International Journal of Pattern Recognition
and Artificial Intelligence
Vol. 37, No. 3 (2023) 2354005 (28 pages)
© World Scientific Publishing Company
DOI: 10.1142/S0218001423540058

A Novel Interactive Image Segmentation Algorithm Based on Maximization of Submodular Function

Huang Tan* and Qiaoliang Li†
Key Laboratory of Computing and Stochastic Mathematics (Ministry of Education)
School of Mathematics and Statistics, Hunan Normal University
Changsha, Hunan 410081, P. R. China
**tanhuang@hunnu.edu.cn*
†*liqiaoliang@hunnu.edu.cn*

Zili Peng
Hunan Youmei Science and Technology Development Co., Ltd.
No. 568 Queyuan Road, Changsha, Hunan 410000, P. R. China
pengzili0603@gmail.com

Received 7 October 2022
Accepted 12 December 2022
Published 4 March 2023

In this paper, an efficient interactive image segmentation method based on maximization of submodular function under user's scribble constraint is presented. The problem of interactive image segmentation is formulated as the maximum entropy rate under user's constraints. The objective function is submodular, and we solve the constrained submodular function maximization by incorporating a new data structure and some aggregating rules into the greedy algorithm. The main steps of our algorithm are as follows. First, the pixels scribbled by the user are clustered separately as target foreground and background clusters. Second, in the process of greedy algorithm, unscribbled pixels are aggregated to the corresponding target cluster according to the proposed aggregating rules. Finally, the segmentation result is presented by the two target clusters. The experiments and comparisons on three standard benchmarks show that our method has good performance. Our method is straightforward and efficient, and the time complexity of our method is between linear and polynomial. Furthermore, we analyze the influence of different scribbles, and propose some optimal scribble strategies.

Keywords: Image segmentation; user's scribble; greedy algorithm; submodular maximization.

1. Introduction

As the basis of computer vision and an important part of image understanding, image segmentation is the first step of image analysis; it is also one of the most difficult problems in image process. During the last decade, many efficient image

*Corresponding author.

segmentation algorithms have been proposed, and these algorithms can be divided into two categories: traditional image segmentation methods[32,46] and image segmentation methods based on machine learning (ML).[16] Automatic segmentation algorithms are effective solutions for applications, such as multimedia indexing and retrieval, which require quick, coarse and region-based segmentation. Some applications, however, require accurate semantic objects so far as to instances segmentation. So, fully automatic segmentation is often inadequate. Some high-level semantic information is needed to close the semantic gap between homogeneous regions and perceived objects. Interactive segmentation algorithms provide a solution through a series of human interactions to supply the high-level information necessary. Typically, users provide some markers in the image as foreground or background to guide the segmentation process, and the algorithm can update the segmentation further by using newly added marker information. By iteratively providing more interactions, the user can refine the segmentation.[20]

During the last 20 years, many classic interactive image segmentation methods, such as Graph cuts,[6] Normalized cut,[44] GrabCut[40] and Random Walk (RW),[18] have been proposed. And there are also many improved interactive image segmentation methods which are on the basis of the theories of these classic methods.[13,14,17,36,47,49,50] In recent years, many interactive image segmentation methods based on deep learning have been proposed to achieve high performance.[7,26,27,29,33,35,45,51]

Submodularity is a property of "diminishing marginal utility". Submodular function optimization is a fundamental problem in discrete optimization and has attracted more and more attention over the past decade in computer vision field.[24] Shen *et al.*[43] proposed a new trajectory clustering method using submodular optimization for better motion segmentation in videos. In Ref. 42, the multi-object tracking problem has been formulated as a submodular maximization problem subject to related constraints. A growing number of vision applications can be formulated as submodular maximization under constraints, e.g. superpixel segmentation,[30] object detection,[3,31] video summarization,[19,52] image saliency[22,53,54] and image segmentation.[41]

Superpixel segmentation based on submodular function optimization is a process of clustering, and can be solved by greedy algorithm.[15,28,34] It has high efficiency. Liu *et al.*[30] proposed an entropy function, which is a monotonically increasing submodular function, as objective function for superpixel segmentation. They presented a novel construction for the graph associated with the image and showed that this construction induced a matroid. The segmentation result was given by the graph topology that maximized the objective function under the matroid constraint. By exploiting the submodular and monotonic properties of the objective function, they developed an efficient greedy algorithm. Their method has outperformed the state-of-the-art superpixel segmentation algorithms in most of the standard evaluation metrics. This method is efficient, and it works well for images with homogeneous foreground and background. However, it is not applicable to images that have heterogeneous foreground and background. The reason for the failed segmentation is that clusters gather only pixels with high similarity. The processes do not pay attention to the targeted object.

In Ref. 41, the authors proposed a framework for maximizing quadratic submodular energy with a knapsack constraint approximately to solve certain computer vision problems. They tried to solve the interactive image segmentation by the proposed framework. In order to get satisfactory segmentation results, they proposed to introduce preprocessing for the image to be segmented with Gaussian Mixture Model (GMM)[39,55] under user's scribble. But their method depends heavily on the effect of GMM, so their method is not accurate and not suitable for real-time applications.

In this paper, an interactive image segmentation method is proposed. This method uses the high-level information provided by user marking, and gives full play to the efficiency of the superpixel segmentation based on submodular function optimization. We formulate the interactive image segmentation problem as a maximum entropy problem under user's scribble. We solve the maximization problem as an optimization problem of submodular function under user's constraint. In order to tackle the new optimization problem, we incorporate a new data structure and some aggregating rules into the greedy algorithm. In the new data structure, we introduce two target clusters, which are created according to user's scribbles that represent the user-specified foreground and background separately. The proposed aggregating rules guide the clustering process which assembles the pixels into the correct target cluster purposefully. Our clustering process considers not only the similarity of pixels but also the user's scribble. The experiments on Berkeley Segmentation Dataset (BSDS500),[2] MSR[5,8,9] and GrabCut[40] show that our method is efficient and effective.

The innovations of this paper are as follows:

(1) Propose an efficient interactive image segmentation method by adding user's marks to superpixel segmentation method based on submodular function optimization.
(2) A new greedy algorithm that incorporates a new data structure and some aggregating rules, is used to solve the optimization problem of submodular function under user's constraint.

The rest of paper is organized as follows: In Sec. 2, we give an overview of the superpixel segmentation with submodular function maximization. In Sec. 3, we give the details of our proposed method. Extensive experiments are performed in Sec. 4, and we conclude our paper in Sec. 5.

2. Related Works

2.1. *Superpixel segmentation with submodular function maximization*

Superpixel in general is a big element in the image that clusters pixels with similar features. We can take these elements as a new basic unit in many image processing algorithms. Superpixels can reduce the dimension of the image and exclude some abnormal pixels during the image process, so as to improve efficiency and reduce errors. There are many popular algorithms for superpixel segmentation, such as SLIC,[1] Mean Shift,[10] Watershed,[48] NCut,[38] Felzenszwalb–Huttenlocher (FH),[15] Turbo Pixels,[25] etc. Reference 12 made a rough categorization of those algorithms, and gave an evaluation of the state-of-the-art algorithms.

Superpixel segmentation with submodular function maximization is a process of clustering, and it can be solved by greedy algorithm.[15,28,34] It gives a $\frac{1}{2}$-approximation ratio algorithm [see Eq. (4) below] for the optimality of the greedy solution and shows its competitive performances with respect to popular superpixel segmentation algorithms.

A function f on a set V is submodular[24] if

$$f(S \cup e) - f(S) \leq f(T \cup e) - f(T), \quad T \subseteq S \subseteq V,\ e \in V - S. \tag{1}$$

The general form of submodular function maximization is as follows:

$$\begin{aligned} &\max_{S \subseteq V} \quad F(S) \\ &\text{subject to} \quad g(S) \leq K, \end{aligned} \tag{2}$$

where the total cost $g(S)$ is kept to be below some ceiling K.

Generally, submodular function maximization under the constraints is NP-hard.[21] If the submodular function is monotonic and nonnegative, greedy algorithm is the best way to give an approximate solution in polynomial time.[23] In each iteration of greedy algorithm, a maximum gain that satisfies the constraint conditions is added to the subset S,

$$S_i = S_{i-1} \cup \{e | \arg\max_{e \in V - S_{i-1}} f\{e | S_{i-1}\}\}, \tag{3}$$

where the gain $f\{e|S_{i-1}\} \triangleq f(S_{i-1} \cup \{e\}) - f(S_{i-1})$.

The approximate solution S_{approx} is α-approximate if

$$f(S_{\text{approx}}) \geq \alpha f(S_{\text{optimization}}), \tag{4}$$

where $\alpha > 0$ and $f(S_{\text{optimization}})$ is the optimal solution of the problem.

Reference 30 proposed a superpixel segmentation based on the maximization of submodular function. The objective function consists of two parts: the entropy rate of a random walk on a graph and a balancing term, presented as Eq. (5),

$$F(A) = H(A) + \lambda B(A), \tag{5}$$

where $H(A)$ is defined by Eq. (9), and it is the entropy rate of a random walk on a graph that guarantees the clusters to be compact and homogeneous, and $B(A)$ is the balance term, which encourages the clusters to have similar sizes. $\lambda > 0$ is the weight of the balance term. The segmentation result is gotten by maximizing $F(A)$ under some given constraints:

$$\begin{aligned} &\arg\max_{A\subseteq E} \quad F(A) \\ &\text{s.t.} \quad N_A = K, \quad A \in I, \end{aligned} \tag{6}$$

where E is the edge set of the graph corresponding to an image, and I is the set of subsets of E. $A \in I$ denotes the matroid constraint,[4] N_A is the cluster number constraint in A and $K > 0$.

