Fast Interactive Image Segmentation by Discriminative Clustering

Dingding Liu
Department of Electrical Engineering
University of Washington
Seattle, WA, USA, 98105
liudd@u.washington.edu

Kari Pulli
Nokia Research Center
955 Page Mill Road
Palo Alto, CA, USA, 94304
Kari.Pulli@nokia.com

Linda G. Shapiro
Department of Electrical Engineering
University of Washington
Seattle, WA, USA, 98105
shapiro@cs.washington.edu

Yingen Xiong
Nokia Research Center
955 Page Mill Road
Palo Alto, CA, USA, 94304
yingen.xiong@nokia.com

ABSTRACT

We propose a novel and fast interactive image segmentation algorithm for use on mobile phones. Instead of using global optimization, our algorithm begins with an initial over-segmentation using the mean shift algorithm and follows this by discriminative clustering and local neighborhood classification. This procedure obtains better quality results than previous methods that use graph cuts on over-segmented regions or region merging based on maximal similarity, yet its running time is smaller by an order of magnitude. We compare and analyze the strengths and limitations of the three approaches and have implemented our algorithm as part of an interactive object cut out application running on a mobile phone.

Categories and Subject Descriptors

I [**Computing Methodologies**]: Image Processing and Computer Vision Application

General Terms

Algorithms

Keywords

Segmentation, Image editing, Mobile phones

1. INTRODUCTION

Image segmentation is an important and unsolved problem in computer vision. Due to the difficulty of fully automatic segmentation on a wide range of different images, interactive image segmentation algorithms have received much attention recently [19, 22, 5, 4, 12, 13, 24, 16, 2, 3]. When they are applied to image editing tasks, the goal is to cut out an object from its background. Generally, some user input such as strokes, seeds or bounding box is used to guide the algorithms to label the unmarked regions correctly. However, the previous work does not satisfy the needs of mobile phones users, since the desired balance of performance, speed, and feasibility required for such users is not provided.

The $gPb - owt - ucm$ algorithm in [2] gets the contour information from the globalized Probability of boundary (gPb) detector [15, 14], applies the oriented watershed transform (owt) [3], and represents the region hierarchy as an ultrametric contour map (ucm) [1]. The segmentation trees generated by this algorithm provide a natural starting point for interactive segmentation by following the procedure of [3]. However, the constrained segmentation algorithm in [3], similar to the $gPb - owt - ucm$ algorithm, needs the input from the Pb operator [15], which involves supervised training using manually labeled images as ground truth and expensive computation of textons using filter banks. Most of the emphasis in this line of work is on the precision of segmentation algorithms and benchmarking of specific databases. However, in object cut-out applications, a few pixels off may not be important visually. Furthermore, as no existing algorithm can be absolutely accurate on all images and speed is critical for mobile phone applications, we prefer to be faster and let users modify their inputs to improve the results.

Efficient energy minimization algorithms, such as graph cuts, are widely used to label the unmarked pixels in interactive segmentation [19, 5, 4, 12, 13, 24]. Much work has been done to improve graph cut segmentation speed [12, 13] and segmentation quality [24]. However, graph-cut algorithms have several limitations. In the multilevel banded approach [13], the quality of the graph-cut segmentation is assumed. However, later research [24, 16] and our own studies show that methods using graph cuts on pixel color features do not always perform well, especially on textured images [24]. Szeliski et al. [21] point out that the lowest energy solution is not always the best one for a vision problem. Moreover, Lombaert et al. [13] admit that the coarsening and uncoarsening may cause errors on high-frequency features in the image. On the other hand, Wang [24] incorporates more fea-

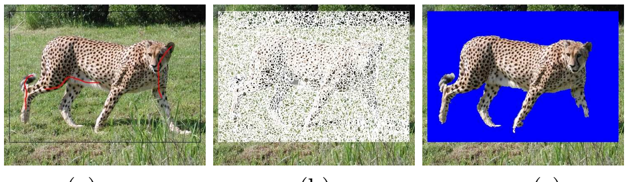
(a) (b) (c)

Figure 1: Framework. (a) Original image. (b) Over-segmented image. (c) Resultant segmentation.

tures such as texture and more complex model-selection to improve the performance of graph-cut-based segmentation. Ning et al. [16] merge the over-segmented regions according to the maximal similarity rule. They show that graph-cuts on pre-segmented regions work much better than on pixels. However, these methods [24, 16] do not take the speed performance into consideration.

