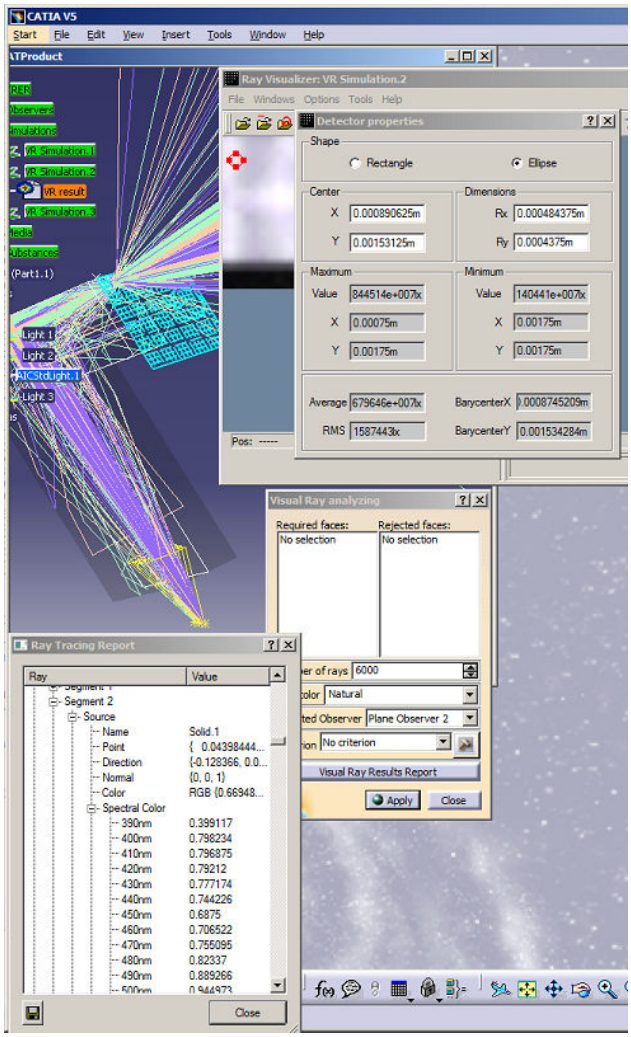


Figure 3: Our ray visualization system in CATIA.

The **Figure 3** illustrates the elaborated by us plugin for industrial CAD system CATIA [CATIA]. The dialog of radiation detector in the top-right corner lets you to choose the rays passing via the specified detector area additionally to the ray criterion specified as it is shown on the **Figure 1** and **Figure 2**. The **Ray Tracing Report** located at the lower left corner of the image provides detailed analyzing of the events, the optical properties of surfaces and media along the ray trajectory. All rays in this report correspond to the displayed rays and satisfy to the criterion set by user. The ray segment color can be represented both in RGB and spectral formats.

5. FUTURE WORK

Currently the suggested approach is used in the ray visualization subsystem. We suppose to adapt the Ray Maps technique for photo realistic images generation for scenes with complex optical surface and media properties. We also plane to use Ray Maps to allow post processing variation of radiation detector parameters such as detector resolution, the angles of registration, etc.

6. AKNOLEDGMENTS

This work was supported by the Russian Leading Scientific Schools Foundation, project NSh- 8129.2010.9, RFBR, grants 10-01-00302 and 11-01-00870, and by INTEGRA Inc. (Japan).

7. REFERENCES

[CATIA] CATIA - Virtual Design for Product Excellence, http://www.3ds.com/products/catia/welcome/

[Hashisuka 2009] Toshiya Hachisuka and Henrik Wann Jensen. Stochastic progressive photon mapping. ACM Transaction on Graphics, 28(5), pp. 1–8, 2009.

[Havran 2000] Vlastimil Havran. Heuristic Ray Shooting Algorithms: Ph.D. thesis / Department of Computer Science and Engineering, Faculty of Electrical Engineering, Czech Technical University in Prague. November 2000 http://www.cgg.cvut.cz/~havran/phdthesis.html.

[Pharr 2004] Matt Pharr, Greg Humphreys, Physically Based Rendering. Elsevier 2004.

[Voloboy, Galaktionov 2006] А.Г. Волобой, В.А. Галактионов Машинная графика в задачах автоматизированного проектирования // "Информационные технологии в проектировании и производстве", № 1, 2006, с. 64-73

[Voloboy et al 2009] А.Г. Волобой, В.А. Галактионов, А.Д. Жданов, Д.Д. Жданов. Средства визуализации распространения световых лучей в задачах проектирования оптических систем // "Информационные технологии и вычислительные системы", № 4, 2009, с. 28-39.

[Zlib 2010] http://zlib.net/

About the authors

Boris Kh. Barladyan, PhD, senior researcher, Keldysh Institute for Applied Mathematics RAS.

E-mail: obb@gin.keldysh.ru

Lev Z. Shapiro, PhD, senior researcher, Keldysh Institute for Applied Mathematics RAS.

Alexey G. Voloboy, PhD, senior researcher, Keldysh Institute for Applied Mathematics RAS.

E-mail: voloboy@gin.keldysh.ru

Usage of Local Estimations and Object Spectral Representation at the Solution of Global Illumination Equation

Vladimir P. Budak, Victor S. Zheltov, Timofey K. Kalakutsky
Light Engineering Department
Moscow Power Engineering Institute (TU), Moscow, Russia
budakvp@mpei.ru, zheltov@list.ru, kalakutskytk@mpei.ru

Abstract

The solution simulation of the global illumination equation by the local estimation of the Monte Carlo method is analyzed in the paper. Local estimation methods allow implementing simulation at the arbitrary reflectance law and directly calculating luminance in every point and direction of 3D scene. The local estimations are more effective methods than straight modeling. Their usage does not demand the mesh creation for the illumination map formation. The possibility of 3D object representation on the basis of the spherical harmonics is also considered in the paper. The spectral representation possesses the essential advantages for some object classes. The algorithms of the local and double local estimation with the usage of object representation on the basis of the spherical harmonics are implemented in the framework of paper. This algorithm allow realizing the physically adequate modeling of the global illumination equation solution and calculating the luminance directly that opens new horizons both in the photometry and computer graphics.

Keywords: *Global Illumination, Monte Carlo, Local estimation.*

1. INTRODUCTION

3D scene simulation is inherently the calculation of object luminance on the basis of the solution of the global illumination equation, which is an integral equation of the second order [1]:

$$L(\mathbf{r},\hat{\mathbf{l}}) = L_0(\mathbf{r},\hat{\mathbf{l}}) + \frac{1}{\pi}\int L(\mathbf{r},\hat{\mathbf{l}}')\sigma(\mathbf{r};\hat{\mathbf{l}},\hat{\mathbf{l}}')\left|(\hat{\mathbf{N}},\hat{\mathbf{l}}')\right| d\hat{\mathbf{l}}' \qquad (1)$$

where $L(\mathbf{r},\hat{\mathbf{l}})$ is the luminance at the point $\mathbf{r}$ in the direction $\hat{\mathbf{l}}$, $\sigma(\mathbf{r};\hat{\mathbf{l}},\hat{\mathbf{l}}')$ is bidirectional scattering distribution function (reflection or transmittance), L_0 is the luminance of the direct radiation straight from sources.

The equation (1) has no analytical solution at an arbitrary law of reflection, so the methods of numerical simulation are used. The Monte Carlo methods, direct and reverse ray tracing are the most popular among them.

The reverse ray tracing suffers from basic defects connected with that the sizes of light sources as a rule are sufficiently small and the expectancy of hitting in the source is very low.

In case of the direct simulation the scene is divided into elements, in which the photon hits are calculated. As a result the illumination map is created. This approach is connected with the complications of mesh generation and the excessive consumption of memory.

The finite element method, which is called in the global illumination theory as a radiosity [3], is used for the solution of equation (1) on the basis of diffuse reflection approximation. In this case the equation (1) could be rewritten as [3]

$$M(\mathbf{r}) = M_0(\mathbf{r}) + \frac{\sigma}{\pi}\int_{\Sigma} M(\mathbf{r}')F(\mathbf{r},\mathbf{r}')\Theta(\mathbf{r},\mathbf{r}')d^2\mathbf{r}', \qquad (2)$$

where $M(\mathbf{r})$ is the luminosity of the surface in point $\mathbf{r}$, $M_0(\mathbf{r})$ is the luminosity of the surface in point $\mathbf{r}$ straightforward from sources, $\Theta(\mathbf{r},\mathbf{r}')$ is the visibility function of element $d^2\mathbf{r}'$ from point $\mathbf{r}$, $F = \dfrac{\left|(\hat{\mathbf{N}}(\mathbf{r}),(\mathbf{r}-\mathbf{r}'))\right|\left|(\hat{\mathbf{N}}(\mathbf{r}'),(\mathbf{r}-\mathbf{r}'))\right|}{(\mathbf{r}-\mathbf{r}')^4}$ is the infinitesimal form-factor.

In our paper we propose to use the local estimation of the Monte-Carlo method, which are well-known in the optics of atmosphere at the solution of radiative transfer equation [2, 4]. Rendering 3D objects represented by the spherical harmonics spectrum is analyzed in the paper.

2. LOCAL ESTIMATION

Equation (1) is not convenient for the statistic simulation, as the unknown function is under the integral in the point $\mathbf{r}'$, but it is determined in the point $\mathbf{r}$. In addition to that the variables $\mathbf{r}'$ and $\hat{\mathbf{l}}'$ are not independent, and combined by the relation

$$\hat{\mathbf{l}}' = \frac{\mathbf{r}-\mathbf{r}'}{|\mathbf{r}-\mathbf{r}'|}. \qquad (3)$$

Accordingly we can write down the equation (1) in the following form

$$L(\mathbf{r},\hat{\mathbf{l}}) = L_0(\mathbf{r},\hat{\mathbf{l}}) + \frac{1}{\pi}\int L(\mathbf{r}',\hat{\mathbf{l}}')\sigma(\mathbf{r};\hat{\mathbf{l}},\hat{\mathbf{l}}')\delta\left(\hat{\mathbf{l}}' - \frac{\mathbf{r}-\mathbf{r}'}{|\mathbf{r}-\mathbf{r}'|}\right)\frac{\left|(\hat{\mathbf{N}},\hat{\mathbf{l}})\right|\left|(\hat{\mathbf{N}}',\hat{\mathbf{l}}')\right|}{(\mathbf{r}-\mathbf{r}')^2}d^2\mathbf{r}'. \qquad (4)$$

This expression contains the δ-function impeded the simulation by the local estimation of the Monte Carlo method. It is possible to eliminate the singularity in the expression (4) integrating over space. In addition to that the equation (2) contains no singularities and fit to simulation by the local estimation. Consequently the estimation for the arbitrary functional I_φ of $L(\mathbf{r},\hat{\mathbf{l}})$ takes a form

$$I_\varphi = \mathtt{M}\sum_{n=0}^{\infty} Q_n k(\mathbf{r},\mathbf{r}'), \qquad (5)$$

where $k(\mathbf{r},\mathbf{r}')$ is the kernel of (4), Q_n is the weight of Markov chain, and $\mathtt{M}$ is the mean operator by the different random ray trajectories.

Markov chain in this case represents the sample of random ray trajectory, which is created using $L_0(\mathbf{r},\hat{\mathbf{l}})$ as an initial probability and $k(\mathbf{r},\mathbf{r}')$ as a transition probability [2, 4]. From every knots of this random ray trajectory $\mathbf{r}$□, which coincide with the intersection points of ray with object surfaces, is calculated the contribution in the illumination in the given point $\mathbf{r}$ by the term $k(\mathbf{r},\mathbf{r}')$ in expression (5).

Of course all the probability distribution must satisfy to the normalizing condition. The distinctions of $L_0(\mathbf{r},\hat{\mathbf{l}})$ and $k(\mathbf{r},\mathbf{r}')$ from this condition generate the statistical weights Q_n. For example, in case of the diffuse reflection it is equal to the surface reflectivity.

The expression (5) was called the local estimation of the Monte Carlo method [2, 4]. It allows estimating the illumination in the given point. Thereby to calculate the illumination in some given

point **r** it is necessary to create the Markov chain in the space, and at every point to calculate the value $k(\mathbf{r},\mathbf{r}')$ for investigated points. The mathematical expectation of this value is equal an illumination.

Let's note that as opposed to the classical ray tracing the local estimation allows estimating in several points at once by one statistical sampling ray that sufficiently accelerates the convergence of calculations.

3. ACCURACY EVALUATION OF LOCAL ESTIMATION

The accuracy estimation of algorithm is possible on the basis of comparison with other known methods or with exact analytical solution of the equation (2) for some special situation. It is known two exact analytical solution of the global illumination equation: for the Ulbricht sphere and for so called Sobolev problem [9]. In case of the Ulbricht sphere we have the uniform distribution of the luminosity over sphere that makes this case ineffective for the accuracy estimation. In the Sobolev problem we solve the (2) for the luminosity distribution in case of the illumination of two parallel diffuse reflecting planes by the point isotropic source located between these planes. In this case the equation (2) is transformed to the set of two integral equations (for the first plane):

$$E_1(\mathbf{r}) = \frac{\rho_2}{\pi}\int \frac{E_2(\mathbf{r}')d^2\mathbf{r}'}{\left[1+(\mathbf{r}-\mathbf{r}')^2\right]^2} + \frac{h_1}{(h_1^2+\mathbf{r}^2)^{3/2}}, \qquad (6)$$

where $E_i(\mathbf{r})$ is the illuminance of i-th plane (i=1,2), ρ_i is the reflection coefficient of i-th plane, h_i is the distance from the source to the i-th plane. We assume the source intensity equal to 1 and $h_1+h_2=1$.

For the solution of the equation set (6) let's execute the Fourier transformation of each equation that results in the set of linear algebraic equations. After the solution of the obtained equation set and inverse Fourier transformation we get the following expression:

$$E_1(r) = \frac{h_1}{(h_1^2+r^2)^{3/2}} + \rho_2\int_0^\infty \frac{e^{-h_1 k}\rho_1 k K_1(k) + e^{-h_2 k}}{1-\rho_1\rho_2 k^2 K_1^2(k)} K_1(k) J_0(kr) k^2 dk, \qquad (7)$$

where $K_1(k)$ is the modified Bessel function of second kind, $J_0(kr)$ is the Bessel function of first kind.

In Figure 1 we can see the numerical comparison of the luminosity distribution calculation by the local estimation method and the expression (7).

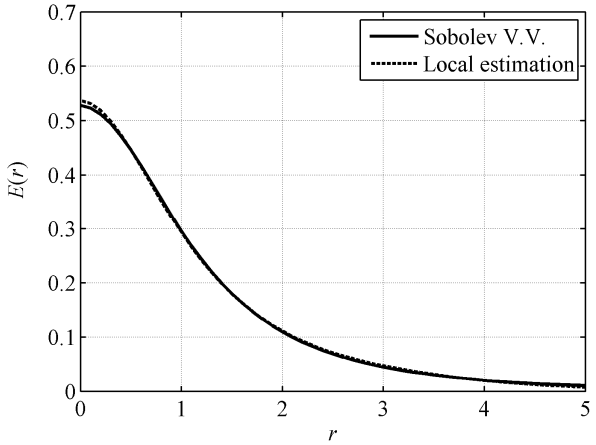


Figure 1. Illuminance distribution in case of the Sobolev problem: $h_1 = h_2 = 0.5$, and reflection coefficients $\rho_1 = \rho_2 = 0.5$.

In this case it was used 2000 rays at the calculation by the local estimation method, and the running time was less than 1 second for the computer AMD Athlon 64 X2 5200.

We realized also the solution of the Sobolev problem by the radiosity, and compared it with the local estimation: the local estimation exceeds the radiosity in computation speed at 80-90 times for the equal calculation accuracy.

4. DOUBLE LOCAL ESTIMATION

The equation of global illumination can be represented in the operator form

$$L = L_0 + \mathbf{K}L. \qquad (8)$$

The solution of this equation is represented in the form of the Neumann series that allows conducting following transformations

$$L = \sum_{n=0}^{\infty}\mathbf{K}^n L_0 = L_0 + \mathbf{K}L_0 + \sum_{n=2}^{\infty}\mathbf{K}^n L_0 = L_0 + \mathbf{K}L_0 + \mathbf{K}^2\sum_{n=0}^{\infty}\mathbf{K}^n L_0 = L_0 + \mathbf{K}L_0 + \mathbf{K}^2 L, \qquad (9)$$

that results in normal a form

$$L(\mathbf{r},\hat{\mathbf{l}}) = L_0(\mathbf{r},\hat{\mathbf{l}}) + \mathbf{K}L_0 + \frac{1}{\pi^2}\int_{(\Sigma)}\sigma(\mathbf{r};\hat{\mathbf{l}}',\hat{\mathbf{l}})\delta\left(\hat{\mathbf{l}}' - \frac{\mathbf{r}-\mathbf{r}'}{|\mathbf{r}-\mathbf{r}'|}\right)F(\mathbf{r},\mathbf{r}')\times$$
$$\times\int_{(\Sigma)} L(\mathbf{r}'',\hat{\mathbf{l}}'')\sigma(\mathbf{r}';\hat{\mathbf{l}}'',\hat{\mathbf{l}}')\delta\left(\hat{\mathbf{l}}'' - \frac{\mathbf{r}'-\mathbf{r}''}{|\mathbf{r}'-\mathbf{r}''|}\right)F(\mathbf{r}',\mathbf{r}'')d^3\mathbf{r}''d^3\mathbf{r}'. \qquad (10)$$

The local estimation corresponding expression (10) can be called as double local estimation [2, 4] and takes the following view

$$I_\varphi = \mathsf{M}\sum_{n=0}^{\infty} Q_n k(\mathbf{r},\hat{\mathbf{l}},\mathbf{r}',\mathbf{l}'), \qquad (11)$$

where

$$k(\mathbf{r},\hat{\mathbf{l}},\mathbf{r}',\hat{\mathbf{l}}') = \frac{1}{\pi^2}\sigma(\mathbf{r}';\hat{\mathbf{l}}'',\hat{\mathbf{l}}')\sigma(\mathbf{r};\hat{\mathbf{l}}',\hat{\mathbf{l}})F(\mathbf{r}',\mathbf{r}''). \qquad (12)$$

In the expression (12) the δ-function is eliminated as a result of integration, and independent variables **r**, $\hat{\mathbf{l}}$, **r**′, $\hat{\mathbf{l}}'$, **r**″, $\hat{\mathbf{l}}''$ correspond to the geometry of ray propagation [4].

Thereby the double local estimation allows modeling the global illumination equation solution and calculating straight the luminance of incident radiation in the given point for the reflectance of orders higher than the first taking into account an arbitrary reflectance law. The first order can be calculated directly.

We do not know currently any software, which is capable to calculate the luminance directly. Thereby the double local estimation for the first time in light engineering and computer graphics allows computing the spatial-angular distribution of luminance in the given point of space.

5. SPECTRAL REPRESENTATION OF OBJECTS

Standard representation for the 3D objects is the mesh representation, in which objects are described by the set of vertices connected into faces. Such representation is universal and allows describing any object. However it is not capable to reproduce exactly many objects, and in case of the essential error influence on the result the solid modeling can be applied, when objects are described analytically. SolidWorks and TracePro are the most known programs using the similar approach.

One of the challenging directions of the 3D object representation is the spectral representation on the basis of spherical harmonics [5] that is the object surface representation by the series of spherical harmonics. In the sequel we will use the term "spectral representation" for short. Such object representation gives a possibility

of the reproduction quality control. So objects located far from a camera can be reproduced with low quality. In terms of photometry the essential advantage of this approach is the continuous representation of surface normals without any approximation. Let's analyze the mathematical techniques underlying the method of spectral representation.

The spherical harmonic $\left\{Y_k^m(\theta,\varphi):\ m \le k\right\}$ is the special function defined on the unit sphere

$$Y_k^m(\theta,\varphi) = (A_{km}\cos m\varphi + B_{km}\sin m\varphi)P_k^m(\cos\theta) \qquad (13)$$

where θ is the zenith angle [0 π], φ is the azimuth angle [0 2π], A_{km}, B_{km} are some coefficients, $P_k^m(\cos\theta)$ is the associated Legendre polynomials. The major property of spherical harmonics is their orthogonality:

$$\int_0^{2\pi}\int_0^{\pi} Y_k^m(\theta,\varphi)Y_{k'}^{m'}(\theta,\varphi)\sin\theta d\theta\varphi = \delta_{k,k'}\delta_{m,m'}\,, \qquad (14)$$

where δ is the delta Kronecker symbol.

Owing to this property any twice continuously differentiable function can be expanded into uniformly and absolutely convergent series of spherical harmonics [6]

$$f(\theta,\varphi) = \sum_{k=0}^{\infty}\sum_{m=0}^{k}(A_{km}\cos m\varphi + B_{km}\sin m\varphi)P_k^m(\cos\theta)\,, \qquad (15)$$

where A_{km}, B_{km} are the Fourier coefficients determined by formulae

$$A_{km} = \frac{\int_0^{2\pi}\int_0^{\pi} f(\theta,\varphi)P_k^m(\cos\theta)\cos m\varphi\sin\theta d\theta d\varphi}{\left\|Y_k^m\right\|^2},$$

$$B_{km} = \frac{\int_0^{2\pi}\int_0^{\pi} f(\theta,\varphi)P_k^m(\cos\theta)\sin m\varphi\sin\theta d\theta d\varphi}{\left\|Y_k^m\right\|^2}, \qquad (16)$$

and $\left\|Y_k^m\right\|^2$ is the norm determined by the expressions

$$\left\|Y_k^m\right\|^2 = \frac{2\pi\varepsilon_m}{2n+1}\frac{(k+m)!}{(k-m)!},\quad \varepsilon_m = \begin{cases}2, m=0\\ 1, m>0\end{cases} \qquad (17)$$

The calculation of factorial is the substantial complication by the expansion. As a result it is the essential difference of coefficients resulting in the extra miscalculation. At the expansions of spherical harmonics it is more convenient to use the Schmidt polynomial determined as

$$\mathrm{Q}_k^m(\mu) = \sqrt{\frac{(k-m)!}{(k+m)!}}P_k^m(\mu)\,. \qquad (18)$$

It is not difficult to see that at the Schmidt polynomial expansion there is no necessity to calculate the factorials for the norm and that essentially improved productivity of calculations.

Therefore, any object defined unambiguously on radius from some point can be expanded by spherical harmonics.

6. RENDERING SPECTRAL OBJECTS

At the global illumination equation solution by the local estimation method it is necessary to calculate surface normal and to determine the cross point of ray with an object. In case of the mesh object representation this problem is not difficult, but in case of the spectral representation it becomes a hard one. We analyze the determination of normal to the given point of object defined in the basis of spherical harmonics.

The normal in the point (r_0,θ_0,φ_0) on the surface defined in the spherical coordinate system is equal the value of its gradient in this point

$$gradU = \frac{\partial U}{\partial r}\hat{\mathbf{e}}_r + \frac{1}{r}\frac{\partial U}{\partial\theta}\hat{\mathbf{e}}_\theta + \frac{1}{r\sin\theta}\frac{\partial U}{\partial\varphi}\hat{\mathbf{e}}_\varphi\,. \qquad (19)$$

For the function described the surface in the spherical coordinate system we can write down the expression

$$U \equiv \sum_{k=0}^{\infty}\sum_{m=0}^{k}(A_{km}\cos m\varphi + B_{km}\sin m\varphi)Q_k^m(\cos\theta) - r = 0\,. \qquad (20)$$

Let's find partial derivatives in accordance with the equation (19). The derivative with respect to r

$$\frac{\partial U}{\partial r} = -1\,, \qquad (21)$$

The derivative with respect to φ

$$\frac{\partial U}{\partial\varphi} = \sum_{k=0}^{\infty}\sum_{m=0}^{k}(-mA_{km}\sin m\varphi + mB_{km}\cos m\varphi)Q_k^m(\cos\theta) \qquad (22)$$

The derivative with respect to θ

$$\frac{\partial U}{\partial\theta} = \sum_{k=0}^{\infty}\sum_{m=0}^{k}(A_{km}\cos m\varphi + B_{km}\sin m\varphi)\frac{\partial Q_k^m(\cos\theta)}{\partial\theta} \qquad (23)$$

Consequently, it is necessary to calculate the derivative of the Schmidt polynomial. It is known the recurrent relation for the Legendre polynomial [7]

$$\sqrt{1-\mu^2}\,\frac{\partial P_{l,k}^m(\mu)}{\partial\mu} = -\frac{i}{2}\Big[\sqrt{(k+m)(k-m+1)}P_{l,k}^{m-1}(\mu) + \\ +\sqrt{(k-m)(k+m+1)}P_{l,k}^{m+1}(\mu)\Big], \qquad (24)$$

that taking into account the relation between the Schmidt and Legendre polynomial [7]

$$P_{0,k}^m(\mu) = i^{-m}\mathrm{Q}_k^m(\mu)\,, \qquad (25)$$

allows to receive after some transformations the final expression

$$\frac{\partial Q_k^m(\mu)}{\partial\theta} = \frac{1}{2}\Big[\sqrt{(k-m)(k+m+1)}Q_k^{m+1}(\mu) - \\ -\sqrt{(k+m)(k-m+1)}Q_k^{m-1}(\mu)\Big]. \qquad (26)$$

It is necessary to analyze separately the case m=0. In this case taking into account a number of the relations [8] it is not difficult to result in

$$\frac{\partial Q_k^0(\mu)}{\partial\theta} = \sqrt{k(k+1)}Q_k^1(\mu)\,. \qquad (27)$$

Using the derived gradient in the local spherical coordinate system $(\hat{\mathbf{e}}_r,\hat{\mathbf{e}}_\theta,\hat{\mathbf{e}}_\varphi)$ it is necessary to represent it in the world Cartesian coordinate system. Taking into account some known relations ones can get the final expression for the normal to object surface represented by spherical coordinates

$$gradU = \hat{\mathbf{i}}\left(-\rho\sin^2\theta\cos\varphi + \sin\theta\cos\theta\cos\varphi\frac{\partial U}{\partial\theta} - \sin\varphi\frac{\partial U}{\partial\varphi}\right) + \\ +\hat{\mathbf{j}}\left(-\rho\sin^2\theta\sin\varphi + \sin\theta\cos\theta\sin\varphi\frac{\partial U}{\partial\theta} + \cos\varphi\frac{\partial U}{\partial\varphi}\right) + \\ +\hat{\mathbf{k}}\left(-\rho\sin\theta\cos\theta - \sin^2\theta\frac{\partial U}{\partial\theta}\right). \qquad (28)$$

The search of the point of ray intersection with the object represented by the spherical harmonics is also a nontrivial task. Let's consider the object represented relative to the origin of the Cartesian coordinate system and the ray

$$\mathbf{r} = \mathbf{r}_0 + \xi\hat{\mathbf{l}}\,. \qquad (29)$$

The cosine of angle θ of the point of ray intersection with the surface can be written as

$$\cos\theta = \frac{(\hat{\mathbf{k}},\mathbf{r})}{|\mathbf{r}|} = \frac{z_0 + l_z\xi}{\sqrt{r_0^2 + \xi^2 + 2\xi(\hat{\mathbf{l}},r_0)}}. \quad (30)$$

For the vector **ρ** we have accordingly

$$\boldsymbol{\rho} = \mathbf{r} - \hat{\mathbf{k}}(\hat{\mathbf{k}},\mathbf{r}) = \mathbf{r}_0 + \xi\hat{\mathbf{l}} - \hat{\mathbf{k}}(z_0 - l_z\xi). \quad (31)$$

The cosine and sine of angle φ have the form

$$\cos\varphi = \frac{(\hat{\mathbf{i}},\boldsymbol{\rho})}{|\boldsymbol{\rho}|} = \frac{x_0 + l_x\xi}{\sqrt{\left(\mathbf{r}_0 + \xi\hat{\mathbf{l}} - \hat{\mathbf{k}}z_0 - \hat{\mathbf{k}}l_z\xi\right)^2}},$$

$$\sin\varphi = \frac{(\hat{\mathbf{j}},\boldsymbol{\rho})}{|\boldsymbol{\rho}|} = \frac{y_0 + l_y\xi}{\sqrt{\left(\mathbf{r}_0 + \xi\hat{\mathbf{l}} - \hat{\mathbf{k}}z_0 - \hat{\mathbf{k}}l_z\xi\right)^2}}. \quad (32)$$

Therefore we get the dependence of angles θ and φ on one variable ξ. The equality takes place in the point of intersection

$$f_{sg}(\theta(\xi),\varphi(\xi)) - \mathbf{r}_0 - \xi\hat{\mathbf{l}} = 0. \quad (33)$$

Solving the equation (33) relatively the parameter ξ we can find all the point of ray intersection with the object surface. In case of the global illumination equation solution for scene rendering we are interested only by the first intersection with the object. It can be found both the standard procedures, for example the bisection method, and more complex and fast algorithms.

7. REALIZATION OF LOCAL ESTIMAION METHOD AND RENDERING OBJECTS REPRESENTED BY SPHERICAL HARMONICS

We implemented the algorithms of the local and double local estimations with 3D object representation in the basis of spherical harmonics. In Figure 2 we see rendering of the simple 3D scene. It is the room represented by the mesh with one column and one omni light source. The head of man in the scene is represented by the surface harmonics. The initial mesh contains 32 654 edges. In Figure 1 it was used $N = 32$ spherical harmonics to play the head. In this case the rendering time increased by 40% in comparison with the visualization of the mesh, but the amount of stored information decreased in more than 1,000 times.

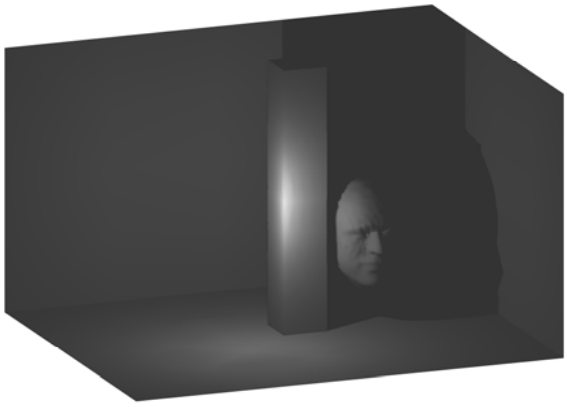


Figure 2. 3D scene rendering by the local estimation method.

8. CONCLUSIONS

The local estimation method of the Monte Carlo methods allows implementing the physically adequate simulation of the global illumination equation solution. In addition it allows obtaining directly the absolute values of luminance in every point and direction of scene. This algorithm opens the new horizons in the photometric research and can be the ground of the new rendering methods on the basis of direct solution of the global illumination equation.

The usage of the object spectral representation on the basis of spherical harmonics is the successful approach for the description of some class objects that allow reducing significantly the information content, which is necessary for the adequate reproduction. The possibility of continuous normal retrieval with the control accuracy in aggregate with the local estimation method is an important approach in the 3D scene rendering and its photometric investigation. The continuous restoring of normals has the special importance in case of luminance angular distribution calculations, where the wrong normal approximating results in considerable degradation of calculation accuracy.

REFERENCES

1. Budak V.P. Visualization of luminance distribution in 3D scene. – M.: MPEI, 2000. (in Russian)
2. Monte-Carlo Methods in Atmospheric Optics / G.I.Marchuk, ed. – Berlin: Springer-Verlag, 1980
3. Foley J.D., van Dam A., Feiner S.K., Hughes J.F. Computer graphics: principles and practice. – Addison-Wesley, 1997.
4. Ermakov S.M., Mikhailov G.A. Course of statistical modeling. - M.: Nauka, 1976. (in Russian)
5. Mousa M., Chaine R., Akkouche S. Frequency-Based Representation of 3D Models using Spherical Harmonics // The 14-th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision "WSCG'2006", Plzen, Czech Republic, January 31 – February 2, 2006. Full Papers Proceedings, p.193-200
6. Tikhonov A.N., Samarsky A.A. Equations of Mathematical Physics. – M.: Nauka, 1976. (in Russian)
7. Vilenkin N.Ya. Special functions and the theory of group representations. Translated from the Russian by V.N. Singh. Translations of Mathematical Monographs, Vol. 22 American Mathematical Society, Providence, R.I. 1968
8. Mishchenko M.I., Travis L.D., Lacis A.A. Scattering, Absorption, and Emission of Light by Small Particles. - Cambridge: Cambridge University Press, 2002
9. Sobolev V.V. Point light source between parallel planes // Doklady of the Academy of Sciences of the USSR, 1944, vol.42, No.4, p.176-177. (in Russian)

Compression of light transport matrix

Andrey Lebedev, Ivan Karpuhin, Andrey Ilyin, Alexey Ignatenko
Department of Computational Mathematics and Cybernetics
Moscow State University, Moscow, Russia
{alebedev, ikarpuhin, ailyin, ignatenko}@graphics.cs.msu.ru

Abstract

We address the problem of relighting of three-dimensional scenes and propose a new method based on reconstruction of light transport matrix. The matrix can be reconstructed on-fly from data obtained by some renderer (we use path-tracing algorithm). We compared our method with existing ones. The proposed method achieves better quality compared to others with the same compression ratio. It is suitable not only for diffuse scenes and can be easily parallelized. In future we can use this method as a new global illumination algorithm.

Keywords: *relighting, photorealistic rendering, light transport matrix, compression.*

1. INTRODUCTION

The relighting problem arises during rendering of three-dimensional scenes. It is particular actual for designers as they often have to modify illumination in modeled scene. Aside from modeling and visualization in architectural rendering we should have an opportunity to integrate a building into existing environment (see Figure 1).

Figure 1: Modeled building (top) and integrated into existing environment (bottom).

When you change the lighting (or environment) in the scene it is necessary to recalculate the whole image. Calculation of a single image can take from half a minute to several hours, depending on the complexity of scenes, and lighting effects. This article discusses the relighting methods based on a preliminary calculation of light transport matrix, addresses compression and recovery problems of light transport matrix as well as the quality of rendered images.

2. REVIEW OF EXISTING METHODS

Initially the relighting problem was solved for some special cases. Debevec [1] modeled reflective properties and geometry of the human face skin to obtain images in various lighting conditions (relighting skin). Very complicated equipment that measured the reflectance of skin for various light sources and camera position was used to model the reflective properties. A special image-based method for face geometry reconstruction was developed. Based on geometrical and reflectivity information the authors changed the illumination of face. Several thousand of images were used for full reconstruction of the human face. Later Wenger [9] improved the method for relighting of human face. He used a special camera for measurements and a special set of light sources. Then the author obtained an image for each light source from a special set and relit the face representing a new light source as a linear combination of basis sources. The relit image was equal to the linear combination with the same coefficients of images for light sources from special set.

Peers [6] formalized the relighting problem as follows:

$$c = T \cdot l \qquad (1)$$

where c - is a vector of image pixel intensities, l - is a vector of light source intensities, T - light transport matrix. (i, j)-element of light transport matrix is a contribution of light source j into the intensity of image pixel i. Due to the linearity of light we have formula (1).

Relighting problem was formalized for a fixed camera position case. Lawrence [4] generalized formula (1) for an arbitrary camera position case considering the light transport matrix as a relationship between light source vector and vertices of polygonal model of the scene. However this approach leads to the problems connected with the calculation of the view-dependent material models.

The light transport matrix is usually huge (sometimes it occupies several gigabytes). That is why the most important trend in relighting is matrix compression. Wang [8] proposed Nystom algorithm for compressed representation of light transport matrix. In this case instead of the entire matrix we store some of its rows and columns, the rest of the matrix was reconstructed on the basis of these rows and columns. Mahajan [5] compressed light transport matrix using the property of locally low rank of this matrix. He divided the whole image into blocks and each block was reconstructed independently using the method of principal components (PCA).

An alternative way to compress light transport matrix is compressed sensing [2]. Sen [7] and Peers [6] investigated the problem of matrix restoration by a small set of measurements. They considered the matrix of light transport in a special basis

(obtained a sparse signal). Then the signal was recovered by compressed sensing methods.

In our work we used a standard formulation of the relighting problem by Peers [6], the idea of application of Nystrom algorithm for matrix recovery and local blocks of Mahajan [5]. We have developed a system for scene relighting and provide several algorithms for light transport matrix compression.

Figure 2: Results of relighting (Conference Room).

3. DESCRIPTION OF RELIGHTING SYSTEM

3.1 Basic implementation

The developed system reconstructs light transport matrix for arbitrary three-dimensional scene and relights this scene using equation (1). Light transport matrix is calculated using path tracing algorithm [3]. Note that the total work time of path tracing algorithm for light transport calculation is practically the same one as for rendering of one image because for all backward ray tracing algorithms we need to compute all paths from camera to various points of scene.

Relighting step is based on equation (1). We used CUDA programming technology to accelerate matrix multiplication. Figure 2 shows the results of relighting of benchmark Conference room scene. The rendering time is less than 1 second for 1024 by 768 resolution on NVIDIA 240M GPU. 20 light sources were used in this scene.

In Fig.3 we can see the result of relighting of scene with complex light effects using area light source (resolution - 32 by 32).

Figure 3: Relighting results of the scene with an area light source.

3.2 Global compression of light transport

The main problem of the basic implementation is the large size of the light transport matrix. For a scene with an area source in Figure 3 the matrix size was 9 GB. There are problems with matrix storage, slow work with the disk that significantly reduces the relighting speed.

Wang [8] proposed to compress the light transport matrix using Nystrom method. Also it was considered an algorithm for rows and columns retrieval for Nystrom method from images of real scenes. By analogy with [8] we have developed several versions of the algorithm for rows and columns selection:

- *Algorithm 1*. Choose arbitrary r_0 rows, consider them as a set of columns with dimension r, clusterize vectors using k-means method. Centers of the clusters are needed columns (see Fig. 4 - marked in green).
- *Algorithm 2*. Choose arbitrary c_0 columns, calculate r_0 lines using k-means.
- *Algorithm 3*. Choose random c_0 columns and random r_0 columns.
- *Algorithm 4*. Choose arbitrary c_0 columns, calculate r_0 lines using k-means. Then choose other c_0 columns - obtain new r_0 lines, etc. As an output we obtain the union of all rows and columns.

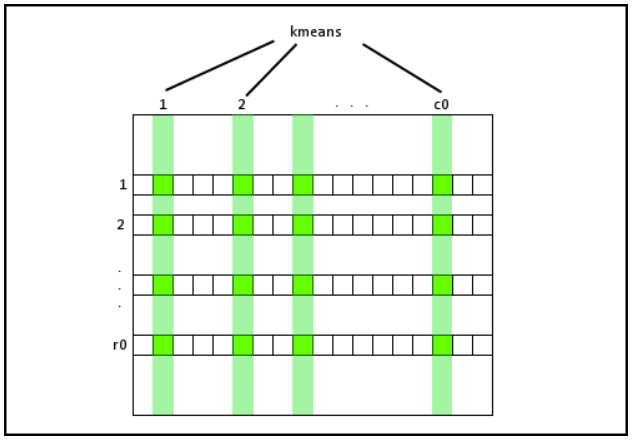


Figure 4: *Scheme for rows and columns selection.*

In [8] Wang also proposed to use a nonlinear transformation of light transport matrix to maximize compression. Our tests show

that the algorithm with a nonlinear transformation is not resistant to noisy images obtained by path tracing. Also it works well on a number of specific scenes (with diffuse materials and blurred caustics). In scenes with bright reflections, visible light source and bright caustics the algorithm generates artifacts (see Fig.5). To remove artifacts we should significantly increase the number of rows and columns.

Figure 5: *Artifacts of nonlinear transformation algorithm.*

3.3 Local compression of light transport

Mahajan in [5] showed the benefits of reconstruction of independent light transport matrices for the different image blocks. We have also implemented algorithms for matrix reconstruction independently for each image block. Local compression of the light transport matrix can adaptively compress the matrix, i.e. use a different number of rows and columns for reconstruction of different blocks. For example, blocks with a uniform illumination usually require fewer rows and columns than the blocks with bright reflection (or caustics).

Another advantage of this approach is the ability to parallelize the algorithm, since image blocks can be processed independently.

3.4 The proposed algorithm

We propose the following algorithm to compress the light transport matrix. First of all, the entire image is divided into blocks. For each block, we use Algorithm 4 for selecting rows and columns (see section 3.2). For on-fly reconstruction of full matrix we use Nystrom method [8].

In the next section, we analyze the algorithms described in section 3.2, and compare them with the proposed one.

4. RESULTS AND COMPARISON

In the first stage, we compared our algorithm with block versions of algorithms 1-3 (see section 3.2). The comparison was performed using different types of scenes: diffuse, with caustics, reflections and mixed scenes.

To assess the quality of synthesized images we used PSNR metric between the images generated using full light transport matrix and its compressed form. Calculating the PSNR we considered different light sources (different color, position and size), then PSNRs were averaged. Fig. 6 presents the results of algorithm comparison of average PSNRs for all types of scenes. All the algorithms show the same result on diffuse scene. Our algorithm shows much better results than the other ones on the scenes with reflections, caustics and visible light source. Figs. 7-8 present rendering results of our algorithm on all types of scenes.

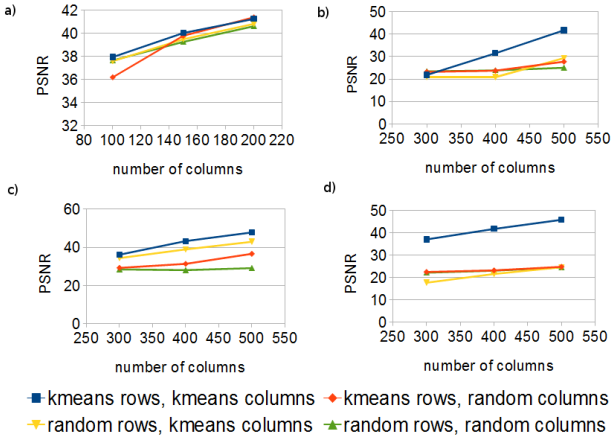


Figure 6: *Dependency of average PSNR on number of columns (a - diffuse scene, b – reflections, c – caustics, d – mixed scene).*

Also, we have compared our algorithm with a nonlinear algorithm of Wang [8]. Earlier in the section 3.2 were mentioned the main problems of nonlinear algorithm. In [8] light transport matrix for diffuse scene was compressed 4000 times. This was achieved by using a huge resolution of the light source. We showed that increasing the resolution of the light source does not lead to the increase of the matrix rank (it remains constant). This is because the new columns of light transport matrix are actually the result of interpolation (i.e., linear combination) of the old ones. We compared the compression ratios of our algorithm and the nonlinear one [8], using a light source with resolution 64 by 64 (see Fig. 9).

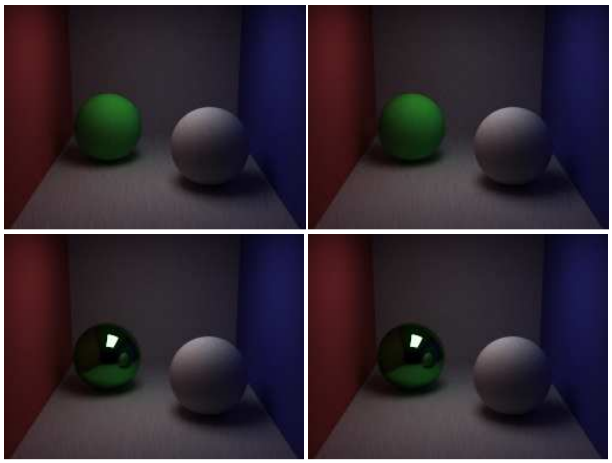


Figure 7: *Rendering results of our algorithm on different types of scenes: diffuse and specular (left – obtained by full matrix, right – by our compressed form).*

The comparison was made for the diffuse scene and the algorithms behave the same way. However, for other types of scenes a nonlinear algorithm is much worse.

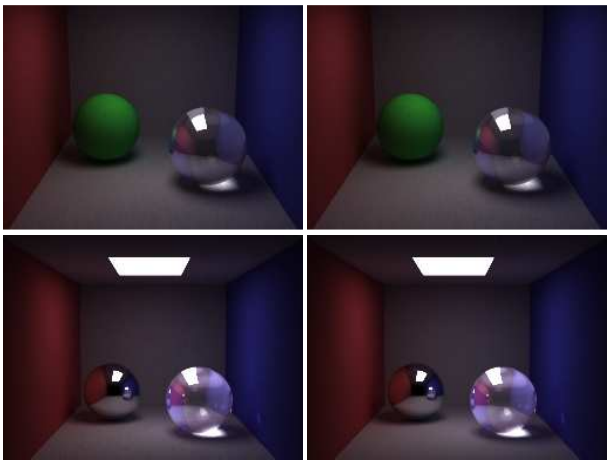


Figure 8: *Rendering results of our algorithm on different types of scenes: caustics and general (left – obtained by full matrix, right – by our compressed form).*

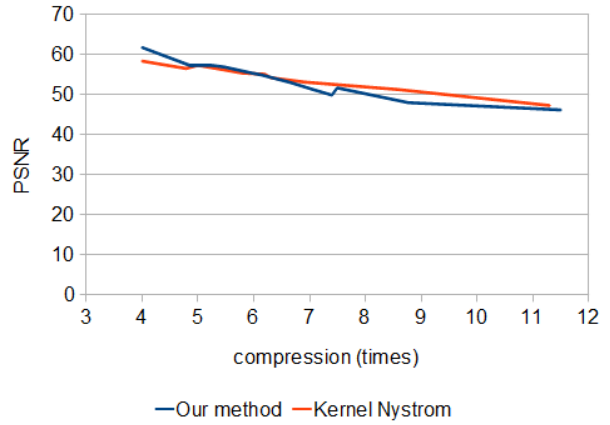


Figure 9: *Dependency of average PSNR on compression ratio for our method and nonlinear one.*

5. CONCLUSION AND FUTURE PLANS

We proposed a new algorithm to compress light transport matrix. The results of algorithm analysis and comparison with other ones show that the proposed algorithm works better than others on some types of scenes, more resistant to noise and it can be easily parallelized.

Our results can be used as a new algorithm for ray tracing with higher convergence rate than available ones. Algorithm for selection of rows and columns obtains the most important for ray tracing paths from the light source to camera. Also we are going to explore the issue of ray tracing acceleration.

6. LITERATURE

[1] Debevec P., Hawkins T., Tchou C., Duiker H.-P., Sarokin W., Sagar, M. Acquiring the reflectance field of a human face. In *Proceedings of ACM SIGGRAPH 2000*, Computer Graphics Proceedings, Annual Conference Series (2000), 145–156.

[2] Donoho D.L. Compressed sensing. *IEEE Transactions on Information Theory* 52, 4 (2006), 1289-1306.

[3] Kajiya J. The rendering equation. *ACM SIGGRAPH Computer Graphics*, 20, 4 (1986), 143-150.

[4] Lawrence J., Stamminger M., Huang F., Ramamoorthi R. Sparsely precomputing the light transport matrix for real-time rendering. *Computer Graphics Forum* 29, 4 (2010), 1335-1345.

[5] Mahajan D., Shlizerman I.K., Ramamoorthi R., Belhumeur P. A theory of locally low dimensional light transport. *ACM Transactions on Graphics* 26, 3 (2007).

[6] Peers P., Mahajan D. K., Lamond B., Ghosh A., Matusik W., Ramamoorthi R., Debevec, P. Compressive light transport sensing. *ACM Transactions on Graphics 28*, 1 (2009), 3:1–3:18.

[7] Sen P., Darabi S. Compressive Rendering: A Rendering Application of Compressed Sensing. *IEEE Transactions on Visualization and Computer Graphics*, 99 (2010), 1-14.

[8] Wang J., Dong Y., Tong X., Lin Z., Guo B. Kernel Nystrom method for light transport. *ACM Transactions on Graphics* 28, 3 (2009), 1-10.

[9] Wenger A., Gardner A., Tchou C., Unger J., Hawkins T., Debevec P. Performance relighting and reflectance transformation with time-multiplexed illumination. *ACM Transactions on Graphics 24*, 3 (2005), 756–764.

About the authors

Andrey Lebedev is a Ph.D. student at Moscow State University, Department of Computational Mathematics and Cybernetics. To get more information about him visit http://lebedev.as.

Ivan Karpuhin is a student at Moscow State University, Department of Computational Mathematics and Cybernetics.

Andrey Ilyin is a researcher at Moscow State University, Department of Computational Mathematics and Cybernetics. His research interests include 3D reconstruction, camera calibration, computer vision and image processing. His contact e-mail is ailyin@graphics.cs.msu.ru.

Alexey Ignatenko is a researcher at Moscow State University, Department of Computational Mathematics and Cybernetics. His research interests include photorealistic 3D rendering, 3D modeling and reconstruction, image-based rendering and adjacent fields. His contact e-mail is ignatenko@graphics.cs.msu.ru.

Algorithms for calculating parameters of virtual scenes in computer vision tasks

Pavel Samsonov, Andrey Ilyin, Aleksey Ignatenko
Department of Computational Mathematics and Cybernetics
Moscow State University, Moscow, Russia
psamsonov@graphics.cs.msu.ru, ailyin@graphics.cs.msu.ru, aignatenko@graphics.cs.msu.ru

Abstract

In this paper we describe methods for determining the parameters of the camera and the light spot, which due to its properties is similar to a point light. The position and the rotation of the camera in the scene coordinates are determined, as well as the position of the light spot, its intensity and color. Parameters are determined from the calibration object in the photograph and some known properties of the camera.

Keywords: *Camera calibration, determination of illumination parameters, calibration object.*

1. INTRODUCTION

In computer graphics many problems are associated with the tasks of three-dimensional reconstruction. For example, a simple synthesis of a textured three-dimensional model of the object by photos. For it not only an array of photographs of the object from different angles is needed, but also the camera positions from which the shots were made. For physically accurate simulation of reflective surface properties of objects we also need light spots' positions, spots' brightness and color. If there are lots of pictures, hand measurements of each of the camera (and maybe a light spot) positions can take a long time. Also, often the accuracy of manual measurements is not enough for such tasks.

In this paper algorithms and methods of automatic determining the camera and lighting settings. The input parameters to the algorithms are some camera settings, the size of a calibration object and a set of photographs, which depict the calibration and monitored objects. The positions of cameras and light spots in different photographs may differ. The output is the camera and light spot parameters that were searched for and undistorted images.

2. EXISTING METHODS REVIEW

Currently, we found several existing methods for camera calibration and light parameters determination.

The best-known and commonly used approaches in the camera calibration are the standard method, proposed by Roger Y. Tsai ([4]) and the more modern method, developed by Zhengyou Zhang ([5]). Standard calibration method is to search for specific points on the image and to solve PnP-task(match n points on the image and in the scene, where n must be at least 4). The newer method involves using a checkerboard(its cells can be quickly and easily found on the image, and in this method not only points are matched, but also lines, what gives some additional accuracy in determining the camera position and distortion coefficients).

Zhengyou Zhang method for calibrating the camera in this task could not be used because calibration object should not take much space in the photo whereas accuracy of Z. Zhang's camera calibration method is highly dependent on the size of the chessboard in the image. There is also difficulty in fixing cones relatively to the checkerboard and placing them on a monochrome base, which would greatly improve the accuracy of finding the tips of the shadows.

In the method of Roger Y. Tsai, the main problem is to find the calibration points in the image. The authors suggested a method based on the two previous: on the edges of the gray card small checkerboards of different sizes were placed, what greatly accelerated the determination of calibration points and increased the number of n matches in the PnP - problem.

Determination of parameters of light is less popular task, and the only method that was found by authors is to determine the parameters of light by a mirror sphere. But in order for results were accurate the method requires perfectly circular mirror sphere of large size, which is difficult to find, and it like the checkerboard takes much space on the photos, substantially degrading the accuracy of reconstruction of the object.

3. THE PROPOSED METHOD

For physical accuracy, some non-linear filters are applied on the original photo. They consider the camera response curve and distortion coefficients.

At the stage of camera calibration special calibration points on the calibration object are searched for and then matrix of external camera calibration and coordinate systems of camera are calculated. At the stage of the light parameters determination position of the light spot is restored by cones on the calibration object and by the shadows from them. Then by several photos with known light spot coordinates the intensity and the color of light are restored.

When realizing the method the OpenCV library was used.

3.1 Calibration object

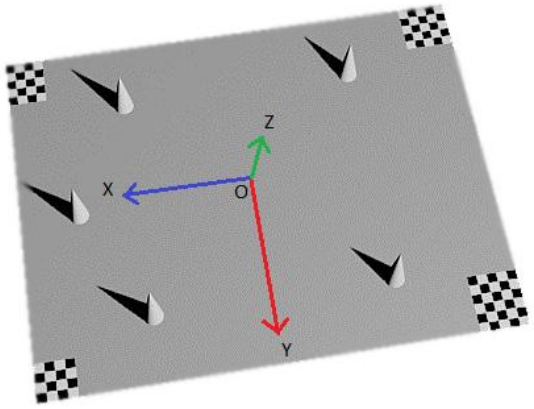


Figure 1: The calibration object synthesized in 3Ds Max

As part of the problem being solved was the design of calibration object (see Figure 1).

The object consists of:

- Base, which is a rectangular gray card of known dimensions
- 4 checkerboards in the corners of the gray card of size 4x4, 4x5, 5x5 and 5x6 with known size of cells
- n cones with a certain height, $n \geq 2$, for the good accuracy of the desired number $n \geq 5$; columns are arranged in an arbitrary manner

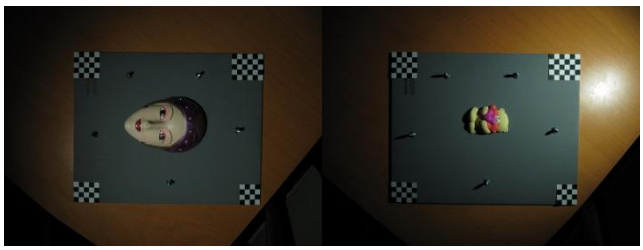


Figure 2: Calibration object with different test objects on it

3.2 Response curve

To achieve physical accuracy, photos should be first processed to restore true colors by non-linear transformation. Response curve is vector function $I = f(M)$, where P is pixel intensity in the image and M is pixel intensity in the camera matrix. To restore initial colors on the matrix, reverse transformation can be used by formula $M = f^{-1}(I)$.

In different cameras response curves may differ very much.

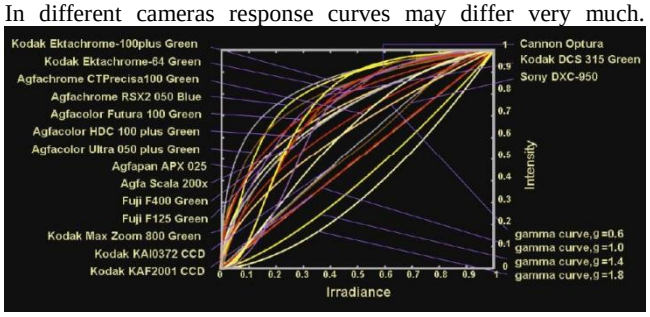


Figure 3: Response curves of some cameras.

Response curve can be achieved from camera's passport characteristics. But sometimes they are incorrect; in this case response curve can be calculated by an external program, for example, "HDRShop" (http://www.debevec.org/HDRShop).

3.3 Camera calibration

For camera calibration it is necessary to compare the coordinates of the scene and points on the image. Coordinates of the scene points can be calculated using the input parameters (properties of the calibration object: the size of a gray card, chess cells, etc.). Points on the calibration object are searched through the following steps:

- Image binarization by thresholding to segment black and white squares
- Find corners of the black squares
- Find contours of the boundaries of the black regions
- Select contours of suitable shape
- Approximate these contours with 4-vertex polygons
- Among these select the quadrangles resembling calibration pattern squares
- Extract corners of the selected quads, having at least one corner in vicinity
- Group the corners of the selected quadrangles in lines according to calibration object size

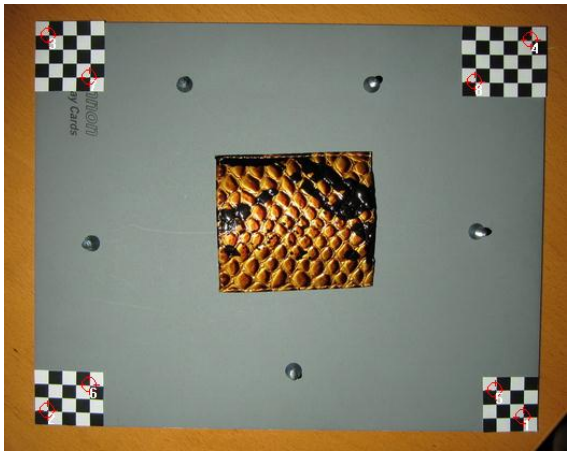


Figure 4: Found calibration points, highlighted with red circles

Then using the method of Roger Y. Tsai from the known relations between gauge points in the scene and the image and from the intrinsic calibration matrix extrinsic and a full calibration are calculated. If necessary, also at this stage the homography matrix and/or distortion coefficients are computed and then distortion is removed from the photographs.

Using 3x4 full calibration matrix(F) to calculate image point(ip) from scene point(sp):

$$v = F * (sp_x,\ sp_y,\ sp_z, 1)$$

$$ip = \left(\frac{v_x}{v_z}; \frac{v_y}{v_z}\right)$$

Using 3x3 homograph matrix(H) to calculate scene point(sp) = (X; Y; 0), from image point(ip) = (x; y):

$$v = F * ip$$

$$sp = \left(\frac{v_x}{v_z}; \frac{v_y}{v_z}; 0\right)$$

Before camera calibration, also base centers should be marked by user. Because cones in the scene may be of any color and size, automatic detection for them is hardly possible.

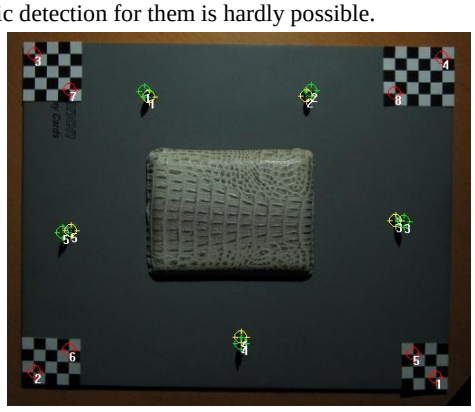


Figure 5: Yellow points are marked by user cone base centers, green points are restored cone peaks.

3.4 Calculation of distortion coefficients

Distortion is a deviation from rectilinear projection (a projection in which straight lines in the scene remain straight in an image). It appears as a failure of a lens to be rectilinear. By matches between calibration points in the scene and on the image distortion coefficients can be calculated. Then using the distortion coefficients image can be undistorted, so straight lines on the image become actually straight. But sometimes it leads to smoothing image in some regions.

More information can be accessed in Brown D.C.'s article ([7]).

3.5 Determination of parameters of light

To determine the position of the light spot cone shadow tips should be found in the photo and translated into world coordinates. Initially, the image is segmented using the Mean Shift algorithm ([6]) to find shadow areas. First, those areas which have common points with small circular areas located between the tips of the cones and the centers of their bases are removed from consideration, greatly reducing the probability of error. Then the image is searched for shadows on the following criteria:

- Distance from the nearest point of the image to the center of the cone base
- Distance from the furthest point of the center of the cone base
- The average brightness of pixels in area (the shadow is almost always darker than the surrounding areas)
- Linear elongation region (cubed distance between the maximum and minimum distance from the center of the cone base points of the domain divided by the square area)

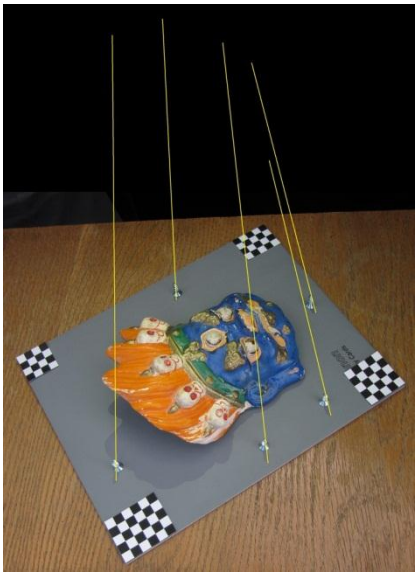


Figure 6: Restoring the position of the light spot by cone shadow tips

We know world coordinates of shadow tips and cone peaks, so we can calculate the approximate intersection point of rays (see Figure 4). If number of these rays equals n, then the number of the midpoints of the common perpendiculars between every two rays will be $N = \frac{n(n-1)}{2}$. So the real point is somewhere in the cloud of these N points.

To recover the coordinates of the light spot from the cloud of points 2 formulae were used: arithmetic mean of these points and the "weight" formula, which calculates the sum where those points for which the distance to other points is less are included with larger weights.

Tests showed that the "weight" formula often provides with more accurate values of the light spot position than the arithmetic mean.

4. RESULTS

Algorithms of camera calibration, distortion correction and determination the position of the light spot have been successfully verified first on synthetic and then on real data.

4.1 Synthetic data

Synthetic data verification was held on two sample images. Known and calculated light spot coordinates were compared to compute error. Error camera position vector length was less than 1.5% of the length of the true position vector of the camera, the error vector of the light spot position length by 5 cones was less than 5% of the true position vector of the spot.

Here are presented two tables of errors for each light spot calculating formula when processing synthesized images. Formula to compute errors was $Err = \frac{|\vec{s} - \vec{t}|}{|\vec{s}|} * 100\%$, where $\vec{s}$ is real position of light spot and $\vec{t}$ is position, which was calculated by the algorithm.

Formula "average mean"		
n	Sample 1	Sample 2
2 cones	90%	14%
3 cones	25%	5%
4 cones	14%	2.8%
5 cones	5.8%	2.6%

Formula "with weights "		
n	Sample 1	Sample 2
2 cones	90%	14%
3 cones	13%	5%
4 cones	6%	3.8%
5 cones	4.7%	2.2%

4.2 Real data

Calculation errors on real data can't be calculated with the same accuracy as on the synthetic. Measurements were made of the error of both restoring the camera and light spot positions. Tests were performed on a large array of photographs in which the positions of the camera and light spot have not changed as the calibration object moved. We compared the position of light spots in the camera coordinate system. The average error was approximately 4.45% (if more precision is needed - just increase the number of cones).

4.3 Re-projection

Also, the accuracy of the restored position of the light spot can be estimated visually. After the calculation of the light spot position, rays are projected through the cone tops back to the gray card, and the closer they are to their initial positions the more accurate the restored light spot position is.

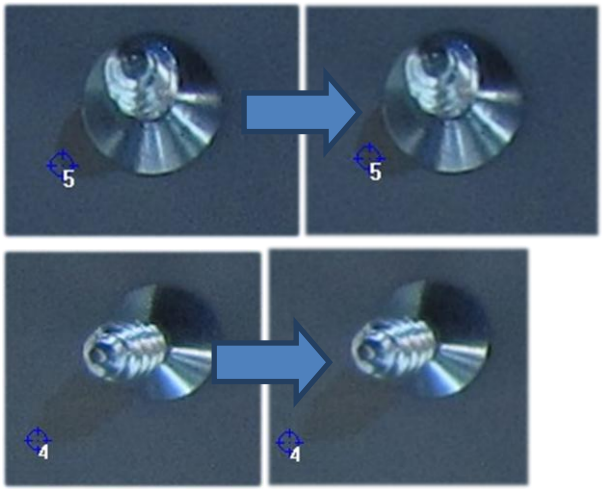


Figure 7: Re-projection

Blue points in circles are marked shadow tips. As can be seen in the photographs, the point shift is little, so the accuracy of reconstruction of the light spot is rather high.

5. CONCLUSION

In general, the methods described in this article were already successfully used by several people in their works. Accuracy of reconstruction by 5 cones is large enough, but if you need additional accuracy, you can add several more cones.

Only automatic detection of calibration points and the tips of the shadows are problematic features of this work. Averagely, about 80% of points are recognized on each photo as the process takes from several seconds to half a minute. In the future the author intends to develop more accurate and faster methods of recognition of these points in the images.

6. LINKS

[1] James A. Paterson, David Claus, Andrew W. Fitzgibbon, “BRDF and geometry capture from extended inhomogeneous samples using flash photography”, *EUROGRAPHICS 2005* Volume 24, Number 3 p. 4.

[2] Alexey Kravtsov and Vladimir Vezhvenets, “Solve of PnP problem”. *Computer graphics and multimedia,* Volume №1(3)/2003.

[3] Alexey Kravtsov and Vladimir Vezhvenets, “General formulation of extrinsic camera calibration task”. *Computer graphics and multimedia,* Volume №1(3)/2003.

[4] Roger Y. Tsai, “A Versatile Camera Calibration Technique for High-Accuracy 3D Machine Vision Metrology Using Off-the-Shelf TV Cameras and Lenses” *IEEE Journal of Robotics and Automation* 1987 Volume 3 Issue 4.

[5] Z. Zhang, “A Flexible New Technology for Camera Calibration” *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 22(11):1330-1334, 2000.

[6] Fukunaga, Keinosuke; Larry D. Hostetler (January 1975). "The Estimation of the Gradient of a Density Function, with Applications in Pattern Recognition". *IEEE Transactions on Information Theory (IEEE)* 21 (1): 32–40.

[7] Brown DC (1966). "Decentering distortion of lenses". *Photogrammetric Engineering*, **7**: 444–462.

Modeling of the light-scattering properties of the metallic coating

Kristina Zipa, Andrey Ilyin, Aleksey Ignatenko
Department of Computational Mathematics and Cybernetics
Moscow State University, Moscow, Russia
kzipa@graphics.cs.msu.ru, ailyin@graphics.cs.msu.ru, aignatenko@graphics.cs.msu.ru

Abstract

The process of the manual creation of the realistic models is labor-intensive. In order to facilitate the modeling process we propose a fully automatic method for determining the light-scattering properties of objects from photographs taken with a digital camera. In this work we also propose new reflectance model of the textured and metallic coating surfaces.

Keywords: material, texture, metallic coating, reflectance properties, light-scattering, reconstruction.

1. INTRODUCTION

Rendering of the photorealistic images of three-dimensional scenes remains a significant problem in computer graphics nowadays. This problem is significant in the process of the creation of computer games, 3D movies and virtual reality systems. Designing of the virtual scenes consists of the creation of the complex objects – models. At the rendering process reflective and light-scattering properties (textures) are assigned to virtual objects for physical and visual similarity with the real world scenes. The modeling of the metallic effect is often used for photorealistic synthesis of the paint-and-lacquer coating surfaces. Nowadays there are many approaches in the modeling of the materials of the objects. In some cases tabular BRDF [1] are used. It requires large storage, but ensures a high quality results. The acquisition of such models requires special photometric devices.

Therefore, in many cases, preference is given to the parametric models of materials for which exists a lot of effective ways of visualization. Using of this kind of models significantly limits a class of described materials. One of the additional means to enhance the realism is the texturing. When rendering, texture is represented by a bitmap image imposed on the surface of the 3D-model to give it color or the illusion of relief. A wide range of surfaces can be described with textures (soil, plants, minerals, fur and leather, etc.).

Metallic is a lamellar shaped particles of metal (such as aluminum, copper, zinc, brass or bronze) is commonly used in multilayered paints. The surface covered with such paint looks sparkling (see Figure 1). Paint with a metallic effect is used in the automotive industry and in the manufacture of different kind of appliances and cosmetics.

Tabular BRDF can be used for local description of metallic effect. In the case of analytical models additional problem accrues: the micro relief reconstruction or modeling of the sparkles visual effects. The main purpose of this paper is to develop algorithms and methods for determining the light-scattering properties of textured surfaces by photographs.

The input data in the proposed approach are the photographic images of a flat surface, made with various lighting conditions, corresponding normal maps to the surface, light sources and camera positions at the moment of shooting. At each input image the presence of a glare from light source is supposed. The presence of metallic effect is also allowed. The investigated material is assumed to be isotropic, not fluorescent.

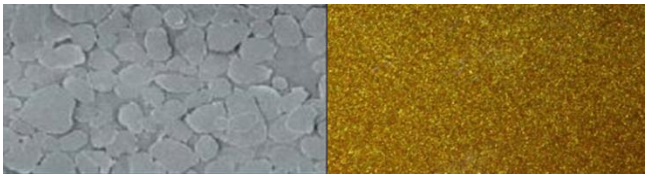


Figure 1: Set of particles (left); Metallic coating (right).

In our work we propose the material model which determines the light-scattering properties of surfaces and metallic effect. Model must satisfy the following requirements:

- Small number of photos for the material model construction;
- Interactive visualization;
- Small amount of memory for storage.

Algorithm for constructing of the proposed model from photographs taken under various lighting conditions was developed as a part of this work. The verification of the model and algorithms was made by comparison of the original photos with the results of visualization under corresponding lighting conditions.

In the following section there is an overview of existing methods of the modeling of the light scattering and reflective properties of surfaces. In **Section 3** the proposed model of a material is described. **Section 4** describes how to find model's parameters from photos. In **Section 5** the basic results of work of algorithm are presented. Finally, main results are provided in **Section 6**.

2. RELATED WORK

To obtain the material models manual modeling with some applications (BRDF Shop) can be used. Parameters of reflective properties of a material are adjusted by the designer. Manual modeling of real world materials can take a lot of time.

For automatic reconstruction of light-scattering properties often high-precision photometric devices or special acquisition systems are used [1]. The sample of the surface covered with the investigated material is illuminated from different directions with the ray of light and the reflected light for various observation angles is measured. The obtained data are usually presented in tabular form. Data storage requires large amounts of memory. For the modern real-time graphic applications the storage problem is still actual. In addition, the material is often assumed to be homogeneous, which eliminates the possibility of texture measuring.

A variety of methods has been developed to determine the textures of materials. Some of them are aimed at texture construction directly during rendering process, for example *BTF* [2], view-dependent textures [3]. Some use preliminary

determining, for example texture weaving [4] and image-based texturing [5]. Bidirectional texture function (BTF) is a group of methods, implying the accumulation of large databases on types of surfaces. View-dependent textures method implies high-quality reconstruction of the texture in the rendering process. Input images are projected onto preliminary reconstructed scene. Methods of preliminary texture reconstruction closely connected with geometry reconstruction. Typically, these approaches are applied to use expensive systems of 3D-scanning. Texture weaving divides a texture into patches and for each patch links it to one of the original images. Texture weaving has a high speed because it uses hardware acceleration, and hash maps.

There are some systems for manual modeling of metallic effect (3dsMax, Photoshop, the system, described in [6]). A part of them don't assume interactive visualization, while others only simulate effect, without allowing the achievement of continuity when moving the light source or camera. In the paper [7] metallic parameters in the form of probability model are determined from the given BRDF table. But the approach does not solve the problem of reconstruction of the metallic parameters from photos. The main drawbacks of existing methods:

- Often special equipment is needed;
- Large amounts of data for rendering;
- Texture or only a homogeneous material reconstruction;
- Occurrence of artifacts related with direct illumination;
- Lack of continuity between rendering of the neighbor frames.

The proposed algorithm is designed to eliminate these shortcomings.

3. PROPOSED MODEL

This section presents the proposed model of light-scattering properties of a material containing metallic.

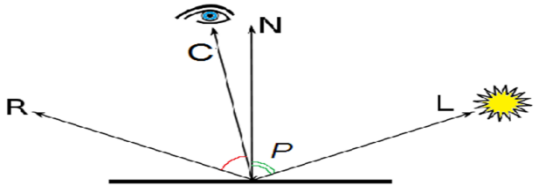


Figure 2: Vectors in proposed model.

Let Ω be a set of points. All examined points are assumed to belong to one of two classes. Either they belong to the investigated material, or to the metallic. That means:

$$\Omega = Material \cup Metallic,\ Material \cap Metallic = \emptyset$$

Let p be an examined point (Figure 2). $\mathcal{M}(p)$ is the model, that describe metallic. Then proposed parametric material model is:

$$M(p, \bar{L}, \bar{C}) = \mathcal{M}(p)\ /+/\ Phong(p)$$

/+/ - is a special addition operation, i.e. either p is point of metallic presence, or it contains only the texture and the specular component. To describe the light-scattering properties of the surface we use Phong reflection model:

$$Phong(p) = d(p) \cdot (\bar{n}(p), \bar{l}(p)) + s(\bar{r}(p), \bar{v}(p))^{\alpha}$$

Specular reflection parameters s and α are the same for all points. Parameter of the scattering power $d(p)$ is different for every point and onwards will be named *texture*.

Consider the metallic model $\mathcal{M}$ in details. On the one hand, the metallic is a set of specifications, on the other - a set of particles. Our model combines these two views.

Each particle is presented in the form of an ellipse. It has geometric parameters and color parameters. Color is presented in the format L*a*b. The geometrical parameters of particles are: the area, the size of the major and minor axis, inclination angle to the horizontal axis. Color parameters are luminosity and color.

The probability metallic model $\mathcal{M}$ depends on the distribution of area, shape and color of the particles.

$\mathcal{M}$ *(Size(m,l,k), Shape(b_1,k_1,b_2,k_2), Color(R_E, G_E, B_E, R_D, G_D, B_D))*

Size(m,l,k) - area distribution (*m,l,k* are parameters of approximated histogram function);

Shape(b_1,k_1,b_2,k_2) - shape distribution (b_1,k_1,b_2,k_2 are parameters of limitation lines for semi-major axis);

Color(R_E, G_E, B_E, R_D, G_D, B_D) - color distribution, where *E* means Expectation, and *D* - Dispersion of RGB.

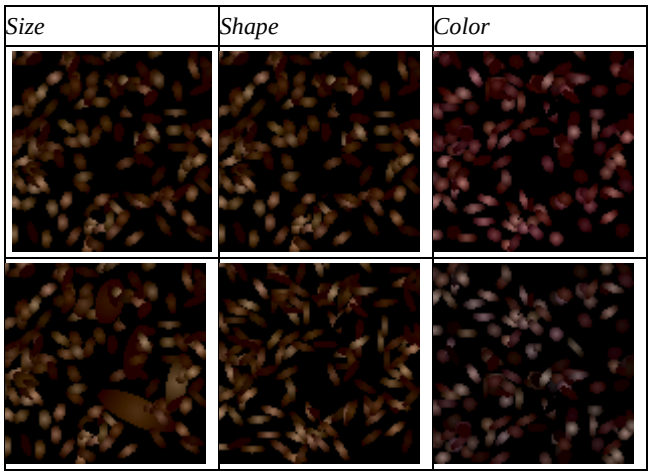


Figure 3: Metallic effect's dependencies from $\mathcal{M}$ parameters.

The proposed model $M(p, \bar{L}, \bar{C})$ takes a small amount of storage in memory, needs only one photo for construction an can be interactively rendered.

4. PROPOSED METHOD

Now we will consider the algorithm for constructing the proposed model from images. Algorithm receives *N* photos of flat surface which contains the texture and metallic. These photos should be taken with different light conditions and camera positions. The output consists of parameters of the proposed model (*T, s, n, Size, Shape, Color*).

Since the model is divided into two independent parts the algorithm of its construction, is also divided into two steps. Thus it is necessary to determine parameters of the Phong and *Metallic* models.

4.1 Determining of Phong Model

At first, consider the algorithm for texture and reflection coefficients of *s, n*. Since the original data contain some errors due to imperfection of the shooting conditions, surface roughness, digital noise, errors while constructing a scene, a bad approximation of the Phong model, the direct solution of linear algebraic equations is impossible. Therefore it is necessary to solve the minimization problem. It's required to find *d (p), s, n*, which minimizes the error function:

$$F(d(p), s, n) = \sum_{k=1}^{N} \left(I_k(p) - d(p) \cdot NL_k(p) + s \cdot NL_k(p)^{\alpha}\right)^2,\quad p \in Material.$$

We use an iterative algorithm. The overall process is shown in Figure 4.

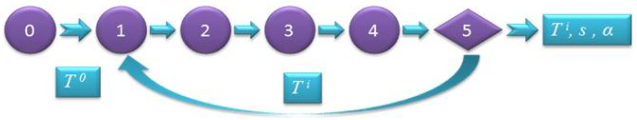


Figure 4: Determining Phong model parameters.

The algorithm iteration consists of five internal steps and of the choice of initial approximation. Parameters of specular component of model are being optimized within every iteration. Globally the process optimizes a texture as long as the quality of the constructed model can be improved. Now we postpone the question of choosing the initial approximation and consider the steps 1-5 in more detail.

4.1.1 Getting specular component

Denote the current iteration *i*. At the previous iteration there was obtained texture $T^{i-1}(p)$. Knowing it we can express the specular component as the difference between the original image and the diffuse component. Diffuse and specular components:

$$Diffuse(p) = T^{i-1}(p) \cdot (\bar{n}(p),\bar{l}(p)).$$

$$Specular(p) = Image(p) - Diffuse(p) = s(\bar{r}(p),\bar{v}(p))^{\alpha}$$

4.1.2 Selection of control points

Specular contain various inaccuracies associated with the approximate texture of $T^{i-1}(p)$ in addition source images also contain noise and various distortions. By optimization of the artifacts have a significant impact on the resulting parameters s^i, α^i. To get rid of these inaccuracies and reduce the number of data for further optimization process, we apply the special filter.

We divide the whole set of points into subsets. Each subset includes only points for which $(\bar{r}(p),\bar{v}(p))$ is the same. In each set we average all included color values. From each set we choose an arbitrary point and put it in a set of control points *Material'*. Each control point corresponds to the average color c_i.

4.1.3 Optimization of specular parameters

Now we solve the problem of optimizing the parameters s^i, α^i over the set of *Material'*. In order to optimize the desired parameters we use Levenberg-Marquardt method. Minimization is performed for each color channel separately. The functional to minimize is:

$$F(s,n) = \sum_{p \in Material'} (s \cdot (\bar{r}(p),\bar{v}(p))^{\alpha} - c(p))^2,$$

c is the color of pixel p.

4.1.4 Recalculation of texture

Subtract from the original image reflectance component.

$$Diffuse(p) = Image(p) - Specular(p).$$

Recalculate the texture as an average:

$$T^i(p) = \frac{1}{N}\sum_{k=1}^{n}\frac{Diffuse_k(p)}{(\bar{n}(p),\bar{t}(p))}$$

Now we can visualize the images with obtained parameters:

4.1.5 The decision of completion

We calculate the error of recovering light-scattering properties of the material in the iteration *i*.

$$Err_i = \sum_{p \in Material} \sum_{k=1}^{N} (Image_k(p) - Image'_k(p))^2$$

If the difference $|Err_{i-1} - Err_i|$ does not exceed the preassigned ε, then the algorithm stops, otherwise the process moves to the next iteration.

4.1.6 Choice of initial approximation

We put an initial approximation to zero at any point. This is equivalent to the recovering *s* and *α* initially from the original image. In this case, the convergence will be slow. In addition, the end result in this case is highly dependent on the type of input data. Requirement for the initial approximation are elimination of the primary defect and reduction in the number of iterations.

Divide the original images at $(\bar{n}(p),\bar{l}(p))$. For each point we take the minimum among all images. So we get the texture T_{min}.

$$T_{min}(p) = T^0(p) + X(p); \qquad s^0 \cdot \frac{(\bar{r}(p),\bar{v}(p))^{\alpha^0}{}_{min}}{(\bar{n}(p),\bar{t}(p))_{min}} \stackrel{\text{def}}{=} X(p)$$

We use the same considerations as in the steps of the basic algorithm but now we have another function:

$$F = s^0 \cdot (\bar{r}(p),\bar{v}(p))^{\alpha^0} - s^0 \cdot \frac{(\bar{r}(p),\bar{v}(p))^{\alpha^0}{}_{min}}{(\bar{n}(p),\bar{t}(p))_{min}} \cdot (\bar{n}(p),\bar{t}(p))$$

After s^0, α^0 have been found $X(p)$ and $T^0(p)$ can be calculated.

Now we can use $T^0(p)$ as an initial approximation for iteration algorithm. The result at the first iteration is significantly better than with zero initial approximation (Figure 5).

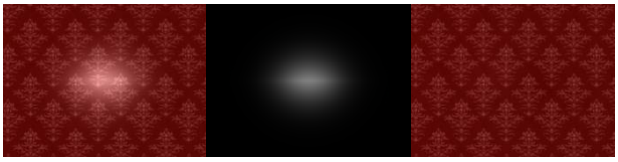


Figure 5: left- T_{min}, *middle- X, right -* T^0.

4.2 Determining of $\mathcal{M}$ Model

The input to the algorithm receives *N* images of a plane surface with a metallic and a synthesized image with the glare and texture without metallic. In this case, the synthesized image should be obtained under the same parameters as the original picture. The resulting algorithm images are synthesized based on the initial approximation to construct a model of Phong.

The whole process of the algorithm can be divided into 3 phases.

In the first phase, the input image is used to construct the image containing only metallic. The difference of input and synthesized images appears as such images.

In the second - from intermediate images a set of particles with different characteristics are determined, and also *Metallic* set. To construct a set of *Metallic* images obtained in the previous step, are adaptively binarized on the metallic and regular areas.

Analyzed images are converted into a format L*a*b. With the help of sequential scanning color images are searched for homogeneous areas. Each found region is considered as a particle. For particle main parameters are counted.

On the third - the parameters of inclusions are used to build model $\mathcal{M}$. To do this we construct histograms and compute probability characteristic for *Size*, *Shape* and *Color*.

To verify the determination of the metallic effect PSNR metric is not applicable, since the noise with the same parameters can modify the different pixels of the image. Consider example with two chessboards as a justification.

Figure 6: Chessboards.

This two pictures have PSNR = 1.7. However, from the observer's point of view, there is no structural difference between them. That is why another metric is proposed. It's named *MD* (*Metallic Difference)*. It has a single parameter R (window size). Its basic steps are following:

- Obtain high-frequency images
- Divided image into blocks of R * R pixels.
- Calculate a set of characteristics for each block;
- Summarize the difference between all blocks of the image;
- Normalize result value in bounds from 0 to 100.

The proposed metric has different from PSNR nature. The lower it is the more similar images are. On Figure 7 there are examples of the metallic reconstruction for metal plate with orange and blue metallic coating.

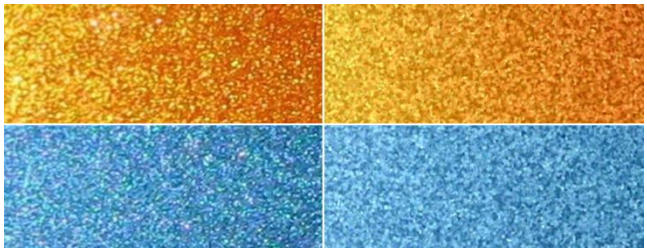


Figure 7: Left- photos of the metallic paint; right- synthesized images (*MD* = 4.05 and 5.40 correspondingly).

However, PSNR still can be used for comparison in case of no metallic presence.

5. EXPERIMENTAL RESULTS

Algorithms were verified on synthetic and real world materials. During tests on synthetic data we generally gains PSNR values between 48-60 dB (while comparing the metallic free initial and rendered images). Example of the input image and reconstructed material is shown on the Figure 9.
The results of the work of the algorithm on real data are illustrated in Figure 8.

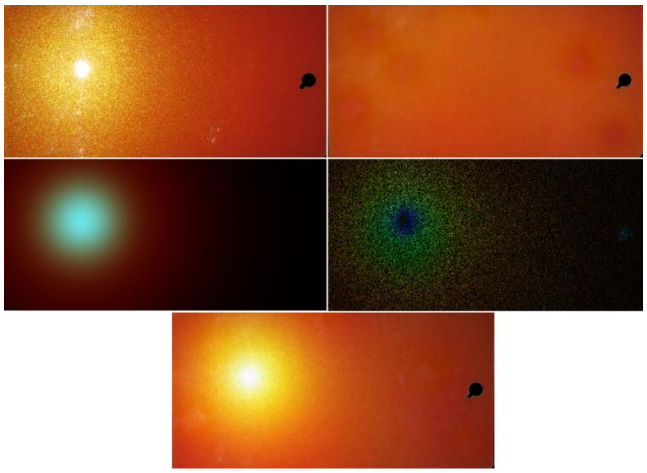


Figure 8: Left to right order**:** original image, generated texture, specular component, metallic component (is multiplied by 10), and the result image with the same light position (*PSNR=19.8, MD = 9.8).*

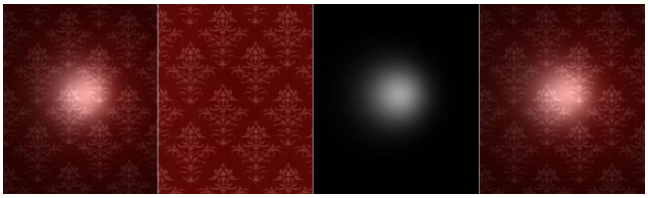


Figure 9: Original image, generated texture and specular component, synthesized image (PSNR=54dB).

6. CONCLUSION

As a result of work we proposed a model for representation of light-scattering properties for textured surfaces with metallic coating. Also we developed an algorithm for constructing the proposed model. The algorithm requires the input of a small number of photographs. However, it recovers with reasonable accuracy the texture and the corresponding parametric model lighting, that allows us to construct a simple probabilistic model of metallic and use it in the future for interactive visualization.

Nevertheless, calibration errors and irregularities of the surfaces are still affecting on the final results and measurement errors. A significant drawback is the requirement for pre-reduction of geometry. However, test results showed that despite the shortcomings, implemented approach has acceptable accuracy for realistic visualization of the constructed model.

7. REFERENCES

[1] Волобой А.Г., Галактионов В.А., Ершов С.В., Летунов А.А., Потемин И.С. Аппаратно-программный комплекс для измерения светорассеивающих свойств поверхностей *"Информационные технологии и вычислительные системы", № 4, 2006.*

[2] G. Müller, J. Meseth, M. Sattler, R. Sarlette and R. Klein. "Acquisition, Synthesis and Rendering of Bidirectional Texture Functions". *Eurographics 2004, STAR.* University of Bonn, Institute for Computer Science II, *2004.*

[3] P. E. Debevec, C. J. Taylor, and J. Malik. "Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach". In *Computer Graphics (SIGGRAPH '96 Proceedings)*, volume 30, pages 11-20, New Orleans, Lousiana, 1996.

[4] M. Callieri, P. Cignoni, and R. Scopigno. "*Reconstructing Textured Meshes from Multiple Range RGB Maps"*, 7th Int.l Fall Workshop on Vision, Modeling, and Visualization 2002, Erlangen (D), Nov. 20 - 22, 2002

[5] Bornik A., Karner K., Bauer J., Leberl F., Mayer H. "High-quality texture reconstruction from multiple views". *The Journal of Visualization and Computer Animation 2001*; 12: 263-276

[6] S.Ershov, K.Kolchin, K.Myszkowski "Rendering Pearlescent Appearance Based on Paint-Composition Modelling", *Eurographics 2001 conf. proc., vol.20 (2001),* number 3.

[7] Aydın Öztürk, Murat Kurt and Ahmet Bilgili. "Modeling BRDF by a Probability Distribution", Proc. of Graphicon'2010, pp. 57-63, Izmir, Turkey, 2010.

Real-time Animation, Collision and Rendering of Grassland

Sergey Belyaev, Igor Laevsky, Vyacheslav Chukanov
Department of Applied Mathematics
St.Petersburg State Polytechnic University, St.Petersburg , Russia
bel@d-inter.ru

Abstract

It is proposed an integrated solution for animation, interaction with dynamic objects and visualization of large grasslands. We use instancing for real-time visualization. For that, a new method for physics animation was developed which does not require saving generalized velocities and moves for each grass blade.

Keywords: Real-time, Animation, Collision, Rendering, Grass.

1. PREVIOUS WORKS

There are three methods for the grass visualization: geometry-based, image-based и volume-based. According to the first method, each grass blade is defined with a set of triangles. The set of the blades shapes a rectangle block. The block is visualized repeatedly in order to cover all the grass area. To speed up visualization, instancing technique is used. This geometry-based method was used in [3], [8] and [10] papers. The problem of this method is visualization of hundreds of millions of triangles. For cutting down this number, it was proposed in [3] to reduce the number of triangles in the blade if the latter is far from camera.

Image-based method distinguishes from the described above with only one feature: the set of blades in the block is replaced with the set of billboards with the texture containing alpha channel; a set of blades is drawn on it. This method was used in [2], [5], [6] and [9] papers. Volume-based method was applied in [7, 8]. Based on these approaches, it is difficult to get high quality of visualization close to the camera.

In order to increase visual realism, all mentioned methods are accompanied by the grass animation. In all cases it is used simplified model: the movements are proportional to the wind force and directed along its velocity vector.

Grass interaction with dynamic objects during the animation process was considered in [12] and [6]. In the first of these papers real interaction was substituted with use of a special texture – the footprint of the moving object. In the second one the object interacted with billboards that simulated the grass.

2. VISUALIZATION

2.1. Basic principles

We use geometry-based approach. According to it, similar to [8] rectangular blocks are created; each of them consists of a set of separate blades. These blocks are visualized on the terrain surface. A set of grass blocks makes up the rectangular grid which is mapped onto terrain such way that the central cell of the grid appears just under camera. When the camera moves on the adjacent grid cell the whole grid moves on the grid cell size.

The grass covering is visualized by multiple repetition of the single grass block according to instancing method with correcting position and shape of each grass blade in compliance with the terrain height and wind force in the given point.

Some part of the grass blocks is not included in instancing. These are the blocks in which interaction of grass blades with other dynamic objects is possible, as well as such interaction has occurred. For these blocks wind simulation calculations use numerical methods that require storing current positions and velocities of the grass blade tops, which excludes instancing. Let us denote these blocks "patches". They are visualized as objects containing a set of grass blades.

The grass block is a square containing N=n*n blades. In terms of DirectX10 each blade is a point containing four vertices and the normal determining rotation angle of the blade plane and the angle to the terrain surface, as well as geometry (length and height) and physics (mass and rigidity) characteristics. This data and the wind force vector are input data for the vertex shader to calculate the current location of the vertices and values of normals in them. Then, the geometry shader builds a cubic spline, that determines the blade triangles, number of which depends on the distance between the blade and the camera.

2.2. Levels of detail and smooth transition between them

We introduce three discrete levels of detail for the grass blocks, as well as possibility of smooth changing of detail within a level. To provide the possibility of smooth changing of detail, a weight coefficient is assigned to each blade in the block. When visualizing, the blade is discarded, if the following condition is valid:

$$w < F(z, \varphi)$$

where w – weight coefficient, F – some function, z – the distance from the camera to the blade, φ - the angle between direction to the camera and normal to the terrain surface in the blade point.

To define F function, note that the number of grass blades on a square unit must be proportional to the visible size of this square on the screen, i.e. proportional to (**n**,**r**) scalar product (where **n** – normal to the terrain surface and **r** – direction to the camera) and inverse proportional to the camera distance d:

$$\frac{(\mathbf{n}, \mathbf{r})}{a_1 + a_2 d + a_3 d^2}$$

In the denominator we take a square trinomial instead of camera distance d in order to avoid big numbers when camera comes near to the blade.

In addition we take into account that on contour areas where (**n**,**r**) scalar product value is close to zero, the grass density should be high. To do that, instead of the scalar product we use the following expression:

$$(1 - t)(\mathbf{n}, \mathbf{r}) + t$$

where

$$t = (1 - |(\mathbf{n}, \mathbf{r})|)^{\alpha}$$

Here α- a big number (we take it 8).

Finally, F function looks as

$$F(z,\varphi) = 1 - clamp\left(\frac{(1-t)(\mathbf{n},\mathbf{r}) + t}{a_1 + a_2 d + a_3 d^2}, 0, 1\right)$$

This function is used both for discarding grass blades and for selecting the block's discrete level of detail. In the first case d is the distance from the camera to the grass blade, in the second – to the block center.

2.3. Lighting calculation

For a single directional light source applied to a point of the surface of a blade, the radiance I is:

$$I = K_a I_a + K_d I_d (\max(\mathbf{N} * \mathbf{L}; 0) + C \max(-\mathbf{N} * \mathbf{L}; 0))$$

Where K_a is the material ambient color, K_d - the diffuse reflectance factor of the blade material, I_a - the intensity of the ambient light, $\mathbf{N}$ - the normal to the grass blade surface, $\mathbf{L}$ - the light direction, C – the factor to account for the color change of light traversing the grass blade fibers, I_d - the intensity of the light source.

Because the grass blades shade each other, intensities of diffused and direct sun light should be considered as functions of distances between the blade base and top. We use the following dependences:

$$I_a = I_a^{**} + l(I_a^{*} - I_a^{**})$$

$$I_d = I_d^{*} l^6$$

where I_a^{*}, I_d^{*} - intensities on the blade top, I_a^{**} - intensity on the base, l – normalized distance from the base to the top.

For Frenel effect simulation while calculating light for the blades on contour terrain areas we increase the calculated intensity value I by I_S value:

$$I_S = K_S l^6 (\mathbf{N}_L * \mathbf{V})^{10}$$

where K_S is the specular reflectance factor of the grass blade material, $\mathbf{N}_L$ - normal to the terrain surface and $\mathbf{V}$ - vector of direction to the camera.

3. ANIMATION

3.1. Inertial animation model

Let us represent the blade model with the chain of n linear segments, connected each to other with joints in which there are spherical springs (Fig. 1). We will denote the rigidity of these springs as k_i where i – number of the joint.

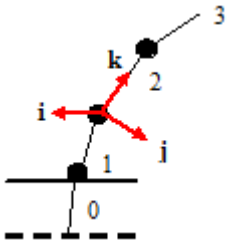


Fig. 1. The blade model for n = 4

The coordinate system is assigned to each segment as shown on Fig. 1. The segments and joints are enumerated bottom-up. Zero segment is dummy and determines initial rotation and tilt angles of the blade when planting. Ground level is on the height of lower end of the first segment (joint 1).

Let the following external forces $\mathbf{f}_i^{\mathbf{e}}$ are applied to the segment centers – the forces, which are the sum of the wind force and the segment gravity (Fig. 2).

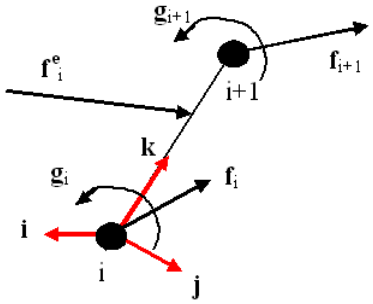


Fig. 2 Forces and moments applied to the segment

Let us write movement equation for i-segment in its coordinate system:

$$\mathbf{J}\dot{\boldsymbol{\omega}}_i = -\boldsymbol{\omega}_i \times (\mathbf{J}\boldsymbol{\omega}_i) - m\mathbf{l} \times \mathbf{R}_i' \mathbf{a}_{i-1} -$$

$$\mathbf{g}_i + \mathbf{R}_{i+1}\mathbf{g}_{i+1} + \mathbf{l} \times (2\mathbf{R}_{i+1}\mathbf{f}_{i+1} + \mathbf{T}_i' \mathbf{f}_i^{\mathbf{e}})$$

$$\mathbf{a}_i^{*} = \dot{\boldsymbol{\omega}}_i \times \mathbf{l} + \boldsymbol{\omega}_i \times (\boldsymbol{\omega}_i \times \mathbf{l})$$

$$\mathbf{a}_i = \mathbf{a}_{i-1} + \mathbf{a}_i^{*}$$

$$\dot{\boldsymbol{\psi}}_i = \boldsymbol{\omega}_i$$

$$\mathbf{f}_i^{*} = -m(\mathbf{R}_i' \mathbf{a}_{i-1} + \mathbf{a}_i^{*})$$

$$\mathbf{f}_i = \mathbf{f}_i^{*} + \mathbf{T}_i' \mathbf{f}_i^{\mathbf{e}} + \mathbf{R}_{i+1}\mathbf{f}_{i+1}$$

Here: $\mathbf{J}$– inertia tensor, $\boldsymbol{\omega}_i$ - vector of angle velocity of i-segment, $\boldsymbol{\psi}_i$ – vector determining rotation increment of the coordinate system of i-segment, relatively to the coordinate system of (i – 1) segment , $\mathbf{g}_i$ - the moment because of spring in i-joint, $\mathbf{R}_i$ – matrix converting vectors from the coordinate system of i-segment to the coordinate system of (i-1)-segment (when i=0 – to the world coordinate system), $\mathbf{T}_i'$ - matrix, converting vectors from the world coordinate system to the coordinate system of i-segment, $\mathbf{a}_i$ - acceleration at the end of i-segment, $\mathbf{l}$= (0,0,l)', where l – a half of the segment length, m – mass of the segment.

For integration of the system we can be used Featherstone algorithm [11]. However, the computational complexity of this integration is too expensive to calculate the animation of several thousand blades of grass in real time.

We can simplify this system, if assume that: angle velocities are small and impact of higher segments on lower is much less, than reverse. The first assumption allows us to discard members containing squares of angular velocities, and the second – to

refuse of the second pass in Featherstone algorithm. As a result, we come to the following simplified algorithm:

$\mathbf{T_0}=\mathbf{R_0}$
for (i=1; i<n; i++) {

$$\mathbf{J}\dot{\boldsymbol{\omega}}_i = -\mathbf{g}_i + \mathbf{l}\times\mathbf{T}_i'\mathbf{f}_i^e$$
$$\dot{\boldsymbol{\psi}}_i = \boldsymbol{\omega}_i$$
$$\mathbf{R}_i = \mathbf{R}_i\mathbf{M}(\boldsymbol{\psi}_i)$$
$$\mathbf{T}_i = \mathbf{T}_{i-1}\mathbf{R}_i$$
$$\mathbf{g}_i = k_i V(\mathbf{R}_i)$$

}

where $\mathbf{M}(\boldsymbol{\psi})$ – matrix of rotation around the vector $\boldsymbol{\psi}$ the value $|\boldsymbol{\psi}|$, V - inverse transform to get rotation vector.

As our experiments showed, this algorithm keeps good visual illusion of animated grass blades.

3.2. Animation of wind force and direction

Similarly to [1] we use the cyclic Perlin noise texture. Wind animation is done by summing up (with various scales c_i) three such textures moving with $\mathbf{w}_i$ velocities (i=1, 2, 3). Resulting wind speed vector $\mathbf{W}$ in the texel with **u** coordinates is calculated with the formula:

$$\mathbf{W[u]} = \sum_{i=1}^{3} k_i \mathbf{T}[\mathbf{w}_i t + c_i \mathbf{R}_i \mathbf{u}]$$

where k_i - scale coefficients, t – time, $\mathbf{R}_i$- rotation matrices defined by $\mathbf{w}_i$ vectors.

While calculating wind force for each blade segment, we consider velocity of the segment:

$$\mathbf{f}_i^W = k_w(\mathbf{W[u]} - \mathbf{v}_i)^\gamma$$

Here k_w - coefficient depending on the blade width, $\mathbf{v}_i$- velocity of the segment center, γ - a constant depending on the grass type (for example, γ =4/3 for sedge). $\mathbf{v}_i$ value is calculated depending on $\overset{*}{\mathbf{v}}_{i-1}$ (velocity of the top of previous segment) according to formula:

$$\mathbf{v}_i = \overset{*}{\mathbf{v}}_{i-1} - \mathbf{T}_i(\mathbf{l}\times\boldsymbol{\omega}_i)$$

Velocities of the segment tops are found from recurrent relations:

$$\overset{*}{\mathbf{v}}_0 = 0$$
$$\overset{*}{\mathbf{v}}_i = \overset{*}{\mathbf{v}}_{i-1} - 2\mathbf{T}_i(\mathbf{l}\times\boldsymbol{\omega}_i)$$

3.3. Virtually-inertial animation model

This model provides results close to that for inertial model, but doesn't require storing current values of angle velocities and general displacements for each grass blade, which allows us to use instancing when rendering.

The idea of the virtually-inertial model is in carrying over inertial component from calculation of the blade shape to calculation of the wind force for this blade. That may be done if we put into centers of wind force texels vertical virtual blades and calculate their shape with inertial model. Afterwards we calculate the moments that must be applied to blades segments in order to get the same shape of static equilibrium:

$$\mathbf{m}_i^W = k_i\boldsymbol{\psi}_i - \mathbf{l}\times\mathbf{T}_i'\mathbf{G}$$

where **G** – gravity (here, as well as in the previous paragraph do not consider the effect of the upper segment to lower). These moments are stored in the virtual wind texture that is used for the grass blade animation when rendering with instancing, instead of the actual wind forces.

Note that the blades covered by one texel of virtual wind texture should be animated differently in spite of they are affected by the same virtual wind. This is because they have various angles when planting, so weight force impact is diverse.

To do so, we calculate the shape of a blade of grass so that the condition of static equilibrium under the action of the virtual wind and gravity, taking into account the slope of grass with seating. The equation of static equilibrium for the i-th segment is as follows:

$$k_i\boldsymbol{\psi}_i = \mathbf{m}_i^W + \mathbf{l}\times(\mathbf{T}_{i-1}\mathbf{M}(\boldsymbol{\psi}_i))'\mathbf{G}$$

With known matrix $\mathbf{T}_{i-1}$ this equation can be solved by simple iteration. As our experiments showed, three iterations are enough for coinciding visual results.

Thus, we arrive at the following algorithm for determining the shape of a blade of grass:

$\mathbf{T} = \mathbf{T}_0$
for (i=1; i<n; i++){

$$k_i\boldsymbol{\psi} = \mathbf{m}_i^W + \mathbf{l}\times(\mathbf{TM}(\boldsymbol{\psi}))'\mathbf{G}$$
$$\mathbf{T}_i = \mathbf{TM}(\boldsymbol{\psi})$$
$$\mathbf{T} = \mathbf{T}_i$$

}

Here, the matrix $\mathbf{T}_0$ defined angle of seating.

3.4. Animation algorithm considering interaction with other dynamic objects

For simulation of grass interaction with dynamic objects, as mentioned in section 2.1, patches instead of blocks are used. Blade data structures in patches contain fields for current generalized velocities and moves, which allows us to use an inertial model. Nevertheless, we continue to use virtual-inertial model until the blade interacts with an object. At this moment the blade segments are bent so that exclude intersections with the object and its generalized velocities are set to zero. Later such blade is calculated with inertial model.

This approach allows us to use an expensive inertial model only for relatively small number (around a thousand) of grass blades.

Note that use of different models for close located blades doesn't lead to visual artifacts due to proximity of these models.

4. RESULTS

Visual results of our program are presented on Fig. 3, 4 screen shots. Fig. 3 shows grass animation with a wind and Fig. 4 shows the result of grass interaction with a dynamic object without and with the wind. Video material can be can be found here:
http://dl.dropbox.com/u/28177387/GrassCar.avi
http://dl.dropbox.com/u/28177387/GrassPlain.avi

Fig. 3. Grass animation with a wind

Fig. 4. Grass interaction with a dynamic object without and with the wind.

The grass field covers the square with the side of 280 meters. The number of grass blades in this square is 10^6. The program performance for three PC configurations is given in the Table 1. The column A contains results when both animation and interaction with dynamic objects are absent. For B column there is animation, but no interaction. At last, C column shows results with both animation and interaction.

It is evident from the Table 1 that about 90% program time is spent for visualization and only 10% - for processing interaction with dynamic object and calculating blade shapes under wind. That proves high efficiency of the developed algorithms and possibility of their use in real-time programs.

PC configuration	A	B	C
	FPS	FPS	FPS
Intel Pentium D CPU 3GHz ATI ASUS EAH5450	28	26	24
Intel Core 2 Duo e8500 Nvidia GeForce 9600 GT	60	57	54
Intel Core 2 Duo e8500 ATI Radeon HD 6870	180	165	150

Table. 1. The program performance

5. ACKNOWLEDGEMENTS

The work was funded by Intel A/O (Agreement #NN/R&D/66/2010).

6. REFERENCES

1. Alexandre Meyer, Fabrice Neyret. “Interactive Volumetric Textures”. Eurographics Rendering Workshop, 1998.
2. Anu Kalra. “Rendering Grass with Instancing in DirectX* 10“, http://isdlibrary.intel-dispatch.com/vc/2325/rendering_grass_32509.pdf, 2009.
3. By Changbo Wang, Zhangye Wang, Qi Zhou, Chengfang Song, Yu Guan and Qunsheng Peng. “Dynamic modeling and rendering of grass wagging in wind”. Comp. Anim. Virtual Worlds, pp.377–389, 2005
4. “Cloth Simulation”. NVIDIA White Paper, sdkfeedback@nvidia.com, 2007
5. David Whtley. “Toward photorealism in virtul botany”. GPU Gems 2, pp. 7-25, 2005
6. J. Orthmann, C. Rezk-Salama, A. Kolb. “GPU-based Responsive Grass”. In *Journal of WSCG, 17, pages 65-72, 2009.*
7. Ralf Habel, Michael Wimmer, Stefan Jeschke. “Instant Animated Grass”. http://www.cg.tuwien.ac.at/research/publications/2007/Habel_2007_IAG/Habel_2007_IAG-Preprint.pdf, 2009.
8. Kevin Boulanger, Sumanta Pattanaik, Kadi Bouatouch. “Rendering Grass in Real Time with Dynamic Lighting”. IEEE Computer Graphics & Applications, vol. 29(1), pp. 32-41, 2009.
9. Kurt Pelzer, Piranha Bytes. “Rendering Countless Blades of Waving Grass”. GPU Gems. Chapter 7, 2004.
10. Perbet F, Cani M-P. “Animating prairies in real-time”. In Proc. the symposium on Interactive 3D graphics’, pp. 103-110, 2001.
11. Roy Featherstone, D. Orin “Robot dynamics: equations and algorithms”. Proceedings 2000 ICRA Millennium Conference IEEE International Conference on Robotics and Automation Symposia Proceedings Cat No00CH37065 (2000)
12. Sylvain Guerraz, Frank Perbet, David Raulo, Franc¸ois Faure, Marie-Paule Can. “A Procedural Approach to Animate Interactive Natural Sceneries”. Computer Animation and Social Agents (CASA), 2003.

Resolution Independent NURBS Curves Rendering using Programmable Graphics Pipeline

Rami Santina
CCT International
rsantina@cctintl.com

Abstract

Non-Uniform Rational B-Splines (NURBS) are widely used, especially in the design and manufacturing industry, for their precision and ability to represent complex shapes. These properties come at the cost of being computationally expensive for rendering. Many methods have tackled NURBS rendering by view based approximations and/or heavy preprocessing. We present a method for resolution independent rendering of curves and shapes, defined by NURBS, by utilizing the high parallelism of the programmable graphics hardware. The computation of the curve is processed directly on the GPU, without the need for complex preprocessing and/or additional storage of the basis functions as textures. Our method enables rendering of a complex NURBS shape in precise form, by defining only the curve's hull. We also present a method to enhance the performance of the preprocessing stage, mainly triangulation, that fits our requirements and speeds up the process. With optimized preprocessing and using only the mobile profile of the programmable graphics pipeline, we achieve a fast and resolution independent method for rendering NURBS based 2D shapes on desktop and mobile devices.

Keywords: *NURBS, Curve Rendering, Resolution Independence, GPU Algorithm, Mobile Graphics.*

1. INTRODUCTION

Resolution independent rendering is becoming a standard requirement for visualizing shapes. Earlier methods presented a resolution independent rendering for Bezier curves. The Bezier form, by definition, limits the curve's shape to the position of the control vertices. Thus, editing the control vertices is the only degree of freedom (DoF) for defining the curve. Such form of editing requires a regeneration of the shape and/or additional control vertices. In addition, designing some shapes using Bezier requires more control vertices and sometimes a higher order definition; this can be reduced by using a more general family of curves. Non-Uniform Rational B-Splines (NURBS) are widely used for a precise design of complex shapes using fewer control vertices, especially in CAD/CAM based applications. The evaluation of a NURBS curve is quite expensive requiring a recursive computation of the basis functions. Many tools transform, as a preprocessing step, the curves to a more simplified form to speed up the rendering.

In this paper, we present a method for Resolution Independent rendering of 2D shapes, defined partially or totally by NURBS curves, using the graphics hardware. Our method defines a simple and memory efficient approach to render 2D NURBS shapes by utilizing the high parallelism of the programmable graphics hardware. The NURBS curve is evaluated, using an implicit function, during rasterization.

The rest of this paper is organized as follows: Related Works are discussed in Section 2. . In Section 3., we present a general overview of our method describing the preprocessing stage and the GPU based NURBS curves rendering algorithm. In Section 4., we enhance the rendering technique to anti-alias the resulting curve. In Section 5., we discuss anti-aliasing for the whole shape to ensure generating a smooth and continuous representation.

2. RELATED WORKS

In [1], the authors presented a method for rendering NURBS curves on the GPU using textures, generated in a preprocessing phase, to store the values of the basis functions depending on the curve's order. This technique produces good results, but requires a computationally expensive preprocessing step and additional memory to store the textures.

To represent our NURBS shapes on the graphics hardware, we depend on the curve's implicit form [2]. In [3] and [4], implicit curve rendering has been used for representing curves and surfaces based on distance approximations. In [5], the authors presented a method for embedding sharp linear features into images to obtain resolution independence while leveraging GPU pixel processing. In [6], curved elements were embedded into texture images at the texel level. In these methods a computationally expensive preprocessing, performed on the CPU, was required.

In [7], the authors presented a method for rendering 2D regions based on quadratic Bezier curves. They defined each curve by a triangle, formed by the curve's control hull. Each of these triangles is rasterized using an implicit equation which defines the Bezier Curve. The mosaic formed by the set of such triangles along with the non-curved triangles gives the form of the final 2D shape. They then extended this process to handle cubic Bezier curves defined by neighboring triangles, depending on a classification of the curve. For preprocessing, the authors rely on constrained delaunay triangulation for generating the non-curved triangles; which is computationally expensive, in its general form. For anti-aliasing of the whole shape, they primarily depend on hardware accelerated Multi-Sampling. This technique produces good results but does not provide high quality anti-aliasing for tiny and skinny shapes, like small text, since the shape as a whole will be blurred.

One of the main applications for [7], is resolution independent font rendering. Earlier methods for font rendering are based mainly on generating a texture processed on the CPU [8], which produces crisp and sharp small text, but are resolution dependent. Our algorithm supports font rendering since the bezier curve is a special case of NURBS. In section 5. we will present a technique which produces high quality text, as an optional extension to our method.

3. ALGORITHM OVERVIEW

The input to our algorithm is a set of outlines which represents the shape's boundaries. Each of the outlines consists of a set of connected vertices. The vertices are of two types, namely: off-curve and on-curve (interpolated by the curve). One or more off-curve vertex defines a curve with endpoints being the closest neighboring on-curve vertices from the same outline. Examples of similar outline based definition are the 2D CAD drawings, scalar vector data, and fonts data (figure 1). In addition, each off-curve vertex has a weight, which defines the influence of the vertex on the curve's final shape. An off-curve vertex with no predefined weight is assigned a weight $w = 1$.

Figure 1: Snapshots of Rendered text, using the NURBS rendering described along this paper, with and without the discarded pixels of the curved triangles showing method compatibility with TTF rendering. **Left:** the glyph of character S from a TTF based font data. **Right:** A conic shape

We convert all the curved regions of an outline to a set of triplets, defined by two on-curve vertices surrounding an off-curve vertex. This representation enables us to map the curved regions to a set of *curved* triangles. For the **quadratic case**, the conversion is simply performed by representing each curve segment by the 3 vertices that influence the curve shape. As for the **cubic case**, we subdivide the curve to the quadratic form [9]. Generalizing the equations to cubic form is doable by formulating the equations in the same steps described along this paper, but will require a preprocessed classification of the curve's shape and a more computationally expensive rendering. The transformation to quadratic is a mathematical approximation. Since the rendered curve is accurate only to the pixel level, this approximation doesn't effect any details of the shape

With all the curved regions defined as a set of triangles, we map the rendering of the curves to texture space; assigning the curve control points as texture coordinates to the curve vertices, this step is detailed in Section 3.2. As for the non curved parts we perform a modified Delaunay triangulation (Section 3.1). Finally, the set of triangles generated by the triangulation along with the curved triangles form the final shape of the 2D object. An example output is shown in figure 2.

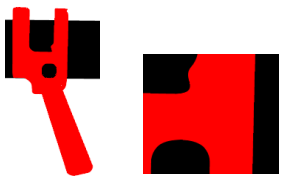


Figure 2: A CAD handle rendered using our method, with different weights. **Right:** Zoom in on part of the shape.

The produced shapes will have the following properties:

1. viewpoint independence, since the curves are computed on the GPU, using an implicit function, during the rasterization phase.

2. minimized memory usage, since we are not using subdivision to approximate the curved parts and we are not using any textures in the algorithm.

3. high performance rendering, since preprocessing (mainly triangulation) is computed once at input definition and rendering is processed directly on the GPU.

4. mobile compatible, since our method only uses features included in the mobile profile (OpenGL ES2)

3.1 Preprocessing

In this section, we present a modified version of constrained delaunay triangulation [10] that fits our requirements, removing the need for any cleanup stages. Note that, this is not a requirement for the algorithm described in this paper; but is a major block in the preprocessing phase which affects the overall performance and memory usage. As stated earlier, the curved regions are transformed into curved triangles which excludes them from the general triangulation step and thus from the algorithm described in what follows.

First, we transform each of the input outlines to a set of connected half edges forming a loop λ. Then, we add each λ, sequentially, to the final set of loops Λ. While adding, we check if this loop is intersecting, constrain, or is constrained by any of the already added loops in Λ, using a simple ray tracing (in/out) test. If the loop constrain or is constrained, we combine the two loops at the closest pair of vertices by adding two sibling half-edges between them. Hence, we get a simplification of the problem since now we have only one well connected loop. If the loop intersects, we split at the intersecting positions, which may result in a maximum of two new vertices. Then, we constrain the first by the contained part of the second and combine the two open ended parts forming another closed loop.

At the completion of the above steps, Λ contains a set of independent closed loops that define the resulting shape. Hence, we can now triangulate each loop separately without any additional cleanup steps.

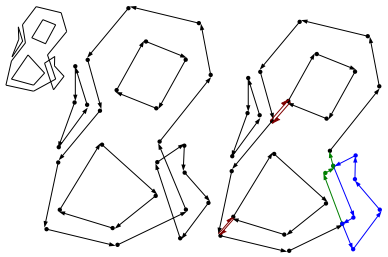


Figure 3: **Top Left:** Input shape defined as a set of connected vertices forming five outlines. **Middle:** Outlines converted to half edges. **Right:** The constraint outlines are merged with the respective outline (red half-edges), and the intersecting outlines are split editing the second large outline (green half-edges) and forming the third outline (blue half-edges)

In figure 3, the five input outlines are transformed into 3 well connected independent outlines. This step simplifies the

triangulation process of the shape, to be a triangulation of independent set of loops, where a cleaning phase is no more required to define the holes. Note that we can not guarantee that a complete delaunay triangulation can be achieved, but we can maximize it using a greedy approach.

3.2 Quadratic NURBS Rendering

Since all the curved parts of the region have been transformed into quadratic NURBS, the following is applied to all the curved triangles.

The general form of a NURBS curve is given by,

$$C(x) = \frac{\sum_{i=0}^{n} N_{i,D}(x) w_i P_i}{\sum_{j=0}^{n} N_{j,D}(x) w_j} \tag{1}$$

where

$$N_{i,1}(x) = \begin{cases} 1 & \text{if } t_i \leq x < t_{i+1} \\ 0 & \text{otherwise.} \end{cases}$$

and

$$N_{i,D}(x) = \frac{(x - t_i) N_{i,d-1}(x)}{t_{i+d-1} - t_i} + \frac{(t_{i+d} - x) N_{i+1,d-1}(x)}{t_{i+d} - t_{i+1}}$$

where t_i corresponds to the knot at location i in the Knot Vector.

Equation 1, gives the definition of a NURBS curve C as a function of the parameter x, where P_i are the control points and $N_{i,D}(u)$ are the basis functions of degree D. w_i are the weights of each control point. The special case of $\frac{0}{0}$ that may arise in one of the basis functions, is taken to be 0.

We first map the curve definition to texture space; by assigning the control points of our quadratic NURBS curve as attributes to the vertices. Hence, the control points p_0, p_1, and p_2 are assigned to the vertices by the set [u v w], where [u v] are the texture coordinates and w is the weight. We set $p_0 = [0\ 0\ w_0]$, $p_1 = [\frac{1}{2}\ \frac{1}{2}\ w_1]$ and $p_2 = [1\ 0\ w_2]$, where p_1 is assigned to the off-curve vertex. During rasterization, the GPU will calculate a texture coordinate for each pixel on the interior of the triangle by interpolating the defined texture coordinates.

Since u belongs to [0 1] and the curve is defined in the domain [0 1], we get the following property: for each value of u generated by the interpolation there exists a value v which is on the curve. In the fragment shader, we determine the fragment position w.r.t. the curve by evaluating the implicit function of the curve [2], which can be derived as:

$$f = v - \frac{w_1 u(1-u)}{(w_0 - 2w_1 + w_2)u^2 + 2(w_1 - w_0)u + w_0} \tag{2}$$

If $f < 0$, then the fragment belongs to the region below the curve (**in**). Otherwise, it belongs to the region above the curve (**out**). With this function we can choose which part of the triangle we which to render, above the curve or below it. Note that, equation 2 is the implicit form of equation 1, where $D = 3$ and applying triple Knot insertion. An illustration of this process is provided in figure 4. In figure 5, a sample output with different weight values is shown. Also we see the basic difference with [7] where the latter can only render the triangle on the far left.

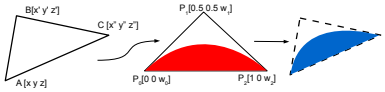


Figure 4: **Left:** Triangle in world coordinates. **Middle:** corresponding mapping in texture space. **Right:** Final Image in screen space.

Figure 5: A triangle rendered with decreasing weight values at the off-curve vertex. Left to Right: w_1=1 $\rightarrow$ 0. In comparison, [7] can only render the left-most shape for the given triangle

4. CURVED REGIONS RENDERING

The in-out function of equation 2 does not provide a smoothly curved boundary of the shape, due to aliasing artifacts. In this section, we enhance equation 2, to provide a smooth interpolation between the in and out parts. We enhance this equation, to handle change factor in the (x, y) directions by computing $\bigtriangledown g(x, y)$ according to the chain rule:

$$\bigtriangledown g = \begin{bmatrix} g_y^x - \frac{w_1((w_0 - w_2)u^2 - 2w_0 u + w_0) g_x^x}{(\alpha u^2 + 2\beta u + w_0)^2} \\ g_y^y - \frac{w_1((w_0 - w_2)u^2 - 2w_0 u + w_0) g_x^y}{(\alpha u^2 + 2\beta u + w_0)^2} \end{bmatrix} \tag{3}$$

where

$$\alpha = w_0 - 2w_1 + w_2\ ,\ \beta = w_1 - w_0$$

and g_b^a denotes $\frac{dt}{da}(b)$ the values of the partial derivative of t w.r.t. a in the b direction, and $t = (u, |v|)$ is the absolute value of the texture coordinates at the current location. Absolute values of the texture coordinates are used since we negate v to define that the **out** region is required instead of the **in** region.

The fragment shader of the programmable pipeline supports the computation of functions of the form g_b^a, by local differencing, since GLSL version 1.x. Having the gradient approximation, we compute $e(u, v)$ which resembles the signed distance from the current pixel to the curve.

$$e(u, v) = \frac{1}{2} - sign(v) \frac{f}{||\bigtriangledown g||} \tag{4}$$

The *sign* function is used to provide the ability to render any of the two regions within the triangle (in or out). Hence to render the out region of the triangle, we negate the sign of the v texture coordinate of p_1. Using the value of $e(u, v)$, we classify the fragment according to the equation:

$$class(u, v) = \begin{cases} in & e(u, v) \geq 1 \\ out & e(u, v) \leq 0 \\ boundary & \text{otherwise.} \end{cases} \tag{5}$$

In Equation 5, we get a classification of three cases in comparison to the previous two, provided by equation 2. If class(u,v) is determined as **in**, we render the fragment with

full color/shade/texture. If the classification is **out**, we discard the fragment or render it with the back color/texture, depending on the rendering technique used. In the **boundary** case, we do a linear interpolation between the back color c_b and the shading color c_f, as defined in equation 6.

$$color = (1 - e(u, v))c_b + e(u, v)c_f \quad (6)$$

Other rendering techniques might need to define only the alpha channel, in that case we use the value of $e(u, v)$ clamped to the region [0 1]. An example of this process is shown in figure 1 where discarded pixels are rendered in red. Figure 6 shows the smoothness provided by the extended approach described above.

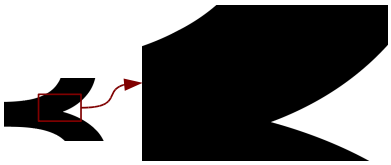


Figure 6: **Left:** A NURBS shape, rendered using our method. **Right:** Zoom in to the area marked in red, showing the smoothness

5. ANTI-ALIASING

In what follows, we will present a View Based Anti Aliasing (VBAA) technique for dealing with skinny small shapes, such as tiny text and condensed P&ID cad drawings. This step is optional since a one pass rendering will provide good results, but to get a crisp overall image an additional rendering pass is required. First, we compute the size d using the following equation:

$$d = r \cdot (M_p \cdot B_w) \quad (7)$$

where r is the radial region of fragments that will affect the final fragment color, M_p is the projection matrix from world coordinates to screen coordinates, and B_w is the bounding box of the shape. We then render the shape into a texture of size d. In the second pass we attach the generated texture to B_w and apply a Gaussian based filter of radius r to produce the final rendered image. We assign horizontal and vertical texels a higher weighting average than diagonal texels. This is to provide the sharp and crisp features of the final shape. A comparison of the output produced by VBAA w.r.t. MSAA is shown in figure 7.

Figure 7: A comparison between rendering using VBAA and hardware MSAA at different sizes. **Top:** A string rendered at two different sizes using the View Based AA. **Bottom:** Same string rendered with 4x MSAA.

6. ACKNOWLEDGMENT

The author is grateful for all the reviewers of this work regarding the algorithm and its application. A special thanks to Sven Gothel and the JogAmp open-source community for the valuable discussions held regarding this work.

7. CONCLUSION

This paper presents a method for a fast and resolution independent rendering of NURBS curves and shapes, without the need for heavy preprocessing. With our method, a user is able to render NURBS shapes in high performance, low memory usage, and high quality anti-aliasing around the curve. We have also shown how our method can be used to render lower order curves (Bezier) such as fonts by setting the weights to one. Finally, since our method does not require heavy computations, it can be used to visualize NURBS shapes on mobile devices. The complete source code of the algorithms presented in this paper is published in JOGL open-source project, part of the JogAmp Community.

8. REFERENCES

[1] Adarsh Krishnamurthy, Rahul Khardekar, and Sara McMains, "Direct evaluation of nurbs curves and surfaces on the gpu," in *Proceedings of the 2007 ACM symposium on Solid and physical modeling*, New York, NY, USA, 2007, SPM '07, pp. 329–334, ACM.

[2] Thomas Warren Sederberg, *Implicit and parametric curves and surfaces for computer aided geometric design*, Ph.D. thesis, West Lafayette, IN, USA, 1983, AAI8400421.

[3] H. Pottmann, S. Leopoldseder, M. Hofer, T. Steiner, and W. Wang, "Industrial geometry: recent advances and applications in cad," *Comput. Aided Des.*, vol. 37, pp. 751–766, June 2005.

[4] Gabriel Taubin, "Distance approximations for rasterizing implicit curves," *ACM Trans. Graph.*, vol. 13, pp. 3–42, January 1994.

[5] Jack Tumblin and Prasun Choudhury, "Bixels: Picture samples with sharp embedded boundaries," in *Rendering Techniques*, Alexander Keller and Henrik Wann Jensen, Eds. 2004, pp. 255–264, Eurographics Association.

[6] Ganesh Ramanarayanan, Kavita Bala, and Bruce Walter, "Feature-based textures," in *Rendering Techniques*, Alexander Keller and Henrik Wann Jensen, Eds. 2004, pp. 265–274, Eurographics Association.

[7] Charles Loop and James Blinn, "Resolution independent curve rendering using programmable graphics hardware," *ACM Trans. Graph.*, vol. 24, no. 3, pp. 1000–1009, 2005.

[8] Sampo Kaasila, "Method and apparatus for moving control points in displaying digital typeface on raster output devices," in *US Patent 5155805*. 1992, Apple Computer, Inc.

[9] Les Piegl and Wayne Tiller, *The NURBS book*, Springer-Verlag, London, UK, 1995.

[10] L. P. Chew, "Constrained delaunay triangulations," in *Proceedings of the third annual symposium on Computational geometry*, New York, NY, USA, 1987, SCG '87, pp. 215–222, ACM.

Real-Time SAH BVH Construction for Ray Tracing Dynamic Scenes

Dmitry Sopin[1], Denis Bogolepov[1], Danila Ulyanov[2]
[1]N.I. Lobachevsky State University of Nizhni Novgorod
[2]R.Y. Alekseev State Technical University of Nizhny Novgorod
sopindm@gmail.com, denisbogol@gmail.com, danila-ulyanov@ya.ru

Abstract

This work is aimed at the development of effective algorithms for building of full SAH BVH trees on GPU in real-time. In this work it is presupposed that all the scene objects are represented by a number of triangles (the so-called "triangle soup"), at the same time the arbitrary changes in the geometry are allowed in the process of rendering. The proposed methods have allowed more than tenfold increase performance compared to the best GPU implementation known.

Keywords: *Ray Tracing, Acceleration Structures, BVH, SAH, Real-Time, Dynamic Scenes, GPGPU, OpenCL.*

1. INTRODUCTION

The ray tracing algorithm was traditionally used in computer graphics for image synthesis of high quality. For getting the results a large amount of computation is needed. That is why for a long time using of the method in interactive applications and real-time systems seemed impractical. The main difficulties were related to two phases of work: building of acceleration structures and visualization based on ray tracing. For the effective solution of the highlighted tasks the programmable graphics accelerators can be used, which have turned into high-performance general-purpose processors over the past few years.

The first tries to implement the ray tracing on the GPU allowed the processing of the scenes with *static* geometry only [1]–[3]. This restriction allowed to build accelerating structure at the stage of pre-processing of the scene and use it on the GPU to make the rendering faster. The extensive research has shown that on standard consumer hardware ray tracing in real-time is possible for all major acceleration structures including uniform and hierarchical grids, kd-trees and bounding volume hierarchies (BVH). The technology has already found application in many fields of tasks but still has been of little use in applications with *dynamic* geometry, such as computer games, simulators and virtual reality systems.

The considerable attention of contemporary research is associated not only with the ray tracing methods but also with algorithms allowing the quick building of effective acceleration structures which would provide the support for dynamic geometry. In this case the formulation of the problem is changing radically because now it is necessary to take into consideration not only the efficiency of data structures at the stage of rendering but also to take into account the time needed for its construction which often appears the main limit for the working speed. The recent works have shown that the supporting for dynamic scenes with using of all the main accelerating structures can be possible [4]. However, in some cases the animation is put under some restrictions, in particular, only the *hierarchical* movements of the primitives or *deformable* scenes are allowed.

In this work for faster rendering BVH trees were chosen because of their advantages. First, this structure provides the most "dense" approximation of the geometry of the scene with a minimum number of nodes – it requires a minimum number of steps in the construction and traversing of the tree. Secondly, in the process of tree building only the centroids of triangles are used, that is why the situation with intersection of the split plane by a triangle is excluded – the primitive always belongs to the one descendant only. Another useful property of BVH trees is in the opportunity for their *updating* instead of a full *rebuild* which is used in several works where the animation is possible only with few limitations. Nevertheless, the processing of the scenes with arbitrary animation is possible only due to the presence of algorithms for fast building of BVH from scratch.

The high performance of ray tracing directly depends on the quality of the tree the best criterion for which serves the *surface area heuristic* (SAH). This heuristic was first proposed in [5] and is defined as follows: at each step of the recursive construction of the tree in the process of splitting of the set of triangles into two parts L and R the cost of that splitting is being computed:

$$SAH(T \to (L,R)) = K_T + K_I \left[\frac{SA(L)}{SA(T)} N_L + \frac{SA(R)}{SA(T)} N_R \right]$$

Here, $SA(X)$ stands for the area of the bounding volume of node X, N_X stands for the number of triangles in X, when K_I and K_T stand for implementation options that determine the cost of the test for intersection and traversal of the tree. The high effect of rendering is achieved through minimizing of the cost of splittings in the process of construction of data structures.

The first attempt to interactively build the SAH BVH trees was made in work [6], in which the author has adapted the *binned* technique that was originally proposed for *kd*-trees, and effectively implemented it on the multi-core CPUs. Subsequently, this implementation was further developed for the Intel MIC architecture, where the best timing estimates were obtained [8]. Recent work [7] has shown that the construction of SAH BVH tree on the GPU can be also possible, but the specified timing estimates turned out to be even higher than the similar estimates for older CPU [6], despite the use of more powerful architecture. Thus, at the present time there are no effective methods for construction of SAH BVH trees on the GPU, which would provide the opportunity for rendering of arbitrary dynamic scenes.

2. PROBLEM STATEMENT

This work is aimed at development of effective algorithms for building of full SAH BVH trees on GPU in *real-time*. In this work it is presupposed that all the scene objects are represented by a number of triangles (the so-called "*triangle soup*"), at the same time the arbitrary changes in the geometry are allowed in the process of rendering.

From a conceptual point of view the algorithm of the tree building is analogous to the algorithms which are described in the works [6]–[8]. Still in contrast with the works mentioned we don't use the simple heuristics similar to "Median Splits" and "LBVH" because they lead to decline in the quality of the generated data structure and consequently to the worse rendering performance. This work proposes a number of methods for the effective mapping of the general algorithm to the architecture of modern GPUs what allowed to accelerate the tree building up to 10 times compared to the best known GPU implementation [7].

3. BUILDING OF THE SAH BVH TREE

3.1 The Basic Algorithm

The process of the tree construction is represented in the form of the set of tasks the subsequent performing of which is realized through the concept of the task queue. The algorithm can be described by the following sequence of steps:

1) The *root* node is added to the task queue that contains all geometric primitives.
2) The first node in queue becomes *current*.
3) The current node is split into 16 bins along all the three axes. For every bin the number of geometrical primitives is calculated and the bounding volume is computed.
4) The optimal split plane is calculated by using SAH.
5) The current node is split into two new nodes containing geometrical primitives located to the left and to the right from the selected plane respectively.
6) For every new node the number of geometrical primitives is compared with the specified threshold number (we used 4). If the number of the primitives exceeds the threshold then the corresponding node is added to the task queue.
7) The current node is removed from the task queue. If the queue isn't empty then we go to the step 2.

3.2 The Adaptation for GPU

The main objective of this work was mapping of the general algorithm on the architecture of modern graphics processors.

The most simple way is mapping of a single task on one *work group* (here and below we use the terminology of OpenCL standard). In this case, we will repeat getting of the same result as in work [7]: low performance on the first levels of the tree, as soon as only some part of the available cores will be used in calculations, and the decline in productiveness at the last levels of the tree associated with an increase in overhead costs needed to support a large number of small tasks.

The following relatively simple method was proposed in work [8] and implemented for the Intel MIC architecture: for large nodes (with the number of primitives bigger than the specified threshold) the resources of the whole processor are used, while the other nodes are processed according to the previous scheme – a single task for a work group. This approach has proved effective for the architecture of Intel MIC, but when mapped on the GPU architecture there appears a number of problems. As in the case with the previous approach, there is a reduce in productivity at the low levels of the tree, and there an "intermediate layer" appears – a part of the levels at which a single task is not able to load the entire graphics processor, but the number of tasks is not large enough to project them onto a work group.

3.2.1 The General Scheme

For an effective use of modern GPU we have developed an improved algorithm for building of SAH BVH tree, which is largely free from the problems mentioned above. The general scheme of the algorithm is as follows:

1) At each step of the construction of the tree *all* the available tasks are fulfilled, regardless of their size. As a result, we are able to utilize the resources of the GPU to the full as well as to reduce the overhead costs associated with the transfer of tasks from the CPU.
2) For the processing of the nodes containing a sufficient number of geometric primitives (we used a threshold of 256), several work groups are used on a node (we used $N/512$, where N – the number of geometric primitives in the node). In conjunction with the optimization from paragraph 1), this approach enables us to use the number of resources close to the optimum.
3) The construction of the tree is divided into 3 stages: the processing of *large* nodes (more than 32K of primitives), the processing of *secondary* nodes (from 256 to 32K primitives) and the processing of *small* nodes (less than 256 primitives). This approach makes it possible to use the special modifications of the algorithm at each level, what results in the possibility to reduce the overhead costs and optimize the utilization of the GPU resources.

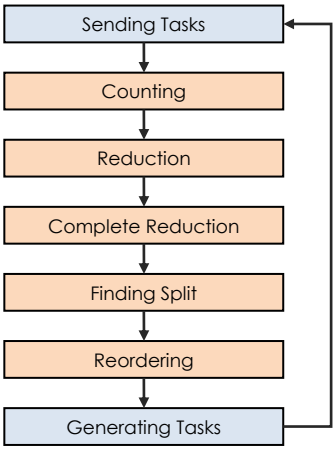


Figure 1: Construction of the tree level.

