A Simple Algorithm of Superpixel Segmentation With Boundary Constraint

Yongxia Zhang, Xuemei Li, Xifeng Gao, and Caiming Zhang

***Abstract*— As one of the most popular image oversegmentations, superpixel has been commonly used as supporting regions for primitives to reduce computations in various computer vision tasks. In this paper, we propose a novel superpixel segmentation approach based on a distance function that is designed to balance among boundary adherence, intensity homogeneity, and compactness (COM) characteristics of the resulting superpixels. Given an expected number of superpixels, our method begins with initializing the superpixel seed positions to obtain the initial labels of pixels. Then, we optimize the superpixels iteratively based on the defined distance measurement. We update the positions and intensities of superpixel seeds based on the three-sigma rule. The experimental results demonstrate that our algorithm is more effective and accurate than previous superpixel methods and achieves a comparable tradeoff between superpixel COM and adherence to object boundaries.**

***Index Terms*— Image preprocessing, image segmentation, oversegmentation, superpixel.**

I. Introduction

THE concept of superpixel was first introduced by Ren and Malik [1] through the definition of the perceptually uniform regions using the normalized cut (NCuts) algorithm. Superpixels group pixels into perceptually meaningful atomic regions that can be used to replace the rigid structure of the pixel grid in images. In this way, image primitives and redundancy can be reduced greatly. Furthermore, for subsequent applications, it is more convenient and effective to compute image features based on regions than pixels. In general, superpixel segmentation algorithms are usually employed as preprocessing steps of many computer vision tasks to improve their performances, such as image segmentation [2], [3], object recognition [4], [5], classification [6], [7], and tracking [8].

An automatic superpixel segmentation algorithm has been researched intensively, such as NCut [1], mean shift [9], graph-based methods [10], Turbopixels [11], a simple liner iterative clustering method (SLIC) [12], optimization-based superpixels [13], [14], VCells [15], flooding-based superpixel generation approach (FCC) [16], and lazy random walk (LRW) [17], [18]. However, each superpixel method has its own advantages and drawbacks that may be suitable for a particular application. For instance, if superpixels are to be used to build a graph, an approach that can produce a more regular lattice, such as NCut [1], could be a better choice. While to segment an image, a method that generates supeprixels with better boundary adherence, such as SLIC [12], is preferred. Although it is difficult to develop an algorithm that is suitable for all applications, three characteristics, i.e., boundary adherence, uniform intensity of superpixels, and compactness (COM) [19], should be considered in designing a superpixel algorithm.

In this paper, we propose a new superpixel generation algorithm, where the image boundaries are especially considered to obtain superpixels with better boundary adherence. The main contributions of this paper are as follows.

1) A boundary term is defined based on pixels and their neighbors to make the edges of superpixels consistent with object boundaries in images. Based on this, a new distance measurement is proposed to control the homogeneity of intensity, COM, and boundary adherence of superpixels.
2) A new strategy is employed to update the positions and intensities of superpixel seeds. Only the highly reliable pixels within one superpixel region are used for updating the cluster center, which helps in relocating the superpixel seed accurately and smoothly improving the accuracy of segmentation.

II. Related Work

A large amount of the literature on image superpixel generation algorithms have been published during the last decades, and most of them are based on graphs, gradient ascent, or k-means. We will give a brief review of these methods from this angle in this section. For the convenience of complexity comparison, we assume that all of these methods process an image containing N pixels and generate K superpixels with n iterations if necessary.

A. Graph-Based Methods

Graph-based approaches of superpixel generation model an image as a graph taking pixels as nodes and the similarity between neighboring pixels as edge weight. Superpixels are

Manuscript received July 10, 2015; revised September 27, 2015, December 3, 2015, and January 23, 2016; accepted February 1, 2016. Date of publication March 8, 2016; date of current version June 30, 2017. This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grant 61332015, Grant 61373078, Grant 61472220, and Grant 61572292 and in part by the NSFC Guangdong Joint Fund under Grant U1201258. This paper was recommended by Associate Editor J. Lu. *(Corresponding author: Caiming Zhang.)*

Y. Zhang and X. Li are with the School of Computer Science and Technology, Shandong University, Jinan 250101, China (e-mail: sduzyx@gmail.com; xmli@sdu.edu.cn).

