Image Segmentation by Iterated Region Merging with Localized Graph Cuts

Bo Peng[a], Lei Zhang[a,1], David Zhang[a], Jian Yang[b]

[a]*Department of Computing, The Hong Kong Polytechnic University, Kowloon, Hong Kong, China*

[b]*School of Computer Science and Technology, Nanjing University of Science and Technology, Nanjing, 210094, China*

Abstract

This paper presents an iterated region merging-based graph cuts algorithm which is a novel extension of the standard graph cuts algorithm. Graph cuts addresses segmentation in an optimization framework and finds a globally optimal solution to a wide class of energy functions. However, the extraction of objects in a complex background often requires a lot of user interaction. The proposed algorithm starts from the user labeled sub-graph and works iteratively to label the surrounding un-segmented regions. In each iteration, only the local neighboring regions to the labeled regions are involved in the optimization so that much interference from the far unknown regions can be significantly reduced. Meanwhile, the data models of the object and background are updated iteratively based on high confident labeled regions. The sub-graph requires less user guidance for segmentation and thus better results can be obtained under the same amount of user interaction. Experiments on benchmark datasets validated that our method yields much better segmentation results than the standard graph cuts and the Grabcut methods in either qualitative or quantitative evaluation.

Keywords: Image segmentation, graph cuts, region merging

[1]Corresponding author. Email:cslzhang@comp.polyu.edu.hk. This research is supported by the Hong Kong SAR General Research Fund (GRF) under grant No. PolyU5351/08E and the National Science Foundation Council (NSFC) of China Key Grant under No. 60973098.

1. Introduction

While it has been widely studied for many decades, automatic image segmentation is still a big challenge due to the complexity of image content. A lot of work shows that the user guidance can help to define the desired content to be extracted and thus reduce the ambiguities produced by the automatic methods. In this paper we consider the most common type of interactive segmentation: segmenting the object of interest from its background.

In the past few years, various approaches to interactive segmentation have been proposed. For example, livewire [1] allows the user to interactively select certain pixels where the segmentation boundary should pass. However, texture or noise in the image might require a lot of interaction in order for an acceptable segmentation. To obtain real time response to the user's actions, independent of the image size, Falcão [2] proposed a modified livewire method, which exploits three properties of Dijkstra's algorithm to compute minimum-cost paths in sub-linear time. Active contour, or snake [3, 4, 5], is defined as an energy-minimizing spline. After initializing the contour close to the original object boundary, the contour will fit the actual object boundary iteratively. Level sets-based segmentation method [6] uses implicit active contour models, in which the numerical computation involving curves and surface is performed without having to parameterize the objects.

Another preferable interactive segmentation method based on combinatorial optimization is graph cuts [7, 8]. It addresses segmentation in a global optimization framework and guarantees a globally optimal solution to a wide class of energy functions. In addition, the user interface of graph cuts is convenient-seeds can be loosely positioned inside the object and background regions, which is easier compared to placing seeds exactly on the boundary, like in livewire [1]. Because graph cuts can involve a wide range of visual cues, a number of recent literature further extended the original work of Boykov and Jolly [7] and developed the use of regional cues [9, 12], geometric cues [13, 14], shape cues [15, 16, 17], stereo cues [12], or even topology priors [18] as global constraints in the graph cuts framework. When foreground and background color distributions are not well separated, the traditional graph cuts [7] cannot achieve satisfying segmentation. Some advanced versions of graph cuts are developed [9, 10, 11, 19], which are more robust and substantially simplify the user interaction. In [10], the user interaction can be applied on both coarse and fine scales, which inherit the advantages in region and boundary-based methods for image segmentation. The work proposed in [11] makes a progressive local selection on the object of interest. Instant visual feedback is provided to the user for a quick and effective image editing.

In the classical graph-based framework, most of segmentation methods consider pixels or groups of pixels as the nodes in a graph. The edge weight estimation usually takes into account image attributes, for example color, gradient and texture. An efficient edge weight assignment method was proposed by Miranda et al. [20], where the object information obtained from user interaction as well as the image attributes are both used for estimating edge weights. Separating from the image segmentation process, it can act as a basic step for high accuracy image segmentation. Some other works studied graph structures for designing image processing operators. Image foresting transform (IFT) [21, 22], for example, defines a minimum-cost path forest in a graph, and provides a mathematically sound framework for many image processing operations. Based on similar graphs, a theoretical analysis between optimum-path forests and minimum cut was given in [23]. Under some conditions, the two algorithms were proven to produce the same result.

In our preliminary work [19], we explore the graph cuts algorithm by extending it to a region merging scheme. Starting from seed regions given by the user, graph cuts is conducted on a propagated sub-graph where the regions are regarded as the nodes of the graph. An iterated conditional mode (ICM) is studied and the maximum a posterior (MAP) estimation is obtained by virtual of graph cuts on each growing sub-graph. The segmentation process is stopped when all the regions are labeled. In [19], the initial segmentation is obtained by meanshift algorithm, which is a sophisticated segmentation technique. While in this paper, the initial segmentation is obtained by the simple watershed algorithm [24]. In each iteration, a semi-supervised algorithm is applied to learn a classifier. Consequently, the most confident labels will contribute for new seed regions in the next iteration.

The proposed method is a novel extension of the standard graph cuts algorithm. Rather than segmenting the entire image all at once, the segmentation is performed incrementally. It has many advantages to do this. First of all, using sub-graph significantly reduce the complexity of background content in the image. The many unlabeled background regions in the image may have unpredictable negative effect on graph cuts optimization. This is why the global optimum obtained by graph cuts often does not lead to the most desirable result. However, by using a sub-graph and blocking those unknown regions far from the labeled regions, the background interference can be much reduced, and hence better results can be obtained under the same amount of user interaction. Second, the algorithm is run on the sub-graph that comprises object/background regions and the surrounding un-segmented regions, thus the computational cost is significantly less than running graph cuts on the whole graph which is based on image pixels. Third, as a graph cuts based region merging algorithm, our method obtains the optimal segmentation on each sub-graph. In interactive image segmentation, user input information helps to enhance the discontinuities between object and background by constructing color data models [9], which represent object and background respectively. Some simple methods such as color histograms can be used to calculate these models. In this work, the construction of the object and the background color models are obtained from the most confident labels by a learned classifier. This scheme automatically collects more reliable information for the next round of segmentation.

Although the user input is helpful in steering the segmentation process to reduce the ambiguities, too much interaction will lead to a tedious and time-consuming work. If the object is in a complex environment from which the background can not be trivially subtracted, a significant amount of interaction may be required. Moreover, the complex content of an image also makes it hard to provide user guidance for accurate segmentation while keeping the interaction as less as possible. Therefore, some algorithms allow further user edit based on the previous segmentation results [9, 10, 11, 25] until the desired result is achieved. In comparison to the traditional graph cuts algorithm, the proposed method is able to reduce the amount of user interaction needed for a desirable segmentation result, or that given a fixed amount of user interaction it increases the quality of the final segmentation result. Experiments show that with poor initialization (i.e. user inputs), the segmentation results of standard graph cuts algorithm might be far from what we expect, while the proposed method can still offer good results. In addition, much better segmentation results can be achieved by the proposed method for images with complex background.

The rest of this paper is organized as follows. A brief review of standard graph cuts algorithm is in Section 2. An iterated conditional mode (ICM) on graph cuts is proposed in Section 3, followed by the region merging based localized graph cuts algorithm. Section 4 presents experimental results of the proposed method on 50 benchmark images in comparison with standard graph cuts and Grabcut. Finally the conclusion is made in Section 5.

2. Image Segmentation by Graph Cuts

Image segmentation can be naturally taken as a labeling problem. Given a set of labels L and a set of sites S (e.g, image pixels or regions),our goal is to assign each of the sites $p \in S$ a label $f_p \in L$. The graph cuts framework proposed by Boykov and Jolly [7] addresses the segmentation on binary images, which solves a labeling problem with two labels. The label set is $L = \{0, 1\}$, where 0 corresponds to the background and 1 corresponds to the object. Therefore, labeling is a mapping from S to L and is denoted by $f = \{f_p | f_p \in L\}$, i.e. label assignments to all pixels [26]. An energy function in a "Gibbs" form is formulated as:

$$E(f) = E_{data}(f) + \lambda E_{smooth}(f) \tag{1}$$

The data term E_{data} consists of constraints from the observed data and measures how sites like the labels that f assigns to them. It is usually defined to be

$$E_{data}(f) = \sum_{p \in S} D_p(f_p) \tag{2}$$

where D_p measures how well label f_p fits site p. For example, we can use intensities of marked sites (seeds) to learn the histograms for the object and the background intensity distributions $Pr(I|"obj")$ and $Pr(I|"bkg")$. Then D_p can be expressed as follows:

$$D_p("obj") = -lnPr(I_p|"obj") \tag{3}$$

and

$$D_p("bkg") = -lnPr(I_p|"bkg") \tag{4}$$

D_p is the penalty of assigning the label f_p to site $p \in S$. The negative log-likelihoods should be small if p likes f_p and vice versa. E_{smooth} is called the smoothness term and measures the extent to which f is not piecewise smooth. The typical form of E_{smooth} is

$$E_{smooth} = \sum_{\{p,q\} \in \mathcal{N}} V_{pq}(f_p, f_q) \tag{5}$$

where $\mathcal{N}$ is a neighborhood system, such as a 4-connected neighborhood system or an 8-connected neighborhood system. The smoothness term typically used for image segmentation is the Potts Model [34], which is

$$V_{pq}(f_p, f_q) = \omega_{pq} \times T(f_p \neq f_q) \tag{6}$$

where:

$$T(f_p \neq f_q) = \begin{cases} 1 & \text{if } f_p \neq f_q \\ 0 & \text{otherwise} \end{cases}$$

