"GrowCut" - Interactive Multi-Label N-D Image Segmentation By Cellular Automata

Vladimir Vezhnevets * Vadim Konouchine

Graphics and Media Laboratory †
Faculty of Computational Mathematics and Cybernetics
Moscow State University,
Moscow, Russia.

Abstract

In this paper we describe a novel algorithm for interactive multi-label segmentation of N-dimensional images. Given a small number of user-labelled pixels, the rest of the image is segmented automatically by a Cellular Automaton. The process is iterative, as the automaton labels the image, user can observe the segmentation evolution and guide the algorithm with human input where the segmentation is difficult to compute. In the areas, where the segmentation is reliably computed automatically no additional user effort is required. Results of segmenting generic photos and medical images are presented. Our experiments show that modest user effort is required for segmentation of moderately hard images.

Keywords: interactive image segmentation, graph cut, cellular automata, image editing, foreground extraction

1 Introduction

Image segmentation is an integral part of image processing applications like medical images analysis and photo editing. A wide range of computational vision algorithms can also benefit from existence of reliable and efficient image segmentation technique. For instance, intermediate-level vision problems such as shape from silhouette, shape from stereo and object tracking could make use of reliable segmentation of the object of interest from the rest of the scene. Higher-level problems such as recognition and image indexing can also make use of segmentation results in matching.

Fully automated segmentation techniques are being constantly improved, however, no automated image analysis technique can be applied fully autonomously with guaranteed results in general case. That is why semi-automatic segmentation techniques that allow solving moderate and hard segmentation tasks by modest effort on the part of the user are becoming more and more popular.

Several powerful techniques for interactive image segmentation have been proposed recently based on graph cuts [Boykov and Jolly 2001], [Rother et al. 2004] and random walker [Grady and Funka-Lea 2004]. They seem to significantly outperform earlier methods both by resulting segmentation quality and required user effort.

This paper's contribution is twofold. First, we propose a new interactive segmentation scheme, based on cellular automata, that has several favorable properties (see section 2 for details):

1. Capable of solving moderately hard segmentation tasks (see examples in this paper);
2. Is easy in implementing and allows efficient parallel implementation;
3. Works with images of any dimension $N \geq 1$;
4. Performs multi-label image segmentation (the computation time does not depend on the number of labels);
5. Is extensible, allows constructing new families of segmentation algorithms with specific properties;
6. Is truly interactive - the user observes the process of computing the segmentation and is able to make modifications and corrections at any time;

Second, and most important, we would like to attract attention to another possible view on segmentation problem, apart from graph theory. Cellular automata family is very rich and diverse, and our hope is that by applying experience and knowledge accumulated in this field to interactive image segmentation task, research community will be able to come up with some novel and ingenious solutions.

1.1 Related work

In this section we briefly outline main features of current state-of-the-art interactive segmentation techniques. We informally divide these methods into two families - proposed for generic image editing, and developed especially for medical images. In reality, most of the methods can be successfully applied in both domains.

Magic Wand - is a common selection tool for almost any image editor nowadays. It gathers color statistics from the user-specified image point (or region) and segments (connected) image region with pixels, which color properties fall within some given tolerance of the gathered statistics.

Intelligent paint - is a region-based interactive segmentation technique, based on hierarchical image segmentation by tobogganing [Reese 1999]. It uses a connect-and-collect strategy to define an objects region. This strategy uses a hierarchical tobogganing algorithm to automatically connect image regions that naturally flow together, and a user-guided, cumulative cost-ordered expansion interface to interactively collect those regions which constitute the object of interest. This strategy coordinates human-computer interaction to extract regions of interest from complex backgrounds using paint strokes with a mouse.

Intelligent scissors - is a boundary-based method, that computes minimum-cost path between user-specified boundary points [Mortensen and Barrett 1998]. It treats each pixel as a graph node and uses shortest-path graph algorithms for boundary calculation. A faster variant of region-based intelligent scissors uses tobogganing for image oversegmentation and then treats homogenous regions as a graph nodes [Mortensen and Barrett 1999].

*e-mail: vvp@graphics.cmc.msu.ru
†www: http://graphics.cmc.msu.ru

Graph Cut is a combinatorial optimization technique, which was applied by [Boykov and Jolly 2001] to the task of image segmentation. The image is treated as a graph - each pixel is a graph node. For the case of two labels (i.e. object and background) the globally optimal pixel labelling (with respect to defined cost function) can be efficiently computed by max-flow/min-cut algorithms. This technique can be applied to N-dimensional images. Given user-specified object and background seed pixels, the rest of the pixels are labelled automatically.