X. Gao is with the Department of Computer Science, University of Houston, Houston, TX 77004 USA (e-mail: xgao6@uh.edu).

C. Zhang is with the School of Computer Science and Technology, Shandong University, Jinan 250101, China, and also with the Laboratory of Digital Media Technology, Shandong University of Finance and Economics, Jinan 250014, China (e-mail: czhang@sdu.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSVT.2016.2539839

generated by minimizing an energy function defined on the graph.

NCut [20] uses contour and texture cues to partition a graph recursively, which consists of all pixels in the images. It defines a cost function on the partition boundaries and generates regular and visually pleasing superpixels by minimizing the function. However, the boundary adherence of superpixels is poor. And the computational cost is very expensive when the number of superpixels or the size of images increases greatly. To improve the efficiency, Felzenszwalb and Huttenlocher [10] proposed an alternative graph-based approach to generate superpixels. It clusters pixels as nodes of a graph and the edges of resulting superpixels align to image boundaries well in practice. This algorithm does not consider COM and superpixels are generated with highly irregular shapes and size.

Moore *et al.* [21] proposed a method to generate superpixels, which conforms to a grid using a graph cut method to find optimal paths or seams splitting images into small vertical or horizontal regions. In this algorithm, an image boundary map that influences the quality and speed of the output greatly should be precomputed. The complexity of this algorithm is $O(N^{(3/2)}\log N)$ without taking the time of this preprocessing step into account.

Veksler *et al.* [13] used a global graph cut optimization method to generate superpixels. Superpixels are obtained by stitching overlapping image patches together such that each pixel belongs to only one of the image patch regions. Liu *et al.* [22] formulated the superpixel generation problem as an objective function of entropy rate on a graph. The entropy rate can help to cluster pixels into compact and homogeneous regions and also favor superpixels to overlap with a single object on the perceptual boundaries.

Shen *et al.* [17] proposed a new superpixel segmentation method using LRW. They apply LRW on an image to obtain the probability of each pixel. Then, they get the initial superpixels according to the probabilities and commute time and optimize superpixels iteratively by an energy function defined on texture measurement and the commute time. It can preserve weak boundaries and segment complicated texture regions very well. However, the complexity of this method is about $O(nN^2)$, which is expensive.

B. Gradient Ascent Methods

Gradient ascent methods generate superpixels based on gradients. Beucher and Lantuéjoul [23] proposed the watershed approach based on an immersion analogy. Beucher and Meyer [24] used it to segment images and achieved a good performance. To improve the efficiency of watershed method, linear complexity algorithms have been proposed to compute it in [25] and [26]. The method in [26] performs a gradient ascent step by starting from local minima to produce watersheds, lines that separate catchment basins. This method is fast with complexity of $O(N \log N)$, while it does not consider COM. The resulting superpixels are often highly irregular in size and shapes and the boundary adherence is poor. To overcome these problems, Machairas *et al.* [27], [28] introduced a spatially regularized gradient algorithm to generate superpixels with regular shapes and achieved a tunable tradeoff between the superpixel regularity and the adherence to object boundaries. The complexity of this method is also linear with respect to the number of pixels in an image.

Comaniciu and Meer [9] applied mean shift, an iterative mode-seeking procedure for locating local maxima of a density function, to find modes in the color or intensity feature space and generate superpixels. However, this algorithm does not constrain COM and generates superpixels with irregular shapes. Its complexity being about $O(nN^2)$ makes it slow.

Similarly, the quick shift method [29] also uses a mode-seeking segmentation scheme. It initializes the segmentation by a median shift procedure. Then, each point in the feature space is moved to the nearest neighbor for increasing the Parzen density estimate. This algorithm can generate superpixels with good boundary adherence, while it is quite slow with the complexity of $O(ndN^2)$ (d is a small constant). And this method does not allow explicitly controlling the size and the number of superpixels.

Levinshtein *et al.* [11] proposed the Turbopixels algorithm to generate superpixels efficiently. This method uses a level-set-based geometric flow evolution process from the uniformly placed seeds in images. In this paper, superpixels are constrained to be compact and with regular size. However, the boundary adherence is poor and the complexity of this algorithm is high.