To further improve the speed and achieve better performance of the interactive segmentation algorithms, we propose a novel algorithm that avoids the expensive global optimization such as done in graph cut. Our algorithm, which is related to work by Li et al. [12] and Ning et al. [16], does not involve supervised training of classifiers or extra texture descriptors. The procedure of the proposed algorithm is illustrated in Fig. 1. The user first chooses which object she wants to segment, using two points to specify a bounding box and strokes to indicate foreground and possibly background. Then the image is over-segmented by the mean-shift algorithm, the regions are clustered according to the user input, and the labelling is refined using local neighbourhood information. To better utilize the user input, our algorithm samples the regions containing the bounding box as background if it is not explicitly indicated by the user as foreground. In most cases, this allows less user input by better utilizing the bounding box information in a different way than Lempitsky et al. [11], but in a more robust way than Rother et al. [19]. Furthermore, we do not require the user to draw the bounding box “not too loose, but sufficiently tight,” as described in [11]. Compared to [19], we extract regions that contain the boundary instead of just a strip of pixels near it.

Our algorithm offers a better alternative that makes better use of the global and local information than simply running graph cuts on regions. It is conceptually simple and easy to implement. We do algorithm analysis and show experimental results that the proposed algorithm is much faster than the graph-cuts based method [12] and the similarity-based region merging algorithm [16], and performs equally well and often better than these algorithms. After improving the speed by an order of magnitude, our algorithm also allows users to quickly and iteratively adjust their inputs and re-do the segmentation if they are not satisfied with the results, instead of waiting for a long time and being surprised by unsatisfactory segmentations.

Our main contributions are that we (i) propose a novel algorithm that is an order of magnitude faster than previous methods and achieves better results; (ii) analyze the effect of the image contents on the speed of our algorithm and compare the most related ones [12, 16]; (iii) provide a better way to fully utilize the user input of corner points of a bounding box for the object and a few strokes; (iv) implement the algorithm on a mobile phone.

Section 2 explains our algorithm in detail. Section 3 analyzes our algorithm and the two most related algorithms [12, 16]. It explains why our algorithm is the fastest among the three. Section 4 compares the experimental results of the three algorithms with respect to segmentation speed, quality and robustness. Section 5 concludes the paper and briefly describes future work.

2. ALGORITHM

We first describe our approach to pre-segmentation using mean-shift, then explain the key part of our algorithm, discriminative clustering by sampling the regions marked by limited user input, and finally describe the local neighborhood classification. The steps of the algorithm are illustrated in Fig. 2.

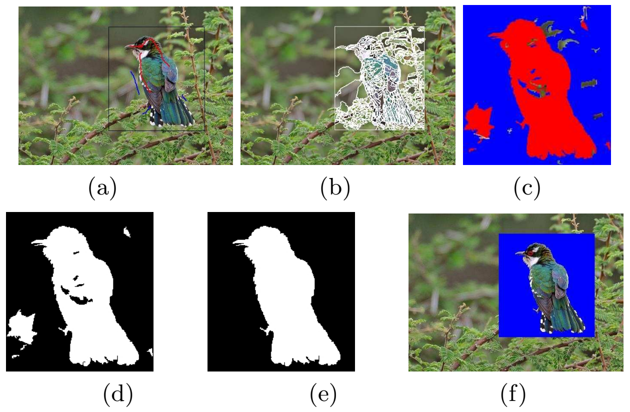
(a) (b) (c) (d) (e) (f)

Figure 2: Steps of the proposed algorithm: (a) Original image with user input. (b) Over-segmentation by the mean-shift algorithm. (c) Result after discimminative clustering, with red indicating foreground, blue indicating background, and original color indicating ambiguous regions. (d) Result after merging ambiguous regions from (c). (e) Result of local neighborhood pruning. (f) Final result on color image with bounding box shown.