Except an additional heuristic rule, the greedy algorithm for solving Eq. (6) is similar to the standard one.[34] The pseudo-code of the algorithm in Ref. 30 is given in Algorithm 1.

There are five input parameters in Algorithm 1: graph G, nonnegative weight w of edges, the set of vertex labels L in graph G, the number of clusters K and the weight λ of balance term $B(A)$. The weight w gives the similarity between vertices of an edge. $|L|$ denotes the number of labels.

In each iteration, Algorithm 1 selects the edge that yields the largest gain in $F(A)$ subject to the constraint (line 3). By the heuristic rule, the algorithm abandons the selected edge that forms a cycle in cluster A (line 4), and it can achieve an additional speed-up. If A satisfies matroid constraint after adding $\hat{e}_{i,j}$ (line 5), then the cluster that v_i or v_j belongs to is aggregated to another one (line 7), and N_A is reduced by 1 (line 8). After the aggregation, the label of related vertices has been changed. The iteration is processed until all the edges have been appended to A or $N_A = K$. The segmentation result can be gotten through the label of vertices at the end of the algorithm.

Algorithm 1. Pseudo-code of the greedy algorithm for solving the maximization of submodular function $F(A) = H(A) + \lambda B(A)$

Input: $G = (V, E), w : E \to R^+, L, K$ and λ.
Output: V.

```
1: $A \leftarrow \phi, U \leftarrow E, N_A \leftarrow |L|$
2: repeat
3:   $\hat{e}_{i,j} \leftarrow \arg\max_{e_{i,j}\in U} \quad F(A \cup \{e_{i,j}\}) - F(A)$
4:   if $L_i != L_j$ then
5:     if $A \cup \{\hat{e}_{i,j}\} \in I$ then
6:       $A \leftarrow A \cup \{\hat{e}_{i,j}\}$
7:       Do cluster aggregating
8:       $N_A \leftarrow N_A - 1$
9:     end if
10:  end if
11:  $U \leftarrow U - \{\hat{e}_{i,j}\}$
12: until $N_A = K$ or $U = \phi$
```

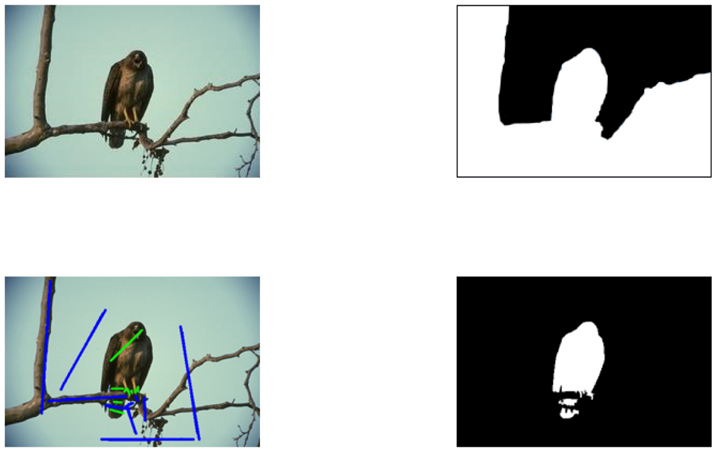


Fig. 1. Results of segmentation by autocluster and our method. The first row shows the result of autocluster and the second row is our result.

2.2. *Motivation*

The method of Ref. 30 is efficient, and the time cost of it is very low. We tried to use the above method to extract an object from an image by setting $K = 2$, but failed for most images. Figure 1 shows one such example.

For most images, the foreground and background are homogeneous inside. The clusters in the graph are aggregated until there are only two clusters left. According to Algorithm 1, two clusters are aggregated when they have high similarity and are adjacent. But if two adjacent clusters with low similarity belong to the same semantic part (see row 1 of Fig. 1, the branch and sky should belong to the background), Algorithm 1 will give a wrong result. Another case is two adjacent clusters having high similarity, but one belongs to foreground while the other belongs to background; here too they will be aggregated wrongly. The reason is that Algorithm 1 aggregates clusters with no semantic information, so it is difficult to meet user's expectations.

Based on the above observations, we add user's marks to the superpixel segmentation method, and modify Algorithm 1 to solve the problem of maximization of submodular function with user's constraints, so that we can overcome the above defects and get the correct segmentation result (see the second row of Fig. 1).

3. Our Approach

3.1. *Object function and constraint*

We construct the objective function by the entropy rate of a random walk on the constructed graph, which is a criterion to obtain compact and homogeneous clusters. The construction leaves the stationary distribution of the random walk unchanged, and the set functions for the transition probabilities $p_{i,j}$ are given in Eq. (7),

$$p_{i,j}(A) = \begin{cases} \dfrac{w_{i,j}}{w_i} & \text{if } i \neq j \quad \text{and} \quad e_{i,j} \in A, \\ 0 & \text{if } i \neq j \quad \text{and} \quad e_{i,j} \notin A, \\ 1 - \dfrac{\Sigma_{j:e_{i,j}\in A} w_{i,j}}{w_i} & \text{if } i = j, \end{cases} \tag{7}$$

where $w_i = \Sigma_{k:e_{i,k}\in E} w_{i,k}$ is the sum of incident weights of the vertex v_i, $w_{i,j} = |I_{v_i} - I_{v_j}|$, where I_{v_i} denotes the intensity of v_i. The stationary distribution of the random walk is given by

$$\mu = (\mu_1, \mu_2, \ldots, \mu_{|V|})^T = \left(\frac{w_1}{w_T}, \frac{w_2}{w_T}, \ldots, \frac{w_{|V|}}{w_T}\right)^T, \tag{8}$$

where $w_T = \sum_{i=1}^{|V|} w_i$ is the normalization constant.

The entropy rate of the random walk on $G = (V, A)$ can be expressed as a set function,

$$H(A) = -\sum_i \mu_i \sum_j p_{i,j}(A) \log(p_{i,j}(A)), \tag{9}$$

where μ is the stationary distribution of the random walk. We rewrite $F(A)$ in Eq. (5) as

$$F(A) = H(A). \tag{10}$$

We do not use the balance term since the foreground and background need not necessarily have the same size. A satisfies the matroid constraint. In Ref. 30, the authors have proven that object function is monotonically increasing and submodular.

Maximization of a submodular function subject to matroid constraint is NP-hard. Greedy algorithm is one of the approximation algorithms for this problem, and it can achieve $\frac{1}{2}$-approximation.[34]

In order to get the correct segmentation result under user's supervision, we add user's scribble U as the constraint of the objective function. The pixels that are scribbled as background are marked as 0, while those scribbled as foreground are marked as 1. Our object function is defined as

$$\begin{aligned} &\arg\max_{A\subseteq E} \; F(A) \\ &\text{s.t.} \quad U, A \in I, \end{aligned} \tag{11}$$

where I is the set of subsets of E, $A \in I$ represents the matroid constraint, U is user's constraint, $U = \{u_i | u_i = \{-1, 0, 1\}, i = 1, 2, \ldots, |V|\}$, and u_i is defined as shown in Eq. (12),

$$u_i = \begin{cases} 0 & \text{if } v_i \text{ is scribbled as background,} \\ 1 & \text{if } v_i \text{ is scribbled as foreground,} \\ -1 & \text{if } v_i \text{ is not scribbled.} \end{cases} \tag{12}$$

3.2. *Algorithm flow*

Our method consists of three main steps: graph construction, initialization of target clusters and maximization of submodular function under user's scribble constraints.

The input includes an image with the target object and user's scribbles. The image is mapped to a graph with our new data structure, and the two target clusters in new data structure are first formed with pixels that are marked by the user. Then the remaining unmarked pixels are aggregated to the right target cluster during the process of maximization of the target submodular function under user's constraints and some aggregating rules. Finally, the segmentation result is presented by the two target clusters.

In the following, we will give the details of the processes.

3.3. *Implementation of cluster processes*

3.3.1. *Graph construction*

Let $G = (V, E)$ be a graph with eight-connected neighborhood. The edge e is expressed as $e = (v_i, v_j)$, where v_i, v_j are the two vertices of e and $w_{i,j}$ is the similarity of v_i and v_j of $e_{i,j}$. The value of $w_{i,j}$ is equal to $p_{i,j}$ in Eq. (7).

A cluster is defined as $c = (l, h, \text{size})$, where l is the label of the cluster and c_l means the cluster of label l. Also, h is the index of a vertex which is called a cluster header in c_l, and there is only one cluster header in c_l. Besides, h_i means the index of cluster header in c_i. Likewise, size denotes the number of vertices of a cluster.

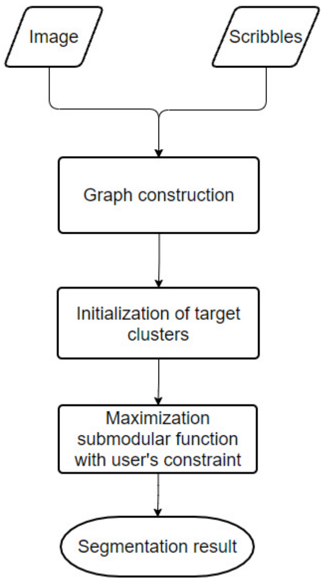


Image
Scribbles
Graph construction
Initialization of target clusters
Maximization submodular function with user's constraint
Segmentation result

Fig. 2. The algorithm flow of our method.

