Interactive Image Segmentation using Optimization with Statistical Priors

Jian Guan and Guoping Qiu

School of Computer Science, The University of Nottingham
Jubilee Campus, Nottingham, NG8 1BB, UK
{jwg, qiu}@cs.nott.ac.uk

Abstract. Interactive image segmentation is important and has widespread applications in computer vision, computer graphics and medical imaging. A recent work has shown that interactive figure ground segmentation can be achieved by computing a transparency image using an optimization framework, where user interactions are used to supply constraints for solving a quadratic cost function with a unique global minimum, which can be efficiently obtained using standard methods. In this paper, we introduce statistical priors as constraints to solve the optimization problem. We show that for some images, the statistical priors can provide good enough constraints to automatically obtain satisfactory figure ground segmentation results. For more difficult cases, we use the segmentation result of the statistical priors as a starting point for interactive figure ground segmentation. We show that segmentation results obtained based on statistical priors can be effectively employed to guide user interaction thus helping to reduce users labor in the interaction process. We also present a new effective adaptive thresholding method for making binary (hard) segmentation based on the computed continuous transparency image. Another contribution of this paper is the extension of the optimization based interactive figure ground segmentation framework to interactive multi-class segmentation, where user can provide multi-class seed pixels instead of just foreground background 2-class seeds, for segmenting the given image into the desired number of regions by performing a one-shot optimization operation, which again has a unique global minimum and can be obtained by solving a large system of linear equations. We present various experimental results, including segmentation error rates on an online image database with human labeled ground truth, to show that our method works well and has direct applications in areas such as interactive image editing.

1 Introduction

Foreground background segmentation has wide applications in computer vision (e.g. scene analysis), computer graphics (e.g. image editing) and medical imaging (e.g. organ segmentation). Fully automatic image segmentation has many intrinsic difficulties and is still a very hard problem. In many applications, such as image editing in computer graphics and organ segmentation in medical imaging, semi-automatic and interactive approaches, where human operators provide strong priors for the computational algorithms to perform segmentation, can not only overcome the inherent technical difficulties of fully automatic image segmentation, but may also be desirable because the operators in many of these applications may want to be able to control the segmentation process and results.

There have been increasing activities in the research community to develop interactive semi-automatic image segmentation techniques [2, 3, 9, 11, 12]. In [2], the authors presented an interactive image segmentation technique based on graph cut. Users labeled seed pixels which indicating definite background and foreground were used as strong priors for segmenting images into figure and ground. In [3], the authors showed that graph cut based segmentation algorithms could be implemented very fast. In [12], the authors presented a segmentation given partial grouping constraints method. User inputs were used as *bias* to a natural grouping process, and the authors formulated such biased grouping problem as a constrained optimization problem that propagates sparse partial grouping information to the unlabelled data by enforcing grouping smoothness and fairness on the labeled data points. They used the normalized cut criterion and solved the optimization problem by eigen-decomposition. In [9], the authors presented an interactive image foreground extraction method that was computationally based on graph cut of [1] but the authors introduced a simpler user interaction technique to reduce user efforts in the interaction process and an iterative model updating procedure to improve accuracy. In [11], an interactive foreground background segmentation method was introduced in the context of image matting. The authors used Belief Propagation to iteratively propagate user labeled pixels to the unlabeled pixels.

A recent work [13] has developed an optimization based figure ground segmentation technique, where a transparency image was computed by optimizing a quadratic cost function with user supplied linear constraints. The optimization problem has a unique global minimum and can be solved efficiently by standard numerical methods. In this paper, we introduce statistical priors as constraints to solve the optimization problem. For some images, the statistical priors can provide good enough constraints to automatically obtain satisfactory figure ground segmentation results. For more difficult cases, user interaction is necessary. In such cases, we use the segmentation result based on the statistical priors as a starting point for interactive figure ground segmentation, and as a guide to help users to place the constraints in the correct locations to generate the desired results. In this way, the statistical priors not only guide the user but also help reducing users' labors in the interaction process. We have also developed a new method to make binary (hard) segmentation based on the computed continuous transparency image. Another contribution of this paper is the extension of the optimization based interactive figure ground segmentation framework to interactive multi-class segmentation, where user can provide multi-class seed pixels instead of just foreground background 2-class seeds, for segmenting the given image into the desired number of regions by performing a one shot optimization operation, which again has a unique global minimum and can be obtained by solving a large system of linear equations.

The organization of the paper is as follows. In Section 2, we present the framework of optimization based figure ground segmentation and describe solutions based on statistical priors and user interaction. In section 3, we present experimental results and demonstrate the possible applications of our method. In section 4, we extend the optimization based figure ground segmentation framework to interactive multi-class image segmentation and present preliminary results.