The model (6) is a piecewise constant model because it encourages labelings consisting of several regions where sites in the same region have the same labels. In image segmentation, we want the boundary to lie on the intensity edges in the image. A typical choice for $\omega_{p,q}$ is as follows:

$$\omega_{pq} = e^{-\frac{(I_p - I_q)^2}{2\delta^2}} \cdot \frac{1}{dist(p,q)} \tag{7}$$

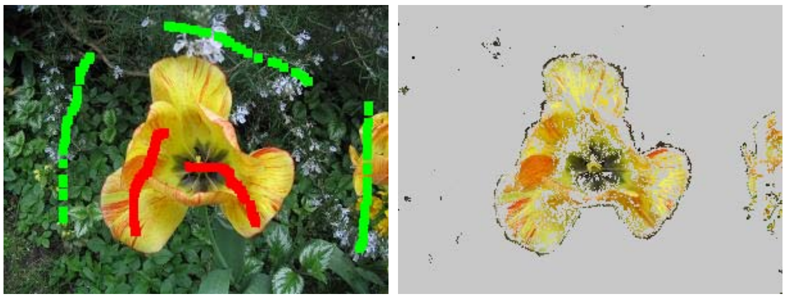
where I_p and I_q are the intensities of sites p and q. For color images, I_p and I_q can be taken as the sum of values in LAB channels. $dist(p,q)$ is the distance between sites p and q. Parameter δ is related to the level of variation between neighboring sites within the same object. The parameter λ is used to control the relative importance of the data term versus the smoothness term. If λ is very small, only the data term matters. In this case, the label of each site is independent from the other sites. If λ is very large, all the sites will have the same label. Minimization of the energy function can be done using the min-cut/max-flow algorithm as described in [7].

Now we need to construct a graph corresponding to the energy function (1). There are two additional nodes: the source terminal s and the sink terminal t, representing the object and the background respectively. Each node in the graph is connected to s and t by two t-links. And each pair of neighboring nodes is connected by an n-link. The weights of t-links for seed pixels can be seen as hard constraint imposed on the segmentation. In initialization, the user will mark some pixels as the object or the background so that these pixels will keep their initial labels in the final result. If pixel p is marked as an object label, the edge between p and s should be set to 0 and edge between p and t set to infinity. N-links correspond to the penalty for discontinuity between the two neighboring pixels. They are derived from the smoothness term E_{smooth} in energy function (1). And the weight of a t-link corresponds to a penalty for assigning the label to the pixel. It will be derived from the data term E_{data} in the energy function (1).

3. Iterated Region Merging with Localized Graph Cuts

3.1. Initial Segmentation by Modified Watershed Algorithm

In the original graph cuts algorithm [7], the segmentation is directly performed on the image pixels. There are two problems for such a processing.



(a) (b)

Figure 1: (a) Original image with user input seeds. The background seeds are in green, and object seeds are in red. (b) The segmentation results by standard graph cuts.

First, each pixel will be a node in the graph so that the computational cost will be high; second, the segmentation result may not be smooth, especially along the edges. Fig.1 shows an example of the graph cuts segmentation result. It can be seen that although there should be clear boundary between the object and background, the graph cuts fails to give a smooth segmentation map by labeling some object pixels as background, or vice versa. Actually, in the early work of Wu and Leahy [31], it was noticed that the minimum cut criteria favored cutting small sets of isolated nodes in the graph.

To alleviate this problem, Veksler [17] included a shape constraint in the graph cuts energy function, which encourages a long object boundary. Some other segmentation criterions were also proposed to solve this problem, such as normalized cuts [32] and ratio cut [33]. In this paper, we adopted a relatively simple but effective strategy to solve this problem by introducing some low level image processing techniques to graph cuts. In [25], Li et al. used watershed [24] for initial segmentation to speed up the graph cuts optimization process in video segmentation. With such initialization, the image can be partitioned into many small homogenous regions, and then each region, instead of each pixel, is taken as a node in the graph. In this way the computational cost can be reduced significantly, while the object boundary can be better preserved. The watershed technique is also used in this paper with some modification.

Watershed algorithm produces coherent over-segmented regions which preserve most structures of the interest object. However, the standard watershed algorithm is very sensitive to noise and thus leads to severe over-segmentation (see Fig. 2(b)). There are some edge-preserving smoothing techniques, such as median filtering, can help to reduce noise and trivial structures. Therefore, to reduce over-segmentation, we apply median filtering on the gradient image before conducting the watershed algorithm. Fig. 2 shows an example. Fig. 2(a) is the gradient image of the original image in Fig. 1(a) and Fig. 2(b) is the watershed segmentation of it. Clearly, there is a severe over-segmentation in Fig. 2(b). Such small regions are not reliable for calculating the region statistics and they will also increase the computational cost in our region merging algorithm Fig. 2(c) is the median filtering output of the gradient image in Fig. 2(a), and Fig. 2(d) is the watershed segmentation result on it. We see that the over-segmentation is significantly reduced, while the contour of the object is well preserved. Note that we can use more sophisticated initial segmentation techniques in the proposed method. To weaken the importance of initial segmentation, the watershed algorithm is adopted for its simplicity.

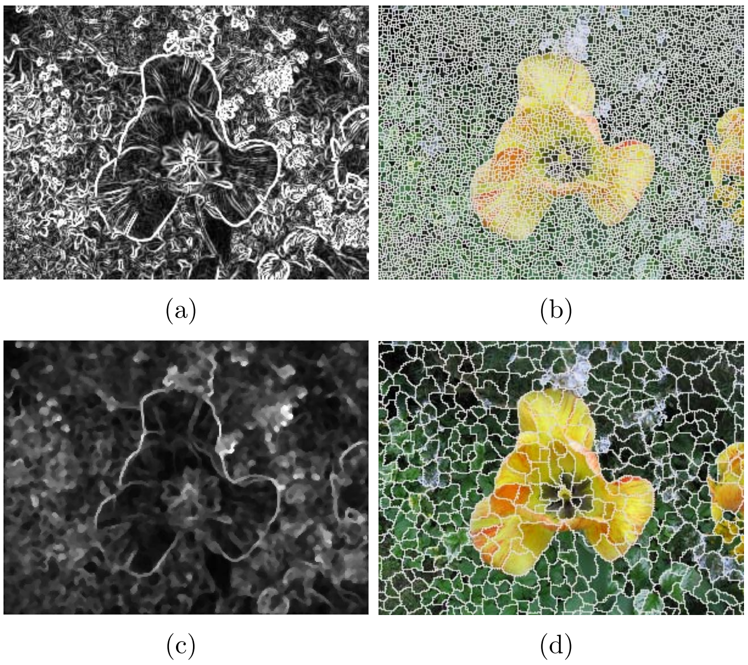


Figure 2: Initial segmentation using modified watershed algorithm. (a) is the gradient image of Fig. 1(a); (c) is the median filtering result of (a); (b) and (d) are the watershed segmentation results of (a) and (c) respectively. We see that the over-segmentation is significantly reduced in (d).

3.2. Iterated Conditional Mode

Although graph cuts technique provides an optimal solution to the energy function (1) for image segmentation, the complex content of an image makes it hard to precisely segment the whole image all at once. In the proposed region merging based segmentation algorithm, the one-shot minimum cut estimation algorithm is replaced by a novel iterative procedure, in which the object/background distributions are updated according to the previous segmentation results and new nodes are added until the whole image is segmented. This problem is studied in a way like the iterated conditional mode (ICM) proposed by Besag [27], where the local conditional probabilities is maximized sequentially.

In computer vision, an image can be represented by a graph $G =< V, E >$, where V is a set of nodes corresponding to image elements (e.g. pixels, regions), and E is a set of edges connecting the pairs of nodes. We say two nodes are incident with an edge and that these nodes are adjacent or neighbors of each other. Edge weights of the graph are computed as the dissimilarity between the connected nodes (e.g. the distance of region histograms). A sub-graph $G' =< V', E' >$ can be defined such that $V' \subseteq V$ and $E' \subseteq E$. In this paper, we consider image regions as the graph nodes, and the neighborhood of a node in V' corresponds to its adjacent regions in the image. Inspired by ICM, we consider the graph-cuts algorithm in a "divide and conquer" style: finding the minima on the sub-graph and extending the sub-graph successively until reach the whole graph. The proposed method works iteratively, in place of the previous one-shot graph cuts algorithm [7].

