Journal of Visual Communication and Image Representation 55 (2018) 393–403

Contents lists available at ScienceDirect

Journal of Visual Communication and Image Representation

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

Toward a unified scheme for fast interactive segmentation☆

Ding-Jie Chen, Hwann-Tzong Chen*, Long-Wen Chang

Department of Computer Science, National Tsing Hua University, Hsinchu 300, Taiwan

ARTICLE INFO

Keywords:
Interactive
Image segmentation
Manifold ranking
Machine-assisted

ABSTRACT

This paper presents an efficient and effective interactive segmentation scheme for extracting the region of a foreground object in an image. Our goal is to design an interactive segmentation algorithm that unifies the bounding-box-based, seed-based, and query-based interaction mechanisms for pursuing (*i*) high efficiency in simple interaction mechanism, (*ii*) few interaction rounds, and (*iii*) short response time. The proposed algorithm starts with a user-provided bounding box and obtains candidate background superpixels for inferring the foreground object. Our algorithm tolerates imprecise bounding boxes and provides two kinds of interactions for acquiring correct labels from the user. The user can either input the seed/scribble annotations or label the algorithm-queried regions. Our algorithm selects the most uncertain region as a query, and this query-based interaction mechanism reduces the burden of the user on deciding suitable annotation locations. The average response time per-interaction of our algorithm is merely 0.014 s. Our experiments demonstrate that the algorithm achieves an efficient unified scheme for interactive image segmentation.

1. Introduction

Image segmentation is a fundamental problem in computer vision. Tasks of fully automated segmentation often suffer from ambiguities in the definition of the region of interest. On the other hand, tasks of fully manual segmentation are certainly time-consuming and by no means preferable. Hence, interactive image segmentation, *ie*, segmentation with a human in the loop, is more suitable for achieving satisfactory segmentation accuracy while keeping the time cost acceptable to the user [1–20]. Since the user inputs are required to guide the segmentation process, in order to get a smooth interaction experience, the response time of the segmentation algorithm is expected to be short enough, and the input mechanism is expected to be as simple as possible.

The existing interactive segmentation algorithms can be classified into four categories: (*i*) bounding-box-based [4,6,10,13,14,18], (*ii*) seed/scribble-based [3,5,7–9,11–13,16–19], (*iii*) contour-based [1,2,20], and (*iv*) query-based [15,21]. In general, it is easier for users to indicate the candidate foreground object via a bounding box. However, the segmentation accuracy is usually limited by how precisely the box is drawn. Both the seed-based algorithms and the contour-based algorithms can tackle the situations of complex-shaped objects as long as sufficient user inputs are given—more rounds of interactions are needed in comparison with the bounding-box-based algorithms. As to the query-based algorithm, though the user only needs to provide the true-or-false answers, sometimes the probability of hitting a foreground region is too low for some very small objects, which means that the number of interactions could be out of user's control to get a satisfactory segmentation result.

We propose a new interactive segmentation algorithm combining the advantages from different categories of interactive image segmentation algorithms. Our algorithm starts with a bounding box that roughly covers the foreground object. Then, our algorithm provides an user-assistance mechanism for subsequent improvements in the segmentation accuracy. The mechanism actively suggests the most uncertain region as a query for the user to respond with the correct binary label, in which the user only needs to give a true-or-false answer. An overview of the proposed algorithm is shown in Fig. 1. In this unified scheme, the initial user input, which is the bounding box, is used to provide the underlying distribution of background labels. Then, we estimate an initial confidence level of foreground object according to the potential background labels and the pairwise superpixel relevance using a manifold ranking strategy. Our user-assistance mechanism then acquires some label information from the user to update the segmentation result and the confidence level of the foreground object.

The contributions of the proposed unified scheme for interactive image segmentation are:

1. The initial estimation of the foreground object confidence is robust. We can allow part of the foreground object to straddle the bounding

☆ This paper has been recommended for acceptance by Radu Serban Jasinschi.

* Corresponding author.
E-mail address: htchen@cs.nthu.edu.tw (H.-T. Chen).

https://doi.org/10.1016/j.jvcir.2018.06.011
Received 15 July 2017; Received in revised form 12 April 2018; Accepted 13 June 2018
Available online 19 June 2018

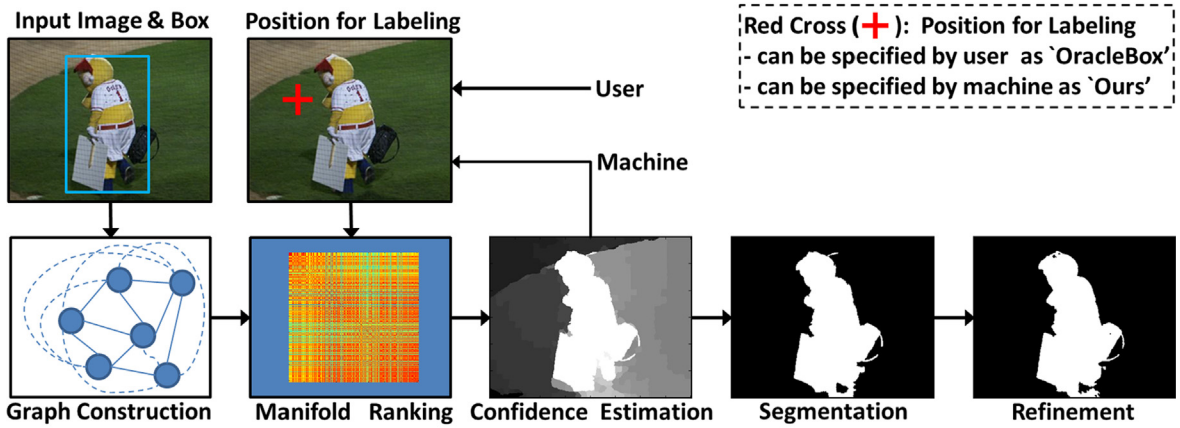


Fig. 1. An overview of the proposed interactive image segmentation algorithm. In 'BnS' approach, the user has to manually explore image positions for labeling. In 'BnQ' approach, the user only needs to provide the binary responses for labeling the automatically generated positions.

box. The user may casually draw a box that is not necessarily "bounding".

2. The response time is very short. The segmentation updating process is simple and efficient, which means that our algorithm is practical to be used in real-time incremental manipulation.
3. The input mechanism is flexible. After providing a bounding box, the user may manually input some labeled seeds to acquire a more accurate segmentation (which is called '*BnS*' as the abbreviation of 'Box and Seeds'), or the user gives true-or-false feedback to the query-region for easily obtaining acceptable segmentation results (which is called '*BnQ*' as the abbreviation of 'Box and Queries').

2. Related work

We classify previous interactive image segmentation algorithms into four categories, depending on the types of user inputs.

2.1. Bounding-box-based

Algorithms of this kind assume that the user specified bounding box encloses the foreground object. Then, figure-ground segmentation is performed according to this assumption. Rother et al. [6] propose the GrabCut algorithm that iteratively estimates the Gaussian mixture models of the foreground and the background areas [4], and then refines the segmentation using graph cuts. Lempitsky et al. [10] model the segmentation as an integer programming problem with the prior from the given bounding box. Tang et al. [13] propose a different energy term for global maximization in graph cut. It can also accept the seed-labels. Wu et al. [14] introduce a multiple-instance-learning algorithm to segment the foreground object inside a given bounding box. Zemene and Pelillo [18] assume that a foreground object is a dominant set [22], and introduce a constrained version of the dominant set algorithm that makes the generated dominant sets contain the user labeled regions.

2.2. Seed/scribble-based

Algorithms of this type solve image segmentation tasks according to user-labeled seeds. The segmentation results usually vary depending on the amount and the positions of the user annotated seeds. Boykov and Jolly [3] represent an image as a graph and treat the user inputs as hard constraints to find an optimal segmentation via graph cuts. Li et al. [5] propose the Lazy Snapping algorithm that contains an object marking step and a boundary editing step. Freedman and Zhang [7] use an object-specific shape prior to a graph cuts framework. Grady [8] models the segmentation result as the probability for each pixel first arrive the labeled pixels. Vicente et al. [9] use explicit connectivity prior to overcoming the shrinking bias effect of graph cuts. Gulshan et al. [11] use a star-convexity shape constraint to segment images under geodesic distance. Anh et al. [12] propose an algorithm to segment a set of multi-view images by interactively cutting a small image subset. Wang et al. [16] combine the region and boundary information to improve the conventional graph cut methods. Feng et al. [17] introduce the cue selection mechanism in a graph cuts framework, which based on the intuition that only one cue is needed at each vertex while optimizing the segmentation energy. Another bounding-box-based algorithm, proposed by Zemene and Pelillo [18], also accepts user scribbles to guide the constrained dominant set. Luo et al. [19] cast the segmentation problem as a multi-classification problem, and then learn a discriminative projection matrix through Fisher linear discriminant analysis for segmentation.