C. K-Means-Based Methods

K-means-based methods generate superpixels by starting from a rough initialization of cluster centers and refining the clusters till some convergence conditions are met.

Achanta *et al.* [12] presented the SLIC algorithm, adopting k-means clustering approach to generate superpixels. It produces superpixels with regular size and shapes. The complexity of SLIC is $O(nN)$.

Wang *et al.* [30] proposed a structure-sensitive technique to generate superpixels by exploiting the geodesic distance in 2011. Recently, Wang and Wang [15] used the VCells algorithm to generate superpixels with regular size and good boundary adherence. It initializes a partition of an image without consideration of image intensities and refines the segmentation by moving boundary pixels one by one to generate superpixels. It is fast and the main computational cost is $O(n\sqrt{K \cdot N})$.

A seed approach based on a simple hill-climbing optimization was proposed in [31]. It starts from an initialized superpixel partition and refines the initialization continuously by modifying its boundaries. This algorithm defines an energy function based on enforcing color similarity between the boundaries and superpixel color histogram. It is fast. However, the target number of superpixel is not under control and the shapes of superpixels are irregular.

The FCC algorithm was proposed in [16]. It offers COM and smoothness constraints and the edges of superpixels adhere to object boundaries in images well. The complexity of this method is about $O(N \log N)$.

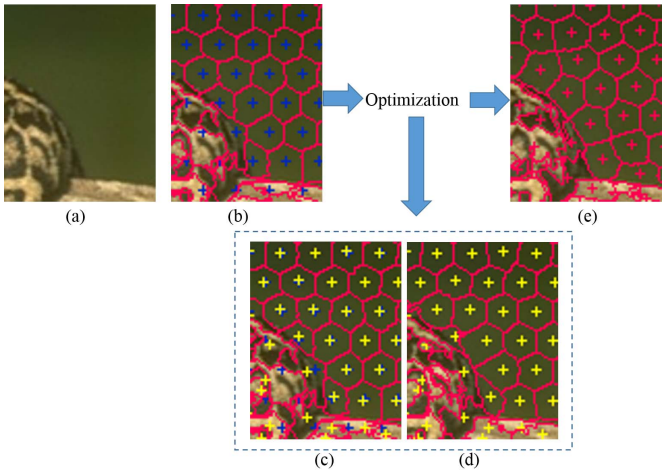


Fig. 1. Pipeline of our algorithm. (a) Input image. (b) Initial superpixels and seed points (blue pluses). (c) Seed relocation by our new strategy [yellow pluses show the relocated seeds from the original positions in (b)]. (d) Superpixel refinement by our method with updated center positions (yellow pluses). (e) Final superpixels. Note that steps in (c) and (d) (rectangle with dashed lines) are performed iteratively till the final superpixels are obtained.

III. Our Approach

The oversegmentation of an image I is to divide it into a collection of subregions, which are referred to as superpixels, namely, $I = \{S_l | l = 1, 2, \ldots, K\}$. Superpixel generation is to assign a unique label for each pixel of an image. A pixel x with label $L(x)$ belongs to the $L(x)$th superpixel region with the center $\tilde{C}_{L(x)}$.

Fig. 1 illustrates the pipeline of the proposed method. Given an image [Fig. 1(a)], we first place seeds with the expected number of superpixels to obtain an initial superpixel segmentation of the image, as shown in Fig. 1(b). To make the edges of superpixels adhere well to object boundaries in an image, we define a new distance measurement to optimize superpixels. We also introduce a new strategy to update the positions and intensities of superpixel seeds iteratively, as shown in Fig. 1(c) and (d). The final generated superpixels are shown in Fig. 1(e). Our major contributions lie in two aspects: 1) a new distance measurement to make the edges of superpixels align better with object boundaries in an image [Fig. 1(d)] and 2) a new strategy to update the positions and intensities of superpixel seeds [Fig. 1(c)].

In the following, Section III-A presents the new distance measurement first, then, Section III-B introduces the optimization strategy of superpixel seeds, and finally, Section III-C gives an overall description of our superpixel generation algorithm.

