Signal Processing: Image Communication 65 (2018) 107–114

Contents lists available at ScienceDirect

Signal Processing: Image Communication

journal homepage: www.elsevier.com/locate/image

Interactive object segmentation in two phases

Ran Shi [a,b,*], King Ngi Ngan [b,c,1], Songnan Li [b,d,2], Hongliang Li [c,3]

[a] School of Computer Science and Engineering, Nanjing University of Science and Technology, China
[b] Department of Electronic Engineering, The Chinese University of Hong Kong, Shatin, Hong Kong
[c] School of Electronic Engineering, University of Electronic Science and Technology of China, Chengdu, China
[d] TCL Corporate Research (Hong Kong) Co., Limited, Hong Kong

ARTICLE INFO

Keywords:
Object segmentation
Contrast
Level

ABSTRACT

This paper addresses the problem of interactive object segmentation with an input rectangle. We present a coarse-to-fine method from region-level segmentation to pixel-level segmentation. In the region-level segmentation, the best combination of adjacent refined superpixels is selected as the coarse segmentation result by measuring its global contrast and tightness degree. Subsequently, we use the coarse segmentation result to aid the construction of the energy function in the pixel-level segmentation. The result can be further refined due to the fusion of region-level and pixel-level segmentation. Experimental results demonstrate that our method can achieve better segmentation performance.

1. Introduction

Object segmentation is an important technique in image processing and computer vision. The performance of many applications depends on the accuracy of the segmentation result, such as image editing [1], content-based image retargeting [2] and compression [3], etc. According to input provided by users, object segmentation can be classified into two categories: one is interactive segmentation, which requires manual input; the other is fully automatic segmentation, such as salient object segmentation [4,5]. Generally, interactive segmentation generates more accurate results by sufficiently utilizing user input.

The goal of interactive segmentation is to accurately extract an object using the least amount of input from the user. Rather than implementing a scribble based input mode [8,9], GrabCut [6] proposed another input mode, i.e. marking a rectangle around the object by clicking only two points. This type of input mode generally gains more popularity due to its convenience. It can also be applied to salient object segmentation because the border of an image can be treated as a defined rectangle [5]. In this paper, we focus on the rectangle based input mode. The pixels within and outside the input rectangle are used to construct an initial object model and an initial background model respectively. However, these two initial models overlap because the input rectangle includes both the object and the background. The overlapped initial models cannot accurately estimate the likelihood of each pixel. In order to rectify this problem, GrabCut proposed an iterative process that jointly refines the segmentation result and models. In one iteration, the segmentation result can be generated using the models, and then the models can be re-constructed according to the latest segmentation result. OneCut [10] approximated the iterative strategy of GrabCut in one Graph Cut [8] by adding a penalty term in the energy function to suppress the affecting induced by the overlap between the models. However, the iteration strategy and the penalty term cannot work well when the overlap is heavy. In [7], some pixels outside the input rectangle are sampled to construct an initial background model. Then, the object model is fitted to 33% of the pixels inside the input rectangle with the smallest probability in the initial background model, and the background model is fitted to 33% of the pixels with the largest probability joined together with the sampled pixels. Although the method may reduce the overlap of models to a certain extent, it is not flexible due to the fixed proportion. Fig. 1 shows one typical example. Since the input rectangle covers most of the background, the segmentation results of [6] and [7] are terrible due to the heavy overlap between the initial models. There is no doubt that quality of the initial models is critical to obtaining a good segmentation result. Thus, an effective interactive segmentation method should reduce the overlap between the models as much as possible. Additionally, in aforementioned methods, the regularization weight in the energy function is pre-defined. It is not adaptive to the quality

* Corresponding author at: School of Computer Science and Engineering, Nanjing University of Science and Technology, China.
E-mail address: rshi@njust.edu.cn (R. Shi).
[1] Fellow, IEEE.
[2] Member, IEEE.
[3] Senior Member, IEEE.

https://doi.org/10.1016/j.image.2018.03.020
Received 15 February 2017; Received in revised form 27 March 2018; Accepted 27 March 2018
Available online 3 April 2018

Fig. 1. An example of an input rectangle covering most part of the background. (a) Original image with the input rectangle; (b) The segmentation result of [6]; (c) The segmentation result of [7].

of initial models. In order to generate better segmentation results, other cues should also be considered. One is the region-level feature, such as superpixels and connectivity prior. The superpixel is widely adopted in image segmentation [11] and object detection [12] tasks. Most pixels within a superpixel share the same attributes. Therefore, the segmentation can incorporate these as soft constraints [11]. In [13,14], the superpixel constraint was introduced into the energy function to further improve segmentation accuracy. However, the quality of the initial model still cannot be well refined when the input is a rectangle. In [5,15], connectivity prior is attributed if most image patches in the background can be easily connected to each other. Another cue is tightness. In [7], tightness is described as that the object should be sufficiently close to each of the sides of the input rectangle. In [16], a natural relationship between tightness and multiple instance constraints is demonstrated and a sweeping line multiple instance learning paradigm is adopted to solve the problem with structural constraints. These two methods sufficiently utilize input topology information to aid segmentation resulting in significant improvement in the accuracy of segmentation results.