Figure 1 illustrates the most general scheme for building of the level of the tree (which is used at the stage of processing of the large nodes). Of all the presented stages only the sending of tasks and the generation of the new tasks are implemented on the CPU. These stages are the least labor-consuming, so their porting to the GPU is not reasonable.

3.2.2 The Generation of the Tasks for GPU

As it was stated earlier, in our implementation for processing of one node the several work groups are used, while several groups of nodes are processed simultaneously. This allows for high loading of the GPU, but, unfortunately, leads to the fact that there is no simple way to determine the amount of work for each specific work group. Therefore, as the additional parameters for work group we pass two data arrays – the number of the node being processed by the work group, and the number of this work group in the node. This information combined with data about nodes (the number of the primitives, the number of the first

primitive, bounding the volume of the node) allows to determine the amount of work for each work group.

3.2.3 Counting

The second stage in the construction of the tree level is counting of the number of primitives in each bin and of the bounding volumes (AABBs) of these primitives for each work group (this is correct as each work group processes the primitives from one node of the tree). In this case, the bins for all the three coordinate planes are calculated – as it was proved by the experiments in certain scenes it makes a significant (up to 10-fold) performance increase. Stage can be divided into two parts:

1) The conversation of information about geometric primitives into the information about the bins of tree nodes.
2) The application of the algorithm for parallel reduction to this data.

To implement the first part we have used the approach similar to that used in works [7] and [8] – the primitives of each work group are divided into parts consisting of 16 elements (in accordance with the number of bins) and each of the mentioned parts is processed by 16 consecutive threads ("*halfwarp*" in the terminology of NVIDIA CUDA). This algorithm can be described by the following pseudo code:

```
for i in 1 to 16 do
  if bin(triangle[i]) = threadId
    bin[threadId].append(triangle[i])
```

Using 16-passes on the 16 elements may seem superfluous, but it has the following advantages:

1) It is optimally mapped on the GPU SIMD architecture.
2) It minimizes memory conflicts (eliminates "*bank conflicts*" and provides "*coalesced*" access).

As an alternative solution to this problem we can propose the calculation by each data stream of the bin for an associated triangle and the subsequent reduction of these data. This approach leads to an increase in required memory (about 16 times) and the significant computational costs associated with the subsequent reduction.

3.2.4 Reduction

With the implementation of this phase the classical algorithm of parallel reduction was used. However, in the process of adaptation one significant change was made.

Firstly, we do not use a *variable* number of iterations of the algorithm. For the processing of the "average" levels of the tree just *one* iteration is enough, for the processing of "high" levels, we used an additional kernel of "final reduction", reducing all the available sets of bins in one (for "small" levels this step is not needed at all). The losses of productivity are not important for this approach, since most of the computational burden falls on the steps of calculation and reduction. The result is a simpler algorithm and reduced the amount of data exchange between the CPU and the GPU.

3.2.5 Search of Optimal Splitting

This is the one of the least resource-consuming parts of the algorithm. On the base of the available data on the bins the optimal split plane is calculated in it with the help of SAH. It was implemented on the GPU to reduce the amount of the traffic between the CPU and the accelerator.

3.2.6 Reordering

The last resource-consuming stage of the tree construction is the reordering of the nodes' elements in accordance with the found splittings (the elements on the left of the split plane are moved to the beginning of the array, the elements on the right – at the end). As a basis for solving this sub problem we used the *radix sort* algorithm, effectively implemented for the GPU in work [11].

This algorithm can be divided into two main parts – the prefix summation of the indices of the record (the place in the array where the element is to be placed) for all the geometric primitives of the node and strictly speaking the reordering of the array elements. Traditionally these parts are performed in several passes which entails additional costs of memory because of the need to store the results of intermediate calculations and they require additional calculations (associated with the launch of a prefix summation on the "global" level [12]).

In our algorithm the *atomic* operations on the global memory were used instead of the "global" prefix summation. As a result the additional steps for computing of the prefix sum on a global level have been replaced by *one* atomic operation for the work group on the local level what allowed to reduce the memory consumption and the number of computations.

3.2.7 Generating new tasks

It also belongs to the least resource-consuming parts of the tree building. At this stage the new nodes (corresponding to the obtained splittings) are added into the resulting tree and, depending on their size, the new tasks are generated. At the stages of processing of large and medium-sized nodes the nodes with the number of primitives which is less than a predetermined threshold (we used the 32K for the large and 256 for the medium ones) are added to the task queue belonging to the next step. On the stage of processing of small units they are considered to be the leaves (here as the threshold value is used 4).

4. TRAVERSING

4.1 The Basic Algorithms

So there are two main approaches to the realization of traversing of the tree on GPU: the *stack-based* and the *stackless*. The stackless approach was widely spread when only shaders were available for the purposes of ray tracing. As soon as writing data from the fragment shader (kernel) was possible only at the current fragment of the output texture, for realization of the full stack it was necessary to use the multi-pass schemes of low efficiency. With the advent of such instruments as CUDA and OpenCL it became possible to use the stack because we can easily get the access to the global device memory for reading and writing. Still to compare the performance we have realized the both variants.

4.1.1 The Stackless Algorithm

The stackless algorithms are normally based on the preprocessing of the tree and providing of additional information for the traversing of the tree. The algorithm based on using of the *escape indices* serves as an example of that approach for BVH tree [9]. The information about the node that is passed the next is counted and saved for every node. The algorithm showed some good results on simple scenes but with the scene getting more complex (what means the tree getting bigger) it turned out to behave much worse than the stack one. Besides that in the situation when the tree is built on every frame it is necessary to take into

consideration any additional processing of the tree because it must be done every time.

4.1.2 The Stack Algorithm

When traversing with a stack we can choose to which node of the tree we are to go further. For the ray tracing tasks the tree is being traversed in *front to back* order, what means that on every step you need to go to the nearest node. In this way the search for the nearest hit is done quicker. For traversing of the tree with a stack it is necessary to organize separate stacks for each thread. For that we need to allocate the fixed-size array in the *private* (in the terms of OpenCL) memory sufficient for storing of the stack of the maximum possible length the same as it is described in work [10].

4.1.3 The Stack in Registers

Though the organization of the private stack for every thread doesn't appear to be a problem anymore still any incoherent references to the external memory are undesirable. We have realized the variant of the traversing with a stack in the GPU registers. Thus for the stack element only 2 bits from the register will be given. With this approach the traversal algorithm has got three main states: to visit the right neighboring node, to visit the left neighboring node and also to go up the tree. A set of these states provides sufficient information for the traversing of the tree in the same order as with a normal stack. However, because of the growing complexity of the kernel code and the need to prepare and read the additional information for each node of the tree, such an approach in the current implementation did not provide for a significant advantage in comparison with the normal stack.

5. RESULTS

To estimate the operating time of the described algorithm the different scenes have been used with complexity from 10K to 3M of triangles. The tests have been conducted on a computer with the GPU NVIDIA GeForce GTX 480 being run by the Linux operating system (with 270.41.06 version of the video driver).

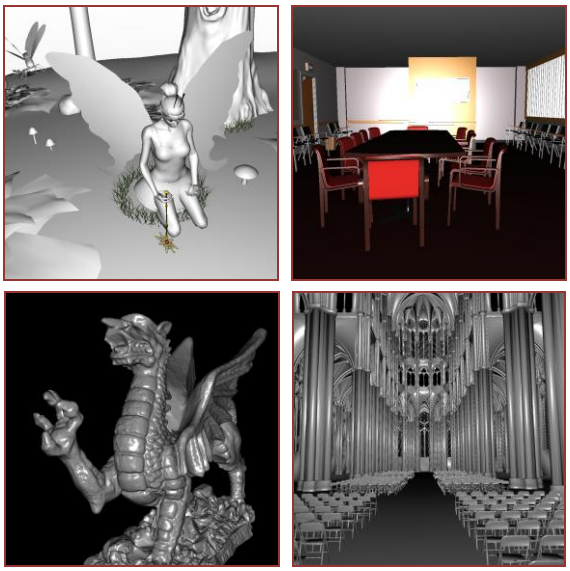


Figure 2: Sample frames from the test scenes (*Fairy Forest*, *Conference*, *Welsh-dragon* and *Cathedral*).

Table 1 contains the comparison of our results with the results from other well-known works.

Table 1: Comparison to other GPU SAH BVH builders: pure build time (*ms*) / rendering performance (*FPS*, 1024×1024).

Scene	Lauterbach [7] (GTX 280)	Wald [8] (Intel MIC)	Ours (GTX 480)
Toasters / 11K	N/A	11 / 105	13 / 83
Fairy Forest / 174K	488 / 21	31 / 29	40 / 25
Cloth / 92K	N/A	19 / 97	19 / 42
Dragon/Bunny / 252K	403 / 8	43 / 55	49 / 15
Conference / 284K	477 / 24	42 / 46	98 / 36
Welsh-dragon / 2.2M	N/A	N/A	362 / 20
Cathedral / 3.2M	N/A	N/A	697 / 14

6. CONCLUSION

In this work the task of building of full SAH BVH on GPU has been studied. The given methods for adapting of the general algorithm allowed to improve the results at more than 10 times compared to the best implementation known [7] (taking into account the difference in the hardware used – up to 5 times). Moreover, the obtained timing estimates are comparable with the estimates for the "compromise" structures (providing for rapid construction, but less effective for the ray tracing – Hybrid BVH, Two-level Grids), obtained in recent works [7], [13]. The current implementation allows to perform the rendering of arbitrary dynamic scenes of up to 800K of triangles and static scenes of up to 5M of triangles.

7. REFERENCES

[1] Timothy J. Purcell. *Ray Tracing on a Stream Processor*. Ph.D. dissertation, Stanford University, March 2004.

[2] Foley T., Sugerman J. *KD-Tree Acceleration Structures for a GPU Raytracer*. In Proceedings of the ACM SIGGRAPH/ Eurographics conf. on Graphics hardware, pp. 15–22, 2005.

[3] J. Gunther, S. Popov, H. Seidel, P. Slusallek. *Real-time Ray Tracing on GPU with BVH-based Packet Traversal*. In Proc. of the IEEE Symposium on Interactive Ray Tracing, 2007.

[4] I. Wald, W. Mark, J. Gunther, ... , P. Shirley. *State of the Art in Ray Tracing Animated Scenes*. Computer Graphics Forum 28(6), 2009, pp. 1691–1722.

[5] J. Goldsmith and J. Salmon. *Automatic Creation of Object Hierarchies for Ray Tracing*. IEEE Computer Graphics and Applications, vol. 7, no. 5, pp. 14–20, 1987.

[6] I. Wald. *On fast Construction of SAH-based Bounding Volume Hierarchies*. In Proceedings of the Eurographics Symposium on Interactive Ray Tracing, 2007, pp. 33–40.

[7] C. Lauterbach, … , D. Manocha. *Fast BVH Construction on GPUs*. Computer Graphics Forum, 28, 2, 375–384, 2009.

[8] I. Wald. *Fast Construction of SAH BVHs on the Intel Many Integrated Core (MIC) Architecture*. IEEE Transactions on Visualization and Computer Graphics, 2010.

[9] Thrane N., Simonsen L. *A Comparison of Acceleration Structures for GPU Assisted Ray Tracing*. Master's thesis University of Aarhus (2005).

[10] Zhou K., ... , Guo B. *Real-time KD-tree construction on graphics hardware*. ACM Trans. Graph. 27, 5, 1–11, 2008.

[11] Satish N., Harris M., Garland M. *Designing efficient sorting algorithms for manycore GPUs*.

[12] Harris M. *Parallel Prefix Sum (Scan) with CUDA*.

[13] J. Kalojanov, … , P. Slusallek. *Two-level Grids for Ray Tracing on GPUs*. In Proc. of the Eurographics conf., 2011.

Automated processing of retinal images

A.A. Chernomorets[1], A.S. Krylov[1], A.V. Nasonov[1], A.S. Semashko[1], V.V. Sergeev[1],
V.S. Akopyan[2], A.S. Rodin[2], N.S. Semenova[2]
[1]Faculty of Computational Mathematics and Cybernetics [2]Faculty of Fundamental Medicine
Lomonosov Moscow State University, Russia
{chernomorets, kryl}@cs.msu.ru

Abstract

The problem of automated processing of retinal images for semi-automatic diagnosis of retinal diseases is considered in this paper.

The main aspects of retinal image processing are discussed. Methods for non-uniform illumination correction, blood vessel detection and macula and optic disc segmentation, finding dark and light spots in the macula area are suggested.

Keywords: *retinal image processing, eye fundus, telemedicine.*

1. INTRODUCTION

Automated analysis of retinal images using software programs makes it possible to perform mass diagnosis (screening) without involving a medical officer. This is very important in terms of time saving and health budget economy and gives a possibility to make the criteria of detection of the most dangerous retinal diseases more objective and to detect retinal diseases at early stages.

The following major task groups can be marked out in the problem of automated processing of retinal images:

1. Preprocessing of retinal images.

2. Segmentation of retinal objects which include blood vessels, macula — the central part of the retina and the optic disc (optic nerve head).

3. Detection and assessment of pathologies.

2. PREPROCESSING OF RETINAL IMAGES

Image preprocessing consists of correction of non-uniform luminosity, color normalization and contrast enhancement. Methods of general image enhancement (sharpening of blurred images, noise suppression, etc.) can be added to this list.

In this work we use a method of luminosity correction that is based on segmentation of background pixels and subsequent computation of luminosity function based only on the background image [1]. The advantage of this approach is that it does not produce ringing effect.

The algorithm consists of two steps:

1. First we segment background pixels based on analysis of statistical values. For each pixel of the image we compute mean value $\mu(x,y)$ and standard deviation $\sigma(x,y)$ within a window of a fixed size, then we compute Mahalanobis distance as follows:

$$D(x,y)=\left|\frac{I(x,y)-\mu(x,y)}{\sigma(x,y)+\sigma_\varepsilon}\right|,$$

where $I(x,y)$ — value of the pixel with coordinates (x,y), $\sigma_\varepsilon > 0$ — small coefficient.

The pixel is classified as the background pixel if $D(x,y) < D_T$. We use $D_T = 0,7$.

2. The second step consists of luminosity correction as

$$I_C(x,y)=128\frac{I(x,y)}{\mu_C(x,y)+\mu_\varepsilon},$$

where $\mu_C(x,y)$ is the mean intensity value of background pixels in the neighborhood of the pixel at (x,y), $\mu_\varepsilon > 0$.

Fig. 1 illustrates the proposed method of luminosity correction.

a) original image b) result

Fig 1: Example of correction of the non-uniform luminosity

3. OBJECT SEGMENTATION IN RETINAL IMAGES

3.1 Vasculature segmentation

There are several general approaches to automated vessel segmentation [2]: matched filtering; ridge detection; vessel tracking that allows connecting disjoint parts of the vessel; pixel-based classification; mathematical morphology.

In the proposed method we apply morphological amoebas to the result of Gabor filtering of the image.

3.1.1 Gabor filtering

2-D Gabor filtering is a convolution with a kernel that can be formulated as

$$g(x,y;\lambda,\theta,\psi,\sigma,\gamma)=\exp\left(-\frac{x_\theta^2+\gamma^2 y_\theta^2}{2\sigma^2}\right)\exp\left(i\left(\frac{2\pi x_\theta}{\lambda}+\psi\right)\right),$$

where $\left.\begin{aligned} x_\theta &= x\cos\theta + y\sin\theta \\ y_\theta &= -x\sin\theta + y\cos\theta \end{aligned}\right\}$ is the rotation operation. In these notations, the parameter λ defines the wavelength of the sinusoidal factor, θ sets the orientation of the filter, ψ is the phase offset, σ is the scale of the filter and γ specifies the ellipticity.

Values of the parameters λ, θ and σ are chosen in order to detect parts of vessels of different directions and widths. We use 6 directions from 0° to 150° with a step of 30° and 4 values for parameter σ: $\frac{1}{4}\sigma_0, \frac{1}{2}\sigma_0, \frac{3}{4}\sigma_0$ and σ_0, where σ_0 is the maximum vessel width. Other parameters are set to: $\lambda = 3\sigma$, $\psi = \pi$, $\gamma = 1$.

The result is the maximum modulus of the filtering over all angles for multiple scales.

Fig. 2 illustrates vessel enhancement by Gabor filtering.

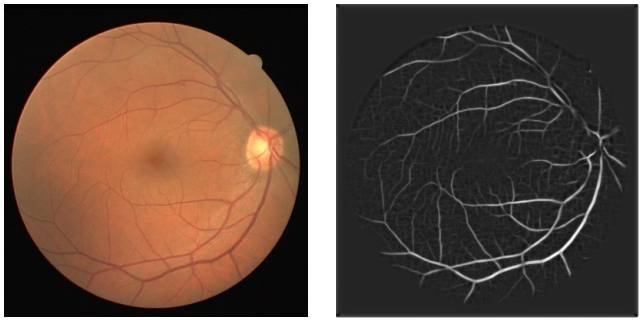


Fig 2: Result of vessel enhancement using Gabor filtering.

3.1.2 Morphological amoebas

The segmentation method is applied to the result of the Gabor filtering. Thresholding does not produce satisfying results, yielding to either a lot of false positive responses or loss of significant amount of smaller vessels. Mathematical morphology allows to enhance thresholding methods.

For segmentation of retinal vasculature we use morphological amoebas, described in [3,4], with modified amoeba distance function. The general idea of morphological amoebas is that the shape of the structuring element is adapted for each pixel of the image.

For each pair of adjacent pixels the distance based on pixels' intensities is computed. We use the following distance function:

$$d((x_0,y_0),(x_1,y_1)) = \sqrt{(x_1-x_0)^2+(y_1-y_0)^2} + \\ +\lambda\sqrt{(I(x_0,y_0))^2+(I(x_1,y_1))^2}, \quad \lambda \geq 0.$$

The summand $\lambda\sqrt{(I(x_0,y_0)^2+I(x_1,y_1)^2}$ makes the distance shorter in dark areas and longer in light areas. It restricts the amoeba expansion only inside the vessel. Parameter λ defines the cost of expansion. The summand $\sqrt{(x_1-x_0)^2+(y_1-y_0)^2}$ does not allow the amoeba to grow indefinitely in case of a constant image.

The structuring element for the pixel (x_0, y_0) consists of all pixels with the shortest path to the pixel (x_0, y_0) not exceeding the given threshold.

The examples of amoebas' shapes are given in Fig. 3.

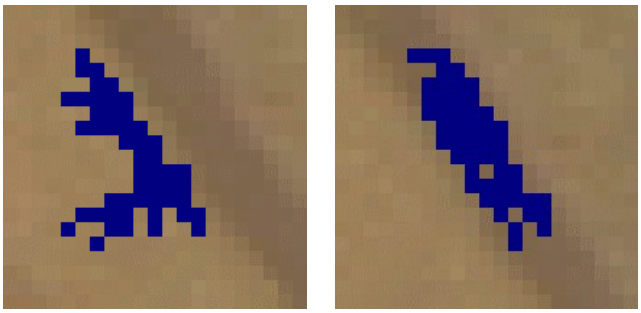


Fig 3: Examples of shapes of morphological amoebas. Pixels that form the structuring element are marked in blue.

The algorithm of segmentation using morphological amoebas consists of the following steps:

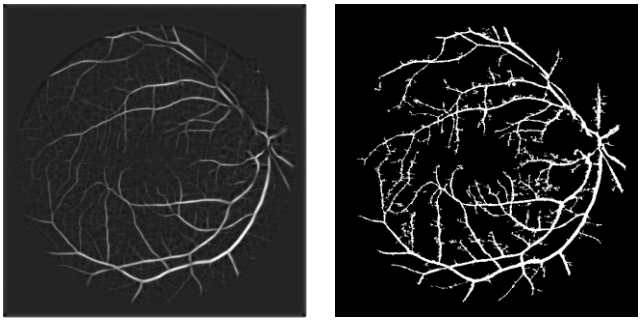
1. Via thresholding of the result of Gabor filtering the set M_0 that consists only of vessel pixels is produced.

2. The set of vessel pixels is morphologically dilated using morphological amoebas. The set is then eroded using circular structuring element and is joined to the set from the previous step:

$$M_{n+1} = M_n \cup Erosion(AmoebasDilation(M_n))$$

3. The process terminates when $M_{n+1} = M_n$.

The result of the segmentation using morphological amoebas is shown in Fig. 4.



Result of Gabor filtering | Result of segmentation

Рис 4: The result of the vessel segmentation using morphological amoebas.

3.2 Macula and optic disc segmentation

The localization of macula and optic disc can be performed via analysis of the vasculature.

We find these regions in assumption that form of the major vessel arcades is close to circle, the center of this circle is the center of the macula and the circle crosses the optic disc.

Let $\{P_k = (x_k, y_k)\}$, $k = 1,...,K$ be a set of coordinates of major vessel pixels. The resulting circle is the one that minimizes the dispersion of distances from pixels P_k to the circle boundary.

The macula region is taken as a circle with 1/3 of the radius of obtained circle and the same center.

A bright region on the circle boundary is taken as the initial approximation of the center of optic disc that is further used for detection of its boundary.

Example of macula and optic disc localization is shown in Fig. 5.

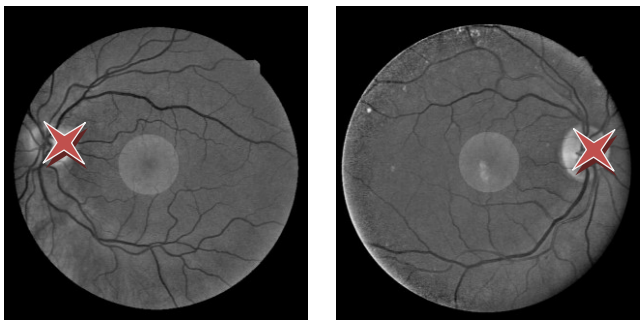


Fig 5: Result of macula and optic disc localization. The macula region is highlighted in the center of the image.

4. DETECTION AND ASSESSMENT OF PATHOLOGIES

4.1 Finding spots in the macula area

Appearing of dark and light spots in the retina can be an indication of the retinopathy.

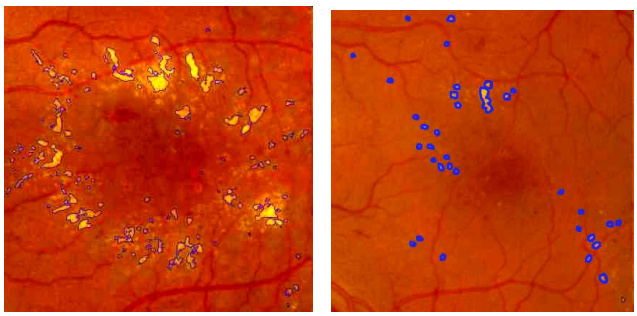
4.1.1 Light spot detection

We perform the detection of lesions in the macular area.

The green channel of the preprocessed image is used for the light spot analysis. Morphological closing and opening is performed to exclude blood vessels from the image. The size of the structuring element is chosen in accordance to image size. Then we calculate dispersion values in every pixel of the image, apply thresholding and find coordinates of light spot candidates.

Next to distinguish between light spots and outliers we fill every light spot candidate area with the value taken from surrounding background pixels, then subtract this image from the original one and apply the threshold to find light spots.

An example of light spot detection is shown in Fig. 6.



Fid 6: An example of light spot detection in retinal images.

4.1.2 Dark spot detection

Like for the light spot detection, the green channel of the preprocessed image is used for dark spot detection. We perform median filtering, then apply thresholding to find dark spot candidates. Next dark spots are separated from outliers. We calculate properties of the candidate areas which include compactness, ellipticity, dispersion, average intensity in red and green channels. The first property is used to exclude large blood vessels. The remaining small vessels are segmented from dark spots using the SVM (Support Vector Machines) method.

An example of dark spot detection is shown in Fig. 7.

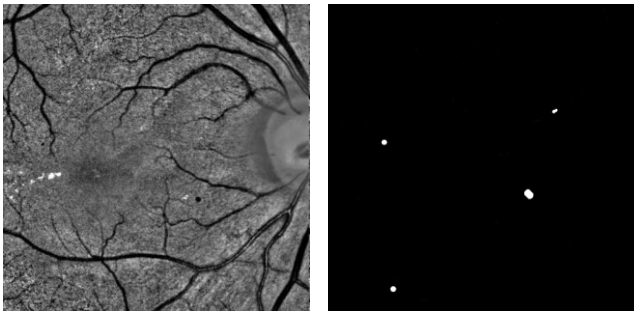


Fig 7: An example of dark spot detection.

4.2 Finding optic disc boundary

The analysis of optic disc boundary is important for glaucoma diagnosis. To find its boundary the method based on active contours was developed [5].

At the preprocessing step the blood vessels are removed from the image using morphological closing and opening in CIE Lab color space. [6]. Next the blood vessel map is constructed from the difference of the obtained image and the source image. Then the edge map is calculated as the modulus of the intensity gradient. Using the Hough transform, the initial contour best approximating the optic disc boundary is taken. The external force field Gradient Vector Flow (GVF) [7] which is used in active contours method is constructed from the boundary map.

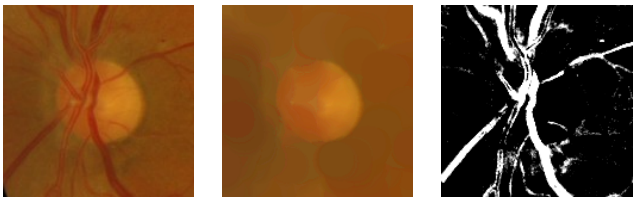


Fig 8: An example of morphological preprocessing. Left: the source image; middle: the result of blood vessel removal; right: the blood vessel map.

At the next step, the preliminary result is calculated. This result is usually inaccurate and the deviation of the obtained boundary from the true optic disc boundary is usually observed at intersections of the optic disc boundary with blood vessels.

At the last step, the correction of the obtained boundary is performed. We construct the ellipsis approximating the obtained boundary. Every point of this ellipsis is assigned a weight depending on the edge map and the blood vessel map. The external force attracting the contour to this ellipsis is added. Another force (the pressure force) directed along the external contour normal is added to the points of intersection of the contour with blood vessels.

Fig 9: Examples of optic disc detection. True contours are shown by white color. The results of the proposed method are shown by green color. Black dashed lines are the preliminary results (without applying active contours).

The following performance values were obtained on DRIVE and MESSIDOR bases: average overlap – 0.89, sensitivity – 94.1%, specificity – 98.4%. This confirms the effectiveness and the reliability of the proposed method.

5. CONCLUSION

The proposed in the paper algorithms for non-uniform illumination correction, blood vessel detection, macula and optic disc segmentation in retinal images, finding light and dark spots in the macula area compose the basis of the telemedicine system of retinal image analysis developed by the authors group.

The target of the future work is creating the integrated software system based on the optimization and coordination of the work of the proposed algorithms.

The work was supported by the state contract №8/3-617н-10 with the Moscow government.

6. REFERENCES

[1] G.D.Joshi, J.Sivaswamy. Colour Retinal Image Enhancement based on Domain Knowledge // *6th Indian Conf. on Computer Vision, Graphics and Image Processing*, 2008, pp. 591–598.

[2] Winder R.J., Morrow P.J., McRitchie I.N., Bailie J.R., Hart P.M.. Algorithms for digital image processing in diabetic retinopathy. Computerized Medical Imaging and Graphics, vol. 33, pp. 608–622, 2009.

[3] M.Welk, M.Breub, O.Vogel. Differential Equations for Morphological Amoebas // *Lecture Notes in Computer Science*, Vol. 5720/2009, 2009, pp. 104–114.

[4] Romain Lerallut, Etienne Decencière, Fernand Meyer. Image filtering using morphological amoebas// *Image and Vision Computing*, Vol. 25, 2007,pp. 395–404.

[5] A. S. Semashko, A. S. Krylov, A. S. Rodin. Using Blood Vessels Location Information in Optic Disk Segmentation // *16th International Conference on Image Analysis and Processing* (ICIAP'2011), accepted for publication.

[6] A. Osareh, M. Mirmehdi, B. Thomas, R. Markham. Colour morphology and snakes for optic disc localisation. // *6th MIUA Conference*, 2002, pp. 21—24.

[7] C. Xu, J. Prince. Snakes, shapes, and gradient vector flow. // IEEE Trans. Image Processing 7, 1998, pp. 359—369.

About authors

A.A. Chernomorets – PhD student of Faculty of Computational Mathematics and Cybernetics (CMC), Lomonosov Moscow State University (MSU).

A.S. Krylov – professor, head of Laboratory of Mathematical Methods of Image Processing, CMC MSU.

A.V. Nasonov – member of scientific staff of CMC MSU

A.S. Semashko – PhD student of CMC MSU.

V.V. Sergeev – student of CMC MSU.

V.S. Akopyan – professor, head of Ophthalmology department, Faculty of Fundamental Medicine (FFM), MSU.

A.S. Rodin – assistant of Ophthalmology department, FFM MSU.

N.S. Semenova – assistant of Ophthalmology department, FFM MSU.

A new stage of development of modern microscopy

Stepan Panov
MECOS company, Moscow, Russia
panovstepan@gmail.com

Abstract

Major achievements in the field of automation microscopy, such as automatic analysis of blood smear, counting leukocyte formula (wbc), automated analysis of feces for parasites, the creation of a virtual slide, online virtual slides are considered.

Machine vision applied to medical imaging, with high accuracy allow differentiating, and then segmenting the objects according to their typical properties.

The main problem is the difference in color and quality of the product depending on the method of its preparation and coating on a slide. To solve this problem applies a starting point for search algorithms and adapt to current conditions.

Keywords: *automated microscopy, virtual slides, virtual microscopes, blood analysis.*

1. INTRODUCTION

Image processing - an integral part of working with any kind of analysis and the results of medical research. History of medical imaging stems from the first electronic image of the musculoskeletal system, but we also look at the lab. Daily passes through a typical laboratory set a variety of tests ranging from blood and ending with a scrape on helminthes eggs. Any sample analyzed specialist based on their knowledge and experience. The accuracy and reliability of the results may cause some doubt in the absence of backup tests. Therefore, a new stage of development of laboratory tests is automated or robotic microscope, which allows you to automatically carry out the study, with follow-up inspection, adjustment and the ability to store the results.

2. ANALYSIS OF BLOOD SMEAR

Многие Many diseases can give the normal ratio of blood cells and abnormal cell morphology, so use only enough flow hematology analyzer. [3] Modern flow hematology analyzer can not detect unusual cells and approximately 20% of the test results are questionable. And there must be, at least, selective microscopic control results of flow hematology analyzer.[2] Microscopic examination is recommended for the differential diagnosis, particularly if the flow hematology analyzer has a deviation from the norm. Hand microscopy is labor-intensive and responsible work, requiring high stress. [4] The creation of an automated system can not only facilitate the work of specialists, but also to preserve the accuracy of detection of atypical cells.

Automatic analysis can be divided into several stages:

- Finding a starting point, the definition of layer thickness.

  We move to the area of a monolayer of erythrocytes then we perform binarization of images and divide the objects and background, assessment of the size of objects and their optical density in order to approach the values for the given values.

- Finding the cells.

  The algorithm is based on the difference in color of the nucleus and other objects. The core is inherent in a blue color.

- Segmentation

  At the end of the analysis it is necessary to determine the types of cells. The algorithm works as follows: Selecting the core. Isolation of the background. Separation of the remaining pixels in the cytoplasm and other objects.

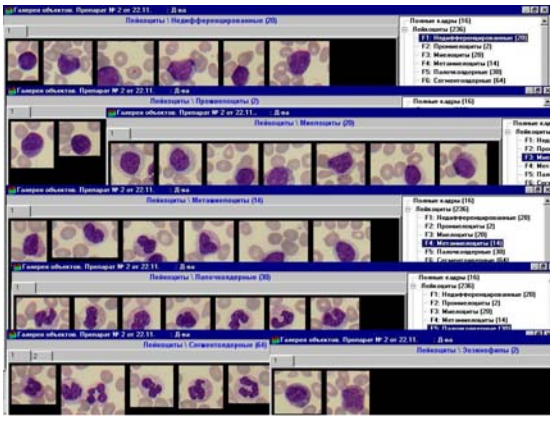


Image 1: Object gallery.

- Classification of erythrocytes.

  Classification is carried out in form, size and optical density. Highlighted the atypical cells (echinocytes, drop-shaped, bite, eliptotsity, and irregularly shaped cells), macro- and microforms. Also allocated special types of cells with an abnormal profile of optical density (target cells, spherocytes, stomatotsity).

- Classification of leukocytes.

  Six major types of cells are distinguished (stab neutrophils, segmented neutrophils, eosinophils, basophils, lymphocytes, monocytes). Color, luminance, geometric and textural characteristics are used.

3. ANALYSIS OF FECES FOR PARASITES

Another area of laboratory microscope is parasitology. In contrast to blood smears, to search for helminthes eggs it is necessary to examine the material at different focal lengths. Determination of the eggs carried by the characteristic black rings (shells). Sample preparation requires special attention to medications. In most cases, a complex station with a slide-camera, mounted directly on the object table of the microscope. Two variants of screening are possible:

:

- In the first case, the installation performs the screening, moving, and focusing the slide chamber on all area and depth, revealing helminthes eggs. Special holder for the slide chamber provides reliable attachment during rapid movements of automatic screening. Detected objects fall into a database for visual identification of the doctor on a computer screen. [1]
- In the second case, we create a digital 3D copy of the sample (virtual slide) and then analyze it in the workplace or remotely

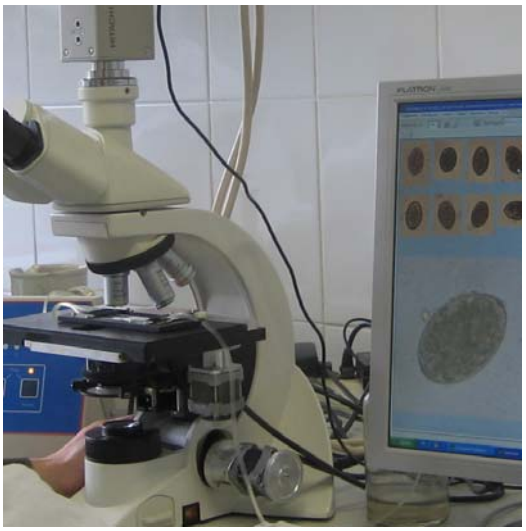


Image2: Automated analysis of the sample on helminthes eggs..

4. VIRTUAL MICROSCOPE

Virtual product is formed from a large number of neighboring primary physical field of view with an adaptive adjustment of the boundaries between them, one without defect border (pan).

Virtual drug can be viewed on a remote computer simulated microscopy (moving, different size) and morphometry of objects. Together with the virtual microscope, it became possible long-term storage of samples useful for quality control and training.

4.1 Virtual slides online

The development of Internet technology allows for the rapid exchange of information between laboratories. The integration of virtual slides with platform GMAPS API has significantly simplify remote research. On slides, placed on Internet site, you can navigate, edit, zoom, measure the dimensions of objects, as well as switch Z layers for 3D virtual slides of urine sediment, helminthes eggs and fecal matter. In contrast to real samples, virtual slides do not deteriorate over time and are available for viewing virtually for any interested man.

5. PROTECTION OF PERSONAL DATA

The generally accepted standard for transmitting medical images is a standard DICOM (Digital Imaging and COmmunications in Medicine). Storing the virtual slide is carried in several ways. The first option - stitched image (large image in JPEG2000 format, with a patient information). The alternative - a collection of sets of tiles, numbered in order. In the case of placement of virtual slides on the Internet, patient information is not reported.

6. CONCLUSION

In conclusion, I want to note that the development of modern microscopy increases the accuracy and performance studies, which undoubtedly plays a crucial role in formulating the correct diagnosis.

7. REFERENCES

[1] Bogdan S. IV Bykov Parpar AA (CGE MBA MEKOS) Automated microscopy of fecal parasites.

[2]Cornet E., Perol J-P., X Troussard. Performance evaluation and relevance of the CellaVision™ DM96 system in routine analysis and in patients with malignant hematological diseases. Int J Lab Hematol. v.30, N 6, 2008.

[3] Honey, VS, AA Parpar Pyatnitsky AM, Sokolinskii BZ (MEKOS). Robotic microscopy introduces standard of quality blood smear analysis.

[4] Pyatnitsky AM, Honey VS Parpar AA (MEKOS) analysis of reticulocytes: manual microscopy, flow analyzers, analyzers or images?

About the author

Stepan Panov is a student at Bauman Moscow State Technical University, Faculty of special machines. He is an engineer at MECOS Company. His contact email is panovstepan@gmail.com

Diffuse Axonal Injury Lesion Segmentation Using Contouring Algorithm

Olga V. Senyukova[1], Valeriy E. Galanine[1], Andrey S. Krylov[1],
Alexey V. Petraikin[2], Tolibdjon A. Akhadov[2], Sergey V.Sidorin[2]
[1] Faculty of Computational Mathematics and Cybernetics, Lomonosov Moscow State University, Moscow, Russia
[2] Children's Clinical and Research Institute Emergency Surgery and Trauma, Moscow, Russia
osenyukova@graphics.cs.msu.ru

Abstract

Diffuse axonal injury (DAI) is a common type of brain damage induced by trauma. Accurate detection and quantification of DAI lesions is essential for assessment of a patient's state and making a prognosis of a traumatic disease. This study is devoted to semi-automated segmentation of DAI lesions on T2*-weighted brain magnetic resonance (MR) images. A radiologist outlines the regions containing lesions, and the lesions are further automatically delineated inside these regions.

A proposed algorithm for automated segmentation of DAI lesions inside specified regions is based on contouring algorithm which treats the image as a 3D topological map where a pixel's intensity corresponds to its height. It builds isolines of an intensity function. Lesion contours are among these isolines.

In order to distinguish true lesion contours from other closed contours obtained by contouring algorithm machine learning approach is exploited. A labeled training base with positive (lesions) and negative (non-lesions or lesion parts) examples of closed contours is used to train the classifier.

The algorithm was evaluated with real T2*-weighted brain MRI images.

Keywords: *Diffuse Axonal Injury, Brain MRI, Brain Lesion Segmentation, Contour Levels.*

1. INTRODUCTION

Recently automated and semi-automated algorithms for medical image processing are widely used by radiologists during evaluation of a patient's state and diagnostics of various diseases. In this paper we introduce a semi-automated method for detection of diffuse axonal injury (DAI) lesions on T2*-weighted brain MRI images.

Diffuse axonal injury occurs in 50% cases of severe traumatic brain injury. Occurrence of DAI lesions in midbrain and brainstem is the most common cause of coma and subsequent disability. An accurate and convenient method for detection of DAI lesions will help a radiologist to estimate current stage of the disease and make a prognosis of a further disease flow.

T2*-weighted MRI was chosen among other MRI modalities because it allows to identify DAI lesions visually rather clearly. The T2*-weighted MRI sequence is the most sensitive to the magnetic susceptibility induced by static field inhomogeneities, arising from paramagnetic blood breakdown products in DAI lesions. They look like small dark spots, usually in brain tissue, sometimes connecting each other. So some automatization can be added to the process. Fully automated algorithms for brain lesion detection usually lack accuracy and stability. True lesions can be easily confused with other formations, for example, blood vessels. Therefore, we developed a semi-automated algorithm.

A radiologist interactively specifies rectangular regions of interest (ROI) which are likely to contain DAI lesions. After this marking procedure each MRI section of the current stack may include none, one or several square regions (Figure 1). Then the lesions inside these specified regions are segmented fully automatically. So there is a trade-off between speed and accuracy.

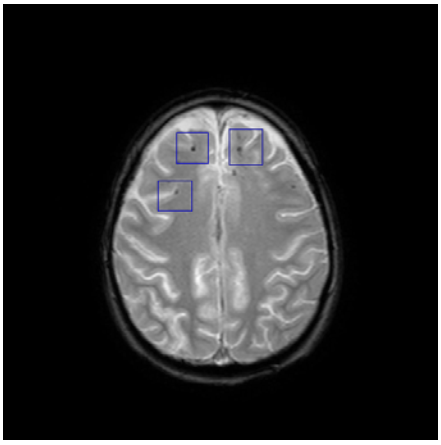


Figure 1: ROI selected on T2*-weighted MRI.

A proposed algorithm for DAI lesion segmentation, i.e. delineation of lesion contours, is based on contouring algorithm which is aimed to obtain closed contours which may potentially correspond to true lesion contours. In order to distinguish lesion contours from other closed contours machine learning is applied.

The rest of the paper is organized as follows. A brief review of previous work is given in section 2. Contouring algorithm purpose and its application to a current problem is described in section 3. Section 4 is devoted to selection of closed contours obtained by contouring algorithm, which correspond to DAI lesions contours, via support vector machine (SVM) classifier. Implementation details and comparison with other methods is given in Section 5. Section 6 includes conclusion and discussion.

2. PREVIOUS WORK

There are several works devoted to processing of MR images of brain with DAI. For the most part they deal with medical and practical aspects of the experiment rather than with image processing algorithms.

Paper [8] presents a method for quantification of severity of DAI by diffusion tensor images (DTI). Several specific characteristics

are measured at multiple locations and their statistical analysis is performed for patients and healthy control subjects. Voxels with statistically significant deviation from normal values are assigned to lesions.

In [7] two types of T2*-weighted MRI sequences are compared for detection of DAI. Lesions are defined as regions of abnormally low signal intensity.

A wide variety of works on detection of brain lesions in general exist. These lesions can be caused by multiple sclerosis, aging, vascular risk factors.

One of fundamental studies in this area is [16]. The paper presents semiautomatic technique for segmentation of white matter lesions (WML) using a supervised artificial neural network classifier. Multimodal MRI data is used. The classifier is trained to assign each pixel to one of five classes – background, white matter (WM), gray matter (GM), cerebrospinal fluid (CSF) and WML – based on its intensity on different MRI modalities.

A widely used method for segmentation of WML is described in [13]. It utilizes a previously proposed concept of fuzzy connected objects. An operator detects few points on dual-echo fast spin-echo MR images, which correspond to WM, GM and CSF. Each of these objects is further automatically detected as a fuzzy connected set. WML's are segmented as 3D fuzzy connected objects in the holes of these objects' union.

An important work on automated multiple sclerosis lesions detection is [11]. The method models distribution of intensities of WM, GM and CSF, and detects lesions as voxels which are not well described by these models. Markov random field is used to add contextual constraints because it is known that multiple sclerosis lesions are usually located in the vicinity of white matter.

In [1] normal tissues and WMLs are segmented by thresholding on some MRI-specific values. Potential WML clusters are also analyzed for 3D shape and surrounding tissue composition.

The method for automated segmentation of multiple sclerosis lesions on multimodal MRI images using fuzzy C-means (FCM) clustering is described in [4]. First, FCM is used to extract a mask for CSF and lesions which distinguishes them from other tissues. Then, FCM is reapplied to the masked image to distinguish lesions from CSF.

The authors of [9] propose to use k-nearest neighbor classifier to segment WM, GM, CSF and WMLs.

In [10] each voxel of brain volume is characterized by a vector of values from multiple MRI modalities – its own and of its neighbors. SVM is used for classification to lesions and non-lesions. Typical non-lesions for SVM training are selected using AdaBoost.

One of the latest works [15] introduces weak labels – rectangular regions of interest specified by an expert, which contain lesions. This concept is close to one used in the present work. Lesions are segmented automatically inside these regions. First, K-means clustering is used to identify lesions cluster. Intensity distribution is calculated based on the found cluster. A confidence map is built for a whole ROI. Voxels with high probability of belonging to lesions are used as seeds for more accurate segmentation by Grow-Cut algorithm [14].

The methods mentioned above represent a wide variety of approaches to brain lesions segmentation but all of them perform voxel-wise segmentation. In this work we propose a method which is based on a quite different approach – machine learning is applied to feature vectors relative to contours, not voxels. DAI lesions usually have a specific shape so we can utilize this a priori knowledge and obtain accurate and smooth lesion contours.

Furthermore, in this study there is no need for 3D segmentation – each MRI section is processed separately so that a radiologist can easily control the process.

3. CONTOURING ALGORITHM

The method proposed in this paper is based on contouring algorithm which is aimed to build smooth closed contours delineating DAI lesions.

Contouring algorithm in general is used for building a contour plot – a set of level curves of different heights of a function of two variables, i.e. isolines of a function. A level curve of height h of a function $f(x, y)$ is the set of all points (x, y) such that $f(x, y) = h$. One or more isolines can exist for level h.

Contour plots are widely used in cartography for building topographic maps, meteorology, geology and many other areas.

Contouring algorithms – algorithms for building contour plots – are described in different works, for example [2]. In this work we use contouring algorithm close to the algorithm described in MATLAB online documentation [12].

In our case $f(x, y)$ is a function which takes a value which equals a pixel's intensity at the point (x, y).

The specificity of a current problem requires that we consider only closed contour lines because only they can delineate DAI lesions. As it is shown on Figure 2 there exists a range of contour levels on which lesions are marked perfectly with closed contour lines.

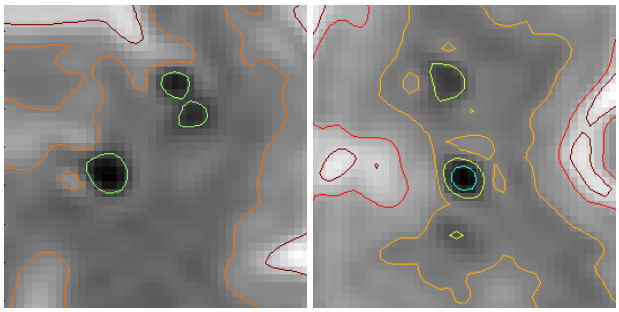


Figure 2: Contour maps of image ROI.

4. SELECTION OF REQUIRED CLOSED CONTOURS VIA SVM CLASSIFIER

After the contour lines are built, a problem which arises is how to select contours corresponding to DAI lesions contours from the whole set of closed contours.

A radiologist can easily do it by sight – he uses a priori knowledge of the shape, intensity and other specific features of DAI lesions. Therefore, we can try to apply machine learning approach to this problem and train a classifier which can

distinguish lesions contours from other closed contours based on these a priori characteristics of lesions.

4.1 Training and applying a classifier

We used the following features to characterize each closed contour:

- square of shape bounded by the contour;
- perimeter;
- mean intensity inside the contour / ROI mean intensity;
- intensity difference inside and outside the contour;
- elongation;
- density.

A feature vector corresponding to a contour is used as a sample for training and testing a classifier.

A training base is constructed as follows. Contour lines for a specified range of levels are built for a set of MRI images inside rectangular ROI specified by a radiologist. Only closed contours are selected among them. For each closed contour a feature vector of above mentioned features is calculated. If a contour corresponds to a DAI lesion contour it is marked as +1, otherwise it is marked as -1. Thus we obtain a labeled training set for a two-class classifier. A classifier model is built on this set. Then, if we want to detect all DAI lesions inside a ROI on a previously unseen MRI image we should do the following:

- Build contour lines for a range of levels.
- Select closed contours among these contour lines.
- Calculate a feature vector for each of these contours.
- Apply a classifier to each of these feature vectors in order to learn if a corresponding closed contour is a DAI lesion contour or not.

4.2 Support vector machine (SVM)

A popular SVM classifier [6] is used in this work. It constructs a hyperplane in a high-dimensional feature space, which separates the classes – a decision boundary. SVM tries to maximize the margin – the perpendicular distance between the decision boundary and the closest of the data points, which are called support vectors. The location of this boundary is determined by a subset of these support vectors. Maximizing the margin leads to better separation between classes.
An important property of SVM is that the determination of the model parameters corresponds to a convex optimization problem, and so any local solution is also a global optimum [3].

5. EXPERIMENTS

A proposed algorithm was implemented on C#. Implementation details and results are given below.

Contour lines for 20 levels in the intensity range $\left[\min_{x,y} I(x,y)+10, \min_{x,y} I(x,y)+70\right]$ were built by the contouring algorithm.

The base for training and validation of SVM classifier consists of 320 lesion contours and 231 non-lesion contours obtained from 41 MRI images. A total set of 8 brain MRI volumes was used. SVM classifier with linear kernel was applied.

Classification accuracy was evaluated by 5-2 cross-validation. On 80% of images DAI lesions were segmented successfully.

Obtained results were compared with results of segmentation by some basic computer vision methods such as edge detection and region growing.

For edge detection Canny edge detector [5] was used. The results are not satisfactory because it gives too much clutter and furthermore it loses the contours of lesions (Figure 3b).

Algorithms based on region growing, which were also tried, start with Gaussian blurring of the original image which allows to compute local intensity minima effectively. These local minima are used as seeds for region growing because DAI lesions are regions of minimum intensity. Each local minimum can contain one pixel or a group of pixels with equal intensities which are lower than intensities of neighboring pixels. Region growing starts from these local minima – more and more pixels which satisfy special criteria are added to each region. One of such criteria is intensity threshold which can be set manually or adaptively. Another threshold used in this experiment was based on the region square. In order to filter seeds obtained from local minima third threshold for intensity value was set and finally inside and outside border mean intensities difference was also checked by threshold value.

Region growing seems to be a better solution than edge detection, but in some cases it also works not correctly which results in re-segmentation of odd pixels or missing some pixels which definitely belong to the lesion (Figure 3c).

The proposed algorithm significantly outperforms region growing and edge detection methods. An example of its results is shown on Figure 3d.

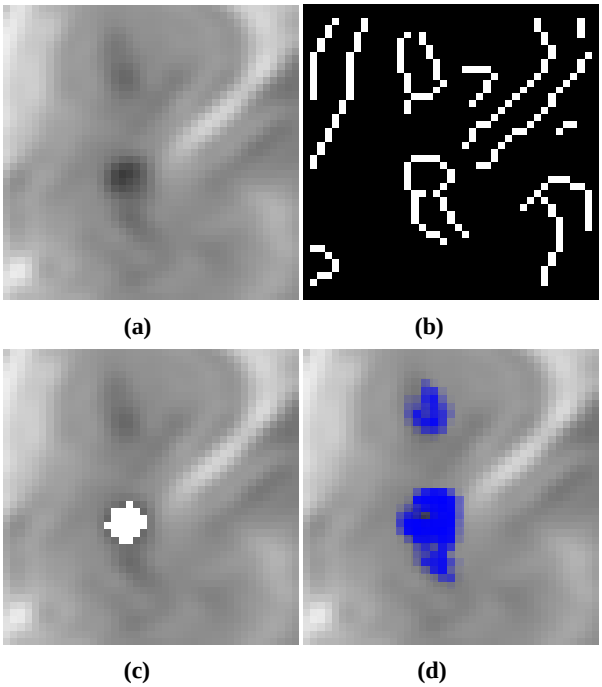


Figure 3: a: original ROI; b: Canny; c: region growing; d: our method.

6. CONCLUSION AND DISCUSSION

A novel approach to semi-automated segmentation of DAI lesions on brain MRI images was proposed. It is based on contouring algorithm which builds level curves for a specified set of levels – these curves are isolines of an intensity function. It was shown that these level curves can catch lesion contours well. In order to select required lesion contours from other closed level contours SVM classifier was used.

A proposed algorithm was evaluated on real brain MRI images provided by Children's Clinical and Research Institute Emergency Surgery and Trauma and demonstrated good results accepted by the radiologist.

However, some difficulties arise in the case of DAI lesions detection in severely damaged areas where these lesions cover almost all the area of the ROI. But in this case there is a fuzzy boundary between DAI and hemorrhage. Thus detection of hemorrhage is another problem and may involve other methods for its solution.

Another difficulty is that in some situations even a radiologist cannot be sure if a dark spot is a DAI lesion or a blood vessel. So one of further directions for this work is matching T2*-weighted brain MRI images with angiography images where blood vessels are well seen, in order to distinguish DAI lesions from blood vessels automatically.

The future work also includes creation of a more representative MRI database of brain damaged by DAI.

7. AKNOLEDGMENTS

This work was supported by federal target program "Scientific and scientific-pedagogical personnel of innovative Russia in 2009-2013".

8. REFERENCES

[1] B. Alfano, A. Brunetti, M. Larobina et al., “Automated segmentation and measurement of global white matter lesion volume in patients with multiple sclerosis”, *J. of Magnetic Resonance Imaging*, vol. 12, no. 6, pp. 799—807, 2000.

[2] M.J. Aramini, “The Design and Implementation of Computer Algorithms for Contour Plotting” (thesis), Stevens Institute of Technology, Hoboken, NJ 07030, (May 2, 1980).

[3] C. Bishop “Pattern Recognition and Machine Learning (Information Science and Statistics)”, Springer-Verlag New York, Inc. Secaucus, NJ, USA, 2006.

[4] A.O. Boudraa, S.M.R. Dehak, Y.-M. Zhu et al., “Automated Segmentation of Multiple Sclerosis Lesions in Multispectral MR Imaging Using Fuzzy Clustering”, *Computers in Biology and Medicine*, vol. 30, no. 1, pp. 23—40, 2000.

[5] J. Canny, “A Computational Approach to Edge Detection”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, no. 6, pp. 679—698, 1986.

[6] C. Cortes and V. Vapnik, “Support-Vector Networks”, *Machine Learning*, vol. 20, pp. 273—297, 1995.

[7] E. Giugni, U. Sabatini, G.E. Hagberg et al., “Fast Detection of Diffuse Axonal Damage in Severe Traumatic Brain Injury: Comparison of Gradient-Recalled Echo and Turbo Proton Echo-Planar Spectroscopic Imaging MRI Sequences”, *American Journal of Neuroradiology*, vol. 26, no. 5, pp. 1140—1148, 2005.

[8] T.A.G.M. Huisman, L.H. Schwamm, P. W. Schaefer et al., “Diffusion Tensor Imaging as Potential Biomarker of White Matter Injury in Diffuse Axonal Injury”, *American Journal of Neuroradiology*, vol. 25, pp. 370—376, 2004.

[9] C. Jongen, J. Van Der Grond, L.J. Kappelle et al. “Automated measurement of brain and white matter lesion volume in type 2 diabetes mellitus”, *Diabetologia*, vol 50, no, 7, pp. 1509—1516, 2007.

[10] Z. Lao, D. Shen, A. Jawad, “Automated Segmentation of White Matter Lesions in 3D Brain MR Images Using Multivariate Pattern Classification”, *3rd IEEE Int. Symp. on Biomedical Imaging: Nano to Macro*, pp. 307—310, 2006.

[11] K. Van Leemput, F. Maes, D. Vandermeulen et al., “Automated Segmentation of Multiple Sclerosis Lesions by Model Outlier Detection”, *IEEE Transactions on Medical Imaging*, vol. 20, no. 8, pp. 677—688, 2001.

[12] MathWorks. MATLAB online documentation. The Contouring Algorithm.
http://www.mathworks.com/help/techdoc/creating_plots/f10-2524.html#f10-2614

[13] J.K. Udupa, L. Wei, S. Samarasekera et al., “Multiple sclerosis lesion quantification using fuzzy-connectedness principles”, *IEEE Transactions on Medical Imaging*, vol. 16, no. 5, pp. 598—609, 1997.

[14] V. Vezhnevets, V. Konouchine, “Grow-Cut” - Interactive Multi-Label N-D Image Segmentation”, *Proceedings of Graphicon 2005*, pp. 150—156, 2005.

[15] Y. Xie, X. Tao, “White Matter Lesion Segmentation Using Machine Learning and Weakly Labeled MR Images”, *Proc. SPIE* 7962, 79622G, 2011.

[16] A.P. Zijdenbos, B.M. Dawant, R.A. Margolin, A.C. Palmer, “Morphometric analysis of white matter lesions on MRI images: method and validation”, *IEEE Transactions on Medical Imaging*, vol. 13, no. 4, pp. 716—724, 1994.

About the authors

Olga Senyukova is an assistant professor at the Chair of Computer Systems Architecture, Graphics & Media Lab, Faculty of Computational Mathematics & Cybernetics, Lomonosov Moscow State University (CMC MSU). Her contact email is osenyukova@graphics.cs.msu.ru

Valeriy Galanine is a student at CMC MSU. His contact email is galich1990@mail.ru

Andrey Krylov is a professor, head of Laboratory of Mathematical Methods of Image Processing, CMC MSU. His contact email is kryl@cs.msu.ru

Alexey Petraikin is a senior researcher at the Radiology Department within the Children's Clinical and Research Institute Emergency Surgery and Trauma (RD CCRIEST). His contact email is petrai@rambler.ru

Tolibdjon Akhadov is a professor, head of RD CCRIEST. His contact email is akhadov@mail.ru

Sergey Sidorin is a radiologist at the RD CCRIEST. His contact email is sid-sid@inbox.ru

Towards a Better Removal of Subsurface Fluoresence Artifacts in Block-Face Imaging

Andrey Tikhonov*
Faculty of Computational Mathematics and Cybernetics, Lomonosov Moscow State University, Moscow, Russia
Pavel Voronin†
Kurchatov NBIC Centre, National Research Centre "Kurchatov Institute", Moscow, Russia

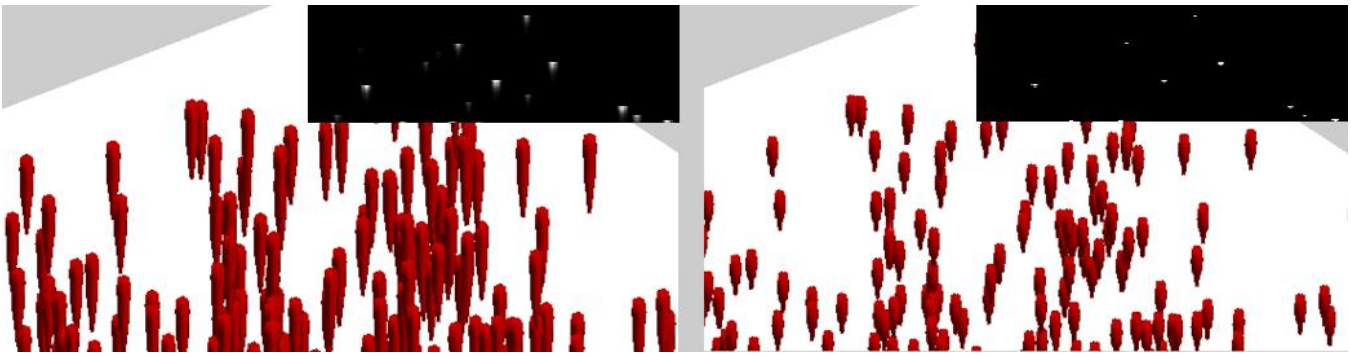


Figure 1: 3D expression map of *c-Fos* gene in an adult mouse brain, before and after subsurface fluorescence removal (fragment, iso-surface rendering); inset: 2D sections.

Abstract

Acquisition, transfer, registration and processing of 3D images of biological objects are all complex procedures that tend to introduce various artifacts, reducing the resolution of the imaging techniques. One such technique is block-face imaging, which suffers greatly from subsurface fluorescence. Removal of this type of artifacts can be reformulated as a deconvolution problem. Although several classic and specialized algorithms have been succesfully applied to this problem, they only work for relatively low amount of blur, sparsely distributed objects and predetermined convolution kernel. Following recent advances in deblurring of 2D natural images, we propose a new Expectation-Minimization algorithm approximating the maximum a posteriori probability solution. The algorithm is not limited to the case of fixed kernel, but can also work with partially known (parametric) or totally unknown kernels. We test the algorithm on model and real data.

Keywords: image processing, 3D deblurring, blind deconvolution, block-face imaging, EM-algorithm

1 INTRODUCTION

The last few years saw rapid development of 3D cryogenic block-face imaging methods. New methods of scanning combined with modern capabilities of visualization techniques allowed researchers to measure and analyze cell-level activity on the scale of the whole brain or even the whole organism [Roy et al. 2010]. These methods can be used in many applications including stem cell therapy, metastatic cancer treatment, gene expression mapping and phenotyping of laboratory animals [Krishnamurthi et al. 2010].

*e-mail:AndreyAlexTikhonov@yandex.ru
†pavel.voronin@gmail.com

While it became easier to obtain high-resolution 3D volumetric data, certain imaging artifacts can significantly reduce the resolving power of the technique. For block-face imaging of fluorescent proteins, the biggest problem is subsurface fluorescence and scattering of light. Effectively a type of blur, it is generally modeled as convolution with a point spread function (PSF):

$$\mathbf{I} = \mathbf{L} \otimes \mathbf{f} + \mathbf{n}. \tag{1}$$

Here we used the following notation:

- $\mathbf{I}$ — observed blurred image;
- $\mathbf{L}$ — clear latent image we want to find;
- $\mathbf{f}$ — kernel (PSF), could be known (non-blind deconvolution) or not (blind deconvolution);
- $\mathbf{n}$ — additive noise, some of its properties are usually known.

Point spread function describes the response of an imaging system to a point source or point object. PSF is usually assumed to be shift-invariant (every voxel is convolved with the same kernel). Such assumption is widely used in practice because it leads to more stable and efficient algorithms. We will follow this assumption, but for many practical cases it is not sufficiently adequate and should be utilized with care.

Note that equation 1 is a Fredholm integral equation of the first kind. It is an ill-posed problem, and some prior knowledge is needed to regularize it.

2 METHODS OVERVIEW

3D deconvolution necessarily involves processing huge amounts of data, which places strong constraints on the

methods' computational demands. This has led to a wide usage of simple (and thus computationally cheap) methods.

So far, the research has been focused on non-blind deconvolution. Same as in 2D, the most commonly used methods are Richardson-Lucy algorithm and Wiener filter [Krishnamurthi et al. 2010]. Additionaly, a new cryo-imaging oriented Next-image algorithm was proposed [Steyer et al. 2009].

Wiener filtering is a well-known approach to deconvolution of noisy images coming from signal processing theory [Gonzalez and Woods 2006]. It works in the frequency domain of equation 1, attempting to minimize the impact of deconvoluted noise at frequencies which have a poor signal-to-noise ratio. It is a non-iterative algorithm, implementing an expicit formula.

The Richardson-Lucy algorithm (also known as Lucy-Richardson deconvolution) is an iterative procedure recovering original image under the assumption that the noise is Poisson-distributed [Richardson 1972], [Lucy 1974]. It has been shown that if this algorithm converges, it converges to the maximum likelihood solution.

Next-image algorithm avoids solving deconvolution problem altogether, turning instead to modeling light propagation in tissue directly [Steyer et al. 2009]. Using a crude approximation and a set of physically measured or semi-automatically estimated parameters, the derived scheme is very simple. It essentially boils down to going through the data, blurring each layer to estimate its contribution to subsurface fluorescence, and subtracting it from the consequtive layers.

While data volume for 3D problems seems to limit the complexity of applicable algorithms, we argue that it can still be handled by more general approaches. For most applications, one can use much more rigid kernel priors than in 2D; furthemore, PSF is often known up to a low-dimensional set of parameters. This can be utilized to make the problem tractable by the more complex mordern MAP-algorithms.

3 PROPOSED ALGORITHM

3D deconvolution is a challenging problem and thus recieved modest development. Meanwhile, methods for 2D image deblurring were developing rapidly during the last decade. Exponential growth of the available computational power seems to have led to the current capabilities of computers being high enough so that we can apply more efficient deblurring techniques from this closely related field of research.

We will combine and adapt several modern approaches to 2D deconvolution to construct a 3D blind deconvolution algorithm.

3.1 Blind deconvolution method

Many modern 2D blind deconvolution algorithms are based on approximating maximum aposteriori probability (MAP) solution:

$$(\mathbf{L},\mathbf{f}) = \arg\max \log p(\mathbf{L},\mathbf{f}|\mathbf{I}). \tag{2}$$

While applying MAP framework, one must construct probabilistic model for the process of image convolution:

$$p(\mathbf{L},\mathbf{f}|\mathbf{I}) \propto p(\mathbf{I}|\mathbf{L},\mathbf{f})p(\mathbf{L})p(\mathbf{f}). \tag{3}$$

Each factor in the right part of the model should be chosen with great care, and many variations have been proposed. Very little is known about about appropriate priors for 3D images, so we based our model on some of the more robust 2D priors.

Once the model is fixed, the problem of blind deconvolution is solved by an iterative EM-like procedure. Before the first iteration, PSF is initialized with some realistic values. Then, we alternate between these two steps till convergence:

- Fix kernel $\mathbf{f}$ and solve non-blind deconvolution problem:

$$(\mathbf{L}^{\mathbf{new}}|\mathbf{f}) = \arg\max \log p(\mathbf{L}|\mathbf{f},\mathbf{I}). \tag{4}$$

- Fix latent image estimate $\mathbf{L}$ and update kernel $\mathbf{f}$:

$$(\mathbf{f}^{\mathbf{new}}|\mathbf{L}) = \arg\max \log p(\mathbf{f}|\mathbf{L},\mathbf{I}). \tag{5}$$

3.2 Estimation of the latent image

To calculate optimal clear image $\mathbf{L}^{\mathbf{new}}$ according to suggested model, we must choose each factor in the probability model. Basically, we use the approach of [Shan et al. 2008] with the exception of the kernel prior. To simplify the optimization process, following [Cho and Lee 2009], we opted for L_2 regularization instead of L_1.

By taking the negative logarithm of the aposteriori probability $p(\mathbf{L},\mathbf{f}|\mathbf{I})$, we restate the probability maximization problem as an energy minimization problem, defining energy $E(\mathbf{L},\mathbf{f}) = -\log p(\mathbf{L},\mathbf{f}|\mathbf{I})$. Thus, the appropriate target function can be formulated as follows:

$$\begin{aligned} E(\mathbf{L},\mathbf{f}) \propto & \left(\sum_{\partial^* \in \Theta} w_{\kappa(\partial^*)} \|\partial^*\mathbf{L} \otimes \mathbf{f} - \partial^*\mathbf{I}\|_2^2 \right) + \\ & + \sum_{\nu \in \{x,y,z\}} \lambda_1 \|\Phi(\partial_\nu \mathbf{L})\|_1 + \\ & + \lambda_2 \Big(\|\partial_\nu \mathbf{L} - \partial_\nu \mathbf{I}\|_2^2 \circ \mathbf{M} \Big) + \|\mathbf{f}\|_2. \end{aligned} \tag{6}$$

Here $M = \{m_i\}$ is an estimate of the local smoothness map with $\circ$ denoting masking (element-wise multiplication); Θ is the set of partial derivative operators: $\Theta = \{\partial_0, \partial_x, \partial_y, \partial_z, \partial_{xx}, \partial_{xy}, \partial_{xz}, \partial_{yy}, \partial_{yz}, \partial_{zz}, \}$, where ∂_0 returns pixel intensities of an image: $\partial_0\mathbf{I} = \mathbf{I}$. Θ is used to regularize the mismatch between the observed blurred image ($\mathbf{I}$) and estimated latent image convolved with PSF ($\mathbf{L} \otimes \mathbf{f}$). Using second order derivatives was proposed by [Shan et al. 2008] as a means to improve the robustness of the algorithm. Φ is a function specifically designed to penalize unrealistic distribution of gradients in the estimated image.

Even after significant simplification, target function of equation 6 is still highly non-convex and has thousands to millions of variables. In order to optimize it efficiently, authors of [Shan et al. 2008] propose a variable substitution scheme as well as an iterative parameter re-weighting technique. The basic idea is to separate the complex convolutions from other terms so that they can be computed using Fourier transform. Transferring the approach to 3D, we get the following two-step scheme for the inner iteration process of computing the new estimate $\mathbf{L}^{new}$:

- Fix $\mathbf{L}^{new}$ (initialize it with $\mathbf{I}$ before the first iteration) and find the pseudogradients $\Psi = \{\psi_i\}$ of $\mathbf{L}^{new}$ that minimize energy:

$$\begin{aligned} E'_{\psi_{i,\nu}} = & \lambda_1 |\Phi(\psi_{i,\nu})| + \lambda_2 m_i(\psi_{i,\nu}) - \\ & - \partial_\nu \mathbf{I}_i^2 + \gamma(\psi_{i,\nu} - \partial_\nu \mathbf{L}_i^2). \end{aligned} \tag{7}$$

This energy can be optimized very fast, independently for each pixel i and for each partial derivative $\partial\nu \in \partial\{x,y,z\}$.

- Fix the gradient estimation Ψ and update the latent image $\mathbf{L}^{new}$:

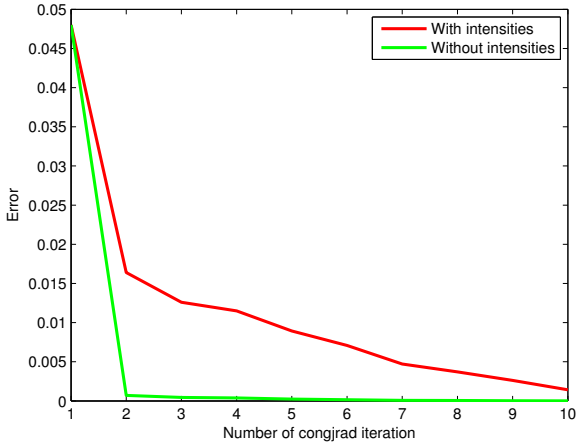


Figure 2: Excluding intensities from the energy improves the optimization convergence.

$$\mathbf{L}^{\mathbf{new}} = \\ = \mathcal{F}^{-1}\left(\frac{\overline{\mathcal{F}(\mathbf{f})}\circ\mathcal{F}(\mathbf{I})\circ\Delta+\gamma\sum\limits_{\nu\in\{x,y,z\}}\overline{\mathcal{F}(\partial_\nu)}\circ\mathcal{F}(\Psi_\nu)}{\overline{\mathcal{F}(\mathbf{f})}\circ\mathcal{F}(\mathbf{f})\circ\Delta+\gamma\sum\limits_{\nu\in\{x,y,z\}}\overline{\mathcal{F}(\partial_\nu)}\circ\mathcal{F}(\partial_\nu)}\right). \quad (8)$$

Here $\mathcal{F}, \mathcal{F}^{-1}$ denote direct and inverse Fourier transform, respectively. $\Delta = \sum_{\partial^* \in \Theta} w_{\kappa(\partial^*)}\overline{\mathcal{F}(\partial^*)}\circ\mathcal{F}(\partial^*)$, and $\overline{(\cdot)}$ is the complex conjugate operator. Division is performed element-wise.

3.3 Estimation of the kernel

We consider two types of convolution kernels — PSF with a sparsity prior only (commonly used in 2D deblurring), and a parametric set of kernels — cheaper and more robust approach.

3.3.1 Unconstrained PSF

Method used in [Shan et al. 2008] for kernel estimation (based on the interior point method) is extremely slow, and even for 2D images it suffers from memory overhead. Consequently, we decided to use the approach of [Cho and Lee 2009], that is much more efficient.

When $\mathbf{L}$ is fixed, energy in eq. 6 is simplified:

$$E(\mathbf{f}) \propto \left(\sum_{\partial^* \in \Theta} w_{\kappa(\partial^*)}\|\partial^*\mathbf{L}\otimes\mathbf{f} - \partial^*\mathbf{I}\|_2^2\right) + \|\mathbf{f}\|_2; \quad (9)$$

Optimization of this energy is a quadratic programming problem:

$$\left(\sum_{\partial^* \in \Theta} w_{\kappa(\partial^*)}||\partial^*\mathbf{L}\otimes\mathbf{f} - \partial^*\mathbf{I}||_2\right) + ||\mathbf{f}||_2 \rightarrow \max_{\mathbf{f}} \\ f_i \geq 0 \\ \sum_i f_i = 1 \quad (10)$$

If we relax the problem, ignoring constrains on $\mathbf{f}$, we can find solution analytically, using Fourier transforms. Doing so would require inverting the corresponding matrix, which in this case is a computationally expensive and unrobust operation. Instead, following [Cho and Lee 2009], we employ conjugate gradients (CG) method to optimize 9. Recent analysis in [Levin et al. 2009; Levin et al. 2011] also indicates the energy in eq. 9 should be further simplified by excluding pixel intensities, $\Theta \rightarrow \Theta' = \Theta \setminus \partial_0$. It leads to a significant improvement of conjugate gradients convergence speed and, at the same time, produces a more robust solution (for what is a typical example for our experiments, see fig. 2).

As convolution for two signals ($p\times p$ and $q\times q$) is equivalent to multiplication of two matrices ($p^2 \times q^2$ and $q^2 \times 1$), the problem can be rewritten in matrix form. Denoting matrix corresponding to the sum in 10 by A, we get the following functional to be minimized at each iteration of CG:

$$E_k(\mathbf{f}) = ||Af - b||_2 + \beta||f||_2 = \\ = (Af-b)^T(Af-b) + \beta f^T f; \quad (11)$$

To use the CG method, we also need to compute the gradient of $E_k(\mathbf{f})$:

$$\frac{dE_k(\mathbf{f})}{d\mathbf{f}} = 2A^T A\mathbf{f} + 2\beta\mathbf{f} - 2A^T b; \quad (12)$$

In general, our optimization procedure produces an unnormalized solution with negative values. So, after the optimization process is finished, we set the small values in the kernel to zero (to be precise, those smaller than $\frac{1}{20}$ of the biggest one); the remaining elements are normalized so that their sum is equal to one.

As CG can not guarantee convergence to the global optimum, the optimization is done in a pyramidal coarse-to-fine manner [Cho and Lee 2009].

3.3.2 Parametric PSF

Now, let's assume the PSF can be parameterized by a small set of variables, $\mathbf{f} = \mathbf{f}(\theta)$. While the energy can still be computed using eq. 11, the equation for the gradient changes somewhat:

$$\frac{dE_k(\mathbf{f})}{d\theta} = 2A^T A\mathbf{f}\frac{d\mathbf{f}}{d\theta} + 2\beta\mathbf{f}\frac{d\mathbf{f}}{d\theta} - 2A^T b\frac{d\mathbf{f}}{d\theta}; \quad (13)$$

Effectively, this constitutes a more robust procedure, since the dimensionality of the problem is tipically much lower and, even more importantly, the kernel is guaranteed to be normalized and have no negative elements.

3.4 Implementation details

We have implemented the algorithms described in previous section for both 2D and 3D using MATLAB. The implementation was evaluated on model data and real images. The first part of the proposed method is non-blind deconvolution, so we used the standard MATLAB algorithms for evaluation.

Also, we tried two different configurations of coarse-to-fine pyramid: 1) to downsample both the kernel and the images, and 2) to downsample the kernel only (while applying conjugate gradients). The second method is simpler, but the first one often produces a better solution, especially for the coarsest approximations.

Note that working with the downsampled images demands an appropriate scaling of the non-blind algorithm's parameters (otherwise, significant artifacts in early levels of the pyramid will result in a grossly suboptimal solution).

4 EXPERIMENTAL RESULTS

We have tested our algorithm on model and real data.

The model 3D scene consists of numerous tightly packed small spheres. Each sphere is a dense intensity source, with luminocity peaking in the center and falling to about half

of that towards the edges. Positions and densities of the spheres are sampled from a normal distribution. Fig. 3 (upper left) shows an example of our model scene.

In our experiments, we used a simple kernel parametrization, a gaussian multiplied by a hyperbola. This PSF that can be scaled in each dimension (and it fits well with our real data):

$$f_{i,j,k}(h,w,d) = \frac{h-k}{h} * \mathcal{N}([w,d]^T, \frac{w}{5}); \qquad (14)$$

The only parameters are $\{h,w,d\}$ — height, width and depth of the kernel, accordingly. The PSF has a paraboloid-like shape, with most of the density concentrated around the axis. Example of our PSF can be seen on fig. 3 (upper right).

After applying the convolution, many of the spheres stick together and cease to be discrete objects. This replicates the effect subsurface fluorescence has on block-face imaging: individual cells can no longer be easily discerned as such, thus decrising the resolution (resolving power) of the technique. Fig.3 (lower left) shows the model scene after applying convolution with the kernel.

Fig.3 (lower right) shows the output of the proposed non-blind deconvolution method. Even the more closely seeded spheres have been been succesfully separated. The estimation of the densities is also pretty close to the original.

Fig.4 compares the result of non-blind and fully blind (no kernel parameterization information) deconvolution schemes. The source image and the kernel used for blur are the same in both cases. Note that although the reconstructed shapes are very similar, non-blind deconvolution did a much better job at estimating densities.

When parameters of the kernel are taken into account, blind deconvolution algorithm finds a good estimate for the PSF in just a few steps of CG. For the model data, the results are visually indistinguishable from those of non-blind deconvoluton (because of the space limitations, we omit them here).

Fig.1 shows an iso-surface rendering of a fragment of a 3D map of c-Fos expression in an adult mouse brain, before and after subsurface fluorescence removal procedure; inset are 2D section of the same data. Kernel parameterization is the same as above. The proposed method clearly results in a significant improval of the quality of the scan.

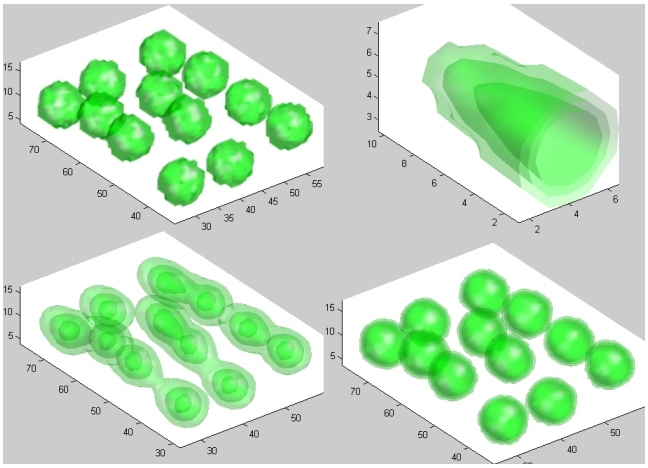


Figure 3: Non-blind deconvolution result. From left to right, from top to bottom: sharp image, convolution kernel, blurred image, restored image

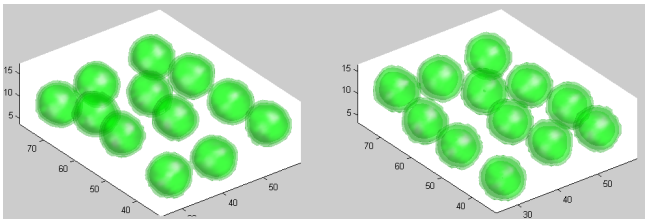


Figure 4: Non-blind (left) versus blind deconvolution.

References

Cho, S., and Lee, S. 2009. Fast motion deblurring. *ACM Transactions on Graphics (SIGGRAPH ASIA 2009) 28*, 5, article no. 145.

Gonzalez, R. C., and Woods, R. E. 2006. *Digital Image Processing (3rd Edition)*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

Krishnamurthi, G., Wang, C. Y., Steyer, G., and Wilson, D. L. 2010. Removal of subsurface fluorescence in cryo-imaging using deconvolution. *Opt. Express 18*, 21 (Oct), 22324–22338.

Levin, A., Weiss, Y., Durand, F., and Freeman, W. T. 2009. Understanding and evaluating blind deconvolution algorithms. *IEEE CVPR* (June).

Levin, A., Weiss, Y., Durand, F., and Freeman, W. T. 2011. Efficient marginal likelihood optimization in blind deconvolution. *IEEE CVPR*.

Lucy, L. B. 1974. An iterative technique for the rectification of observed distributions.

Richardson, H. W. 1972. Bayesian-Based Iterative Method of Image Restoration. *Journal of the Optical Society of America 62*, 1 (Jan.), 55–59.

Roy, D., Gargesha, M., Steyer, G. J., Hakimi, P., Hanson, R. W., and Wilson, D. L. 2010. Multiscale characterization of the pepck-cmus mouse through 3d cryo-imaging. *Journal of Biomedical Imaging 2010* (January), 5:1–5:11.

Shan, Q., Jia, J., and Agarwala, A. 2008. High-quality motion deblurring from a single image. *ACM Transactions on Graphics (SIGGRAPH)*.

Steyer, G. J., Roy, D., Salvado, O., Stone, M. E., and Wilson, D. L. 2009. Removal of out-of-plane fluorescence for single cell visualization and quantification in cryo-imaging. *Annals of Biomedical Engineering 37*, 8, 1613–1628.

AUTHORS

- Andrey Tikhonov is a fifth year student of the faculty of Computational Mathematics and Cybernetics in Lomonosov Moscow State University.
- Pavel Voronin is a researcher in Kurchatov NBIC Centre, National Research Centre "Kurchatov Institute", Moscow.

Several approaches for improvement of the Direct Volume Rendering in scientific and medical visualization

Nikolay Gavrilov, Alexandra Belokamenskaya, Vadim Turlapov
Laboratory of Computer Graphics
Lobachevsky State University of Nizny Novgorod, Russia
{gavrilov86, alexandra.belokamenskaya, vadim.turlapov}@gmail.com

Abstract

This paper presents Direct Volume Rendering (DVR) improvement strategies, which provide new opportunities for scientific and medical visualization which are not available in due measure in analogues: 1) multi-volume rendering in a single space of up to 3 volumetric datasets determined in different coordinate systems and having sizes as big as up to 512x512x512 16-bit values; 2) performing the above process in real time on a middle class GPU, e. g. nVidia GeForce GTS 250 512 MB; 3) a custom bounding mesh for more accurate selection of the desired region in addition to the clipping bounding box; 4) simultaneous usage of a number of visualization techniques including the shaded Direct Volume Rendering via the 1D- or 2D- transfer functions, multiple semi-transparent discrete iso-surfaces visualization, MIP, and MIDA. The paper discusses how the new properties affect the implementation of the DVR. In the DVR implementation we use such optimization strategies as the early ray termination and the empty space skipping. The clipping ability is also used as the empty space skipping approach to the rendering performance improvement. We use the random ray start position generation and the further frame accumulation in order to reduce the rendering artifacts. The rendering quality can be also improved by the on-the-fly tri-cubic filtering during the rendering process. Our framework supports 4 different stereoscopic visualization modes. Finally we outline the visualization performance in terms of the frame rates for different visualization techniques on different graphic cards.

Keywords: *GPGPU, ray casting, direct volume rendering, medical imaging, empty space skipping, section by arbitrary mesh.*

1. INTRODUCTION

In scientific visualization it is often necessary to deal with some volumetric regular scalar datasets. These data may be obtained by some numerical simulation or via the scanning equipment such as tomographs. The output data of the CT scan is a series of the slices, i.e. two-dimensional scalar arrays of 16-bit integer values. The stack of such slices can be interpreted as a volumetric dataset which can be visualized as a 3D object.

Since the 90s, the Direct Volume Rendering shows itself as an efficient tool for the visual analysis of volumetric datasets [1-4]. Different established approaches [4] make possible the implementation of the real-time volume rendering by using the parallel and high-performance computations on the GPU. The recent progress in GPGPU computations makes the real-time multi-volume rendering possible [1]. In this paper we make a review of general details of our Ray Casting implementation, which allows for the performance and quality improvement of the available visualization methods.

1.1 The framework development motivation

There was a need in a volumetric visualization of the molecular dynamic simulation of the electron bubble transport process. There was a number of different volumetric datasets being obtained during this simulation, representing different scalar and vector fields. These datasets should be visualized together, in a single space.

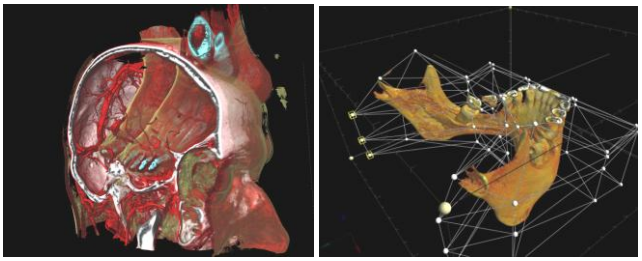


Figure 1: Features that we have implemented in our visualization software: the multi-volume rendering (left); volumetric clipping via the custom polygonal mesh (right).

However, there was no scientific visualization software available, which could satisfy our needs in the visual analysis of the scientific multi-volume time-varying volumetric datasets we had.

We wanted to be able to add several desired specific features in our own visualization framework. For instance, the interactive volumetric section via the arbitrary polygonal mesh can be efficiently used for the interactive and suitable selection of the region of interest, and we have not found any solutions with this feature. There is no multi-volume rendering software available as well, so its development is still in the research domain. Below we make a review of some significant visualization software solutions we have encountered.

The Voreen (http://www.voreen.org/) is an open source volume rendering engine, which can be nicely used for some scientific datasets visualization and the rendering quality is good enough. But when dealing with some big dataset (e.g. 512^3 cells) it appears to be hardly a real-time visualization. Moreover, there is no support for the multi-volume datasets rendering if we want to render several datasets together in one space.

The OsiriX (http://osirix-viewer.com) viewer is designed for the medical staff and has a lot of features, but it is the MacOs-only software. And while it is the medical imaging software, it can be hardly used for the scientific data visualization, because the user may want to obtain some specific data visual representation.

The Fovia's (http://fovia.com) CPU-based Ray Casting efficiency shows that even such GPU-suitable algorithms, like the Ray Casting, are still may be implemented fully on the CPU with the same or better results. However, while the graphical cards' performance, availability and architecture flexibility increases, the CPU-based efficient Ray Casting implementation is a very difficult task. Besides, the Fovia has no the desired features like multi-volume rendering and polygonal mesh based volumetric clipping.

2. METHODS AND ALGORITHMS

In this section we make a brief inspection of the Ray Casting and some other algorithms we have implemented in our system. All of these algorithms and are implemented as the GLSL shaders.

2.1 Rendering methods

Due to the high flexibility of the Ray Casting (RC) method there is a huge amount of different possible visualization techniques. The RC algorithm calculates each pixel of the image by generating (casting) rays on the screen plane and traversing them towards the observer viewing direction. Each of the RC-based rendering algorithms takes the ray start position and its direction as the input parameters, and the pixel color as the algorithm output.

There are six rendering techniques in our framework and each of them supports multi-volume rendering, i.e. these algorithms can handle several volumetric datasets, which are arbitrary located in the world space. The dataset's location is determined by the transformation 3x4 matrix. In addition to the volume's position and orientation, this matrix also defines the spacing by x, y and z components, which allows for the proper volume scaling. So to perform a sampling from the arbitrary located dataset we multiply the sampling point by this matrix, so we will get the sampling point in the volume's local coordinate system.

In order to perform the GPU-based Ray Casting, we use GLSL shaders to calculate the screen plane's pixel colors, like it is done in the Voreen framework. Each of the rendering methods in our framework is defined by some GLSL fragment shader program, i.e. a text file with the GLSL code. These methods differ from each other, but they uniformly handle all visualization parameters, e.g. the observer position, transfer functions, iso-surfaces, anaglyph matrix, etc. So it is easy for us to add new RC-based rendering techniques in our visualization framework.

2.1.1 Maximum Intensity Projection (MIP)

The Maximum Intensity Projection (MIP) is one of the most usable volumetric visualization modes in the medical imaging [3]. It is easy for clinicians to interpret an MIP image of blood vessels.

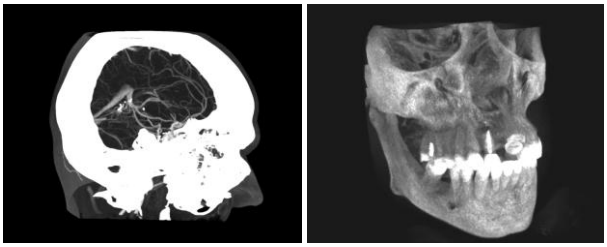


Figure 2: MIP images of the brain vessels (left) and the dental datasets.

The maximal intensities of the volumetric dataset are projected onto the screen plane. Usually the projected intensity determines the pixel's color in the gray-scaled manner (Figure 2). It medical imaging there is a window/level concept. The window is represented by two scalar values – the window width and window center.

2.1.2 The shaded Direct Volume Rendering (sDVR)

The shaded DVR technique is also used in a medical exam, but its usage is more limited [2]. In contrast to the MIP technique here we can use the early ray termination approach, which considerably improves the rendering performance without any image visible changes. This termination can be done because of the DVR algorithm nature – while traversing throw the volume data, the ray accumulates color and opacity, i.e. the optical properties, defined by the transfer function (TF). While the opacity increases, the contribution to the final pixel's color decreases. This opacity accumulation is commonly used to visualize the data as a realistic volumetric object.

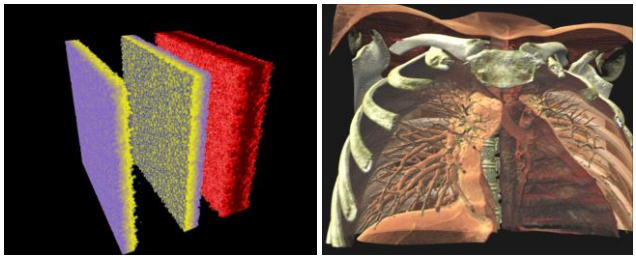


Figure 3: The scientific (left) and medical (right) data visualization by the DVR and sDVR techniques.

2.1.3 Semi-transparent iso-surfaces

Semi-transparent iso-surfaces are usually used for the scientific data visualization, because these surfaces are easier to interpret when examining some new phenomena. One of the main advantages of this method over the DVR's one is a surface's visualization high quality: here we can search for more exact intersection with the surface, a so called Hitpoint Refinement [5], while in DVR there are no surfaces. There are opaque regions in DVR, but theirs accurate visualization requires more difficult algorithm, and its laboriousness considerably reduces the performance of the GPU-implementation.

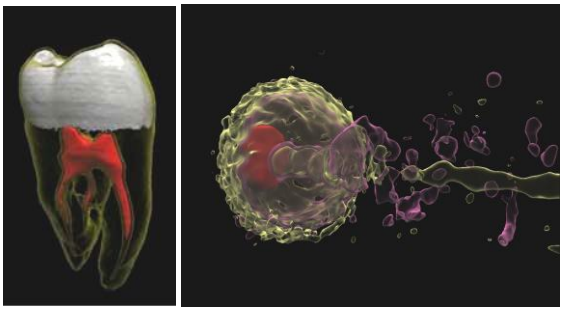


Figure 4: Semi-transparent iso-surfaces via the Ray Casting algorithm. The CT tooth dataset (left) and the simulated electron bubble transport process (right).

This method searches for ray's intersections with the iso-surfaces. When the ray passes across the iso-value, we calculate a more exact intersection point determined by the ray positions and sampled values on the current and previous steps. The linear interpolation is enough to obtain a desired result.

2.2 Optimization strategies

2.2.1 Early ray termination

The Early ray termination technique is a common optimization strategy for a Ray Casting algorithm. It is possible to terminate the RC algorithm for each individual ray if the accumulated opaqueness is close to 1. However, in rendering techniques like the MIP it is necessary to browse the whole ray path until leaving the bounding volume.

2.2.2 Empty space skipping via the volumetric clipping

The custom bounding polygonal mesh can be used either for the rendering acceleration [5] or for the data clipping, as an alternative to the dataset's segmentation. We use OpenGL frame buffers to store the distances from the viewpoint to the mesh front and back faces. The buffers may contain up to 4 ray path segments (in [5] there is only one segment), so that the mesh is not required to be convex. Of course, before using these buffers we should fill them with some relevant data. To do it we draw the bounding mesh by calling common OpenGL instructions and perform rendering into the texture. We use a specific GLSL shader program to fill pixels with the relevant data instead of the standard OpenGL coloring. The program calculates the current distance between the observer and the fragment position. The only varying parameter is the fragment position, i.e. the point on the mesh surface. After the buffers are filled with the relevant distances, the Ray Casting may be performed (Figure 5). For each ray it is known, what segments of the ray path should be traversed.

2.3 Rendering artifacts reduction

Because of the finite steps' number the ray may skip some meaningful features in the dataset, even if the ray step is much less than the voxel's size. As a result the 'wood-like' image artifacts may appear. This wood-likeness appears because each ray starts from the same plane (e.g. from the bounding box face). This artifacts' regularity can be removed by randomization of the ray start positions. The final image will contain 'noisy' artifacts instead of 'regular' ones. If a user does not change the viewpoint and other visualization settings, these random frames can be accumulated by such a way that a user will see an average image that contains no noise (Figure 8).

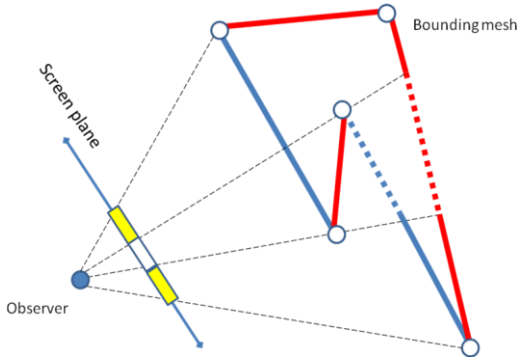


Figure 5: The illustration of the volumetric clipping algorithm. The yellow regions of the screen plane identify pixels, where are only one path segment inside the bounding mesh. In the white region there are two path segments for the rays in the Ray Casting algorithm.

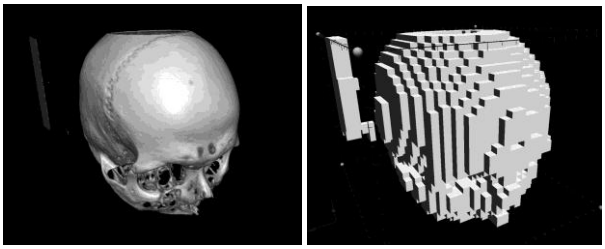


Figure 6: The source dataset (left) and its acceleration structure (right) visualization. Visible 'bricks' identify regions of interest that contain some visible features.

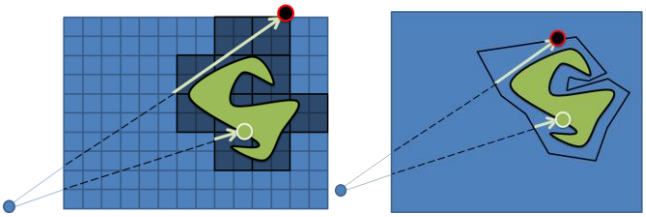


Figure 7: Empty space skipping: via the regular acceleration structure (left); via the bounding mesh (right).

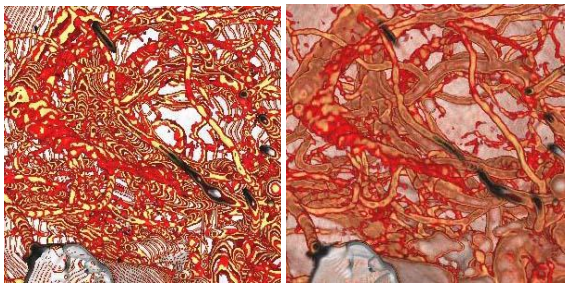


Figure 8: Before (left) and after (right) the DVR 'wood-like' artifacts' reduction by the frames accumulation approach.

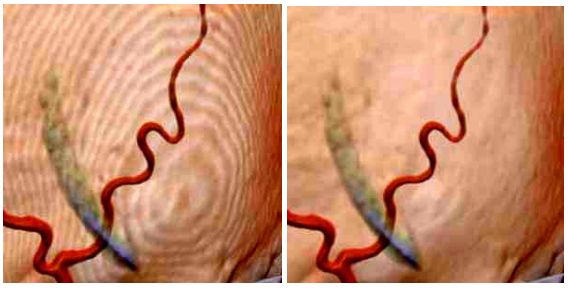


Figure 9: Tri-linear (left) and tri-cubic (right) on-the-fly samplings for the Ray Casting algorithm.

The visualization quality can be improved also via the tri-cubic filtering instead of the common tri-linear one. We have extended the algorithm for the two-dimensional case, presented in [7]. Considering the embedded bi-linear texture filtering ability, we can make only four samplings to calculate the bi-cubic interpolated value. In order to perform a single tri-cubic sampling it is necessary to make 8 basic tri-linear samplings from the same dataset, but in fact we will calculate the data value, determined by 64 nearest voxels' values.

3. PERFORMANCE ACCELERATION RESULTS

In table1 we have outlined obtained rendering performance improvement by using the space skipping technique based on the clipping via bounding polygonal mesh (Figure 6). The Acceleration 1 and 2 in table1 mean that tri-linear and tri-cubic on-the-fly samplings were used in experiments.

Data	Size	Acceleration 1	Acceleration 2
Head	512^3	1.6	3
Dental	512x512x331	1.7	3.2
Feet	512x512x250	2	3.6

Table 1: Rendering performance improvement by volumetric polygonal clipping approach.

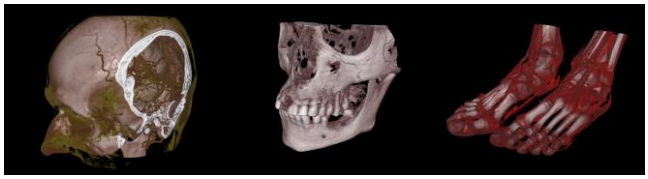


Figure 10: Test CT datasets (left to right): head, dental, feet.

4. CONCLUSIONS

The proposed framework can be used for the multi-modal medical data examination. A user can load datasets of the CT and MRI modalities as DICOM slices. Each dataset is determined by its own coordinate system and a user can properly place datasets together in the space via the control points, so that the user sees the comprehensive 3D-picture of the medical examination. The real-time rendering is performed on consumer graphic cards.

The framework is also good for stereo demonstrations via the stereo-pare of projectors, anaglyph, interlaced rendering or stereo-monitors (like Zalman with passive polarized glasses) or with shutter glasses via the nVidia 3D Vision technology.

The volumetric clipping option is performed via the arbitrary polygonal mesh. The triangles number is not sufficient and does not impair the rendering performance, so the mesh may have a rather complex structure and may be fitted to the visible features in the space. The clipping may be used either for the performance improvement or for the interactive manual data segmentation in order to select the volumetric region of interest. We define the bounding mesh as the triangulated visible cells of the regular acceleration structure. This acceleration technique improves the rendering performance up to four times.

5. AKNOLEDGMENTS

This work was supported by the federal target program "Research and scientific-pedagogical cadres Innovative Russia" for 2009-2013, state contract № 02.740.11.

6. REFERENCES

[1] Kainz B. et al, 2009. Ray Casting of Multiple Volumetric Datasets with Polyhedral Boundaries on Manycore GPUs. Proceedings of ACM SIGGRAPH Asia 2009, Volume 28, No 152.

[2] Lundström C., 2007. Efficient Medical Volume Visualization: An Approach Based on Domain Knowledge. Linköping Studies in Science and Technology. Dissertations; No. 1125.

[3] Geoffrey D., 2000. Data explosion: the challenge of multidetector-row CT. In IEEE Transactions on European Juornal of Radiology. Vol. 36, Issue 2, pp 74-80.

[4] Klaus E. et al, 2004; *Real-Time Volume Graphics*, A.K. Peters, New York, USA.

[5] Scharsach H., 2005. Advanced GPU raycasting. In Central European Seminar on Computer Graphics, pp. 69–76.

[6] Gordon L., 1999. Semi-automatic generation of transfer functions for direct volume rendering. A Thesis Presented to the Faculty of the Graduate School of Cornell University in Partial Fulfillment of the Requirements for the Degree of Master of Science.

[7] Daniel R. et al, 2008. Efficient GPU-Based Texture Interpolation using Uniform B-Splines. Journal of Graphics, GPU, & Game Tools, Vol. 13, No. 4, pp 61-69.

[8] Grimm S. et al, 2004. Memory Efficient Acceleration Structures and Techniques for CPU-based Volume Raycasting of Large Data. Proceedings of the IEEE Symposium on Volume Visualization and Graphics 2004. pp 1 – 8.

Visualization of Simulated 3d–Geometry in Transfer Equation Solution Problems Using Monte-Carlo Technique

Elena Klass, Sergey Ulyanov
Central Research and Development Institute of Chemistry and Mechanics
Moscow, Russia
elenaklass@yandex.ru, ulyanov.sergey@gmail.com

Abstract

Standard 3d-visualization programs assume object surface representation as a set of polygons. When the ionizing radiation transfer equation is solved in 3d-geometry using Monte-Carlo technique an object geometry is usually introduced as a set of second order surfaces. The current work is focused on 3d modeling of an object, where the second order surfaces are visualized using marching cubes algorithm, representing the surfaces as a set of cubic blocks. The results of the proposed method application for visualization of a specific simulated geometry are presented.

Keywords: *3d-visualization, second order surfaces, quadrics, polygonization, triangulation.*

1. INTRODUCTION

Originally, the practice of geometrical modeling applied to solving transfer equations using Monte-Carlo technique, has adopted an approach that utilized second order surfaces and body-primitives to represent real object designs. [1, 3-5, 9]. To increase the accuracy of geometry description the simulation model can be represented also by a big number of voluminous cubic elements (voxels) with a size of some mm^3. For example, the problems of radiation protection are actively dealt with using mathematical models describing a human body by a set of cubic blocks, based on computer tomography data [7, 10]. Similar geometry representation is applied as well in CAD-systems. While solving transfer problems using Monte-Carlo technique the principle drawback of the voxel representation of an object under study is that geometrical models consisting of numerous cells require significant increase in run time. This is connected with peculiarities of particle tracking using the method of Monte-Carlo. In particular, after generation of an interaction point and the particle free path length it is necessary to search for the cell boundary to be crossed by the generated track. After the boundary is found similar search is undertaken for the next cell, crossed by the track. If the cell size is small and the track length is big the run time to model one trajectory explodes by orders of magnitude. Due to that in practice an approach is preserved based on a simplified representation of object geometry by second order surfaces.

The validation of the geometrical description is performed by special programs. Standard approach to monitor the geometry is to visualize 2d cross sections parallel to the reference system axes. [4]. In recent years visualization packages in 3d have been produced for the geometrical module of Monte-Carlo programs MCNP5 and MCNPX [3, 11, 12], using ray tracing as well as 3d geometry in the form of second order surfaces. This makes it possible to obtain more obvious idea of the object model. The goal of the current work was to create an algorithm, making it possible to produce a 3d object model for the geometrical module of national program "РОБОТ" [14].

2. METHOD OF GEOMETRICAL MODEL REPRESENTATION IN "РОБОТ" PROGRAM

To introduce the problem geometry a user [14] performs division of the object and surrounding space by so called zones. A zone is a region in space, limited by first and second order surfaces. Necessary condition here is that the zone should not be crossed by surfaces, which elements constitute its boundaries. The zone is called simple if under the problem conditions it can be considered uniform in composition and compound, if this requirement is not met. The division by zones is performed in sequential allocations of regions in space using the above mentioned surfaces specifying the surface sign. The following parameters are introduced in the main geometrical array for each zone:

- numerical ID of the surface equation with corresponding sign
- material ID of the zone
- IDs of all equations, limiting the zone and ID of the equation determining the zone.

The latter equation ID list is necessary to estimate path lengths from interaction points up to the zone boundaries. All geometrical zones, identified in the first stage, are described similarly. In the next stage the description of each zone, identified as "compound" or "composite", is performed. The division is continued until all the simulated region is covered with simple zones. Description principle of geometrical zones is maintained during all stages. The final geometrical zones description array is composed of a series of sequential division arrays of the simulation region.

In the process of the problem solution in order to find which zone contains particular point its coordinates (x,y,z) are sequentially introduced in equations, determining 1^{st}, 2^{nd}, 3^{d}, etc. zones accounting for sign of the equation. Point (x,y,z) belongs to zone v, described by equation j if the following condition is true:

$$sign(v_j)(c_{1j}x^2+c_{2j}y^2+c_{3j}z^2+c_{4j}xy+c_{5j}xz+c_{6j}yz+c_{7j}x+c_{8j}y+c_{9j}z+c_{10j}) > 0,$$

where $\mathrm{sign}(v_j)$ is the sign of equation, describing the zone. In case the point appears to belong to a compound zone the search is continued starting with its first subzone. As a results, the geometrical module contains points in space with corresponding two numbers: material ID and zone ID.

3. GEOMETRICAL OBJECT 3D-MODELING ALGORITHM

As specified above, each geometrical region in the object description follow the format of РОБОТ program using first and second order boundary equations and represents a connected set. The proposed visualization algorithm depends on the complexity of the surface equations bounding the zone. If the zone is bounded by first order equations polygonization of the zone is performed. If second order equations exist in the description, triangulation of the zone is done using marching cubes algorithm [6].

Visualization algorithm proposed in the current work contains the following.

3.1 Validation of geometrical zones

In the first stage the validation of the geometry of each zone is performed, including search for unbounded regions in space as well as the correctness of the defined boundaries and their signs. The zone geometry is considered to be defined correctly if the space, bounded by first and second order surfaces, is closed, i.e. represents some volume. For this purpose during processing (sequential or parallel) of each zone a search problem is solved of minimization of the enclosing rectangular parallelepiped. This subproblem is reduced to search for the extrema in the degrees of freedom, satisfying the inequality system of the surface equations, Such problem statement can be easily reduced to the problem of mathematical programming:

$$f(\vec{x}) \rightarrow min,\ \vec{x} \subseteq E^3,$$

subject to:

$$g_i(\vec{x}) \leq 0,\ i=[1,m],$$

where:

$\vec{x}$ – coordinates vector in space $\vec{x} = (x, y, z)^T$;

$f(\vec{x})$ – objective function, represented by three degrees of freedom $f(\vec{x}) = x$, $f(\vec{x}) = y$ and $f(\vec{x}) = z$;

$g_i(\vec{x})$ – bounding functions, corresponding to 1st and 2nd order surface equations;

m – number of boundary conditions.

As a result, to find six bounding faces of the parallelepiped it is necessary to solve the problem of mathematical programming six times in order to find global minima and maxima of each degree of freedom. Since boundary conditions are generally nonlinear functions, to solve the defined problem it is necessary to use one of the nonlinear optimization methods. In our solution we applied the flexible tolerance algorithm [15] due to the fact that this method is direct (do not use derivatives), and not very sensitive to the choice of initial feasible solution. One can utilize linear programming technique in the specific case when all boundary conditions are first order equations.

In case the described parallelepiped exists, i.e. finite limits were found for all reference system axes and it has not degenerated in a point or plane, then it is decided that the zone was defined correctly (represents a bounded region in space) and is subject to visualization (Figure 1).

3.2 Zone visualization

Based on surface types describing a specific zone it is possible to identify two major cases: only 1st order equations and combined description. Such classification can drastically increase the visualization algorithm efficiency by simplified polygonization of zones, described only by 1st order equations.

3.2.1 Polygonization of zones, defined by 1st order equations

If a zone is described only by 1st order equations, then each of its bounding surfaces corresponds to maximum one polygon, lying on its surface. For each plane, bounding a zone, points of intersections with pairs of other such planes are found. The resulting points are filtered to select those, belonging to the zone surface (i.e. satisfy unstrict inequalities, defining the zone). As a result vertex coordinates of the polygon lying on the zone surface are obtained. Then, a normal to the plane under consideration is built, directed outside (Figure 1) and obtained points are sorted anti-clockwise with respect to this normal, which is necessary to correct mapping of the polygon.

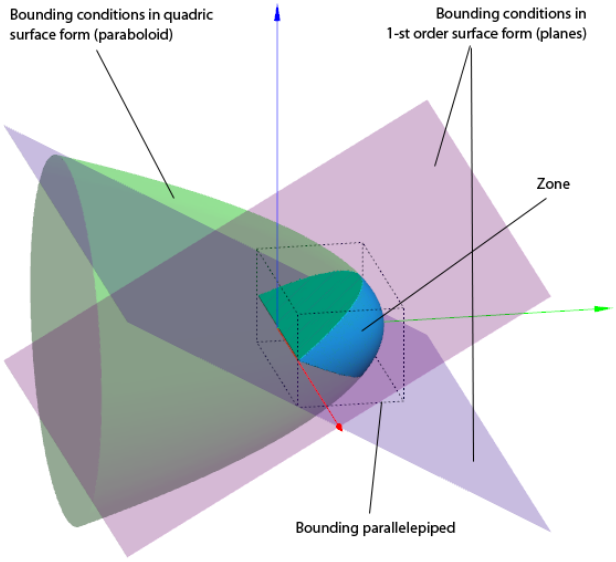


Figure 1: Zone, bounding conditions and parallelepiped example.

3.2.2 Triangulation of zones, defined by 2nd order equations

The process of visualization of the zones, defined by 2nd order equations, consists of dividing the three dimensional space into cubic blocks of predefined size, each of which experience triangulation.

To perform triangulation of 2nd order equation zones marching cubes algorithm is used [6].

It was mentioned above, that during validation of geometry definition an enclosing parallelepiped is found for each zone. In the process of zone triangulation the space bounded by its faces is split into cubes of different sizes. Then each cube is processed to find intersections with bounding surfaces of the zone.

Each cube vertex is probed for belonging to the geometrical zone under search. Eight Boolean quantities are obtained or one byte of information. This number in a pre-calculated table corresponds to a set of triangles, forming part of the image. The triangles are defined by indexes of cube edges, containing their vertexes. Precise location of each such vertex is determined by the solution of the problem of cube edge intersection with corresponding zone

surface. If one vertex of the cube edge lies within the zone and the other is outside, then it is assumed that this cube edge intersects the zone surface only ones. If both vertexes either belong or do not belong to the zone, then the intersection is missing at all. If the distance between the zone boundaries is less than the cube edge length, the zone can remain invisible. That is why when marching cubes algorithm is used the cube sizes are selected based on characteristic sizes of the geometry.

4. RESULTS

Based on the described algorithm a program application was developed, making it possible to perform validation and visualization of object geometry of defined complexity (Figure 2, 3).

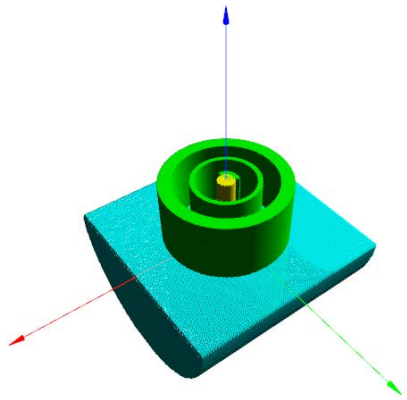


Figure 2: Example of visualized object.

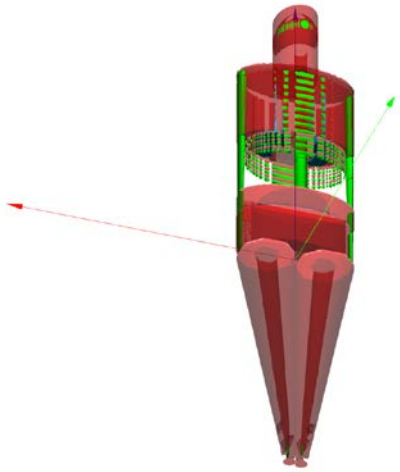


Figure 3: Visualization of the components of a heterogeneous model of the human body (MIRD-phantom [8, 13]).

5. CONCLUSION

Algorithm of 3d-visualization of geometrical file, containing object description in the form of 2nd order surfaces, widely used in radiation fields Monte-Carlo simulation programs is developed.

6. REFERENCES

[1] *GEANT. Detector Description and Simulation Tool. CERN, Geneva, Switzerland, 1993.*

[2] *Geant4 User's Documents Version: Geant4 3.2 June 2001.*

[3] *Graphical User Interface for High Energy Multi-Particle Transport. Final Report November 30th 2008 Home Page: http://www.mcnpvised.com.*

[4] *MCNP4C: Monte Carlo N-Particle Transport Code System. ORNL RSICC Computer Code Collection, CCC-700, April, 1990.*

[5] *MCNP – A General Monte Carlo N-Particle Transport Code, Version 5, X-5 Monte Carlo Team, LA-UR-03-1987, April 24, 2003.*

[6] Paul Bourke. *Polygonising A Scalar Field* http://paulbourke.net/geometry/polygonise/

[7] L. de Carlan, P. Roch, E. Blanchardon and D. Franck. *New method of voxel phantom creation: application for whole-body counting calibration and perspectives in individual internal dose assessment. Radiation Protection Dosimetry, 2005, 116(1-4):160-*164.

[8] Cristy M. *Mathematical phantoms representing children of various ages for use in estimates of internal dose. U.S. Nuclear Regulatory Commission Rep. NUREG/CR-1159 (also Oak Ridge National Laboratory Rep. ORNL/NUREG/TM-367).*

[9] Emmett M.B. *The MORSE Monte-Carlo Radiation Transport Code System. Report-ORNL-4972, 1975.*

[10] J.F. Evans, T. E. Blue, N. Gupta. *"Absorbed dose estimates to structures of the brain and head using a high-resolution voxel-based head phantom." Med Phys. 2001 May;28(5):780-6.*

[11] R.A. Schwarz, L.L. Carter, W Brown. *"Particle Track Visualization Using the MCNP Visual Editor, "Proc. Am. Nucl. Soc. Topical Radiation Protection for Our National Priorities Medicine, the Environment and, the Legacy, 324-331, 2000, Spokane, Washington.*

[12] A.L. Schwarz, R.A. Schwarz, L.L. Carter. *"3D Plotting Capabilities in the Visual Editor for Release 5 of MCNP," Nuclear Mathematical and Computational Sciences: A Century in Review, A Century Anew, April 6-11, 2003 on CD-ROM, American Nuclear Society, La Grange Park, Illinois (2003).*

[13] Snyder W.S., Ford M.R.,Warner G.G. and Fischer H.I. *MIRD. Pamphlet 5, Medical Internal Radiation Commitee, J. Nucl. Med., 1969, Suppl. 3, p. 7-12, 46-52.*

[14] Класс Е.В., Вязьмин С.О., Шаховский В.В. и др. *Состояние и возможности комплекса программ РОБОТ по расчету трехмерной защиты методом Монте-Карло. Тез. докл. // Защита от ионизирующих излучений ядерно-технических установок. ФЭИ, Обнинск, 1994. С. 106-108. (6-я Всесоюзн. конф.).*

[15] Химмельблау Д. *Прикладное нелинейное программирование, «Мир», Москва, 1974*

Optimization of Numerically Controlled Five Axis Machining Using Curvilinear Space Filling Curves

Stanislav Makhanov
School of Information and Computer Technology
Sirindhorn International Institute of Technology Thammasat University, Bangkok, Thailand
makhanov@siit.tu.ac.th

Abstract

The paper presents optimization and visualization methods for tool path planning of 5 axis milling machines. The first method is based on the grid generation techniques whereas the second method exploits the space filling curve technologies. Combination of the two techniques is superior with regard to the conventional methods and with regard to the case when the two methods are applied independently

Keywords: *CAD/CAM, CNC machines, grid generation, space filling curves .*

1. INTRODUCTION

Milling machines are programmable mechanisms for cutting industrial parts. The axes of the machine define the number of the degrees of freedom of the cutting device. Five axes provide that the cutting device (the tool) is capable of approaching the machined surface at a given point with a required orientation. The machines consist of several moving parts designed to establish the required coordinates and orientations of the tool during the cutting process (see Figure 1 and Figure 2). The movements of the machine parts are guided by a controller which is fed with a so-called NC program comprising commands carrying three spatial coordinates of the tool-tip and a pair of rotation angles needed to rotate the machine parts to establish the orientation of the tool. One may think about such a machine as a 3D plotter with five degrees of freedom.

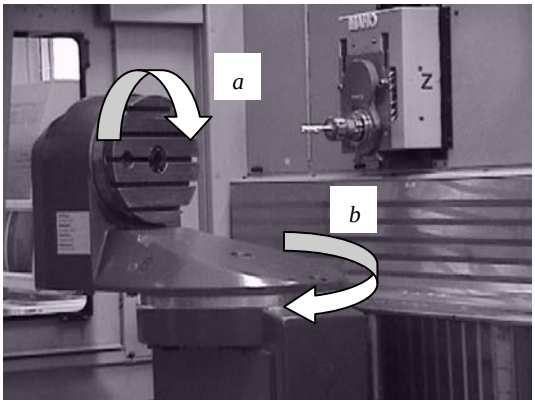


Figure 1: Five-Axis milling machines MAHO600E (Deckel Maho Gildemeister) *a* and *b* are the rotation axis

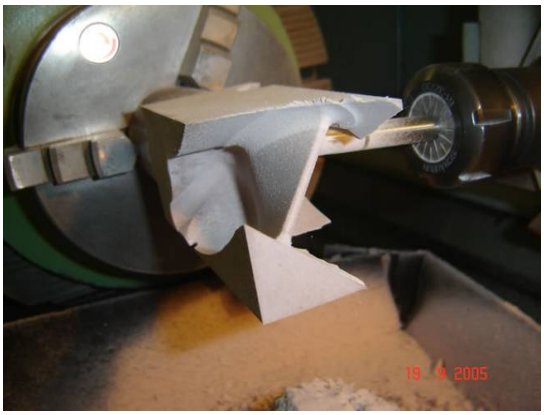


Figure 2: MAHO600E during cutting operations

The tool path is a sequence of positions arranged into a

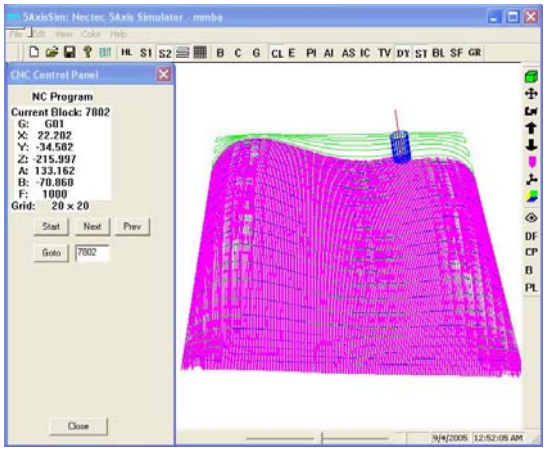


Figure 3: Zigzag tool path and the machining strips in the workpiece coordinate system

structured spatial pattern. The conventional engineering patterns are the zigzag and the spiral (Figure 3).

Tool path planning for five axis machining requires a multi criteria optimization governed by estimates of the difference between the required and the actual surface. Additionally, the criteria vector may include the length of the path, the negative of the machining strip (strip maximization), the machining time, etc

[8], [10] . Besides, the optimization could be subjected to constraints [11]

2. CURVILINEAR SPACE FILLING CURVES APPROACH

The most popular space filling curve (SFC) for tool path planning is the recursive Hilbert's curve [7] considered for numerous applications including the tool path planning [6].

Cox et al. [5] used various forms of SFCs such as the Moore's curve. However, Hilbert's curve is particularly appealing in tool path planning as its local refinement property can be used to adaptively increase the density of the path only where necessary. However, each local refinement of the tool path based on the Hilbert's curve increases the tool path density in the refined region by a factor of 2 resulting in a lower machining efficiency due to the increased total path length. Besides, the Hilbert's curve has an undesirable property that it leads to a path, where the tool is frequently changing directions which slows down the machining process and produces large kinematics errors. Adaptive SFC [1] has less turns, however, a basic rectangular grid used to construct the adaptive SFCs is often inefficient since a small step between the tracks could be required only in certain areas. The grid is also inefficient in treating complex geometries appearing in the case of the so-called trimmed surfaces having the boundaries created by intersections with other surfaces.

On the other hand the above geometrical complexities and sharp variations of the surface curvature have been proven to be successfully treated by numerically generated curvilinear zigzag tool paths obtained from adaptive grids topologically equivalent to the rectangular grids. In [9] a modification of a classic grid generation method based on the Euler-Lagrange equations for Winslow functional [4] has been adapted to the curvilinear zigzag tool path generation. The zigzag tool path is constructed by solving numerically Euler-Lagrange equations for a functional representing desired properties of the grid such as smoothness, adaptivity to the boundaries and to a certain weight (control) function [2]. A similar idea to use a Laplacian curvilinear grid was suggested independently in [3]. However, these techniques have several major drawbacks. Chief among them is slow convergence for complicated constraints. Besides, the approach requires an equal number of the cutter contact points on each track of the tool. Therefore, if the kinematics error changes sharply from track to track, the method may require an excessive number of points.

This paper introduces a new modification of the grid refinement which fits better in the framework of tool path optimization and is designed specifically for the SFC generation. The method does not require equal number of points on each track. It automatically evaluates the number of the required grid lines. As opposed to the preceding approach, where the weight function represents either the kinematics error or an estimate of the kinematics error (such as the surface curvature or the rotation angles), the proposed algorithm iteratively constructs an adaptive control function designed to represent the scallop height constraints. Additionally, instead of the Winslow functional the new optimization is based on the harmonic functional derived from the theory of harmonic maps. The functional not only provides the smoothness and the adaptivity but under certain conditions guarantees the numerical convergence. Finally, this approach merges with the SFC techniques. In this case, the grid is not converted to the tool path directly. Instead, it becomes the *basic grid* required for the SFC generation. With this modification, the curvilinear space filling curve (CSFC) tool path can be constructed for surfaces with complex irregular boundaries, cuts off, pockets, islands, etc. Besides, the adaptive grid allows to efficiently treat complex spatial variability of the constraints in such a way that the SFC is created on a grid having the small cells only where necessary.

Finally, the combination of the two techniques is superior with regard to the case when the two methods are applied independently. A variety of examples are presented when the conventional methods are inefficient whereas the proposed algorithms allow constructing the required tool path with the length close to the minimal. The numerical experiments are complemented by the real machining as well as by the test simulations on the Unigraphics 18.

3. TOOL PATH GENERATION FOR A SURFACE WITH CURVILINEAR BOUNDARIES AND POCKETS

This example demonstrates the use of the CSFC to construct tool paths to machine surfaces with complex irregular boundaries, cuts off, and islands. Consider a surface shown in Figure 4(a). Figure 4(b) shows the corresponding CSFC and Figure 4(d) the machining result obtained with the use of the solid modeling engine of the Unigraphics. The surface has been machined by a flat-end tool of radius 3 mm and the machined surface tolerance of 0.05 mm. Consequently, the method is capable of creating tool path for surfaces with complex non rectangular boundaries and islands.

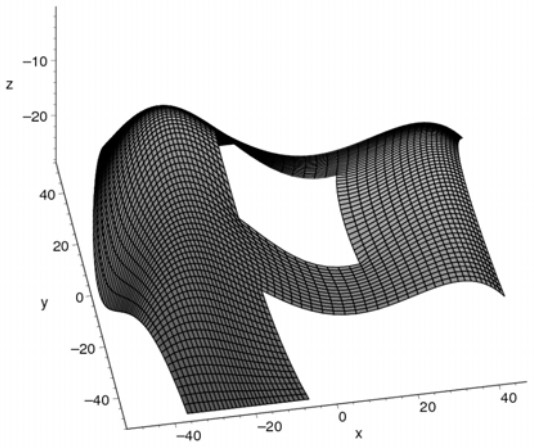


Figure 4 (a): The machined surface

4. POINT MILLING OF INDUSTRIAL IMPELLER

Frequently, the blades of the impellers (Figure 5) are produced by the so-called five-axis *swarf milling* made by a side of the tool. In this case the contact between the workpiece and the cutter is characterized by a contact line rather than a contact point. However, the cutter contact line may lead to large errors. This example demonstrates machining of the blade by 5-axis by (more accurate) *point milling* with the use of CSFC .

In order to demonstrate the advantages of the proposed method, the geometrical complexity of the example blade is increased as

follows. We assume that the blade is broken along a prescribed curve and requires restoration so that we will cut only the mussing part of the blade.

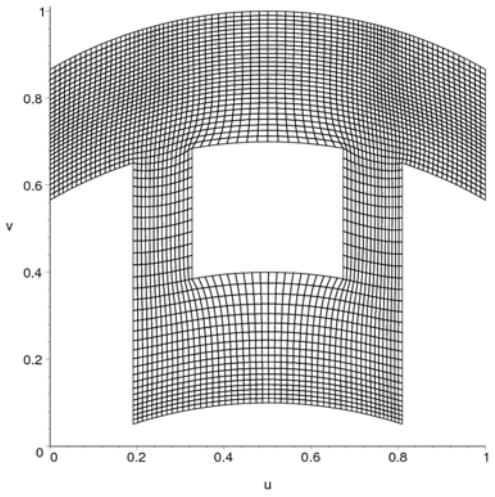
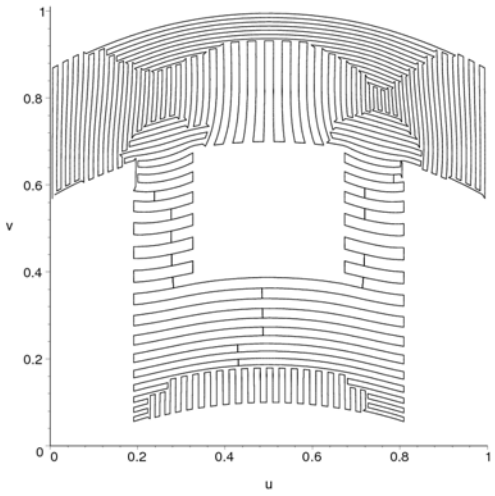


Figure 4 (b): The curvilinear grid



. **Figure 4 (c):** The curvilinear space filling curve

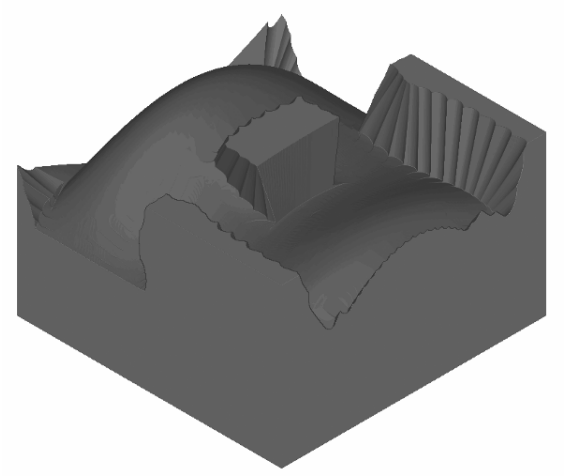


Figure 4 (d): Virtual machining in Unigraphics

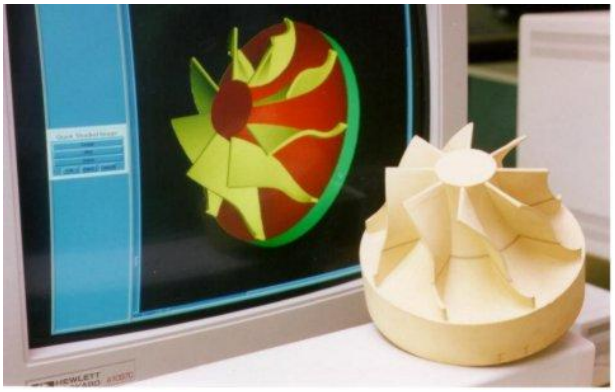


Figure 5: An industrial impeller

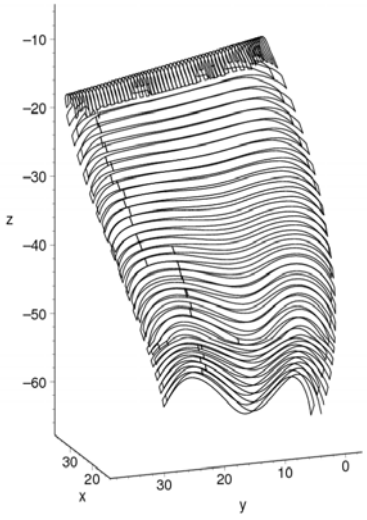


Figure 6(a): CSFC for the “broken blade”

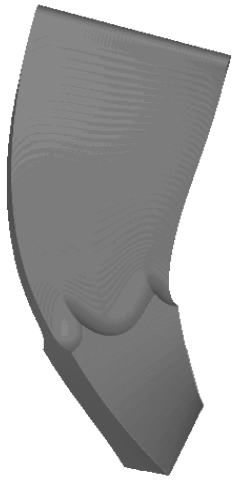


Figure 6(b): Virtual machining of the "broken blade"

Figure 6(c): Prototype blade (wood)

Figure 7: Blade for turbo machinery

The corresponding CSFC tool path is shown in Figure 6(a). A virtual cutting using the proposed CSFC tool path is shown in Figure 6(b) whereas a real prototype of the blade (wood) in Figure 6(c) For demonstration purposes the size of scallops has been chosen so that the CSFC is visible on the blade surface. Finally, Figure 7 shows an industrial blade suitable for the turbo machinery produced with the proposed method.

5. CONCLUSION

Numerically generated adaptive curvilinear grid is introduced to replace the rectangular grid for construction of the space filling tool path for 5 axis machining. With this modification the CSFC can be constructed for surfaces with complex irregular boundaries, cuts off, pockets, islands, etc. The CSFCs are applicable to produce industrial blades for turbo machinery.

6. REFERENCES

[1]. Anotaipaiboon, W. and Makhanov, S.S., Tool path generation for five-axis NC machining using adaptive space-filling curves. Int. J. Prod. Res., 2005, 43, 1643-1665.
[2]. Brackbill, J.U. and Saltzman, J.S., Adaptive zoning for singular problems in two dimensions, J. Comput. Phys., 1982, 46, 342-368.
[3]. Bieterman, M. B. and Sandstrom, D. R. 2003. A curvilinear tool-path method for pocket machining. Journal of Materials Processing Technology, 125(4):709–715
[4]. Charakhch'yan, A. A. and Ivanenko, S. A. 1997. A variational form of the Winslow grid generator. Journal of Computational Physics, 136(2):385–398.
[5]. Cox, J.J., Takezaki, T., Ferguson, H.R.P., Kohkonen, K.E. and Mulkay, E.L., Space-filling curves in tool-path applications. Comput.-Aided Design, 1994, 26, 215-224.
[6]. Griffiths, J.G., Toolpath based on Hilbert's curve. Comput-Aided Design, 1994, 26, 839-844.
[7]. Lam, W.M. and Shapiro, J.H., A class of fast algorithms for the Peano-Hilbert space-filling curve. In Proceedings ICIP-94, IEEE Computer Society, Vol. 1, 1994, pp. 638-641.
[8]. Lo, C.-C., Efficient cutter-path planning for five-axis surface for machining with a flat-end cutter. Computer Aided Design, 1999, 31, 557-566.
[9]. Makhanov, S.S. and Ivanenko, S.A.,Grid Generation as Applied to Optimize Cutting Operations of a Five-axis Milling Machine, Applied Numerical Mathematics, 46, (2003), 353-377
[10].Rao, A. and Sarma, R., On local gouging in five-axis sculptured surface machining using flat end tools. Comput.-Aided Design, 2000, 32, 409-420.
[11].Yoon, J.-H., Tool tip gouging avoidance and optimal tool positioning for 5-axis sculptured surface machining. Int. J. Prod. Res. 2003, 41, 2125-2142.

The system for visualization of synoptic objects

Sergey Melman, Valery Bobkov, Vladimir May
Institute of Automation and Control Processes
Vladivostok, Russia
{gruzd, bobkov, may}@iacp.dvo.ru

Abstract

This work is devoted to developing tools for the visual analysis of tropical cyclones based on satellite data. The implemented system has an extensible set of algorithms for the loading, processing and visualization of data, mainly the spatial scalar fields. The well-known algorithms and author's developments using CUDA technology, and shaders were used in creation of visualization system.

Keywords: *visual analysis tool, animation, volume visualization, scalar field, synoptic objects, tropical cyclone, shaders, CUDA.*

1. INTRODUCTION

This report concerns the problem of visualization of synoptic objects. This problem is important today because the tools for visual analysis are required for research, modeling and forecasting of flows, cyclones and other synoptic objects in atmosphere and ocean. The developments of such tools were started for a long time ago in the world. For the first time there were a software with a mostly 2D visualization. For example, the system of special effects visualization for television needs, the system WeatherEye from European center of research and weather forecasts METOFFICE or Terra3D [8]. The limitation of this system is the total absence of interactivities. The interactive 2D visualization of satellite and aero photos weather maps was implemented in other software products such as Cat5Data, F5Data, TornadoTarget. There are also the systems of data number notation and data visualization on the graphics: Lightning/2000 , LNM View [9], Weather Display [10]. These products are intended for weather forecasts interpretation and representation of current detectors registration. It required more advanced and perfect 3D visualization methods for tasks of modeling and forecasting of difficult weather situations such as tropical cyclones (TC), storms. In some systems for these purposes the universal 3D visualization packs as VolPack [11] are used. The limitation of similar approach is the absence of the special tools for preprocessing and analyzing of visualized data, and the usage of standard data formats that in turn demands additional efforts for converting of initial data and calculating results into standard data formats. Therefore the creation of special systems for synoptic 3D data visualization is still relevant today. Among innumerable real systems of such rank it should be chosen the system CAMVis from NASA [1][5] and «System of storm visualization in 3D environment» with the usage of stereovision system VERTEX with the support of NOAA project [6]. These systems differ with high self-descriptiveness and quality of visualization using the power of supercomputers. However, they are unique and not replicable. They are characterized by sufficiently complex software architecture and require specialized computer equipment. They are developed taking into account the needs dictated by the specific data formats and tasks in a specific monitoring center. The expensive super computers and computer clusters are used.