GrabCut [Rother et al. 2004] extends graph-cut by introducing iterative segmentation scheme, that uses graph-cut for intermediate steps. The user draws rectangle around the object of interest - this gives the first approximation of the final object/background labelling. Then, each iteration step gathers color statistics according to current segmentation, re-weights the image graph and applies graph-cut to compute new refined segmentation. After the iterations stop the segmentation results can be refined by specifying additional seeds, similar to original graph-cut.

Medical images have their own unique properties. In many cases they are graylevel and objects that should be segmented are very different in their structure and appearance from the objects that are common in photo editing. Probably that is why much research effort was applied for developing specific and efficient segmentation methods for the medical images domain. Nevertheless, some segmentation techniques like Graph Cuts can successfully be applied to medical images also. Some specific 'medical' segmentation techniques are reviewed further.

Marker-based watershed transformation uses watershed transform, supported by user-specified markers (seeds) for segmenting graylevel images [Moga and Gabbouj 1996]. Watershed transform treats the image as a surface with the relief specified by the pixel brightness, or by absolute value of the image gradient. The valleys of the resulting 'landscape' are filled with water, until it reaches the 'mountains'. Markers placed in the image specify the initial labels that should be segmented from each other.

Random walker [Grady and Funka-Lea 2004] given a small number of pixels with user-defined seed labels (labels number can be grater than 2), analytically determines the probability that a random walker starting at each unlabelled pixel will first reach one of the pre-labelled pixels. By assigning each pixel to the label for which the greatest probability is calculated, image segmentation is obtained. This method provides unique segmentation solution into connected segments, with some robustness against 'weak' boundaries. A confidence rating of each pixel's membership in the segmentation is also estimated.

Interactive region growing is a descendant of one of the classic image segmentation techniques. Initially, the seed pixel inside the object of interest is specified, and then neighboring pixels are iteratively added to the growing region, while they conform to some region homogeneity criterium. The main problem is 'leaking' of the growing region through 'weak' boundaries, however some treatment of this problem was proposed [Heimann et al. 2004]. This method works with two labels only - object and background.

The performance of described photo editing methods was evaluated in [Rother et al. 2004] (except for the intelligent paint). The authors have clearly shown, that methods based on graph cuts allow achieving better segmentation results with less user effort required, compared with other methods. One of the few drawbacks of the graph-based methods is that they are not easily extended to multi-label task and the other is that they are not very flexible - the only tunable parameters are the graph weighting and cost function coefficients. For example, additional restrictions on the object boundary smoothness or soft user-specified segmentation constraints cannot be added readily. As for the intelligent paint, judging by the examples supplied by the authors, the advantage of their method over the traditional 'magic wand' is in speed and number of user interactions. As it appears from the algorithm description and presented results, it is unlikely that intelligent paint would be capable of solving hard segmentation problems like in [Rother et al. 2004], [Boykov and Jolly 2001]. Precise object boundary estimation is also questionable, because the finest segmentation level is obtained by initial tobogganing oversegmentation, which may not coincide with actual object borders.

Speaking about medical images, the best performing method is random walker (judging by the provided examples). It leaves behind both watershed segmentation and region growing behind in quality and robustness of segmentation. The quality of segmentation comparable to is graph cuts, but random walker is capable of finding the solution for number of labels > 2. However, it is rather slow and its implementation is not an easy task. Also, method extension to achieve some special algorithm properties (i.e. controllable boundary smoothness) is not straightforward. It should be mentioned, that multi-labelling tasks *can* be solved by min-cut graph algorithms [Boykov et al. 2001], but no attempt to apply this multi-labelling method to interactive image segmentation is known to us.

1.2 Proposed method

We take an intuitive user interaction scheme - user specifies certain image pixels (we will call them **seed** pixels) that belong to objects, that should be segmented from each other. The task is to assign labels to all other image pixels automatically, preferably achieving the segmentation result the user is expecting to get. The task statement and input data is similar to [Boykov and Jolly 2001] and [Grady and Funka-Lea 2004], however the segmentation instrument differs.