2 Interactive Figure Ground Segmentation using Optimization

For a given color image $I(z)$, where $z\in(x, y)$ is the co-ordinate vector, there is a corresponding (hidden) transparency image $\alpha(z)$, where $0\leq\alpha(z)\leq1$ is called the alpha matte in computer graphics [4]. We consider each pixel as being generated as an additive combination of a proportion $\alpha(z)$ of foreground color with a proportion $1-\alpha(z)$ of background color. For a definite background pixel, we have $\alpha(z) = 0$ and for a definite foreground pixel, we have $\alpha(z) = 1$. For pixels that are between foreground and background, we have $0<\alpha(z)<1$. In figure ground segmentation, we have to decide whether a pixel with $0<\alpha(z)<1$ belongs to foreground or background. It is in these areas, the task of segmentation becomes hard. In many situations, even human observers will have difficulty in deciding whether some particular pixels should belong to background or foreground. There exist a large amount of uncertainty and, if a crisp decision has to be made, it is unlikely that a unique answer is possible. Therefore, instead of making a binary decision in the modeling stage, we wish to compute a continuous transparency image $\alpha(z)$ and then making hard segmentation based on $\alpha(z)$. This approach may offer more flexibility in making the binary decision (Section 3.1 presents a possible approach) and may also be conveniently used for applications such as image composition (Section 3.5).

If two pixels both belong to the foreground (or background), it is reasonable to assume that they will have similar colors and other photometric properties. In other words, pixels with similar photometric properties should have similar α values. Within the background or foreground, it is also reasonable to assume that the image is smooth, i.e., neighboring pixels will have similar photometric properties hence α values. If two pixels one belongs to the background, the other belongs to the foreground, it is also reasonable to assume that they will have very different photometric properties, that is, two pixels with different photometric properties should have different α values.

Based on such reasoning, we can formulate a cost function for the transparency image $\alpha(z)$ such that an $\alpha(z)$ that satisfies the above assumptions will correspond to a minimum of the cost function. We will follow a similar approach developed in a recent work [13] to formulate the cost function but will introduce statistical priors as a new means to set the constraints for initially solving the optimization problem. For cases where statistical priors are inadequate, user interactions can build on the result of the statistical priors to produce desired results quickly.

2.1 The Cost Function

This sub section essentially follows that of [13]. Let $G(z) = \Phi(I(z))$ be image features (such as color, texture, etc), computed around the pixel at location z, where Φ is the feature extraction operator. We call $G(z)$ the photometric features. Let $N_g(z)$ be the set that contains the *geometric* neighbors of z, $t \in N_g(z)$ if $|t - z| < R_g$, let $N_p(z)$ be the set that contains the *photometric* neighbors of z, $t \in N_p(z)$ if $|G(t) - G(z)| < R_p$, where R_g and R_p are some preset constants determining the size of the neighborhoods. Let $N_m(z) = N_g(z) \cup N_p(z)$ which includes all pixels that are either the geometric neighbors or the photometric neighbors of z.

Based on some reasonable assumptions about the transparency image and assuming that two spatially adjacent pixels should have similar α values, two photometrically similar pixels should also have similar α values, and the difference in α values between two pixels should be proportional to the pixels' photometric distance, the α image may be obtained by minimizing the cost function defined as

$$E(\alpha(z)) = \sum_z \left(\alpha(z) - \sum_{t \in N_m(z)} \left(w_g(z,t) + \lambda w_p(z,t) \right) \alpha(t) \right)^2 \quad \textbf{(1)}$$

where $w_g(z, t)$ is the geometric neighbor similarity weighting function between two pixels z and t, $w_p(z, t)$ is the photometric neighbor similarity weighting function between two pixels z and t, and λ is a constant that measures the relative importance of the geometric neighbor similarity and the photometric neighbor similarity, all similarity weightings inside the neighborhood sum to one.

$$\sum_{t \in N_m(z)} \left(w_g(z,t) + \lambda w_p(z,t) \right) = 1 \quad \textbf{(2)}$$