As we can see the similar systems have narrow directivity and are created for monitoring centers needs.

This report represents the visualization system of synoptic objects that is development of work described in [3]. The system is focused first of all on visual analysis of TCs and is operated on general computer engineering. The needs of satellite monitoring center in IACP FEB RAS caused necessity of system creation and demands to it. The main difficulties and demands to this system are:

- high quality rendering for the visual analysis of TCs dynamics;
- huge amounts of data;
- real-time calculations of addition parameters and real-time animation;
- necessity of combining data from different satellites;
- noise pollution and unspecified format of input data.

Our contribution:

- development of methods and algorithms of animated TCs visualization through calculating of physical field anomalies;
- usage of shaders and CUDA technology for animation mode support and acceleration of application-specific calculations;
- the implementation of a working version of the system for synoptic objects visualization, that is oriented on visual analysis of TCs according to the requirements of professionals from IACP FEB RAS satellite center.

2. CYCLONE ANIMATION

2.1 Input data, recovering

The data of satellite observations of spatial physical fields in atmosphere (temperature, pressure, humidity) in TC regions are used for TC visualization. To track the dynamic of the TC it is necessary that the TC was in satellites' zone of visibility from its initial stage to dissipation taking into account the fact that the TC is moving.

After the preprocessing the satellite data are presented as profiles (regular grid) according to the levels of height above grade. There are two difficulties in data operations. The first one is traditional difficulty – absence of meaning data in some points and presence of errors in measurements. The cause of the second difficulty is that in-orbit satellites are not able to provide required steady «visibility» of TC, so as the satellites fly along fixed orbits, and the TC path of motion is unpredictable. Therefore it is necessary to use integral measurements of two or more satellites, for example such satellites as NOAA 15, 16, 18, 19. Though two adjacent satellite measurements, as a rule, are splitted by time and refer to different regions. The time intervals between measurements can differ from 30 minutes to 12 hours. If let us assume that TC has slight changes in these time intervals, so it's possible to use the methods of space-temporal data recovering.

The different methods of data extrapolation and interpolation were considered. According to experiments the using of simple methods of recovering through neighbor and linear interpolation is enough in this case. Two methods of data joint using are applied in this case: a) integration of measurements and b) crossing of measurements. They both have apparent disadvantages and advantages.

2.2 Calculation of anomalies

If it is known that profiles in the region have a TC and the location of its center is also evident, so it is possible to build a profile mean value on each level of height (normal condition). The meanings from TC outside are used for calculating of mean value, as it is known that the condition of characteristics on the TC outside is close to regular condition. Officially on each level of height the data inside ring with an inner radius of 250 km, and an outer radius of 500 km are used for calculation of mean value. The radius of 500 km restricts the ring, because the processes not relating to the TC can affect to mean value. The anomaly is calculated as the meaning difference on the level and mean value for this level. Then for each level the univariate dependence function of anomaly size from distance till the TC center is built. Thus we receive anomalies distribution through the height and remoteness from the TC center.

2.3 Visualization of tropical cyclone core according to anomaly map

During anomaly visualization the area - a core of the TC can be clearly extracted. There are maximal anomalies in the TC core. The 3D regular grid with the center in the center of TC is used for the TC core visualization. Each grid cell depending on the distance from the TC center is filled with the value from anomalies map. The grid dimension depends on compromise between the quality of visualization and calculations work content. In example below the optimal choice was a grid 64x64x64.

2.4 Visualization algorithms

The following algorithms of animated scalar field visualization are used in this system version:

1) algorithm of volumetric textured visualization;
2) algorithm of volumetric many-particle visualization ;
3) algorithm of isosurface visualization;
4) algorithm of tracing on shaders.

The first three of algorithms listed above were developed by authors in previous system version. There is a short characteristic below, and more detailed description can be found in [4].

Algorithm of volumetric textured visualization is based on using the OpenGL and 3D textures supported by hardware. The field of scalar data is interpreted as 3D-textere according to special scheme, and the graphics accelerator takes the further data interpolation between the grid cells.

The idea of algorithm of volumetric many-particle visualization is based on particles displaying in compliance with the meaning of scalar in considering points of volume. The scalar field images have the following characteristics for visualization: color of particles, illumination, occurrence probability. Animation is the result of permanent change of particles positions (flickering). Thus the user can conclude about scalar field structure by the character of particles distribution on frames chain.

The method of “marching cubes” is used for conversion of voxel isosurface configuration into polygonal configuration [2]. This method provided hardware support of scalar fields isosurface visualization.

3D textures, CLUT textures and shaders-technologies were used for algorithm of rays tracing on shaders realization. The first step is data upload in video buffer. The data are reduced to a format of textures with account of accelerator demands. Now these data can be downloaded in four-component 3D texture, where the first three components contain the special prepared normal in a point, and the forth one contains a restricted meaning of a scalar field.

The texture with ColourLookUpTable (CLUT) (Figure 1) is downloaded into GPU buffer on the second step. The CLUT can be created by CLUT-editor.

Figure 1: Example of CLUT texture. The color feature is taken from texture upper line, and the feature of clarity is taken from the lower one.

Further the vertex and pixel shaders are downloaded into accelerator. As the physical dimensions of scalar field are known, there is a binding to the projection of Mercator coordinate system. The cube with side equal to 1 (one) is added through process of scaling and flushing into the frame buffer, in this case the track OpenGL for cube color uses shaders downloaded before.

OpenGL provides the visibility only of those cube surfaces that are located closer to the view point. As a result shader program gets the point of entry modified into the texture coordinate system. Now it should be calculated the “look” vector in the vertex shader, that spread from the view point to the point of entry.

The light source takes place in the view point position, where the “look” vector and light vector overlap that reduces time for calculations. There is a color accumulation in the fragment shader on the principle of “from close to distant” until the ray crosses the cube bounds or the accumulated clarity reaches the meaning of “unclear pixel”. Then the further color accumulation has no meaning. On each step of color meaning accumulation the meaning of modified gradient and scalar in the point is read out from data texture that is transferred into color through CLUT.

The visualization characteristics during animation (calculation of textures, gradients, minmax scalar fields meanings and polygonal models constructing) are calculated in real time. The effective traditional algorithms are used for animation in the space.

It should be noted that module system structure lets easily to expand the base of visualization algorithms.

3. MULTIPROCESSING

One of the requirements to a system was a possibility of its operation on personal PC with stuff hardware and on the multi processing computer system. That’s why during the system development the following direction of calculations acceleration and real-time mode support are taken into consideration: using of multi-core CPU, using of accelerators with shader- and CUDA technologies.

As the physical fields are given in the regular 3D grid points, the operations above them have a grid character. According to

analysis the steps of synoptic data processing depending on its computational cost and opportunity to paralleling can be divided between the center processor (CPU) and accelerator (GPU) in the following way (Figure 2). The step of downloading, data compensation and anomaly calculation is realized on CPU in multithreading. Interpolation and characteristics transformation are executed on CUDA-core GPU. Data preparation to visualization is executed on CPU and GPU. The visualization is performed by traditional OpenGL and shader-technology.

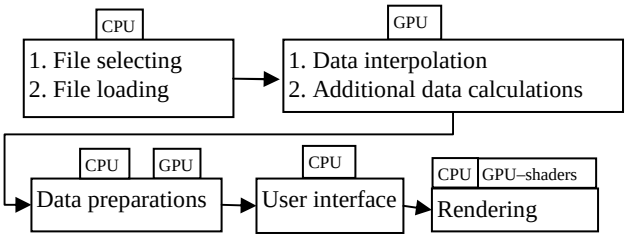


Figure 2: Steps of data processing

As all applicable calculations putting into GPU, have a grid character, all of them are well paralleled and bring approximately the same contribution in general data processing acceleration.

4. SYSTEM STRUCTURE

The system provides users with an interface of data control and selection; tools for their filtration and transformation; management of the renders options and camera angle (Figure 3).

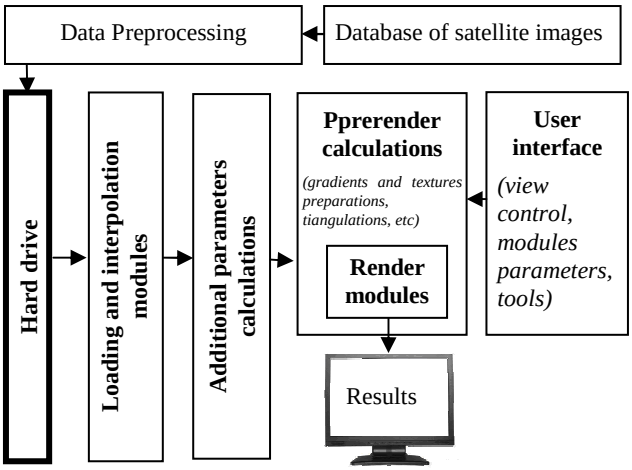


Figure 3: The users interface of the system for a data control and visualization.

5. EXPERIMENT

The system work is illustrated on the base of TC Melor real data with its time of life from 29.09.2009 till 10.10.2009.

CLUT (Figure 4) was used for visualization of the temperature anomalies in the TC core.

Figure 4: Color distribution in relation with the temperature of anomalies (0K in the middle – transparent)

The yellow color shows the anomalies with a temperature higher than 4.5K, the red color – anomalies from 1K to 4.5K (degree Kelvin), the blue color - anomalies from -4.5 K to -1K, the green color - anomalies below -4.5. The scale of grey color defines transparency.

The track of TC Melor is shown on the Figure 5 [7]. Figure 6 - Figure 8 presents the result of system work: TC Melor visualization and its expended fragments that have a special meaning for the picture interpretation.

The TC dynamic reflects the TC Melor beginning and development till its break-down. The daily TC “breath” is well seen during the animated visualization of TC dynamic. The TC eye (calm zone) forming is clearly seen during its crossing over Japanese island. The clear-cut warmth of a core (of red color) is on the 20 level that is equal to 10 km height. Also the forming of temperature anomaly zones above the TC core is also well seen in troposphere.

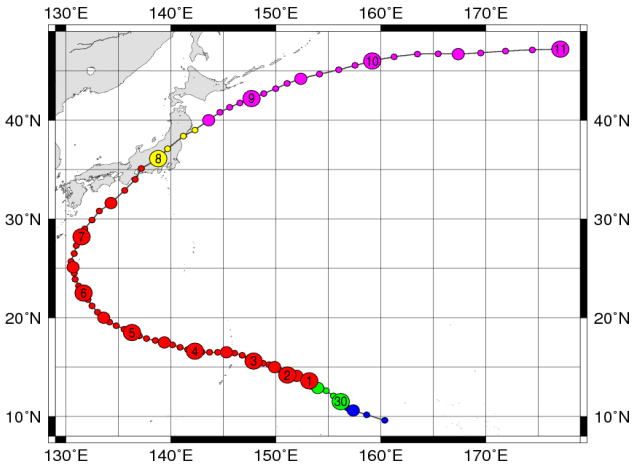


Figure 5: TC Melor path.

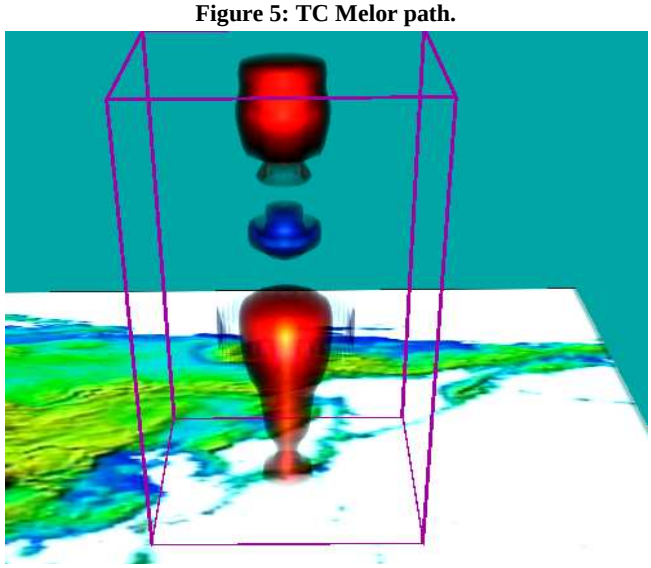


Figure 6: TC Melor crossing over Japanese islands.

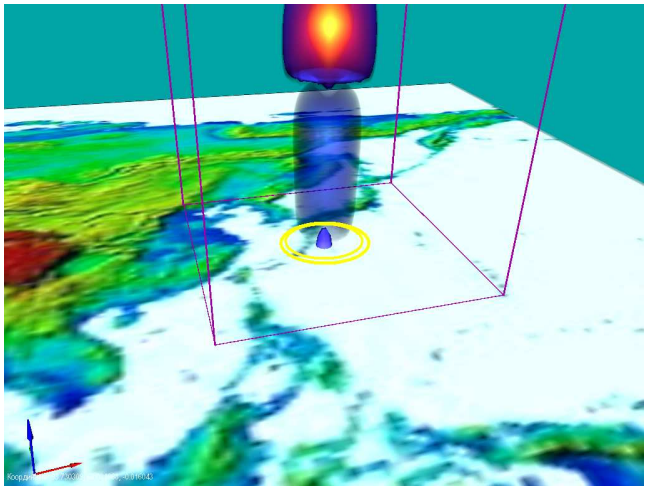


Figure 7: «TC's eye».

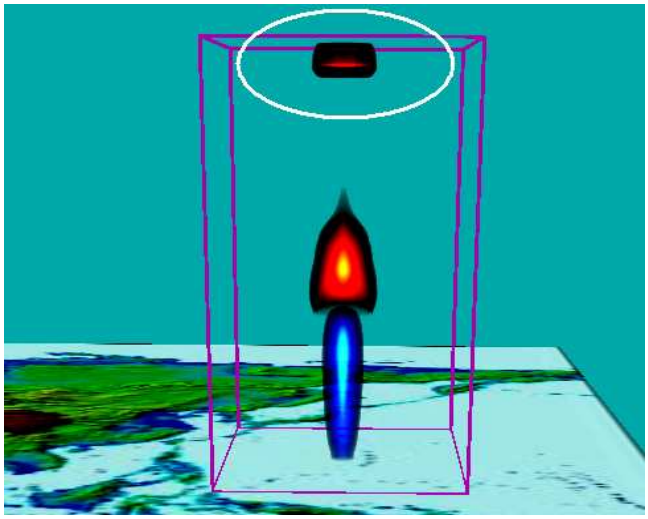


Figure 8: Excitation in troposphere.

6. CONCLUSION

The paper presents the current version of the system for visualization of physical fields of synoptic objects, focused on visual analysis of TCs. Experiments showed the effectiveness of the chosen method of displaying the dynamics of TCs with the help of visualization of temperature anomalies. The realized possibilities of multicore processing of the applied data using CUDA technology and rendering using shaders have provided acceptable speed of processing of huge amounts of the satellite data and a real-time animation mode of visualization. The system is currently used in the satellite monitoring center in the IACP FEB RAS. It used in the mode of trial operation for visual inspection of the primary processing of operational satellite measurements. Also it used to study TCs in the retrospective data. Further development of the system is planned in several directions: enhancing of visualization algorithms; implementation of the system on a hybrid multiprocessor architecture using multiple graphics cards; system adaptation to the demands of professionals of Roshydromet.

This work was supported by grant of FEB RAS.

7. REFERENCES

[1] **Green, B., C. Henze, B.-W. Shen:**, *"Development of a scalable concurrent visualization approach for high temporal- and spatial-resolution models"*. Eos Trans. AGU, 91(26), West. Pac. Geophys. Meet. Suppl., Abstract A23B-142. AGU 2010 Western Pacific Geophysics Meeting, Taipei, Taiwan, June 22-25, 2010. Highlighted by NASA Advanced Supercomputing (NAS) Division, NASA Ames Research Center.

[2] **Lorensen W.E., H.E. Cline:** *"Marching Cubes: a high resolution 3D surface reconstruction algorithm"*, SIGGRAPH 87, 1987, pp.163-169.

[3] **Melman S. Bobkov V.:***"Visualization of synoptic objects"*,17-th International Conference on Computer Graphics and Vision, GraphiCon 2007. Conference Proceedings. Pp 264-268.

[4] **Melman S.** *"Volume rendering in analysis of physical fields of synoptic objects"*, Information technology №1, pp 62-66

[5] **Shen, B.-W., G. Bryan, W.-K. Tao, C. Henze, S. Cheung, J.-L. F. Li, and P. Mehrotra**: *"Coupling NASA Advanced Multi-Scale Modeling and Concurrent Visualization Systems for Improving Predictions of Tropical High-Impact Weather (CAMVis)"*. Earth Science Technology Forum (ESTF) 2010.

Arlington, Virginia, June 22-24, 2010

[6] **Wang Shao Rong, Rui You, Sheng Li, Yi Song Chen, Guo Ping Wang: "***Realtime Visualization of Hurricane in Distributed Environment"*, Applied Mechanics and Materials (Volumes 40 - 41), Volume: Advances in Science and Engineering, pp.: 948-954, 2010

[7] http://agora.ex.nii.ac.jp/digital-TC/news/2009/TC0918/index.html.en

[8] http://www.met.fu-berlin.de/terra3d/en/index.htm

[9] http://www.thiesclima.com/soft_e.htm

[10] http://www.weather-display.com/index.php

[11] http://graphics.stanford.edu/software/volpack

About the authors

Melman Sergey Vladimirovich is a senior software engineer of Computer Graphics Laboratory, IACP FEB RAS, Vladivostok, Russia. His contact email is gruzd@iacp.dvo.ru

Bobkov Valery Alexandrovich, Computer Graphics Laboratory Chief, IACP FEB RAS, Vladivostok, Russia. His contact email is: bobkov@iacp.dvo.ru

May Vladimir Pavlovich is a senior scientist of Computer Graphics Laboratory, IACP FEB RAS, Vladivostok, Russia. His contact email is may@iacp.dvo.ru

Simple Geometry Compression for Ray Tracing on GPU

Kirill Garanzha[1] Alexander Bely[2] Vladimir Galaktionov[1]
[1] Keldysh Institute of Applied Mathematics, Russian Academy of Sciences
[2] CentiLeo

Abstract

In this short paper we describe simple approach to loosely compress the vertex data for ray tracing large triangular models on the GPU. The disadvantage of the GPU is limited memory capacity. The advantage of the GPU is high performance computation. Sometimes it is hard to load large models to the GPU and we suggest compressing the vertices that form 3D models and acceleration data structure using 16bit representations instead of 32bit floats. At the same time the aggregate vertex precision varies between 22 and 24 bits and the amount of cracks is minimized. The advantages of NVIDIA CUDA platform are used to implement our approach efficiently.

Keywords: *ray tracing, GPU, CUDA, compression.*

1. INTRODUCTION

Scenes for feature film rendering and CAD/medicine visualization may have large geometric complexity and can easily contain ten or hundred million polygons. Demands for greater photorealism, more realistic materials, complex lighting and global illumination push computational bounds which often results in long render times and more research in data compression methods.

Complex lighting can be implemented with ray tracing rather simply. However efficient ray tracing implementation on the GPU is still a challenge. The GPUs such as NVIDIA GTX480 and GTX580 have around 1.5Tflops of compute power and only 1.5GB of main memory. Our challenge is to load a model with several ten million triangles to these GPUs.

One of the most popular 3D model representations is triangular representation which encodes the surface of the 3D model. One of the most popular acceleration structures for ray tracing is the Bounding Volume Hierarchy (**BVH**) of Axis Aligned Bounding Boxes (**AABBs**) [4].

In this paper we present a simple and efficient method for compact vertex coordinates and BVH storage which is dedicated for convenient ray tracing queries on modern GPU (our geometry compression is optimized for GPU ray tracing target unlike general compressed geometry/mesh representations by Deering and Chow [8] [9]). We spend an average of 15.5bytes per triangle for vertex coordinate and BVH data storage. This allows spending 750-950MB (for vertex coordinates and BVH) of GPU memory to ray trace 50-60 million triangle models (see Figure 1) which are common for wide range of CAD visualization tasks. Several features of NVIDIA CUDA platform [7] which are implemented in hardware are used to simplify and accelerate our data decompression method. Our basic approach is quantization of vertex coordinates in the limited space of bounding box.

2. ALGORITHM

Every node of the BVH contains the AABB and the link to the pair of children nodes (the BVH leaf contains the link to the primitives).

Figure 1: *The scene contains 58M triangles and is rendered on GTX 480 with 1.5GB memory: 3 area light sources, specular or diffuse reflections (the maximum ray path length is 3). The image is rendered progressively at 1024x768 (every image pass takes 130ms to render).*

As the rays are usually non-coherent this may result in a non-coherent memory access for the threads within the same CUDA warp. When the rays processed by the same CUDA warp access non-coherent data the best way to improve the memory access time is to access the data packed in float4 or int4 elements (basic 16byte CUDA types). The memory access time is further improved when we use CUDA textures [7]:

```
float4 elem = tex1Dfetch(texAABBdata, addr);
```

BVH data layout and fetching. We use the ray tracer implementation similar to the one by Aila and Laine [1]. The ray traverses through the binary BVH where two sibling nodes are stored together (the top root node is not stored; see Figure 2, left image). These 2 sibling nodes are sorted during ray traversal and the ray prefers to descend to the closest one until the leaf node is reached. Each AABB needs 6 words to encode the box, two sibling AABBs need 12 words to encode two boxes. During the ray traversal we fetch these 12 words using three float4 texture fetches (see Figure 2, array *Aabbs*) and then consider them as 2 boxes inside the "*ray-pair of boxes*" intersection.

Vertex data layout and fetching. An input triangle mesh is converted to a quadrilateral mesh using triangle connectivity information [3]. We process triangles one by one. Each triangle is connected to some neighboring triangle if they share a common edge and have the same material. From multiple candidates we select a pair of connected triangles that form a quad with the smallest perimeter. A quad mesh has 1.5 times less connectivity information (four vertex links per quad) than corresponding triangle mesh (six vertex links per two triangles).

Every new scene primitive has 4 vertices and one shared edge. Such a quad is stored as four consecutive 3D vertices (see Figure 2, array *Primitives*).

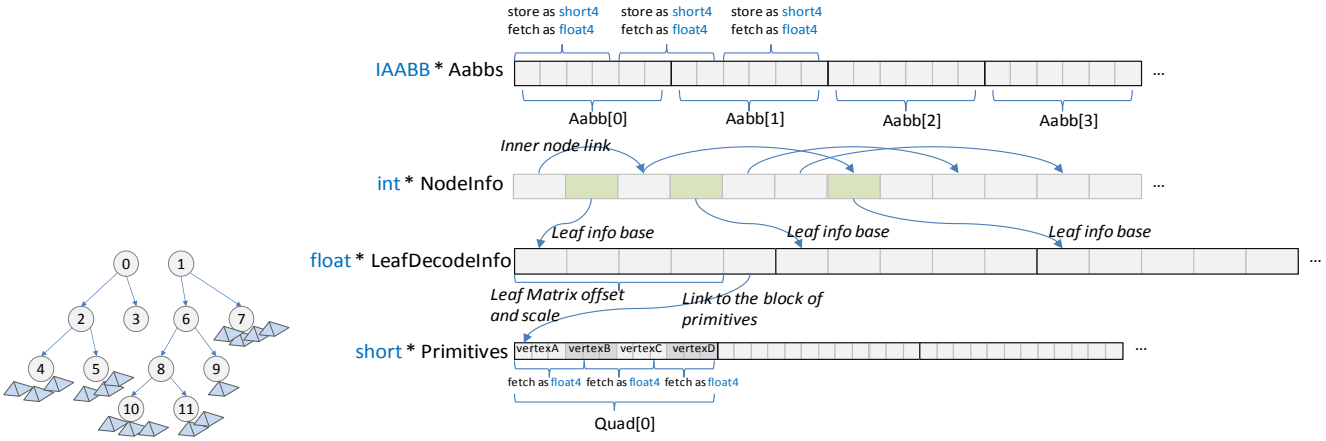


Figure 2: *Left: hierarchical view of the BVH without root; all the sibling nodes are stored together. Right: memory representation of the BVH and the list of primitives. Each i-th Aabb from Boxes array and i-th word from NodeInfo array forms the BVH node. The inner node (32^{nd} bit of NodeInfo[i] equal to zero) refers to the pair of children nodes. The leaf node (green rectangle) has 32^{nd} bit of NodeInfo[i] equal to one and refers to the block of five 4byte words of array LeafDecodeInfo. The first 4 words encode leaf decode matrix offset and scale, the fifth word is casted to integer value which refers to the block of several primitives united by this leaf.*

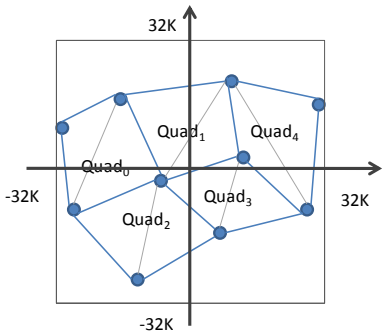


Figure 3: *Geometry Quantization. The BVH leaf is enclosed into a bounding cube and the vertex coordinates are quantized within the range $-2^{15}..2^{15}$.*

Vertex data compression. In the BVH generated using Surface Area Heuristic (SAH) [6] the leaves are well separated from each other. We assume that leaf extents are small enough compared to the whole mesh. All the quad vertices within the leaf are embedded into a 16-bit cube (coordinates ranging from -2^{15} to 2^{15}) and quantized (see Figure 3) using *Mencode* matrix (see the code for quad vertex compression in the *Listing 1*). We then linearize all quad vertices eliminating the connectivity information and vertex link indirection. Each quad now takes 24 bytes (i.e. 12 bytes/triangle). Decode information is stored per leaf: *float3 Center* and *float maxwidth* components of *Mdecode* matrix are stored in the first 4 words of the block allocated for this leaf in the array *LeafDecodeInfo* (see Figure 2).

Vertex decompression (decoding) is very fast. We use CUDA texture fetch whose mode is set to *cudaReadModeNormalizedFloat*:

```
// Bind the array of linearized quads to texQuad.
texture<short4,1,cudaReadModeNormalizedFloat> texQuads;
cudaBindTexture(0, texQuads, Primitives, numPrims * sizeof(QUAD_COMPRESSED));

// Read two connected triangles (shared edge is used in intersection test to
// reduce the number of cross/dot products
float3 A, B, C, D; {
  float4 q1 = tex1Dfetch(texQuads, 3 * quadIdx);
  float4 q2 = tex1Dfetch(texQuads, 3 * quadIdx + 1);
  float4 q3 = tex1Dfetch(texQuads, 3 * quadIdx + 2);
  A = make_float3(q1.x, q1.y, q1.z);
  B = make_float3(q1.w, q2.x, q2.y);
  C = make_float3(q2.z, q2.w, q3.x);
  D = make_float3(q3.y, q3.z, q3.w);
}
```

With this normalized fetch mode the float values are mapped from original short value range $-2^{15}..2^{15}$ to the values in the range *-1..1*.

The rays are specified in the world space. When we intersect them with the BVH-leaf they are transformed to the local leaf space (i.e. *-1..1*) using the *Center* and *maxwidth* components of the *Mdecode* which are stored per leaf in the *LeafDecodeInfo* array.

16-bit quantization (16bit vertex precision within the leaf) can encode 2^{48} voxels in a localized BVH-leaf 3D space (assumed to be relatively small compared to the world space). The amount of the leaves which can be placed in the row along any of the scene dimensions can determine the final precision of this vertex representation. If we can place 100-250 BVH-leaves along *x*, *y* or *z* scene dimension then the vertex precision would be 23-24bits in the world space. This precision depends on the size of the AABB of the average BVH leaf compared to the whole scene AABB.

```
Matrix4 CompressVertices(QUAD_COMPRESSED * outLeafQuads, QUAD * inLeafQuads,
                         int numQuads)
{
  AABB SBox = make_aabb();
  for(int i = 0; i < numQuads; i++) {
    Extend(SBox, inLeafQuads[i].A);
    Extend(SBox, inLeafQuads[i].B);
    Extend(SBox, inLeafQuads[i].C);
    Extend(SBox, inLeafQuads[i].D);
  }

  // Scale data
  float3 Center = (SBox.Min + SBox.Max) * 0.5f;
  float3 Size = SBox.Max - SBox.Min;
  float maxwidth = max(Size.x, max(Size.y, Size.z));

  // geometry is stored in a 16bit quantization
  // space in a cube [-32K,-32K,32K] .. [32K,32K,32K] with a center at [0,0,0]
  Matrix4 Mencode = scale(65534.0f / maxwidth) *
                    translate(-Center.x, -Center.y, -Center.z);

  // used to get the original position of object during ray traversal
  Matrix4 Mdecode = translate(Center.x, Center.y, Center.z) * scale(maxwidth);

  // optimized vertex coordinates
  // (shift and scale the vertices to the local compression space)
  for(int i = 0; i < numQuads; i++) {
    outLeafQuads[i].A = make_short3(transformPoint(Mencode, inLeafQuads[i].A));
    outLeafQuads[i].B = make_short3(transformPoint(Mencode, inLeafQuads[i].B));
    outLeafQuads[i].C = make_short3(transformPoint(Mencode, inLeafQuads[i].C));
    outLeafQuads[i].D = make_short3(transformPoint(Mencode, inLeafQuads[i].D));
  }

  return Mdecode;
}
```

Listing 1*: Code fragment for compressing quad vertices of the BVH leaf.*

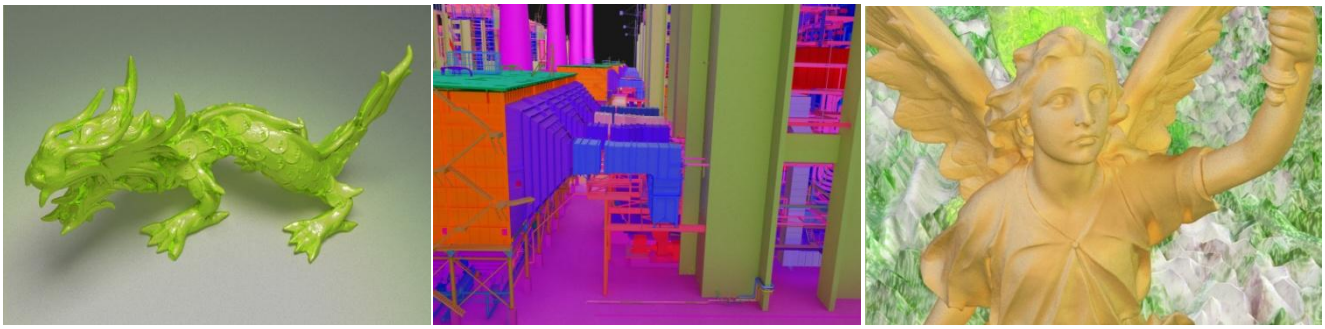


***Figure 4**: Dragon, PowerPlant and Lucy rendered with 3 light sources and enabled compression mode. No any visible artifacts/cracks.*

We have tested this simple vertex coordinate compression with a variety of large 3D models (CAD scenes, laser scanned objects, etc.) and have not observed any cracks between the leaves of the leaves of the BVH.

Although, the cracks are possible for small models which contain the primitives with large extents. Example: two primitives are connected in the original mesh. When the BVH is generated the primitives fall into different leaves. These leaves have independent bounding cubes (which are used for quantization, see Figure 3). For large extent primitives (compared to the scene extent) this can be the reason of visible cracks between the leaves of the BVH.

In our scenes we can mix the models which have 32-bit vertex precision or reduced vertex precision.

BVH data compression. For large models which contain several tens of millions of polygons we generate the SAH-based BVHs with up to 16 quads (equal to 32 triangles) per leaf. These "fat" leaves reduce the BVH size (the arrays *Aabbs*, *NodeInfo* and *LeafDecodeInfo* are smaller if the leaves have more primitives).

Speculative ray traversal [1] has more effect (compared to non-speculative) for the BVHs with "fat" primitives.

We also apply a 16bit quantization to the plane coordinates of the AABB. Each of the 6 plane coordinates is stored in a 2byte word. This encoding is done similarly to the vertex data encoding: all the world-space bounding boxes of the BVH-leaves are embedded into a global 16bit bounding cube (similar to the one on the Figure 3). All the BVH bounding boxes are stored using short data type. Ray traversal decodes the AABBs using the same normalized texture fetch mode which converts the data stored in short type to the float type.

With this quantization the short representation bounding boxes can be slightly extended compared to the boxes computed originally with float type. This extension may result in a few more ray-primitive intersection tests during the course of ray traversal.

3. RESULTS

We test the implementation of our compression method using CUDA 3.2, 64bit WindowsXP and GTX480 card (with 1.5GB of memory where only 1.3GB can be allocated with *cudaMalloc*).

The BVH is built offline using the Surface Area Heuristic (SAH) [6]. All the scenes are rendered with 1024x768 images resolution with 3 area light sources and 3 bounce rays.

Figure 1 represents the 58M triangle scene (a PowerPlant with 13M triangles, a Lucy model with 28M triangles, a Thai model with 10M triangles and a Dragon model with 7M triangles). In a compressed mode we spend 900MB for BVH storage (max 16 quads / leaf) and vertex data storage (which result in 15.5 bytes per triangle storage). Without this kind of memory optimization this kind of model could consume up to 2.7GB of storage which can't fit into GTX480 memory.

	non-compressed (4 triangles / BVH leaf)		compressed (quads + quantization + 16quads/BVH leaf)	
	Storage, MB	Render, ms	Storage, MB	Render, ms
Dragon,7M triangles	420	45	109	51
Thai, 10M triangles	600	47	155	55
PowerPlant, 13M triangles	780	97	201	110
Lucy, 28M triangles	1680	n/a	434	65
All together, 58M triangles	3480	n/a	900	130

***Table 1:** Non-compressed vs. compressed stats.*

In Table 1 we present comparison statistic for two modes: non-compressed (36 bytes for vertex data storage per triangle, 28 bytes for AABB storage, 4 triangles per BVH leaf) and compressed (grouping into quads, quantization and 16 quads per BVH leaf).

In the non-compressed mode each triangle requires up to 60 bytes to store the vertices and acceleration structure. Because of this we can't ray trace the models larger than 13M triangles on a 1.5GB graphics card.

With a maximum of 32 quads per primitive we can reduce the storage by 5% and decrease the ray tracing time by another 10%.

With a compression mode enabled we have 3.9x compression rate for the data storage and 10% slower ray tracing (if we compare render timings for PowerPlant and Dragon which both fit into GPU for both compressed/non-compressed modes). However, compressed-mode allows loading the model with 60-70M triangles to the GPU.

When compression mode is switched the amount of data to be accessed is reduced and there should be less influence of data access BW on the ray tracing time. But at the same time we have increased the number of primitives per BVH leaf from 2 quads (4 triangles) per leaf till 16 quads per leaf: the average number of ray-primitive intersection tests per ray was increased by 90%. Instead of 2x slowdown we got 10% slowdown for models that fit into memory (for both modes) because of the more utilization of GPU computation units per memory access units. This simple

experiment confirms that for GPU it can be better to store less and compute/recomputed more.

Potential improvement. The advantage of our approach is its simplicity compared to the hierarchical AABB compression [2], [5] which is harder to implement on GPU. Though, in the future we would like to experiment with such compression methods as well as with the larger primitive (not quad, but a triangle fan or "cluster" [3]).

Another interesting direction of future work would be to compress the dataset on the GPU as well as acceleration structure generation on the GPU.

4. CONCLUSION

In this paper we have presented a simple method to compress the vertex coordinates dataset and the bounding volume hierarchy by almost 4x and keep efficient ray tracing at the same time. This method allows rendering 4x larger models on the same GPU at low implementation cost.

5. AKNOWLEDGEMENTS

This work was supported by the Russian Leading Scientific Schools Foundation, project NSh- 8129.2010.9, RFBR, grants 10-01-00302.

6. REFERENCES

[1] Aila T., Laine S.: Understanding the Efficiency of Ray Traversal on GPUs. In Proc. of High Performance Graphics (2009).

[2] Fabianowski B. and Dingliana J.: Compact BVH Storage for Ray Tracing and Photon Mapping. Proceedings of the 9th Eurographics Ireland Workshop, Dublin, Ireland, 2009.

[3] Garanzha K.: The Use of Precomputed Triangle Clusters for Accelerated Ray Tracing in Dynamic Scenes. Proceedings of the Eurographics Symposium on Rendering 2009, Girona, Spain, 2009.

[4] Kay T., Kajiya J.: Ray tracing complex scenes. In Proc. of SIGGRAPH (1986), 269–278.

[5] Mahovsky J.: Ray Tracing with Reduced-Precision Bounding Volume Hierarchies. PhD thesis, University of Calgary, Calgary, Alberta, Canada, 2005.

[6] MacDonald J., Booth K.: Heuristics for ray tracing using space subdivision. The Visual Computer 6, 3 (1990), 153–166.

[7] NVIDIA. 2011. NVIDIA CUDA Programming Guide Version 4.0.

[8] Chow M.: "Optimized Geometry compression for real-time rendering", Proceedings on the IEEE Visualization'97.

[9] Deering M.: "Geometry Compression", Computer Graphics, 1995, 13-20.

Topological Mapping Complex 3D Environments Using Occupancy Octrees

Konstantine Kazakov, Vitaly Semenov, Vladislav Zolotov
Institute for System Programming of the Russian Academy of Sciences (ISP RAS), Moscow, Russia
step@ispras.ru

Abstract

Global path planning is a challenging problem arisen in many fields of research. Unfortunately, path planning algorithms have relatively high complexity that extremely grows with the input data volume. Being oriented on accurate metric schemes, traditional local path planning methods have significant limitations in the case of large-scale environments. Their inability to use overall information on the whole environment creates critical shortcoming in global planning. Topological schemes try to overcome this drawback by representing the original environment by means of route graphs. Topological schemes scale better than metric ones, but being resistant to geometric representation errors may yield incorrect or suboptimal solutions.

In the paper we propose an effective method of generating topological maps for global path planning in complex large-scale 3D environments. The method utilizes adaptive spatial decomposition in conformity to occupancy octree structures and uses original criteria for identifying spaces and gates. Thus generated maps provide for whole coverage of environments and enable to effectively resolve multiple path planning requests using graph search algorithms. Conducted experiments proved the feasibility and effectiveness of the method presented as well as its suitability to industry meaningful problem statements.

Keywords: *Global path planning, Topological and metric schemes, Collision Detection.*

1. INTRODUCTION

Global path planning is a challenging problem arisen in many fields of research, including global positioning systems (GPS), autonomous robot navigation and very large-scale integration design (VLSI). The problem is of particular interest to construction planning community facing the requirements of trustworthiness and feasibility of project schedules [10]. Correct schedules must avoid any conflicting situations at project sites and assure the existence of collision-free paths for installed construction elements and deployed equipment. Ultimately it would enable detecting and anticipating problems at earlier planning phases and reducing risks and waste at the final phases.

Unfortunately, global path planning algorithms have relatively high complexity that extremely grows with the input data volume. Well-known metric schemes like configuration spaces, generalized cones, voronoi diagrams, visibility graphs, cell decompositions correspond directly to original geometric representation of the explored environment and provide implicit information about the traversability relations among different places. Being based on metric schemes, popular path planning methods such as probabilistic roadmaps (PRM), rapidly exploring random trees (RRT), and potential fields succeed on local statements. However, their inability to use overall a priory information on the whole environment creates critical shortcomings in global planning.

Topological schemes try to overcome these drawbacks by representing the original environment by means of route graphs. Typically, vertices of such graphs are associated with identifiable locations and edges — with possible routes between them. Topological schemes scale better than metric ones, but being resistant to geometric representation errors may yield incorrect or suboptimal solutions [7]

Integration of metric and topological schemes looks most promising approach to leverage advantages of both paradigms. It can be performed by extracting a topological map from metric representations and by annotating topological elements with metric information. Occupancy grids are usually chosen as a metric representation which is easy to calculate and to update. To reduce their volume and to obtain a reasonable partitioning, the grid cells are grouped in homogeneous regions. Each such region becomes a vertex of the generated topological map and if there is a feasible path joining two regions then their vertices are connected via corresponding edge.

Extensive research efforts have been directed toward topological mapping problems. This partitioning can be performed by using quadtrees [13]. However, the optimality of the resulting partition depends strongly on the distribution of the obstacles in the environment [2]. Sometimes obstacle boundaries are modeled by means of straight lines [1]. The main disadvantage of this method is that it can not deal correctly with irregularly shaped regions nor with walls which are not parallel or orthogonal. The topological map can be extracted from the grid by analyzing the shape of free space by means of the mathematical morphology image processing tool [4]. However, only explored regions can be represented at topological level. Region shape criteria were proposed in [12] to split free space into homogeneous regions. It is reported in [6] that these maps, as well as other self organizing maps like the colored map and the growing neural gas, produce unintuitive tessellations of free space. Wall histograms were proposed in [6] to resolve these problems. However, this method provides no suitable topologies for navigating in free space. Many researchers tried to obtain adequate topological map by identifying features like virtual doors, narrow passages, corners, middle lines [3], [9]. Although their methods work well for particular domestic cases, scarcely that they match to complex 3D environments, particularly, indoor and outdoor environments simulating real construction project sites.

In the paper we propose an effective method of generating topological maps for complex indoor/outdoor environments. The method utilizes adaptive spatial decomposition in conformity to occupancy octree structures and uses original criteria for identifying spaces and gates. Thus generated maps provide for whole coverage of environments and enable to effectively resolve multiple path planning requests using graph search algorithms.

The rest of the paper is organized as follows. Section 2 describes peculiarities of adaptive octree structures utilized as a metric occupancy representation. In Section 3 we present an original algorithm and criteria for extracting a topological map from the metric representation. The algorithms to find suboptimal routes in the topological graph and some results of conducted experiments are mentioned in Section 4. In conclusions we summarize benefits of the topological mapping method proposed.

2. OCCUPANCY OCTREE

To simplify the discussed motion planning problem and to avoid computationally expensive analysis of whole 3D environments, so-called spatial decomposition is usually applied. It assumes subdividing the environment space into cells and determining occupancy status for each localized cell. Using the occupancy metric scheme, path planning problem can be solved more efficiently by successive finding neighboring free cells and navigating over them from a starting point of the moved object to its destination point. Importantly, it can be done under very general assumptions about the simulated environment.

We suggest the environment is composed of objects which may be geometric primitives, algebraic implicit and parametric surfaces like quadrics, NURBS and Bezier patches, convex and non-convex polyhedrons, solid bodies given by constructive solid geometry or boundary representation. No matter which geometric models are adopted. It is assumed only that there is a common function for the interference identification between any environmental object and any given box. The testing result is an occupancy status taking the values 'grey', 'black' or 'white' and pointing whether the box is partially occupied, entirely full, or entirely empty correspondingly.

In this paper we confine ourselves to the case of static environments, although the method presented can be applied to dynamic and pseudo-dynamic environments too. As opposed to many works addressing to path planning, no specific restrictions are imposed upon the geometric representation of both simulated environment and moved objects considered as its intrinsic parts.

Adaptive cell decomposition is used to reduce the number of cells suffered to the analysis and to waste less memory storage space and computation time. Contrary to regular cell decomposition, it is a good tactic for the complex indoor/outdoor environments simulating construction sites and containing large regions with the same traversability. At the same time adaptive cell decomposition imposes problems for dynamic and pseudo-dynamic environments. One good solution is the use of framed trees as suggested in [11]. However, in high clutter environments framed structures can be less efficient than regular grids due to the overhead required to keep track of the cell sizes and locations.

To generate the occupancy octree the well-known adaptive decomposition technique is applied. It begins by imposing a large size cell over the entire planning space. If a grid cell is partially occupied, it is sub-divided into eight equal subparts or octants, which are then reapplied to the planning space. These octants are then recursively subdivided again and again until each of the cells is either entirely full or entirely empty. The subdivision process is interrupted also for refined cells if their size becomes equal or smaller than a given tolerance of the generated metric representation. The resulting octree has grid cells of varying size and concentration, but the cell boundaries coincide very closely with the obstacle boundaries. An example of the octree representation is shown in Figure 1b. It was generated for a simple building model presented in Figure 1a.

The cells of the deployed octree are then marked as 'grey', 'black' or 'white' depending on their occupancy status that points out whether the cell is partially occupied, entirely full, or entirely empty. Note that sometimes it is difficult to identify entirely full cells for the environments consisted of, so-called, polygon soups rather than solid primitives or assemblies. In such situations the subdivision process has to be recursively continued under suggestion that the cells are partially occupied regions.

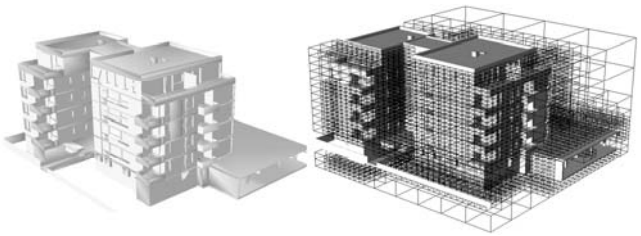


Figure 1: (a) Geometric representation of the building, (b) Deployed occupancy octree.

To make the metric representation more constructive for spatial reasoning and topology extracting, it is proposed to enrich the octree structure by Euclidian distance field values. For this purpose empty cells store distances to the nearby obstacles. To simplify computations, the distance estimates can be obtained using already deployed metric scheme and avoiding consumable analysis of the original geometric representation. In the proposed algorithm the distances are computed from the center of each empty cell to the nearest faces, edges or corners of the neighboring cells (entirely or partially) occupied by the environment objects. Neighboring cells are determined in a way similar to the backtracking method [5] by finding common ancestor and by descending from it to adjacent cells of the explored octant.

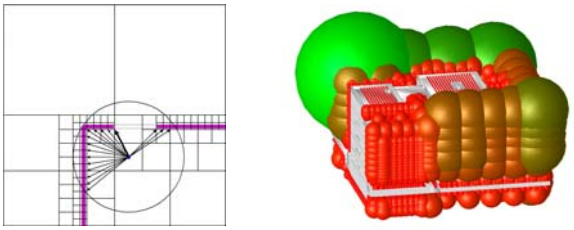


Figure 2: (a) Estimated search area and neighboring occupied cell, (b) Distance field distribution.

A principal distinction of the presented algorithm is that it collects also indirect neighbors which are located not far than search radius r from the explored octant. An initial estimation for the search radius is given by the expression $\sqrt{3}(\frac{3}{2}a - d)$, where a is the linear size of the explored octant and d is the size of the smallest cell coincident with the given metric tolerance. Indeed, by construction the parent of the explored empty octant may be only a grey cell and it must contain at least one occupied child

cell. Evidently that it is located not far than $\sqrt{3}\,(a-d)$ from the center of the parent cell and not far than the given value from the center of the explored octant. As new neighbors are found, the estimated search radius can be refined and replaced by the current value of the distance. This leads to additional savings on traversing and processing cells. An estimated search area and an occupied cell nearby to the central empty octant are shown in Figure 2a as a circle and a bold arrow correspondingly. Figure 2b illustrates the distance field distribution computed for the building model and the occupancy octree presented above.

3. EXTRACTING TOPOLOGICAL MAP

As explained, the occupancy octree is an useful metric representation to perform path planning in 3D environments by successive navigating over empty octants and avoiding the environment obstacles. However, being represented by huge number of cells the octree prevents efficient coverage in large-scale environments. This observation made us to follow "space-gate" reasoning paradigm and to support corresponding topological scheme. Both spaces and gates are considered as non-overlapping, simply connected subsets of empty cells of the occupancy octree. Principal difference between introduced categories is that the spaces approximate large free regions of the environment, whereas the gates — small narrow regions.

Therefore, the objective of this study is to extract a topological scheme from available metric information. This problem attracted many researches which tried to obtain adequate topological scheme by identifying features like virtual doors, narrow passages, corners, middle lines [9]. Although their methods work well for particular domestic cases, scarcely that they match to complex environments simulating real construction projects. The proposed algorithm for extracting a "space-gate" topological scheme looks very promising from this point of view.

Underlying principle for detecting free regions is to find the cells where the distance field reaches local maxima. Each such maximum originates a subset of empty, simply connected cells surrounding it and having distance values not exceeding the local maximum. Opposite to centerline algorithms oriented on rectangular occupancy grids and steepest descent by gradient vector approximations, our algorithm is applicable to arbitrary irregular grids and octrees which of special value for the discussed global path planning problems.

To explain how the algorithm works, let us define binary relations of the adjacency, dominancy and origination among empty cells of the occupancy octree C. The cells $c', c'' \in C$ are adjacent (or $c' \sim c''$) if and only if there is a common face belonging to boundaries of both cells. The cell $c' \in C$ dominates over the cell $c'' \in C$ (or $c' > c''$) if and only if the cells are adjacent and estimated distance value for the cell c' is larger than the corresponding value for the cell c''. And, finally, the subset $C' \subseteq C$ is originated from the cell $c' \in C'$ ($c' \triangleright C'$) if and only if for any $c'' \in C'$ there is a sequence $c_1, c_2, ..., c_n \in C'$ so that $c' > c_1, c_1 > c_2, ..., c_n > c''$ and there is no dominating cell $c''' \in C'$ for c' so that $c''' > c'$. The introduced adjacency relation is symmetric, reflexive and transitive. The dominancy relation is transitive.

We define the regions to be subsets of cells $R_1, R_2, R_3, ... \subseteq C$ obtained by transitive closure of the dominancy relation on a set of all empty cells of the octree C. Then the spaces $S_1, S_2, S_3, ... \subseteq C$ can be defined as non-intersecting subsets of regions $S_1 = R_1 \setminus \{R_2 \cup R_3 \cup ...\}$, $S_2 = R_2 \setminus \{R_1 \cup R_3 \cup ...\}$, $S_3 = R_3 \setminus \{R_1 \cup R_2 \cup ...\}$. Consider now a remained subset $G = \{R_1 \cap R_2\} \cup \{R_1 \cap R_3\} \cup \{R_2 \cap R_3\} ... \subseteq C$ representing all the intersected regions with more than one originating cell. Cells of the subset G with the same combination of originating cells are grouped to form the gates $G_1, G_2, G_3, ... \subseteq G \subset C$. One would assume that the extracted spaces and gates have no mutual intersections $C_i \cap C_j = \varnothing$, $G_k \cap G_l = \varnothing$, $C_i \cap G_k = \varnothing$, $i \neq j$, $k \neq l$ and they form an exact cover of the original set $S_1 \cup S_2 \cup ... \cup G_1 \cup G_2 \cup ... = C$.

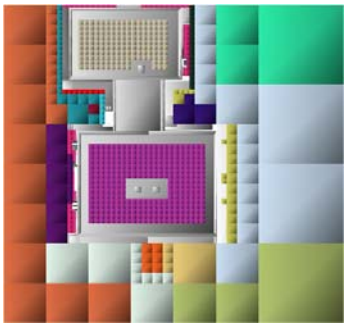


Figure 3: Space and gate regions identified in the occupancy octree.

The introduced definitions and obtained formula give a constructive algorithm to identify spaces and gates and to extract a topological map from the metric representation. An important advantage of the algorithm is an avoidance of any domestic feature analysis and its applicability to both indoor and outdoor environments. An example of the space and gate identification is shown in Figure 3.

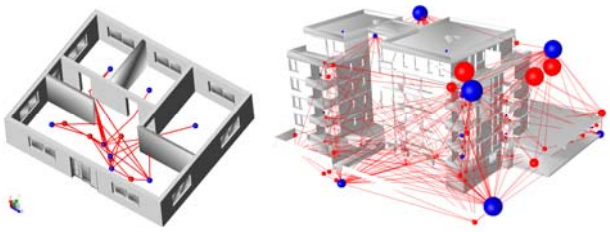


Figure 4: A bipartite "space-gate" topological graph for indoor and outdoor environments.

All the recognized spaces and gates match to corresponding vertices of the generated bipartite topological graph. Space vertices are connected by edges with incident gate vertices, but not with other space vertices. Similarly, gate vertices are connected with incident space vertices, rather than with other gate vertices. Thus, spaces and gates are alternated when navigating

over environment and traversing the topological graph. A bipartite topological graph generated for indoor and outdoor environments is shown in Figure 4.

4. SOME EXPERIMENTAL RESULTS

Using the "space-gate" topological map and graph search algorithms like classical Dijkstra's and Tarjan's algorithms, we can determine what the possible path to get to a certain destination is. The spatial path can be easy formed from waypoints lying in the centers of incident spaces and gates associated with the found route. Then the path is checked against potential collisions and, if necessary, it is corrected using known modification of the RRT algorithm that assumes growing trees from neighboring waypoints in opposite directions. For details see randomized kinodynamic planning [8]. Figure 5 provides an example of the resulting path in the presented building model obtained using both global and local planning strategies.

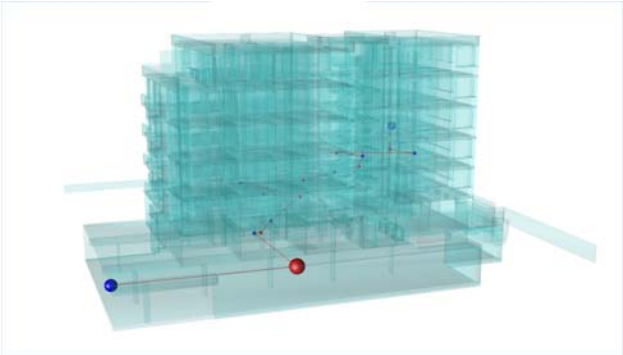


Figure 5: A resulting path obtained using both global and local strategies.

To validate the topological mapping method proposed and to estimate its practical benefits, we have implemented a program and conducted timing experiments, in which a middle-size building model was applied. Although the model was strongly structured in accordance with IFC standard, in fact a polygon soup consisted from about 500,000 triangles was taken as original 3D geometry data. A series of the experiments corresponded to different tolerance values of the metric representation limited by the regular grid sizes.

The experiments showed that the method demonstrates polynomial complexity grow avoiding any exponential explosion. For the grid 52 x 56 x 30 the analysis took totally 107 CPU seconds on a typical computer configuration Core 2 Duo E8600 processor (2.13 GHz), 2GB of RAM (800 MHz). After the topological map has been constructed, search in the graph and final validation of the found spatial routes took no more than a few seconds that looks like quite promising result giving an opportunity to interactively resolve path planning problems.

5. CONCLUSION

Thus, the effective method of generating topological maps for large-scale 3D environments has been proposed. The method utilizes occupancy octree structures and uses original criteria for identifying spaces and gates. Thus generated maps provide for whole coverage of environments and enable to effectively resolve multiple path planning requests using well-known graph search algorithms. Conducted experiments proved the feasibility and effectiveness of the method presented as well as its suitability to global path planning problems in complex indoor-outdoor environments.

6. REFERENCES

[1] Alero A., Milln J.R and Floreano D. *(1999) "Efficient learning of variable-resolution cognitive maps for autonomous indoor navigation." IEEE Trans. on Robotics and Automation, 15(6), 990-1000*

[2] Chen D.Z., Szczerba R.J. and Uhran J.J. *(1997) "A framed-quadtree approach for determining euclidean shortest paths in 2D environment" IEEE Trans. on Robotics and Automation, 13(5), 668-680.*

[3] Choi J. W., Choi M. Y. and Chung W. K. *(2009) "Incremental Topological Modeling using Sonar Gridmap in Home Environment." In International Conference on Intelligent Robots and Systems, 3582-3587.*

[4] Fabrizi E. and Saffiotti A. *(2000) "Extracting Topology-Based Maps from Gridmaps." In International Conference on Robotics and Automation, 2972-2978.*

[5] Kim J., Lee S. *(2009) "Fast neighbor cells finding method for multiple octree representation" In IEEE Int. Symposium on Computational Intelligence in Robotics and Automation. 540-545.*

[6] Kraetzschmar G., Sabaltong S., Enderle S. and Palm G. (2000) *"Application of neurosymbolic integration for environment modeling in mobile robots", in Wermter and Sun (eds) Hybrid Neural Systems, Springer, Berlin.*

[7] Lamarche F. *(2009) "TopoPlan: a topological path planner for real time human navigation under floor and ceiling constraints." Computer Graphics Forum. 28 (2). 649-658.*

[8] Lavalle S. M. *(2006) "Planning algorithms." Cambridge University Press, Cambridge, UK.*

[9] Myung H., Jeon H. M., Jeong W. Y. and Bang S. W. *(2009) "Virtual Door-Based Coverage Path Planning for Mobile Robot." In Federation of International Robot-soccer Association, 197-207.*

[10] Semenov V.A., Kazakov K.A., Morozov S.V., Tarlapan O.A., Zolotov V.A., Dengenis T. *(2010) "4D modeling of large industrial projects using spatio-temporal decomposition." In eWork and eBusiness in Architecture, Engineering and Construction, eds. K. Menzel and R. Scherer, 89-95.*

[11] Szczerba R.J., Chen D.Z., Uhran J.J. *(1998) "Planning Shortest Paths among 2D and 3D Weighted Regions Using Framed-Subspaces" International Journal of Robotics Research. 17(5). 531-546.*

[12] Thrun S., Bucken A., Bulgard W., Fox D., Frohlinghaus T., Hennig D., Hofmann T., Krell M. and Schmidt T. (1998) *"Map learning and high-speed navigation in RHINO" MIT/AAAI Press, Cambrige-MA.*

[13] Zelinsky A. *(1992) "A mobile robot navigation exploration algorithm", IEEE Trans. on Robotics and Automation, 8, 707-717.*

3D Curve-Skeletons Extraction and Evaluation

Denis Khromov, Leonid Mestetskiy*
Department of Computational Mathematics and Cybernetics
Moscow State University, Moscow, Russia
denis.v.khromov@gmail.com, l.mest@ru.net

Abstract

A novel definition of the 3D curve-skeleton is presented. Many existing approaches to the problem can be formalized in the given definition. The main advantage of the presented mathematical model is that it allows strict quality assesment of the produced curve-skeleton. The definition is based on the usage of fat curves. A fat curve is a 3D object which allows to approximate tubular fragments of the shape. A set of fat curves is used to approximate the entire shape; such a set can be considered as a generalization of the 2D medial axis. An example algorithm which obtains curve-skeletons due to the given definition is also presented. The algorithm is robust and efficient.

Keywords: *curve-skeleton, medial axis, fat curve, shape analysis.*

1. INTRODUCTION

The medial axis, first introduced in [1], has been proved to be very useful for 2D shape analysis. The medial axis of a closed bounded set $\Omega \subset \mathbb{R}^2$ is the set of points having more than one closest point on the boundary; or, equivalently, the medial axis is the set of centers of the maximum inscribed in Ω circles. The medial axis is a graph embedded in $\mathbb{R}^2$. This graph emphasizes geometrical and topological properties of the shape Ω. Such graphs are usually called skeletons. There are efficient algorithms for 2D medial axis computation.

It would be natural to try use the same approach for 3D shape analysis. A medial axis of a 3D shape $\Omega \subset \mathbb{R}^3$ is a set of points having more than one closest point on the boundary. But such an object is not a graph since it may contain 2D sheets [2]. Those sheets may be very complex, and there are some methods which try to simplify the inner structure of the medial axis in 3d [3]. Therefore 3D medial axis is as difficult for the processing as the initial shape Ω. So there is a problem: how to define a skeleton of a 3D shape as a graph embedded in $\mathbb{R}^3$ so that this graph would have all of the advantages of the 2D medial axis?

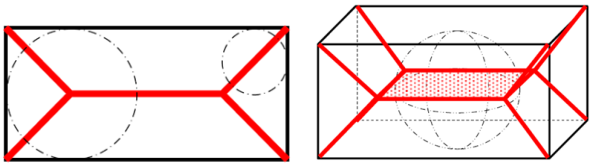


Figure 1: Medial axis of a 2D rectange and a 3D box.

Such graphs are called curve-skeletons. To date, there are lots of publications on curve-skeletons. However, unlike 2D case, where the strict mathematical definition of the medial axis was given decades ago, the definition of a 3D curve-skeleton still hasn't been presented. Usually, curve-skeleton is defined as the result of applying some algorithm to the 3D shape. There is no way to compare these algorithms with each other because their working principles and results may have totally different nature. It's very difficult to evaluate the quality of the skeletons produced by those algorithms, since there is no formal criterion for such an evaluation. There's only visual evaluation, which is subjective and not mathematical at all.

In [4][5] the authors presented the classification of curve-skeleton algorithms. The curve-skeleton is intuitively defined as a 1D thinning of the 3D object. The authors also made a list of possible properties of a curve-skeleton. Some of these properties are strict (for example, topological equivalency between curve-skeleton and the original shape), others are intuitive and should be formalized (centeredness of the skeleton and possibility of reconstruction of the original 3D object). Almost every published algorithm computes skeletons which have some of these properties. In such cases, these properties are considered to be advantages of the algorithm.

One of the popular approaches is based on the thinning of voxel images. Such thinning may be done directly (deleting boundary voxels step-by-step, [6] [7]) or with the distance function [8]. The skeletons produced by such methods are not continuous but discrete objects. Algorithms of this class are not universal because they're applicable to the voxel images only. Finally, there is no mathematical criterion to evaluate and compare different techniques of thinning.

It seems natural to try to extract 1D curve-skeleton from the 2D medial axis. The medial axis itself is a very complicated object, which consists of quadratic surfaces, so it's usually replaced by some approximation. However, extraction of 1D piece from the medial axis usually based on some successfully found heuristic. For example, in [9] such an extraction is done with the help of the geodesics on the boundary surface. As in the previous case, there's no strict criterion for evaluation and comparison of various heuristics of a 1D curve-skeleton extraction.

There are some other techniques used to compute curve-skeleton, such as usage of optimal cut planes [10] or physical interpretation of the problem [11]. However, these methods are also successfully found heuristics which allow to compute some object visually corresponding to the human idea of the curve-skeleton. And again, formal mathematical evaluation of these algorithms doesn't seem to be possible.

In this paper, a strict definition of the curve-skeleton is given. The model being proposed

1. allows to evaluate the correspondence between the curve-skeleton and the original object;
2. approximates the given shape with a fixed precision;
3. doesn't depend on the type of the shape description (polygonal model, voxel image or point cloud).

Also, an algorithm which computes the curve-skeleton according to the definition, is presented.

*This is supported by RFBR, grant 11-01-00783.

2. DEFINITIONS

Let $\mathscr{C}$ be a set of smooth curves in $\mathbb{R}^3$. For every curve $c \in \mathscr{C}$, there is a set $\mathscr{R}_c$ of continuous non-negative functions defined on c.

Definition 1 *A fat curve is a pair (c, r), where $c \in \mathscr{C}, r \in \mathscr{R}_c$. The curve c is said to be an axis of the fat curve, and the function r is its radial function.*

Definition 2 *An image of the fat curve (c, r) is a set of points*

$$I(c, r) = \{\mathbf{x} \in \mathbb{R}^3 | \exists \mathbf{y} \in c : \rho(\mathbf{x}, \mathbf{y}) \leq r(\mathbf{y})\}. \quad (1)$$

Definition 3 *A boundary of the fat curve (c, r) is a set of points*

$$\partial I(c, r) = \{\mathbf{x} \in I(c, r) | \forall \mathbf{y} \in c : \rho(\mathbf{x}, \mathbf{y}) \geq r(\mathbf{y})\}. \quad (2)$$

The fat curve is an object which is very convenient to approximate tubular 3D shapes (see Fig. 2).

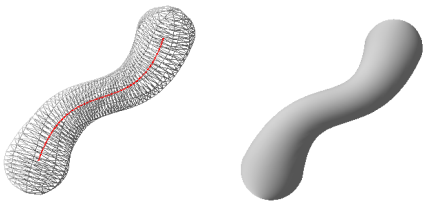


Figure 2: Fat curve.

Definition 4 *Let C be a set of fat curves such that axis of these fat curves intersect each other in their endpoints only. A fat graph F over a set C is a graph whose edges are fat curves from C and vertices are endpoints of their axis.*

Definition 5 *A boundary ∂F of a fat graph F is an union of boundaries of the fat curves composing F.*

Let $\mathscr{F}_{\mathscr{C}}$ be a set of all possible fat graphs.

Consider an embedded in $\mathbb{R}^3$ connected 3D manifold Ω with the boundary $\partial\Omega$. We'll approximate Ω with some fat graph.

Definition 6 *A distance between the point $\mathbf{x} \in \mathbb{R}^3$ and the fat graph F is a distance between $\mathbf{x}$ and the closest point on $\mathscr{F}_{\mathscr{C}}$'s boundary:*

$$\rho(\mathbf{x}, G) = \min_{\mathbf{y} \in \partial F} \rho(\mathbf{x}, \mathbf{y}). \quad (3)$$

Definition 7 *A distance between a manifold Ω and a fat graph F is a value*

$$\varepsilon(\Omega, F) = \int_{\mathbf{x} \in \partial\Omega} \rho^2(\mathbf{x}, F) dS. \quad (4)$$

Approximation quality can be evaluated by two values: distance $\varepsilon(\Omega, F)$ and complexity of the fat graph F. A complexity of a fat graph can be evaluated as

1. sum of lengths of axis of fat curves composing the fat graph;
2. number of the fat curves.

If the set $\mathscr{C}$ is rather wide, it's better to use the first criterion in order to avoid too crooked curves. However, if $\mathscr{C}$ is narrow and doesn't contain such curves, the second criterion can be used since it's very simple.

In these terms, the problem of approximation with a fat graph can be defined as following

1. compute an approximation with the smallest possible $\varepsilon(\Omega, F)$ and a fixed fat graph complexity;
2. compute an approximation with the least possible complexity and
$$\varepsilon(\Omega, F) < \varepsilon_0, \quad (5)$$
where ε_0 is a fixed value.

The fat graph can also be defined for planar curves. 2D medial axis would be an example of such a graph if we define the radial function at a point $\mathbf{x}$ equal to the distance from $\mathbf{x}$ to the boundary. Image of this special graph coincides with the whole shape, so its approximation error is zero.

3. IMPLEMENTATION

The main issue which wasn't discussed in the previous chapter but seems to be very important in the practical implementations of the method is how to choose the first approximation of the skeleton. It's possible to use any existing algorithm which produces curve-skeletons. But the proposed scheme has the advantage that the first approximation of the skeleton doesn't have to be very nice and accurate. It can be easily fitted into the shape afterwards using numerical methods.

That means that we can use some algorithm which is inaccurate but very fast, hoping to improve it during the fat graph fitting. One way to do so is to use the 2D medial axis of some Ω's planar projection. There're some facts in favor of this decision.

- It seems that medial information plays the key role in the human vision and visual perception[5]. But the human vision is planar, so if some 3D graph feels to be a good curve-skeleton, its projection would be also considered as a graph which is very close to the 2D medial axis of the object.
- 2D medial axis is a well-defined and examined object. There are fast and robust algorithms for 2D skeletonization.
- As mentioned above, the 2D medial axis can be considered as a fat graph which has zero approximation error. It's possible to try to bring this property in 3D as close as possible.
- 2D medial axis of a polygonal shape consists of smooth curves of degree 1 and 2, which can be fitted in the manner described above.

An example of the object and its planar projection is shown in the Fig. 3.

Each knot of the 2D skeleton is a projection of at least 2 points on the surface (see Fig. 4). A maximal inscribed ball tangent to the surface $\partial\Omega$ at those two points is a good mapping of the knot into the 3D space.

The main problem of this method is the possibility of occlusions. For example, if one of the legs of the horse in Fig. 3 was occluded by another one, the usage of this particular projection would lead to an incorrect result. For some models it's possible to find a projection which gives no occlusions. The incorrect projection with occlusions produces significant approximation error. But for some

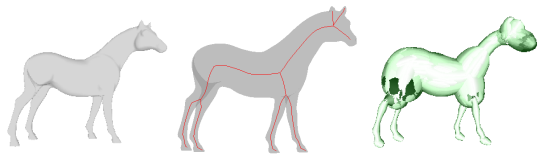


Figure 3: A model of a horse (left), its orthogonal planar projection with the 2D medial axis (center) and the approximating fat graph (right).

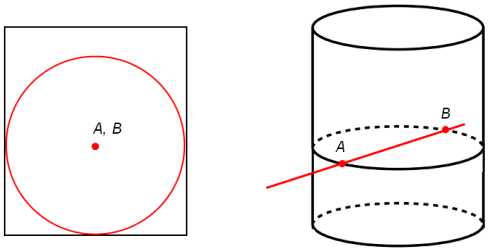


Figure 4: 2D projection (left) of a 3D cylinder (right); center of the maximum inscribed circle is a projection of points A, B.

shapes it's impossible (or very difficult) to find a good projection with no serious occlusions. This problem is solved by the preliminary segmentation. The shape is divided into tubular segments. Each segments is approximated with its own fat graph produced by some particular planar projection. Finally, all these partial fat graphs are joined into one.

The segmentation is defined by a set of points

$$Q_s = \{\mathbf{q_1}, \ldots, \mathbf{q_s}\}, \mathbf{q_i} \in \partial\Omega. \quad (6)$$

Let ρ_Ω be a geodesic distance on the surface $\partial\Omega$. Then each segment S_i is defined as a set of points closest to $\mathbf{q_i}$:

$$S_i = \{\mathbf{x} \in \partial\Omega | \forall k, 1 \leq k \leq s, \rho_\Omega(\mathbf{x}, \mathbf{q_i}) \leq \rho_\Omega(\mathbf{x}, \mathbf{q_k})\}. \quad (7)$$

An example is shown on the Fig. 5.

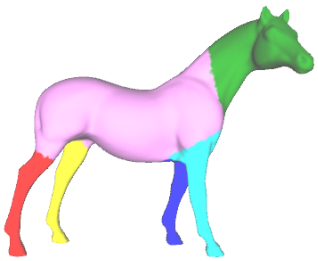


Figure 5: Segmentation of the model.

If the segmentation based on the set Q_s is not detailed enough, we can replace it with a new one

$$Q_{s+1} = Q_s \cup \{\mathbf{q_{s+1}}\}, \mathbf{q_{s+1}} = \arg\max_{\mathbf{x}\in\partial\Omega} \rho_\Omega(\mathbf{x}, Q_s), \quad (8)$$

where

$$\rho_\Omega(\mathbf{x}, Q_s) = \min_{1\leq i\leq s} \rho_\Omega(\mathbf{x}, \mathbf{q_i}). \quad (9)$$

The segmentation process starts with the set Q_2 which consists of two points which are most distant from each other.

The short summary of this chapter gives us the following scheme of the algorithm.

1. Segmentation of the model.
2. Choosing the best 2D projection for each segment (the word "best" means that this projection leads to the least value of the approximation error on the next step).
3. Computation of the approximating fat graph for each segment using the planar skeleton of the projection obtained on the previous step.
4. Join all of the fat graphs into one and final fitting using the numerical methods.

4. EXPERIMENTS

The described algorithm was successfully implemented. As mentioned above, it's impossible to compare the quality of the skeletons produced by other methods, since there has been no numerical criterion for evaluation of the difference between the curve-skeleton and the given shape. We'll prove the capacity of our approach in the way that is common in the literature on the curve-skeletons, which is based on the visual evaluation and experimental proof of the claimed properties.

First of all, we demonstrate the examples of curve-skeletons produced by the described algorithm (see Fig. 6). 3D models which have been chosen for the experiment are widely used to evaluate various computer geometry algorithms, so they're appropriate for the visual comparison with other methods.

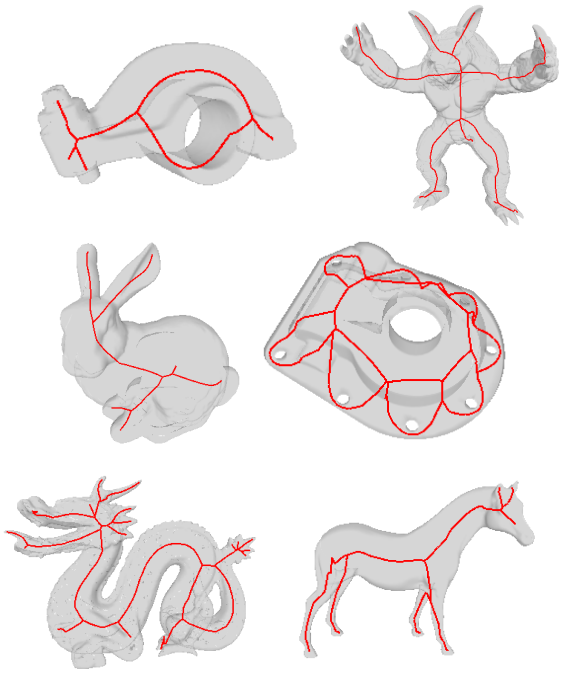
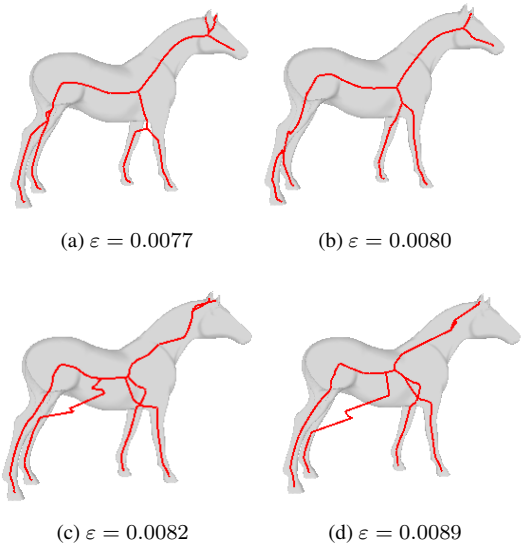


Figure 6: Examples.

It's useful to discuss the properties listed in [4]. Homotopy equivalence between the curve-skeleton and the shape is not guaranteed,

since the fat graph with relatively large amount of edges approximates wide non-tubular fragments of the shape with a loop consisting of a pair of edges. However, this property can be easily provided by more strict requirements for the topological class. Invariance under isometric transformations (i.e. transformations in which the distances between points are preserved) is obvious. The possibility of reconstruction of the original shape is provided by the definition of a fat graph: the image of the fat graph is a 3D manifold which approximates the original object with a known precision. The reliability (which means that every boundary point is visible from at least one curve-skeleton location) is not guaranteed, but it's achieved when the fat graph has enough edges. The robustness is implied by the robustness of the function $\varepsilon(\Omega, F)$.

In order to justify the meaningfulness of the function $\varepsilon(\Omega, F)$, which is the core part of the described method, we've prepared a number of various curve-skeletons of the same object. These skeletons were made without any fitting and with badly tuned parameters of the algorithm. The curve-skeletons and their corresponding approximation error values are shown on the Fig. 7. It's pretty obvious that the greater the value of $\varepsilon(\Omega, F)$, the worse the visual quality of the produced curve-skeleton. That implies that the proposed definition is not only theoretically grounded but also has some practical utility.



(a) $\varepsilon = 0.0077$ (b) $\varepsilon = 0.0080$

(c) $\varepsilon = 0.0082$ (d) $\varepsilon = 0.0089$

Figure 7: Curve-skeletons of the horse with different approximation error.

5. CONCLUSION

In the paper, a new mathematical model for curve-skeleton formalization was presented. This model allows to compare and research various approaches for the 3D skeletonization. Also, a new algorithm for skeletonization was described, implemented and discussed. The further research involves the following issues:

- elaboration of the model, in particular, approximation evaluation via Hausdorff metric;
- further development of the algorithm, better selection of the first approximation and formalization of the iterative fitting.

6. REFERENCES

[1] Harry Blum, "A Transformation for Extracting New Descriptors of Shape," in *Models for the Perception of Speech and Visual Form*, Weiant Wathen-Dunn, Ed., pp. 362–380. MIT Press, Cambridge, 1967.

[2] Peter Giblin and Benjamin B. Kimia, "A formal classification of 3d medial axis points and their local geometry," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, pp. 238–251, January 2004.

[3] Ming-Ching Chang and Benjamin B. Kimia, "Regularizing 3D medial axis using medial scaffold transforms," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2008, p. Accepted, IEEE Computer Society.

[4] Nicu D. Cornea and Deborah Silver, "Curve-skeleton properties, applications, and algorithms," *IEEE Transactions on Visualization and Computer Graphics*, vol. 13, pp. 530–548, 2007.

[5] Kaleem Siddiqi and Stephen Pizer, *Medial Representations: Mathematics, Algorithms and Applications*, Springer Publishing Company, Incorporated, 1st edition, 2008.

[6] Kálmán Palágyi and Attila Kuba, "A parallel 12-subiteration 3d thinning algorithm to extract medial lines," in *Proceedings of the 7th International Conference on Computer Analysis of Images and Patterns*, London, UK, 1997, CAIP '97, pp. 400–407, Springer-Verlag.

[7] Yu-Shuen Wang and Tong-Yee Lee, "Curve-skeleton extraction using iterative least squares optimization," *IEEE Transactions on Visualization and Computer Graphics*, vol. 14, pp. 926–936, July 2008.

[8] Alexandru Telea and Jarke J. van Wijk, "An augmented fast marching method for computing skeletons and centerlines," in *Proceedings of the symposium on Data Visualisation 2002*, Aire-la-Ville, Switzerland, Switzerland, 2002, VISSYM '02, pp. 251–ff, Eurographics Association.

[9] Tamal K. Dey and Jian Sun, "Defining and computing curve-skeletons with medial geodesic function," in *Proceedings of the fourth Eurographics symposium on Geometry processing*, Aire-la-Ville, Switzerland, Switzerland, 2006, pp. 143–152, Eurographics Association.

[10] Andrea Tagliasacchi, Hao Zhang, and Daniel Cohen-Or, "Curve skeleton extraction from incomplete point cloud," *ACM Trans. Graph.*, vol. 28, pp. 71:1–71:9, July 2009.

[11] Jen-Hi Chuang, Chi-Hao Tsai, and Min-Chi Ko, "Skeletonization of three-dimensional object using generalized potential field," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, pp. 1241–1251, November 2000.

ABOUT THE AUTHOR

Leonid Mestetskiy is a professor at Moscow State University, Department of Computational Mathematics and Cybernetics. His contact email is `l.mest@ru.net`.

Denis Khromov is a Ph.D. student at Moscow State University, Department of Computational Mathematics and Cybernetics. His contact email is `denis.v.khromov@gmail.com`.

Multithreaded approach for lossless LiDAR data compression

Domen Mongus, Denis Špelič, Borut Žalik
Faculty of Electrical Engineering and Computer Science, University of Maribor,
Smetanova ul. 17, SI 2000 Maribor, Slovenia
{domen.mongus, denis.spelic, zalik}@uni-mb.si

Abstract

Light detection and ranging (LiDAR) has the capability of capturing a huge amount of highly accurate spatial data. However, the size of the data causes a lot of problems associated with its exchange and storage. In this paper, a method for lossless LiDAR data compression is presented. Although, efficient methods in terms of compression ratio have already been proposed, their time efficiency can be improved. For this purpose, a multithreading schema was developed to increase the use of the computer resources (i.e. multi-core central processor unit and direct memory access). In this way, the overall compression time has been reduced over 70%.

Keywords: *LiDAR, multithreading, lossless compression, predictive coding.*

1. INTRODUCTION

In the recent years, light detection and ranging (LiDAR) has become one of the prime remote sensing technologies [1, 2, 3]. Most of its capabilities arise from the use of the laser light to measure the range from the distant objects. The range is calculated based on the time delay between the transmission of the laser pulse and detection of its reflection [4]. Since a short wavelength signal is used, LiDAR achieves high accuracy and performs the measurements extremely fast [5]. As such, it is able to capture large amount of highly accurate and dense data in a short time. Because of this, it becomes one of the most widely used techniques within a wide range applications [6, 7, 8].

LiDAR is especially important in geosciences, where relatively large Earth' s surface can be gathered by airborne LiDAR systems[1, 2, 3, 6, 7, 8]. The airborne LiDAR systems are mounted on aircrafts and obtain geographical position of measured points by the use of the global positioning system (GPS) and inertial measurement unit (IMU) [5]. GPS is used to define the position of the aircraft, while IMU measures the roll, the pitch, and the heading of the aircraft. By that, and by measuring the scan angle, the angular orientation of each point is established and thus, the position of the point can be defined (see Fig. 1). Furthermore, such systems are capable to distinguish between different reflections of the emitted laser pulse. In this way, they can retrieve some points from the Earth's surface even below the vegetation [4].

The gathered points are usually saved in a LAS file, which represents the industrial standard for storing and exchanging LiDAR data [9]. In a LAS format, points are represented by *xyz*-coordinates. Scalar values are associated with each point thet represents, for example, an intensity, a reflection number or user specified data. Exact specification depends on the version of the LAS format. Because LiDAR systems can perform over 200.000 measurements per second (which allows retrieving over 35 points per square meter), such files become extremely large. They often contain several tens of millions of points and their size can easily reach more than a few gigabytes. Therefore, to store such files is difficult, while their exchange over the local networks and internet is practically impossible. Because of this, the compression of LiDAR data is of great relevance to the remote sensing community [5]. However, as huge files have to be compressed (and decompressed), the time for compression represents another important factor for efficient maintaining of the data.

In this paper, we present a multithreaded approach for LiDAR data compression, which exploits multi-core central processor units to speed up the compression process. In section 2, related work on LiDAR data compression is briefly presented. Brief description of the used compression method is given in section 3. In section 4, the multithreaded approach is explained in details. The results are presented in section 5. Section 6 concludes the paper.

2. RELATED WORK

Early works on point compression were closely related to compressing the topology of the triangular meshes. Thus, one of the earliest methods presented in [10] uses triangular mesh, beside which the geometry of points is compressed. The method uses a prediction of the next points position. For this purpose, the triangular mesh is divided into stripes. Points are then compressed based on the order of their appearances in such stripes, where the position of the next point is encoded using linear prediction schema [11]. In this way, only the differences between predicted and actual positions of points are stored. Since this approach compresses the topology as well as the geometry of triangular mesh, it is not suitable for LiDAR data, where neighbouring relations of points are not known (we are talking about unstructured points). The efficient compression of unstructured point-cloud usually requires to find the points that are close enough in the space, to enable utilization of the prediction paradigm. The method described in [12] establishes a correlation between the points by using an octree-based space partitioning schema. This algorithm predicts the location of the next point with regard to the approximated planes in the octree cells. A similar spatial hierarchy is used in the algorithm of Huang et al. [13]. It also supports the compression of scalar values attached to the points. Both, the point coordinates and the attached scalar values are represented by using an average value in the surrounding tree cells. Other approaches apply prediction vectors to estimate the coordinates of the next point in the stream. For example, the algorithm proposed in [13] predicts the location of the next point using the vector between the last two points before the observed one. Besides this, they introduced additional rules for rotating of the prediction vector.

The algorithm proposed in [14] is one of the earliest, aimed for LiDAR data compression. It utilizes the Delaunay triangulation [15] for planes approximation and the wavelet transform to update the values of the points coordinates. This method is very efficient with regard to the compression ratio but, simultaneously, it is slow and lossy. The algorithm proposed by Isenburg [16] is also intended to compress LiDAR data. At the moment, it is accessible only as a demo program, while the implementation details are not known to the public. However, in terms of compression ratio, even more efficient algorithm was presented by Mongus and Žalik in [17] (execution version is available at [18]). To seed-up the compression process, a multithreaded implementation is suggested in this paper.

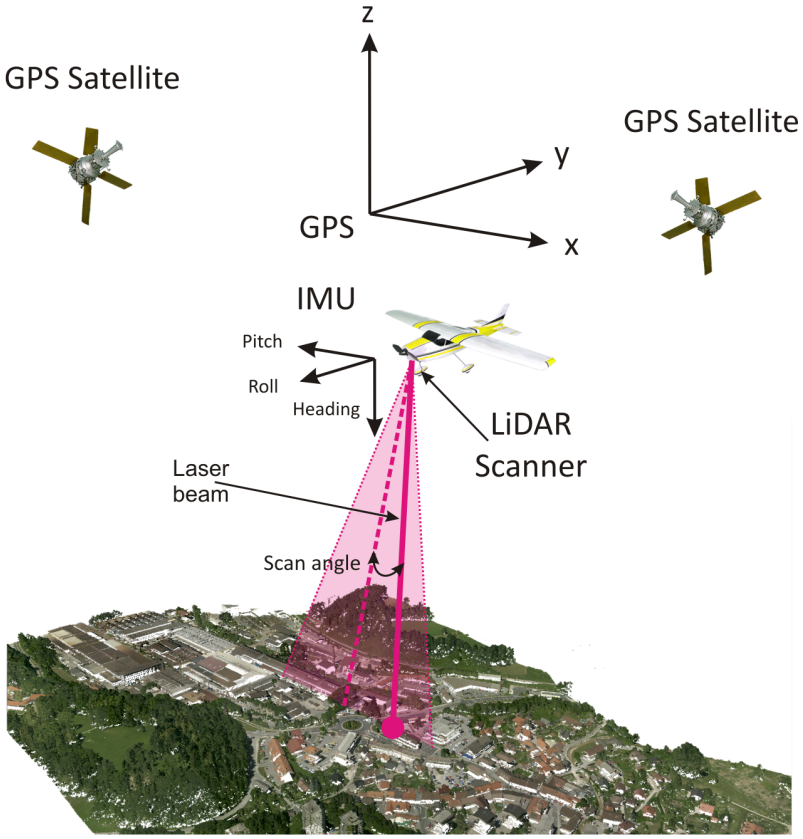


Figure 1: Gathering of airborne LiDAR data.

Because of this, this method will be briefly explained in the next section.

3. LOSSLESS LIDAR DATA COMPRESSION ALGORITHM

The considered method for compression of LAS files uses domain-specific information about the LiDAR data gathering. In this way, the correlation between points can be established without an additional space partitioning. The method works in three steps:

1. Points are encoded with a predictive coding scheme.
2. The errors in the prediction are coded with the variable-length-coding (VLC).
3. VLC values are compressed with arithmetic coder (AC) and stored in the output file.

In the prediction model, three prediction rules are used to estimate the positions of the points, while constant prediction rule is used do predict their scalar values (details are given in [17]):

- Constant prediction rule presumes that the values of the same attribute of two successive points are the same. Because the consequent scalar values of LiDAR points are often very similar, the constant prediction rule is highly efficient. On the other hand, the positions of successive points are never the same. Therefore different prediction rules are proposed for them.
- Prediction rule for x-coordinate uses the average distance between x-coordi-nates of the successive points. In addition, the deviation of the last few samples (100 have been used in our case) are used, and by the help of the linear interpolation, included in the final prediction.
- Prediction rule for y-coordinate exploits the difference between successive points x-coordinate to predict the y-coordinate of the coded point. Usually, the large distance in x-coordinate results in a large distance in y-coordinate, too. Thus, the history of the coded points is searched to find two successive points with a similar difference in x-coordinate. If such points are found, their differences in y-coordinate should match, too. If the match is not found, the linear approximation between previous y-coordinates is used for prediction.
- Prediction rule for z-coordinate applies a similar concept as the prediction for y-coordinate. However, it uses both x and y coordinates.

Because predictions made by described prediction model are accurate, the absolute values of prediction errors are small and the VLC can be efficiently applied. In the VLC step, the description byte is added to each value, where the information about sign and length (in bytes) of each value are stored. Thus, the zero bytes can be removed from each value. Furthermore, description bytes, as well as non-zero bytes of the same importance, are arranged in the separated byte streams. Each of those streams is then independently compressed by arithmetic coding (AC) [11] and stored to output file. This independency makes the algorithm suitable for multiprocessor programming, introduced in the next section.

4. MULTITHREAD APPROACH FOR LIDAR DATA COMPRESSION

As already stated, most of the LiDAR point attributes are coded independently and therefore they can be processed simultaneously. Thus, by the use of multithreading, multi-core central process units (CPU) and direct memory access (DMA) [19] can be exploited for a considerable reduction of the time needed for LiDAR data compression. In the Fig. 2, the scheme of the proposed approach is presented.

A set of LiDAR points, which represents an input in our method, is rearranged into a set of data-streams. Each data-stream contains the values of the same LiDAR point attribute. Because only predictions of *xyz*-coordinates are mutually dependent (prediction for *y* is dependent on *x* values and prediction for *z* is dependent on *x* and *y* values), one thread is created for each data-stream and one for processing *xyz*-coordinates. It is obvious that most of the CPU time is used to predict *xyz*-values and thus, corresponding thread should have a higher priority. In this way, the processing of the data-streams is more synchronized and consecutively less CPU time is lost in the last synchronisation step. However, after the predictive coding step for the *xyz*-coordinates is completed, VLC can be preformed for *xyz*-coordinates simultaneously as well. Therefore, two more threads are created (for *y*-coordinates and *z*-coordinates), while the priority of the current one (*x*-coordinates) is set back to normal. Furthermore, in the VLC step, each data-stream is further split into four byte-streams. Because each byte-stream can be handled by AC independently, additional threads are created (see Fig. 2).

In the last step, the threads are synchronised and compressed data is stored in the output file. Since it is time consuming to write the data to the out file, a pipeline is created for using DMA to speed up the data storing. Thus, in the first step, each of the threads enters the queue for the AC. One after another, byte-streams are accepted by the AC and the corresponding threads are killed. When processing of a byte-stream is terminated by AC, the data is passed to the DMA. Because DMA does not use the CPU for writing, the CPU can process the next byte-stream. In this way, the arithmetic coding and the data storing are performed simultaneously, reducing the overall data compression time. However, because we cannot predict which of the threads finishes the first, the order of the byte-streams is stored in the output file, too. Nevertheless, the number of threads is limited and the space needed to store the order of the byte-streams is irrelevant in regards to the total file size.

5. RESULTS

The presented multithreading approach was tested against the original method accessible at [18]. Both implementations were done in C# under MS Windows 7 Profession. The time efficiency of multithreading approach is demonstrated on five LAS files that contain different number of points. Tests were performed on computer system with Intel Core 2 Quad CPU Q6600 2.45 GHz, 4 GB of RAM. The results are presented in Table 1.

From the results we can see that the proposed multithreaded approach is, for over 70%, faster than the original approach. The CPU utilization is increased from 24% to 94%. The original method has not been designed for multi-core CPUs. Because of this, its utilisation cannot be higher than *1/N*, where *N* is the number of CPU cores. Thus, in our case (*N=4*) the maximal utilization is 25%. When dealing with multi-core CPUs utilization, the distribution of computational workload should be as equal as possible. With the presented approach, the CPU utilization confirms that almost optimal workload distribution has been achieved. The difference be-

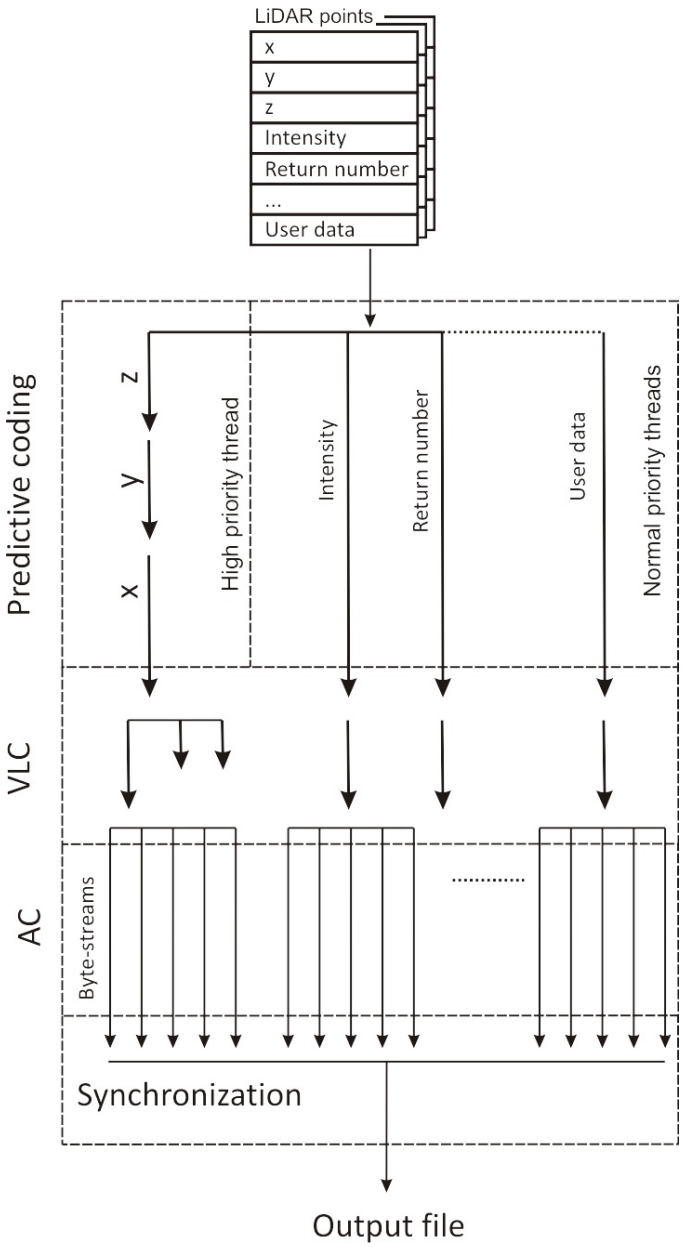


Figure 2: Multithread aproach for LiDAR data compression.

Table 1: Time efficiency of lossless LiDAR data compression algorithms

	LAS File 1	LAS File 2	LAS File 3	LAS File 4	LAS File 5
Original file size (kB)	64.194	101.649	108.037	175.304	429.743
Compressed file size(kB)	9.887	9.289	14.339	20.250	40.455
Number of points	2.345.998	3.581.247	3.951.030	6.411.089	15.716.290
Original method	14.85 sec.	19.95 sec.	24.35 sec.	36.82 sec.	142.80 sec.
Multithreaded method	4.30 sec.	5.66 sec.	7.18 sec.	10.35 sec.	39.60 sec.
Time reduction	**71.0%**	**71.6%**	**70.55%**	**71.9%**	**72.3 %**

tween optimal utilisation and the achieved utilisation is due to the additional calculations needed for synchronizations of threads and thread scheduling. Because the thread synchronisation time is equal regardless to the file size, slight increase in the time reduction can be noticed when larger files are compressed. The only exception is LAS File 3, where data density is particularly low and worse compression ratio is achieved as well.

6. CONCLUSION

A new multithread compression schema for lossless LiDAR data compression has been presented in this paper. The proposed approach exploits multi-core CPU and DMA of modern computer systems to speed up the data compression. It has been shown that in these ways, the total compression time has considerably decreased, due to the multithreading. When dealing with multiple threads, different threads share the same cash. Thus, by proper synchronisation, less time-costly data transfers are required [19]. Furthermore, if one thread gets a lot of cache misses, other threads can still employ free computer resources that would otherwise be idle. In this way the overall compression time has been reduced over 70%.

7. REFERENCES

[1] F. Ackermann, "Airborne laser scanning - present status and future expectations," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 54(2-3), pp. 64–67, 1999.

[2] C. Briese, N. Pfeifer, and P. Dorninger, "Applications of the robust interpolation for dtm determination," *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 34, pp. 55–61, 2002.

[3] M. R. Belmont, "Application of non-uniform to uniform data mapping to: Shallow angle lidar with the introduction of independent variable techniques," *Signal Processing*, vol. 87(10), pp. 2461–2472, 2007.

[4] A. Wehr and U. Lohr, "Airborne laser scanning - an introduction and overview," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 54(2-3), pp. 68–82, 1999.

[5] D. F. Maune, *Land development handbook (3th ed.)*, chapter Aerial mapping and surveying, pp. 877–910, McGraw-Hill Professional, 2008.

[6] H. Lee, K. C. Slatton, B. E. Roth, and W.P. Cropper, "Prediction of forest canopy light interception using three-dimensional airborne lidar data," *International Journal of Remote Sensing*, vol. 30(1), pp. 189–207, 2009.

[7] S. Coveney, A. S. Fotheringham, M. Charlton, and T. McCarthy, "Dual-scale validation of a medium-resolution coastal dem with terrestrial lidar dsm and gps," *Computers & Geosciences*, vol. 36(4), pp. 489–499, 2010.

[8] J. L. Guerrero-Rascado, B. Ruiz, G. Chourdakis, S. A. Raymetrics, G. Georgoussis, and L. Alados-Arboledas, "One year of water vapour raman lidar measurements at the andalusian centre for environmental studies (ceama)," *International Journal of Remote Sensing*, vol. 29(17-18), pp. 5437–5453, 2008.

[9] American Society for Photogrammetry and Remote Sensing (ASPRS), "Las specification," avaliable at http://www.asprs.org/.

[10] G. Taubin and J. Rossignac, "Geometric compression through topological surgery," *ACM Transactions on Graphics*, vol. 17(2), pp. 84–115, 1998.

[11] D. Salomon, M. Giovanni, and D. Bryant, *Data Compression: The Complete Reference*, Springer, 2009.

[12] R. Schnabel and R. Klein, "Octree-based point-cloud compression," in *Eurographics Symposium on Point-Based Graphics*, M. Botsch and B. Chen, Eds. 2006, pp. 147–156, Eurographics Association.

[13] Y. Huang, J. Peng, and C. C. J. Kuo, "A generic scheme for progressive point cloud coding," *IEEE Transactions on Visualization and Computer Graphics*, vol. 14, pp. 440–453, 2008.

[14] B. Pradhan, S. Kumar, S. Mansor, A. R. Ramli, and A. R. B. M. Sharif, "Light detection and ranging (lidar) data compression," *KMITL Journal of Science and Technology*, vol. 5, pp. 515–526, 2005.

[15] B Žalik, "An efficient sweepline delaunay triangulation algorithm," *Computer-Aided Design*, vol. 37, pp. 1027–1038, 2005.

[16] M. Isenburg, "Lastools: converting, viewing, and compressing lidar data in las format," avaliable at: http://www.cs.unc.edu/∼isenburg/lastools/.

[17] D. Mongus and B. Žalik, "Efficient method for lossless lidar data compression," *International Journal of Remote Sensing*, vol. 32(9), pp. 2507 – 2518, 2011.

[18] D. Mongus and B. Žalik, "Las compression algorithm," avaliable at: http://gemma.uni-mb.si/lascompression/.

[19] M. Herlihy and N. Shavit, *The Art of Multiprocessor Programming*, Morgan Kaufmann, 2008.

ABOUT THE AUTHORS

Domen Mongus is a Ph.D. student at University of Maribor, Faculty of Electrical Engineering and Computer Science,Department of computer science. His contact email is `domen.mongus@uni-mb.si`.

Ph.D. Denis Špelič is a researcher at University of Maribor, Faculty of Electrical Engineering and Computer Science,Department of computer science. His contact email is denis.spelic@uni-mb.si.

Borut Žalik is a professor at University of Maribor, Faculty of Electrical Engineering and Computer Science,Department of computer science. His contact email is zalik@uni-mb.si.

Novel Peer Group Filtering Method Based On The CIELab Color Space For Impulse Noise Reduction

Yu–Ren Lai[a], Kuo–Liang Chung[a], Wei–Ning Yang[b], Chyou–Hwa Chen[a], and Le–Chung Lin[a]
[a]Department of Computer Science and Information Engineering
[b]Department of Information Management
National Taiwan University of Science and Technology
No. 43, Section 4, Keelung Road, Taipei, Taiwan 10672, R. O. C.
E-mail:{D9715011, k.l.chung}@mail.ntust.edu.tw, yang@cs.ntust.edu.tw, {similar, M9915011}@mail.ntust.edu.tw

Abstract

This paper presents a novel peer group filtering method, called the NPGF method, for impulse noise reduction. The main contributions of the proposed method are two-fold. First, we propose that the impulse noise detection is performed in the CIELab, instead of the RGB, color space to enhance noise detectability. Secondly, the proposed method employs two different-sized windows in determining the status for each pixel, alleviating the problems in correcting non-corrupted pixels in the neighborhood of edges in the textured regions. Based on three typical color images, experimental results demonstrate that the proposed NPGF method achieves better performance in noise detection when compared to the existing method.

Keywords: *Color image denoising, Impulse noise reduction, Peer group filter, CIELab color space.*

1. INTRODUCTION

The impulse noise reduction problems have been widely investigated because of their fundamental importance for image processing. During image acquisition or transmission, impulse noises are often introduced. Many impulse noise reduction methods have been developed [1]–[18]. These methods perform filtering operations on an image where the intensities of noisy or corrupted pixels are modified while preserving the intensities of non-corrupted or noise-free pixels to improve the image quality. The previous developed impulse noise reduction methods could be broadly classified into four categories — vector median based filtering method [1]–[3], fuzzy-logic based filtering method [4]–[7], switching filtering based methods [8]–[11], and peer group based methods [14]–[18].

In the category of vector median filtering methods, Astola *et al.* [1] proposed a VMF method, which is the earliest vector median based filtering method. The VMF method, an extension of the scalar median filter, is a vector processing technique and can be derived as a maximum likelihood estimation approach when the probability density is the double exponential. In the category of fuzzy-logic based filtering methods, fuzzy logic approach is used to deal with nonlinear image noise and process the inherent uncertainty in image structures. Camarena *et al.* [7] proposed a two-step fuzzy procedure which first performs a quick diagnosis based on the rank-ordered difference statistics [19], to determine the status of pixels in simpler cases. and then a strict diagnosis is used to deal with the pixels which are more difficult to classify. Then, the corrupted pixels are modified using the VMF method and the rest of the pixels is unchanged. In the family of switching filters, the methods are based on a detection-correction strategy where the filters are only applied to the corrupted pixels, indicating that the switching based filtering method can preserve more image edges. Jin and Li [8] proposed a switching filtering method by using quaternion rotation theory [12]. The methods in [10]–[11] use the difference between the central pixel and neighboring pixels in four directions to determine the status of central pixel. Neuvo and Ku [13] proposed the first peer group based filtering method. The central idea behind the peer group filtering method is that the pixel with significantly different intensity from those of neighboring pixels is more likely to be a noise. This method counts the number of neighboring pixels with similar intensities to deduce if the pixel is a noise. Thresholds on the similarity and the number of similar pixels are used and the pixel with small number of similar pixels is deduced to be a noise. Camarena *et al.* [17] proposed a fast peer group based filtering method in which a pixel is identified as non-corrupted when the size of peer group is larger than a threshold in the fuzzy metrics context. Morillas *et al.* [16] used a reduced ordering of color vectors to detect and replace the corrupted pixels for simultaneous reduction of impulse noises and preservation of the textured edges. Camarena *et al.* [18] further proposed a two-stage peer group filtering method, called the IFPGF method, to detect the corruption status of a pixel. In the first stage, a pixel is classified as either non-corrupted or undetermined. Only the undetermined pixels enter the second stage for further investigation. This two-stage method suffers from the problem of false alarm near the edges in textured regions of an image.

This paper presents a novel peer group filtering method, called the NPGF method. The ideas behind the proposed method are two-fold. First, we show that to achieve good pixel corruption detection, working in CIELab color space [20] achieves better corruption detection than other color spaces. Secondly and more importantly, we propose a novel two-window approach for detecting corrupted pixels which suppresses the false alarms near the edges in textured regions of images. Based on three typical colored images, experimental results show that the proposed NPGF method achieves better performance in noise detection when compared to the IFPGF method.

The rest of the paper is organized as follows. Section 2 presents in detail the IFPGF by Camarena *et al.* [18] and the proposed NPGF method. In Section 3, the experimental results are demonstrated to show the superiority of the proposed NPGF method. The final section concludes the paper.

2. THE PROPOSED PEER GROUP FILTERING METHOD BASED ON THE CIELAB COLOR SPACE

In this section, we first illustrate the basis of peer group filtering methods for impulse noise reduction, using the image shown in Fig. 1 as an example.

To simplify the discussion, gray values, instead of the values in the CIELab color space, are used in Fig. 1. To determine if a pixel is corrupted, a window of certain size (e.g., 5×5) is constructed with the pixel located in the center. The peer group corresponding to the

central pixel of a window is defined as the set of pixels which have similar gray values (e.g., difference in gray values less than or equal to some similarity-threshold) with the central pixel. If the size of the peer group is large (e.g., larger than or equal to some size-threshold) then we tend to classify the corresponding pixel as a non-corrupted pixel. Using similarity-threshold 5 and size-threshold 10, the central pixel in Fig. 1 is classified as a corrupted pixel since the size of the peer group is 3. Camarena *et al.* [18] proposed an improved fast peer group filtering method, called IFPGF, to increase the filtering speed. The IFPGF first divide the image into non-overlapping 5×5 windows. The central pixel of each window can either be classified as a non-corrupted if the peer group is large or an undetermined pixel. Once the central pixel is classified as a non-corrupted pixel, all the pixels in the corresponding peer group are classified as non-corrupted. Then, for each undetermined pixel, a 5×5 window is constructed to determine its status.

General peer group filtering methods suffer the problem of falsely deducing a pixel in a textured region as a corrupted pixel since the size of the corresponding peer group tends to be small. To alleviate this problem, we propose a novel two-stage peer group filtering method, called NPGF, which employs two different-sized windows to determine the peer group. In the first stage, a 5×5 window is constructed and the similar pixels with the central pixel are included in the peer group. In the second stage, for each of the eight pixels around the central pixel which are similar to the central pixel, a 3×3 window is constructed. For each 3×3 window, the pixels similar to the central pixel, excluding the identified similar pixels in the first stage, are included in the peer group corresponding to the central pixel in the first stage. Finally, the pixel with large peer group is deduced as a non-corrupted pixel; otherwise, a corrupted pixel. The central idea behind the proposed two-stage peer group filtering method is to use the central pixel of the 3×3 window as the bridge for capturing the gradual changes on the gray values of edges in the textured regions.

3	3	3	3	4
3	3	3	4	4
3	3	96	4	4
3	4	4	4	93
4	4	4	93	93

Figure 1: Example for illustrating peer group filtering.

The proposed impulse noise detection is performed in the CIELab, instead of the RGB, color space to enhance the noise detectability since the color distance between neighboring pixels is larger in the CIELab color space. To transform an RGB-based input image into the CIELab color space, Hunt [20] first transforms the image from the RGB color space into the CIEXYZ color space by

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0.4124 & 0.3575 & 0.1804 \\ 0.2126 & 0.7151 & 0.0721 \\ 0.0193 & 0.1191 & 0.9502 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}. \quad (1)$$

Then the CIEXYZ-based image is converted into the CIELab-based image by

$$L = 116 \times f(Y/Y_n) - 16$$

$$a = 500 \times [f(X/X_n) - f(Y/Y_n)]$$

$$b = 200 \times [f(Y/Y_n) - f(Z/Z_n)],$$

with

$$f(t) = \begin{cases} t^{\frac{1}{3}}, & \text{if } t > (\frac{6}{29})^3, \\ \frac{1}{3}(\frac{29}{6})^2 t + \frac{4}{29}, & \text{elsewhere,} \end{cases}$$

where $(X_n, Y_n, Z_n) = (95.047, 100.00, 108.883)$ is the position of the white point in the CIEXYZ color space. The color distance in CIELab color space between pixels p and q is defined as

$$d(p,q) = \sqrt{(L_p - L_q)^2 + (a_p - a_q)^2 + (b_p - b_q)^2},$$

where (L_p, a_p, b_p) and (L_q, a_q, b_q) are coordinates of pixels p and q, respectively, in the CIELab color space.

Denote by $|PG(p)|$ the size of the peer group corresponding to pixel p. Let $W_i(p)$ denote the window of size $i \times i, i = 3, 5$, with central pixel p. The pseudo-code for determining if pixel p is corrupted and the following noise-reduction operation using the proposed NPGF can be described as follows.

1. Determine $PG(p)$:
   - 1.1 For $p_i \in W_5(p)$ and $p_i \neq p$, if $d(p_i, p) \leq t_s$ then $p_i \in PG(p)$.
   - 1.2 For $p_j \in PG(p)$ and $p_j \neq p$, construct $W_3(p_j)$. For $p_k \in W_3(p_j)$, $p_k \notin PG(p)$, and $p_k \neq p_j$, if $d(p_k, p_j) \leq t_s$ then $p_k \in PG(p)$.
2. If $|PG(p)| \geq t_z$ then deduce pixel p as non-corrupted.
3. For each deduced corrupted pixel, replace its CIELab coordinates by applying the arithmetic mean filtering operation on a 3×3 window centered at the corrupted pixel.

Since the proposed NPGF method alleviates the problem of misidentifying non-corrupted pixels as corrupted in the textured regions, it can preserve, as supposed to, the edges in the textured regions.

3. EXPERIMENTAL RESULTS

We compare the proposed NPGF method with the IFPGF method based on three typical test color images, Lena, Flower, and Statue, as shown in Figure 2. For fair comparison with the IFPGF method, the proposed NPGF method is only used to determine the status of the undetermined pixels generated by the first stage in the IFPGF method. The similarity threshold $t_s = 15$ was used for images Lena and Flower and $t_s = 20$ for image Statue in the proposed NPGF method. The corresponding similarity thresholds 30 and 40 were used in the IFPGF method. As for the threshold t_z on the peer group size, number 12 was used for both NPGF and IFPGF methods. All comparisons are performed and implemented with Borland C++ Builder 6.0 and run on a standard PC with AMD Athlon 64x2 4800+ CPU (2.5GHz) and 1.87GB of RAM.

We compare the accuracy in detecting the corrupted pixels based on five conventional accuracy measures: (a) recall, (b) specificity, (c) precision, (d) accuracy, and (e) F-measure. Recall is the proportion of correctly deduced corrupted pixels within the true corrupted pixels. Specificity is the proportion of correctly deduced non-corrupted pixels within the true non-corrupted pixels. Precision is the proportion of true corrupted pixels within the deduced corrupted pixels. Accuracy is the weighted average of recall and specificity with weights proportional to the numbers of true corrupted and non-corrupted pixels. The F-measure is the harmonic mean of recall and precision. The F-measure is high only when both recall and precision are high since the harmonic mean of two

proportions tends to be low if one of the two proportions is low. Let TP and TN denote respectively the number of pixels that are correctly deduced as corrupted and non-corrupted pixels. And denote respectively by FP and FN the number of pixels that are erroneously deduced as corrupted and non-corrupted pixels. Then we have

$$\begin{aligned} \text{recall} &= \frac{TP}{TP+FN}, \\ \text{specificity} &= \frac{TN}{TN+FP}, \\ \text{precision} &= \frac{TP}{TP+FP}, \\ \text{accuracy} &= \frac{TP+TN}{TP+FP+TN+FN}, \\ \text{F-measure} &= \frac{2}{\frac{1}{\text{recall}}+\frac{1}{\text{precision}}}. \end{aligned}$$

Empirical results are listed in Tables 1 through 4, for different percentages of corruption, respectively. Based on the empirical results, the following general conclusions are obvious:

1. The proposed NPGF has higher detection accuracy than the IFPGF method on all the accuracy measures considered.

2. The proposed NPGF has better but not significantly higher specificity than the IFPGF method for all the cases considered.

3. The proposed NPGF has significantly higher recall and precision than the IFPGF method for all the cases considered, leading to a significantly higher F-measure.

4. The proposed NPGF has significantly higher recall but not significantly higher accuracy since the images only encompass a small proportion of corrupted pixels.

Table 1: Comparisons of detection accuracy with corruption percentage 5%.

NPGF (IFPGF)	recall	specificity	precision	accuracy	F-measure
Lena	94.05% (93.48%)	99.85% (99.14%)	96.29% (79.28%)	99.56% (98.86%)	95.16% (85.80%)
Flower	96.29% (81.57%)	99.78% (99.47%)	94.80% (87.26%)	99.61% (98.58%)	95.54% (84.32%)
Statue	95.06% (77.94%)	99.71% (99.25%)	93.09% (81.96%)	99.48% (98.18%)	94.06% (79.90%)

Table 2: Comparisons of detection accuracy with corruption percentage 10%.

NPGF (IFPGF)	recall	specificity	precision	accuracy	F-measure
Lena	96.92% (96.71%)	99.67% (98.04%)	96.66% (79.71%)	99.40% (97.91%)	96.79% (87.39%)
Flower	98.30% (90.38%)	99.58% (98.84%)	95.56% (87.73%)	99.45% (97.99%)	96.91% (89.04%)
Statue	97.55% (86.32%)	99.34% (98.35%)	93.20% (82.99%)	99.16% (97.15%)	95.33% (84.62%)

Table 3: Comparisons of detection accuracy with corruption percentage 20%.

NPGF (IFPGF)	recall	specificity	precision	accuracy	F-measure
Lena	98.24% (98.15%)	99.21% (95.87%)	96.62% (81.63%)	99.02% (96.33%)	97.42% (89.13%)
Flower	99.11% (94.12%)	98.90% (97.38%)	95.16% (88.43%)	98.94% (96.73%)	97.09% (91.19%)
Statue	98.55% (93.01%)	98.64% (96.64%)	93.95% (84.86%)	98.62% (95.91%)	96.20% (88.75%)

Table 4: Comparisons of detection accuracy with corruption percentage 30%.

NPGF (IFPGF)	recall	specificity	precision	accuracy	F-measure
Lena	98.67% (98.30%)	98.84% (93.76%)	97.11% (83.62%)	98.79% (95.12%)	97.88% (90.37%)
Flower	99.02% (94.59%)	98.39% (96.25%)	95.90% (90.43%)	98.58% (95.75%)	97.44% (92.46%)
Statue	98.85% (94.38%)	97.61% (94.53%)	94.00% (86.04%)	97.98% (94.49%)	96.36% (90.02%)

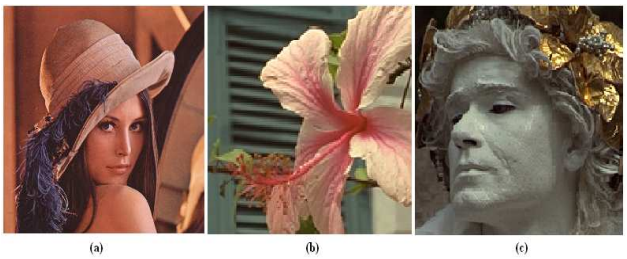


Figure 2: Three typical test images, (a) Lena, (b) Flower, and (c) Statue.

4. CONCLUSION

We propose a novel peer group-based filtering method to reduce the impulse noises for the color images. The key contributions of this work are two-fold. First, we propose that the impulse noise detection is performed in the CIELab, instead of the RGB, color space to enhance noise detectability. Secondly, the proposed method employs two different-sized windows in determining the status for each pixel, alleviating the problems in correcting non-corrupted pixels in the neighborhood of edges in the textured regions. Experimental results demonstrate that the proposed NPGF method achieves better detection ability in terms of all the accuracy measures considered when compared to the IFPGF method.

5. ACKNOWLEDGEMENTS

Kuo–Liang Chung and Wei–Ning Yang are supported by the National Science Council of R.O.C. under Contract NSC98-2923-E-011-001-MY3 and NSC100-2218-E-011-006, respectively.

6. REFERENCES

[1] J. Astola, P. Haavisto, and Y. Neuvo, "Vector Median Filters," *Proceedings of IEEE,* vol. 78, no. 4, pp. 678–689, 1990.

[2] K. N. Plataniotis and A. N. Venetsanopoulos, Color Image Processing and Applications, Springer-Verlag, Berlin, 2000.

[3] R. Lukac, B. Smolka, K. Martin, K. N. Plataniotis, and A. N. Venetsanopoulos, "Vector filtering for color imaging," *IEEE Signal Processing Magazine, Special Issue on Color Image Processing,* vol. 22, no. 1, pp. 74–86, 2005.

[4] S. Schulte, V. De Witte, M. Nachtegael, D. Van der Weken, and E. E. Kerre, "Fuzzy random impulse noise reduction method," *Fuzzy Sets and Systems,* vol. 158, no. 3, pp. 270–283, 2007.

[5] S. Schulte, V. De Witte, M. Nachtegael, D. Van der Weken, and E. E. Kerre, "Histogram-based fuzzy colour filter for image restoration," *Image and Vision Computing,* vol. 25, no. 9, pp. 1377–1390, 2007.

[6] S. Schulte, S. Morillas, V. Gregori, and E. E. Kerre, "A new fuzzy color correlated impulsive noise reduction method," *IEEE Transactions on Image Processing,* vol. 16, no. 10, pp. 2565–2575, 2007.

[7] J. G. Camarena, V. Gregori, S. Morillas, and A. Sapena, "Two–step fuzzy logic-based method for impulse noise detection in colour images," *Pattern Recognition Letters,* vol. 31, no. 13, pp. 1842–1849, 2010.

[8] L. Jin and D. Li, "An efficient color impulse detector and its application to color images," *IEEE Signal Processing Letters,* vol. 14, no. 6, pp. 397–400, 2007.

[9] K. S. Srinivasan and D. Ebenezer, "A new fast and efficient decision-based algorithm for removal of high-density impulse noises," *IEEE Signal Processing Letters,* vol. 14, no. 3, pp. 189–192, 2007.

[10] Y. Dong and S. Xu, "A new directional weighted median filter for removal of random-valued impulse noise," *IEEE Signal Processing Letters,* vol. 14, no. 3, pp. 193–196, 2007.

[11] L. Jin and D. Li, "A switching vector median filter based on the CIELAB color space for color image restoration," *Signal Processing,* vol. 87, no. 6, pp. 1345–1354, 2007.

[12] C. E. Moxey, S. T. Sangwine, and T. A. Ell, "Hypercomplex correlation techniques for vector images," *IEEE Transactions on Signal Process.,* vol. 51, no. 7, pp. 1941–1953, 2003.

[13] Y. Neuvo and W. Ku, "Analysis and digital realization of a pseudorandom gaussian and impulsive noise source," *IEEE Transactions on Communications,* vol. 23, no. 9, pp. 849–858, 1975.

[14] J. Y. F. Ho, "Peer region determination based impulsive noise detection," *Proceedings of International Conference on Acoustics, Speech and Signal Processing ICASSP,* vol. 03, no. 3, pp. 713–716, 2003.

[15] B. Smolka and A. Chydzinski, "Fast detection and impulsive noise removal in color images," *Real-Time Imaging,* vol. 11, no. 5–6, pp. 389–402, 2005.

[16] S. Morillas, V. Gregori, and G. Peris-Fajarnes, "Isolating impulsive noise pixels in color images by peer group techniques," *Computer Vision and Image Understanding,* vol. 110, no. 1, pp. 102–116, 2008.

[17] J. G. Camarena, V. Gregori, S. Morillas, and A. Sapena, "Fast detection and removal of impulsive noise using peer groups and fuzzy metrics," *Journal of Visual Communication and Image Representation,* vol. 19, no. 1, pp. 20–29, 2008.

[18] J. G. Camarena, V. Gregori, S. Morillas, and A. Sapena, "Some improvements for image filtering using peer group techniques," *Image and Vision Computing,* vol. 28, no. 1, pp. 188–201, 2010.

[19] S. Morillas, V. Gregori, G. Peris-Fajarnes, and P. Latorre, "A fast impulsive noise color image filter using fuzzy metrics." *Real-Time Imaging,* vol. 11, no. 5–6, pp. 417–428, 2005.

[20] R. W. G. Hunt, Measuring Colour, 2nd ed., Ellis Horwood, Chichester, UK, 1995.

ABOUT THE AUTHOR

Yu–Ren Lai is a Ph.D. student at National Taiwan University of Science and Technology, Department of Computer Science and Information Engineering. His contact email is D9715011@mail.ntust.edu.tw.

Kuo–Liang Chung is a chair professor at National Taiwan University of Science and Technology, Department of Computer Science and Information Engineering. His contact email is k.l.chung@mail.ntust.edu.tw.

Wei–Ning Yang is an associate professor at National Taiwan University of Science and Technology, Department of Information Management. His contact email is yang@cs.ntust.edu.tw.

Chyou–Hwa Chen is a professor at National Taiwan University of Science and Technology, Department of Computer Science and Information Engineering. His contact email is similar@mail.ntust.edu.tw.

Le–Chung Lin is a master student at National Taiwan University of Science and Technology, Department of Computer Science and Information Engineering. His contact email is M9915011@mail.ntust.edu.tw.

Image enhancement quality metrics

Andrey Nasonov, Andrey Krylov*
Laboratory of Mathematics Methods of Image Processing
Department of Computational Mathematics and Cybernetics
Moscow State University, Moscow, Russia
{nasonov, kryl}@cs.msu.ru

Abstract

The paper presents a new adaptive full reference metrics for the quality measurement of image enhancement algorithms. The idea of the proposed metrics is to find areas related to typical artifacts of image enhancement algorithms. Two types of artifacts are considered: blur and ringing effect. The concept of basic edges is used to find areas of these artifacts which are invariant to image corruption and image enhancement methods. The metrics are illustrated with an application to image resampling and image deblurring.

Keywords: *Image metrics, image enhancement, blur artifact, ringing artifact.*

1. INTRODUCTION

Restoration of high-frequency information of an image is a common problem in image processing. High-frequency information is corrupted of either lost during various image corruption and degradation procedures like downsampling or blurring. It is not possible to completely reconstruct lost high-frequency information, therefore artifacts appears in restored images. Typical artifacts of image enhancement algorithms caused by loss of the high frequency information are blur and ringing effect near sharp edges.

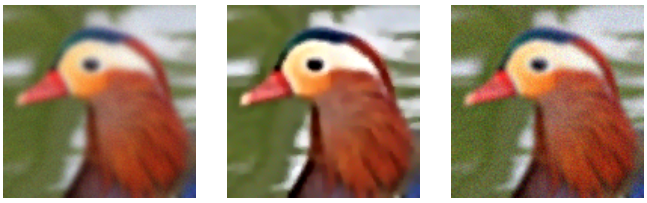
Development of image metrics is important for the objective analysis of image resampling, deringing, deblurring, denoising and other image enhancement algorithms.

Image metrics perform comparison of the ground truth image and the restored image. Since the ground truth image is unavailable in most cases, the simulation approach is used. In this approach, artifact free images are corrupted to simulate the effect which is aimed to be suppressed by the being evaluated image enhancement algorithm. Then the corrupted images are restored using the given algorithm and compared to the corresponding reference images using image metrics.

There exist large variety of image metrics ranging from simple but fast approaches like MSE, PSNR to more complicated metrics based on modeling the human visual system [1]. Most of image metrics can provide the estimation of perceptual image quality but they cannot be used to develop effective image enhancement algorithms because they do not focus on typical artifacts caused by the corruption of high-frequency information. Two image enhancement algorithms can give the same metrics values but the results can be very different if the first algorithm processes edges well and corrupts non-edge area while the second one corrupts only edges. Such an example for image deblurring is shown in Fig.1.

There also exist no-reference quality estimation algorithms that measure specific artifacts like blur and ringing for certain image restoration algorithms like image compression [2, 3, 4] but they are not applicable to the general case.

*The work was supported by federal target program "Research and Development in Priority Fields of S&T Complex of Russia in 2007–2013".



Blurred image — Results of image deblurring

Figure 1: Deblurring of the noisy blurred image by the unsharp mask with two different parameters. PSNR values are the same, but the edges are sharper in the left result image while the non-edge area is better in the right image.

In this paper, we develop metrics for image enhancement algorithms. The proposed metrics are focused on finding the areas related to the considered typical image enhancement artifacts: edge blur and ringing effect. According to the parameters of image corruption and image enhancement method, it is possible to find the areas related to these artifacts and calculate image quality metrics in these areas separately. This information can be useful to help find the most problem areas of the given image enhancement algorithm.

An algorithm to find the area related to ringing effect is proposed in [5], but this algorithm has limitations and cannot be applied for most of image enhancement algorithms. Our proposed method is based on the concept of basic edges — sharp edges which are distant from other edges thus surviving after image corruption. The perceptual metrics for these areas are suggested.

The proposed metrics estimate the quality of different image enhancement methods by analyzing the image quality in the areas of blur and ringing effect. We use the simulation approach so image degradation type and its parameters are considered to be known.

In section 2, we analyze blur and ringing effect for image enhancement of low-resolution images, blurred images and images with ringing effect. In section 3, we find the edges suitable for image quality estimation. In section 4, we introduce our metrics to estimate the quality of image enhancement methods. Applications of the proposed metrics to image resampling and image deblurring are shown in section 5.

2. ARTIFACT ANALYSIS

Since both blur and ringing effect are the results of loss of high frequency information, these effects should be considered together. If all frequencies above $\frac{1}{2p}$ Hz are truncated in Fourier transform, ringing oscillations appear and edges are blurred. The length of single ringing oscillation and edge width are equal to p pixels. The example of high frequency truncation is shown in Fig. 2. Although the number of ringing oscillations is unlimited for the high frequency cut off, usually no more than 1-2 oscillations are noticeable.

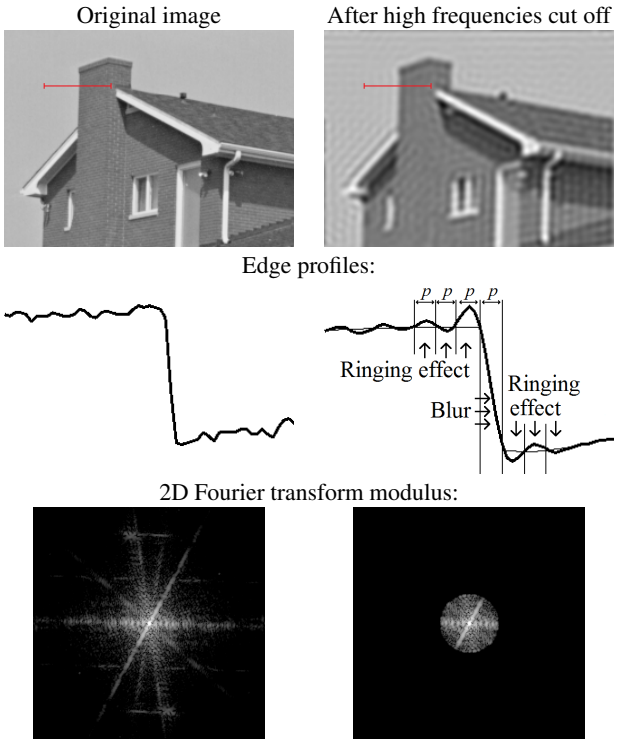


Figure 2: Appearing of blur and ringing effect after high frequency information cut off for $p = 4$.

In practice, the high frequency information is usually corrupted but not completely absent, and the cut off frequency cannot be obtained directly from Fourier transform. In this case additional investigations are required to estimate blur and ringing effect parameter. This parameter can be predicted from image degradation type a priori.

Low-resolution images are constructed using downsampling procedure which includes low-pass antialiasing filtering followed by the decimation procedure. During the decimation with scale factor s, the frequencies greater than $\frac{1}{2s}$ are discarded. The cut off is not ideal because of two-dimensionality of the image. For any linear image resampling method producing blur and ringing effect, its parameter p depends only on scale factor s and $p = s$. For non-linear image resampling methods we use $p = s$ too.

In image deringing the parameter p is already known from the definition of the problem.

Blurred images are the results of low-pass filtering followed by adding noise. We consider Gaussian blur with known radius σ and a noise with Gaussian distribution and standard deviation equals to d. There is no frequency cut off, and parameter p depends on image deblurring method. For unsharp mask, we use $p = k\sigma$, where $2.5 \le k \le 3$.

In the appendix, these results are confirmed experimentally.

3. BASIC EDGES

Blur and ringing effect appears near sharp edges. But any sharp edge cannot be used for image quality analysis. Some edges can disappear or can be displaced after image corruption. If these edges are used for blur and ringing analysis, the results will be incorrect.

There are two effects observed in images with corrupted high frequency information:

1. Masking effect. If an edge with low gradient value is located near an edge with high gradient value, it will disappear after image blurring.

2. Edge displacement. If two edges with the same or close gradient values are located near each other, they will be displaced after image blurring.

To find the edges which do not suffer from masking effect and edge displacement during image corruption, we put the following conditions:

1. An edge point is not masked by nearby edges

$$g_{i_0,j_0} > \max_{i,j} \phi((i-i_0)^2 + (j-j_0)^2), \qquad (1)$$

where $g_{i,j}$ is the image gradient modulus in pixel (i,j), $\phi(d) = \exp\left(-\frac{d^2}{2p^2}\right)$.

2. The distance from the edge point to the nearest edge is greater than a threshold R. This operation is performed using mathematical morphology [6]. We use $R = 3p$.

3. The gradient modulus $g_{i,j}$ is greater than a threshold g_0. The condition is used to reduce the influence of noise to blur and ringing effect.

We call the edges passed all these conditions as *basic edges* and the edges passed only the first condition as *non-masked edges*.

4. IMAGE QUALITY METRICS

After detection of basic edges, we perform image segmentation. According to the analysis of the profile of the step edge after high-frequency cut-off with parameter p (see Fig. 2), we divide the image into three sets:

1. The set M_1 containing all pixels for which the nearest non-masked edge is a basic edge and the distance to this edge is less or equal than $p/2$. Blur effect is the most likely to appear in this set.

2. The set M_2 containing all pixels for which the nearest non-masked edge is a basic edge and the distance to this edge is less or equal than $2p$ and greater than $p/2$. Ringing effect is the most likely to appear in this set.

3. The set M_3 of all pixels with the distance to the nearest non-masked edge greater than $2p$. This set contains no non-masked edges and corresponds to flat and textures areas in the image.

The example of finding these sets is shown in Fig. 3.

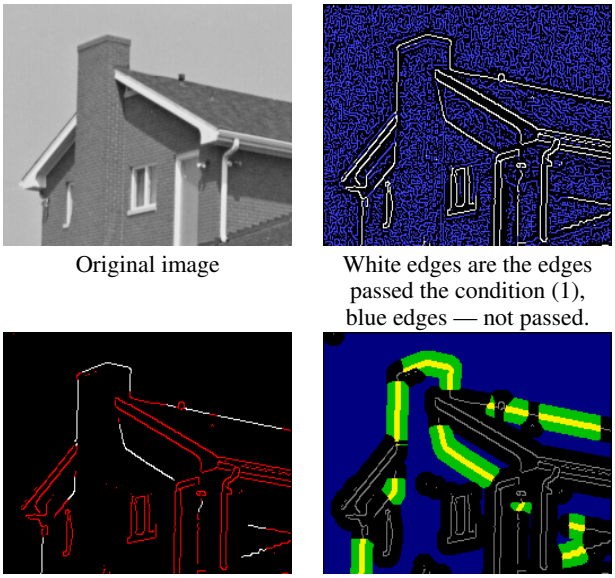
To measure image quality, we calculate metrics values in the sets M_1, M_2 and M_3. Any metrics can be used here. We use $SSIM$ [7] due to its simplicity and good correlation with the perceptual image quality:

$$SSIM(M,u,v) = \frac{(2\mu_u\mu_v + c_1)(2\sigma_{uv} + c_2)}{(\mu_u^2 + \mu_v^2 + c_1)(\sigma_u^2 + \sigma_v^2 + c_2)},$$

where μ_u, μ_v are the averages of u and v respectively, σ_u^2, σ_v^2 — variances, σ_{uv} - the covariance of u and v, L is the dynamic range of the pixel-values (typically this is 255), $k_1 = 0.01$ and $k_2 = 0.03$. The values μ_u, μ_v, σ_u^2, σ_v^2, σ_{uv} are calculated only in the set M.

Now we are ready to introduce the image quality value vector for image u with ground truth image v and given blur-ringing parameter p:

$$\begin{aligned} QV(u,v,p) &= (Q_1,Q_2,Q_3,Q_4) = \\ &= (SSIM(M_1,u,v), SSIM(M_2,u,v), \\ &\qquad SSIM(M_3,u,v), SSIM(u,v)). \end{aligned} \qquad (2)$$



Original image

White edges are the edges passed the condition (1), blue edges — not passed.

White edges are basic edges, red — non-masked non-basic edges

Yellow area — set M_1, green — M_2, blue — M_3.

Figure 3: The result of basic edges detection for $p = 4$.

QV value is a vector containing $SSIM$ values calculated in the sets M_1, M_2, M_3 and in the entire image. Higher values mean better image quality. The sets M_1, M_2, M_3 are constructed for the image v with given parameter p.

5. APPLICATIONS

The proposed metrics are demonstrated by its application to construct combined algorithms for image resampling and image deblurring.

We consider the case when there are two image enhancement algorithms which give relatively the same values of some general purpose metric but produce different artifacts: the first algorithm has strong blur artifact while the second one has strong ringing artifact. This difference is detected by the proposed metrics.

