Hindawi Publishing Corporation
The Scientific World Journal
Volume 2014, Article ID 296074, 9 pages
http://dx.doi.org/10.1155/2014/296074

Research Article

Automatic Foreground Extraction Based on Difference of Gaussian

Yubo Yuan,[1] Yun Liu,[1] Guanghui Dai,[1] Jing Zhang,[1,2] and Zhihua Chen[1]

[1] *Department of Computer Science and Engineering, East China University of Science and Technology, Shanghai 200237, China*
[2] *State Key Lab for Novel Software Technology, Nanjing University, Nanjing 210093, China*

Correspondence should be addressed to Jing Zhang; jingzhang@ecust.edu.cn and Zhihua Chen; czh@ecust.edu.cn

Received 24 April 2014; Revised 8 July 2014; Accepted 8 July 2014; Published 20 July 2014

Academic Editor: Shan Zhao

A novel algorithm for automatic foreground extraction based on difference of Gaussian (DoG) is presented. In our algorithm, DoG is employed to find the candidate keypoints of an input image in different color layers. Then, a keypoints filter algorithm is proposed to get the keypoints by removing the pseudo-keypoints and rebuilding the important keypoints. Finally, Normalized cut (Ncut) is used to segment an image into several regions and locate the foreground with the number of keypoints in each region. Experiments on the given image data set demonstrate the effectiveness of our algorithm.

1. Introduction

In the image processing, the foreground is an integral part of the objective image. It takes important advantage in many applications [1–4]. For example, in the field of object recognition, in 2011, Rosenfeld and Weinshall [5] proposed an algorithm to extract a foreground mask and to identify the locations of objects in the image. In the field of object tracking, in 2012, Wang et al. [6] used partial least squares (PLS) analysis to label the foreground and background of an image and the results showed that the proposed tracking algorithm was very powerful with the labeled foreground. In the field of content-based image retrieval, in 2006, Shekhar and Chaudhuri [7] investigated the influence of the foreground. In the field of image editing, in 2008, Levin et al. [8] indicated that the process of extracting a foreground object from an image based on limited user input was an important task in image editing.

In current methods, foreground extraction can be classified into two categories [9], one is the interactive foreground extraction and the other is the automatic one. The interactive foreground extraction can accurately find artificial areas from the input images; however, it can do nothing when the task is to extract foregrounds from thousands of images. In this case, the technology of extracting foregrounds automatically is becoming more and more important. Moreover, it can be applied in many fields, such as image segmentation, image enhancement, object recognition, and content-based image retrieval.

In 2011, Kim et al. [10] proposed an automatic method to extract a foreground object captured from multiple viewpoints. Their result was the high quality alpha mattes of the foreground object consistently across all different viewpoints. In 2012, Zhang et al. [11] proposed a technique of automatic foreground-background segmentation based on depth from coded aperture. Their entire progress was fully automatic, without any manual intervention. In 2013, Hsieh and Lee [12] proposed an automatic trimap generation technique. The experimental results showed that the trimap generated by the proposed method effectively improves the matting result. Moreover, they processed the enhancement of the accuracy of the trimap results in a reduction of regions, so that the extraction procedure can be accelerated.

In recent years, with the development of video technology, the research on the automatic foreground extraction will become more and more popular. In this paper, we present a novel approach for automatic foreground extraction based on difference of Gaussian. We employ the difference of Gaussian (DoG) to find candidate keypoints. After filtering and rebuilding the candidate keypoints, we get the refined keypoints of an input image. With Ncut, we extract the foreground from the original image.