In this paper, we propose a two-phase interactive segmentation method. In the first phase, we generate a coarse region-level segmentation result by selecting the best combination of connected refined superpixels in terms of contrast and tightness degree. In the second phase, the object and background models are constructed based on the region-level segmentation result, and then Graph Cut [8] is used to generate a fine pixel-level segmentation result. The regularization weight in the energy function is adapted by the contrast and tightness degree measured in the first phase. One example of the entire procedure of our two-phase segmentation method is shown in Fig. 2. Compared to existing methods [7,10,13,16] that improve the segmentation result by adding various constraints in the pixel-level segmentation, the contributions of our work are as follows:

1. The quality of initial models is improved by separating the object and the background in the region-level segmentation.
2. The regularization weight in the energy function is estimated according to the quality of the region-level segmentation result.
3. The region-level constraint is transferred to the pixel-level segmentation by constructing the models and the adjustment of the regularization weight.

The rest of this paper is organized as follows: Section 2 describes our two-phase interactive segmentation method in detail. Experimental results are presented in Section 3. Finally, Section 4 concludes our paper.

2. Proposed method

2.1. Region-level segmentation

In order to sufficiently reduce the overlap between the object and the background models while imposing a region-level constraint on the segmentation result, we conduct region-level segmentation in the first phase. Given an image I, Felzenszwalb and Huttenlocher' s graph based method (FH) [17] is used to generate an original superpixels map (SM) which only keeps the superpixels mostly located in the input rectangle (IR). The constraint induced by the superpixels is that an object is generally over-segmented into several superpixels, but none straddles its boundaries [18]. Supposed that there is only one object in IR, it can be roughly segmented by aggregating some connected superpixels. So, the region-level segmentation is to select the best combination of superpixels from candidates.

2.1.1. Merging

The contrast can be classified into two categories: global contrast and local contrast. If the local contrast between two superpixels is low, they tend to share a same label (object or background). Therefore, we design a merging step as illustrated in Fig. 3 to integrate this property into the following selection step. Based on SM, we can extract its boundaries and obtain an original binary boundary map BM. Similarly, we can resize I into a smaller one (each side is one kth of the original) and generate a corresponding superpixel boundary map bm_k for the resized image. Since the resizing actually reduces the diversity of pixels, the boundaries of superpixels in bm_k are re-distributed. Some boundaries in BM can still exist in bm_k. However, some will mostly disappear. We treat the boundaries which cannot be well maintained after the resizing as weak boundaries. Otherwise, they are considered as strong boundaries. The weak adjacent boundary between two superpixels indicates that their local contrast is low. Therefore, we may merge them in SM. The main problem is how to choose a proper scale. It is difficult to accurately select a suitable scale to definitely distinguish between strong and weak boundaries. Therefore, we generate multiple low scales of boundary maps to comprehensively evaluate the strength of boundaries. On the other hand, since large scales may lead to over-resizing (one extreme case is that I can be resized into one pixel), we also need to control the scales to guarantee suitable discrimination of the strong and weak boundary. Specifically, we adopt 0.5 as the interval of k and count it from 1.5. For a current boundary map bm_k, we use Hu moment [19] to measure its dissimilarity D_{bm_k} with BM. If $D_{bm_k} < \mu(D_{bm_k}, \ldots, D_{bm_{1.5}}) + std(D_{bm_k}, \ldots, D_{bm_{1.5}})$ where $\mu(\cdot)$ is the mean of all dissimilarities including bm_k and $std(\cdot)$ is their corresponding standard deviation, we continuously generate boundary map with a lower scale. Otherwise, it indicates that the current boundary map may not be

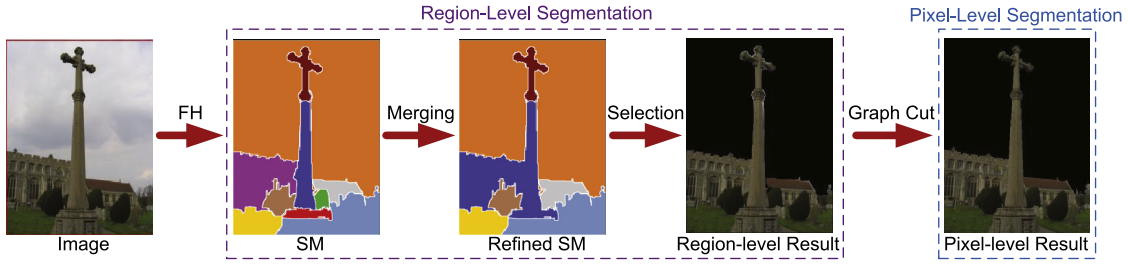

Fig. 2. The whole procedure of our method. From left to right, they are the image with an input rectangle, its superpixel map (SM) generated by FH method, refined superpixel map after merging, region-level segmentation result by selection and pixel-level result by graph cut.

