Multidimensional filtering in application to progressive video rendering

Gruzdev A., Frolov V., Vostryakov K., Ignatenko A.

Lomonosov Moscow State University, Computational Mathematics and Cybernetics Department, Moscow, Russia

aleks-gruzdev92@yandex.ru, vfrolov@graphics.cs.msu.ru, kvostryakov@nvidia.com, ignatenko@graphics.cs.msu.ru

Abstract

In this paper we propose a new approach to progressive video rendering. We apply a multidimensional filtering to samples with additional information about a scene along with color. Our filter takes noisy output of path tracing with a low amount of paths per pixel and yields the resulting movie quicker than tracing a large amount of rays. Our approach adds only a linear member O(*N*) to a path tracing computing complexity (where *N* is amount of pixels). With the help of progressive scheme the resulting video quality is improving after the every iteration. Using our approach we can get a video of a good quality even computing 10 paths per pixel.

Keywords: *path tracing, multidimensional filtering, progressive rendering, video rendering, video filtering*.

1. INTRODUCTION

Photorealistic rendering is one of the main tasks of computer graphics. A properly high quality and physical correctness are provided by the Monte Carlo ray tracing algorithm [Ritschel et al 2011] (and its modifications: backward ray tracing, path tracing), but it has a high computing complexity that complicates application of ray tracing for real-time rendering. Often it takes minutes to render every frame of a movie even for a simple scene. When rendering process has finished and a user estimates the result he may want to change it by tweaking scene parameters, so the long rendering process should be performed again.

A small amount of rays can be used for faster frames rendering, but it leads to highly noised results, so a filtration is necessary in this case. But many fast denoising methods can't safe small details and sharp edges. On another hand, many high quality algorithms have a high complexity and can't work in real-time.

Aiming at achieving a high speed we apply a fast filtrating algorithm to noisy ray tracing results. It removes the most part of noise saving details and edges. The rest of noise is expected to be removed with the help of progressive rendering scheme after a few iterations.

2. RELATED WORKS

2.1. Bilateral filtering

Bilateral filtering [Tomasi and Manduchi 1998] smoothes images while preserving edges, by means of a non-linear combination of nearby image values. The method is non-iterative, local, and simple. It combines gray levels or colors based on both their geometric closeness and their photometric similarity, and prefers near values to distant values in both domain and range. In contrast with filters that operate on the three bands of a color image separately, a bilateral filter can smooth colors and preserve edges in a way that is tuned to human perception. Also, in contrast with standard filtering, bilateral filtering produces no phantom colors along edges in color images, and reduces phantom colors where they appear in the original image. The main disadvantage of the bilateral filtering is its computing complexity which doesn't allow to apply the classic algorithm for real-time filtering.

Bilateral filtering is computationally expensive due to the adaptive kernel recomputation at every pixel. [Pham and van Vliet 2005] present a **separable implementation of the bilateral filter**. Separable implementation of a multi-dimensional bilateral filter offers equivalent adaptive filtering capability at a fraction of execution time compared to the traditional filter.

In [Yang et al 2009] a new **real-time bilateral filtering** algorithm with computational complexity invariant to filter kernel size is proposed. Also, the algorithm lends itself to a parallel implementation. The method gives the same output accuracy and can be about 10x faster on average than the state-of-the-art.

2.2. Non-local denoising

Essentially different method of a **non-local image denoising** is proposed by [Buades et al 2005]. This algorithm computes for every pixel a weighted average of all pixels in the image. It compares spatial neighborhoods of pixels and gives large weights to the similar ones and small weights to others, so gathering information from the whole image, though that leads to a nonlinear complexity of the algorithm. In comparison to bilateral filtering [Tomasi and Manduchi 1998] the non-local approach shown better results of denoising, giving the best results on periodic images. In [Seo and Milanfar 2008] the non-local denoising is implemented for 3D filtration for video through enlargement neighborhoods along time axis.

2.3. Time-coherence video filtering

A modification of the non-local image denoising is presented by [Liu and Freeman 2010]. An integrating of optical flow into the method is a key to ensure temporal coherence in video filtering, and an approximate K-nearest neighbor matching reduces the high complexity of the classical algorithm.