Given the observed data d_p of site p, the label f_p of site p and the set of labels $f_{S-\{p\}}$ which is at the site in S-$\{p\}$, where $f_p \in L$ and S-$\{p\}$ is the set difference. We sequentially assign each f_i by maximizing conditional probability $P(f_p|d_p, f_{S-\{p\}})$ under the MAP-MRF framework. There are two assumptions in calculating $P(f_p|d_p, f_{S-\{p\}})$. First, the observed data $d_1, \ldots, d_m$ are conditionally independent given f and that each d_p depends only on f_p. Second, f depends on labels in the local neighborhood, which is Markovianity, i.e. $P(f_p|d_p, f_{S-\{p\}}) = P(f_p|f_{N_p})$, where N_p is a neighborhood system of site p. Markovianity depicts the local characteristics of labeling. With the two assumptions we have:

$$P(f_p|d_p, f_{S-\{p\}}) = \frac{P(d_p|f_p) \cdot P(f_p|f_{N_p})}{P(d)} \tag{8}$$

where $P(d)$ is a normalizing constant when d is given. There is:

$$P(f_p|d_p, f_{S-\{p\}}) \propto P(d_p|f_p) \cdot P(f_p|f_{N_p}) \tag{9}$$

where $\propto$ denotes the relation of direct proportion. The posterior probability satisfies:

$$P(f_p|d_p, f_{S-\{p\}}) \propto e^{-U(f_p|d_p, f_{N_p})} \tag{10}$$

where $U(f_p|d_p, f_{N_p})$ is the posterior energy and satisfies:

$$\begin{aligned} U(f_p|d_p, f_{N_p}) &= U(d_p|f_p) + U(f_p|f_{N_p}) \\ &= U(d_p|f_p) + \sum_{p' \in N_p} U(f_p|f_{p'}) \end{aligned} \tag{11}$$

$U(d_p|f_p)$ is the data term corresponding to function (1), and $\sum_{p' \in N_p} U(f_p|f_{p'})$ is the smoothness term which relates to the number of neighboring sites whose labels $f_{p'}$ differ from f_p. The MAP estimate is equivalently found by minimizing the posterior energy:

$$f^{k+1} = \arg\min_f U(f|d, f_N^k) \tag{12}$$

where f_N^k is the optimal labeling of graph nodes obtained in previous k iterations. The labeling result in each iteration is reserved for later segmentation. This process is done until the whole image is labeled.

3.3. Iterated Region Merging

The proposed iterated region merging method starts from the initially segmented image by the modified watershed algorithm in Section 3.1. Fig. 3 illustrates the iterative segmentation process by using an example. In each iteration, new regions which are in the neighborhood of newly labeled object and background regions are added into the sub-graph, while the other regions keep their labels unchanged.

The proposed algorithm is summarized in Table 1. The inputs consist of the initial segmentation from watershed segmentation and user marked seeds. The object and background data models are updated based on the labeled regions from the previous iteration. In Section 3.4, the algorithm to construct data models will be discussed in detail.

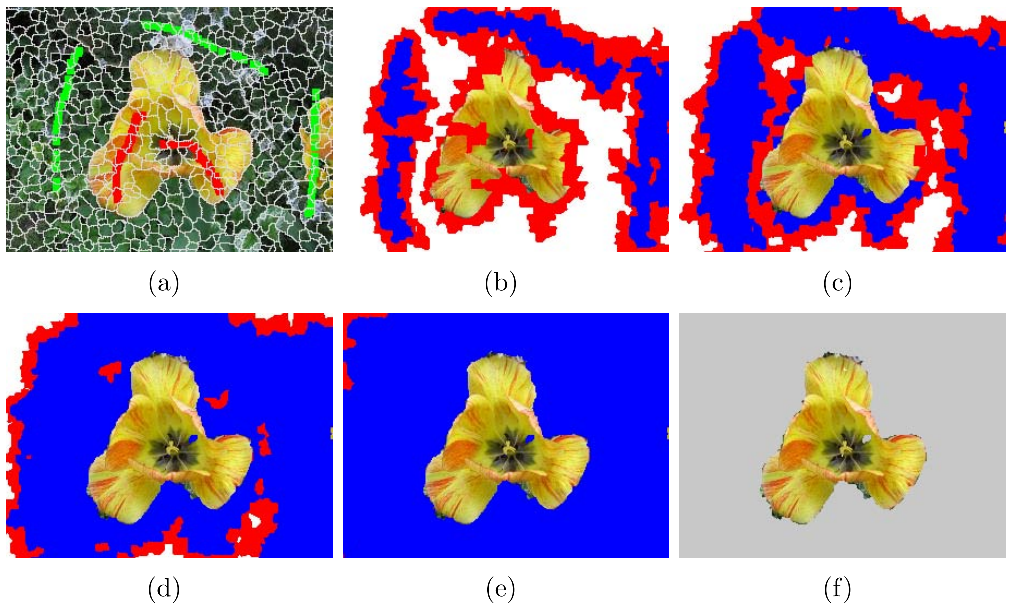


Figure 3: The iterative segmentation process. (a) Initial segmentation. (b)-(e) show the intermediate segmentation results in the 1st, 2nd, 3rd and 4th iterations. The newly added regions in the sub-graphs are shown in red color and the background regions are in blue color. We can see the target object is well segmented from the background in (f).

Table 1: Iterated region merging with localized graph cuts

```
Algorithm1 : RegionMergingGraphCuts()
Input:
– Initial segmentation of the given image.
– User labeled object regions $R_o$ and background regions $R_b$.
Output: Segmentation result.
----------------------------------------------------------------
1. Build object and background data models based on labeled re-
   gions $R_o$ and $R_b$.
2. Build subgraph $G' =< V', E' >$, where $V'$ consist of $R_o$, $R_b$ and
   their adjacent regions.
3. Update object and background data models using the SelectLa-
   bels() algorithm (refer to section 3.4).
4. Use graph cuts algorithm to solve the min-cut optimization on
   $G'$, i.e.
              $\arg\min_{f} U(f|d, f_N^k).$
5. Update object regions $R_o$ and background regions $R_b$ according
   to the labeling results from step 4.
6. Go back to step 2, until no adjacent regions of $R_o$ and $R_b$ can
   be found.
7. Return the segmentation results.
```

3.4. Update Object/Background Models

Incorporating user input information in segmentation is one of the most interesting features of graph cuts method [8]. There is a lot of flexibility in how the information can be used to adjust the algorithm for a desired segmentation, for example, initializing the algorithm or editing the results. With the given information, the object and background models can be learned for formulating the data term in function (1), which describes how well label f_p fits site p. In step 2 of the proposed Algorithm 1, the models are updated based on the previously labeled regions. However, if all the labeled regions are used to update the models, the misclassified regions will probably reinforce themselves in the next round of iteration. Therefore, we propose a semi-supervised approach in which the labeled regions in the $(i-1)th$ iteration are partially selected to be the seeds for the ith iteration. This model updating process is independent of the graph cuts optimization algorithm, aiming is to increase the confidence levels of the color models. The main idea of our object/background models updating process can be summarized as follows: in each iteration, a set of confident labels is chosen by a semi-supervised approach, such that the corresponding regions are taken as confident regions. Based on these confident regions, new object/background models are constructed for the graph cuts segmentation, as an integral step of the proposed Algorithm 1.

There are a number of semi-supervised algorithms which use both labeled and unlabeled data to build classifiers. With the merits of less human effort and higher accuracy, they are of great interest in practice. The Yarowsky algorithm [28] is a well-known semi-supervised algorithm, which is widely used in computational linguistics. Some variants of the original Yarowsky algorithm [29, 30] were also developed to optimize specific objective functions. In this section, we adopt it to build better object/background models for the proposed iterated segmentation algorithm.