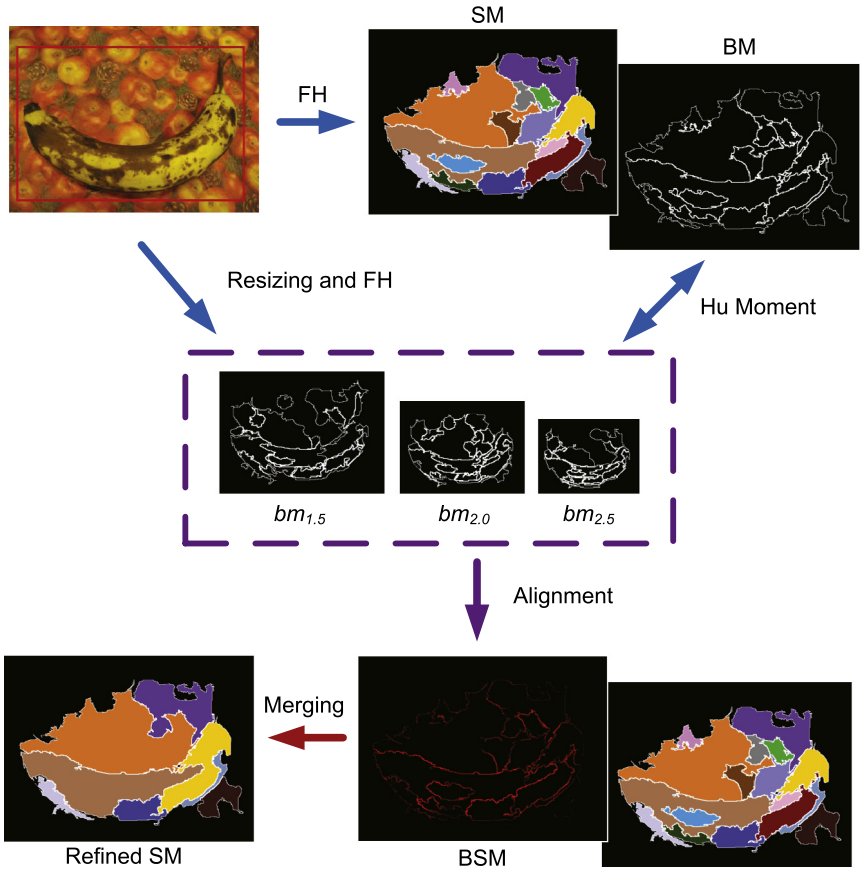

Fig. 3. An example of the merging procedure. The first row shows the source image I with IR and its corresponding SM and BM. Then we can obtain selected multiple scales of boundary maps shown in the second row. BSM and the refined SM after merging are shown in the third row.

reliable, so we discard it and stop the resizing process. One example of the maintained boundary maps is shown in the second row of Fig. 3. Next, the selected boundary maps are aligned by resizing them back to the original scale. For each pixel on the boundary of BM, its strength is the average value of corresponding pixels in those aligned maps. Consequently, we can obtain a boundary strength map BSM of BM. Different from traditional methods [20,21] measuring the similarity between two superpixels using the original scale, our method is more abstract in order to reduce the influence of image details. Our method is similar to [22], but we do not fix the scales. Since each image has its own discrimination of the boundary strength, our method can adaptively select suitable scales according to the dissimilarity with BM. As shown in the third row of Fig. 3, lighter red lines in BSM indicate stronger boundaries. In SM, the merging starts from the first superpixel with the topmost location. For superpixel sp_i and its certain adjacent superpixel sp_j, their boundaries are bsp_i and bsp_j respectively. The new superpixel after merging is nsp_i and its boundary is $bnsp_i$. We judge whether their adjacent boundary ab_{ij} is a weak boundary by the following conditions,

$$mean(ab_{ij}) + \tau \cdot std(ab_{ij}) < mean(bnsp_i) \quad (1)$$

$$std(bnsp_i) < max(std(bsp_i), std(bsp_j)) \quad (2)$$

where $mean(\cdot)$ is mean strength of all pixels on ab_{ij} and τ is to control the weight of $std(\cdot)$. If ab_{ij} is a weak boundary, its strength should be lower than that of its surrounding boundary $bnsp_i$. If the strength distribution of one adjacent boundary is not uniform, it is hard to judge its strength. So, the standard deviation term is introduced to prevent the high contrast part from being merged. In the implementation, τ is set to 0.4. Meanwhile, the merging should generate the boundary with uniform strength, as enforced by the second condition. Otherwise, one superpixel with very strong boundary may mistakenly be merged into another superpixel with weak boundary. We continue the above merging step until there is no adjacent superpixel to be merged with sp_i. Then, another superpixel is chosen to merge its adjacent superpixels.

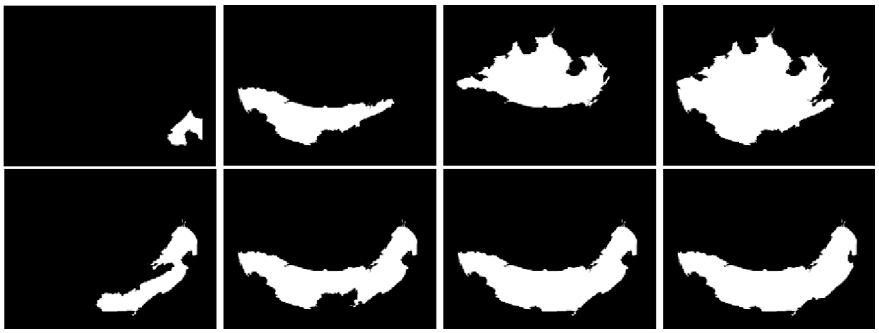

Fig. 4. Some combinations of connected refined superpixels.

The merging procedure continues until no superpixel can be merged with others. The refined SM after merging is illustrated in Fig. 3.

