A Statistical Approach to Interactive Image Segmentation

Ahmed A. Hussein
School of Information Science and Engineering
Northeastern University
Liaoning 110819, China
Email: ah_h72@yahoo.com

Xiaochun Yang
School of Information Science and Engineering
Northeastern University
Liaoning 110819, China
Email: yangxc@mail.neu.edu.cn

***Abstract*—Interactive segmentation aims to separate an object of interest from the rest of an image. This problem in computer vision is known to be hard, and very few fully automatic vision systems exist which have been shown to be accurate and robust under all sorts of challenging inputs. Most of the previous works require users to trace the whole boundary of the object. When the object has a complicated boundary, or the object is in a highly-textured region, users have to put great effort into iteratively correcting the selection. To solve this problem, most researchers rely on computing cost function for the purpose of control on the boundary of the region. Minimizing cost function yields a large, sparse system of linear equations. This paper presents a new interactive algorithm by introducing new equations for image foreground background separation. In addition we propose *Edge-stopping* function, a powerful tool for automatically controlling the update of seed colors with the boundaries of the image. Results show that the new method works effectively and provides an alternative computational algorithm for building interactive image editing tools.**

Keywords: interactive segmentation; interactive learning; denoising.

I. Introduction

Interactive image segmentation is the process of extracting an object in an image with additional hints from the user. This problem in computer vision is known to be hard, and very few fully automatic vision systems exist which have been shown to be an ill-posed problem due to the fact that there is (i) no clear definition of a correct segmentation; and (ii) no agreed-upon objective measure that defines the goodness of a segment. For this reason a number of interactive systems have been developed which allowed users to help the vision algorithm to achieve the correct solution by giving hints [1].

Interactive segmentation aims to separate an object of interest from the rest of an image. It is a classification problem where each pixel is assigned one of two labels: foreground (F) or background (B). The interaction comes in the form of sets of pixels marked by the user by help of brushes to belong either to foreground (F) or background (B).

The algorithms used in previous work can be categorized into the following five main approaches depending on their methodology and user-interfaces [2]:

MagicWand is probably the simplest technique. Given a user-specified *seed* point (or region), a set of pixels is computed which is connected to the seed point, where all pixels in the set deviate, from the color of the seed point, only within a given tolerance [3].

Intelligent Scissors allows a user to choose a *minimum cost contour* by roughly tracing the object's boundary with the mouse. As the mouse moves, the minimum cost path from the cursor position back to the last seed point is shown. One problem is that this technique is not effective for objects with a long boundary (e.g. a tree with many branches) [4].

Segmentation in Discrete Domain gives some user constraints in the form of foreground and background, brushes, i.e. regional constraints, the optimal solution is computed efficiently with graph cut [1], [6], [7]. Boykov and Jolly were the first to formulate a simple generative MRF model in discrete domain for the task of binary image segmentation [5]. This basic model can be used for interactive segmentation.

Segmentation in Continuous Domain finds a segmentation that minimizes a boundary (surface) under some metric, typically image-based Riemannian metric. Traditionally, techniques such as level-sets were used, which however are only guaranteed to find a local optimum [8], [9].

Paint-Selection is very similar to the brush interface. The key difference is that a new segmentation is visualized after each mouse movement [10].

Interest in interactive vision systems has grown in the last few years, which has led to a number of workshops and special sessions in vision, graphics, and user-interface conferences.

This paper presents a new interactive algorithm by introducing new equations for image foreground background separation. Users select image regions by directly painting the object of interest with a paint brush. Unlike conventional painting operations, users need not paint over the whole object. Instead, the selection can be automatically expanded from users' paint brush and aligned with the object boundary.

Most of the previous works require users to trace the whole boundary of the object. When the object has a complicated boundary, or the object is in a highly-textured the region, users have to put great effort into iteratively correcting selection.

To solve this problem, most researchers rely on computing cost function for the purpose of control on the boundary of the region. Minimizing cost function yields a large, sparse system of linear equations.