To reflect the assumptions that geometrically close pixels are likely to have similar α values and geometrically far apart pixels are likely to have different α values, and photometrically similar pixels are likely to have similar α values and photometrically different pixels are likely to have different α values, the neighbor similarity functions $w_p(z, t)$ and $w_g(z, t)$ should be small if $|z - t|$ is large and small if $|z - t|$ is large, and they should be small if $G(z)$ and $G(t)$ are different and large if $G(z)$ and $G(t)$ are similar. We use following functions

$$\begin{aligned} &if \;\; I(t) \in N_g(z) \quad w_g(z,t) \propto e^{-\|t-z\|^2 / \sigma_{gg}^2} e^{-\|G(t)-G(z)\|^2 / \sigma_{gp}^2} \\ &if \;\; I(t) \in N_p(z) \quad w_p(z,t) \propto e^{-\|t-z\|^2 / \sigma_{pg}^2} e^{-\|G(t)-G(z)\|^2 / \sigma_{pp}^2} \end{aligned} \quad \textbf{(3)}$$

where σ_{gg} and σ_{gp}, σ_{pg} and σ_{pp} are the variances of the geometric co-ordinates and photometric features inside $N_p(z)$ and $N_g(z)$ respectively.

A similar cost function of the form, $\alpha^T(I-W)^T(I-W)\alpha$, also appeared in the literature in the context of image segmentation using normalized cut [10], nonlinear dimensionality reduction [8], and coloring black and white images [6].

Note that one of the purposes of using photometric neighbors is to bring spatially far away but photometrically very similar pixels to have influence on each other's α values. In theory can be achieved by making the geometric neighborhood larger (at the extreme to include the whole image). However, this will have two problems. Firstly, a larger neighborhood will make the problem less sparse thus making the problem computationally much more demanding. Secondly, making a pixel influenced by too many pixels (with unknown status), it may also risk introducing inaccuracy and uncertainty into the model.

2.2 Optimize the Cost Function with Statistical Priors

To solve (1), we need to constrain the problem. One possible solution is to linearly constrain the problem by finding statistically almost definite background pixel locations z_i and set $\alpha(z_i) = 0$; and statistically almost definite foreground locations z_j and set $\alpha(z_j) = 1$. Using these linear constraints, the quadratic function in (1) has a unique global minimum. This optimization problem yields a large, sparse system of linear equations, which may be solved efficiently using a number of standard tools [1, 7].

There are many possible ways for finding the statistical priors. Here we present a possible (not necessarily the optimal) approach. We first classify the pixels into two classes, and then within each class, we find the pixels that are most likely to belong to the foreground or the background. There are numeral methods in the literature that can be used to achieve this purpose. Note that accurate foreground background separation is not the goal here but instead the purpose is to be able to identify some foreground and some background pixels with reasonably high confidence and use these pixels as constraints for solving the optimization problem of (1). Although any previous automatic image segmentation and data clustering algorithms can be used, for simplicity, we have developed our implementation based on *k*-means clustering algorithm. The procedure is described as follows:

Step 1: Using the pixels photometric features $G(z)$ and the k-means algorithm to classify the pixels into 2 classes, C_1 and C_2.

Step 2: Again use k-means to classify $\forall\, z \in C_1$ into k sub classes, $c_{11}, c_{12}, c_{13}, \ldots, c_{1k}$ and classify $\forall\, z \in C_2$ into k sub classes, $c_{21}, c_{22}, c_{23}, \ldots, c_{2k}$

Step 3: Compute the probabilities $p_{11} = p[z \in c_{11}]$, $p_{12} = p[z \in c_{12}]$, …, $p_{1k} = p[z \in c_{1k}]$; $p_{21} = p[z \in c_{21}]$, $p_{22} = p[z \in c_{22}]$, …, $p_{2k} = p[z \in c_{2k}]$

Step 4: Set $c_{1\max} = c_{1i}$, if $p_{1i} \geq p_{1j}\ \forall$ j; set $c_{2\max} = c_{2i}$, if $p_{2i} \geq p_{2j}\ \forall$ j

Step 5: Find the set $\Omega_1 = \{z \mid z \in c_{1\max}\}$; and the set $\Omega_2 = \{z \mid z \in c_{2\max}\}$

Step 6: For the set Ω_1, find the connected pixel regions $s_{11}, s_{12}, \ldots, s_{1m}$; and for the set Ω_2, find the connected pixel regions $s_{21}, s_{22}, \ldots, s_{2n}$

Step 7: If $z \in s_{1i}$ and $s_{1i} > S$; set $\alpha(z) = 0$; if $z \in s_{2j}$ and $s_{2j} > S$ set $\alpha(z) = 1$; $\forall\, i, j$; where S is the preset connected region size (area).

In words, we first classify the pixels into two classes. Within each class, we find the pixels that occur with the highest probability. We then perform connected region labeling on these most frequently occurred pixels. Only those pixels that form a connected region that is larger than a preset size will be used as priors for solving the optimization problem of (1). Once these constraints are identified, then solving the optimization problem become straightforward, i.e., solving a large, sparse linear system of equations.