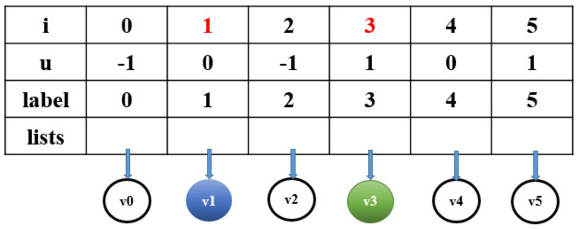
Now we define a new data structure for a vertex v, which consists of four parts as follows:

$$v = (i, u, \text{label}, \text{lists}), \tag{13}$$

where i is the index of the vertex, u indicates whether the vertex belongs to foreground or background that is scribbled by the user and u is defined as shown in Eq. (12). label is the label of the cluster that the vertex belongs to. The lists contains all vertices of a cluster, and the cluster header's lists is not $\emptyset$. When two clusters are aggregated, the lists is used to aggregate all vertices of two clusters. When the size of lists_i is 0, it means that v_i has been aggregated to a cluster c_l, and label_i is changed to l.

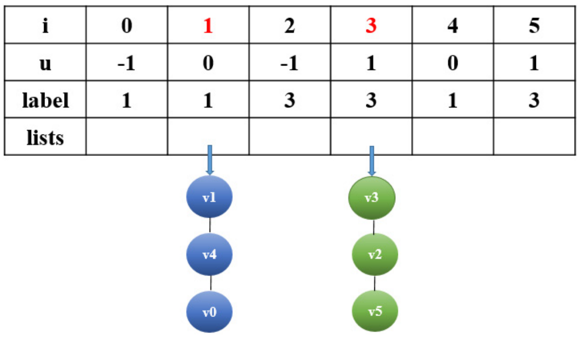
Figure 3 shows an example of the initialization of vertices (we only show six vertices for simplicity). Each of the vertices is treated as an independent cluster c_l, and $l = i$, $h_l = i$ and lists_i only contains v_i. u_i is assigned according to user's scribble.

The example in Fig. 4 shows the segmented result of vertices. All the vertices have been aggregated to one of the target clusters c_3 and c_1, separately. Vertices v_0 and v_4 are included in lists_1, v_2 and v_5 are included in lists_3 and their lists has been changed to $\emptyset$.



i	0	1	2	3	4	5
u	-1	0	-1	1	0	1
label	0	1	2	3	4	5
lists						

Fig. 3. (Color online) Initialization of vertices. Blue disk denotes background vertices, green disk denotes foreground vertices and black circles are vertices that do not belong to both foreground and background temporarily.



i	0	1	2	3	4	5
u	-1	0	-1	1	0	1
label	1	1	3	3	1	3
lists						

Fig. 4. Segmented result of vertices.

3.3.2. *Initialization of target clusters*

In our algorithm, two clusters are left at the end of the process, and we call them the target cluster (tc). Each vertex in the graph must be aggregated to the target cluster during the clustering process, and the two target clusters form the segmentation result finally. The labels of the two target clusters are denoted by f (foreground) and b (background), and the two target clusters are denoted by c_f and c_b. Target clusters can help to get the segmentation result under user's expectation.

In this stage, we first randomly choose two clusters from the initial graph such that $u_{h_f} = 1$ and $u_{h_b} = 0$, separately, and make the two clusters as tc. Then we aggregate other scribbled vertices to tc separately. In Fig. 5, $b = 1$, $f = 3$, $h_1 = 1$ and $h_3 = 3$. v_4 scribbled as background has been aggregated to c_b and v_5 scribbled as foreground has been aggregated to c_f. v_0 and v_2 that are not scribbled will be aggregated to c_f or c_b in the next stage.

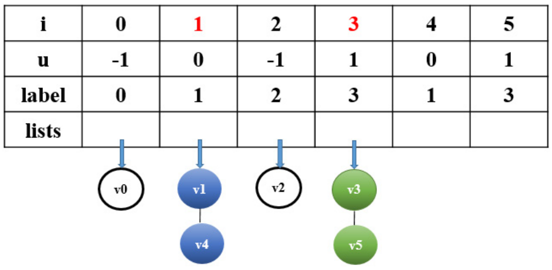
As shown in Fig. 1, image segmentation with the cluster method often failed when the target object is composed of several parts with low similarity. By the initialization of target cluster, we can make tc across parts of target object with low similarity, and some parts with high similarity can also be aggregated to different tc separately. Furthermore, during the process of initializing tc, we delete the related edges with $u_i = u_j$ and $u_i \neq -1$, which can reduce the amount of calculation in the next process.

3.3.3. *Maximization of submodular function with user's constraint*

The maximization of the submodular function with user's constraint is implemented by our improved greedy algorithm with some aggregating rules. The flowchart of our greedy algorithm with user's constraint is shown in Fig. 6.

In the flowchart in Fig. 6, A is the subset of E and N_c is the number of clusters in G. A is the set of remaining edges after initializing the target cluster. In each iteration, we choose an edge $e_{i,j}$ with maximum gain from A and aggregate the two clusters that v_i and v_j belong to. The iteration is terminated until only two target clusters are left in the graph. The following are some rules for cluster aggregation:

(a) If $\text{label}_i == \text{label}_j$, v_i and v_j are not aggregated.



i	0	1	2	3	4	5
u	-1	0	-1	1	0	1
label	0	1	2	3	1	3
lists						

Fig. 5. Initialization of target cluster.

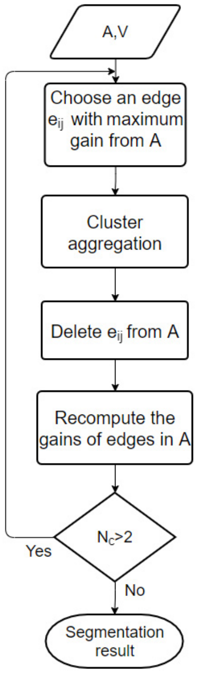

A,V
Choose an edge e_{ij} with maximum gain from A
Cluster aggregation
Delete e_{ij} from A
Recompute the gains of edges in A
N_C>2
Yes
No
Segmentation result

Fig. 6. Flow structure of our greedy algorithm for the maximization of the submodular function with user's constraint.

(b) If $\text{label}_i \neq \text{label}_j$, v_i and v_j are aggregated with the following rules:

(1) If v_i and v_j have been aggregated to tc, then label_i and label_j will be changed to label_{h_b} or label_{h_f}, separately, and they will not be aggregated. Figure 7 is an example of this situation where $v_5 \in c_f$ and $v_8 \in c_b$, so we need not aggregate them.

(2) If v_i has been aggregated to tc and v_j has not been (the converse is valid too), then c_{label_j} will be aggregated to tc. Because the edge $e_{i,j}$ has the highest gain in A, we can assert that v_j is more similar to tc than v_i, so c_{label_j} will be aggregated to tc. An example of this situation is shown in Figs. 7 and 8.

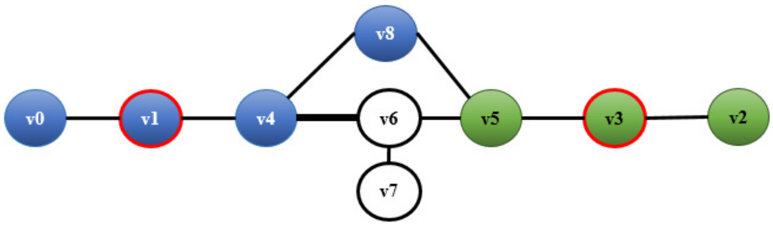
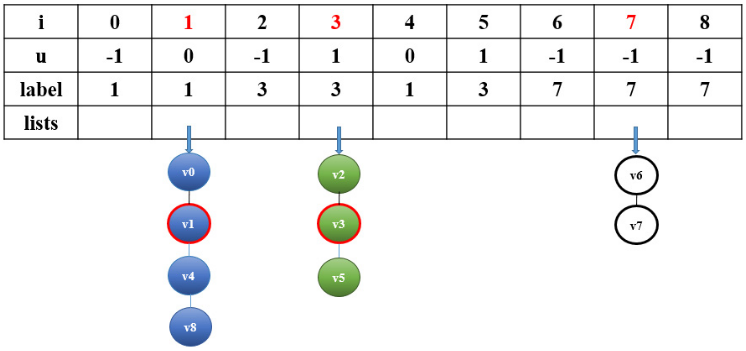


Fig. 7. Edge $e_{8,5}$ connecting the two target clusters. Black circles v_6 and v_7 are the vertices that have been aggregated to a general cluster, while the edges $e_{4,6}$ and $e_{5,6}$ connect the general cluster and two target clusters.

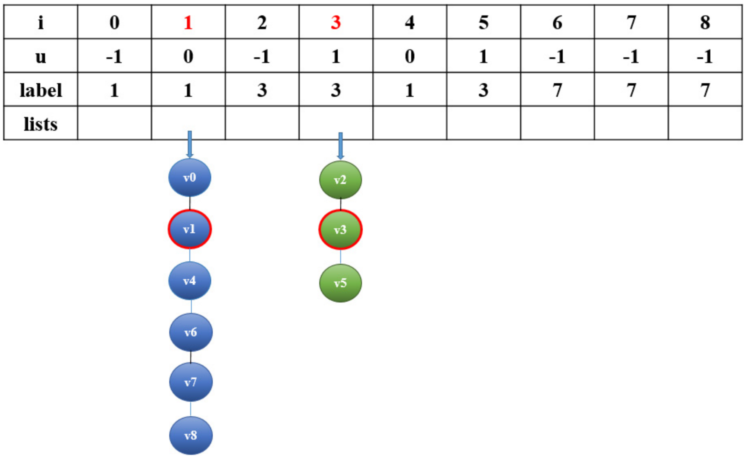


i	0	1	2	3	4	5	6	7	8
u	-1	0	-1	1	0	1	-1	-1	-1
label	1	1	3	3	1	3	7	7	7
lists									

Fig. 8. The information of vertices in the situation of rule (2).