Suppose $\phi_x(j)$ is the probability that instance x belongs to the jth class, and $\pi_x(j)$ is the score of the model in predicting label j for the region x. An object function based on cross-entropy is defined as [29]:

$$l(\phi,\pi) = \sum_{x\in X} H(\phi_x||\pi_x) = \sum_{x\in X}\sum_{j} \phi_x(j) log \frac{1}{\pi_x(j)} \tag{13}$$

The minimization of function (13) encourages the unlabeled data becomes labeled, and its assigned label agrees with the model prediction. Since the

Table 2: Algorithm of label selection for constructing color model in the *ith* iteration

```
Algorithm2: SelectLabels()
Input:
– Seeds regions $Y^0 = R_o \cup R_b$
– Labeled regions $X$ after the 1st iteration, which contain $Y^0$ and their adjacent regions $\perp$.
Output: labeling $Y^{i+1}$.
1. For $i \in \{0, 1, \ldots\}$ do.
2. $\wedge^i = \{x \in X | Y^i \neq \perp\}$.
3. Train classifier on $(\wedge^i, Y^i)$; resulting in $\pi^i$.
4. For each example $x \in X$
   4.1 set $\hat{y} = argmax_j \pi_x^{i+1}(j)$
   4.2 set
       $Y^{i+1} = \begin{cases} Y_x^0 & \text{if } x \in \wedge^0 \\ \hat{y} & \text{if } x \in \pi^i \vee \pi_x^{i+1}(\hat{y}) > 1/L \\ \perp & \text{otherwise} \end{cases}$
5. If $Y^{i+1} = Y^i$, stop. Otherwise, go 1.
6. return $Y^i$.
```

goal is to build color models based on previously labeled regions, we would like to choose the regions whose predictions are most confident according to the Yarowsky algorithm. With the fact that seeds regions in the $(i-1)th$ iteration are confident for the graph cut segmentation, we only have to decide which are the confident regions resulting from the graph cut in the $(i-1)th$ iteration. The Algorithm 2 in Table 2 describes the process of how to choose the labeled regions in the $(i-1)th$ iteration for constructing color models of the *ith* iteration, which is corresponding to step 3 in Algorithm 1.

In Algorithm 2, the outer loop is given a seed set Y^0 to start with. In step 2, a labeled training set $\wedge^i$ is constructed from the most confident predictions Y^i. The score $\pi_x(j)$ is related to all the feature values in a sample x , and is given by:

$$\pi_x(j) = \frac{1}{|F_x|} \sum_{f \in F_i} \theta_{fj} \tag{14}$$

where $\theta_{fj} = \frac{|\wedge_{fj}| + 1/L|V_f|}{|\wedge_f| + |V_f|}$, $|F_x|$ is the number of features of a region x, L is the number of labels, $| \wedge_{fj} |$ is the number of regions with label j and feature f;

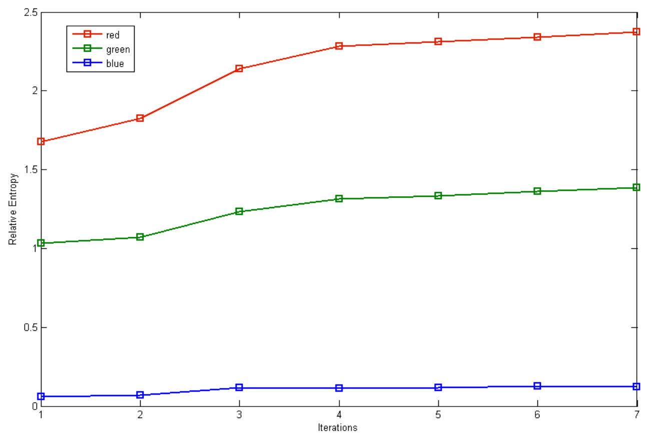


Figure 4: Relative entropy of the object and background distributions (Fig.3) in different iterations. The three plots represent the red, green and blue color channels respectively.

$|V_f|$ and $|\wedge_f|$ are respectively the numbers of unlabeled and labeled regions that have feature f. The feature used here is the average RGB color of a region. Abney [29] proved that the definition of score $\pi_x(j)$ can promise the object function (13) to decrease with the iteration number until it reaches a minimum. The predicted label for region is given in step 4.1 in Algorithm 2, where it is assumed that the classifier makes confidence-weighted predictions.

To check the relationship between the object and background distributions, we use the relative entropy to evaluate the distance between them. It is defined as the Kullback-Leibler distance from the distribution of foreground to that of the background, i.e. $D_{KL}(p||q) = \sum_{x \in X} p(x) log \frac{p(x)}{q(x)}$, where $p(x)$ and $q(x)$ are the probability density functions of the object and background respectively. Fig.4 shows the value of relative entropy in all the 7 iterations for the image in Fig.3(a). As the value of relative entropy goes up from the first iteration, the data models of the object and background become more and more distinguishable. This leads to a higher probability of well separating the object from the background.

In the proposed algorithm, segmentation is obtained on different levels of sub-graphs. In light of graph cuts, the segmentation keeps the property of global optimality on each sub-graph. Adding new seeds according to the previous optimal labeling, it increases the amount of useful information that can be used for further segmentation while avoiding introducing much inter-

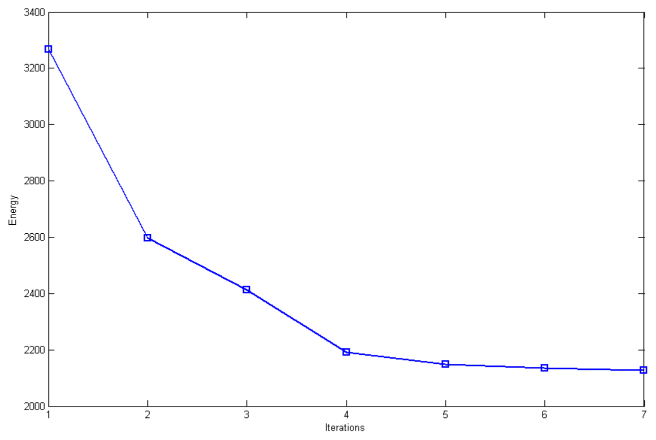


Figure 5: The energy evolution of the segmentation results in Fig. 3. Graph cuts energy decreases in the iterated segmentation process.

ference information from unknown regions. Fig.5 shows the energy evolution of the image segmentation process in Fig.3. Fig.6 shows another example. With the user input seeds (Fig.6(g)), the amount of object and background seeds increases automatically based on the segmentation result in each iteration. It is straightforward that our algorithm guarantees the monotonic decrease of energy because iterative minimization can be taken as a multi-step minimization of the total energy.

4. Experimental Results

We evaluate the segmentation performance of the proposed method in comparison with the graph cuts algorithm [7] and *GrabCut* [9]. Since we use watershed for initial segmentation, for a fair comparison, we also extend the standard graph cuts to a region based scheme, i.e. we use the regions segmented by watershed, instead of the pixels, as the nodes in the graph. *GrabCut* algorithm is also an interactive segmentation technique based on graph cuts and has the advantage of reducing user's interaction under complex background. It allows the user to drag a rectangle around the desired object. Then the color models of the object and background are constructed according to this rectangle. Hence in total we have four algorithms in the experiments: the pixel based graph cuts (denoted by GC_p), the region based graph cuts (GC_r), the *GrabCut* and the proposed iterated region merging method

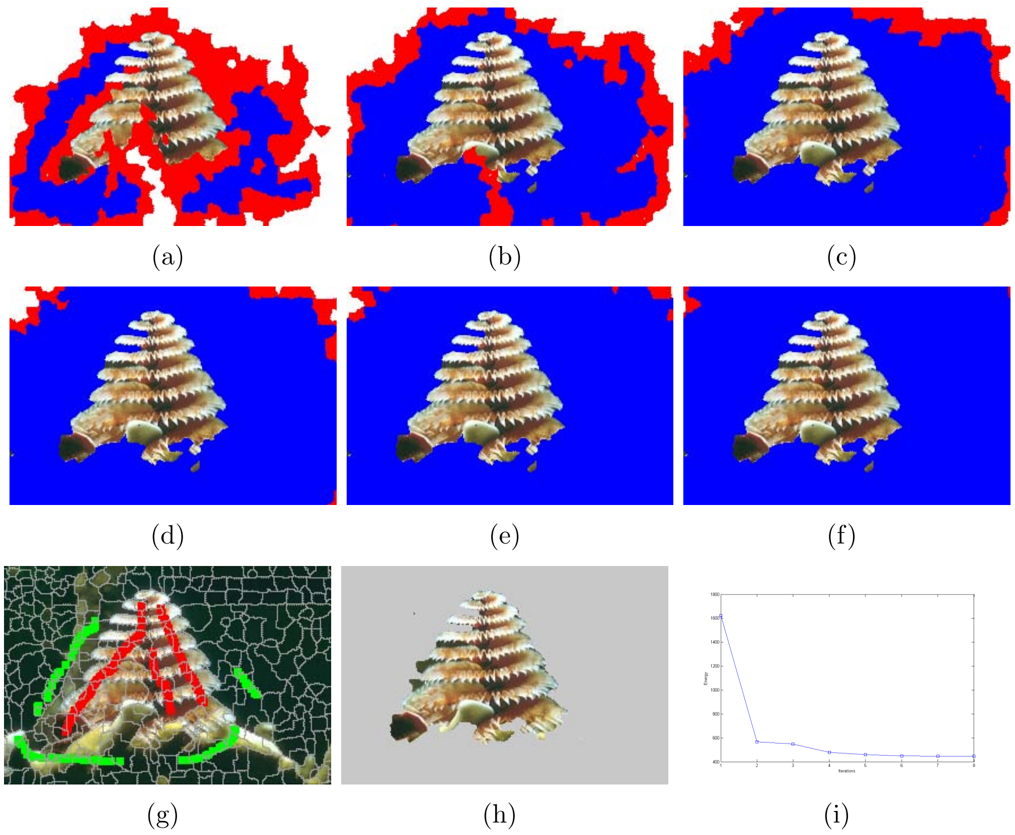


Figure 6: Another example of energy evolution. (a)-(f) show the object and background seeds in different iterations based on the user input seeds shown in (g). (h) shows the final segmentation result, and (i) shows the energy values, which are calculated on the whole graphs by using the seeds obtained in each iteration. We see that the energy decreases monotonically.

with localized graph cuts (denoted by *IRM-LGC*). The software of the proposed method can be downloaded at http://www4.comp.polyu.edu.hk/ cslzhang/code.htm.