2.3 Human Interaction for Result Refinement

As we shall demonstrate in section 3 that using the statistical priors can achieve satisfactory results for some images, see Fig. 1. However, as any automatic image segmentation algorithms, the results are image dependent. Fortunately, in our current framework, we can build on the result of the statistical priors and manually add extra constraints or remove inappropriate constraints to solve the optimization problem again to achieve the desired result.

Often in interactive image segmentation, the users do not know exactly where to start unless they are very experienced. The initial results obtained using the statistical priors can serve as a guide to help users to place the constraints in the strategically correct locations, which can not only increase the chance for success but also can help reducing the user effort. Please see Fig. 2 in Section 3 for examples of such interaction.

3 Experimental Results

The implementation of the algorithm consists of several straightforward steps. First we need to decide the types of photometric features to use and find the geometric and photometric neighbors for each pixel, and then we compute $\alpha(z)$ by optimizing $E(\alpha(z))$ using the statistical priors. If the results are not satisfactory, then extra constraints are added on by the user until satisfactory results are obtained.

The selection of geometric neighbors is straightforward and in all our experiments we use the 8 spatially connected pixels as geometric neighbors. We then need to select the photometric features for finding the photometric neighbors and for computing the similarity weighting measures (3) in the feature space. In all our experiments, we convert the color image into La^*b^* color space and use all pixels from the 3x3 window that center on a pixel to form the photometric feature for that pixel. We use a very simple method to efficiently search the photometric neighbors. For each pixel, we randomly sample K ($K \approx 150$ in all our experiments) pixels from an M x N window (17 x 17 in all our results) center on the pixel. From these K pixels, we find 4 that are the most similar to the pixel as its photometric neighbors (It is neither desirable nor it turns out necessary to have large numbers of photometric neighbors). Each pixel will have 8 geometric neighbors and 4 photometric neighbors. Therefore each α pixel is linked to 12 (or fewer) neighbors via their similarity weightings. The relative geometric and photometric weighting constant λ is determined by experiment, which may vary from image to image. We found that results are not sensitive to λ and setting $\lambda = 1$ works well for most images.

Solving the minimization problem of (1) using the linear constraints yields a large, sparse system of linear equations. In our current implementation, we use Matlab's build-in least squares solver for sparse linear system to directly solve for $\alpha(z)$. On a Pentium 4 PC with 1.8GHz CPU, our current implementation takes about 6 minutes to compute an α image for a 500x300 image. However, fast methods and even dedicated hardware are available to solve this problem much faster [1, 5, 7].

3.1 Adaptive Thresholding α

The optimization process of (1) produces a continuous image $\alpha(z) \in [0, 1]$. If $\alpha(z) = 0$, it is a background and if $\alpha(z) = 1$, it is a foreground pixel. In order to make a binary decision, i.e., segmenting figure and ground, we need to make pixels where $0 < \alpha(z) < 1$

either belong to the foreground or the background. Although there are many possible ways to make such binary decision, we have developed a simple yet extremely effective method. For $0<\alpha(z)<1$, we place a rectangular window $W(z)$ around the pixel z and choose the smallest possible window size which will also include at least one $\alpha=0$ pixel and one $\alpha=1$ pixel. We then calculate the average of all α's$\neq 0$ and α's$\neq 1$ inside the window as the threshold $T(z)$. That is, we select the window size adaptively by specifying that a window have to include at least one definite background and one definite foreground pixels. In computing the threshold value, we exclude the definite foreground and definite background pixels in the window. Hard (binary) segmentation is achieved as: If $\alpha(z) \geq T(z)$ then $\alpha(z)=1$ (foreground); and if $\alpha(z) < T(z)$ then $\alpha(z)=0$ (background).