2.3. Contour-based

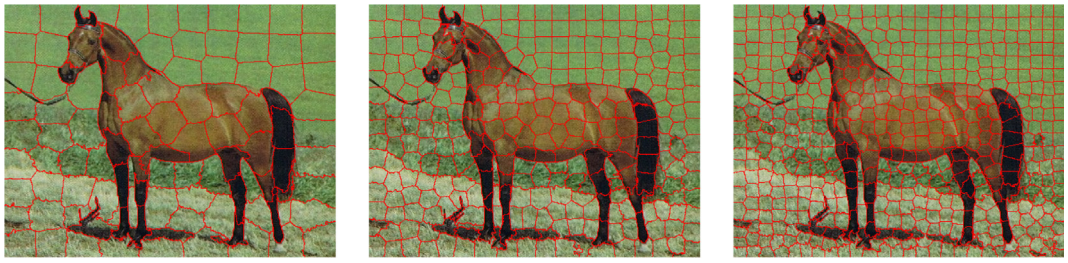
Algorithms in this category require the user to sketch the boundary of the foreground object roughly. Kass et al. [1] present a contour deformable algorithm to warp the user-specified contour for segmentation. Mortensen and Barrett [2] propose an intelligent scissors algorithm, which requires the user to put some seeds around the foreground object. Then the object contour is calculated via the shortest path for separating the figure and ground. Chan and Vese [20] propose an active-contour approach based on Mumford-Shah functional and level sets for segmenting objects whose boundaries have no need to be defined by the gradient.

2.4. Query-based

Algorithms in this category actively ask users the correct labels of uncertain regions for updating the segmentation. Rupprecht et al. [15] introduce an active region-querying image segmentation algorithm. The queried region is calculated using the geodesic distance transform and the Markov Chain Monte Carlo sampling. Chen et al. [21] propose an interactive segmentation algorithm which based on a transductive inference process. A query-pixel is selected by an entropy measurement on feature similarity and uncertainty [23].

2.5. Ours

This work illustrates a unified scheme for fast interactive image segmentation by integrating bounding-box-based, seed/scribble-based, and query-based mechanisms. Most of the existing methods [4,6,10,3,5,7–9,11–13,16–19] initialize a segmentation task with a



(a) $|\mathcal{V}| = 100$ (b) $|\mathcal{V}| = 300$ (c) $|\mathcal{V}| = 500$

Fig. 2. Examples of generating different numbers of superpixels using the SLIC algorithm [26]. Red boundaries separate each non-overlap region as one superpixel-level vertex. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

bounding box or seeds/scribbles, and then refine the segmentation using seed/scribble based annotations. The users of those methods hence take the responsibility of providing qualified annotations to obtain satisfying segmentation results. Our work provides an alternative for segmentation refinement in two aspects: For manually labeled seed/scribble as the most of the existing methods, we propose the '*BnS*' approach. For query-based interaction to reduce the user's burden on deciding suitable annotation locations, we propose the '*BnQ*' approach. Both approaches tolerate imprecise bounding-box inputs and improve the segmentation quality. With the introduced foreground-object-confidence update rule, our approaches efficient and effective refine the segmentation with a human in the loop. The experimental results show the advantages of both approaches to the improvement of segmentation accuracy and response time.

3. The proposed algorithm

Given an image $\mathscr{I}$ represented as a graph $\mathscr{G}$, a segmentation algorithm aims to partition the vertices in $\mathscr{G}$ into disjoint foreground set $\mathscr{F}$ and background set $\mathscr{B}$. We illustrate an algorithm for segmenting an image with respect to its *foreground object confidence*, which is derived from the *relevance* among vertices discussed in a graph-based ranking problem. Briefly, our algorithm starts with a bounding box for estimating the *initial* foreground object confidence. Then, our user-assistance mechanism iteratively queries the user an image region to acquire the user response for updating the foreground object confidence. Finally, the segmentation derived from the foreground object confidence obtaining in the last round is refined with the guided filter. An overview of the proposed algorithm is shown in Fig. 1 and Algorithm 1.

While labeling a graph, which represents an image to segment, we name a vertex that has acquired its label via querying the user as the 'labeled' vertex. The other vertices that have not acquired labels are named as 'not-yet-labeled' vertices. In the graph-based ranking problem [23–25], a ranker is used to calculate the relevances between the labeled vertices and the not-yet-labeled vertices for inferring the labels of those not-yet-labeled vertices. Two factors may affect the ranking accuracy of the ranker. The first factor is the function of the ranker for measuring the relevances between vertices. We use the manifold ranking algorithm proposed by Zhou et al. [24], which shows good performance in the saliency detection problem [25]. In Section 3.2, we describe the graph-based ranking problem and the first factor of an optimal solution proposed by Zhou *et al.* The second factor is how to choose the set of labeled vertices. Exhaustive search is infeasible since there are $2^{|\mathscr{V}|}$ possible segmentations of $\mathscr{V}$. We use the uncertainty of label confidence [15,21] to query the most uncertain vertex per round for acquiring its label from the user. We detail the second factor and the proposed effective confidence update rule in Section 3.4 and 3.6.

3.1. Graph construction

Given an image $\mathscr{I}$, we can partition it into $\mathscr{V} = \{v_1, \cdots, v_{|\mathscr{V}|}\}$, which is a set of non-overlap regions. Each region contains numerous pixels to form a superpixel. Each region can be seen as a vertex. Fig. 2 shows examples of changing the number of superpixels. We then construct a weighted 2-regular graph $\mathscr{G} = (\mathscr{V}, \mathscr{E}, \omega)$ as [25] for an input image. In $\mathscr{G}$, the vertex set $\mathscr{V}$ is the set of superpixels. The edge set $\mathscr{E}$ contains all links between every two adjacent superpixels. Two superpixels are adjacent if (*i*) they are neighboring to each other in the image, or (*ii*) they share the same neighboring superpixel, or (*iii*) they both locate on the four sides of the image. Two superpixels are neighboring if they respectively have at least one pixel to form a 4-connected relation. We define the weighting [1] function ω: $\mathscr{E} \to \mathbb{R}_0^+$ as [25] in the following form

$$\omega_{ij} = e^{-\theta_1 \|m_i - m_j\|_2}, \tag{1}$$

where m_i and m_j denote the CIE-Lab mean color features of the adjacent superpixels v_i and v_j.

3.2. Graph-based manifold ranking

To deal with the ranking problem, the manifold ranking algorithm [24] exploits the intrinsic manifold structure of vertices to estimate their relevances with respect to the labeled vertices for ranking the vertices. According to the estimated relevances, our algorithm can infer the label of those not-yet-labeled vertices reasonably.

In the graph-based ranking problem [23–25], some labeled vertices are given for ranking the rest not-yet-labeled vertices according to the relevances between them. Let $\mathbf{f}$: $\mathscr{V} \to \mathbb{R}^{|\mathscr{V}|}$ denote a ranking function (which we call 'ranker'), which is used to assign a relevance value f_i to each vertex v_i. The ranker $\mathbf{f}$ is in the form of vector, *i.e.*, $\mathbf{f} = [f_1, f_2, \cdots, f_{|\mathscr{V}|}]^T$. A vertex v_i of a higher relevance value f_i means the higher confidence to assign it the same label as the labeled vertices. Let $\mathbf{y} = [y_1, y_2, ..., y_{|\mathscr{V}|}]^T$ denote a binary indicator vector, in which $y_i = 1$ if v_i is a labeled vertex, and $y_i = 0$ otherwise. Let $\mathbf{D_W} = [d_{ij}]_{|\mathscr{V}|\times|\mathscr{V}|}$ denote the diagonal matrix with each diagonal entry representing the row sum of the matrix $\mathbf{W} = [\omega_{ij}]_{|\mathscr{V}|\times|\mathscr{V}|}$. The optimal ranker with respect to the labeled vertices is computed by solving the optimization function as follows:

$$\mathbf{f}^* = \underset{f}{\operatorname{argmin}} \frac{1}{2}\left(\sum_{i,j=1}^{|\mathscr{V}|} w_{i,j} \left\| \frac{f_i}{\sqrt{d_{ii}}} - \frac{f_j}{\sqrt{d_{jj}}} \right\|^2 + \mu \sum_{i=1}^{|\mathscr{V}|} \|f_i - y_i\|\right), \tag{2}$$

where $\mu > 0$ is a parameter to weight the first smoothness term and the second fitting term. A good ranker aims to assign the similar relevance values for neighboring vertices, which is described by the first smoothness term, and not to differ too much concerning the labeled vertices, which is characterized by the second fitting term.