Similar to other region-based methods [12, 16], we assume that the presegmentation does not over-merge the regions and that the user has marked most of the distinctive features of the image.

2.1 Pre-segmentation by the mean-shift algorithm

In our experiments, we let the user pick the upper left and lower right of the bounding box for the object to be cut out and then add the strokes to specify the foreground region and possibly the background as in Fig. 2 (a). Since the bounding box can serve as the background sampling, in most cases the user only needs to indicate the foreground, and only if the background is complex, a few background strokes may be added.

Similar to [12] and [16], the selected image is first over-segmented as in Fig. 2 (b) using the mean-shift algorithm [7]. There are three reasons for our choice of the mean-shift algorithm for the initial over-segmentation. First, we have observed that it preserves the boundaries better than other methods [8, 18]. Second, its speed has been improved significantly in recent years [9, 17]. Third, there are less parameters to tune, and our algorithm is not sensitive to the change of parameters as long as they are within a reasonable range.

2.2 Merge regions by discriminative clustering

This step is the most critical step that makes our algorithm an order of magnitude faster than other methods.

After the user input is given, the algorithm calculates the mean color of marked foreground and background regions. Suppose there are N_f foreground regions M_f marked by the user, and N_b marked background regions M_b. For each region $A_i \in M_f$ and $B_j \in M_b$, their mean colors μ_{A_i} and μ_{B_j} in CIELab color-space are computed. The minimal difference between the mean colors of the marked foreground and background regions is also computed to be taken as the threshold d_{thresh} for later clustering usage. That is, $d_{thresh} = min(\mu_{A_i} - \mu_{B_j})$. If d_{thresh} itself is smaller than a threshold λ, then it is doubled. The intuition is that if some marked foreground and background segments have very similar colors, we should allow more ambiguous regions to be further processed (Section 2.3). In our implementation, λ is set to 2.

Two three dimensional kd-trees are constructed, one for each type of marked regions. In the foreground kd-tree, each node stores the mean color of one marked foreground region, and in the background kd-tree, each node stores the mean color of one marked background region.

For each unmarked region R_u, the algorithm takes its mean color as a query point in the 3D space and does a nearest neighbor search in those two kd-trees. This quickly returns the marked regions A_{nn} and B_{nn} that have the most similar mean color to it, and the color differences d_{uf} and d_{ub}.

Let $diff_d = d_{uf} - d_{ub}$. If $diff_d > d_{thresh}$, R_u is marked as background. If $diff_d < -d_{thresh}$, R_u is marked as foreground. The output of this step on the bird image is Fig. 2 (c), where the foreground regions are marked red, background regions are blue, and the ambiguous regions are just marked with their original colors. We can see that some regions with dark green remain ambiguous.

2.3 Local neighborhood region classification and pruning

After initial cluster merging, there may still be unmarked regions that are not classified as either foreground or background. Whereas in the first step we classify the unmarked regions only by their mean colors without considering the spatial information, we now utilize the local neighborhood information to better prune the segmentation result. There are two steps:

1. If there are N_u remaining unmarked regions from the first processing step, each of them is assigned the label of the most similar of its neighboring regions in terms of their mean color. If the most similar neighboring region is also an unmarked region, they are merged together to become a new unmarked region and the process repeats. If there is a tie again in terms of the most similarly labeled neighboring region, the label of the region that has the most similar color variance is used. For the bird image, this step generates the image shown in Fig. 2 (d), with white indicating foreground and black indicating background.
2. After all the regions have been marked as either foreground or background, we apply a connected component algorithm [20] to find isolated foreground or background regions that are surrounded by regions of the opposite labeling. We specify a set of rules to decide whether the isolated regions' label should be changed. They are changed to the opposite label only when the following conditions are satisfied.
   (a) The region was not marked by the user.
   (b) The region is not the biggest region with that label.
   (c) The region is smaller than its surrounding regions.