Our method uses cellular automaton for solving pixel labelling task. The method is iterative, giving feedback to the user while the segmentation is computed. Proposed method allows (but not requires) human input during labelling process, to provide dynamic interaction and feedback between the user and the algorithm. This allows to correcting and guidance of the algorithm with user input in the areas where the segmentation is difficult to compute, yet does not require additional user effort where the segmentation is reliably computed automatically.

Important properties of our method, that we would like to outline are:

1. Capable of solving moderately hard segmentation tasks (see examples in this paper);
2. Works with images of any dimension $N \geq 1$;
3. Performs multi-label image segmentation (the computation time is not directly affected by the number of labels);
4. Is extensible, allowing construction of new families of segmentation algorithms with specific properties;
5. Interactivity - as the segmentation is refined with each iteration, user can observe the evolution and refine the segmentation "on the fly";

6. The algorithm is simple in both understanding and implementation;

7. Using cellular automata allows fast parallel implementation;

The rest of the paper is organized follows. Section 2 describes the proposed method in detail. Section 3 gives the results, section 4 provides discussion and comparison with other methods. Finally the conclusive remark is given in section 5.

2 Method details

2.1 Basic method

Cellular automata (CA) were introduced by Ulam and von Neumann [von Neumann 1966]. Since then they've been used to model wide variety of dynamical systems in various application domains, including image denoising and edge detection [Popovici and Popovici 2002], [Hernandez and Herrmann 1996]. A cellular automaton is generally an algorithm discrete in both space and time, that operates on a lattice of sites $p \in P \subseteq Z^n$ (pixels or voxels in image processing).

A (bi-directional, deterministic) cellular automaton is a triplet $A = (S, N, \delta)$, where S is an non-empty *state set*, N is the *neighborhood system*, and $\delta : S^N \to S$ is the local transition function (rule). This function defines the rule of calculating the cell's state at $t+1$ time step, given the states of the neighborhood cells at previous time step t.

Commonly used neighborhood systems N are the von Neumann and Moore neighborhoods:

- von Neumann neighborhood

$$N(p) = \{q \in Z^n : \|p-q\|_1 := \sum_{i=1}^{n} |p_i - q_i| = 1\}; \quad (1)$$

- Moore neighborhood

$$N(p) = \{q \in Z^n : \|p-q\|_\infty := \max_{i=\overline{1,n}} |p_i - q_i| = 1\}; \quad (2)$$

The cell state S_p in our case is actually a triplet $(l_p, \theta_p, \vec{C}_p)$ - the label l_p of the current cell, 'strength' of the current cell θ_p, and cell feature vector $\vec{C}_p$, defined by the image. Without loss of generality we will assume $\theta_p \in [0,1]$.

A digital image is a two-dimensional array of $k \times m$ pixels. An unlabelled image may be then considered as a particular configuration state of a cellular automaton, where cellular space P is defined by the $k \times m$ array set by the image, and initial states for $\forall p \in P$ are set to:

$$l_p = 0,\ \theta_p = 0,\ \vec{C}_p = RGB_p; \quad (3)$$

where RGB_p is the three dimensional vector of pixel's p color in RGB space. The final goal of the segmentation is to assign each pixel one of the K possible labels.

When user starts the segmentation by specifying the segmentation seeds, the seeded cells labels are set accordingly, while their strength is set to the seed strength value (more about it in section 2.3). This sets the initial state of the cellular automaton.

(a) (b) (c)

Figure 1: Segmentation of a color image. (a) Source image, (b) user-specified seeds, (c) segmentation results

At iteration $t+1$ cell labels l_p^{t+1} and strengths θ_p^{t+1} are updated as follows:

Code 1 Automata evolution rule

```
// For each cell...
for ∀p ∈ P
    // Copy previous state
    l_p^{t+1} = l_p^t;
    θ_p^{t+1} = θ_p^t;
    // neighbors try to attack current cell
    for ∀q ∈ N(p)
        if g(‖C_p − C_q‖_2) · θ_q^t > θ_p^t
            l_p^{t+1} = l_q^t
            θ_p^{t+1} = g(‖C_p − C_q‖_2) · θ_q^t
        end if
    end for
end for
```

Where g is a monotonous decreasing function bounded to $[0,1]$, we use a simple one:

$$g(x) = 1 - \frac{x}{\max \|\vec{C}\|_2}; \quad (4)$$

To supply an intuitive explanation to the pseudocode above we can use biological metaphor. We can treat pixel labelling process as growth and struggle for domination of K types of bacteria. The bacteria start to spread (grow) from the seed pixels and try to occupy all the image. That is why we called the method 'GrowCut'. The rules of bacteria growth and competition are obvious - at each discrete time step, each cell tries to 'attack' its neighbors. The attack force is defined by the attacker cell's strength θ_q, and the distance between attacker's and defender's feature vectors $\vec{C}_q$ and $\vec{C}_p$. If the attack force is greater than defender's strength - the defending cell is 'conquered' and its label and strength are changed. The result of these local competitions is that the strongest bacteria occupy the neighboring sites and gradually spread over the image.