For estimating the relevances between the labeled vertices and the

[1] For calculating the edge weight between superpixels v_i and v_j, the weight $e^{-\alpha\|m_i - m_j\|_2^2}$ is used in [24], and the weight $e^{-\beta\|m_i - m_j\|_2}$ is used in [25], where α and β are parameters.

not-yet-labeled vertices on an image that is represented as a graph $\mathscr{G} = (\mathscr{V}, \mathscr{E})$, Zhou et al. [24,23] define the optimal solution as follows:

$$\mathbf{f}^* = \mathbf{r} = (\mathbf{D_W} - \theta_2 \mathbf{W})^{-1}\mathbf{y}, \tag{3}$$

where $\theta_2 = 1/(1+\mu)$. The vector $\mathbf{r} = [r_1, r_2, ..., r_{|\mathscr{V}|}]^T$ is calculated by Eq. (3), namely, the relevance vector we used to infer the label of the not-yet-labeled vertices. A vertex v_i with a higher relevance r_i means higher confidence to assign it the same label as the labeled vertices.

3.3. Initialization of the foreground object confidence

As aforementioned, the bounding box only provides a rough clue about the foreground object. Nevertheless, it is reasonable to assume that the background superpixels outside the box outnumber the foreground superpixels outside the box. In practice, we treat the superpixels outside the bounding box as the initial background vertices in manifold ranking, and we can thus estimate the initial foreground object confidence of the superpixels inside the box according to the *irrelevances* with respect to these initial background superpixels.

Given a bounding box b in an image that is represented as a superpixel set $\mathscr{V}$, we can partition the set $\mathscr{V}$ into $\mathscr{V}_b$ and its complement $\overline{\mathscr{V}_b}$, where $\mathscr{V}_b$ denotes the set of superpixels covered by the area of the bounding box b. We define a $|\mathscr{V}|$-by-1 binary indicator vector $\mathbf{y}$, in which the elements corresponding to $\overline{\mathscr{V}_b}$ are set as one and the remaining elements are set as zero. Then, according to Eq. (3), the vector $\mathbf{r}$ represents the relevances of all superpixels to the potential background superpixels indicated by $\mathbf{y}$. All elements of the vector $\mathbf{r}$ are non-negative, and we divide each element with the maximum $\max(\mathbf{r})$ for normalizing each element of the vector $\mathbf{r}$ into the range $[0, 1]$. We then define the foreground object confidence $\mathbf{c}$ as the irrelevance by

$$\mathbf{c} = \mathbf{1} - \mathbf{r}, \tag{4}$$

where $\mathbf{1}$ is an $|\mathscr{V}|$-by-1 all-one vector. A vertex v_i with a higher foreground object confidence c_i means higher confidence to assign it the label of the foreground object.

Given a foreground object confidence $\mathbf{c}$, the corresponding segmentation can be obtained via thresholding. Precisely, we take the median value of the range of $\mathbf{c}$, *i.e.*, 0.5, to define the foreground set $\mathscr{F}$ and the background set $\mathscr{B}$. Hence, $\mathscr{F} = \{v_i | c_i \geqslant 0.5\}$ and $\mathscr{B} = \{v_i | c_i < 0.5\}$.

3.4. Updating the foreground object confidence

Once we obtain the initial foreground object confidence, the subsequent task aims to update the confidence with respect to the reliable user feedback. The proposed confidence updating rule is introduced as follows.

We first construct the relevance matrix $\mathbf{R}$ with size $|\mathscr{V}| \times |\mathscr{V}|$ by

$$\mathbf{R} = (\mathbf{D_W} - \theta_2 \mathbf{W})^{-1}. \tag{5}$$

The entry $\mathbf{R}[i, j]$ in the relevance matrix $\mathbf{R}$ describes the relevance between any two superpixels v_i and v_j. We further normalize each row of the relevance matrix to get $\widetilde{\mathbf{R}} = (\mathbf{D_R^{-1}})\mathbf{R}$, where $\mathbf{D_R}$ is a diagonal matrix with each diagonal entry equal to the row sum of $\mathbf{R}$. Hence, the i-th row in $\widetilde{\mathbf{R}}$ represents the row-normalized relevance of the superpixel v_i to each superpixel, and $\widetilde{\mathbf{R}}[i, j]$ denotes the entry at the i-th row and j-th column of $\widetilde{\mathbf{R}}$. $\widetilde{\mathbf{R}}[i, j]$ represents the row-normalized relevance between any two superpixels v_i and v_j, and $\widetilde{\mathbf{R}}[i, i]$ represents the *self-relevance* of the superpixel v_i. Given the graph $\mathscr{G}$, the i-th row of $\widetilde{\mathbf{R}}$ is similar to the transition probability of random walk on $\mathscr{G}$ that starts at the node v_i, and the self-relevance, *i.e.*, $\widetilde{\mathbf{R}}[i, i]$, is always the highest transition probability of the i-th row.

We then denote the foreground object confidence c_i of the superpixel v_i at the t-th round of interaction as c_i^t. The proposed confidence update rule is defined by

$$c_i^{t+1} = \begin{cases} c_i^t + \Delta(i, \Omega(i;t+1)), & \text{if } \ell(\Omega(i;t+1)) \in \mathscr{F}, \\ c_i^t - \Delta(i, \Omega(i;t+1)), & \text{otherwise,} \end{cases} \tag{6}$$

where the function $\Delta(\cdot,\cdot)$ computes the displacement of confidence value, the function $\ell(\cdot)$ retrieves the user-defined label. Note that an updated value that exceeds the range $[0, 1]$ is truncated directly. The function $\Omega(i;t+1)$ gives the index of *the most relevant superpixel among all user-labeled superpixels during these* $t+1$ *rounds* to superpixel v_i. The function $\Delta(\cdot,\cdot)$ is further defined as

$$\Delta(i, j) = \frac{\widetilde{\mathbf{R}}[i, j]}{\widetilde{\mathbf{R}}[i, i]}. \tag{7}$$

The superpixel v_j is the most relevant superpixel to v_i if $\widetilde{\mathbf{R}}[i, j] > \widetilde{\mathbf{R}}[i, k]$ for all v_k. Since $\widetilde{\mathbf{R}}[i, i] \geqslant \widetilde{\mathbf{R}}[i, k]$ for all v_i and v_k, and the summation of each row in $\widetilde{\mathbf{R}}$ is one, the function $\Delta(\cdot,\cdot)$ is in the range from 0 to 1.

For superpixel v_i at the $(t+1)$-th round of interaction, once the most relevant labeled superpixel $v_{\Omega(i;t+1)}$ is defined, Eq. (7) assigns the adjusted confidence value[2] in proportion to the relevance between the superpixel pair v_i and $v_{\Omega(i;t+1)}$ with respect to the self-relevance of the superpixel v_i. Intuitively, Eq. (6) ensures that the label of superpixel v_i is only affected by the most relevant labeled-superpixel $v_{\Omega(i;t+1)}$.

3.5. Refinement

Superpixel-level segmentation may not align well with the exact boundary of an object. After obtaining the preliminary segmentation result, we use the guided filter [27] to refine a segmentation. We only change the pixels near the segmentation boundary (within a radius of 8 pixels). Fig. 3 shows some refinement examples. In general, refining a superpixel-level segmentation via the guided filter [27] can align the object boundaries properly with pixel-level precision. Fig. 4.

3.6. Interaction mechanisms

Our algorithm assumes that the user provides a bounding box at first. At each round of interaction, our algorithm automatically shows the user one pixel, which represents a whole superpixel, to query whether the pixel belongs to the foreground object. Fig. 9(e) shows a query example. Hence the user only needs to provide binary responses concerning the queried pixels. A superpixel is chosen for the query if the superpixel is the most uncertain one that is equally likely to be assigned to the foreground or the background, *i.e.*, having a confidence value close to 0.5.

3.7. Pseudo code

We summarize the main steps of the proposed 'BnS' and 'BnQ' in Algorithm 1.

Algorithm 1 Unified Interactive Segmentation

Input: *flag*: binary indicator (true for 'BnS'; false for 'BnQ'); $\mathscr{I}$: an image; b: bounding box; (θ_1, θ_2): parameters; T: the number of interactions;
Output: S: segmentation of $\mathscr{I}$;
1: Construct the graph $\mathscr{G} = (\mathscr{V}, \mathscr{E}, \omega)$ of the image $\mathscr{I}$, and compute the weight matrix $\mathbf{W}$ and the diagonal matrix $\mathbf{D}$;
2: Construct the vector $\mathbf{y}$ with the bounding box b and the superpixel set $\mathscr{V}$;
3: Compute the initial foreground object confidence $\mathbf{c}$ by Eq. (3) and Eq. (4);

[2] Our algorithm updates the confidence values of all not-yet-quired superpixels per interaction for estimating their labels. Include more interactions makes our algorithm acquire more reliable labeled superpixels and hence increase the segmentation accuracy.