In this paper, we introduce *edge-stopping* function as a

978-1-61284-774-0/11/$26.00 ©2011 IEEE

powerful tool for automatically controlling the update of seed colors with the boundaries of the image. Computation of edge-stopping function depends on the geometry and structure of the image.

Our algorithm has the following advantages: (i) enabling interchangeability. The user is free to choose the colors for the purpose of indicating regions. In addition, the algorithm does not require indicating the background of the image, as we will show in the results; (ii) introduction of the edge-stopping function instead of the cost function; and (iii) image composition. Results show that the new method works effectively and provides an alternative computational algorithm for building interactive image editing tools.

The rest of the paper is organized as follows: Section 2 explains how our algorithm update seed color. Section 3 demonstrates experiments. Finally, we conclude with a discussion in Section 4.

II. SEED COLOR UPDATES

The algorithm assumes the seed color to be a noise. Colors are given in a region Ω_c in such that $\Omega_c \ll \Omega$. The main idea is instead of removing the noise, the removal noise algorithm will be developed to update the seed color (noise) to its neighbors with respect the boundaries of the region. The edge-stopping function [12] will be introduced to automatically control the updates of that seeds with the boundaries of the image. Fig. 1 explains the framework overview of our algorithm.

A. Image Gradient

First-order derivatives of a digital image are based on various approximations of the 2-D gradient. The gradient of an image $I(x, y)$ at location (x, y) is defined as the vector.

$$\nabla I = \begin{pmatrix} I_y \\ \\ I_x \end{pmatrix} = \begin{pmatrix} \frac{\partial I}{\partial x} \\ \\ \frac{\partial I}{\partial y} \end{pmatrix}. \tag{1}$$

It is well known from vector analysis that the gradient vector points in the direction of maximum rate of change of I at coordinates (x, y). An important quantity in edge detection is the magnitude of this vector, denoted $\|\nabla I\|$, where

$$\|\nabla I\| = \sqrt{\left(\frac{\partial I}{\partial x}\right)^2 + \left(\frac{\partial I}{\partial y}\right)^2}. \tag{2}$$

This quantity gives the maximum rate of increase of $I(x, y)$ per unit distance in the direction of ∇I. It is a common (although not strictly correct) practice to refer to ∇I also as the *gradient*. The computation of the gradient of an image is based on obtaining the partial derivatives $\frac{\partial I}{\partial x}$ and $\frac{\partial I}{\partial y}$ at every pixel location [11].

Equation 2 can be discrete on a square lattice, with brightness values associated to the center lattice and its vertices, as shown in Fig. 2. Although we used the 8-neighbors for the element $I(x, y)$ in our experiments, in this section we only use the 4-neighbors in order to simplify the idea. The symbols N and S refers to forward partial derivative and backward

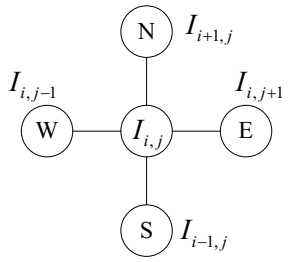


Fig. 2. The structure of the Pixel ($I_{i,j}$) with its 4-nighbors.

partial derivative to $\frac{\partial I}{\partial x}$ respectively. And E and W refers to forward partial derivative and backward partial derivative to $\frac{\partial I}{\partial y}$ respectively. Hence we can rewrite Equation 2 as:

$$\begin{aligned} \nabla I_{i,j}(\mathsf{N}) &= (I_{i+1,j} - I_{i,j}) \\ \nabla I_{i,j}(\mathsf{S}) &= (I_{i-1,j} - I_{i,j}) \\ \nabla I_{i,j}(\mathsf{E}) &= (I_{i,j+1} - I_{i,j}) \\ \nabla I_{i,j}(\mathsf{W}) &= (I_{i,j-1} - I_{i,j}) \end{aligned} \tag{3}$$

Note that we neglected the power for each equation, because the equations will rise to the power 2 in the subsequent section.