[Bartovcak and Vrankic 2012] present an **adaptive pixel-wise algorithm** based on temporal averaging. Processing blocks of pixels requires lots of resources, and this approach observes a video as a group of 1D signals – one time depended signal per each pixel. The proposed method is simple and intuitive, has a lower computing complexity than some other algorithms. Giving comparable to other methods results, nevertheless it has a weakness of edges processing.

[Tawara et al 2004] propose to extend traditional photon density estimation methods for global illumination computation by using spatio-temporal bilateral filtering to reduce stochastic noise, while preventing excessive blurring in reconstructed lighting. This method is suitable for practical animation systems, where the rendering speed is the key factor even at the expense of lower accuracy in the lighting computation.

2.4. Multidimensional filtering

[Gastal and Oliveira 2012] present a new approach to efficiently performing high-quality **high-dimensional filtering**. The method

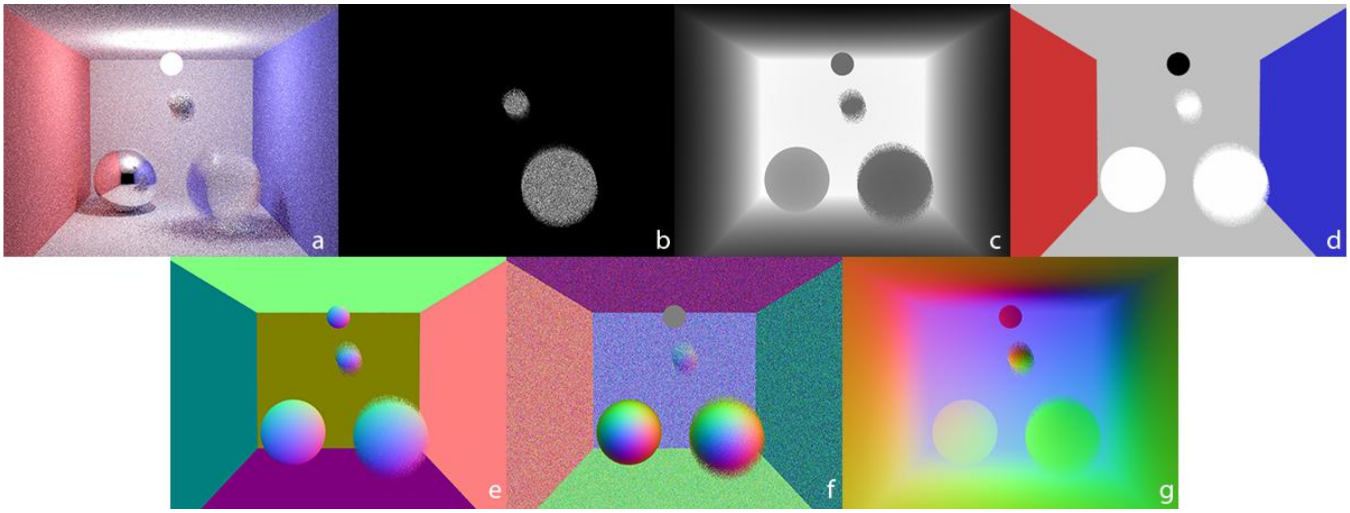


Figure 1. *Components of the sample vector: a) color; b) time; c) depth; d) material color; e) normal direction; f) reflected ray direction; g) shadow ray direction*

accelerates filtering by evaluating the filter's response on a reduced set of sampling points and using these values to interpolate the filter's response at all input pixels. The resulting filter is quite flexible, being capable of producing responses that approximate either standard bilateral filter [Tomasi and Manduchi 1998]. The presented filter can be implemented for a large number of dimensions, so it can be applied for video filtering or filtering with additional information. For a proper choice of the sampling points the total cost of the filtering operation is linear both in a number of pixels and in a number of dimensions.

[Sen and Darabi 2012] propose **random parameter filtering** for noisy results of Monte Carlo ray tracing with a low amount of samples per pixel. The method considers a sample as a high-dimensional vector of scene parameters, that allows using more information about a pixel than its color, and computes functional relationships between sample values and random parameter inputs. Then the approach uses all information to compute weights of every pixel when applying a cross-bilateral filter [Petschnigg et al 2004], which removes only the noise caused by the random parameters but preserves important scene detail.