The calculation continues until automaton converges to stable configuration, where cell states seize to change. The example of image segmentation is shown in figures 1 and 2. The method is guaranteed to converge as the strength of each cell is a increasing and bounded.

(a) (b) (c) (d) (e)

Figure 2: Bacteria evolution steps. White - 'object' labelled bacteria, black - 'background' labelled bacteria, gray - neutral territory. (a) - step 1, (b) - step 10, (c) - step 25, (d) - step 40, (e) - step 65.

2.2 Adding controllable boundary smoothness

The basic scheme is very simple and yet is able to achieve quality segmentation (see figures 1, 4, 5). But in some images, the resulting segments boundary can be ragged, see figure 3 (a). It can be acceptable, or even necessary when the task is to capture the smallest detail of the boundary (i.e. in medical applications), but this can be an unwanted artifact when editing a generic high-resolution photo. To achieve smoother boundary, we propose an extension to the automata in section 2.1. Local transition rule is modified with two additional conditions. First: the cell, that has too many enemies around - $enemies^t(p) \geq T_1$ is prohibited to attack its neighbors. Second: the cell that has $enemies^t(p) \geq T_2$ is forced to be occupied by the weakest of it's enemies, no matter the cell's strength. The enemies number is defined by:

$$enemies^t(p) = \max_{l=1,K}\Big(\sum_{q\in N(p), l_q^t \neq l_p^t} 1\Big) \tag{5}$$

The thresholds values T_1, T_2 control the boundary smoothness. Reasonable values for 2D images and Moore neighborhood range from 9 (no smoothing) to 6. The examples of segmented border for identical initial seeds and different threshold values are presented in figure 3.

(a) (b)

Figure 3: Different boundary smoothness restrictions. (a) $T_1, T_2 = 9$, (b) $T_1, T_2 = 6$.

2.3 User interaction

For the discussion on user interaction we will consider the case of two labels - object and background. To our opinion, this is the most often case of segmentation task in photo editing. However, this is only for the sake of clarity, everything said here is true for the case of labels $K > 2$ also.

The segmentation is started by specifying the initial seeds. This is done by user's strokes with 'object' and 'background' brushes (see figure 1.b - red pixels correspond to 'object' brush strokes, blue - to 'background' strokes). Each paint stroke of a defined brush sets the initial labels and strengths of seed pixels.

After the initial seeds are set, the automata evolution starts. The initial, incomplete user-labelling is often sufficient to allow the entire segmentation to be completed automatically, but by no means always. While the cell labels are being computed, the user can observe the progress and interactively correct and guide the labelling process if necessary. User editing takes the form of brushing pixels for adding new segmentation constraints. Each new paint stroke changes the states of the underlying pixels and affects the automaton evolution. This strategy allows to extract regions of interest from complex backgrounds using simple paint strokes with a mouse.

Just as in [Rother et al. 2004] it is usually sufficient to brush, roughly, just part of a wrongly labelled area. Note that correction can be done not only after the segmentation process is finished, but in 'the middle' of the segmentation computation.

One important difference from the methods based on graph cuts is that seeds do not necessarily specify hard segmentation constraints. In other words - user brush strokes need not to specify only the areas of firm foreground or firm background, but instead can adjust the pixels state continuously, making them 'more foreground' or 'a little more background' for example. This gives more versatile control of the segmentation from the user part and makes the process tolerable to inaccurate paint strokes. This is especially helpful for segmenting accurate boundaries of objects like flowers or plant leaves, like in figure 7.

The seed 'strength' is controlled by the initial strength θ' that is set by the user's brush strokes. Hard constraints specify the pixels that should not change during the evolution, which is easily achieved by setting seed cells strength to 1. Soft constraints set their initial strength values < 1 or just increase or decrease current cell strength by some value.