In the example, $f = 3$ and $b = 1$. The current edge with the highest gain is $e_{4,6}$, v_4 belongs to c_b and v_6 belongs a general cluster c_7 ($h_7 = 7$). There is another edge $e_{5,6}$ where v_5 belongs to c_f. We will aggregate c_7 to c_b. As shown in Fig. 8, v_7 contains all the vertices in c_7 and v_1 is the cluster header in c_b. The aggregation process first finds all the vertices from lists_7, changes their label to label_1, then adds lists_7 to lists_1 and finally changes $\text{lists}_7 = \emptyset$. Figure 9 shows the result of aggregation.

(3) If v_i and v_j do not belong to tc, we won't be sure which tc they should be aggregated to, so we aggregate them together temporarily according to their size. The cluster will finally be aggregated to the target cluster in the later iteration process. The cluster aggregation algorithm is shown in Algorithm 2.



i	0	1	2	3	4	5	6	7	8
u	-1	0	-1	1	0	1	-1	-1	-1
label	1	1	3	3	1	3	7	7	7
lists									

Fig. 9. The information of vertices for aggregation result.

Algorithm 2. Pseudo-code of cluster aggregation(v_i, v_j)

```
Input: $v_i, v_j$.
  if label$_i$ $\neq$ label$_j$ then
    if (($v_i \in$ tc) and ($v_j \notin$ tc)) or (($v_i \notin$ tc) and ($v_j \in$ tc)) then
      Aggregate as rule (2)
    else
      if size$_i$ $<$ size$_j$ then
        label$_{ci}$ = label$_j$, $v_{ci} \in$ lists$_i$
        lists$_i$ $\rightarrow$ lists$_j$
        size$_i$ = 0
        lists$_i$ = $\emptyset$
      else
        label$_{cj}$ = label$_i$, $v_{cj} \in$ lists$_j$
        lists$_j$ $\rightarrow$ lists$_i$
        size$_j$ = 0
        lists$_j$ = $\emptyset$
      end if
    end if
  end if
```

4. Experiment

In this section, we compare our method with seven representative methods on three standard datasets (BSDS500,[2] MSRA[5,8,9] and GrabCut[40]). Among these methods, Random Walk[18] and Sub-Markov Random Walk (SRW)[13] are classic methods; Content-based Propagation of User Markings for Interactive Segmentation (CPMSEG)[11] and Interactive Image Segmentation based on Appearance Model and Orientation Energy (AMOE)[37] are two recent traditional methods; and Deep Extreme Cut (DEXTR),[33] Interactive Image Segmentation with Latent Diversity (LDSEG)[26] and Interactive Image Segmentation with First Click Attention (FCA-NET)[27] are deep learning-based methods. We choose a part of these datasets and user's scribbles (1000 pictures from MSRA, 30 pictures from GrabCut and 96 pictures from BSDS500). User's scribbles for images of the three datasets are given by ourselves. Except for DEXTR and FCANET for which we give the extreme points (left-most, right-most, top and bottom pixels) and click points separately, the scribbles are consistent for the other seven comparative methods and suitable for the characteristics of these methods. The seven comparative algorithms use authoritative code and default parameters for experiments. Our experiments are processed in Windows 11 with Intel Core i7 CPU and 16G memory, and do not use GPU for fairness.

Next, we first show the segmented results of our method and the other seven methods. Then we give some scribble strategies for our method. Finally, we give the quantitative results and time complexity analysis of our method.

4.1. *Segmentation results*

Figures 10–12 show some results under the datasets MSRA, GrabCut and BSDS500, respectively. We use green strokes to represent the foreground scribble, and blue represents the background scribbles. Many images from these datasets have weak boundaries between foreground and background, or the region inside the foreground or background is heterogeneous. For most images in these datasets, we have better results than the other seven methods.

Our method is based on the process of clustering, and user's scribbles are used to aggregate vertices under their expectation. Different from Random Walk, in each iteration, we choose two clusters with higher similarity to aggregate. Thus, our method can avoid the aggregation of two clusters which belong to foreground and background separately into one cluster (see the third and fifth rows in Fig. 10). The segmented results of SRW depend on the initial process of GMM. If the result of GMM does not have a high contrast ratio between foreground and background, the

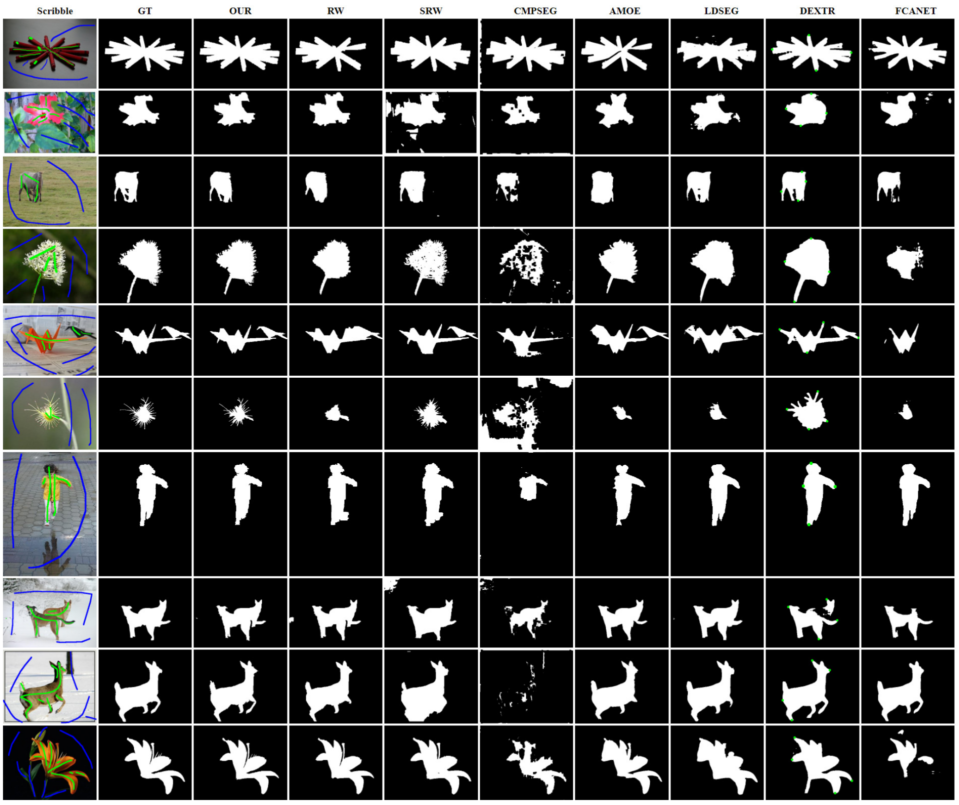


Fig. 10. (Color online) Segmentation results under MSRA. The first column shows the images with scribble; the second column shows ground truths; the third column shows the results of our method; and the next seven columns are the results of RW, SRW, CPMSEG, AMOE, LDSEG, DEXTR and FCANET. Note that we mark the extreme points with green disk for DEXTR.

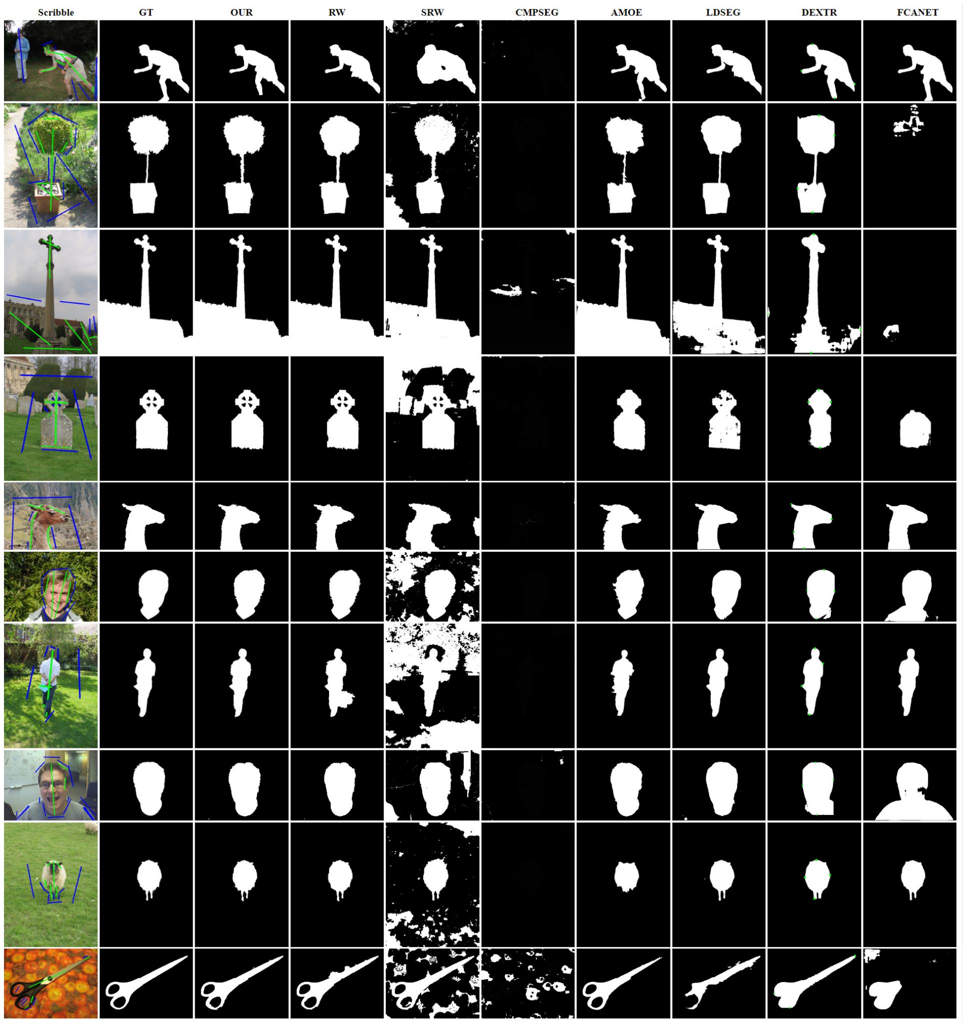


Fig. 11. (Color online) Segmentation results under GrabCut. The first column shows the images with scribble; the second column shows ground truths; the third column shows the results of our method; and the next seven columns are the results of RW, SRW, CPMSEG, AMOE, LDSEG, DEXTR and FCA-NET. Note that we mark the extreme points with green disk for DEXTR.