2.5. Filtering for progressive rendering

[Schwenk et al. 2012] presents an approach for filtering in progressive Monte Carlo rendering. This method performs a strong denoising with saving sharp edges and is able to display the first resulted image after ray tracing a few paths per pixel. Using light path classification high-variance and low-variance noise is separated into different buffers; the bilateral filtering is applied only to high-variance noise. High complexity of bilateral filtering is compensated by not each frame denoising and accumulating new samples during filtration process.

2.6. Summary

The reviewed methods can be used for filtering video, though all of them have disadvantages for our task. The non-local approaches ([Buades et al 2005], [Tomasi and Manduchi 1998], [Seo and Milanfar 2008], [Liu and Freeman 2010]) have very high computing complexity, so they don't fit progressive video rendering well. The pixel-wise algorithm [Bartovcak and Vrankic 2012] has a weakness saving noise at edges, while the bilateral filtering process edges better. The bilateral filters ([Tomasi and Manduchi 1998], [Yang et al 2009], [Pham and van Vliet 2005]) show good results in image and video denoising, and can be applied for filtering results of 3D rendering ([Tawara et al 2004]), though they can't be applied for high-dimensional data without

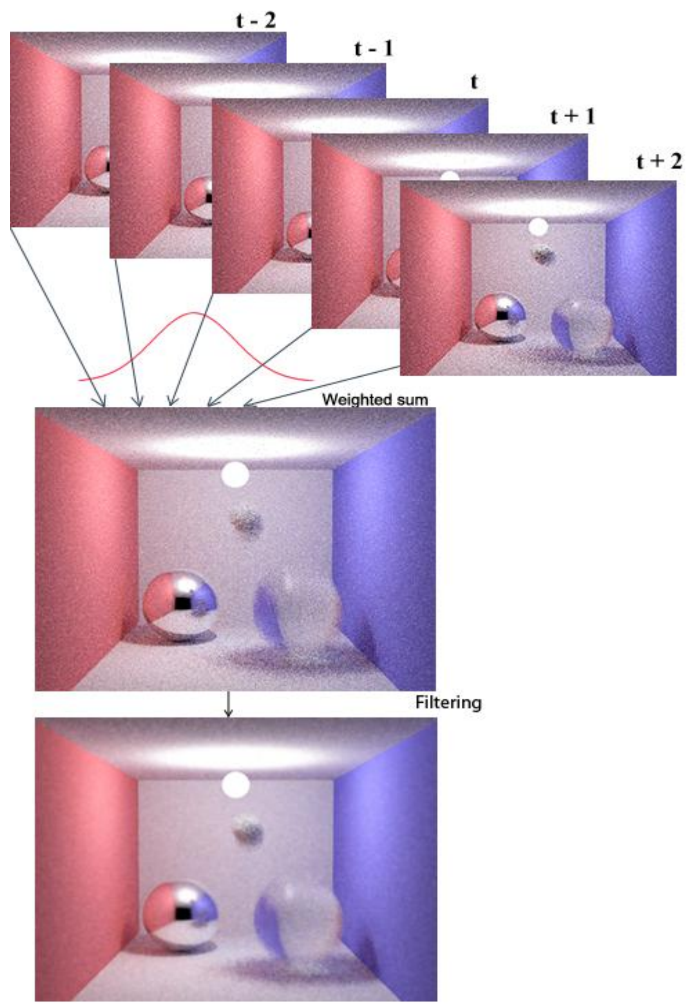


Figure 2. *Steps of the algorithm. Computing a weighted sum and filtering.*

modifications. The filtering for progressive Monte Carlo rendering [Schwenk et al. 2012] is not suitable for our task due to bilateral filtering disadvantages and an absence of ability to accumulate new samples during filtration process. Both [Gastal and Oliveira 2012] and [Sen and Darabi 2012] approaches can work with additional information and be applied for a large number of dimensions, though the first one is faster and implement a more general approach. So we use the method of high-dimensional filtering. In contrast to [Gastal and Oliveira 2012] we apply it to filtering of videos saving time coherence but for single images.

3. SUGGESTED APPROACH

In our implementation frames are rendered by Monte Carlo path tracer with a low amount of paths per pixel (1-8 paths). The algorithm takes at the input a sequence of frames with additional information about a scene: time $\boldsymbol{t}$ of a frame for filtering, parameter $\boldsymbol{\sigma}$ of blurring strength, a size of a window. We define the window as a sequence of rendered frames in a definite time range, including time $\boldsymbol{t}$ as well. (Fig. 2)

Information from all frames in the window is used for filtering the current frame. We use a Gaussian weighted sum while an accumulating samples from the frames. Also we can produce a new frame if the time $\boldsymbol{t}$ is set between existing frames. In future work we are going to avoid using a separate filter pass for time and use filtering capability of the multidimensional filter [Gastal and Oliveira 2012].