In Sections 4.1 and 4.2, the four algorithms are evaluated qualitatively. In Section 4.3, the segmentation results are evaluated quantitatively. Some discussions are made in Section 4.4. Our experiment database contains 50 benchmark test images selected from online resources [2] [3], where 10 of them contain objects with simple background and the others are images with relatively complex background. Every image in our database has a figure-ground assignment labeled by human subjects.

4.1. Comparison with Graph Cuts

In this subsection, the segmentation results are compared between the proposed algorithm and algorithms GC_p and GC_r. Note that GC_r algorithm is used as the first step in lazy snapping [10]. This experiment can thus partially compare the performance of lazy snapping and *IRM-LGC*. However, a direct comparison of the two methods is not a fair choice, since lazy snapping has another refinement step which adjusts the mis-located boundaries produced by the first step. Fig.7 shows some images with simple background. In these examples, it is relatively easy to extract the objects from the background. Therefore some of the results by GC_p or GC_r are not too bad, while the proposed method works better.

Extracting objects of interest from complex background is a more challenging task. Fig.8 shows some images with relatively complex background and their segmentation results. In these images, the objects contain weak boundaries due to poor contrast and noise, and the colors of some background regions are very close to those of the objects. Given the same amount of user input, the proposed *IRM-LGC* achieves much better segmentation results than the GC_p and GC_r algorithms.

4.2. Comparison with GrabCut

Fig.9 compares the results of *IRM-LGC* and *GrabCut*. The left column shows the original images with the seeds points. The middle column shows the segmentation results of *GrabCut*. Implementation of *GrabCut* uses 5 GMMs to model RGB color data and parameter λ is set to be 50. The right

[2] http://www.research.microsoft.com/vision/cambridge/segmentation/
[3] http://www.cs.berkeley.edu/projects/vision/grouping/segbench/

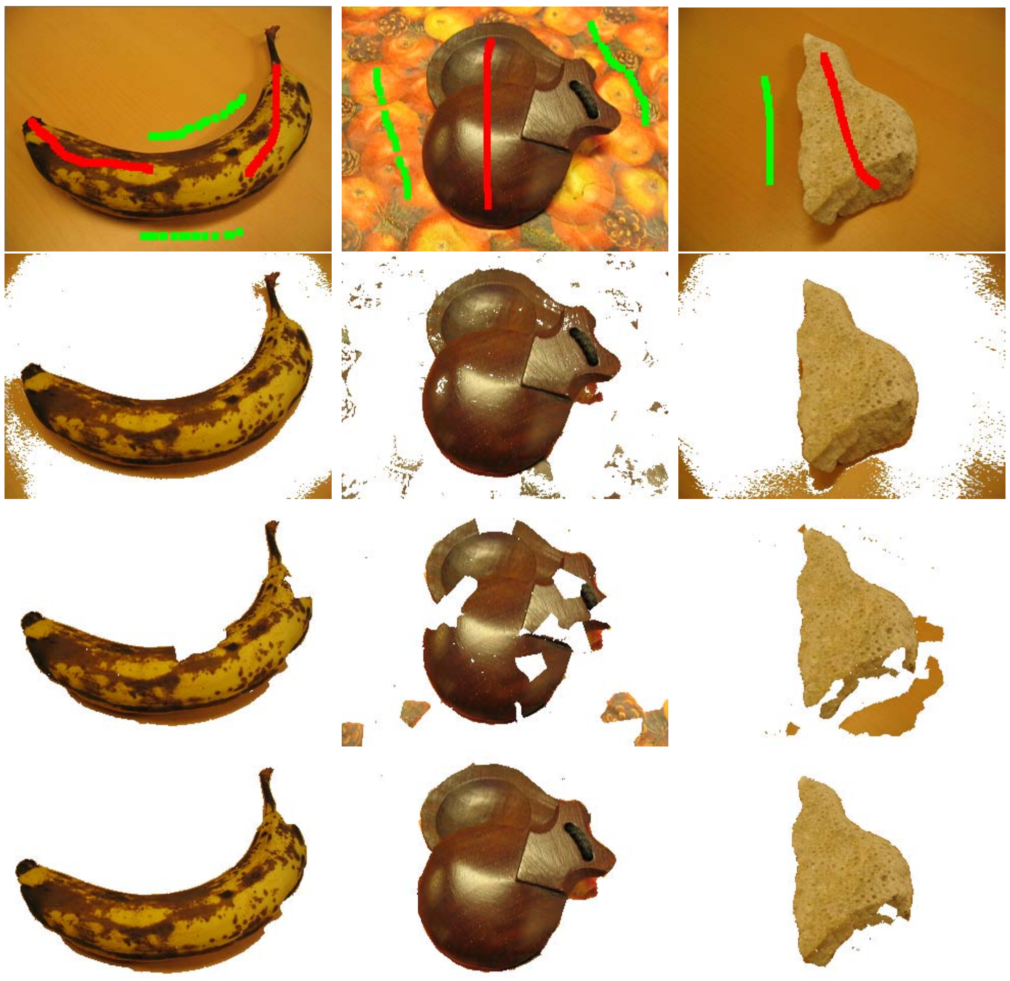


Figure 7: Segmentation results of images with simple background. The first row shows the original images with seeds. Red strokes are for the object and the green strokes are for the background. The second to the forth row show the segmentation results by GC_p, GC_r and $IRM\text{-}LGC$ respectively.

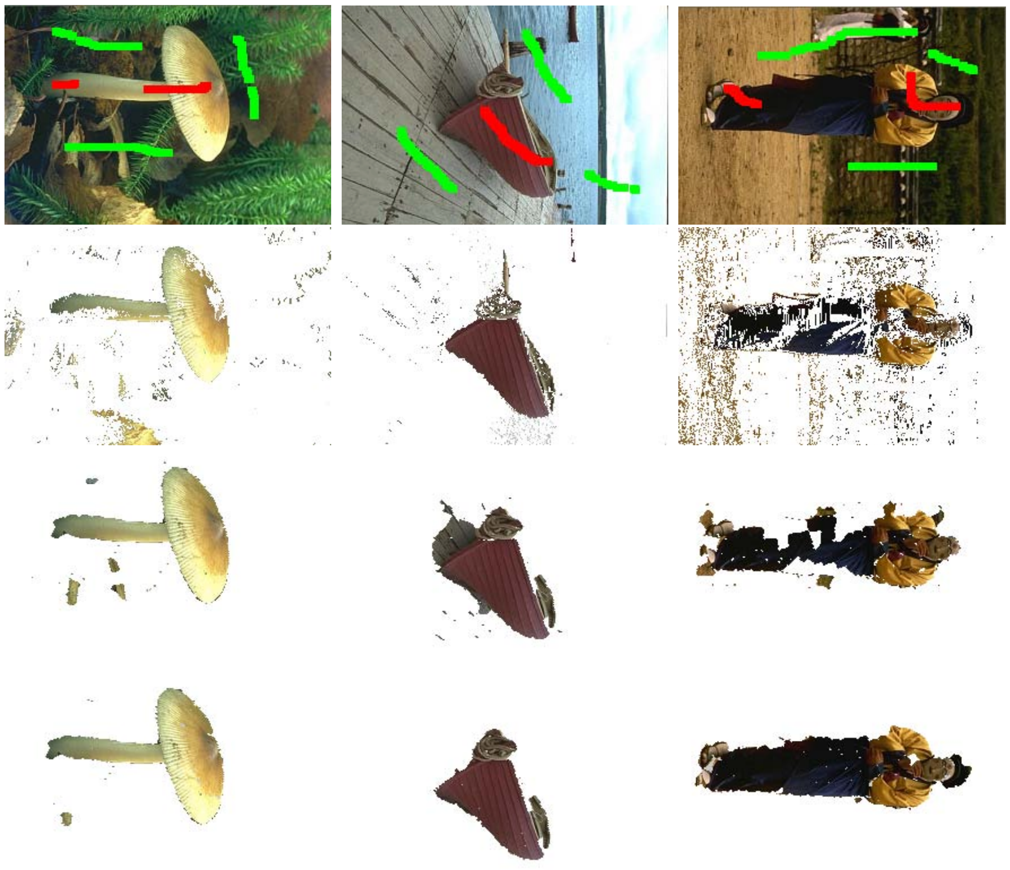


Figure 8: Segmentation results of images with complex background. The first row shows the original images with seeds. From the second to the forth row, there are the segmentation results obtained by GC_p , GC_r and $IRM\text{-}LGC$ respectively.