A. New Distance Measurement

For a pixel, the superpixel to which it belongs is determined by the shortest distance from the pixel to those seeds of its surrounding superpixels. In this paper, we design a new distance measurement between a pixel and a superpixel seed.

One of the most important properties of an ideal superpixel algorithm is that the edges of superpixels should align with the object boundaries in an image. What is more, it is desired for superpixels to have homogenous appearance and be compact. Accordingly, our distance measurement between pixel x and the lth superpixel seed consists of three terms, i.e., boundary adherence, homogeneous intensity, and COM, which is defined as

$$D(x, l) = w_b \times B(x, l) + w_i \times I(x, l) + \alpha \times w_c \times C(x, l). \tag{1}$$

Here, w_b, w_i, and w_c are weight functions that are designed to vary the focus on different characteristics during the optimization procedure. For instance, when the segmentation accuracy and COM are getting better, the weight function w_b for boundary adherence will be larger. They are defined as

$$\begin{aligned} w_b &= \frac{I(x, l) + C(x, l)}{A} \\ w_i &= \frac{B(x, l) + C(x, l)}{A} \\ w_c &= \frac{I(x, l) + B(x, l)}{A} \\ A &= 2(I(x, l) + B(x, l) + C(x, l)). \end{aligned}$$

α is a parameter to flexibly control the COM of the resulting superpixels. For each pixel, it can be labeled by selecting the superpixel of its neighboring one with the smallest D from the pixel to its seed. By performing such a labeling strategy for all the pixels sequentially, we can obtain a superpixel segmentation of an image.

1) Boundary Term: In general, image boundaries are detected by examining the gradient values of pixels. The larger the gradient of a pixel is, the more likely it will be on image boundaries. However, with the presence of noise, pixels that are not on objects boundaries could have large gradient values as well. In our algorithm, we introduce a new measurement based on neighbors of one pixel to compute its possibility of lying on object boundaries in an image, which is calculated by

$$w(x) = e^{\frac{\sum_{x_i} G_\delta |g(x_i) - g(x)|}{|R^\omega(x)|}}, \quad x_i \in R^\omega(x) \tag{2}$$

where $R^\omega(x)$ is the collection of all the pixels in a $\omega \times \omega$ sized window with pixel x as its center. $|R^\omega(x)|$ is the number of pixels in $R^\omega(x)$. In this paper, we set $\omega = 3$. $g(x)$ is the gradient of pixel x and G_δ is the Gaussian coefficient with size 3×3 as shown in Fig. 2.

1	4	1
4	20	4
1	4	1

Fig. 2. 3×3 Gaussian template.

If pixel x is in a flat region where the intensities of pixels x_i surrounding it are homogeneous, the value of $g(x_i)$ is small and quite similar to $g(x)$. As a result, $w(x)$ is small. If pixel x is in a region with many details, $g(x)$ and $g(x_i)$ are similar and large. From the computation in (2), we know that $w(x)$ is small. If pixel x is on image boundaries, the intensities of pixels surrounding it greatly vary along the

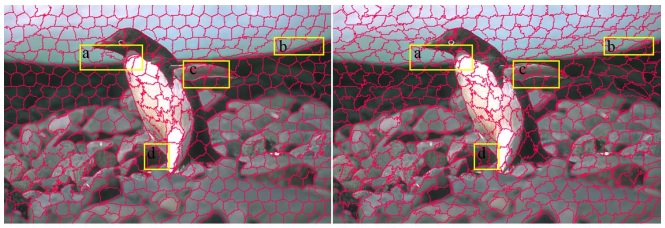


Fig. 3. (a) and (b) Results of our algorithm using (2) and $g(x)$ as $w(x)$ to evaluate the probabilities of pixels being on image boundaries, respectively.