Each sample is considering as a high-dimensional vector: ***S* = (*R*, *G*, *B*, *T*, *Z*, *Mr*, *Mg*, *Mb*, *Nx*, *Ny*, *Nz*, *Dx*, *Dy*, *Dz*, *Sx*, *Sy*, *Sz*)**, where ***R*, *G*, *B*** – computed color; ***T*** – time; ***Z*** – ray depth value; ***Mr*, *Mg*, *Mb*** – material color; ***Nx*, *Ny*, *Nz*** – direction of a normal to the first ray hit point; ***Dx*, *Dy*, *Dz*** – direction of the reflected ray after the first bounce; ***Sx*, *Sy*, *Sz*** – direction of the shadow ray. (Fig. 1) All additional data can be got from the classical path tracer and don't require any considerable modification of the path tracing algorithm.

As a stopping criterion of the [Gastal and Oliveira 2012] algorithm, a manifolds tree depth limit is chosen. The depth 2 is considered as a good balance of denoising quality and computing speed. The fixed depth and dimensions amount make the filtration complexity O(N).

The algorithm works iteratively, processing the whole video sequence and outputting the resulted video on the each iteration. The following main steps of processing one video frame can be marked out:

1) Accumulating information to the current frame from neighbor frames by computing samples as a sum of all frames with the Gaussian weights. We use the Gauss distribution with mathematical expectation equal $\boldsymbol{t}$ and dispersion equal 1 (the value can be varied like an algorithm parameter).

2) Performing high-dimensional filtration of the current frame, using the method of [Gastal and Oliveira 2012]. We reduce the blurring strength $\boldsymbol{\sigma}$ after the each iteration [Hachisuka et al 2012]:

$$\frac{\sigma_{i+1}^2}{\sigma_i^2} = \frac{i+a}{i+1},$$

where $\boldsymbol{i}$ – iteration number, $\boldsymbol{a}$ – a parameter from (0; 1). This way we achieve consistency of the Monte-Carlo estimation and filtering blur will be vanished when the number of iterations gets infinite.