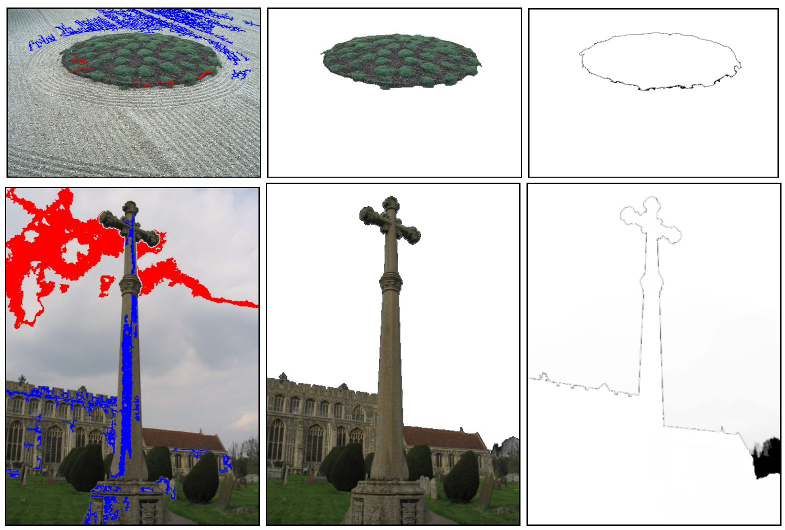

Fig. 1. Results of foreground background segmentation using statistical priors only. Left column shows the original image with statistical priors (red and blue pixels). Middle column shows the segmented foreground images. Right column shows the segmentation errors as compared with the ground truth. Original and ground truth images used in [14] and are available online [15]. Note that most of the error pixels are labeled as mixed pixels in the ground truth.

3.2 Segmentation using Statistical Priors

In this experiment, we use the statistical priors only as the linear constraints to solve the optimization problem in (1). The statistical priors are obtained as described in Section 2.2 and the computed continuous α is binarized (hard segmented) using the procedure of Section 3.1. Fig.1 (previous page) shows examples of segmentation results. It is seen that statistical priors alone can produce very good results in these cases. Note that everything is automatic in this mode.

3.3 Human Interaction for Result Refinement

Fig. 2. Human interaction build on statistical priors result to produce good segmentation result. 1st column: original image with statistical priors (yellow and light blue pixels). 2nd column: α images based on statistical priors only. 3rd column: User interaction based on statistical priors results, red and blue scribbles are put on by the user to be used as extra constraints. Note that in the Mushroom image, the white squares indicate the statistical priors should be removed from these regions. 4th column: segmented images after human interaction. 5th column: the errors of the segmentation as compared with the ground truth. Original and ground truth images used in [14] and are available online [15].

Even though our method can do a reasonable job in segmenting non-complicated scenes by using statistical priors alone, for more challenging scenes, as any automatic segmentation scheme, our method can also fail. The advantage of our method is that it allows user interaction. Users can build on the results of the statistical priors and manually place extra constraints in the correct locations to guide the optimization process to achieve good segmentation results. Fig. 2 shows examples of such interaction process. It is seen that statistical priors alone did not work very well. However, the results of the statistical priors indicated to the user where he/she should place the background and foreground constraints in order to achieve a good segmentation. It also allows the user to remove inappropriate priors which have caused the optimization process to fail to produce a good result. Therefore, the statistical priors can be used as a guide for user interaction which is helpful in reducing user labor.

3.4 Segmentation Results for a Dataset with Ground Truth

In this section, we present results performed on a database used in [9, 14] and which is available online [15]. The data set consists of 50 color images, each has a “trimap” and a ground truth image. The trimap specifies definite background and definite foreground pixels and the pixels between the foreground and background are unknown, see Fig. 3.

We use the trimap to set the constraints for our optimization based image segmentation to compute the α in those unknown areas. We then use the adaptive thresholding procedure of section 3.1 to make hard (binary) segmentation. We also computed the segmentation error rate as defined in [14] which measures the ratio of misclassified pixels over unknown pixels. We also excluded the possible mixed pixels on object boundaries for error rate calculation. The average error rate on all 50 images was **4.65%** with a standard deviation of **3.26%**.