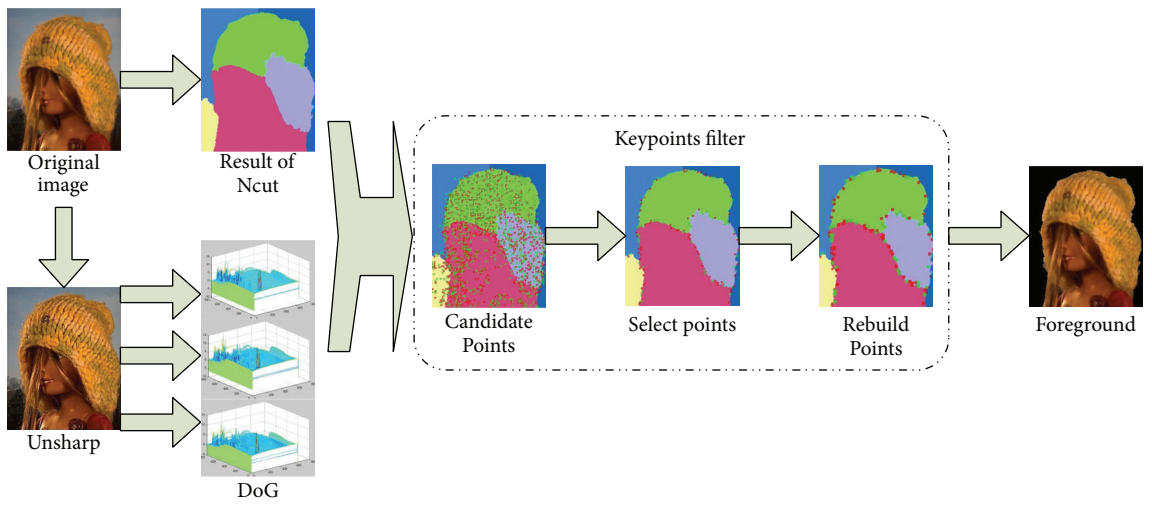


Figure 1: Flowchart of the basic procedure of FMDOG. The result of difference of Gaussian (DoG) and Normalized cut (Ncut) is combined by the keypoints filter. The last image is the foreground we extract in this model image.

The rest of this paper is organized as follows. In Section 2, we introduce the key steps of proposed extraction algorithm, including basic framework, candidate keypoints locating, and keypoints filter. In Section 3, the algorithm is presented. In Section 4, some excellent experimental results are shown. Finally, we give a conclusion with this research work.

2. Foreground Extraction Based on Difference of Gaussian

2.1. Framework. The motivation of this work is to develop a useful technology to extract the foreground from an input image. The contributions of this paper are as follows.

(i) A new procedure to find candidate keypoints in different color layers is proposed. It helps us to find the point that has a more obvious color difference than its neighbours. It is implemented efficiently by using a difference of Gaussian function to find candidate keypoints. We fulfill this task in different color layers.

(ii) Novel filtering operators are constructed to remove the pseudo-keypoint and rebuild the important keypoints. This stage can be summarized into two steps. The first is to reduce the candidate keypoints to find the edge of the foreground by the result of Ncut. Another step is to rebuild the points to increase the proportion of candidate keypoints by a novel approach. At last, we call these candidate points keypoints.

(iii) Novel operator for foreground extraction is proposed. We locate the foreground by the proportion of keypoints and segment the foreground by the result of Ncut.

The basic procedures to locate the foreground are illustrated as in Figure 1.

2.2. Regional Segmentations Based on Normalized Cut. In this work, we use a well-known technique to segment the input image, namely, the Ncut method [15]. Based on the studies in [15], Ncut is from graph-partitioning method. We map the image into a graph $\mathbf{G} = (\mathbf{V}, \mathbf{E}, \mathbf{W})$, where $\mathbf{G} = (\mathbf{V})$ is the set of nodes, and $\mathbf{G} = (\mathbf{E})$ is the set of edges connecting the nodes. A pair of nodes u and v is connected by edge and is weighted by $w(u, v) = w(v, u) \geq 0$ to measure the dissimilarity between them. The basic optimal model of graph cut can be given by

$$\mathbf{C}^* = \arg\min_{\mathbf{C}\subset\mathbf{V}} \{\text{cut}(\mathbf{C})\}, \tag{1}$$

in which

$$\text{cut}(\mathbf{C}) = \sum_{u\in\mathbf{C}, v\in\mathbf{V}-\mathbf{C}} w(u, v). \tag{2}$$

In [15], Shi and Malik proposed Ncut method based on the following optimal model:

$$\mathbf{C}^* = \arg\min_{\mathbf{C}\subset\mathbf{V}} \{\text{Ncut}(\mathbf{C})\}, \tag{3}$$

in which

$$\text{Ncut}(\mathbf{C}) = \frac{1}{\text{normal}(\mathbf{C})}\text{cut}(\mathbf{C}), \tag{4}$$

where

$$\text{normal}(\mathbf{C}) = \frac{\text{assoc}(\mathbf{C}, \mathbf{V}) \cdot \text{assoc}(\mathbf{V} - \mathbf{C}, \mathbf{V})}{\text{assoc}(\mathbf{C}, \mathbf{V}) + \text{assoc}(\mathbf{V} - \mathbf{C}, \mathbf{V})}, \tag{5}$$

in which assoc(**C**,**V**) denotes the total connection from nodes in **C** to all nodes in the graph and assoc($\mathbf{V} - \mathbf{C}$, **V**) is similarly defined.