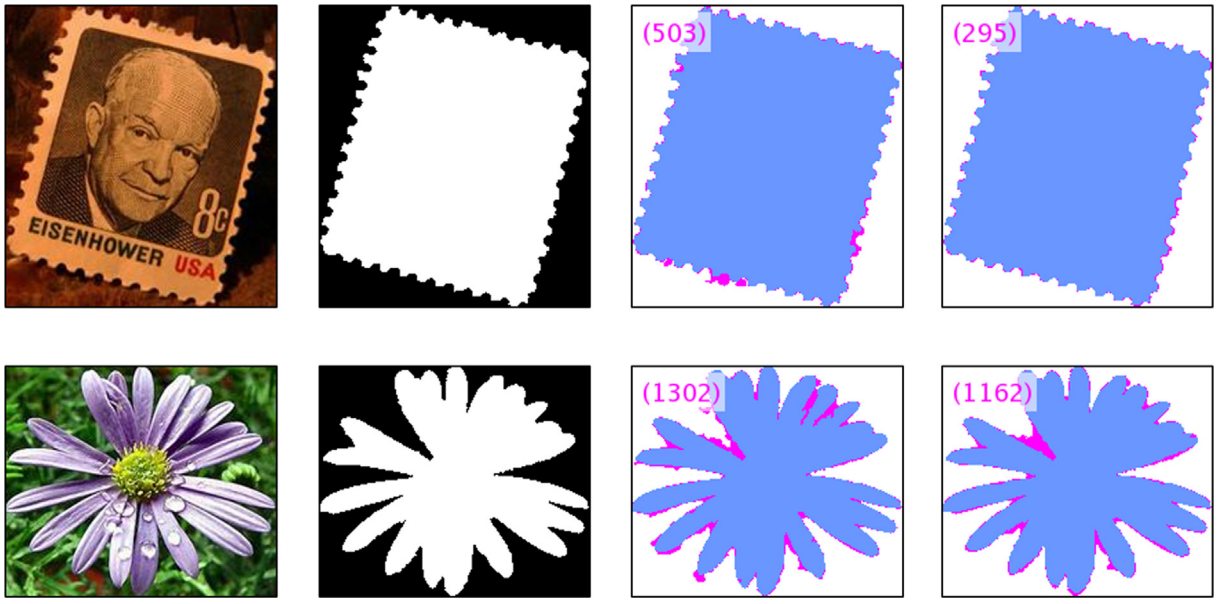


Fig. 3. Some refinement examples. The first column shows the input images. The second column shows the ground-truth segments. The third column shows the superpixel-level segmentations before refinement. The fourth column shows the refined segmentations. In the last two columns, the blue and magenta regions indicate the overlaps or the differences between the segmentations and the ground-truth segments. The magenta number denotes the number of error pixels.

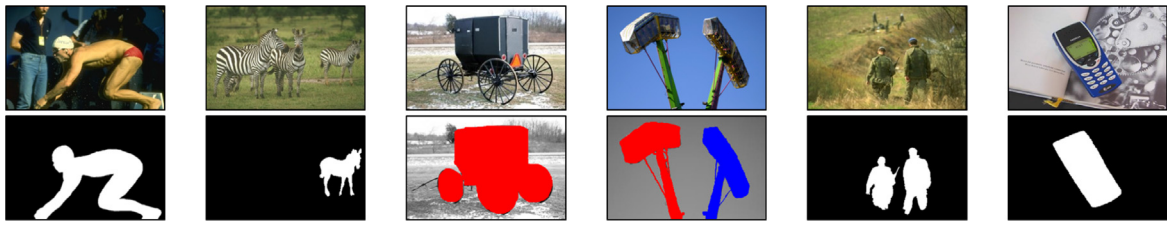


Fig. 4. Some examples of the ground-truth foreground objects from different datasets. Each saturated color denotes a ground-truth foreground object except the black color (background).

4: Normalize the relevance matrix $\mathbf{R}$ of Eq. (5) to form $\widetilde{\mathbf{R}}$;
5: **for** $t = 2$ to T
6: **if** *flag* bf then
7: Obtain the user binary annotation to alter the indicator vector $\mathbf{y}$;
8: **else**
9: Query user the v_i which has the confidence c_i most close to 0.5;
10: Obtain the 1-bit user feedback to alter the indicator vector $\mathbf{y}$;
11: **end if**
12: Update the foreground object confidence $\mathbf{c}$ by Eq. (6) and Eq. (7);
13: **end for**
14: Compute the superpixel-level segmentation by $\mathscr{F} = \{v_i | c_i \geqslant 0.5\}$ and $\mathscr{B} = \{v_i | c_i < 0.5\}$;
15: Refine the superpixel-level segmentation by guided filter to from S;

4. Experiments

We evaluate the proposed method on six datasets in four parts, including evaluations on thirty rounds of interactions, evaluations on one round of interaction, average response time, and some segmentation examples.

4.1. Protocol

(*i*) To evaluate the influence of bounding-box quality, we sample 29 random boxes plus one exact bounding box surrounding the ground-truth foreground object. Based on the intersection over union (IoU)[3] metric, these boxes are sorted by IoU-50% to IoU-100%. Hence, the 30-th box is the one that tightly encloses the ground-truth foreground object. See Fig. 9b. (*ii*) To evaluate seed/scribble-based algorithms in a supervised manner, we select the centroid of the biggest connected component among the exclusive-or regions between the machine segmentation and the ground-truth segmentation at each round as the procedure in [17]. Notice that all seed/scribble-based algorithms and our variants ('BnS,' 'SnS,' and 'NaiveBnS') perform interactive image segmentation with this protocol. (*iii*) While testing the segmentation accuracy with the skeleton labels, we first compute the skeleton pixels of the foreground object and the background regions. Next, we take the bounding box that fully encloses the foreground object skeleton pixels for bounding-box-based algorithms. For seed/scribble-based algorithms and ours, the skeleton pixels of the foreground object and the background regions are treated as the foreground seeds and the background

[3] The IoU is defined as $\frac{|S \bigcap G|}{|S \bigcup G|}$, where S denotes the sampled *bounding box* and G denotes the ground-truth *bounding box* that tightly encloses the foreground object. The value of this metric ranges from 0% to 100%, the higher value means the sampled box more tightly encloses the foreground object.

Table 1
The statistics of the six datasets.

Dataset	GrabCut	BSDDCU	WSDB1	WSDB2	ECSSD	MSRA
No. of Images	50	100	100	100	1000	1000
No. of Ground-truths	50	100	300	554	1000	1000

seeds. See Fig. 9d.

4.2. Datasets

The evaluation datasets are listed as follows and their statistics are summarized in the Table 1: (*i*) GrabCut dataset [6]. (*ii*) BSDDCU dataset [28]. (*iii*) Weizmann database [29], which contains two sub-datasets, *i.e.*, one-object dataset (WSDB1) and two-object dataset (WSDB2). (*iv*) Extended complex scene saliency dataset (ECSSD) [30]. (*v*) MSRA dataset [31].

4.3. The algorithms for comparison

Our algorithm is compared with various state-of-the-art interactive segmentation algorithms, including three bounding-box-based algorithms and five seed/scribble-based algorithms listed as follows[4]: MilCut-graph (MCg) and MilCut-struct (MCs) [14], GrabCut (GC) [6], Lazy Snapping (LS) [5], Random Walks (RW) [8], Interactive Graph Cuts (IGC) [3], Geodesic Star Convexity (GSC) [11], and OneCut with seeds (OCs) [13]. Notice that, every segment in each individual ground-truth annotation is selected as a ground-truth foreground object for evaluation.

4.4. Evaluation metric

To measure the segmentation quality, we employ the median of the Dice score [32] defined as

$$dice(S, G) = \frac{2|S \bigcap G|}{|S| + |G|}, \tag{8}$$

where S denotes the machine segmentation and G denotes the ground-truth foreground object.

4.5. Parameter setting

We use the SLIC algorithm [26] to generate the set $\mathscr{S}$ with roughly 800 superpixels [5]. The parameters $\theta_1 = 60$ and $\theta_2 = 0.999$ are empirically set[6] as [21] for all experiments.

4.6. Thirty rounds of interactions

The first experiment aims to evaluate the segmentation accuracy against 30 rounds of interactions. In our algorithm, only the initial box is provided by the user; after initialization, at each round our algorithm actively asks the user to decide the label of an automatically chosen pixel-location marked by a red cross sign as shown in Fig. 9e. We use the procedure in [17] to automatically synthesize the next seed position as a new user-input.

4.6.1. The importance of our confidence update step

In the confidence update step, we design Eq. (6) and Eq. (7) to update the object confidence with respect to the user feedback. Eq. (6) and Eq. (7) are important to make the unified scheme available in the interactive segmentation problem. Naively using Eq. (3) and Eq. (4) to update the foreground object confidence is not working. The difficulty is that the initialized background superpixels outnumber the labeled foreground superpixels acquired from the interactions, and thus the foreground confidence is hard to update via Eq. (3) and Eq. (4). In Fig. 5, we denote the naive combination approach as 'NaiveBnS[30-Refine],' which updates the confidence via Eq. (3) and Eq. (4) and adopts the highest quality box and refinement. By comparing 'BnS[30-Refine]' with 'NaiveBnS[30-Refine]' among the six datasets, we can see that the segmentation accuracy of the naive update approach is hard to improve with the user feedback. In contrast, our confidence update step clearly improves the performances of 'BnS' and 'BnQ' with the user feedback.