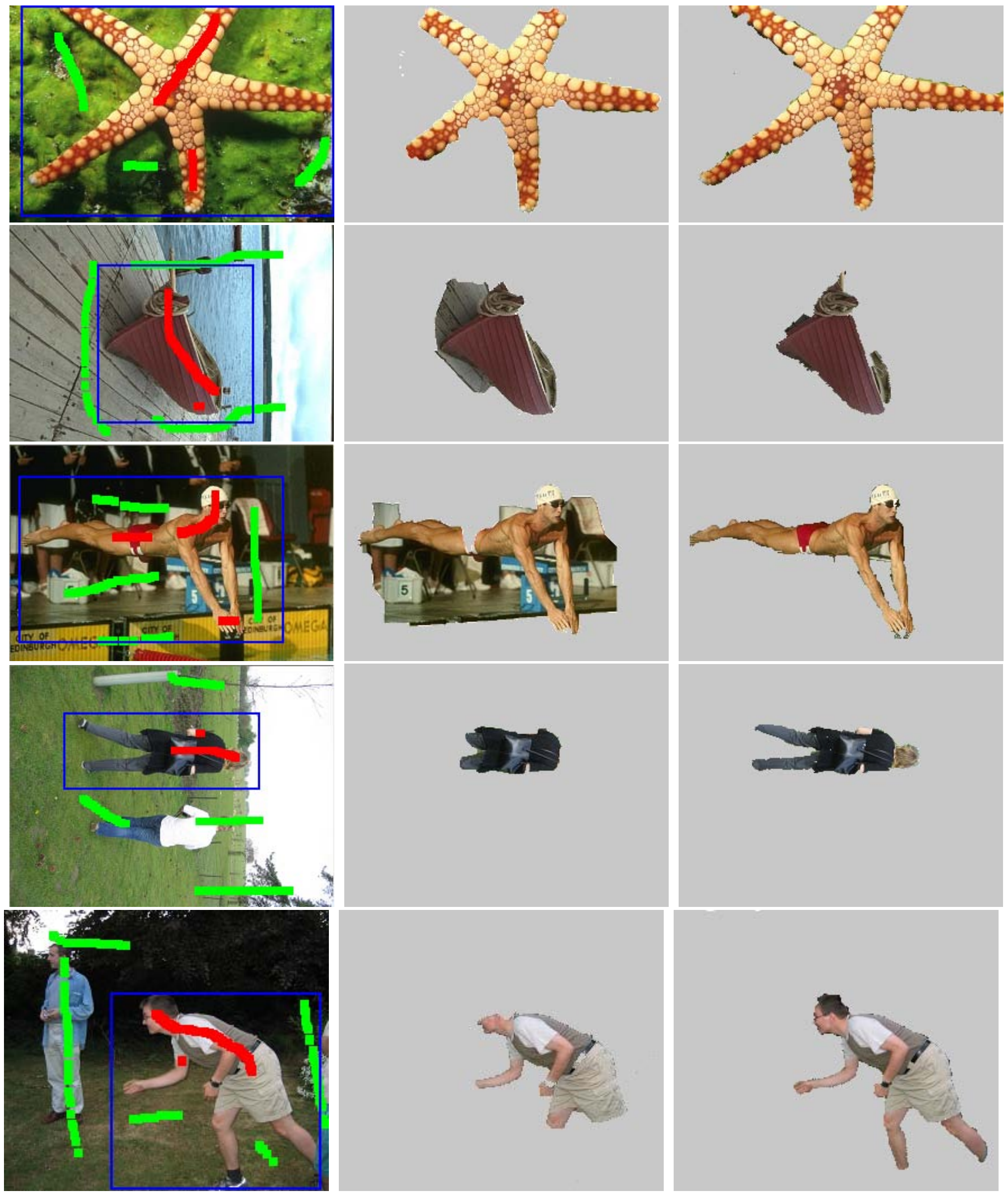


Figure 9: Segmentation results by *GrabCut* and the proposed method. The left column shows the original images with seeds. The blue rectangle is the interaction used in *GrabCut*, while the red and green strokes are the object and background seeds used in the proposed algorithm. The middle column shows the results of *GrabCut*. The right column shows results of *IRM-LGC*.

Table 3: The TNF, TPF, FNF and FPF results by different methods.

Algorithms	TPF(%)	FNF(%)	TNF(%)	FPF(%)
GrabCut	83.65	16.35	96.59	3.41
GC_p	82.72	17.28	92.37	7.63
GC_r	88.01	11.99	93.78	6.22
IRM-LGC	91.29	8.71	97.75	2.25

column is results of *IRM-LGC*. When the objects to be segmented contain similar colors with the background, *GrabCut* might fail to correctly segment them. Although our algorithm uses more user interaction than *GrabCut*, this tradeoff leads to more precise segmentation results.

4.3. Quantitative Evaluation

To better evaluate our algorithm, a quantitative evaluation of the segmentations is given by comparing with ground truth labels in the database. The qualities of segmentation are calculated by using four measures: the true-positive fraction (TPF), false-positive fraction (FPF), true-negative fraction (TNF) and false-negative fraction (FNF):

$$TPF = \frac{|A_A \cap A_G|}{|A_G|}, FPF = \frac{|A_A - A_G|}{|\overline{A_G}|}$$

$$TNF = \frac{|\overline{A_A \cup A_G}|}{|\overline{A_G}|}, FNF = \frac{|A_G - A_A|}{|A_G|}$$

where A_G represents the area of the ground truth of foreground and its complement is $\overline{A_G}$; A_A represents the area of segmented foreground by the tested segmentation method. Table 3 lists the results of TPF, FNF, TNF and FPF by the three methods over the 50 test images. We see the proposed method achieves the best TPF, FNF, TNF and FPF results.

As mentioned, the proposed *IRM-LGC* image segmentation method uses a modified watershed algorithm for initial segmentation. The median filtering of the gradient image controls the watershed segmentation output. To examine how the initial segments affect the final result of *IRM-LGC*, we applied the algorithm to different initial segmentation with different granularities, i.e. different numbers and sizes of regions in the initial segmentation. This can be done by changing the filtering times and using different sizes of filter windows. Fig.10 shows an example. The first row shows three initial segmentations by the modified watershed algorithm, where the number of regions is 203, 372 and 1296 respectively. The second row shows the final segmentation results. We can see that segmentation quality is not sensitive to the initial segmentation. Fig.11 compares the segmentation quality of the same image with 42 different initial segmentations, from which we can clearly see that the segmentation results are not influenced much by the initialization.

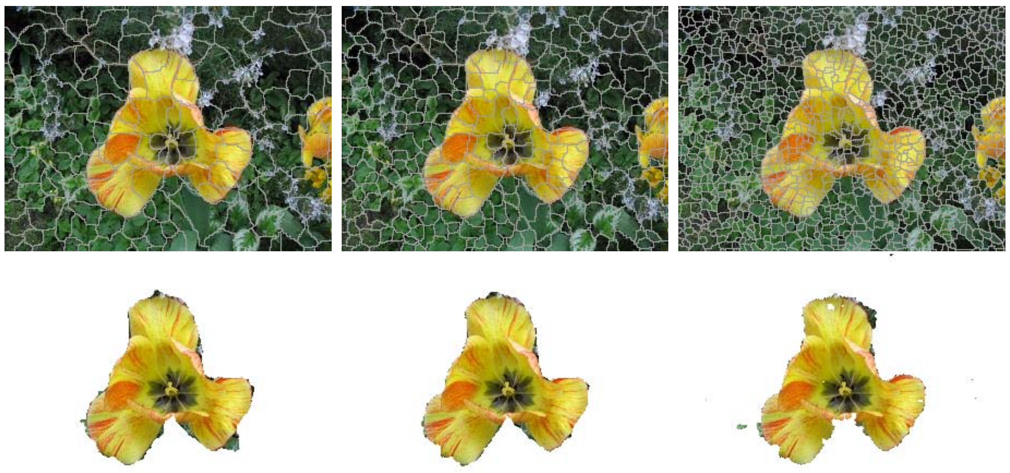


Figure 10: Initial segmentation of an image with different numbers of regions. In the first row, from the left to the right, there are 203, 372 and 1296 regions in the initial segmentation respectively. The second row shows the final segmentation results.