In practice, Ncut is a powerful segmentation method. After employing the Ncut algorithm, if we denote the jth part of the ith input image as $\mathbf{R}_{ij}$, the input image $\mathbf{I}_i$ is segmented into M_i parts

$$\mathbf{I}_i = \bigcup_{j=1}^{M_i} \mathbf{R}_{ij}. \tag{6}$$

2.3. Candidate Keypoints Locating. An input image $I(x, y)$ can be seen as a surface on square domain $\Omega = [a, b] \times [c, d]$. In the discrete space, if the total number of pixels is $m \times n$, the domain can be represented as $\{1, 2, \ldots, m\} \times \{1, 2, \ldots, n\}$.

Candidate keypoints of an input image are detected firstly by difference of Gaussian (DoG) algorithm. The difference of Gaussian with the scale σ and constant multiplicative factor k can be computed by

$$D(x, y, \sigma, k) = L(x, y, k\sigma) - L(x, y, \sigma), \quad (x, y) \in \frac{\Omega}{\partial\Omega}, \tag{7}$$

in which $L(x, y, \sigma)$ is the scale space and $\partial\Omega$ denotes the boundary of the input image. $L(x, y, \sigma)$ can be obtained by

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y), \tag{8}$$

where $*$ is the convolution operation and

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2}. \tag{9}$$

In practice, the size is usually chosen as $\sigma = 1.6$ and the constant multiplicative factor is chosen as $k = \sqrt{2}$.

We will detect maxima and minima of the difference of Gaussian images by comparing a pixel to its 26 neighbors in 3×3 regions at the current and adjacent scales. Once the value of a pixel is maxima or minima, we regarded this pixel as a *candidate point*. Mathematically, for any $(x, y) \in \Omega$, we get the map $(g(x, y))$ of the candidate points, where

$$g(x, y) = \begin{cases} 1, & \text{the value of } g(x, y) \text{ is maxima or minima;} \\ 0, & \text{otherwise.} \end{cases} \tag{10}$$

2.4. Keypoints Filtering. In general, the candidate points can not be used to detect right foreground regions. They always gather in the areas with mixed colors. However, these areas may be in the background. Motivated by this observation, a point filtering function is constructed to reduce the candidate points in the background and rebuild some new candidate points in the foreground. We call these candidate points as keypoints.

The filtering function is formulated as

$$f(x, y) = \begin{cases} 0, & S(x, y) > 1; \\ 1, & S(x, y) \le 1, \end{cases} \tag{11}$$

in which $f(x, y) = 1$ means that (x, y) is the candidate point; otherwise (x, y) is not. The selection function can be formulated as follows:

$$S(x, y) = \sum_{x-2 \le i \le x+2, y-2 \le j \le y+2} g(i, j). \tag{12}$$

Remark 1. We create a 5×5 filter to reduce the candidate points. Only one candidate point can remain in the 5×5 filter.

Through the filter, the number of the candidate points is decreased dramatically. In particular, for the dense candidate points in one region, the reduction is obvious.

In the next step, we employ the Ncut to find the edges of an input image. With Ncut, we segment the image into many regions and locate the edge points.

We select the more useful candidate points along the edges. We keep the candidate points on the edges. This process can be formulated as follows:

$$c(x, y) = \begin{cases} 1, & f(x, y) = 1, (x, y) \text{ is one edge point;} \\ 0, & \text{otherwise.} \end{cases} \tag{13}$$

In the next step, we will rebuild some new candidate points.

Firstly, we need to define the focus of an image. We regard center of the region that contains the largest number of candidate points as the focus. The region that the focus in (R_f) can be computed by

$$R_f = \begin{cases} R_i, & \max\left(\sum_{(x,y) \in R_i} c(x, y)\right) > \sum_{(x,y) \notin R_i} c(x, y); \\ R, & \text{otherwise,} \end{cases} \tag{14}$$

where R_i is the ith region of an input image and

$$R = \bigcup R_i. \tag{15}$$

The initial focus is located in the center of the image. The focus may shift to the other regions that contains the largest number of candidate points.