final segmentation result will not be good (see the fourth and sixth rows in Fig. 10). AMOE has the same problem of SRW; it uses an extra box that includes the object to get the characteristics of the object, thereby improving the contrast of the object from background. The closer the box is to the object, the more accurate the characteristics will be. Otherwise, the quality of the segmentation results will be affected. So, we need to give the box of object carefully. CPMSEG defines the structure of interest with a subset of pixels that are marked by the user, and finds regions with the defined structure. In CPMSEG, one structure is defined to describe the feature of

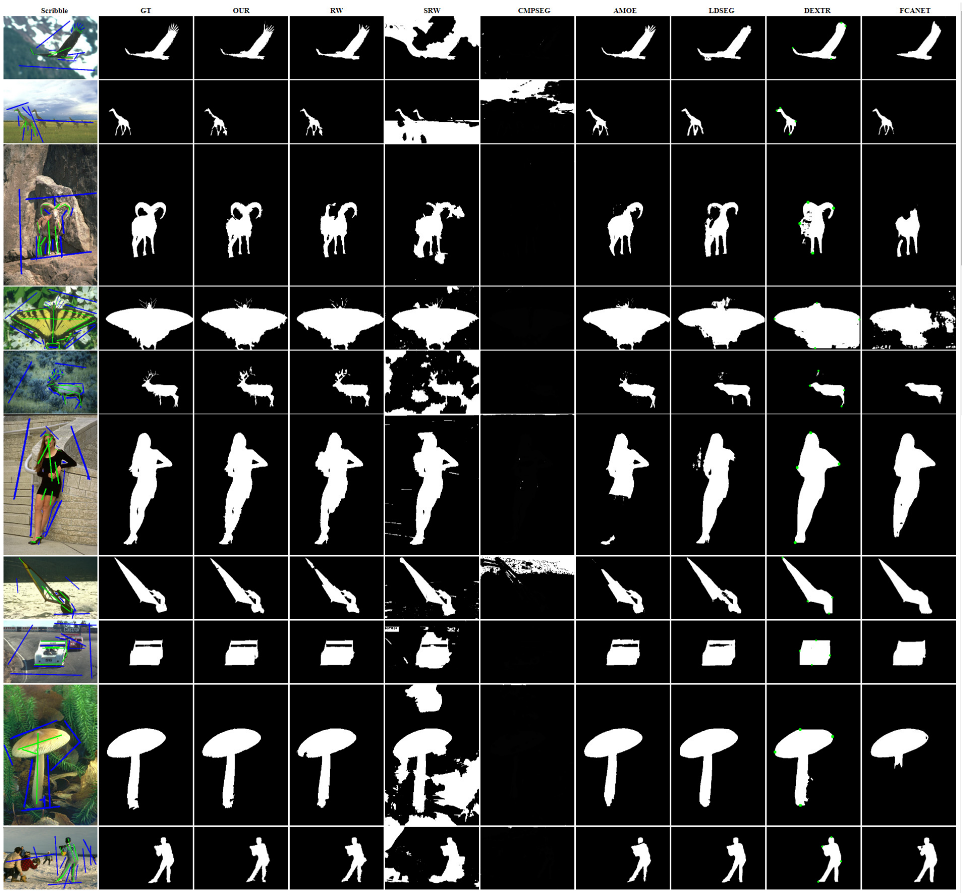


Fig. 12. (Color online) Segmentation results under BSDS500. The first column shows the images with scribble; the second column shows ground truths; the third column shows the results of our method; and the next seven columns are the results of RW, SRW, CPMSEG, AMOE, LDSEG, DEXTR and FCA-NET. Note that we mark the extreme points with green disk for DEXTR.

a single texture or a homogeneous region. If the marked pixels of a label cross a heterogeneous region, the defined structure may not describe such region correctly and cannot get satisfactory segmentation result. This method is suitable for images with homogeneous foreground and background. We can observe that the segmentation results of CPMSEG for MSRA are much better than GrabCut, because the target objects of images in MSRA and BSDS500 are more homogeneous. By pre-training, the three deep learning-based methods DEXTR, LDSEG and FCANET need fewer scribbles and can get the correct segmentation results, but these results are rough, especially for the images with complex boundaries and those with some parts of the background inside the object (see the 4th, 6th and 10th rows in Fig. 10; the 2nd, 4th and 10th rows in Fig. 11; and the first, fourth, fifth and sixth rows in Fig. 12).

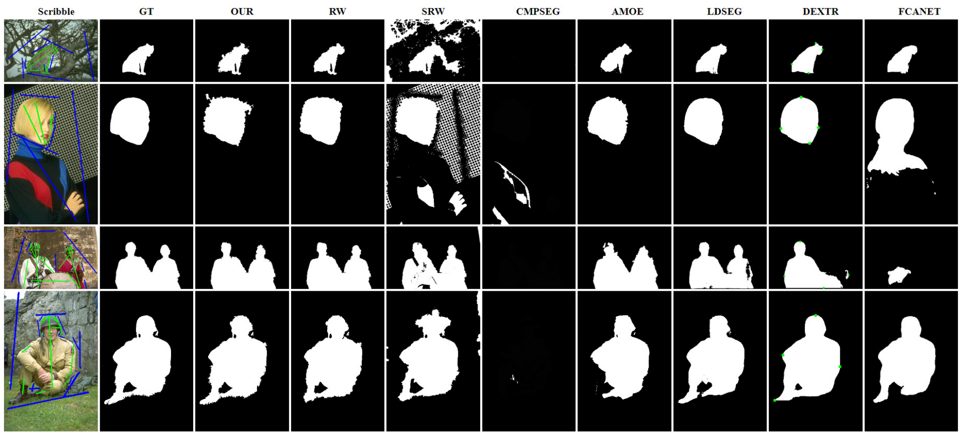


Fig. 13. Some bad results. The third column is ours which has rough boundary (we can see these rough boundaries by zooming in the figure).

In Fig. 13, we show some unsatisfactory results. We can see that these images have complex texture in the background, and the boundary between foreground and background is fuzzy and intermittent. The resulted segmenting boundary is rough. The reason is that the process of aggregating two clusters is based on local information. Figure 14 shows two adjacent vertices (v_0 and v_1) on the boundary in the second image of Fig. 13. By user's expectation, v_0 belongs to background and v_1 belongs to foreground. The two vertices have high similarity and the edge they belong to has the biggest gain. If v_1 has been aggregated to the foreground target cluster and v_0 belongs to a general cluster [Fig. 14(a)], which conforms to the aggregating situation of (2) of rule (b) in Sec. 3.3.3, v_0 will be aggregated to the foreground target cluster. If the two vertices belong to two different general clusters separately, which conforms to the aggregating situation of (2) of rule (b) in Sec. 3.3.3, their clusters will be aggregated together. There is incorrect aggregation in these two

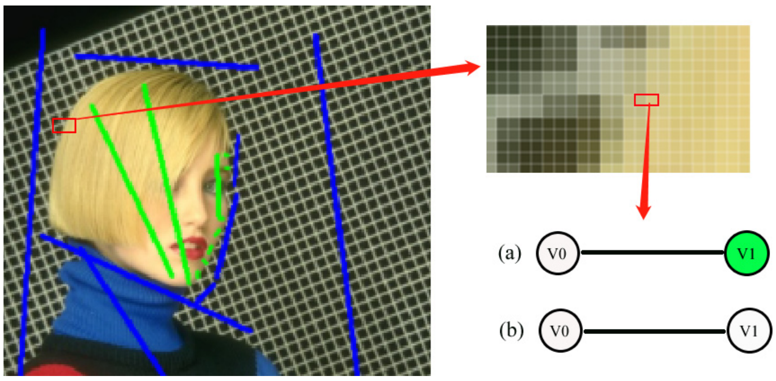


Fig. 14. Wrong aggregation of two vertices.

kinds of situations. We can fix the drawback by adding more scribbles, which is to make v_0 and v_1 conform to the aggregating situation of (2) of rule (b) in Sec. 3.3.3. On the other hand, we find that the resulted boundaries of three deep learning-based methods are smooth, although the segmentation results are not absolutely correct.

4.2. *Scribble strategy*

The user's scribble is very useful in the segmenting process. Many algorithms are sensitive to user's scribble. For example, algorithms like SRW and GrabCut require more information to get a good result, since these algorithms use GMM for pre-processing. So, it is needed to cover as many different color blocks as possible for user's scribbles.