B. Edge-Stopping Function

We would want to encourage replacing each pixel of the region with seed pixel with respect the boundaries. This could be achieved by setting the interior of each region to be 1 and 0 at the boundaries. The segmenting would then take place separately in each region with no interaction between regions. The region boundaries would remain sharp. The edge-stopping function $g(\nabla I)$ in Equations 4 and 5 is a powerful tool to do that. The images in this paper adopted edge-stopping function proposed in [12].

$$g(\nabla I) = e^{-\left(\frac{\|\nabla I\|}{K}\right)^2}. \tag{4}$$

$$g(\nabla I) = \frac{1}{1 + \left(\frac{\|\nabla I\|}{K}\right)^2}. \tag{5}$$

The scale-spaces generated by these two functions are different: the first privileges high-contrast edges over low-contrast ones, the two privileges wide regions over smaller ones. K is the local contrast (difference in the image intensities on the left and right) of the edge and fixed by hand at some fixed value.

Consider the image intensity differences, Fig. 3(a), between the pixel $I_{i,j}$ and its neighbors within the desired region. These neighbors differences will be small, zero mean, and normally distributed. Hence the value of each edge-stopping function in the direction N, S, E, and W equals to 1. In Fig. 3(b), the neighbor difference will not be normally distributed between the $I_{i,j}$ and its neighbor E because the pixel E is included within a boundary (intensity discontinuity). So the pixel E will skew the estimate of $I_{i,j}$ and the edge-stopping function in E equals to 0.

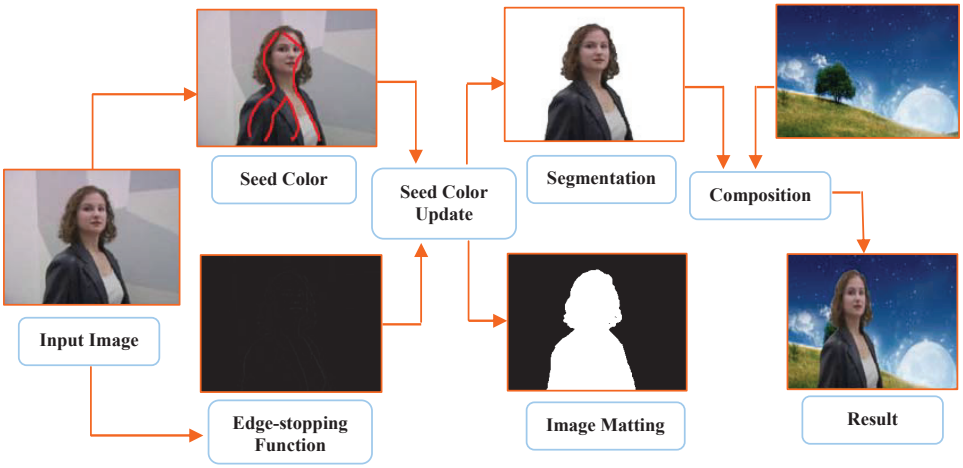
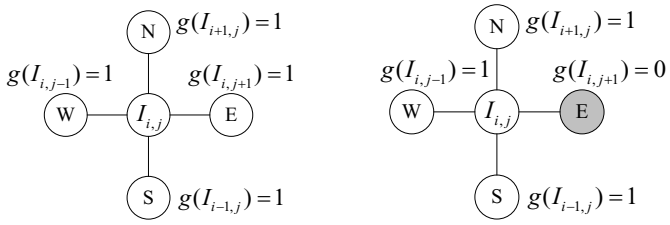


Fig. 1. Framework overview.



(a) The pixel normally distributed.

(b) The pixel not normally distributed (intensity discontinuity).

Fig. 3. Local neighborhood of pixel ($I_{i,j}$).

Fig. 4. A text image segmentation.