The candidate points will be rebuilt towards the focus. For an input image, there are four boundaries. We reserved the boundary information included in each area in the candidate points of this area. Some weight is added to each already existed point. The weight is determined by the boundary information of the candidate point. The more boundaries it contains, the smaller the weight is. The weight can be computed by

$$w(x, y) = \begin{cases} 1, & \text{the region this point in contains 0 boundary;} \\ 0.5, & \text{the region this point in contains 1 boundary;} \\ 0, & \text{otherwise.} \end{cases} \tag{16}$$

We regard the candidate points which are obtained from the above steps as keypoints. At last, we locate the foreground by the number of keypoints in each region. The process of filtering keypoints is displayed in Figure 2.

3. Proposed Algorithm to Extract the Foreground

In this section, we give the basic procedures of our proposed algorithm. The main steps include difference of Gaussian

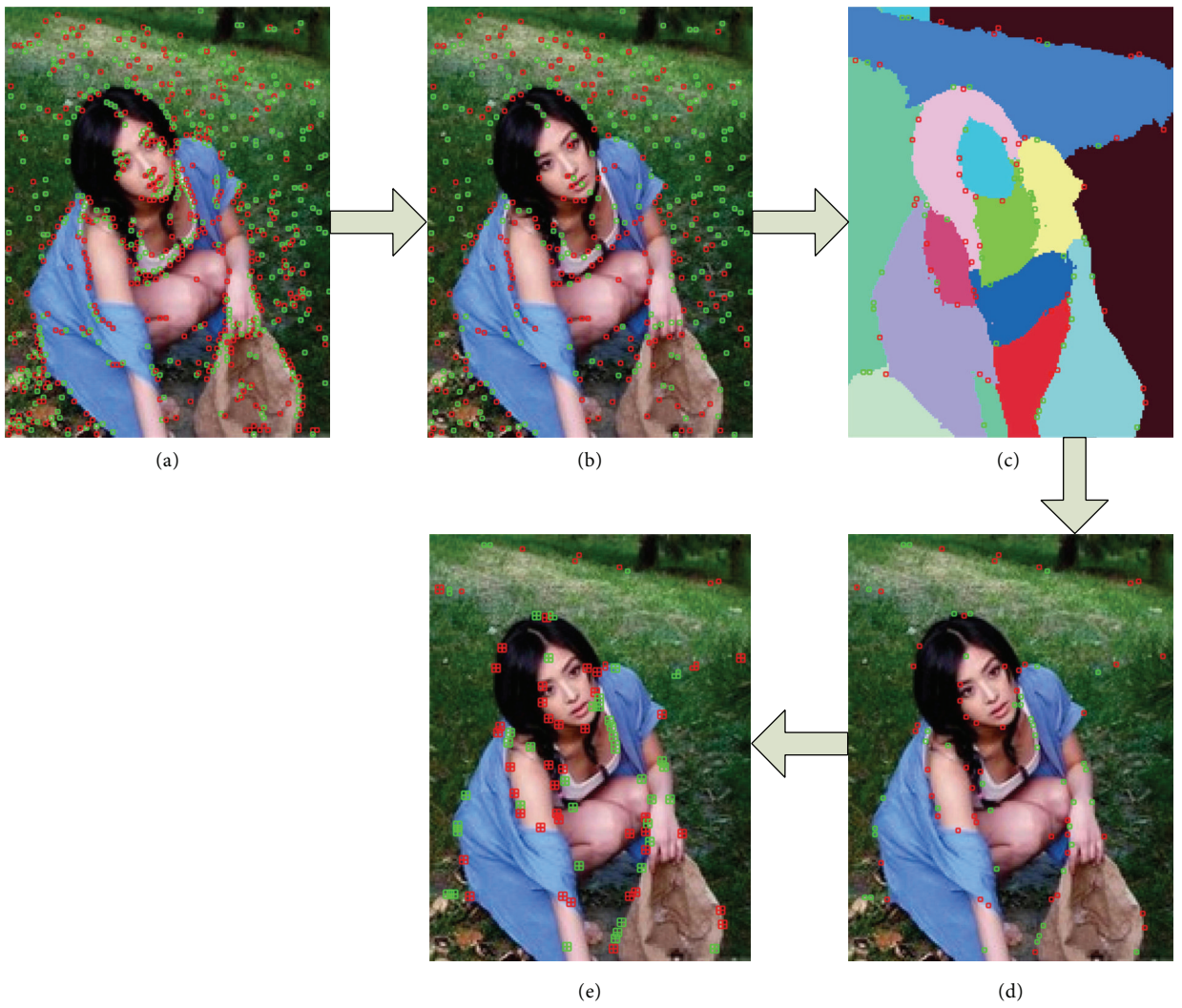


Figure 2: Basic procedure of the keypoints filter. (a) Initial result of difference of Gaussian (DoG). (b) Candidate points we get through the 5×5 filter. (c) Information of edges. (d) Information of edges on the original image. (e) Result of candidate points rebuild.