2.1.2. Selection

After the above merging step, we generate M refined superpixels. Then, we can list N possible combinations of connected refined superpixels. Fig. 4 shows some combinations of connected refined superpixels based on Fig. 3. Our goal is to select one of them as the region-level segmentation result. The criteria of this selection are global contrast $G(\cdot)$ and tightness $T(\cdot)$. For one combination C_i, its global contrast is measured as below,

$$G(C_i) = B(H^{C_i}, H^{\overline{C}_i}) + bias(C_i) \tag{3}$$

We construct the $32 \times 32 \times 32$-bin RGB histograms H of C_i and $\overline{C}_i$. $\overline{C}_i$ is the rest part of I excluding possible holes $h(C_i)$ formed by C_i, i.e $\overline{C}_i = I \backslash (C_i \cup h(C_i))$. These three regions are indicated in Fig. 6(a). $B(\cdot)$ calculates the Bhattacharyya coefficient [23,24] of these two histograms as the contrast degree. A common problem is to distinguish between the object whose inner part is quite different from its surrounding (e.g. flower) and the object with holes (e.g. ring). So, if the holes exist, we exclude them from $\overline{C}_i$ and add a $bias(\cdot)$ term to distinguish above two cases,

$$bias(C_i) = \begin{cases} B(H^{IR}, H^{\overline{IR}}) - B(H^{h(C_i)}, H^{\overline{IR}}) & if\ Card(h(C_i)) > 0 \\ 0 & otherwise \end{cases} \tag{4}$$

where $Card(\cdot)$ is the number of these holes and $\overline{IR}$ is the region outside the input rectangle. We assume that the region within the natural hole is similar to the background. Since $\overline{IR}$ is the background and IR includes the background and the object, we treat the difference between these two regions as a reference. If the difference between $\overline{IR}$ and $h(C_i)$ is smaller than that between $\overline{IR}$ and IR, it means that the hole is more similar to the background and we should preserve it by increasing $G(C_i)$. Otherwise, the holes should be a part of the object and $G(C_i)$ is decreased using a negative value of $bias(C_i)$. Fig. 6 shows the comparison of the selection with and without $bias(\cdot)$. We can see that the pistil of the flower is preserved by this term.

As mentioned above, tightness is an useful criterion to eliminate the ambiguity of the object in a certain region. Therefore, we also consider this criterion in our selection. As suggested in [7], we can draw a tight rectangle RT inside IR with a margin. We can use two coordinates to describe RT such as $(p^{lt}_{RT}(x), p^{lt}_{RT}(y))$ and $(p^{rb}_{RT}(x), p^{rb}_{RT}(y))$ where lt and rb represent the left-top and right-bottom corners respectively as shown in Fig. 5(a). Similarly, we draw a bounding rectangle of C_i and describe it by two coordinates $(p^{lt}_{c_i}(x), p^{lt}_{c_i}(y))$ and $(p^{rb}_{c_i}(x), p^{rb}_{c_i}(y))$ as shown in Fig. 5(b). Then, the tightness of C_i is measured as below,

$$T(C_i) = exp(-\frac{\sum_{u\in\{lt,rb\}}\sum_{v\in\{x,y\}} max(p^u_{RT}(v) - p^u_{c_i}(v), 0)}{\alpha L}) \tag{5}$$

where L is the diagonal length of IR and a scaling factor α is set to 0.2. We can see that the tightness degree can be measured relatively according to the size of IR. Compared with the rigid tightness restriction in [7], we relax this criterion by measuring the tightness degree in order to relieve the users' input.

Finally, we combine these two criteria together to select the best segmentation result C^*.

$$C^* = \arg\max_{i\in N} G(C_i)T(C_i) \tag{6}$$

One example of C^* is shown in Fig. 6(b).

2.2. Pixel-level segmentation

The region-level segmentation can roughly separate the object and the background. Although the result may not be accurate in pixel-level, it can be treated as a soft region-level constraint. Therefore, we transfer the region-level segmentation result to the pixel-level segmentation by using C^* and $\overline{C^*}$ to construct the object model and the background model respectively. Then, Graph Cut [8] can generate the pixel-level segmentation result R^*. The energy function of Graph Cut is,

$$E = \sum_p U^p \cdot l_p + \gamma \sum_{\{p,q\}\in\varepsilon} V^{pq} \cdot |l_p - l_q| \tag{7}$$

where p and q represent pixels and ε is a set of pairs of neighboring pixels. l_p is a label of p which is the "object" or "background". U and V is the data term and the smoothness term respectively. γ as a regularization weight is a weight to balance these two terms. The methods of construction of the models and this energy function are the same as those used in Grabcut [6]. The difference is that we estimate γ by a formula given below rather than setting it to be a constant as in [6]

$$\gamma = \lambda(1 + \frac{B(H^{IR}, H^{\overline{IR}})}{B(H^{C^*}, H^{\overline{C^*}})}) \tag{8}$$

where λ is a scaling factor. The denominator in Eq. (8) measures the difference between C^* and $\overline{C^*}$. However, this difference is relative. Therefore, we still use the difference between IR and $\overline{IR}$ as a reference to quantize this relative difference. If the quantized difference is large, it means that the object and the background are well separated. Thus, we can assign the data term a larger weight, or equivalently a smaller γ in Eq.(7). Otherwise, the segmentation result should rely more on the smoothness term. Furthermore, the design of this regularization weight can also be transferred to the traditional GrabCut. Similarly, we can use a current segmentation result to estimate an adaptive γ in the next iteration. Fig. 7 shows different results generated by GrabCut with a fixed regularization weight and adaptive regularization weights which actually assign larger weights to the data term in the iterations. This more proper assignment guarantees the completeness of the result. The comparison illustrates that the design of the adaptive regularization weight can also improve the performance of GrabCut itself.