The combined algorithm is constructed as a linear combination of two image enhancement algorithms u, v

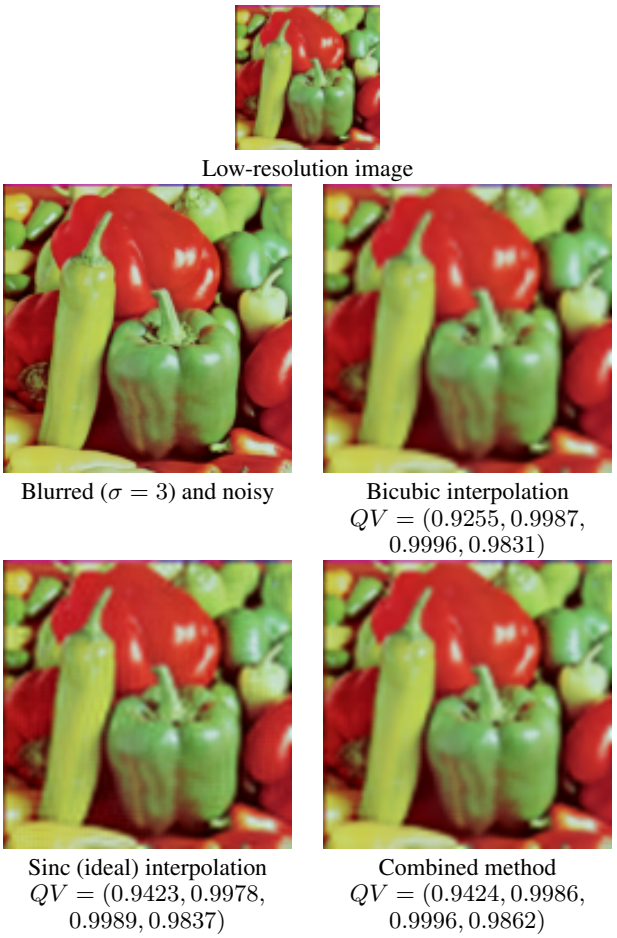
$$w_{i,j} = a(d_{i,j})u_{i,j} + (1 - a(d_{i,j}))v_{i,j},$$

where $a(d)$ is the weight coefficient depending on the distance to the closest non-masked edge $d_{i,j}$ in the blurred image.

Consider u and v such that $QV_1(u) < QV_1(v)$ and $QV_2(u) > QV_2(v)$. In this case we use

$$a(d) = \begin{cases} 0, & d < \frac{p}{2}, \\ \frac{2d-p}{p}, & \frac{p}{2} \le d < p, \\ 1, & d \ge p. \end{cases}$$

The result for combination of bicubic interpolation and sinc interpolation for the problem of image resampling is shown in Fig. 4. To the problem of image deblurring, the result for combination of unsharp mask and regularized total variation (TV) deconvolution in low-frequency domain is shown in Fig. 5. In both cases the combined method shows better $SSIM$ calculated in the whole image than two given methods. Also Q_1, Q_2 and Q_3 values of the combined methods are better than the corresponding best values of the given methods. This make possible to say that the results of combination based on the results of the proposed metrics are better than the results of the methods used for combination.



Low-resolution image

Blurred ($\sigma = 3$) and noisy

Bicubic interpolation $QV = (0.9255, 0.9987, 0.9996, 0.9831)$

Sinc (ideal) interpolation $QV = (0.9423, 0.9978, 0.9989, 0.9837)$

Combined method $QV = (0.9424, 0.9986, 0.9996, 0.9862)$

Figure 4: Application of the proposed metrics to improve the results of image resampling methods.

6. CONCLUSION

New full-reference metrics for quality measurement of image enhancement algorithms were developing. These metrics were approbated on image resampling and image deblurring. It looks promising for combining two different image enhancement algorithms to obtain better result.

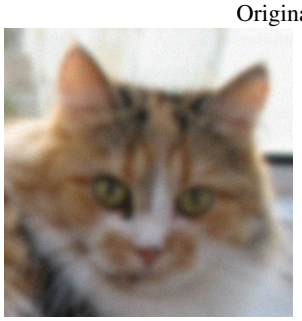
7. REFERENCES

[1] W. S. Lin, *Digital Video Image Quality and Perceptual Coding*, chapter Computational Models for Just-Noticeable Difference, pp. 281–303, CRC Press, 2006.

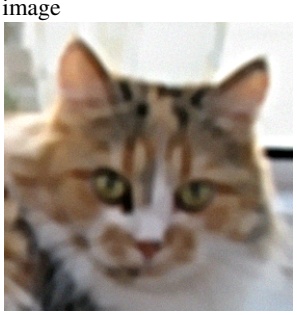
[2] R. Ferzli and L. J. Karam, "Human visual system based no-reference objective image sharpness metric," *ICIP'06*, pp. 2949–2952, 2006.

[3] M. Balasubramanian, S. S. Iyengar, J. Reynaud, and R. W. Beuerman, "A ringing metric to evaluate the quality of images restored using iterative deconvolution algorithms," *Proc. of the 18th Int. Conf. on Systems Engineering (ICSENG'05)*, pp. 483–488, 2005.

Original image

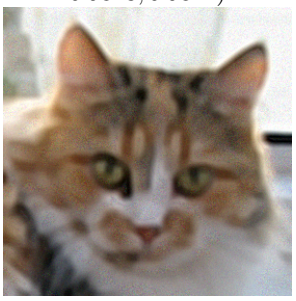


Blurred ($\sigma = 3$) and noisy
$QV = (0.8168, 0.9747,$
$0.9861, 0.9828)$



Regularized deconvolution
$QV = (0.9520, 0.9754,$
$0.9843, 0.9844)$

Unsharp mask with $\alpha = 2$
$QV = (0.8791, 0.9852,$
$0.9788, 0.9833)$



Combined method
$QV = (0.9409, 0.9901,$
$0.9984, 0.9889)$

Figure 5: Application of the proposed metrics to improve the results of deblurring methods.

[4] A. Punchihewa and A. Keerl, "Test pattern based evaluation of ringing and blur in jpeg and jpeg2000 compressed images," *4th International Conference on Signal Processing and Communication Systems (ICSPCS2010)*, pp. 1–7, 2010.

[5] P. Marziliano, F. Dufaux, S. Winkler, and T. Ebrahimi, "Perceptual blur and ringing metrics: Application to jpeg2000," *Signal Processing: Image Communication*, vol. 19, no. 2, pp. 163–172, 2004.

[6] A. V. Nasonov and A. S. Krylov, "Basic edges metrics for image delurring," *Proceedings of 10th Conference on Pattern Recognition and Image Analysis: New Information Technologies*, vol. 1, pp. 243–246, 2010.

[7] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.

[8] A. S. Krylov, A. S. Lukin, and A. V. Nasonov, "Edge-preserving nonlinear iterative image resampling method," *ICIP'09*, pp. 385–388, 2009.

8. APPENDIX

We have performed frequency analysis of different image enhancement algorithms to confirm the preposition from Section 2 that parameter p can be estimated from image degradation method. For every image we calculate the cumulative spectrum function $A(w)$ (CSF):

$$A(w) = \int_0^{2\pi} |\hat{f}(w\cos\theta, w\sin\theta)|^2 d\theta,$$

where $\hat{f}(w_1, w_2)$ is linearly interpolated discrete Fourier transform of the image f.

The analysis consists in calculating the difference between CSFs $A(w)$ for reference images from the set of standard images (baboon, cameraman, house, goldhill, lena, peppers) and CSFs of enhanced images.

Frequency power functions for the different methods of image resampling, deringing and deblurring are shown in Fig. 6. It can be seen that the change of the curve shape happens in the expected point $w = \frac{1}{2p} = \frac{1}{4}$.

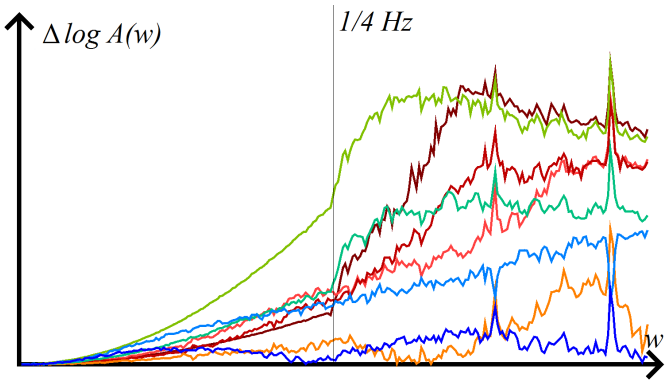


Image resampling ($s = 2$):
— ideal (zero filling);
— Lanczos3;
— bicubic;
— regularization based interpolation with low regularization parameter [8].

Image deringing ($p = 2$):
— Gaussian blur, $\sigma = 1$;
— TV projection.

Image deblurring ($\sigma = 0.7$):
— TV regularization in low-frequency domain;
— unsharp mask, $\alpha = 6$.

Figure 6: Cumulative spectrum functions differences for different image corruption and enhancement methods.

ABOUT THE AUTHORS

Andrey Nasonov is a member of scientific staff at Laboratory of Mathematical Methods of Image Processing, Faculty of Computational Mathematics and Cybernetics, Lomonosov Moscow State University. His contact email is `nasonov@cs.msu.ru`.

Andrey Krylov is a professor, head of Laboratory of Mathematical Methods of Image Processing, Faculty of Computational Mathematics and Cybernetics, Lomonosov Moscow State University. His contact email is `kryl@cs.msu.ru`.

Fast Rank Algorithms Based on Multiscale Histograms

Maria V. Storozhilova, Dmitry V. Yurin
Faculty of Computational Mathematics and Cybernetics,
Lomonosov Moscow State University, Moscow, Russia
mariastorozhilova@gmail.com, yurin@cs.msu.su

Abstract

Rank algorithms for ε_V and KNV neighborhood average calculating are used seldom due to their computational complexity. In this paper fast versions of these algorithms have been proposed. They are based on multiscale histograms. Also the impulse noise suppression method is proposed.

Keywords: *Rank algorithms, median, ε_V neighborhood, KNV neighborhood, multiscale histograms.*

1. INTRODUCTION

The main problem of rank algorithms [5] is their computational complexity, thus only median filtering and maximum/minimum elements search are widely spread in practical applications. Rank algorithms do not blur the edges of objects and fit good for the impulse noise suppression. The approach based on use, maintaining and updating of histograms [2] lets to decrease median filtering complexity from $O(r^2 \log r)$ to $O(r)$, where r is a neighborhood radius. One of the latest works in this area describes the algorithm [3] which lets to decrease histogram updating complexity to $O(1)$ during an image processing. However, the approach [3] has maximum performance only for square neighborhood area and does not provide any optimization of median calculation process.

This work considers algorithms for fast calculation of ε_V and KNV neighborhood average and fast search of arbitrary element in a rank series. For histogram construction and updating both [2] or [3] approaches may be used.

2. MULTISCALE HISTOGRAMS

The highest and the roughest level L_0 of multiscale histogram (see Figure 1) contains the total number of points in current pixel's local neighborhood and the sum of their brightness (the interval from 0 to I_{max}, the maximum intensity of the image). The next lower level (L_1) of histogram contains the same information for 2 sections in the intensity range (0 to $I_{max}/2$ and from the $I_{max}/2+1$ to I_{max}), at the level of L_2 - for 4 sections. In other words every element of higher level includes two corresponding elements of lower level.

The lowest level is the usual histogram where each element corresponds to one intensity value. This level of histogram contains the number of pixels in the neighborhood with corresponding brightness values. The number of this level coincides with the number of bits that represents the image intensity (L_{max}).

Let us consider the histogram element v_0 on the L_3 level and its neighborhood (element $h_{L3}[3]$, see Figure 1). The absolute difference between the remote from v_0 tail of $h_{L3}[3]$ element and v_0 position on L_{max} level will be called the distance from the neighborhood to the adjacent left or right element. For example distance from $h_{L3}[3]$ to $h_{L3}[2]$ equals 13-8 = 5, from $h_{L3}[3]$ to $h_{L3}[4]$ equals 19-13 = 6. Left or right element, which distance from the neighborhood is minimal will be called the closest element.

During the histogram construction process for the current pixel, counter in the appropriate cell on each level of the multiscale histogram is incremented or decremented. Thus, for grayscale images with 256 shades of gray 8 histogram levels should be updated simultaneously, i.e. the complexity of the histogram construction process increases 8 times. However, various mean values might be computed with logarithmic complexity (8) instead of linear (256) in the number of 256 intensity gradations. Acceleration is especially notable in the calculation of complex expressions [5], for example average (ε_V(med(KNV(v_0))).

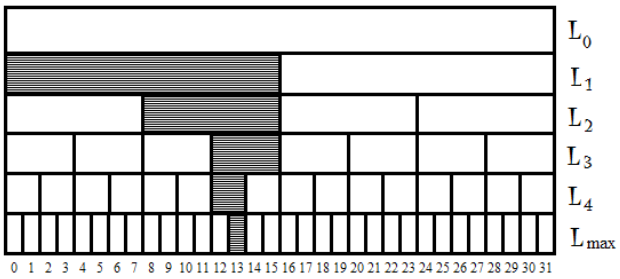


Figure 1: Multiscale histogram.

3. ε_V NEIGHBORHOOD

Let us introduce some useful definitions: the rank series $\{v(r)\}$ is a one-dimensional sequence of N pixels of the neighborhood whose elements are sorted in ascending order with respect to their values: $\{v(r): v(r) \le v(r+1), r=0,1,..N-1\}$. Pixel v_0 rank R is the number of the element in the rank series.

Definition 1. ε_V neighborhood is a subset of pixels $\{v(r)\}$ whose values deviate from the value of the central pixel v_0 at most by predetermined quantities -ε and +ε:

$$\varepsilon_V(v_0) = \{v(r) : v_0 - \varepsilon \le v(r) \le v_0 + \varepsilon\}, r = 0..N-1.$$

ε_V neighborhood average calculation is a simplified analogue of bilateral filter [4]. Bilateral filtering has high computational complexity and fast algorithms give only approximate results. The following method is proposed for mean value calculation in ε_V neighborhood:

Algorithm 1. ε_V neighborhood average.

Input: ε – value of epsilon parameter;

v_0 – intensity of the current pixel;

$v_L := v_0 - \varepsilon$;

$v_R := v_0 + \varepsilon$;

$L_i := L_{max}$ – level number;

$n = 0$ – number of elements in summation;

$s = 0$ – sum of histogram elements;

h – histogram vector;

Output: *average*(v_L; v_R; s; n; L_i);

1: **if** v_R is even **then**

2: $s := s + h_{Li}[v_R] \cdot v_R$;

3: $n := n + h_{Li}[v_R]$;

4: $v_R := v_R - 1$;

5: **if** v_L is odd **then**

6: $s := s + h_{Li}[v_L] \cdot v_L$;

7: $n := n + h_{Li}[v_L]$;

8: $v_L := v_L + 1$;

9: **if** ($v_L < v_R$) **then**

10: *average*($v_L/2$; $v_R/2$; s; L_{i-1})

11: **else**

12: average value is calculated: average := s/n;

First, equidistant segment borders (v_L and v_R) are calculated for the center point on the most detailed level of the histogram. Then, while $v_L < v_R$, algorithm recursively shifts to a higher level. After shift the borders are rounded to the next power of two. For the current level summation involves only the segments that fall under the rounding on each side of the current section.

Thus, number of operations for mean calculation in the local neighborhood of current pixel (for arbitrary central element v_0 and ε value) will not exceed 2*7 additions and comparisons, and one division for an image in 256 shades of gray. I.e. the proposed algorithm has the logarithmic complexity of $O(L_{max})$ rather than linear complexity of $O(2^{Lmax})$. Independence on the choice of v_0 means that the entire class of algorithms for ε_V neighborhood average calculating [5] is implemented and the central element may be the current pixel, or the mean value or the median value, etc.

4. KNV - NEIGHBORHOOD

Definition 2. *KNV*-neighborhood is a subset of a specified number K of pixels $\{v_{n,m}\}$ whose values are nearest to the value of the central pixel v_0:

$$KNV\ (v_0) = \left\{ v(r) : \sum_{r=p}^{p+K-1} |v_0 - v(r)| = \min_p \right\}$$

Considered above algorithm for ε_V neighborhood average has a disadvantage, connected with the problem of right ε choice (it is also a problem for bilateral filtering). The value of ε should depend on image content and must be calculated or set considering a priori image information. Moreover, a single ε value choice for the entire image can be impossible. Therefore algorithms with a priori clear parameter values are of great interest. Particularly – the algorithm for KNV neighborhood average is among them.

From the definition it is clear that, for example, angles of objects ≥ 90º will not be rounded off in the case of two-color image with parameter K = ¼ N; with parameter K = ⅛ N angles less sharp than 45º will not be rounded either. Such a priori clear dependence of the results of algorithm work on its parameter makes it applicable for use, even considering its higher complexity.

Algorithm 2. KNV neighborhood average.

Input: k – value of K parameter;

v_0 – the intensity of the current pixel;

v_L, v_R – left and right borders of summing range;

L_i – the number of the level, where $h_{Li+1}[v_0] > K$;

$n = 3$ – the number of elements, which is necessary to add on a considering histogram level;

$s = 0$ – sum of histogram elements;

h – histogram vector;

Output: *average_KNV*(v_0,v_L,v_R,k,s,n,L)

1: **if** $v_0 - (v_L \cdot 2^{Lmax-L})) > ((v_R + 1) \cdot 2^{Lmax-L} - 1) - v_0$ **then**

2: **if** $h_L[v_R] < k$ **then**

3: $s := s + h_L[v_R] \cdot v_R$;

4: $k := k - h_L[v_R]$;

5: $v_R := v_R + 1$;

6: $n := n - 1$;

7: **else if** $n \neq 2$

8: $s := s - h_L[v_L] \cdot v_L$;

9: $k := k + h_L[v_L]$;

10: $v_L := v_L + 1$;

11: $n = 0$;

12: **else**

13: **if** $h_L[v_L] < k$ **then**

14: $s := s + h_L[v_L] \cdot v_L$;

15: $k := k - h_L[v_L]$;

16: $v_L := v_L - 1$;

17: $n := n - 1$;

18: **else if** $n \neq 2$

19: $s := s - h_L[v_R] \cdot v_R$;

20: $k := k + h_L[v_R]$;

21: $v_R := v_R - 1$;

22: $n = 0$;

23: **if** n ≠ 0

24: *average_KNV* (v_0,v_L,v_R,k,s,n,L);

25: **else**

26: *average_KNV* (v_0,$v_L \cdot 2$,$v_R \cdot 2$,k,s,3,$L+1$);

First the roughest level, containing the number of elements not greater than K, is searched starting with the most detailed level (L_{max}). Initial sum is the element of this level, which contains the central pixel of the neighborhood (v_0). It is proposed to add not more than 3 closest to v_0 elements at each level. If adding one of these elements results that an intermediate sum will contain more than K counts, n=3 or n=1, the opposite border element is subtracted from the sum and step to the lower level is performed. This step allows keeping the symmetry of the neighborhood. At the most detailed level (L_{max}) a single, closest to the center, element is added first. After this step, the neighborhood becomes completely symmetrical. In order to keep exactly K elements in the resulted sum it is necessary to add no more than 2 border elements from each side.

Lemma 1. Proposed algorithm allows constructing symmetrical neighborhood.

Proof: Let L_0 represent the level, at which the first element $h_{L0}[i]$ was added, it contains v_0 – the central pixel of the neighborhood. The L_0 also contains odd number (1) of elements.

Let us propose that, the neighborhood contains odd number M of elements on the level L_0, the central element of the neighborhood contains v_0. Then, at the level of L_0+1 the neighborhood contains 2M corresponding elements (because of a histogram construction, see Figure 1.). Three nearest (see Chapter 2) elements, which are not yet included into the neighborhood, will be added by turn. Let us consider 2 cases.

1) Let addition the first item failed. Then the opposite extreme element is excluded from the neighborhood. In that case the distance between left and right borders of the neighborhood will differ by no more than the size of one element (see Chapter 2) of the level of L_0+1 of histogram.

2) Let addition the first item succeeded. The neighborhood will be more symmetrical after adding the first element on current level, because the element was added from the border where the distance from that border to v_0 is minimal.

The addition of the second element.

1) Let the addition of the second element failed. Then the level handling is over.

2) Let addition the second element succeeded. Then the procedure of addition of the third element is similar to the procedure of addition of the first element.

Thus, on the level of L_0+1 the neighborhood contains the odd number of elements, and the central element of the neighborhood contains v_0. Then the distance between left and right borders of the neighborhood differs by no more than the size of one element of the level of L_0+1 of histogram.

Finally, because of the principle of mathematical induction, all the levels from L_0 to L_{max} contain the odd number of elements, the central element of each level contains v_0 and the dissymmetry of the neighborhood is lesser than the size of the element on each level. Addition of the nearest element on the level of L_{max} guarantees that the neighborhood will be symmetrical.

Lemma 2. The algorithm guarantees that the neighborhood will contain less than K elements for any $L \geq L_0$. Addition of one element on the right and on the left sides guarantees that the neighborhood will contain not less than K elements.

Proof. Let L_0 represent the level, at which the first element $h_{L0}[i]$ was added, it contains v_0 – the central pixel of the neighborhood. The L_0 also contains odd number (1) of elements. Then the neighborhood will contain ≥ K counts with nearest left and right elements.

Let us assume that the histogram contains the odd number of elements on the level of L_0, the central element of the neighborhood contains v_0. Then, at the level of L_0+1 the neighborhood contains 2M corresponding elements (because of a histogram construction Img.1.). Three nearest elements (see Chapter 2), which are not yet included into the neighborhood, will be added by turn. Let us consider 2 cases.

1) Let addition the first item failed. Then the opposite extreme element is excluded from the neighborhood. In that case the number of counts in the neighborhood will be < K. But the neighborhood will contain ≥ K counts with nearest left and right elements.

2) Let addition the first item succeeded. Then the neighborhood contains < K counts.

The addition of the second element.

1) Let the addition of the second element failed. Then the level handling is over. In that case the neighborhood will contain ≥ K counts with nearest left and right elements.

2) Let addition the second element succeeded. Then the procedure of addition of the third element is similar to the procedure of addition of the first element.

Thus, the neighborhood contains ≥ K counts with nearest left and right elements on each level from L_0 to L_{max}. If one more element from each border of the neighborhood is considered together with previous 3 elements on the level of L_{max} (in all 5 elements on the level) then neighborhood will be guaranteed to contain ≥ K counts.

Algorithm 2 may go beyond the borders of the histogram vector while adding or subtracting elements on each level of the histogram. To avoid the growth of difficulty because of permanent index checks, it is proposed to supplement the histogram with zeros on its full size on the right and on the left on each level.

5. SEARCH FOR AN ARBITRARY ELEMENT IN RANK SERIES

The following algorithm is proposed:

Algorithm 3. Search for element with a rank R in a series.

Input: R – element rank in a series;

$L := L_1$ – number of the level, where recursion starts;

h – histogram vector;

$v_0 := 0$ – first element of L_1 level;

Output: *rank*(R,v_0,L)

(For levels L_1 to L_{max}-1)

1: $v_0 := v_0 \cdot 2$;

2: **if** $h_L[v_0] \geq R$ **then**

3: rank(R,v_0,L+1);

4: **else**

5: R := R - $h_L[v_0]$;

6: rank(R,v_0+1,L+1);

(For level L_{max})

1: $v_0 := v_0 \cdot 2$;

2: **if** $h_L[v_0] \geq R$ **then**

3: v_0 – required element;

4: **else**

5: v_0+1 – required element;

Starting from the level L_1 a histogram element with the number of counts less than R is sought. If the first histogram element contains ≥ R elements, the search is continued in the lower-level element that the current element consists of. If the considering histogram element contains less than R elements, their quantity is subtracted from R, and the search is continued in the lower-level element that the next element on current level consists of.

Search of the item with the given rank is performed only using the first element of the lower level, which was obtained in the previous step. Just one check is needed at the level L_{max}. Thus, the search of the element with a given rank for a grayscale image in

256 shades of gray requires not more than 7 subtractions and comparisons.

6. IMPULSE NOISE SUPPRESSION

The following algorithm is proposed. A rank series is created for local neighborhood of central pixel (v_0). If v_0 is contained in K leftmost or rightmost elements in rank series, it is replaced by the mean value. This mean value is calculated over all elements of the rank series except of K leftmost, or rightmost elements, or over N-2K points. Other pixels remain unchanged. Detailed description of the algorithm is not given due to paper size limitations.

7. RESULTS

N is the number of elements in the neighborhood, r is the neighborhood radius, R is the rank for Algorithm 3.

Figure.2 shows source image with added impulse noise and the result of its filtering using the algorithm from section 6.

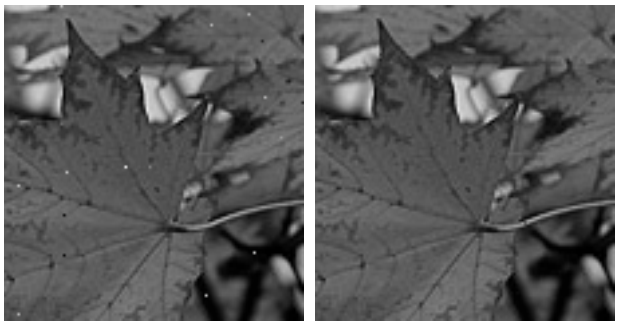


Figure. 2. Source and filtered image

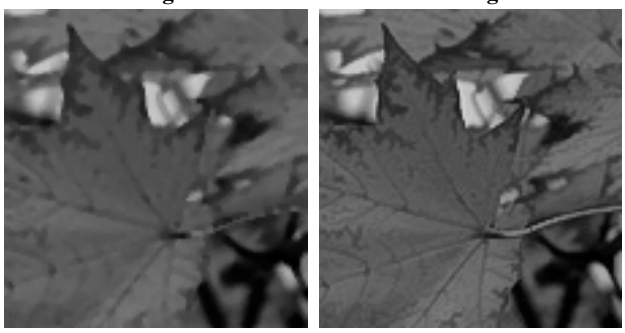


Figure. 3. Image filtered with median (left), with KNV (right)

Figure. 3 demonstrates the smoothing properties of the KNV neighborhood average on denoised image. The smoothed image on the left was calculated with the Algorithm 3 with R = N/2. The right image was calculated with the Algorithm 2, K = 1/4N. Leaf corners became rounder than on the source image after filtering with median. But KNV-based filtering left these corners as sharp as they were on denoised image.

	I_1, r= 12	I_1, r = 37	I_1, r = 62	I_2, r = 37	I_2, r = 62
ε_V	0.031	0.032	0.032	0.110	0.110
KNV	0.093	0.125	0.125	0.244	0.282
rank(N/2)	0.030	0.032	0.032	0.110	0.100
ctmf [3]	0.141	0.172	0.203	0.625	0.656

Figure. 4. Time comparison for images I_1 (size = 375x486) and I_2 (size = 1000x1000).

Figure. 4 demonstrates time statistics of proposed algorithms compared with [3] (times for rank algorithms do not include histogram updating steps).

8. CONCLUSION

This paper proposes methods for fast calculation of ε_V and KNV neighborhood average and fast search of an arbitrary element in a rank series (including minimum, maximum and median) with the use of multiscale histograms. Described algorithms do not suggest any method for histogram construction and maintaining. For this task the classic approach [2] is used. The fast algorithm from [3] is for future work. The algorithms have been implemented in C++ language with wide use of metaprogramming techniques [1] for cycles and recursions unrolling.

9. ACKNOWLEDGMENT

The research is done under support of Federal Target Program "Scientific and Scientific Pedagogical Personnel of Innovative Russia" in 2009-2013 and the Russian Foundation for Basic Research, project no. 09-07-92000-HHC_a.

10. REFERENCES

[1] Alexandrescu A., *Modern C++ Design: Generic Programming and Design Patterns Applied*. Addison-Wesley. ISBN 978-0201704310, February 2001.

[2] Huang T., Yang G., and Tang G., *"A Fast Two-Dimensional Median Filtering Algorithm,"* IEEE Trans. Acoust., Speech, Signal Processing, vol. 27, no. 1, pp. 13–18, 1979.

[3] Perreault S., Hebert P., *"Median Filtering in Constant Time"*, IEEE Transactions on Image Processing, vol. 16, pp. 2389-2394, 2007.

[4] Tomasi C., Manduchi R., "*Bilateral Filtering for Gray and Color Images*", iccv, pp.839, Sixth International Conference on Computer Vision (ICCV'98), 1998

[5] Yaroslavsky L.P., Kim V., "*Rank Algorithms for Picture Processing, Computer Vision",* Graphics and Image Processing, v. 35, 1986, p. 234-258

About the author

Maria V. Storozhilova, is a student at Chair of Mathematical Physics, Faculty of Computational Mathematics and Cybernetics, Lomonosov Moscow State University, Russia. Her contact email is. mariastorozhilova@gmail.com

Dmitry V. Yurin (PhD) is a senior researcher at laboratory of Mathematical Methods of Image Processing, Faculty of Computational Mathematics and Cybernetics, Lomonosov Moscow State University, Russia. His contact email is yurin_d@inbox.ru

Short-term solar flare forecast

Victor Chernyshov, Dmitry Laptev, Dmitry Vetrov
Department of Computational Mathematics and Cybernetics
Moscow State University, Moscow, Russia
webcreator18@gmail.com, laptev.d.a@gmail.com, vetrovd@yandex.ru

Abstract

In this paper a new automated hybrid method for short-term flare forecasting is introduced and suggested for future use.

At the initial stage we created a flare base, and an image base for 1996–2009 years interval.

Further, we derived simple and efficient parametric precedent model, which turned our prediction problem into two-class classification problem, and developed machine learning-based procedures for features extraction from both magnetograms and continuum images.

We develop an experimental protocol to estimate the accuracy of obtained decision rules and report 63% to 82% on balanced data (maximum worst-case), 73% to 90% on real data depending on the choice of precedent model configuration.

Keywords: *solar flare forecast, precedent model, spherical correction, sunspots clustering, magnetogram segmentation, active regions localization.*

1. INTRODUCTION

A *solar flare* is a sudden brightening observed over the Sun surface or the solar limb, which is interpreted as a large energy release. *Sunspots* are temporary phenomena on the photosphere of the Sun that appear in visible spectrum as dark spots compared to surrounding regions. An ***a**ctive **r**egion (AR)* on the Sun is an area with an especially strong magnetic field. Sunspots usually form in active regions.

X-rays emitted by solar flares can affect Earth's ionosphere and disrupt long-range radio communications and disturb operation of navigation systems. The most violent eruptions may affect satellite or cause problems with power grid.

Solar flares research has shown that X-ray flares are closely related to sunspots and active regions [1]. So, a number of flare forecasting methods based on this relationship has been proposed. McIntosh [2] revised sunspot classification and specially dedicated system called Theophrastus was developed in 1987. The method depends on human expert.

In 2009 Qahwaji and Colak presented an automated hybrid computer platform for the short-term prediction of significant solar flares using Solar and Heliospheric Observatory (SOHO)/Michelson Doppler Imager (MDI) images[3]. Proposed method incorporates sunspot grouping (both MDI continuum and magnetogram images are used), McIntosh-based classification, and, afterwards, flare prediction using neural networks.

Yu et al. [4] analyzed the influence of sequences of magnetic-based parameters on the flare level and proposed bayesian solar flare prediction models.

Very recently Falconer et al. introduced their tool for empirical forecasting of major flares from a proxy of active region free magnetic energy [5]. Their method is mainly focused on measuring the proxy of the active regions free magnetic energy, and the empirical relationship is then used to convert the free magnetic energy proxy into an expected event rate.

In our research we included features of both types (magnetic and continuum), derived adequate precedent model in order to use Support Vector Machine (SVM) classification algorithm. As a result, fully-automated testing system was built and good short-term prediction results achieved.

The rest of the paper is organized as follows. The solar data used in this paper are described in Section 2.. A precedent model is proposed in Section 3.. Features extraction is discussed in Section 4.. The implementation of the system and testing results are reported in Section 5..

2. DATA

In this study we used data from the publicly available NOAA solar flare catalogue[1] and images in FITS format from SOHO/MDI in the resolution 1024px x 1024px (can be downloaded via web interface[2]). SOHO/MDI provides both continuum ("white-light", 2–5 per day) and magnetogram observations (in the vicinity of the Ni I 6767.8 Å photospheric absorption line, 6–14 per day) of the Sun.

We created a flare base, consisting of 24658 events, and an image base, including 40573 magnetograms and 14927 continuum images from 1996–2009 years interval. All flares are divided into flare series (using the NOAA active region number): 1397 series, maximum flares in series is equal to 154, minimum — 1, in average — 8. We use notation (i_f, j_f), where i_f — series number, j_f — flare number in the series.

3. PRECEDENT MODEL

Solar flares are classified as A, B, C, M or X according to the peak flux as measured on the GOES spacecraft. Hereinafter we use the following notation: *strong flares* are flares not not weaker than M_f, *weak flares*, respectively, are weaker than M_f. The exact values of parameters we used are given in the end of the section. It is assumed that precedent model is built on the training stage of the method, so, we know when and where a flare occurred.

Stage 0. For simplicity, let's fix flare F, which will correspond to a precedent. It uniquely localizes an active region on every image within flare's prehistory. We consider only images within flare's T_{ph} days prehistory. Hereinafter the words "preceding", "closely located", "nearest" should be treated in the context of time.

Stage 1: base precedent. We choose a magnetogram image and the nearest preceding magnetogram image located not closer than Δ_f days. Denote the first image as "head magnetogram". For the head image we determine the most closely located continuum image ("head continuum"), for which we also find the nearest preceding image located not closer than Δ_f days. Two pairs of images and active region on them we denote as *base precedent*.

Stage 2: positive and negative precedents. *Positive precedent*

[1] http://www.ngdc.noaa.gov/stp/solar/solarflares.html
[2] http://sohowww.nascom.nasa.gov/data/archive/

(class 1) is a base precedent, which meets following requirements:
1. Time from a head magnetogram to the flare does not exceed T_f.
2. Flare strength is not less than M_f.
3. The nearest flare, which meets 1. and 2., corresponds to the head magnetogram.
Negative precedent (class 0) is a base precedent, which is not a positive one.

T_f, M_f, Δ_f, T_{ph} are structural parameters of the precedent model. In our research we suppose $M_f \in M_{set} = \{C5.0, M1.0, M5.0, X1.0\}$, $T_f \in T_{set} = \{1.0, 1.5, 2.0\}$ days, $\Delta_f = 0.25$ days, $T_{ph} = 4.0$ days, so, we investigate different configurations.

4. FEATURES

A general scheme for features extracting is the following:
1. We fix a head magnetogram and build a precedent according the model.
2. Sunspot groups are localized on the nearest to the flare cont-image.
3. If there is a no correspondence between the flare and one of the sunspot groups, we start building another precedent. Otherwise we have the sunspot group and all its characteristics (including bounding box parameters).
4. Steps 2–3 are performed for the second cont-image. After that we are able to extract cont-features from the pair of cont-images (Section 4.1).
5. We apply tracking procedure to the found sunspot group (to be precise, to the center of its bounding box) to locate active regions on both magnetograms and extract from them magn-features (Section 4.2).

The described above scheme is applied for the whole image base for every configuration $(M_f, T_f) \in \{M_{set} \times T_{set}\}$. As a result we have $|M_{set}| \cdot |T_{set}|$ datasets. We use a notation $X_{M_f, T_f} \in \mathbf{R}^{m \times n}$ for features table, $Y_{M_f, T_f} \in \{0, 1\}^{m \times 1}$ for class labels, where m — number of features, n — number of precedents.

4.1 Continuum-based features

4.1.1 Continuum images preprocessing

If we have plane projection of a semisphere, we can not avoid distortions: the same areas on a semisphere in general case are not the same after projection. A point on a semisphere is denoted as $P'(x',\, y',\, z')$, it corresponds to a point $P'_{xy}(x',\, y')$ on the visible image. Supposed that spherical correction maps a point P'_{xy} to a pixel $P(x,\, y)$ on spherically corrected image.

Assuming that proposed mapping keeps ratio 1

$$\frac{\beta}{\frac{\pi}{2}} = \frac{\sqrt{x^2 + y^2}}{R} \tag{1}$$

$$\alpha = \arctan(\frac{y}{x}), \rho = r \sin\beta,\; x' = \rho\cos\alpha,\; y' = \rho\sin\alpha,$$

ρ is the length of vector $O'P'$ projection on plain $O'XY$,

$\beta = \widehat{ZO'P'}$, $O'Z$ is "eye-pointed" axis.

we get coordinates of P':

$$\begin{cases} x' = sgn(x) r \cos(\arctan(\frac{y}{x})) \sin(\frac{\pi}{2} \frac{\sqrt{x^2+y^2}}{R}), \text{if } x \neq 0 \\ y' = sgn(x) r \sin(\arctan(\frac{y}{x})) \sin(\frac{\pi}{2} \frac{\sqrt{x^2+y^2}}{R}), \text{if } x \neq 0 \\ x' = 0,\; y' = 0, \text{if } x = y = 0 \\ z' = \sqrt{r^2 - (x')^2 - (y')^2} \end{cases}$$

To get approximation of the intensity in $P'(x',\, y')$, we use bilinear interpolation.

Further a term "image" is used for images with applied spherical correction procedure.

Before the preprocessing we create an average background cont-image (pixel-by-pixel median through 500 cont-images); it is also called "quiet Sun image".

We also perform Gaussian blurring with $\sigma = 0.7$ and $window\ size = 7$ and cut-off beyond $\alpha_{cut} = 65^\circ$ (heliocentric degrees) to get rid of limb darkening and overblurring effects.

4.1.2 Adaptive binarization

We worked out simple adaptive binarization procedure to localize sunspot groups in the Sun (see figure 1 c): we set knowingly high threshold $T_0 = 0.98$ and start decreasing it with the step $\tau = 0.002$. At every step binarizarion is performed, thus, we know the number of connected components and their sizes. If connected components number doesn't exceed $N_{max\_comp} = 100$, and among them there are no components with area is more than $0.12 * (R * \sin\alpha_{cut})^2$), then we have found appropriate binarization threshold T_{bin}. Exceeding maximum loop iterations number $N_{max\_iter} = 80$ is an alternative loop exit condition.

4.1.3 Umbra and penumbra segmentation

When viewed through a telescope, sunspots have a dark central region known as the *umbra*, surrounded by a somewhat lighter region called the *penumbra*. Umbra and penumbra characteristics can be used for solar flare prediction [2].

Area of the sunspot is denoted as S_F. Quiet Sun intensity value I_q is a non-zero intensity, corresponding to the peak on the histogram of a preprocessed image (we used a partition of $[0, 1]$ in $N_{hist\_bins} = 1000$ equal intervals). Our approach to umbra and penumbra segmentation is based on the method proposed by Zharkov et al. [6], which incorporates umbra (T_u) and penumbra (T_p) thresholds:
1. if $S_F \leq 5$ pixels then assign the thresholds: $T_p = 0.91 * I_q$; $T_u = 0.6 * I_q$
2. if $S_F > 5$ pixels then assign the thresholds: $T_p = 0.93 * I_q$; $T_u = \max(0.55 * I_q, \mathbb{E}P - \Delta P)$, where $\mathbb{E}P$ is the mean intensity value, ΔP is the standard deviation for pixel intensities in the region F.

Our modification uses only T_p threshold. Thus, sunspot pixels, which are not in umbra, are supposed to be in penumbra; due rather flexible adaptive binarization, T_{bin} could be considered as T_p analogue (see figure 1 e).

4.1.4 Sunspots clustering

After localizing sunspots we need to combine them into several sunspot groups. For this purpose we use agglomerative hierarchical clustering procedure with euclidian metrics and Ward linkage (see figure 1 f).

4.1.5 Extracted features

Described above stages of cont-images processing make calculating cont-features for a pair of images possible. Cont-features are calculated using head continuum image, their speed of change — using the both images. A sunspot group is unambiguous defined by the solar flare, corresponding to a precedent.

Extracted cont-features:
1. umbra square of the sunspot group,
2. penumbra square of the sunspot group,
3. speed of change for 1–2 (px/sec).

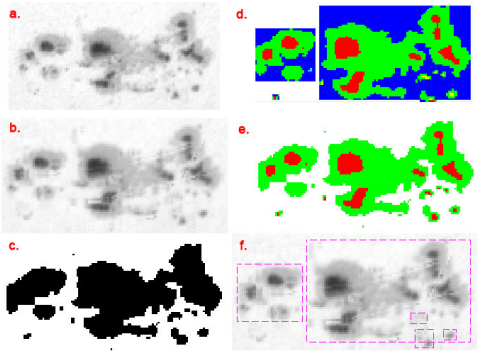


Figure 1: Vicinity of the sunspot group, localized on cont-image 4240.0003, at different stages of features extraction: (a) — initial, (b) — after spherical correction, (c) — the result of adaptive binarization, (d) — segmented umbra and penumbra (using Zharkov et. al method), (e) — segmented umbra and penumbra using our method, (f) — after clustering all neighboring sunspots form a sunspot group.

4.2 Magnetic-based features

A magnetogram image is a representation of the variations in strength of a magnetic field. Black regions correspond to strong positive magnetic field, white regions — to strong negative magnetic field, neutral Sun regions are marked with grey color. We extract magn-features in three stages: performing segmentation of a magnetogram, finding active regions, counting features.

4.2.1 Segmentation

The first step is to find the areas of strong positive magnetic fields, strong negative magnetic fields and neutral areas. Proposed method is based on variational approximation [7] with the use of global constraints [8], which give us a possibility to include some physically-driven conditions in our model (i. e. the equality of positive and negative fluxes within AR).

Let i be the pixel of an image with the value of the magnetic field I_i, N — number of pixels, Z_i — class label for pixel i, $\mathcal{E}$ — neighborhood system.

In these terms the discrete optimization task could be formulated as

$$p(I|Z) \propto \prod_{i=1}^{N} \varphi_i(Z_i) \prod_{j \in \mathcal{E}(i)} \phi(Z_i, Z_j) \to \max_Z \qquad (2)$$

Where $\phi(Z_i, Z_j) = e^{C_{pair}[Z_i \neq Z_j]}$ — pairwise term, $\varphi_i(Z_i)$ — unary term: $\varphi_i(1) = e^{-C_1\sqrt{|2000-I_i|}}$, $\varphi_i(2) = e^{-C_1\sqrt{|2000+I_i|}}$, $\varphi_i(3) = e^{-C_2|I_i|}$.

This task is solved with factorized approximation

$$p(Z|I) \approx q(Z) = \prod_{i=1}^{N} q_i(Z_i) \qquad (3)$$

Minimizing KL-divergence between $q(Z)$ and $p(Z|I)$ the following equation could be obtained:

$$q_j^*(Z_j) = \frac{\exp(\mathrm{E}_{i \neq j} \log p(Z)}{\int \exp(\mathrm{E}_{i \neq j} \log p(Z) dZ_j}$$

Using the exact form of $p(Z)$ we obtained iterative process

$$q_i^{new}(Z_i) = \frac{1}{C} \exp\left(\log(\varphi_i(Z_i)) - C_{pair} \sum_{t \in \mathcal{E}(i)} \sum_{j \neq i} q_j^{old}(Z_j)\right)$$

4.2.2 Active region search

For localizing an active region on the Sun we use the method introduced in [9], which is based on the branch and bounds approach to maximizing the functional defined on a rectangle. The initial bounding rectangle on a magn-image is obtained through tracking and the following enlarging it in 2.5 times.

Let R be the rectangle, $A_i = q_i(1) + q_i(2)$ and $B_i = q_i(3)$. The active region could be found by maximizing the following functional:

$$F(R) = \alpha \sum_{i \in R} A_i - \sum_{i \in R} B_i + \beta\sqrt{Area(R)} \sum_{i \in border\ of\ R} B_i.$$

The global optimum could be found using the function $\hat{F}(\mathbf{R})$ which is the top border of $F(R)$ and equal $F(R)$if $\mathbf{R} = \{R\}$. The optimization procedure is then performed as follows: at the first step $\mathbf{R}$ consists of all the possible rectangles R, the sets $\mathbf{R}$ are placed in the priority queue in a decreasing order of $\hat{F}(\mathbf{R})$, in every step the first element of the queue is divided into two and returned back to the queue. If the first element of the queue consists of just one rectangle, that is the answer.

In our case the function $\hat{F}(\mathbf{R})$ could be chosen as follows:

$$\hat{F}(\mathbf{R}) = \alpha \sum_{i \in R_{small}} A_i - \sum_{i \in R_{big}} B_i + \beta\sqrt{Area(R_{big})}$$
$$\Big(\max_{R \in \mathbf{R}} \sum_{i \in left\ border\ of\ R} B_i + \max_{R \in \mathbf{R}} \sum_{i \in top\ border\ of\ R} B_i +$$
$$\max_{R \in \mathbf{R}} \sum_{i \in right\ border\ of\ R} B_i + \max_{R \in \mathbf{R}} \sum_{i \in bottom\ border\ of\ R} B_i\Big) \qquad (4)$$

Finally, we get a refined bounding box, specifying active region on a magn-image more precisely. The result of the procedure with $\alpha = 4, \beta = 0.05$ is presented in figure figure 2 a,

4.2.3 Neutral line

The neutral line concept is supposed to give several informative features. We define it as follows: *neutral line* is a line separating the regions of strong magnetic fields obtained from segmentation with different polarities.

Neutral line extraction is applied to a refined active region. We use both direct implementation of the definition and the robust algorithm that cuts off small regions and small fragments of the line. The resulting simple neutral line is presented in white in figure 2 b, the robust line is given in black in the same figure.

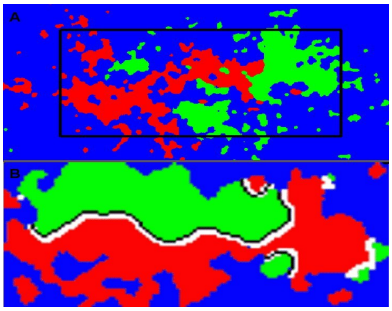


Figure 2: (a) — the localized active region, (b) — standard (white) and robust (black) versions of the neutral line.

4.2.4 Extracted features

Magn-features are calculated using head magnetogram image, their speed of change — using the both images. An active region is unambiguously defined by the solar flare, corresponding to a precedent. An AR is specified by it's bounding box.

Extracted magn-features:
1. sum of magnetogram values in the area corresponding to a positive/negative segment inside bounding box (positive/negative flux),
2. maximum absolute value of the magnetic flux in the bounding box (maximum absolute flux),
3. area of the dilated line with the dilation coefficient equal to 5 for simple and robust algorithms (simple/robust line area),
4. sum of the positive/negative magnetic flux inside the area of the dilated line with the dilation coefficient equal to 5 for simple and robust algorithms (simple/robust line positive/negative flux),
5. speed of change for all of the features above.

5. TESTING SYSTEM AND RESULTS

For testing purposes we use libSVM implementation of Support Vector Machine two-class classifier. After several numerical experiments with different kernel and structural parameters we have found that the best result of SVM with RBF-kernel is worse than with linear one. So, further only linear SVM with the only structural parameter C is used. We implemented an exhaustive search of the optimal parameter value over the set $C_{set} = \{2.25^i \,|\, i \in \{-4, -3, \ldots, 4\}\}$.

Several precedents can have correspondence to one flare. So, we can define max-based decision rule: we obtain class labels for all test precedents in usual way, then we perform postprocessing organized as follows. For every precedent we calculate maximum among class labels of the precedents, which correspond to the same flare as our precedent. Thus, we decrease probability of missing positive precedents.

Every calculated dataset $(X_{M_f,T_f}, Y_{M_f,T_f})$ is divided into three non-intersecting approximately equal parts: $train$ (to learn our classifier), $test$ (to find the most optimal configurations of the structural parameters), and $TEST$ (to get testing results). The union of $train$ and $test$ is denoted as $TRAIN$.

Although negative precedents are much more numerous than positive (strong flare is a rare event), we should learn and optimize our classifier on balanced datasets $train$ and $test$.

Except this, to get rid of the similarity between the precedents in one series of flares, we decided to put the precedents, corresponding to flares belonging to the same flare series, either all in train or all in test.

Although a reasonable amount of features is calculated, we don't exactly know, which of them are really informative for one or the other precedent model configuration. Therefore, we implemented full search over all subsets $\{F \,|\, F \subseteq \mathbf{2}^{F_{set}}, F \neq \emptyset\}$. F_{set} includes the following features: umbra square of the sunspot group and it's speed of change, negative flux and it's speed of change, maximum absolute flux and it's speed of change, robust line negative flux and it's speed of change, speed of change of the simple line negative flux.

To obtain the final results table 1 we run our testing procedure for every configuration $(M_f, T_f) \in \{M_{set} \times T_{set}\}$; partitioning into $TRAIN$ and $TEST$ was fixed; the results were averaged over 5 different partitions of $TRAIN$ into $train$ and $test$.

M_f \ T_f	1.00	1.50	2.00
C5.0	36.0 **26.9**	36.6 **29.3**	37.0 **30.1**
M1.0	31.7 **22.3**	36.2 **22.2**	36.5 **22.2**
M5.0	26.7 **12.3**	29.3 **13.7**	25.7 **13.7**
X1.0	19.9 **9.4**	17.3 **10.2**	19.0 **11.6**

Table 1: Average error rates (%) on $TEST$ dataset: for balanced (first number) and unbalanced (second number) it's versions.

Since real data are unbalanced (strong flares are much less than weak ones), second number (bold) in every cell can be considered as unbiased error rate of our method; a cell is chosen according our forecast needs. Finally, we have 63% to 82% on balanced data (maximum worst-case), 73% to 90% on real data.

In the nearest future we intend to incorporate in our method some additional physically-driven features, include SHO/HMI images support, and build fully-automated web-compliant prediction system.

6. REFERENCES

[1] Zhongxian Shi and Jingxiu Wang, "Delta-sunspots and x-class flares," *Solar Physics*, vol. 149, pp. 105–118, jan 1994.

[2] Patrick S. McIntosh, "The classification of sunspot groups," *Solar Physics*, vol. 125, pp. 251–267, 1990.

[3] T. Colak and R. Qahwaji, "Automated Solar Activity Prediction: A hybrid computer platform using machine learning and solar imaging for automated prediction of solar flares," *Space Weather*, vol. 7, no. 6, jun 2009.

[4] Daren Yu, Xin Huang, Huaning Wang, Yanmei Cui, Qinghua Hu, and Rui Zhou, "Short-term solar flare level prediction using a bayesian network approach," *The Astrophysical Journal*, vol. 710, no. 1, pp. 869, 2010.

[5] David Falconer, Abdulnasser Barghouty, and et. al, "A tool for empirical forecasting of major flares, coronal mass ejections, and solar particle events from a proxy of active-region free magnetic energy," *Space Weather*, vol. 9, no. 4, apr 2011.

[6] S. Zharkov, V. Zharkova, and et. al, "Automated recognition of sunspots on the soho/mdi white light solar images," in *Knowledge-Based Intelligent Information and Engineering Systems*, vol. 3215, pp. 446–452. 2004.

[7] Michael I. Jordan, Zoubin Ghahramani, and et. al, "An introduction to variational methods for graphical models," *Mach. Learn.*, vol. 37, pp. 183–233, November 1999.

[8] D. Kropotov, D. Laptev, and et. al, "Variational segmentation algorithms with label frequency constraints," *Pattern Recognit. Image Anal.*, vol. 20, pp. 324–334, September 2010.

[9] C. H. Lampert, M. B. Blaschko, and T. Hofmann, "Beyond sliding windows: Object localization by efficient subwindow search," in *Computer Vision and Pattern Recognition, 2008. IEEE Conference*, 2008, pp. 1–8.

7. ACKNOWLEDGEMENTS

This work is supported by Microsoft Research grant. Authors would like to thank Anatoly Petrukovich, Victor Lempitsky and Pushmeet Kohli for valuable discussions throughout the research.

Image Segmentation Techniques: Comparison and Improvement

Charbel Fares[1], Alain Bou Malham[2]
[1]Holy Spirit University of Kaslik, Faculty of Sciences, Lebanon.
[2]Digipen Middle East
charbelfares@usek.edu.lb, aboumalh@digipen.edu

Abstract

Region Growing is an image segmentation approach particularly used in Artificial Intelligence in which neighboring pixels are examined and added to a region class if no edges are detected. This process is iterated for each boundary pixel in the region. If adjacent regions are found, a region-merging algorithm is used, in which weak edges are dissolved and strong edges are left intact. Region Growing offers several advantages over conventional segmentation techniques. The algorithm is also very stable with respect to noise. Regions will never contain too much of the background, as long as the parameters are defined correctly. Other techniques that produce connected edges, like boundary tracking, are very unstable. We can take advantage of several image properties, such as low gradient or gray level intensity value, at once. There are, however, several disadvantages to region growing. It is very expensive computationally. It takes both serious computing power (processing power and memory usage) and a decent amount of time to implement the algorithms efficiently. In this paper, we presented a list of segmentation techniques and we proposed a new segmentation algorithm. The results of an objective evaluation of these segmentation techniques are shown in a comparative study. Moreover, we proposed a hybrid variant that combines two techniques. For each of these two techniques, we will examine three characteristics: Correctness, Stability with respect to parameter choice, and Stability with respect to image choice.

Keywords: *Image Segmentation, Hybrid Approach, Region Growing, Region Merging, Mean Graph.*

1. INTRODUCTION

In computer vision, segmentation refers to the process of partitioning a digital image into multiple regions. The goal of segmentation is to simplify and/or change the representation of an image into something that is more meaningful and easier to analyze. Image segmentation is typically used to locate objects and boundaries (lines, curves, etc.) in images. The result of image segmentation is a set of regions that collectively cover the entire image, or a set of contours extracted from the image. Each of the pixels in a region is similar with respect to some characteristic or computed property, such as color, intensity, or texture.

Segmented images are now used routinely in a multitude of different applications, such as, diagnosis, treatment planning, in the robotics, localization of pathology, geology, study of anatomical structure, meteorology, computer-integrated surgery, among others. However, image segmentation remains a difficult task due to both the variability of object shapes and the variation in image quality. In spite of the most complex algorithms developed nowadays, segmentation continues being very dependent on the application. With the aim of obtaining segmentation methods more exact and more effective, several techniques have been proposed in the literature. Unfortunately, segmentation is a complex problem with no exact solution. Noise and other image artifacts can cause incorrect regions or boundary discontinuities in segmented objects.

The main challenge in object detection is the amount of variation in visual appearance. For example, cars vary in shape, size, coloring, and in small details such as the headlights, grill, and tires. Visual appearance also depends on the surrounding environment. Light sources will vary in their location with respect to the object, their intensity, and their color. Nearby objects may cast shadows on the object or reflect additional light on the object. The appearance of the object also depends on its pose; that is, its position and orientation with respect to the camera. For example, a human face will look much different when viewed from the side than viewed frontally. An object detector much accommodates all these variations and still distinguishes the object from any other patterns that may occur in the visual world.

The rest of the paper is structured as follows. In section 2 the proposed algorithm is shown. The evaluation and comparison are shown in section 3. Finally, conclusions are sketched in Section 4.

2. PROPOSED ALGORITHM: MEAN GRAPH

Unsupervised image segmentation algorithms have matured to the point that they provide segmentations which agree to a large extent with human intuition. The time has arrived for these segmentations to play a larger role in object recognition. It is clear that unsupervised segmentation can be used to help cue and refine various recognition algorithms. However, one of the stumbling blocks that remain is that it is unknown exactly how well these segmentation algorithms perform from an objective standpoint.

Most presentations of segmentation algorithms contain superficial evaluations which merely display images of the segmentation results and appeal to the reader's intuition for evaluation. There is a consistent lack of numerical results, thus it is difficult to know which segmentation algorithms present useful results and in which situations they do so. Appealing to human intuition is convenient, however if the algorithm is going to be used in an automated system then objective results on large datasets are to be desired.

We present the results of an objective evaluation of two popular segmentation techniques: the mean shift [1] and the graph-based segmentation algorithms [2]. As well, we look at a hybrid variant that combines these algorithms. For each of these algorithms, we examine three characteristics:

1. *Correctness*: the ability to produce segmentations which agree with human intuition. That is segmentations which correctly identify structures in the image at neither too fine nor too coarse level of detail.

2. *Stability with respect to parameter choice*: the ability to produce segmentations of consistent correctness for a range of parameter choices.
3. *Stability with respect to image choice*: the ability to produce segmentations of consistent correctness using the same parameter choice on a wide range of different images.

If a segmentation scheme satisfies these three characteristics, then it will give useful and predictable results which can be reliably incorporated into a larger system.

2.1 Segmentation Algorithms

We have chosen to look at mean-graph segmentation as it is generally effective. The efficient graph-based segmentation algorithm was chosen as an interesting comparison to the mean shift in that its general approach is similar; however it excludes the mean shift filtering step itself, thus partially addressing the question of whether the filtering step is useful. The combination of the two algorithms is shown as an attempt to improve the performance and stability of either one alone. Then we describe each algorithm and further discuss how they differ from one another.

2.1.1 Mean Shift Segmentation

The mean shift segmentation technique is one of many techniques under the heading of ``feature space analysis". The mean shift technique is comprised of two basic steps: a mean shift filtering of the original image data (in feature space), and a subsequent clustering of the filtered data points. Below we will briefly describe each of these steps and then discuss some of the strengths and weaknesses of this method.

- **Filtering:** The filtering step of the mean shift segmentation algorithm consists of analyzing the probability density function underlying the image data in feature space. Consider the feature space consisting of the original image data represented as the (x, y) location of each pixel, plus its color in L*u*v* space (L*, u*, v*). The modes of the probability density function (pdf) underlying the data in this space will correspond to the locations with highest data density. In terms of segmentation, it is intuitive that the data points close to these high density points (modes) should be clustered together. Note that these modes are also far less sensitive to outliers than the means of, say, a mixture of Gaussians would be.

The mean shift filtering step consists of finding the modes of the underlying pdf and associating with them any points in their basin of attraction. Unlike earlier techniques, the mean shift is a non-parametric technique and hence, we will need to estimate the gradient of the pdf, f(x), in an iterative manner using kernel density estimation to find the modes. For a data point x in feature space, the density gradient is estimated as being proportional to the mean shift vector:

$$\widehat{\nabla f}(\mathbf{x}) \propto \frac{\sum_{i=1}^{n} \mathbf{x}_i g\left(\left\| \frac{\mathbf{x}-\mathbf{x}_i}{h} \right\|\right)}{\sum_{i=1}^{n} g\left(\left\| \frac{\mathbf{x}-\mathbf{x}_i}{h} \right\|\right)} - \mathbf{x} \qquad (1)$$

where x_i are the data points, x is a point in the feature space, n is the number of data points (pixels in the image), and g is the profile of the symmetric kernel G. We use the simple case where G is the uniform kernel with radius vector h. Thus, the above equation simplifies to:

$$\widehat{\nabla f}(\mathbf{x}) \propto \left[\frac{1}{|S_{\mathbf{x},h_s,h_r}|} \sum_{\mathbf{x}_i \in S_{\mathbf{x},h_s,h_r}} \mathbf{x}_i \right] - \mathbf{x} \qquad (2)$$

where S_x,h_s,h_r represents the sphere in feature space centered at x and having spatial radius h_s and color (range) radius h_r, and the x_i represent the data points within that sphere. For every data point (pixel in the original image) x we can iteratively compute the gradient estimate in Eqn. 2 and move x in that direction, until the gradient is below a threshold. Thus we have found the points where $\widehat{\nabla f}(\mathbf{x}') = 0$, the modes of the density estimate. We can then replace the point x with x0, the mode with which it is associated. Finding the mode associated with each data point helps to smooth the image while preserving discontinuities. Intuitively, if two points xi and xj are far from each other in feature space, then xi $\notin$ $S_{x,hS,hr}$ and hence xj doesn't contribute to the mean shift vector gradient estimate and the trajectory of xi will move it away from xj . Hence, pixels on either side of a strong discontinuity will not attract each other. However, filtering alone does not provide segmentation as the modes found are noisy. This "noise" stems from two sources. First, the mode estimation is an iterative process; hence it only converges to within the threshold provided (and with some numerical error). Second, consider an area in feature space larger than $S_{xj,hs,hr}$ and where the color features are uniform or have a gradient of 1. Since the pixel coordinates are uniform by design, the mean shift vector will be 0 in this region, and the data points will not move and hence not converge to a single mode. Intuitively, however, we would like all of these data points to belong to the same cluster in the final segmentation. For these reasons, mean shift filtering is only a preprocessing step, and a second step is required in the segmentation process: clustering of the filtered data points {x'}.

- **Clustering:** After mean shift filtering, each data point in the feature space has been replaced by its corresponding mode. As described above, some points may have collapsed to the same mode, but many have not despite the fact that they may be less than one kernel radius apart. Clustering is described as a simple post-processing step in which any modes that are less than one kernel radius apart are grouped together and their basins of attraction are merged. This suggests using single linkage clustering, which effectively converts the filtered points into segmentation [3].

- **Discussion:** Mean shift filtering using either single linkage clustering or edge-directed clustering produces segmentations that correspond well to human perception. This algorithm is quite sensitive to its parameters. The mean shift filtering stage has two parameters corresponding to the bandwidths (radii of the kernel) for the spatial (hs) and color (hr) features. Slight variations in hr can cause large changes in the granularity of the segmentation, as shown in Figure 1. By adjusting the color bandwidth we can produce over-segmentations as in Figure 1-b which shows every minute detail, to reasonably intuitive segmentations as in Figure 1-f which delineate objects or large patches, to under-segmentations as in Figure 1-g which obscure the important elements completely. This issue is a major stumbling block with respect to using mean shift segmentation as a reliable preprocessing step for other algorithms, such as object recognition. For an object recognition system to actually use a

segmentation algorithm, it requires that the segmentations produced be fairly stable under parameter changes and that the same parameters produce stable results for different images, thus easing the burden of parameter tuning.

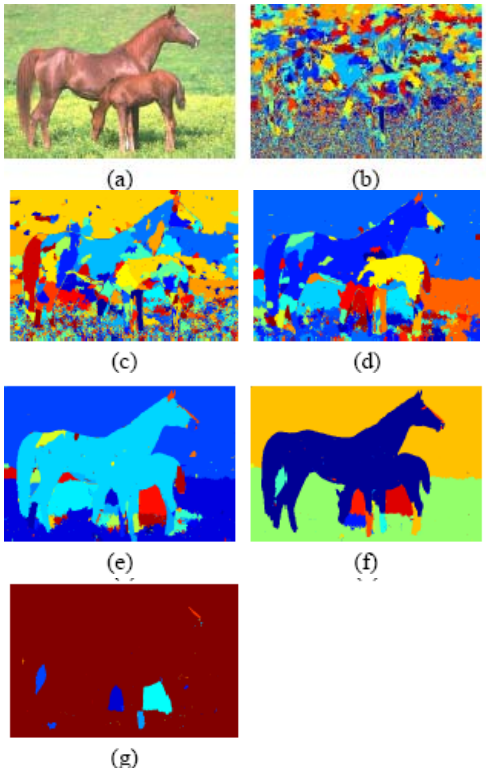


Figure 1: *Changing scores for different segmentation granularities: (a) Original image, (b)-(g) mean shift segmentations using scale bandwidth (hs) 7 and color bandwidths (hr) 3, 7, 11, 15, 19 and 23 respectively.*

2.1.2 Efficient Graph-based Segmentation

Efficient graph-based image segmentation is another method of performing clustering in feature space. This method works directly on the data points in feature space, without first performing a filtering step, and uses a variation on single linkage clustering. The key to the success of this method is adaptive threshold. To perform traditional single linkage clustering, a minimum spanning tree of the data points is first generated (using Kruskal's algorithm [4]), from which any edges with length greater than a given hard threshold are removed. The connected components become the clusters in the segmentation. This method eliminates the need for a hard threshold, instead of replacing it with a data-dependent term. More specifically, let G = (V,E) be a (fully connected) graph, with m edges and n vertices. Each vertex is a pixel, x, represented in the feature space. The final segmentation will be S = (C1, ...,Cr) where Ci is a cluster of data points. The algorithm is shown in table 1.

We can make the algorithm more efficient by considering only the 100 shortest edges from any vertex instead of the fully connected graph. This does not result in any perceptible quality loss. In contrast to single linkage clustering which uses a constant K to set the threshold on edge length for merging two components in Eqn. 3, efficient graph-based segmentation uses a variable threshold. This threshold effectively allows two components to be merged if the minimum edge connecting them does not have length greater than the maximum edge in either of the components' minimum spanning trees, plus a term. $\tau = \frac{k}{|C_{\mathbf{x}_i}^{t-1}|}$. as defined here, τ is dependent on a constant k and the size of the component. Note that on the first iteration, li = 0 & lj = 0, and $|C_{\mathbf{x}_i}^0| = 1$ & $|C_{\mathbf{x}_j}^0| = 1$. So k represents the longest edge which will be added to any cluster at any time, k = lmax. Also, as the number of points in a component increases, the tolerance on added edge length for new edges becomes tighter and fewer mergers are performed, thus indirectly controlling region size. However, it is possible to use any non-negative function for τ which reflects the goals of the segmentation system. The merging criteria in Eqn. 3 allows efficient graph-based clustering to be sensitive to edges in areas of low variability, and less sensitive to them in areas of high variability. This is intuitively the property we would like to see in a clustering algorithm. However, the results it gives do not have the same degree of correctness as mean shift-based segmentation, as demonstrated in Fig. 2. This algorithm also suffers somewhat from sensitivity to its parameter, k.

Table 1: *Algorithm of Graph-Based Segmentation*

1. Sort $E = (e_1, ..., e_m)$ such that $|e_t| \leq |e_{t'}| \, \forall t < t'$
2. Let $S^0 = (\{\mathbf{x}_1\}, ..., \{\mathbf{x}_n\})$, in other words each initial cluster contains exactly one vertex.
3. For $t = 1, ..., m$
   (a) Let $\mathbf{x}_i$ and $\mathbf{x}_j$ be the vertices connected by e_t.
   (b) Let $C_{\mathbf{x}_i}^{t-1}$ be the connected component containing point $\mathbf{x}_i$ on iteration $t-1$, and $l_i = \max_{\text{mst}} C_{\mathbf{x}_i}^{t-1}$ be the longest edge in the minimum spanning tree of $C_{\mathbf{x}_i}^{t-1}$. Likewise for l_j.
   (c) Merge $C_{\mathbf{x}_i}^{t-1}$ and $C_{\mathbf{x}_j}^{t-1}$ if
   $$|e_t| < \min\{l_i + \frac{k}{|C_{\mathbf{x}_i}^{t-1}|}, l_j + \frac{k}{|C_{\mathbf{x}_j}^{t-1}|}\} \quad (3)$$
   where k is a constant.
4. $S = S^m$

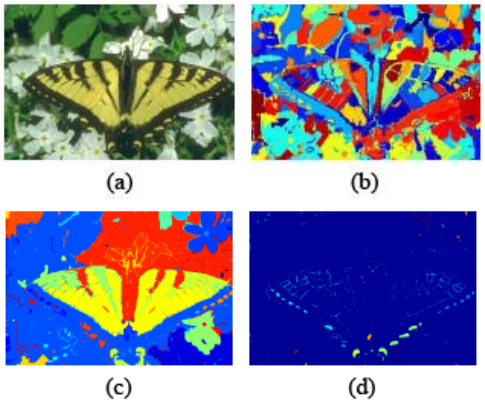


Figure 2: *Changing scores for different parameters using efficient graph-based segmentation: (a) Original image, (b)-(d) efficient graph-based segmentations using scale bandwidth (hs) 7, color bandwidth (hr) 7 and k values 5, 25, and 125 respectively.*

2.1.3 Hybrid Segmentation Algorithm

An obvious question emerges when describing the mean shift based segmentation method and the efficient graph based

clustering method: can we combine the two methods to give better results than either method alone? More specifically, can we combine the two methods to create more stable segmentations that are less sensitive to parameter changes and for which the same parameters give reasonable segmentations across multiple images? In an attempt to answer these questions, the third algorithm we consider is a combination of the previous two algorithms: first we apply mean shift filtering, and then we use efficient graph-based to give the final segmentation. The result of applying this algorithm with different parameters can be seen in Fig 3. Notice that for hr = 15 the quality of the segmentation is high. Also notice that the rate of granularity change is slower than either of the previous two algorithms, even.

3. EVALUATION AND COMPARISON

The first comparison we performed considered the correctness of the fourth algorithms. All three algorithms had the potential to perform equally well on the dataset given the correct parameter choice. On average over the parameter set, however, the hybrid algorithm performed slightly better than the mean shift algorithm, and both performed significantly better than the graph-based segmentation. We can conclude that the mean shift filtering step is indeed useful, and that the most promising algorithms are the mean shift segmentation and the hybrid algorithm.

The second comparison we performed considered stability with respect to parameters. In this comparison, the hybrid algorithm showed less variability when its parameters were changed than the mean shift segmentation algorithm. Although the amount of improvement did decline with increasing values of k, the rate of decline was very slow and any choice of k within our parameter set gave reasonable results. Although the graph-based segmentation did show very low variability with k = 5, changing the value of k decreased its stability drastically.

Finally, we compared the stability of a particular parameter choice over the set of images. Once again, we see that the graph-based algorithm has low variability when k = 5, however its performance and stability decrease rapidly with changing values of k. The comparison between the mean shift segmentation and the hybrid method is much closer here, with neither performing significantly better. For the three characteristics measured, we have demonstrated that both the mean shift segmentation and hybrid segmentation algorithms can create realistic segmentations with a wide variety of parameters; however the hybrid algorithm has slightly improved stability.

The complexity of our proposed algorithm (Mean-Graph) is O(n*m).

4. CONCLUSIONS

In this article we compared and evaluated image segmentation algorithms. Our work consists of comparing the performance of segmentation algorithms based on three important characteristics: correctness, stability with respect to parameter choice, and stability with respect to image choice. If an algorithm performs well with respect to all of these characteristics, it has the potential to be useful as part of a larger vision system. For our comparison task, we chose to compare two popular segmentation algorithms: mean shift-based segmentation and graph-based segmentation scheme. We also proposed a hybrid algorithm which first performs the first stage of mean shift-based segmentation, mean shift filtering, and then applies the graph-based segmentation scheme, as an attempt to create an algorithm which preserves the correctness of the mean shift-based segmentation but is more robust with respect to parameter and image choice. Thus, we would choose to incorporate the hybrid method into a larger system.

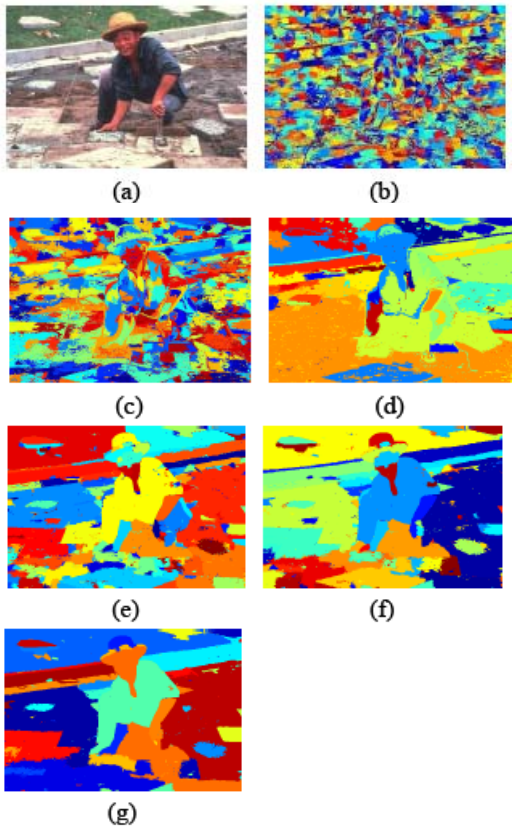


Figure 3: *Changing scores for different parameters using a hybrid segmentation algorithm which first performs mean shift filtering and then efficient graph-based segmentation: (a) Original image, (b)-(g) segmentations using scale bandwidth (hs) 7, and color bandwidth (hr) and k value combinations (3, 5), (3, 25), (3,125), (15, 5), (15, 25), (15,125) respectively.*

5. REFERENCES

[1] D. Comaniciu, P. Meer, "Mean shift: A robust approach toward feature space analysis", IEEE Trans. on Pattern Analysis and Machine Intelligence, 2002, 24, pp. 603-619

[2] P. Felzenszwalb, D. Huttenlocher, "Efficient Graph-Based Image Segmentation", Intl Journal of Computer Vision, 2004, 59 (2).

[3] C. Christoudias, B. Georgescu, P. Meer, "Synergism in Low Level Vision", Intl Conf on Pattern Recognition, 2002, 4, pp. 40190.

[4] Leo Grady and Eric L. Schwartz; "Isoperimetric Graph Partitioning for Data Clustering and Image Segmentation" IEEE Transactions on Pattern Analysis and Machine Intelligence, VOL. XX, 2004.

Image segmentation by means of optimal approximations

Mikhail Kharinov
St. Petersburg Institute for Informatics and Automation of RAS,
St. Petersburg, Russia
khar@iias.spb.su

Abstract

In the report the segmentation problem of a priory unknown image is discussed. The solution is developed in framework of Mumford–Shah model. To formalize the result of segmentation of any image into each number of segments avoiding an appeal to prescribed image segmentation method or algorithm the optimal image approximations are used. The insufficiency of iterative segment merging employed in Mumford–Shah model to obtain the optimal approximations of an image is established. The improvement of optimization results is demonstrated.

Keywords: *Mumford–Shah Model, Optimal Approximation, Segment Number, Standard Deviation.*

1. INTRODUCTION

Image segmentation is an important problem in preliminary image analysis described as certain partitioning of an image into proper segments. The one of known correct solutions consists in specifying a certain functional, which attains the minimal value for the segmented image compared with the values of this functional for arbitrarily partitioning of an image into segments. At that the definition of image segmentation result does not appeal nor to a visual perception nor to a priori prescribed image segmentation technique. So for image of any content it appears possible to obtain the minimal values of the numerical functional and calculate the proper segmented representations. But to do so some known theoretical and computational obstacles should to be overcome. We start to solve the mentioned task in terms of optimal piecewise constant image approximations which are used in the Mumford–Shah model [2, 4, 6, 7, 9–12].

According to the Mumford–Shah model, a functional to be minimized usually depends on standard deviation of approximation from an image and the total length of boundaries between image segments. Accounting for segment boundaries was introduced to describe a non-trivial minimum of the functional for a certain number of segments. At the same time it complicates the optimization problem and forces the use of an extra control parameter [6, 7, 9, 10]. In special case of ignoring of segment borders the optimization reduces to minimizing the standard deviation of approximation from an image [2]. In this case the optimal approximation appears trivial and coincides with the partition of an image into separate pixels, or segments of identical pixels corresponded to zero value of the functional. However, for meaningful statement of the problem avoiding any functional complications it seems perfect to minimize the functional for each possible segment number that guarantees the existence of solutions and simplifies the interpretation of the functional. On the first sight the requirement of minimizing of functional for each segment seems a generalization of the problem statement that commonly used. But it should be noted that the consideration of a sequence of optimal approximations instead of single approximation maintains the logic of practical computations as the selection of the desired image approximation from a set of calculated ones is often provided within a supplementary «stopping condition» [2, 6, 11] that treated in addition to initial minimization requirement. Nevertheless, for more effective utilization of the model it is useful to clarify and carefully experimentally verify the concept of optimal approximation avoiding an overestimation of the results of computer calculations.

2. OPTIMAL APPROXIMATION PROBLEM

Let us illustrate the notion of optimal approximation by the example of the digital image of the four pixels, comparing the image and its approximation by the standard deviation $StdDev$ of approximation from the image (Fig. 1).

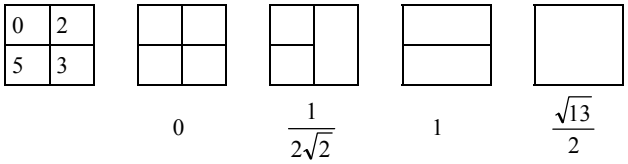


Figure 1: Image partitions inducing optimal approximations.

Leftmost matrix in Fig. 1 is an image example. Intensity values of pixels are written in the cells. Other matrices describe the partitions of the image that are converted to piecewise constant approximations by filling the segments with averaged intensity values. The values of $StdDev$ are written under the matrices.

The piecewise constant image approximation is called *optimal* if in comparison with any other approximation constituted of the same number of segments less differs from the image by the value of the standard deviation $StdDev$, calculated as the square root of the additive square error SE divided by the number of pixels N: $StdDev = \sqrt{SE/N}$.

By direct enumeration of possibilities one can check up that Fig. 1 demonstrates just partitions inducing the optimal approximations. Comparing in Fig. 1 the partitions of the image into two and three segments, it is easy to notice that segments from different partitions are not nested within each other but overlap one another. Therefore, it is impossible to obtain the sequence of optimal approximations of an image into $n = 1, 2, 3, ...N$ segments by iterative segment merging, which generally gives inaccurate results since the second iteration.

It should be noted that the computational problem of the optimal approximation has no an exact practical solution so far, even for individual standard images of actual dimensions. Therefore the

reliable computation of non optimal, but only optimized image approximations is further discussed (Fig. 2).

31.60

Figure 2: Nearly optimal piecewise constant image approximation with two segments. The original image is shown on the left and its contrasted approximation is shown on the right.

Fig. 2 by the example of the standard Lenna image demonstrates an optimized approximation which contains two segments. To increase a visual clarity, the averaged within each segment intensity values of pixels in piecewise constant image approximation replaced with the values 0 and 255. The optimized approximation called a *nearly optimal* is characterized by a minimum value of standard deviation *StdDev* compared with other piecewise constant approximations that obtained in available algorithms. The appropriate value of standard deviation *StdDev* is written under the approximation.

In Fig. 2 either segment of nearly optimal approximation covers several objects connected to each other with connective lines and other ties. Connectivity in this case means that to recolor the approximation it is enough to click once on the white and black pixels in "Flood Fill" mode of any image editor. To notice the mutual connectivity of the black or white objects, one should pay attention to the partially black and partially white frame around the perimeter of the image approximation. Ignoring the increase of the *StdDev* caused by the connective elements, it may be derived that the lower limit of the minimum possible standard deviation exceeds the value 30.65. So we suppose that the value of the standard deviation $StdDev = 31.60$ is small enough to nearly optimal approximation possesses the features of the optimal approximation. Most likely, the latter merely presents a slightly larger number of objects. Thus, the nearly optimal approximation of the image prompts a very simple idea of constitution of optimal approximation that presents the object in the image as mutually connected regions.

3. MUMFORD–SHAH SEGMENTATION

The original approach to the problem of image segmentation as a problem of optimization was founded by D. Mumford and J. Shah in former century in [9, 10] followed by a number of subsequent key studies [2, 4, 6, 11, 12] that are implemented into practice of image processing e.g. in program package «ENVI» (Environment for Visualizing Images, http://www.ittvis.com/portals/0/pdfs/envi/Feature_Extraction_Module.pdf). These studies base on the formalization of segmentation problem as the minimization of a functional presented as:

$$SE + \lambda L = \min, \tag{1}$$

where SE (abbreviated «square error») is the sum of squared deviations of intensities within the segment 1 and 2 from intensity mean values, L is the total length of the boundaries between adjacent segments and λ is a parameter which may depend on the number of segments according to version [6] and also on the image itself according to the version FLSA (Full λ–Scheduled Algorithm) [4, 11, 12]. Though the dependence of the lambda from an image violates the clear interpretation of functional, which is divided into two terms, just this version of the Mumford–Shah model is preferably implemented to the practice of image processing due to exclusion of control parameters. In the special case of $\lambda = 0$ the task is reduced to minimization of the standard deviation *StdDev* of the image from its piecewise constant approximation [2].

In accordance with Mumford–Shah model the segmentation task is solved using the algorithms of iterative segment merging. At that the criterion for merging of adjacent segments 1 and 2 according to FLSA version is presented as:

$$\frac{\Delta SE(1,2)}{l(1,2)} \equiv \frac{(\Delta I(1,2))^2}{l(1,2)} \cdot \frac{S(1)S(2)}{S(1)+S(2)} = \min, \tag{2}$$

where $\Delta I(1,2)$ is the difference of intensities averaged within the segments, $S(1)$ and $S(2)$ are the segment squares (pixel numbers), $l(1,2)$ is the length of common boundary between the segments 1 and 2, $\Delta SE(1,2) \equiv SE(1 \cup 2) - SE(1) - SE(2)$ is the value of nonadditivity of SE, estimated for segments 1, 2 and their union $1 \cup 2$.

In the version [6] of Mumford–Shah model the merging criterion is expressed as:

$$\Delta SE(1,2) - \lambda \cdot l(1,2) = \min \tag{3}$$

and in the special case [2] for zero λ is reduced to:

$$\Delta SE(1,2) = \min \tag{4}$$

The latter is called in [12] «weighted intensity difference».

The formulae (3) and (4) simply describe the increment of the functional value for current iteration of segment merging and provide the strict minimization of (1) at merging of first pair of pixels. To avoid the control parameters in [11] the estimation of λ as the minimal value that formally equates to zero the expression (3) whence formula (2) follows was suggested detrimentally to the source idea of minimizing of functional (1). Perhaps that is why in mentioned papers with exception of [7] there is a shortage of the experimental data that would clearly indicate the obtained values of minimized functional.

In any case the optimized image approximations do not perfectly coincide with the optimal ones, due to the fact that the sequence of the partitions inducing the optimal approximations, in general,

is not hierarchical. Moreover the results of the iterative segmentation of an image into several segments depend on the variable intensity of image pixels, the initial partition of the image into indivisible *superpixels*, and even from rounding errors. Therefore, a quantitative assessment of the optimality of the discussed approximations of the image seems to be useful for providing of robust segmentation by selecting the best approximation of available ones. As a unified criterion of approximation optimality it is convenient to accept the standard deviation $StdDev$ of the image from piecewise constant approximation.

4. STANDARD DEVIATION ANALYSIS

In Fig. 3 the dependencies of standard deviation $StdDev$ on the segment number in the range from 1 to 1000 are presented for standard Lenna image.

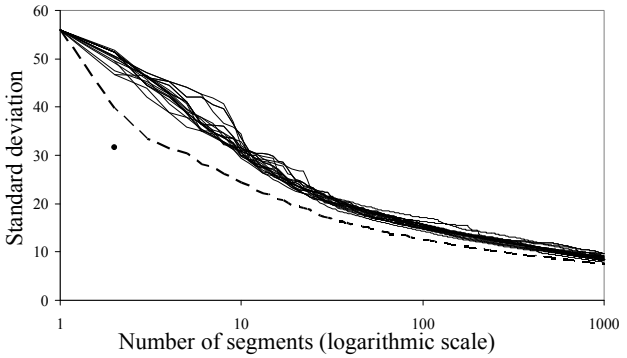


Figure 3: The dependencies of $StdDev$ on segment number.

In Fig. 3 the twenty black interlaced curves show the dependencies of $StdDev$ on the segment number for different versions of Mumford–Shah model. These weakly decreasing curves describe the hierarchies of nested partitions obtained by iterative merging of adjacent segments with the minimal «weighted intensity difference» (4), minimal value of criterion (2) and a number of experimentally chosen another functions of segment features. To generate the partitions inducing the appropriate piecewise constant image approximations the variation of initial image partitioning into superpixels and normalization of intensities to invariant values with respect to certain image transformations were also used.

Dashed curve corresponds to a non–hierarchical sequence of partitions with overlapping of segments from different partitions. This curve describes the partitions inducing piecewise constant image approximations obtained by iterative merging of adjacent segments with the minimal value of criterion (4) in turn with automatic correction of the segment boundaries accompanied with reduction of the standard deviation value by means of a pilot version of algorithm implementing the methodology of active contours [1, 3, 5, 8] combined with Monte Carlo method [7]. Then the selection of soundness segmentation results according to the monotony condition:

$$n_1 \le n_2 \quad \Rightarrow \quad StdDev(n_1) \ge StdDev(n_2) \tag{5}$$

was performed, where n_1, n_2 are the numbers of segments in partitions.

The bold dot in Fig. 4 marks the partition inducing the nearly optimal approximation of the image with two segments presented above in Fig. 2. To obtain this partition the segment boundaries were corrected in interactive mode as in [1].

The twenty curves in Fig. 3 fill some strip over the infimum of possible values of the standard deviation for successive values of the segment number. Dashed curve describing a sequence of overlapping partitions, lies lower than twenty curves describing the hierarchical partitions, and for images partitioning into relatively small number of segments shows a noticeable decrease in the standard deviation. It is remarkable that the value of the standard deviation for nearly optimal approximation marked with the bold dot turns out to be one third less than the value for the approximation calculated in Mumford–Shah model by segment merging only. The discussed value of standard deviation is also significantly less than the improved value obtained by automatic segmentation. Thus the opportunities to improve the automatic segmentation have not yet been exhausted.

Based on our first results on optimized approximations obtaining for standard and other images followed by their improvement, we venture to assert that for an image approximations which are equalized in segment number, the increase in the value of the standard deviation as a rule is accompanied with the smearing of some visually observed objects (Fig. 4).

16.47 12.67

Figure 4: Optimized image approximations with 99 segments.

Fig. 4 by example of standard Lenna image demonstrates the approximations consisting of equal number of segments. The values of $StdDev$ are written under the approximations.

The approximation on the right in Fig. 4 characterized by a smaller value of standard deviation looks more similar to the original than the approximation on the left, characterized by a larger value of standard deviation. The latter is also characterized by better visual quality. With regard to the visual quality of an image approximation it should be noted, that the loss of those or other objects may improve the visual quality of remaining ones though implies some increase in standard deviation value.

5. CONCLUSION

It should be noted that Mumford–Shah model essentially involves the generation of a multilevel hierarchical partitions of an image into the segments of computed shapes, which at first sight, requires the multiple iterations and laborious optimization of algorithms especially in speed. However, the problem of direct program speedup owing to reducing of the iteration number is solved

quite simply in the algorithms of synchronous segments merging over the whole image as in version SLBM (Synchronous locally best merging) of the model [4]. But, as rightly noted by the authors of [4], in this case the optimality of approximations may be disturbed due to deviation from the adopted in a model technique of generation of optimized approximations by successive segment merging. This obstacle is avoidable by applying a special data structure that supports the storage of a hierarchy of approximations in lesser part of RAM and provides the conversion of one hierarchy into another to obtain the desired hierarchical sequence of partitions. The applied scheme of optimized computing and our variant of mentioned version of the model providing the generation of mostly binary segment hierarchy and the increasing of processing speed up to over half a million pixels per second is planned to outline in the report.

Probably just the laboriousness of programming restrains the widest propagation of the Mumford–Shah model. Nevertheless, at the present level of computer processing of digital image the segmentation problem, consisting in accurate definition of segmented image, is more practical than theoretical. The theoretical solution is provided, at least, in the Mumford–Shah model that treats the segmented image as an optimal piecewise constant image approximation.

The main limitation of the practical implementation of the Mumford-Shah model consists in that traditional methods for generating of optimized approximations are often restricted to utilization of iterative segment merging. Since the iterative segment merging is insufficient to obtain strictly optimal approximations, the practical results of Mumford–Shah segmentation do not quite match the desired optimal results. In this case, the optimized approximations essentially depend on the used set of segmentation algorithms and are characterized by excessive values of functional. Functional values themselves are quite inactively used in practical calculations. Meanwhile, they seem to be necessary to compensate the effect of variability of image content and object representation, as well as a number of other factors affecting an image partitioning into a relatively small number of segments by means of iterative algorithms.

Probably the standard deviation should be perfectly investigated as the basic functional before the others functional assumptions. Utilization of active contour methods in frameworks of the Mumford–Shah model improves the segmentation results which assessed by the values of the standard deviation, but causes the renewal of computational challenges that arise at a new level. Overcoming these challenges to obtain the optimal image approximations that characterized by reliably minimal values of the standard deviation and more complete disclosure of capabilities of the Mumford-Shah model is the subject of further research. Particularly we focus on lower estimation of attainable minimum of standard deviation for each number of segments, the development of deterministic algorithms for fast generating of optimal approximation series and the generalization of data structures for storage and transformations of a sequence of overlapping image partitions.

In the report additionally to the example of the standard image Lenna the experimental results are illustrated in more detail by the example from the Berkeley Segmentation Dataset and Benchmarks (http://www.eecs.berkeley.edu/Research/Projects/CS/vision/bsds/).

6. ACKNOWLEDGMENTS

This work is supported by the Russian Foundation for Basic Research, grant no. 11–07–00685–a.

7. REFERENCES

[1] Bucha V.V, Ablameyko S.V. Algorithm of Interactive Image Segmentation based on dynamic programming // *Informatics*, United Institute of Informatics Problems of the National Academy of Sciences of Belarus (UIIP NASB), Vol. 1 (9), 2006. — pp. 5–15. (in Russian)

[2] Bugaev A.S., Khelvas A.V. Pilot studies and development of methods and tools for analysis and automatic recognition of media flows in global information systems // *Scientific and Technical Report*, Moscow: Moscow Institute of Physics and Technology (MIPT), Vol. 1, 2001. — 140 p. (in Russian).

[3] Chan T.F., Vese L.A. Active contours without edges // *IEEE Trans. Image Process.*, Vol. 10, 2001. — pp. 266–277.

[4] Crisp D.J., Tao T.C. Fast Region Merging Algorithms for Image Segmentation // *The 5th Asian Conf. on Computer Vision (ACCV2002)*, Melbourne, Australia, 23–25 January 2002. — pp. 1–6.

[5] Kaas M., Witkin A., Terzopoulos D. Snakes: Active Contour Models // *Int. Journal of Computer Vision*, Vol. 1, No 4, 1988. — pp. 321–331.

[6] Koepfler G., Lopez C., Morel J. A Multiscale Algorithm for Image Segmentation by Variational Method // *SIAM Journal on Numerical Analysis*, Vol. 31, No 1, 1994. — pp. 282–299.

[7] Yan Nei Law, Hwee Kuan Lee, Yip A.M. A Multi-resolution Stochastic Level Set Method for Mumford-Shah Image Segmentation // *IEEE Trans. Image. Process.* , Vol. 17, Issue 12, 2008. — pp. 2289–2300.

[8] Mendi E., Milanova M. Image Segmentation with Active Contours based on Selective Visual Attention // *Proc. of the 8th WSEAS Int. Conf. on Signal Processing*, 2009. — pp. 79–84.

[9] Mumford D., Shah J. Boundary detection by minimizing functionals, I // *Proc. IEEE Comput. Vision Patt. Recogn. Conf.*, San Francisco, 1985. — pp. 22–26.

[10] Mumford D., Shah J. Optimal Approximations by Piecewise Smooth Functions and Associated Variational Problems // *Communications on Pure and Applied Mathematics*, Vol. XLII, No 4, 1989. — pp. 577–685.

[11] Redding N.J., Crisp D.J., Tang D.H., Newsam G.N. An efficient algorithm for Mumford–Shah segmentation and its application to SAR imagery // *Proc. Conf. Digital Image Computing Techniques and Applications (DICTA '99)*, 1999. — pp. 35–41.

[12] Robinson B.J., Redding N.J., Crisp D.J. Implementation of a fast algorithm for segmenting SAR imagery // *Scientific and Technical Report*, Australia: Defense Science and Technology Organization, Australia, 01 January 2002. — 42 p.

About the author

Mikhail Kharinov is senior researcher of St. Petersburg Institute for Informatics and Automation of RAS. His contact email is khar@iias.spb.su.

SAR Image Classification Utilizing Hausdorff Similarity

Xiao Yuan, Tianze Chen, Tao Tang, Yi Su
School of Electronic Science and Engineering
National University of Defense Technology, Changsha, China
yuanxiao@nudt.edu.cn

Abstract

Aiming at the image classification problem of Synthetic aperture radar (SAR), a classifier based on image intensity similarity is constructed. This approach calculates the similarity between two images by Hausdorff kernel, which is then fed into Support vector machines (SVM) is used, to eventually accomplish the task of classification. Experiments carried on field and simulated data verify its efficiency.

Keywords: *SAR, Image Classification, SVM, Kernel.*

1. INTRODUCTION

SAR image automatic target recognition (ATR) has developed rapidly during the last decades. With the launching of Moving and Stationary Target Acquisition and Recognition (MSTAR) project, and the releasing of images of several military vehicles, many researches on SAR ATR have been carried out on this public database. From a system point of view, SAR ATR can be categorized into template-based and model-based recognition. The main difference lies at the way how to generate the reference image; the recognition method itself has no distinguished difference.

SVM is an efficient classifier with great generalization ability, which was developed with statistical learning theory by Vapnik[8]. Since its naissance, popularity has been gained due to many attractive features; SAR ATR community also has benefit from it. Adopting SVM, Bryant[3] benchmarked its performance on MSTAR data set in 1999, and Zhao[9] gave a more thorough discussion in 2001. Since then, researches came forth and state-of-the-art results have been obtained with SVM[1]. However, most methods ignored image structure and context information which are much helpful for recognizing a target. To conquer this drawback, in this paper, we employ Hausdorff similarity between images to utilize image structural information, and feed the similarity as a kernel into SVM for classification.

The rest of the paper is structured as follows. Section 2 introduces the theory of SVM and kernel methods, section 3 discusses the Hausdorff similarity as a kernel function, section 4 presents the experiments, and a conclusion is given in section 5.

2. SUPPORT VECTOR MACHINE PRELIMINARY

The basic idea of Support vector machine[8] is to construct a hyperplane as the decision plane, here we give a brief introduction to the methodology for classifying two perfectly separated classes.

Given samples $\mathbf{x}_i$ and corresponding labels y_i: if $\mathbf{x}_i \in w_1$, $y_i = 1$ and if $\mathbf{x}_i \in w_2$, $y_i = -1$, The hyperplane which separates two classes is:

$$d(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}\mathbf{x} + b = 0 \tag{1}$$

where $\mathbf{w}$ denotes the normal vector to the hyperplane, and classification function is thus $\mathrm{sgn}(d(\mathbf{x}))$. Then optimization problem is to determine $\mathbf{w}$ and b that minimize $\|\mathbf{w}\|^2 / 2$ under the constraint:

$$y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b) \geq 1 \tag{2}$$

The solution can be found by quadratic programming method:

$$\mathbf{w}^* = \sum_{i=1}^{N} \alpha_i^* y_i \mathbf{x}_i \tag{3}$$

Then the hyperplane decision function is written as:

$$\begin{aligned} d(\mathbf{x}) &= \mathrm{sgn}\left[(\mathbf{w}^* \cdot \mathbf{x}) + b^* \right] \\ &= \mathrm{sgn}\left[(\sum_i \alpha_i^* y_i (\mathbf{x}_i \cdot \mathbf{x}) + b^* \right] \end{aligned} \tag{4}$$

In case of non-linear separation, samples can be projected into another feature space through a transform $\phi : \mathbf{x} \to \phi(\mathbf{x})$, and then the classification function becomes:

$$\begin{aligned} d(\mathbf{x}) &= \mathrm{sgn}\left[(\sum_i \alpha_i^* y_i \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}) + b^* \right] \\ &= \mathrm{sgn}\left[(\sum_i \alpha_i^* y_i K(\mathbf{x}_i, \mathbf{x}) + b^* \right] \end{aligned} \tag{5}$$

where the kernel function is defined as $K(\mathbf{x}_i, \mathbf{x}) = \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x})$.

Generally, the crucial factor of SVM classifier is the kernel function, which has certain desirable should satisfy Mercer condition[4], and can reflect similarity between input data etc.. Without a prior knowledge of the problem, polynomial and Gaussian kernels are commonly used to fit the training data. With the investigation of SAR images, noise and scintillation phenomena are usually the cases. We employ Hausdorff kernel for SAR classification considering its ability of handling structural information.

3. HAUSDORFF SIMILARITY

Hausdorff distance[5] has been used in computer vision nearly exclusively to match binary patterns of contours or edges. Hausdorff similarity can be induced from the directed Hausdorff distance. It's equivalent to fix a maximum distance between two sets, and see if the two sets are within that distance[6]. The definitions are given below.

3.1 Hausdorff Distance

Given two finite point sets A and B, the Hausdorff distance between A and B is defined as:

$$H(A,B) = \max(h(A,B), h(B,A)) \quad (6)$$

where

$$h(A,B) = \max_{a\in A} \min_{b\in B} \|a-b\| \quad (7)$$

and $\|\cdot\|$ denotes some underlying norm on the points.

3.2 Hausdorff Similarity

To compute the Hausdorff similarity[6] between two $N\times N$ gray level images A_{im} and B_{im}, neighborhood O and threshold ε are defined to compensate pixel migration and intensity distortion, The Hausdorff similarity between two images is thus:

$$H(A_{im}, B_{im}) = \sum_{i,j=1}^{N} U(\varepsilon - |A_{im}[i,j] - B_{im}[\hat{i},\hat{j}]|) \quad (8)$$

where U is the unit step function, and $(\hat{i},\hat{j}) \in O$ denotes the pixel in B_{im} most similar to $A_{im}[i,j]$ in O. Expression (8) evaluates the overlapping degree of two images with relaxed constraint on pixel correspondence. In order to restore symmetry which makes a metric has mathematical meaning, the Hausdorff similarity is defined by taking average:

$$K_{Hau}(A_{im}, B_{im}) = \frac{1}{2}(H(A_{im}, B_{im}) + H(B_{im}, A_{im})) \quad (9)$$

Note that though the function is not a real kernel unless the neighborhood is appropriately redefined, it could be used in most of the image classification cases[2]. In fact, this similarity measure has been shown to have nice features, such as be tolerant to small changes due to geometric distortions and noise. And a fast algorithm can be implemented with simple logical operations, which enhance the efficiency significantly. We refer the more detailed algorithm to [6].

4. EXPERIMENTS

In this section we present two groups of experiments on MSTAR field images and RadBase simulated images respectively.

4.1 Experiments on MSTAR Field Images

MSTAR database consists of a large number of 128x128 SAR images with one foot resolution, among which 3 types of targets used most frequently are BMP2, BTR70 and T72 as shown in Figure 1. There are 1622 training images recorded with a 17° depression angle and 1365 testing images recorded with a 15° depression angle.

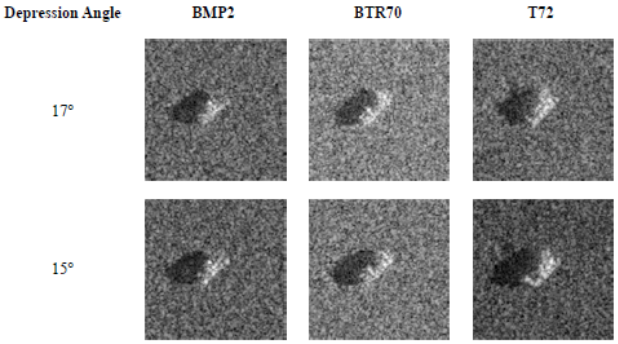


Figure 1: MSTAR SAR Images of Three Targets.

The MSTAR database assumes the same scale for all SAR images. It eliminates the need for scale, rotation and shift operations. In the benchmark work of Bryant[3], although SAR images vary a lot between azimuth gap great than 5°, dividing certain target into 360/(5*2)=36 different classes produced even disaster. SVM is in fact a two-class classifier, with the increasing of class numbers, more rejections appear naturally. To avoid this situation, we construct different SVMs for every single testing image with the training images located within a azimuth interval predefined.

On the other hand, image size, neighborhood size, and threshold are three parameters involved in Hausdorff similarity calculation. With consideration of image characteristics, we choose eight kinds of azimuth intervals (5°-40° for every 5°), three image sizes (32x32、48x48、64x64), and three neighborhood sizes (3x3、5x5、7x7) in our experiments. The threshold between pixels is fixed to 5, and the SVM parameter C is fixed to 1000. The classification results are listed from Table 1 to Table 3.

Table 1: Probability of Correct Classification for Image Size of 32x32

Azimuth Interval	Neighborhood Size		
	3x3	5x5	7x7
5°	0.3993	0.9868	0.9736
10°	0.3941	0.9897	0.9700
15°	0.3883	0.9905	0.9495
20°	0.37	0.9861	0.9311
25°	0.37	0.9802	0.9026
30°	0.3626	0.9729	0.8769
35°	0.359	0.9531	0.8608
40°	0.3568	0.9355	0.844

Table 2: Probability of Correct Classification for Image Size of 48x48

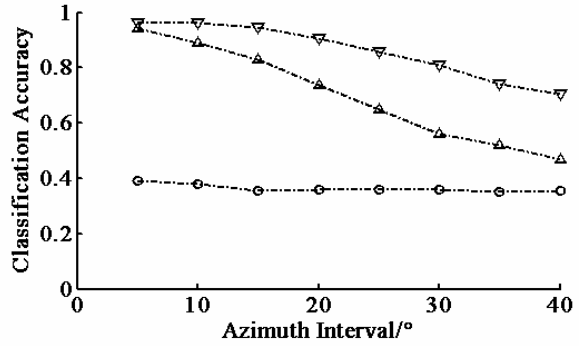
Azimuth Interval	Neighborhood Size		
	3x3	5x5	7x7
5°	0.9656	0.9817	0.4007
10°	0.9919	0.9927	0.3839
15°	0.9963	0.9897	0.3531
20°	0.9971	0.9868	0.3663
25°	0.9971	0.981	0.3495

30°	0.9956	0.9685	0.3436
35°	0.9956	0.9509	0.3465
40°	0.9912	0.9399	0.3538

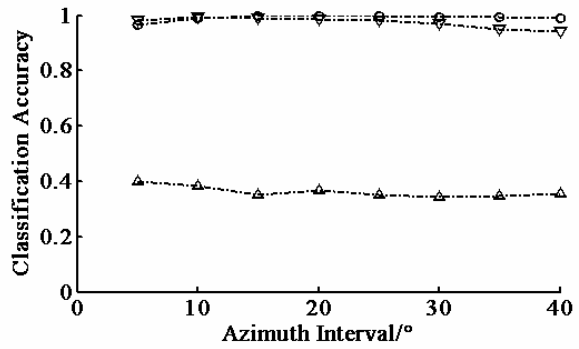
Table 3: Probability of Correct Classification for Image Size of 64x64

Azimuth Interval	Neighborhood Size		
	3x3	5x5	7x7
5°	0.3897	0.9604	0.9407
10°	0.3788	0.9626	0.8886
15°	0.356	0.9465	0.8293
20°	0.359	0.907	0.7377
25°	0.3575	0.8593	0.6469
30°	0.3612	0.8095	0.5604
35°	0.3531	0.7399	0.5216
40°	0.356	0.7055	0.4667

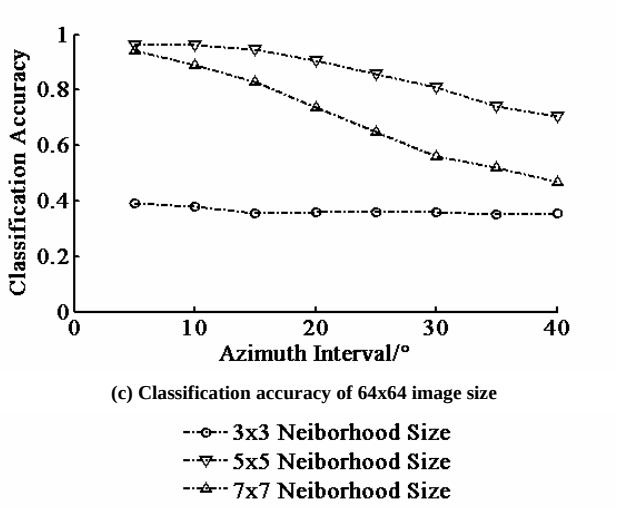
The experiments reveal that Hausdorff similarity achieves outstanding performance under certain parameter combinations, such as (32x32 image, 5x5 neighborhood), (48x48 image, 3x3 neighborhood) and (48x48 image, 5x5 neighborhood). While for some other combinations, the classification accuracy is stably low. There are also some parameter combinations in which azimuth matters a lot, performance decreases from high to low rapidly varying the azimuth intervals. We conclude that parameter settings will certainly affect Hausdorff similarity computation, and for these three targets in one foot SAR images, (48x48 image, 3x3 neighborhood, 20° Azimuth Interval) produces the highest performance. The graphic results are shown in Fig. 2.



(a) Classification accuracy of 32x32 image size



(b) Classification accuracy of 48x48 image size



(c) Classification accuracy of 64x64 image size

Figure 2: Experiment results under different conditions

The comparison to other SAR image classification algorithms[1] are listed in Table 4. In these experiments, our proposed algorithm provides significantly improved classification accuracy with moderate features.

Table 4: Results for the 3-class MSTAR Classification Problem

Classifier	Feature Set	Number of Features	Classification Accuracy
DAGSVM	7 EFS coefficients	26	0.9346
SVM	80x80 normalized image	6400	0.9099
Nearest Neighbor	QP normalized image	2304	0.9410
Max Value	MINACE	4096	0.9060
Hausdorff Similarity SVM	48x48 normalized image	2304	0.9971

4.2 Experiments on RadBase Simulated Images

In this section, we carry out experiments on RadBase simulated images with the optimal parameter setting to further validate our proposed algorithm with Hausdorff kernel SVM. RadBase[7] is a RCS calculation software developed by Surface Optics Corporation (SOC). It calculates the RCS and complex field data of targets as a function of frequency, polarization, and target/observer geometry using a hybrid geometrical/physical optics approach. We generate whole azimuth images for three targets with 17° depression angle and one foot resolution. Due to 10 errors occurred during simulation, the actual number of images is 1070. Example images are shown in Fig. 3.

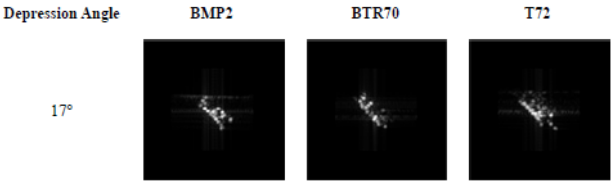


Figure 3: RadBase SAR Images of Three Targets.

The 1070 images are portioned into two parts randomly, one part as training data, and the left as testing data. The Hausdorff kernel and SVM parameters are as related in Section 4.1. Classification accuracy varied with training image portion is listed in Table 5.

Table 5: Probability of Correct Classification under Different Training Data Portions

Portion of Training Images	Testing Image Number	Correctly Classified Image Number	Classification Accuracy
0.1	959	817	0.8519
0.2	849	822	0.9682
0.3	758	747	0.9855
0.4	630	628	0.9968
0.5	529	528	0.9981
0.6	453	452	0.9978
0.7	297	296	0.9966
0.8	217	217	1
0.9	105	105	1

It can be seen that, when the portion of training images exceed 20%, the proposed classifier performs with satisfaction. While the portion increases, it outperforms its counterparts in section 4.1 rapidly and eventually achieve an incredible accuracy of 100%. It is because of the noiseless condition evolves with simulation, which is seldom the case in real applications.

5. CONCLUSION

The Hausdorff similarity SVM for SAR image classification is proposed, experiments on MSTAR field data and RadBase simulated data both yield excellent results. The introduced kernel function can deal with spatial contextual constraint as well as gray-scale values, which leading to a benign metric for SAR images. As for its relationship with Hausdorff distance, this approach can also be applied to more refined features like peak points, ridges, and ravines etc. It's worthy to notice that parameter selection is a key step to achieve fine performance.

6. ACKNOWLEDGMENTS

This work was supported by National Science Foundation project 61002023.

7. REFERENCES

[1] G. C. Anagnostopoulos. *SVM-based target recognition from synthetic aperture radar images using target region outline descriptors. Nonlinear Analysis, vol. 71, pp. e2934-e2939, 2009.*

[2] A. Barla, F. Odone, and A. Verri. *Hausdorff Kernel for 3D Object Acquisition and Detection." pp. 20-33.*

[3] M. Bryant, and F. Garber. *SVM classifier applied to the MSTAR public data set. pp. 355-360.*

[4] N. Cristianini, and J. Shawe-Taylor. *An Introduction to support Vector Machines and Other Kernel based Learning Methods, Cambridge, England: Cambridge University Press, 2000.*

[5] D. P. Huttenlocher, G. A. Klanderman, and W. J. Rucklidge. *Comparing Images Using the Hausdorff Distance" IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 15, no. 9, pp. 850-863, 1993.*

[6] F. Odone, E. Trucco, and A. Verri. *General Purpose Matching of Grey Level Arbitrary Images, Lecture Notes on Computer Science LNCS 2056. pp. 573-582.*

[7] *RadBase™ User's Guide Version 2.0, August 2000*

[8] V. N. Vapnik. *Statistical Learning Theory: John Wiley & Sons, Inc., 1998.*

[9] Q. Zhao, and J. C. Principe. *Support Vector Machines For Synthetic Aperture Radar Automatic Target Recognition, IEEE Transactions on Aerospace and Electronic Systems, vol. 37, no. 2, pp. 643-654, 2001.*

About the author

Xiao Yuan is a Ph.D. student at National University of Defense Technology, School of Electronic Science and Engineering. His contact email is yuanxiao@nudt.edu.cn.

Tianze Chen is a lecturer at National University of Defense Technology, School of Electronic Science and Engineering. His contact email is tzchen77@126.com .

Tao Tang is a lecturer at National University of Defense Technology, School of Electronic Science and Engineering. His contact email is tangtaonudt@gmail.com.

Yi Su is a professor at National University of Defense Technology, School of Electronic Science and Engineering. His contact email is y.su@yeah.net.

Face recognition system using 2D and 3D information fusion

Svetlana V. Korobkova, Archil Tsiskaridze
Vocord Company, Moscow, Russia
svetlana.korobkova@vocord.ru

Abstract

The system using both 3D and 2D face information has been developed and tested. We use 3D face surface reconstructed from 4 images. These face images are obtained from 4 calibrated high definition cameras without any additional structured illumination. 2D face texture information is obtained using frontal face 3D view with face texture. Neural networks are used both for 2D texture recognition and for 2D and 3D fusion. The developed face recognition system achieves the accuracy of 98,5%.

***Keywords**: 3D object reconstruction, 3D face recognition, 2D face recognition, texture face recognition, face recognition methods fusion.*

1. INTRODUCTION

2D face recognition systems are the most prevalent recognition systems when it is required that the person can be scanned without any collaboration. All face recognition systems using 2D face images have a row of shortcomings. Such systems are sensitive to illumination conditions, direction and type of light source. The illumination on the image depends on the type of the light source, face geometry and reflectance and camera parameters. This makes it very difficult to design 2D face model which could correctly deal with all these parameters. 3D face recognition systems are free of these drawbacks. 3D face surface remains the same in any lighting conditions [5].

Another drawback of 2D face recognition systems is pose variance. 2D face recognition algorithms, which are capable of processing and recognizing faces with different poses have to include special processing steps, such as applying special transformation function to faces with pose variations [6]. The accuracy of such methods depends highly on the accuracy of landmark points detection on the face with pose variations and shows bad results when a part of face is poorly illuminated. Moreover this task cannot be strictly solved in 2D case in theory because of perspective projection property of 2D images. To overcome this problem it has been proposed to record and keep faces with pose variations in the database [9]. Nevertheless this approach requires storing of large amount of data and reduces face recognition system performance. Several more approaches have been proposed to deal with faces with pose variations. They are eigen light-fields method, flexible appearance models, statistical face models for interpolating to unseen views of the face [4, 7], and aggregate face models [12]. The use of 3D information makes the task of constructing frontal face view much easier to solve and provides rather accurate projected frontal view of the probe face [1].

According to face recognition methods overview [11], among state-of-the-art face recognition methods it is worth to mention global methods (with correct recognition rate 90-96%) and statistical methods (93-100%). Face recognition methods based on parametric face model construction have the accuracy about 88-96%.

After the analysis of the state of the art technology in face recognition we made a decision to use both 3D and 2D face information for recognition which led to achieving high accuracy rates of the system.

The developed face recognition system implements the method which uses the fusion of 3D face surface information and 2D face texture information. Due to this face recognition system uses the advantages of both approaches and makes it possible to overcome the drawbacks of each of them when they are used as 2 separate systems. Using this approach made it possible to achieve 98.5% correct recognition rate.

The results of 2D and 3D face similarity rate are obtained separately and used as the inputs of a neural network classifier. The output value of this classifier is the similarity measure for the gallery and the probe face image.

The input data for the face recognition system are the 4 images containing subject face, obtained from the high definition calibrated cameras. These images are processed (face search), and after that 3D face surface is reconstructed. Using the 3D face model and combination of face textures from the input images the 2D face texture is formed. Then both face 3D surface form and 3D face texture are used to find the corresponding gallery face in case it exists.

The developed face recognition system consists of the following processing steps:

- Face and landmarks search
- 3D face surface reconstruction and construction of frontal face texture;
- 3D face recognition;
- 2D texture face recognition;
- 2D and 3D recognition results fusion.

2. 3D FACE SURFACE RECONSTRUCTION

3D face surface reconstruction is done using images from 4 time-synchronized cameras as input. Cameras have 1280 x 1024 resolution and frame frequency is 10 pictures per second. Cameras are grouped in 2 vertical pairs, which makes it possible to place them on both sides of the checkpoint system. The distance between the cameras in a pair is 20 cm, the distance between 2 vertical camera pairs is about 1 m.

The proposed algorithm for matching points search on the images obtained from 2 cameras forming a stereo pair is implemented with subpixel accuracy of ¼ pixel value (due to image interpolation). This makes possible to reach overall effective accuracy about 0.2 mm, which comes from camera sensors discreteness. Such accuracy is enough for further high quality face recognition.

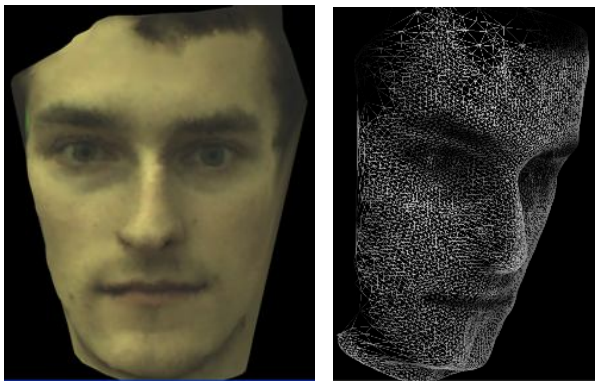
After the reconstruction using 2 camera stereo pairs is done the resulting 3D point sets are combined into one point set representing the face. The procedure of point sets combination is based on preliminary intrinsic and extrinsic camera calibration. Intrinsic camera parameters describe camera and lens properties and extrinsic describes camera positions and orientation. So the resulting point sets reconstructed from 2 stereo pairs on both sides of the checkpoint lay in the same coordinate system.

3D face surface reconstruction consists of the following steps:

- Building 3D point sets using each stereo pair and combining 2 point sets;
- Preliminary point set filtration;
- Building triangulated 3D face model (figure 1);
- Combining several face surfaces, obtained at different points of time into an aggregate 3D face surface.

To make 3D point set triangulation we use 2.5D triangulation method. The main idea of this method is projection 3D points to a 2D plane and then using 2D triangulation to produce 3D triangulated face surface. This method overcomes 3D triangulation methods both in reliability and in performance if applied to face 3D surfaces. Triangulation takes several milliseconds for the whole face surface.

The search for matching points on the images can be solved for each point independently. This feature of the algorithm makes possible significant speed up of the calculations due to performing them using parallel computation on a graphical processing unit. The reconstruction of a 3D object the image of which is 500x500 pixels (the usual size of the face in the image) with a 4 pixel step for 2 images from a stereo pair takes 5.1 sec. on Intel Core2 Q6600 2.4 GHz processor and only 0.17 sec. on NVIDIA GTX260 graphical processing unit. This provides 30 times speed up of the 3D surface reconstruction calculation.



(а) (б)

Figure 1: 3D face reconstruction. (а) – 3D surface with texture, (б) – triangulated face surface.

3D surface reconstruction can be made even under the condition of poor illumination of the face image (about 400-600 lux) with the distance from cameras to the reconstructed object up to 2.5 m. When using cameras with higher resolution, which become cheaper and easier to find, will make it possible to achieve good reconstruction results for even larger distances between cameras and the object of interest.

The 3D face model is based on parametric description, The construction of the face model is based on searching face parameters which fit closely to the surface obtained from cameras. If 3D face obtained from cameras includes artifacts (i.e. glasses, mustaches), they are eliminated, and thus do not contribute to recognition score of 3D recognition algorithm, see figure 2).

Figure 2: Face with glasses, glasses do not have volume and appear only as texture trails.

3. 3D FACE RECOGNITION

3D face recognition is done in two steps:

1. Fast database sample comparison using feature set of small length;
2. Precise comparison of several most similar database face samples.

Figure 3: Reconstructed 3D face surface - side view (while cameras are placed in front of the face).

4. 2D TEXTURE FACE RECOGNITION

To perform 2D face recognition first we form a synthetic face image, face texture is projected to the 3D face surface and the face is rotated to make frontal view of this face. The resulting image is then regarded as the input of 2D face recognition algorithm and called face texture. For this face texture we perform illumination correction procedure which yields face texture image with the specified mean illumination level.

For face recognition we use a set of classifiers, each of them analyzes different face parameters and outputs similarity rate for 2 face textures – probe face texture and database face texture. These classifiers are chosen to be neural network.

Each neural network classifier takes a vector of difference features as input. For each component of the difference feature vector we take its absolute value, so that classification is symmetric. The output of each classifier is the estimation of probability (the value from 0 to 1) that 2 faces are of the same person by this classifier.

The outputs of these classifiers are used as the input values for the overall classifier, which determines the probability of 2 face textures being of the same person, which is considered as the output of 2D face recognition system part.

The training on separate face texture neural network classifiers and of the overall classifier has been conducted on a training set of about 700000 difference features. We have collected and sorted face database and know the person name for each texture sample.

5. RESULTS

The developed face recognition system using both 3D and 2D information for recognition was tested on big enough database containing both 3D information and 2D texture for each face instance and known classes (person names) for these samples. We have reached 98,5 recognition accuracy rate at FAR = FRR point. The ROC plot for the proposed face recognition algorithm using 2D and 3D information fusion is shown in figure 4.

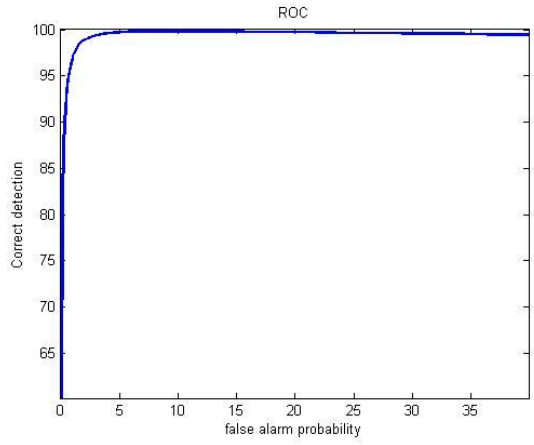


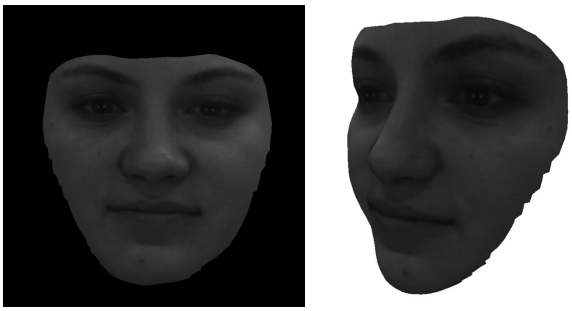
Figure 4: ROC of the face recognition system.

Our face recognition system proved to be stable when probe faces had different poses (such as rotation (+/- 30 degrees), leaning down and looking up faces (50/30 degrees up and down respectively)), moreover the system still can recognize a person if some parts of face are occluded (or there are no key points on this region, i.e. eyes, mouth).

The proposed face recognition algorithm is stable when input face has slight face expression variations: smiles, open mouth, squint faces.

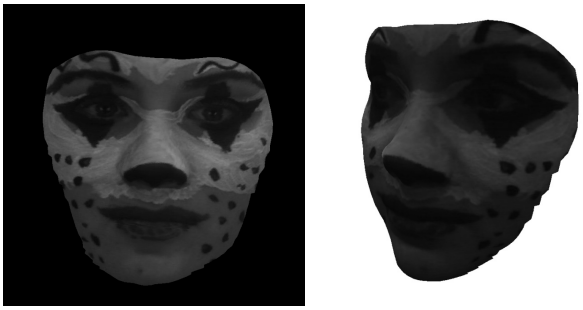
We have conducted experiments with painted faces and found out that face recognition system still gives correct answer for heavily painted faces due to the usage of 3D face information which is unchanged in case of painted faces.

Figure 5 shows the face sample (3D form and 2D texture) without painting. This sample has been used as a database sample. Figure 6 shows a painted face of the same person. This face was used as the probe face. Our system still could recognize the person whom this face belongs to, in spite of great difference of the textures.



а) б)

Figure 5: Original face: а) – texture, б) – 3D surface



а) б)

Figure 6: Painted face: а) – texture б) – 3D surface

Our system is able to work in 2D-only recognition mode, This mode is important if no preliminary 3D information about the face is available. In this case recognition is based only on 2D information (i.e. single personal photograph) stored in the database. However, the benefit of our full approach compared to other 2D face recognition methods is pose correction before image comparison with database. This correction is carried out based on our 3D reconstruction with face rotation and severely improves recognition rate.

To evaluate 2D recognizer itself we tested it on FERET data base. Consider situation when 2D photograph available is also not perfect: i.e. it is taken from another camera under poor illumination conditions, with variable head pose, glasses etc, In such situation it is important for 2D recognizer to show stable results within some image variation range, even if test image (acquired by the system) is purely frontal after 3D correction.

The basic steps of 2D recognizer are as follows and include preprocessing of the input image consisting of the following steps:

- face detection;
- landmark points search;
- deleting background;
- illumination correction.

After we get the input face image of specific size and pose we use it as the input for 2D face neural network classifiers similar to texture classifiers, but trained on the difference features of real face images.

This 2D face recognizer has been tested on FERET database, consisting of more than 2000 face images of about 700 different people. The images included pose variations with 10 degrees. Each person has more than 1 face instance in the database. Face pictures in FERET database were taken at different points of time, with different poses and in different illumination conditions.

Also the faces in FERET database have such variation as presence or absence of glasses, beard and moustache.

Our 2D only recognition system reached 88,5% correct recognition rate at FAR=FRR point.

6. CONCLUSION

We have designed and implemented face recognition system (figure 7) which uses both 3D and 2D information for recognition. 3D face information is obtained using only high definition cameras without any help of additional structured light sources. The system can deal faces with different poses, rotation, occluded parts of faces, faces with emotions and painted faces. 2D and 3D face data fusion make possible to achieve 98,5% correct recognition rate.

Figure 7: 2D+3D face recognition system.

7. REFERENCES

[1] Blanz, V., Grother, P., Phillips, J., and Vetter, T. (2005). *Face recognition based on frontal views generated from non-frontal images. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 454–461.*

[2] Chua, C. and Jarvis, R. (1997). *Point signatures - a new representation for 3D object recognition. International Journal of Computer Vision, 25(1): pp. 63–85.*

[3] Gordon, G. (1992). *Face recognition based on depth and curvature features. IEEE Computer Vision and Pattern Recognition (CVPR), pp. 808–810.*

[4] Gross, R., Matthews, I., and Baker, S. (2002). *Eigen light-fields and face recognition across pose. International Conference on Automatic Face and Gesture Recognition.*

[5] Hesher, C., Srivastava, A., and Erlebacher, G. (2003). *A novel technique for face recognition using range imaging. International Symposium on Signal Processing and Its Applications, pp. 201–204.*

[6] Kim, T. and Kittler, J. (2005). *Locally linear discriminant analysis for multimodally distributed classes for face recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence, 27(3): pp. 318–327.*

[7] Lanitis, A., Taylor, C., and Cootes, T. (1995). *Automatic face identification system using flexible appearance models. Image and Vision Computing, 13(5): pp. 393–401.*

[8] Lee, Y., Song, H., Yang, U., Shin, H., and Sohn, K. (2005). *Local feature based 3D face recognition. International Conference on Audio- and Video-based Biometric Person Authentication, pp. 909–918.*

[9] Li, Y., Gong, S., and Lidell,H. (2000). *Support vector regression and classification based multiview face detection and recognition. International Conference on Face and Gesture Recognition, pp. 300–305.*

[10] Moreno, A., Sanchez, A., Velez, J., and Diaz, F. (2003). *Face recognition using 3D surface extracted descriptors. In Irish Machine Vision and Image Processing Conference.*

[11] Papatheodorou, T. and Rueckert, D. (2007). *3D face recognition, book Face recognition, Vienna, Austria, 2007, pp. 417-423.*

[12] Prince, S. and Elder, J. (2006). *Tied factor analysis for face recognition across large pose changes. British Machine Vision Conference.*

About the author

Svetlana Korobkova is a senior researcher at Vocord. Her contact e-mail is svetlana.korobkova@vocord.ru.

Archil Tsiskaridze is a researcher at Vocord. His contact e-mail is svetlana.korobkova@vocord.ru.

System of Audio-Visual Streams Recording and Synchronization for the Smart Meeting Room

Ronzhin Al.L., Karpov A.A.
Speech and Multimodal Interfaces Laboratory
Institution of the Russian Academy of Sciences St.Petersburg Institute for Informatics and Automation of RAS (SPIIRAS),
St. Petersburg, Russia
{ronzhinal, karpov}@iias.spb.su

Abstract

The problem of automatic detection and recording of active speaker talks among more than thirty participants located in the medium meeting room is not solved completely. In the developed smart meeting room techniques of video tracking and audio source localization are implemented for recording AVI files of speaker messages. Video processing of streams from five cameras serves for registration participants in fixed chair positions, tracking main speaker and recording view to the audience. The experiments showed that the developed audiovisual system captured messages of all the speakers. The detection error of beginning and ending of speech message had acceptable rate.

Keywords: *Video surveillance, Computer vision, Speaker detection, Smart meeting room.*

1. INTRODUCTION

Choosing current active speaker and recording his/her activity during an event are the main tasks for meeting recording and supporting teleconference systems [1, 2]. Panoramic and personal cameras could be employed for simultaneous recording of all participants. Such approach is suitable for small events, where all the participants are located at one table. Increase in participant number leads to space extension, which should to be processed, as well as cost of recording technical equipment.

Several approaches of information presenting such as oral statement, presentation, whiteboard notes, demonstration of audiovisual data may be used for support educational events such as teleconference, lecture, workshop, meeting, which are carried out in rooms with state of art multimedia equipment. General lecture scenario implies that students have to write most fully information of lecture talk. However, students usually may write only short notes and main words. So in order to provide participants with meetings materials the audiovisual system for meetings recording and processing was developed [3]. The first prototype of such system was developed in the Cornell University [4], which consists of two cameras for lecture talk and presentation slides recording. Another system is FLYSPEC [5], which was developed at 2002 year by FX Palo Alto Labs and its intended for supporting teleconference. Two video sensors were implemented in this system: high resolution camera and Pan/Tilt/Zoom (PTZ) camera. The system may automatically control second camera or by analysis of participants requests.

Automatic analysis of audio and video data recorded during a meeting is not trivial task, since it is necessary to track a lot of participants, which randomly change position of their body, head and gaze. In order to detect participant activity several approaches based on using panoramic cameras, intelligent PTZ cameras, distributed camera systems were employed [5, 6]. Besides video monitoring, motion sensors and microphone arrays could be implemented for detecting participant's location and selection of the current speaker [7]. The sound source localization technique is effective for small lections or conference rooms. Personal microphones for all the participants or system of microphone arrays, which set on several walls of smart room, are employed for audio re-cording in medium rooms [8, 9].

Description of the technological framework of the smart meeting room is presented in Section 2. An algorithm describing the interaction of sound source localization and video tracking modules during speaker detection, recording audio video files and synchronization all the data streams is presented in Section 3. The results of the experiment are discussed in Section 4.

2. TECHNOLOGICAL FRAMEWORK OF THE SMART MEETING ROOM

A premises of 72 square meters located inside the institute building was supplied for intelligent meeting room in 2008 at the financial support of the Russian Foundation for Basic Research. Monitoring of the room is performed by 15 video cameras mounted on the walls, ceiling and tables and provides tracking of moving objects, face detection and other functions. Three T-shape 4 channel microphone arrays mounted on the different walls serve for localization of sound sources, far-field recording and following speech processing. Besides video recording the personal web cameras mounted on the tables have internal microphones and are used for recording speech of each meeting participant.

A wide touchscreen plasma panel and multimedia projector (projection screen) are located one under another in the left side of the room and provide output of multimodal information. Operated electro gears are connected to the projection screen and curtain rail. The curtains are made from special light-tight cloth in order to suppress the outside influence on the illumination changing in the room. The processing of recorded audio-visual data, control the multimedia and electro mechanic equipment are performed by six four-cored computers, two multichannel audio boards Presonus FirePod, as well as some devices providing cable and wireless network. The referred service equipment is mounted in a rack and located on the adjacent room from the meeting one. Thus, users inside the meeting room could see only appliances for input/output information, but other devices and computational resources are invisible. To provide service and the same time be hidden for a user is one of the main features of ambient intelligence.

The developed smart room is intended for holding small and medium events with up to forty-two participants. Also there is the

ability to support of distributed events with connection of remote participants. Two complexes of devices are used for tracking participants and recording speakers: (1) personal web-cameras serve for observation of participants, which are located at the conference table; (2) three microphone arrays with T-shape configuration and five video cameras of three types are used for audio localization and video capturing of other participants, which sit in rows of chairs in other part of the room. Description of the first complex could be found in [10]. Status of multimedia devices and participant activity are analyzed for whole mapping current situation in the room.

Location and description of places of seats, multimedia equipment (TV set, Projector), five AXIS Internet-cameras (two PTZ-cameras, two wireless cameras, camera with wide angle lens, installed on the ceiling in the center of the room),10 personal webcams Logitech AF Sphere (on the conference table) could be found in [11]. Three microphone arrays located in the center of the left and right walls and over the sensor plasma (touch screen) serve for sound sources localization and recording phrases of participants. Each array of microphones has T-shaped configuration [12]. The applied software for processing multi-channel audio streams was firstly used at development of multimodal information kiosk [13]. The conference table with installed personal web-cameras for placement of participants of small meetings (round tables) up to 10 people is located on the left side of the hall. The right side of the hall contains rows of seats, which can accommodate up to 32 participants, tracking of which is implemented by the distributed system of cameras and microphone arrays.

3. ALGORITHM OF SPEAKER RECORDING

The algorithm of camera pointing to the current active speaker in the zone of chairs and following recording of his/her speech should be considered in detail. Sound source localization and object tracking by ceiling camera are implemented here. Both modules work together during the all time of the event in the smart room.

The object detection module carries out the search and following tracking of the participants inside in the room. Also the module marks the occupied chairs [14, 15], which will be used as hypotheses for speaker position. The scheme of the algorithm of the speaker detection and recording is shown in Figure 1.

The appearance of a sound signal in the chair zone launches the voice activity detection process and makes query (the event E_1) to the object detection module in order to check the presence of a participant in the chair, which is closest to the determined coordinates of the sound source. If the chair is marked as occupied then corresponding response is transmitted to the module of speech recording as well as the camera serving this zone is being pointed to the selected chair with the current active participant.

To avoid missing of the speech, the decision about useful sound segment is made every 20 ms. Short remarks and noise with duration less half of second are discarded in order to exclude false speaker detection. The frame rate of the camera, which captures the speaker in the chair zone, achieves thirty frames per second. However, the camera pointing takes up to couple seconds owing to a stabilisation period after mechanical setting of direction angles and zooming of the objective lens. So the recording of images to the *bmp* files is started after the camera pointing is accomplished.

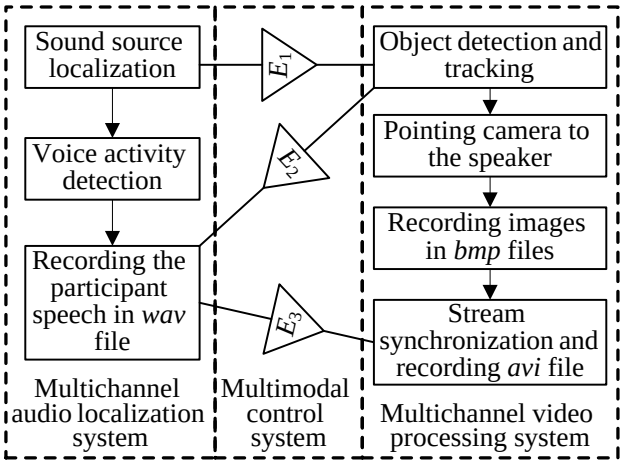


Figure 1: Algorithm of speaker detection and recording.

At the same time, in the multichannel audio localization system the *wav* file with participant speech is recorded in case of speech boundaries detection and positive response from the object detection module (the event E_2) about presence of a participant in the selected chair. After recording *wav* file the corresponding notification (the event E_3) with path to the file transmitted to the video processing system. Then the system goes to the sound source detection and localization stage again.

Participant could make some pauses during the talk that leads to the detection ending boundary of the phrase and recording the separate wav file. As a result during the talks the system could write several audio files, which belongs to the same participant (more precisely put, belongs to chair coordinates assigned to this speaker). Name of audio file includes information about chair number, from which speech signal was recorded.

The creation of *avi* file is started after silence of the current speaker during five seconds or, that more frequent case, detection of an active speaker on other chair, conference table or in the presentation area. The main difficulty of recording *avi* file consists in synchronization of the sets of audio and image files. Frame rate of the camera is not constant owing to various download of the computer, constraints of network and camera hardware. So, the synchronization process is based on analysis of duration and creation time of the *wav* files. Figure 2 shows scheme of synchronization algorithm. All audio files are processed in consecutive order. At first, system detects time interval, in which audio and video files were recorded.