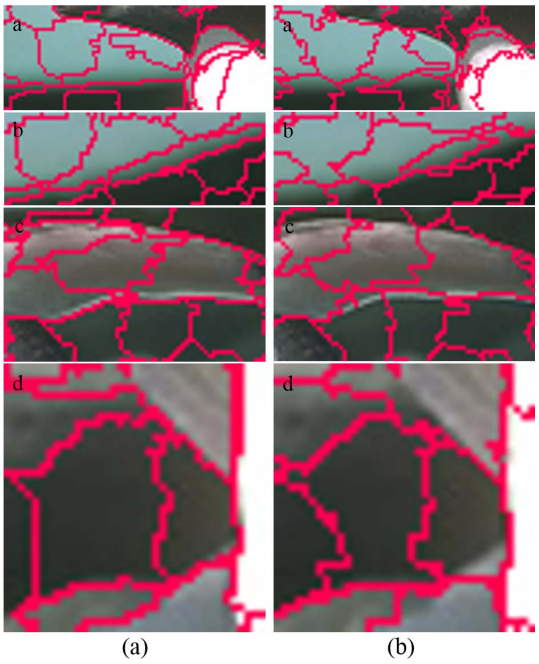


Fig. 4. Closeups of images in Fig. 3. (a) Closeups of the image in Fig. 3(a). (b) Closeups of the image in Fig. 3(b).

orthogonal direction of boundaries and the gradient values of pixels near the boundaries are large, while the others are small. Consequently, $w(x)$ is large based on the computation in (2). The larger the value of $w(x)$ is, the higher probability of the pixel x on object boundaries is. Fig. 3 shows the results of using (2) and $g(x)$ as $w(x)$ to evaluate the probabilities of pixels being on image boundaries, respectively. The closeups of these images are shown in Fig. 4(a) and (b) individually. It is clear that edges of superpixels in Fig. 3(a) using (2) as $w(x)$ better adhere to object boundaries in the image than the superpixels in Fig. 3(b), which are generated using $g(x)$ as $w(x)$. Based on this observation of the behavior of pixels on image boundaries, we design a boundary term to make the edges of superpixels align to object boundaries in images as follows:

$$B(x,l) = (1 - w(x))e^{\frac{n_{lx}}{\lambda_1 \delta^2}} + w(x)e^{\frac{w(x)}{\lambda_2 \zeta^2}} \tag{3}$$

where λ_1 and λ_2 are parameters, δ is the mathematical expectation of n_{lx}, and ζ is the average of $w(x)$ in an image. In a local window centered with pixel x, n_{lx} is the number of pixels whose label is not equal to l normalized by the total

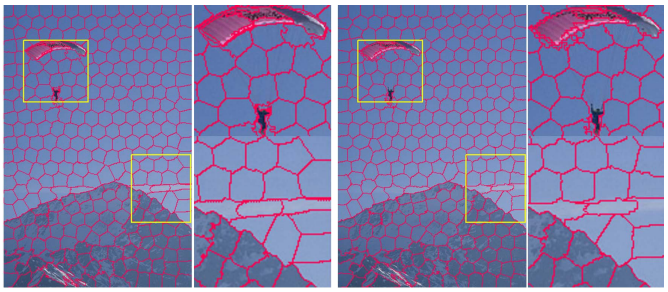


Fig. 5. Results of superpixels and the magnified regions (a) with boundary item and (b) without boundary term.

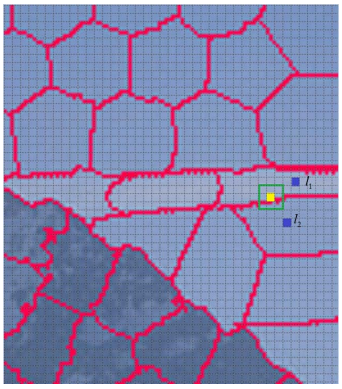


Fig. 6. Local region in the bottom yellow rectangle of Fig. 5(a). The seed positions of the l_1th and l_2th superpixels are marked in blue.

number of pixels in this window. n_{lx} is calculated as

$$n_{lx} = \frac{|R^{\omega}(x)'|}{|R^{\omega}(x)|}, \quad R^{\omega}(x)' = \{p | p \in R^{\omega}(x) \& L(p) \neq l\} \tag{4}$$

where $R^{\omega}(x)$ is the same with the one in (2). $|\bullet|$ is the number of elements in the collection $\bullet$. To some extent, the value of n_{lx} indicates the probability of whether pixel x is on the edges of superpixels.