Now, we analyze the scribble strategy of our method. During the clustering process, we should avoid two clusters that belong to the same semantic part (foreground or background) from being separated; meanwhile, we also need to avoid two clusters which belong to different parts from being assembled. As shown in Fig. 15(a), the object is the union of the regions A–E in the red rectangle. But the clustering process of Ref. 30 can only extract region A [see Fig. 15(b)]. In our method, the marked vertices are aggregated to the target clusters first. Then the two target

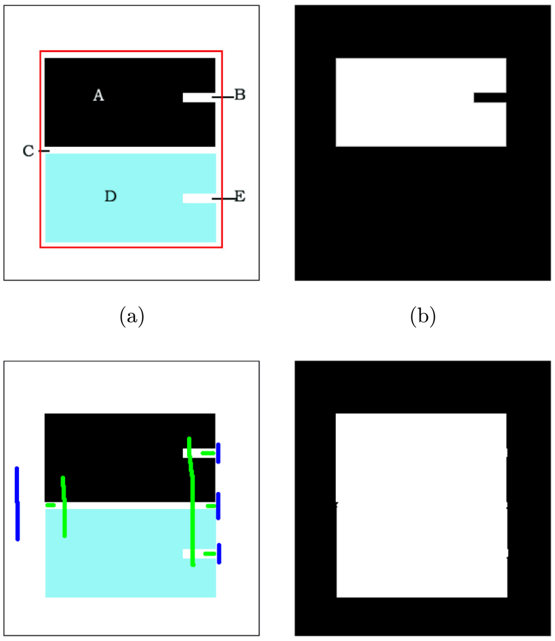


Fig. 15. Image segmentation with scribbling strategy. From left to right are the image to be segmented, wrong result by autocluster, scribbles of foreground and background and the segmentation result with our scribbling strategy.

clusters are treated as the aggregate center and the other clusters will be aggregated into them directly or indirectly. The homogeneous region is aggregated to a cluster correctly. When this region meets other regions, the algorithm will choose the one with the highest similarity to aggregate. But the two regions may not belong to the same targeted region. In order to help the clustering process to find the right cluster to aggregate, we must make a scribble across the two regions, such that the target cluster covers the two regions. Another situation is shown in Fig. 15(a), where the regions B and E are more similar to background, so they will be aggregated to tc_b. To avoid this wrong aggregation, we should make a scribble to keep the two regions apart. Such scribble will prevent tc_b from extending to the regions B and E. In addition, to avoid diffusion of wrong target clusters, we should make the scribbles of background close to foreground. This scribble strategy is appropriate for images with a fuzzy boundary between foreground and background. Figures 15(c) and 15(d) show the effectiveness of our scribble strategy.

Figure 15(c) gives the scribbles which satisfy the above strategy. The difficulty in extracting the object in Fig. 15(a) is due to that the regions B, C and E have the same color as the background. The scribbles are used to distinguish the foreground and background, and they will make the vertices in these regions to be assembled to the right target cluster. Figure 15(d) shows the correct segmentation result with the scribbles of Fig. 15(c).

We apply the scribble strategy to the segmentation process in Sec. 4.1. The object of the seventh image in Fig. 10 is a kid, which can be considered to include four parts: black hair, face, yellow clothes and white trousers. For this kind of object, we provide scribbles across the four parts that can help to aggregate these parts together. There are some objects which have borders similar to the background, such as the fifth image in Fig. 11 and the third and fifth images in Fig. 12. In these situations, it is difficult for the clustering process to extract the objects from background. We provide some scribbles on the bounds of these objects and assist our algorithm to get the correct segmentation results.

4.3. *Quantitative results*

We evaluate our method with five evaluation parameters: DICE, Precision (Pr), True Positive Rate (TPR), Rand Index (RI) and False Positive Rate (FPR). For simplicity, we abbreviate the segmentation result as SEG and the ground truth as GT.

The parameters can be defined as follows:

$$\begin{aligned}\mathrm{DICE} &= \frac{2\mathrm{TP}}{2\mathrm{TP}+\mathrm{FP}+\mathrm{FN}},\\ \mathrm{Pr} &= \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}},\\ \mathrm{TPR} &= \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}},\end{aligned}$$

$$\begin{aligned} \mathrm{FPR} &= \frac{\mathrm{FP}}{\mathrm{FP}+\mathrm{TN}}, \\ \mathrm{RI} &= \frac{\mathrm{TP}+\mathrm{TN}}{\mathrm{TP}+\mathrm{FP}+\mathrm{TN}+\mathrm{FN}}, \end{aligned} \tag{14}$$

where FP, FN, TP and TN represent false positives of the fuzzy union, false negatives, true positives and true negatives, respectively.

DICE is the regional similarity of SEG and GT; Pr is the proportion of true segmented foreground in the foreground in GT; TPR is the proportion of true segmented foreground in GT; and RI is the proportion of true segmented foreground and background in an image. For these four indices, the higher the matching degree is, the higher are the values. FPR is the proportion of false segmented foreground in background and a low value represents a high degree of matching. In order to keep the display consistent with other criteria, we use Specificity $= 1 - \mathrm{FPR}$ instead of FPR. In addition, we use Misclassification Rate (MR) to judge the failed segmentation ($\mathrm{MR} = 1 - \mathrm{RI}$), and we define failed segmentation as $\mathrm{MR} > 0.20$.

Tables 1–3 show the average values of compared criteria under the datasets MSRA, GrabCut and BSDS500, separately. In general, our method has better performance for the five evaluation criteria than the other seven methods. It is worth noting that high values do not always mean a good result. Regarding the TPR value

Table 1. Comparison of the average evaluation criteria for MSRA.

	DICE	Pr	Specificity	TPR	RI
Ours	0.93	**0.92**	**0.98**	0.95	**0.97**
RW	0.92	0.91	0.97	0.94	**0.97**
SRW	0.91	0.86	0.96	**0.98**	0.96
CPMSEG	0.55	0.76	0.97	0.51	0.88
DEXTR	0.88	0.83	0.97	0.95	**0.97**
LDSEG	**0.94**	**0.92**	**0.98**	0.96	**0.97**
FCANET	0.81	0.88	0.94	0.83	0.92
AMOE	0.92	0.92	0.97	0.93	**0.97**

Table 2. Comparison of the average evaluation criteria for GrabCut.

	DICE	Pr	Specificity	TPR	RI
Ours	**0.97**	**0.98**	**0.99**	**0.96**	0.97
RW	0.95	0.97	0.99	0.94	0.97
SRW	0.72	0.62	0.83	0.95	0.96
CPMSEG	0.37	0.72	0.94	0.33	0.84
DEXTR	0.89	0.92	0.98	0.88	0.96
LDSEG	0.94	**0.98**	**0.99**	0.91	**0.98**
FCANET	0.79	0.87	0.91	0.83	0.89
AMOE	0.95	**0.98**	**0.99**	0.92	**0.98**

Table 3. Comparison of the average evaluation criteria for BSDS500.

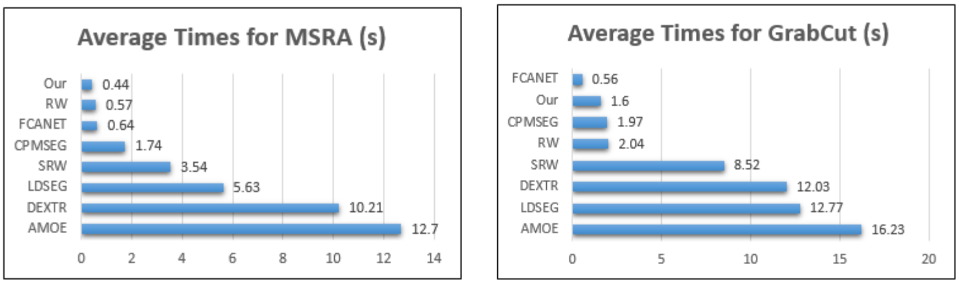
	DICE	Pr	Specificity	TPR	RI
Ours	**0.95**	**0.98**	**0.99**	0.94	**0.99**
RW	0.95	0.95	0.99	0.94	0.98
SRW	0.67	0.59	0.85	**0.95**	0.86
CPMSEG	0.15	0.44	0.96	0.12	0.83
DEXTR	0.91	0.91	0.98	0.92	0.97
LDSEG	0.94	0.96	**0.99**	0.93	0.98
FCANET	0.82	0.94	**0.98**	0.78	0.94
AMOE	0.89	0.97	**0.99**	0.86	0.98

of SRW in Tables 1 and 3, it is the highest of the seven methods, but the segmentation result is not the best. The reason is that some extracted objects not only cover most of the foreground in GT, but also part of the background. According to the definition of TPR, though the segmentation result is not good, the value is high. The average Misclassification Rates for our method, RW, SRW, CPMSEG, DEXTR, LDSEG, FCANET and AMOE are 0.015, 0.02, 0.1, 0.18, 0.03, 0.019, 0.08 and 0.013, and the average failed segmentation rates are 0.007, 0.009, 0.04, 0.18, 0.001, 0.002, 0.1 and 0. From the failed segmentation rates, we can conclude that there are more failed segmented images of our method than the two deep learning-based methods though our method has a lower Misclassification Rate. This shows that the three deep learning-based methods are more stable than ours. The main reason is that our method is based on local information. For the images that have complex texture in background and/or the boundaries between foreground and background are fuzzy and intermittent, we need to provide crucial scribbles to extract the object from background. Otherwise, some small parts of object won't be extracted. The compared deep learning-based methods do the segmentation process with the characteristics of the object (global information). In most cases, the segmentation results of these methods cover the ground truth, even if these results contain some parts of background. The AMOE has the lowest average Misclassification Rate and average failed segmentation rate, but the segmentation results are not good, particularly around the edge of the objects.

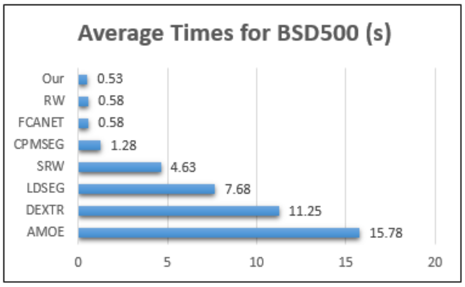
4.4. *Time complexity analysis*

In this subsection, we compare the segmentation times. Figures 16(a)–16(c) show the average times of the segmentation process of the eight methods. The difference of times spent for the three datasets is due to the different sizes of the images (MSRA: 400×300, GrabCut: 640×480 and BSDS500: 481×321).