4.6.2. The refinement step and the initial box

We then explore the effects of the refinement step and the initial box's quality in Fig. 5. Comparing 'BnS[30-NotRefine]' with 'BnS[30-Refine]' among six datasets, we can see that the refinement step can improve the segmentation accuracy by 2% in average, except the WSDB2 dataset. Next, by comparing 'BnS[30-Refine]' with 'BnS[01-Refine]' or by comparing 'BnQ[30-Refine]' with 'BnQ[01-Refine]' among the six datasets, we can find that using the highest quality box shows merely slight improvement in segmentation accuracy in comparison with using the lowest quality box.

It is interesting that the segmentation accuracy of 'BnS[30-NotRefine]' is better than that of 'BnS[30-Refine]' in Fig. 5(d). We observe that the WSDB2 dataset contains some rough (expansion) segmentations as the ground-truths, which can be observed in the Fig. 6(d). The superpixel-level machine segmentations depicting the foreground objects are usually the sub-area of the expansion ground-truths. Unfortunately, the refinement step usually sharpens the superpixel-level machine segmentations and hence reduces the median Dice score more between the machine segmentations and the expansion ground-truths. However, note that the range of the y-axis in Fig. 5(d) is very narrow, hence the difference of the segmentation accuracy between 'BnS[30-NotRefine]' and 'BnS[30-Refine]' is small.

In sum, Fig. 5 shows that (*i*) the refinement step benefits the segmentation accuracy, and (*ii*) our algorithm is not sensitive to the choice of an initial bounding box. Both high and low IoU boxes yield comparably good segmentation accuracy.

4.6.3. Comparison with state-of-the-art algorithms

Fig. 7 shows the experimental results compared with various state-of-the-art interactive segmentation algorithms. In our algorithm, 'BnQ,' the user only needs to respond to the automatically generated queries and provide the binary labels of those selected superpixels. The 'BnS' and 'SnS' are two approaches initialized with one box and one seed respectively. Notice that, we follow the automatic seed selection strategy [17] for automatically selecting a seed per interaction, and all methods except 'BnQ' use such a strategy.

In Fig. 7, owing to the fact that 'BnS' has clearly better segmentation accuracy than 'SnS,' the idea to infer the foreground region via manifold ranking with collecting the potential background superpixels from an initial bounding box is well founded. Furthermore, the segmentation accuracy of the 'BnS' approach is obviously better than all other algorithms. Actually, the performance of 'BnS' is what our machine-assisted 'BnQ' approach pursues. 'BnQ' has better segmentation accuracy than other state-of-the-art algorithms in the first 12 to 20 rounds of interactions. According to our experiments, we suggest using twenty rounds

[4] The programs of MCg and MCs are provided by the authors. The programs of GC and LS are implemented by Gupta and Ramnathhttp://www.cs.cmu.edu/mohitg/segmentation.html. The code of RW is fromhttp://cns.bu.edu/lgrady/software.html by Leo Grady. The programs of IGC and GSCs are fromhttp://www.robots.ox.ac.uk/vgg/research/iseg/. The code of OCs is fromhttp://vision.csd.uwo.ca/code/.

[5] The number of superpixels is empirically selected accounting for the segmentation accuracy and the computational time. The larger number of superpixels improves the segmentation accuracy, especially for the small and thin foreground objects, yet increases the computational time while calculating the manifold ranking Eq. (3). Note that the number of superpixels generated by the SLIC algorithm [26] usually slightly less than the specified amount.

[6] For comparison, the parameters in [25] are $\theta_1 = 10$ and $\theta_2 = 0.99$.

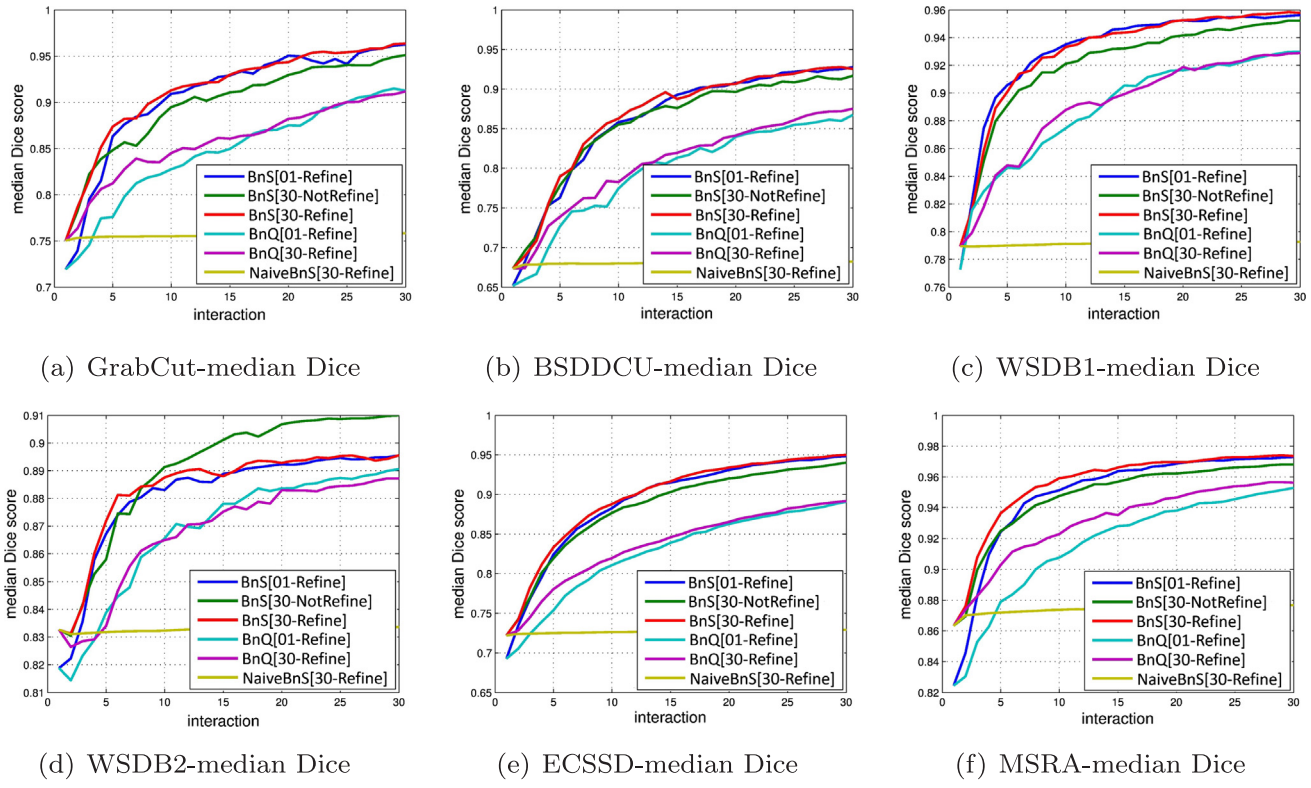


Fig. 5. We compare different methods by the segmentation accuracy against the number of interactions among the six datasets. The number in '[·]' denotes the ID of a bounding box, where '01' means the lowest IoU box and '30' means the best box. Note that, the sub-figures have different and narrow ranges of the median Dice score in the vertical axis.

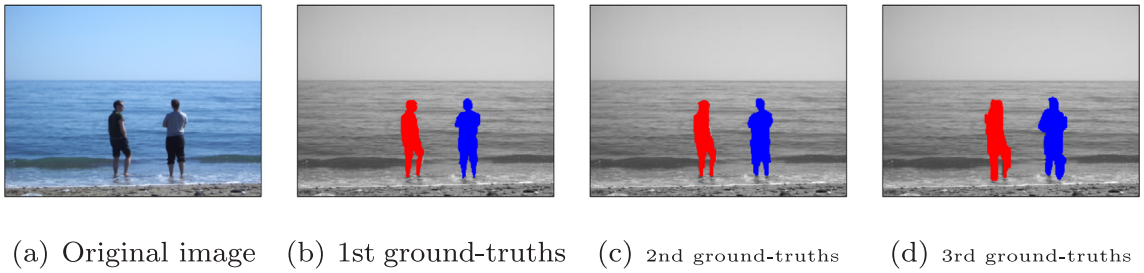


Fig. 6. An example of an original image and the corresponding three ground-truths in the WSDB2 dataset. The red and blue regions denote two different foreground object ground-truths.

of interactions to reach the segmentation accuracy at least 83% mean Dice score. The experimental results clearly show that our unified approaches reduce the needed interactions to achieve a certain degree of segmentation accuracy.