However, the initial γ may cause that the tightness degree of C^* cannot be maintained by R^* in our method. One example is shown in Fig. 8. We can see that the head is lost in the R^* although C^* covers it.

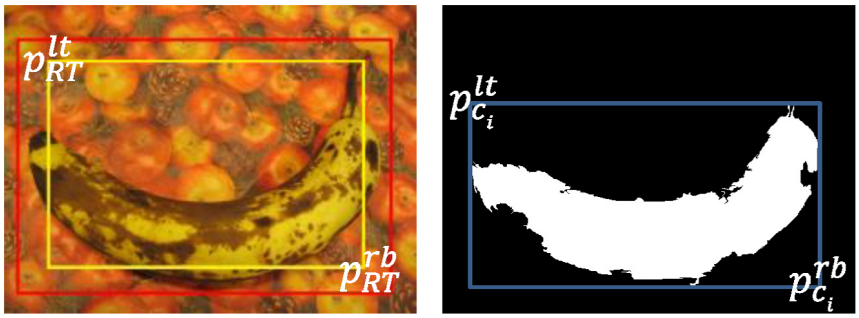

Fig. 5. An example of the tight rectangle and the bounding box of one combination. (a) The yellow tight rectangle based on the red input rectangle, (b) the blue bounding box of one combination.

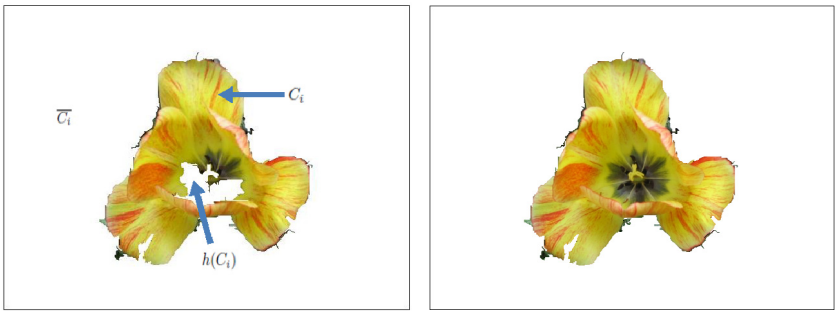

Fig. 6. An example of the effect of the bias term. (a) C_i, $\overline{C_i}$ and $h(C_i)$ are indicated and this combination is selected as C^* without the bias term, (b) C^* selected with the bias term.

Fig. 7. An example of the effect of different regularization weights in GrabCut. (a) Source image with the input, (b) The segmentation result using a fixed regularization weight whose value is 50, (c) The segmentation result using adaptive regularization weights whose average value is 43.2.

Fig. 8. An example of the effect of different λ in the pixel-level segmentation. (a) Source image with the input, (b) The region-level segmentation result, (c) One pixel-level segmentation result using an unsuitable λ and (d) Another pixel-level segmentation result using a suitable λ.

The reason is that the smoothness term mainly determines the labels of the head and the body. So, after generating R^*, we need to evaluate its tightness again. If $T(R^*) < T(C^*)$, the initial λ is reduced by a fixed value θ and then R^* is re-generated. It means that we reduce γ such that R^* approaches C^* in order to hold its tightness. This process is performed iteratively until $T(R^*) \geq T(C^*)$ and $\lambda > 0$. Otherwise, R^* is refined by the tightest one in the iterations. Thus, we maintain the tightness degree of R^* during the global refinement by searching for a suitable λ.

3. Experimental results

3.1. Database and evaluation method

In order to evaluate the performance of the proposed segmentation method, we tested it on three segmentation databases:

1. GrabCut database [6] contains 50 natural images with the input rectangle and the pixel-wise ground truth.
2. ASD [25] database contains 1000 images selected from MSRA database [26] for the salient object segmentation. The pixel-wise ground truth is provided by [27]. In order to mimic the input rectangle given by GrabCut database, the input rectangle in ASD database is annotated by extending 10% of each side of the ground truth's bounding box.
3. PSOD database contains 215 images from SOD [28] database of 300 images. It excludes the image with multiple separated objects in SOD or has been selected in GrabCut database. The pixel-wise ground truth is provided by [28]. The input rectangle is also annotated by extending 10% of each side of the ground truth's bounding box.

For each segmentation result, we evaluated its error rate by calculating the percentage of mislabeled pixels inside the input rectangle [7]. The average error rate of all results in the database indicates the overall performance. In our all experiments, the margin of each side of RT is set to 0.06 of the largest IR dimension as suggested in [7]. The initial λ and θ are empirically set to 30 and 5 respectively in the pixel-level segmentation step.

3.2. Performance on GrabCut database

Since GrabCut database is most commonly used for evaluating the performance of the interactive segmentation method with the rectangle input, we firstly test our method on this database.

3.2.1. Performance of region-level segmentation

As mentioned before, the quality of our region-level segmentation is critical to the final pixel-level segmentation result. So, we evaluate the quality of our region-level segmentation and compare it with MCG [22] which is one state of the art object proposal method. For our region-level segmentation method, the average numbers of superpixels within IR before and after the merging step are 12.3 and 8.1 respectively. Based on the merged superpixels, our region-level segmentation result is generated in the selection step. For MCG, we select one object proposal with the lowest error rate as its segmentation result from its top 20 object proposals. Therefore, we actually use the upper bound of the performance of MCG to do the comparison. The average error rates of our method and MCG are 9.4% and 13.1% respectively. Since our method adopts adaptive scales in the merging step and sufficiently utilizes the input rectangle in the selection step, it can achieve lower error rate. The comparison demonstrates that our method can generate better region-level segmentation results for the subsequent segmentation.