Overall, the average time of our method is shorter than the other comparative methods. Except for FCANET, the average times of other compared methods change significantly as the size of image gets bigger. The average time of FCANET is more stable than others. SRW and AMOE need preprocessing, and the subsequent process



(a) (b)



(c)

Fig. 16. The average times of the eight segmentation methods for the three datasets: (a) MSRA, (b) GrabCut and (c) BSDS500.

requires iteration. So they need more time to get the segmentation results than the other traditional methods (CMPSEG, RW and our method) in our experiments. Though DEXTR only needs four extreme points, it needs to spend lots of time to do complex computation around these extreme points, and get the box close to the object. So the average time of DEXTR is long. The segmentation of LDSEG includes two steps. First, LDSEG gets multiple optional segmentation masks through the segmentation network, then chooses one of these optional segmentation masks by a selection network. LDSEG also needs long time to get the segmentation result. Next, we will analyze the time complexity of our method.

We construct an undirected graph $G = \langle V, E\rangle$ with eight-connected neighborhood. W denotes the width of the graph, H denotes the height of the graph, $|V| = W * H$ and $|E| = 4WH - 3(W + H) + 2$. For $W \geq 1$ and $H \geq 1$, $|E| < 4|V|$.

As shown in Fig. 2, our algorithm is divided into three steps: graph construction, initialization of target clusters and clustering with maximization of the submodular function with user's constraint.

In the first stage, the properties of vertices are initialized. The running time of this step is $T_1 = 4|V|$.

In the second stage, the scribbled vertices are aggregated to tc, and the related edges are deleted. We denote the set of scribbled vertices by V_s, and denote the edges that connect scribbled vertices by E_s, where $E_s = |V_s| - 2$. Afterward, the number of remaining edges $A = |E| - |E_s|$. The running time of this stage comes to: $T_2 = |V_s| + |E_s| = 2|V_s| - 2$.

The third stage is a circular structure. There are three steps for each iteration. (1) One edge with the biggest gain is chosen from A and the two clusters that relate to the edge are aggregated to tc. (2) The edge is deleted from A and gains of the remaining edges in A are recomputed. (3) The edges in A are resorted by gain. For efficiency, a heap is used to organize A and only the edge with the biggest gain is taken to the top. For the whole circular process, the total time $T_3 = |A| + |V_A| + 2\frac{(1+|A|)*|A|}{2} + |A|\lg|A|$, where V_A is the set of vertices that are related to A, and they are linearly correlated. According to the expression for T_3, the third component of it for recomputing the gain of edges requires the most computational resources, and its time complexity is $O(A^2)$. However, we find in experiments that the circular process need not traverse all the edges in A for most of the images (see Fig. 17), so the time complexity of recomputing the gain of edges is actually about $O(mA), 0 < m \leq 1$. Therefore, the time complexity of this stage is $O(|A|\lg|A|)$.

According to the construction of the graph, we know that A and $|V_A|$ are linearly correlated and $|A| = n|V_A| = n(|V| - |V_s|)$, where $1 \leq n < 4$. The total time spent is

$$\begin{aligned}
T &= T_1 + T_2 + T_3 \\
&= 4|V| + 2|V_s| + |A| + |V_A| + 2\frac{(1+|A|)|A|}{2} + |A|\lg|A| - 2 \\
&= 4|V| + 2|V_s| + |A| + (|V| - |V_s|) + (1+|A|)|A| + |A|\lg|A| - 2 \\
&= 5|V| + |V_s| + 2|A| + |A|^2 + |A|\lg|A| - 2 \\
&= 5|V| + |V_s| + 2n(|V| - |V_s|) + n^2(|V| - |V_s|)^2 \\
&\quad + n(|V| - |V_s|)\lg n(|V| - |V_s|) - 2.
\end{aligned} \tag{15}$$

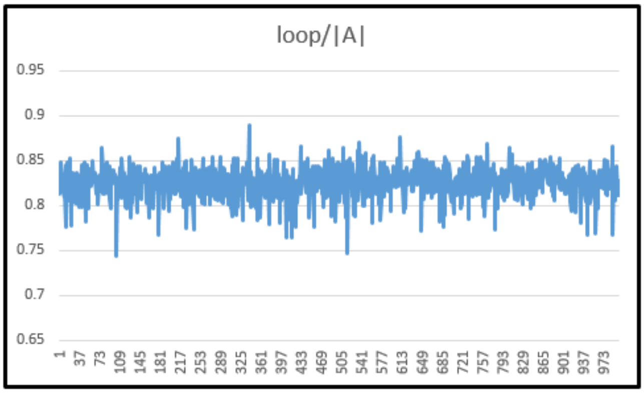


Fig. 17. The proportion of circular number to $|A|$ under the clustering process of 1000 images in MSRA.

Combined with the analysis of the time complexity of the third stage, we can conclude that the total time complexity is $O((|V| - |V_s|)\lg(|V| - |V_s|)), |V| > |V_s|$, which is between linear time complexity and polynomial time complexity. Our method is simple and efficient.

5. Conclusion

In this paper, we proposed a greedy algorithm for the maximization of submodular function with user's constraint to segment the images. In our clustering algorithm, we first make two initial clusters for the foreground and background by user's scribble. Then during the process of the greedy algorithm for maximization of object function with submodularity, we use some aggregate rules for cluster aggregation. The method gets two clusters and extracts the object from the image as per user's expectation. Experimental results of image segmentation demonstrated that our method outperforms some classic, recent and deep learning-based methods. However, our method is sensitive to the fuzzy boundary, therefore, when the object boundary is similar to background, the boundary of the segmentation result may not be smooth. In future work, we plan to explore some submodular functions which can express the relationship between two pixels better. On the other hand, we will try to improve our method by integrating the high-level semantic information and ensure high efficiency. In addition, our method aggregates clusters according to local information and user's scribbles, so it is possible to improve our method to a parallel algorithm for large-resolution images.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (NSFC No. 12071126). We declare that we have no financial and personal relationships with other people or organizations that can inappropriately influence our work, and there is no professional or other personal interest of any nature or kind in any product, service and/or company that could be construed as influencing the position presented in, or the review of, the manuscript entitled.

References

1. R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua and S. Süsstrunk, SLIC superpixels compared to state-of-the-art superpixel methods, *IEEE Trans. Pattern Anal. Mach. Intell.* **34**(11) (2012) 2274–2282.
2. P. Arbelaez, M. Maire, C. Fowlkes and J. Malik, Contour detection and hierarchical image segmentation, *IEEE Trans. Pattern Anal. Mach. Intell.* **33**(5) (2011) 898–916.
3. J. A. Bilmes and W. Bai, Deep submodular functions, preprint, arXiv:1701.08939 [cs.LG] (2017).
4. J. E. Bonin, J. G. Oxley and B. Servatius (eds.), in *Matroid Theory: Proceedings of the 1995 AMS-IMS-SIAM Joint Summer Research Conference*, Graduate Texts in Mathematics, Vol. 197 (American Mathematical Society, Providence, 1996), pp. 234–260.

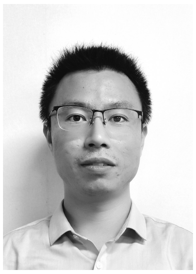
5. A. Borji, M. Cheng, H. Jiang and J. Li, Salient object detection: A survey, preprint, arXiv:1411.5878 [cs.CV] (2014).
6. Y. Boykov and V. Kolmogorov, An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision, *IEEE Trans. Pattern Anal. Mach. Intell.* **26**(9) (2004) 1124–1137.
7. X. Chen, Z. Zhao, F. Yu, Y. Zhang and M. Duan, Conditional diffusion for interactive segmentation, in *Proc. 2021 IEEE Conf. Computer Vision and Pattern Recognition (CVPR)* (2021), pp. 7325–7334.
8. M.-M. Cheng, N. J. Mitra, X. Huang, P. H. S. Torr and S.-M. Hu, Global contrast based salient region detection, *IEEE Trans. Pattern Anal. Mach. Intell.* **37**(3) (2015) 569–582.
9. M. M. Cheng, J. Warrell, W.-Y. Lin, S. Zheng, V. Vineet and N. Crook, Efficient salient region detection with soft image abstraction, in *Proc. 2013 IEEE Int. Conf. Computer Vision (ICCV)* (2013), pp. 1529–1536.
10. D. Comaniciu and P. Meer, Mean shift: a robust approach toward feature space analysis, *IEEE Trans. Pattern Anal. Mach. Intell.* **24**(5) (2002) 603–619.
11. V. A. Dahl, M. J. Emerson, C. H. Trinderup and A. B. Dahl, Content-based propagation of user markings for interactive segmentation of patterned images, in *Proc. 2020 IEEE/CVF Conf. Computer Vision and Pattern Recognition (CVPR)* (2020), pp. 4280–4288.
12. S. David, H. Alexander and L. Bastian, Superpixels: An evaluation of the state-of-the-art, *Comput. Vis. Image Understand.* **166**(1) (2018) 1–27.
13. X. Dong, J. Shen, L. Shao and L. V. Gool, Sub-Markov random walk for image segmentation, *IEEE Trans. Image Process.* **25**(2) (2016) 516–527.
14. X. Dong, J. Shen, L. Shao and M.-H. Yang, Interactive cosegmentation using global and local energy optimization, *IEEE Trans. Image Process.* **24**(11) (2015) 3966–3977.
15. P. F. Felzenszwalb and D. P. Huttenlocher, Efficient graph-based image segmentation, *Int. J. Comput. Vis.* **59**(2) (2004) 167–181.
16. Q. Geng and H. Yan, Image segmentation under the optimization algorithm of krill swarm and machine learning, *Comput. Intell. Neurosci.* **2022** (2022) 8771650.
17. L. Gorelick, O. Veksler, Y. Boykov and C. Nieuwenhuis, Convexity shape prior for binary segmentation, *IEEE Trans. Pattern Anal. Mach. Intell.* **39**(2) (2017) 258–271.
18. L. Grady, Random walks for image segmentation, *IEEE Trans. Pattern Anal. Mach. Intell.* **28**(11) (2006) 1768–1783.
19. M. Gygli, H. Grabner and L. Van Gool, Video summarization by learning submodular mixtures of objectives, in *Proc. 2015 IEEE Conf. Computer Vision and Pattern Recognition (CVPR)* (2015), pp. 3090–3098.
20. C. L. Hiba Ramadan and H. Tairi, A survey of recent interactive image segmentation methods, *Comput. Vis. Media* **6** (2020) 355–384.
21. R. Iyer and J. Bilmes, Algorithms for approximate minimization of the difference between submodular functions, with applications, in *Proc. Twenty-Eighth Conf. Uncertainty in Artificial Intelligence* (2012), pp. 407–417.
22. Z. Jiang and L. S. Davis, Submodular salient region detection, in *Proc. 2013 IEEE Conf. Computer Vision and Pattern Recognition (CVPR)* (2013), pp. 2043–2050.
23. B. Korte and J. Vygen, *Combinatorial Optimization* (Springer, Berlin, 2012).
24. A. Krause and D. Golovin, Submodular function maximization, in *Tractability*, eds. L. Bordeaux, Y. Hamadi, P. Kohli and R. Mateescu (Cambridge University Press, 2001), pp. 71–104.
25. A. Levinshtein, A. Stere, K. Kutulakos, D. Fleet, S. Dickinson and K. Siddiqi, Turbo-Pixels: Fast superpixels using geometric flows, *IEEE Trans. Pattern Anal. Mach. Intell.* **31**(12) (2009) 2290–2297.