For the bird image, this step generates the image Fig. 2 (e). This mask is used to cut out the object as shown in Fig. 2 (f).

3. ALGORITHM ANALYSIS AND COMPARISON

Given the same set of user input, we explain why this clustering-based algorithm on pre-segmented images has an absolute advantage over [12, 16] in terms of speed, and then show actual measurements in Section 4.

3.1 Comparison to graph cut with presegmentation

Li et al. [12] formulate the object-cutout problem as a graph-cut problem where the nodes are segmented regions from the pre-segmentation using the watershed algorithm [23]. A graph $G = \langle V, E \rangle$ is constructed over the nodes V, and the edges E are the set of all arcs connecting adjacent regions. The labeling problem is to assign a unique label $x_i = 1$ for a foreground region and $x_i = 0$ for a background region. They obtain the solution $X = x_i$ by minimizing Gibbs energy $E(X)$ [10]:

$$E(X) = \sum_{i \in v} E_1(x_i) + \lambda \sum_{(i,j) \in \varepsilon} E_2(x_i, x_j), \qquad (1)$$

where E_1 is likelihood energy, E_2 is prior energy, and λ is a parameter. Selecting the value of λ is not mentioned in [12], and we assume it is a constant with value of 1.

To set up the energy function, for E_1, the minimum color differences from every region in the unmarked area R_i to the marked foreground region d_i^F and background d_i^B need to be computed. For E_2, the color differences between each region R_i and its adjacent regions need to be computed. Then the max-flow algorithm is applied [6].

In our proposed algorithm, the pre-segmentation stage is similar to [12]. Then our first step (described in Section 2.2) requires less time than setting up E_1 in their method. The nearest neighbor search algorithm using a kd-tree further speeds up the algorithm as finding the nearest neighboring regions is an $O(logN)$ operation, where N is the total number of user-marked regions. Section 2.3 uses fast image processing operations as well.

3.2 Comparison to region merging by maximal similarity

Ning et al. [16] proposed a maximal similarity based region merging (MSRM) algorithm for interactive image segmentation, which is an iterative method utilizing the local information of the regions.

The algorithm consists of two stages. In the first stage, the marked background regions search for unmarked neighboring regions. If among the neighboring regions of an unmarked neighboring region, the marked background region is the most similar to the unmarked neighboring region, then the unmarked region is merged with the background. In the second stage, the unmarked regions are merged with neighboring unmarked regions using the same rule. Both stages need several iterations to stop. Moreover, after each merging operation, the feature descriptor of the region, a color histogram with 4096 bins, needs to be re-calculated. This makes the algorithm very slow, especially when the pre-segmentation gives many small regions.

In contrast, our algorithm does not involve iterations after pre-segmentation, and the mean color of each region is used as the feature descriptor.

4. EXPERIMENTS AND RESULT ANALYSIS

The proposed algorithm provides a faster way to segment the image based on pre-segmented regions. With the strokes and bounding box input by the user, it will label the unmarked regions by utilizing both global and local information of them.

In Section 4.1, we compare the speed and segmentation quality using our algorithm with graph-cuts and MSRM on regions. Most of the data is from the Berkeley segmentation database and PASCAL Visual Object Classes Challenge 2009 dataset. The experimental results clearly demonstrate the advantages of the proposed algorithm in terms of speed and quality. In Section 4.2, we show the time measurement of the algorithm on a mobile phone. In Section 4.3, we analyze the robustness and stability of our algorithm with respect to different strokes, different pre-segmentations, and difficult images. We also discuss the limitations of the proposed algorithm.