4.6.4. Comparison with query-based algorithms

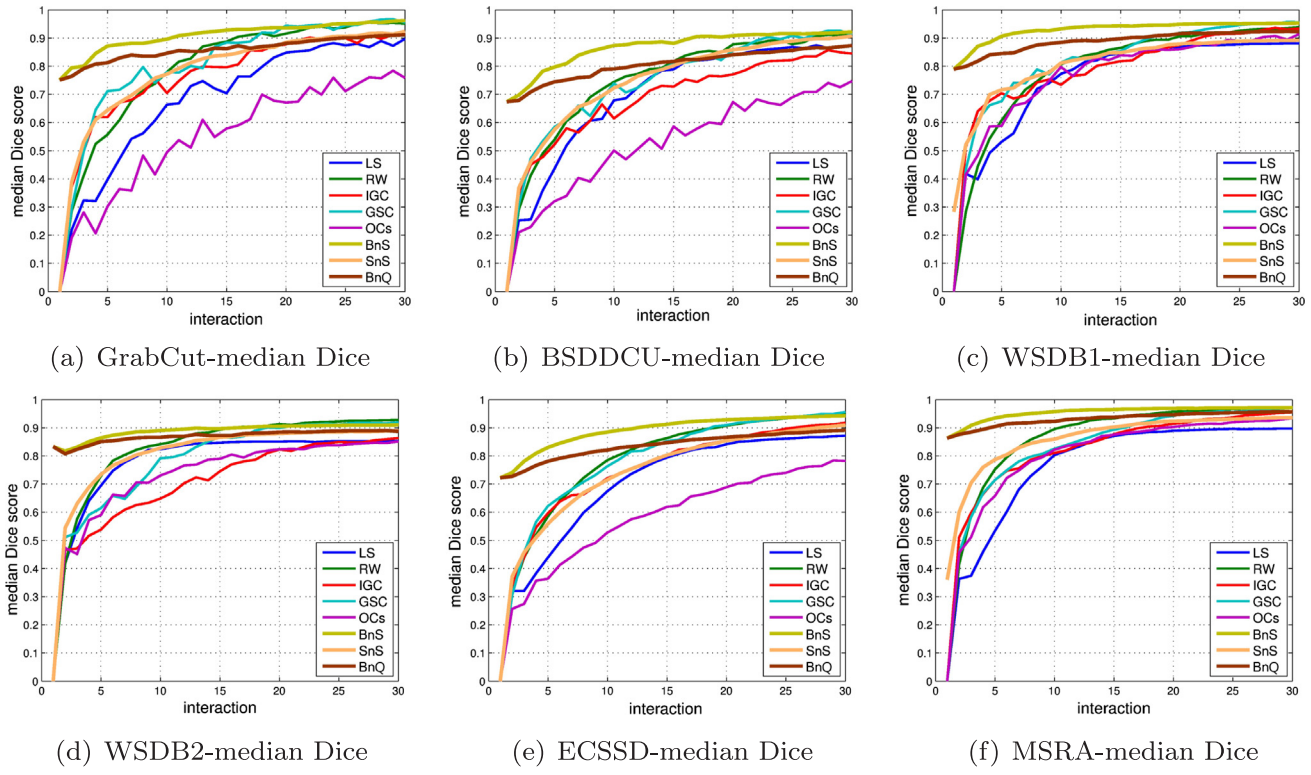
In the previous sections, we refer to two query-based algorithms, proposed by Rupprecht et al [15] and Chen *et al.* [21]. We compare our 'BnS' and 'BnQ' approaches with theirs on segmentation accuracy, as shown in Fig. 8 We use the datasets as Rupprecht *et al.* [15] and Chen *et al.* [21] for evaluation: (*i*) BSDS dataset [33]. (*ii*) IBSR dataset, where the brain dataset and their manual segmentations are available athttp://www.cma.mgh.harvard.edu/ibsr/. (*iii*) SBD dataset [34]. Note that, there are 18 subjects in IBSR dataset, and we extract 90 brain slices ranging from the 20th slice to the 109th slice for each subject. We can see that our approaches outperform their algorithms on three datasets, except that the 'BnQ' approach is on par with the others on IBSR and SBD datasets after 13 rounds. With the aid of an initial box, our algorithm avoids searching the region of interest at the first few interactions and hence has a better chance to revise the segmentation.

4.7. One round of interaction

The second experiment evaluates the one-round segmentation accuracy, in which each experiment is performed as one round of interaction either using the information derived from the skeleton labels or using the initial box for segmentation. The skeleton labels are computed from both foreground and background regions for each ground-truth foreground object. See Fig. 9 for illustration. For bounding-box-based algorithms, we take the bounding box that fully encloses all the foreground skeleton pixels. For seed/scribble-based algorithms and ours, the skeleton pixels of both foreground and background regions are treated as the foreground and background seeds.

4.7.1. Using the skeleton labels

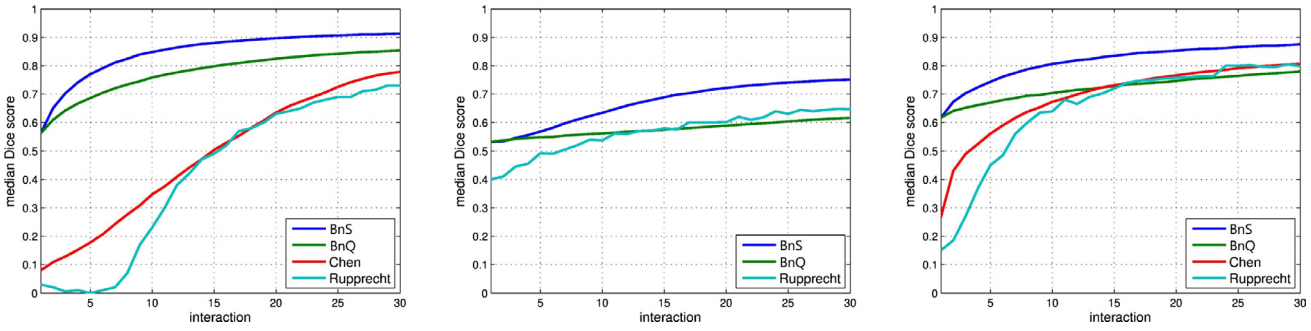
Here we evaluate each method with using the metric of mean Dice score for reference. Notice that, we take the bounding box that fully encloses the foreground object skeleton pixels for bounding-box-based algorithms (MCg, MCs, GC), and we directly take the skeleton pixels of the foreground object and the background regions as the foreground seeds and the background seeds for all other algorithms (LS, RW, IGC,



(a) GrabCut-median Dice (b) BSDDCU-median Dice (c) WSDB1-median Dice

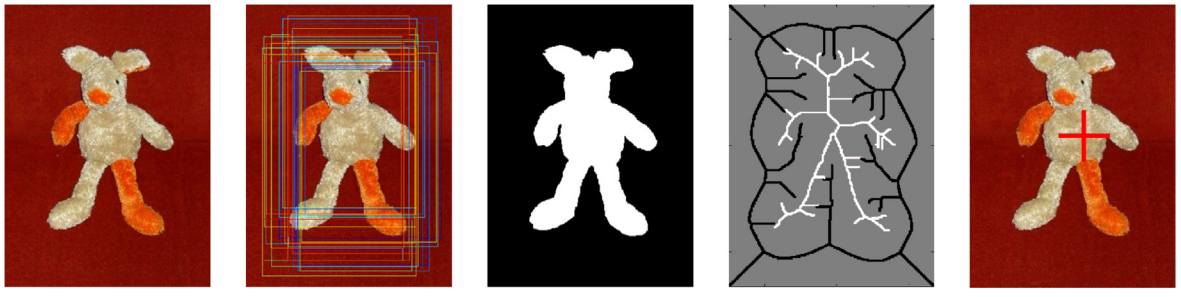
(d) WSDB2-median Dice (e) ECSSD-median Dice (f) MSRA-median Dice

Fig. 7. Evaluations on the segmentation accuracy against the number of interactions. Note that, all methods except the proposed 'BnQ' approach are fully user supervised, *i.e.*, the user has to observe the segmentation results and then to specify the labeled seeds one by one. On the other hand, our machine-assisted method actively queries regions for the user to respond with the correct binary labels, in which the user only needs to give the true-or-false feedback simply.



(a) BSDS-median Dice (b) IBSR-median Dice (c) SBD-median Dice

Fig. 8. This figure shows the comparison results on three additional datasets (BSDS, IBSR, and SBD) presented in [15]. The algorithms are evaluated by the segmentation accuracy against the number of interactions. 'Chen' means the algorithm proposed in [21]; 'Rupprecht' means the algorithm proposed in [15]. We directly replicate their results in this figure according to their experiments in [15,21].



(a) Input image (b) 30 boxes (c) Ground-truth (d) Skeleton-labels (e) Query example

Fig. 9. Input examples. (a) The input image. (b) Various input boxes for evaluating the sensitivity of initial bounding box. (c) The ground-truth segmentation. (d) The input skeleton labels computed from both foreground and background regions of (c). (e) The query example.