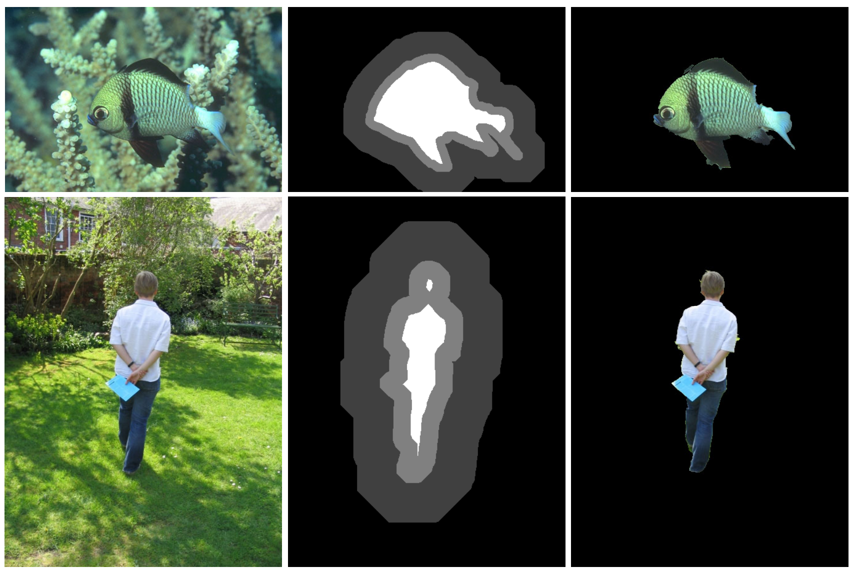

Fig. 3. Example of segmentation results using trimap. Left column: original images. Middle column: the trimap images with user defined foreground (white) background (dark grey) and unknown (light grey). Right column: foreground results. The error rate for the top image is 4.43% and the error rate for the bottom image is 0.89%.

These are excellent results that are comparable to those presented in [14]. Note that in our technique, we do not have to train the segmentation model whilst the method of [14] used 15 images to train the segmentation model and tested the model on 35 images. Our result is obtained using default parameter settings and everything is done automatically. Also our results are for all 50 images rather than a subset. Fig. 3 shows examples of the segmentation results performed on this data set.

3.5 Image Composition/Matting

Lets consider a situation where we have pixels that are near the boundary u between the foreground and background objects as shown in Fig. 4. Assuming that the background signal $B(t)$ and the foreground signal $F(t)$are smooth surfaces, these pixels can be approximately modeled as

$$I(z_f) = \int_{-\infty}^{u} F(t)h(t-z)dt + \int_{u}^{+\infty} B(t)h(t-z)dt \approx \alpha(z_f)F + (1-\alpha(z_f))B \tag{4}$$

$$I(z_b) = \int_{-\infty}^{u} F(t)h(t-z)dt + \int_{u}^{+\infty} B(t)h(t-z)dt \approx \alpha(z_b)F + (1-\alpha(z_b))B$$

where $\alpha(z_f) = \int_{-\infty}^{u} h(t-z_f)dt$ and $\alpha(z_b) = \int_{-\infty}^{u} h(t-z_b)dt$

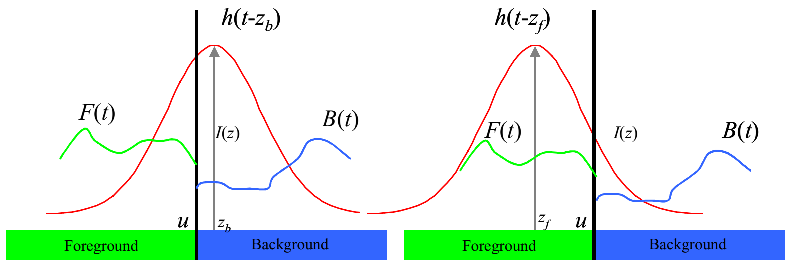

Fig. 4. Background and foreground contribute to the pixels near the boundary between foreground and background objects, $h(t)$ is the point spread function of the camera system.

If the pixel is near the boundary but is on the foreground side at z_f, then $\alpha(z_f) > (1-\alpha(z_f))$, i.e., $\alpha(z_f) > 0.5$. The farther way it is from the boundary, the value of $\alpha(z_f)$ will be closer to 1. If the pixel is near the boundary but on the background side at z_b, then $\alpha(z_b) < (1- \alpha(z_b))$, i.e., $\alpha(z_b) < 0.5$. The farther way it is from the boundary, the value of $\alpha(z_b)$ will be closer to 0. Note that for definite background pixels we have $a(z) = 0$ and for definite foreground pixels we have $a(z) = 1$.

This shows that in image formation, in the areas that are near the boundary between the background and the foreground, if a pixel is closer to the foreground, then its α value should be more similar to that of the foreground; and if it is closer to the background, its α should be more similar to that of the background. The α image computed based on (1) has exactly this property because one of the assumptions of (1) is that two spatially close pixels should have similar α's. Therefore the computed α image using (1) can be seen as the alpha matte [4] and may be directly used for image composition using following image composition equation:

$$I(z) = \alpha(z)F(z) + (1-\alpha(z))B(z) \tag{5}$$

Fig. 5 shows examples of using the computed α as the transparency layer for extracting the alpha matte and for image composition.

Fig. 5. Image composition using the computed α image as the transparency layer. Left column: images containing the foreground objects. Middle column: the α images computed by our optimization framework. Right column: composition image based on equation (5).