Then to get normal FPS (25 frame per second) starts image duplication in determinate time intervals. Edition of *avi* file is carried out during processing of packets of *bmp* files recorded in interval time approximately equal one second. A data packet structure consists of its duration, first frame number and frames total amount in packet. An analysis of current structure need for elimination of asynchrony appears at recording of audio and video streams, because it allows calculating additional frames total amount. After processing of all *bmp* and *wav* files selected and duplicated images add to an *avi* file, then *wav* files add to it.

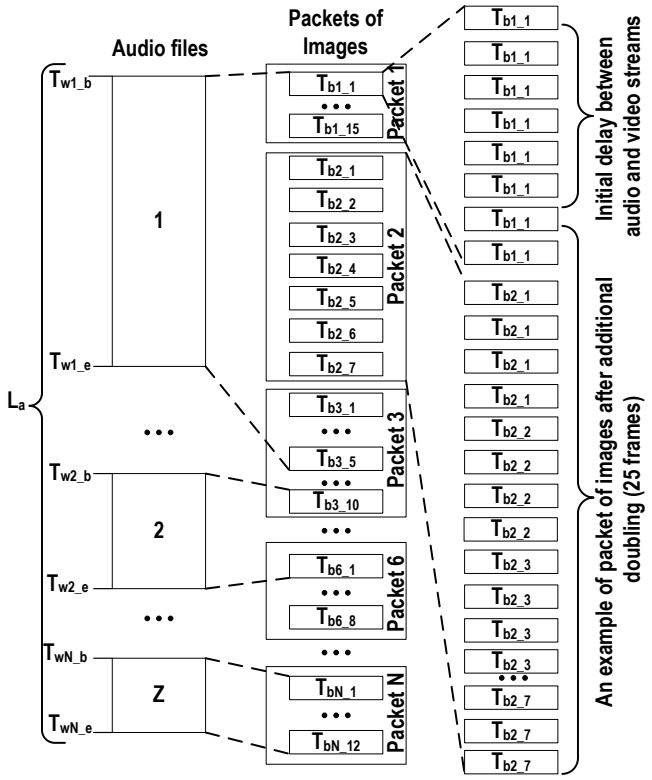


Figure 2: An example of audio and video streams synchronization for recording *avi* file

The described algorithms serve for recording remarks of participants sitting thirty-two chairs of the right side of the smart room. At the end of the meeting the set of *avi* files with all the remarks are recorded. Analogical algorithm is used for tracking main speaker in the presentation area. The description of the approach, which is used to capture activities of participants sitting at the conference table, as well as the logical-temporal model for compilation multimedia content for remote participants of the meeting and support teleconference, is presented in [10].

4. EXPERIMENTS

For an estimation algorithm of detecting and recording active participant speech four criteria were used.

(1) Initial delay between audio and video streams L_{b_d} calculates as difference between first *wav* file creation time T_{w1_b} and *bmp* file T_{b1_1} creation time, which corresponded with a T_{w1_b} time: $L_{b\_d} = | T_{w1\_b} - T_{b1\_1} |$;

(2) A length of recorded *avi* file L_a calculates as summing up of *wav* files length for current speech: $L_a = \sum_{i=1}^{N} T_{wi\_e} - T_{wi\_b}$

(3) Duplicate frames total amount calculates as summing up of L_{b_d} and all duplicated frames in all image packets $P_i : N_{f\_d} = L_{b\_d} + \sum_{i=1}^{N} P_i$; $P_i = P_{AF\_i} + P_{RF\_i}$;

$$P_{AF\_i} = \frac{(P_{FN\_i} - P_{F\_i})}{P_{F\_i}}; \qquad P_{RF\_i} = (P_{FN\_i} - P_{F\_i}) \% P_{F\_i};$$

$$P_{FN\_i} = F_D * \frac{T_{bN\_i} - T_{b1\_i}}{1000};$$

(4) A mean FPS F_a in a video buffer calculates as summing up of image packets size divide on the packets total amount: $F_a = \sum_{i=1}^{i<N} F_i$.

The estimation of the algorithm of detecting and recording active participant speech was carry out in the SPIIRAS smart room. Main attention was paid on detecting active participants in the zone of chairs. Each tester performed the following scenario: (1) take a sit in the room; (2) wait visual confirmation on a smart board about registration of participant in the chair; (3) pronounce the digit sequence from one to ten; (4) move to another chair.

During the experiments were recorded 36 avi files in a discussion work mode. After manual checking was detected that 89% are files with speaker's speech and 11% false files with noises. Such noises are carrying out in process of tester standing up from a chair, because in such moment chair's mechanical details carry out high noise. Also mistakes in detecting sitting participants influence on appearance of false files. Table 1 shows results of estimation files with speaker's speech.

L_{b_d}, ms			L_a, ms			N_{f_d}, frames		
Min	**Max**	**Mean**	**Min**	**Max**	**Mean**	**Min**	**Max**	**Mean**
80	2440	724	5312	6432	5608	32	104	59

Table 1: The estimation of algorithm of detecting and recording active participant speech work

A result of experiments shows, that *avi* file in mean consists of 137 frames, 59 of it are duplicated frames, as well as has length 5 seconds. Calculated mean FPS in video buffer is 24 frames per second, this is due to the fact that rounding of values at calculating a required total amount of additional frames in image packets. The total amount of duplicated frames includes initial delay between audio and video streams. Also such total amount of duplicated frames is carry out with changing camera FPS as a result of noises in a network devices as well as limited writing speed of storage devices. Analyses of received data shows that *avi* files form by system include all speeches and a small percent of false records.

5. CONCLUSION

The audiovisual monitoring system of participants was developed for automation of recording events in the smart room. It consists of the four main modules, which realize multichannel audio and video signal processing for participants localization, detection of speakers and recording them. The proposed system allows us to automate control of audio and video hardware as well as other devices installed in the smart room by distant speech recognition of participant command. The verification of the system was accomplished on the functional level and also the estimations of

detection quality of participants, and camera pointing on speaker and speaker detection error were calculated.

6. ACKNOWLEDGMENT

This work is supported by Russian Foundation for Basic Research (projects 10-08-00199-a 11-08-01016-a).

7. REFERENCES

[1] Busso C., Hernanz S., Chi-Wei Chu, Soon-il Kwon, Sung Lee, Georgiou P.G., Cohen I., and Narayanan S. *Proc. IEEE International Conference on Multimedia and Expo: Smart room: Participant and speaker localization and identification. Philadelphia, USA, March 18-23. 2005, pp. 1117–1120.*

[2] Zhang C., Yin P., Rui Y., Cutler R., Viola P., Sun X., Pinto N., and Zhang Z. *IEEE Transactions on Multimedia: Boosting-Based Multimodal Speaker Detection for Distributed Meeting Videos. Vol.10, No.8, 2008, pp. 1541-1552.*

[3] Lampi F. *Automatic Lecture Recording. Dissertation. The University of Mannheim, Germany. 2010.*

[4] Mukhopadhyay, S., Smith, B. *Passive capture and Structuring of Lectures, Proceedings of ACM Multimedia 1999, Orlando, FL, USA, Vol.: 1, pp. 477-487.*

[5] Liu, Q., Kimber, D., Foote, J., Wilcox, L., Boreczky, J. *FLYSPEC: a multi-user video camera system with hybrid human and automatic control, Proceedings of ACM Multimedia 2002, Juan-les-Pins, France, pp. 484-492.*

[6] Erol B. and Li Y. *in Proc. ICASSP: An overview of technologies for e-meeting and e-lecture. 2005, pp. 6-12.*

[7] Kellermann W. *SPECOM 2009: Towards Natural Acoustic Interfaces for Automatic Speech Recognition. St. Petersburg, June 25-29, 2009, pp. 8-17.*

[8] Brutti A., Omologo M. and Svaizer P. *Hands-Free Speech Communication and Microphone Arrays (HSCMA): Comparison be-tween different sound source localization techniques based on a real data collection. Trento, Italy, May 2008.*

[9] Waibel A., Stiefelhagen R. *Computers in the human interaction loop. Berlin: Springer, 2009, 374 p.*

[10] Ronzhin, A., Budkov, V., and Karpov, A., *Multichannel System of Audio-Visual Support of Remote Mobile Participant at E-Meeting / Springer-Verlag Berlin Heidelberg, S. Balandin et al. (Eds.): NEW2AN/ruSMART 2010, LNCS 6294, 2010, pp. 62–71.*

[11] Al.L. Ronzhin, M.V. Prischepa, Budkov V. Yu., A.A. Karpov, A.L. Ronzhin. *Distributed System of Video Monitoring for the Smart Space. In. Proc. GraphiCon'2010. Saint-Petersburg, Russia, 2010 pp. 207-214. (in Rus.).*

[12] Maurizio O., Piergiorgio S., Alessio B., Luca C. *Machine Learning for Multimodal Interaction: Speaker Localization in CHIL Lectures: Evaluation Criteria and Results. Berlin: Springer, 2006, pp. 476–487.*

[13] A. Ronzhin, A. Karpov, I. Kipyatkova, M. Zelezny. *TSD 2010: Client and Speech Detection System for Intelligent Infokiosk. Springer-Verlag Berlin Heidelberg, Petr Sojka et al. (Eds.): TSD 2010, LNAI 6231, 2010, pp. 560–567.*

[14] Schiele B., Crowley J. L. *European Conference on Computer Vision: Object recognition using multidimensional receptive field histograms. Vol. I, pp. 610–619, April 1996.*

[15] Viola P., Jones M., Snow D. *In Proc.of IEEE ICCV: Detecting pedestrians using patterns of motion and appearance. Pages II: 734–741, 2003.*

About the author

Ronzhin Alexander Leonidovich – PhD student of Speech and Multimodal Interfaces Laboratory of the Institution of the Russian Academy of Sciences St.Petersburg Institute for Informatics and Automation of RAS (SPIIRAS), ronzhinal@iias.spb.su

Karpov Alexey Anatol'evich – PhD, senior researcher of Speech and Multimodal Interfaces Laboratory of the Institution of the Russian Academy of Sciences St.Petersburg Institute for Informatics and Automation of RAS (SPIIRAS), karpov@iias.spb.su

Calibration maintenance for a four camera acquisition system

Oleg Stepanenko
Vocord Company
Moscow, Russia
oleg.stepanenko@vocord.ru

Abstract

This paper discusses the method of calibration maintenance for four camera acquisition system (the mentioned system will be referenced as the 3D system in this paper). 3D system creates 3D models of human faces. The relative orientation between two vertical stereo pairs could slowly change in time because of a vibration. Thus camera calibration needs to be maintained.

We show how 3D system can maintain calibration without being stopped. The new calibration parameters are being continuously recalculated from new multiple dynamic scene images and previous calibration.

Keywords: *Camera calibration, Stereo, Essential parameters estimation.*

1. INTRODUCTION

3D system consists of four cameras that have to be calibrated (Fig.1). The cameras are synchronized.

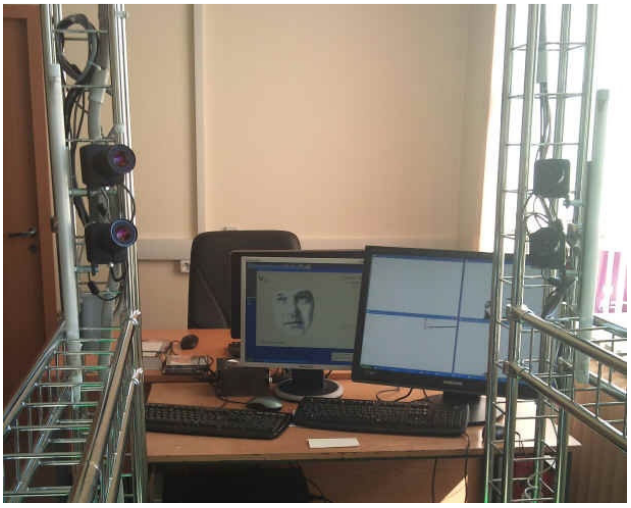


Figure 1: Four camera 3D acquisition system.

Four cameras are coupled in two vertical stereo pairs. Two cameras in each pair are mounted at the common stable basis and have constant orientation in respect to each other.

We had observed that 3D system with two stereo pair is much superior in respect to one that has only one stereo pair. Each stereo pair recovers only its half of the face. Taken from left and right sides the face is fully recovered under wide angle of rotation.

Coupling left and right vertical stereo together however puts another problem: the relative orientation between two vertical stereo pairs may have small alteration because of the vibration of constructive elements which position stereo pairs in space. Thus 3D models of human faces will have alteration in time. In Fig.2 we present a 3D model of human face before the calibration has been changed. In Fig.3 we present a 3D model of same human face after the calibration has been slightly changed. In Fig.3 the face is distorted obviously: nose becomes shorter.

The classical camera calibration [4, 5] is performed by capturing a reference object with a known Euclidean structure (for example, chessboard pattern). This approach can be reasonably used on stages when 3D system is stopped. On those stages 3D system cannot create 3D models of human faces. But 3D system cannot be stopped every time we want to find out if calibration has changed considerably or not. There are also severe conditions in which human traffic may exist all day and night, thus the face recognition system has to operate without break (i.e. at airports).

Figure 2: 3D model of human face before camera calibration has been changed

Figure 2: 3D model of human face after camera calibration has been changed (the shape of nose gets a distortion)

A question arises: how new camera calibration can be obtained from dynamic scene images and previous calibration?

We use the following features of the mentioned problem: cameras in pairs keep relative orientation, the observed scene is dynamic.

2. NOTATION AND PROBLEM DEFINITION

2.1 Notation

The pinhole camera model is used here. A 3D point has coordinates $M = [x, y, z]^T$ in a world coordinate system. The retinal image coordinates of a 3D point are $m = [u, v]^T$. World coordinates and retinal image coordinates are related by

$$s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \mathbf{P} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \text{ or } s\tilde{m} = \mathbf{P}\tilde{M},$$

where s is a scale coefficient, $\mathbf{P}$ is a 3×4 perspective projection matrix. We use tilde sign to denote the augmented vector $\tilde{x}$ (adding 1 as its last element) of a vector x.

Matrix P can be written as $P = \mathbf{A}[\mathbf{R}\ \mathbf{t}]$ with $= \begin{bmatrix} \alpha_u & c & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix}$, where $\mathbf{A}$ is a 3×3 matrix of intrinsic parameters, and $(\mathbf{R}, \mathbf{t})$ is a displacement (rotation and translation) from the world coordinate system to the camera coordinate system. $\mathbf{R}$ is a 3×3 rotation matrix. $\mathbf{t} = [t_x, t_y, t_z]^T$ is a translation vector. Translation may be presented in a form of skew-symmetric matrix: $[\mathbf{t}]_\times = \begin{bmatrix} 0 & -t_z & t_y \\ t_z & 0 & t_x \\ -t_y & t_x & 0 \end{bmatrix}$.

One of the cameras (top camera from one pair) is chosen as reference camera and has $\mathbf{R} = \mathbf{I}$ and $\mathbf{t} = \mathbf{0}$, where $\mathbf{I}$ is 3×3 identity matrix and $\mathbf{0}$ is zero 3×1 vector . The parameters in $\mathbf{A}$ are obtained through calibration procedure [4, 5].

2.2 Problem definition

We consider multiple perspective images of a dynamic scene, and have to determine the relation between the multiple images and the camera pair displacement. This arises from a situation described further. Four cameras take an image sequence. We assume the images are projections of a moving human (to be specific, top part of human body), the cameras in pairs are calibrated (i.e. their intrinsic parameters are known), cameras in pairs have a known displacement (orientation and translation).

3. ALGORITHM

3.1 Main steps

The main steps of the algorithm are:

Step 1: Establish two sets of pixel correspondences between camera pairs at multiple points of time in a dynamic scene. One set is established between two images that are taken by top cameras. The other set is established between two images that are taken by bottom cameras (Fig.4). Select a set that is bigger.

Step 2: Establish pixel correspondences between two images that are taken by top and bottom cameras from each pairs at multiple points of time in a dynamic scene (Fig.5).

Step 3: Triangulate 3D point sets from pixel correspondences that have been established on the step 1 and 2. Thus we have two sets of 3D points that are triangulated from one pair and the other pair.

Step 4: Match 3D points from sets that are mentioned above in step 3. Form set of pixel correspondences from 3D matched points.

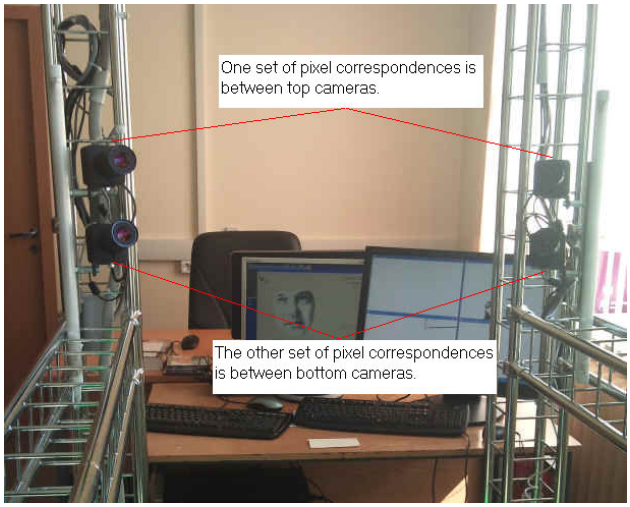


Figure 4: Two sets of pixel correspondences that are establishing at the step 1

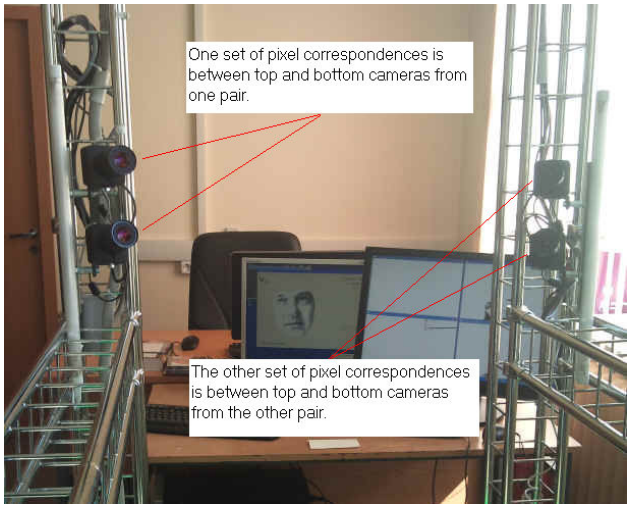


Figure 5: Two sets of pixel correspondences that are establishing at the step 2

Step 5: Estimate the essential parameters with 8-point algorithm [3]. The obtained matrix is denoted by $\boldsymbol{E} = [\mathbf{t}]_\times \mathbf{R}$. Recover the displacement parameters $\mathbf{t}$ and $\mathbf{R}$ from **E.** This displacement takes place between top cameras from different pairs. One of those top cameras is the reference camera.

Step 6: Refine the displacement parameters.

3.2 Detection of false matches on step 4

In order to detect false matches on step 4 we have to establish correspondences between two sets of 3D points. The criterion being minimized is the 3D Euclidean distance between points. Suppose we have i^{th} and j^{th} 3D points from the sets of points that has been triangulated from top and bottom cameras of one pair (coordinates are M_i) and the other pair (coordinates are M_j). The Euclidean distance from point M_i to point M_j is $d(M_i, M_j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2}$.

After matching procedure has ended we have n 3D point correspondences: $\{(M_i, M_j)\}$. For each of the mentioned

correspondences we can determine $\{(m_i, m_j)\}$, where m_i are pixel coordinates of M_i on the image of top camera from one pair, m_j are coordinates of M_j on the image of top camera from the other pair.

Thus we have two sets of matched 3D points and matched pixel correspondences that are taken by two cameras from the different pairs.

3.3 Estimate the essential parameters with 8-point algorithm on step 5

Due to the presence of incorrect correspondences (outliers), essential parameters estimators must be robust. The Random Sample Consensus – RANSAC methods [1] have become the methods of choice for outlier removal in essential parameters estimation [2]. We use RANSAC-like techniques on step 5. We start by selecting (at random) a subset of k correspondences, which is then used to compute the essential parameters estimation. The cost function of the full set of correspondences is then computed. The cost function expresses numbers of inliers that are within a certain neighborhood form their predicted epipolar lines. The random selection process is repeated S times, and the sample set with largest number of inliers is kept as the final solution.

Assuming that the set of correspondences may contain up to a portion ε of outliers, the probability that one of S samples is good is given by $P = 1 - (1 - (1 - \varepsilon)^k)^S$. In our implementation, we determine $\varepsilon = 25\%$, $k = 35$, $P = 0.999$, thus S=200000. The algorithm can be speed up considerably by means of CUDA technology.

The standard 8-point algorithm [3] is used to estimate the essential matrix.

3.4 Refine the displacement parameters on the step 6

The nonlinear minimization on step 6 is done with the Levenberg-Marquardt algorithm. The criterion being minimized is the sum of squared reprojection errors. The Levenberg-Marquardt algorithm is one of the most popular methods for iterative minimization, when cost function to be minimized is of this type [2]. The optimization is carried out for all displacement parameters.

In Fig.6 we present a 3D model of the same human face after calibration has been recovered. In Fig.6 a shape of the face is recovered: nose gets former shape.

4. CONCLUSION

In this paper we have used a general scheme of displacement estimation from multiple calibrated images [2, 3, 6] in the field of a four camera acquisition system.

Steps 1, 2, 3, 4 of our algorithm were developed specifically for our field of investigation and we haven't found any references to such methods implementation in literature.

3D system has to be accurately calibrated. But 3D system has not to be stopped every time when we want to do calibration procedure with reference object with a known Euclidean structure.

Due to the scene being dynamic we have matched points from cameras that fill the observed scene fairly uniformly.

Figure 6: 3D model of human face after camera calibration has been recovered (nose gets former shape)

The new camera calibration can be obtained from dynamic scene images and previous calibration. Our experimental results suggest that our method can be applied when displacement alteration is rather small. Experience has shown that it was enough to run our algorithm once at the middle of the day. Nevertheless classical camera calibration must be done as soon as 3D system may have a break in its work (for example, at the end of the day or before a new day).

5. REFERENCES

[1] *M. Fischler and R. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. CACM, 24(6):381–395, June 1981.*

[2] *R. Hartley and A. Zisserman Multiple View Geometry in Computer Vision. Cambridge University Press, Cambridge, UK, 2000.*

[3] Y. Ma, S. Soatto, J. Kosecka and Shankar Sastry. *An Invitation to 3D Vision: From Images to Models. Springer Verlag, December 2003.*

[4] R. Y. Tsai. *A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf cameras and lenses. IEEE Journal of Robotics and Automation, 3(4): 323-344, Aug. 1987.*

[5] Z. Zhang. *A flexible new technique for camera calibration.* IEEE Transactions *on Pattern Analysis and Machine Intelligence*, Vol.22, No.11, pages 1330-1334, 2000

[6] Z. Zhang. *Motion and Structure From Two Perspective Views: From Essential Parameters to Euclidean Motion Via Fundamental Matrix. Journal of the Optical Society of America* , Vol.14, no.11, pages 2938-2950, 1997

About the author

Oleg Stepanenko (Ph.D, Associate Professor) is a scientist at Vocord Company, Department of Advanced Developing. His contact email is oleg.stepanenko@vocord.ru.

Tracking CSOs Using PHD Filter from Image Observations

Yang Xu, Hui Xu, Wei An, Liangkui Lin

(School of Electronic Science and Engineering, National University of Defense Technology, Changsha, P. R. China)

Abstract

Aiming at tracking Closely-Spaced Objects (CSOs) from image observations, a new method using the Probability Hypothesis Density (PHD) filter is proposed. To circumvent the unresolved measurements problem, a detection process is used firstly to extract the connected sets of object pixels that likely correspond to the unresolved targets of interest. Then the representative measurements are constructed to cast the CSOs tracking in the framework of PHD filter. The newly resolved targets are naturally modeled as spawned targets thus can be detected and estimated immediately by setting appropriate target spawn intensity. Gaussian mixture (GM) implementation is used for this filter, and simulations are carried out to verify the effectiveness of the proposed method.

Keywords: *Closely Space Objects (CSOs), optical sensor, pix-cluster, GM-PHD filter.*

1. INTRODUCTION

A major challenge in multi-target tracking by optical sensors is that the targets are often closely spaced during certain period. The term "closely spaced objects" implies that two or more objects are close enough to one another on the sensor focal plane to create an unresolved measurement where the number, the focal plane coordinates, and the radiant intensities of the objects are not immediately apparent[4]. As a result, the tracking problem becomes quite difficult, because it brings together all issues of multi-target tracking: miss detection, target birth, spawn and termination, unresolved targets and an environment with a high false alarm rate. Furthermore, because of the possibly physical intersection of the targets, temporary varying sensor-to-target geometry and sensor measurement noise, most of existing association-based approaches, such as nearest neighbor, joint probabilistic data association (JPDA), multiple hypothesis tracking (MHT) are not appropriate due to the impractical or error-prone measurement-to-track association[5].

Recently, the PHD filter proposed by Mahler is a computation tractable approximation to the optimal multi-object Bayes filter[6]. PHD filter gives both targets number and individual target state estimates while avoiding the combinatorial problem that arises from data association. However, the PHD assumes each measurement is from either single target or clutter. To track CSOs using PHD filter, in this paper, we present a method which tracks the unresolved CSOs on the focal plane as conventional point targets, then detects and estimates individual targets states just after they are resolved. Simulations are carried out to verify effectiveness of the method.

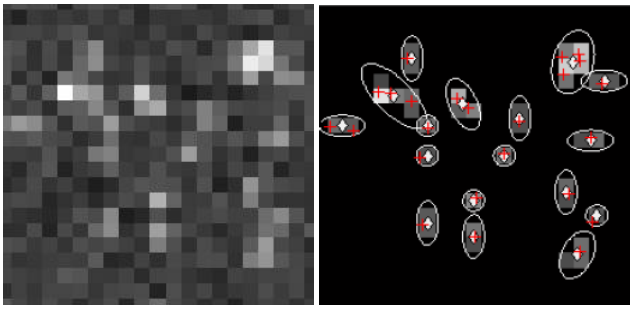
2. MEASUREMENT MODEL

2.1 Image preprocessing

Due to the close target locations and the varying sensor-to-target geometry, measured signals from CSOs may overlap on the sensor focal plane, forming a connected pixel-cluster which represents multiple targets. Consequently, we adopt a detection process including thresholding and cluster identification to the original image observations. The purpose of detection is to identify connected sets of object pixels that likely correspond to targets of interest and to remove background pixels that are likely caused by noise, false targets, clutter, etc. An adaptive thresholding approach is used to extract the objects from the background[2]. After the detection step, the image consists of object pixels and background pixels. The Hoshen-Kopelman (HK) Cluster identification algorithm[3] is adopted with 4 neighbor criteria to identify connected sets of object pixels, i.e., a pixel-cluster, that likely correspond to the return from the same target class. Fig. 1 shows the one original image before and after the processing. We see that most of the background pixels are removed and measurements of CSOs are represented as pixel-clusters.

2.2 Representative measurements

To support tracking, given a pixel cluster extract form image observation at time k, we compute a representative measurement $\tilde{\boldsymbol{z}}_k$ with a covariance estimate $\boldsymbol{R}_{\tilde{\boldsymbol{z}}_k}$. Assume that the pixel-cluster is described in (x, y) focal plane row, column coordinates. Let $P = \{(x_1, y_1), \ldots (x_n, y_n)\}$ denote the set containing the n pixel coordinates of this pixel-cluster. The signal strength at a coordinate (x, y) is given by $s(x, y)$. As covariance estimate, we use a second moment, intensity-weighted estimate:



(a) Original image (b) Image after processing

Figure 1: Image observations from optical sensor, "+" denote individual target positions, "◊" denote centroids of pixel-clusters.

$$\tilde{\boldsymbol{z}}_k := (z_x, z_y)^T = \frac{1}{s_P}(s_x, s_y)^T, \quad \boldsymbol{R}_{\tilde{\boldsymbol{z}}_k} := \begin{bmatrix} \sigma_x^2 & \sigma_{xy}^2 \\ \sigma_{yx}^2 & \sigma_y^2 \end{bmatrix} \tag{1}$$

where

$$s_P = \sum_{(x,y)\in P} s(x,y), s_x = \sum_{(x,y)\in P} s(x,y)x, s_y = \sum_{(x,y)\in P} s(x,y)y, \tag{2}$$

and

$$\sigma_x^2 = \frac{1}{s_P}\sum_{(x,y)\in P} s(x,y)(x-z_x),\ \sigma_y^2 = \frac{1}{s_P}\sum_{(x,y)\in P} s(x,y)(y-z_y),$$
$$\sigma_{xy}^2 = \frac{1}{s_P}\sum_{(x,y)\in P} s(x,y)(x-z_x)(y-z_y). \tag{3}$$

For the single-pixel pixel-cluster case, $\tilde{\boldsymbol{z}}_k = (x-0.5, y-0.5)^T$, $\boldsymbol{R}_{\tilde{\boldsymbol{z}}_k} = diag\left(\sigma_{sp}^2, \sigma_{sp}^2\right)$, where we define the σ_{sp}^2 such that 3 sigma circumscribes a single pixel, i.e., 0.7071 pixel units, $\sigma_{sp}^2 = 2(0.5)^2/9$.

3. PHD FILTERING

3.1 PHD filter

The PHD filter is a computation tractable approximation to the optimal multi-object Bayes filter based on RFS[6]. It is a recursion propagating the 1st moment, called the intensity function or PHD, associated with the multi-target posterior.

Let $D_{k|k}$ and $D_{k|k-1}$ denote the respective intensities associated with the multi-target posterior density and the multi-target predicted density, the PHD recursion is given as follows[7]:

$$D_{k|k-1}(\boldsymbol{x}) = \gamma_k(\boldsymbol{x}) + \int (p_{S,k}(\boldsymbol{x}')\, f_{k|k-1}(\boldsymbol{x}\mid\boldsymbol{x}') + b_{k|k-1}(\boldsymbol{x}\mid\boldsymbol{x}'))D_{k-1|k-1}(\boldsymbol{x}')d\boldsymbol{x}' \tag{4}$$

$$D_{k|k}(\boldsymbol{x}) = \left(1 - P_{D,k}(\boldsymbol{x})\right)\cdot D_{k|k-1}(\boldsymbol{x}) + \sum_{\boldsymbol{z}\in Z_k} \frac{P_{D,k}(\boldsymbol{x})\, g_k(\boldsymbol{z}\mid\boldsymbol{x}) D_{k|k-1}(\boldsymbol{x})}{\kappa_k(\boldsymbol{z}) + \int P_{D,k}(\xi)\, g_k(\boldsymbol{z}\mid\xi) D_{k|k-1}(\xi)d\xi} \tag{5}$$

where $\gamma_k(\boldsymbol{x})$, $b_{k|k-1}(\boldsymbol{x}\mid\boldsymbol{x}')$, $\kappa_k(\boldsymbol{z})$ are the intensity for target birth, target spawn and clutter respectively, $p_{S,k}(\boldsymbol{x}')$ is the probability of target survival, $f_{k|k-1}(\boldsymbol{x}\mid\boldsymbol{x}')$ is the transition density, $g_k(\boldsymbol{z}\mid\boldsymbol{x})$ is the likelihood and the $D_{k-1|k-1}(\boldsymbol{x}')$ is the PHD at time $k-1$.

The integration of PHD over the target state space $\int D_{k|k}(\boldsymbol{x})d\boldsymbol{x}$ gives the targets number estimate, and the local maximum of the intensity can be used to generate the estimates of target states.

3.2 Implementation

To cast the CSOs tracking with unresolved measurements in the framework of PHD filter, we present an alternative method as following two steps. Firstly, we model the collection of unresolved CSOs (also called target cluster in this paper) states and the collection of their representative measurements as random finite sets (RFS) similarly to the single target case. Let $\tilde{\boldsymbol{x}}_k$ denote the state vector containing position and velocity of each target cluster on focal plane at time k, and the measurement $\tilde{\boldsymbol{z}}$ is given as in Section 2.2. Then, the RFSs for the states and measurements are given as

$$\tilde{X}_k = \left\{\tilde{\boldsymbol{x}}_{k,1}, \cdots \tilde{\boldsymbol{x}}_{k,N_k}\right\}$$
$$\tilde{Z}_k = \left\{\tilde{\boldsymbol{z}}_{k,1}, \cdots \tilde{\boldsymbol{z}}_{k,M_k}\right\} \tag{6}$$

where N_k and M_k are the number of elements of each RFS.

Secondly, we model the resolved targets from the target cluster as spawn targets which are naturally adapted by PHD filter. Eq. (4) reveals that the resolving process of CSOs can be represented as targets spawn in the recursion of PHD. Thus, by setting appropriate target spawn intensity, the newly resolved targets can be detected immediately and the number and states of which are jointly estimated.

Gaussian mixture implementation is used for the PHD filter. Assume that each target cluster in CSOs follows a linear Gaussian dynamical model on the focal plane and the sensor has a linear Gaussian measurement model, the transition density and likelihood for target cluster are given as

$$f_{k|k-1}(\tilde{\boldsymbol{x}}\mid\tilde{\boldsymbol{x}}') = \mathcal{N}\left(\tilde{\boldsymbol{x}}; \boldsymbol{F}_{k-1}\tilde{\boldsymbol{x}}', \boldsymbol{Q}_{k-1}\right) \tag{7}$$

$$g_k(\tilde{\boldsymbol{z}}\mid\tilde{\boldsymbol{x}}) = \mathcal{N}\left(\tilde{\boldsymbol{z}}; \boldsymbol{H}_k\boldsymbol{m}, \tilde{\boldsymbol{R}}_k\right) \tag{8}$$

where $\mathcal{N}(\cdot;\boldsymbol{m},\boldsymbol{P})$ denotes a Gaussian density with mean $\boldsymbol{m}$ and covariance $\boldsymbol{P}$, $\boldsymbol{F}_{k-1}$ is the state transition matrix, $\boldsymbol{Q}_{k-1}$ is the process noise covariance, $\boldsymbol{H}_k$ is the observation matrix, and $\tilde{\boldsymbol{R}}_k$ is the observation noise covariance associated with the representative measurement $\tilde{\boldsymbol{z}}_k$. It is worth mentioning that the linear model assumption can be relaxed to the nonlinear case by combining with EKF/UKF or particle based method[1].

The dynamical model for spawn targets is also linear Gaussian obviously. Then the intensities of the birth and spawn RFSs are Gaussian mixtures of the form

$$\gamma_k(\tilde{\boldsymbol{x}}) = \sum_{i=1}^{J_{\gamma,k}} \omega_{\gamma,k}^{(i)} \mathcal{N}\left(\tilde{\boldsymbol{x}}; \boldsymbol{m}_{\gamma,k}^{(i)}, \boldsymbol{P}_{\gamma,k}^{(i)}\right) \tag{9}$$

$$b_{k|k-1}(\tilde{\boldsymbol{x}}\mid\tilde{\boldsymbol{x}}') = \sum_{j=1}^{J_{\beta,k}} \omega_{\beta,k}^{(j)} \mathcal{N}\left(\tilde{\boldsymbol{x}}; \boldsymbol{F}_{\beta,k-1}^{(j)}\tilde{\zeta} + d_{\beta,k-1}^{(j)}, \boldsymbol{Q}_{\beta,k-1}^{(j)}\right) \tag{10}$$

where $J_{\gamma,k}, \omega_{\gamma,k}^{(i)}, \boldsymbol{m}_{\gamma,k}^{(i)}, \boldsymbol{P}_{\gamma,k}^{(i)}, i = 1, \cdots J_{\gamma,k}$, are given model parameters that determine the shape of the birth intensity; $J_{\beta,k}, \omega_{\beta,k}^{(j)}, \boldsymbol{F}_{\beta,k-1}^{(j)}, d_{\beta,k-1}^{(j)}$, and $\boldsymbol{Q}_{\beta,k-1}^{(j)}, j = 1, \cdots J_{\beta,k}$, determine the shape of the spawning intensity of a target from the target cluster with the previous state $\tilde{\zeta}$.

We assume that the survival and detection probabilities are state independent, i.e., $p_{S,k}(\tilde{\boldsymbol{x}}) = p_{S,k}$, $p_{D,k}(\tilde{\boldsymbol{x}}) = p_{D,k}$. Then, given

the posterior intensity at time $k-1$ as the Gaussian mixture

$$D_{k-1|k-1}(\tilde{\boldsymbol{x}}) = \sum_{i=1}^{J_{k-1}} \omega_{k-1}^{(i)} \mathcal{N}\left(\tilde{\boldsymbol{x}}; \boldsymbol{m}_{k-1}^{(i)}, \boldsymbol{P}_{k-1}^{(i)}\right), \quad (11)$$

the prediction and update of PHD filter is given as follows[9],

$$D_{k|k-1}(\tilde{\boldsymbol{x}}) = \gamma_k(\tilde{\boldsymbol{x}}) + p_{S,k} \sum_{i=1}^{J_{k-1}} \omega_{k-1}^{(i)} \mathcal{N}\left(\tilde{\boldsymbol{x}}; \boldsymbol{m}_{S,k|k-1}^{(i)}, \boldsymbol{P}_{S,k|k-1}^{(i)}\right) + \sum_{j=1}^{J_{k-1}} \sum_{l=1}^{J_{\beta,k}} \omega_{k-1}^{(j)} \omega_{\beta,k}^{(l)} \mathcal{N}\left(\tilde{\boldsymbol{x}}; \boldsymbol{m}_{\beta,k|k-1}^{(j,l)}, \boldsymbol{P}_{\beta,k|k-1}^{(j,l)}\right) \quad (12)$$

$$D_{k|k}(\tilde{\boldsymbol{x}}) = \left(1 - p_{D,k}\right) D_{k|k-1}(\tilde{\boldsymbol{x}}) + \sum_{i=1}^{J_{k|k-1}} \sum_{\tilde{\boldsymbol{z}} \in \tilde{Z}_k} \omega_k^{(i)}(\tilde{\boldsymbol{z}}) \mathcal{N}\left(\tilde{\boldsymbol{x}}; \boldsymbol{m}_{k|k}^{(i)}(\tilde{\boldsymbol{z}}), \boldsymbol{P}_{k|k}^{(i)}\right) \quad (13)$$

where the means $\boldsymbol{m}_{S,k|k-1}^{(i)}$, $\boldsymbol{m}_{\beta,k|k-1}^{(j,l)}$, $\boldsymbol{m}_{k|k}^{(i)}(\tilde{\boldsymbol{z}})$ and covariances $\boldsymbol{P}_{S,k|k-1}^{(i)}$, $\boldsymbol{P}_{\beta,k|k-1}^{(j,l)}$, $\boldsymbol{P}_{k|k}^{(i)}$ are computed with the Kalman filter.

Specifically, the updated weights $\omega_k^{(i)}(\tilde{\boldsymbol{z}})$ are computed as

$$\omega_k^{(i)}(\tilde{\boldsymbol{z}}) = \frac{p_{D,k} \omega_{k|k-1}^{(i)} \mathcal{N}\left(\tilde{\boldsymbol{z}}; \hat{\tilde{\boldsymbol{z}}}_{k|k-1}^{(i)}, \boldsymbol{S}_{k|k-1}^{(i)}\right)}{\kappa_k(\tilde{\boldsymbol{z}}) + p_{D,k} \sum_{l=1}^{J_{k|k-1}} \omega_{k|k-1}^{(l)} \mathcal{N}\left(\tilde{\boldsymbol{z}}; \hat{\tilde{\boldsymbol{z}}}_{k|k-1}^{(l)}, \boldsymbol{S}_{k|k-1}^{(l)}\right)} \quad (14)$$

where $\hat{\tilde{\boldsymbol{z}}}_{k|k-1}^{(i)} = \boldsymbol{H}_k \boldsymbol{m}_{k|k-1}^{(i)}$ is the predicted measurement and $\boldsymbol{S}_{k|k-1}^{(i)} = \boldsymbol{H}_k \boldsymbol{P}_{k|k-1}^{(i)} \boldsymbol{H}_k^T + \tilde{\boldsymbol{R}}_k$ is the innovation covariance.

4. SIMULATIONS

Simulations and results on typical CSOs tracking are presented in this section to examine the performance of the proposed method.

4.1 Scenarios and OSPA metric

Consider a typical CSOs scenario where all the targets are closely spaced to create only one representative measurement in the center of focal plane at the beginning, and the CSOs become resolved to generate more measurements because of the independent movement of each constituent target as time elapses. Fig. 2 shows the processed image observations of CSOs with 50 targets at different time instants.

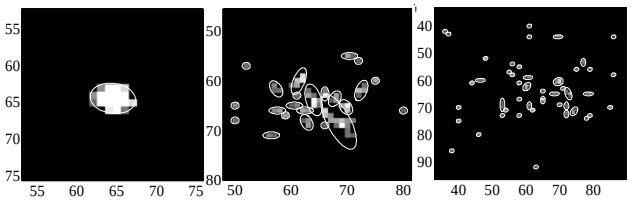
The survival and detection probability for each target cluster are set as $p_{S,k} = 0.99$, $p_{D,k} = 0.95$ respectively. The parameters for target model in Eq. (7) is give as

$$\boldsymbol{F}_{k-1} = \begin{bmatrix} \boldsymbol{I}_2 & \Delta \boldsymbol{I}_2 \\ \boldsymbol{0}_2 & \boldsymbol{I}_2 \end{bmatrix}, \quad \boldsymbol{Q}_{k-1} = \sigma_v^2 \begin{bmatrix} \frac{\Delta^4}{4} \boldsymbol{I}_2 & \frac{\Delta^3}{2} \boldsymbol{I}_2 \\ \frac{\Delta^3}{2} \boldsymbol{I}_2 & \Delta^2 \boldsymbol{I}_2 \end{bmatrix} \quad (15)$$

where $\Delta = 1\,s$ is the image sampling period, and $\sigma_v = 0.5\ (pixel/s^2)$ is the standard deviation of process noise. The measurement follows the observation model in Eq. (8) with $\boldsymbol{H}_k = [\boldsymbol{I}_2\ \boldsymbol{0}_2]$, $\tilde{\boldsymbol{R}}_k = \sigma_\varepsilon^2 \boldsymbol{I}_2 + \boldsymbol{R}_{\boldsymbol{z}_k}$, where $\sigma_\varepsilon = 1\ (pixel)$ is the standard deviation of measurement noise. Since all the targets are born in the center of focal plane, the intensity of the birth RFS is set as a single Gaussian $\gamma_k(\boldsymbol{x}) = 0.1\mathcal{N}\left(\boldsymbol{x}; \boldsymbol{m}_\gamma, \boldsymbol{P}_\gamma\right)$, where $\boldsymbol{m}_\gamma = [64, 64, 0, 0]^T$ and $\boldsymbol{P}_\gamma = diag\left([20, 20, 5, 5]\right)$ is used to model spontaneous birth in the vicinity of $\boldsymbol{m}_\gamma$. The RFS of targets spawned from an unresolved target cluster with the previous state $\tilde{\zeta}$ is Poisson with the intensity

$$b_{k|k-1}(\tilde{\boldsymbol{x}} \mid \tilde{\zeta}) = 0.1\mathcal{N}\left(\tilde{\boldsymbol{x}}; \tilde{\zeta}, \boldsymbol{Q}_\beta\right) \quad (16)$$

with $\boldsymbol{Q}_\beta = diag\left([20, 20, 10, 10]\right)$. The intensity of the RFS of clutter is set as $\kappa_k(\tilde{\boldsymbol{z}}) = \lambda_c V u(\tilde{\boldsymbol{z}})$, where $u(\cdot)$ is the uniform density over the focal plane, $V = 128 \times 128\ \text{pixel}^2$ is the "volume" of the focal plane, and $\lambda_c = 10^{-3}$ is the average number of clutter measurement per unit volume.



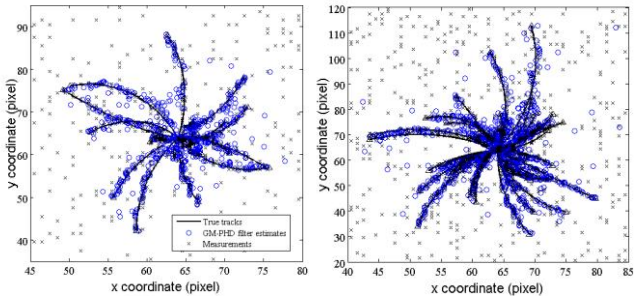
(a) t=10s (b) t=60s (c)t=100s

Figure 2: Processed images at different time instants

We use the optimal subpattern assignment (OSPA) metric[8] to evaluate the performance of proposed method. The OSPA distance is interpreted as a *p*th order per-target error, comprised of a *p*th order per-target localization error and a *p*th order per-target cardinality error. Let $d^{(c)}(\boldsymbol{x}, \boldsymbol{y}) := \min\left(c, \|\boldsymbol{x} - \boldsymbol{y}\|\right)$ for $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{X}$, where the $\mathcal{X}$ is target state space, and $\prod_k$ denote the set of permutations on $\{1, \cdots k\}$ for any positive integer k. Then, for $p \geq 1, c > 0$, and two arbitrary finite sets $X = \{\boldsymbol{x}_1, \cdots \boldsymbol{x}_m\}$ and $Y = \{\boldsymbol{y}_1, \cdots \boldsymbol{y}_n\}$, these components are define as

$$\bar{e}_{loc}^{(c)}(X, Y) := \left(\frac{1}{n} \cdot \min_{\pi \in \prod_n} \sum_{i=1}^{m} d^{(c)}(\boldsymbol{x}_i, \boldsymbol{y}_{\pi(i)})^p \right)^{1/p}$$
$$\bar{e}_{card}^{(c)}(X, Y) := \left(\frac{c^p (n-m)}{n} \right)^{1/p} \quad (17)$$

If $m \leq n$; and $\bar{e}_{loc}^{(c)}(X, Y) := \bar{e}_{loc}^{(c)}(Y, X)$, $\bar{e}_{card}^{(c)}(X, Y) := \bar{e}_{card}^{(c)}(Y, X)$ if $m > n$. The order parameter p determines the sensitivity of the metric to outliers, and the cutoff parameter c determines the relative weighting of the penalties assigned to cardinality and localization errors. We use the Eq. (17) with $p = 2$ and $c = 10\ (pixel)$ to evaluate the location estimate error and targets number estimate error of our filter for CSOs tracking.



(a) $N_t = 20$ (b) $N_t = 50$

Figure 3: Filter results of CSOs with different targets number.

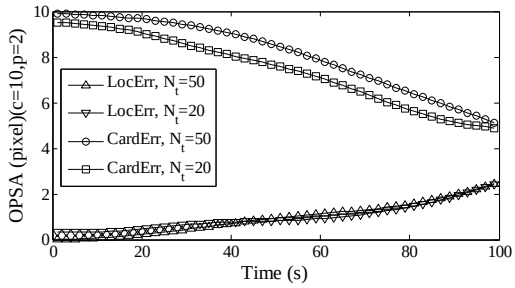


Figure 4: 100 MC run average OSPA components versus time with different targets number

4.2 Results and discussion

The true trajectories and filter estimates of CSOs with targets number as $N_t = 20$ are shown in x and y coordinates on focal plane in Fig. 3 (a), simulation duration is set to 100s. We see that the filter eliminates most of the clutters and gives the target estimates close to their true positions. As expected, the state estimate of each target cluster in CSOs is given at the beginning, then the individual targets are detected and their states are estimated as soon as resolved. And the effectiveness of our method is also proved in the scenario with a larger targets number $N_t = 50$, the results are shown in Fig. 3 (b).

We analyze the performance of our method by further examining the "localization" and "cardinality" components of the OSPA metric given in Eq. (17). The 100 Monte Carlo run averages of these two components versus time in same scenario for different targets numbers are shown in Fig. 4. In terms of localization errors, the maximum values are less than 2 pixels both for the two cases, which reveal the satisfactory localization accuracy of the filter over entire simulation time. Additionally, there are roughly equal error amounts for these two different targets number cases, which indicates that the localization error of our method is insensitive to the targets number thus can adapt to CSOs tracking with even higher target density. As for the cardinality errors, both for these two cases, we see high values close to the cutoff value in the early simulation time, and the errors decrease consistently as time elapses. In addition, larger targets number case ($N_t = 50$) has greater error amount. The reason after these trends is obvious. Since our filter tracks the targets as clusters when they are unresolved at the beginning period, the estimated target clusters number will be significantly smaller than the true value N_t , which results in large cardinality error. When these clusters become resolved as time passing, the newly resolved targets are detected thus decrease the errors. Consequently, it is believed that the cardinality estimating performance of our method can be further improved by combining with some super-resolution methods.

5. CONCLUSION

In this paper, an effective method using PHD filter is proposed for CSOs tracking from image observations. Different from the conventional individual object tracking methods, the new method tracks the unresolved target cluster similarly to the single target case, and jointly detecting the individual targets and estimating their states just after they are resolved. Simulation results show that its performance is satisfactory and useful for the large number of CSOs tracking in clutter environment. It is worth mentioning that combined with EKF/UKF or particle filter, this method can be adapted to accommodate nonlinear target model. Future work will consider a real-time practical application of the proposed method.

6. REFERENCES

[1] D. Clark, B.-T. Vo and B.-N. Vo, *"Gaussian Particle Implementations of Probability Hypothesis Density Filters," in IEEE Aerospace Conf., Big Sky, MT, USA, pp. 1–11, Mar. 2007.*

[2] R. Gonzalez and R. Woods, Digital Image Processing, *Pearson Education, 2002.*

[3] J. Hoshen and R. Kopelman, *"Percolation and cluster distributionPercolation and cluster distribution. I. Cluster multiple labeling technique and critical concentration algorithm," Physical Review B, vol. 14, no. 8, pp. 3438-3445, 1976.*

[4] J. Korn, H. Holtz, and M. S. Farber, *"Trajectory estimation of closely spaced objects (CSO) using infrared focal plane data of an STSS Platform," in Signal and Data Processing of Small Targets 2004, SPIE, Bellingham, WA, pp. 387-399, 2004.*

[5] F. Lian, C.-Z. Han, W.-F. Liu, X.-X. Yan and H. -Y. Zhou, *"Sequential Monte Carlo implementation and state extraction of the group probability hypothesis density filter for partly unresolvable group targets-tracking problem," IET Radar, Sonar and Navigation, vol. 4, iss. 5, pp. 685-702, 2010.*

[6] R. P. S. Mahler, *statistical multisource-multitarget information fusion, Artech House, 2007.*

[7] R. P. S. Mahler, *"Multitarget Bayes filtering via first-order multitarget moments," IEEE Transactions on Aerospace and Electronic Systems, vol. 39, no. 4, pp. 1152-1178, 2003.*

[8] D. Schuhmacher, B.-T. Vo and B.-N. Vo, *"A Consistent Metric for Performance Evaluation of Multi-Object Filters," IEEE Transactions on Signal Processing, vol. 56, no. 8, pp. 3447-3457, 2008.*

[9] B.-N. Vo and W. Ma, *"The Gaussian Mixture Probability Hypothesis Density Filter," IEEE Transactions on Signal Processing, vol. 54, no. 11, pp. 4091-4104, 2006.*

About the author

Yang Xu is a Ph.D. student at National University of Defense Technology, School of Electronic Science and Engineering, his contact email is xuyang012@nudt.edu.cn.

Hui Xu is a professor at National University of Defense Technology, School of Electronic Science and Engineering, his contact email is simon863@vip.sina.com.

Wei An is a professor at National University of Defense Technology, School of Electronic Science and Engineering, her contact email is nudtanwei@tom.com.

Scale-Space Color Blob Detection

Ekaterina V. Semeikina, Dmitry V. Yurin
Department of Computational Mathematics and Cybernetics
Moscow State University, Moscow, Russia
esemeikina@graphics.cs.msu.ru, yurin@cs.msu.ru

Abstract

Feature detection in color images frequently consists in image conversion from color to grayscale and then application of one of many known grayscale detectors. This approach has a few disadvantages: some features become indistinguishable in grayscale and features ordering based on grayscale detector response do not accord with features order of importance from human's perception point of view. In this paper the method for direct detection of blobs in color images is proposed. The proposed algorithm is based on scale space approach and estimates blobs sizes. Two modifications of the proposed method are given and compared.

Keywords: *color blob detection, feature points, scale-space, hessian matrix.*

1. INTRODUCTION

Point and linear feature detection is a base problem of image mosaicing, image registration, 3D recovery, pattern recognition, and scene analysis. It has been shown [8] that blobs are the most appropriate point features for the applications which need feature matching. In contrast to corners [4], blobs [8] have more stable location and size. The algorithm for color blob detection will be proposed in the paper.

Feature detection in color images often consists of conversion from color to grayscale mode and application of one of grayscale detectors. This approach has disadvantages described below.

Most of feature detectors [1, 4, 11] consist of three steps. The first of them is application of some transform to image in order to construct Feature Response Image (FRI). Typical examples of FRI are gradient absolute value image (for Canny edge detection [1]), difference of Gaussians (grayscale blob detection [8]), Harris functional (corner detection [4]). The second is extrema detection (or non maxima suppression [1]) in FRI. The third step is thresholding or hysteresis [1]: sufficiently large extrema are considered to be features. Using of thresholds gives rise to the first argument for color image analysis without conversion to grayscale: a possibility of *feature distinguishability reduction*. Equal brightness of a feature and background is a rare situation, but visibility decreasing of some features after conversion to grayscale is typical.

For a wide class of algorithm correct feature sorting in order of their importance is significant. [5] can be considered as an examples of the algorithms which estimate model parameters (homography or essential matrix) using matched pairs of features from different views. Estimation is fulfilled in two stages: using pairs of "most important" features (with high FRI value) and refinement with all feature pairs. If feature ordering is not stable, sets of most important features from different views can contain images of different points of 3D scene and matching will be incorrect.

The corner [10] and edge [2, 3] detectors for color images are known. Unfortunately such approach is not applicable to features dependent on second image derivatives. In this paper two variants of color blob detection are proposed. In our method sizes of blobs are defined adaptively in scale-space. We know the only work [9] on color blob detection, comparative discussion of the proposed method and [9] will be given. Also comparison with detection in converted to grayscale images is presented.

2. SCALE-SPACE

Blobs in image can be described using derivatives of an image brightness function. However, the input image $I(x,\ y)$ is given at discrete pixel mesh in the plane $(x,\ y)$. Scale-space theory [7] proposes to use instead discrete image $I(x,\ y)$ its version $L(x,\ y,\ t)$ blurred with the Gaussian kernel $G(x,\ y,\ t)$:

$$L(x,y,t) = G(x,y,t) \otimes I(x,y),$$
$$\text{where } G(x,y,t) = \frac{1}{2\pi\sigma^2} e^{-\left(x^2+y^2\right)/2\sigma^2},\ t = \sigma^2 \qquad \textbf{(1)}$$

Blurred function $L(x,\ y,\ t)$ is infinitely differentiable. Using convolution properties the derivatives of $L(x,\ y,\ t)$ can be calculated via convolution of $I(x,\ y)$ with the corresponding derivatives of Gauss function. Thus image derivatives depend on the blurring parameter $t = \sigma^2$ called *scale* [7].

In blob detection there is no a priory known scale, which should be used for derivatives calculation. A blob should be detected at the scale where it is visible better. Such scale is proportional to the blob size. This leads to consideration of one-dimensional family of images $L(x,\ y,\ t)$ blurred with different t.

Let us notice that since Gauss function is used as the blurring kernel, then function $L(x,\ y,\ t)$ satisfies diffusion equation [7]:

$$L_t = \frac{1}{2} \cdot \left(L_{xx} + L_{yy}\right). \qquad \textbf{(2)}$$

3. GRAYSCALE BLOB DETECTION AT A FIXED SCALE

Let us consider Taylor series (3) of image brightness at the point $(x,\ y)$ in order to explain popular blob [8] detection techniques basics.

$$dL(x,\ y,\ t_0) = \bar{\mathbf{g}}^T(x,y)\begin{pmatrix} dx \\ dy \end{pmatrix} + \frac{1}{2}\begin{pmatrix} dx \\ dy \end{pmatrix}^T \mathbf{H}(x,\ y)\begin{pmatrix} dx \\ dy \end{pmatrix} + \ldots,$$
$$\text{where } \mathbf{H}(x,\ y) = \mathbf{H}(x,\ y,\ t_0) = \begin{pmatrix} L_{xx} & L_{xy} \\ L_{xy} & L_{yy} \end{pmatrix} \text{ is Hessian matrix, scale } t = t_0 \text{ is fixed.} \qquad \textbf{(3)}$$

In the middle of a blob first derivatives (gradient) are small and the second term becomes the main. Image brightness $L(x,\ y,\ t_0)$ defines the surface and eigenvalues λ_1, λ_2 ($|\lambda_1| \geq |\lambda_2|$) of Hessian matrix characterizes curvature of this surface in direction of eigenvectors $\vec{v_1}$, $\vec{v_2}$. If both curvature values have the same sign and similar magnitudes then the feature is blob. Thus grayscale blob detection procedure consists of FRI construction $\lambda_2(x,\ y)$ and local extrema detection.

In order to avoid square root calculation Laplacian (4) is frequently used as FRI for blob detection:

$$trace\,(\mathbf{H}) = L_{xx} + L_{yy} = \lambda_1 + \lambda_2\,. \qquad \textbf{(4)}$$

Another popular FRI for blob detection is determinant of Hessian matrix $\det(\mathbf{H})$.

4. COLOR BLOB DETECTION AT A FIXED SCALE

4.1 Color variation vector and Hessian matrix for color image

Feature detection using Hessian matrix eigenvalues cannot be directly applied to color images. Usually in order to solve this problem grayscale image is constructed as a projection of color image to some direction in color space, for example, $(0.299,\ 0.587,\ 0.114)$. In this work we propose to select color projection direction adaptively in every image point as a direction in color space of the fastest color change.

In order to introduce vector of the fastest color change let us apply Laplace operator to each of image color channels and let us introduce auxiliary vector $\vec{C}$:

$$\vec{C} = \frac{1}{3}\cdot\left[R_{xx}+R_{yy},\ \ G_{xx}+G_{yy},\ \ B_{xx}+B_{yy}\right]^T. \qquad \textbf{(5)}$$

Application of the Laplace operator, i.e. convolution with second derivatives of Gauss function, means difference between weighted mean $\vec{C}_- = \frac{1}{3}\cdot\left[R_-,\ \ G_-,\ \ B_-\right]^T$ over the point neighbourhood of radius r_0 and weighted mean $\vec{C}_+ = \frac{1}{3}\cdot\left[R_+,\ \ G_+,\ \ B_+\right]^T$ over outer ring neighbourhood (Figure 1). Here $r_0 = \sigma\sqrt{N_G}$, where N_G is the dimensionality of used Gaussian function, in our case $N_G = 2$. So we can say that $\vec{C} = \vec{C}_+ - \vec{C}_-$ is a *color variation vector* in the feature neighbourhood.

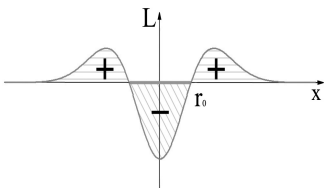


Figure 1: Second derivative of the 1D Gauss function.

Direction of $\vec{C}$ is a vector of the fastest color change (6):

$$\vec{c} = \left[c_R \quad c_G \quad c_B\right]^T = \frac{1}{\sqrt{3}}\vec{C}\Big/\left|\vec{C}\right| \qquad \textbf{(6)}$$

In order to form Hessian matrix, derivatives of the adaptively projected image should be constructed. Derivatives calculation in the point (x_i, y_i) via convolution with Gaussian derivatives would require forming of separate image L_i, using the fixed coefficients $\vec{c}(x_i, y_i)$ in all points of L_i.

$$\begin{aligned} L_i\ (x, y) &= c_R(x_i, y_i)\cdot R\ (x, y) + \\ &+ c_G(x_i, y_i)\cdot G\ \ (x, y) + \ c_B(x_i, y_i)\cdot B\ (x, y) \end{aligned}. \qquad \textbf{(7)}$$

Fortunately we can avoid building of this image set $\{L_i\}$ and calculate derivatives in all points directly from color components derivatives:

$$\begin{aligned} \hat{D}(L(x,y)) = &\ c_R(x,y)\cdot\hat{D}(R(x,y)) + \\ &+ c_G(x,y)\cdot\hat{D}(G(x,y)) + c_B(x,y)\cdot\hat{D}(B(x,y)) \end{aligned}' \qquad \textbf{(8)}$$

where $\hat{D}(\ldots)$ is differentiation operator.

Since only adaptive image derivatives are needed for blob detection, *the proposed method does not contain image conversion from color to grayscale*.

4.2 Detection at a fixed scale

Hessian matrix can be formed using derivatives (8). Further blob detection is analogous to detection in grayscale images using eigenvalues λ_1, λ_2 ($|\lambda_1| \geq |\lambda_2|$) of Hesse matrix.

Let us chose threshold β characterizing maximum allowed oblongness of a blob. Local extrema of FRI $\lambda_2(x,y)$, where $\lambda_2/\lambda_1 > 1/\beta > 0$, are detected as blobs. These local extrema can be detected via scanning by 3×3 frame and comparing central pixel on FRI with its 8 neighbour pixels. In our experiments $\beta = 8$ have been used.

In the previous work [6] $|\lambda_2(x,y)|$ was used as FRI and local maxima needed to be detected. Detection of maxima instead of maxima and minima slightly saves computation time but leads to artifacts.

4.3 "Continuity" and "compatibility" properties

Like classical color extensions of corner [10] and edge [2, 3] detectors the proposed method obeys an important "*continuity*" property: small changes in RGB channel values results in small FRI changes. It also obeys property of "*compatibility*" with grayscale detector: when image color components are equal:

$$R(x,y) \equiv G(x,y) \equiv B(x,y)\,, \qquad \textbf{(9)}$$

formulae used to construct FRI of color image come to formulae used for grayscale images. Thus the results of detection using color and grayscale algorithms are the same when (9) is true.

5. TWO VARIANTS OF SCALE-SPACE BLOB DETECTION

Unlike edge detection, where incorrect scale selection frequently leads only to some change of the edges shape, blobs will be missed if incorrect scale is used for detection. In the current sec-

tion the method for scale-space detection which allows to detect feature at the scale where it is better visible is proposed.

For blob detection in scale-space we construct Hessian matrices (3) for a set of sequential scales t_j using derivatives (8). Then we construct a set of FRIs $\lambda_2(x,\ y,\ t_j)$ and detect local extrema via scanning with window $3\times3\times3$.

This scale-space blob detection method can be modified if simultaneous blob and ridge detection is needed. Ridge detection [6] uses Hessian matrix $\mathbf{H}(x,\ y,\ t)$, characterizing brightness curvature in 3D space $(x,\ y,\ t)$:

$$\mathbf{H}(x,y,t)=\begin{pmatrix} L_{xx} & L_{xy} & L_{xt} \\ L_{xy} & L_{yy} & L_{yt} \\ L_{xt} & L_{yt} & L_{tt} \end{pmatrix}=\{diffusion\ equation\ (2)\}=$$

$$=\begin{pmatrix} L_{xx} & L_{xy} & \dfrac{L_{xxx}+L_{xyy}}{2} \\ L_{xy} & L_{yy} & \dfrac{L_{xxy}+L_{yyy}}{2} \\ \dfrac{L_{xxx}+L_{xyy}}{2} & \dfrac{L_{xxy}+L_{yyy}}{2} & \dfrac{L_{xxxx}+2L_{xxyy}+L_{yyyy}}{4} \end{pmatrix} \quad \textbf{(10)}$$

Using of second eigenvalue of Hessian matrix $\mathbf{H}(x,\ y,\ t)$ (10) as FRI gives result equivalent to result of detection with $\mathbf{H}(x,\ y)$ (3): sets of detected local extrema are almost the same (see example in Figure 2) and contrasts of the corresponding extrema are close. To compare extrema contrast two values have been calculated: in case of local maxima $c_1=\lambda_{mean}/\lambda_{\max}$ and $c_2=\lambda_{premax}/\lambda_{\max}$, where $\lambda_{\max}$ is detected maximum value, λ_{premax} is maximum value among $\Lambda_{3\times3\times3}$ - neigbourhood of $\lambda_{\max}$ of size $3\times3\times3$, λ_{mean} is mean value among $\Lambda_{3\times3\times3}$. Analogically in case of local minima: $c_1=\lambda_{mean}/\lambda_{\min}$ and $c_2=\lambda_{premin}/\lambda_{\min}$. Mean values over the test base, containing synthetic and natural images, are $\langle c_1\rangle\approx2.3$ and $\langle c_2\rangle\approx1.02$.

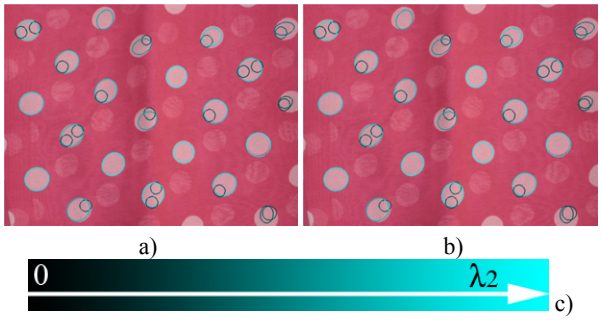


Figure 2: The blobs detected (response threshold 1.25): a) using $\mathbf{H}(x,\ y)$, b) using $\mathbf{H}(x,\ y,\ t)$, c) color of ellipses correspond to blob response.

Figure 3 illustrates eigenvalues of $\mathbf{H}(x,\ y)$ and $\mathbf{H}(x,\ y,\ t)$ at a set of scales. For a figure in the left part four FRI sets have been calculated: $\lambda_1(x,\ y,\ t)$ and $\lambda_2(x,\ y,\ t)$ of $\mathbf{H}(x,\ y)$ and $\mathbf{H}(x,\ y,\ t)$. Each diagram in the right part of Figure 3 contains the same row of FRI taken at sequential scales. Rows y_{cc} and y_{ec} of FRIs are shown, where circle blob in the original image has center $(x_{cc},\ y_{cc})$ and elliptic blob has center $(x_{ec},\ y_{ec})$. It can be seen that in the middle of circle or elliptic blob there is evident extremum of $\lambda_2(x,\ y,\ t)$ and both eigenvalues λ_1, λ_2 have close values.

6. COMPARISON

Adaptive projection (8) preserves features distinguishability towards background unlike methods using a fixed direction or the adaptive method [9] proposed by Ming and Ma (11):

$$\hat{D}(L\ (x,y)) = \frac{R(x,y)}{S(x,y)}\cdot\hat{D}(R\ (x,y))+ \\ +\frac{G(x,y)}{S(x,y)}\cdot\hat{D}(G(x,y))+\ \frac{B(x,y)}{S(x,y)}\cdot\hat{D}(B(x,y))\Big)' \quad \textbf{(11)}$$

where $S(x,y)=R(x,y)+G(x,y)+B(x,y)$.

Unlike the proposed algorithm method [9] take into consideration only the color in current point and do not use background color. Let us consider an example blob of color $(255,\ 0,\ 0)$ against a background of color $(255,\ 0,\ 5)$. In formula (11) red component, which does not distinguish feature and background, has the highest contribution while the most important blue component is suppressed, so projection direction is $(0.98,\ 0,\ 0.02)$. In our method (8) a component has the higher weight the higher change between feature and background for this component. The color variation vector is proportional to $(0,\ 0,\ 1)$ for the above example.

Comparative examples of blob detection with the proposed algorithm and detection after conversion to grayscale are given in Figure 4. It can be seen that after conversion to grayscale most noticeable from the human perception point of view color features are missed. At the same time these features have been detected by the proposed method with high response.

7. CONCLUSION

Two variants of scale-space algorithm for blob detection in color images have been developed. Up today the color detectors of features dependent on first derivatives (edges and corners) were known. In this paper we have proposed an other approach to using color information. In contrast to previous approach our method can be applied to features dependent on second derivatives, particularly blobs. It has been showed that the proposed method has advantages in comparison with the only previously known color blob detector and with detection after conversion to grayscale.

8. ACKNOWLEDGEMENTS

The work was supported by the Federal Program "Scientific and Scientific Pedagogical Personnel of Innovative Russia" in 2009–2013 and the Russian Foundation for Basic Research, project no. 09-07-92000-HHC_a.

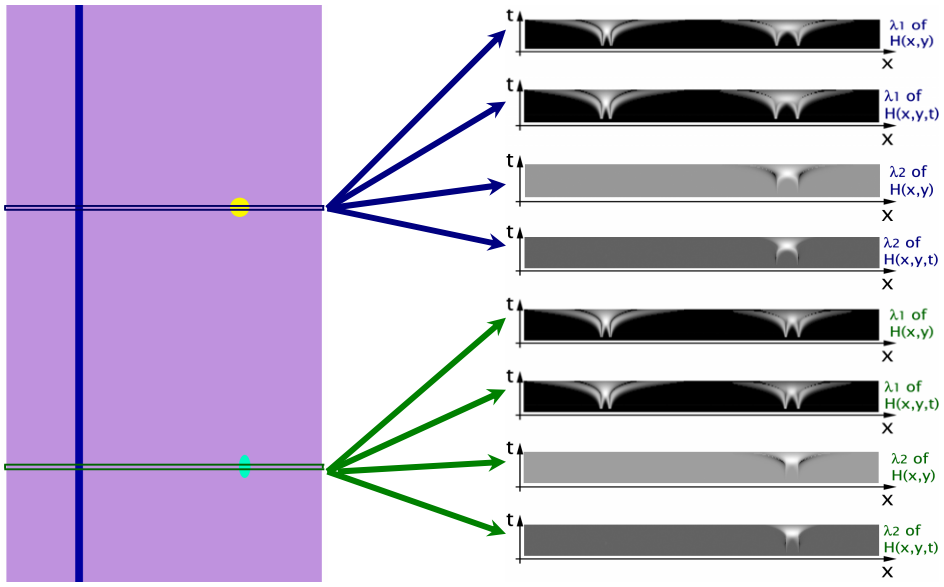


Figure 3: Eigenvalues of $\mathbf{H}(x, y)$ and $\mathbf{H}(x, y, t)$ at a set of scales. For a figure in the left part four FRI sets have been calculated: $\lambda_1(x, y, t)$ and $\lambda_2(x, y, t)$ of $\mathbf{H}(x, y)$ and $\mathbf{H}(x, y, t)$. Each diagram in the right part contains the same row of FRI taken at sequential scales. Shown rows are taken in the middle of circle and elliptic blobs. Values in diagrams are normalized to [0, 255].

Figure 4: Comparison of results obtained from the proposed algorithm ((c) and (d)) and detection after conversion to grayscale ((e) and (f)). Response threshold is 0.75. After conversion to grayscale most noticeable color features are missed.

9. REFERENCES

[1] J. Canny. *A computational approach to edge detection.* IEEE Trans. PAMI, 1986, V. 8, P. 34 - 43.

[2] A. Cumani. *Edge Detection in Multispectral Images.* Computer Vision, Graphics and Image Processing: Graphical Models Image Processing, 1991, V. 53, No. 1, P. 40 - 51.

[3] S. Di Zenzo. *A Note on the Gradient of Multi-Image.* Comp. Vision Graphics Image Processing, 1986, V. 33, P. 116 - 125.

[4] C. Harris, M. Stephens. *A combined corner and edge detector*. Proc. 4th Alvey Vision Conf., V. 15, 1988, P. 147–151.

[5] R. Hartley, A. Zisserman. *Multiple View Geometry in Computer Vision.* Cambridge University Press, 2004, P. 672.

[6] N. Khanina, E. Semeikina, D. Yurin. *Color Blob and Ridge Detection.* Proc. of 10th Conf. on Pattern Recognition and Image Analysis: New Inform. Technologies, 2010, V. 1, P. 289–292.

[7] T. Lindeberg. *Scale–Space Theory in Computer Vision*. Kluwer Academic Publishers, Dordrecht, 1994.

[8] D. G. Lowe. *Distinctive image features from scale-invariant keypoints.* IJCV 2004, V. 60, No 2, P. 91 - 110.

[9] A. Ming, H. Ma. *A Blob Detector in Color Images*. Proc. of the 6th ACM CIVR, 2007, P. 364 - 370.

[10] P. Montesinos, V. Gouet, R. Deriche. *Differential Invariants for Color Images*. Proc. of 14th ICPR, 1998, P.838–840.

[11] J. Shi, C. Tomasi. *Good features to track.* Proceedings of the IEEE CVPR'94, 1994, P. 593 - 600.

About the authors

Ekaterina V. Semeikina is a Ph.D. student at Chair of Mathematical Physics, Faculty of Computational Mathematics and Cybernetics, Lomonosov Moscow State University, Russia.

Her contact email is esemeikina@graphics.cs.msu.ru

Dmitry V. Yurin (PhD) is a senior researcher at laboratory of Mathematical Methods of Image Processing, Faculty of Computational Mathematics and Cybernetics, Lomonosov Moscow State University, Russia. His contact email is yurin_d@inbox.ru

Parallel SIFT-detector implementation for images matching

Anton I. Vasilyev, Andrey A. Boguslavskiy, Sergey M. Sokolov
Keldysh Institute of Applied Mathematics of Russian Academy of Sciences, Moscow, Russia
{ahbac, anbg}@mail.ru sokolsm@keldysh.ru

Abstract

This paper describes the parallel SIFT-detector implementation on the basis of the NVIDIA CUDA technology for the images matching. The SIFT-detector implementation was applied for the images matching in the stereo-system mounted on the moving car and for images from the onboard UAV-camera.

Keywords: *real-time computer vision, SIFT, CUDA, images matching.*

1. INTRODUCTION

Programmability of general purpose graphic processor units does them attractive for computer vision tasks in order to accelerate the processing and reduce the computational load of the central processor unit. One of the widespread GPU programming technologies is NVIDIA CUDA. The unified architecture of a set of graphic processor units NVIDIA GeForce/GTX and programming model with application of C-like language [10, 11] lies in its basis.

In the given work the GPU application for the SIFT-detector [1] implementation is described. This algorithm is often applied to detect reliable repeated features (blobs/keypoints). The difficulty of its application in real-time systems in practice is connected with the essential computing time consumption for the image preprocessing and the descriptors generating. Parallel implementation on the basis of CUDA has allowed to apply SIFT for the tie points searching in the stereoscopic visual system mounted on the moving car and for images from the onboard UAV-camera.

2. PARALLEL IMPLEMENTATION OF THE SIFT-DETECTOR WITH THE USE OF CUDA

There is a series of published papers on the subject of the SIFT-detector implementation for GPU, for example, [3-8], including open source on C++ [2] and CUDA [6, 7]. The implementation [7] is not documented and the work is presented without any time evaluations in use. In turn, in work [6] the most high-efficiency time evaluations are showed for realisation with shaders. In work [9] the comparison procedure of n-dimensional vectors on the basis of Euclidean metric with CUDA and the primitives library of computing linear algebra CUBLAS [10, 11] is described. A distinctiveness of the many papers is to evaluate an execution time of a general procedure, whereas in applied tasks it is possible to get an optimization and time gain due to the prior problem knowledge consideration. For example, there is no necessity to calculate the blobs orientations in the pure camera translation mode (without rotation). This speeds up the formation time of the descriptors array and reduces the processing time. Another example: the physical limits consideration allows to reduce an execution time of the descriptors matching, when the comparison of descriptors from the certain image area is performed, instead of the comparison of all descriptors. Thus, at designing of the own SIFT-detector implementation for the images matching, two main purposes were put: the possibility of the detector tuning to take into account features of the concrete applied problem and the possibility of the reuse in modular systems of computer vision of real time. As a whole, it is possible to tell that in the chosen approach to the SIFT-detector algorithm implementation was made the GPU-acceleration process using the GPU memory economically.

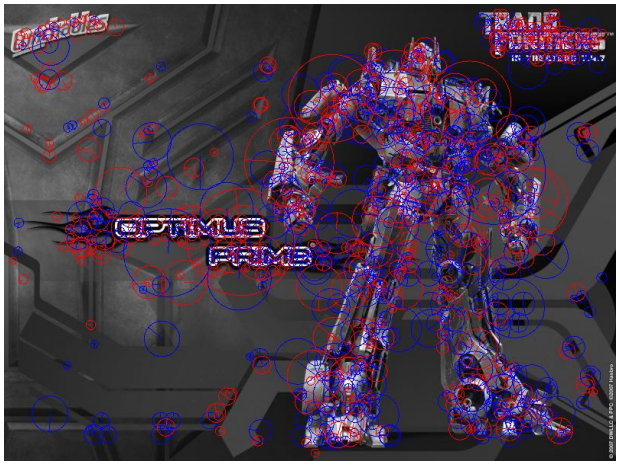


Figure 1: Example of the image processing (800x600, the image is taken from [6] to compare implementations)

2.1 Features of the SIFT-detector implementation

In the SIFT-detector [1] it is possible to emphasize following basic stages:

1) The amount of octaves (and sublevels in the octave) for the image is defined (an amount of the reduced copies with sizes multiple to degrees 2).
2) Formation of the scale-space representation (SSR) for the image: for each octave - the array construction of images, smoothed by Gauss filter (Gss array further), the array size depends of a number of sublevels
3) Formation of the scale-invariant map of edges. Subtraction of the adjacent smoothed images in Gss array is performed for each octave, i.e. outcomes of subtractions are added into the array (DoG array further).
4) The keypoints selecting for each octave: search of local extremes (pixels) for everyone DoG and between adjacent DoGs.
5) Qualification (subpixel position and scale) is fulfilled for each keypoint.

6) An orientation is assigned for each keypoint: the histogram construction of the gradient orientations in a neighborhood of blob (Gss) is fulfilled for corresponding (scale keypoint) smoothed image; the orientation corresponding to maximum bin gets out as a blob orientation.
7) The descriptor is calculated for each blob: the histograms array of gradient orientations is formed round a blob.

Implementation of the described algorithm with the use of the NVIDIA CUDA technology was constructed as (the algorithm stages are presented on figures 2-6):

1) Memory allocation (on GPU):
   - GPU-memory allocation for the image and its copying into the GPU-memory.
   - Memory allocation for SSR on GPU.
2) The octaves and sublevels processing is shown on figures 2-3 (it is realised sequentially for economic use of RAM GPU-memory; green and blue blocks are invoked GPU-memory for current iteration, at that green blocks use the matrix representation of GPU-data):
   - The Gss array generating (it is fulfilled on GPU only).
   - The partial DoG array generating (it is fulfilled on GPU only).
   - The local extremes search and subpixel qualification of them in the matrix representation (it is fulfilled only on GPU).
   - The keypoints redistribution (fig. 3): from the matrix into the linear array (it is fulfilled on GPU and CPU).
3) The orientations assigning for each blob is presented on figures 4-5 (after, all octaves were processed)
   - The orientations calculation for each blob: SSR is used (it is fulfilled on GPU only).
   - If several orientations are defined for one blob, the redistribution is performed in order to construct the array of blobs, having one orientation (it is fulfilled on GPU and CPU).
4) The descriptors generating: SSR is used (it is fulfilled on GPU only).

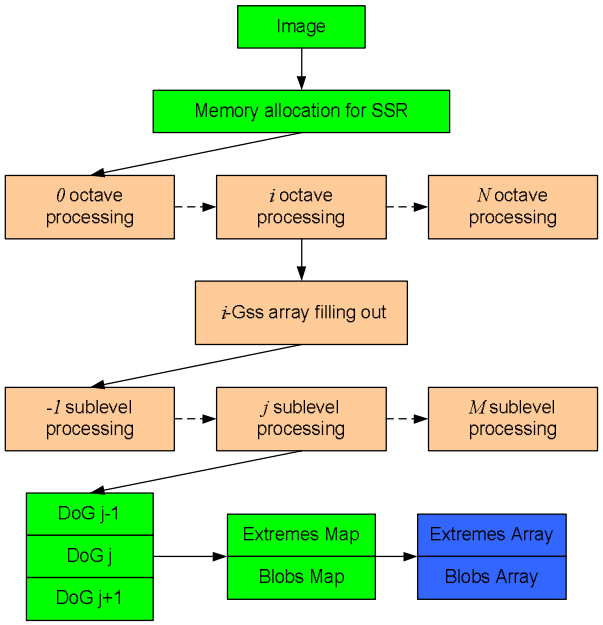


Figure 2: General scheme of the blobs extraction

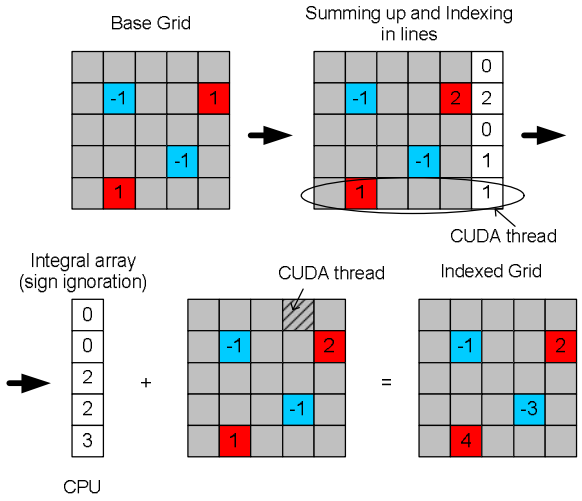


Figure 3: Example of the blobs redistribution (the base grid is extremes map; the indexed grid is used to convert a map to linear array; the integral sum is used to form the integral array)

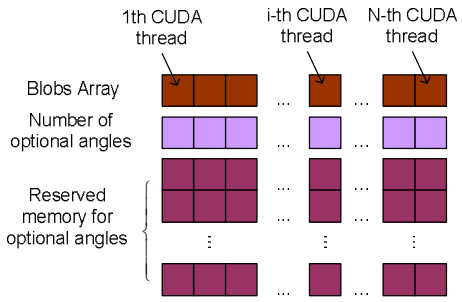


Figure 4: General scheme of the orientations calculation

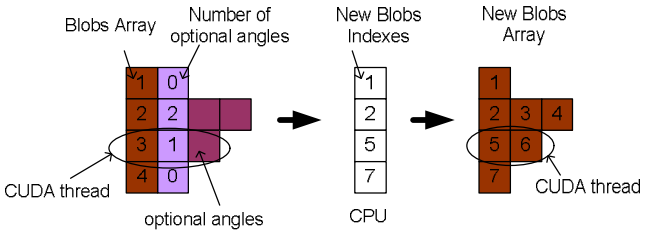


Figure 5: Example of the orientations calculation and reindexation (The integral sum using for the formation of new indexes array)

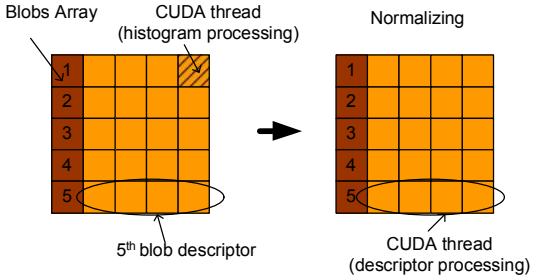


Figure 6: Example of the descriptors generating

Essential differences of our implementation from work [6]:

1) Texture memory is not used for the images arrays storage, as a consequence, there are losses in the access speed but indexation on the image SSR becomes simpler.
2) The octaves and sublevels processing is realised sequentially (economic use of RAM GPU-memory).
3) The data redistribution (reindexation of array elements and the transform from the sparse matrix representation into the linear representation also) has been realised with the use of GPU and CPU.
4) The orientations calculation is performed in parallel for all blobs on the SSR image, GPU and CPU are used for reindexation of the blobs array
5) The Descriptors generating is performed in parallel for all blobs on the SSR image, and the each histogram of descriptor is formed by the separate CUDA thread.
6) The octave index and extreme type are used as blob parameters
7) The detector tuning includes next parameters:
   - Start octave index (down-sampling or up-sampling for the base octave)
   - the amount of octaves and sublevels
   - the maximum quantity of multi orientations for blobs (inclusive of nothing)
   - the SIFT-descriptor size (a number of histograms and their size)

2.2 Features of the matching implementation

The matching procedure (fig. 7) contains the following operations:

1) The physical limits matrix (Boolean matrix) is prepared on the basis of the discovered blobs
2) The calculation of Euclidean distances matrix is performed with the use of the physical limits matrix

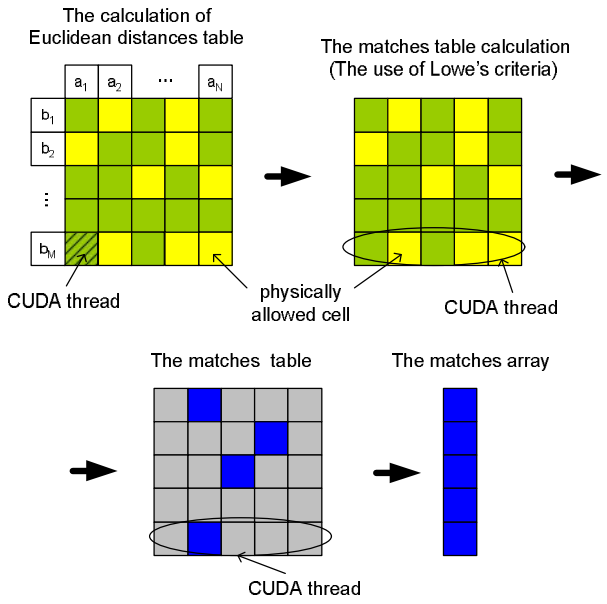


Figure 7: General scheme of the matching procedure (green cells are not admissible physically, blue cells are matches)

3) The matches table is calculated based on the physical limits matrix and Euclidean distances matrix by means of Lowe's criteria [1]
4) The matches table is converted from the matrix representation into the linear array

3. TIME EVALUATIONS AND RESULTS IN PRACTICE

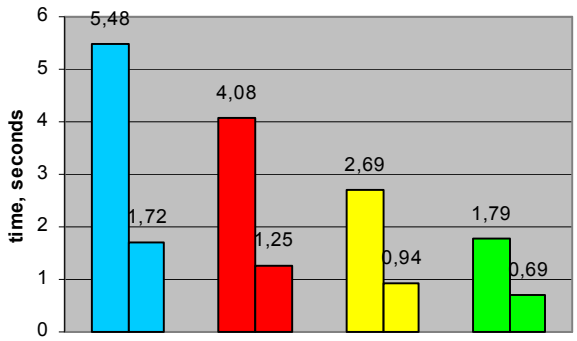
The processing example of the separate image (800x600 pixels, all outcomes are shown for GeForce GTX 260) is represented in tab. 1 for the execution time evaluation of our implementation. Settings were set: 0 start octave index (no up- or down-sampling), 4 octaves with 5 sublevels, no more than two orientations for one blob, 128 elements (4x4x8) in the SIFT-descriptor. On fig. 1 the processing result is shown (blobs/extremes: red - minima, dark blue – maxima). The picture (fig. 1) is taken from work [6] for the purpose of comparison with the published implementation [6].

Table 1: Time evaluations

Keypoints extraction (subpixel position and scale, the linear array formation)			37.5 ms/974 blobs	
The orientations assigning and regrouping			5.1 ms/1139 blobs	
The descriptors generating			43.89 ms	
Total time			87.2 ms	
Key points extraction				
Octave	0	1	2	3
The Generating of Gss-arrays pyramid, ms	13.25	3.13	1.18	0.68
Transform from sparse matrix into linear array (indexing), ms	2.36	1.08	0.67	0.475
Blobs searching (including indexing), ms	10.47	4.48	2.71	2.32
A number of the discovered blobs	612	256	89	17

The received time evaluations concede the implementation (based on shaders) presented in work [6], for example, the image processing on fig. 1 is about 53.9 ms/1052 blobs (in the multi orientation mode). However in our implementation the octaves processing is performed in a consecutive way instead of a parallel way. Thereby resources of RAM GPU-memory are used more economically, as it's important for the UAV-shots processing. For example, the pair of shots (4016x2672 pixels everyone) is formed simultaneously for twin-lens UAV-camera. In turn, the processing of one such shot took about two seconds, whereas it's not possible to process such shot without down-sampling in [6] (the application was downloaded from the site).

The physical limits consideration for the two UAV-shots matching (2008x1336 in size, figures 9-11 in supplementary materials) is presented on the plot 1. The processing time of each of pictures is about 500 ms, at that size of blobs array is about 10.000 elements. The histograms are shown for two cases, when descriptor contains 128 and 32 elements (big and small bins).



Plot 1: The matching time evaluations for UAV-shots with the physical limits consideration (128- and 32- dimensional descriptor): cyan bins – general procedure (no limits), red bins consider the motion without rotation (no orientations), yellow bins – the motion without rotation and the consideration of similar extremes types, green bins – the motion without rotation, the consideration of similar extremes types and octaves indexes

The processing of two shots (nonsequential shots of video, 720x576 pixels in size, figures 12-13 in supplementary materials) took about 300 ms: image 1 – 96 ms/1377 blobs, image 2 – 97 ms/1437 blobs, matching (without any physical limits) – 108 ms/67 pairs. If we use the physical limits (without rotation, similar extremes types, 32 elements in the descriptor) then the processing time is about 108 ms: image 1 – 44 ms/1251 blobs, image 2 – 45 ms/1341 blobs, matching – 19 ms/120 pairs.

Our implementation work analysis by means of the CUDA visual profiler is presented on fig. 8 (in supplementary materials). From drawing it can be seen, that the greatest time is spent for the SIFT-descriptors generating and convolution operation, and the summarized contribution for copying operation of the data between CPU↔GPU makes less than 5 %.

4. CONCLUSION

In the given work the images matching based on the SIFT-detector is shown for two various applied problems: the video processing for stereoscopic measurements and the UAV-shots processing for mosaic formation. The serial-parallel scheme of the blobs extraction, the possibility of the detector tuning and the physical limits consideration allowed to increase the processing speed without essential losses of the used GPU-memory. The next step of development for this implementation will be an inclusion the CUFFT library in order to form the image scale-space representation and the small-size octaves processing in parallel.

5. REFERENCES

[1] D. G. Lowe, “Distinctive image features from scale-invariant keypoints,” International Journal of Computer Vision, vol. 60, pp. 91–110, 2004

[2] A. Vedaldi. (2011) Sift ++ source code and documentation. http://www.vlfeat.org/

[3] Warn S., Emeneker W., Cothren J., Apon A. “Accelerating SIFT on Parallel Architectures”, Cluster Computing and Workshops, 2009

[4] S. Sinha, J.-M. Frahm, M. Pollefeys, and Y. Genc, “Feature tracking and matching in video using programmable graphics hardware,” Machine Vision and Applications, March 2007

[5] S. Heymann, K. Muller, A. Smolic, B. Frohlich, and T. Wiegand. SIFT Implementation and Optimization for General-Purpose GPU, in WSCG ’ 07, 2007

[6] C. Wu (2011) "SIFTGPU: A GPU implemenatation of scale invariant feature transform (SIFT)", http://www.cs.unc.edu/~ccwu/siftgpu/#lowesift

[7] Bjorkman M., A CUDA implementation of SIFT http://www.csc.kth.se/~celle/

[8] Kayombya G.-R., "Implementation and Optimization of SIFT on an OpenCL GPU", 2010, http://beowulf.csail.mit.edu/18.337/projects/reports/Kayombya_report.pdf

[9] V. Garcia, E. Debreuve, F. Nielsen, M. Barlaud "KNN Search: fast GPU-based implementations and application to high-dimensional feature matching", ICIP, 2010

[10] NVIDIA, “CUDA technology,” 2011, http://www.nvidia.com/object/cuda_home_new.html

[11] Borescov A.V., Harlamov A.A. “CUDA basics”. – Moscow, DMK-Press, 2010 (In Russian).

About the author

Anton Vasilyev, Keldysh Institute of Applied Mathematics of Russian Academy of Sciences, postgraduate student

Andrey Boguslavskiy, Keldysh Institute of Applied Mathematics of Russian Academy of Sciences, senior scientist

Sergey Sokolov, Keldysh Institute of Applied Mathematics of Russian Academy of Sciences, leading scientist

SUPPLEMENTARY MATERIALS

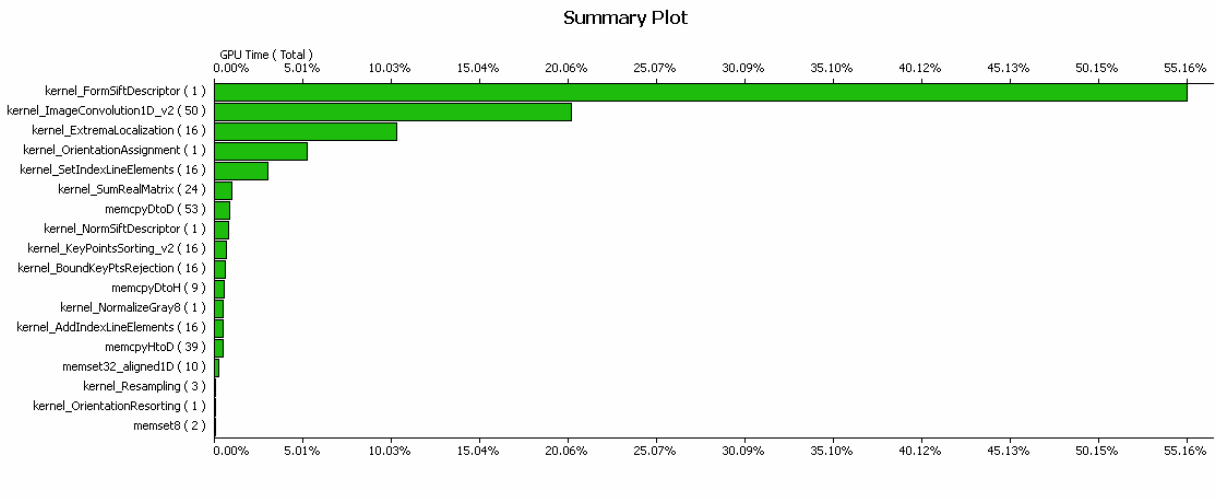


Figure 8: The program work analysis outcomes by means of the CUDA visual profiler

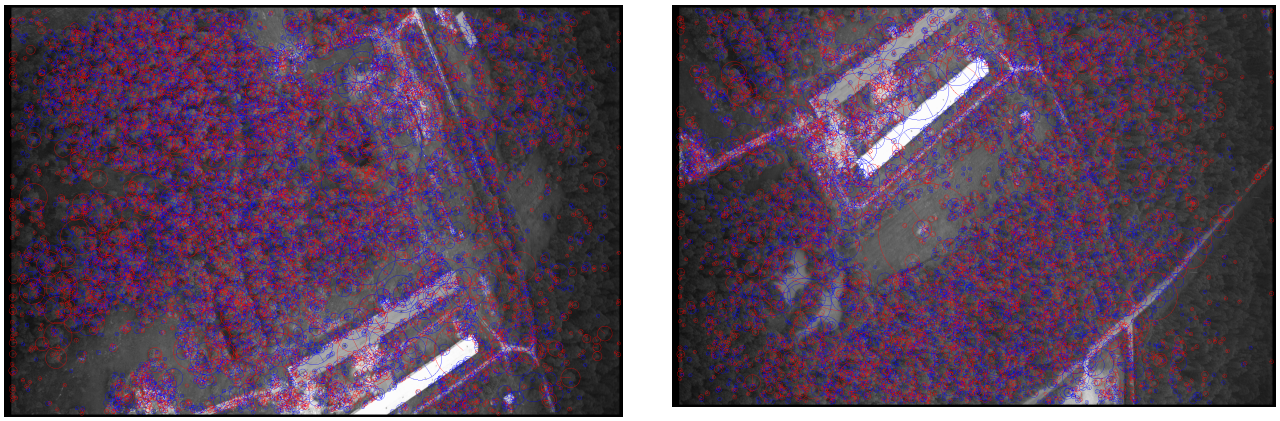


Figure 9: The UAV-shots processed by the SIFT-detector (7 octaves, 5 sublevels, 2 orientations): 539 ms/11058 blobs and 539 ms/9258 blobs

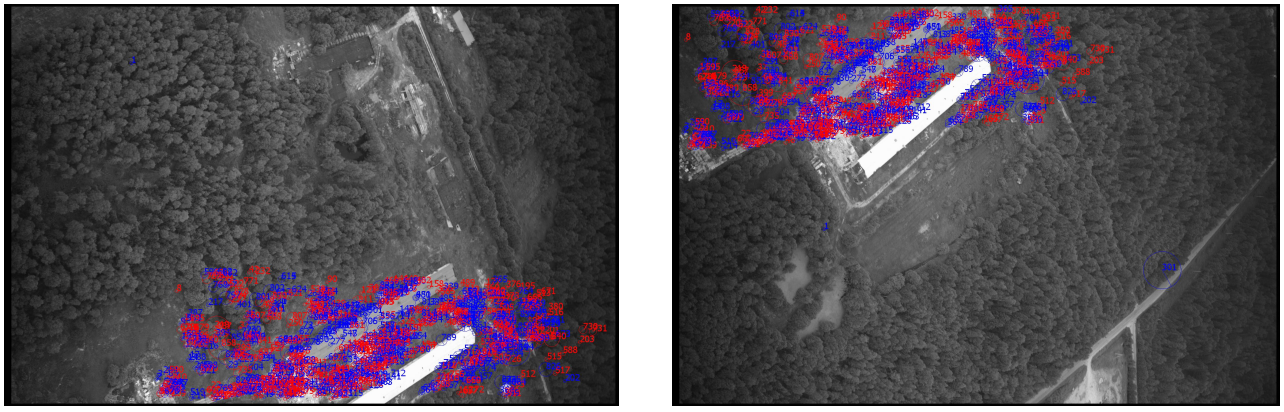


Figure 10: The UAV-shots matching (128-dimensional descriptors, without RANSAC filtering): 5.47 s/865 pairs

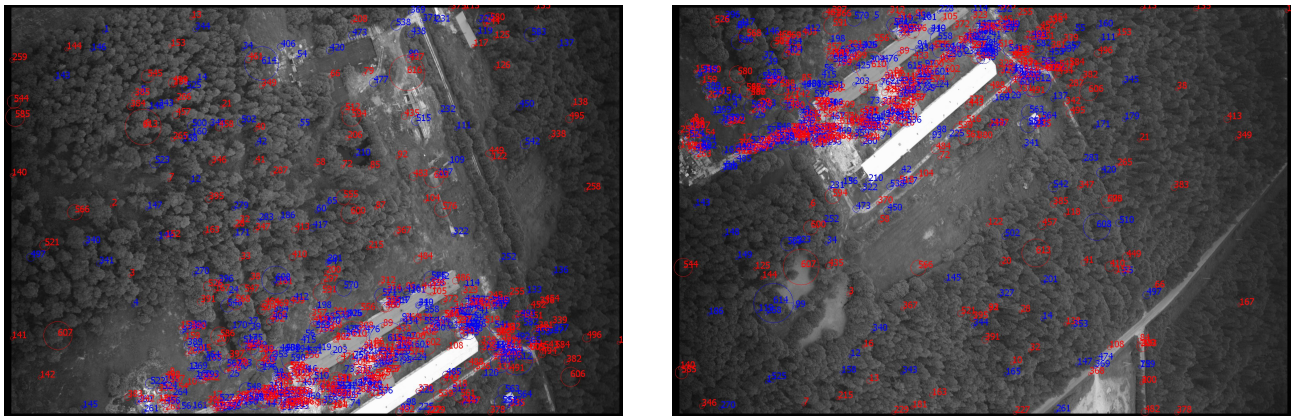


Figure 11: The UAV-shots matching (4 octaves, 5 sublevels, 0 orientations, 32-dimensional descriptors, the consideration of similar extremes types and octaves indexes): 201 ms/9749 blobs, 201 ms/8104 blobs, 695 ms/616 pairs

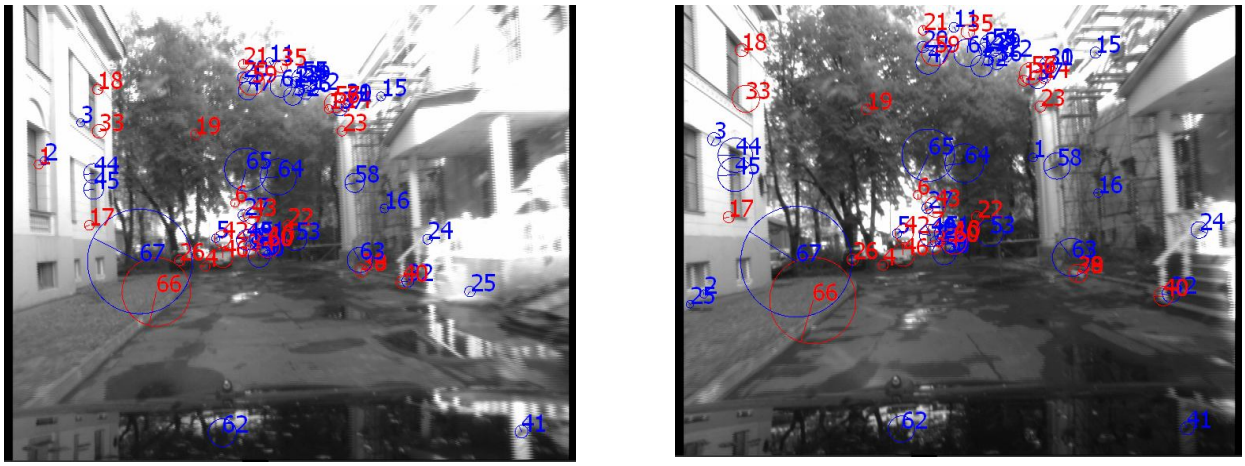


Figure 12: The shots matching from the camera mounted on the moving car (4 octaves, 5 sublevels, 2 orientations, 128-dimensional descriptors, no limits): 96 ms/1377 blobs, 97 ms/1437 blobs, 108 ms/67 pairs

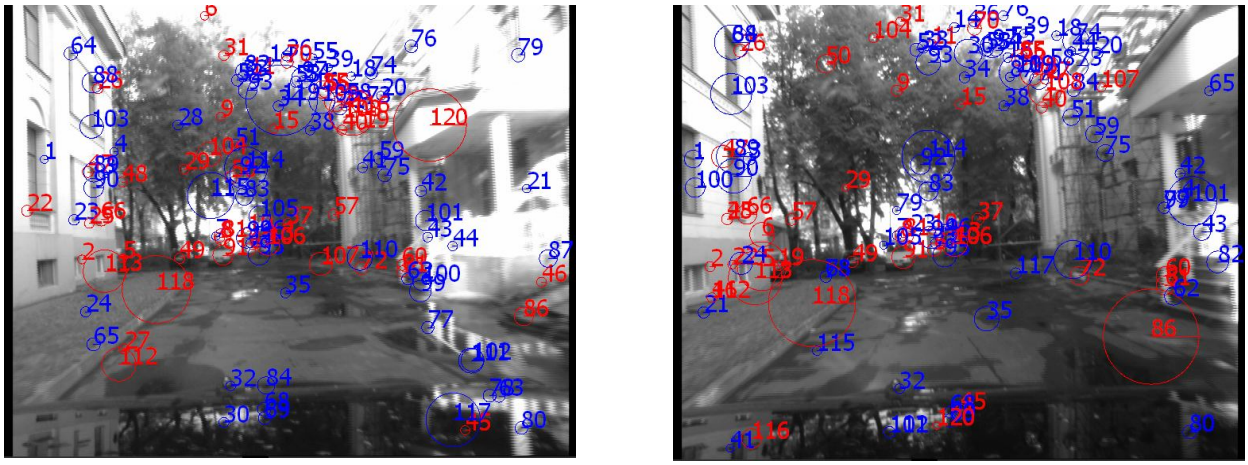


Figure 13: The shots matching from the camera mounted on the moving car (4 octaves, 5 sublevels, 0 orientations, 32-dimensional descriptors, the similar extremes types): 44 ms/1251 blobs, 45 ms/1341 blobs, 19 ms/120 pairs

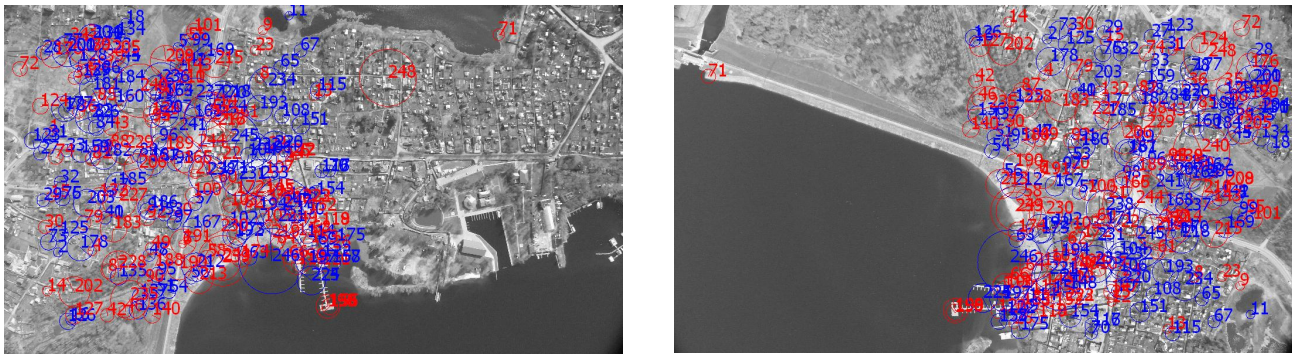


Figure 14: The UAV-shots matching (4224x2376 in size, start octave index 2, 7 octaves, 5 sublevels, 2 orientations, 128-dimensional descriptors, the consideration of similar extremes types and octaves indexes): 102 ms/1019 blobs, 91 ms/695 blobs, 15 ms/248 pairs

Automatic merging of DSMs extracted by image matching algorithms

Verkeenko Mikhail
The Federal State Unitary Enterprise
«State Research Institute of Aviation Systems», Moscow, Russia
envenom.gm@gmail.com

Abstract

In this paper a new approach to building 3D model using existing Digital Surface Models (DSMs) extracted by image matching algorithms is proposed. This problem arises when the 3D object to be reconstructed has a very complicated form and its full surface cannot be restored using only one stereo pair. A new algorithm based on image orientation data and 2D Delaunay triangulation is presented. This method is compared with another widespread approach for 3D surface triangulation. Two types of objects have been used for this article: extended bas-relief and the closed object "Monument to the Heroes of Plevna".

Keywords: *3D model triangulation, DSM, photogrammetry, image matching, Delaunay triangulation, merging of triangulations*

1. INTRODUCTION

At current, digital models of objects are using in many areas: electronic geographic information systems, virtual and augmented reality, mission planning and rehearsal, urban planning etc.. Also, a variety of models is in demand now: typical mass-produced houses of urban development, unique historical buildings, palaces, castles and unique works of architecture, such as the bas-reliefs and sculptures. Automated construction of digital models of objects based on ground-based images and reference data is a science-intensive and highly relevant problem. To date, photogrammetric systems can give the same dense cloud of points, as well as laser scanning [1, 8]. This explains the increasing importance of developing special algorithms and digital photogrammetric systems for data processing ground surveys.

We are not facing problems in case of using only one stereo pair for the reconstruction of the object's surface; since we can use 2D Delaunay triangulation for the corresponding projection of matched point cloud on a plane of stereo pair. Unfortunately, it's unable to reconstruct all closed objects using only one stereo pair. Therefore a problem of merging a number of DSMs arises.

2. RELATED WORKS

Since the problem of surface reconstruction from point cloud is very important, it was proposed many methods over past decades. Several approaches are based on combinatorial structures such as 3D Delaunay triangulations, alpha shapes [5], or Voronoi diagrams [3]. These schemes typically create a triangle mesh that interpolates all or a most of the points. Other schemes directly reconstruct an approximating surface, typically represented in implicit form [4]. A Poisson Surface Reconstruction (PSR) technique [3], developed by Kazhdan et al. (2006), has been adopted for mesh generation. This technique is a novel approach that expresses surface reconstruction as the solution to a Poisson equation. Kazhdan et al. (2006) demonstrate that the Poisson algorithm can facilitate the reconstruction of surfaces with greater detail than previously achieved. This algorithm uses points positions and their normals for surface reconstruction.

Another class of approaches analyzes same triangles in neighboring triangulations and uses them for surfaces transformation and merging. But these methods are unstable in case of different image acquisition conditions, which can cause difference between matched points or reconstructed triangles in the intersection zone [7].

The main feature of our approach is to use an advantage of image orientation data. It can be helpful for moving the problem from 3D space into 2D.

3. INPUT DATA

The input data for the merging procedure is:

- A set of point clouds corresponding to each stereo pair (Figure 1). For each point cloud a triangulation that accurately approximates the actual surface of the object can be computed (Figure 2). Point clouds must have areas of intersection; otherwise the procedure of merging cannot be executed.

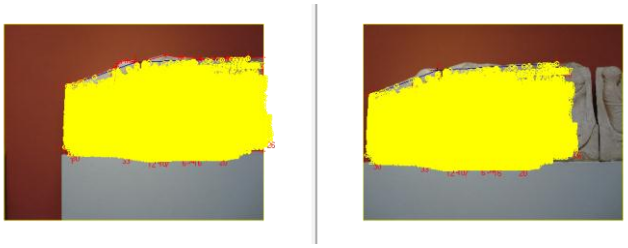


Figure 1: Point cloud for a part of bas-relief

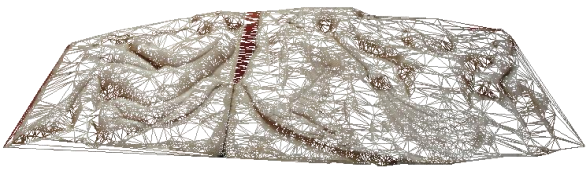


Figure 2: Triangulation for a part of bas-relief

- Possible constraints for triangulation of current point cloud.
- Orientation for each stereo pair, allowing to project points from 3D space onto an image plane, to restore point position in the 3D space based on 2D points from left and right stereo pair images and to determine

position and view vector of camera attached to the current image (Figure 3).

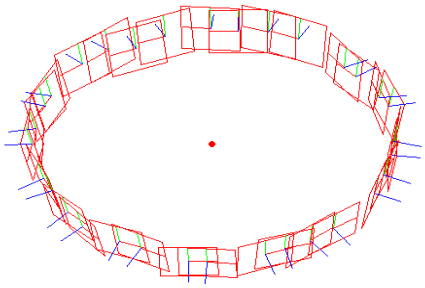


Figure 3: Orientation scheme for the object "Monument to the Heroes of Plevna"

4. THE PROPOSED METHOD

The proposed method consists of few steps:

- Creating the initial triangulation (mesh) for each of the matched stereo pairs and marking one of them as "current".
- Iterative process
    - Searching for adjacent triangulations, which have zones of intersection with the current triangulation.
    - Merging of 2 triangulations.

Iterative process will change triangulation only in the zone of intersection.

4.1 Preparing initial triangulations

This is a fairly simple step, since it's sufficient to project the points of each cloud on the stereo pair and to triangulate projected points using Delaunay triangulation procedure [2]. We have to take triangulation constraints into consideration, if it's necessary. The resulting triangulation quite accurately reflects the actual geometry of the matched surface, as it's performed on the points extracted from respective images. Also, it solves the problem with overlapping of triangles – all triangles are visible from the corresponding point of view.

4.2 Searching for neighbor triangulations and determination of the mutual intersection zone

4.2.1 Search for neighbors

Firstly, we must introduce a criterion to mark a triangulation as a neighbor. We can check intersections between bounding shapes(Oriented Bounding Box (OBB) for tested triangulation or 3D convex hull for tested triangulation) and current triangulation's points for this purpose (It would be better to utilize OBB if the computational time is critical). If there are few triangulations, that are neighbors, then we can use them together or separately, processing the first triangulation with maximum intersection count. In this work preferences were given to the latest version, as the most evident. The only exception is the closure of the object. We have a high probability that a mutual intersection zone exists in case of the exact orientation and matching.

4.2.2 Determination of the mutual intersection zone

After finding the neighbor triangulation, that is suitable for merging, we should more accurately identify the zone of mutual intersection. We can take intersection points from subsubsection 4.2.1 and identify appropriate triangles from current triangulation. Points and triangles from the neighbor triangulation can be defined similar. Thus we get:

- Point cloud inside the mutual intersection zone (red).
- Triangles outside the intersection zone (green).

In practice, often turns out that this data cannot be used for merging without obtaining various artifacts, because of some "garbage" triangles inside the intersection zone and "jagged and ragged" border between outside triangulation and point cloud inside the intersection zone.

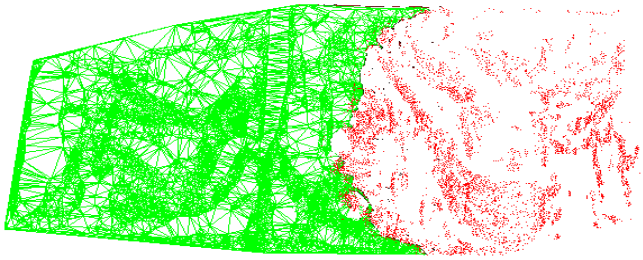


Figure 4: "Outside" triangulation (green) and point cloud inside the intersection zone (red)

In this way, we have to smooth the border and to remove this garbage. Alternately project point cloud onto 2 planes: neighbor triangulation plane (equal to the image plane from which it was extracted) and another plane which has corresponding camera position near the intersection zone. For each projection we will find 2D bounding box. Additionally we will build graph which uses triangles as nodes for each "outside" triangulation. Using the information about triangles that are exactly inside "old" intersection zone, we can make tracing of triangulation to 2D bounding box boundaries and delete traced and garbage triangles inside the intersection zone. Additionally, we will add new points from deleted triangles into point cloud. As a result, the border between outside triangulation and intersection zone will be more "smoothly", so we can avoid some problems with artifacts in triangulation by merging. After this operation we have to delete repeating points from the intersection zone using some kind of criterion based on distance between points for example. If there are few intersection zones, then we must process them separately.

4.3 Building triangulation in the mutual intersection zone

Now we can make triangulation of target point cloud in the intersection zone. 2D Delaunay triangulation is the simplest way. Therefore we must choose the best projection for our 3D point cloud.

It's obviously, that it's not desirable to use real camera projections described in subsubsection 4.2.2, because the point cloud contains points from opposite borders and these boundaries can be greatly distorted in case of using such projections. So, it's necessary to create any synthetic projection, which will be an average between real projections and will not cause huge distortions in the borders geometry.

The solution is based on the next approach. We have to take two nearby positions of cameras and have to calculate any point A,

lying on the view ray $\overline{Vec1}$ one of them (Figure 5). Then we make additional plane using three points: Cam1, Cam2 and A; and project view ray of camera 2 $\overline{Vec2}$ on this plane. Point B is an intersection of this projection and $\overline{Vec1}$. This point is also a center of circle with radius $R = \left|\overline{(Cam1, B)}\right|$. The synthetic position of camera is a point C, which is lying in the middle of circular arc between points Cam1 and Cam2'. The view vector of a new camera is the vector between points B and C, and the up vector is the normal to the plane (Cam1, Cam2, A).

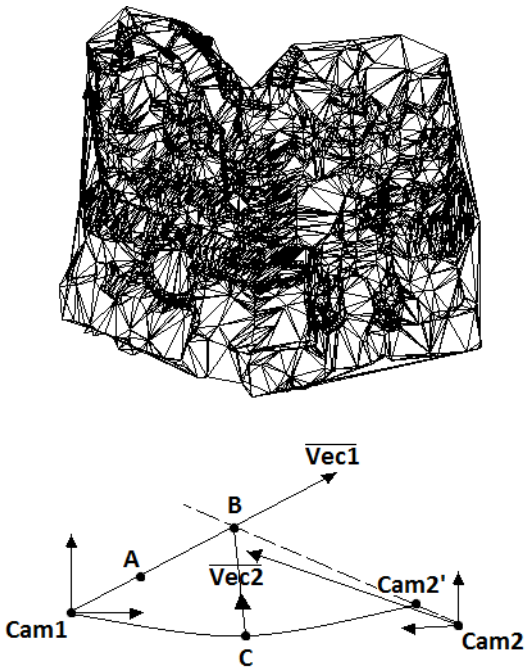


Figure 5: Calculating synthetic camera's position and view vector

After choosing a new projection, we can get 2D coords of point cloud to implement 2D Delaunay triangulation. It should be mentioned, that we have to take borders between outside triangulations and an intersection zone into consideration. In this case, we can use constraint Delaunay triangulation to consider these borders [2]. There is still one remaining issue – Delaunay triangulation is building triangles into convex hull, so there can be some intersections between edges of "old" and "new" triangles. To avoid this problem the intersection of built triangles with part of outside triangulation can be checked.

5. COMPARSION WITH PSR APPROACH

In the section 2 Poisson Surface Reconstruction approach for reconstructing surface from oriented point cloud was mentioned [3]. This approach is implemented for example in a well-known Computational Geometry Algorithms Library CGAL (www.cgal.org). As already mentioned, to perform this algorithm is required to have only points positions and normals, which are easily obtained from our orientation and matching data.

The results of two algorithms are presented in the figure 6 and figure 7. As you can see, our approach preserves the geometry in its initial condition, including the number of polygons, in addition, it executes a little faster (at 50K points proposed algorithm takes 15 s and PSR takes 20 s). To obtain acceptable results by merging DSMs is desirable to have a blunder-free surface. Also, this method does not require dense point cloud.

6. CONCLUSION

An approach for merging DSMs extracted by image matching was proposed in this paper. The proposed method takes advantages of the image orientation and produces accurate and complete surface of an object without artifacts and holes. The proposed method was compared to a PSR approach; advantages in terms of accuracy and initial condition of the reconstructed surface, as well as the computational time have been shown. There is also an example of merging surfaces for the rather complicated closed object (figure 8).

7. FUTURE WORK

It is planned to test this method on a large number of objects, to optimize it and to develop an algorithm for automatic creation of photorealistic textures. This application will be a part of a photogrammetric system, containing modules for automatic orientation of images, automatic matching and reconstruction of surfaces, merging of extracted DSMs and texturing. Consequently, we can develop an effective workflow for automatic building of realistic and precise digital models of existing objects.

8. REFERENCES

[1] Блохинов Ю.Б., Веркеенко М.С.. Алгоритмы построения цифровых трехмерных моделей уникальных объектов, Известия РАН. Теория и системы управления, М., №4, 2011, с.xxx-xxx

[2] Скворцов А.В.. Триангуляция Делоне и её применение, Издательство Томского университета, 2002. 127 с.

[3] Amenta N., Bern M., Kamvysselis M.. A new Voronoi-based surface reconstruction algorithm. Computer Graphics (SIGGRAPH '98) (1998), 415–21.

[4] Carr J., Beatson R., Cherrie H., Mitchel T., Fright W., Mccallum B., Evans T.. Reconstruction and representation of 3D objects with radial basis functions. SIGGRAPH (2001), 67–76.

[5] Edelsbrunner H., Mucke E.. Three-dimensional alpha shapes. TOG (1994), 43–72.

[6] Michael Kazhdan, Matthew Bolitho, Hugues Hoppe. Poisson Surface Reconstruction, Eurographics Symposium on Geometry Processing, 2006.

[7] Thomas R. Hudson. Merging VRML models: extending the use of photomodeller, University of Virginia, 1998.

[8] Zhang L.. Automatic Digital Surface Model Generation from Linear Array Images // DISS.ETH NO. 16078. Zurich, 2005. 219 p.

About the author

Verkeenko Mikhail – Ph.D. student at State Research Institute of Aviation Systems.

Email: envenom.gm@gmail.com

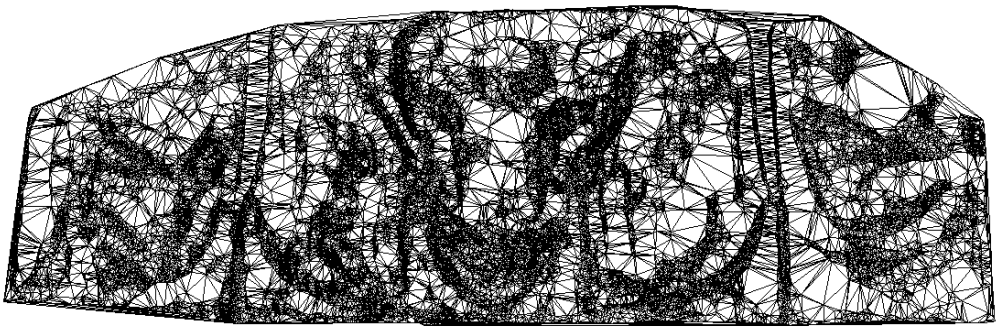


Figure 6: Result of proposed approach

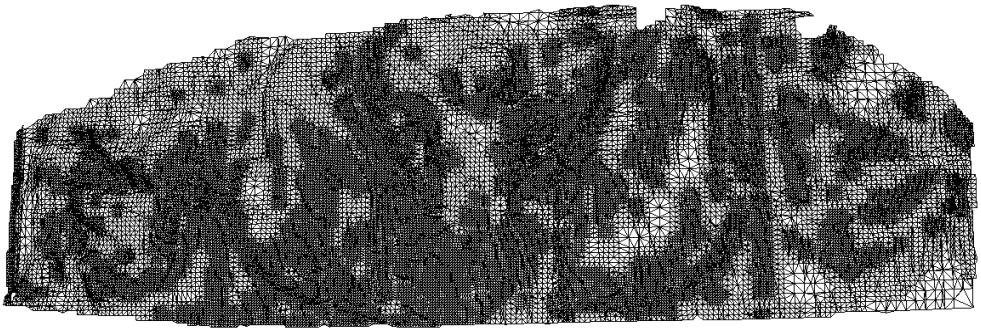


Figure 7: Result of Poisson surface reconstruction

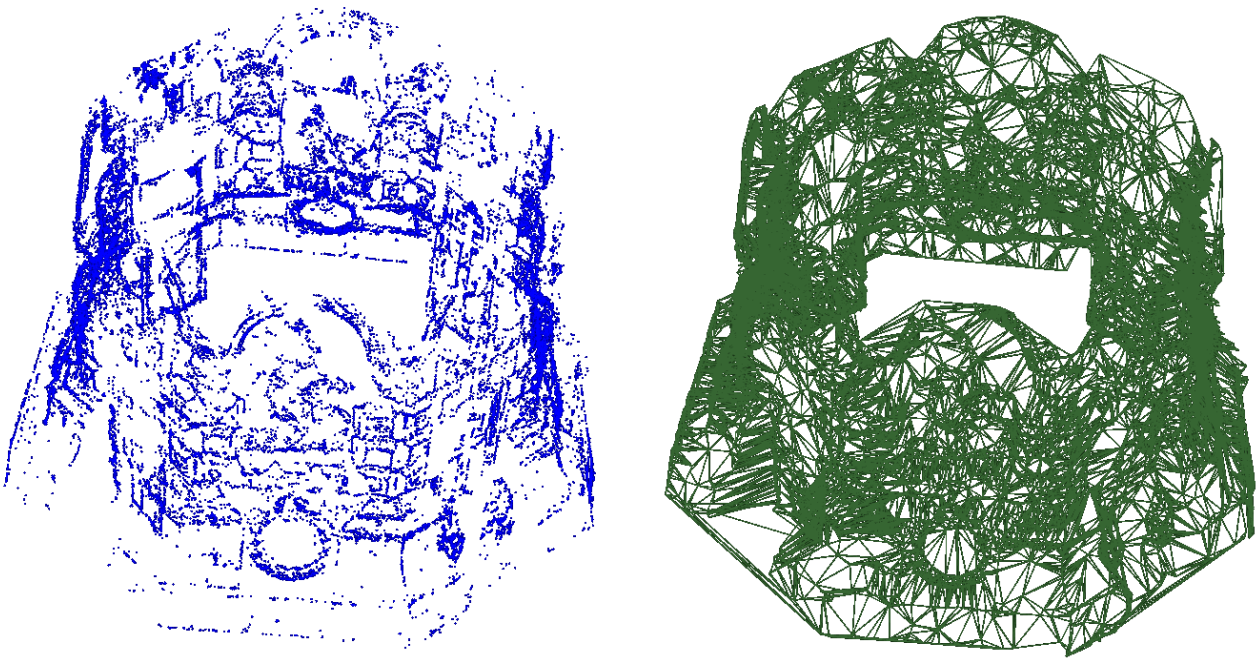


Figure 8: Point cloud and triangulation of the object “Monument to the Heroes of Plevna”

Фильтрация Цифровых Дактилоскопических Изображений

Гудков Владимир Юльевич
Челябинский Государственный Университет, Челябинск, Россия
Diana@Sonda.ru

Аннотация

В работе предлагается новый метод фильтрации дактилоскопических изображений, устойчивый к изменению ширины линий и просветов, рыхлости линий и другим дефектам изображения. Результатом фильтрации является бинарное изображение, по скелету которого определяют шаблон, используемый при идентификации личности.

Ключевые слова: *Отпечаток пальца, килевидый фильтр, дактилоскопическое изображение.*

1. ВВЕДЕНИЕ

В компьютеризированных системах (КС) идентификация изображений обычно выполняется после их обработки [1, 2]. Хотя преобразование Фурье является основой методов обработки изображений в частотной области, применение рядов Фурье для функций с локальными особенностями неэффективно [3, 4]. Связано это с тем, что базисная функция рядов Фурье – синусоида – является по природе гладкой и строго периодической функцией, определенной на бесконечности. Она фактически не в состоянии описывать произвольные сигналы. Вследствие этого возникла потребность в функциях, называемых вейвлетами [2, 4, 5]. Однако для фильтрации дактилоскопических изображений (ДИ) обычно применяют фильтр Габора [6]. К сожалению, он имеет недостатки, определяемые особенностями ДИ: изменяющийся период смежных линий, изменяющаяся ширина просветов между линиями и кривизна линий приводят к ошибкам фильтрации. На рис. 1 выделены окрестности общих признаков (завитка и дельты) и флексорной складки, в которых наблюдаются перечисленные особенности ДИ. Это приводит к увеличению ошибок идентификации личности [6] и вынуждает разработчиков КС применять новые методы фильтрации [7, 8].

Рис. 1 Изображение отпечатка пальца

2. ПОСТАНОВКА ЗАДАЧИ

Изображение как множество действительных чисел формируют в виде $F=\{f(x,y)\,|\,(x,y)\in X\times Y\}$ в прямоугольной области G мощностью $|G|=x_0y_0$, где $X=0..x_0-1$ и $Y=0..y_0-1$. Структурно обработку изображения представляют в виде пирамиды $\Re$ слоев из взаимосвязанных иерархий [5]. Сегментация l-го слоя k-й иерархии $F_k^{(l)}$ разбивает слой на x_hy_h непересекающихся квадратных сегментов $S_{hk}^{(l)}(x,y)$ с длиной стороны 2^{h-k} и вершинами $(x,y)\in X_h\times Y_h$, где $k<h$ и h – номер иерархии; $X_h=0..x_h-1$ и $Y_h=0..y_h-1$. Доступ к каждой точке сегмента $S_{hk}(x,y)$ записывают в координатах $(u,v)\in\overline{X}_{hk}\times\overline{Y}_{hk}$:

$$\begin{cases}\overline{X}_{hk}=&\{u+x2^{h-k}\,|\,x\in X_h\wedge u\in 0..2^{h-k}-1\},\\ \overline{Y}_{hk}=&\{v+y2^{h-k}\,|\,y\in Y_h\wedge v\in 0..2^{h-k}-1\}.\end{cases}\quad(1)$$

Размер области h-й иерархии: $x_h=\left\lceil\frac{x_0}{2^h}\right\rceil$ и $y_h=\left\lceil\frac{y_0}{2^h}\right\rceil$, где $\lceil a\rceil$ – наименьшее целое число, превышающее вещественную величину a. Для формализации методов классификационного анализа (КА) применяют аппарат апертур. Прямолинейные щелевые $A_h(x,y,\alpha,w)$ и $A_h^-(x,y,\alpha,w)$ апертуры как множества точек и углов в виде элементов упорядоченных троек (u,v,β) определяют по формулам:

$$\begin{cases}A_h(x,y,\alpha,w)=\{(u,v,\beta)=(x+\rceil w_x\lceil,y+\rceil w_y\lceil,\beta)\,|\,w\in Z_w\},\\ A_h^-(x,y,\alpha,w)=\{(u,v,\beta)=(x+\rceil w_x\lceil,y+\rceil w_y\lceil,\beta)\,|\,w\in Z_w^-\},\end{cases}\quad(2)$$

где $w_x=w\cos(\alpha)$ и $w_y=w\sin(\alpha)$; $(x,y)\in X_h\times Y_h$ – центр апертуры; $(u,v)\in X_h\times Y_h$ – точка апертуры; w – размер апертуры; множества $Z_w=1..w$ и $Z_w^-=-w..-1\cup 1..w$; α – угол направления апертуры; $\rceil\alpha\lceil$ – ближайшая целая часть вещественного числа a. Угол, определяющий направление из центра (x,y) в точку (u,v) апертуры, находят в виде

$$\beta=\operatorname{arctg}\left(\frac{v-y}{u-x}\right)+\pi n\ \text{ при }\ n\in 0..1\,.$$

Задача фильтрации опирается на специальные слои признаков, вычисленные на предшествующих стадиях обработки ДИ: качества $\mathrm{Q}_h=[q_h(x,y)]$, меток сегментации $\mathrm{C}_h=[c_h(x,y)]$, периодов линий $\mathrm{T}_h=[t_h(x,y)]$, направлений $\Theta_h=[\theta_h(x,y)]$ и величин $\mathrm{V}_h=[v_h(x,y)]$ кривизны линий. Иерархия h пирамиды $\Re$ определяется алгоритмом сегментации.

3. ФИЛЬТРАЦИЯ

Изображение итерационно дифференцируют, сглаживают, а в финале бинаризуют. Для точек одного сегмента значения признаков, считываемых с вершины сегмента, рассматривают как константы. Обработку изображения в окрестности петель, дельт и завитков специализируют. Инициализация: $F_0^{(p)}=\{f_0^{(p)}(x,y)\}=F$, $F_0^{(b)}=\{f_0^{(b)}(x,y)\}=[0]$, где $F_0^{(p)}$ – слой фильтрованного ДИ; $F_0^{(b)}$ – слой бинарного ДИ. Номер итерации $j=1$.

3.1 Дифференцирование

Дифференцирование заключается в вычислении одномерной свертки, ориентированной по направлению кривизны линий, в точках оснований сегментов $\{S_h\}$. Введем набор углов дезориентации

$$A=\{\alpha,\alpha+\beta,\alpha-\beta\} \tag{3}$$

с опорным углом $\alpha=\theta_h(x,y)$, где $\theta_h(x,y)$ –направление кривизны линий, и углом отклонения в виде

$$\beta=\begin{cases}\beta_{\max}\dfrac{\varepsilon-d}{\varepsilon}, & \text{если } d\le\varepsilon,\\ 0, & \text{иначе.}\end{cases}$$

Здесь $d=\sqrt{(x-a)^2+(y-b)^2}$ – расстояние от отсчета $(x,y)\in X\times Y$ до ближайшего общего признака с координатами $(a,b)\in X\times Y$; ε – заданная окрестность общего признака; $\beta_{\max}$ – наибольший угол отклонения апертур (до 60°). Вне окрестностей общих признаков $\beta=0$.

Дифференцирование информативных сегментов с метками $c_h(x,y)\in\{1\}$ и качеством $q_h(x,y)\le lev(j,q_{\max})$ сводится к расчету по формуле

$$f_0^{(b)}(x,y)=\begin{cases}f\Big(\max\limits_{\alpha\in A} f^{\alpha}(x,y)\Big), & \text{если } \min\limits_{\alpha\in A} f^{\alpha}(x,y)>0,\\ f\Big(\min\limits_{\alpha\in A} f^{\alpha}(x,y)\Big), & \text{если } \max\limits_{\alpha\in A} f^{\alpha}(x,y)<0,\\ f\Big(\operatorname{med}\limits_{\alpha\in A} f^{\alpha}(x,y)\Big), & \text{иначе,}\end{cases} \tag{4}$$

где $f^{\alpha}(x,y)$ – свертка в направлении $\alpha\in A$ по (3) в виде

$$f^{\alpha}(x,y)=\sum_{i=-w}^{w}h(i)\,\xi^{\alpha}(i);$$

элементы набора $\{\xi^{\alpha}(i)\}$ формируют перечислением величин, последовательно собираемых в отсчетах (u,v) прямолинейной щелевой апертуры по (2) в виде

$$\{\xi^{\alpha}(i)\}=\{f_0^{(p)}(u,v)\,|\,(u,v)\in A_0^{-}(x,y,\alpha,w)\vee(x,y)\}; \tag{5}$$

размер апертуры $w=\rceil k_1\,t_h(x,y)[$ ($k_1=0{,}7$ в реализации); $(x,y)\in F_0^{(p)}$ – центр апертуры; $lev(.,.)$ – функция уровня качества (рис. 2), зависящая от номера итерации j и лучшей оценки качества $q_{\max}=\max\limits_{(x,y)\in X_h\times Y_h} q_h(x,y)$.