(DoG), generation of candidate points, keypoints filtering, and locating the foreground. The pseudo-codes are listed in Algorithm 1.

Remark 2. In Algorithm 1, *thresh* and *NcutNum* are two constants given by users or experts. Otherwise, they can be determined by a learning procedure. In this paper, *thresh* = 0.2 and *NcutNum* = 12 are determined by the observation value on many experimental results.

Remark 3. It is a big trouble problem for noisy images, because it is very difficult to detect the keypoints with DoG. In this case, we need to employ the operator to move the noise away at the beginning step of this algorithm.

4. Experiments

4.1. Images Data Set. We evaluate our extraction technique in two different data sets. The first data set consists of 27 images that are the most popular images used to extract foreground interactively. The second one is created by ourselves which contains 26 images. The latter one is more complicated than the front.

Some excellent results in the first data set are shown in Figure 3 and the other excellent results in the second data set are shown in Figure 4. Observations on Figures 3 and 4, our proposed algorithm, can extract the foreground beyond 95%. The original images in Figure 3 have rich color and texture in the foreground. When we use the DoG to check out the candidate keypoints, the number of keypoints in the foreground is larger than in the background. In this case, it is easy to extract the foreground. However, few of the images in the first data set are difficult to extract foreground automatically because there is confusion between foreground and background. For the images with outstanding target and complicated background, although more keypoints in the background are obtained, we can get effective keypoints by the keypoints filtered function. So we can extract the foreground with high performance, for example, the lady, the dog, and the postbox.

```
(1)  Input: image, thresh, NcutNum;
(2)  Loading image data: ImRGB = imread("image");
(3)  Image segmentation with Ncut: ImNut = Ncut(ImRGB, NcutNum);
(4)  Boundary computing: ImBoundary = FindBoundary(ImNcut));
(5)  Initial the keypoints set: Keypoint = zeros(size(ImRGB));
(6)  Computing the candidate points by difference of Gaussian (DoG):
(7)  for n = 1 : 3 do
(8)      Loading single layer data: ImLay = ImRGB(:, :, n);
(9)      Detecting the keypoints with DoG: ImlayDoG = DoG(ImLay);
(10)     Edge detection: ImEdge = edge(ImlayDog);
(11)     Finding Focus: ImF = FindFocus(ImEdge);
(12)     Rebuilding keypoints:
             LayerKeypoint(n) = Rebuild(ImEdge, ImF, ImBoundary));
(13) end for
(14) Combining the keypoints with the three layers:
                 ImKeypoint = union(LayerKeypoint);
(15) Locating the foreground regions by the numbers of keyponts:
(16) for n = 1 : NcutNum do
(17)     Initial the number of keypoints and the total pixels of the nth region:
                 TotalKeypoints = 0; NumKeypoint(n) = 0;
(18)     Calculating the total number of keypoints of each region:
(19)     for x = 1 : size(ImRGB, 1) do
(20)         for y = 1 : size(ImRGB, 2) do
(21)             if ImNcut(x, y) == n then
(22)                 if abs(ImKeypoint(x, y)) == 1 then
(23)                     TotalKeypoints = TotalKeypoints + 1;
(24)                     NumKeypoint(n) = NumKeypoint(n) + 1;
(25)                 end if
(26)             end if
(27)         end for
(28)     end for
(29)     Locating the foreground regions by the numbers of keyponts:
(30)     if NumKeypoint(n) ≤ thresh ∗ TotalKeypoints then
(31)         for (x, y) ∈ Region(n) do
(32)             ImRGB(x, y, :) = 0;;
(33)         end for
(34)     end if
(35) end for
(36) Output: Foreground, Foreground = ImRGB;
```

Algorithm 1: Extraction Algorithm.