Unlike graph cuts, adding seeds in already correctly segmented area (i.e. near the border) can help to achieve better segmentation in troubled areas. This is especially true when accurate border in blurry or camouflaged area is estimated.

3 Results

We demonstrate our segmentation method in several examples including photo editing and medical image segmentation. We show original data and resulting segments generated by our technique, for a given set of seeds.

3.1 Medical images

Figure 4 shows segmentations that have been obtained on several medical images from the DICOM image samples [Barre n. d.] - user-specified seeds and resulting segmentation.

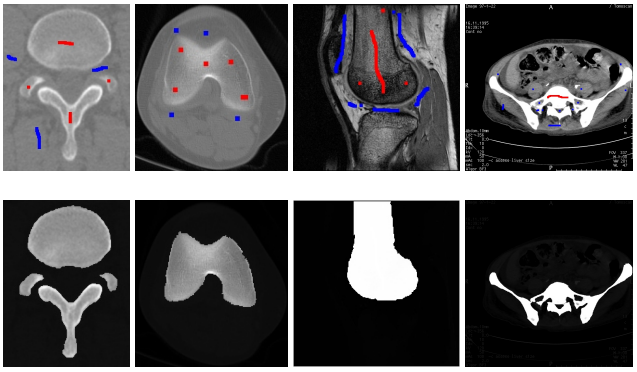

Figure 4: Medical images segmentation results.

3.2 Photo editing

In Figures 5 and 7 we show results of segmentation of several photographs.

Naturally, the hope is that the method can quickly identify the right object, with little user interaction required. If the user needs to specify too many seeds, this is not much better than a manual segmentation. The performance of an algorithm can be assessed by the amount of efforts required from the user. Thus, the results of our segmentation are shown along with seeds that we entered by the user.

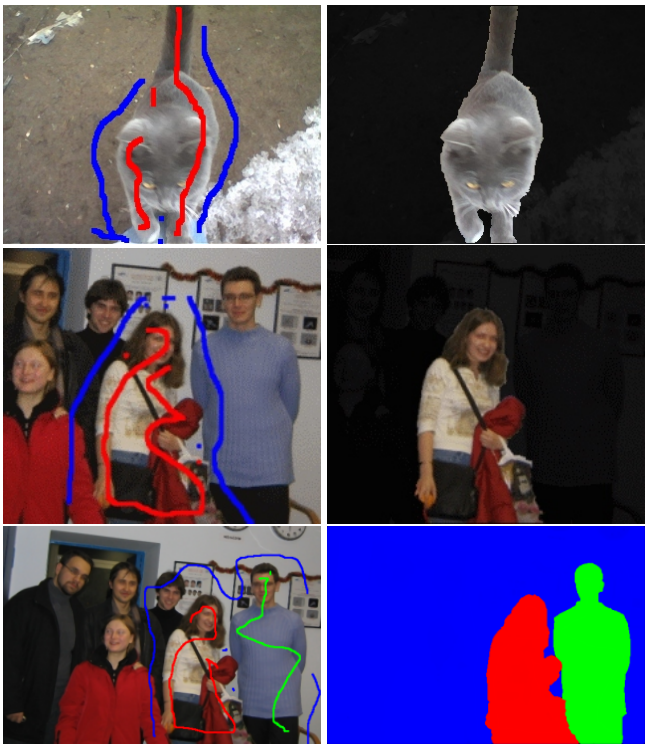

Figure 5: Generic photos segmentation results.

4 Discussion

As we have already emphasized in the introduction, our hope is to stir up the research community, motivating to search new ideas in the field of cellular automata and evolutionary computation and applying them to interactive image segmentation.

We expect that results exceeding our current can be obtained. However, our current method can already compete with elegant achievements of graph theory. In this section we will try to compare current top performing methods with ours and point out advantages and disadvantages of our scheme.

We take four methods - Graph Cuts, GrabCut, Random Walker and GrowCut and compare them by several criteria: segmentation quality, speed and convenience for the user. Accurately speaking, the methods differ seriously by the amount of information that they extract from the image. GrabCut uses most information - it computes the evolving color statistics of foreground and background and takes into account color difference between neighboring pixels. Graph Cuts differs in using color statistics collected from the user-specified seeds only, computed before the segmentation start. Random Walker uses only intensity difference between neighboring pixels. Our current GrowCut variant also does not take advantage of object color statistics, however it can be easily extended to maintain regions color statistics and use them in automaton evolution.