C. Updating Seed Colors

In this step, we deal with the seeds colors that have been set by the user as noise. Instead of deleting noise(seed) our algorithm works on the update of noise to all the neighboring pixels with respect the boundaries of the region. A classical formulation by [12] to remove noise from images will be developed to use it to segment any images. We present equation (6), a general and fast equation to solve the problem of segmenting images. The execution depends on the geometry and structure of the image. Given by its gradient information, the noise or seed color is then updated by applying our equation that annotates a few seed color scribbles provided by the user. Note that modifying the image according to this equation is equivalent to filtering the image with a mean filter except the pixel that skew the estimate of pixel $I_{i,j}$. The goal of this stage is the update of the seed color information in Ω_{seed}, provided by the user via color strokes, to the rest of Ω.

$$I_{i,j}^{t+1} = \left(\frac{\sum_{s=1}^{p} (g(\nabla I_{(i,j),s}) \cdot \nabla I_{(i,j),s})}{\sum_{s=1}^{p} g(\nabla I_{(i,j),s})} \right)^{t}, \qquad (6)$$

where $I_{i,j}^{t+1}$ is a discretely sampled image, $I_{i,j}$ denotes the pixel position in the discrete, and t denotes discrete time steps(iterations), s represents the spatial neighborhood of pixel $I_{i,j}$ (N, S, E, and W), and p is the number of neighbors (In this illustration p equals to four, except at the image boundaries).

III. Experimental Results

In this section we develop a system to examine the proposed algorithm and prove the accuracy and efficiency of our core algorithm. The algorithm has been implemented on Windows Vista, with Borland Delphi language version 7.0. The memory capacity of the computer is 3.00 GB and the CPU is Intel Core^{TM} 2Duo 2.00GHz.

A. Effect of Image Segmentation

We have tested our system on a variety of images. Fig. 4 shows that our system is capable of removing unwanted complicated objects from the image and pulling out complicated features such as a text with low to moderate user effort.

Fig. 4 shows a text image used for segmenting, 475×315 pixels. A text image is marked with some seed color by the user (The bottom left of the figure). The algorithm produces a image matting (The top right of the figure) and the image extraction (the bottom right of the figure). To perform a extracted image process, the running time of the system is less than 2.09 seconds and k = 30.

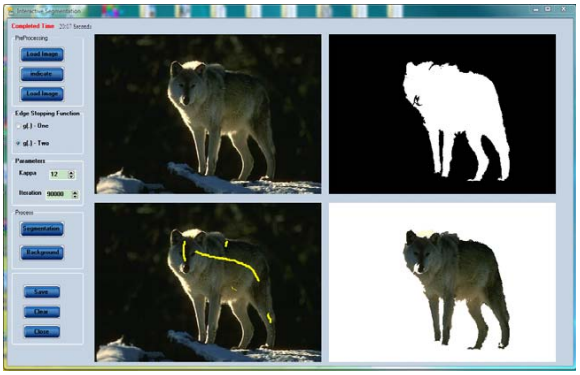


Fig. 5. A wolf image segmentation.

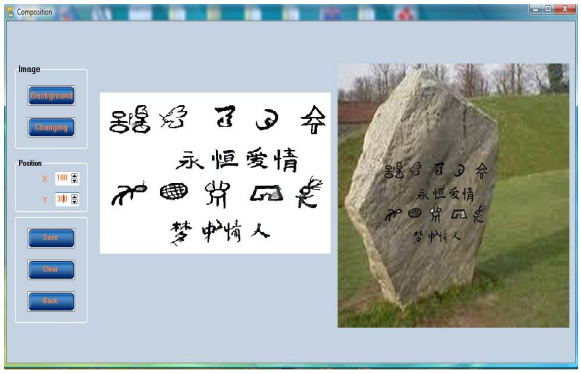


Fig. 6. A rock image composition.

Fig. 5 shows a wolf image used for segmenting, 481×321 pixels. Two typical problems are faced here. One is the presence of a highly textured area - the body of the wolf is occluded by fur, which has to be skipped smoothly. The edge-stopping function in Equation 4 could successfully skip the high-contrast edge over low-contrast ones.

The second problem occurs where the hair meets the edge at the boundary of the wolf's body. The problem is of multiple close edges, generated by the changing color of the hair. The user wants, of course, the specific edge separating the hair from the background. All he has to do is to correctly fix the k value by hand at some fix value.