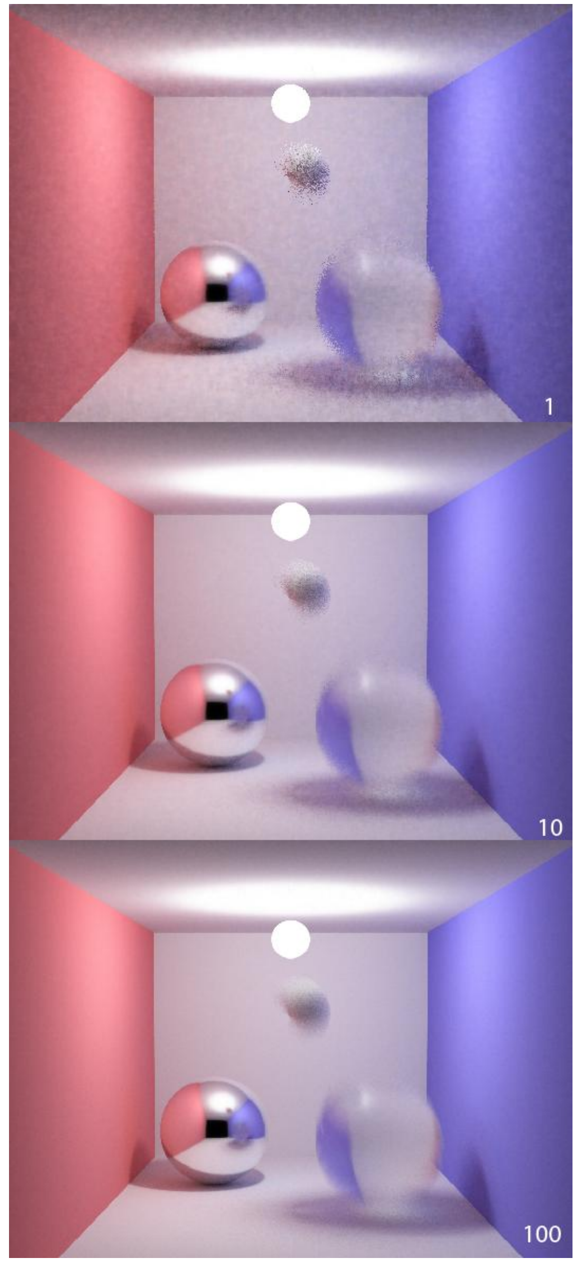
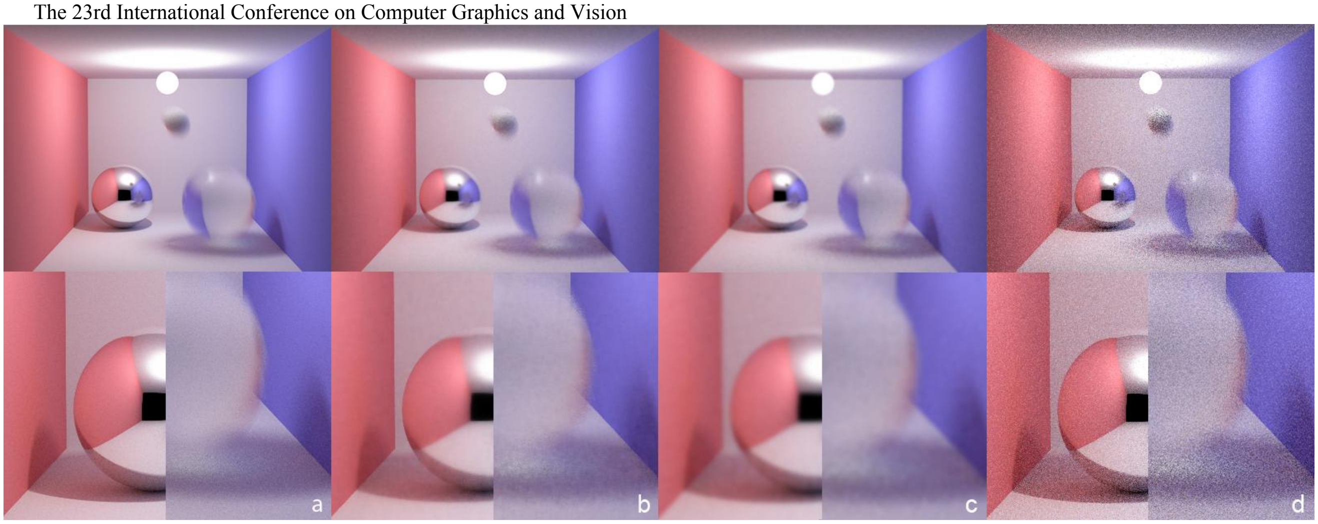


Figure 3. *Output of the algorithm after 1, 10 and 100 iterations*

3) The resulted samples are added to a current frame buffer with cumulative values of all previous iterations. The buffer contents can be displayed right after this step.

4. RESULTS

Our implementation performs a progressive video rendering through path tracing with 1 sample per pixel and multidimensional filtering per iteration. The method reaches a good output quality after 10 iterations (Fig. 3). Also the motion blur effect, that usually radically influences performance, can be



a) Etalon rendering. 100 spp. *b) Our approach. 10 spp.* *c) Progressive rendering with Gaussian blurring with 10 spp.* *d) Non-filtered path tracing rendering with 10 spp.*

Figure 4. *Comparison of progressive filtering methods.*

achieved with the help of this algorithm with only small additional performance cost.

Taking into account the additional information gives an advantage over other non-multidimensional algorithms. Figure 4 shows that our method yields a result comparable to etalon path tracing after 10 iterations, while Gaussian filter keeps edges blurry. Path tracing of the equal amount of paths outputs a noisy image.

The algorithm was implemented on MATLAB with the help of the filtration scripts of [Gastal and Oliveira 2012]. Frames samples were got from smallpt path tracer [Beason 2010].

Taking into consideration the linear complexity of the [Gastal and Oliveira 2012] filtering and a high speed of tracing 1 path per pixel we expect this algorithm to work in the real-time on the modern GPU. This is our next goal.