In order to the effectiveness of our proposed method, two good algorithms are employed to extract the foreground from our 27 images. They are well-known in the saliency detection. One is regional contrast (RC) method and the other is two-stage scheme (TSS) method.

4.2. Regional Contrast Based Saliency Extraction Algorithm (RC). Cheng et al. [13] proposed a regional contrast based saliency extraction algorithm (RC), which simultaneously evaluates global contrast differences and spatial coherence. Their algorithm was simple, efficient and yields full resolution saliency maps.

At the beginning of the algorithm, an input image is mapped into a graphic, which is used to segment the image by GB [16]. The mapping operator is as follows:

$$\mathbf{I} \longrightarrow \mathbf{G} = (\mathbf{V}, \mathbf{E}, \mathbf{W}), \tag{17}$$

where **V** denotes the pixel in the image. **E** denotes the edge connected between adjacent pixels. **W** is the weight of the edge.

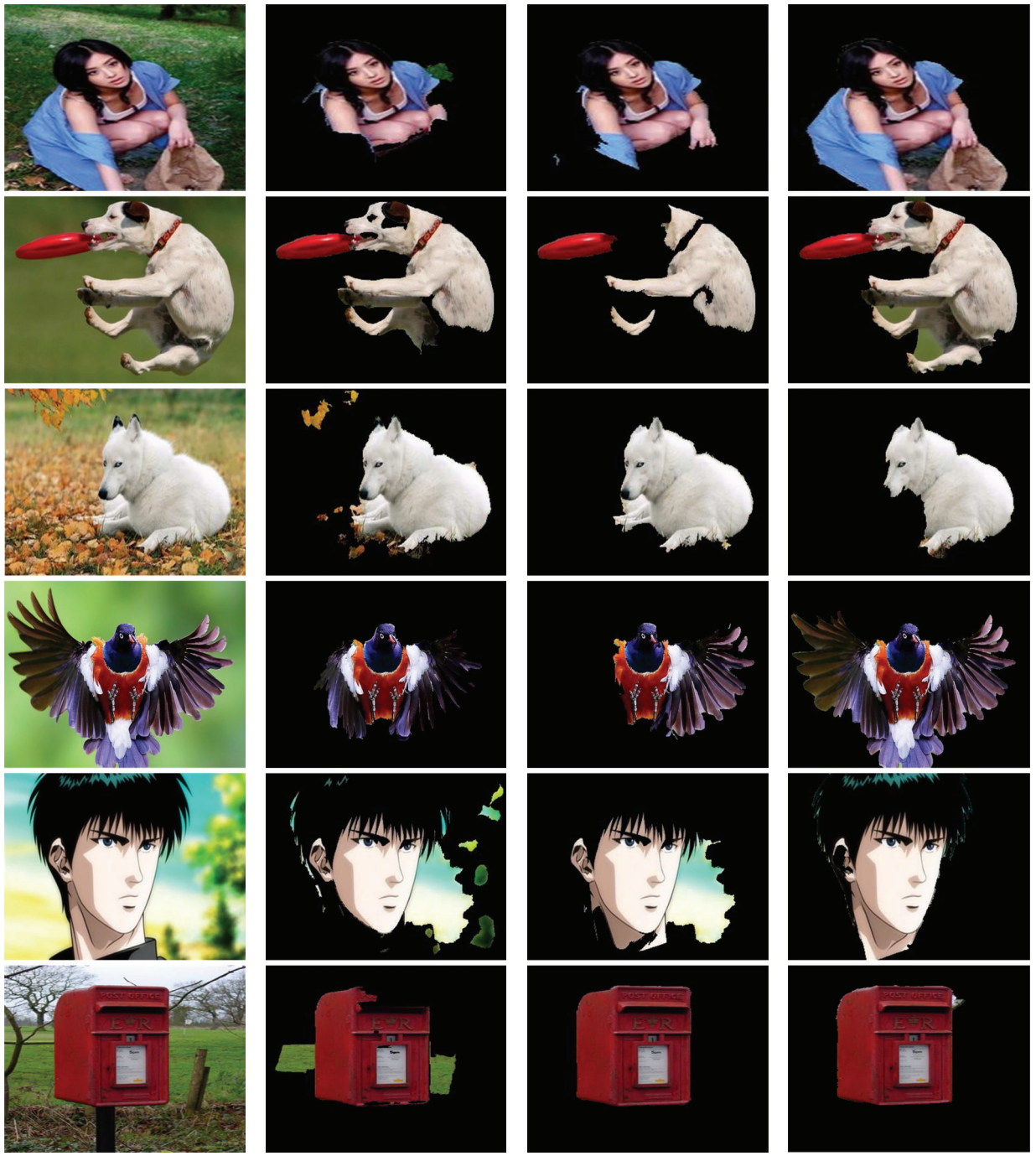
With the minimum spanning tree and the smallest weight value between two vertexes, the input image is segmented into several regions as follows:

$$\mathbf{I} = \{s_0, s_1, \ldots, s_K\}, \tag{18}$$

in which the number K means that the image is mapped into K parts.