4.1 Comparison with other approaches

Since all three algorithms depend on the pre-segmentation result, we perform the same pre-segmentation for each algorithm and measure the time from when the pre-segmentation is done till the final result is output. The process is illustrated in Fig. 2 from (b) to (f). Table 1 shows that given the same user input and presegmentation result, our approach is an order of magnitude faster than either the graph-cuts over regions (GCR) or the MSRM. We measure the time on a desktop with Pentium 4 CPU 3.00GHz and 3.25GB of RAM.

Table 1: Comparison of region labelling time of the three algorithms

Time (s)	Number of pre-segmented regions	Our algorithm	MSRM	Graph cuts
Man	954	0.203	22.56	30.607
Cheetah	1021	0.313	48.342	72.137
Baby	379	0.078	5.391	10.47
Cow	344	0.062	6.482	7.79
Mushroom	496	0.125	15.039	15.808
Bird	637	0.079	9.143	12.923

These results make sense intuitively, based on our analysis in Section 3.1. For graph-cuts on regions, the more pre-segmented regions there are, the bigger the graph is. Both setting up and performing optimization are more expensive. The MSRM algorithm is iterative and there are several region updates in each iteration, while the proposed algorithm only involves simple clustering, local neighborhood analysis on much fewer ambiguous regions, and simple image processing.

The improvement of speed does not compromise the effectiveness of the proposed algorithm. On the contrary, it works as well or better than the other two methods that have higher complexities. Our purpose in each application is to cut out the object that the user chooses with red strokes.

Before we go into details to compare the quality of these three algorithms, we do want to mention that from our experiments, the GCR may perform better if a postprocessing step like in Section 2.3 is added, but no such methods were reported in the literature. Moreover, the point we want to emphasize is our algorithm's absolute advantage in terms of speed. If a postprocessing step is added to the GCR algorithm, it will only make it take even longer time than the proposed method.

The first row in Fig. 3 shows a comparison of these algorithms on a baby image. Given the same set of user input (a), the GCR algorithm mistakenly labeled the baby's eye, part of its lips and its forehead as background, and some black table leg as foreground (b). According to Eqn. (1), when some unmarked regions have bigger differences with their neighboring regions, the global minimum will let the cut go through those area. MSRM labeled the white part of the baby's cloth as background (c). Our proposed algorithm generated the best result.

The second row shows a comparison of the algorithms to an image with a man in front of a highly textured background. Given the same set of user input (a), the GCR algorithm again mislabeled part of the person's cloth as background and part of the leaves as foreground (b). The MSRM algorithm included some leaves as the foreground (c). In comparison, the proposed algorithm works best (d).

The mushroom image in the third row has a cluttered background. Given the same set of user input (a), the GCR algorithm again labels part of the background as foreground (b), and the MSRM algorithm labels even more background as foreground (c). Our algorithm yields the best result (d).

The bird image in the forth row has green color in the foreground as well as in the background. Given the same set of user input (a), the GCR algorithm labels some of the green leaves as foreground and some of the bird's feather as background (b). The MSRM algorithm mistakenly labels more background near the bird's head as foreground. Our proposed algorithm correctly cuts the bird out.

The cow image in the fifth row poses similar challenges. The color of gray grass is similar to the cow's color. As a result, the MSRM algorithm misses the left part of the cow's face (c). The result from the GCR algorithm (b) is not as good as ours (d). The sixth row's cheetah image is very challenging due to the fact that there is texture on both foreground and background. It takes $72.137s$ for the GCR algorithm to generate a result (b) that is not as good as our result (d) generated in $0.313s$. For the MSRM algorithm, after $48.342s$, the result image still mistakenly labels part of the left rear leg of the animal as background (c).