3.2.2. Relative performance of pixel-level segmentation

In order to analyze the effectiveness of every step in our method, different configurations of our method are listed as follows:

- C1: The merging step in the region-level segmentation is skipped and the initial λ is used without adjustment in the pixel-level segmentation step.
- C2: The merging step in the region-level segmentation is performed. But the initial λ is still used without adjustment.
- C3: The merging step is conducted and λ is adjusted according to $T(R^*)$.

Table 1
Performance of different interactive segmentation methods.

Methods		Error rate (%)
Graph Cut		6.7
OneCut		6.7
LP-Pinpoint		5.0
MILCut-Struct		4.2
MILCut-Graph		3.6
Ours	C1	4.6
	C2	4.1
	C3	3.4

We tested our methods with different configurations and compared them with Graph Cut implemented by [7], OneCut [10], LP-Pinpoint [7], MILCut-Struct [16] and MILCut-Graph [16]. The error rates of these methods are shown in Table 1. Firstly, we can see that our basic selection strategy (global contrast + tightness) is able to provide good region-level segmentation result for constructing good initial object and background models. It sufficiently reduce the overlap between the initial object and the background models induced by the input mode so that C1 outperforms most other methods. Secondly, when we consider the local contrast information by adding the merging step, the quality of the region-level segmentation result is further improved. The better initial object and background models make C2 achieve lower error rate, which justify the use of the local contrast for the region-level constraint. Thirdly, compare the performance of C2 with that of C3, it demonstrates the effectiveness of the adjustment of λ in the pixel-level segmentation. In other words, C3 can achieve the best performance by not only considering the region-level constraint but also fine-tuning λ as well as maintaining the tightness degree in the pixel-level segmentation.

3.2.3. Iterative process of pixel-level segmentation

In the traditional GrabCut, the segmentation result can be improved by iteratively refining the object and background models. We follow this procedure by re-constructing the object and background models according to the current segmentation result. The initial γ is re-estimated and the λ is also fine-tuned in each iteration. Our method is compared with LooseCut [14], GrabCut reported in [7] and GrabCut-Pinpoint [7]. The error rates of these methods are shown in Table 2.

The number in the parentheses indicates the number of iterations. We can see that the error rates of all methods are reduced due to the refinement of the object and background models. Furthermore, our method can achieve the best performance with the fewest iterations. Some comparison of the segmentation results are shown in Fig. 9. These results demonstrate the effect of the fusion of the region-level segmentation and the pixel-level segmentation. In the region-level segmentation, on one hand, some object pixels can be preserved although they have high contrast with other neighboring object pixels. On the other hand, some background pixels which are similar to the object can be removed. It guarantees good initial models in the pixel-level segmentation. Furthermore, since the region-level segmentation result is treated as a soft constraint, the errors induced by the region-level segmentation can be corrected in the pixel-level segmentation. Therefore, it is reasonable that our method outperforms LooseCut, GrabCut and GrabCut-Pinpoint which only perform the segmentation with poor initial models in the pixel-level. More segmentation results are shown in Fig. 10. We can see that our method can generate high quality segmentation result although the background is complex or the object and the background are similar.

For the running time, the code is implemented in C++ and we test it on a PC with Intel 2.5 GHz CPU and 16 GB RAM. The average processing time for one image is 27.5 s. The most time is cost on exhaustively searching for the possible combination of superpixels in the region-level segmentation. So, if we use parallel processing to handle the searching in the future, the processing time can be greatly shortened.

Table 2
Performance of different interactive segmentation methods with iterations.

Methods	LooseCut (5)	GrabCut (5)	GrabCut-Pinpoint (5)	Ours (2)
Error rate (%)	7.9	5.9	3.7	3.0

Table 3
Performance of different interactive segmentation methods on ASD and PSOD databases.

Method	Error rate (%)	
	ASD	PSOD
LooseCut(5)	5.8	23.5
GrabCut(5)	4.8	18.8
GrabCut-Pinpoint (5)	4.5	16.7
Ours(2)	4.3	15.3

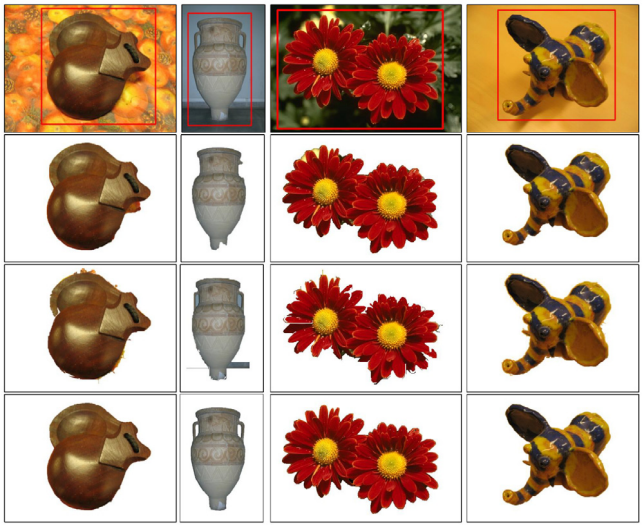

Fig. 9. Comparison of segmentation results. The images from the first row to the fourth are the source images with the input rectangles, the results generated by GrabCut-Pinpoint, our region-level segmentation results and our final results.