For each segmented region, its salient values are calculated by comparing itself with the value of the other regions in Lab color space. In the same region, each pixel has the same salient value. The spatial distance information is also the important factor that influences salient value, so we consider it in the saliency detection. If one segmented region is close to

Figure 3: Results in the first data set. (a) Original images. (b) Foreground extracted with RC [13]. (c) Foreground extracted with TSS [14]. (d) Foreground extracted with FMDOG.



(a) (b) (c) (d)

Figure 4: Results in the second data set. (a) Original images. (b) Foreground extracted with RC [13]. (c) Foreground extracted with TSS [14]. (d) Foreground extracted with FMDOG.

current segmented region, the saliency influence of it is big. Otherwise, the influence is small.

The formula that adds spatial weights is as follows [13]:

$$\overline{S}(s_k) = \sum_{s_k \neq s_i} \exp\left(-\frac{D_s(s_k, s_i)}{\sigma_s^2}\right) w(s_i)\, \overline{D_s}(s_k, s_i), \quad (19)$$

where $w(s_i)$ is the weight value of region s_i, which is defined as the number of pixels in s_i. $D_s(s_k, s_i)$ is the distance between region s_k and s_i, which is defined as the Euclidean distance between their centers of gravity. σ^2 is used to control the strength of spatial weight. If the value σ^2 is big, the impact of the spatial weight is great, and the region far from the current region will have a stronger impact. Here, σ^2 is set as 0.4, and

the pixel coordinates are all normalized to $[0, 1]$. $\overline{D}_s(s_k, s_i)$ is the color distance metric between s_k and s_i.

The color distance formula between s_1 and s_2 is defined as follows [13]:

$$\overline{D}_s(s_1, s_2) = \sum_{i=1}^{n_1}\sum_{j=1}^{n_2} f(s_{1,i})\, f(s_{2,j})\, D(s_{1,i}, s_{2,j}), \quad (20)$$

in which $f(s_{k,i})$ is the frequency of ith color $s_{k,i}$ among all n_k colors in kth segmented region $s_k, k = \{1, 2\}$.

4.3. Two-Stage Scheme for Bottom-Up Saliency Detection (TSS). In 2013, Yang et al. [14] proposed a two-stage scheme (TSS) for bottom-up saliency detection using ranking with background and foreground queries. In this subsection, we introduce the basis procedures and the primal ideas of the TSS. The following sentences are referred from [14].

At first, an input image is represented as a close-loop graph with super-pixels as nodes (a graph $\mathbf{G} = (\mathbf{V}, \mathbf{E}, \mathbf{W})$ with super-pixels as nodes, $\mathbf{V}$ is a set of nodes and $\mathbf{E}$ is a set of undirected edges). The weight between two nodes is defined by

$$w_{ij} = e^{-\|c_i - c_j\|/2}, \quad i, j \in \mathbf{V}, \quad (21)$$

in which c_i and c_j denote the mean of the super-pixels corresponding to two nodes in the feature space and is a constant that controls the strength of the weight.

The graph-based ranking technique is employed to calculate the similarity of the image elements (pixels or regions) with foreground cues or background cues. The basic idea is that for a given node as a query, the remaining nodes are ranked based on their relevances to the given query. The goal is to learn a ranking function, which defines the relevance between unlabelled nodes and queries. The saliency of the image elements is defined based on their relevances to the given seeds or queries. The saliency map of the first stage is binary segmented (i.e., salient foreground and background) using an adaptive threshold, which facilitates selecting the nodes of the foreground salient objects as queries. The selected queries cover the salient object regions as much as possible (i.e., with high recall). The threshold is set as the mean saliency over the entire saliency map. Once the salient queries are given, an indicator vector $\mathbf{y}$ is formed to compute the ranking vector $\mathbf{f}$ using the equation