26. Z. Li, Q. Chen and V. Koltun, Interactive image segmentation with latent diversity, in *Proc. 2018 IEEE Conf. Computer Vision and Pattern Recognition (CVPR)* (2018), pp. 577–585.
27. Z. Lin, Z. Zhang, L.-Z. Chen, M.-M. Cheng and S.-P. Lu, Interactive image segmentation with first click attention, in *Proc. 2020 IEEE Conf. Computer Vision and Pattern Recognition (CVPR)* (2020), pp. 13339–13348.
28. Y. Liu, E. K. P. Chong, A. Pezeshki and Z. Zhang, Submodular optimization problems and greedy strategies: A survey, *Discrete Event Dyn. Syst., Theory Appl.* **30**(3) (2020) 381–412.
29. Y. Liu, L. Chu, G. Chen, Z. Wu, Z. Chen, B. Lai and Y. Hao, PaddleSeg: A high-efficient development toolkit for image segmentation, preprint, arXiv:2101.06175 [cs.CV] (2021).
30. M.-Y. Liu, O. Tuzel, S. Ramalingam and R. Chellappa, Entropy-rate clustering: Cluster analysis via maximizing a submodular function subject to a matroid constraint, *IEEE Trans. Pattern Anal. Mach. Intell.* **36**(1) (2014) 99–112.
31. D. Liu, K. S. Zhou, D. Bernhardt and D. Comaniciu, Search strategies for multiple landmark detection by submodular maximization, in *Proc. 2010 IEEE Conf. Computer Vision and Pattern Recognition (CVPR)* (2010), pp. 2831–2838.
32. S. Mahajan, Image segmentation and optimization techniques: A short overview, *Medicon Eng. Themes* **2**(2) (2022) 47–49.
33. K.-K. Maninis, S. Caelles, J. Pont-Tuset and L. Van Gool, Deep extreme cut: From extreme points to object segmentation, in *Proc. 2018 IEEE Conf. Computer Vision and Pattern Recognition (CVPR)* (2018), pp. 616–625.
34. G. L. Nemhauser, L. A. Wolsey and M. L. Fisher, An analysis of approximations for maximizing submodular set functions, *Math. Program.* **14**(1) (1978) 265–294.
35. PaddlePaddle Contributors, PaddleSeg: End-to-end image segmentation kit based on PaddlePaddle (2019), https://github.com/PaddlePaddle/PaddleSeg.
36. Z. Peng, S. Qu and Q. Li, Interactive image segmentation using geodesic appearance overlap graph cut, *Signal Process., Image Commun.* **78** (2019) 159–170.
37. S. Qu, H. Tan, Q. Li and Z. Peng, Interactive image segmentation based on the appearance model and orientation energy, *Comput. Vis. Image Understand.* **217** (2022) 103371.
38. X. Ren and J. Malik, Learning a classification model for segmentation, in *Proc. 2003 IEEE Int. Conf. Computer Vision (ICCV)*, Vol. 1 (2003), pp. 10–17.
39. D. Reynolds, Gaussian mixture models, in *Encyclopedia of Biometrics*, eds. S. Z. Li and A. Jain (Springer, US, Boston, 2009), pp. 659–663.
40. C. Rother, V. Kolmogorov and A. Blake, GrabCut: Interactive foreground extraction using iterated graph cuts, *ACM Trans. Graph.* **23** (2004) 309–314.
41. J. Shen, X. Dong, J. Peng, X. Jin, L. Shao and F. Porikli, Submodular function optimization for motion clustering and image segmentation, *IEEE Trans. Neural Netw. Learn. Syst.* **30**(9) (2019) 2637–2649.
42. J. Shen, Z. Liang, J. Liu, H. Sun, L. Shao and D. Tao, Multiobject tracking by submodular optimization, *IEEE Trans. Cybern.* **49**(6) (2019) 1990–2001.
43. J. Shen, J. Peng and L. Shao, Submodular trajectories for better motion segmentation in videos, *IEEE Trans. Image Process.* **27**(6) (2018) 2688–2700.
44. J. Shi and J. Malik, Normalized cuts and image segmentation, *IEEE Trans. Pattern Anal. Mach. Intell.* **22**(8) (2000) 888–905.
45. K. Sofiiuk, I. Petrov, O. Barinova and A. Konushin, f-BRS: Rethinking backpropagating refinement for interactive segmentation, in *Proc. 2020 IEEE Conf. Computer Vision and Pattern Recognition (CVPR)* (2020), pp. 8623–8632.

46. Y. Song and H. Yan, Image segmentation techniques overview, in *Proc. Asia Modelling Symp.*, Vol. 1 (2017), pp. 103–107.
47. M. Tang, L. Gorelick, O. Veksler and Y. Boykov, GrabCut in one cut, in *Proc. 2013 IEEE Int. Conf. Computer Vision (ICCV)* (2013), pp. 1769–1776.
48. L. Vincent and P. Soille, Watersheds in digital spaces: An efficient algorithm based on immersion simulations, *IEEE Trans. Pattern Anal. Mach. Intell.* **13**(6) (1991) 583–598.
49. Q. Wang, J. Gao and Y. Yuan, A joint convolutional neural networks and context transfer for street scenes labeling, *IEEE Trans. Intell. Transp. Syst.* **19**(5) (2018) 1457–1470.
50. T. Wang, J. Yang, Z. Ji and Q. Sun, Probabilistic diffusion for interactive image segmentation, *IEEE Trans. Image Process.* **28**(1) (2019) 330–342.
51. G. Wang, M. A. Zuluaga, W. Li, P. Rosalind, P. A. Patel and A. Michael, DeepIGeoS: A deep interactive geodesic framework for medical image segmentation, *IEEE Trans. Pattern Anal. Mach. Intell.* **41**(7) (2019) 1559–1572.
52. J. Xu, L. Mukherjee, Y. Li, J. Warner, J. M. Rehg and V. Singh, Gaze-enabled egocentric video summarization via constrained submodular maximization, in *Proc. 2015 IEEE Conf. Computer Vision and Pattern Recognition (CVPR)* (2015), pp. 2235–2244.
53. G. Zhong, B. Li, J. Cao and Z. Su, Object level saliency by submodular optimization, in *Proc. 5th Int. Conf. Digital Home* (2014), pp. 105–110.
54. F. Zhu, Z. Jiang and L. Shao, Submodular object recognition, in *Proc. 2014 IEEE Conf. Computer Vision and Pattern Recognition (CVPR)* (2014), pp. 2457–2464.
55. Z. Zivkovic, Improved adaptive Gaussian mixture model for background subtraction, in *Proc. 2004 IEEE Conf. Computer Vision and Pattern Recognition (CVPR)* (2004), pp. 28–31.

Huang Tan received his M.S. degree in Computer Science and Technology from Hunan Normal University, Changsha, Hunan, China, in 2006. Currently, he is pursuing his Ph.D. degree in Computational Mathematics at Hunan Normal University. His research interests mainly include image segmentation and deep learning on graph.

Qiaoliang Li received his M.S. degree from South China Normal University, Guangzhou, China, in 1992. He received his Ph.D. degree from the University of Science and Technology of China, Hefei, China, in 1997. He is currently a Full Professor with the School of Mathematics and Statistics, Hunan Normal University, Changsha, Hunan, China. His research interests include submodular optimization, computer vision and graph theory.



Zili Peng received his Ph.D. degree in Computational Mathematics from Hunan Normal University in China in 2018. He is currently a researcher at Hunan Youmei Science and Technology Development Co., Ltd, where he focuses on computer vision and deep learning in his research.