4 Interactive Multiple Class Segmentation using Optimization

Interactive image segmentation methods such as those in [2, 9, 11] can only segment images into 2 classes in their energy formulation. If multiple regions segmentation is desired, then these methods have to be applied recursively. We can extend the interactive optimization foreground background segmentation framework to interactive multiple class segmentation. Assuming we would like to segment the image into n classes, c_1, c_2, …, c_n, then we construct n α images, $\alpha_i(z)$, i =1, 2, …, n, each correspond to a class. We have, if $z \in c_i$ then $\alpha_i(z) = 1$, and $\alpha_j(z) = 0$ for $\forall j \neq i$. To construct the n α images, we can formulate the problem as a constrained optimization the n α images, we can formulate the problem as a constrained optimization problem as:

$$\text{Minimizing: } \sum_i \sum_z \left(\alpha_i(z) - \sum_{t \in N_m(z)} \left(w_g(z,t) + \lambda w_p(z,t) \right) \alpha_i(t) \right)^2 \quad \textbf{(6)}$$

$$\text{Subject to: } \sum_{t \in N_m(z)} \left(w_g(z,t) + \lambda w_p(z,t) \right) = 1 \text{ and } \sum_i a_i(z) = 1$$

This optimization problem is a straightforward extension from (1). Within each class, we assume that if two pixels have similar photometric properties or are close spatially, then they should have similar α values. Conversely, they should have different α. The weighting functions between the same geometric neighbors and the same photometric neighbors are the same for different α images. However, the second constraint in (6) ensures that a single pixel will not be classified into 2 classes simultaneously (all α values at a given co-ordinate sum to 1). Another interpretation of this second condition is that $\alpha_i(z) = p(z \in c_i)$, the probability of the pixel at z belongs to class c_i.

To solve the optimization problem in (6) the conditions are insufficient, we can also employ user input to provide more constraints to make the problem well conditioned. Assuming at locations $z_1, z_2, \ldots z_n$, user have specified pixels as belonging to $c_1, c_2, \ldots c_n$ respectively, then the constraints are: $\alpha_i(z_j) = \delta(i, j)$, for i =1, 2, …, n and j =1, 2, …, n, where δ is the Dirac delta function. With these constraints, then the optimization problem again yields a large, sparse system of linear equations. Now the system is n times as large as the foreground background segmentation problem. A straightforward solution using standard methods will require large amount of memory. Although the system is still sparse, fast solution may require special programming considerations. Once the α images are computed, then segmentation can be achieved as $z \in c_i$ if $\alpha_i(z) = \max\{\alpha_1(z), \alpha_2(z), \ldots, \alpha_n(z)\}$.

We have implemented the interactive multiple class image segmentation algorithm in Matlab using its build in linear system solver. Fig. 6 shows examples of interactively segmenting the images into 3 classes simultaneously. It is seen that the method clearly works which demonstrates the potential of the new extension. However, computationally, we found that straightforward solution requires a huge amount of memory space and is computationally expensive. We are currently working on more efficient implementations.