$$\mathbf{f} = \mathbf{A}\mathbf{y}. \quad (22)$$

In (22), $\mathbf{A}$ can be regarded as a learnt optimal affinity matrix and can be determined by the supervised manifold learning (details can be seen in Section 2.2 from [14]). As is carried out in the first stage, the ranking vector $\mathbf{f}$ is normalized between the range of 0 and 1 to form the final saliency map. It is calculated by

$$S_{fq}(i) = \bar{\mathbf{f}}(i), \quad i = 1, 2, \ldots, N, \quad (23)$$

where i indexes super-pixel node on graph and $\bar{\mathbf{f}}$ denotes the normalized vector.

4.4. Results. The results of these methods are displayed in Figures 3 and 4. It is obvious that better results can be obtained by our method in most cases.

5. Conclusion

In this paper, a novel approach for automatic foreground extraction is proposed. It is based on the difference of Gaussian (DoG). We create a keypoints filter to obtain the keypoints which are used to locate the foreground region in the image. Normalized cut (Ncut) is used to cut the image into different regions and find the information of the boundaries. This approach can be better applied to the image of which foreground is easy to identify by interactive foreground extraction. So our experiments are taken on the data set for interactive foreground extraction.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This research has been supported by the National Natural Science Foundations of China (Grants 61370174,61001200), Open Project Program of the State Key Lab of CAD&CG (Grant no. A1213), Zhejiang University, and Natural Science Foundation of Shanghai Province of China (11ZR1409600).

References

[1] J. Starek, A. Maki, S. Nobuhara, A. Hilton, and T. Matsuyama, "The multiple-camera 3-D production studio," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 19, no. 6, pp. 856–869, 2009.

[2] Q. Chen, D. Li, and C. Tang, "KNN matting," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 9, pp. 2175–2188, 2013.

[3] S. M. Yoon and G. J. Yoon, "Alpha matting using compressive sensing," *Electronics Letters*, vol. 48, no. 3, pp. 153–155, 2012.

[4] S. M. Prabhu and A. N. Rajagopalan, "Natural matting for degraded pictures," *IEEE Transactions on Image Processing*, vol. 20, no. 12, pp. 3647–3653, 2011.

[5] A. Rosenfeld and D. Weinshall, "Extracting foreground masks towards object recognition," in *Proceeding of the IEEE International Conference on Computer Vision (ICCV '11)*, pp. 1371–1378, Barcelona, Spain, November 2011.

[6] Q. Wang, F. Chen, W. Xu, and M. Yang, "Object tracking via partial least squares analysis," *IEEE Transactions on Image Processing*, vol. 21, no. 10, pp. 4454–4465, 2012.

[7] R. Shekhar and S. Chaudhuri, "Content-based image retrieval in presence of foreground disturbances," *IEE Proceedings: Vision, Image and Signal Processing*, vol. 153, no. 5, pp. 625–638, 2006.

[8] A. Levin, D. Lischinski, and Y. Weiss, "A closed-form solution to natural image matting," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 2, pp. 228–242, 2008.

[9] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[10] S. Kim, Y.-W. Tai, Y. N. Bok, H. Kim, and I. Kweon, "Two-phase approach for multi-view object extraction," in *Proceedings of the 18th IEEE International Conference on Image Processing (ICIP '11)*, pp. 2361–2364, Brussels, Belgium, September 2011.

[11] Y. Zhang, Y. Wang, Y. Li, and D. Weng, "Automatic foreground-background segmentation based on depth from coded aperture," in *Proceedings of the International Symposium on Photonics and Optoelectronics (SOPO '12)*, pp. 21–23, Shanghai, China, May 2012.

[12] C.-L. Hsieh and M.-S. Lee, "Autom atic trimap generation for dig ital image matting," in *Signal and Information Processing Association Annual Summit and Conference*, pp. 1–5, October 2013.

[13] M. Cheng, G. Zhang, N. J. Mitra, X. Huang, and S. Hu, "Global contrast based salient region detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '11)*, pp. 409–416, June 2011.

[14] C. Yang, L. Zhang, H. Lu, and X. Ruan, "Saliency detection via graph-based manifold ranking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '13)*, pp. 3166–3173, 2013.

[15] J. B. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 888–905, 2000.

[16] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph-based image segmentation," *International Journal of Computer Vision*, vol. 59, no. 2, pp. 167–181, 2004.