4.1 Segmentation quality

Judging by our own experiments, Graph Cuts, GrabCut (we used our own implementation of both methods) and GrowCut are all able to produce high quality segmentation in natural and medical images, so in this case mostly the amount user effort required should be the measure of performance. However, there are some interesting features in these methods results, mentioned in next paragraph. Concerning Random Walker, we've not been able to test the method by ourselves, but the results presented in the paper [Grady and Funka-Lea 2004] and in the presentation poster are very close to what we've been able to achieve on similar images.

Generally, methods based on graph cuts tend to produce smoother (actually, shorter) boundary than our basic method. However our modified algorithm with controllable boundary smoothness (see section 2.2) makes this difference less perceptible.

Several authors mention that segmentation methods, based on minimum graph cuts tend to cut away small isolated image segments [Grady and Funka-Lea 2004], [Veksler 2000]. We performed some investigation on how often this problem actually arises. In our experience, this situation occurs only in cases when regional term $R(a)$ in eq. (1) in [Boykov and Jolly 2001] becomes insignificant compared to boundary term $B(A)$. One such example is shown in figure 6, one can identify both smoother boundary of the graph cuts segmentation, and isolated background segments on the right part of the image. This can usually be avoided by tuning the relative importance coefficient λ.

(a) (b) (c)

Figure 6: Graph cuts and GrowCut segmentation difference for same seeds. The Graph cuts produce smoother boundary, but has problems with the background on the right image part, where two isolated clusters instead of the whole region were segmented

It also should not be forgotten that both Random Walker and GrowCut are capable of solving multi-labelling tasks.

Random Walker boasts of creating segments connected to the initial seeds always. So as GrowCut does. Graph Cut potentially can (and sometimes does) create segments not connected to initial seeds, if the regional term $R(A)$ is stronger than boundary term $B(A)$. Same applies GrabCut. However, whether this an advantage or a drawback is probably the question of exact segmentation task.

4.2 Speed

The method speed is surely important. The reaction time to the user input should be small enough not to produce discomfort and irritation, otherwise the method cannot be called interactive. An important measure is not only how fast the initial segmentation is achieved, but also how fast can the segmentation be recomputed, when user adds more seeds to refine the initial segmentation.

The fastest are algorithms based on graph cuts. The initial segmentation computation time for Graph Cuts reported by the authors is 'less than a second' for 'most 2D images (up to 512x512)' on '333MHz Pentium III'. This looks terrific, but judging from the Intel processor manufacturer site, the frequency for Pentium III processor family ranges from 450 MHz to 1.33 GHz. This suggests that actually running times are given for 1.33 GHz processor, which is still a very good result. GrabCut processing time was given for 450x300 image and equals to 0.9*sec* on 2.5GHz CPU with 512 Mb RAM. Recomputing segmentation after new user-added seeds is even faster than that. However, our own implementation of these methods, based on Vladimir Kolmogorov's publicly available code [Boykov and Kolmogorov 2004] was not able to reach this speed.

Random walker cannot boast of such speed, for the computation time for 256x256 image on **dual** Intel Xeon 2.4GHz processor with

1GB RAM is almost 5 (4.831) seconds. No indications that adding more seeds will need less time to compute a refined segmentation were found in the method description. But again, Random Walker solves a harder task of multi-labelling.

Our current implementation goes somewhere in between. It takes about 4 seconds on 256x256 image on 2.5GHz processor. However, this is the *total* automaton evolution time (till complete stop). Usually the desired segmentation result is obtained much earlier (about 1.5-2 seconds) and the later automaton evolution does not change the segmentation and can be stopped by the user, when he is satisfied with the result. Another important issue is that our methods allows user intervention at any iteration step, so adding additional seeds throughout the computation does not increase the whole working time. Again - our method solves the task of multi-labelling.

4.3 User convenience

In our opinion, most important performance measure is the convenience of the image segmentation instrument, based on the algorithm. Or, in other words, how much user effort is needed to achieve satisfying segmentation result.

The results reported by Boykov et al. [Boykov and Jolly 2001] say that it takes from 1 to 3-4 minutes to segment an image with Graph Cuts, depending on the scene. Our own experiments have shown that usually it takes about 1-2 minutes for the user, armed with GrowCut to segment an image like shown in figures 4, 5, 7.