We use the max-flow algorithm [37] to implement the proposed *IRM-LGC* method. The worst case running time complexity for this algorithm is $O(mn^2|C|)$, where n is the number of nodes, m is the number of edges and $|C|$ is the cost of the minimum cut in the graph. In each iteration of *IRM-LGC*, the number of nodes and edges are largely reduced in comparison of the pixel based graph cuts algorithm. Our experiment is implemented on a PC with Intel Core 2 Duo 2.66 GHz CPU, 2GB memory. The running time to perform min-cut/max-flow algorithm on the whole graph which is based on image pixels is around 10-20ms, while the proposed *IRM-LGC* takes far less than 1ms. However, it should be noted that the majority of time for our algorithm is spent on constructing color models and updating the graph ($\sim$ 0.3s per iteration), thus the speedup on the min-cut/max-flow part would be

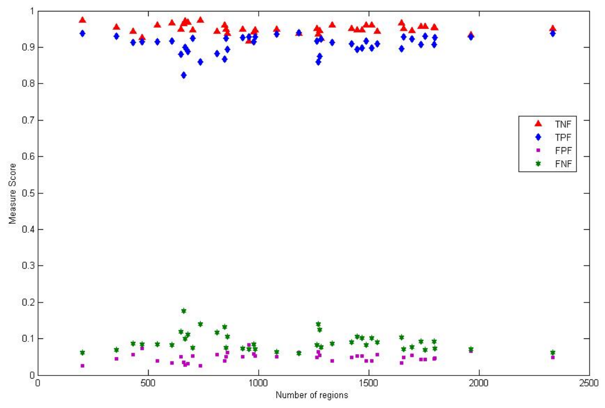
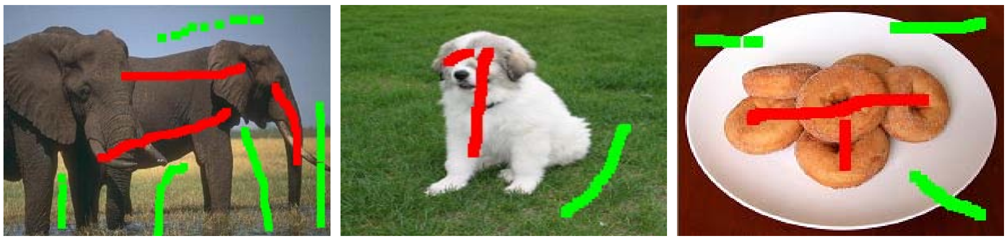


Figure 11: Segmentation qualities vs. initial segmentation in different granularities. For the original image used in Fig.10, 42 different initial segmentations are obtained and used in the proposed algorithm. The segmentation quality is measured by TPF, FPF, TNF and FNF scores.

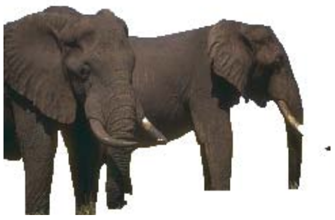
relatively modest for the overall algorithm.

4.4. Discussion

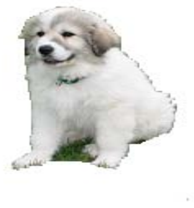
In graph cuts based segmentation, parameter λ is used to weight the data and smoothness terms. In recent years, some literature [35, 36] has studied the parameter selection for graph cuts. There are two problems in graph cuts algorithm about the selection of λ. First, given different images, graph cuts with a fixed value of λ cannot lead to satisfactory segmentation. The appropriate λ values would vary largely among different images, so the user may have to spend a significant amount of time searching for it. Fortunately, the proposed *IRM-LGC* is not sensitive to the selection of λ across different images. This can be illustrated by the following experiments. In practice we found that the region based graph cuts (i.e. GC_r) has similar property to pixel based graph cuts(i.e. GC_p) in parameter selection. Sometimes, GC_r may not lead to satisfying segmentation result throughout the searching space of λ. Thus to study on a more general case, the GC_p is used in the following experiments. Fig.12 shows some examples of the segmentation by GC_p and *IRM-LGC*. For a comparable quality of the segmentation results by the two methods, the best value of parameter λ in GC_p varies a lot for different images (2nd row in Fig.12); however, a constant λ in *IRM-LGC* can lead to satisfying segmentations across different images (3rd row in Fig.12).



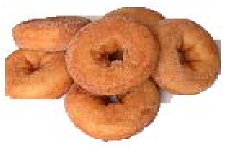
(a) Images with user input seeds



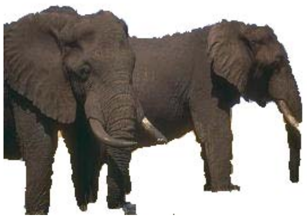
(b) GC_p,$\lambda = 18$



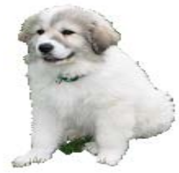
(c) GC_p,$\lambda = 50$



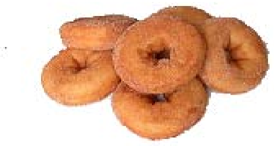
(d) GC_p,$\lambda = 170$



(e) $IRM\text{-}LGC$,$\lambda = 50$



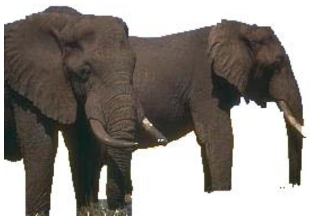
(f) $IRM\text{-}LGC$,$\lambda = 50$

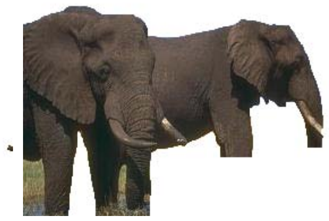


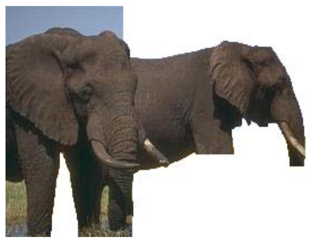
(g) $IRM\text{-}LGC$,$\lambda = 50$

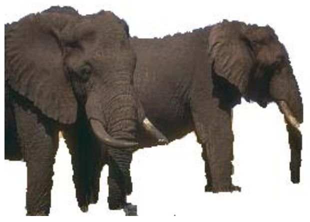
Figure 12: The values of parameter λ in GC_p and $IRM\text{-}LGC$ for different images.

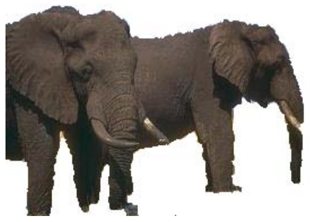
The second problem of standard graph cuts is that different values of λ will result in very different segmentation results for the same image. Fig.13 compares GC_p and *IRM-LGC* by increasing the value of parameter λ. The original image with user input seeds is in Fig.12(a). In Fig.13(a), GC_p produces a relatively good segmentation with $\lambda = 2$. In Fig.13(b) and Fig.13(c), it produces big segmentation errors with $\lambda = 50$ and $\lambda = 150$, respectively. However, by using *IRM-LGC*, we can obtain similar and good segmentation results for a wide range of values: $\lambda = 2$, $\lambda = 50$ and $\lambda = 150$.


(a) GC_p,$\lambda = 2$


(b) GC_p,$\lambda = 50$


(c) GC_p,$\lambda = 150$


(d) *IRM-LGC*,$\lambda = 2$


(e) *IRM-LGC*,$\lambda = 50$

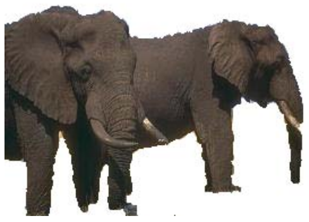

(f) *IRM-LGC*,$\lambda = 150$

Figure 13: Image segmentation with different parameter values. (a-c) show the segmented objects by GC_p and (d-f) show the segmented objects by *IRM-LGC*.

IRM-LGC can reduce greatly the search range of λ. On most of the test images in our database, λ is roughly between 50 and 100 for the proposed method, while for GC_p, the values vary from 10 to 200. An explanation for this is that if the data term in energy function can provide sufficient information for labeling, the graph node does not need a strong relationship with its neighbors. The proposed method gives good object/background models as iteration process goes on, thus the changes of λ for various image can be reduced. This brings much benefit for users in real applications.

Although graph cuts algorithm has relaxed the user input compared with some other algorithms, such as livewire [1], the input seeds cannot always efficiently indicate the background regions, therefore when the connecting regions of the object and background have similar colors, they are still hard to be segmented correctly. It is empirically found that if the input seeds can cover the main features of the object and background, good segmentation result can be obtained. Some promising work [20, 23] has exploited effective methods for arc weight estimation during the seeds marking process. Their work takes into account image attributes and object information in order to enhance the discontinuities between object and background, whereas a visual feedback can be provided to the user for the next action. We will investigate how to incorporate these methods into our work in the future. Fig.14 shows a failure example. The regions circled in red only connect to object regions on the sub-graph, so they are easily assigned to the same label. Moreover, our method uses an initial segmentation to partition the image into regions, incorrect partition in initialization will also affect the final segmentation result.

IRM-LGC is independent of the initial segmentation step. However, under-segmented regions from the naive watershed algorithm cannot be re-partitioned due to the region-merging style of *IRM-LGC*. To reduce the over-segmentation and well keep the coherence of regions, more sophisticated pre-segmentation algorithms can be adopted for the initialization. For example, connected filters with morphological reconstruction operators can eliminate or merge connected components produced by watershed algorithm [21]. Hence they might be used as a more suitable tool for improving the initial segmentation quality than median filters.