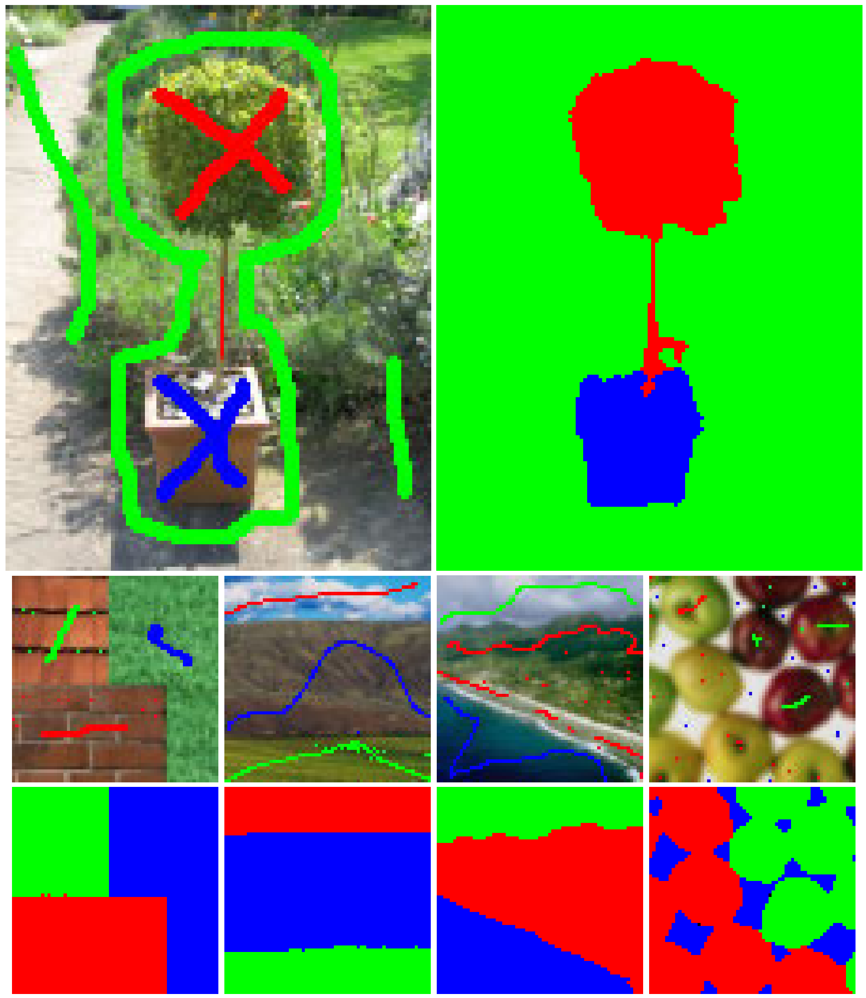

Fig. 6. Examples of interactive multiple class image segmentation using optimization. The figures shows the original images with user indicated 3 classes of seed pixels (red, green and blue color scribbles), and the segmentation results in red, green and blue regions.

5 Concluding Remarks and Future Work

In this paper, we have presented a method that introduces statistical priors to an optimization based interactive image foreground background segmentation framework. We have shown that for simple cases, the statistical priors will be sufficient to produce satisfactory results automatically. We have shown segmentation results obtained based on statistical priors can be effectively employed to guide user interaction thus helping to reduce users labor in the interaction process. We have also successfully extended the foreground background two-class segmentation framework to interactive multiple class segmentation.

Preliminary results have demonstrated that the interactive multiple class segmentation works well. But we also encountered problems in solving the multiple class segmentation problems for large images and for large number of classes, chiefly caused by the memory problems in the process of solving the large system of linear equations. In our future work, we will investigate better and more efficiency ways to solving the computational problems. Our goal is to achieve real time interactive image segmentation of arbitrary number of classes using the optimization framework.

References

1. Hackbusch, W. 1985. Multi-grid Methods and Applications, Springer, Berlin, 1985
2. Boykov, Y.,and Jolly, M. P. 2001. Interactive graph cuts for optimal boundary and region segmentation of objects in N-D images. In Proc. IEEE Int. Conf. on Computer Vision
3. Boukov, Y and Kolmogorov, V. 2004. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. IEEE Trans on PAMI, Vol. 26, 1124-1137
4. Chuang, Y.-Y., Curless, B., Salesin, D., and Szeliski, R. 2001. A Bayesian approach to digital matting. In Proc. IEEE CVPR 2001, CD-ROM.
5. Geiselmann, W., Shamir, A., Steinwandt R., and Tromer, E. 2005. Scalable hardware for sparse systems of linear equations, with applications to integer factorization. Proc. CHES 2005, LNCS 3659, 131--146, Springer
6. Levin, A., Lischinski, D. and Weiss, Y. 2004. Colorization using Optimization. SIGGRAPH, ACM Transactions on Graphics, 689-694
7. Press, W. H. and Teukolsky, S. A. et al. 2002. Numerical Recipes in C++ - The Art of Scientific Computing, Cambridge, MA, Cambridge University Press
8. Roweis, S and Saul, L. 2000. Nonlinear dimensionality reduction by locally linear embedding. Science, v.290, no.5500 , 2323-2326
9. Rother, C., Kolmogorov, V., and Blake, A. 2004. "GrabCut": interactive foreground extraction using iterated graph cuts. ACM Trans. Graph. 23(3): 309-314
10. Shi, J.and Malik, J. 2000. Normalized Cuts and Image Segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence, 22(8), 888-905
11. Wang, J. and Cohen, M., 2005. An iterative optimization approach for unified image segmentation and matting. In Proc. IEEE Int. Conf. On Computer Vision
12. Yu, S. X. and Shi, J. 2004. Segmentation given partial grouping Constraints. IEEE TPAMI, 173–183
13. Qiu, G. and Guan, J., 2006. Interactive Image Matting using Optimization, Report-VIPLAB-01-2006 , Visual Information Processing Lab, School of Computer Science and Information Technology, University of Nottingham
14. A. Blake, C. Rother, M. Brown, P. Perez, and P. Torr. Interactive image segmentation using an adaptive GMMRF model. Proc. Eur. Conf. on Computer Vision, ECCV (2004).
15. http://www.research.microsoft.com/vision/cambridge/segmentation