5 Conclusion and future work

In this paper, we have proposed a novel algorithm for interactive image segmentation, based in cellular automaton. Experiments have shown, that user effort required for segmenting generic photos and medical images is rather modest, no harder than in state-of-the-art graph-based methods like Graph Cuts, GrabCut and Random Walker.

Proposed method combines the advantages, distributed between the mentioned methods (multi-label segmentation, N-dimensional images processing, speed high enough for interactive segmentation), and offers more - algorithm extensibility by varying the automaton evolution rule, more interactivity and user control of the segmentation process.

Future work will concentrate on exploring theoretical properties of the algorithm, research into exploiting color statistics during segmentation, and practical issues like faster implementation.

Figure 7: More photo segmentation results.

6 Acknowledgements

We gratefully acknowledge valuable comments from the Graphicon reviewing committee members, Yuri Boykov and Olga Veksler. We also would like to thank our lab's head Yuri Bayakovsky for attracting our attention to the field of cellular automata some years ago.

References

BARRE, S. Medical image samples. http://www.barre.nom.fr/medical/samples/.

BOYKOV, Y., AND JOLLY, M.-P. 2001. Interactive graph cuts for optimal boundary and region segmentation of objects in n-d images. In *Proc. of the International Conference on Computer Vision*, vol. 1, 105–112.

BOYKOV, Y., AND KOLMOGOROV, V. 2004. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Trans. Pattern Anal. Mach. Intell. 26*, 9, 1124–1137.

BOYKOV, Y., VEKSLER, O., AND ZABIH, R. 2001. Fast approximate energy minimization via graph cuts. *IEEE Trans. Pattern Anal. Mach. Intell. 23*, 11, 1222–1239.

GRADY, L., AND FUNKA-LEA, G. 2004. Multi-label image segmentation for medical applications based on graph-theoretic electrical potentials. In *ECCV Workshops CVAMIA and MMBIA*, 230–245.

HEIMANN, T., THORN, M., KUNERT, T., AND MEINZER, H.-P. 2004. New methods for leak detection and contour correction in seeded region growing segmentation. In *20th ISPRS Congress, Istanbul 2004, International Archives of Photogrammetry and Remote Sensing*, vol. XXXV, 317–322.

HERNANDEZ, G., AND HERRMANN, H. J. 1996. Cellular automata for elementary image enhancement. *CVGIP: Graphical Model and Image Processing 58*, 1, 82–89.

MOGA, A., AND GABBOUJ, M. 1996. A parallel marker based watershed transformation. In *ICIP96*, II: 137–140.

MORTENSEN, E. N., AND BARRETT, W. A. 1998. Interactive segmentation with intelligent scissors. *Graphical Models and Image Processing 60*, 5, 349–384.

MORTENSEN, E. N., AND BARRETT, W. A. 1999. Toboggan-based intelligent scissors with a four-parameter edge model. In *CVPR*, 2452–2458.

POPOVICI, A., AND POPOVICI, D. 2002. Cellular automata in image processing. In *Fifteenth International Symposium on Mathematical Theory of Networks and Systems*.

REESE, L. 1999. *Intelligent Paint: Region-Based Interactive Image Segmentation.* Master's thesis, Department of Computer Science, Brigham Young University, Provo, UT.

ROTHER, C., KOLMOGOROV, V., AND BLAKE, A. 2004. Grabcut - interactive foreground extraction using iterated graph cuts. *Proc. ACM Siggraph.*

VEKSLER, O. 2000. Image segmentation by nested cuts. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, vol. 1, 339–344.

VON NEUMANN, J. 1966. *Theory of Self-Reproducing Automata*. University of Illinois Press. Ed. and Completed by A. Burks.

7 About the authors

Vladimir Veznevets is a research fellow at Graphics and Media Laboratory of Moscow State Lomonosov University. He graduated with honors Faculty of Computational Mathematics and Cybernetics of Moscow State University in 1999. He received his PhD in 2002 in Moscow State University also. His research interests include image processing, pattern recognition, computer vision and adjacent fields. His email address is vvp@graphics.cs.msu.ru.

Vadim Konouchine is a third year student of Moscow State Lomonosov University, Faculty of Computational Mathematics and Cybernetics. His research interests lie in the fields of statistics and image processing. His email address is kvad@inbox.ru.