3.2.4. The limitation

According to our basic assumption of the tightness, we tested two extreme input rectangles: one uses the bounding box of ground truth as the tightest input rectangle; another is the loosest input rectangle whose corresponding inside tight rectangle is the bounding box of ground truth. The error rates of these two cases are 4.0% and 5.2% respectively. For the first case, the constraint of the tightness is easy to satisfy and the global contrast plays a major role in selecting the region-level segmentation result. Conversely, since the inside tight rectangle is larger in the second case, the tightness mainly determines the region-level segmentation result. So, we can see that a good balance between the global contrast and the tightness is necessary to generate accurate segmentation results.

3.3. Performance on other databases

In order to evaluate the robustness of our method, we also tested it on ASD and PSOD databases. For these two databases, the parameters in our method are set as the same as for the Grabcut database. The error rates of the different methods on these two databases are shown in Table 3. The number in the parentheses indicates the number of iterations. Due to high contrast between the object and the background in images contained in ASD, all methods can achieve low error rates. Conversely, PSOD contains many images with complicated scenes. The contrast between the object and the background is more difficult to measure. Therefore, the performances of all methods on PSOD are much lower than those on ASD. However, our method can also achieve better performance. It demonstrates good robustness of our method. Some comparison of segmentation results on these two databases are shown in Figs. 11 and 12. We can see that our method can generate more accurate segmentation results, especially where most part of the background is covered by the input rectangle.

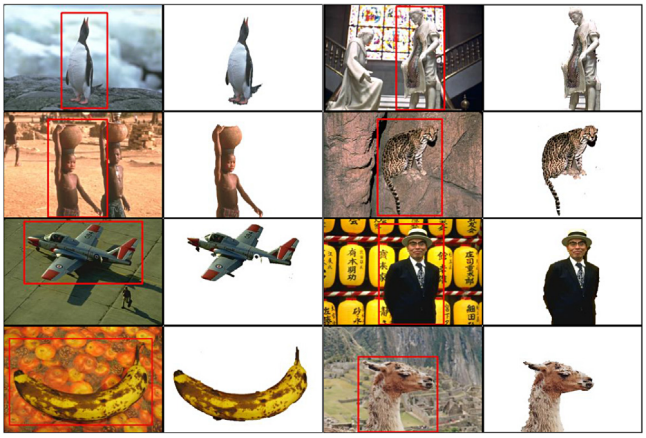

Fig. 10. More of our segmentation results of images with complex background or high similarity between the object and the background.

Fig. 11. Comparison of segmentation results on ASD database. The images from the first row to the fifth are the source images with the input rectangles, the results generated by LooseCut, the results generated by GrabCut, the results generated by GrabCut-Pinpoint, and our results.

4. Conclusion

We treat the interactive object segmentation problem as how to measure the two types of contrast in the two levels. Here, two types

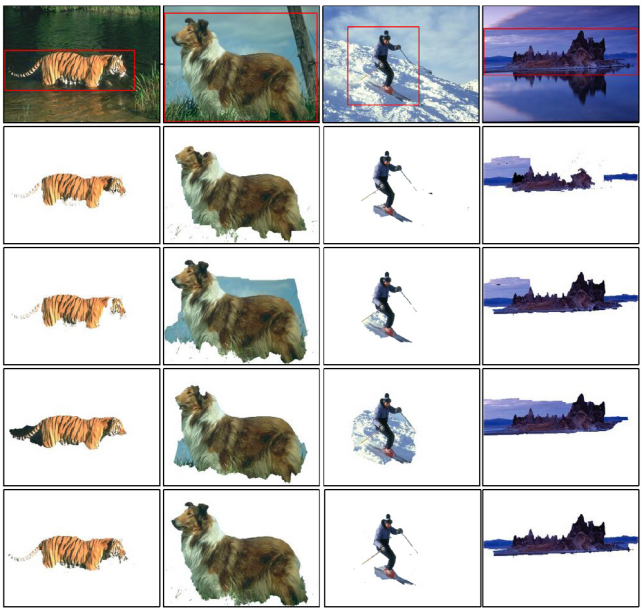

Fig. 12. Comparison of segmentation results on PSOD database. The images from the first row to the fifth are the source images with the input rectangles, the results generated by LooseCut, the results generated by GrabCut, the results generated by GrabCut-Pinpoint, and our results.

of contrast are global contrast and local contrast; two levels are region-level and pixel-level. In the region-level segmentation, the merging step measures the local contrast among the superpixels and the selection step evaluates the global contrast between one candidate combination of refined superpixels and the rest of the region. In the pixel-level segmentation, the data term and the smoothness term in the energy function measure the pixel-level global contrast and local contrast. The region-level segmentation result is used to construct the initial object and background models, and estimate the initial regularization weight γ. Therefore, we can transfer the soft region-level constraint to the pixel-level segmentation. Moreover, the segmentation result is further improved by integrating the tightness degree into these two levels.

Acknowledgments

This work is partially supported by a grant from the Research Grants Council of the Hong Kong SAR, China (Project CUHK 14201115). The authors would like to thank Prof. Lempitsky for his help on performing experiments.