Table 2
Quantitative comparisons on segmentation accuracy with the skeleton labels: The mean Dice scores (%) among the six datasets. The best three scores of each dataset are colored in red, green, and blue.

mean Dice	MCg	MCs	GC	LS	RW	IGC	GSC	OCs	BnQ
GrabCut	77.0	77.8	81.8	92.6	93.7	93.3	96.6	91.2	94.7
BSDDCU	78.4	77.4	78.6	90.0	90.9	84.9	92.2	86.3	91.3
WSDB1	82.5	82.4	80.1	89.6	93.4	94.6	94.9	93.2	95.0
WSDB2	79.9	79.8	68.9	86.1	89.7	81.1	90.1	82.7	88.1
ECSSD	82.1	81.0	79.8	91.7	95.5	93.5	95.5	91.8	94.8
MSRA	85.2	85.3	84.6	91.2	96.6	95.8	96.7	94.2	97.2

Table 3
The average response time (seconds) per round of different algorithms. The measurement is done on an Intel i7-4770 3.40 GHz CPU with 8 GB RAM. The timing results are obtained using the MSRA dataset.

Algorithm	MCg	MCs	GC	LS	RW	IGC	GSC	OCs	BnS	BnQ
Second	1.84	1.32	2.78	0.33	0.72	0.34	0.61	0.47	0.008	0.014

GSC, OCs, BnQ). Table 2 shows the Dice scores of different algorithms using the skeleton labels as user inputs. Though the proposed algorithm is not always the best in this evaluation, it is usually ranked at the top three places.

4.7.2. Using the bounding box only

It is also worth mentioning that our method is not sensitive to the choice of an initial bounding box. In Fig. 5, we have seen that both of the highest and lowest IoU boxes can yield good segmentation accuracy using our algorithm. This experiment further explores the robustness of segmentation accuracy with respect to the initial box. We randomly sample 29 boxes with IoU higher than 50% plus one bounding box that tightly encloses the ground-truth foreground object (IoU-100%), as shown in Fig. 9(b). These boxes can be sorted from imprecise to precise according to IoU. Hence, the 30th box is the one that tightly encloses the ground-truth foreground object. Note that, the values on each line are independent, they denote the segmentation accuracy of each method with different bounding box input.Table 3.

We compare our algorithm with other bounding-box-based segmentation algorithms (MCg, MCs, GC) in Fig. 10. We can see that the quality of the initial bounding box has a significant effect on other bounding-box-based segmentation algorithms. On the contrary, our algorithm is not sensitive to the choice of an initial bounding box. Hence, the robustness of our algorithm concerning the initial estimation of the foreground object confidence is evident. The results again imply

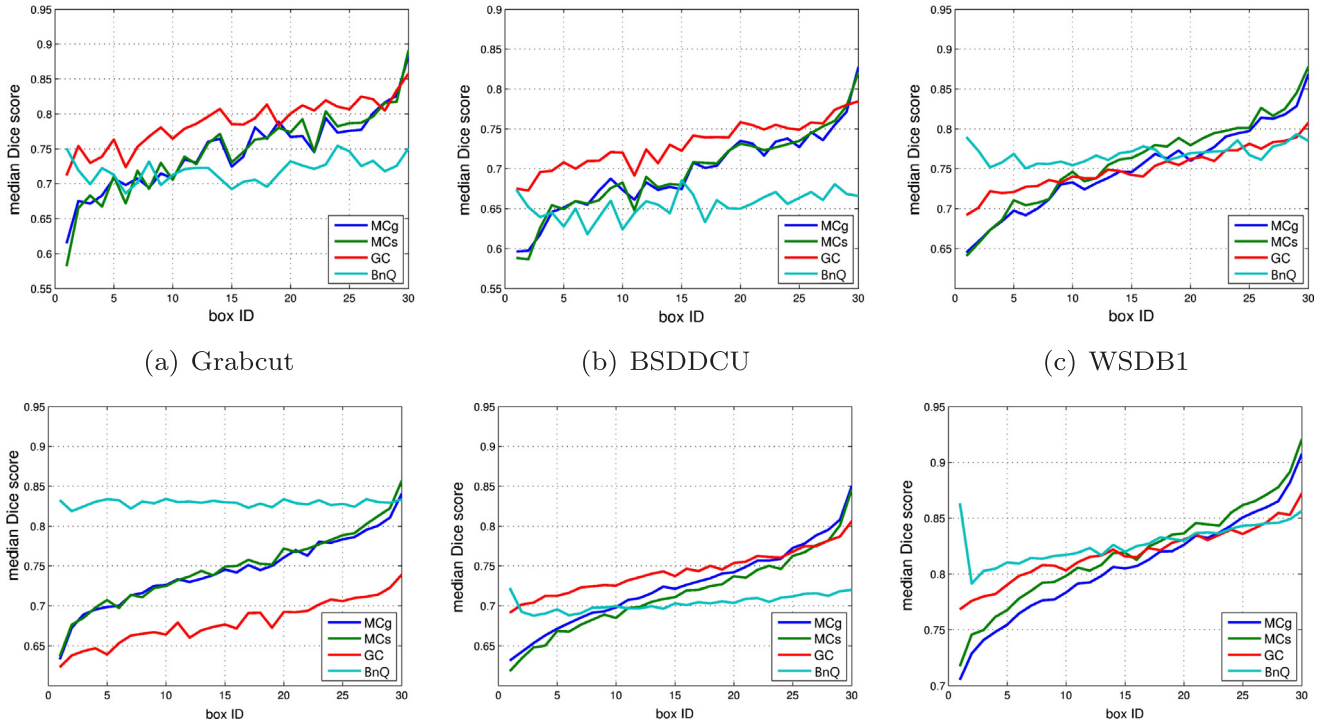


Fig. 10. Evaluation of the segmentation accuracy in median Dice against the initial bounding boxes with different IoU. The number on the x-tick denotes the ID of the bounding box, where '01' means the lowest IoU box and '30' means the best box. Note that, the values on each line are independent, they denote the segmentation accuracy of each method that performs segmentation with only one bounding box input and without any further interactions.