As in traditional graph cuts algorithm, in the proposed *IRM-LGC* the user input information is also crucial for obtaining desirable segmentation. Since the newly added seeds in each iteration depend on the segmentation results in the previous iteration, the misclassified regions will probably destroy the rest part of segmentation. In the future work, other strategies of seeds selection will be taken into account. For example, the work in [20] does not use the seeds from previous delineation to re-compute the edge weights. It makes the well-segmented regions unchanged and therefore, the segmentation process becomes more traceable.

5. Conclusion

This paper proposed an iterative region merging based image segmentation algorithm by using graph cuts for optimization. The proposed algorithm starts from the user labeled sub-graph and works iteratively to label the sur-

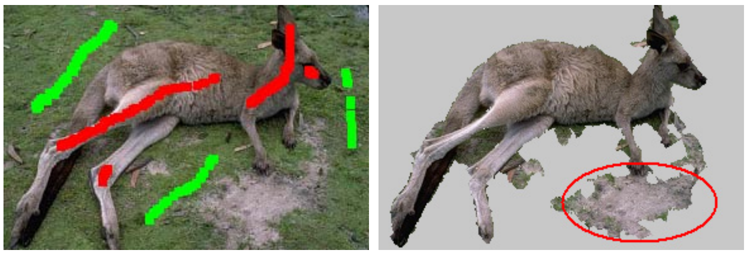


Figure 14: A failure example of the proposed method.

rounding un-segmented regions. It can reduce the interference of unknown background regions far from the labeled regions so that more robust segmentation can be obtained. With the same amount of user input, our algorithm can achieve better segmentation results than the standard graph cuts, especially when extract the object from complex background. Qualitative and quantitative comparisons with standard graph cuts and GrabCut show the efficiency of the proposed method. Moreover, the search space of parameter λ in graph cuts is also reduced greatly by the iterated region merging scheme.

References

[1] A.X. Falcão, J.K. Udupa, S. Samarasekara, and S. Sharma. User-steered image segmentation paradigms: Live wire and live lane. In *Graphical Models and Image Processing*, volume 60, pp. 233–260, 1998.

[2] A.X. Falcão and J.K. Udupa and F.K. Miyazawa. An ultra-fast user-steered image segmentation paradigm: Live-wire-on-the-fly. In *IEEE Trans. on Medical Imaging*,vol.19, no.1,pp. 55–62, Jan. 2000.

[3] K.H. Zhang, L. Zhang, H.H. Song and W. Zhou. Active contours with selective local or global segmentation: a new formulation and level set method. *Image and Vision Computing*, vol. 28, issue 4, pp. 668-676, April 2010.

[4] K.H. Zhang, H.H. Song and L. Zhang. Active Contours Driven by Local Image Fitting Energy. *Pattern recognition*, vol. 43, issue 4, pp. 1199-1206, April 2010.

[5] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 2:321—331,1988.

[6] S. Osher and J.A. Sethian. Fronts propagating with curvature dependent speed: Algorithm based on hamilton jacobi formulations. *Journal of Computational Physics*,79:12–49, 1988.

[7] Y. Boykov and M. P. Jolly. Interactive graph cuts for optimal boundary and region segmentation. In*International Conference on Computer Vision*, volume I, pp.105–112, 2001.

[8] Y. Boykov and G. Funka Lea. Graph cuts and efficient n-d image segmentation. *International Journal of Computer Vision*, 69(2):109–131, September 2006.

[9] C. Rother, V. Kolmogorov, and A. Blake. Grabcut-interactive foreground extraction using iterated graph cuts. In *ACM Transactions on Graphics* (SIGGRAPH),pp. 309–314. 2004.

[10] Yin Li, Jian Sun, Chi-Keung Tang and Heung-Yeung Shum. Lazy Snapping. SIGGRAPH. Vol. 23,pp.303-308,2004.

[11] Jiangyu Liu, Jian Sun, and Heung-Yeung Shum. Paint Selection. SIGGRAPH, 2009.

[12] V. Kolmogorov, A. Criminisi, A. Blake,G. Cross, and C. Rother, Bilayer segmentation of binocular stereo video. In *IEEE Conference of Computer Vision and Pattern Recognition*,pp.407–414. 2005.

[13] Y. Boykov and V. Kolmogorov. Computing geodesics and minimal surfaces via graph cuts. In *International Conference on Computer Vision*, vol. I, pp. 26–33,2003.

[14] V. Kolmogorov and Y. Boykov. What metrics can be approximated by geo-cuts, or global optimization of length/area and flux. In *International Conference on Computer Vision*.vol. I, pp. 564–571,2005.

[15] D. Freedman and T. Zhang. Interactive graph cut based segmentation with shape prior. *In IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pp. 755–762, 2005.

[16] P.Das, O.Veksler, V. Zavadsky, Y. Boykov. Semiautomatic Segmentation with Compact Shape Prior. *Image and Vision Computing*,volume 27, Issue 1-2, pp.206-219, January 2009.

[17] O.Veksler. Star Shape Prior for Graph-Cut Image Segmentation. *Proceedings of the 10th European Conference on Computer Vision*, Part III, pp. 454-467, 2008.

[18] Y. Zeng, D. Samaras, W. Chen, Q. Peng. Topology cuts: A novel min-cut/max-flow algorithm for topology preserving segmentation in N-D images. *Computer Vision and Image Understanding*, Volume 112, Issue 1, pp. 81-90, October 2008.

[19] B. Peng, L. Zhang, J. Yang. Iterated Graph Cuts for Image Segmentation. The Ninth Asian Conference on Computer Vision (ACCV), 2009.

[20] P.A.V. Miranda and A.X. Falcão and J.K. Udupa. Synergistic Arc-Weight Estimation for Interactive Image Segmentation using Graphs. *Computer Vision and Image Understanding.* vol. 114, no. 1, pp. 85–99, 2010.

[21] A.X. Falcão and J. Stolfi and R.A. Lotufo. The Image Foresting Transform: Theory, Algorithms, and Applications. *IEEE Trans. on Pattern Analysis and Machine Intelligence.* vol. 26, no. 1, pp.19–29, 2004.

[22] A.X. Falcão and F.P.G. Bergo. Interactive Volume Segmentation with Differential Image Foresting Transforms. *IEEE Trans. on Medical Imaging*,vol. 23, no. 9, pp.1100–1108,2004.

[23] P.A.V. Miranda and A.X. Falcão. Links Between Image Segmentation Based on Optimum-Path Forest and Minimum Cut in Graph. *Journal of Mathematical Imaging and Vision.* issn 1573-7683, Springer Netherlands, 2009.

[24] L. Vincent, P. Soille. Watersheds in digital spaces: an efficient algorithm based on immersion simulations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13, 6, pp.583-598, 1991.

[25] Y. Li, J. Sun, H. Shum Video Object Cut and Paste. *In Proceedings of ACM SIGGRAPH*, Vol. 24, pp. 595-600, 2005.

[26] Y. Boykov, O. Veksler, R. Zabih. Efficient Approximate Energy Minimization via Graph Cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 12, p. 1222-1239, November 2001.

[27] J.Besag. On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society*, series B 48,259-302, 1986.

[28] D. Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*, 189–196,1995.

[29] S. Abney. Understanding the Yarowsky Algorithm. *Computational Linguistics*, 30(3):365-395, 2004.

[30] G.Haffari, A. Sarkar. Analysis of Semi-Supervised Learning with the Yarowsky Algorithm. *Technical Report*,2007-07.

[31] Z. Wu and R. Leahy. An optimal graph theoretic approach to data clustering Theoryand its application to image segmentation. In*IEEE Transactions on Pattern Analysis and Machine Intelligence*,vol.15,no.11,pp.1101-1113, November,1993.

[32] J. Shi and J. Malik. Normalized cuts and image segmentation. *Conference of Computer Vision and Pattern Recognition*, pages 731-7,1997.

[33] S. Wang, J. M. Siskind. Image Segmentation with Ratio Cut, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(6):675-690, 2003.

[34] Y. Boykov, O. Veksler, and R. Zabih. Markov random fields with effcient approximations. *In IEEE Conference on Computer Vision and Pattern Recognition*, pp.648–655, 1998.

[35] V. Kolmogorov, Y. Boykov, C. Rother. Applications of parametric maxflow in computer vision. *IEEE 11th International Conference on Computer Vision*, pp.1-8, 2007.

[36] B. Peng, O. Veksler. Parameter Selection for Graph Cut Based Image Segmentation. *British Machine Vision conference*(BMVC), 2008.

[37] Y. Boykov, V. Kolmogorov. An Experimental Comparison of Min-Cut/Max-Flow Algorithms for Energy Minimization in Vision. *In IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 9, pp. 1124-1137, Sept. 2004.