Ядро свертки для значений $t=\rceil k_2\,t_h(x,y)[$ ($k_2=0{,}5$ в реализации), показанное на рис. 3, вычисляют по формуле

$$\mathbf{H}=[h(i)]=\left[J_1^{-1}(i,m)-J_2^{-1}(i,n)\right], \tag{6}$$

где $-w\le i\le w$ – индекс элемента ядра свертки; m,n – обучаемые параметры обратных функционалов; функционалы

$$J_1(i,m):\begin{cases}\left\{k_3\cos^m\left(90\dfrac{i}{t}\right)-\bar{h}\right\}, & \text{если } -t\le i\le t,\\ \{0\}, & \text{иначе}\end{cases}\to\{m\}$$

и

$$J_2(i,n):\begin{cases}\left\{k_4\cos^n\left(90\dfrac{|i|-t}{t}\right)-\bar{h}\right\}, & \text{если } -t\le i\le t,\\ \left\{k_5\cos^n\left(90\dfrac{|i|-t}{t}\right)-\bar{h}\right\}, & \text{иначе}\end{cases}\to\{n\};$$

коэффициенты $k_3>k_4>k_5$. Коэффициенты ядра свертки (6) рассматривают как центрированную случайную величину, получаемую вычитанием средней арифметической

$$\bar{h}=\frac{1}{2w+1}\sum_{i=-w}^{w}h(i).$$

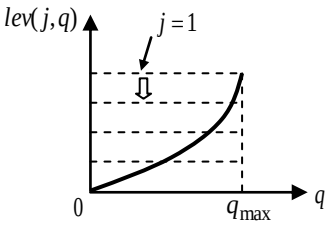


Рис. 2. Функция уровня качества

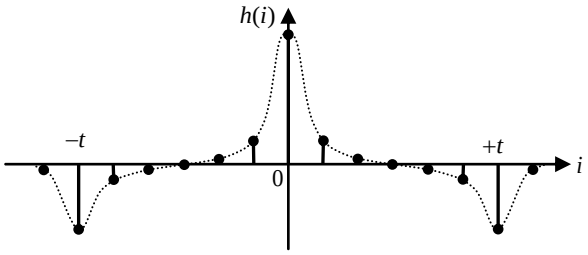


Рис. 3. Маска дифференциального фильтра

Функция уровня качества (см. рис. 2) в первой итерации разрешает фильтрацию всех информативных сегментов (верхняя пунктирная линия). С ростом номера j итерации ее значение уменьшается, а фильтрацию повторяют только для сегментов с качеством $q_h(x,y)\le lev(j,q_{\max})$ (пунктирные линии смещают вниз). При этом маска фильтра цепляется за структуру изображения в соседних сегментах, что помогает лучше фильтровать дефектные сегменты [6].

Дифференциальный фильтр (см. рис. 3) детектирует фазу сигнала, а амплитуда отклика пропорциональна локальной контрастности ДИ. Маска фильтра обеспечивает вычитание взвешенных значений боковых отсчетов из взвешенных значений центральных отсчетов. Коэффициенты ядра свертки резко возрастают в центре апертуры и на расстоянии в половину периода от центра. Отклик фильтра усиливается для малых и ослабляется для больших значений сигнала. Это

приводит к уменьшению зависимости величины отклика фильтра от величины локального контраста изображения. Она еще более уменьшается в схеме итерационной фильтрации. На темных линиях отклик фильтра отрицателен, на просветах – положителен, а на ровном фоне – равен нулю. За счет килевидной формы маски фильтр корректно обрабатывает узоры с изменяющейся шириной линий и просветов, с размытыми краями линий, с рыхлыми линиями. Амплитуда отклика фильтра, в отличие от фильтра Габора [4, 7], устойчива.

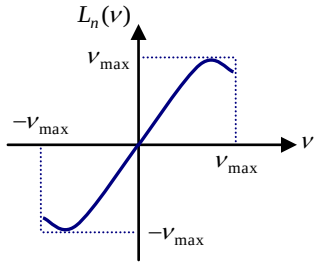


Рис. 4. Функция усиления откликов

На рис. 4 показана функция усиления $f(.)$, используемая в формуле (4). Ее находят как интерполяционный многочлен L_n степени n при $L_n(0) \equiv 0$, который масштабируют наибольшим откликом фильтра

$$v_{\max} = \max_{(x,y)\in X\times Y} \sum_{i=-w}^{w} h(i)\,\xi^{\alpha}(i)$$

при $\alpha = \theta_h(x,y)$ и $\{\xi^{\alpha}(i)\}$ по (5). Усиление слабых и ослабление сильных сигналов позволяет аккумулировать отклики фильтра вблизи «горочки», отодвигая их от зоны насыщения. Так как края апертуры цепляются за соседние более хорошие сегменты, а величины откликов фильтра $f(.)$ для различных сегментов соизмеримы, линии прогнозируют, а не копируют по подобию соседних сегментов (как при забеливании линии). Чем больше номер итерации j, тем сильнее эффект прогнозирования перпендикулярно линиям. Фактически фильтр меняет размерность, а в каждом блоке такого составного фильтра применяют уникальный период $t_h(x,y)$ (посегментно). Интерполяционный многочлен L_n обучают по качеству распознавания частных признаков.

На рис. 5 показаны апертуры для различных углов дезориентации из набора A по (3). Для отсчетов, попадающих в окрестность ε общих признаков, углы из набора A расходятся. Вблизи центра окрестности щелевые апертуры распределяются по окружности и понятие ориентации линий фактически исчезает. Эксперименты показали, что такой подход себя оправдывает. Вне окрестности ε равенство углов из A используют для ускорения процедуры.

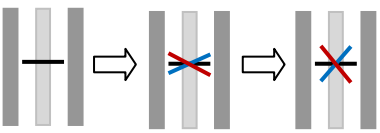


Рис. 5. Дезориентация щелевых апертур

Дифференциальный фильтр рассчитывают как составной блочный фильтр с переменными параметрами: коэффициентом усиления, величиной периода, кривизной и механизмом, компенсирующим изменение скважности сигнала. Это сильно отличает дифференциальный фильтр от фильтра Габора [6] и Гаусса-Эрмита [3].

3.2 Сглаживание

Сглаживание в основаниях информативных сегментов с метками $c_h(x,y) \in \{1\}$ и качеством $q_h(x,y) \le lev(j, q_{\max})$ (см. рис. 2) сводится к применению сглаживающего фильтра по формуле

$$f_0^{(p)}(x,y) = \begin{cases} \max\limits_{\alpha\in A} f^{\alpha}(x,y), & \text{если } \operatorname{med}\limits_{\alpha\in A} f^{\alpha}(x,y) > 0, \\ \min\limits_{\alpha\in A} f^{\alpha}(x,y), & \text{если } \operatorname{med}\limits_{\alpha\in A} f^{\alpha}(x,y) < 0, \\ f^{\alpha}(x,y), & \text{иначе,} \end{cases} \qquad (7)$$

где $f^{\alpha}(x,y)$ – одномерная свертка для направления $\alpha \in A$ по (3) или $\alpha = \theta_h(x,y)$ в виде

$$f^{\alpha}(x,y) = \mathbf{H} * \Xi_0^{(\alpha)}(x,y)$$

с ядром свертки $\mathbf{H}$ и нормой $\|\mathbf{H}\|$ как суммой ее элементов; набор для отсчета (x,y) состоит из элементов вида

$$\Xi_0^{(\alpha)}(x,y) = \left\{\xi_0^{(\alpha)}(u,v)\right\}.$$

Элементы набора $\Xi_0^{(\alpha)}(x,y)$ считывают из слоя $F_0^{(b)}$ прямолинейной щелевой апертурой по (2) в виде

$$\{\xi_0^{(\alpha)}(u,v)\} = \{f_0^{(b)}(u,v)\},$$

где $(u,v) \in A_0(x,y,\alpha_0,w) \vee A_0(x,y,\alpha_1,w) \vee (x,y)$; направления апертур для угла $\alpha \in A$ по (3) определяются направлением и величиной кривизны в виде

$$\alpha_0 = \alpha + 90 - \frac{1}{\kappa} v_h(x,y) \quad \text{и} \quad \alpha_1 = \alpha - 90 + \frac{1}{\kappa} v_h(x,y);$$

$v_h(x,y)$ – величина кривизны; κ – масштабный коэффициент кривизны (3–5 в реализации). Размер апертуры

$$w = \rceil k_6\, t_h(x,y) \lceil$$

определяется периодом $t_h(x,y)$ и коэффициентом k_6, причем $2w+1 \le t_h(x,y)$.

Элементы для набора $\Xi_0^{(\alpha)}(x,y)$ выбираются вдоль ломаной линии длиной $2w+1$, ориентация и угол излома которой определяются направлением и величиной кривизны. В ядре свертки используется число элементов, не превышающее значение периода. Вектор $\mathbf{H}$ может состоять из единиц или подчеркивать центральные элементы апертуры по экспоненциальному закону. Сортировка элементов здесь не применяется.

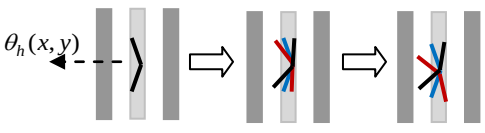


Рис. 6. Расхождение щелевых апертур

На рис. 6 показано расхождение углов ориентации щелевых апертур при движении к центру окрестности ε общих признаков, внутри которой углы из набора A по (3) расходятся. В центре окрестности понятие ориентации линий фактически исчезает. Вне окрестности ε равенство углов из A используют для ускорения процедуры.

В первой итерации в слой $F_0^{(p)}$ фильтруют все величины отсчетов из слоя $F_0^{(b)}$. На последующих итерациях края апертуры цепляются за соседние более хорошие сегменты. Происходит сглаживание вдоль линий и их прогноз. Чем больше номер итерации j, тем дальше прогнозируются линии по их длине с учетом их кривизны. Фактически фильтр меняет размерность, а в каждом блоке такого составного фильтра применяют период $t_h(x,y)$ (посегментно). Коэффициенты и ядро свертки **H** фильтра обучают по качеству распознавания частных признаков.

Номер итерации инкрементируют, дифференцирование по (4) и сглаживание по (7) повторяют, пока $j \le j_{\max}$.

3.3 Бинаризация

Бинаризация в основаниях всех информативных сегментов с метками $c_h(x,y) \in \{1\}$ сводится к применению дифференциального фильтра в модифицированном виде

$$f_0^{(b)}(x,y) = \begin{cases} b, & \text{если } \sum_{\alpha \in A} f^{\alpha}(x,y) \ge 0, \\ 0, & \text{иначе} \end{cases}, \qquad (8)$$

где $f^{\alpha}(x,y)$ – одномерная свертка для направления $\alpha \in A$ по (3) в виде

$$f^{\alpha}(x,y) = \sum_{i=-w}^{w} h(i)\,\xi^{\alpha}(i);$$

элементы набора $\{\xi^{\alpha}(i)\}$ формируют перечислением величин, собираемых последовательно в отсчетах (u,v) прямолинейной щелевой апертуры размером w по (5); множество $\{b,0\}$ задает величину яркости просветов и линий; ядро свертки **H** определяют по (6) (см. рис. 3).

На этом фильтрация ДИ заканчивается.

Итак, фильтрация выполняется на основе разделимой импульсной характеристики последовательным применением дифференциального и сглаживающего фильтров. Разнесение составных частей импульсной характеристики по этапам процедуры уменьшает количество отсчетов, задействованных в свертке, и в среднем не превышает числа 40 (для сравнения фильтр Габора размерностью 33×33 считывает 1089 точек [4, 6]). Временные характеристики фильтрации существенно улучшаются. В целом две части фильтра образуют составной блочный фильтр с параметрами: коэффициентом усиления, величиной периода, размером апертуры, направлением и величиной кривизны, механизмом, компенсирующим изменение скважности сигнала в локальном профиле изображения [1]. В окрестности общих признаков фильтр специализируют. Это дает преимущества КА в скорости, качестве и помехоустойчивости, недостижимые при использовании фильтра Габора. Первое дифференцирование ДИ и бинарное ДИ показаны на рис. 7.

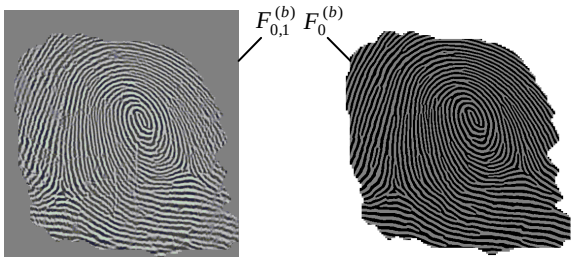


Рис. 7. Первое дифференцирование и бинарный слой

4. ЗАКЛЮЧЕНИЕ

Предложен итерационный метод килевидной фильтрации ДИ, который в каждом сегменте опирается на оценку качества, период линий, направление и величину кривизны линий. Фильтрация развивается от «хороших» сегментов к «плохим», многократно на них повторяясь, а в окрестности общих признаков она специализируется. При синтезе разделимого импульсного отклика фильтра использованы следующие новые решения: 1) дифференциальный фильтр с килевидной характеристикой по уравнениям (3-6); 2) сглаживающий фильтр по уравнению (7); 3) фильтр бинаризации по (8), учитывающий только знак отклика дифференциального фильтра с килевидной характеристикой.

В дальнейшем планируется исследование задачи синтеза фильтра с маской, изогнутой по кривой 3 порядка.

5. ССЫЛКИ

1. Обработка и анализ изображений в задачах машинного зрения / Ю.В. Визильтер, С.Ю.Желтов, А.В.Бондаренко и др. – М.: Физматкнига, 2010. – 672 с.

2. Гонсалес, Р. Цифровая обработка изображений / Р. Гонсалес, Р. Вудс; пер. с англ.; под ред. П.А. Чочиа. – М.: Техносфера, 2006. – 1072 с.

3. Крылов А.С. Метод моментов Гаусса-Эрмита для анализа изображений отпечатков пальцев / А.С. Крылов, Е.В. Лазарева, О.С. Ушмаев // Графикон: тр. конф. – М.: МГУ, 2008. – с. 31.

4. Новейшие методы обработки изображений / А.А. Потапов, А.А. Пахомов, С.А. Никитин, Ю.В. Гуляев. – М.: Физматлит, 2008. – 496 с.

5. Яне, Б. Цифровая обработка изображений / Б. Яне; пер. с англ. А.М. Измайловой. – М.: Техносфера, 2007. – 584 с.

6. Maltoni D. Handbook of fingerprint recognition / D. Maltoni, D. Maio, A.K. Jain, S. Prabhakar. – London: Springer-Verlag, 2009. – 496 p

7. Pat. 7194115 USA, Int. Cl. G06K 9/00, G05B 19/00. Fingerprint identification method and apparatus / U. Kaoru (Japan); NEC corporation. – Field: July. 26, 2001; Date of patent: Mar. 20, 2007; U.S.Cl. 382/124. – 26 p.

8. Pat. 7194393 USA, Int. Cl. G06F 17/10. Numerical model for image feature extraction / W. Xiangshu (Calif.), H. Ming (Calif.); Cogent Systems (Calif.). – Field: Jan. 19, 2006; Date of patent: Mar. 20, 2007; U.S.Cl. 703/2. – 16 p.

Об авторах

Гудков Владимир Юльевич – доцент Челябинского государственного университета. Его адрес: Diana@Sonda.ru.

Определение локальных сдвигов изображений радужных оболочек глаз методом проекционной фазовой корреляции

Е.А. Павельева, А.С. Крылов
Лаборатория Математических Методов Обработки Изображений
Факультет Вычислительной Математики и Кибернетики
Московский Государственный Университет имени М.В. Ломоносова, Москва, Россия
E-mail: paveljeva@yandex.ru, kryl@cs.msu.ru

Аннотация

В статье предложен метод проекционной фазовой корреляции, в котором спектральные характеристики изображений вычисляются на основе разложения функций интенсивности изображений в ряд по функциям Эрмита. При этом в спектральной области отсутствует эффект Гиббса, связанный с периодичным продолжением изображения при подсчете спектра изображения через дискретное преобразование Фурье, что особенно важно в случае входных изображений небольших размеров. Метод применен в задаче идентификации человека по радужной оболочке глаза для определения локальных сдвигов изображений.

Ключевые слова: *радужная оболочка глаза, фаза, функции Эрмита.*

1. ВВЕДЕНИЕ

Фаза изображения несет достаточно много информации [1]. Метод фазовой корреляции изображений (POC, Phase-Only Correlation) [2] позволяет одновременно определить "похожесть" и сдвиги изображений. Для пары изображений исследуется поведение разности фаз этих изображений. Фаза изображения вычисляется через дискретное преобразование Фурье, что подразумевает периодическое продолжение изображений, что приводит к систематическим ошибкам вычислений (эффект Гиббса).

В данной работе предложен метод проекционной фазовой корреляции изображений, в котором преобразование Фурье изображения вычисляется с использованием разложения функции интенсивности изображения в ряд по функциям Эрмита (собственным функциям преобразования Фурье). Периодического продолжения изображений не происходит, что позволяет уменьшить ошибки, возникающие в методе фазовой корреляции. Метод проекционной фазовой корреляции имеет преимущество на изображениях небольших размеров, что позволяет применять этот метод для определения локальных сдвигов изображений.

2. МЕТОД ФАЗОВОЙ КОРРЕЛЯЦИИ

Рассмотрим две функции $f(x,y)$ и $g(x,y)$, определенные в прямоугольнике $[0,M]\times[0,N]$. Пусть $F(u,v)$ и $G(u,v)$ - их преобразования Фурье:

$$F(u,v)=\frac{1}{\sqrt{MN}}\sum_{x=0}^{M}\sum_{y=0}^{N}f(x,y)\,e^{-\frac{i2\pi ux}{M}}e^{-\frac{i2\pi uy}{N}}=A_F(u,v)e^{i\varphi_F(u,v)},$$

где A_F - модуль преобразования Фурье, а φ_F - его фаза.

Взаимным фазовым спектром (cross-phase spectrum) двух спектральных функций $F(u,v)$ и $G(u,v)$ называется функция $R_{FG}(u,v)=\dfrac{F(u,v)\,\overline{G(u,v)}}{\left|F(u,v)\,\overline{G(u,v)}\right|}=e^{i(\varphi_F(u,v)-\varphi_G(u,v))}$, т.е. спектральная функция с единичным модулем, фаза которой равна разности фаз функций. Обратное преобразование Фурье является фазовой корреляцией (POC-функцию) [2]:

$$POC_{fg}(x,y)=\frac{1}{\sqrt{MN}}\sum_{u=0}^{M}\sum_{v=0}^{N}R_{FG}(u,v)e^{\frac{i2\pi ux}{M}}e^{\frac{i2\pi vy}{N}}.$$

Если $g(x,y)=f(x-a,y-b)$, т.е. функции одинаковы с точностью до смещения, тогда $G(u,v)=e^{-i2\pi(\frac{ua}{M}+\frac{vb}{N})}F(u,v)$. POC-функция является дельта-функцией с пиком в точке (a,b). Метод фазовой корреляции опирается на следующее свойство: если две функции $f(x,y)$ и $g(x,y)$ "похожи", то их POC-функция имеет выраженный пик. Высота пика определяет меру похожести функций, а положение пика соответствует относительному смещению двух функций.

Метод фазовой корреляции применяется в различных задачах обработки изображений. В том числе и в задаче идентификации по радужной оболочке глаза [3]. При этом показано, что если брать обратное преобразование Фурье не от всего спектрального сигнала, а только от его части, соответствующей низким частотам, то пик POC-функции получается более четким и устойчивым (рис. 1). Под частью спектрального сигнала (в %) в работе подразумеваются частоты, значения амплитудного спектра в которых составляют данный процент от всей спектральной энергии. При этом высокие частоты обнуляются.

3. МЕТОД ПРОЕКЦИОННОЙ ФАЗОВОЙ КОРРЕЛЯЦИИ

В работе предложен метод проекционной фазовой корреляции, основанный на проекционном методе Эрмита [4]. Исходные функции раскладываются в ряд по функциям Эрмита $\psi_n(x,y)$ – собственным функциям непрерывного преобразования Фурье.

(a)

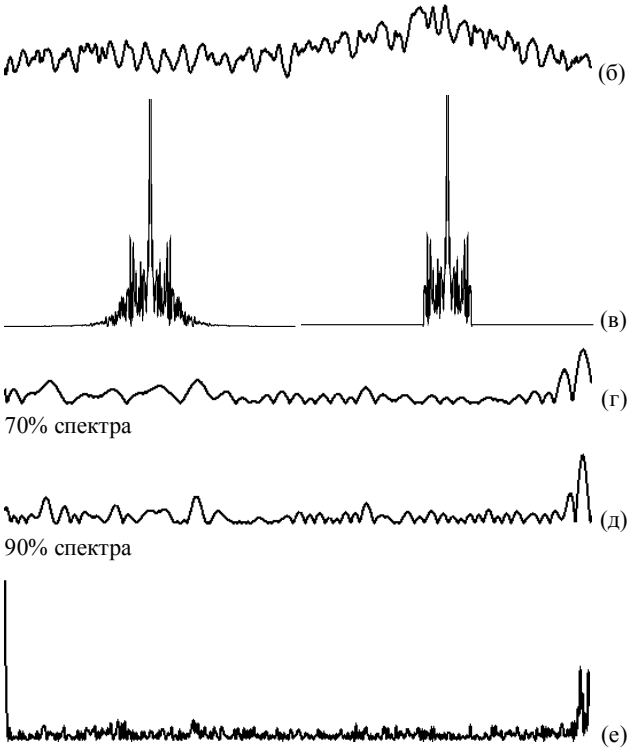


Рисунок 1. Пример одномерных РОС-функций при различной обрезке амплитудного спектра. (а), (б) исходные функции, (в) слева – амплитудный спектр функции (а), справа – урезанный амплитудный спектр (85% спектра), (г)-(е) РОС-функции, вычисленные после различного обнуления высоких частот.

Одномерные функции Эрмита определяются

$$\psi_n(x) = \frac{1}{\sqrt{2^n \cdot n! \cdot \sqrt{\pi}}} \cdot e^{-x^2/2} \cdot H_n(x),$$

$$H_0(x) = 1, \quad H_1(x) = 2 \cdot x,$$

$$H_n(x) = 2 \cdot x \cdot H_{n-1}(x) - 2 \cdot (n-1) \cdot H_{n-2}(x).$$

Двумерные функции Эрмита представимы в виде произведения одномерных:

$$\psi_{m,n}(x,y) = \psi_m(x) \cdot \psi_n(y).$$

Функции Эрмита $\psi_{m,n}(x,y)$, $m,n = \overline{0,\infty}$ образуют полную ортонормированную систему в $L_2(R^2)$. Пусть исследуемые функции $f(x,y)$ и $g(x,y)$ заданы в прямоугольнике $[-A,A]\times[-B,B]$, продолжим их нулем.

Каждая функция Эрмита $\psi_n(x)$ локализована на отрезке $[-\sqrt{2n+1}, \sqrt{2n+1}]$ (на этом отрезке расположены все перегибы $\psi_n(x)$ и $\psi_n(x) \underset{x\to\infty}{\to} 0$).

Для приближения функции $f(x,y)$ линейной комбинацией функций Эрмита $\psi_{ij}(x,y)$, $i = \overline{0,\mathrm{m}}$, $\mathrm{j} = \overline{0,\mathrm{n}}$ растянем все используемые функции Эрмита по оси x в $k = A/\sqrt{2m+1}$ раз и по оси y в $l = B/\sqrt{2n+1}$ раз (чтобы функция Эрмита с максимальным номером (m,n) была локализована в том же прямоугольнике, что и функция $f(x)$). Получим:

$$f(x,\mathrm{y}) \approx \tilde{f}(x,\mathrm{y}) = \sum_{i=0}^{m}\sum_{j=0}^{n} c_{ij}\,\psi_{ij}(x,\mathrm{y}) = ,$$

$$= \sum_{\mathrm{i}=0}^{m} \psi_i(\mathrm{x}) \sum_{j=0}^{n} c_{ij}\,\psi_j(y) = \sum_{i=0}^{m} c_i(y)\,\psi_i(x)$$

где коэффициенты Эрмита

$$c_{i,j} = \iint f(x,y)\,\psi_{i,j}(x,y)dxdy =$$

$$= \int \psi_j(y)dy \int f(x,y)\,\psi_i(x)dx = \int c_i(y)\psi_j(y)dy.$$

Далее для приближенных функций $\tilde{f}(x,y)$ и $\tilde{g}(x,y)$ применяется метод фазовой корреляции. Однако, вместо дискретного преобразования Фурье, используется непрерывное преобразование Фурье. Функции Эрмита являются собственными функциями непрерывного преобразования Фурье с собственными значениями $\pm 1, \pm i$: $F(\psi_n) = (-i)^n \psi_n$. Следовательно, аппроксимация преобразования Фурье имеет следующий вид:

$$F[\tilde{f}] = F\left[\sum_{i=0}^{m}\sum_{j=0}^{n} c_{ij}\,\psi_{ij}(x,\mathrm{y})\right] = \sum_{i=0}^{m}\sum_{j=0}^{n} c_{ij}\,F\left[\psi_{ij}(x,\mathrm{y})\right] = ,$$

$$= \sum_{i=0}^{m}\sum_{j=0}^{n} c_{ij}(-i)^{i+j}\psi_{ij}(x,\mathrm{y})$$

т.е. вещественная Re_F и мнимая Im_F части преобразования Фурье быстро считаются через коэффициенты Эрмита. Фаза φ_F вычисляется как $\varphi_F = arctg\dfrac{\mathrm{Im}_F}{\mathrm{Re}_F}$.

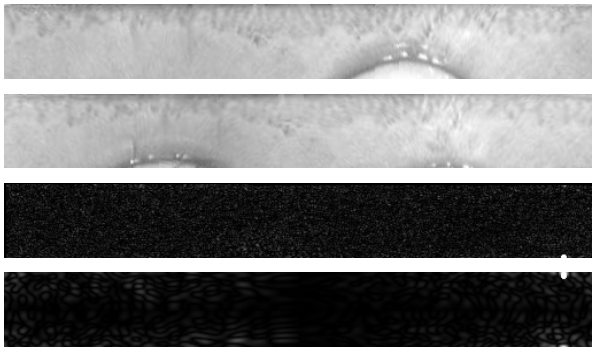
Далее вычисляется функция взаимного фазового спектра $R_{FG}(u,v)$, и для подсчета РОС-функции $POC_{fg}(x,y) = F^{-1}[R_{fg}(u,v)]$ используется свойство вычисления исходной функции через дважды примененное преобразование Фурье: $f(x,y) = F[F(-u,-v)]$. Таким образом, раскладывая функцию $R_{FG}(-u,-v)$, симметричную функции взаимного фазового спектра $R_{FG}(u,v)$, в ряд по тем же функциям Эрмита $\psi_{ij}(x,y)$, $i = \overline{0,\mathrm{m}}$, $\mathrm{j} = \overline{0,\mathrm{n}}$ и считая преобразование Фурье от полученного разложения, получается аналог искомой РОС-функции, называемый нами HPPOC-функцией (Hermite Projection Phase-Only Correlation):

$$HPPOC_{fg}(x,y) = \mathrm{F}[R_{FG}(-u,-v)].$$

4. ПРИМЕНЕНИЕ МЕТОДА ПРОЕКЦИОННОЙ ФАЗОВОЙ КОРРЕЛЯЦИИ В ЗАДАЧЕ ИДЕНТИФИКАЦИИ ПО РАДУЖНОЙ ОБОЛОЧКЕ ГЛАЗА

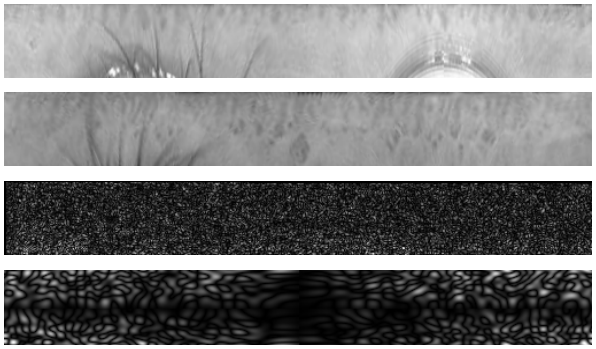
На стадии предобработки [5] на изображении глаза выделяется радужная оболочка, которая переводится в прямоугольное нормализованное изображение. Угол

поворота одного глаза относительно другого соответствует циклическому сдвигу одного нормализованного изображения относительно другого. Будем искать сдвиг методами POC и HPPOC. В случае изображений одной радужной оболочки (рис. 2), пики POC и HPPOC-функций выражены четко, при этом позиция пика соответствует сдвигу второго изображения относительно первого (повороту второго глаза относительно первого), а высота пика определяет меру похожести функций. В случае изображений разных радужных оболочек (рис. 3) нет четкого пика POC и HPPOC-функций.



(а) (б) (в) (г)

Рисунок 2. (а), (б) нормализованные изображения глаз одного человека, (в) их POC-функция, (г) их HPPOC-функция. Изображения взяты из базы данных CASIA-IrisV3 [6].

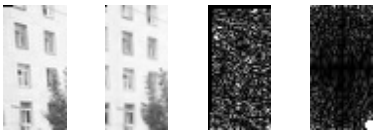


(а) (б) (в) (г)

Рисунок 3. (а), (б) нормализованные изображения глаз разных людей, (в) их POC-функция, (г) их HPPOC-функция.

5. СРАВНЕНИЕ МЕТОДОВ ФАЗОВОЙ КОРРЕЛЯЦИИ И ПРОЕКЦИОННОЙ ФАЗОВОЙ КОРРЕЛЯЦИИ

Метод фазовой корреляции использует дискретное преобразование Фурье, поэтому предполагает периодическое продолжение функций и, далее, обнуление части спектра. Вследствие этого, в спектре исследуемых функций могут возникать нежелательные эффекты (эффект Гиббса). Эти эффекты наиболее заметны в случае небольших изображений и могут приводить к ошибкам в работе метода (отсутствию четкого пика POC функции для “похожих” изображений) (Рис. 4(в)). Метод проекционной фазовой корреляции работает с исходными функциями без периодического продолжения, поэтому этих ошибок не возникает (Рис. 4(г)).



(а) (б) (в) (г)

Рисунок 4: (а), (б) исходные “похожие” изображения небольшого размера, (в) POC-функция – нет четкого пика (г) HPPOC-функция – есть четкий пик.

6. ПРИМЕНЕНИЕ МЕТОДА ПРОЕКЦИОННОЙ ФАЗОВОЙ КОРРЕЛЯЦИИ ДЛЯ УТОЧНЕНИЯ ЛОКАЛЬНЫХ СДВИГОВ ИЗОБРАЖЕНИЙ

При различных условиях съемки радужных оболочек (различия освещения, повороты головы при съемке) радужная оболочка растягивается нелинейно [7]. Поэтому может получиться так, что не существует единого сдвига для различных локальных частей нормализованного изображения (см. рис. 5). Аналогичные эффекты наблюдаются в общем случае двух изображений одних объектов (рис. 6 (а), (б)).

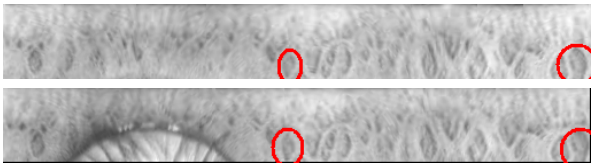


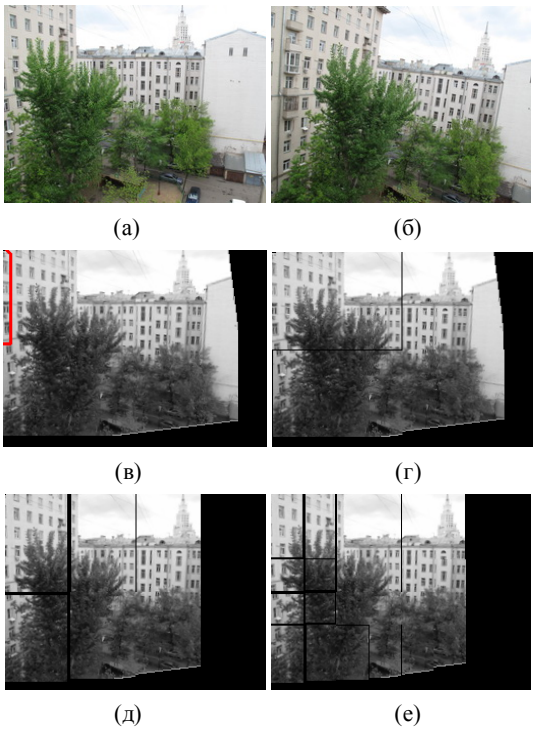
Рисунок 5. Пример локальных сдвигов радужных оболочек. Центральные части точно совпадают друг с другом, тогда как выделенная правая часть первого изображения расположена левее той же части второго изображения.

Повернув второе изображение (рис. 6(б)) методом полярного преобразования Эрмита [5] в спектральной области и, определив сдвиг второго изображения относительно первого методом HPPOC, получим изображение (в). Однако, после применения метода проекционной фазовой корреляции, остались локальные сдвиги изображений (к примеру, на изображении (в) присутствуют окна, которых нет на исходном изображении (а)). Для уточнения локальных сдвигов будем применять наш метод иерархически. Первый уровень иерархии – преобразование изображения (б) в (в) для определения глобального смещения исходных изображений. Далее на каждом следующем уровне иерархии будем делить изображение на 4 равные части. На каждом уровне иерархии уточняются локальные сдвиги изображений. Итоговое изображение практически совпадает с исходным.

Применим описанную выше процедуру к нормализованным изображениям радужных оболочек глаз (Рис. 7). Черные области на рисунке 7 говорят о том, что пик HPPOC функции в этих областях “нечеткий”, т.е. отношение высоты пика к высоте следующего по величине локального максимума HPPOC функции больше пороговой величины k=2.5. Таким образом, метод HPPOC показывает, какие области радужных оболочек “похожи”, а какие нет.

В случае изображений глаз разных людей, черные области занимают либо всю, либо большую часть изображения (Рис. 8), что говорит о том, что локальные области радужных оболочек “не похожи”. В дальнейшем предполагается использовать метод проекционной фазовой корреляции

локально в комбинации с методом ключевых точек [5] для идентификации человека по радужной оболочке глаза.



(а) (б) (в) (г) (д) (е)

Рисунок 6. (а), (б) исходные изображения; (в) повернутое и сдвинутое изображение (б) (1 уровень иерархии); (г), (д), (е) изображение (б) после 2, 3, 4 уровня иерархии.

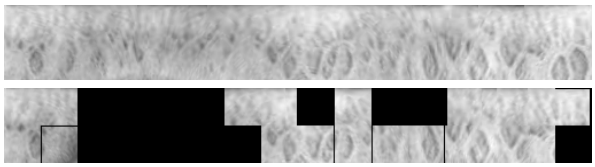
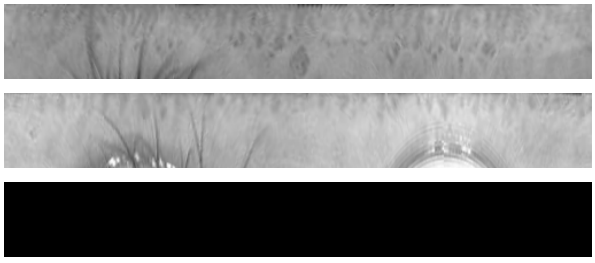


Рисунок 7. Локальные сдвиги областей радужных оболочек одного человека (исходные изображения см. на рис. 5).



(а) (б) (в)

Рисунок 8. (а), (б) исходные изображения разных людей; (в) изображение (б) после нахождения локальных сдвигов.

7. ЗАКЛЮЧЕНИЕ

В работе предложен метод проекционной фазовой корреляции, который показал значительно лучшие результаты по сравнению с методом фазовой корреляции на изображениях небольшого размера. Это связано с тем, что метод HPPOC не требует периодического продолжения изображений в отличие от метода POC, поэтому не возникает нежелательных ошибок от скачков на границах. Поэтому метод HPPOC позволяет находить локальные сдвиги изображений.

Работа выполнена при поддержке ФЦП «Научные и научно-педагогические кадры инновационной России» на 2009–2013 годы и гранта РФФИ 10-07-00433-а.

8. СПИСОК ЛИТЕРАТУРЫ

[1] R. P. Millane, W. H. Hsiao. *The basis of phase dominance,* Optics Letters v. 34, No. 17, p. 2607-2609, 2009.

[2] S. Nagashima, K. Ito, T. Aoki, H. Ishii, K. Kobayashi. *High Accuracy Estimation of Image Rotation using 1D Phase-Only Correlation*, IEICE Trans.Fund. v. E92-A, p. 235-243, 2009.

[3] K. Miyazawa, K. Ito, T. Aoki, K. Kobayashi, H. Nakajima. *A Phase-Based Iris Recognition Algorithm*, LNCS (ICB 2006), No. 3832, p. 356--365, 2006.

[4] A. Krylov, D.Korchagin. *Fast Hermite Projection Method,* LNCS, v.4141, p.329-338,2006.

[5] Е. Павельева, А. Крылов. *Поиск и анализ ключевых точек радужной оболочки глаза методом преобразования Эрмита*, Информатика и ее применения, т.4, в.1, с.79-82,2010.

[6] База данных CASIA-IrisV3 http://www.cbsr.ia.ac.cn/IrisDatabase.htm

[7] L. Ma, T. Tan, Y. Wang, and D. Zhang, *Efficient iris recognition by characterizing key local variations* IEEE Trans. on Image Processing, vol. 13, no. 6, p. 739–750, 2004.

Об авторах

Павельева Елена Александровна – аспирант ф-та ВМК МГУ. E-mail: paveljeva@yandex.ru.

Крылов Андрей Серджевич – д.ф.-м.н., зав. лаб. математических методов обработки изображений ф-та ВМК МГУ. E-mail: kryl@cs.msu.ru.

About the authors:

Elena A. Pavelyeva is a PhD student of Faculty of Computational Mathematics and Cybernetics, Lomonosov Moscow State University. E-mail: paveljeva@yandex.ru.

Professor. Andrey S. Krylov is the Head of the Laboratory of Mathematical methods of Image Processing, Faculty of Computational Mathematics and Cybernetics, Lomonosov Moscow State University. E-mail: kryl@cs.msu.ru

Биометрическая идентификация по радужной оболочке глаза: текущее состояние и перспективы

О.С. Ушмаев
Институт проблем информатики РАН
E-mail: olejku@mail.ru

Аннотация

В статье представлен обзор проблематики биометрической идентификации по радужной оболочке глаза. Отмечены ключевые особенности радужной оболочки глаза. Дана краткая характеристика существующих технологий. Определены перспективные направления дальнейших исследований в области распознавания радужной оболочки глаза.

Ключевые слова: *биометрическая идентификация, радужная оболочка глаза.*

1. ВВЕДЕНИЕ

Идентификация по радужной оболочке глаза является одной из наиболее перспективных биометрических технологий идентификации личности. Этому способствуют такие факторы, как высокая избирательность изображений радужной оболочки глаза, постоянство изображения радужной оболочки глаза во времени, удобная система информативных признаков.

Историю биометрической идентификации по радужной оболочке глаза можно отсчитывать от опубликования патента [1]. За 25 лет идентификация по радужной оболочке глаза прошла большой путь, который показал основные достоинства этой биометрии. Радужная оболочка является внутренним органом, защищенным от механического воздействия. Текстура радужки крайне устойчива во времени. Видимая в ближнем ИК текстура не зависит от генома. Это подтверждают эксперименты с однояйцевыми близнецами. Различия между их радужками идентичны различиям в общей популяции. Расширение зрачка позволяет эффективно отличать текстуру на линзах от текстуры радужной оболочки глаза. Последние испытания показывают достаточно высокую точность идентификации. Практически единственным недостатком биометрии радужной оболочки глаза является маленький размер радужки, что затрудняет ее применение в некооперативных сценариях.

2. КРАТКАЯ ХАРАКТЕРИСТИКА ТЕХНОЛОГИЙ ИДЕНТИФИКАЦИИ ПО ИЗОБРАЖЕНИЮ РАДУЖНОЙ ОБОЛОЧКЕ ГЛАЗА

Технологии идентификации по радужной оболочке глаза являются технологиями распознавания образов. Поэтому они включают достаточно стандартные блоки: получение изображений радужной оболочки глаза с сенсоров, предобработка изображений, вычисление информативных признаков, сравнение информативных признаков.

Большинство работающих в настоящее время систем и технологий идентификации по радужной оболочке глаза основаны на принципах, предложенных Дж. Даугманом в статье [2]. Для распознавания используются изображения, полученные в ближнем ИК-диапазоне с активной подсветкой. Выбор такого диапазона обусловлен физическими свойствами радужной оболочки глаза. На рис. 1 показаны поглощающая способность меланина, который отвечает за пигментацию радужной оболочки глаза, и чувствительность ПЗС-матрицы для различных длин волны. Как видно из рисунка при длинах волны более 700 нм меланин практически не поглощает излучения, в то же в время сенсоры еще достаточно чувствительны.

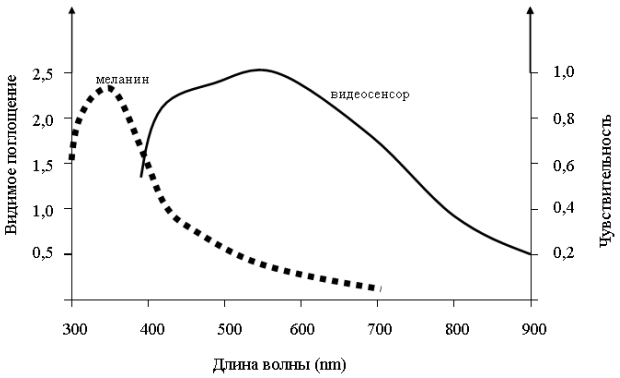


Рисунок 1. Поглощение меланина и чувствительность сенсора

На втором шаге, на полученном в ближнем ИК-диапазоне изображении локализуется радужная оболочка глаза. Границами радужной оболочки глаза являются окружности, на которых достигает максимума следующий функционал:

$$\left| G_\sigma(r) * \frac{\partial}{\partial r} \oint\limits_{r,x_0,y_0} \frac{I(x,y)}{2\pi r} ds \right|, \qquad (1)$$

где $I(x,y)$ - изображение, $G_\sigma(r)$ - сглаживающий гауссиан с нулевым математическим ожиданием и дисперсией σ^2.

На третьем шаге, после определения границ радужки, изображение радужной оболочки глаза нормализуется с целью компенсировать расширение зрачка. Координатная сетка представлена на рис.2. В частном случае, когда центры граничных окружностей совпадают, нормализация является переводом в полярную систему координат.

На четвертом шаге вычисляются информативные признаки. В статье [2] в качестве информативных признаков было предложено использовать локальную фазу φ изображения, получаемую сверткой с ядром фильтра Габора:

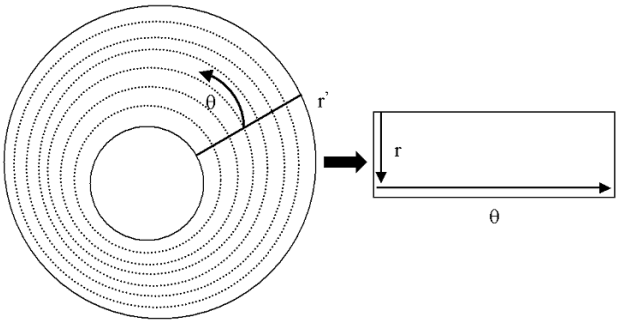


Рисунок 2. Нормализация изображения радужной оболочки глаза

$$z(r.\phi) = \iint_{\rho,\phi} e^{-i\omega(\theta-\phi)} e^{-(r-\rho)^2/\alpha^2} e^{-(\theta-\phi)^2/\beta^2} I(\rho,\phi) d\rho d\phi \,, \quad (2)$$

$$e^{i\varphi_\omega(r.\phi)} = z(r.\phi) / \|z(r,\phi)\| \,. \quad (3)$$

Фазовая информация квантуется. В изначальном варианте использовались 2 бита. В общем случае можно использовать большее число бит (рис. 3). Таким образом длина описания (шаблона) радужной оболочки глаза зависит от количества точек, в которых вычисляется фаза, и числа битов на кодирование фазы. При идентификации предъявленный шаблон радужной оболочки побитно сравнивается с шаблонами, хранящимися в системе. В качестве меры различия радужных оболочек глаза используется расстояние Хэмминга.

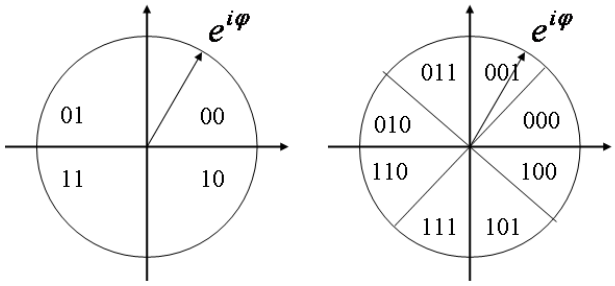


Рисунок 3. Пример кодирования фазы двумя битами (слева) и тремя битами (справа).

В дальнейшем было предложено множество способов улучшения каждого из шагов алгоритма Дж. Даугмана. При локализации (сегментации) даже в самом благожелательном для технологии идентификации сценарии изображение радужной оболочки глаза частично закрыто веком и ресницами. Также на изображении радужной оболочки присутствуют блики. Из последних исследований следует отметить [4-6].

Также в литературе представлено много альтернативных систем информативных признаков. Однако, почти все они построены на схожих с [2] принципах. Вместо габоровских вейвлетов используются другие преобразования [7-10]. Несколько улучшена процедура идентификации за счет учета локальных деформаций изображений [10-12].

Благодаря прогрессу в области методов и алгоритмов обработки изображений радужной оболочки глаза, биометрия радужной оболочки глаза является одной из наиболее точных. Как заявлено Дж. Даугманом в [13], анализ базы из 632 500 изображений радужных оболочек глаз показал потенциальную возможность нулевых ошибок распознавания на массиве такого размера. То есть потенциально ошибка 2-го рода составляет менее 10^{-10}, что может полноценно конкурировать с десятипальцевой дактокартой. Однако такие ошибки достижимы только на базах изображений очень высокого качества. В менее благоприятных условиях, согласно результатам испытаний NIST [14] лучшие алгоритмы идентификации позволяют получить ошибку первого рода порядка 1% при ошибке второго рода 0,1%, что сравнимо с отпечатком пальца или изображения лица.

Такая разница определяет основную цель исследований в области идентификации по радужной оболочке глаза: улучшить точность идентификации в неблагоприятных и некооперативных условиях [15,16].

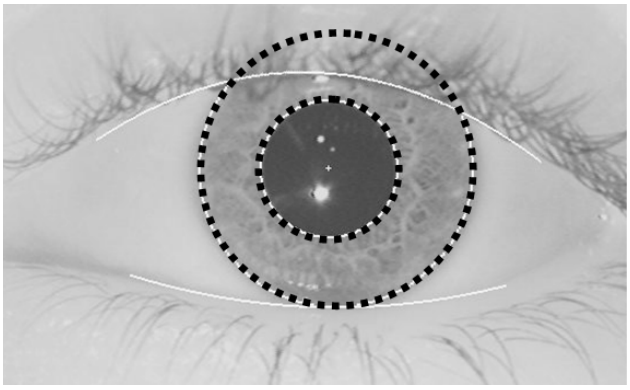


Рисунок 4. Изображение радужной оболочки глаза

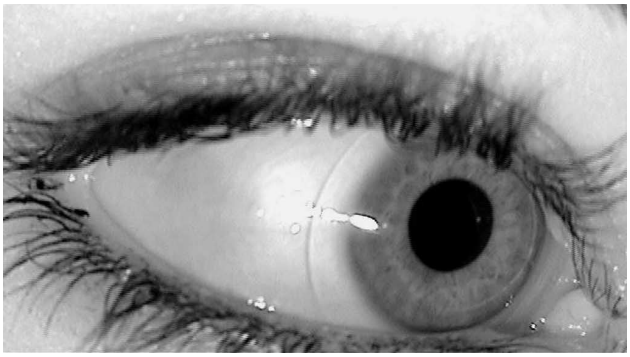


Рисунок 5. Изображение радужной оболочки глаза при некооперативном получении.

3. АКТУАЛЬНЫЕ НАПРАВЛЕНИЯ ИССЛЕДОВАНИЙ

В области биометрической идентификации по радужной оболочке глаза можно определить пять основных направлений. Во-первых, остается актуальным развитие и совершенствование базовых средств идентификации по радужной оболочке глаза: локализация, кодирование, идентификация. Особенно востребованы средства для распознавания радужной оболочки глаза в видимом диапазоне и низком разрешение, а также в некооперативных сценариях, когда радужная оболочка видна под углом (рис.5).

Во-вторых, прогресс в области идентификации по радужной оболочки глаза может быть достигнут за счет совершенствования сенсоров. Наиболее актуальным является получение пригодного для идентификации изображения в движении и на расстоянии более одного метра. Перспективным является использование изображений в видимом диапазоне. В [17] показано, что использование для идентификации других участков спектра совместно с ближним ИК может улучшить точность идентификации.

В-третьих, актуальным является поиск новых признаков, отличных от фазовой информации. В [18-19] предложены новые признаки (ключевые точки), которые оперируют с локальными особенностями текстуры радужной оболочки глаза.

В-четвертых, точность идентификации по радужной оболочке глаза может быть увеличена за счет использования информации об области глаз (periocular biometrics).

В-пятых, высокая точность идентификации по радужной оболочке глаза делает привлекательным ее использование в приложениях, связанных с криптографией и защищенной идентификацией [20].

4. ЗАКЛЮЧЕНИЕ

Идентификация по радужной оболочке глаза остается одной из самых перспективных технологий биометрической идентификации личности. Особенно востребованным является реализация потенциала радужной оболочки глаза для применения в некооперативных сценариях идентификации совместно с изображением лица и возможно другими бесконтактными биометрическими идентификаторами. Поэтому наиболее актуальными направлениями исследований является улучшение распознавания в некооперативных сценариях за счет совершенствования сенсоров, совершенствования системы информативных признаков, а также за счет интеграции с другими модальностями. Отдельный интерес представляет использование радужной оболочки глаза в криптографических приложениях и защищенной идентификации.

Работа выполнена при поддержке гранта Президента России МД72-2011.9 и гранта РФФИ 10-07-00433.

5. СПИСОК ЛИТЕРАТУРЫ

[1] L. Flom, A.Safir, US Patent #4 641,349, U.S. Government Printing Office, Washington, DC, 1987.

[2] J. Daugman, "High confidence visual recognition of persons by a test of statistical independence," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 15(11), pp. 1148-1161, 1993.

[3] N. Kollias, "The spectroscopy of human melanin pigmentation,"/ Melanin: Its Role in Human Photoprotection. Valdenmar Publishing Co., p. 31 – 38, 1995.

[4] H. Proenca ”Iris Recognition: On the Segmentation of Degraded Images Acquired in the Visible Wavelength,” IEEE Transaction on Pattern Analysis and Machine Intelligence, vol. 32, no.8, p. 1502-1516, 2010.

[5] S. Shah and A. Ross, “Iris Segmentation Using Geodesic Active Contours,” IEEE Trans. Information Forensics and Security (TIFS), vol. 4, no. 4, p. 824-836, 2009.

[6] J. Daugman, “New Methods in Iris Recognition,” IEEE Trans. Systems, Man, and Cybernetics, Part B, vol. 37, no.5, 2007, p. 1167-1175, 2007.

[7] D.M. Monro, S. Rakshit, and D. Zhang, “DCT-Based Iris Recognition,” IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 29, no. 4, 2007, pp. 586-596.

[8] Z. Sun and T. Tan, “Ordinal Measures for Iris Recognition,” IEEE Trans. Pattern Analysis and Machine Intelligence, vol.31, no. 12, 2009, pp. 2211-2226.

[9] L. Ma, T. Tan, Y. Wang, and D. Zhang, “Efficient iris recognition by characterizing key local variations,” IEEE Trans. on Image Processing, vol. 13, no. 6, p. 739–750, 2004.

[10] C. Belcher and Y. Du, “Region-Based SIFT Approach to Iris Recognition,” Optics and Lasers in Eng., vol. 47, no. 1, 2009, p. 139-147.

[11] J. Thornton, M. Savvides, and B.V.K. Kumar, “A Bayesian Approach to Deformed Pattern Matching of Iris Images,” IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 29, no. 4, 2007, pp. 596-606.

[12] S.A.C. Schuckers et al., “On Techniques for Angle Compensation in Nonideal Iris Recognition,” IEEE Trans. Systems, Man, and Cybernetics, Part B, vol. 37, no. 5, 2007, pp.1176-1190.

[13] J. Daugman, “Probing the Uniqueness and Randomness of Iris Codes: Results from 200 Billion Iris Pair Comparisons,”Proc. IEEE, vol. 94, no. 11, 2006, pp. 1927-1935.

[14] 3. P.J. Phillips et al., FRVT 2006 and ICE 2006 Large-Scale Results, NISTIR 7408, Nat’l Inst. Standards and Technology, 2007; http://iris.nist.gov/ice/ice2006.htm.

[15] J. Matey et al., “Iris Recognition in Less Constrained Environments,” Advances in Biometrics: Sensors, Algorithms and Systems, Springer, 2008.

[16] A.Ross, “Iris Recognition: The Path Forward,” IEEE Computer, vol. 43, no. 2, p. 30-35, 2010.

[17] C. Boyce et al., “Multispectral Iris Analysis: A Preliminary Study,” Proc. IEEE CS Workshop on Biometrics (CVPRW 06), IEEE CS Press, 2006, pp. 51-59.

[18] Е. Павельева, А. Крылов, “Поиск и анализ ключевых точек радужной оболочки глаза методом преобразования Эрмита,” Информатика и ее применения, т.4, в.1, с.79-82,2010.

[19] M.Sunder, A. Ross “Iris Image Retrieval Based on Macro-features.” ICPR 2010, p. 1318-1321.

[20] F. Hao, R. Anderson, and J. Daugman, “Combining Crypto with Biometrics Effectively,” IEEE Trans. Computers, vol. 55, no. 9, 2006, pp. 1081-1088.

Обучение алгоритмов выделения кожи на цветных изображениях лиц с использованием самоорганизующихся нейронных сетей и морфологических классификаторов на разрезах графов

Ю.В. Визильтер, В.С. Горбацевич, С.Л. Каратеев, Н.А. Костромов
viz@gosniias.ru

Рассмотрены два способа обучения алгоритмов выделения кожи на цветных изображениях лиц – на основе самоорганизующейся нейронной сети типа «растущий нейронный газ» и морфологической классификации путем построения минимальных разрезов графов соседства на обучающей выборке. В качестве рабочего цветового пространства использовалось пространство CIE Lab. Показана эффективность обоих использованных методов, исследованы различия полученных результатов обучения.

Работа выполнена при поддержке РФФИ, гранты №11-08-01114-а, №11-08-01039-а.

1. ВВЕДЕНИЕ

Задача выделения человеческих лиц на цифровых изображениях получила широкое распространение в связи с бурным развитием информационных сетей и охранных систем. Существует много алгоритмов и методов выделения человеческих лиц на изображениях, но наиболее широко применяемым является известный алгоритм Виолы-Джонса, основанный на использовании процедуры обучения типа AdaBoost и Хааро-подобных признаков [1]. Недостатком этого алгоритма является, необходимость практически попиксельного сканирования изображения окнами различных размеров, что приводит к заметной потере производительности при обработке изображений большёго размера. Для преодоления описанных выше трудностей применяются алгоритмы предобработки позволяющие сузить область поиска и тем самым повысить производительность. При этом широкое распространение получили методы, основанные на цветовой сегментации изображений по признаку принадлежности человеческой кожи.

В работе рассматривается два различных способа построения подобных классификаторов - на основе самоорганизующейся нейронной сети типа «растущий нейронный газ» и морфологическое обучение методом минимального разрезания графа соседства для обучающей выборки.

2. МОРФОЛОГИЧЕСКОЕ ОБУЧЕНИЕ МЕТОДОМ МИНИМАЛЬНОГО РАЗРЕЗАНИЯ ГРАФА СОСЕДСТВА ДЛЯ ОБУЧАЮЩЕЙ ВЫБОРКИ

Морфологический подход к синтезу классификаторов основан на рассмотрении задачи синтеза метрического классификатора как задачи оптимальной сегментации (optimal labeling) конечной выборки точек метрического пространства. При этом «форма» и «сложность» классификаторов трактуются в терминах «формы» и «сложности» изображений (образованных метками классов на точках выборки), то есть в терминах математической морфологии [3]-[7].

Для алгоритмической реализации процедур синтеза метрических классификаторов используется техника построения минимальных разрезов графов [8]-[12], применяемая к графам соседства элементов обучающей выборки.

Рассмотрим задачу обучения с учителем. Пусть даны пространство объектов $\mathfrak{A}$, конечное множество классов $C=\{c_1, \ldots, c_l\}$, и известно разбиение объектов по классам: $c_{\mathfrak{A}}(a)$: $a \in \mathfrak{A} \mid\!\rightarrow c \in C$. Обозначение $c_{\mathfrak{A}}$ указывает на то, что функция определена на $\mathfrak{A}$.

Производится описание объектов из $\mathfrak{A}$ дескрипторами из пространства описаний (признаков) X: $x_{\mathfrak{A}}(a)$: $a \in \mathfrak{A} \mid\!\rightarrow x \in X$.

Случайным образом формируется конечная выборка объектов $A \subseteq \mathfrak{A}$, $\|A\|<+\infty$ и соответствующая выборка описаний $X \subseteq X$, $\|X\|<+\infty$. Каждому значению x ставится в соответствие класс c породившего его объекта a:

$$c_X(x)\colon x_{\mathfrak{A}}(a) \in X \mid\!\rightarrow c_{\mathfrak{A}}(a) \in C.$$

По обучающей выборке c_X требуется построить такой распознающий алгоритм или классификатор

$$f_X(x)\colon x \in X \mid\!\rightarrow c \in C,$$

который обеспечивает наилучшее разбиение X на классы из C. «Наилучшее разбиение» формализуется при помощи тестовой выборки

$$c'_Y(x)\colon x \in Y \mid\!\rightarrow c \in C,\ Y \subseteq X,\ Y \cap X = \varnothing,\ \|Y\|<+\infty,$$

и критерия эмпирического риска на выборке Y:

$$J_Y(f_X) = d_H(f_Y, c'_Y) / \| Y \|,$$

$$d_H(f_Y, c'_Y) = \sum_{x \in Y} 1(f(x) \neq c'(x)),$$

где 1(true)=1, 1(false)=0, $\| Y \| = \sum_{x \in Y} 1$. Здесь расстояние Хэмминга d_H имеет смысл числа ошибок классификации на тестовой выборке Y.

Отсюда критерий среднего ожидаемого эмпирического риска имеет вид

$$J_X(f_X) = E_{Y \subseteq X} \{J_Y(f_X)\},$$

где $E_{Y \subseteq X} \{.\}$ – математическое ожидание по всем возможным выборкам $Y \subseteq X$.

Таким образом, может быть сформулирована задача построения оператора оптимального синтеза θ, доставляющего минимум критерию $J_X(f_X) = J_X(\theta c_X)$:

$$\theta\colon c_X \in \Omega_X \mid\!\rightarrow f_X \in \Omega_X,$$

$$\theta = \arg\min_{\theta'} \{J_X(\theta' c_X)\}. \qquad (1)$$

Здесь Ω_X и Ω_X – множества всех возможных разбиений выборки X и пространства X по классам из C.

В большинстве известных подходов от задачи синтеза (1) сразу переходят к задаче обучения классификаторов заданного класса при помощи обучающего правила известного типа:

$$\theta \in \Theta\colon c_X \in \Omega_X \mid\!\rightarrow f_X \in F_X \subseteq \Omega_X,$$

$$\theta = \arg\min_{\theta' \in \Theta} \{J_Y(\theta' c_X)\}, \qquad (2)$$

где F_X – класс классификаторов, Θ - класс алгоритмов обучения классификаторов из F_X на выборках $X \subseteq X$.

Кроме того, вместо недоступного критерия $J_X(f_X)$, на практике используется критерий наблюдаемого эмпирического риска $J_X(\theta c_X)$, который имеет глобальный минимум в точке $f_X \equiv c_X$, заведомо непригодный для неизвестной тестовой выборки Y. Этой проблеме посвящена теория оценки и контроля переобучения, созданная Вапником и Червоненкисом [2]. Здесь эмпирический риск оценивается по обучающей выборке, но сложность решающего правила искусственно ограничивается. Для этого вводится понятие сложности классификатора $Q(f_X)$, а точнее сложности класса

классификаторов $Q(F_x)$. Соответственно вместо задачи (2) решается задача минимизации наблюдаемого риска с регуляризацией по сложности класса обучаемого классификатора:

$$\theta \in \Theta\colon c_X \in \Omega_X \mid\!\rightarrow f_x \in F_x \subseteq \Omega_x,$$

$$\theta = \arg\min_{\theta' \in \Theta} \{J_X(\theta' c_X) + \alpha Q(F_x)\}, \quad (3)$$

где $\alpha \geq 0$ – параметр регуляризации, определяющий компромисс между точностью на обучающей выборке X и сложностью решающего правила, от которой зависит поведение f_x на тестовой выборке Y из (2).

Морфологический поход к машинному обучению направлен непосредственно на решение задачи (1) и позволяет исключить из рассмотрения априорно заданные классы классификаторов и алгоритмов обучения. При этом решение задачи (1) отыскивается в виде композиции решений подзадач:

$$\theta_\alpha = \delta_\alpha \psi_\alpha, \quad (4)$$

где ψ_α - оператор (процедура) синтеза оптимального отклика классификатора на обучающей выборке X с учетом его сложности (локальной некомпактности)

$$\psi_\alpha\colon c_X \in \Omega_X \mid\!\rightarrow f_X \in \Omega_X,$$

$$\psi_\alpha = \arg\min_{\psi'} \{J_X(\psi' c_X) + \alpha Q_X(\psi' c_X)\}; \quad (5)$$

δ_α - оператор (процедура) оптимальной корректной интерполяции (расширения) классификатора f_X на $\mathcal{X}$ с учетом сложности получаемого классификатора f_x:

$$\delta_\alpha\colon f_X \in \Omega_X \mid\!\rightarrow f_x \in \Omega_x,$$

$$\delta_\alpha = \arg\min_{\delta'} \{J_{NN}(\delta' f_X) + \beta Q(\delta' f_X)\}. \quad (6)$$

Здесь

$J_{NN}(\delta f_X) = \{+\infty$, если $\exists x \in X\colon \delta f_X(x) \neq f_X(x)$;

$d_H(\delta f_X(x), \delta^{NN} f_X(x))$ – в противном случае$\}$;

d_H –расстояние Хэмминга; δ^{NN} – простейший оператор интерполяции классификатора, соответствующий правилу ближайшего соседа (Nearest Neighbor).

Поскольку в задаче (1) функционал J имеет вид расстояния Хэмминга, из Утверждения 1 следует, что оператор θ_α (4) является алгебраическим проектором:

$$\theta_\alpha^2 = \theta_\alpha \Rightarrow \forall x \in X\colon \theta_\alpha f_X(x) = f_X(x). \quad (7)$$

Кроме того, на основе θ_α образуется система вложенных классов решающих правил, монотонная относительно α:

$$\forall \alpha \geq \beta \Rightarrow F^\alpha{}_x \subseteq F^\beta{}_x : Q(F^\alpha{}_x) \leq Q(F^\beta{}_x), \quad (8)$$

,где $F^\alpha{}_x = \{f_X\colon \theta_\alpha f_X = f_X\}$ – множество классификаторов (разбиений), стабильное относительно проектора θ_α. В морфологиях изображений такая система вложенных проективных классов рассматривается как множество Пытьевских «форм» нарастающей сложности. В задаче синтеза классификаторов последовательность «форм» используется для решения проблемы переобучения методом минимизации структурного риска.

Определим такой критерий QX(fX), который отдает предпочтение решающим правилам fX, более компактным на выборке X. Для этого определим систему вложенных окрестностей $Ok(x) \subseteq X$, $x \in X \subseteq X$, $k=1,\ldots,\|X\|-1$, состоящих из k ближайших соседей. Введем локальную меру некомпактности fX в окрестности Ok(x):

$$Q_k(x, f_X) = q_H(O_k(x)) / \| O_k(x) \|,$$

$$q_H(O_k(x)) = \sum_{y \in O_k(x)} 1(f_X(x) \neq f_X(y)),$$

$$\| O_k(x) \| = \sum_{y \in O_k(x)} 1. \quad (9)$$

Тогда глобальная мера k-некомпактности имеет вид:

$$Q^k{}_X(f_X) = Q_H(X, f_X) / \| X \|,$$

$$Q_H(X, f_X) = \sum_{x \in X} Q_k(x, f_X). \quad (10)$$

Значение $Q^k{}_X(f_X)$ (11) характеризует эмпирическую оценку вероятности того, что один из k ближайших соседей в разбиении $f_X(x)$ будет отнесен к другому классу. При любых фиксированных k и X усложнению классификатора f_X соответствует нарастание меры k-некомпактности $Q^k{}_X(f_X)$. С другой стороны, при увеличении параметра k в (10) преимущество получают более простые и «гладкие» разделяющие поверхности (см. [10]).

С учетом критерия (10) задача (5) сводится к хорошо известной задаче оптимальной разметки графа на основе скрытой Марковской модели [13], для которой существует эффективное приближенное решение методом минимального разреза графа, вычислимого за низко полиномиальное время относительно числа узлов графа (объектов в выборке) при любом конечном числе классов. Более того, для случая двух классов метод разреза графов может давать точное глобально оптимальное решение.

Алгоритм нахождения минимального разреза на графе с двумя терминальными вершинами позволяет находить минимум функционала энергии вида:

$$E(T) = E_0 + \sum_{i=1..N} E_i(t_i) + \sum_{(i,j) \in V} E_{ij}(t_i, t_j), \quad (11)$$

где N- число нетерминальных вершин графа; $T = \langle t_1, \ldots, t_N \rangle$, $t_1, \ldots, t_N \in \{0,1\}$ – метки ассоциирования каждой нетерминальной вершины с одной из терминальных; $E_i(0)$, $E_i(1) \in \{0,1\}$ – унарные потенциалы; $E_{ij}(t_i, t_j)$ – парные потенциалы, задаваемые четверкой действительных коэффициентов $E_{ij}(0,0)$, $E_{ij}(0,1)$, $E_{ij}(1,0)$, $E_{ij}(1,1)$; V –подмножество пар индексов переменных, задающее систему соседства на T.

Энергия (12) считается субмодулярной [12], если:

$$\forall (i,j) \in V\colon E_{ij}(0,0) + E_{ij}(1,1) \leq E_{ij}(0,1) + E_{ij}(1,0). \quad (12)$$

Для субмодулярной энергии (11)-(12) метод построения минимального разреза графа [8], [11] гарантирует нахождение точного минимума [9], [12].

Для реализации задачи синтеза двухклассового классификатора (5),(10) примем: $C = \{0,1\}$, $N = \|X\|$, $T = \langle t_1, \ldots, t_N \rangle$, $t_1 = f_X(x_1), \ldots, t_N = f_X(x_N)$; $E_i(x) = 1(f_X(x) \neq c_X(x))$, $E_{ij}(t_i, t_j) = 1(f_X(x_i) \neq f_X(x_j))$; $V = \{(i,j)\colon j \in O_k(x_i)\}$. Легко убедиться, что соответствующая энергия (11) будет субмодулярной, а значит, метод минимального разреза графа k-соседства для выборки c_X действительно оптимален и порождает α-семейства проекторов.

3. САМООРГАНИЗУЮЩАЯСЯ НЕЙРОННАЯ СЕТЬ – РАСТУЩИЙ НЕЙРОННЫЙ ГАЗ

В качестве второго способа цветовой сегментации рассматривается алгоритм кластеризации цветового пространства, основанный на аппроксимации цветового пространства изображения самоорганизующейся сетью, обучаемой по алгоритму растущего нейронного газа [15],[16]. Главным преимуществом данного алгоритма является осуществление так называемой “адаптивной” кластеризации входных данных. Т.е пространство не только разделяется на кластеры, но и определяется необходимое их количество исходя из топологии распределения самих данных. Начиная всего с двух нейронов, алгоритм последовательно изменяет (по большей части, увеличивает) их число, одновременно создавая набор связей между нейронами, наилучшим образом отвечающую распределению входных векторов, используя подход соревновательного хеббовского обучения. Каждый нейрон характеризуется т.н. «локальной ошибкой». Соединения между узлами характеризуются «возрастом». Пример такой структуры показан на рис.1.

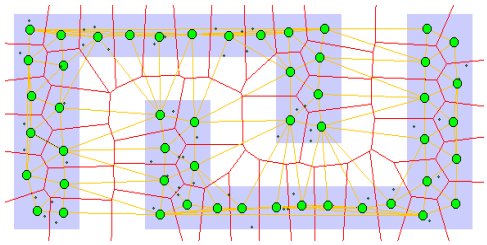


Рис 1. Структура нейронного газа: распределение кластеров (зеленый), связей (оранжевый) и топология данных (синий), конкретные сигналы показаны в виде отдельных точек.

Алгоритм работы растущего нейронного газа кратко можно описать следующим образом:

1. Инициализация: создать два узла с векторами весов, разрешенными распределением входных векторов, и нулевыми значениями локальных ошибок; соединить узлы связью, установив ее возраст равным 0.
2. Подать на вход нейросети вектор x.
3. Найти два нейрона s и t, ближайших к x, т.е. узлы с векторами весов w_s и w_t, такими, что $\|w_s - x\|^2$ минимальное, а $\|w_t - x\|^2$ второе минимальное значение расстояния среди всех узлов.
4. Обновить локальную ошибку нейрона-победителя s путем добавления к ней квадрата расстояния между векторами w_s и x: $E_s \leftarrow E_s + \|w_s - x\|^2$
5. Сместить нейрон-победитель s и всех его топологических соседей (т.е. все нейроны, имеющие соединение с победителем) в сторону входного вектора x на расстояния, равные долям ε_w и ε_n от полного.
   $w_s \leftarrow w_s + \varepsilon_w \cdot (w_s - x)$
   $w_n \leftarrow w_n + \varepsilon_n \cdot (w_n - x)$
6. Увеличить на 1 возраст всех соединений, исходящих от победителя s.
7. Если два лучших нейрона s и t соединены, обнулить возраст их связи. В противном случае создать связь между ними.
8. Удалить все соединения, возраст которых превышает age_{max}. Если после этого имеются нейроны, не имеющие связей с другими узлами, удалить эти нейроны.
9. Если номер текущей итерации кратен λ, и предельный размер сети не достигнут, создать новый нейрон r по следующим правилам:
   - Найти нейрон u с наибольшей локальной ошибкой.
   - Среди соседей u найти нейрон v с максимальной ошибкой.
   - Создать узел r "посредине" между u и v: $w_r = (w_u + w_v) / 2$
   - Заменить связь между u и v на связи u и r, v и r.
   - Уменьшить ошибки нейронов u и v, установить значение ошибки нейрона r.
     $E_u \leftarrow E_u \cdot \alpha$
     $E_v \leftarrow E_v \cdot \alpha$
     $E_r \leftarrow E_u$
   - Уменьшить ошибки всех нейронов j на долю β.
     $E_j \leftarrow E_j - E_j \cdot \beta$
10. Если критерий останова не выполнен, перейти к шагу 2.

4. РЕЗУЛЬТАТЫ ЭКСПЕРИМЕНТАЛЬНОГО ИССЛЕДОВАНИЯ

Тестирование проводилось на базе изображений людей снятых при различных условиях съемки. Изображения были предварительно размечены вручную на области принадлежности пикселей классу кожи. Выборка точек для обработки получена путем перевода тестовых изображений в цветовое пространство CIE Lab с целью отделения цветовых компонент от яркости. Это повышает компактность представления, т.к. кожа имеет характерный цвет, а не яркостную составляющую. Обучение производилось на 10% точек от общего объема выборки в 800 000 точек.

При построении графа соседства использовался алгоритм триангуляции Делоне с динамическим кэшированием треугольников [17]. Нахождение оптимальных разрезов графов осуществлялось с использованием библиотеки [18]. Полученная вероятность правильной классификации цвета пикселя (кожа/не кожа) на тестовой выборке – 0,937.

Для самоорганизующейся нейронной сети были рассмотрены результаты при 32, 128 и 256 кластерах, полученных после кластеризации обучающей выборки. Полученная вероятность правильной классификации цвета пикселя (кожа/не кожа) на тестовой выборке – 0,925.

Как видно, численные значения результатов обучения, полученных двумя описанными методами в задаче цветовой сегментации кожи на изображениях лиц, оказались достаточно близки. Однако более подробное рассмотрение выделенных кластеров демонстрирует существенные различия в их форме. На рис.2 показана обучающая выборка в плоскости ab цветового пространства CIE Lab. Красными точками помечены пикселы кожи, зелеными – других классов.

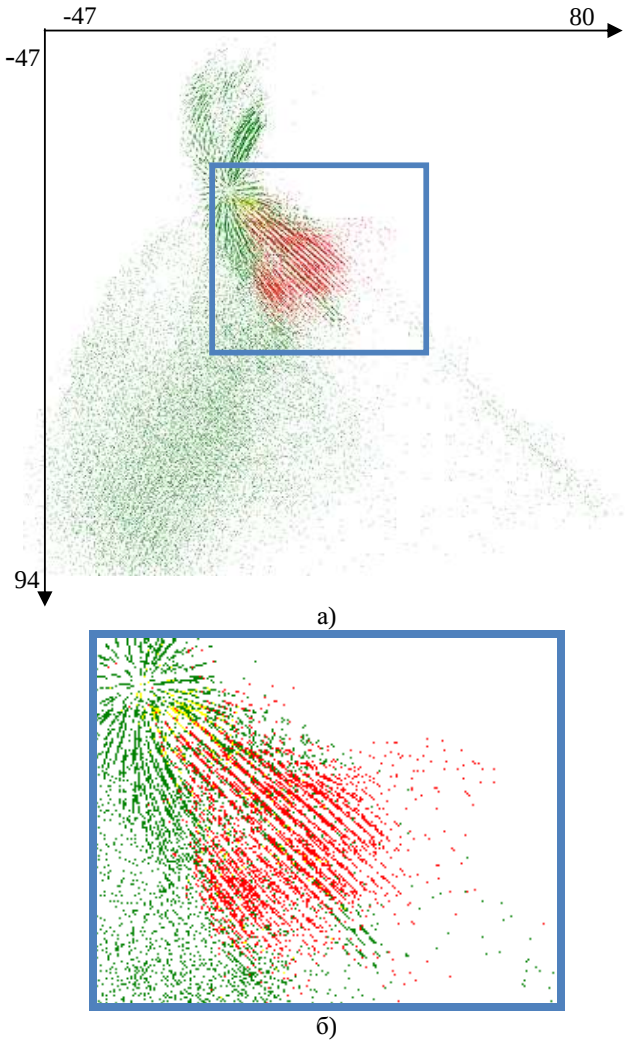


Рис2. а) Обучающая выборка в цветовой плоскости ab. б) Увеличенный фрагмент обучающей выборки содержащий точки “кожи”(красные) и “не кожи” (зеленые). Желтым показаны точки, имеющие обе метки.

На рис.3 приводится результат переразметки обучающей выборки после применения процедуры обучения на основе разреза графа соседства, а на рис.4 – после обучения на основе «растущего нейронного газа». Рис.5 демонстрирует различия в форме кластеров, полученных двумя способами обучения.

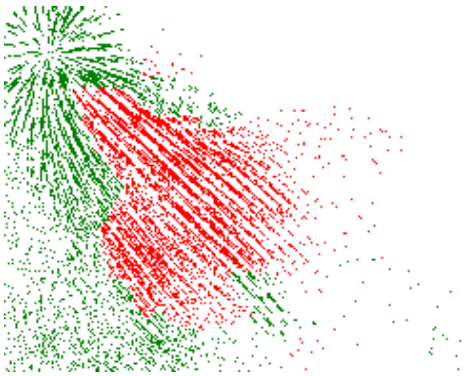


Рис3. Результат переразметки обучающей выборки после морфологического обучения на основе разреза графа.

Рис4. Результат переразметки обучающей выборки после обучения на основе «растущего нейронного газа».

Рис5. Различия в результатах обучения (рис 3, 4). Жирным выделены точки, классифицируемые по-разному.

Как видно, значительные отличия в форме полученных кластеров цвета кожи указывают на существенно различную природу этих процедур обучения, что позволяет в дальнейшем рассматривать возможность их комплексирования с целью повышения вероятности правильной классификации.

Литература

[1] Paul Viola, Michael Jones “Robust Real-Time Object Detection” Second International Workshop On Statistical And Computational Theories Of Vision – Modeling, Learning, Computing, And Sampling Vancouver, Canada, July 13, 2001.
[2] Вапник В. Н. Восстановление зависимостей по
[3] Pavel M. Fundamentals of Pattern Recognition. Marcel Dekker. Inc., New York, 1989.
[4] Serra J. Image Analysis and Mathematical Morphology, Academic Press, London, 1982.
[5] Пытьев Ю.П. Морфологический анализ изображений. Доклады АН СССР, 1983. Т. 269. № 5. С. 1061-1064.
[6] Пытьев Ю.П., Чуличков А.И. Методы морфологического анализа изображений // М.: ФИЗМАТЛИТ, 2010. 336с.
[7] Визильтер Ю.В. Обобщенная проективная морфология. «Компьютерная оптика», Том 32, №4. 2008, С.384-399.
[8] L. Ford and D. Fulkerson. Flows in Networks. Princeton University Press,1962.
[10] D. Greig, B. Porteous, and A. Seheult. Exact maximum a posteriori estimation for binary images. Journal of the Royal Statistical Society, 51(2):271–279, 1989.
[11] Y. Boykov and V. Kolmogorov. Computing geodesics and minimal surfaces via graph cuts. In Proc. IEEE International Conf. Computer Vision (ICCV), pages 26–33, 2003.
[12] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. IEEE Trans. Pattern Analysis and Machine Intelligence (PAMI), 26(9):1124–1137, 2004.
[13] V. Kolmogorov and R. Zabih. What energy functions can be minimized via graph cuts?. IEEE Trans. Pattern Analysis and Machine Intelligence (PAMI), 26(2):147–159, 2004.
[14] Geman S., Geman D. Stochastic relaxation, Gibbs distributions, the Bayesian restoration of images. – IEEE Trans. Pattern Analysis, Machine Intelligence, 1984, № 6, pp.721-741.
[15] B. Fritzke. A growing neural gas network learns topologies.In G. Tesauro, D. S. Touretzky, and T. K. Leen, editors, *Advances in Neural Information Processing Systems 7*, pages 625-632. MIT Press, Cambridge MA, 1995a.
[16] B. Fritzke. Fast learning with incremental RBF networks. *Neural Processing Letters*, 1(1):-5, 1994b.
[17] Скворцов А.В. Обзор алгоритмов построения триангуляции Делоне. Вычислительные методы и программирование. 2002. Т3. с.14-39.
[18] Yuri Boykov, Vladimir Kolmogorov. MAXFLOW - software for computing mincut/maxflow in a graph.V.3.01: http://www.cs.ucl.ac.uk/staff/V.Kolmogorov/software.html

Young Scientists School

GraphiCon'2011

September 26–30, 2011
Moscow, Russia

REAL-TIME DEPTH MAP OCCLUSION FILLING AND SCENE BACKGROUND RESTORATION FOR PROJECTED-PATTERN-BASED DEPTH CAMERAS

Yuriy Berdnikov

Moscow State University
Graphics & Media Lab
yberdnikov@graphics.cs.msu.su

Dmitriy Vatolin

Moscow State University
Graphics & Media Lab
dmitriy@graphics.cs.msu.ru

ABSTRACT

In this paper we present our approach to real-time filtering of depth maps taken using projected-pattern-based depth cameras and to restoration of scene backgrounds for images taken using aligned RGB and depth cameras. An original depth map contains a numerous occlusions, and stereo-from-depth-map video generation leaves many uncovered areas. To solve this problem we propose an adaptive occlusion-filling algorithm for depth map processing and for restoration of scene backgrounds using depth map. Our goal is to accurately fill occlusions while maintaining real-time processing speed using common workstations.

Index Terms— depth map, disparity map, kinect, projected patterns, depth camera, background restoration, depth restoration

1. INTRODUCTION

In November, 2010 Microsoft released the first widely available and relatively inexpensive depth camera: Microsoft Kinect. This depth camera uses projected patterns technology for depth estimation: an infrared light source projects a dot pattern on the scene and an infrared camera captures an image. Because of the relative displacement of projector and camera, the displacement of the projected dots in the shot depends on the distance between the camera and the scene point. Such cameras have many drawbacks: they cannot work with shiny or self-illuminating surfaces, and all foreground objects cause occlusions. Moreover, a non-trivial problem crops up when filling occlusions for stereo pairs because of different foreground and background depths.

Approaches to filling occlusions filling usually work slowly, if they are to produce high-quality results. For example, the approach described in [1] uses normal maps and the AdaBoost classification algorithm, which requires training and extensive computational resources. The approaches of [2] and [3] require no training and work much faster, but they still cannot support real-time processing. Patch-based approaches [4] are highly parallel and could potentially be implemented using GPUs to achieve real-time speeds, but they have a common drawback: either they use only spatial information, or they use temporal information improperly? creating visual artifacts in moving scenes.

The approaches described in [5] and [6] are much more suitable for the present task because of simplicity. [5] uses a weighted combination of simple spatial and temporal inpainting; [6] uses fast spatial filling. Neither method. however, uses the specific characteristics of a depth map created using a projected patterns based depth camera, resulting in a priori decreasing their efficiency. To avoid such problems? we developed a new algorithm.

This research was partially supported by the grant number 10-01-00697-a by Russian Foundation for Basic Research

2. PROPOSED METHOD

2.1. Depth Map Occlusion Classification

(a) RGB frame (b) Depth frame

Fig. 1. Combined video and example of the occlusion classes.

There are two occlusion classes :

1) Occlusions caused by the edges of foreground objects (example marked on 1 with red color)

2) Occlusions caused by shiny surfaces and other object characteristics and random factors (example marked on 1 with green color)

To separate this cases, we developed the following separation criteria:

1) Occlusions on the foreground object edges exist only if the foreground is right of background (this is true for IR projector place in the right of IR camera)

2) The occlusion width equals the disparity difference between the foreground and background

$$W_{occ} = Disp(D_{foreground}) - Disp(D_{background})$$

Here $Disp(x)$ is a conversion function between the depth indicated by the camera and the pattern disparity. This function is unique to each exact camera. We approximate this function using a polynomial function

$$Disp(x) = a_n x^n + ... + a_1 x + a_0$$

where the coefficients a_n, ..., a_1, a_0 are obtained from the camera calibration datasheet or via manual calibration using sum-of-square-difference (SSD) optimization.

Analyzing each occlusion horizontally, we determine the most probable width for the occlusion

$$W_{mp} = Disp(D_f) - Disp(D_b)$$

$$D_f = \frac{\int_{\Omega_1} w_i D_i}{\int_{\Omega_1} w_i}$$

$$D_b = \frac{\int_{\Omega_2} w_i D_i}{\int_{\Omega_2} w_i}$$

Here, $D(i)$ is the depth of the current pixel, Ω_1 and Ω_2 are the foreground and background areas neighboring the current occlusion, and w_i is the weight of current pixel; this weight depends on the pixel's distance from the occlusion.

Comparing W_{mp} and W_{curr}, we can determine which class this occlusion area belongs to.

2.2. Depth Map Occlusion Filling

For occlusion class 1 we propose using "deepest neighbor" method:

$$D_i = max(D_f, D_b)$$

This method is physically correct, because such occlusions are caused by the object's IR shadow, and these areas must be threated as a background.

For occlusion class 2, we propose using the interpolation method:

$$D_i = \frac{\alpha D_f + \beta D_b}{\alpha + \beta}$$

where α and β are the distances to the nearest foreground and background pixels.

2.3. Background Estimation and Scene Occlusion Filling

Assuming that the camera is static or that the camera motion has been compensated using some appropriate method, we can use the "deepest in history" method for background reconstruction: if the depth of the current pixel is greater than the depth of the current background, we update the background depth and RGB values.

To avoid missing changes in the background color (like camera noise or a dynamic background such as TV-set), we update the background depth values only if $D_{new} > D_{current}$, and we update the RGB values when $D_{new} > \delta D_{current} - \gamma$, where $\delta \leq 1$ and $\gamma \geq 0$

To generate an image using a virtual displaced camera, we preform object displacement on the basis of the object's depth for the current scene and the restored background, and we fill occlusions in the current scene using background data.

3. RESULTS

First video sequence is 300 frames long with moderate foreground motion and static background, 180th frame was taken as an example. Second video sequence is 200 frames long, contains very slow motion and camera shaking and 150th frame was taken as an example.

Both video sequences are taken using Microsoft Kinect, 640x480 @ 30 fps.

Algorithm has been implemented in c++ and show real-time performance on Intel Core I5 2.56 GHz.

Fig. 2. Source RGB frame with corrected zoom and camera misalignment

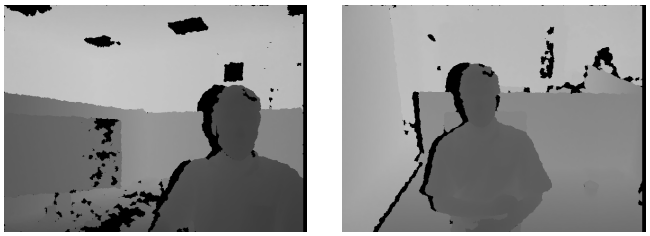


Fig. 3. Source depth frame

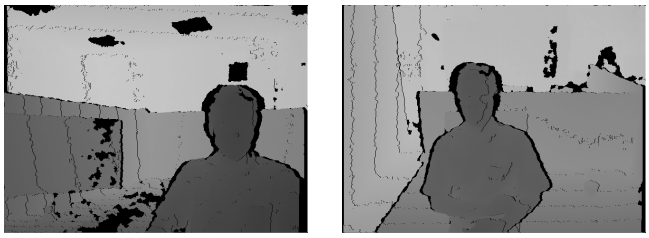


Fig. 4. Depth frame with corrected camera displacement

Fig. 5. Depth frame after occlusion filling

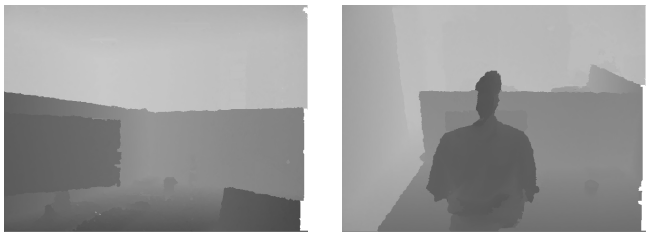


Fig. 6. Estimated scene background depth

Fig. 7. Estimated scene background RGB

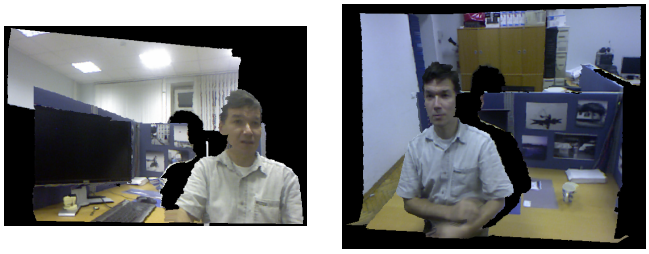


Fig. 8. 3D visualization without background restoration

Fig. 9. 3D visualization with background restoration

4. FURTHER WORK

In the short term, we plan to improve the occlusion filling algorithm by considering temporal information and by using fast motion-compensation algorithms to enable the use of non-static cameras.

5. CONCLUSION

In this work we presented our approach to real-time depth map filtering and occlusion filling as well as background restoration for video that combiner RGB and depth information, where the depth was determined using a projected-patterns-based depth camera. Lastly, we described our intended directions for future work.

6. ACKNOWLEDGMENTS

This research was partially supported by the grant number 10-01-00697-a by Russian Foundation for Basic Research.

7. REFERENCES

[1] Christoph Strecha Pascal Fua Engin Tola, Andrea Fossati, "Large occlusion completion using normal maps," *ACCV*, November 2010.

[2] C. Rother J. Shotton A. Criminisi, A. Blake and P. H. S. Torr, "Efficient dense stereo with occlusions for new view-synthesis by four-state dynamic programming," *International Journal of Computer Vision*, vol. 71, num. 1, pp. 89–110, November 2010.

[3] Atanas Gotchev Lucio Azzari, Federica Battisti, "Comparative analysis of occlusion-filling techniques in depth image-based rendering for 3d videos," *ACM Multimedia*, 2010.

[4] Kentaro Toyama Antonio Criminisi, Patrick Perez, "Object removal by exemplar-based inpainting," *WI Proc. IEEE Computer Vision and Pattern Recognition*, 2003.

[5] Barenburg B. Magalhaes J. P. Gunnewick R. Klein, Berretty R-P. M., "Coherent spatial and temporal occlusion generation," *SPIE, the International Society for Optical Engineering*, ISSN 0277-786X 2009 2010.

[6] Richard McKenna Yu-Sung Chang Manuel M. Oliveira, Brian Bowen, "Fast digital image inpainting," *Proceedings of the International Conference on Visualization, Imaging and Image Processing*, 2001.

Fig. 10. Source RGB frame with corrected zoom and camera mispointing

Fig. 11. Source depth frame

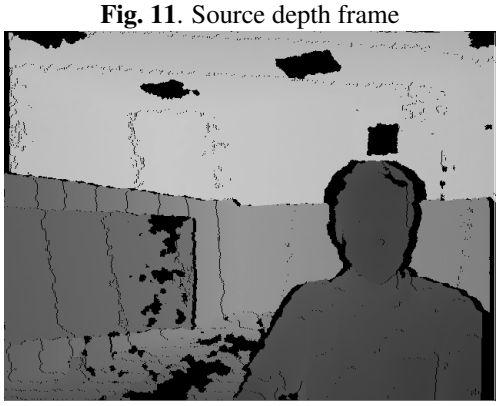


Fig. 12. Depth frame with corrected camera displacement

Fig. 13. Depth frame after occlusion filling

Fig. 14. Estimated scene background depth

Fig. 15. Estimated scene background RGB

Fig. 16. 3D visualization without background restoration

Fig. 17. 3D visualization with background restoration

Automatic Logo Removal for Semitransparent and Animated Logos

Erofeev Mikhail, Dmitriy Vatolin
Department of Computational Mathematics and Cybernetics
Moscow State University, Moscow, Russia
{merofeev, dmitriy}@graphics.cs.msu.ru

Abstract

Adding a visual logo to a video sequence is a popular method of identifying the owner of that sequence. In this paper we propose a fully automatic method for removing opaque, semitransparent and animated logotypes from video sequences.

Keywords: *Logotype removal, video processing.*

1. INTRODUCTION

In this paper we consider logo removal as a process of automatic detection of a logotype's shape and position in each frame of the video, followed by complete removal of that logotype from the video. The most common types of logos are the following:

1. Opaque static logo—the logo image is overlaid on all video frames
2. Semitransparent static logo—the source frame is alpha blended with the logo image
3. Animated logo—the logo image changes with time, but usually this change is periodic

The proposed method allows for removal of all these types of logos.

2. RELATED WORK

Several related works address TV logos. These works can be classified into two categories.

2.1 Logo detection using a logo database

Methods in this group use previously collected information about a set of logos to detect which one is shown on the screen.

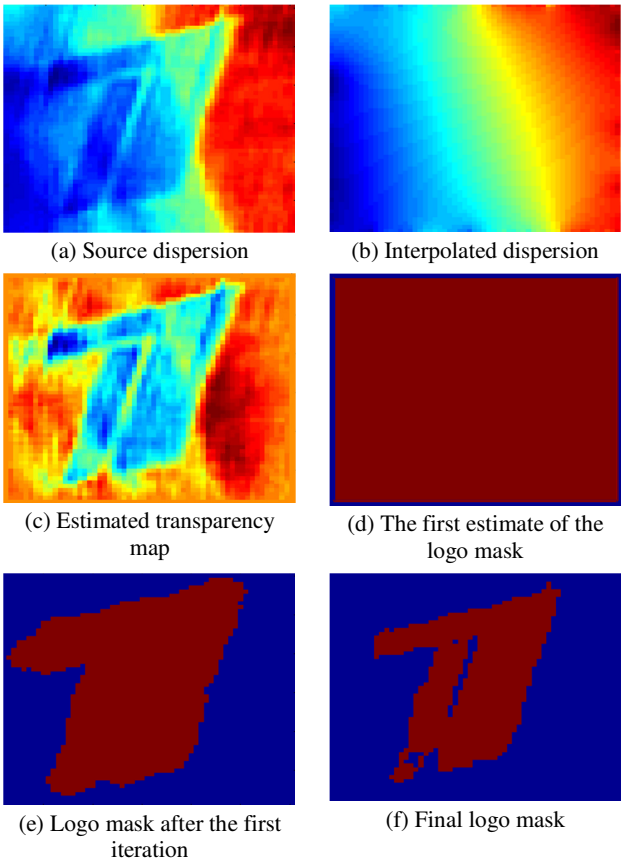
In [3], the authors propose a real-time approach to detecting opaque, semitransparent and partially animated logos. They also propose using average gradient values to detect semitransparent logos. Our approach uses average gradient values to estimate the position of a static logo. In [1], the authors reported good detection results for animated logos when using the unified logo boundary representation.

Nevertheless these approaches cannot be applied in the case of arbitrary logotypes, because of the need to build a database containing all existing logos.

2.2 Logo detection and removal with assumptions

The second group of works solves the problem of automatically removing an arbitrary logotype from a video sequence. The authors in these cases make some additional assumptions about the logo (for example, the logotype doesn't change its color or shape) to permit detection of the logo in the video data.

In [2], the authors propose methods for detecting and removing semitransparent and opaque logos. They assumed that the video content—except for the logo—changes over time. To exploit these temporal variations for logo detection, the sequence is divided into segments with static content. A similar concept is used in our approach to acquire a high-contrast dispersion map.



(a) Source dispersion (b) Interpolated dispersion

(c) Estimated transparency map (d) The first estimate of the logo mask

(e) Logo mask after the first iteration (f) Final logo mask

Figure 1: Main steps of building logo mask

In [4], the authors use a multi-stage approach to logo detection. First, simple binary segmentation is performed, thereby filtering out a large portion of the pixels that belong to moving objects and to the background. The second step involves a Bayesian classifier and neural networks to detect the "coarse" logo. The third step consists of some post-processing to refine the logo mask.

Both of these approaches fail to remove animated logos.

3. PROPOSED APPROACH

Our method consists of two main steps: logo detection and logo removal. For the first step we determine the exact position of the logo and its binary mask. Also for semitransparent logos, some additional information about their color and transparency is collected. The second step uses the data from the previous step to completely remove the logo from the image

3.1 Logo detection

Our method treats static and animated logos in fundamentally different ways. Static opaque and static semitransparent logos are handled in the same way. We discuss the cases of static logos and animated logos separately.

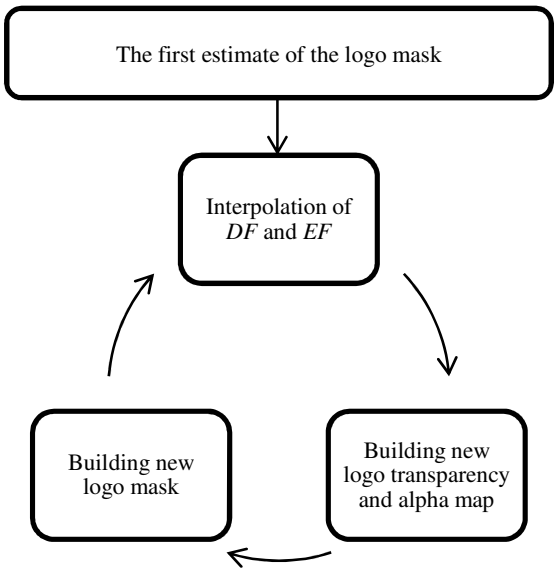


Figure 2: Algorithm of building logo mask

3.1.1 Static logo

For static logos, the frame containing the logo can be considered a source frame that is alpha blended with logo image.

$$I(t) = (1 - \alpha)F(t) + \alpha L$$

Here *I(t)* is the frame containing the logo, *F(t)* is the source frame and α is the transparency map. For opaque logos, the transparency map consists only of the value 1 for points that are part of the logo and the value 0 for all other points.

The first step in logo detection is estimating the logo's position. For this purpose we calculate a gradient field for each frame and then compute the average gradient map for several frames.

$$G(n) = \frac{1}{n}\sum_{i=1}^{n} \|\nabla \mathrm{I}(ki)\|$$

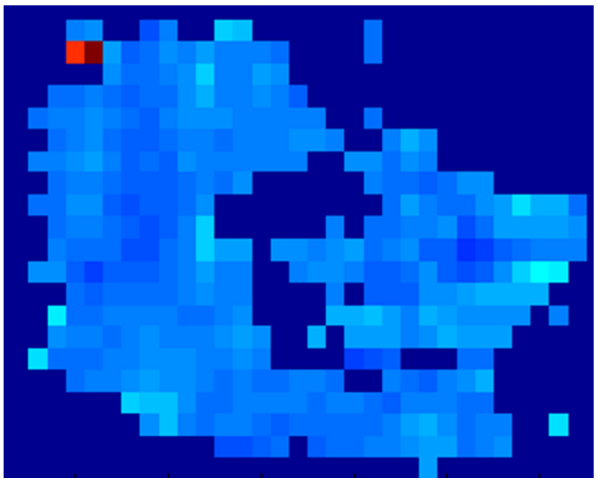
Here *k* is a time scaling factor (*k*=20 in our experiments); such time scaling allows us to dismiss the edges of slowly moving objects. For each *G(i)* we determine a set bounding boxes containing areas with high values (the value should be higher than that of 80% of the other *G(i)* points). If a bounding box doesn't change size and position for several instances of *G(i),* that bounding box is assumed to contain a logo image.

All computations at this point are carried out inside the selected bounding box. The last step of logo detection is intended to build the final binary mask of the logotype, its transparency map and its color. We collect a set *R* of *n* (*n*=200) logo region images that are as different as possible. For each video frame we cut out a logo region *r*. If *R* contains less than *n* images, *r* is inserted into the set. If *R* already has *n* images and the following condition is true

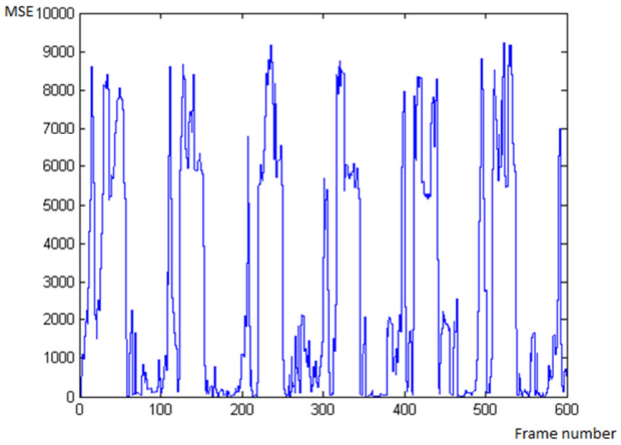
$$\min_{r' \in R} \|r - r'\| > \min_{r_1 \in R, r_2 \in R. r_1 \neq r_2} \|r_1 - r_2\|,$$

then *r* is inserted into *R* and the set of the most similar regions R_s is constructed. Then we randomly select one region from the set R_s and remove it from *R*.

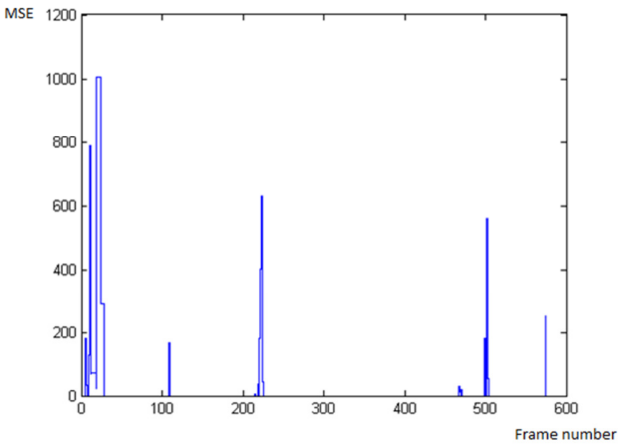
$$R_s = \arg\min_{r_1}\left(\min_{r_2 \in R, r_1 \neq r_2} \|r_1 - r_2\|\right)$$



(a) Period-quality measure for each block
(blocks with higher quality have brighter color)

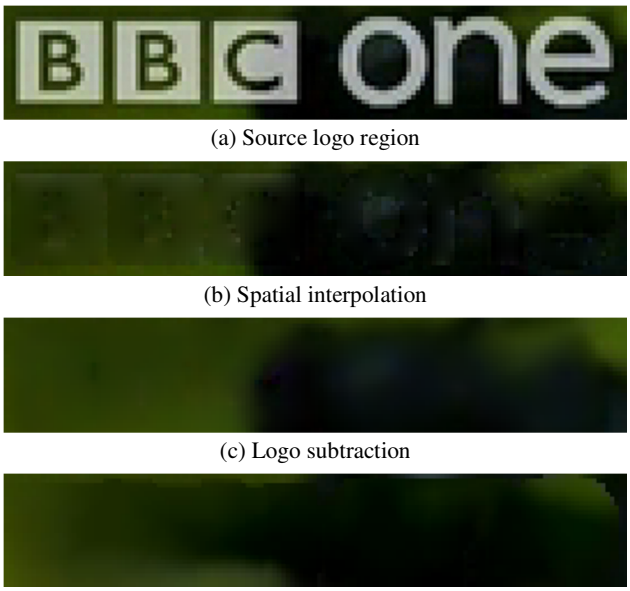


(b) Distances function for block containing an animated logo



(c) Distances function for block containing no logo

Figure 3: Example of quality and distance functions



(a) Source logo region

(b) Spatial interpolation

(c) Logo subtraction

(d) Motion estimation

Figure 4: Source region and examples of several methods of reconstructing frames

For all images in *R*, dispersion and mathematical expectation in the time domain are computed (an example of dispersion is shown in fig. 1(a)).

Now we make assumption that dispersion and mathematical expectation of the whole video are equal to dispersion and mathematical expectation of *R*.

$$DI = DR$$
$$EI = ER$$

Next, the iterative process of building the logo mask is carried out. Fig. 2 shows the flowchart for this algorithm. The first estimate of the logo mask is the rectangle covering the entire logo region, except for a single-pixel-wide border around it, as shown in fig. 1(d). Taking into account only points not covered by the logo mask, we interpolate other points in the dispersion map by solving Laplace's equation.

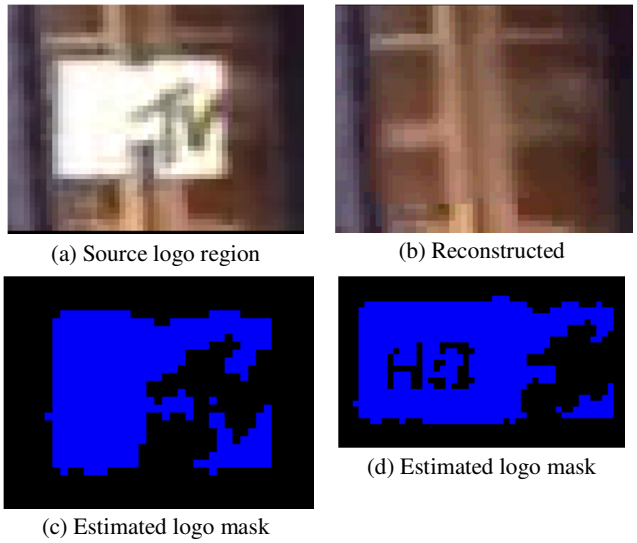
$$\Delta DF = 0$$

The solution *DF* of this equation is the interpolated dispersion (an example of which is shown in fig. 1(b)) behind the logotype. *EF* is interpolated in the same way.

We can compute a transparency map (an example of which is shown in fig. 1(c)) and color for each logo point.

$$DI(t) = D[(1-\alpha)F(t) + \alpha L]$$
$$\mathrm{EI}(t) = \mathrm{E}[(1-\alpha)F(t) + \alpha L]$$
$$\alpha = 1 - \sqrt{\frac{DI(t)}{DF(t)}}$$
$$L = \frac{EI(t) - (1-\alpha)EF(t)}{\alpha}$$

Using simple binary segmentation with a threshold of $\alpha > 0.2$, we can more accurately estimate the logo mask (an example of which is shown in fig. 1(e)). The new mask is used to compute a new transparency and color map. In most cases three iterations of this process are enough to obtain an accurate logo mask (an example of a final mask is shown in fig. 1(f)).



(a) Source logo region

(b) Reconstructed

(c) Estimated logo mask

(d) Estimated logo mask

Figure 5: Source and reconstructed regions from video sequence with animated logo. (c-d) masks for different phases of animated logo

3.1.2 *Animated logo*

Most animated logos change their color, shape and position periodically. The first step of our approach for animated logo detection is estimation of this period. To this end we divide the frame into regular blocks $B_{i,j}^t$ (where i and j are spatial coordinates and t is the frame number) that are 16×16 pixels each. For each block we estimate its period of change $T_{i,j}$ and period-quality measure $Q_{i,j}$.

$$T_{i,j} = \arg\min_t \sum_{t'} \left\| B_{i,j}^t - B_{i,j}^{t'} \right\|$$
$$Q_{i,j} = \left(\min_t \sum_{t'} \left\| B_{i,j}^t - B_{i,j}^{t'} \right\| \right)^{-1}$$

To save time our method doesn't compute these values directly. We compute the function of distances between blocks for the first frame and a subsequent frame.

$$d_{i,j}(t) = \left\| B_{i,j}^1 - B_{i,j}^t \right\|$$

We can assume that

$$\left\| B_{i,j}^{t_1} - B_{i,j}^{t_2} \right\| \approx \left| d_{i,j}(t_1) - d_{i,j}(t_2) \right|.$$

Using this assumption, $T_{i,j}$ and $Q_{i,j}$ can be computed much faster.

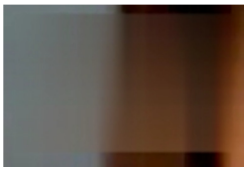
Fig. 3(a) shows an example of $Q_{i,j}$ for each block. Fig. 3(b) illustrates a distance function for the block containing an animated logotype. This function changes periodically, and the block has a large period-quality measure. Fig. 3(c) shows another distance function computed for a block containing no logo. This function changes chaotically, and its corresponding block has a low period-quality measure. The block $B_{i,j}$ with the highest $Q_{i,j}$ is assumed to contain an animated logo with a period of change $T_{i,j}$.

When the period of change is known, we consider frame sets $A_i(t)$.

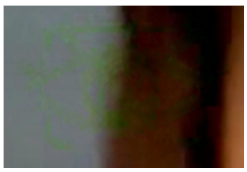
$$A_i(t) = \{ I(Tt + i) | t \in N \}$$

(a) Source logo region

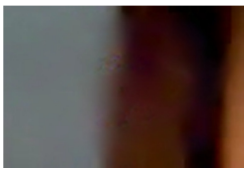
(b) Fuzzy Logo Filter



(c) Logoaway



(d) DeLogo



(e) Our approach

Figure 6: Comparison of logo removal tools

Here T is the estimated period of change. Each A_i can be regarded as a video sequence containing a static logo, and the methods discussed in the previous section can be used for its detection.

3.2 Logo removal

Logo removal is the last step of the proposed approach. We implemented and tested several methods of reconstructing frames behind the logo.

3.2.1 Spatial interpolation

This method uses only the binary logo mask. We solve Laplace's equation to interpolate points covered by the logo mask. This method yields good results when applied to smooth areas, but it fails to when applied to textured areas. An example of the results is shown in fig. 4(b).

3.2.2 Logo subtraction

This method can deal only with semitransparent logotypes and uses the transparency map and logo color data collected in the previous step. To compute the frame without the logo, the following formula is used.

$$F(t) = \frac{I(t) - \alpha L}{1 - \alpha}$$

Fig. 4(c) shows an image obtained using this method.

3.2.3 Motion estimation

This method uses the logo's binary mask and motion information from previous frames. The method can deal with any type of logo. Fig. 4(d) shows results.

4. RESULTS

We tested our methods on several video sequences containing different types of logos. The average frame rate for the logo detection step for HD video was 3, and the average frame rate for the removal step was 4. Fig. 4-5 show example frames for which our approach removed the logo.

Objective comparison of our approach with those of several publicly available logo removal tools was performed. A video sequence without any logotypes was taken as the ground-truth sequence. Opaque and semitransparent logos were added to this sequence. Each tool was used to remove the logo from the test sequences, and the PSNR was measured relative to the ground-truth sequence and its output. Results for this comparison are shown in the table below. Fig. 6 shows output of these tools. Unfortunately we cannot provide any comparison for animated logos because of the lack of a publicly available tool for automatically removing such logotypes.

Filter name	Required additional user input	Opaque logo PSNR (dB)	Semitransparent logo PSNR (dB)
Fuzzy Logo Filter [5]	No	36.29	36.30
Logoaway [6]	Logo bounding box	36.35	36.36
DeLogo [7]	Logo mask	38.64	35.80
Our approach	No	39.38	39.44

As we can see in the table above our approach has the best PSNR value among publicly available tools and requires only video with logotype as input. Fig. 6 also shows that output of our method contains less visually noticeable distortions then output of other methods.

In this paper we have presented logo detection and removal techniques for several types of logo. For static logos we analyze dispersion map of the logo region to carry out logo mask. For animated logos we analyze periodical changes in the input video sequence to determine logo region. Also our animated logo detection method is the first animated logo detection method that doesn't require additional source data.

5. ACKNOWLEDGEMENTS

This research was partially supported by grant 10-01-00697-a from the Russian Foundation for Basic Research.

6. REFERENCES

[1] E. Esen, M. Soysal,T. Ateş, A. Saracoğlu and A. Alatan. *"A Fast Method for Animated TV Logo Detection"*. Content-Based Multimedia Indexing, 2008. CBMI 2008.

[2] K. Meisinger, T. Troeger,M. Zeller, and A. Kaup, *"Automatic TV logo removal using statistical based logo detection and frequency selective inpainting,"* presented at the Eur. Signal Processing Conf., Sep. 2005.

[3] A. dos Santos and H. Kim. *"Real-Time Opaque and Semi-Transparent TV Logos Detection"*. In WACV, 2007.

[4] W. Q. Yan, J. Wang, and M. S. Kankanhalli, *"Automatic Video Logo Detection and Removal"* Multimedia Systems, 10(5), pp. 379-391, July 2005.

[5] http://wiki.atrox.at/index.php/TV_LogoRemove

[6] http://www.videohelp.com/tools/Virtualdub_Logoaway_filter

[7] http://neuron2.net/delogo132/delogo.html

Автоматическое сопоставление изображений и облака трехмерных точек

Глеб Кривовязь[1], Алексей Черников, Александр Велижев[2]
Лаборатория компьютерной графики и мультимедиа, Факультет вычислительной математики и кибернетики
Московский Государственный Университет имени М.В.Ломоносова, Москва, Россия
[1]gkrivovyaz@graphics.cs.msu.ru, [2]avelizhev@graphics.cs.msu.ru

Аннотация

В данной статье рассматривается задача сопоставления изображений и облака трехмерных точек. Предлагается новый алгоритм решения задачи, развивающий существующий подход [11] оценки соответствия между положением камеры и облаком трехмерных точек с помощью максимизации функции взаимной информации фотоизображения и карты глубины. Идея нового метода заключается в одновременном использовании двух изображений сцены. Такой подход позволяет наложить дополнительные ограничения на решение задачи за счет учета геометрических связей между изображениями. Проведенное тестирование показывает улучшение результатов относительно базового метода, что подтверждает перспективность предложенного подхода.

Ключевые слова: *сопоставление изображений, облако трехмерных точек, лазерный сканер, параметры внешнего ориентирования камеры, фундаментальная матрица.*

1. ВВЕДЕНИЕ

Все более широкое применение систем лазерного сканирования ставит новые задачи, связанные с обработкой получаемых с их помощью данных – облаков трехмерных точек. Одной из таких задач является сопоставление изображений и облака трехмерных точек некоторой сцены.

Одновременное использование изображений и облаков трехмерных точек открывает новые возможности при анализе сцен. Действительно, эти данные различной природы органично дополняют друг друга, предоставляя информацию как о цвете, так и о трехмерной форме наблюдаемых объектов. Лазерное сканирование и фотосъемка используются совместно, например, при трехмерной реконструкции зданий и построении трехмерной модели местности (Рис. 1).

Однако, совместное использование изображений и данных лазерного сканирования эффективно лишь в случае, когда известно точное взаимное расположение соответствующих сенсоров системы в общей системе координат. Но подобная информация, как правило, доступна с недостаточной точностью. Положение и ориентация системы, с которой производится съемка (например, автомобиля или самолета) определяется с помощью датчиков глобального позиционирования (GPS) и инерциальной системы (INS), подверженных сбоям и погрешностям в измерениях. По этой причине актуальной задачей является уточнение взаимного расположения датчиков системы с помощью алгоритмов компьютерного зрения. В литературе данная задача зачастую называется задачей сопоставления (регистрации) изображений и облаков трехмерных точек.

В рамках данной работы задача рассматривается как проблема уточнения положения и ориентации камеры в мировой системе координат при известном положении в пространстве каждой точки облака (т.е. данные лазерного сканирования считаются геопривязанными). Такая постановка задачи характерна для сценария, когда облако трехмерных точек получено заранее и требуется уточнить по нему параметры внешнего ориентирования камеры в процессе фотосъемки.

2. СУЩЕСТВУЮЩИЕ МЕТОДЫ

Существующие алгоритмы решения рассматриваемой задачи можно разбить на две группы. В первую группу входят методы, основанные на восстановлении разреженного облака трехмерных точек по набору изображений и последующем сопоставлении разреженного и плотного (полученного с лазерного сканера) облаков.

(а) (б)

Рис 1: (а) Изображение; (б) Построенная трехмерная модель. Пример из статьи [11]

Методы второй группы, напротив, работают с двумерным представлением плотного облака точек - как правило, в виде карт глубины (Рис. 2) или удаленности - и сопоставляют его с изображениями сцены. Таким образом, задача сводится к задаче сопоставления либо в трехмерном, либо в двумерном пространстве. Рассмотрим методы каждой группы подробнее.

Алгоритмы первой группы, как правило, получают разреженное облако трехмерных точек путем вычисления структуры из движения по набору изображений. Так, в работе [9] авторы используют инкрементальный подход к вычислению разреженной трехмерной модели сцены и метод связок [7] для уточнения параметров камеры и координат трехмерных точек. Сопоставление облаков трехмерных точек производится в два этапа: начальное сопоставление и последующее уточнение регистрации по плоскостям. Недостатком подхода является необходимость пользовательского ввода на этапе начального сопоставления.

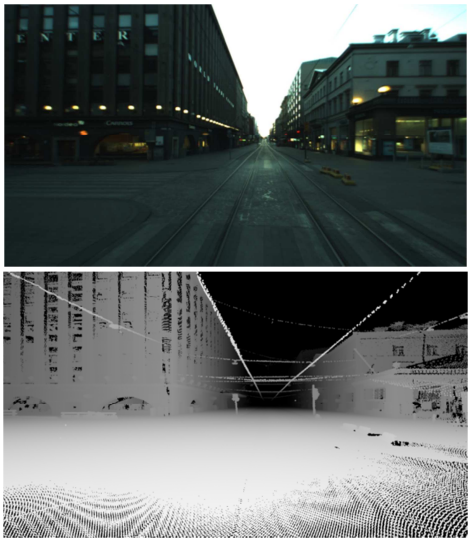


Рис 2: Изображение сцены (сверху) и карта глубины (снизу)

В статье [15] авторы используют алгоритм плотного стерео для восстановления трехмерной структуры сцены и метод ICP [16] для сопоставления облаков трехмерных точек. Такой подход требует наличия видеоданных достаточно высокого разрешения, и также не является полностью автоматическим: начальное приближение для метода ICP подбирается вручную.

Некоторые методы напрямую сопоставляют изображения с облаком трехмерных точек. В работе [15] это делается вручную для отдельных кадров видео, тогда как авторы [14] используют алгоритм, основанный на поиске и сопоставлении линейных объектов. Такой подход встречается и в более ранней работе [12], в которой для поиска прямых в облаке точек выполнялась сегментация плоскостей, и находились их пересечения.

Методы второй группы, основанные на сопоставлении в двумерном пространстве, в свою очередь, подразделяются на два типа: использующие геометрические особенности на изображениях и опирающиеся на статистические метрики. К числу первых относится, например, работа [5]. В ней производится сопоставление линий на изображении, найденных с помощью алгоритма [2], и линий на карте глубины. Минусом алгоритма является низкая скорость работы, а также предположение о наличии достаточно числа линий на изображениях. Представляет также интерес работа [8], в которой облако трехмерных точек проецируется на плоскость земли (метод разработан для регистрации изображений и облака точек, полученных с воздуха) и сформированное изображение сопоставляется с бинарным, представляющим собой наиболее характерные участки сцены при съемке сверху. Задача сопоставления формулируется в терминах минимизации энергии.

В отличие от методов, использующих отдельные геометрические особенности, второй тип алгоритмов двумерного сопоставления основан на вычислении некоторой статистики по всему изображению для оценки качества сопоставления. Как подчеркивают в [1], такой подход избавляет от необходимости искать особые точки, линии и прочие характерные элементы, которых может и не быть на изображении. Напротив, статистические методы учитывают также информацию с плоских и однородных участков сцен, таких как крыши, стены, поля и т.д. В [1] в качестве метрики сходства изображений применялась χ^2-статистика. Другой распространенной метрикой является взаимная информация, применявшаяся, например, в [3, 13].

Большое значение в контексте данной работы имеет статья [11], авторы которой используют в качестве метрики качества величину взаимной информации между изображением и картой глубины. Отталкиваясь от начального приближения, алгоритм сходится к некоторому решению методом симплексного спуска. Такой подход представляется наиболее перспективным. Он не требует восстановления трехмерных точек по изображению, не подразумевает пользовательского ввода и поиска геометрических особенностей в картах глубины. Однако его недостатком является работа лишь с одним изображением, тогда как обычно доступен целый набор снимков сцены. Целью данной работы является развитие метода [11] на случай использования одновременно пары изображений.

3. ПРЕДЛОЖЕННЫЙ АЛГОРИТМ

Идея данной работы - использовать подход из статьи [11], добавив в процесс оптимизации второе изображение и изменив оптимизируемую функцию так, чтобы дополнительно учитывать геометрические связи между снимками. Предложенный алгоритм состоит из двух основных шагов:

1. Поиск соответствующих точек на изображениях. Подчеркнем, что поиск производится один раз до запуска оптимизационного процесса, поскольку изображения не меняются в ходе оптимизации.
2. Итеративная оптимизация параметров внешнего ориентирования. В отличие от [11], в целевую функцию включается слагаемое, отвечающее за согласованность между фундаментальной матрицей и соответствующими точками, найденными на первом шаге.

Далее рассмотрим каждый шаг подробнее.

3.1 Поиск соответствующих точек на изображениях

Для обнаружения соответствующих точек используется стандартная схема сопоставления изображений. Сначала применяется детектор особых точек Харриса [6]. После этого окрестности найденных точек описываются с помощью дескриптора SIFT [10], инвариантного к масштабу и повороту. Соответствия между особыми точками устанавливаются на основе расстояния по дескриптору: для

каждой точки первого изображения выбирается ближайшая к ней точка второго изображения. При этом проверяется выполнение и обратного условия. Наконец, ошибочные соответствия удаляются с помощью алгоритма устойчивой оценки параметров RANSAC [4] с фундаментальной матрицей в качестве модели и суммой квадратов расстояний от точек до эпиполярных линий в качестве функции ошибки. Заметим, что фундаментальная матрица, вычисляемая на данном этапе, используется только для оценки правильности найденных соответствий.

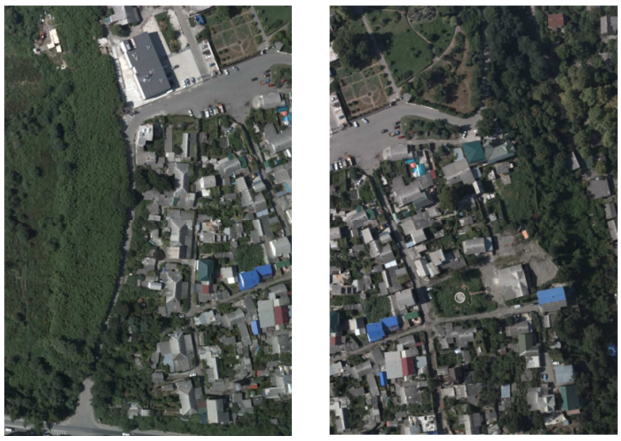


Рис 3: Пример тестовых изображений

Стоит отметить, что выбор уголков Харриса в качестве детектора особых точек обусловлен характером исходных данных – для тестирования в рамках данной работы использовались аэрофотоснимки постоянного масштаба (Рис. 3). В случае, например, наземной съемки мог бы потребоваться детектор, учитывающий масштаб. Однако это не повлияло бы на общую схему предлагаемого в работе алгоритма.

3.2 Оптимизация параметров камеры

Оптимизация параметров внешнего ориентирования камеры производится итеративно методом симплексного спуска. Целевая функция имеет следующий вид:

$$F = M(I_1, P_1) + M(I_2, P_2) - \lambda \cdot E(I_1, P_1, I_2, P_2) \qquad (1)$$

Здесь M – функция взаимной информации, E – функция ошибки соответствия особых точек, I_1, I_2 – изображения сцены, P_1, P_2 – параметры внешнего ориентирования камеры в мировой системе координат в моменты съемки каждого из двух изображений (более точно, $P_1 = [R_1|T_1], P_2 = [R_2|T_2]$, где R_1, R_2 – матрицы поворота, T_1, T_2 – вектора смещений), λ - весовой коэффициент.

Аналогично [11], взаимная информация M вычисляется для изображения и карты глубины. Она характеризует степень согласованности (взаимной зависимости) значений яркости и глубины сцены. В [11] показано, что функция достигает своего максимума при правильном положении и ориентации камеры. Подчеркнем, что изображение сцены остается неизменным в процессе оптимизации, однако карта глубины меняется, поскольку зависит от внешних параметров камеры P. Карта глубины строится путем проецирования точек облака на плоскость изображения. Предполагается, что внутренние параметры камеры (фокусное расстояние и координаты принципиальной точки) известны и не меняются, что соответствует большинству практических задач.

Карта глубины представляет собой полутоновое изображение, в котором яркость пикселя кодирует удаленность соответствующей трехмерной точки от камеры (зависимость линейная), см. Рис. 2. Те пиксели, в которые не была спроецирована ни одна трехмерная точка (это возможно при недостаточной плотности облака трехмерных точек), не учитываются при вычислении взаимной информации.

Функция E отражает согласованность между парами соответствующих точек, найденными на первом шаге, и фундаментальной матрицей F, вычисленной по текущим значениям параметров внешнего ориентирования P_1, P_2 (алгоритм вычисления фундаментальной матрицы при известных внешних и внутренних параметрах камеры приведен в [7]).

Пусть x – особая точка на первом изображении, а x' и $l' = x \cdot F$ - соответствующие ей точка и эпиполярная линия на втором изображении. Расстояние от точки x' до прямой l' характеризует согласованность пары точек $\{x, x'\}$ и фундаментальной матрицы F. Функция E вычисляется как среднее расстояние от точки до эпиполярной линии по всем точкам на обоих изображениях.

Коэффициент λ в формуле (1) необходим для приведения двух метрик различной природы к единому масштабу, он подбирается эвристически. Вопрос определения целевой функции, состоящей из однородных частей, является предметом будущего исследования. Отметим также, что предложенный подход можно рассматривать как одну из вариаций метода связок [7], с измененной целевой функцией. Таким образом, представляет интерес обобщение алгоритма на случай произвольного числа изображений.

4. РЕЗУЛЬТАТЫ

Тестирование алгоритма производилось на данных аэрофотосъемки (Рис. 3). Критерием качества служил процент успешных сопоставлений. Для каждого изображения были известны правильные значения параметров внешнего ориентирования камеры, поэтому для проверки успешности сопоставления использовался численный критерий (в работе [11] выполнялась визуальная проверка).

Для получения начального приближения в известные значения положения и ориентации камеры вносился случайный шум, что имитировало погрешность работы датчиков GPS и INS. Сопоставление считалось успешным, когда выполнялось следующее условие:

$$\sum_i \frac{\tilde{n}_i}{A_i} < t \sum_i \frac{n_i}{A_i},$$

где $n_i, \tilde{n}_i$ – абсолютные величины отклонений параметра i от правильного значения до и после сопоставления соответственно; A_i – максимальная величина (амплитуда) шума по параметру i; $i \in \{x, y, z, \omega, \varphi, \kappa\}$ – три координаты и три угла; t – порог, регулирующий, во сколько раз в среднем стали точнее значения параметров (точность должна возрасти не менее чем в $\frac{1}{t}$ раз).

При тестировании использовались четыре пары изображений. Для каждой пары тест повторялся 150 раз, для различных начальных приближений. При этом параметры камеры для одного из изображений фиксировались в правильном положении, а для второго – подвергались шуму и оптимизировались. Такой сценарий соответствует ситуации, когда сопоставление для одного из изображений уже было произведено любым из существующих методов, например

[11], и требуется уточнение параметров для остальных снимков. Одновременное уточнение параметров для обеих камер – предмет дальнейшего развития данной работы.

Тест	Успешные сопоставления	
	Предложенный метод	Метод [11]
1	57.3%	40.0%
2	58.6%	35.3%
3	45.3%	10.6%
4	50.6%	23.3%

Таблица 1. Результат сравнения предложенного и базового метода

Результаты экспериментального сравнения предложенного и базового метода представлены в Таблице 1. В ходе тестов использовались следующие значения параметров:

$$A_x = 40, A_y = 40, A_z = 40, A_\omega = 1, A_\varphi = 1, A_\kappa = 1, t = 0.5$$

Значения для координат указаны в метрах, для углов – в градусах. Для весового коэффициента было выбрано значение $\lambda = 30$.

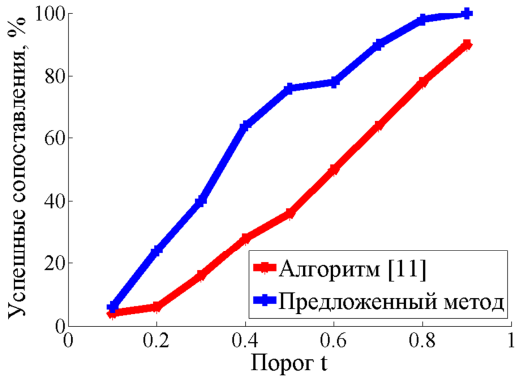


Рис 4: Зависимость доли успешных сопоставлений от значения порога t

На Рис. 4 представлен график изменения доли успешных исходов в зависимости от значения порога t. Как видно из результатов тестов, предложенный алгоритм превосходит базовый метод, что говорит о перспективности идеи данной работы.

5. ЗАКЛЮЧЕНИЕ

В данной работе был представлен новый алгоритм сопоставления изображений и облака трехмерных точек. Алгоритм развивает один из известных подходов к решению данной задачи [11] на случай нескольких изображений. За счет добавления в процесс оптимизации второго изображения на решение накладываются дополнительные ограничения, обусловленные геометрическими связями между различными снимками одной сцены. Показано, что такой подход превосходит базовый метод по доле успешно произведенных уточнений параметров.

6. БЛАГОДАРНОСТИ

Работа выполнена при поддержке федеральной целевой программы «Научные и научно-педагогические кадры инновационной России на 2009-2013 годы», контракт №1189.

7. ЛИТЕРАТУРА

[1] Boughorbal F. Registration and Integration of Multisensor Data for Photorealistic Scene Reconstruction // SPIE Proceedings. 2000. P 74-84.

[2] Canny J. A Computational Approach to Edge Detection // IEEE Transactions on Pattern Analysis and Machine Intelligence. 1986. 8. N 6. P 679–698.

[3] Collignon A., Maes F., Delaere D., Vandermeulen D., Suetens P., Marchal G. Automated Multimodality Medical Images Registration using Information Theory // Conference on Information Processing in Medical Imaging Proceedings. 1995. 14. P 263-274.

[4] Fischler M.A., Bolles R.C. Random Sample Consensus: A Paradigm for Model Fitting with Application to Image Analysis and Automated Cartography // Communications of the ACM. 1981. 24. N 6. P 381-395.

[5] Frueh C., Sammon R., Zakhor A. Automated Texture Mapping of 3D City Models with Oblique Aerial Imagery // International Symposium on 3D Data Processing, Visualization and Transmission. 2004. 2. P 396-403.

[6] Harris C., Stephens M. A. Combined Corner and Edge Detector // Alvey Vision Conference Proceedings. 1988. P 147–151.

[7] Hartley R., Zisserman A. Multiple View Geometry in Computer Vision Second Edition // Cambridge University Press. 2004.

[8] Kaminsky R., Snavely N., Seitz S., Szeliski R. Alignment of 3D Point Clouds to Overhead Images // IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops. 2009. P 63-70.

[9] Li Y., Low K. Automatic Registration of Color Images to 3D Geometry // Computer Graphics International (CGI). 2009. P 21-28.

[10] Lowe D. Distinctive Image Features from Scale-invariant Keypoints // International Journal of Computer Vision. 2004. 60. N 2. P 91-110.

[11] Mastin A., Kepner J., Fisher J. Automatic Registration of LIDAR and Optical Images of Urban Scenes // IEEE Conference on Computer Vision and Pattern Recognition. 2009. P 2639-2646.

[12] Stamos I., Allen P.K. Geometry and Texture Recovery of Scenes of Large Scale // Computer Vision and Image Understanding. 2002. 88. N 2. P 94-118.

[13] Viola P., Wells W. Alignment by Maximization of Mutual Information // International Conference on Computer Vision. 1997. 24. N 2. P 137 -154.

[14] Wolberg G., Zokai S. Multiview Geometry for Texture Mapping 2D Images onto 3D Range Data // IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR). 2006. 2. P 2293-2300.

[15] Zhao W., Nister D., Hsu S. Alignment of Continuous Video onto 3D Point Clouds // IEEE Transactions on Pattern Analysis and Machine Intelligence. 2005. 27. N 8. P 1305-1318.

[16] Znengyou Z. Iterative Point Matching for Registration of Free-form Curves and Surfaces // INRIA. 1994. 13. N 2. P 119-152.

Система быстрого обнаружения параметрических кривых на серых и цветных изображениях с контролем достоверности

Алексей Левашов, Дмитрий Юрин
Факультет вычислительной математики и кибернетики
Московский государственный университет, Москва, Россия
alexeylevashov89@gmail.com , yurin@cs.msu.su

Аннотация

Предлагается многоэтапная система поиска параметрических кривых на изображении. Сначала производится поиск граничных линий, затем, после ряда подготовительных процедур, результат сохраняется в виде цепочек связанных пикселей. Эти цепочки анализируются целиком и по-фрагментно. Анализ каждого фрагмента на соответствие модели выполняется сначала на основе рандомизированных методов, что позволяет достичь высокого быстродействия, и, в случае успешного прохождения такого теста, производится точная оценка параметров модели методом наименьших квадратов и достоверности гипотезы на основе критерия хи-квадрат. По-фрагментный анализ, фазы роста фрагмента и слияния фрагментов, постоянный контроль достоверности в смысле хи-квадрат позволяют находить параметрические кривые различных типов одновременно, а из набора моделей выбирать наиболее простую, описывающую кривую с точностью, соответствующей погрешности исходных данных.

Ключевые слова: *edge detector, параметрические кривые, векторизация.*

ВВЕДЕНИЕ

Зачастую в задачах компьютерного зрения целесообразно применять упрощенный подход, основанный на поиске известных ориентиров. В роли таких ориентиров могут выступать штанги, люки, ребра корпуса – т.е. объекты, видимые в виде простых геометрических фигур. Таким образом, задача сводится к быстрому и достоверному поиску на изображении кривых, описываемых аналитическим выражением с небольшим количеством параметров и оценке этих параметров.