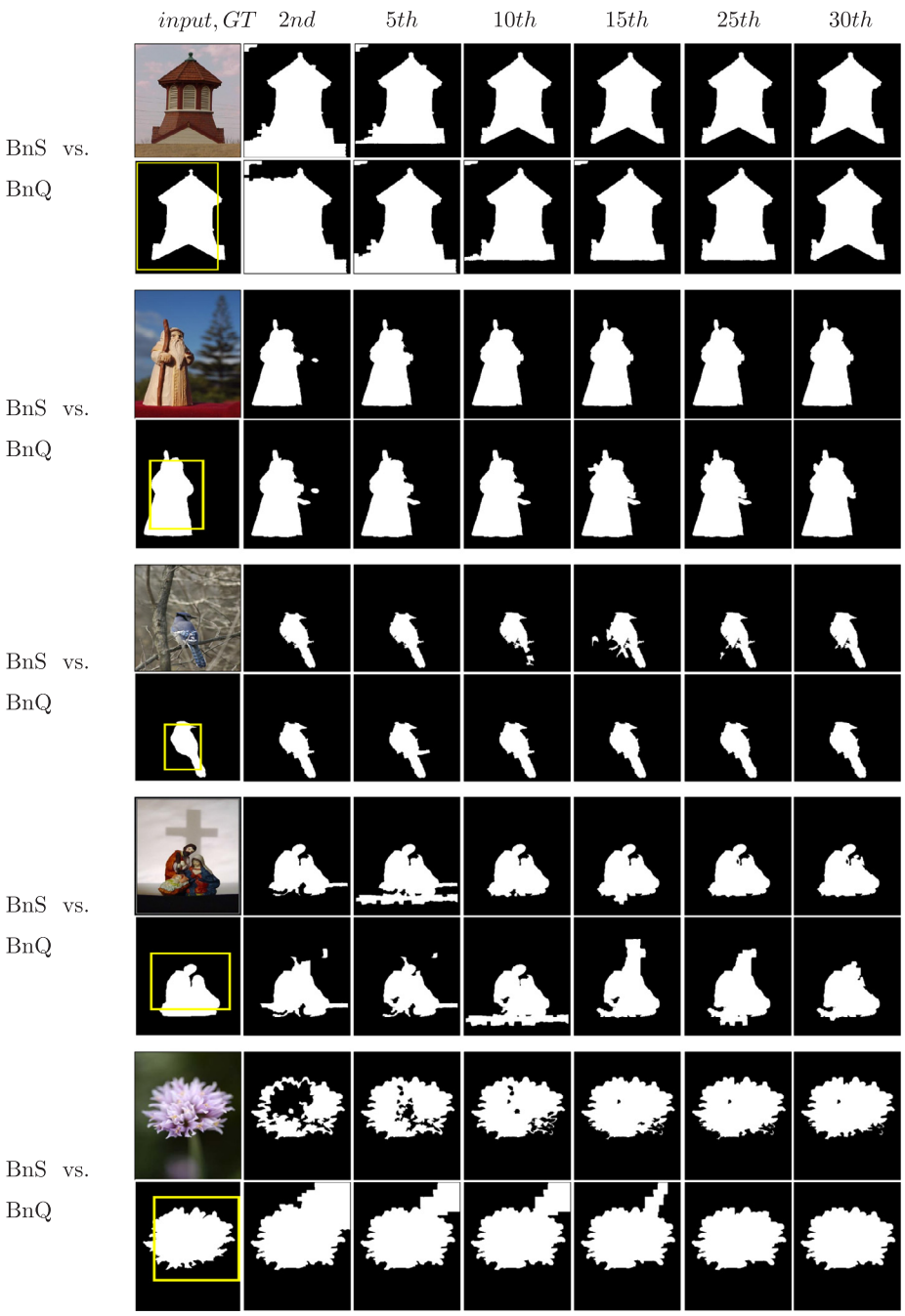


Fig. 11. Segmentation results. The first column shows the input image and the ground-truth object region with the initial bounding box. The first row except the first column shows the segmentation results over 30 rounds of performing the 'BnS' method. The second row shows the segmentation results over 30 rounds of performing our algorithm. From the second column to the last column we illustrate the segmentation results of the 2nd, 5-th, 10-th, 15-th, 25-th, and 30-th round, respectively.

that our method has a lower requirement on the initial bounding box and is, therefore, easier to use.

4.8. Response time

Our algorithm is efficient for interactive segmentation. The average response time per round for our method under the user manually inputs the labels (as 'BnS') is less than 0.01 seconds. For user assistance mechanism (as 'BnQ'), the average response time per iteration is less than 0.02 second with the additional computational cost on handling query superpixels. It takes about 0.2 seconds for graph construction and 0.1 seconds for refinement. However, both the steps only need to be done once, before and after the interaction phase, respectively. Hence, both steps wouldn't degrade the efficiency of the entire algorithm. Therefore, the total computational time of our algorithm to derive the final segmentation in 30 rounds of interactions is less than 0.4 seconds.

4.9. Segmentation examples

Finally, we show some segmentation examples of our algorithm in Fig. 11. We use the lowest-IoU box as the initial bounding box. In Fig. 11, the first row (except the first column) shows the segmentation results of 'BnS,' and the second row shows the segmentation results of 'BnQ.' Fig. 11 demonstrates that both of the two variants of our algorithm can obtain the satisfactory segmentation results in few interactions.

5. Conclusion

This work provides a unified scheme for interactive image segmentation, in which the initial box defines potential background labels, the manifold ranking strategy efficiently estimates foreground confidence, and the interaction mechanism allows two forms of user inputs: fully-user-supervised as 'BnS' or machine-assisted as 'BnQ.' If the user is willing to specify the labeled seeds one by one, then the 'BnS' approach is suitable for such a scenario. On the other hand, an interesting property of the 'BnQ' approach is that the interactions, besides the initial box, could be done with 1-bit feedback, which can be used on hand-held devices or for hands-free scenarios. The extensive evaluations and comparisons demonstrate that the proposed algorithm is not sensitive to the choice of the initial bounding box, and it yields a simple, flexible, and short-response-time scheme for fast interactive segmentation.

References

[1] M. Kass, A.P. Witkin, D. Terzopoulos, Snakes: Active contour models, Int. J. Comput. Vision 1 (4) (1988) 321–331.
[2] E.N. Mortensen, W.A. Barrett, Intelligent scissors for image composition, in: SIGGRAPH, 1995.
[3] Y. Boykov, M. Jolly, Interactive graph cuts for optimal boundary and region segmentation of objects in N-D images, in: ICCV, 2001.
[4] A. Blake, C. Rother, M.A. Brown, P. Pérez, P.H.S. Torr, Interactive image segmentation using an adaptive GMMRF model, in: ECCV, 2004.
[5] Y. Li, J. Sun, C. Tang, H. Shum, Lazy snapping, ACM Trans. Graph. 23 (3) (2004) 303–308.
[6] C. Rother, V. Kolmogorov, A. Blake, grabcut: interactive foreground extraction using iterated graph cuts, ACM Trans. Graph. 23 (3) (2004) 309–314.
[7] D. Freedman, T. Zhang, Interactive graph cut based segmentation with shape priors, in: CVPR, 2005.
[8] L. Grady, Random walks for image segmentation, IEEE Trans. Pattern Anal. Mach. Intell. 28 (11) (2006) 1768–1783.
[9] S. Vicente, V. Kolmogorov, C. Rother, Graph cut based image segmentation with connectivity priors, in: CVPR, 2008.
[10] V.S. Lempitsky, P. Kohli, C. Rother, T. Sharp, Image segmentation with a bounding box prior, in: ICCV, 2009.
[11] V. Gulshan, C. Rother, A. Criminisi, A. Blake, A. Zisserman, Geodesic star convexity for interactive image segmentation, in: CVPR, 2010.
[12] N.T.N. Anh, J. Cai, J. Zheng, J. Li, Interactive object segmentation from multi-view images, J. Visual Commun. Image Representation 24 (4) (2013) 477–485.
[13] M. Tang, L. Gorelick, O. Veksler, Y. Boykov, Grabcut in one cut, in: ICCV, 2013.
[14] J. Wu, Y. Zhao, J. Zhu, S. Luo, Z. Tu, Milcut: A sweeping line multiple instance learning paradigm for interactive image segmentation, in: CVPR, 2014.
[15] C. Rupprecht, L. Peter, N. Navab, Image segmentation in twenty questions, in: CVPR, 2015.
[16] T. Wang, Z. Ji, Q. Sun, Q. Chen, S. Han, Image segmentation based on weighting boundary information via graph cut, J. Visual Commun. Image Representation 33 (2015) 10–19.
[17] J. Feng, B. Price, S. Cohen, S. Chang, Interactive segmentation on rgbd images via cue selection, in: CVPR, 2016.
[18] E. Zemene, M. Pelillo, Interactive image segmentation using constrained dominant sets, in: ECCV, 2016.
[19] L. Luo, X. Wang, S. Hu, X. Hu, L. Chen, Interactive image segmentation based on samples reconstruction and FLDA, J. Visual Commun. Image Representation 43 (2017) 138–151.
[20] T.F. Chan, L.A. Vese, Active contours without edges, IEEE Trans. Image Processing 10 (2) (2001) 266–277.
[21] D. Chen, H. Chen, L. Chang, Interactive segmentation from 1-bit feedback, in: ACCV, 2016.
[22] M. Pavan, M. Pelillo, Dominant sets and pairwise clustering, IEEE Trans. Pattern Anal. Mach. Intell. 29 (1) (2007) 167–172.
[23] D. Zhou, O. Bousquet, T.N. Lal, J. Weston, B. Schölkopf, Learning with local and global consistency, in: NIPS, 2003.
[24] D. Zhou, J. Weston, A. Gretton, O. Bousquet, B. Schölkopf, Ranking on data manifolds, in: NIPS, 2003.
[25] C. Yang, L. Zhang, H. Lu, X. Ruan, M. Yang, Saliency detection via graph-based manifold ranking, in: CVPR, 2013.
[26] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, S. Süsstrunk, SLIC superpixels compared to state-of-the-art superpixel methods, IEEE Trans. Pattern Anal. Mach. Intell. 34 (11) (2012) 2274–2282.
[27] K. He, J. Sun, X. Tang, Guided image filtering, IEEE Trans. Pattern Anal. Mach. Intell. 35 (6) (2013) 1397–1409.
[28] K. McGuinness, N.E. O'Connor, A comparative evaluation of interactive segmentation algorithms, Pattern Recogn. 43 (2) (2010) 434–444.
[29] S. Alpert, M. Galun, R. Basri, A. Brandt, Image segmentation by probabilistic bottom-up aggregation and cue integration, in: CVPR, 2007.
[30] J. Shi, Q. Yan, L. Xu, J. Jia, Hierarchical image saliency detection on extended CSSD, IEEE Trans. Pattern Anal. Mach. Intell. 38 (4) (2016) 717–729.
[31] R. Achanta, S.S. Hemami, F.J. Estrada, S. Süsstrunk, Frequency-tuned salient region detection, in: CVPR, 2009.
[32] T. Sensen, A method of establishing groups of equal amplitude in plant sociology based on similarity of species and its application to analyses of the vegetation on danish commons, Kongelige Danske Videnskabernes Selskab 5 (4) (1948) 1–34.
[33] C.C. Fowlkes, D.R. Martin, J. Malik, Local figure-ground cues are valid for natural images, J. Vision 7 (8) (2007) 1–9.
[34] S. Gould, R. Fulton, D. Koller, Decomposing a scene into geometric and semantically consistent regions, in: ICCV, 2009.