5. ACKNOWLEDGMENTS

The work is supported by the following grant: RBRF- 12-01-31027 MOL_A

6. REFERENCES

[1] [Bartovcak and Vrankic 2012] Bartovcak, D. and Vrankic, M.: *Video denoising based on adaptive temporal averaging.* In Engineering Review vol. 32, issue 2, 64-69. 2012

[2] [Beason 2010] Beason, K.: *Path tracer smallpt.* http://www.kevinbeason.com/smallpt/

[3] [Buades et al 2005] Buades, A., Coll, B. and Morel, J.M.: *A non-local algorithm for image denoising.* In IEEE Conference on Computer Vision and Pattern Recongnition (CVPR). (2005)

[4] [Gastal and Oliveira 2012] *Adaptive manifolds for real-time high-dimensional filtering.* In ACM SIGGRAPH 2012 vol 31, issue. 4, July 2012

[5] [Hachisuka et al 2012] Hachisuka, T., Jarosz, W., Bouchard, G., Christensen, P., Frisvad, J.R., Jakob, W., Jensen, H.W., Kaschalk, M., Knaus, C., Selle, A., Spencer, B.: *State of the art in photon density estimation*. In ACM SIGGRAPH 2012 Courses (New York, NY, USA, 2012), SIGGRAPH '12, ACM, pp. 6:1-6:469. 4, 6.

[6] [Liu and Freeman 2010] Liu, C. and Freeman, W.T.: *A high-quality video denoising algorithm based on reliable motion estimation.* In European Conference on Computer Vision (ECCV). 2010.

[7] [Petschnigg et al 2004] Petschnigg, G., Agrawala, M., Hoppe, H., Szeliski, R., Cohen, M., and Toyama, K.: *Digital photography with flash and no-flash image pairs.* In ACM Transactions on Graphics, vol 23, no. 3, pp 664-672, Proceedings of the ACM SIGGRAPH Conference, 2004.

[8] [Pham and van Vliet 2005] Pham, T.Q. and van Vliet, L.J.: *Separable bilateral filtering for fast video preprocessing.* In: International Conference on Multimedia and Expo. (2005)

[9] [Ritschel et al 2011] Ritschel, T., Dashsbacher, C., Grosch, T., Kautz, J.: *The state of the art in interactive global illumination.* In Computer Graphics Forum, vol 31, issue 1, p 160-188, February 2012.

[10] [Schwenk et al. 2012] Schwenk, K., Kuijper, A., Behr, J., Fellner, D.W.: *Practical noise reduction for progressive stochastic ray tracing with perceptual control.* In IEEE Computer Graphics and Applications, vol 32, issue 6, pp 46-55. Nov.-Dec. 2012.

[11] [Sen and Darabi 2012] Sen, P. and Darabi, S.: *On filtering the noise from random paramenters in Monte Carlo rendering.* In ACM Transactions on Graphics, vol. 31, issue 3, no. 18, May 2012.

[12] [Seo and Milanfar 2008] Seo, H.J. and Milanfar, P.: *Video denoising using higher order optimal space-time adaptation.* In IEEE 1-4244-1484-9/08. (2008)

[13] [Tawara et al 2004] Tawara, T., Myszkowski, K., Dmitriev, K., Havran, V., Damez, C., Seidel, H.-P.: *Exploiting temporal coherence in global illumination*. SCCG 2004, ISBN:1-58113-967-5

[14] [Tomasi and Manduchi 1998] Tomasi, C. and Manduchi, R.: *Bilateral filtering for gray and color images*. In: ICCV. (1998) 839–846.

[15] [Yang et al 2009] Yang, Q., Tan K-H., Ahuja, N.: *Real-time O(1) bilateral filtering.* 978-1-4244-3991-1/09 IEEE

About the author

Gruzdev Alexey is a student at Moscow State University, Department of Computational Mathematics and Cybernetics. His contact email is aleks-gruzdev92@yandex.ru.

Frolov Vladimir is a postgraduate at Keldysh Institute of Applied Mathematics. His contact email is vfrolov@graphics.cs.msu.ru.

Vostryakov Konstantin is an engineer at NVIDIA. His contact email is kvostryakov@nvidia.com.

Ignatenko Alexey is a researcher, candidate of physical and mathematical science at Moscow State University, Department of Computational Mathematics and Cybernetics. His contact email is ignatenko@graphics.cs.msu.ru.