References

[1] Y. Li, J. Sun, C.-K. Tang, H.-Y. Shum, Lazy snapping, ACM Trans. Graph. (ToG) 23 (3) (2004) 303–308.
[2] A. Shamir, S. Avidan, Seam carving for media retargeting, Commun. ACM 52 (1) (2009) 77–85.
[3] C. Guo, L. Zhang, A novel multiresolution spatiotemporal saliency detection model and its applications in image and video compression, IEEE Trans. Image Process. 19 (1) (2010) 185–198.
[4] J. Zhang, S. Sclaroff, Exploiting surroundedness for saliency detection: a Boolean map approach, IEEE Trans. Pattern Anal. Mach. Intell. 38 (5) (2016) 889–902.
[5] W.-C. Tu, S. He, Q. Yang, S.-Y. Chien, Real-time salient object detection with a minimum spanning tree, in: Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition, 2016, pp. 2334–2342.
[6] C. Rother, V. Kolmogorov, A. Blake, Grabcut: Interactive foreground extraction using iterated graph cuts, in: ACM Transactions on Graphics (TOG), vol. 23(3), ACM, 2004, pp. 309–314.
[7] V. Lempitsky, P. Kohli, C. Rother, T. Sharp, Image segmentation with a bounding box prior, in: Proceedings of the IEEE International Conference on Computer Vision, IEEE, 2009, pp. 277–284.
[8] Y.Y. Boykov, M.-P. Jolly, Interactive graph cuts for optimal boundary & region segmentation of objects in ND images, in: Proceedings of the IEEE International Conference on Computer Vision, vol. 1, IEEE, 2001, pp. 105–112.
[9] M. Jian, C. Jung, Interactive image segmentation using adaptive constraint propagation, IEEE Trans. Image Process. 25 (3) (2016) 1301–1311.
[10] M. Tang, L. Gorelick, O. Veksler, Y. Boykov, Grabcut in one cut, in: Proceedings of the IEEE International Conference on Computer Vision, 2013, pp. 1769–1776.
[11] Z. Li, X.-M. Wu, S.-F. Chang, Segmentation using superpixels: A bipartite graph partitioning approach, in: Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition, IEEE, 2012, pp. 789–796.
[12] B. Alexe, T. Deselaers, V. Ferrari, What is an object? in: Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition, IEEE, 2010, pp. 73–80.
[13] T. Wang, Q. Sun, Z. Ji, Q. Chen, P. Fu, Multi-layer graph constraints for interactive image segmentation via game theory, Pattern Recognit. 55 (2016) 28–44.
[14] H. Yu, Y. Zhou, H. Qian, M. Xian, Y. Lin, D. Guo, K. Zheng, K. Abdelfatah, S. Wang, Loosecut: interactive image segmentation with loosely bounded boxes, 2015. ArXiv preprint arXiv:1507.03060.
[15] Y. Wei, F. Wen, W. Zhu, J. Sun, Geodesic saliency using background priors, in: Computer Vision–ECCV 2012, 2012, pp. 29–42.
[16] J. Wu, Y. Zhao, J.-Y. Zhu, S. Luo, Z. Tu, Milcut: A sweeping line multiple instance learning paradigm for interactive image segmentation, in: Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition, 2014, pp. 256–263.
[17] P.F. Felzenszwalb, D.P. Huttenlocher, Efficient graph-based image segmentation, Int. J. Comput. Vis. 59 (2) (2004) 167–181.
[18] B.C. Russell, W.T. Freeman, A.A. Efros, J. Sivic, A. Zisserman, Using multiple segmentations to discover objects and their extent in image collections, in: Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition, vol. 2, IEEE, 2006, pp. 1605–1614.
[19] M.-K. Hu, Visual pattern recognition by moment invariants, IRE Trans. Inform. Theory 8 (2) (1962) 179–187.
[20] K.E. Van de Sande, J.R. Uijlings, T. Gevers, A.W. Smeulders, Segmentation as selective search for object recognition, in: Proceedings of the IEEE International Conference on Computer Vision, IEEE, 2011, pp. 1879–1886.
[21] P. Rantalankila, J. Kannala, E. Rahtu, Generating object segmentation proposals using global and local search, in: Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition, 2014, pp. 2417–2424.
[22] P. Arbeláez, J. Pont-Tuset, J. Barron, F. Marques, J. Malik, Multiscale combinatorial grouping, in: Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition, 2014, pp. 328–335.
[23] A. Bhattachayya, On a measure of divergence between two statistical population defined by their population distributions, Bull. Calcutta Math. Soc. 35 (1943) 99–109.
[24] J. Ning, L. Zhang, D. Zhang, C. Wu, Interactive image segmentation by maximal similarity based region merging, Pattern Recognit. 43 (2) (2010) 445–456.
[25] R. Achanta, S. Hemami, F. Estrada, S. Susstrunk, Frequency-tuned salient region detection, in: Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition, IEEE, 2009, pp. 1597–1604.
[26] T. Liu, Z. Yuan, J. Sun, J. Wang, N. Zheng, X. Tang, H.-Y. Shum, Learning to detect a salient object, IEEE Trans. Pattern Anal. Mach. Intell. 33 (2) (2011) 353–367.
[27] Z. Liu, W. Zou, O. Le Meur, Saliency tree: A novel saliency detection framework, IEEE Trans. Image Process. 23 (5) (2014) 1937–1952.
[28] V. Movahedi, J.H. Elder, Design and perceptual validation of performance measures for salient object segmentation, in: Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition-Workshops, IEEE, 2010, pp. 49–56.