Существует множество методов, решающих данную проблему. Первым является метод Хафа [4], который оказался очень не эффективным при поиске многопараметрических кривых. Существенным улучшением данного подхода является использование разреженных массивов, а также применение рандомизированных выборок [1, 5, 8]. Однако рандомизированные методы пропускают кривые на сложных изображениях из-за низкой вероятности того, что случайная выборка точек принадлежит одной кривой. Оба подхода плохо отслеживают кривые малой длины, а также ориентированы на поиск кривых только одного типа. Другой более точный подход к решению данной задачи основан на использовании бимлетов [2]. Однако подобные методы также ориентированы на нахождение кривых только одного типа.

В настоящей работе предлагается метод, анализирующий цепочки связанных пикселей – граничных линий, что становится возможным при использовании высококачественных детекторов граничных точек. Путем рандомизированных проверок отбрасываются граничные линии, заведомо не удовлетворяющих ни одной из рассматриваемых гипотез. Согласно оставшимся гипотезам оценка параметров выполняется методом наименьших квадратов, достоверность контролируется методом хи-квадрат.

1. АЛГОРИТМ

Общая схема алгоритма состоит из нескольких шагов. Сначала применяется детектор граничных линий Канни – для изображений в оттенках серого, Дизензо-Кумани – для цветных изображений [6] с вычислением производных путем свертки с производными функции Гаусса и процедурой подавления не максимальных точек [3]. При использовании этих детекторов получаются линии с малым числом разрывов и, за исключением незначительного числа точек, – шириной в 1 пиксель. После выполнения данного шага получаем изображение с граничными линиями. Следующим этапом является векторизация, т.е. граничные точки собираются в виде списка кривых, а кривые – в виде списка точек. Каждая точка представляется в памяти как пара координат, а также записывается направление и величина градиента в данной точке. Весь дальнейший анализ ведется только с таким списком границ. Тут есть две проблемы, которые мешают этапу векторизации. Первая – граничные линии могут быть не единичной толщины, что представляет трудность в записывании их в массив. Вторая – возможны точки ветвления, тем самым придется хранить список векторов в виде дерева или даже графа. Однако в точках ветвления точность детектирования границ обычно невысока. Поэтому точки ветвления удаляются вместе с небольшой окрестностью, а линии не единичной толщины утончаются.

Таким образом, получаем схему алгоритма:

1) Детектор граничных линий [6].
2) Утончение граничных линий [9].
3) Удаление точек ветвления.
4) Векторизация изображения.
5) Анализ каждой граничной линии в отдельности.
6) Объединение параметрических кривых с близкими значениями параметров.

Шаг 6 не рассматривается подробно, из-за ограничений объема статьи, но он важен, если кривые имеют разрывы.

1.1 Удаление точек ветвления

Пусть P – пиксель на изображении. Считаем, что $P=1$, если P – граничная точка, 0 – иначе. Вокруг каждой граничной точки P берется окно 3x3 пикселя $P_1, P_2, ..., P_9$ (Рис 1)

и вычисляются характеристики $J(P_1)=\sum_{i=2}^{9}P_i$ и $K(P_1)=$количеству паттернов 01 в последовательности $P_2,...,P_9,P_2$. Точка удаляется вместе с ее окрестностью, если выполняется условие $J(P_1)\geq 5 \ \vee \ K(P_1)>2$. В отличие от [5], точки удаляются сразу при сканировании. Если в процессе сканирования хотя бы одна точка была удалена, следует потом просканировать еще раз.

$P_9(i-1,j-1)$	$P_2(i-1,j)$	$P_3(i-1,j+1)$
$P_8(i,j-1)$	$P_1(i,j)$	$P_4(i,j+1)$
$P_7(i+1,j-1)$	$P_6(i+1,j)$	$P_5(i+1,j+1)$

Рис 1: Индексация пикселей в окне 3x3

1.2 Векторизация

Переход от пиксельных данных к векторному представлению существенно упрощает дальнейшую работу. Целесообразно хранить данные как список списков – список кривых, каждая из которых является списком точек. Здесь требуется произвольный доступ и хранение в виде массива. Было применено следующее решение. Память выделялась массивами по 2^{16}=65536 элементов по мере исчерпания. Непрерывная адресация (обобщенный массив) осуществляется путем использования старших бит индекса для выбора массива, а 16 младших – для выбора элемента. На этой памяти выделялись элементы для всех списков. По окончании сбора точек, данные сортировались так, что точки различных кривых оказывались последовательно размещенными в обобщенных массивах, последовательно одна кривая за другой.

Если предположить, что мы ищем исключительно гладкие кривые, то, если на граничной линии есть уголки, имеет смысл анализировать только сегменты между уголками. Находить такие уголки можно с помощью детектора уголков Харриса или Форстнера со скользящим окном, однако можно ускорить алгоритм, рассматривая уже собранные списки. Для каждого граничного пикселя по его соседям строим ковариационную матрицу градиентных направлений и по второму собственному числу, если оно больше некоторого порога и является локальным максимумом, выделяем его как уголок. Как показано на Рис 2 карты уголков, построенные таким методом, остаются плавными.

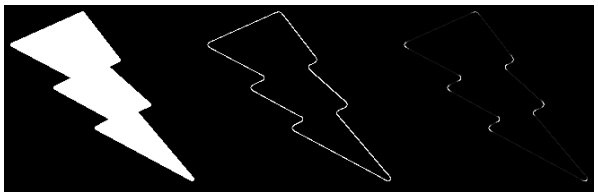


Рис 2: Карта угоков

1.3 Алгоритм анализа граничной линии

Т.к. одна граничная линия может состоять из нескольких параметрических кривых, то важной задачей является быстро и достоверно разбить данную линию на модели. Эту задачу выполняет *Алгоритм 1.*

Пусть P – массив точек данной граничной линии. Под сегментом $P[n_1,...,n_2]$ будет пониматься набор точек из P с индекса n_1 и заканчивая индексом n_2. Под функцией $\texttt{size}(P)$ понимается число элементов массива P.

Пусть $\mathbf{M}$ – это заданный априори набор математических моделей параметрической кривой, на соответствие которым проверяются фрагменты кривой. В выражении $h\in\mathbf{M}$ под h может пониматься окружность, прямая, эллипс и любая другая аналитическая кривая. Под конкретной параметрической кривой понимается $\{h, params\}$, где $params$ – это параметры кривой модели h. Для каждого такого h необходимо знать следующие алгоритмы:

1) $\texttt{LeastSquareMatching}(h,n_1,n_2)$ – метод наименьших квадратов для точек из сегмента $P[n_1,...,n_2]$, который находит оптимальные параметры для h.
2) $\texttt{ChiSquareFitting}(\{h, params\},n_1,n_2)$ – критерий достоверности χ^2 модели h с параметрами $params$ для сегмента $P[n_1,...,n_2]$.
3) $B(h)$ – минимальное необходимое количество точек для построения модели h.
4) $\texttt{FindParams}(P',h)$ – нахождение параметров модели h по P' точкам, при чем $\texttt{size}(P')=B(h)$.
5) $a\in m$ – способ определить лежит ли точка a на параметрической кривой $m=\{h, params\}, h\in\mathbf{M}$.

Пусть S – упорядоченный набор $S_0,...,S_{n-1}$ разделителей, которые делят массив на сегменты. Разделитель это индекс элемента в массиве. Множество точек между двумя ближайшими разделителями – это сегмент массива. Считаем, что $S_{-1}=S_0$ и $S_n=S_{n-1}$. Изначально S состоит из 2-х элементов: начального и конечного индексов P.

Все найденные кривые будут записаны в M. M – это множество троек $\{m,n_1,n_2\}$, где n_1 – индекс начала сегмента, n_2 – индекс конца сегмента, а $m-\{h, params\}, h\in\mathbf{M}$ параметрическая кривая, описывающая данный сегмент.

Идея *Алгоритма 1* заключается в следующем: сначала рассматривается все точки P, и пробуется с помощью *Алгоритма 3.* найти параметрическую кривую описывающий данный сегмент. Если такого сделать не удается, то сегмент делится пополам и процедура повторяется для 2-х половинок. Как только был найден сегмент, для которого нашлась оптимальная кривая, то выполняются *Алгоритм 2a* и *Алгоритм 2b* для определения точных границ сегмента для найденной кривой. Деление пополам продолжается до тех пор, пока сегмент не будет меньше $C_{\min}$ – константы, определяющей минимальное количество точек в сегменте.

Алгоритм 1. $\texttt{Separation}()$: Разбиение P на сегменты, удовлетворяющие моделям.

1. $S \leftarrow [0, size(P)-1]$
2. $M \leftarrow \{\}$
3. **while** $\exists i : (S_{i+1} - S_i > C_{\min} \wedge !\exists m : \{m, S_i, S_{i+1}\} \in M)$
4. **do for** $i = 0,\ i < size(S) - 1$
5. $m \leftarrow$ `FindParamCurve`(S_i, S_{i+1})
6. **if** m `найдена` **then**
7. $n_1 \leftarrow$`ExpandLeft`(m, S_{i-1}, S_i)
8. $n_2 \leftarrow$`ExpandRight`(m, S_{i+1}, S_{i+2})
9. $S_i \leftarrow n_1$
10. $S_{i+1} \leftarrow n_2$
11. $M \leftarrow M \cup \{m, n_1, n_2\}$
12. **else**
13. $S \leftarrow [S_0, ..., S_i, \frac{S_i + S_{i+1}}{2}, S_{i+1}, ..., S_{n-1}]$

В процессе деления сегментов пополам, скорее всего будет обнаружена только часть кривой, и целесообразно расширить границы найденного сегмента так, чтобы расширенный сегмент описывал кривую максимальной длины. C_r – количество случайных точек, которое берется при расширении сегмента. В *Алгоритме 2a* индекс n_2 – текущее положение левой границы сегмента для кривой, а n_1 - граница максимально возможного расширения слева.

Алгоритм 2a. `ExpandLeft`(m, n_1, n_2):

1. **if** $(n_2 - n_1) \le 1$
2. **then return** n_2
3. $i \leftarrow \frac{n_1 + n_2}{2}$
4. **while** $i < n_2 - 1$ **do**
5. **if** $P_i \in m$ **then**
6. $T_r \leftarrow C_r$ случайных точек из $P[i, ..., n_2]$
7. **if** $\forall a \in T_r : a \in m$ **then**
8. **if** $\forall a \in P[i, ..., n_2] : a \in m$ **then**
9. $n_1 \leftarrow 2i - n_2$
10. $n_2 \leftarrow i$
11. **goto 1**
12. $i \leftarrow \frac{i + n_2}{2}$
13. **return** n_2

В цикле 4-12 пытаемся сдвинуть правую границу на $(n_2 - n_1)/2^n$, проверяя при этом, принадлежат ли присоединяемые точки к m. После добавления в строчках 9 – 11 повторяем алгоритм для уточнения границы слева от новой найденной.

Алгоритм 2b `ExpandRight`(m, n_1, n_2) Аналогичен *Алгоритму 2a*: n_1 - индекс правой границы найденного сегмента, а n_2 - индекс максимально возможного смещения границы.

В *Алгоритме 3* находится оптимальная кривая для сегмента $P[n_1, ..., n_2]$. C_{χ^2} – степень достоверности критерия χ^2.

Алгоритм 3. `FindParamCurve`(n_1, n_2): Определение модели, описывающий сегмент $P[n_1, ..., n_2]$

1. $m \leftarrow$ *пустая модель*
2. $T_{\chi^2} \leftarrow 0$
3. **for each** $h \in \mathbf{M}$ **do**
4. **if** QuickTest(h, n_1, n_2) **then**
5. $params \leftarrow$ LeastSquareMatching(h, n_1, n_2)
6. $T_{\chi^2}' \leftarrow$ ChiSquareFitting$(\{h, params\}, n_1, n_2)$
7. **if** $T_{\chi^2}' > C_{\chi^2}$ **then**
8. **if** $T_{\chi^2}' > T_{\chi^2}$ **then**
9. $T_{\chi^2} \leftarrow T_{\chi^2}'$
10. $m \leftarrow \{h, params\}$
11. **return** m

В *Алгоритме 4* происходит быстрая проверка, может ли сегмент $P[n_1, ..., n_2]$ описываться кривой типа h. Используются константы C_q – количество тестов быстрой проверки и C_q^{accept} – минимальное необходимое количество пройденных тестов быстрой проверки. В алгоритме берется случайным образом $B(h)+1$ точек из текущего сегмента, по $B(h)$ точкам строится кривая типа h и проверяется, лежит ли дополнительная точка на кривой. Так делается C_q раз.

Алгоритм 4. `QuickTest` (h, n_1, n_2): Быстрая проверка гипотезы h на сегменте $P[n_1, ..., n_2]$

1. $count \leftarrow 0$
2. **for** $i = 0, i < C_q$ **do**
3. $P' \leftarrow B(h)$ `точек из` $P[n_1, ..., n_2]$, `выбранных случайно`
4. $c \leftarrow$ `случайно выбранная точка из` $P[n_1, ..., n_2]$

5. $params \leftarrow \texttt{FindParams}(P', h)$

6. **if** $c \in \{h, params\}$ **then**

7. $count \leftarrow count + 1$

8. **return** $count > C_q^{accept}$

2. РЕЗУЛЬТАТЫ

Временные эксперименты были сделаны на процессоре Intel Core 2 Duo 2.00 GHz, оперативной памяти 2 Гб. Искались одновременно две модели: прямая и окружность. Времена работы алгоритма и его частей представлены в Таблице 1 в сравнении с алгоритмом [1]. Прочерком показаны тесты, которые алгоритм [1] не выполнил. Не было замечено чтобы разработанный алгоритм пропускал искомые кривые за исключением случаев когда их форма была сильно искажена на стадии детектирования граничных линий.

3. ЗАКЛЮЧЕНИЕ

Разработана система поиска параметрических кривых на изображениях. Сравнение с алгоритмом [1] показало близкие времена работы, но предложенный метод находит одновременно различные типы кривых и не имеет ограничений по сложности изображения. Комбинация рандомизированных проверок с методом наименьших квадратов и критерием хи-квадрат обеспечивает высокое быстродействие без потери надежности. Алгоритм не имеет ограничений на длины искомых кривых.

4. БЛАГОДАРНОСТИ

Работа выполнена при поддержке ФЦП «Научные и научно-педагогические кадры инновационной России» на 2009 – 2013 годы и гранта РФФИ 09-07-92000-ННС_а.

ЛИТЕРАТУРА

[1] Chen T.C., Chung K.L., *An Efficient Randomized Algorithm or Detecting Circles. // CVIU 2001, V.83, P.172–191.*

[2] Donoho D.L., Huo X., Jermyn I., Jones P., Lerman G., Levi O., Natterer F. *Beamlets and Multiscale Image Analysis //In Multiscale and Multiresolution Methods, Springer 2001, P.149-196.*

[3] Devernay F. *A Non-Maxima Suppression Method for Edge Detection with Sub-Pixel Accuracy //INRIA Tech. Report RR-2724, 20 p, 1995.*

[4] Duda R., Hart P. *Use of the Hough Transformation to Detect Lines and Curves in Pictures. Comm. ACM 15: 1 – 15, 1972.*

[5] McLaughlin R.A. *Randomized Hough Transform: better ellipse detection // IEEE TENCON - Digital Signal Processing Applications, V. 1, P. 409-414, Nov. 1996.*

[6] Pratt W.K. *Digital Image Processing: PIKS Scientific inside (4th ed.) // Wiley-Interscience, John Wiley & Sons, Inc., Los Altos, California, 2007, 782 p.*

[7] Triggs B., Sdika M. *Boundary Conditions for Young - van Vliet Recursive Filtering // IEEE Transactions on Signal Processing, V.54, No.5, May 2006.*

[8] Xu L., Oja E. *Randomized Hough Transform // in Encyc. of Artif. Intell., Ed.By: J.Ramón, R.Dopico; J.Dorado; A.Pazos, IGI Global publishing comp., 2008, P.1354—1361.*

[9] Zhang T.Y., Suen C.Y. *A fast parallel algorithm for thinning digital patterns // Communications of the ACM, V. 27, No. 3, P. 236--239, 1984.*

Об авторах

Левашов Алексей Евгеньевич – студент магистратуры факультета Вычислительной математики и кибернетики Московского государственного университета. alexeylevashov89@gmail.com

Юрин Дмитрий Владимирович, к.ф.-м.н., с.н.с. лаборатории Математических методов обработки изображений факультета Вычислительной математики и кибернетики Московского государственного университета им. М.В.Ломоносова. yurin_d@inbox.ru

	Размеры Изображения	Кол-во точек	Кол-во списков	Время работы, мс									Изображения
				1	2	3	4	5	6	7	8	9	
1	256×256	13063	1511	11	4	3	6	8	21	5	54	3	
2	256×256	11839	1473	11	3	3	5	7	18	7	55	4	
3	256×256	9167	1000	11	2	3	4	5	14	6	20	0	
4	256×192	7938	689	9	2	3	8	5	18	5	20	2	
5	559×559	39076	3034	58	14	10	13	8	45	3	-	-	
6	375×486	27881	2544	34	7	6	11	17	41	2	-	-	
7	1000×667	137387	21235	139	68	40	59	74	241	2	-	-	
8	1024×1024	228526	30932	372	106	88	85	128	407	0	-	-	

Таблица 1. Временные показатели. Пронумерованные колонки содержат времена работы (1) детектор граничных линий (2) утончение линий, (3) удаление узлов, (4) векторизация, (5) нахождение кривых, (6) общее время исполнения предложенного алгоритма без детектора граничных линий, (7) количество окружностей найденных нашим алгоритмом, (8) поиск окружностей [1], (9) количество окружностей найденных алгоритмом [1].

АЛГОРИТМ НАХОЖДЕНИЯ РАДУЖКИ ДЛЯ СИСТЕМЫ ОТСЛЕЖИВАНИЯ НАПРАВЛЕНИЯ ВЗГЛЯДА НА ОСНОВЕ ДОСТУПНОЙ ВИДЕОАППАРАТУРЫ

Иван Малин
Факультет прикладной математики и физики
Московский авиационный институт, Москва, Россия
ivan.malin@gmail.com

Аннотация

Большое количество важной информации о человеческом поведении, интересах и распределении внимания может быть получено на основе анализа траектории движения его взгляда. Системы отслеживания направления взгляда используются с растущей популярностью в различных отраслях науки и техники. Однако большинство современных систем являются слишком дорогими и требуют специального оборудования. В данной работа предлагаются новые методы, позволяющие определить точку взгляда пользователя, применяя только доступную аппаратуры. Под доступной аппаратурой понимаются веб-, фото и видеокамеры бытового уровня.

Ключевые слова: *отслеживание взгляда, распознавание радужки, компьютерное зрение*

1. ВВЕДЕНИЕ

В настоящее время количество информации, представленной в сети Интернет, огромно, и продолжает расти с увеличивающимися темпами. Для удобства поиска и работы с ними, эти данные должны быть представлены в удобной и логичной форме. Для определения, насколько удобен для пользователя интерфейс программы или веб-сайта, проводятся исследования юзабилити. Одним из ключевых понятий подобных исследованиях является распределение внимания пользователя. В связи с этим, одним из широко используемых инструментов при проведении юзабилити-тестов являются системы отслеживания направления взгляда. Следует отметить, что это только одно из наиболее очевидных приложений подобных систем, также они используются в задачах человеко-машинного взаимодействия, маркетинговых и медицинских исследованиях, тренировочных симуляторах, системах виртуальной реальности и компьютерных играх. Большинство современных реализаций таких устройств нуждается в специальном оборудовании для корректной работы, например, инфракрасных камерах и светодиодах, стереоскопических камерах и прочих спецсредствах. Целью нашей работы является разработка метода получения точки взгляда, который может быть применен в системе, использующей доступное оборудование: веб-, фото и видеокамеры бытового уровня.

Задача определения точки взгляда по видеозаписи глаза может быть разделена на две подзадачи: выделение ряда признаков на кадре и построение функции отображения пространства этих признаков в координаты точки взгляда на экране. Разные методы предлагают различные признаки для выделения. Ими могут быть блики на роговице, создаваемые инфракрасными светодиодами (иногда установленными по определенному геометрическому шаблону), центр и границы зрачка или радужки, уголки глаз и т. д.

Наши исследования наиболее популярных методов показали, что наилучшим признаком для выделения в условиях использования доступной видеоаппаратуры является граница радужки. Зрачок не подходит в силу низкого контраста с радужной оболочкой. Фактически, получить достаточный контраст между зрачком и радужкой (особенно для людей с темными глазами) не представляется возможным без использования дополнительной подсветки, направленной на глаз. Это возможно сделать при работе в инфракрасном диапазоне, однако в видимом диапазоне направленный свет может повредить глаз.

На рисунке 1 представлены изображения глаза в разных условиях освещения.

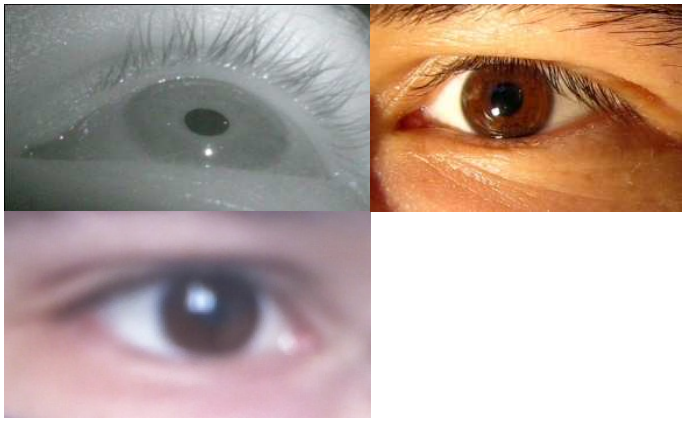


Рис. 1. Глаз в инфракрасном диапазоне, в видимом спектре с дополнительной подсветкой, в условиях естественного освещения.

В ходе нашего исследования было испробовано несколько известных методов для выделения радужки, включая метод

Хаффа [3] и пороговую бинаризацию. Однако при их применении не была достигнута требуемая устойчивость к плохим условиям освещения. Пороговая бинаризация не подходит для выделения радужки в связи с невозможностью подобрать корректное значение порога, отделяющее радужку от век и теней около глаза. Метод Хаффа показал плохие результаты на сильно зашумленных и размытых изображениях, а также потребовал тонкой настройки параметров для выделения границ методом Кенни.

2. ПРЕДЛАГАЕМЫЙ МЕТОД

2.1. Поиск окружности

Сначала рассмотрим более подробно задачу нахождения окружности на изображении. Будем рассматривать полутоновое изображение как функцию яркости, определенную на *прямоугольнике* $I(x,y), x \in [0..W], y \in [0..H]$. Одним из способов нахождения окружности является определение максимума функционала $\Phi(I, x0, y0, r) = \int_{O_{x0,y0,r}} \nabla I \cdot \bar{n}\, dl$, где $O_{x0,y0,r}$ является окружностью радиуса r с центром в точке $(x0, y0)$, а $\bar{n}$ – нормализованный радиус вектор, имеющий начало в $(x0, y0)$ и конец в (x, y). Другими словами, мы хотим найти таку окружность, что в каждой ее точке направление радиус вектора от ее центра к этой точке будет наиболее схоже с направлением градиента яркости изображения в этой же точке.

В дискретном пространстве будем рассматривать изображение как $(i,j) = I_{ij}, i = 1..H, j = 1..W$. Сначала найдем градиент изображения – рассчитаем вектор надиента яркости для каждого пиксела – путем свертки изображения с двумя масками Собела [3] горизонтального и вертикального направлений. Рисунок 3 показывает градиент изображения. Направление градиента на рисунке закодировано цветом, аналогично кодированию цветом комплексных значений в работе [4] – тон пиксела кодирует направление вектора, а яркость – длину. Будем обозначать градиент изображения как $\nabla I(i,j)$.

Затем насчитаем две маски одинакового размера – градиентную маску M_{ij} и радиальную R_{ij}. Градиентная маска – это двухмерный массив, каждый пиксел которого содержит двухмерный вектор единичной длины, направленный из центра маски к этому пикселу. Каждый пиксел радиальной маски содержит расстояние между центром маски и данным пикселом.

$$M_{ij} = \left(i - \frac{M_h}{2}, j - \frac{M_w}{2}\right), i = 1..M_W, j = 1..M_H$$

$$R_{ij} = \sqrt{\left(i - \frac{M_h}{2}\right)^2 + \left(j - \frac{M_w}{2}\right)^2}, i = 1..M_W, j = 1..M_H$$

Здесь i обозначат строку, j – столбец, M_h – высоту маски и M_w – ширину маски. После этого насчитывается куб данных $C_{ijr}, i = 0..H-1, j = 0..W-1, r = r_{min}..r_{max}$:

$$C_{ijr} = \sum_{k=1..M_h, l=1..M_w \,:\, r=R_{kl}} M_{kl} \cdot \nabla I\left(i - \frac{M_h}{2} + k, j - \frac{M_w}{2} + l\right)$$

Затем мы находим максимальное значение среди всех элементов кубы, и координаты i, j и r максимального элемента дадут нам координаты центра и радиус искомой окружности. Другими словами, это просто организуемый способ исчерпывающего перебора наилучших параметров окружности.

На рисунке 2 изображены градиентная и радиальная маски.

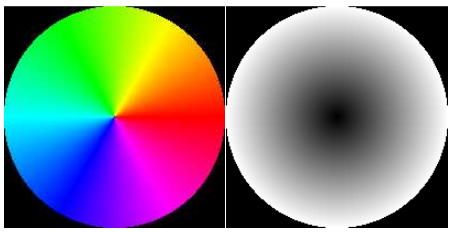


Рис. 2. Градиентная и радиальная маски.

На рисунке 3 изображены результаты работы алгоритма поиска окружности на реальном изображении глаза.

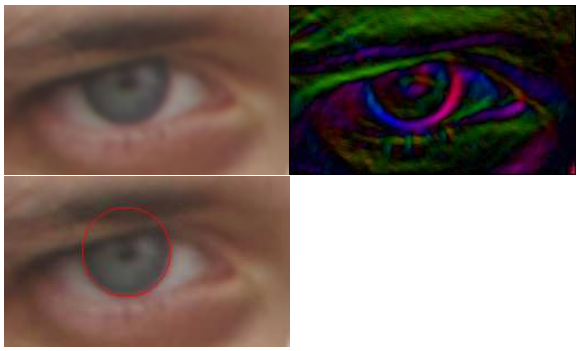


Рис. 3. Исходное изображение, градиент изображения, найденная окружность.

Мы также протестировали алгоритм на сильно зашумленных и размытых синтезированных изображениях, где он также показал хороший результат. Один из них представлен на рисунке 4.

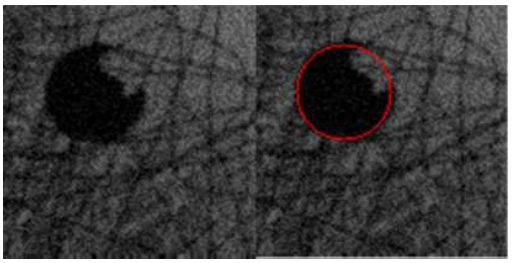


Рис. 4. Результат работы алгоритма определения окружности на сильно размытом и зашумленном синтезированном изображении.

2.2. От окружности к эллипсу

В общем случае проекцией окружности на плоскость является эллипс. Поэтому описанный выше метод не может дать точное положение радужки, если угол между оптическими осями глаза и камеры велик. Однако его можно использовать для поиска некоторого приближения искомого эллипса. Уточнение параметров эллипса может быть осуществлено путем простого исчерпывающего перебора в пространстве параметров с некоторым заданным шагом. При этом целевая функция похожа на используемую при поиске окружности – это свертка с градиентной маской. Но теперь у нас есть целый набор градиентных масок вместо одной – по одной для каждой комбинации параметров эллипса. Мы так же совмещаем центры масок последовательно с каждым пикселом изображения и суммируем скалярные произведения векторов градиентной маски и изобажения для всех пикселов, накрытых маской. Маска и ее положение с максимальным откликом определяют параметры наилучшего эллипса.

На рисунке 5 изображены некоторые градиентные маски, используемые для поиска эллипса. Рисунок 6 показывает результат работы алгоритма поиска эллипса. Красная окружность на рисунке показывает предварительно найденное приближение, результирующий эллипс нанесен зеленым цветом.

Рис. 5. Эллиптические градиентные маски.

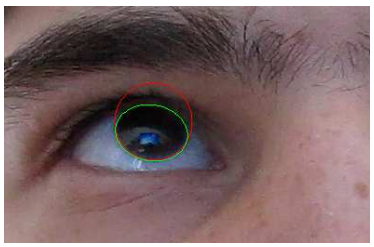


Рис. 6. Результат работы алгоритма определения эллипса.

Данная задача отнимает наибольшую часть времени вычисления в процессе поиска. Эллипс определяется пятью независимыми параметрами, например, координатами точки центра, длиной наибольшей оси, отношением длин осей и углом поворота наибольшей оси. Вычисление требует $O(M \cdot N \cdot m^2 \cdot K)$ операций умножения, где M и N определяют размеры области поиска центра эллипса, m обозначает длину маски в пикселах, а K – количество используемых масок.

В приведенном выше примере мы уменьшили размер изображения таким образом, чтобы радиус радужки примерно был равен 25 пикселам. Поиск осуществлялся в области 20*20 пикселов вокруг центра предварительно найденной окружности. Было применено около 2000 градиентных масок, соответствующих различным параметрам эллипса. Процесс поиска занял 500 мс на процессоре AMD Turion 2GHz.

В дальнейшей работе над методом следует уделить внимание его ускорению. В частности, следует опробовать методы численной оптимизации, работающие быстрее полного перебора.

2.3. Функция отображения

Для определения координат точки взгляда на экране по координатам центра зрачка в видеокадре, требуется определить соответствующее преобразование. Для построения такого преобразования требуется сперва провести процедуру калибровки. Пользователю предлагается посмотреть последовательно на девять калибровочных точек: четыре по краям экрана, четыре в серединах сторон и одна в ентре. В результате данной процедуры мы получим девять пар $\langle S_i, F_i \rangle, i = 1..9$, где S_i – точки взгляда на экране, а F_i – центр зрачка на видеокадре (точка S_9 совпадает с точкой O).

Теперь определим две полярные системы координат. Одна из них лежит в плоскости экрана, другая – в плоскости кадра. Началом координат первой системы является центр экрана. Началом второй – точка F_9 (см рис. 7). Будем обозначать центр зрачка в кадре как (φ, L), а точку взгляда на экране как (φ', L'). L и L' являются растояниями до точки от начала координат соответствующей системы. φ и φ' – углы поворота радиус-вектора. Для системы координат кадра угол отсчитывается по часовой стрелки относительно направления OF_1., для системы координат экрана – против часовой относительно OS_1.

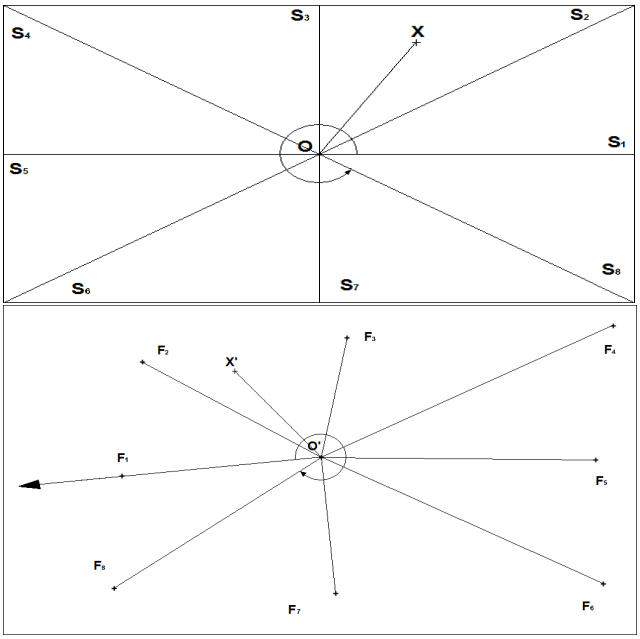


Рис. 7. Экранная система координат и система координат кадра.

Теперь определим простое правило для преобразования координат из системы кадра в систему снимка.

$$\varphi = \frac{\varphi' - \varphi'_i}{\varphi'_{i+1} - \varphi'_i}(\varphi_{i+1} - \varphi_i) + \varphi_i$$

$$L = \left(\frac{\varphi' - \varphi'_i}{\varphi'_{i+1} - \varphi'_i} \cdot \frac{L_{i+1}}{L'_{i+1}} + \frac{\varphi'_{i+1} - \varphi'}{\varphi'_{i+1} - \varphi'_i} \cdot \frac{L_i}{L'_i}\right) L'$$

3. РЕЗУЛЬТАТЫ ЭКСПЕРИМЕНТОВ

Приведем некоторые экспериментальные результаты работы наших алгоритмов. В качестве входных данных использовалась видеозапись с разрешением 640x480 пикселов, сделанная фотокамерой Canon Powershot SX20. Кадры были уменьшены до размера 320x240 пикселов, и вручную была выделена область обработки вокруг глаза для увеличения скорости детектирования радужки. Запись видео производилась в условиях электрического освещения лампой накаливания без дополнительной подсветки. На рисунке 8 показаны результаты работы алгоритма выделения радужки.
Рисунок 8 показывает работу функции отображения координат. Черными квадратами обозначены точки на экране, в которые предлагалось посмотреть пользователю (включая калибровочные), а серыми линиями нанесена вычисленная траектория точки взгляда. Пользователь находился в 60 см от экрана разрешением 1280x800 пикселов. Максимальная ошибка составила 95 пикселов, что составляет 6.2% от диагонали экрана. Среднеквадратическая ошибка составила 48 пикселов (3.2% диагонали экрана).

4. ДАЛЬНЕЙШАЯ РАБОТА

Результаты экспериментов показали, что предлагаемые алгоритмы могут быть использованы для простой и дешевой системы отслеживания направления взгляда, работающей на основе видеоаппаратуры бытового уровня. Однако все еще имеется ряд проблем, требующих решения. Во-первых, метод работает медленно, что описано в разделе, посвященном определению эллипсов. Возможно, использование подходящего метода стохастической оптимизации вместо перебора по сетке для определения параметров эллипса позволило бы ускорить работу. Также алгоритм легко поддается распараллеливанию. Второй большой проблемой является необходимость соблюдать неподвижность головы пользователя для корректной работы функции отображения. Использование более сложной геометрической модели и отслеживание движений головы потенциально может позволить системе не ограничивать свободу движений пользователя.

ЛИТЕРАТУРА

[1] Ying Huang, Zhiliang Wang, and An Ping "Non-Contact Gaze Tracking with Head Movement Adaptation based on Single Camera", *World Academy of Science, Engineering and Technology 59 2009*

[2] Dong Hyun Yoo Myung Jin Chung, "Non-contact Eye Gaze Estimation System using Robust Feature Extraction and Mapping of Corneal Reflections"

[3] Gary Bradksi, Adrian Kaehler, "Learning OpenCV", *O'Reilly Media, 2008*

[4] Bjorn Johansson**, "**Rotational Symmetries a Quick Tutorial", *2003*

Об авторе

Иван Малин – аспирант кафедры вычислительной математики и программирования Московского авиационного института.

e-mail: ivan.malin@gmail.com

Нейросетевой алгоритм обнаружения малоразмерных объектов на облачных фонах

Н.Ю. Шубин, В.С Муравьев, С.И. Муравьев
Факультет автоматики и информационных технологий в управлении
Рязанский государственный радиотехнический университет, Рязань, Россия, aitu@rsreu.ru

Аннотация

Постоянное совершенствование датчиков изображений и средств вычислительной техники приводит к расширению сферы практического применения методов и алгоритмов анализа изображений. Так, комплексы обработки изображений реального времени начали активно применяться в системах автоматического обнаружения и сопровождения объектов. Областями их использования являются: управление дорожным движением, контроль производственных процессов, исследование Земли из космоса, робототехника, медицина и целый ряд других.

Особое внимание в последние два десятилетия уделяется проблеме обнаружения малоразмерных воздушных объектов. В работе предложен нейросетевой алгоритм решения данной задачи, ориентированный на анализ изображений в темпе их поступления от видеодатчика. Описана процедура обучения нейронной сети, приведены результаты сравнительных экспериментальных исследований.

Ключевые слова: нейронная сеть, обнаружение, малоразмерный объект, бинарное изображение.

1. ВВЕДЕНИЕ

Одной из важных задач, решаемых в оптико-электронных системах наблюдения за воздушным пространством, является обнаружение, оценка параметров и сопровождение объектов. Необходимо отметить, что ранее для этих целей использовались преимущественно радиолокационные станции (РЛС). Однако РЛС присущ ряд существенных недостатков, к которым можно отнести присутствие “слепой зоны”, чувствительность к постановкам помех, высокую стоимость и значительные габариты. Кроме этого, во многих случаях применение активных способов наблюдения нежелательно, ввиду ограничений по энергопотреблению и необходимости обеспечения скрытности средств обзора воздушного пространства. Одним из способов решения указанных проблем является использование современных телевизионных и тепловизионных датчиков для получения изображения объекта с последующим анализом поступающей видеоинформации. Зачастую интересующие объекты на изображениях могут быть малоразмерными или точечными, так как они имеют небольшие габариты и находятся на значительном удалении от датчика изображения.

Отметим ряд особенностей, усложняющих решение задачи обнаружения и оценки параметров воздушных объектов по сравнению с классическими задачами технического зрения:

– присутствие геометрических преобразований изображений, возникающих вследствие изменения ориентации датчика в пространстве;

– высокая динамика фона;

– требование обеспечения высокой вычислительной эффективности алгоритма, связанное с необходимостью обработки в реальном масштабе времени последовательности изображений при ограниченных аппаратных ресурсах.

– возможность работы алгоритма в замкнутом контуре сопровождения.

Вместе с тем при разработке алгоритма обнаружения малоразмерных объектов необходимо принимать во внимание свойства облачного фона, которому свойственно наличие корреляционных связей между соседними элементами изображения. В данной работе предлагается нейросетевой алгоритм обнаружения объектов, основанный на пространственной обработке изображений. Рассматриваемый подход может использоваться в таких областях как контроль и мониторинг воздушного пространства, астронавигация, метеорные исследования, в военных и специальных приложениях.

2. ОПИСАНИЕ АЛГОРИТМА

Для успешного решения задачи обнаружения объектов, необходимо располагать как можно более полной информацией о функциях яркости объектов и фона, а также знать параметры распределения шума. Оптимальный в смысле выбранного критерия качества алгоритм можно построить только при наличии полной априорной информации о свойствах объекта и фона. В большинстве случаев требуемая информация бывает недоступна. Реальные условия всегда характеризуются той или иной степенью неопределенности в отношении информации о фоноцелевой обстановке, которая обусловлена как невозможностью точного предсказания свойств объекта и фона, так и непредвиденными изменениями этих свойств во времени. Проблемы, связанные с математическим синтезом оптимальных алгоритмов приводят к необходимости поиска эвристических подходов.

Для решения задач классификации широкое распространение получил метод опорных векторов [1], однако высокое качество решения задачи достигается за счет увеличения вычислительной сложности, что затрудняет его использование в системах обработки изображений реального времени. На практике в случае отсутствия формализованной информации о фоноцелевой обстановке могут применяться нейронные сети (НС), которые обобщают в процессе обучения основные отличия необходимые для выработки классификационных решений о принадлежности точек изображения объекту либо фону.

Первоначальный этап обработки в большинстве случаев заключается в формировании многомерного признакового пространства. Общеизвестно, что качество решения задачи классификации в большой степени зависит от выбранной для описания системы признаков. Не вызывает сомнения целесообразность использования телевизионных, тепловизионных и радиолокационных яркостных

изображений. Располагая этими естественными признаками, путем их линейной либо нелинейной обработки получают пространственные, градиентные, спектральные, текстурные и ряд других характеристик [2]. Однако вопрос о требуемой размерности признакового пространства остается открытым. Так, с целью выделения наиболее информативных свойств возможно выполнять редукцию многомерного пространства с помощью преобразования главных компонент [3].

Представим совокупность полученных M признаков в виде трехмерного массива чисел размерностью $N_x \times N_y \times M$, где N_x, N_y – ширина и высота наблюдаемого изображения. Для определенности положим M нечетным и введем в рассмотрение многомерное окно размерностью $L \times L \times M$ с центром в точке $(i, j, (M-1)/2)$, перемещающееся вдоль координат i и j. На каждом шаге из исходного массива в пределах окна выбираются данные, из которых формируется входной вектор из $N_{вх} = L^2 M$ элементов, поступающий на вход нейронной сети. Нейронная сеть может осуществлять отнесение по нечеткому правилу центрального элемента к классу “объект” или “фон”, а в соответствующий элемент выходного двумерного изображения записывать значение, соответствующее степени истинности. Таким образом, после завершения обхода получается полутоновое изображение. Можно утверждать, что предложенный подход к классификации имеет много общего с процедурой многомерной фильтрации [4].

Во многих случаях на практике потребителя интересует бинарное изображение, которое может быть получено путем глобальной пороговой обработки результата нейронечеткой классификации. Пороговое значение может быть задано в зависимости от заданной степени истинности.

После получения бинарного изображения можно уменьшить степень фрагментации сегментов (связных областей) путем морфологической обработки, заключающейся в последовательном применении операций морфологического закрытия и открытия с квадратными структурирующими элементами. Переход от итогового бинарного изображения к представлению результатов обработки в виде списка параметров связных областей изображения производится с помощью процедуры разметки и параметризации. Исходя из параметров найденных сегментов и имеющейся априорной информации, можно принять решение о наличии интересующих объектов в обрабатываемой области кадра.

Важным вопросом является выбор структуры НС, который осуществляется в соответствии с особенностями и сложностью задачи. Для решения некоторых отдельных типов задач, на сегодняшний день, уже существуют оптимальные архитектуры [5]. Если же задача не может быть сведена ни к одному из известных типов, разработчику приходится решать проблему синтеза новой архитектуры сети. При этом обычно руководствуются несколькими ключевыми принципами:

– вычислительные возможности сети возрастают с увеличением числа ячеек сети, плотности связей между ними и числа выделенных слоев;

– введение обратных связей наряду с увеличением возможностей сети к решению многих практических задач ставит вопрос о ее динамической устойчивости;

– сложность алгоритмов функционирования сети способствует улучшению способности к обобщению данных.

Однако в большинстве случаев приемлемый вариант получается на основе интуитивного подбора, и дать подробные рекомендации не представляется возможным. В данной работе в качестве базовой структуры сети был выбран персептрон с одним скрытым слоем и сигмоидальными активационными функциями (рисунок 1), что было обусловлено изученностью его свойств, простотой реализации и скоростью работы. Количество нейронов в скрытом слое $N_{скр}$ удовлетворяет неравенству $N_{скр} < N_{вх} - 1$.

3. КОМПЬЮТЕРНОЕ МОДЕЛИРОВАНИЕ

3.1 Обучение сети

Планируется, что разрабатываемый алгоритм будет использован в бортовых системах обработки информации, что накладывает существенные ограничения на вычислительную трудоемкость.

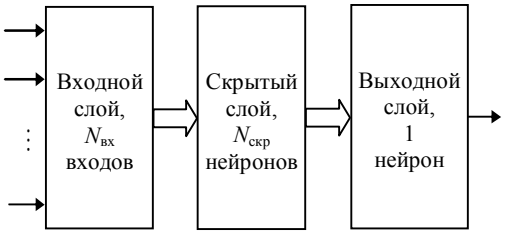


Рисунок 1. Структура трехслойного персептрона.

В текущем варианте реализации в качестве базового признака было выбрано яркостное изображение. Полученные результаты, характеризующие работоспособность разрабатываемого алгоритма, носят промежуточный характер, и в ближайшей перспективе предполагается выполнять комплексирование мультиспектральных изображений. Для обучения сети использовался метод сопряженных градиентов, который обеспечивает компромисс между качеством обучения и вычислительной эффективностью.

С целью проведения обучения НС был создан обширный банк видеопоследовательностей ИК диапазона. В качестве датчиков применялись охлаждаемые тепловизоры и болометрические приборы, на основе сканирующих линеек и матричного типа, чувствительные к излучению с длинами волн 3-5 мкм и 8-14 мкм. Выбранные сюжеты характеризовались наличием как однородного, так и облачного фонов, отношение сигнал/шум (отношение локального контраста объекта к СКО шума) достигало значения 2,5 и менее. Каждый сюжет содержал один объект, что объясняется необходимостью проектирования алгоритма для работы в замкнутой системе автоматического сопровождения объектов. Учитывая размеры объектов и статистические характеристики фона, значение L было принято равным 7. В качестве реальных объектов выступали мини-БПЛА, Ми-8, МиГ-31 и другие. Из тестовой базы было отобрано 50 характерных кадров, содержащих различные типы сцен. Примеры исходного и инвертированного бинарного изображений приведены на рисунке 2.

На каждом кадре формировались обучающие примеры, т.е. участки изображений размерами 7x7, представляемые в виде вектора, содержащего 49 элементов.

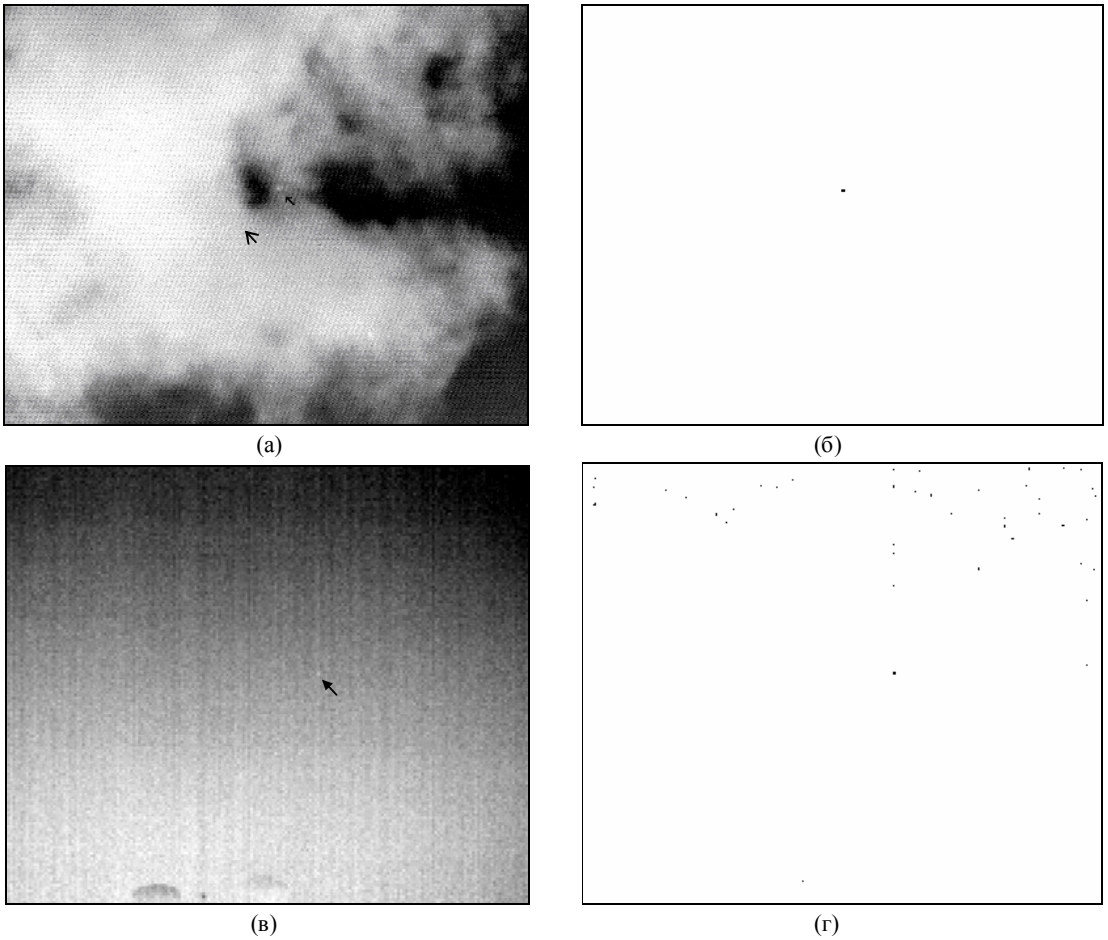


Рисунок 2. Примеры: исходных изображений (а, в) и инвертированных бинарных изображений (б, г). Стрелками обозначено положение объекта.

Выбор фоновых участков производился произвольным образом из разных областей исходного кадра. Участки изображений, содержащих объект, выбирались так, чтобы яркость центра объекта отличалась от точек окрестного фона. Для фона и для объекта формировалось одинаковое число обучающих примеров, по 250 для каждого. Обучающие примеры для разных классов подавались на вход сети поочередно. Максимальное количество циклов обучения (эпох) составило несколько тысяч, минимальная ошибка обучения задавалась на уровне 10^{-6}. Для улучшения процесса обучения входные данные нормализовывались на основе найденных оценок математического ожидания и дисперсии. По результатам проведенного компьютерного моделирования $N_{скр}$ было взято равным 16.

3.2 Экспериментальные исследования

Экспериментальная проверка алгоритма проводилась на натурных тестовых видеопоследовательностях ИК диапазона продолжительностью от 500 до 1000 кадров. Для оценки работоспособности предложенного алгоритма использовалась следующая методика. На *n*-м кадре рассчитывался бинарный параметр правильного обнаружения $N_{no}^{(n)}$ по следующему правилу

$$N_{no}^{(n)} = \begin{cases} 1, \text{ если } |\hat{i}_{ц}^{n} - i_{цэ}^{n}| \le 1, \left|\hat{j}_{ц}^{n} - j_{цэ}^{n}\right| \le 1, n = \overline{1,N}; \\ 0, \text{ иначе,} \end{cases}$$

где $(i_{цэ}^{n}, j_{цэ}^{n})$ – эталонные значения координат центров объектов на *n*-м кадре, в качестве которых использовались данные, полученные человеком-экспертом, $(\hat{i}_{ц}^{n}, \hat{j}_{ц}^{n})$ – измеренные значения координат центров объектов, N – число кадров в видеосюжете.

Зная $N_{no}^{(n)}$ на каждом кадре, можно вычислить частоту правильного обнаружения P_{no} как

$$P_{no} = \frac{1}{N}\sum_{n=1}^{N} N_{no}^{(n)}.$$

Дополнительно оценивалась частота ложных тревог $P_{лт}$ как усредненное отношение количества точек, неверно отнесённых к объекту, к общему количеству точек изображения, не принадлежащих объекту. При варьировании порогового коэффициента можно построить кривую зависимости $P_{no} = f(P_{лт})$, называемую рабочей характеристикой обнаружения. Пример зависимостей $P_{no} = f(P_{лт})$ полученных для двух видеосюжетов

представлен на рисунке 3 (характерные кадры из тестовых видеопоследовательностей представлены на рисунке 2). При проведении сравнительных исследований использовался алгоритм обнаружения воздушных объектов, описанный в [6]. Суть подхода состоит в применении параллельной двумерной фильтрации наблюдаемого изображения двумя фильтрами с масками разного вида и размерности, подстройки размерности маски фильтра для лучшего обнаружения интересующего объекта, сравнения обработанного изображения с глобальным порогом, морфологической фильтрации бинарного изображения. Данный алгоритм выбран для сравнения, так как он успешно используется в системах обработки изображений реального времени семейства "Охотник", выпускаемых ФГУП ГРПЗ [7], а его эффективность подтверждена опытом практического применения.

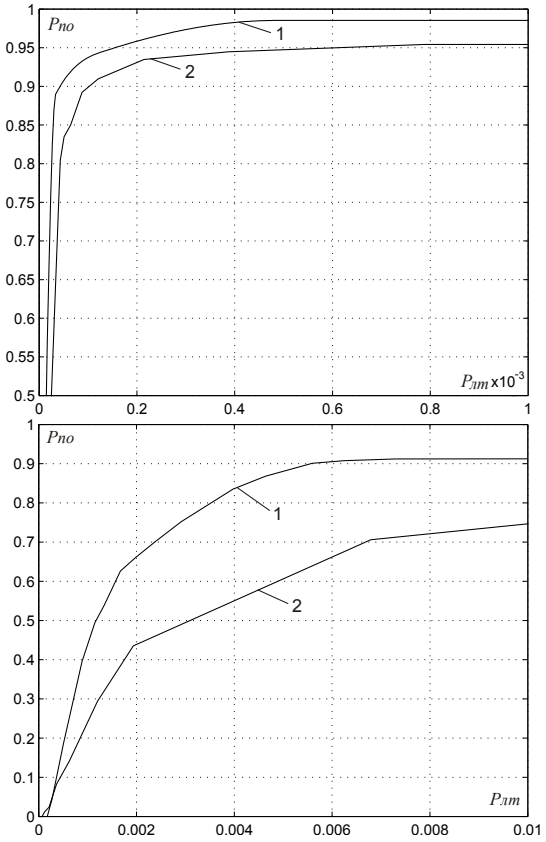


Рисунок 3. Графики зависимости $P_{пв} = f(P_{лт})$. *Кривая 1* получена для нейросетевого алгоритма, *кривая 2* – для предложенного ранее подхода.

Анализ полученных результатов показывает, что прирост частоты правильного обнаружения при фиксированной частоте ложных тревог в среднем составляет около 10%. Исследования показывают, что значение СКО ошибки измерения координат центров объектов на изображении не превышает 1,5 пикселей.

4. ЗАКЛЮЧЕНИЕ

В работе предложен нейросетевой алгоритм обнаружения малоразмерных воздушных объектов, наблюдаемых на фоне облачного неба. Алгоритм ориентирован на реализацию в системах обработки изображений реального времени. Приведены предварительные результаты сравнительных экспериментальных исследований, свидетельствующие о возможности применения рассматриваемого подхода. Улучшения работы алгоритма планируется достичь путем увеличения размерности признакового пространства и использования других архитектур нейронной сети.

5. СПИСОК ЛИТЕРАТУРЫ

[1] Хайкин С. Нейронные сети: полный курс. Пер. с англ., 2006.– 1104с.

[2] Heesung Kwon, Sandor Z. Der, Nasser M. Nasrabadi. Adaptive multisensor target detection using feature-based fusion // *Optical Engineering*, vol. 41, no. 1, 2002, pp. 69-80.

[3] Image Fusion: algorithms and applications, edited by Tania Stathaki, London, Academic Press, 2008, 500 p.

[4] Mukul V. Shirvaikar, Mohan M. Trivedi. A neural network filter to detect small targets in high clutter background // *IEEE Transactions on Neural Networks*, vol.6, no.1, 1995, pp. 252-257.

[5] Markos Markou, Sameer Singh. Novelty detection: a review – part 2: neural network based approaches // *Signal Processing*, vol. 83, 2003, pp. 2499 – 2521.

[6] Алпатов Б.А., Блохин А.Н., Муравьев В.С. Алгоритм обработки изображений для систем автоматического сопровождения воздушных объектов // *Цифровая обработка сигналов*. – 2010. – №4. – С.11-17.

[7] Алпатов Б.А., Блохин А.Н., Костяшкин Л.Н., Романов Ю.Н., Шапка С.В. Семейство многофункциональных систем обработки видеоизображений "Охотник" // *Цифровая обработка сигналов*. – 2010. – №4. – С.44-51.

NEURAL NETWORK ALGORITHM FOR SMALL OBJECT DETECTION ON CLOUDY BACKGROUND

Abstract

Continuous improvement of video sensors and computer equipment leads to an expansion of practical applications of the image analysis. The real-time image processing systems have been actively used for automatic object detection and tracking. They are in high demand in traffic management, control of product manufacturing, remote sensing, robotics, medicine and other fields.

Much attention in last two decades is paid to the problem of small target detection on cloudy background. In this work the neural network approach with the orientation on image analysis of the real scale video sequences has been suggested. The network learning procedure has been described and the preliminary results of comparative experimental research have been also presented.

Авторы

Шубин Н.Ю., аспирант кафедры Автоматики и информационных технологий в управлении (АИТУ) Рязанского государственного радиотехнического университета (РГРТУ)

Муравьев В.С., к.т.н., научный сотрудник кафедры АИТУ РГРТУ

Муравьев С.И., к.т.н., доцент кафедры АИТУ РГРТУ.

Двухуровневый структурный подход к детектированию объектов

Николай Сергиевский, Александр Харламов
Кафедра интеллектуальных информационных систем и технологий
Московского Физико Технического Института
dereyly@gmail.com, kharlamov@analyst.ru

Abstract

The article describes a method for creating an effective set of bounding boxes. To solve this problem a two level approach applies. At the first level a stable structure which is a subset of the object is searching. It should be noted that the probability of detection of such structure on the background (or random location of another object) is small. At the second level the keypoints are searching. After that it is created a model of interaction between the elements of the first and second level, which is the result of a set of hypotheses about the location of the object, i.e. bounding boxes.

Keywords: object detection, object recognition, SIFT, SURF

Аннотация

В статье описывается метод построения эффективного набора детектирующих окон. Для решения этой задачи применяется двухуровневый подход. На первом уровне осуществляется поиск стабильной структуры, такого подмножества объекта, для которого вероятность обнаружения на фоне (случайном месте на сцене) мала. На втором уровне осуществляется поиск характерных точек. После этого строится модель взаимодействия элементов первого и второго уровня, результатом которой оказывается набор гипотез о расположении объекта, т.е. детектирующих окон.

1. Введение.

В направлении распознавания образов в настоящее время наблюдается большой интерес к детектированию объектов на сцене, как с практической стороны, так и со стороны фундаментальных исследований, направленных на понимание и моделирование процессов зрительного восприятия. В данный момент разработано много баз для оценки алгоритмов детектирования, например, конкурс Pascal [1]. Анализируя данные, полученные в рамках этого конкурса (таблица 1), можно заметить, что объекты с большой внутриклассовой вариативностью дают небольшой процент правильного детектирования.
Отчасти это связано с ошибками классификации (таблица 2), но так же большую роль играет выбор метода определения границ объекта и параметров оценки, например размера сканирующего окна.

2. Поиск особенностей на изображении.

В существующих подходах к автоматическому определению границ объектов [2,3] используют методы детектирования и распознавания, основанные на одноуровневых алгоритмах извлечения признаков, таких как SIFT, SURF, MSER. Несколько работ было посвящено определению распределения взаимного расположения характерных точек [4]. На первом этапе работы алгоритма из работы [4] на изображении находят все характерные точки, затем выбирают характерные точки, для которых геометрический центр фигуры, их включающей, лежит внутри объекта, после чего характерные точки классифицируются по дескрипторам. Затем выявляется статистика взаимного распределения расстояний между точками принадлежащим разным классам. Модель строится на основании вычисления степени правдоподобия полученных распределений. При этом возникает проблема комбинаторного взрыва, поскольку максимальное количество характерных точек, которое реально можно соотнести не превышает шести. Подобные проблемы возникают и при использовании подхода, основанного на взаимодействии суперпикселей [5]. Возможно снижение комбинаторной нагрузки и повышение эффективности алгоритмов за счет использования структурного подхода.
В этом случае решаются две задачи разных уровней:
1. поиск стабильных структур;
2. поиск характерных точек.
К детектору первого уровня (поиск стабильной структуры) предъявляют требования, чтобы данная структура почти всегда принадлежала объекту, а вероятность обнаружения объекта на фоне (случайном месте на сцене) была мала. Это не значит, что данная структура не может принадлежать другим классам объектов как подмножество их собственной структуры. Такое детектирование не должно происходить в случайных точках объекта или сцены.
На втором уровне ищутся характерные точки с помощью одного из стандартных методов (Harris, DOG).
Далее должно быть принято компромиссное решение между детектированием объекта целиком и детектированием объекта по характерным точкам. Повысить качество данного компромиссного решения может анализ расстояний между классифицированными характерными точками и стабильной структурой.

3. Построение детектирующего окна.

Стабильные структуры могут нести контекстную информацию о расположении объекта. Эта информация извлекается из взаимного распределения положения структур и границ объектов (Рис 1).

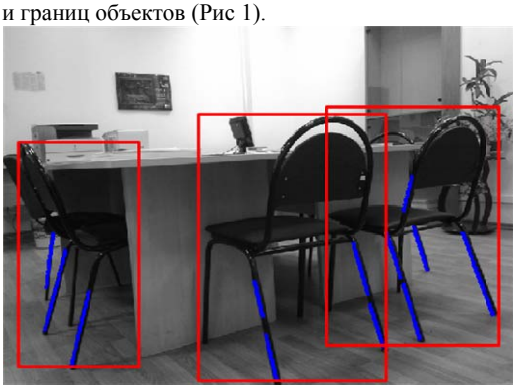


Рис 1. Пример взаимного распределения стабильных структур (ножек стульев) и рамок объектов.

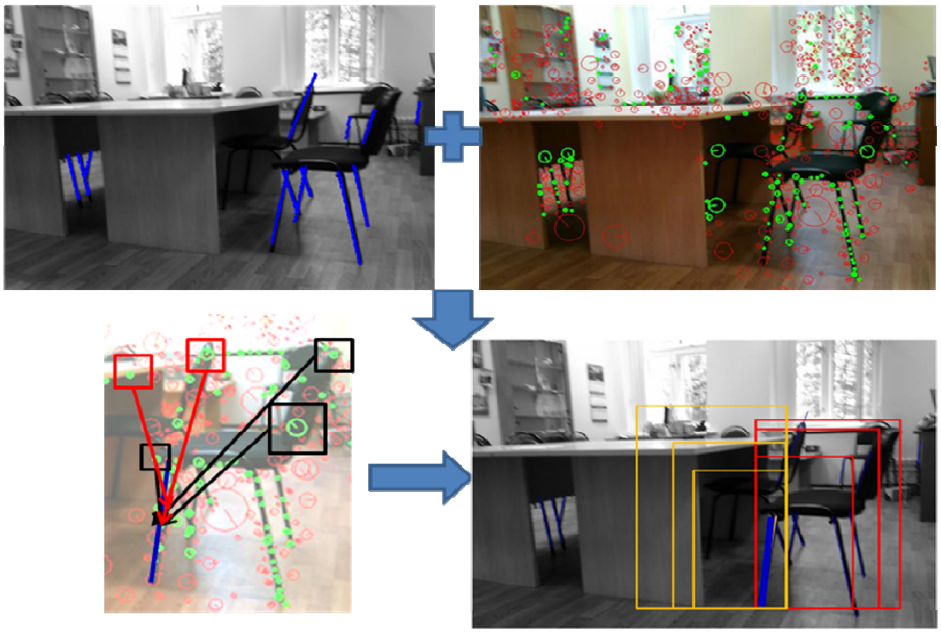


Рис 2. Диаграмма построения гипотез детектирующего окна. Верхний левый рисунок: результат поиска стабильных структур (ножек). Верхний правый: характерные точки SURF, где зелеными кругами обозначаются характерные точки, классифицированные как часть объекта. Нижний левый: взаимодействие стабильной структуры и характерных точек, черным отмечено взаимодействие внутри объекта, красным – ложное срабатывание. Нижний правый: порождаемые гипотезы.

После обучения происходит поиск стабильных структур на каждом изображении из обучающей и контрольной выборки. Далее вычисляются распределения положения объекта и детектирующего окна объекта относительно стабильной структуры, при этом учитываются размеры и положение самой стабильной структуры. Таким образом каждая структура порождает гипотезы о местоположении объекта (Рис. 2). Гипотезы уточняются за счет характерных точек и их взаимодействия со стабильными структурами. Стабильные структуры могут образовывать связи друг с другом и формировать более сильные гипотезы, для этого вычисляются распределения местоположения объекта относительно нескольких опорных точек (опорными точками могут быть как характерные точки, так и стабильные структуры).
С использованием базы детектируемых объектов для каждого изображения ищутся характерные точки, после чего на основании полученного дескриптора производится обучение классификатора точки на принадлежность объекту (Рис 2). Точки, принадлежащие объекту, образуют словарь для алгоритма «мешок-слов», и далее на основании этого словаря идет повторная внутриклассовая классификация (без учителя).

4. Детектирование стульев.

Небольшой эксперимент, проведенный в рамках описанного подхода, показал точность 17.3% на проверочной выборке соревнования паскаль. Для получения результата было взято тренировочная выборка паскаль и дополнительно 600 объектов из LabelMe. После фазы генерации детектирующих окон использовалась классификация квазилинейным SVM по мешку слов, построенному на плотном SIFT (PHOW), размер мешка слов составил 300 элементов.

5. Выводы.

В работе представлен подход к детектированию сложных объектов на основе использования двух уровней поиска разнородных элементов на изображении с их взаимодействием, и построения эффективного множества границ объекта для его последующей классификации.
Предлагаемый подход направлен на уменьшение вариативности результатов для подходов, основанных на взаимном распределении данных, за счет введения однонаправленных связей типа «звездочка» (элемент верхнего уровня – соотнесенные с ним элементы нижнего уровня) между разными уровнями. Исследование может быть расширено на решение задачи семантической классификации объектов путем замены характерных точек на сегменты объекта.

6. Список литературы.

[1] http://pascallin.ecs.soton.ac.uk/challenges/VOC/
[2] *P. Felzenszwalb, D. McAllester, D. Ramanan.* A Discriminatively Trained, Multiscale, Deformable Part Model, CVPR 2008
[3] S.Vijayanarasimhan, K. Grauman. Efficient Region Search for Object Detection, CVPR 2011.
[4] *R. Fergus, P. Perona, R. Fergus.* Object Class Recognition by Unsupervised Scale-Invariant Learning CVPR 2003
[5] *B. Fulkerson, A. Vedaldi, S Soatto.* Class Segmentation and Object Localization with Superpixel Neighborhoods, CVPR 2009

Таблица 1. Часть таблицы результатов Pascal Voc2010 по детектированию объектов.

aero plane	**bicycle**	**bird**	**boat**	**bottle**	**car**	**Cat**	**chair**	**dining table**	**tv/ monitor**
50,5	24,4	17,1	13,3	10,9	32,9	36,5	5,6	6,6	27,2
54,2	48,5	15,7	19,2	29,2	43,5	41,7	16,9	26,7	40,8
53,3	55,3	19,2	21	30	46,7	41,2	20	20,7	40,3
49,1	52,4	17,8	12	30,6	32,8	37,3	17,7	27,7	38,4
46,1	52,6	13,8	15,5	28,3	44,5	26,6	17,6	16,1	-
40,4	34,7	2,7	8,4	26	33,8	17,2	11,2	14,4	35,7
37,9	33,7	2,7	6,5	25,3	33,1	15,5	10,9	12,5	34,1
52,4	54,3	13	15,6	35,1	49,1	31,8	15,5	13,5	38
56,7	39,8	16,8	12,2	13,8	36,9	47,7	12,1	26,5	41,7
58,4	39,6	18	13,3	11,1	37,8	43,9	10,3	20,8	41,9

Таблица 2. Часть таблицы результатов Pascal Voc2010 по классификации объектов.

bird	53,1	39,5	42,9	47,3	64,5	63	66,7	66,8	49,9	54,6	46,1	61,7	69,9	69,9
chair	71,1	46,5	58,7	49,1	72	66,9	72,2	71,2	59,9	60	59,4	67,5	79,3	79,4

Алгоритм вычитания фона, основанный на поблочных классификаторах

Евгений Шальнов, Владимир Кононов, Вадим Конушин
Лаборатория Компьютерной Графики и Мультимедиа,
Московский Государственный Университет имени М.В.Ломоносова, Москва, Россия
{eshalnov, vkononov, vadim}@graphics.cs.msu.ru

Аннотация

Алгоритмы вычитания фона широко применяются в задачах видеонаблюдения и активно изучаются в последнее время. Большинство существующих алгоритмов вычитания фона основываются на сравнении цвета пикселей или блоков очередных кадров со своими цветовыми моделями. При этом значительные перепады в освещении обычно приводят к большому числу ошибок, а иногда и к полной деградации работы системы.

В методе, предлагаемом в данной статье, весь кадр видео разбивается на блоки, для каждого из которых на основе бустинга строится отдельный классификатор, определяющий перекрыт данный блок или нет. Благодаря использованию признаков, основанных на сравнении цветов разных пикселей в блоке, достигается большая устойчивость к изменениям освещенности и дрожанию камеры. Требуя значительного времени при обучении под новую сцену, на этапе выполнения алгоритм демонстрирует высокую скорость, сравнимую со скоростью простейших алгоритмов вычитания фона.

Было проведено сравнение точности сегментации видео с помощью данного алгоритма и нескольких существующих аналогов на выложенных в открытый доступ видеороликах. Сравнение показало высокую точность работы предложенного алгоритма.

Ключевые слова: *вычитание фона, машинное обучение, бустинг.*

1. ВВЕДЕНИЕ

Эта статья посвящена проблеме выделения в видео, снятом статичной камерой, объектов переднего плана в реальном времени. Одним из наиболее распространенных применений такой задачи являются алгоритмы слежения за людьми, которые используют вычитание фона для поиска объектов. Такие алгоритмы используются в охранном видеонаблюдении, освещении спортивных трансляций. Также алгоритмы вычитания фона используются для создания дополнительных эффектов во время проведения видеоконференций, например для замены фона.

Подобные приложения накладывают следующие ограничения на алгоритмы вычитания фона: (1) устойчивость к небольшим изменениям освещения и дрожаниям камеры; (2) вычислительная простота для возможности работы в реальном времени.

Стандартные методы вычитания фона основаны на построении модели фона по правилам, не зависящим от

Рисунок – 1. Задача вычитания фона. Слева направо: фон, кадр с объектом переднего плана, искомая маска объекта.

рассматриваемой области изображения. Из-за этого для описания сложных случаев изменения освещения или дрожания камеры необходимо либо использовать сложную модель фона, требующую больших вычислительных затрат, либо обновлять модель фона динамически во время работы алгоритма, что приводит к существенным ошибкам в работе на тех кадрах, на которых происходит эволюция модели.

В этой статье предлагается алгоритм вычитания фона с использованием поблочных классификаторов для сегментирования блоков на передний и задний план. С помощью машинного обучения происходит выявление наиболее информативных признаков для каждого конкретного блока. Это позволяет с одной стороны быстро обрабатывать кадры видеопоследовательности, так как используются простые признаки, а с другой стороны достигать высокой точности сегментации пикселей, так как набор простых признаков уникален для каждого блока.

Статья построена следующим образом. В секции 2 приводится обзор текущих методов решения задачи, в секции 3 описывается предложенный алгоритм. В секции 4 приводятся результаты экспериментов и сравнения, а в секции 5 подводится заключение.

2. СУЩЕСТВУЮЩИЕ МЕТОДЫ

Большинство методов вычитания фона можно разбить на 3 категории:

- Попиксельные [1,2,11]
- Поблочные [11]
- Алгоритмы, основанные на Марковских или Условных случайных полях [4,5]

Методы первой категории обрабатывают все пиксели очередного кадра независимо. Цвет каждого пикселя на текущем кадре сравнивается с его цветовой моделью. Типичными цветовыми моделями являются нормальное распределение [11], смесь нормальных распределений [1] и непараметрические модели [2]. Метод [1] предполагает, что значение цвета пикселя фона – случайная величина, чье распределение можно аппроксимировать смесью нормальных

распределений. Среди недостатков данного метода можно выделить необходимость использования большого числа гауссиан в смеси для моделирования фона, что приводит к существенным вычислительным затратам. Кроме того, данный метод чувствителен к небольшим дрожаниям камеры.

Методы второй категории независимо обрабатывают не пиксели, а целые блоки. То есть весь кадр разбивается на блоки, и для каждого из них принимается независимое решение. Несмотря на то, что потенциально такие методы не могут быть идеально точными, зачастую они дают более приемлемый результат, т.к. используют для принятия решения информацию с целой области. В эту категорию входят, например, метод на основе построения LBP-гистограмм [10] и метод на основе метрики SSD. Последний метод хранит модель фона в виде одного изображения. Результат на каждом кадре производится с помощью сравнения значения метрики близости с порогом. Метрика считается поблочно и равна сумме квадратов разностей между цветами соответствующих пикселей фона и текущего кадра. Предложенный метод также попадает в эту категорию.

В последнее время были предложены методы вычитания фона, проводящие сегментацию пикселей изображения, используя модели Марковского и Условного случайного поля для учета пространственных зависимостей между пикселями [4-5]. С помощью данных методов можно дополнительно требовать, чтобы границы областей объектов переднего плана проходили преимущественно по краям на входном изображении. Но, к сожалению, методы данной категории являются более медленными, и почти не используются на практике в системах видеонаблюдения.

В статье [5] авторы предлагают алгоритм вычитания фона для использования в видеоконференциях. Поэтому, они используют несколько иную постановку задачи: в качестве переднего плана выступает лишь человек, ближайший к видеокамере. Люди на заднем плане, считаются частью фона. Для решения данной задачи предлагается использовать информацию о наиболее вероятном движении объектов переднего плана. С помощью данного метода можно уменьшить ошибки сегментации из-за движений в фоновом изображении, но с другой стороны необходимо заранее знать наиболее вероятные модели движения для объектов переднего плана, что возможно не в каждом случае.

Все алгоритмы вычитания фона условно можно разделить на быстрые, но слабоустойчивые к изменениям освещения и дрожаниям камеры, и качественные, но требующие существенных временных затрат на обработку каждого кадра. Мы предлагаем метод, который может одновременно быстро и качественно решать задачу, основываясь на классификаторах, натренированных для каждого блока.

3. ПРЕДЛОЖЕННЫЙ МЕТОД

Мы предлагаем алгоритм вычитания фона, работающий с блоками кадра, который устойчив к изменениям освещения и способен работать в приложениях реального времени. Этот эффект достигается за счет использования простых признаков для принятия решения о принадлежности блока к переднему или заднему планам, причем для каждого блока выбираются наиболее информативные для него признаки. Использование простых признаков не требует больших вычислительных затрат, что позволяет работать в реальном времени. Наиболее информативные признаки для каждого блока кадра удается выявлять с помощью алгоритма машинного обучения.

Для тренировки алгоритма машинного обучения необходима выборка положительных и отрицательных примеров. В общем случае для ее создания необходимо много данных, размеченных вручную. Эта процедура требует больших временных затрат и не гарантирует получение репрезентативной выборки. Поэтому мы генерируем выборку без объектов переднего плана, имитируя возможные изменения в освещении и настройках видеокамеры, а также выборку, имитирующую возможные перекрытия фона. Для такой генерации достаточно всего одного кадра сцены без объектов переднего плана.

Таким образом, в предложенном алгоритме можно выделить следующие шаги:

1. Генерация положительных и отрицательных примеров для обучения классификаторов.
2. Построение классификаторов.
3. Использование обученных классификаторов для получения маски объектов переднего плана в видеопоследовательности.

3.1 Построение обучающей выборки

На первом шаге алгоритма происходит построение обучающей выборки для тренировки классификаторов. Данная выборка состоит из положительных и отрицательных примеров для фонового изображения (Рис. 2).

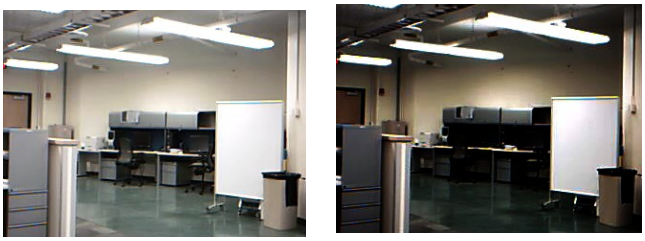


Рисунок – 2. Кадры с положительными примерами из обучающей выборки.

Так как предлагаемый алгоритм должен быть устойчив к изменениям в освещении, то в положительные примеры добавляются кадры с различными вариантами освещения. Кроме того используются примеры, полученные при изменении контрастности и небольшим размытием исходных кадров из обучающего видео.

Для описания метода построения отрицательных примеров необходимо сначала рассмотреть признаки, используемые при обучении. В качестве признаков классификатора используется разность соответствующих цветовых каналов пар случайных пикселей блока. Так как при этом используется не значение цвета, а его значение относительно цвета другого пикселя, то данный классификатор более устойчив к изменениям освещения сцены.

При построении отрицательных примеров мы моделируем ситуацию, когда один или оба пикселя из пары, используемой при построении значения признака, перекрыты объектом. Таким образом в используется 3 способа построения значения признака отрицательного примера: (1) цвет первого пикселя пары берется из изображений с положительными примерами, цвет второго — равномерно распределенная случайная

величина; (2) цвет первого пикселя — случайная величина, а цвет второго — из изображения; (3) цвета обоих пикселей задаются случайно.

3.2 Построение классификаторов

На следующем этапе работы алгоритма происходит обучение классификаторов. В предлагаемом алгоритме используется бустинг в качестве алгоритма машинного обучения [8-9]. С его помощью удается выявлять наиболее информативные признаки для каждого конкретного блока.

3.3 Использование обученных классификаторов

На третьем этапе работы алгоритма применяются обученные классификаторы для сегментации тестового видео.

Так как используется бустинг в качестве алгоритма машинного обучения, то в результате каждый обученный классификатор представляет собой взвешенную сумму простых правил. Правилами являются сравнение конкретного признака с порогом. Это означает, что сложность классификации отдельного блока линейно зависит от сложности применения отдельного простого правила. За счет этого удается достичь быстрой обработки блоков кадра.

Для уменьшения ошибок классификации на данном этапе применяется медиана по результатам сегментации на нескольких последовательных кадрах.

4. ЭКСПЕРИМЕНТЫ И СРАВНЕНИЕ

Предложенный метод был протестирован на наборе видеопоследовательностей [6], снятых внутри помещения, в каждой из которых происходили существенные изменения освещения. Предложенный алгоритм вычитания фона показал хорошие результаты работы на данной базе (рис. 3).

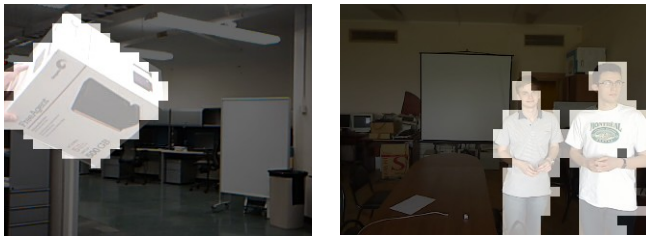


Рисунок – 3. Примеры работы предложенного метода.

В текущей версии нашего алгоритма при существенных изменениях освещения происходят ошибки при сегментации на объекты переднего и заднего плана (рис. 4). Данное явление связано с выбранным методом построения обучающей выборки. В настоящее время используется простое приближение алгоритма настройки камеры при изменении освещения. Одним из основных направлений последующей работы будет устранение этого недостатка.

Рисунок – 4. Примеры неудачной работы метода.

Было проведено экспериментальное сравнение нашего метода с алгоритмами, основанными на использовании смеси гауссиан и сравнении блоков по метрике SSD.

Использовалась встроенная в MATLAB реализация смеси гауссиан из Computer Vision System Toolbox. Полученные результаты продемонстрированы на рис. 5. Для сравнения использовались попиксельные метрики точности (precision) и полноты (recall). Графики на рис. 5 создавались изменением порога на значение метрики или выхода классификатора.

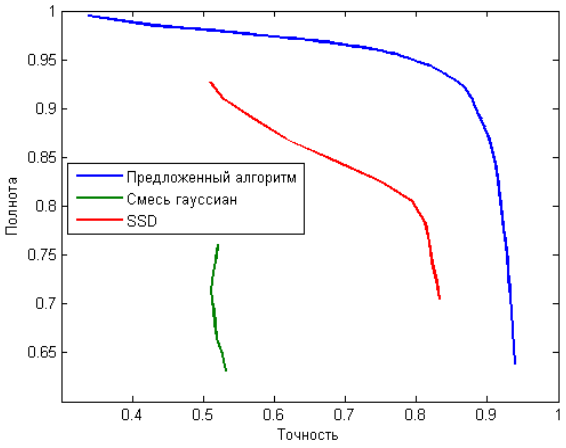


Рисунок – 5. Сравнение алгоритмов.

В первую очередь стоит отметить нетипичный график алгоритма, использующего смесь гауссиан. Так как изменяемый при построении соответствующего графика параметр (минимальная априорная вероятность фона) подается на вход встроенной в MATLAB функции, мы не можем точно объяснить данный факт. Видимо, этот параметр влияет не только на выход алгоритма (как порог на вероятностный выход), но и изменяет внутреннее состояние алгоритма (например, влияет на обновление цветовой модели в пикселях).

Также стоит разъяснить низкую итоговую точность данного алгоритма. В случае стабильной работы, когда не происходит сильного изменения освещения или настроек камеры, точность алгоритма, основанного на смеси гауссиан, превосходит точность двух других сравниваемых методов (см. рис. 6). Однако, как только значительно изменяется освещение, число ошибок резко возрастает (см. рис. 7). Во многих системах встраивают отдельный шаг для обработки таких ситуаций. Например, по резкому изменению гистограммы яркости пикселей кадра определяют момент изменения освещения, после чего вносятся изменения в модель фона (или наоборот, изменяют цвета пикселей очередного кадра видео). Мы использовали данный алгоритм из MATLAB без изменений, не встраивая подобную обработку. При её встраивании, мы допускаем, что алгоритм, основанный на смеси гауссиан смог бы показать на тестовых видеороликах наибольшую точность. Однако обычно такие обработки являются эвристиками, использующими ряд вручную подбираемых параметров, из-за чего они являются неустойчивыми. Предложенный алгоритм справляется с данной проблемой без каких-либо дополнительных модификаций.

Алгоритм, основанный на сравнении блоков по метрике SSD, справляется с изменением освещения лучше. Объяснить это

можно тем, что используя информацию сразу по некоторой окрестности, при классификации блоков он может использовать менее жесткие пороги, чем попиксельные методы. Тем не менее, метрика SSD не является оптимальной метрикой для сравнения блоков, из-за чего итоговая точность сегментации данным алгоритмом оказалась ниже, чем у предложенного подхода.

Помимо устойчивости к изменениям освещения среди достоинств предложенного метода можно выделить быструю обработку кадров за счет использования простых признаков. За секунду алгоритм обрабатывает 6 кадров размера 320x240 с использованием блоков размером 10x10 при неоптимизированной реализации на MATLAB на компьютере Intel Core 2Duo 2.53 GHz, 2048 Mb RAM. При этом алгоритм, основанный на использовании смеси гауссиан, обрабатывает 4-5 кадров в секунду, а алгоритм, использующий сравнение блоков по метрике SSD – 28 кадров в секунду.

Реализация на C++ может существенно ускорить предложенный алгоритм. Также он допускает простое и эффективное распараллеливание.

Стоит отметить, что предлагаемый алгоритм вычитания фона требует длительного процесса обучения. В среднем на обучение одного классификатора требуются около 2 минут. Таким образом, на обучение классификаторов для всех блоков потребовалось 17 часов.

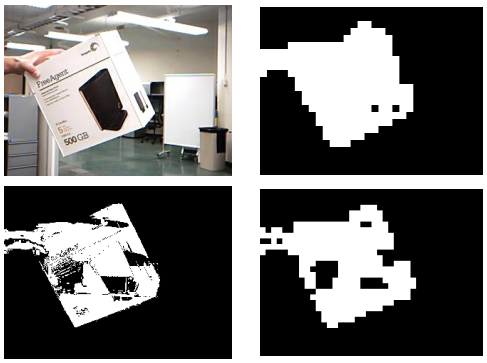


Рисунок – 6. Результаты работы алгоритмов при несильном изменении освещения. Верхний ряд: кадр видео с объектом переднего плана, результат работы предполагаемого метода. Нижний ряд: Результат работы GMM, результат работы SSD.

5. ЗАКЛЮЧЕНИЕ

В данной статье предложен алгоритм вычитания фона в видеопоследовательности на основе построения поблочных классификаторов с использованием алгоритма машинного обучения. Алгоритм показал устойчивость к изменениям освещения. При этом низкая вычислительная сложность алгоритма позволяет после обучения классификаторов обрабатывать видео в реальном времени.

Алгоритм, предложенный в данной статье, находится на стадии разработки. В дальнейшем мы планируем улучшить моделирование положительных и отрицательных примеров для создания выборок. Это позволит правильно решать задачу в случаях сильного изменения освещения и параметров камеры, а также избавит от существующих недостатков.

6. БЛАГОДАРНОСТИ

Работа была выполнена при поддержке гранта РФФИ №11-01-00957-а.

7. ССЫЛКИ

[1] Chris Stauffer, W.E.L. Grimson, Adaptive background mixture models for real-time tracking, CVPR 1999

[2] Ahmed Elgammal, David Harwood, Larry Davis, Non-parametric model for background subtraction, ICCV, 2000

[3] Antoine Monnet, Anurag Mittal, Nikos Paragios, Visvanathan Ramesh, Background modeling and subtraction of dynamic Scenes, ICCV, 2003

[4] Jian Sun, Weiwei Zhang, Xiaoou Tang, Heung-Yeung Shum, Background cut, CVPR, 2006

[5] A. Criminisi, G. Cross, A. Blake, V. Kolmogorov, Bilayer segmentation of live video, CVPR, 2006

[6] http://www.cs.utexas.edu/~changhai/icra10-datasets/datasets.html

[7] V. Kolmogorov, A. Criminisi, A. Blake, G. Cross, C. Rother, Bi-layer segmentation of binocular stereo video, CVPR, 2005

[8] J. Friedman, T. Hastie, and R. Tibshirani, Additive logistic regression: a statistical view of boosting. Annals of statistics, 38:337–374, 2000.

[9]http://graphics.cs.msu.ru/ru/science/research/machinelearning/adaboosttoolbox

[10] Marko Heikkila and Matti Pietika, A texture-based method for modeling background and detecting moving objects, IEEE, 2006

[11] C. Wren, A. Azarbayejani, T. Darrel, and A. Pentland, Pfinder: Real time tracking of the human body. In IEEE Trans. on Pattern Analysis and Machine Intelligence, 19 (7), pp. 780–785, 1997

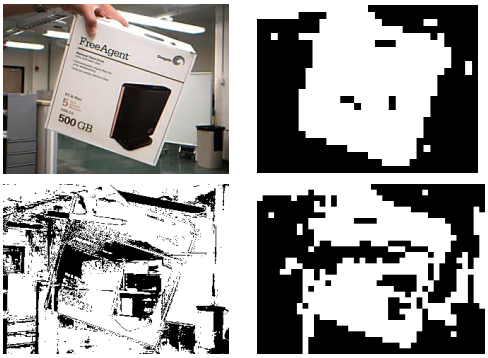


Рисунок – 7. Результаты работы алгоритмов при значительном изменении освещения.

Компьютерное слежение с масштабированием, основанное на градиентном спуске

Альберт Шапошников, Елена Шапошникова
ООО «Tomsklabs», Томский государственный университет, Томск, Россия
albertelena@mail.ru

АННОТАЦИЯ

В статье построена функция схожести двух изображений одного объекта на разных кадрах видеопоследовательности. Найдены частные производные этой функции по координатам центра и размерам области слежения. На основании этих производных построен алгоритм слежения, функционирующий при независимых изменениях длины и ширины области слежения. Алгоритм основан на градиентном спуске.

Ключевые слова: *слежение, цветовые гистограммы, сдвиг среднего, функция схожести, масштабирование.*

ANNOTATION

A similarity function between the images of the object in the different video frames is constructed. The partial derivatives of this similarity function with respect to the abscissa and ordinate of the center of the object and the width and height of the object are taken. Using these derivatives the tracking algorithm is constructed. This algorithm takes into account the independent changes of the width and height of the object. This algorithm is based on the gradient descent and not uses the Mean Shift.

Keywords*: tracking, histograms, mean shift, gradient descent, scaling.*

1. ВВЕДЕНИЕ

Компьютерная обработка и анализ видеоизображений – многогранная и быстроразвивающаяся область. Использование видеоинформации в современном мире стремительно нарастает. Это и различные системы мониторинга, технического зрения, видеотелефонии, регистрирующие и передающие огромные объемы видеоданных, и различные автономные системы (роботы), принимающие решения на основе анализа видеоизображений, и программы изучения космоса с беспилотными летательными аппаратами, и медицина, и интернет, и многие другие направления.

Примеры успешного применения обработки и анализа видеоизображений можно найти в физике, астрономии, биологии, медицинской радиологии, промышленности, в оборонной и правоохранительной сферах.

При этом наряду со значительным повышением уровня развития техники, весьма существенную роль играют методы обработки видеоинформации.

В задачах анализа видеоизображений одним из ключевых элементов является поиск на изображении некоторого объекта. Такая задача распадается на две части. Во-первых, нужно дать описание образца объекта, пригодное для эффективного сравнения этого образца с другим изображением объекта. Во-вторых, разработать алгоритм поиска области изображения (окна), для которой и производить это сравнение. Вопрос описания образца объекта (получения характеристического вектора) изучен широко [8,9]. Описание образца объекта может быть выбрано инвариантным к повороту объекта, изменению его цвета или размера. Для выбора области изображения (окна) подходы тоже имеются [8,9].

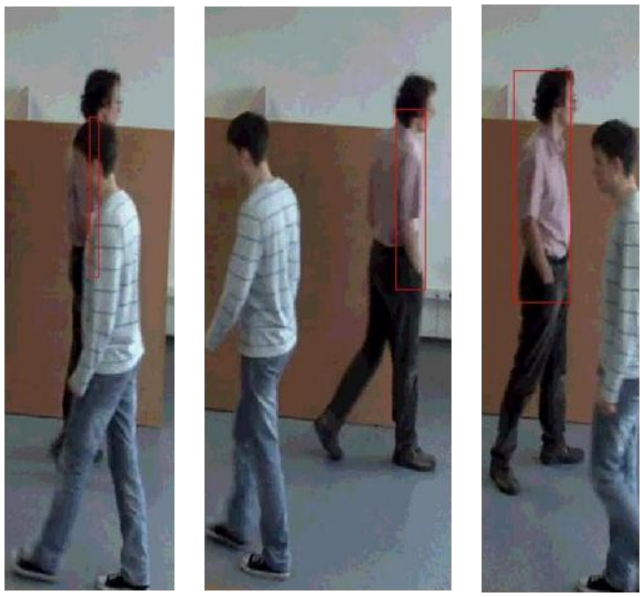
Для задач слежения поиск и подходящего описания объекта, и окна можно сузить, поскольку анализу подвергается последовательность контекстно связанных между собой изображений (кадров). Так, например, в алгоритме сдвига среднего Mean Shift [1,3–6] поиск наиболее подходящей области изображения начинается с области, зафиксированной на предыдущем кадре. В процедуре Mean Shift, введенной и подробно описанной в [4,5], объекты описываются с помощью нормированных цветовых гистограмм, то есть характеристическим вектором в данном случае является нормированная цветовая гистограмма. Для гистограмм на основании коэффициента Бхаттачария строится вещественнозначная функция схожести. На первом кадре последовательности (видеоряда) каким-либо образом отмечается объект, а затем для каждого следующего кадра видеоряда определяется ближайший локальный максимум функции схожести. Процедура Mean Shift оказалась эффективной и получила дальнейшее развитие в работах [1,2,3,6,7]. Однако при изменении пропорций и размеров объекта при его движении длительное слежение производить не удавалось, так как «нечуткое» окно захватывает слишком много фона и «соскальзывает» с объекта. В [3,6] эта проблема решена в частных случаях и найден способ слежения за объектом, у которого изменяются размеры, но не изменяются пропорции (человек приближающийся лицом не должен повернуться боком).

В предлагаемой работе построен алгоритм отыскания области кадра, имеющей наибольшую схожесть (максимум функции схожести в смысле коэффициента Бхаттачария) с заранее выделенным образцом. Данный алгоритм учитывает изменения как положения, так и размеров объекта. В отличие от [1–7] для отыскания локальных максимумов применен градиентный спуск.

Описанный в статье алгоритм определения области с гистограммой, максимально похожей на заданную, был реализован в MatLab'e. Для эксперимента был выбран ролик со свободно прохаживающимися людьми. Целью алгоритма было слежение за произвольно выбранным человеком. На первом кадре молодой человек в зеленой кофте был обведен вручную прямоугольником, и для получившейся области была посчитана гистограмма. Затем был запущен алгоритм, и на каждом следующем кадре по максимуму схожести цветовых гистограмм определялись размеры и положение области слежения.

На рисунке 1 приведены первый, 86-й, 131-й, 165-й и 168 с момента начала слежения кадры. На первом кадре размер прямоугольника 85×296 пикселей. Размеры прямоугольников на других кадрах получились 23×296, 58×271, 24×273, 24×273. Как видно, на 131 кадре молодой человек, за

которым ведется наблюдение, частично заслонен другим человеком – прямоугольник сузился. На 165-ом кадре молодой человек виден полностью, но лицом к нам и дальше от исходного положения – прямоугольник изменил размеры, причем снова не пропорционально первоначальным размерам, а согласно цели – максимуму схожести с первоначальной гистограммой.



Кадр 241. Кадр 261. Кадр 231

Рис 1.

Рис 2.

Отметим также, что в прямоугольник на 131 кадре не попадает часть темного фона от первоначальной области на первом кадре – изменился фон, а лица на первоначально обведенной области на первом кадре вообще видно не было, поэтому коэффициент схожести значительно снизился до 0,88.

Кадр 315. Кадр 412. Кадр 420.

Рис 3.

Однако, и положение, и размеры области, найденной на текущем кадре, адекватны размерам и положению видимой части области, отмеченной на первом кадре. На 165 и 168 кадрах объект слежения заслонен, но не потерян.

Кадр 1. Кадр 87. Кадр 98.

Рис 4.

На рисунках 2,3, 4 и 5 другой пример слежения. На первом кадре объект слежения был обведен вручную красным прямоугольником. Затем по цветовой гистограмме обнаруживалась область наибольшей схожести. В нарезке представлены 1, 87, 98, 231, 241, 261, 315, 412, 420, 421, 426 и 432 кадры. Особо отметим, что изображения не были масштабированы, но только обрезаны

На 1 кадре объект стоит близко спиной. На 87 и 98 кадрах он же, но дальше и левым боком. Размеры обводящего прямоугольника заметно изменились, причем они взаимно независимы. Аналогичную картину можно наблюдать на кадрах 231, 261, 315 и 432.

Кадр 421. Кадр 426. Кадр 432.

Рис 5.

На кадрах 412–432 объект слежения заслонили. Заметно, как по мере того, как слева остается видна все меньшая часть объекта, прямоугольник постепенно сокращается. Затем обнаруживается другая часть вновь появившегося объекта наблюдения. И по мере того, как объект все больше открывается, прямоугольник разрастается.

2. ОПИСАНИЕ ОБЪЕКТА ЦВЕТОВОЙ ГИСТОГРАММОЙ И ПОСТРОЕНИЕ ФУНКЦИИ СХОЖЕСТИ

Пусть задана функция $\mathrm{f}: P \to B$, где $P = [0, d_1] \times [0, d_2]$ – прямоугольник, B – конечное множество. Обозначим через $\chi_{\mathrm{f}^{-1}(b)}(x)$, где $x \in P$, характеристическую функцию полного прообраза $\mathrm{f}^{-1}(b)$ элемента $b \in B$, то есть $\chi_{\mathrm{f}^{-1}(b)}(x) = 1$, если $\mathrm{f}(x) = b$, и $\chi_{\mathrm{f}^{-1}(b)}(x) = 0$ в противном случае.

Для каждого $b \in B$ определим величину $H_b = \iint_P \chi_{\mathrm{f}^{-1}(b)}(x)\,dx_1 dx_2$. Можно предполагать, что этот интеграл существует (это выполнено, если цветовые пятна плотные и непрерывные, то есть множество точек разрыва функции $\chi_{\mathrm{f}^{-1}(b)}(x)$, а все они первого рода, имеет меру нуль).

Множество $\{H_b | b \in B\}$ называется гистограммой кадра.

Введем в рассмотрение семейство функций $\mathrm{g}(x) = \max\left(0, 1 - \left(\frac{x_1 - y_1}{l_1}\right)^2 - \left(\frac{x_2 - y_2}{l_2}\right)^2\right)$, где $x = (x_1, x_2) \in P$, x – переменная; $y = (y_1, y_2) \in P, l_1, l_2 \in R$, y_1, y_2, l_1, l_2 – параметры.

Пусть теперь $G \subset P$, G – эллиптическая область с центром в точке $y = (y_1, y_2)$ и полуосями l_1, l_2. Выбор эллиптической области обусловлен тем, что объект в кадре определяется своим центром $y = (y_1, y_2)$ и размерами $l = (l_1, l_2)$, то есть эллиптическая область хорошо подходит для выделения объекта в кадре.

Для определения гистограммы эллиптической области G рассмотрим функции $H_b(y, l) = \iint_P \chi_{\mathrm{f}^{-1}(b)}(x)\mathrm{g}(x)\,dx_1 dx_2$, где $y = (y_1, y_2)$, $l = (l_1, l_2)$. Величина $H_b = H_b(y_0, l_0)$ есть вес той части области G с центром в точке y_0 и размерами l_0, которая имеет данный цвет b, – вес цвета b. Цвет, который в центре области, вносит больший вклад в гистограмму. Заметим, что $\sum_{b \in B} H_b$ есть величина, характеризующая область G. Будем называть ее весом области.

Тогда множество чисел $H = \{H_b | b \in B\}$ будет гистограммой области. Ясно, что гистограмма H изменяется при изменении положения центра $y = (y_1, y_2)$ и размеров $l = (l_1, l_2)$ области.

Однако для построения функции схожести сами гистограммы $H = \{H_b | b \in B\}$ непригодны, так как H_b существенно зависят от размеров l области. А при пропорциональном увеличении или уменьшении (масштабировании) изображения объекта функция схожести должна идентифицировать получившиеся изображения как изображения одно и того же объекта. Поэтому рассмотрим $h_b = \frac{H_b}{\sum_{b \in B} H_b}$ – отношение веса цвета b к весу всей области, то есть относительный вес цвета.

Тогда $h = \{h_b | b \in B\}$ – нормированная гистограмма области G. Ясно, что нормированная гистограмма h изменяется при изменении $y = (y_1, y_2)$ и $l = (l_1, l_2)$, то есть является функцией переменных y_1, y_2, l_1, l_2.

Для двух нормированных гистограмм h_0 и h определим, следуя [4,5], функцию схожести $\rho = \sum_{b \in B} \sqrt{h_{0b} \cdot h_b}$, где $h_0 = \{h_{0b} | b \in B\}$ – некоторая гистограмма, с которой производим сравнение.

Тогда ρ есть функция, зависящая от параметров $y = (y_1, y_2)$ и $l = (l_1, l_2)$. Для этой функции $\rho = \rho(y, l)$ получаем $\rho: R^3 \to R$, если размеры объекта l_1 и l_2 связаны каким-либо соотношением, например, пропорциональны как при приближении одноцветного шара, или $\rho: R^4 \to R$, если l_1, l_2 не зависят друг от друга, или $\rho: R^5 \to R$, если ввести еще один параметр α, который обеспечивает изменение угла поворота области. При введении в рассмотрение параметра α в равенстве

$H_b(y,l) = \iint\limits_P \chi_{\mathrm{f}^{-1}(b)}(x)\mathrm{g}(x)dx_1dx_2$ в качестве функции $\mathrm{g}(x)$ можно использовать

$$g = \max(0, 1 - \left(\frac{(x_1 - y_1)\cos\alpha - (x_2 - y_2)\sin\alpha}{l_1}\right)^2 - \\ - \left(\frac{(x_1 - y_1)\sin\alpha + (x_2 - y_2)\cos\alpha}{l_2}\right)^2),$$

$\alpha \in [-90^\circ, 90^\circ]$.

Для простоты запишем $\rho = \sum_{b\in B}\rho_b$, где $\rho_b = \sqrt{h_{0b}\cdot h_b}$.

3. ГРАДИЕНТНЫЙ СПУСК

Найдем градиент функции схожести $\rho : R^4 \to R$

$$\mathrm{grad}\rho = \left(\frac{\partial\rho}{\partial y_1}, \frac{\partial\rho}{\partial y_2}, \frac{\partial\rho}{\partial l_1}, \frac{\partial\rho}{\partial l_2}\right)$$

Сначала находим $\frac{\partial H_b}{\partial y_i} = \frac{2}{l_i^2}\iint\limits_P \chi_{f^{-1}(b)}(x)\chi_{\mathrm{supp}(\mathrm{g}(x))}(x)(x_i - y_i)dx_1dx_2$, где $\mathrm{supp}(\mathrm{g})$ – носитель функции g, то есть множество точек, в которых функция принимает ненулевое значение, $\frac{\partial H_b}{\partial l_i} = \frac{2}{l_i^3}\iint\limits_P \chi_{f^{-1}[b]}(x)\chi_{\mathrm{supp}(\mathrm{g}(x))}(x)(x_i - y_i)^2 dx_1dx_2$,

$$\frac{\partial h_b}{\partial y_i} = \frac{\frac{\partial H_b}{\partial y_i} - h_b\sum_{b\in\mathrm{B}}\frac{\partial H_b}{\partial y_i}}{\sum_{b\in\mathrm{B}} H_b},$$

$$\frac{\partial h_b}{\partial l_i} = \frac{\frac{\partial H_b}{\partial l_i} - h_b\sum_{b\in\mathrm{B}}\frac{\partial H_b}{\partial l_i}}{\sum_{b\in\mathrm{B}} H_b}, \text{ где } i = 1,2.$$

Тогда имеем $\frac{\partial\rho}{\partial y_i} = \sum_{b\in\mathrm{B}}\frac{1}{2h_b}\frac{\partial h_b}{\partial y_i}\rho_b$, $\frac{\partial\rho}{\partial l_i} = \sum_{b\in\mathrm{B}}\frac{1}{2h_b}\frac{\partial h_b}{\partial l_i}\rho_b$, $i = 1,2$.

Для функции схожести ρ (в предположении ее дифференцируемости) имеем

$$\rho \approx \rho(y_0, l_0) + \mathrm{grad}\rho|_{(y_0,l_0)}\cdot(\Delta y_1, \Delta y_2, \Delta l_1, \Delta l_2) = \\ = \rho(y_0, l_0) + t\left|\mathrm{grad}\rho|_{(y_0,l_0)}\right|^2,$$

где $(\Delta y_1, \Delta y_2, \Delta l_1, \Delta l_2) = t\cdot\mathrm{grad}\rho|_{(y_0,l_0)}$. То есть вектор приращений выбираем в направлении градиента, так как в каждой точке вектор градиента по численному значению и по направлению характеризует наибольшую скорость возрастания величины, в данном случае схожести.

Вообще говоря, знание градиента в точке дает лишь направление шага, а для определения величины шага этого недостаточно. Однако знание максимального значения функции схожести $\rho = 1$ дает приблизительную оценку для $t \approx \frac{1-\rho(y_0,l_0)}{|\mathrm{grad}\rho|^2}$. Откуда для величины шага имеем

$$(\Delta y_1, \Delta y_2, \Delta l_1, \Delta l_2) \approx \frac{1-\rho(y_0,l_0)}{|\mathrm{grad}\rho|^2}\mathrm{grad}\rho.$$

4. ЗАКЛЮЧЕНИЕ

Отметим, что предложенный выше метод слежения не учитывает угол поворота объекта относительно оси, перпендикулярной плоскости кадра. То есть он эффективен, если объект слежения движется, оставаясь параллельным самому себе (поступательно) и его наибольшее измерение параллельно одной из сторон кадра. Например, такая ситуация складывается при слежении камерой вдоль коридора, по которому движутся люди. При слежении за вращающимся некруглым объектом следует ввести угол поворота. В этой задаче функция схожести $\rho : R^5 \to R$ и мы будем иметь, как указано выше, пять параметров. Рассмотрение этой модели позволит расширить класс обрабатываемых видеопоследовательностей. Например, можно будет производить компьютерное слежение сверху за автомобилями на перекрестках.

5. ССЫЛКИ

[1] R. Szeliski, Computer Vision: Algorithms and Applications. Microsoft Research. URL: http://szeliski.org/Book .

[2] D. Comaniciu, V. Ramesh, and P. Meer. Real-Time Tracking of Non-Rigid Objects Using Mean Shift // Proc. Conf. Computer Vision and Pattern Recognition, 2000. Vol. 2. P. 142-149.

[3] D. Comaniciu, P. Meer. Mean Shift: A Robust Approach Toward Feature Space Analysis // IEEE Transactions on Pattern Analysis and Machine Intelligence, May 2002. Vol. 24. № 5. P. 603-619.

[4] R. Collins, Mean-shift Blob Tracking through Scale Space // IEEE Computer Vision and Pattern Recognition (CVPR03), Madison, WI, June 16-22, 2003. P. 234-240.

[5] D. Comaniciu, V. Ramesh, and P. Meer, The Variable Bandwidth Mean Shift and Data-Driven Scale Selection // Proc Eighth Int'l Conf. Computer Vision, July 2001. Vol. I. P. 438-445.

[6] S. T. Birchfield, S. Rangarajan. Spatiograms Versus Histograms for Region-Based Tracking // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), San Diego, California, June 2005. Vol. 2. P. 1158-1163.

[7] C. O'Conaire, N. E. O'Connor, A. F. Smeaton. An improved spatiogram similarity measure for robust object localization // Proceedings of the 2007 IEEE International Conference on Acoustics, Speech, and Signal Processing, 2007. Vol. 1. P. 1069-1072.

[8] Ning He, Jiaheng Cao, Lin Song, Kernel Based Spatiogram Tracking Using Improved Similarity Measure // Ninth IEEE International Conference on Computer and Information Technology, 2009. Vol. 1. P. 124-127.

[9] Р. Гонсалес, Р. Вудс, С. Эддинс. Цифровая обработка изображений. М.: Техносфера, 2006.

ОБ АВТОРАХ

Елена Васильевна Шапошникова - доцент каф. общей математики ММФ ТГУ. e-mail: albertelena@sibmail.com.

Альберт Игоревич Шапошников - научный консультант ООО «Tomsklabs». e-mail: albertelena@mail.ru

The new molecular surface descriptors

A.M. Shestov, A.V. Koptsova, M.I. Kumskov

Department of Computational Mathematics, Faculty of Mechanics and Mathematics, Lomonosov Moscow State University, 119992 Moscow, Russian Federation.
E-mail: qsar_msu@mail.ru, shestov.msu@gmail.com

ABSTRACT

A new method for extracting geometric features (we call them singular points) of triangulated molecular surface is proposed. For this purpose we developed an algorithm of molecular surface segmentation, which combines several existing techniques. New 3d structural descriptors (we will call them the molecular surface descriptors) are constructed on the base of singular points, which take into account a geometry of molecular surface and local physico-chemical properties of the compound. These descriptors are used for the solution of the QSAR problem. The algorithm is developed and its complexity and needed memory are estimated. The project is implemented on the C++ language and tested on the 2 training sets – the glycosides and the toxic compounds.

Keywords: *QSAR, Descriptors, Molecular Surface, Segmentation.*

1. INTRODUCTION

1.1 QSAR problem definition

The QSAR problem is as follows: to determine the relationship between the structure of chemical compounds and the properties [8]. The solution of this problem can be presented in two stages: the choice of the molecular graphs description and the discriminant function construction. The second stage can be solved by general methods of the pattern recognition [15]. The difference between the QSAR problem and other problems of pattern recognition is the first stage of the solution (the molecular graphs description). The molecular graphs description is the problem which this paper is devoted to.

Let us consider the basic definitions.

The molecular graph is a singly labeled graph whose vertices are interpreted as atoms of the molecule, and the edges – as covalent bonds between pairs of atoms. Vertices and edges may have additional attributes such as coordinates for the vertices, or information about whether a connection is ring. Labels of vertices are the names of the atoms in the periodic table, labels of the edges are the types of bonds between atoms (single, double, triple, aromatic). Each vertex of the graph corresponds to the three real numbers – the location of the atom in some coordinate system.

The descriptor is any property which numerical value can be computed for any molecular graph.

Consider for each atom A_i of a molecule G a ball $B_o(V_i, r_i)$, which radius r_i is a **Van der Waals radius** [16] of the atom V_i and the center V_i coincides with the center of A_i. **The Van der Waals (VDW) surface *V*(*G*)** is the border of the union of these balls (see fig. 1): $V(G) = \partial \bigcup_{i=1}^{n} B_o(V_i, r_i)$

Consider a probe sphere ***S*** of radius ***r***. This sphere is "rolling" over the **Van der Waals surface *V*(*G*).** **The molecular surface** (or **Connolly surface**) $M_r(G)$ [2] of radius **r** of a molecule ***G*** is the surface that is accessible to a probe sphere of the radius ***r*** rolling over the **Van der Waals surface** (see fig. 1):

$$M_r(G) = \partial\{x \in R^3 : \exists y \notin \bigcup_{i=1}^{n} B(P_i, r_i), \rho(x, y) \le r\}$$

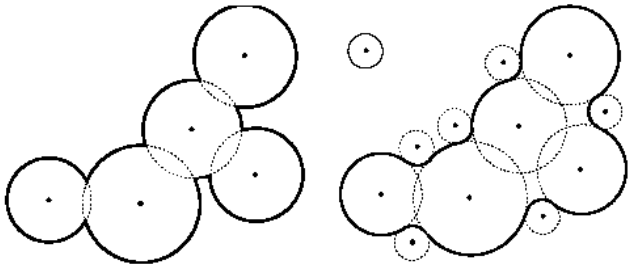


Fig. 1 VDW (left) and molecular (right) surfaces

1.2 Definition of the problem of the molecular surface descriptors construction

Decide that we are given a chemical compound ***M*** and its triangulated molecular surface ***G***. The problem is as follows: to develop a way of describing a molecular surface ***G*** with a set of singular points $\boldsymbol{\Sigma}=\{\boldsymbol{\xi}_i\}$, $i = 1,\ldots,n$, which fulfill the following conditions:

1. Every singular point $\boldsymbol{\xi}_i$ must refer to some geometric feature $\boldsymbol{g_j}$ of the surface ***G***, e.g. *the description must be informative.*
2. Every geometric feature $\boldsymbol{g_j}$ must refer to one and only one singular point $\boldsymbol{\xi}_i$, e.g. *the description must be total and nonredundant.*

Then, decide we are given a training set $\boldsymbol{L}=\{\boldsymbol{M_i}\}$, *i=1,…,m.*, where each compound $\boldsymbol{M_i}$ have a triangulated molecular surface $\boldsymbol{G_i}$. When a set of singular points $\boldsymbol{\Sigma}_i$ is constructed for each molecule $\boldsymbol{M_i}$ of the training set, the alphabets of descriptors (we will call them the molecular surface descriptors further) are constructed in the following way:

1. Several local physical or chemical properties are calculated in each singular point $\boldsymbol{\xi}_j$ for each molecule $\boldsymbol{M_i}$ (e.g. electrostatic potential, lipophilicity, hydrophoby, molecular fields [5]).
2. Each singular point is classified according to the type of singularity, which it refers to, and the value of calculated

property: $\boldsymbol{l_j} = T(\boldsymbol{\xi_j})$. This is the way the alphabet of descriptors A^1 is calculated.

3. Each pair of singular points (ξ_i, ξ_j) is classified according to types $T(\boldsymbol{\xi_j})$, $T(\boldsymbol{\xi_i})$ and the type of distance between them $T(\|\boldsymbol{\xi_j}, \boldsymbol{\xi_i}\|)$: $\boldsymbol{l_{ij}} = T(\boldsymbol{\xi_j}, \boldsymbol{\xi_i})$. This is the way the alphabet of descriptors A^2 is calculated.
4. Each triple of singular points (ξ_i, ξ_j, ξ_l) is classified according to types $T(\boldsymbol{\xi_j})$, $T(\boldsymbol{\xi_i})$, $T(\boldsymbol{\xi_l})$ and the type of triple of distances between them $T(\|\xi_i, \xi_j\|, \|\xi_i, \xi_l\|, \|\xi_j, \xi_l\|)$: $\boldsymbol{l_{ijl}} = T(\xi_i, \xi_j, \xi_l)$. This is the way the alphabet of descriptors A^3 is calculated.

The constructed alphabets are used for the predicting activity of the new compounds.

1.3 The topicality of the problem of the molecular surface descriptors construction

The activity of a molecule is encoded in its shape. Of all geometric properties of a molecule, its surface play an essential role as it delineates the region covered by the protein and therefore defines its region of interactions. Characterization of the compound's molecular surface therefore plays an essential role for analyzing and predicting biomolecular complexes, as well as for modeling the energetic of formation of such complexes. As the surface of a molecule also defines its interface with the solvent it bathes in, its form is crucial for understanding the salvation [9].

So, a set of singular points $\boldsymbol{\Sigma}=\{\boldsymbol{\xi_i}\}$, classified according to the neighborhood shape and physico-chemical properties, is informative from the chemical point of view.

One of the ligand-receptor interaction models is a "triangle of activity" [8]. *So, a set of pairs and triples of singular points, classified according to types of singular points and type of distances between them, is also informative from the chemical point of view.*

Also the developed method of molecular surface segmentation can be used for docking [6].

2. THE DESCRIPTION OF THE ALGORITHM

2.1 The idea of the singular points extraction

The idea is to divide a molecular surface G into the set of segments $\{S_i\}$, $i = 1 \ldots n_s$, in such a way, that each segment S_i is a region of a specific shape – convex, concave or saddle. Then we take centers of constructed segments as singular points. Because each region of a specific shape S_i contains one and only one geometric feature, the obtained set of singular points will fulfill the conditions formulated in chapter 3. This idea is new approach for extracting geometric features.

So, the key question is how to segment a molecular surface.

2.2 The general approach to the segmentation of triangulated surfaces

Let us consider general approach for segmenting triangulated surface.

Decide we are given a triangulated surface $\boldsymbol{G}$, which consists of a set of vertices $\boldsymbol{V}=\{\boldsymbol{v_i}\}$, $i=1 \ldots n_v$ and a set of triangles $\boldsymbol{T}=\{\boldsymbol{t_i}\}$, $i=1 \ldots n_t$. The most common approach for segmenting a triangulated surface G can be presented in two stages:

1. The choice of scalar function $F:G \rightarrow \boldsymbol{R}$ and its calculation for each vertex $\boldsymbol{v_i}$.
2. Segmentation of the surface according to the value of F.

The methods of segmentation can be generally divided into 2 types – *feature-based segmentation* and *part-based segmentation*. For our purpose we need the *feature-based segmentation.*

Let us consider the first stage. The most frequently used functions are mean and Gaussian curvatures and some their combinations. We can't compute these curvatures directly, because only triangulation of the molecular surface is given. So we need to estimate it. A review of different methods for the principal directions estimation can be found in [4]. The segmentation method based on the curvature can be found in work [14] . Also there were developed functions for segmenting the molecular surface precisely – Connolly function [1] and atomic density function [10].

Let's consider the second stage. There are 2 general approaches – *extracting regions* and *extracting borders.* When we extract regions, we unite elements of surface (vertices or triangles) with similar values of F into same regions. The examples can be found in [7][14] . When we extract borders, we treat elements of surface (vertices or triangles) with high values of F as borders between regions. The examples can be found in [10][1] .

In this work we use Haussian and mean curvature as F and after that extracted regions. The choose of curvature is explained by two facts –

1. Due to the method of construction of the molecular surface curvature perfectly suits for its segmentation. This is explained in the next chapter.
2. Unlike the Connolly function and the atomic density function curvature is easy calculated.

2.3 The idea of the triangulated molecular surface segmentation

The molecular surface is composed of fragments of spheres, toruses and reentrant surfaces of spheres. A part of sphere is obtained when a probe sphere touches only one sphere of VDW surface, a part of torus – when a probe touches 2 spheres, a part of reentrant surface of sphere – when a probe touches 3 spheres [12].

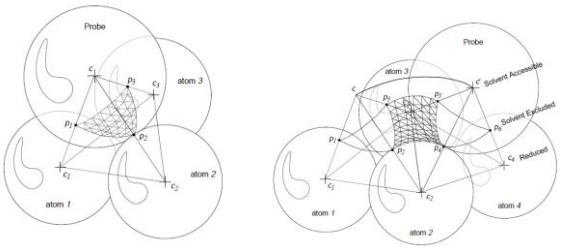


Fig. 2 Torus and reentrant surface of sphere

So, parts of different spheres of WDV surface are separated by parts of toruses, parts of reentrant surfaces of spheres are separated by parts of toruses too.

These 3 different types of molecular surface compounds (spheres, toruses and reentrant surfaces of spheres) can be distinguished by the sign of their Gaussian and mean curvatures. Denote Gaussian curvature as $\boldsymbol{K}$ and mean curvature as $\boldsymbol{H}$. Then a sphere has $K>0$, $H>0$, torus - $K<0$, reentrant surface of sphere - $K>0$, $H<0$.

So, the idea of algorithm is:

1. We assign each vertex to one of 3 types (*convex, concave, saddle*) according to its Haussian and mean curvature.

2. Extract regions of spheres (*convex regions*), regions of reentrant surface of spheres (*concave regions*), as they are separated by regions which have negative Gaussian curvature.

2. Regions which have negative Gaussian curvature (*saddle regions*) in most cases form a small number of connectivity components. So they can be segmented by the algorithm of clustering *k-means* [11].

The strong point of the proposed idea is the simplicity of calculations – we need only a sign of curvature, not its value. Because of this the result of this algorithm must suffer less from irregularities in triangulation of a surface.

2.4 The implementation of the algorithm of segmentation

2.4.1 Molecular surface construction

For the molecular surface constructing we used the program Accelrys Discovery Studio 2.5 [17]. It differs from the other programs for the molecular surface construction with 2 facts –

1. The quality of triangulation is better than in other programs.
2. The triangulated surface can be saved in *wrl* format, which provides not only vertices and triangles, but also a normal to surface in each vertex. So, we have no need to estimate surface normals. And estimation of normals is not a simple question, many articles are devoted to this problem.

2.4.2 The curvature calculation

We chose 2 algorithms for curvature estimation – estimation by a paraboloid [4] and estimation by a cubic-order surface [4]. The first is used for the Gaussian curvature calculation and the second – for the mean curvature calculation.

The essence of these methods is the following: we try to estimate the neighborhood surface of the vertex of the triangulation by a paraboloid or by a cubic-order surface. We obtain a overdetermined system of linear equations, which we solve by the least-squares method. The result is the analytical form of the paraboloid or the cubic-order surface. Then we calculate principal curvatures, Haussian(K) and mean(H) curvatures of the obtained surface. Then we assign each vertex to one of three types: convex ($K>0$, $H>0$), saddle ($K<0$), concave ($K>0$, $H<0$).

There occur errors because of the irregularities in triangulation, such that for the vertex v_i of type k_1 all adjacent vertices v_j belong to another type k_2.We use the following solution: we assign vertex v_i to the type k_2.

2 another types of errors are considered in the next chapter.

After this stage of algorithm all vertices of triangulated surface belong to one of 3 types: concave, convex or saddle.

2.4.3 Convex and concave regions extraction

There are 2 variants of segmentation – to take vertices as elements of segments or to take triangles. If we take triangles we must develop a way to classify them as we did it for vertices. We do it in the following way: triangle t is assigned to a convex (concave) type if and only if all its vertices v_{t1}, v_{t2}, v_{t3} have the convex (the concave) type. Otherwise t is assigned to a saddle type.

This procedure helps to fix the following error of curvature calculation: some convex or concave regions which are meant to be disconnected become connected. If we take triangles as elements of segments different concave (convex) regions will be disconnected.

2.4.4 Saddle regions segmentation

Regions with $K<0$ (*saddle regions*) in most cases form a small number of connectivity components. Each connectivity component is segmented distinguishly. We tried 2 algorithms for segmentation – *k-means* [11] and *mountain clustering* [13]. In this stage we take vertices as elements of segments. For each algorithm we need to define distances between vertices. We take the shortest geodesic distance as distance between vertices. Because of it each obtained cluster has only one component of the connectivity and can be treated as a segment. The shortest distances are calculated by the Dijkstra algorithm with a binary heap as a priority queue [3]. *k-means* provides better segmentation, so we chose it instead of the *mountain clustering*.

After this step the surface is totally segmented.

2.4.5 Singular points extraction

In each segment we take as a singular point the vertex which is the closest to the geometric center of a segment:

$$\xi_i = arg\ min_{v_j \in S_i} \left\| v_j - \frac{1}{n_i} \sum_{x_m \in S_i} v_m \right\|$$

We can see segmented molecular surface with singular points on the figure.

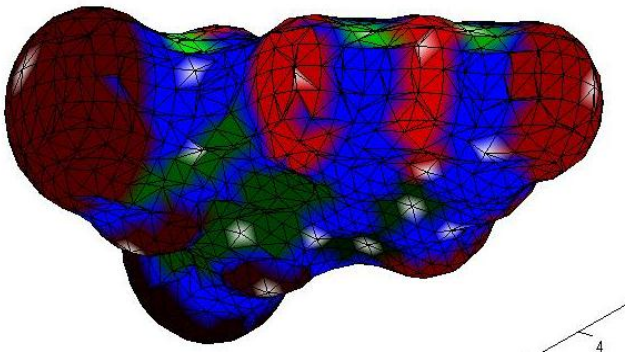
3. THE COMPLEXITY OF THE ALGORITHM OF SINGULAR POINTS EXTRACTION

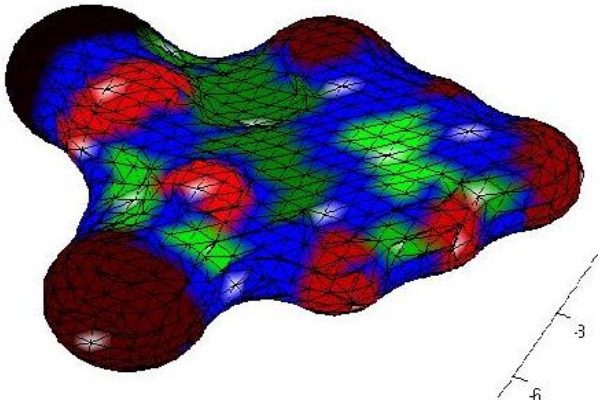
The complexity is - $\boldsymbol{\Theta(N_t{}^2 ln\ N_t)}$.

The memory which we need - $\boldsymbol{\Theta(N_t{}^2)}$, where N_t is a number of triangles in the triangulation of the molecular surface.

4. THE RESULTS

On the figures you can see segmented molecular surfaces with singular points.





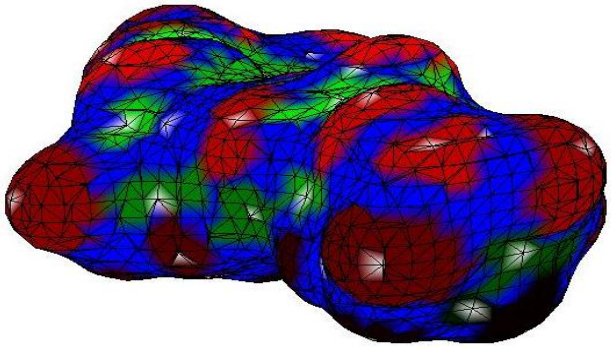


Fig. 3 Segmented molecular surfaces with singular points

On the base of singular point we constructed the alphabet of descriptors A^2 (see the definition). The constructed descriptors were tested on the 2 training sets – the glycosides and the toxic compounds. For each set we constructed 8 molecular-descriptor matrices [8]. We took a percentage of correctly classified compounds as the prediction quality. The results are the following:

№	The prediction quality	Ejections
1	0.7962	2
2	0.8454	0
3	0.8113	4
4	0.8504	3
5	0.8504	3
6	0.7909	0
7	0.8055	2
8	0.8545	0

Table 1 The glycosides

№	The prediction quality
1	0.49
2	0.48
3	0.44
4	0.51
5	0.52
6	0.45
7	0.49
8	0.51

Table 2 The toxic compounds

5. THE CONCLUSION

The algorithm of describing a molecule with a set of singular points is proposed. The complexity of the algorithm and the needed memory are estimated. 3 alphabets of descriptors are constructed on the base of singular points. Experiments show, that these descriptors a suitable for predicting the activity of new compounds.

The further direction of the work is to build a fuzzy classification of singular points. Then the prediction quality will continuously depend on the initial parameters of the algorithm.

6. ACKNOLEDGMENTS

This work is supported by the Russian Foundation for Basic Research, project no. 10–07–00694.

7. REFERENCES

1. F. Cazals, F. Chazal, T. Lewiner: "Molecular Shape Analysis based upon the Morse-Smale Complex and the Connolly Function". *Annual Symposium on Computational Geometry, Proceedings of the nineteenth annual symposium on Computational geometry, San Diego, California, USA, SESSION: Topology* (2003), 351 – 360.
2. M. Connolly: "Analytical molecular surface calculation", *J. Appl. Crystallogr*., i.16 (1983), 548–558.
3. T. Corman, C. Leiserson: "Introduction to Algorithms"
4. J. Goldfeather, V. Interrante: "A Novel Cubic-Order Algorithm for Approximating Principal Direction Vectors", *ACM Transactions on Graphics (TOG)*, v. 23, i. 1 (2004) .
5. T. Halgren: "Merck molecular force field", *J.Comp Chem*, i. 17 (1996), 490-641.
6. H. Holtje, W. Sippl, D. Rognan, G. Folkers: "Molecular modeling. Basic principles and applications".
7. G. Lavoue, F. Dupont, A. Baskurt: "A new CAD mesh segmentation method, based on curvature tensor analysis", *Computer-Aided Design*, i. 37 (2005), 975–987.
8. G. Makeev, M. Kumskov , I. Svitan'ko , I.Zyryanov: "Recognition of Spatial Molecular Shapes of Biologically Active Substances for Classification of Their Properties", *Pattern Recognition and Image Analysis*, v.6, i.4 (1996), p.795-808.
9. V. Natarajan, P. Koehl, Y. Wang, B. Hamann: "Visual Analysis of Biomolecular Surfaces", *Mathematics and Visualization*, i.5 (2008), 237-255.
10. V. Natarajan,Y. Wang, P.-T. Bremer,V. Pascucci, B. Hamann "Segmenting molecular surfaces", *Computer Aided Geometric Design*, i. 23 (2006), 495–509
11. A. Perevoznikov, A. Shestov, E. Permyakov, M.Kumskov: "A Way to Increase the Prediction Quality for the Large Set of Molecular Graphs by Using the *k*–NN Classifier", *Pattern Recognition and Image Analysis*, v. 21, i.2 (2011).
12. M. Sanner, A.Olson, J. Spehner.: "Reduced Surface: an Efficient Way to Compute Molecular Surfaces" *Biopolymers*, v. 38, i.3 (1996), 305-320.
13. S.Shtovba: "Introduction into fuzzy logic", 12.2.1.
14. T. Srinark, C. Kambhamettu: "A novel method for 3D surface mesh segmentation", *Proceedings of the 6th IASTED International Conference on Computers, Graphics, and Imaging* (2003), 212-217.
15. G. Tu, R. Gonsales: "Principles of pattern recognition", ed. "Mir" (1978).
16. U. Zefirov, P. Zorkiy : "Van der Waals radii of atoms In Crystal Chemistry and Structural Chemistry (History)". *Problems of Crystal Chemistry*, ed. "Science" (1992), 6-24.
17. http://accelrys.com/products/discovery-studio/visualization-download.php

Применение Методов Машинного Обучения к Задаче Автоматического Распознавания Пола и Возраста Людей по Изображению Лица

Лев Шмаглит, Владимир Хрящев
Физический факультет
Ярославский Государственный Университет имени П.Г.Демидова, Ярославль, Россия
Lev_shmaglit@yahoo.com, vhr@yandex.ru

Аннотация

В данной работе представлен алгоритм автоматического распознавания пола и возраста людей по изображению лица. В его основе лежат современные методы машинного обучения, такие как линейный дискриминантный анализ и метод опорных векторов. Разработанный алгоритм позволяет решать задачу разбиения анализируемых объектов на три класса («мужчины», «женщины» и «дети») с точностью 89%.

Ключевые слова: *Выделение Лиц, Распознавание Пола, Машинное Обучение.*

1. ВВЕДЕНИЕ

Алгоритмы, осуществляющие автоматический анализ и распознавание лица человека, находят применение в системах технического зрения, робототехнике, системах видеонаблюдения и контроля доступа, в интерфейсах взаимодействия человек-компьютер [6]. Одной из наиболее актуальных задач, решаемых данным классом алгоритмов, является классификация анализируемых объектов по полу (gender recognition) [4]. Распознавание пола востребовано в технологии Digital Signage – представление информации с электронных носителей (дисплеев, проекционных систем), установленных в общественных местах. Кроме того, распознавание пола может использоваться, к примеру, для сбора и оценки демографических показателей, а также как важный этап предобработки при решении задачи идентификации личности, поскольку оно позволяет вдвое (в случае одинакового числа мужчин и женщин в базе данных) уменьшить число кандидатов при анализе, и, таким образом, вдвое ускорить процесс идентификации. Следует отметить, что методы, применяемые в задаче распознавания пола, являются универсальными, и поэтому могут быть успешно использованы для решения других задач в области распознавания образов [6].

Для организации полностью автоматической системы, распознавание пола используется совместно с алгоритмом выделения лиц, который отбирает кандидатов для анализа [3]. От качества работы алгоритма выделения лиц во многом зависит итоговый результат всей системы, поскольку неточности при определении местоположения лица на изображении могут привести к принятию ошибочного решения на этапе распознавания.

Выделенные фрагменты подвергаются предобработке для приведения их к единому разрешению и яркостным характеристикам. Нами использовалось масштабирование до разрешения 40×40 пикселей. Эта величина была подобрана экспериментально, как компромисс между скоростью и качеством распознавания. Для устранения различий в степени освещенности и контрасте к выделенным фрагментам применялась процедура выравнивания гистограммы яркости.

На последнем этапе выделенные и обработанные фрагменты поступают на вход классификатора, который принимает решение об их принадлежности к тому или иному классу. В нашей работе, в отличие от классической постановки задачи распознавания пола, рассматриваются три класса: «мужчины», «женщины» и «дети». Для решения такой задачи предлагается организовать систему из трех двухклассовых классификаторов («мужчины» - «женщины»; «мужчины» - «дети»; «женщины» - «дети»), обученных на наборе тренировочных изображений, и принимающих совместное решение путем простого голосования. Подобный подход позволяет упростить классификатор и ускорить процесс обучения без потери качества распознавания.

Дальнейшее повествование построено следующим образом: вначале приведено описание предлагаемого алгоритма (рассмотрены используемые методы обучения классификатора и этап автоматического выделения лиц), затем представлена методика и результаты его тестирования.

2. АВТОМАТИЧЕСКОЕ ВЫДЕЛЕНИЕ ЛИЦ

Анализируя произвольное изображение, необходимо сначала определить, имеются ли на этом изображении лица, где находится каждое лицо и каков его размер. Эту задачу решают алгоритмы выделения лиц. Их работа заключается в сканировании входного изображения окном, имеющим определенную форму и различный масштаб, и в определении к какому классу относится изображение внутри этого окна («лицо», либо «не лицо»). За последние несколько лет было предложено множество алгоритмов выделения лиц, использующих различные подходы [3].

В нашей работе был использован современный алгоритм выделения лиц, базирующийся на обучающей сети SNoW (Sparse Network of Winnows) [5]. Этот алгоритм отличается высокой точностью в определении местоположения лица и низкой ошибкой классификации. Рассмотрим подробнее его структуру и принцип работы.

Алгоритм на базе SNoW можно условно разбить на три этапа:

2.1 Переход от значений пикселей к локальным SMQT признакам

SMQT [5] (Successive Mean Quantization Transform) – это преобразование, которое позволяет извлечь из локальной области изображения составляющую, не зависящую от освещенности. Оно заключается в квантовании области изображения с порогом квантования, равным среднему значению пикселей, входящих в эту область. При этом

локальная область задается как блок разрешением 3×3 пикселя.

Для нахождения лиц на изображении используется окно детектора разрешением 32×32 пикселя. Сканирование осуществляется с шагом в один пиксель, как по горизонтали, так и по вертикали. Для того чтобы выделять лица разного размера, изображение многократно масштабируется с коэффициентом масштабирования 1,2. Чтобы избежать краевых эффектов, к окну детектора применяется овальная маска, содержащая 648 пикселей. Каждому из этих 648 пикселей соответствует свой SMQT признак.

Переход к SMQT признакам позволяет алгоритму адаптироваться к изменению освещенности объекта, так как SMQT признаки извлекают из изображения не зависящие от освещенности компоненты.