A wolf image is marked with some seed color by the user (The bottom left of the figure). The algorithm produces a image matting (The top right of the figure) and the image extraction (the bottom right of the figure). To perform a extracted image process, the running time of the system is less than 20.07 seconds and k = 12.

B. Effect of Image Composition

The goal of image composition is to combine objects or regions from various still photographs to create a seamless, believable, image which appears convincing and real. However, such methods suffer from overflow and underflow problems, i.e., the resulting image may contain pixels with color values out of the displayable or printable range (e.g. $[0, 255]$ for 8-bit images. Yang et al [13] propose a gamut fitting method to solve this out-of-gamut problem in image composition.

Fig. 6 shows an example of image composition using our interactive segmentation algorithm. We first extract foreground object from original image and then combine the extracted foreground with a new background to form a new image. The right part in Fig. 6 shows a rock image composition with the result of the left part in Fig. 4.

IV. Conclusion

In this paper we present a new system for interactive image segmentation. We formulated a new and easy-to-understand equations to solve the problem of segmenting natural images. The approach suggests a method that helps graphic artists segment an image with less manual effort by selected seed color. Our system does not need to compute the cost function to segment the natural images. Instead, we introduce *edge-stopping* function as a powerful tool for automatically controlling the update of seed colors with the boundaries of the image. In demonstration, we show that the new method works effectively and provides an alternative computational algorithm for building interactive image editing tools.

Acknowledgment

The work is partially supported by the National Natural Science Foundation of China (Nos. 60973018, 60973020) and the Fundamental Research Funds for the Central Universities (Nos. N090504004, N100704001).

References

[1] H. Nickisch, P. Kohli, C. Rother, and C. Rhemann, Learning an interactive segmentation system. In Proc. of the Seventh Indian Conf. on Computer Vision, Graphics and Image Processing (ICVGIP), pp. 274-281, 2010.

[2] C. Rother, V. Kolmogorov, Y. Boykov and A. Blake, Interactive foreground extraction using graph cut. Microsoft Technical Report: MSR-TR-2011.

[3] Adobe Systems Incorp. *Adobe Photoshop User Guide*. 2002.

[4] E. N. Mortensen and W. A. Barrett, Intelligent scissors for image composition. In Proc. ACM Siggraph, pp. 191-198, 1995.

[5] Y. Boykov and M-P. Jolly, Interactive graph cuts for optimal boundary and region segmentation of objects in N-D images. In Proc. of the Eighth IEEE International Conference on Computer Vision (ICCV), pp. 105-112, 2001.

[6] X. Bai and G. Sapiro, Geodesic matting: A framework for fast interactive image and video segmentation and matting. *Int. J. Computer Vision*, 82(2): 113-132, 2009.

[7] V. Gulshan, C. Rother, A. Criminisi, A. Blake, and A. Zisserman, Geodesic star convexity for interactive image segmentation. In Proc. IEEE Conf. Computer Vision and Pattern Recog., 2010.

[8] V. Kolmogorov and Y. Boykov, What metrics can be approximated by geo-cuts, or global optimization of length/area and flux. In Proc. IEEE Int. Conf. on Computer Vision (ICCV), pp. 564 - 571, 2005.

[9] M. Unger, T. Pock, D. Cremers, and H. Bishop, Tvseg-interactive total variation based image segmentation. In Proc. British Machine Vision Conf. (BMVC), 2008.

[10] J. Liu, J. Sun and H.-Y. Shum, Paint selection. In Proc. ACM Siggraph, 2009.

[11] R. C. Gonzalez, R. E. Woods, S. L. Eddins, Digital image processing using matlab. *Pearson Prentice Hall*. 2004.

[12] P. Perona, J. Malik, Scale-space and edge detection using ansitropic diffusion. *IEEE Trans. Pattern Anal Machine Intell*, vol. 12, pp. 629-639, 1990.

[13] W. Yang, J Cai, and J. Zheng, Solving the out-of-gamut problem in image composition. In Prof. of 17th IEEE Int. Conf. on Image Processing (ICIP), pp. 3977-3980, 2010.