Table 2: Comparison of time with respect to robustness on cheetah image

Time (s)	User input set 1	User input set 2	User input set 3
Number of pre-segmented regions	1021	1021	3461
Our algorithm	0.25	0.281	0.782
MSRM	37.388	50.763	104.432
Graph cuts	64.34	66.777	444.818

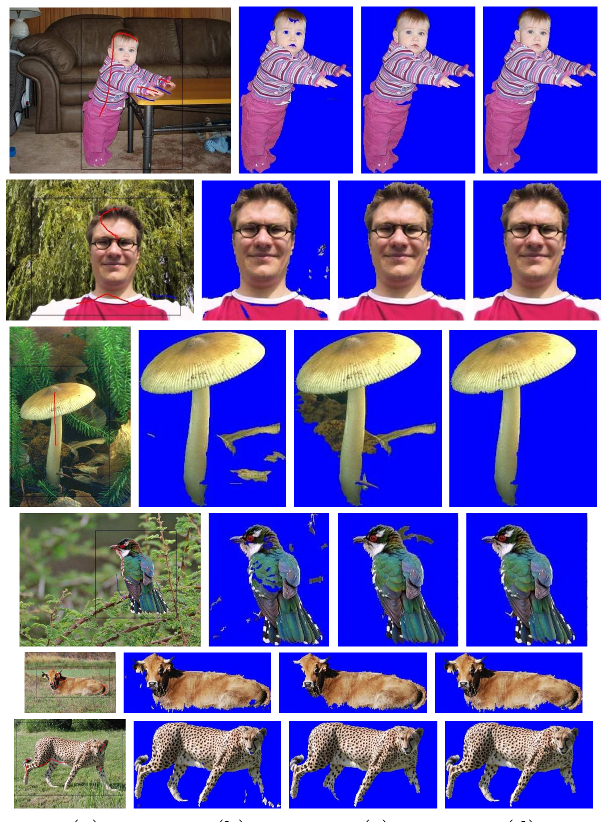

Figure 3: Comparisons between the graph-cuts over regions (GCR), the MSRM method [16] and the proposed algorithm. (a) The source images with user input stokes, where blue strokes are used to mark the background regions, and red strokes are used to mark the foreground regions. (b) Results obtained by the GCR method. (c) Results obtained by the MSRM method. (d) Results obtained by the proposed algorithm.

Table 3: Time measurement on Nokia N900

Time (s)	Man	Cheetah	Baby	Cow	Mushroom	Bird
	1.73	4.28	0.95	0.77	1.58	0.77

4.2 Result on a mobile phone

Our algorithm is implemented and tested on a Nokia N900 phone with an ARM Cortex-A8 600 MHz processor, 256 MB RAM, and a 3.5 inch touch-sensitive widescreen display. Table 3 shows the measurement of time for the same set of images. Since this is unoptimized research code, the time is satisfactory.

4.3 Robustness analysis

Similar to the other two methods, the number of pre-segmented regions and user input have an effect on the proposed algorithm's time, but our algorithm is more robust to these changes. Taking the cheetah image as an example, we use two different initial pre-segmentations and two more different sets of user inputs. The time comparisons are in Table 2, and the final segmentation results are in Fig. **??**. From these results, we can see that the changes affect the proposed algorithm's speed much less than the other algorithms, while the segmentation quality remains stable.

During our many experiments, we also found that low-contrast images with limited color information pose problems for all three algorithms. Such difficult images would first make the pre-segmentation potentially incorrect. This is the limitation to all segmentation methods that depend on pre-segmentation.

5. CONCLUSIONS AND FUTURE WORK

We have proposed a new region-based interactive image segmentation algorithm that can significantly increase the speed of segmentation without compromising its precision. It utilizes the global and local information, as well as the fact that regions are more robust to noise than single pixels, to avoid the expensive graph-cut minimization. We also extract the user input information by sampling the region's mean color instead of a single pixel's color. Our algorithm strikes a balance among precision, speed and feasibility on mobile phones.

In the future, we want to investigate the possibility of further decreasing the users' input without decreasing the speed or compromising the precision. We want to extend this method to interactive segmentation of video as well. It is also interesting to see whether incorporating user input information in various pre-segmentation algorithms would help. We also plan to find a way to combine the individual pixel information to further improve the algorithm. In summary, we aim to accomplish more elaborate yet easy and fast image editing tasks on mobile phones.