2.2 Классификатор на базе обучающей сети SNoW

Обучающая архитектура SNoW (Sparse Network of Winnows) представляет собой разреженную сеть линейных элементов в пространстве признаков [5]. Большим достоинством сети SNoW является возможность создания весовых таблиц для классификации. Пусть x – значение одного пикселя входного изображения, W – набор SMQT признаков $M(x)$, тогда классификатор может быть получен посредством использования весовой таблицы для не лиц $h_x^{nonface}$, весовой таблицы для лиц h_x^{face}, и определения порога θ. Решающее правило такого классификатора может быть представлено в виде:

$$\theta = \sum_{x \in W} h_x^{nonface}(M(x)) - \sum_{x \in W} h_x^{face}(M(x)).$$

Поскольку обе таблицы работают в одном домене, их можно объединить в одну весовую таблицу: $h_x = h_x^{nonface} - h_x^{face}$.

Для обучения классификатора использовалась база тренировочных изображений лиц и не лиц. Изображения лиц были получены с помощью web-камеры. Далее на них вручную были отмечены три точки: правый глаз, левый глаз и центральная точка внешней границы верхней губы. После этого лицо деформировалось в блок 32×32 пикселя с различным расположением опорных точек. Деформация необходима для того, чтобы сымитировать различное положение лица по отношению к камере. В результате описанной процедуры был получен обучающий набор лиц емкостью порядка 1 миллиона образцов. База обучающих изображений не лиц первоначально состояла из случайным образом сгенерированных изображений. После того, как она была дополнена ошибками классификации, ее емкость также составила порядка 1 миллиона образцов.

Каждому обучающему изображению соответствует свой набор SMQT признаков. Пусть тренировочная база состоит из $i = 1,2,\ldots,K$ наборов SMQT признаков $M_i(x)$ и соответствующих каждому набору значений классов c_i («лицо» или «не лицо»). Тогда весовые таблицы лиц и не лиц могут быть обучены с помощью закона обновления весов, получившего название Winnow Update Rule [5]. Изначально обе таблицы содержат нули. При первом обращении к элементу таблицы в процессе обучения ему присваивается значение 1. Три параметра системы задаются вручную: порог γ, повышающий коэффициент $\alpha > 1$ и понижающий коэффициент $\beta < 1$. Их значения были заданы следующим образом: $\alpha = 1{,}005$, $\beta = 0{,}995$, $\gamma = 200$. Если $\sum_{x \in W} h_x^{face}(M_i(x)) \le \gamma$ и c_i – лицо, то веса обновляются согласно формуле:

$$h_x^{face}(M_i(x)) = \alpha h_x^{face}(M_i(x)), \qquad \forall x \in W.$$

Если $\sum_{x \in W} h_x^{face}(M_i(x)) > \gamma$ и c_i – лицо, то значение весов уменьшается:

$$h_x^{face}(M_i(x)) = \beta h_x^{face}(M_i(x)), \qquad \forall x \in W.$$

Эта процедура повторяется до достижения неизменного значения весов. Обучение весовой таблицы для не лиц происходит точно таким же образом. И, наконец, общая весовая таблица находится по формуле $h_x = h_x^{nonface} - h_x^{face}$.

2.3 Организация каскадной структуры

С целью увеличения быстродействия алгоритма полный классификатор SNoW разбивается на более слабые классификаторы, которые организуются в каскадную структуру (алгоритм Split-up SNoW). При этом не требуется дополнительного обучения слабых классификаторов.

Пусть все возможные значения одного признака ограничиваются множеством $P_i, i = 1,2,\ldots,2^N$, тогда вклад каждого признака в разделение классов можно оценить по формуле: $v_x = \sum_{i=1}^{2^N} |h_x(P_i)|, \quad \forall x \in W$. Отсортировав признаки по соответствующим им значениям v_x, можно получить рейтинг лист признаков (importance list). Пусть $W' \in W$ – это подпространство, содержащее определенное количество признаков с наибольшим рейтингом. Тогда: $\theta' = \sum_{x \in W'} h_x(M(x))$ можно определить как слабый классификатор, который выделяет все лица из тренировочной базы, но имеет больший по сравнению с полным классификатором уровень неверного выделения. Увеличить количество слабых классификаторов в каскаде можно за счет тех же самых операций, выбирая подпространства следующим образом: $W' \in W'' \in \ldots \in W$.

Каскад состоит из 5 слоев. Разбиения происходят при 20, 50, 100, 200 и 648 признаках. Признаки в каждый слой каскада отбираются согласно их рейтингу (importance list). Каждый слой отбирает кандидатов для следующего слоя и передает следующему слою значения уже посчитанных для этих кандидатов SMQT признаков.

3. ПРОЦЕДУРА РАСПОЗНАВАНИЯ ПОЛА

На этапе распознавания пола и возраста нами применялись два различных метода обучения классификатора: метод опорных векторов (МОВ) [1] и KDDA (Kernel Direct Discriminant Analysis) [2]. Эти методы являются одними из наиболее эффективных в задаче разделения изображений на два класса [6].

3.1 Линейный дискриминантный анализ

Рассмотрим алгоритм построения классификатора KDDA, основанный на линейном дискриминантном анализе. Основная идея заключается в том, чтобы представить входные обучающие изображения в виде n-мерных векторов, а затем найти такое подпространство, в котором проекции векторов, принадлежащих разным классам, были бы расположены как можно дальше друг от друга, а проекции векторов, принадлежащих одному и тому же классу, наоборот, – ближе друг к другу.

Однако в случае анализа лиц классы могут оказаться линейно неразделимыми. Для решения проблемы линейной неразделимости классов, алгоритм KDDA использует неявное проецирование векторов-признаков в пространство потенциально намного более высокой размерности (еще выше, чем пространство изображений), в котором классы могут оказаться линейно разделимыми (рис. 1).

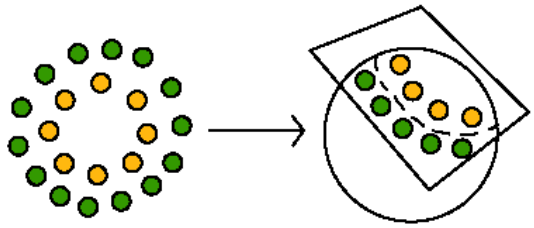


Рис 1: Пример разделения линейно неразделимых классов с помощью перехода к пространству более высокой размерности.

Проецирование осуществляется с помощью аппарата ядерных функций. Неявное проецирование с помощью ядерных функций не приводит к усложнению вычислений, что позволяет успешно использовать линейный классификатор для линейно неразделимых классов.

В качестве ядра в нашей работе использовалась радиальная базисная функция Гаусса:

$$k(z_1, z_2) = \exp\left(\frac{-\|z_1 - z_2\|^2}{\sigma^2}\right).$$

В полученном пространстве признаков считается, что объекты, принадлежащие различным классам линейно разделимы, и применяется D-LDA [2]. Такой выбор связан с тем, что применение классического LDA в задачах распознавания невозможно. Например, в нашей работе обучающая выборка состоит из изображений размерностью 40 × 40 = 1600 пикселей, значит размерность матриц межклассовой и внутриклассовой дисперсии (S_{BTW} и S_{WTH}) составляет 1600×1600. Кроме того, что работа непосредственно с матрицей S_{WTH} затруднительна из-за её размерности, обучающая выборка должна состоять из 1600 изображений для каждого класса, иначе матрица S_{WTH} будет вырожденной. Для решения данной проблемы в LDA использовано предварительное уменьшение размерности с помощью метода главных компонент. Таким образом, размер матрицы S_{WTH} уменьшается за счет удаления близких нулю собственных значений.

В алгоритме D-LDA применяется другой способ для решения данной проблемы. Основная идея алгоритма заключается в том, что в «нулевом» пространстве S_{WTH} может содержаться значимая информация для дискриминантного анализа, если проекция S_{BTW} не нулевая в данном направлении, и поэтому полезная информация будет утеряна. В отличие от классического LDA, при поиске матрицы для отображения пространства изображений на пространство признаков вначале диагонализируется матрица S_{BTW}, при этом уменьшается размер матрицы за счет удаления нулевых и близких к нулю собственных значений, которые не содержат полезной информации. Затем диагонализируется матрица S_{WTH}. Последовательность диагонализации матриц имеет значение, только если матрица S_{WTH} вырожденная, что в нашем случае имеет место быть. Матрица A для проецирования пространства изображения на пространство признаков в алгоритме D-LDA выбирается из следующего условия:

$$A = \arg\max_A \frac{\left|A^T S_{BTW} A\right|}{\left|A^T S_{BTW} A + A^T S_{WTH} A\right|}.$$

Пространство признаков строится на основе собственных векторов матрицы A на этапе обучения. На этапе тестирования для принятия решения, к какому классу относится вектор в пространстве признаков, используется метод ближайших соседей.

3.2 Метод опорных векторов

Применение метода опорных векторов (МОВ) [1] к задаче разделения анализируемых изображений на два класса заключается в поиске оптимальной гиперплоскости в признаковом пространстве, отделяющей один класс от другого. В качестве признаков служат значения пикселей изображения, представленные в виде n-мерного вектора. Оптимальной считается гиперплоскость, которая максимизирует ширину полосы между классами (рис. 2).

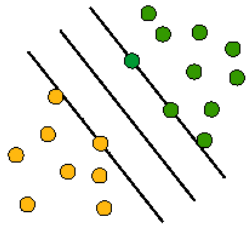


Рис 2: Пример разделяющей полосы классификатора на базе МОВ.

Разделяющая гиперплоскость определяется как линейная комбинация небольшого набора тренировочных векторов, называемых опорными векторами. Обозначим набор собственных векторов как $\{X_1, \ldots, X_m\}$, а соответствующие им коэффициенты линейной комбинации – $\{\alpha_1, \ldots, \alpha_m\}$. МОВ – это линейный классификатор, поэтому для разделения линейно неразделимых классов, применяется проецирование векторов-признаков в пространство потенциально намного более высокой размерности. Формально решающее правило

классификатора на базе MOB для входного изображения X может быть записано следующим образом:

$$f(X) = \mathrm{sgn}\left(\sum_{i=1}^{m} y_i \alpha_i k(X_i, X) + b \right),$$

где $k(\cdot,\cdot)$ – это ядерная функция, а b – смещение.

В качестве ядерной функции, как и для метода KDDA, использовалась радиальная базисная функция Гаусса.

4. МЕТОДИКА И РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ

Для обучения и проведения тестирования была составлена база, состоящая из 2770 изображений, содержащих фронтально расположенные лица «мужчин», «женщин» и «детей». На каждом изображении из этой базы были выделены лица алгоритмом SNoW. При этом уровень выделения составил 98,2%. Ложные выделения были отсеяны вручную, после чего к оставшимся изображениям была применена предобработка. Далее было отобрано по 400 лиц каждого класса. Эти лица были разделены на обучающий набор (по 300 лиц каждого класса) и тестовый набор (по 100 лиц). Были обучены два классификатора: один – с помощью алгоритма KDDA, и второй – на базе метода опорных векторов.

В табл. 1 и 2 представлены результаты распознавания для каждого класса при использовании алгоритмов классификации на базе KDDA и MOB соответственно.

Таблица 1: Распознавание на базе классификатора KDDA.

Решение / **Входное изображение**	Мужчины	Женщины	Дети
Мужчины	**88%**	12%	0%
Женщины	7%	**83%**	10%
Дети	7%	23%	**74%**

Таблица 2: Распознавание на базе классификатора MOB.

Решение / **Входное изображение**	Мужчины	Женщины	Дети
Мужчины	**94%**	5%	1%
Женщины	8%	**81%**	11%
Дети	1%	7%	**92%**

Классификатором KDDA было верно распознано 245 фрагментов из 300, таким образом, его средний уровень распознавания составил 81,6%. Классификатор MOB показал уровень распознавания 89% (267 верных фрагментов из 300). При этом оба алгоритма допускают больше ошибок при различении «детей» от «женщин».

Пример работы системы распознавания пола и возраста представлен на рис. 3.

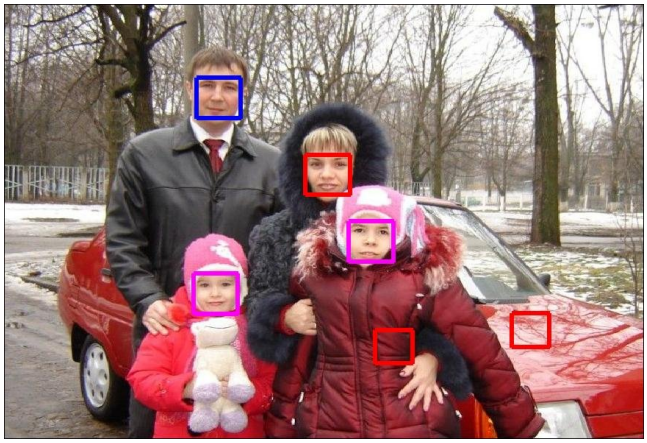


Рис 3: Распознавания пола и возраста людей на тестовом изображении.

Здесь фрагменты, распознанные как класс «мужчины», условно выделены синим цветом; «женщины» – красным; «дети» – фиолетовым. Из рис. 3 видно, что лица были распознаны верно, однако, было допущено два ложных выделения, которые были распознаны как класс «женщины».

5. ЗАКЛЮЧЕНИЕ

Итак, был представлен алгоритм распознавания пола и возраста по изображению лица. Для обучения классификатора использовалось два различных метода: линейный дискриминантный анализ и метод опорных векторов. Наибольшую эффективность продемонстрировал классификатор на базе MOB. Уровень распознавания в задаче разбиения изображений на три класса составил порядка 90%.

Дальнейшая работа будет направлена на увеличение уровня распознавания путем применения интеллектуальной предобработки, обучения алгоритма на собственных ошибках, увеличения обучающего набора изображений, оптимизации параметров классификаторов.

6. ССЫЛКИ

[1] Burges C. A Tutorial on Support Vector Machines for Pattern Recognition // Data Mining and Knowledge Discovery, V. 2, P. 121-167, 1998.

[2] Gao H., Davis J. Why direct LDA is not equivalent to LDA // Pattern Recognition Letters 39, № 5, P. 1002-1006, 2006.

[3] Kriegman D., Yang M.H., Ahuja N. Detecting faces in images: A survey // IEEE Transactions on Pattern Analysis and Machine Intelligence, V. 24, № 1, P. 34-58, 2002.

[4] Makinen E., Raisamo R. An experimental comparison of gender classification methods // Pattern Recognition Letters 29, № 10, P. 1544-1556, 2008.

[5] Nilsson M., Nordberg J., Claesson I. Face Detection Using Local SMQT Features and Split Up SNoW Classifier // Proceedings of IEEE Int. Conf. ICASSP, V. 2, P. 589-592, 2007.

[6] Потапов А.С. Распознавание образов и машинное восприятие: общий подход на основе принципа минимальной длины описания. - СПб.: Политехника. — 2007.

Обработка изображений в браузерных приложениях на потоковых графических процессорах

Андрей Сморкалов
Лаборатория систем мультимедиа
Марийский Государственный Технический Университет, Йошкар-Ола
asmorkalov@mail.ru

Аннотация

В статье рассматриваются подходы к организации обработки изображений в браузерных приложениях с использованием вычислительных возможностей графических потоковых процессоров. Идея применения потоковых процессоров в этом случае имеет большие перспективы, т.к. скорость выполнения javascript-кода значительно уступает скорости выполнения программ на таких языках, как C/C++, а пиковая производительность потоковых процессоров в десятки раз выше производительности центральных процессоров.

Ключевые слова: *Обработка изображений, Браузерные приложения, Потоковые процессоры, WebGL.*

1. ВВЕДЕНИЕ

Графические потоковые процессоры характеризуются большой степенью параллелизма, обладая сотнями и даже тысячами вычислительных ядер, но накладывают на выполняемые алгоритмы большое число ограничений (в основном, из-за изначальной ориентированности на задачи 3D-графики), которые делают невозможным прямой перенос алгоритмов обработки изображений для выполнения на потоковых процессорах. Среди этих ограничений отсутствие рекурсии, прямого доступа к памяти на запись, неопределенный результат чтения из результирующего изображения и др.

Между тем, превосходство потоковых над обычными процессорами в десятки раз по вычислительной мощности делает перспективным обработку двумерных изображений именно на потоковых процессорах [6].

Не менее перспективной является идея использования потоковых процессоров для обработки изображений в браузерных приложениях в связи с их ограниченным доступом к вычислительным возможностям центрального процессора. Прямое исполнение кода на центральном процессоре из браузерного приложения невозможно без использования специально разработанного дополнительного модуля для браузера (плагина). Механизм поддержки плагинов варьируется от браузера к браузеру, что вносит дополнительные трудности и требует разработки плагинов под каждый поддерживаемый браузер.

Актуальным является разработка программной системы для браузерных приложений, предлагающей удобную абстракцию для решения основных задач обработки изображений. Одним из основных достоинств такой системы будет то, что от программиста при решении задач обработки изображений не потребуется знания аппаратных особенностей потоковых процессоров и умения программировать их на низком уровне.

2. ОБЗОР СУЩЕСТВУЮЩИХ МЕТОДОВ

Долгое время обработку изображений в браузерных приложениях можно было реализовать с помощью двух основных подходов: с использованием Java-апплета и с использованием Flash-компонента. Так, например, для обработки изображений в java-апплетах широко используется библиотека ImageJ [1].

При использовании обоих подходов требуется установка специальных плагинов, что является во многих случаях нежелательным или невозможным (по соображениям безопасности, ввиду отсутствия плагина для нужной платформы или браузера, ввиду недостаточной квалификации пользователя для установки, а также по другим соображениям). Кроме того, в обоих случаях обработка изображений производится байт-кодом, выполняемым на виртуальной машине Flash или Java. Таким образом, скорость обработки изображений будет значительно уступать скорости программ на компилируемых языках, таких как C/C++, и тем более скорости обработки изображений на графических потоковых процессорах.

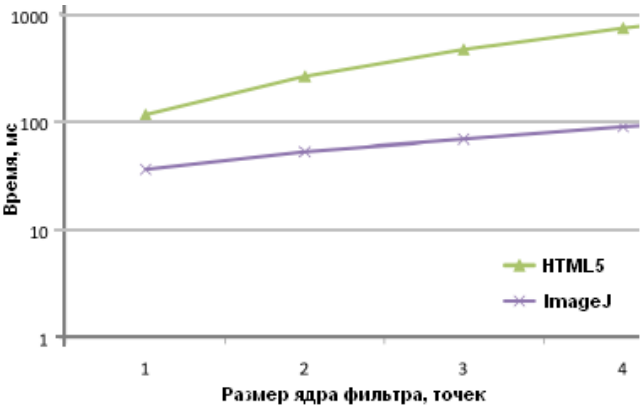


Рис 1. Сравнение производительности box-фильтра с помощью HTML5 canvas и ImageJ.

С появлением стандарта HTML5 стала возможна обработка изображений непосредственно на языке javascript с использованием элемента canvas [3]. Преимущество метода заключается в отсутствии необходимости установки каких-либо плагинов, однако

1. Программа обработки изображений выполняется на виртуальной машине javascript.

2. Использование многоядерности процессоров затруднено.

Эти факторы негативно отражаются на производительности подхода. По приводимому в [3] графику видно, что скорость описываемого подхода в несколько раз уступает скорости обработки изображений с помощью ImageJ (рис. 1).

В настоящее время все большее распространение получает технология PixelBender [2], реализованная в Adobe Flash начиная с версии 10.0. Эта технология позволяет в ограниченном объеме использовать возможности потоковых процессоров, разрабатывая программы для них на специальном языке, специфичном для Flash.

Существенные минусы PixelBender:

1. Технология реализована внутри Flash, а значит требует установки плагина и имеет все перечисленные проблемы технологий, требующих плагинов.

2. Специальный язык ограничивает доступ к вычислительным возможностям потоковых процессоров, ограничивает простор для оптимизации и т.д.

10 февраля 2011 года была опубликована спецификация открытого стандарта WebGL [5] – реализации OpenGL ES 2.0 для использования в браузерных приложениях. WebGL дает возможность разрабатывать OpenGL-программы непосредственно на javascript, включая использование вершинных и пиксельных шейдеров. При этом становится доступной вся функциональность потоковых процессоров.

Для использования WebGL не требуется установки дополнительных модулей, поддержка технологии реализуется непосредственно в браузере. Перечисленные качества делают технологию перспективной для организации обработки изображений в браузерных приложениях.

3. МАТЕМАТИЧЕСКАЯ МОДЕЛЬ

Составим математическую модель обработки изображений, опираясь на особенности и возможности WebGL. Будем рассматривать изображение в цветовой модели RGBA:

$U(x, y) = \{f_R(x, y), f_G(x, y), f_B(x, y), f_A(x, y)\}$, (1)

где $f_R(x, y)$, $f_G(x, y)$, $f_B(x, y)$, $f_A(x, y)$ — это дискретные функции, заданные табличным методом, x, y – координаты внутри изображения. $f_R(x, y)$ представляет красный канал изображения, $f_G(x, y)$ - зеленый, $f_B(x, y)$ - синий, $f_A(x, y)$ - альфа-канал. Значения этих функций лежат в диапазоне [0, 1]. $x \in [0, w-1)$, $y \in [0, h-1)$, где w и h – ширина и высота изображения.

Результатом преобразования G изображения A на основе изображения B будем называть R = G(A, B, x, y), (2) где A — это исходное изображение, B – растеризуемая фигура, G — преобразующая функция.

Цветовой маской будем называть множество $M = \{M_R, M_G, M_B, M_A\}$ (3), где M_R, M_G, M_B, M_A могут принимать значения из множества {0, 1}. Результатом преобразования G изображения A на основе изображения B с учетом цветовой маски M будем называть $R_q = G_q(A, B, M, x, y) = G_q(A, B, x, y) * M_q + A * (1 – M_q)$ (4), где q – любой из цветовых каналов.

Геометрическая фигура $S = \{V, F, \{r, g, b, a\}\}$ (5) задается множеством двумерных векторов вершин V и множеством индексов вершин F, а также цветом фигуры {r, g, b, a}. Растеризация представляет собой преобразование, результатом которого является изображение $U(x, y) = G_R(G_P(S, M_P))$ (6), где G_R – растеризующее преобразование, M_P – матрица проецирования 4x4, G_P — проецирующее преобразование.

В результате преобразования G_P получается спроецированная фигура $S_P = G_P(S, M_P) = \{P, F, \{r, g, b, a\}\}$. (7)

Спроецированная фигура S_P задается множеством двумерных векторов спроецированных вершин $P = \{P_1,\ldots, P_N\}$, множеством индексов вершин $F = \{F_1, ..F_{3 * M}\}$, а также цветом фигуры {r, g, b, a}. Отметим, что $P_i = V_i * M_P$ (8) для всех $i \in \{1..N\}$. Три идущих подряд индекса вершин задают спроецированный полигон (треугольник).

$G_R(x, y) = \{\{r, g, b, a\}$, если существует тройка $(F_{3 * k + 1}$, F3 * k + 2, $F_{3 * k + 3})$, где k – целое число от 1 до M / 3, такая, что (x, y) принадлежит треугольнику, образованному вершинами P_{i1}, P_{i2}, P_{i3}, где $i1 = F_{3 * k + 1}$, $i2 = F_{3 * k + 2}$, $i3 = F_{3 * k + 3}$, {0.0, 0.0, 0.0, 0.0} в противном случае} (9).

4. МОДЕЛЬ ФИЛЬТРАЦИИ ИЗОБРАЖЕНИЙ

На основе анализа архитектуры и особенностей потоковых процессоров [4] и возможностей WebGL была разработана следующая модель фильтрации (преобразования) изображений. В основе модели лежат четыре основные понятия: Texture, Drawing Target, Filter и Filter Sequence. Texture - это изображение в форме (1), хранимое в памяти графического потокового процессора. Drawing Target - это объект, который определяет изображение-результат и цветовую маску (3).

Filter - задает преобразование изображения (2) и в общем случае представляет собой функцию с набором предопределенных и пользовательских параметров, возвращающую цвет точки для заданных координат. Функция, например, может быть задана на GLSL-подобном языке, расширенном дополнительными функциями для обработки изображений. Параметры функции строго типизированы, имеют уникальные имена и определяются описанием в форме XML. В качестве растеризуемой фигуры B используется триангулированный прямоугольник с размером равным размеру результирующего изображения R.

```
<filter>
 <function name="GetColor">
  <arguments>
   <image>buffer</image>
   <image>oldBuffer</image>
  </arguments>
  <body>
   vec4 newColor = (buffer.getOfsPixel(-1.0, -1.0) +
   buffer.getOfsPixel(-1.0,0.0) + buffer.getOfsPixel(-1.0,1.0) +
   buffer.getOfsPixel(0.0, -1.0) + buffer.getOfsPixel(0.0, 1.0) +
   buffer.getOfsPixel(1.0, -1.0) + buffer.getOfsPixel(1.0, 0.0) +
   buffer.getOfsPixel(1.0, 1.0)) / 4.0;
   vec4 oldColor = oldBuffer.getCurrentPixel();
   float clr = newColor.r - oldColor.r;
   if (clr less 0.32)
    clr = 0.32;
   if (clr more 1.0)
    clr = mod(clr, 1.0);
   return vec4(clr, clr, clr, 1.0);
  </body>
 </function>
</filter>
```

Листинг 1: Пример фильтра для расчета кадра дождевой поверхности

В листинге 1 приведен пример фильтра, заданного в предлагаемой форме. Функция GetColor должна возвратить цвет текущего обрабатываемого пикселя, аргументами функции являются два изображения buffer и oldBuffer. Функция getCurrentPixel возвращает цвет текущего пикселя в

заданном изображении, а getOfsPixel цвет пикселя со смещением относительно текущего. Типы используемых вне декларативного объявления переменных соответствуют типам данных языка GLSL.

```
<filter>
  <function name="GetColor">
  <arguments>
    <image>background</image>
    <image>rain</image>
  </arguments>
  <body>
      vec4 newColor = background.getCurrentPixel();
      vec4 oldColor = rain.getPixel(%x, %height - %y) * 0.5;
      vec4 oldColor2 = oldColor + 0.5;
            return vec4(newColor.r * oldColor2.r + oldColor.r,
newColor.g * oldColor2.g + oldColor.g, newColor.b *
oldColor2.b + oldColor.b, 1.0);
  </body>
  </function>
</filter>
```

Листинг 2: Пример фильтра для применения рассчитанной дождевой поверхности на произвольное изображение.

В листинге 2 приведен еще один пример фильтра. В этом фильтре осуществляется прямой доступ к пикселям изображения с помощью функции getPixel, а координаты рассчитываются согласно значениям предопределенных переменных %x и %y (координаты текущего пикселя), %width и %height (размеры результирующего изображения).

В основе модели фильтрации лежит условие: изображение-результат и изображения-параметры одного и того же фильтра не могут совпадать, т.е. $F(X) \neq X$. (10)

Это условие заложено в модель в целях обеспечения корректности выполнения преобразований на потоковых процессорах, т.е. для организации независимой параллельной обработки фрагментов изображения.

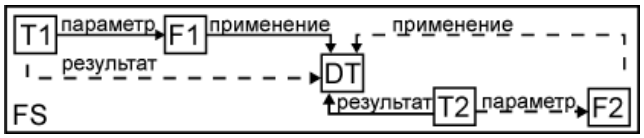


Рис.2: Взаимоотношения объектов в системе на пример filter sequence FS типа "пинг-понг".

Filter Sequence - задает последовательность нескольких фильтров с их параметрами, в общем случае позволяя организовать конвейер обработки, на основе которого можно реализовать сложное преобразование для обработки изображений. Таким образом, Filter Sequence представляет собой композицию преобразований изображений и в общем случае выражается формулой $G(X) = G_1(G_2(..G_i(X)))$, где G(X) - преобразование, задающее Filter Sequence, а G_1, G_2... G_i - составляющие его преобразования-фильтры.

На примере в рис. 2. текстура T1 является параметром фильтра F1, который применяется к drawing target DT. Текстура T2 является результатом применения фильтра F1 к DT. В тоже время текстура T2 является параметром фильтра F2, который применяется к DT. Текстура T1 является результатом применения фильтра F2 к DT. Сначала применяется F1, затем F2, исходное изображение берется из T1, результат оказывается там же. Сплошными линиями показаны связи, существующие во время применения первого фильтра, пунктирными - второго.

5. МОДЕЛЬ ПОДДЕРЖКИ ОПЕРАЦИЙ РИСОВАНИЯ

Операции рисования являются отдельным классом задач обработки изображений как при реализации на центральном процессоре, так и на потоковых. Они могут использоваться для достижения той или иной функциональности графических редакторов, например для поддержки графического комментирования материала на виртуальной доске в образовательной 3D-среде. Данные для осуществления операций могут поступать как в виде событий от клавиатуры или мыши, так и в виде данных от сервера (воспроизведение ранее записанного занятия).

Способы реализации поддержки рисования в системах обработки изображения на центральном процессоре детально проработаны. Однако, при использовании потоковых процессоров классические алгоритмы (например, алгоритм Брезентхема) оказываются неприменимы из-за архитектурных ограничений, что требует разработки методов поддержки операций рисования специально для использования с применением потоковых процессоров.

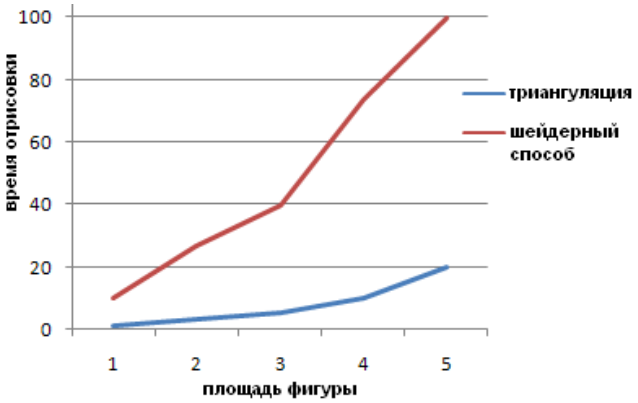


Рис. 3: Замер производительности поддержки операций рисования с помощью растеризации по аналитическому описанию и с помощью триангуляции.

Теоретически, геометрическая фигура может быть отрисована непосредственно по аналитическому описанию с применением шейдерных программ, что свело бы эту задачу к задачам фильтрации. Однако, эффективность такого подхода невелика, т.к. фактическая площадь отрисовываемой фигуры может быть на порядки меньше, чем площадь растеризовываемого триангулированного прямоугольника. Даже если в качестве данного прямоугольника брать ограничивающий объем фигуры (bounding box), неэффективность подхода остается значительной (см. рис 3).

Поэтому для более эффективной реализации необходимо триангулировать геометрические фигуры, т.е. представлять их в форме выражения (5). Триангуляцию каждой конкретной фигуры предлагается реализовать вычислением множества вершин по специфичным для выбранной фигуры формулам, т.к. триангуляция в общем виде избыточно сложна.

5.1 Программная архитектура

Программная архитектура (рис. 4) предполагает поддержку некоторой абстрактной палитры инструментов, которая может быть расширена новыми инструментами для реализации поддержки требуемых операций. Инструменты самостоятельно выполняют триангуляцию фигур и передают

на растеризацию триангулированную фигуру с установленными параметрами цвета и прозрачности.

Для каждого изображения с поддержкой операций рисования создается объект палитры инструментов, а также Drawing Target. Он содержит в себе основную текстуру, временную текстуру и буфер глубины. Размер основной и временной текстуры совпадают. Реализация текущего инструмента может запросить сохранение во временную текстуру изображения из основной текстуры перед очередным этапом работы. До окончания этапа содержимое временной текстуры может быть скопировано в основную, а поверх растеризована актуальная рисуемая фигура. Под этапом понимается полный цикл рисования одной фигуры (линии, прямоугольника и т.д).

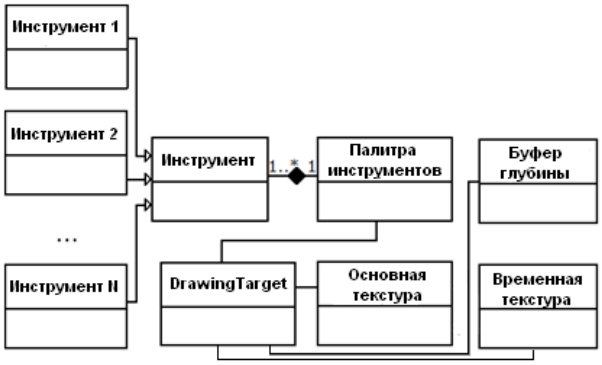


Рис. 4: Архитектура инструмента

Объект палитры инструментов содержит в себе коллекцию инструментов и хранит информацию о том, какой из инструментов активен в текущий момент. Этот объект получает уведомления о событиях, инициирующих процесс отрисовки (например, от мыши, от управляющего сервера и т.д.), и передает их текущему инструменту. Все инструменты удовлетворяют одному и тому же программному интерфейсу, так что палитра инструментов взаимодействует с любым конкретным инструментом без учета того, как этот инструмент устроен внутри.

5.2 Поддержка прозрачности

Для поддержки прозрачности предлагается использовать композицию из двух преобразований. Первый проход выражается преобразованием G(C, D, x, y) = min(C(x, y) + D(x, y), 1.0) (11) , где C и D представляют собой функции одного и того же канала изображения. Это преобразование применяется только к альфа-каналу с помощью маски (3). Оно соответствует поддерживаемому аппаратно преобразованию с параметрами source factor = GL_ONE, destination factor = GL_ONE, blend mode = GL_FUNC_ADD.

Второй проход выражается преобразованием G(A, B, x, y)q = min(A(x, y)q * B(x, y)A + (1.0 - B(x, y)A) * B(x, y)q), 1.0) (12). Преобразование (12) применяется только к RGB-каналам изображения. Оно соответствует поддерживаемому аппаратно преобразованию с параметрами source factor = GL_SRC_ALPHA, destination factor = GL_ONE_MINUS_SRC_ALPHA, blend mode = GL_FUNC_ADD. Полученное преобразование удовлетворяет предъявляемым к прозрачности требованиям:

1. Смешивание RGB-каналов должно производиться с учетом значения альфа-канала растеризуемой фигуры.
2. При растеризации фигуры на абсолютно непрозрачную точку изображения, значение альфа-канала не должно измениться.
3. При растеризации фигуры на абсолютно прозрачную точку изображения, значение альфа-канала должно увеличиться.

6. ВЫВОДЫ

Был предложен перспективный подход к обработке изображений в браузерных приложениях, который позволяет осуществлять довольно сложные преобразования в режиме реального времени. Подход успешно апробирован на ряде реальных примеров (рис. 5) и в ходе экспериментальных разработок браузерного плеера записанных виртуальных 3D-лекций (с поддержкой обработки изображений и операций рисования на виртуальной доске).

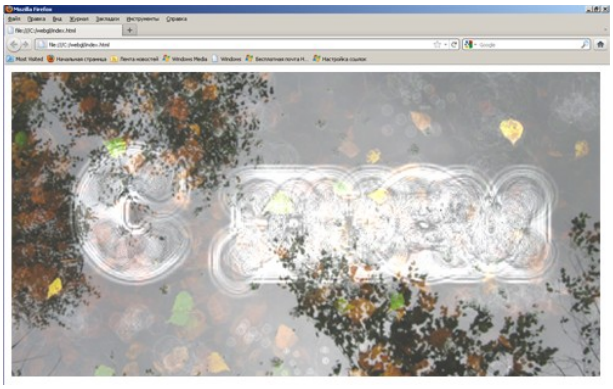


Рис. 5: Пример фильтра водной поверхности, работающего в реальном времени на потоковых процессорах в браузерном приложении в Firefox 4.

Метод показывает высокую производительность. Так box-фильтр и фрактал Мандельброта рассчитываются предлагаемым методом в ~15-110 раз быстрее, чем при использовании подхода [3], в зависимости от выбранного сочетания центрального и графического потокового процессоров.

7. ССЫЛКИ

[1] Abramoff, M.D., Magelhaes, P.J., Ram, S.J. *"Image Processing with ImageJ". Biophotonics International, volume 11, issue 7, pp. 36-42, 2004.*

[2] *Adobe Pixel Bender Guide.* http://www.adobe.com/content/dam/Adobe/en/devnet/pixelbender/pdfs/pixelbender_guide.pdf

[3] Kai Uwe Barthel, Karsten Schulz, *ImageJ in the web? - Image processing in the browser using HTML5. In proceedings of ImageJ Conference 2010.*

[4] Kayvon Fatahalian, Mike Houston. *A closer look at GPUs. Communications of the ACM, October 2008, Vol. 51, NO. 10.*

[5] The Khronos Group. *WebGL 1.0 Specification.* https://www.khronos.org/registry/webgl/specs/1.0/

[6] А.Ю. Сморкалов, М.Н. Морозов. Поддержка дискретных вейвлет-преобразований для компрессии в системе обработки изображений на потоковых графических процессорах. Материалы всероссийской научно-практической конференции "Информационные технологии в профессиональной деятельности и научной работе". - Йошкар-Ола: МарГТУ, 2010.- С. 91-96.

Метод Текстовой Стеганографии на Основе Модификации Цветовых Координат Символов

Надежда Урбанович, Владимир Пласковицкий
Факультет издательского дела и полиграфии
Белорусский государственный технологический университет, Минск, Беларусь
nadya_ur@rambler.ru

Аннотация

В статье рассматриваются особенности предлагаемого авторами метода компьютерной текстовой стеганографии на основе изменения цветовых параметров символов текста. Метод может применяться, в том числе, для защиты прав интеллектуальной собственности. Описывается программное средство для изучения и анализа эффективности метода.

Ключевые слова: *текстовая стеганография, цвет символа, защита авторского права.*

1. ВВЕДЕНИЕ

Известны два основных направления решения задачи защиты информации от несанкционированного использования: криптография и стеганография.

Целью криптографии является скрытие содержимого сообщений за счет их шифрования. В отличие от этого, в стеганографии скрывается сам факт существования тайного сообщения. Дальнейшее изложение касается компьютерной стеганографии, предусматривающей размещение информации в текстовых документах.

Слово «стеганография» имеет греческие корни и буквально означает «тайнопись». С помощью различных методов можно тайно (с различным уровнем тайности) передавать сообщение, «осажденное» в документ (файл).

Кратко охарактеризуем основные понятия, относящиеся к предметной области: сообщение, контейнер и ключ. Термин «контейнер» употребляется в отечественной литературе большинством авторов, поскольку является дословным переводом устоявшегося английского термина «container», обозначающего несекретную информацию, которую используют для сокрытия сообщений. По сути же, контейнер в стеганографической системе является ни чем иным как носителем сокрытой информации, поэтому вполне возможно использование и такого термина. В некоторых источниках термин контейнер также заменяют названием «стего», который также является производным от английского сокращения «stego» (полное название «stegano»).

Контейнером (носителем) называют несекретные данные, которые используют для сокрытия сообщений.

В компьютерной стеганографии в качестве контейнеров могут быть использованы различные оцифрованные данные: не только текстовые электронные документы, но и растровые графические изображения, звук, видео и др.

Сообщением (стегосообщением) называют секретную информацию, наличие которой в контейнере необходимо скрыть.

Ключом называют секретную информацию, известную только законному пользователю, которая и определяет конкретный вид алгоритма сокрытия [1].

Кроме скрытой передачи сообщений, стеганографические методы являются одним из самых перспективных инструментов для аутентификации и маркировки авторской продукции с целью защиты авторских прав на цифровые объекты от пиратского копирования. В качестве стегосообщения можно использовать данные об авторе, дату и место создания произведения, номера документов, подтверждающих авторство, дату приоритета и т.п. Такие специальные сведения могут рассматриваться в качестве доказательств при рассмотрении споров об авторстве или для доказательства нелегального копирования.

Многие существующие методы текстовой стеганографии [2] недостаточно эффективно скрывают сообщения, факт наличия секретной информации (ключ) является очевидным либо почти очевидным. Этот вывод сделан авторами данной статьи на основе сравнительного анализа известных методов текстовой стеганографии с помощью специально разработанного программного средства. Таким образом, актуальной является задача разработки новых методов, повышающих устойчивость к атакам, т. е. снижающим вероятность извлечения сообщения из контейнера.

2. СУЩНОСТЬ МЕТОДА МОДИФИКАЦИИ ЦВЕТОВЫХ ПАРАМЕТРОВ СИМВОЛА

Цвет символа в текстовом процессоре Microsoft Word представлен в цветовой модели RGB. Незначительное изменение цвета символа не воспринимается человеческим глазом. Используя данную физиологическую особенность, можно незаметно производить встраивание информации.

Подобный подход используется в случаях, когда контейнером является изображение (графический файл). Известный метод имеет название Least Significant Bit (LSB). При его реализации встраивание производится в последние 1–2 бита цвета пикселя изображения. При адаптации вышеупомянутого алгоритма к тексту встраивание можно производить в последние 3–5 битов цвета символа. Увеличение числа используемых бит цвета в тексте, по сравнению с графикой, происходит из-за того, что изображение, как правило, содержит градации и переходы от одного цвета к другому. Текст — монотонен и выполняется в большинстве случаев одним цветом, поэтому становится возможным увеличение используемого для встраивания цветового диапазона.

Например, необходимо внедрить секретное сообщение «101» в текст-контейнер «А», используя текстовый процессор MS Office Word 2007. В данном процессоре цвет символов представлен в системе RGB (цвет представлен тремя стандартными цветами: red, green, blue) с 8 битами на канал (каждый из 3 стандартных цветов представлен 8 битами — числом от 0 до 255). Цвет текста-контейнера «А» — черный: данный цвет представлен в MS Office Word 2007 как (00000000, 00000000, 00000000). Встраивание

секретного сообщения будем производить в младшие биты цвета символов. В итоге будет получен цвет: (00000001, 00000000, 00000001).

Результат внедрения сообщения «101» в текст-контейнер (символ «A»), при использовании текстового процессора MS Office Word 2007, показан на рисунке 1 (показано увеличенное графическое изображение). Как видно, визуально оба символа одинаковы.

Рис. 1: Применение метода изменения цвета символов

а) стандартное графическое изображение символа; б) модифицированный символ (со встроенным секретным сообщением «101»)

3. ОПИСАНИЕ ПРОГРАММНОГО СРЕДСТВА НА ОСНОВЕ ИЗМЕНЕНИЯ ЦВЕТОВЫХ КООРДИНАТ СИМВОЛОВ ТЕКСТА

Описанный выше метод реализован в программном продукте, характеризующемся следующими техническими параметрами.

Язык разработки: C# (Framework 2.0 + библиотеки Microsoft.Vbe.Interop.dll, Microsoft.Office.Interop.Word.dll, office.dll).

Требования к ОС: Windows + Framework 2.0 (XP, Vista, Seven).

Поддерживаемые форматы документов для скрытия: любые, который можно открыть с помощью Office Word и способные хранить цвет символов: Word 93-2010 (*.doc, *.docx), *.rtf (межплатформенный формат хранения размеченных текстовых документов) , *.odt *(открытый формат документов для офисных приложений*). Поддержка данных форматов должна быть доступна из установленного на компьютере пользователя приложения MS Word.

Общий принцип работы приложения проиллюстрирован на диаграмме деятельности (рис. 2).

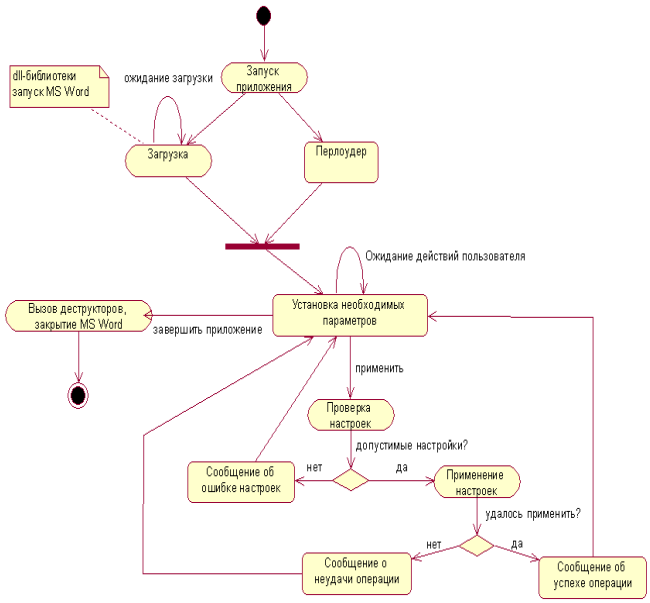


Рис. 2: Диаграмма деятельности программного средства.

Главное окно программного средства имеет вид, представленный на рис. 3.

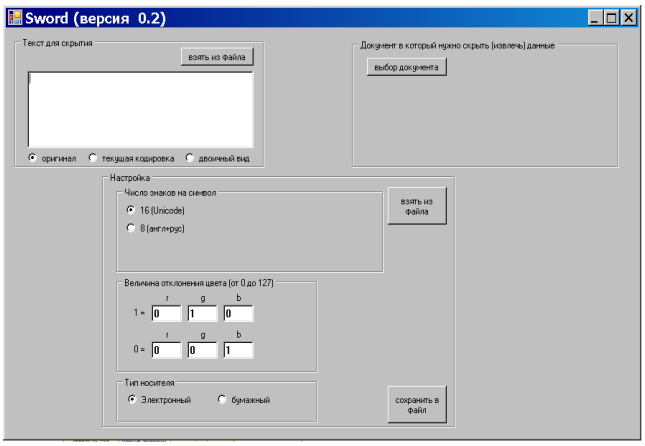


Рис. 3: Общий вид программы

Установка необходимых параметров осуществляется с помощью трех блоков, в которых задается: стегосообщение (в случае извлечения заполнять не надо), контейнер и настройки, на основе которых происходит скрытие/извлечение сообщения (ключ).

Последовательность скрытия текста:

- перевод текста в двоичный вид (на основе выбранной кодировки);
- проверка того, достаточно ли в документе-контейнере символов для скрытия стегосообщения. Если последнее состоит из K символов и при этом выбрана кодировка, в которой один символ представлен Q знаками, то в документе должно быть не меньше K*Q*2 знаков. Умножение на 2 предусматривает то, что каждому скрываемому символу нужна пара, по которой будет определяться отклонение его цвета;
- если проверка прошла успешно, составляется список символов, в которых будет производиться скрытие. Эта процедура необходима для того, чтобы модифицированные символы располагались через случайный промежуток. Поэтому нужно убедиться, что алгоритм, задающий этот шаг, укладывается в границы документа. Правило расстановки следующее: анализируется максимальное расстояние, на котором могут располагаться скрываемые символы. Например, это расстояние равно S. Для каждого символа генерируется шаг в диапазоне от 2 до (2*S-2), т. е. иногда шаг будет превышать это значение, иногда наоборот - будет меньше. Затем происходит составление списка. Если в момент составления происходит отклонение в большую от S сторону, то значение S уменьшается на единицу и процедура повторяется. В худшем случае S будет равняться 2, т. е. скрытые символы будут идти с плотностью через один;
- если список составлен успешно, происходит осаждение символов стегосообщения. Время встраивания пропорционально числу скрываемых

символов и практически не зависит от объема документа-контейнера;

- чтобы просмотреть позиции, в которых скрыты символы, необходимо воспользоваться кнопкой «Показать», а затем отметить эти символы (кнопка «Отметить»). После этого соответствующие символы будут помечены отдельным (например, синим маркером; рис. 4);

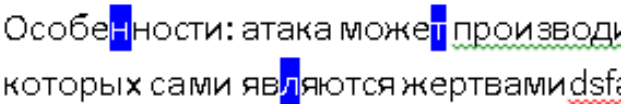


Рис. 4: Вид документа с выделением модифицированных символов

- перед сохранением либо изменением документа необходимо очистить эти метки (кнопка «Очистить»);
- параметры, которые используются для скрытия текста, можно сохранять в файлы для последующего считывания. Эти файлы являются обычными текстовыми документами, в которых хранятся настройки, структурированные специальным способом. Для того чтобы отличить файлы с такими настройками от других текстовых документов, первые сохраняются с расширением *.sword.

Последовательность извлечения текста:

- выбор документа-контейнера, содержащего стегосообщение;
- настройка параметров (ключа), на основе которых он был скрыт (если были сохранены, можно загрузить из файла);
- запуск операции извлечения (кнопка «Извлечь»);
- после извлечения текст будет помещен в соответствующее поле. Время, затраченное на извлечение, пропорционально числу символов в документе и практически не зависит о числа скрытых символов. Скорость извлечения составляет приблизительно 50 знаков в секунду. Во время дешифровки происходит уведомление пользователя о текущем состоянии выполнения с помощью специального текстового поля.

Стегосообщение можно вводить с клавиатуры, вставлять из буфера, открывать из файла. При выборе необходимого документа предоставляется три фильтра, а также возможность выбора любого файла (см. рис. 5).

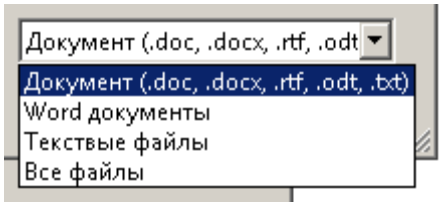


Рис. 5: Возможные типы документов

Во время работы с текстом можно просматривать его в двоичной системе (рис.6). Редактировать текст в этом режиме нельзя.

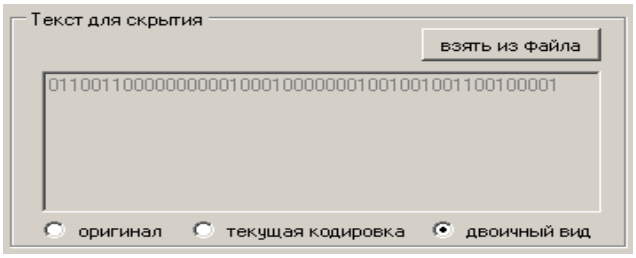


Рис. 6: Стегосообщение, представленной в двоичном виде

По умолчанию выбранный документ не отображается на экране, чтобы просмотреть его, необходимо нажать на кнопку «Показать». После открытия документа с его содержимым можно производить любые действия. При внедрении исследуется именно текущее содержание документа, а не то, которое было на момент открытия.

Если документ будет закрыт пользователем не из приложения MS Word, то он автоматически закроется и в программе. Он также будет закрыт, если в программе выбрать другой документ, поэтому при необходимости его надо предварительно сохранить.

Алгоритм модификации цветовых координат символов текста состоит в следующем. В настройках задается отклонение от основного цвета для символов «0» и «1» в системе RGB. В процессе скрытия берутся два идущих подряд символа: первый — образец, второй — тот, в котором будет скрыт текст. Если в настройках запрещено прятать текст в пробельных участках (тип носителя — «бумажный»), тогда сдвиг осуществляется до тех пор, пока оба символа будут не пробельными.

Цвет символа, в котором будет производиться скрытие, формируется исходя из цвета символа-образца и заданного в настройках смещения. По умолчанию это смещение добавляется к основному цвету. Если при этом значение выходит за допустимый диапазон — отнимается. Например, если цвет образца задан как (0,200,100), а смещение задано как (100,60,50), то результирующим будет цветовая координата (100,140,150). Чтобы избежать ситуации, когда значение может выйти за оба диапазона, в полях для отклонения не рекомендуется задавать значения больше 128.

4. ЗАКЛЮЧЕНИЕ

Описанный метод и программное средство используются в настоящее время для изучения возможности размещения в текстовых документах информации, предусматривающей защиту прав интеллектуальной собственности. В ходе достаточно объемных экспериментов установлено, что модификация до 4-х младших символов цветовой координата каждого канала (RGB) в 100% случаев остается незамеченной пользователем, которому не известен факт осаждения в документе невидимой информации.

5. БЛАГОДАРНОСТИ

Работа была выполнена при частичной поддержке гранта ГБ 11-025.

6. ССЫЛКИ

[1] Конахович Г. Ф., Пузыренко А.Ю. *Компьютерная*

стеганография. Теория и практика. – К.: «МК-Пресс», 2006. – 288 с.

[2] Ярмолик В. Н., Портянко С. С., Ярмолик С.В. *Криптография, стеганография и охрана авторского права – Минск: Изд.центр БГУ, 2007. – 240с.*

Об авторах

Надежда Урбанович — студентка БГТУ. Ее адрес: nadya_ur@rambler.ru.
Владимир Пласковицкий — студент БГТУ. Его адрес: vaplas20@gmail.com.

Реализация проекции сферического зеркала

Виноградов Владислав Игоревич, ННГУ им. Лобачевского, факультет ВМК. vlad.vinogradov47@gmail.com

Любов Дмитрий Викторович, ННГУ им. Лобачевского, факультет ВМК. dmitriy.lyubov@gmail.com

Носов Сергей Николаевич, ННГУ им. Лобачевского, факультет ВМК. sergei.nosov@gmail.com

Майоров Алексей Юрьевич, ННГУ им. Лобачевского, факультет ВМК. mayorov.alexey@gmail.com

АННОТАЦИЯ

В данной статье рассматривается вопрос адаптации медиа-контента, созданного для цифровых планетариев, под полнокупольные системы, использующие сферическое зеркало. Описывается программный комплекс, предоставляющий возможность калибровки проекционной аппаратуры и воспроизведения видео на сферическом экране.

Ключевые слова: цифровые планетарии, полнокупольные системы, сферический экран, fisheye-проекция, сферическое зеркало.

1 ВВЕДЕНИЕ

В настоящее время все большую популярность набирают цифровые полнокупольные системы. Это системы, в которых изображение проецируется на экран в форме полусферы. Изначально в качестве подобных экранов использовались купола планетариев. Сегодня широко доступны портативные полнокупольные системы, которые могут быть установлены в небольших помещениях. Такие системы чаще всего представляют собой надувной купол небольшого диаметра (от 3 до 8 метров), один проектор и специальное сферическое зеркало

Рис. 1 Проектор и сферическое зеркало

Так же на данный момент создано достаточное количество полнокупольного медиа-контента, который в основном состоит из научных и научно-популярных фильмов и видео-роликов. Так как все эти материалы предназначены для показа на сферическом экране, то при их съемке используется специальная fisheye - камера, обеспечивающая обзор 360°x180°. Получившиеся при этом кадры имеют fisheye - проекцию (Рис. 2). Данные кадры можно напрямую без конвертирования использовать при показе на сферических экранах с многопроекторными или fisheye-проекционными системами. Однако они не подходят для систем, использующих сферическое зеркало из-за отличий в проекции. Ввиду этого остро стоит задача адаптации имеющегося контента в fisheye-проекции к показу на малом куполе с использованием сферического зеркала (Рис. 2).

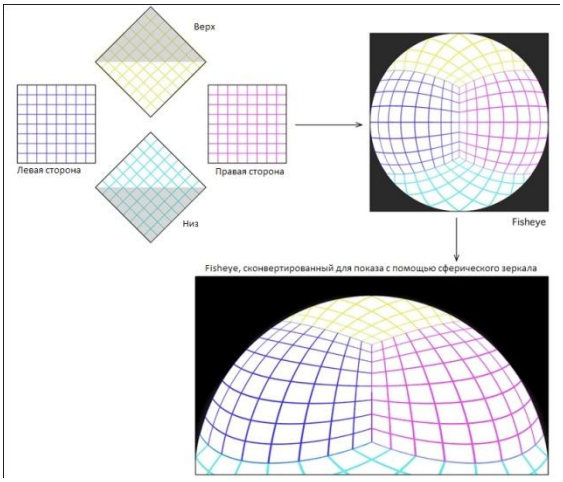


Рис. 2 Fisheye - проекция и конвертированный для сферического зеркала вариант

Также важной задачей является обеспечение возможности показа на сферическом экране стандартного видео, предназначенного для плоского экрана.

2 ПОСТАНОВКА ЗАДАЧИ

Конечной целью является создание программного комплекса, позволяющего решать следующие задачи:

1. калибровка проекционной аппаратуры полнокупольной системы (купол - сферическое зеркало - проектор);
2. воспроизведение видео, изначально находящегося в fisheye-формате.
3. Воспроизведение видео, предназначенного для показа на плоских экранах.

При создании программы калибровки необходимо выделить множество параметров, однозначно описывающих конфигурацию аппаратуры. Также необходимо определиться

с форматом, в котором будут храниться и использоваться результаты работы программы.

Для решения второй задачи – воспроизведения видео на куполе – было решено дополнить необходимой функциональностью один из существующих видеоплееров. В качестве такого плеера был выбран медиаплеер VLC, имеющий открытый исходный код. Его преимущество заключается в возможности расширения функционала за счет плагинов. Следовательно, необходимо реализовать в виде плагина видеофильтр для преобразования fisheye-видео. Данный плагин должен загружать информацию, полученную от программы калибровки, и осуществлять необходимое преобразование исходного изображения.

Для решения последней задачи медиа-контент, предназначенный для плоского экрана, необходимо перевести в проекцию fisheye. Следовательно необходимо разработать алгоритм, позволяющий производить подобное преобразование без потери качества и искажения исходного изображения.

3 РЕАЛИЗАЦИЯ ПРЕОБРАЗОВАНИЯ ИЗОБРАЖЕНИЯ

3.1 Моделирование полнокупольной системы

Первым делом обозначим множество параметров, которые однозначно определяют состояние конкретной полнокупольной системы. Выделим подобные множества для каждого из элементов системы. Начнем с купола:

- Dr Радиус купола. Является главной характеристикой купола.
- Dp = (0, 0, 0) - Центр купола. Для удобства расположим начало координат в центре купола.
- Da Угол наклона купола. Купол может располагаться не только параллельно полу, но и под наклоном вплоть до 90 градусов (вертикальный купол).

Далее рассмотрим параметры второй существенной части системы - сферического зеркала:

- Mr Радиус зеркала.
- Mp Положение центра зеркала в системе координат купола.

Теперь рассмотрим параметры проектора:

- Pp Положение проектора в системе координат купола.
- euler вектор углов Эйлера, отражающих возможные наклоны проектора.
- ThrowRatio Отношение расстояние от проектора до экрана к ширине получаемого изображения.

И наконец рассмотрим параметры входного и выходного изображений:

- angle Угол поворота исходного изображения вокруг своего центра.
- width Требуемая ширина выходного изображения.
- height Требуемая высота выходного изображения.
- SF Скалирующий множитель.
- mirrored Зеркальность относительно вертикальной оси.

Упрощенная схема полнокупольной системы представлена на рис. 1.

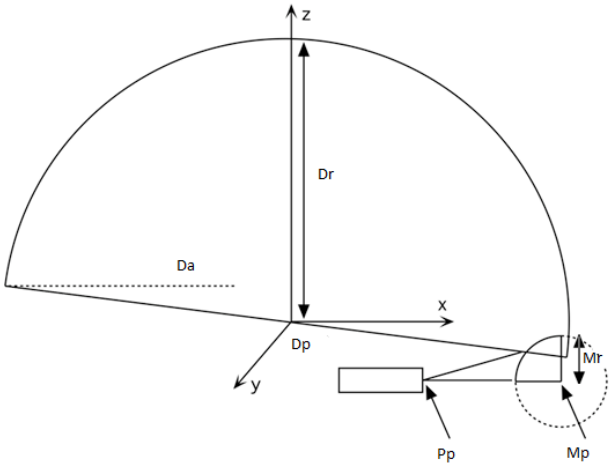


Рис. 3 Схема полнокупольной системы

3.2 Алгоритм преобразования fishye-кадра к кадру, использующему проекцию сферического зеркала

После того, как мы определили все необходимые для вычислений параметры, перейдем к описанию алгоритма преобразования. За основу возьмем известный метод трассировки лучей и проследим путь луча от проектора до поверхности купола. Функция генерации луча, отвечающего за проецируемый пиксель (i, j) может быть описана следующим псевдо-кодом:

```
GenerateRay (Ray ray, Vec3 view, Vec3 up,
             Vec3 side, i, j)
{
  Real aspect = width/height;
  Vec2 scale = {1/ThrowRatio, aspect/ThrowRatio}
  Vec2 screen = { i/height * scale.x;
                  (j/width - 0.5) * scale.y }
  ray.Direction = view + side * screen.x +
                  up * screen.y
  ray.Origin = Pp
}
```

Листинг 1 Генерация луча

Здесь вектора view, up, side задают положение проектора и вычисляются с использованием углов Эйлера проектора.

Следующим шагом после генерации луча является его трассировка. Для каждого луча нужно найти точку его пересечения с зеркалом, а затем вычислить отраженный луч и найти точку его пересечения с куполом - Q. Точка пересечения луча и сферы может быть найдена из решения квадратного уравнения.

Далее необходимо по координатам точки на куполе восстановить ее координаты на исходном fisheye-

изображении. Ниже приведен псевдо-код, вычисляющий текстурные координаты точки в текстуре, которая впоследствии будет спроецирована на сферу (текстурные координаты находятся в интервале [0, 1]):

```
// поворот Q вокруг оси Y на угол Da
Vec3 Q1 = y axis rotation(Da, Q)
Vec3 Q2 = Q1 / Dr
Real r = SF * (2 * acos(Q2.z) / PI)
Real a = atan(Q2.y / Q2.x) + angle
Vec2 TexCoord = {r * cos(a) / 2.0 + 0.5,
                 r * sin (a) / 2.0 + 0.5 }
```

Листинг 2 Трассировка лучей

4 Алгоритм работы с виртуальным экраном

Идея преобразования видео, предназначенного для просмотра на плоском экране, к fisheye-проекции состоит в использовании так называемого виртуального экрана. Составляется трехмерная сцена, состоящая из экрана, на котором отображается исходное видео. Экран может быть плоским, либо цилиндрическим и находиться в различных позициях (перед зрителями, сбоку, сверху и т. п.). Можно объединить несколько экранов в одной сцене для одновременного отображения нескольких видео файлов. После составления сцена проецируется на плоский кадр с использованием fisheye-камеры (Рис. 4). Затем полученное fisheye-видео будет проецироваться на сферический экран с помощью описанных выше методов.

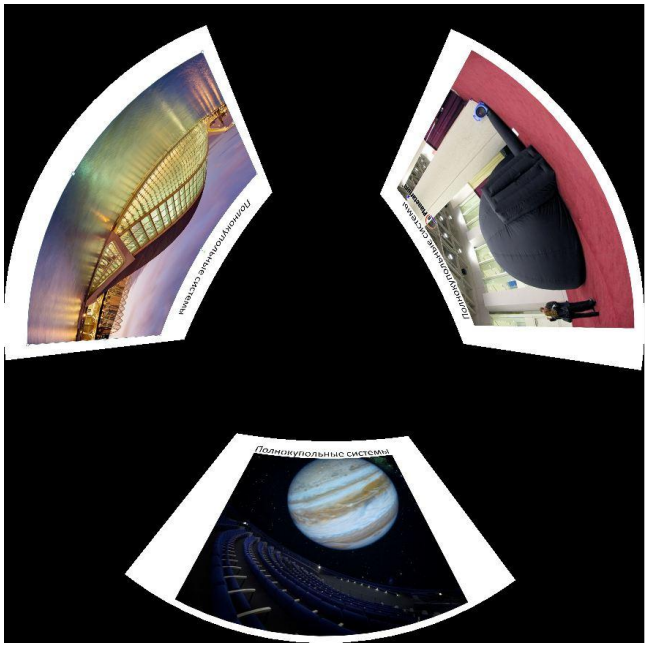


Рис. 4 Fisheye-кадр с виртуальными экранами

В отличии от обычной камеры, для которой проецирование выражается простым умножением трехмерных координат на проекционную матрицу, проецирование для fisheye-камеры не является аффинным преобразованием координат. Вместо этого преобразование выражается алгоритмически с использованием шейдеров (Листинг 3). Так как fisheye-проекция не зависит от параметров полнокупольной системы (единственный параметр проекции - разрешение результирующего изображения), то данное преобразование можно выполнять заранее.

```
vec3 fisheye(vec3 v)
{
  vec3 win;

  float view_scaling_factor = 1.0 / Fov * 180.0 /
              PI * min(Viewport[2], Viewport[3]);


  float oneoverh = 1.0 / sqrt(v.x * v.x +
                   v.y * v.y);
  float a = 0.5 * PI + atan(v.z * oneoverh);
  if (a > 0.5 * PI)
    a = 0.25 * (PI * PI) / (PI - a);
  float f = (a * view scaling factor) * oneoverh;

  win.x = Viewport[0] + Viewport[2] / 2.0 +
          v.x * f;
  win.y = Viewport[1] + Viewport[3] / 2.0 +
          v.y * f;
  win.z = (-v.z - zNear) / (zFar - zNear);
  win.x = 2.0 * (win.x - Viewport[0]) /
          Viewport[2] - 1.0;
  win.y = 2.0 * (win.y - Viewport[1]) /
          Viewport[3] - 1.0;
  win.z = 2.0 * win.z - 1.0;
  return win;
}
```

Листинг 3 Преобразование координат в fisheye - проекции

5 ДЕТАЛИ РЕАЛИЗАЦИИ ПРОГРАММЫ ДЛЯ КАЛИБРОВКИ

Программа калибровки позволяет настроить перечисленные выше параметры алгоритма.Процедура непосредственного вычисления карты преобразования путем трассировки лучей реализована на шейдерах OpenGL. Благодаря этому, вычисления выполняются в реальном времени.

Для реализации программы калибровки была выбрана библиотека Qt - кроссплатформенная, свободная (существует некоммерческая версия) библиотека для языка C++. Одним из плюсов библиотеки Qt является встроенная поддержка OpenGL, что упростило разработку приложения.

6 СОЗДАНИЕ ПЛАГИНА ДЛЯ VLC

VLC является продуктом с открытым исходным кодом. Заключительным этапом разработки является добавление необходимой функциональности в VLC посредством создания плагина. Для построения плеера из исходного кода используются make-файлы с надстройками из automake. Для включения в эту систему плагина необходимо добавить файлы с исходными кодами и объявить в них функции инициализации и деинициализации, предоставляемые библиотеками VLC.

Большую часть производимых операций решено было вынести из кода плагина и сгруппировать в специализированной библиотеке. Она содержит в себе функции инициализации и запуска необходимого фильтра, на вход которого подается буфер с исходным изображением и буфер, в котором будет содержаться результирующее изображение. После чего производится расчет карт смещения

пикселов с помощью загруженных из конфигурационного файла параметров и вышеизложенного алгоритма. Стоит отметить, что создание подобной карты проводится однократно, при запуске приложения. Далее используется уже готовая, хранящаяся в памяти, карта смещения. Благодаря этому удается достичь производительности в 60 кадров в секунду.

7 РЕЗУЛЬТАТЫ

В результате проведенных работ был реализован программный комплекс, состоящий из программы калибровки оборудования и плагина для видео плеера, позволяющих проигрывать fisheye-видео на малых полнокупольных системах, использующих проекцию сферического зеркала (Рис. 5). Данный программный комплекс был протестирован на малом куполе Нижегородского Планетария. Планируется использование разработанной системы для демонстрации научно-популярных фильмов.

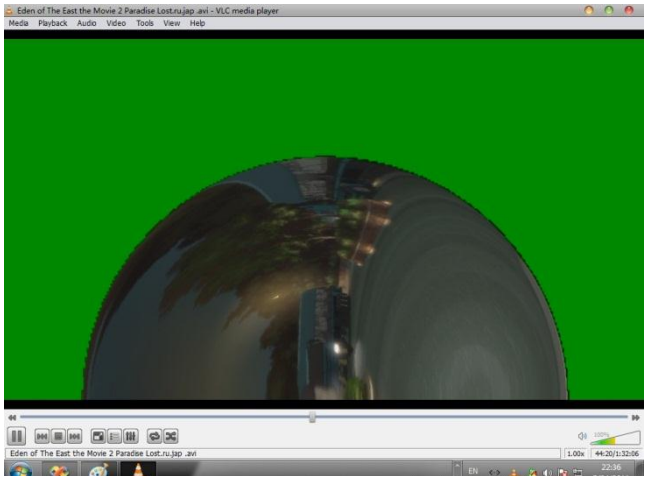


Рис. 5 Работа плагина для VLC

Также был разработан алгоритм и тестовая программа для показа на куполе видео, предназначенного для плоских экранов, путем проецирования его на виртуальный экран.

8 СПИСОК ЛИТЕРАТУРЫ

1. Роджерс Д, Адамс Дж. Математические основы машинной графики / Пер. с англ. – М.: Мир, 2001. – 604с.

2. Рост Р. Дж. OpenGL. Трехмерная графика и язык программирования шейдеров. Для профессионалов – СПб.: Питер, 2005. – 428 с.

3. Paul Bourke. Spherical mirror (Mirrordome) - A new approach to hemispherical dome projection. Planetarian, Vol 34(4), December 2005, pp 5-9.

4. P. D. Bourke. Using a spherical mirror for projection into immersive environments (Mirrordome). Graphite (ACM Siggraph), Dunedin Nov/Dec 2005. Proceedings of the 3rd international conference on Computer graphics and interactive techniques in Australasia and South East Asia. pp 281-284.

Поиск пути в 3D-пространстве с учетом динамических объектов

Винокурова Светлана
Лаборатория систем мультимедиа,
Марийский технический университет, Йошкар-Ола, Россия
svetlana_curls@mail.ru

Аннотация

В статье рассматривается расширение метода навигационного графа (НГ) для поиска путей в 3D-пространстве с использованием множества НГ, соответствующих каждому объекту, вместо единого графа. Метод позволяет учитывать динамические объекты без лишних вычислительных затрат в рамках единого алгоритма, а также существенно сокращает как объем ручной работы задания НГ, так и общее время работы алгоритма без ущерба для адекватности найденного пути.

Ключевые слова: *Поиск путей, Динамические объекты, Навигационный граф.*

1. ВВЕДЕНИЕ

Поиск пути (ПП) – задача нахождения наилучшего, оптимального маршрута между двумя точками пространства. В зависимости от контекста критерии оптимальности могут быть различные, но во всех случаях на найденном пути не должно встречаться препятствий. Важным достоинством системы ПП является возможность учитывать динамические объекты и корректировать найденный путь при необходимости. Динамическим объектом в системе считается объект, который может изменить позицию и угол поворота, появится или исчезнуть в любой момент времени.

ПП - важная часть виртуальной 3D-среды. ПП используется в компьютерных играх, различных симуляторах и тренажерах, чтобы обеспечивать корректное перемещение персонажей, управляемых компьютером. Кроме того ПП применяется в виртуальных мирах для управляемого пользователем персонажа.

2. ОБЗОР СУЩЕСТВУЮЩИХ МЕТОДОВ

Для ПП существуют 4 основных метода.

Первый из них подразумевает разбиение всего пространства на клетки одинаковой формы, для каждой из которых задано, проходима клетка или нет, и поиск пути алгоритмом A* [6]. Подразумевается, что объект, для которого идет ПП, занимает 1 клетку (Рис.1). Метод успешно используется в 2D-пространстве, но плохо применим к 3D-миру, т.к. имеет в этом случае серьезные недостатки:

- 3D-мир состоит из объектов произвольной формы, которые плохо ложатся на клеточную структуру пространства.
- Трудно учитывать перемещающиеся препятствия, так как при произвольной позиции они могут занимать до 4 клеток сразу (если размер препятствия 1 клетка).
- Перемещение по клеткам выглядит неестественно в трехмерном пространстве. Методы сглаживания пути сплайнами могут привести к проблемам с непроходимостью отдельных участков пути.
- Большое время ПП при большом количестве клеток.

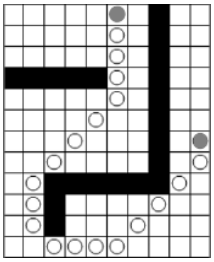


Рис. 1: Пример поиска пути на клеточном пространстве с помощью алгоритма A*.

Метод НГ предполагает ПП на графе, где вершины графа - трехмерные точки, ребра графа - отрезки, соединяющие эти точки, с ценой равной длине отрезка [2]. Все ребра графа являются проходимыми (персонаж может пройти по ним, не наткнувшись на препятствие). Задача ПП сводится к нахождению ближайших вершин к начальной и конечной точке, а затем к ПП на графе между этими вершинами с использованием критерия минимального веса общего пути (Рис.2). Метод хорошо подходит для 3D-пространства, однако можно принять во внимание 4 основных недостатка метода НГ [5]:

1. Большая трудоемкость задания графа, т.к. в некоторых случаях требуется задать чрезмерное число вершин графа.
2. Перемещаемые при помощи ПП объекты проходят пустое пространство по неестественной траектории, т.к. следуют по заданным ребрам графа. Проявление проблемы можно смягчить, если увеличить число вершин графа, что усугубит недостаток № 1 (Рис.3).
3. В ситуации, когда динамический объект находится на ребре НГ, нет корректного способа построить путь через это ребро.
4. НГ плохо подходит для ПП для перемещающихся объектов разных размеров, т.к. при размещении вершин графа приходиться ориентироваться только на 1 конкретный размер персонажа.

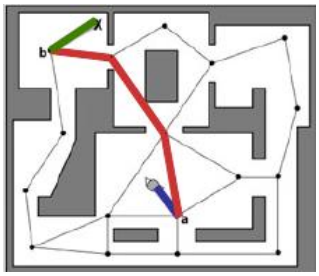


Рис. 2: Пример поиска пути с помощью НГ.

Метод сочетания эвристик [3] в 3D-пространстве предполагает применение специального алгоритма разрешения проблемной ситуации для небольшого числа распространенных случаев необходимости обхода препятствия. При его помощи полноценный ПП организовать невозможно.

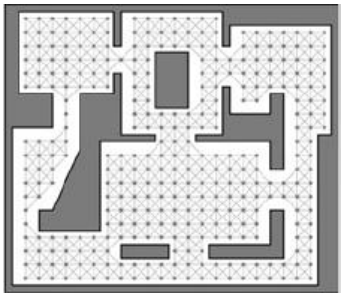


Рис. 3: Увеличение числа вершин графа для улучшения естественности найденного пути.

Метод navigation mesh [4] предполагает задание полигональной 3D-модели проходимого пространства (Рис.4). С помощью этого метода трудно учитывать динамические объекты, так как это требует больших вычислительных затрат.

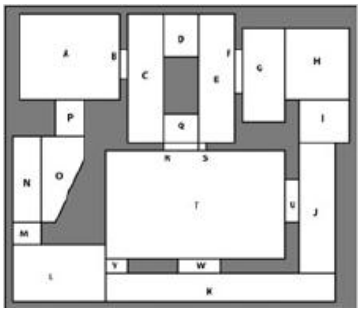


Рис. 4: Пример navigation mesh.

3. ПОИСК ПУТИ НА ОСНОВЕ УСОВЕРШЕНСТВОВАННОГО МЕТОДА НГ

На основе собственного расширения метода НГ была разработана система ПП для 3D-мира. Для ПП используется множество графов $A = \{A_i, i = 1..n\}$ (1), где $A_i = \{V, R\}$ (2) – НГ одного объекта, заданный множеством вершин V и множеством ребер $R = \{R_j, j = 1..m\}$ (3), где $R_j = \{V_{j1} \in V, V_{j2} \in V, T_A, T_D, j1 \neq j2\}$ (4), $T_A \in \{0,1\}$ – признак проходимости, T_D - контекстная информация о проходимости пространства вокруг ребра.

На практике НГ для каждой 3D-модели задается визуально с помощью специального инструмента [8].

ПП выполняется по известному отрезку пути S, соединяющему начальную и конечную точки $P_н$ и $P_к$. При поиске выполняются следующие шаги:

1. Поиск множества $A_{изм}$ динамических объектов, у которых изменились угол поворота или позиция. Замена объединенного графа на его составляющие, если найденный объект находился в составе объединенного графа.

   $A_{изм} = \{A_i$, такие, что $p_i \neq p_i'$ или $q_i \neq q_i'\}$ (5), где p_i, q_i – позиция и угол поворота объекта, соответствующего A_i, в текущий момент времени, а p_i', q_i' – в момент предыдущего ПП.

2. Пересчет координат ребер НГ найденных динамических объектов в соответствии с новой позицией и углом поворота.

3. Поиск множества $A_{пер}$ динамических объектов, НГ которых пересекаются друг с другом.

4. Объединение пересекающихся НГ динамических объектов и создание временного НГ, который учитывается при ПП вместо пересекающихся.

5. Поиск множества точек пересечения (ТП) P отрезка пути с НГ объектов. Первоначально выполняется проверка пересечения отрезка пути с bounding box НГ объекта и, только если проверка прошла успешно, выполняется поиск пересечений с ребрами графа. Это делает подход не менее эффективным, чем иерархические методы ПП [1].

   $P = \{P_i$, такие что $P_i \in S$ и существуют j, k такие, что $Pi \in R_j, R_j \in A_k\}$ (6).

6. Если найдена всего 1 ТП, то идет поиск сегментов НГ, перпендикуляр к которым из начальной или конечной точки имеет длину $< \varepsilon$. Из числа найденных сегментов выбирается тот, длина перпендикуляра к которому меньше других, и к множеству P добавляется P1 - ТП с перпендикуляром, образуя расширенное множество ТП P':

   $P' = P \cup P1$ (7).

7. Рассчитывается сортированное по удаленности от начальной точки пути множество ТП P_s.

   $P_s = \{P_i$, такие что $D(P_i, P_н) < D(P_i + 1, P_н)$ при $i \in 1..Np - 1\}$ (8),

   где D(A, B) – расстояние от точки A до точки B, Np – количество элементов множества P'.

8. Множество ТП разбивается на несколько частей P_{sk}, в каждой из которых идут подряд точки одного и того же графа.

   $P_{sk} = \{P_i$, такие что $P_i \in P_s$, а также $P_i \in A_k$, $P_{i+1} \in A_k$ при $i \in 1..Np - 1\}$ (9).

9. Внутри получившихся частей списка ПП ведется отдельно, а крайние точки соседних частей

соединяются прямым отрезком. При ПП внутри одного НГ рассматриваются варианты:

- 1 ТП — путь касается НГ и поиск оптимального пути не требуется.
- 2 и более ТП — выполняется ПП по крайним ТП по критерию минимальной суммарной длины ребер. Ребра учитываются в соответствии с признаком проходимости T_A.

10. Компоновка полученных отрезков путей в единый путь. Объединение производится за счет того, что берутся найденные пути для каждого графа и крайние точки этих путей соединяются прямой линией.

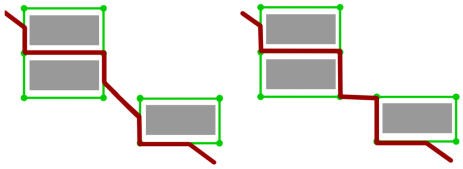


Рис. 5: ПП в модифицированном (слева) и базовом методах (справа) НГ. Серые прямоугольники - объекты, зеленые линии – НГ объектов, красные линии - найденные пути.

Описанный алгоритм, по крайней мере, не уступает в адекватности найденного пути базовому подходу, а в ряде случаев и превосходит его. Например, в усовершенствованном методе (Рис.5) путь по пространству свободному от препятствий зависит от координат начальной и конечной точек и проходит более естественно, в отличие от базового метода, где при относительно небольшом смещении начальной и конечной точек, путь проходит по одному и тому же ребру. Таким образом, расширенный алгоритм свободен от недостатков № 2, упомянутых в п.2.

Модифицированный алгоритм также позволяет реализовать ряд возможностей для сокращения объема ручной работы при задании НГ, что избавляет новый подход от недостатка метода НГ № 1:

1. Т.к. для каждого объекта используется свой НГ, пустое, гарантированно проходимое пространство не требует заполнения путями графа (Рис.6).
2. Для повторяющихся объектов НГ может быть скопирован автоматически.
3. Для обходных путей простой геометрической формы может быть использована автогенерация пути на основе данных о bounding box объекта и его положении в пространстве.

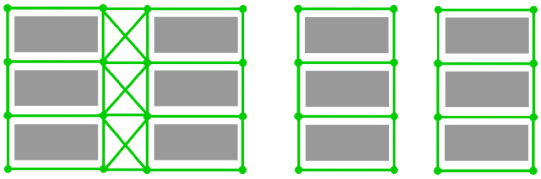


Рис. 6: НГ в базовом (слева) и модифицированном (справа) методах

4. УЧЕТ ДИНАМИЧЕСКИХ ОБЪЕКТОВ

Динамические объекты учитываются также как статические, заданные переносом (позицией) и углом поворота на момент начала ПП.

Рассмотрим приемы, которые позволяют учитывать динамические объекты в рамках общего алгоритма ПП.

4.1 Столкновение в момент следования по пути

Так как динамические объекты могут изменить свою позицию и угол поворота, то необходим механизм разрешения коллизии в момент следования по пути. В таких случаях предлагается находить путь заново из точки столкновения и, таким образом, учитывать новое положение динамического объекта.

4.2 Изменение состояния проходимости ребра

Ребро НГ, по которому идет найденный путь, может полностью оказаться в непроходимом пространстве. Чтобы учесть этот случай, в момент коллизии ребро отмечается как непроходимое с помощью признака проходимости T_A и при новом ПП не учитывается (Рис.7). Когда у динамического объекта изменяется позиция или угол поворота, все его ребра отмечаются как проходимые.

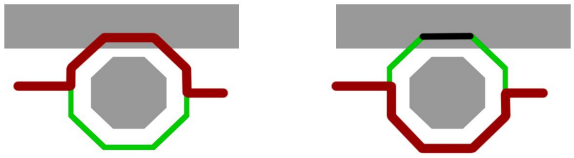


Рис. 7: Найденный путь обхода динамического объекта при первой попытке ПП (слева) и второй попытке (справа). Черным цветом выделено ребро, отмеченное как непроходимое после первой попытки.

4.3 Учет динамических объектов, расположенных на ребре НГ статического объекта

Для каждого ребра статического НГ задается контекстная информация T_D – информация о проходимости пространства вокруг ребра: ребро можно обойти слева, справа, слева и справа, нельзя обойти совсем. Эта информация используется для учета динамических объектов, расположенных на ребре НГ статического объекта. Возможны 2 случая:

- Ребро нельзя обойти. Оно помечается как непроходимое (Рис.8).

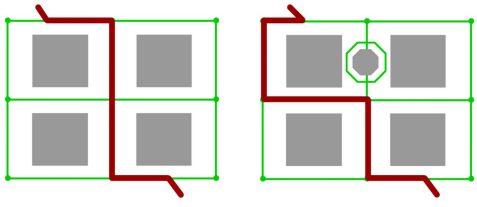


Рис. 8: ПП с динамическим объектом на ребре без заданного признака проходимости статического НГ (справа) и без него (слева).

- Ребро можно обойти. Тогда в п.10 алгоритма при сборке пути участок пути по рассматриваемому ребру между первой и последней ТП с динамическим объектом заменяется обходным путем динамического объекта. Направление обхода соответствует информации о проходимости (Рис.9).

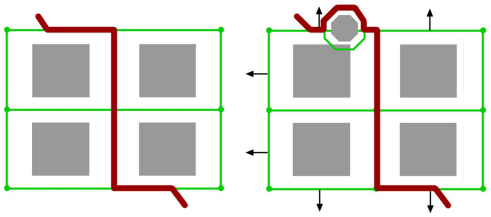


Рис. 9: ПП с динамическим объектом на ребре с заданным признаком проходимости статического НГ (справа) и без него (слева).

Пункты 4.2 и 4.3 позволяют расширенному методу НГ избавиться от третьего недостатка базового метода (п.2).

4.4 Слияние НГ

Так как динамические объекты могут занимать произвольное положение в пространстве, то возможна ситуация, когда их НГ пересекаются. Может оказаться, что ребро одного НГ динамического объекта проходит через другой динамический объект, и это ребро входит в найденный путь, который окажется некорректным. Чтобы избежать такой ситуации, необходимо слияние графов, такое, что все ребра объединенного графа окажутся проходимыми (Рис.10).

Алгоритм слияния:

1. Поиск ТП графов.
2. Ребра, которым принадлежат ТП, делятся ТП на части. Эти части рассматриваются как новые ребра графа вместо исходного.
3. Проверяем все ребра и отбрасываем те, которые находятся внутри одного из исходных графов.

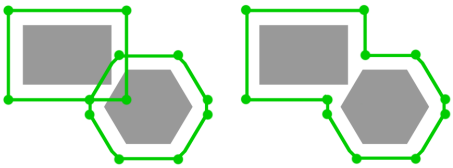


Рис. 10: НГ динамических объектов до слияния (слева) и после слияния (справа).

5. СИНХРОНИЗАЦИЯ

Часто требуется показать перемещение персонажа по пути не только на своем компьютере, но и на компьютерах других пользователей. При этом учет динамических объектов не должен усложнить синхронизацию.

Путь считается только для своего персонажа. Для других пользователей пересылается набор контрольных точек пути. Это существенно сокращает объем расчетов. В худшем случае п.4.1, путь будет скорректирован и послан заново, как и в ситуации смены пользователем направления пути.

6. ВЫВОДЫ

Чтобы получить представление об эффективности предложенного подхода, требуется оценить сложность учета динамических объектов. У п.1 две части. У первой части сложность O(n)

Шаг	Описание	Сложность
1.	Поиск динамических объектов, у которых изменился поворот или позиция	O(N), где N – число динамических объектов
2.	Пересчет координат вершин НГ найденных динамических объектов	$O(\sum_{i=1}^{k} len(V_i))$, где k – число найденных динамических объектов, len – количество элементов множества
3.	Поиск пересекающихся динамических объектов	O(k*N)

Табл. 1: Оценка сложности учета динамических объектов

Исходя из оценки сложности (Табл.1), можно сделать вывод, что алгоритм подходит для применения в реальном времени. При этом расширенный алгоритм свободен от 3 из 4 упомянутых в п.2. недостатков базового метода, что делает его более перспективным.

На основе описанного алгоритма была разработана программная система ПП в 3D-пространстве. Система была успешно апробирована в образовательной 3D-среде «Виртуальная Академия» [7].

7. REFERENCES

[1] A. Botea, M. Muller, and J. Schaeffer, “Near Optimal Hierarchical Path-finding,” *Journal of Game Development,* vol. 1, issue 1, 2004.

[2] Xiao Cui, Hao Shi, “A*-based Pathfinding in Modern Computer Games,” *IJCSNS International Journal of Computer Science and Network Security*, vol.11, no.1, January 2011.

[3] Mika M., Charla C. Simple, “Cheap Pathfinding,” *AI Game Programming Wisdom,* 2002.

[4] John C. O'Neill, “Efficient Navigation Mesh Implementation,” *Journal of Game Development*, vol. 1, no. 1, pp. 71-90, 2004.

[5] Paul Tozour, “Fixing Pathfinding Once and For All”, 2008. http://www.ai-blog.net/archives/000152.html.

[6] Peter Yap, “Grid-Based Path-Finding,” *AI '02 Proceedings of the 15th Conference of the Canadian Society for Computational Studies of Intelligence on Advances in Artificial Intelligence.*

[7] http://vacademia.com.

[8] http://www.youtube.com/watch?v=NMunrmtG00o.

Модификация алгоритма нелокального усреднения на основе анализа главных компонент для удаления шума из цифровых изображений

Владимир Волохов, Евгений Сергеев, Иван Мочалов
Лаборатория цифровые цепи и сигналы
Ярославский государственный университет им. П. Г. Демидова, Ярославль, Россия
volokhov@piclab.ru, sergeev@piclab.ru, dcslab@uniyar.ac.ru

Аннотация

В настоящей работе рассматривается метод восстановления цифровых изображений из зашумленных данных, основанный на методе нелокального усреднения. Приведены результаты моделирования, показывающие основные особенности данного метода. Проведено сравнение полученных результатов с результатами, синтезированными на основе «классического» алгоритма нелокального усреднения.

Ключевые слова: *алгоритм нелокального усреднения, анализ главных компонент, область обучения, векторы обучения, сохранение границ, фильтрация цифровых изображений.*

1. ВВЕДЕНИЕ

На данный момент времени задача фильтрации цифровых сигналов, изображений и видеопоследовательностей является по-прежнему актуальной. Последнее связано с непосредственным использованием указанных типов данных (далее цифровых изображений) в огромном количестве бытовых и научно-технических приложений, а также с тем, что теоретический предел [2] восстановления различных цифровых изображений современными алгоритмами фильтрации, на данный момент полностью не достигнут. Предполагая, что данные являются искаженными аддитивным белым гауссовским шумом, выделим несколько стандартных подходов к решению вышеозначенной задачи.

Алгоритмы локальной обработки цифровых изображений [5] возможно определить как методы, в которых весовые коэффициенты, используемые для взвешивания исследуемых данных, находящихся в окрестности оцениваемого пикселя (пикселов), зависят от расстояния до этого пикселя (пикселов), то есть принимают малые значения при большей пространственной удаленности исследуемых данных от оцениваемого пикселя (пикселов), в противном случае большие.

Алгоритмы нелокальной обработки цифровых изображений [5] возможно определить как методы, в которых весовые коэффициенты, используемые для взвешивания исследуемых данных, зависят от различия между значениями этих данных и значением оцениваемого пикселя (пикселов). В данном случае исследуемые данные, находящиеся в различных пространственных позициях на цифровом изображении, могут вносить одинаковый вклад в итоговую оценку.

Целью настоящей работы является анализ и построение модификации алгоритма нелокального усреднения, предложенного в работе [1]. Вначале кратко опишем основные особенности данного метода применительно к задаче шумоподавления.

2. АЛГОРИТМ НЕЛОКАЛЬНОГО УСРЕДНЕНИЯ

Предположим, что анализируемое цифровое изображение $\mathbf{x}$ искажено аддитивным белым гауссовским шумом $\mathbf{n}$ с нулевым математическим ожиданием и дисперсией σ^2.

1. Для обрабатываемого пикселя i зашумленного изображения $\mathbf{y} = \mathbf{x} + \mathbf{n}$ описываем квадратную окрестность, фиксированного размера, центрированную на этот пиксел.

2. Определяем подобность обрабатываемого пикселя i зашумленного изображения $\mathbf{y}$ с пикселом j того же изображения, используя взвешенное евклидово расстояние $\left\| y(N_i) - y(N_j) \right\|_{2,a}^2$, где N_i и N_j – квадратные окрестности, центрированные на пикселы i и j, соответственно, a – положительное число, определяющее среднеквадратическое отклонение гауссова ядра, используемого для вычисления взвешенного евклидова расстояния.

3. Определяем вес подобного к i пикселя j в итоговой оценке пикселя i:

$$w(i,j) = \frac{1}{z(i)} \cdot e^{-\frac{\left\| y(N_i) - y(N_j) \right\|_{2,a}^2}{h^2}}, \quad z(i) = \sum_j e^{-\frac{\left\| y(N_i) - y(N_j) \right\|_{2,a}^2}{h^2}},$$

где h – параметр, влияющий на степень фильтрации цифрового изображения.

4. Формируем итоговую оценку пикселя i на основе следующего выражения:

$$\hat{x}(i) = \sum_j w(i,j) y(j). \qquad (1)$$

Необходимо отметить, что основным достоинством данного алгоритма является высокое качество сохраненных границ цифрового изображения, а основными недостатками высокая вычислительная сложность и остаточный шум в локальных, однородных областях изображения. Настоящая работа позволяет решить вторую из вышеозначенных проблем. Для этого предлагается использовать комбинированную схему обработки, включающую в себя последовательное применение к изображению модификации алгоритма локальной обработки данных, основанную на анализе главных компонент [6] и алгоритма нелокального усреднения [1]. Этапы предложенного метода кратко описаны ниже.

3. ОПИСАНИЕ ПРЕДЛАГАЕМОГО АЛГОРИТМА

1. Оцениваем дисперсию шума σ^2 на входном зашумленном изображении $\mathbf{y} = \mathbf{x} + \mathbf{n}$. Для этого можно воспользоваться достаточно распространенной формулой из теории М-оценок:

$$\hat{\sigma}^2 = \frac{Медиана\,(ДВК_1)}{0{,}6745}, \qquad (2)$$

приведенной, например, в [6]. Здесь $ДВК_1$ – диагональные вейвлет-коэффициенты первого уровня вейвлет-разложения.

2. Разбиваем входное зашумленное изображение на совокупность перекрывающихся блоков, как представлено на рис. 1. Внутри каждого из таких блоков можно выделить: *область обучения*, *область фильтрации* и *область наложения блоков*. Размеры рассматриваемых областей могут варьироваться.

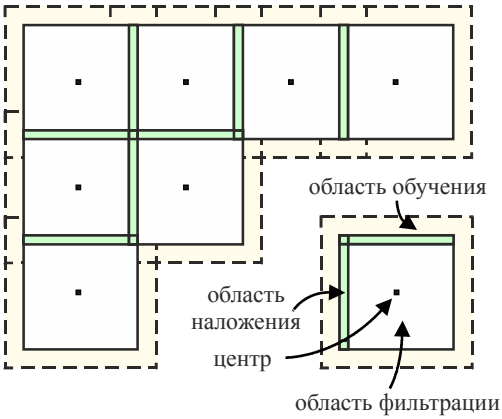


Рис 1: Локальная обработка изображения из работы [6]. Множество пикселов на цифровом изображении оценивается внутри области фильтрации, с использованием статистики набранной в области обучения.

3. Внутри области обучения выбираем всевозможные блоки размера $N \times N$ (*векторы обучения*). Последние, будучи представленными в виде векторов-столбцов длиной N^2 каждый, позволяют сформировать некоторую матрицу S_y^{I} размера $N^2 \times M$, столбцами которой и являются рассматриваемые векторы-столбцы. Здесь M – число векторов обучения, найденных в области обучения, а римские цифры I, II или III обозначают этап обработки изображения.

4. На основе матрицы S_y^{I} составляем ковариационную матрицу Q_y^{I}. Для матрицы Q_y^{I} находим собственные числа и соответствующие им собственные векторы (*главные компоненты* данных, заключенных в матрице S_y^{I}).

5. Для всех $l = 1,\ldots,N^2$ и $i = 1,\ldots,M$ находим проекции $y_{li}^{\mathrm{I}} = x_{li}^{\mathrm{I}} + n_{li}^{\mathrm{I}}$ множества векторов, заключенных в матрице S_y^{I}, на множество собственных векторов, найденных на предыдущем шаге [6]. Здесь y_{li}^{I} (*l*-я проекция вектора *i* из матрицы S_y^{I} на множество собственных векторов матрицы Q_y^{I}) представляет сумму *l*-ой проекция вектора *i* неискаженных данных и *l*-ой проекция вектора *i* шума.

6. Осуществляем обработку (фильтрацию) полученного множества проекций с использованием линейной среднеквадратической оценки:

$$\hat{x}_{li}^{\mathrm{I}} = \frac{\sigma_l^2}{\sigma_l^2 + \sigma^2} \cdot y_{li}^{\mathrm{I}}, \qquad (3)$$

представленной в [6]. Здесь σ^2 – дисперсия шума, а σ_l^2 – дисперсия *l*-ой проекции векторов $i = 1,\ldots,M$ неискаженных данных, которую можно найти с использованием оценки максимального правдоподобия [6]:

$$\hat{\sigma}_l^2 = \max\left[0, \frac{1}{M}\sum_{i=1}^{M}(y_{li}^{\mathrm{I}})^2 - \sigma^2\right]. \qquad (4)$$

7. На основе множества обработанных данных $\hat{x}_{li}^{\mathrm{I}}$ восстанавливаем матрицу S_y^{I}, а на основе последней отдельную обработанную область на изображении. Повторяя аналогичную операцию для остальных областей фильтрации с учетом их наложения, можно обработать зашумленное изображение целиком [6] и получить первую, «грубую» оценку $\hat{\mathbf{x}}^{\mathrm{I}}$ неискаженного изображения.

8. Используя зашумленное изображение, повторяем шаги 2-5, рассмотренные выше. При этом устанавливаются другие размеры областей обучения, областей фильтрации и областей наложения, а также размеры векторов обучения.

9. Выражение (3) заменяем следующим соотношением:

$$\hat{x}_{li}^{\mathrm{II}} = \frac{\left|z_{li}^{\mathrm{II}}\right|^2}{\left|z_{li}^{\mathrm{II}}\right|^2 + \sigma^2} \cdot y_{li}^{\mathrm{II}}, \qquad (5)$$

где y_{li}^{II} и z_{li}^{II} – *l*-е проекции вектора *i* из матриц S_y^{II} и S_z^{II} на множество собственных векторов матриц Q_y^{II} и Q_z^{II} для зашумленных данных и данных, полученных на основе «грубой» оценки сформированной на шаге 7, соответственно.

10. Повторяя рассуждения, изложенные на шаге 7, получаем вторую, «более точную» оценку $\hat{\mathbf{x}}^{\mathrm{II}}$ исходного изображения.

11. Применяем алгоритм нелокального усреднения [1] ко второй оценке $\hat{\mathbf{x}}^{\mathrm{II}}$ исходного изображения, устанавливая параметр, влияющий на степень фильтрации цифрового изображения – h с использованием следующего выражения:

$$h = 0{,}5 \cdot \sqrt{\sigma^2 - \sigma^2_{\mathbf{y}-\hat{\mathbf{x}}^{\mathrm{II}}}}, \qquad (6)$$

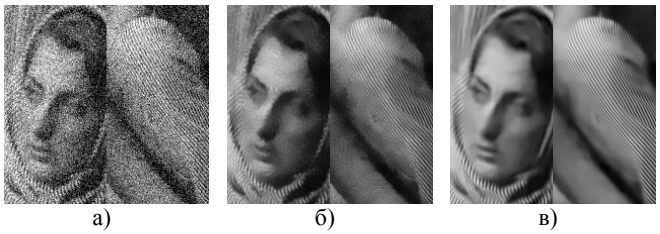
где $\sigma^2_{\mathbf{y}-\hat{\mathbf{x}}^{\mathrm{II}}}$ – дисперсия сигнала разности между зашумленным изображением $\mathbf{y}$ и второй оценкой $\hat{\mathbf{x}}^{\mathrm{II}}$ исходного изображения. Результатом этого этапа является итоговое восстановленное изображение $\hat{\mathbf{x}}^{\mathrm{III}}$.

4. РЕЗУЛЬТАТЫ МОДЕЛИРОВАНИЯ

Рассматриваемый в настоящей работе алгоритм был реализован с использованием пакета Matlab. Для проведения исследований использовался «классический» набор тестовых полутоновых изображений размера 256×256 и 512×512 пикселов, предложенных для анализа на интернет сайте http://www.cs.tut.fi/~foi/GCF-BM3D.

Визуальные и численные (пиковое отношение сигнала к шуму – ПОСШ и коэффициент структурного подобия – КСП

[7]) результаты моделирования кратко приведены на рис. 2, на примере «классического» тестового изображения «Барбара» размера 512×512 пикселов.



а) б) в)

Рис 2: а) Фрагменты зашумленного изображения «Барбара», ПОСШ = 17,54 дБ, КСП = 0,300; б) фрагменты обработанного изображения «Барбара», ПОСШ = 26,43 дБ, КСП = 0,713, полученные на основе метода [1]; в) фрагменты обработанного изображения «Барбара», ПОСШ = 28,08 дБ, КСП = 0,816, полученные на основе предложенного метода.

Как видно из приведенных фрагментов восстановленных изображений, предложенный алгоритм позволил полностью подавить шум, а также хорошо сохранить локальные особенности (криволинейные контура, границы и т.п.) анализируемого изображения. Последнее связано с наличием адаптивного подхода к анализу рассматриваемых данных, а также с использованием комбинированной схемы их обработки. Дополнительные численные результаты по сравнению работоспособности алгоритма фильтрации [1] и предложенного метода, приведены в табл. 1 (для метрики ПОСШ) и табл. 2 (для метрики КСП). В табл. 1 и табл. 2 лучшие результаты обозначены жирным шрифтом.

Тест. изобр.	σ = 5	σ = 15	σ = 20	σ = 25	σ = 35
Лена, 512×512	**37,54** 37,29	33,04 **33,83**	31,56 **32,66**	30,35 **31,69**	28,48 **30,09**
Лодки, 512×512	**35,39** 35,05	31,09 **31,43**	29,67 **30,21**	28,53 **29,20**	26,73 **27,55**
Барбара, 512×512	**36,05** 36,00	31,49 **32,41**	29,86 **31,13**	28,52 **30,01**	26,43 **28,08**
Фотограф, 256×256	33,91 **34,32**	**30,58** 30,22	**29,37** 28,97	**28,31** 27,91	**26,42** 26,17

Таблица 1: Численные результаты работы алгоритма нелокального усреднения [1], ПОСШ в дБ (выше) и предложенного алгоритма, ПОСШ в дБ (ниже).

Тест. изобр.	σ = 5	σ = 15	σ = 20	σ = 25	σ = 35
Лена, 512×512	**0,939** 0,935	0,858 **0,890**	0,815 **0,872**	0,771 **0,856**	0,689 **0,825**
Лодки, 512×512	**0,926** 0,917	0,820 **0,838**	0,773 **0,805**	0,729 **0,776**	0,647 **0,723**
Барбара, 512×512	**0,957** 0,956	0,884 **0,915**	0,840 **0,894**	0,796 **0,869**	0,713 **0,816**
Фотограф, 256×256	**0,946** 0,943	0,861 **0,878**	0,818 **0,847**	0,776 **0,820**	0,695 **0,776**

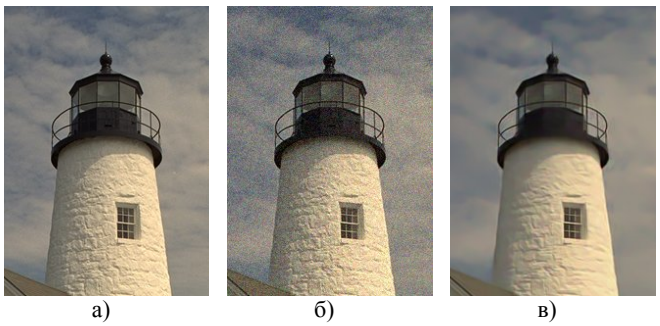
Таблица 2: Численные результаты работы алгоритма нелокального усреднения [1], КСП (выше) и предложенного алгоритма, КСП (ниже).

5. ДОПОЛНИТЕЛЬНЫЕ ПРИЛОЖЕНИЯ

Анализ литературы по удалению шумов из цифровых изображений показывает то, что современные методы подавления аддитивного белого гауссовского шума на полутоновых изображениях могут дополнительно использоваться в ряде других задач цифровой обработки изображений. Краткое описание и решение некоторых из них, приведено ниже.

5.1 Фильтрация цветных изображений

Задача фильтрации цветных изображений является достаточно актуальной с точки зрения современных практических приложений, поэтому на данный момент времени существует множество подходов к ее решению. Возможным вариантом обработки, рассматриваемым в настоящей работе, является прямая, поканальная обработка RGB-изображения, полученного, например, после выполнения операции интерполяции байеровских шаблонов. Необходимо отметить, что в процессе моделирования перехода от RGB-изображения к изображению с разделенной яркостной и цветовой информацией [3] не происходило, а аддитивный белый гауссовский шум подмешивался с одинаковыми характеристиками в каждый канал по отдельности. Последнее возможно, например, в случае формирования цифрового изображения с использованием трех независимых ПЗС- или КМОП-матриц. Пример обработки цветного тестового изображения базы данных Kodak, представленной для анализа на интернет сайте http://www.cipr.rpi.edu/resource/stills, приведен на рис. 3.



а) б) в)

Рис 3: а) Фрагмент исходного изображения; б) фрагмент зашумленного изображения, ПОСШ = 24,67 дБ, КСП = 0,655; в) фрагмент обработанного изображения, ПОСШ = 31,22 дБ, КСП = 0,845, полученный на основе предложенного метода.

5.2 Фильтрация смешанных шумов

Возможными усложнениями модели аддитивного белого гауссовского шума, рассматриваемой в рамках настоящей работы, могут служить модели смешанных шумов. Пример подобной модели, рассматриваемый в работе [4] для описания шума КМОП-матриц, может иметь следующий вид:

$$\mathbf{y} = \mathbf{x} + (\sigma_1 + \sigma_2 \mathbf{x})\mathbf{n}, \qquad (7)$$

где σ_1 и σ_2 – это константы, описывающие степень зашумления, а $\mathbf{n}$ – белый гауссовский шум с нулевым

математическим ожиданием и единичной дисперсией. При значении $\sigma_2 = 0$ модель (7) переходит в модель обычного аддитивного белого гауссовского шума.

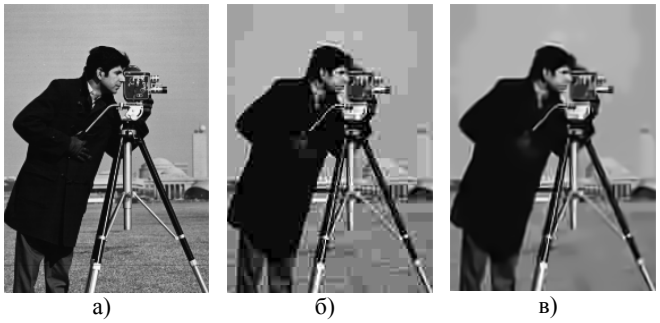
В силу неоднородности дисперсии шума на цифровом изображении, при наличии модели зашумления (7), то есть в силу зависимости шума от полезного сигнала, прямое использование методики восстановления исходного изображения из зашумленных данных, изложенной в пункте 3 настоящей работе, не представляется возможным. Поэтому для решения данной проблемы был использован подход на основе обобщенной гомоморфной фильтрации [4], позволяющий с использованием определенного преобразования (логарифмического типа) представить зашумленные данные $\mathbf{y}$ в виде суммы полезного сигнала и аддитивного белого гауссовского шума, обработать их с использованием предложенной методики и восстановить, используя обратное (к логарифмическому) преобразование данных. Пример обработки тестового изображения «Лодки», приведен на рис. 4, для случая $\sigma_1 = 25$, $\sigma_2 = 0{,}1$.

а) б) в)

Рис 4: а) Фрагмент исходного изображения «Лодки»; б) фрагмент зашумленного изображения «Лодки», ПОСШ = 16,66 дБ, КСП = 0,228; в) фрагмент обработанного изображения «Лодки», ПОСШ = 27,05 дБ, КСП = 0,710, полученный на основе предложенного метода.

5.3 Устранение артефактов блочности

Рассмотрим ситуацию, в которой модель зашумления полезного сигнала с использованием аддитивного белого гауссовского шума $\mathbf{y} = \mathbf{x} + \mathbf{n}$ описывает результат сжатия изображения с использованием алгоритма JPEG (рис. 5б).



а) б) в)

Рис 5: а) Фрагмент исходного изображения «Фотограф»; б) фрагмент сжатого изображения «Фотограф», ПОСШ = 25,03 дБ, КСП = 0,756; в) фрагмент обработанного изображения «Фотограф», ПОСШ = 25,91 дБ, КСП = 0,787, полученный на основе предложенного метода.

В данном случае шумовую компоненту $\mathbf{n}$ можно гипотетически принять за искажение, связанное с артефактами блочности на цифровом изображении (рис. 5б). Тогда решение задачи «фильтрации артефактов блочности» будет сводиться к нахождению дисперсии σ^2 шумовой компоненты $\mathbf{n}$. Возможный вариант поиска σ^2, с использованием априорных знаний о матрице квантования коэффициентов для стандарта сжатия JPEG, может быть найден в [3]. Пример обработки тестового изображения «Фотограф», приведен на рис. 5, для случая высокой степени сжатия цифрового изображения (качество сжатия $Q = 6$).

6. ЗАКЛЮЧЕНИЕ

В работе был предложен метод, позволяющий решать задачу удаления аддитивного белого гауссовского шума из цифровых изображений. К достоинствам алгоритма можно отнести возможность сохранения локальных особенностей изображений, а также адаптивность к анализируемым данным. К основным недостаткам алгоритма можно отнести высокую вычислительную сложность.

7. ЛИТЕРАТУРА

[1] Buades A., Coll B., Morel J. M. Nonlocal image and movie denoising // Int. J. Computer Vision, 2008. V. 76, № 2. P. 123 – 139.

[2] Chatterjee P., Milanfar P. Is denoising dead? // IEEE Trans. Image Processing, 2010. V. 19, № 4. P. 895 – 911.

[3] Foi A., Katkovnik V., Egiazarian K. Pointwise shape-adaptive DCT for high-quality denoising and deblocking of grayscale and color images // IEEE Trans. Image Processing, 2007. V. 16, № 5. P. 1395 – 1411.

[4] Hirakawa K., Parks T. W. Image denoising using total least squares // IEEE Trans. Image Processing, 2006. V. 15, № 9. P. 2730 – 2742.

[5] Katkovnik V., Foi A., Egiazarian K., Dabov K. From local kernel to nonlocal multiple-model image denoising // Int. J. Computer Vision, 2010. V. 86, № 8. P. 1 – 32.

[6] Muresan D. D., Parks T. W. Adaptive principal components and image denoising // Proc. IEEE Int. Conf. Image Processing, 2003. V. 1. P. 101 – 104.

[7] Wang Z., Bovik A. C., Sheikh H. R., Simoncelli E. P. Image quality assessment: from error visibility to structural similarity // IEEE Trans. Image Processing, 2004. V. 13, № 4. P. 600 – 612.

Об авторах

Владимир Волохов – сотрудник лаборатории цифровые цепи и сигналы Ярославского государственного университета им. П. Г. Демидова. Его адрес: volokhov@piclab.ru.

Евгений Сергеев – сотрудник лаборатории цифровые цепи и сигналы Ярославского государственного университета им. П. Г. Демидова. Его адрес: sergeev@piclab.ru.

Иван Мочалов – аспирант лаборатории цифровые цепи и сигналы Ярославского государственного университета им. П. Г. Демидова. Его адрес: dcslab@uniyar.ac.ru.

Отслеживание объектов с использованием инфракрасных маркеров

Рамиз Зейналов[1], Антон Якубенко[1], Илья Толкунов[2], Александр Мачихин[3],
[1]Московский государственный университет имени М. В. Ломоносова, Москва, Россия,
[2]ООО "ЭВЕРЕСТ ВИТ", Москва, Россия,
[3]Научно-технологический центр уникального приборостроения Российской академии наук, Москва, Россия,
rzeynalov@graphics.cs.msu.ru, toh@graphics.cs.msu.ru, itolkunov@pamega.ru, aalexanderr@mail.ru

Аннотация

В данной работе описывается метод отслеживания (трекинга) объектов с помощью стереопары, которая представляет собой две стационарные высокоскоростные камеры. Видимый диапазон камер отсекается с помощью диафрагмы, поэтому изображения обрабатываются в инфракрасном диапазоне. Объект обклеен инфракрасными маркерами, которые представляют собой диоды, излучающие свет в инфракрасном диапазоне. Для каждого момента времени требуется вычислить трёхмерные положения маркеров в пространстве и рассчитать по ним численные характеристики их движения. Особенность поставленной задачи состоит в том, что используется исключительно инфракрасный диапазон, на котором видны только инфракрасные маркеры, причём они неотличимы друг от друга. Как правило, в задачах, где применяется триангуляция точек, на входе есть видимый диапазон, который позволяет использовать дескрипторы особых точек для сопоставления проекций на разных видах, или оптический поток, позволяющий находить движущийся объект. Также при наличии видимого диапазона используются маркеры, которые несут в себе информацию о своём идентификаторе. В случае ИК-диапазона форма отслеживаемого объекта зачастую известна заранее. В имеющейся ситуации такие возможности отсутствуют, поэтому стоит задача получения межвидовых соответствий для маркеров.

Ключевые слова: *инфракрасные маркеры, трекинг, сопоставление точек, высокоскоростные камеры.*

1. ВВЕДЕНИЕ

Задача заключается в том, чтобы отследить движение объекта неизвестной формы, помеченного инфракрасными маркерами, в пространстве. Для отслеживания объекта используются именно ИК-маркеры, т.к. объект движется достаточно быстро и его изображение в видимом диапазоне очень сильно смазывается, в то время как ИК-маркеры выглядят достаточно контрастно, что делает их отслеживание более лёгким. После этого нужно определить центр масс объекта и характеристики его движения, такие как линейные и угловые скорости и ускорения объекта относительно его центра масс.

Задача трекинга объектов в настоящее время весьма востребована, поэтому уже существует множество различных её решений. Однако большинство из этих решений опирается на то, что форма объекта известна заранее [6], либо работа осуществляется в видимом диапазоне [9, 5], что позволяет использовать дескрипторы особых точек [1] или оптический поток [4] для поиска межвидовых соответствий.

2. ОБЗОР СУЩЕСТВУЮЩИХ РЕШЕНИЙ

Для отслеживания движущихся объектов на практике часто используются дескрипторы особых точек, такие как SIFT и SURF [1], которые позволяют найти особые точки на объекте [3, 12]. Для отслеживания небольшого количества объектов на статической сцене иногда используется оптический поток для поиска движущихся объектов [4, 12, 8]. Для этого берутся соседние кадры, вычисляется их разность. Те регионы, в которых эта разность больше определённого порога, считаются регионами движения, то есть регионами, в которых движется объект интереса. Просматривая попарно все кадры последовательности, можно получить проекции траекторий движения объекта на камеры. В нашей задаче оптический поток бесполезен, так как при его использовании не будут найдены межвидовые соответствия маркеров.

В некоторых задачах трекинга требуется отслеживать перемещение объекта заранее известной формы [6]. В таких случаях нередко этот объект помечен маркерами, что позволяет эффективнее отслеживать его на изображениях. Когда форма отслеживаемого объекта известна, можно использовать свойства, связанные с расположением маркеров, и легче отличать один маркер от другого. Это т.н. техника Motion Capture ([10, 13]).

Такие подходы не пригодны для решения поставленной задачи, потому что форма отслеживаемого объекта заранее неизвестна и должна быть получена в процессе решения задачи.

3. ПРЕДЛОЖЕННЫЙ МЕТОД

Предложенное решение задачи состоит из трёх фаз: калибровка системы, вычисление трёхмерных координат точек, вычисление характеристик движения и формы объекта.

3.1 Калибровка

Как и в других случаях, когда стереопара используется в качестве измерительной системы, требуется процесс калибровки для получения внутренних параметров камер (K – матрица камеры, содержащая принципиальную точку и фокусные расстояния, D – коэффициенты дисторсии) и их взаимного расположения (R – матрица поворота, T – вектор переноса) в общей мировой системе координат. Для калибровки используется трёхмерный шаблон, составленный из инфракрасных маркеров, расположенных на двух плоских стендах под углом 120 градусов друг к другу. Расположение маркеров в разных плоскостях позволяет получить более высокую точность калибровки при меньшем числе кадров. Каждый стенд составлен из маркеров, образующих сетку 9 на 9, причём между соседними маркерами расстояния одинаковые. На каждой сетке на центральной линии

(перпендикулярной к общему ребру сеток) частота маркеров удвоена, частота маркеров удвоена также и на общем ребре сеток (рис. 1). Это необходимо для обеспечения эффективного распознавания шаблона в тех случаях, когда он виден не полностью хотя бы на одной из камер. В качестве критерия качества восстановленных трёхмерных координат Q_i будем использовать ошибку репроекции точки, то есть расстояние от проекции p_j до проекции полученной трёхмерной точки Q_j с матрицей проекции камеры P_i:

$$E_{reproj}^{k,j} = \left\| p_j - un(P_k \cdot Q_j) \right\|, P_k = K_k \cdot \left(R_k \mid T_k \right)$$
$$un(P) = \frac{P}{P(3)} \quad (1)$$

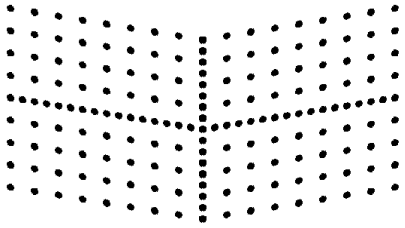


Рисунок 1: Калибровочный шаблон. Схематично.

Для распознавания маркеров сначала используется предобработка, а именно медианная фильтрация для удаления импульсного шума, гауссово размытие для удаления прочих шумов и для удаления видимого диапазона, если он не был подавлен физически оптическими фильтрами (или диафрагмой) на объективе. Для повышения контраста используется гамма-коррекция.

Для нахождения маркеров используется пороговая бинаризация и поиск связных компонент. Компоненты фильтруются по площади в пикселях – отсеивается шум и отдельные пиксели. Координаты пикселей ищутся как математическое ожидание яркости пикселей в связной компоненте.

Для распознавания шаблона сначала вычисляется пересечение основных линий шаблона (с удвоенной частотой маркеров) – две горизонтальные и одна вертикальная, которые вместе составляют крест в центре шаблона. Это пересечение ищется как крест, состоящий из пяти точек, наименее удалённых друг от друга. Пусть это будут точки P_c, P_{right}, P_{left}, P_{top}, P_{bottom}. После этого находятся линии креста – последовательно добавляются ближайшие точки. Затем ищутся точки, которые находятся в тех же строках/столбцах шаблона, что и точки, следующие за P_{right}, P_{left}, P_{top}, P_{bottom}. Назовём эти точки P_1, P_2, P_3 и P_4, начиная с правого верхнего квадранта шаблона по часовой стрелке. Затем берутся вторые точки на этих линиях и вычисляются гомографии из прямоугольника с соотношением сторон 2 : 1 в четырёхугольники, одна сторона которых лежит на вертикальной основной линии между точками, следующими за P_{top} и P_{bottom}, а противоположная ей находится между точками P1 и P2 (P3 и P4 соответственно для левой части шаблона). Используя гомографии, получаем следующие точки: сначала прогнозируем их положения, затем уточняем, пытаясь найти ближайшую точку на изображении, если она есть. После каждого такого шага гомографии уточняются. В результате получаем соответствия точек шаблона и точек на изображении. Если хотя бы один из них не выполнен, операция распознавания считается неуспешной.

3.2 Вычисление движения

Стадия вычисления движения является наиболее трудоёмкой и состоит из нескольких последовательных частей:

1. Предобработка изображений
2. Распознавание маркеров
3. Двухмерный трекинг проекций маркеров между кадрами
4. Вычисление межвидовых соответствий точек
5. Вычисление трёхмерных положений точек
6. Трёхмерное сопоставление точек
7. Вычисления характеристик движения объекта.

Предобработка изображений, распознавание маркеров выполняется так же, как и при калибровке системы.

Сначала производится трекинг проекций точек на каждой камере отдельно. Для этого на первом кадре все маркеры получают свои уникальные идентификаторы $P_{1,1}..P_{1,M}$, а на каждом следующем кадре для каждой точки $P_{i-1,j}$ на предыдущем кадре ищется ближайшая точка $P_{i,k}$ на данном кадре. Если расстояние между точками $P_{i-1,j}$ и $P_{i,k}$ меньше порога $T_{track2d}$, точка на новом кадре получает идентификатор точки $P_{i-1,j}$ на предыдущем кадре. Если на новом кадре остались точки, которые не получили идентификаторы, им назначаются новые идентификаторы. Таким образом, каждая точка на каждом кадре каждой камеры получает некоторый идентификатор, то есть получаются связи положений маркеров в различные моменты времени.

После этого необходимо восстановить трёхмерные координаты точек. Для этого нужно установить межвидовые соответствия. Об этом подробно написано в п. 4. Имея межвидовые соответствия точек, можно восстановить координаты трёхмерных точек путём триангуляции проекций. Для триангуляции используется алгоритм итерационных наименьших квадратов (Iterative-LS, [3]).

Когда становятся известны трёхмерные координаты отдельных точек в разные моменты времени, некоторые маркеры могут встречаться в последовательности несколько раз с разными идентификаторами. Это происходит, например, тогда, когда маркер в какой-то момент исчез из вида, затем через некоторое время появился (и, соответственно, мог успеть переместиться на значительное расстояние по неизвестному закону) и стал снова виден. В таком случае он будет иметь уже другой идентификатор. Для решения этой проблемы нужно восстановить межкадровые соответствия, так как один и тот же маркер может быть представлен в разные моменты времени разными идентификаторами. Эти соответствия восстанавливаются с помощью сопоставления отдельных фрагментов объекта, то есть, если какая-то часть объекта (состоящая из трёх и более точек) была видна некоторое время, затем одна точка пропала из вида, а другая появилась, можно вычислить относительное расположение исчезнувшей и появившейся точек, используя для этого систему координат трёх видимых точек.

3.3 Вычисление центра масс

Когда известны положения объекта в каждый момент времени, можно осуществлять анализ движения. Для этого сначала градиентным спуском находится положение центра

масс объекта. При этом оптимизируется суммарный момент вращения всех видимых маркеров объекта:

$$\sum_{i=1}^{N}\sum_{j=1}^{M}\left\|\vec{r}_i\right\|\cdot\left\|\vec{V}_i-\frac{(\vec{r}_i,\vec{V}_i)}{\left\|\vec{r}_i\right\|}\cdot\vec{r}_i\right\|,\vec{r}_i=\vec{P}_i-\vec{P}_0\,, \qquad (2)$$

где N – количество кадров, M – количество точек, V_i – скорость точки в системе координат объекта, P_i – координаты точки, P_0 – координаты центра масс, r_i – радиус-вектор точки в системе координат объекта. В качестве начального приближения берётся средняя точка объекта.

Затем вычисляются такие характеристики движения, как линейные и угловые скорости и ускорения центра масс объекта.

4. УСТАНОВКА МЕЖВИДОВЫХ СООТВЕТСТВИЙ

Для того чтобы произвести триангуляцию, необходимо узнать межвидовые соответствия точек, то есть соответствие проекций точек на разных камерах. Для этого используются эпиполярные ограничения, которые позволяют ограничить поиск соответствующей точки на другом виде до одной прямой. Однако одних только эпиполярных ограничений не хватает: точки могут лежать вблизи одной эпиполярной линии. Чтобы решить эту проблему, осуществляется перебор возможных вариантов. Это делается таким образом, чтобы минимизировать ошибку репроекции (1) и с учётом непрерывности движения проекций точек на каждом виде.

Рассмотрим пару кадров с разных камер в один момент времени. На первом кадре пусть будут точки $P_1..P_N$, на втором – $Q_1..Q_M$, причём может быть $M \neq N$, так как количество видимых маркеров в один момент времени может быть разным для двух камер. Для каждой точки каждого вида будем искать возможные соответствия на другом виде. Делать это будем путём построения эпиполярных линий для каждой рассматриваемой точки. Возможными соответствиями для данной точки будем считать точки на другой камере, находящиеся от соответствующей эпиполярной линии на расстоянии, не превышающем порог T_{EC}. Когда находим все возможные соответствия для каждой точки, нужно будет выбрать из них наилучшие, причём если точке P_i соответствует точка Q_j, то этой точке не могут соответствовать никакие другие точки, и наоборот. Для этого вычислим ошибки репроекции (1), то есть расстояния от исходных проекций точек на изображения до проекций полученных трёхмерных точек, для всех возможных пар соответствий (i; j) Затем будем брать соответствие с минимальной ошибкой репроекции из построенного списка возможных соответствий. Если возможных соответствий больше нет или минимальная ошибка репроекции (1) в этом списке превышает значение порога T_{reproj}, то заканчиваем искать соответствия для этого кадра, иначе мы нашли соответствие (i_0; j_0) . После этого удаляем все пары, конфликтующие с найденной, то есть пары вида (i_0; j) и (i; j_0), где $i \neq i_0$ и $j \neq j_0$, и выбираем из списка новую пару.

Таким образом, найдены соответствия для точек на одной паре кадров. Чтобы получить соответствия для всей последовательности, используется стратегия голосования. Это нужно для минимизации ошибок ложных соответствий и для повышения общей устойчивости алгоритма. Пусть всего на всех кадрах появилось M_1 и M_2 разных маркеров для первой и второй камеры соответственно. Введём матрицу голосования M (M_1 x M_2) и для каждой пары кадров пройдём по всем полученным соответствиям (i; j), увеличивая на единицу значение $M_{i,j}$. Таким образом получим матрицу голосов за каждое встретившееся соответствие. После этого можно искать глобальные для всей последовательности соответствия, извлекая их из полученной матрицы. Для этого будем выполнять следующие действия: пока матрица имеет ненулевые размерности и включает в себя ненулевые элементы, будем искать в ней максимальный элемент с индексами (i_0; j_0). Такие элементы будем записывать в список глобальных соответствий, при этом удаляя из неё строку с индексом i_0 и столбец с индексом j_0. В результате получаем глобальный список межвидовых соответствий маркеров. Имея список соответствий, можно производить триангуляцию, то есть вычисление координат трёхмерной точки по известным её проекциям на изображения и положениям камер в пространстве. В процессе триангуляции также можно предсказывать положения исчезнувших точек, то есть таких точек, которые были некоторое время видны на обеих камерах, после чего на одном из видов; аналогично для появившихся точек. Для этого вычисляются средние по нескольким ближайших кадрам значения скорости в пространстве и из положения точки в пространстве строится луч с направлением, которое характеризует скорость, строится луч для проекции из камеры, на которой данную точку видно, затем вычисляется псевдопересечение этих лучей. Если ошибка репроекции меньше порога, точка оставляется для дальнейших вычислений. Пусть $R_{i,j}$ – точка с номером j в момент времени i в пространстве, $P_{i,j}$ и $Q_{i,j}$ – её проекции на изображения первой и второй камеры соответственно. В случае исчезновения из вида первой камеры точки j на кадре i имеем: проекции $P_{i-K,j}$, $P_{i-K+1,j}$,.. $P_{i-1,j}$ известны, $P_{i,j}$ неизвестна, проекции $Q_{i-K,j}$, $Q_{i-K+1,j}$,.. $Q_{i,j}$ известны, точки в пространстве $R_{i-K,j}$, $R_{i-K+1,j}$,.. $R_{i-1,j}$ известны, $R_{i,j}$ неизвестна. Кроме того, известны калибровки камер и их взаимное расположение, то есть известны матрицы полных калибровок камер C_1 и C_2, которые включают в себя внутренние и внешние калибровки. По точкам $R_{i-K,j}$, $R_{i-K+1,j}$,.. $R_{i-1,j}$ можно вычислить среднюю скорость точки $R_{i,j}$ $V_{i,j}$. К тому же, зная проекцию $Q_{i,j}$ и калибровку второй камеры, можно построить луч, на котором должна лежать точка $R_{i,j}$. Координаты точки $R_{i-1,j}$ и её скорость $V_{i,j}$ задают луч в пространстве, на котором должна находиться точка $R_{i,j}$. Таким образом, точку $R_{i,j}$ можно определить как псевдопересечение этих лучей.

5. ТЕСТИРОВАНИЕ

Для тестирования алгоритма предлагаются такие метрики, как средняя ошибка репроекции (3) и процент использованных проекций точек при триангуляции.

$$\frac{1}{M}\sum_{j=1}^{M}E_{reproj}^{k,j} \qquad (3)$$

Ошибка репроекции является самым распространённым критерием оценки качества результата триангуляции, которая позволяет судить о том, насколько верно вычислены трёхмерные координаты точек и насколько их проекции отличаются от исходных. Кроме того, этот критерий свидетельствует о верности полученных межвидовых соответствий. Процент использованных проекций точек

позволяет судить, насколько эффективно использованы исходные данные, и делать выводы, какую часть трёхмерных точек удалось восстановить в процессе решения.

Тестирование производилось как на реальных, так и на синтетических данных. Реальных и синтетических данных было по 5 наборов. В качестве реальных данных брались результаты съёмки движения объекта с инфракрасными маркерами, при котором объект произвольным образом перемещался в пространстве. Синтетические данные представляли собой наборы изображений, такие же, как и реальные данные. Данные были с шумом и без шума, в движении объекта присутствовали перемещения и значительные вращения. Для создания изображений генерировался объект в движении. В качестве шума был выбран такой шум, при котором проекции смещались на случайный вектор, распределённый по закону нормального распределения с дисперсией в 2 пикселя. Во всех данных часто возникали ситуации, когда маркеры на изображениях камер при движении совмещались.

Однако даже в данных без шума нет 100-процентной точности и эффективности, так как нередко бывают ситуации, когда маркеры на изображении накладываются друг на друга (т.н. “marker swapping”, [11, 13]), и при этом не всегда удаётся верно восстановить соответствия. Во всех данных имелось небольшое количество точек (семь маркеров для реальных, восемь - для синтетических), что затрудняло распознавание объекта на некоторых кадрах, и, соответственно, вычисление его движения. В таблице 1 представлены обобщённые данные о результатах тестирования системы.

Данные	Средняя ошибка репроекции, пикс.	Процент использованных проекций, %
Реальные	0.49-0.80	66.73-79.69
Синтетические	0.02-0.278	81.30-84.63
Синтетические с высоким уровнем шума	0.91-1.09	59.84-75.13

Таблица 2. Результаты тестирования.

Анализ результатов тестирования подтвердил предположение о важности для алгоритма наличия достаточного числа маркеров на объекте, чтобы в каждый момент времени обе камеры могли видеть хотя бы три общих маркера, что позволило бы восстанавливать форму всего объекта из отдельных фрагментов. Однако число маркеров не должно быть слишком большим, чтобы не возникало проблем с частыми их наложениями друг на друга [11, 13], приводящими к падению качества результата в получении межвидовых соответствий.

6. ЗАКЛЮЧЕНИЕ

Предложенный метод эффективно решает задачу трекинга объектов. От существующих методов он отличается тем, что не требуется знать геометрию отслеживаемых объектов. Кроме того, на эти объекты не накладывается требование к неоднородности текстуры для возможности использования дескрипторов особых точек или оптического потока. Более того, во входных данных вообще не требуется наличие видимого диапазона, т.к. обработка целиком происходит в ИК-диапазоне, что значительно облегчает поиск маркеров. Научная новизна предложенного подхода заключается в том, что здесь не требуется заранее знать межвидовые соответствия для точек. Эти соответствия вычисляются предложенным алгоритмом поиска соответствий. Кроме алгоритма определения соответствий, предлагается новый шаблон калибровки и надёжный алгоритм его распознавания.

В дальнейшем планируется развитие проекта на случай использования более двух камер, что позволит значительно повысить точность и эффективность трекинга.

7. ЛИТЕРАТУРА

[1] H. Bay, T. Tuytelaars, L. V. Gool, SURF: Speeded Up Robust Features, 2008, CVIU'08, Vol. 110, No. 3, pp. 346-359

[2] C. Beall, B. Lawrence, V. Ila, F. Dellaert, 3D Reconstruction of Underwater Structures, 2010, IROS

[3] R. Hartley, P. Sturm, Triangulation, 1994

[4] Z. Khan, R. Herman, K. Wallen, T. Balch, An outdoor 3-D visual tracking system for the study of spatial navigation and memory in rhesus monkeys, 2005, Behavior Research Methods, vol. 37

[5] Z. Kim, Realtime Obstacle Detection and Tracking Based on Constrained Delaunay Triangulation, 2006, ITSC '06. IEEE, pp. 548-553

[6] M. Loaiza, A. Raposo, M. Gattass, A Novel Optical Tracking Algorithm for Point-Based Projective Invariant Marker Patterns, 2007

[7] D. P. Noonan, P. Mountney, D. S. Elson, A. Darzi, G.-Zh. Yang, A Stereoscopic Fibroscope for Camera Motion and 3D Depth Recovery during Minimally Invasive Surgery, 2009, ICRA '09, pp. 4463-4468

[8] S. Smith, Real-time motion segmentation and shape tracking, 1995, In Proc. 5th Int. Conf. on Computer Vision

[9] R. Subbarao, P. Meer, Y. Genc, A Balanced Approach to 3D Tracking from Image Streams, 2005, in Proc. IEEE and ACM ISMAR, pp. 70-78

[10] M. Weber, H. B. Amor, T. Alexander, Identifying Motion Capture Tracking Markers with Self-Organizing Maps, 2008, Virtual Reality Conference, VR'08. IEEE, pp. 297-298

[11] G. Welch, E. Foxlin, Motion Tracking: No Silver Bullet, but a Respectable Arsenal, 2002, IEEE Comput. Graph. Appl. 22, 6, 24–38.

[12] A. Yilmaz, M. Shah, Contour-Based Object Tracking with Occlusion Handling in Video Acquired Using Mobile Cameras, 2004, IEEE TPAMI, vol. 26, pp. 1531-1536

[13] Y. Zhao, J. Westhues, P. Dietz, J. Barnwell, S. Nayar, M. Inami, M. Nol, V. Branzoi, E. Bruns, Lighting Aware Motion Capture using Photosensing Markers and Multiplexed Illuminators, 2007, ACM TOG, Vol. 26, Issue 3, Article 36

Index of Authors

Index of Authors (Russian)

УДК 004.9
ББК 32.973.26-018.2
Г78

ГрафиКон’2011: 21-я Международная конференция по компьютерной графике и зрению: Москва, МГУ имени М.В. Ломоносова, 26–30 сентября 2011 г.: Труды конференции. – М.: МАКС Пресс, 2011. – 270 с.
ISBN 978-5-317-03808-3

УДК 004.9
ББК 32.973.26-018.2

Подготовка оригинал-макета:
М.М. Мизотин, А.В. Насонов

Напечатано с готового оригинал-макета

Подписано в печать 05.09.2011 г.
Формат 60x90 1/8. Усл.печ.л. 33,75. Тираж 100 экз. Изд. № 367.

Издательство ООО “МАКС Пресс”
Лицензия ИД N 00510 от 01.12.99 г.

119992, ГСП-2, Москва, Ленинские горы, МГУ им. М.В. Ломоносова,
2-й учебный корпус, 627 к.
Тел. 939-3890, 939-3891. Тел./Факс 939-3891.