6. REFERENCES

[1] P. Arbeláez. Boundary extraction in natural images using ultrametric contour maps. *Proc. POCV*, 2006.

[2] P. Arbeláez and L. Cohen. Constrained image segmentation from hierarchical boundaries. In *Proc. CVPR*, pages 1–8, 2008.

[3] P. Arbeláez, M. Maire, C. Fowlkes, and J. Malik. From contours to regions: An empirical evaluation. In *Proc. CVPR*, pages 2294–2301, 2009.

[4] A. Blake, C. Rother, M. Brown, P. Perez, and P. Torr. Interactive image segmentation using an adaptive GMMRF model. *Lecture Notes in Computer Science*, pages 428–441, 2004.

[5] Y. Boykov and M. Jolly. Interactive graph cuts for optimal boundary and region segmentation of objects in ND images. In *Proc. ICCV*, pages 105–112, 2001.

[6] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE PAMI*, 26(9):1124–1137, 2004.

[7] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE PAMI*, 24(5):603–619, 2002.

[8] P. Felzenszwalb and D. Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59(2):167–181, 2004.

[9] D. Freedman, P. Kisilev, and I. Haifa. Fast Mean Shift by Compact Density Representation. In *Proc. CVPR*, pages 1818–1825, 2009.

[10] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE PAMI*, 6:721–741, 1984.

[11] V. Lempitsky, P. Kohli, C. Rother, and T. Sharp. Image segmentation with a bounding box prior. In *Proc. ICCV*, 2009.

[12] Y. Li, J. Sun, C.-K. Tang, and H.-Y. Shum. Lazy snapping. *ACM Trans. Graph.*, 23(3):303–308, 2004.

[13] H. Lombaert, Y. Sun, L. Grady, and C. Xu. A Multilevel Banded Graph Cuts Method for Fast Image Segmentation. In *Proc. ICCV*, pages 259–265, 2005.

[14] M. Maire, P. Arbelaez, C. Fowlkes, and J. Malik. Using contours to detect and localize junctions in natural images. In *Proc. CVPR*, pages 1–8, 2008.

[15] D. Martin, C. Fowlkes, and J. Malik. Learning to detect natural image boundaries using local brightness, color, and texture cues. *IEEE PAMI*, 26(5):530–549, 2004.

[16] J. Ning, L. Zhang, D. Zhang, and C. Wu. Interactive image segmentation by maximal similarity based region merging. *Pattern Recognition*, 43(2):445–456, 2010.

[17] S. Paris and F. Durand. A topological approach to hierarchical segmentation using mean shift. In *Proc. CVPR*, pages 1–8, 2007.

[18] X. Ren and J. Malik. Learning a classification model for segmentation. In *Proc. ICCV*, pages 10–17, 2003.

[19] C. Rother, V. Kolmogorov, and A. Blake. Grabcut: Interactive foreground extraction using iterated graph cuts. *ACM SIGGRAPH*, 23:309–314, 2004.

[20] L. Shapiro and G. Stockman. *Computer Vision*. Prentice Hall, 2002.

[21] R. Szeliski, R. Zabih, D. Scharstein, O. Veksler, V. Kolmogorov, A. Agarwala, M. Tappen, and C. Rother. A comparative study of energy minimization methods for markov random fields. *Lecture Notes in Computer Science*, 3952:16–29, 2006.

[22] S. Vicente, V. Kolmogorov, and C. Rother. Joint optimization of segmentation and appearance models. In *Proc. ICCV*, 2009.

[23] L. Vincent and P. Soille. Watersheds in digital spaces: an efficient algorithm based on immersion simulations. *IEEE PAMI*, 13(6):583–598, 1991.

[24] J. Wang. Discriminative Gaussian mixtures for interactive image segmentation. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 386–396, 2007.