Equation (3) consists of two terms, while the second one is a constant for each pixel. The reason is that whether a pixel is on object boundaries in images or lies on previous superpixel edges, it is likely to belong to superpixel edges in the next iteration. That means as long as one value between of $w(x)$ and n_{lx} is large, the value of $B(x,l)$ should be large. Accordingly, the formulation of $B(x,l)$ should be the sum of these two parts as (3). The boundary term is to ensure that edges of superpixels adhere to object boundaries in an image as good as possible.

Figs. 5 and 6 show the effect of boundary term. Fig. 5(a) presents the resulting superpixels generated by our proposed method with the distance measurement combined with the boundary term and more boundaries of the image are kept than that in superpixels shown in Fig. 5(b), which are generated without the boundary term. We explain the work of the boundary term with reference to Fig. 6, which is a local region in the bottom yellow rectangle in Fig. 5(a). For the yellow pixel x in Fig. 6, the l_1th superpixel and the l_2th superpixel are candidates it may belong to. On the one hand, the colors

of these two superpixels are almost the same, so these two intensity terms $I(x, l_1)$ and $I(x, l_2)$ are similar. On the other hand, the location of pixel x is on the border between these two superpixels whose seeds are marked in blue, which makes the COM terms $C(x, l_1)$ and $C(x, l_2)$ almost the same. As a result, whether pixel x belongs to the l_1th superpixel or the l_2th superpixel depends on the boundary term. Considering (3), for most of pixels surrounding pixel x belong to the l_1th superpixel, based on the computation in (4), $n_{l_1 x}$ is smaller than $n_{l_2 x}$, so $B(x, l_1) < B(x, l_2)$ and $D(x, l_1) < D(x, l_2)$. Accordingly, the pixel x is more likely to be labeled l_1.

2) Intensity Term: In our algorithm, we employ an intensity term, aiming at generating superpixels where all pixels contained within each of them have similar color intensities. The designed intensity term uses color distance computed in CIELAB space, which is defined as

$$d_c(x, l) = \sqrt{(L_x - L_l)^2 + (A_x - A_l)^2 + (B_x - B_l)^2} \quad (5)$$

where L_x, A_x, and B_x are the L, A, and B values of pixel x, respectively, and L_l, A_l, and B_l are the corresponding channels of the seed of the lth superpixel. The intensity term is formulated as

$$I(x, l) = e^{\frac{d_c(x,l)}{\lambda_3 \eta^2}} \quad (6)$$

where λ_3 is a parameter and η is the average of the standard variance of color distance for all superpixels. From (6), we can see that to minimize the intensity term, a pixel with color values as close as possible to those of the seed of a superpixel is preferred.

3) Compactness Term: Ideal superpixels should be compact. In our algorithm, we use the Euclidean distance to control the COM of superpixels. This term is defined as

$$C(x, l) = e^{\frac{\sqrt{(r(x)-r_l)^2+(c(x)-c_l)^2}}{\lambda_c \cdot s}} \quad (7)$$

where λ_c is a parameter and $(r(x), c(x))$ and (r_l, c_l) are coordinates of pixel x and the lth superpixel seed in the xy plane, respectively. To optimize the shape of a superpixel close to that of a hexagon, we divide the distance between pixel x and the seed a parameter $s = (2\sqrt{3}N/9K)^{1/2}$, which is the side length of a regular hexagon. The smaller the COM term is, the closer pixel x is to the seed of the lth superpixel and the more compact of the superpixel is.

B. Optimize Superpixel Seeds Based on Three-Sigma Rule

The majority of superpixel segmentation algorithms generates superpixels based on k-means strategy in an iterative way. For each iteration, the seed of each superpixel is updated by averaging all of the pixels belonging to it. However, this may lead to inhomogeneous intensity within a superpixel, especially when there are obvious differences among the pixels belonging to this superpixel. Fig. 7(d) and (e) is an example of such an issue after two consecutive iterations using all the pixels contained in a superpixel to update its seed. From Fig. 7(e), we can observe that the superpixel in the yellow rectangle contains more pixels of background in the result of latter iteration, while its boundary is far away from the object boundaries in the image. To tackle this issue, we use only reliable pixels that have the most homogeneous appearance with the superpixel seeds to optimize the positions and intensities of seeds. The position and intensity of the lth superpixel seed are updated by

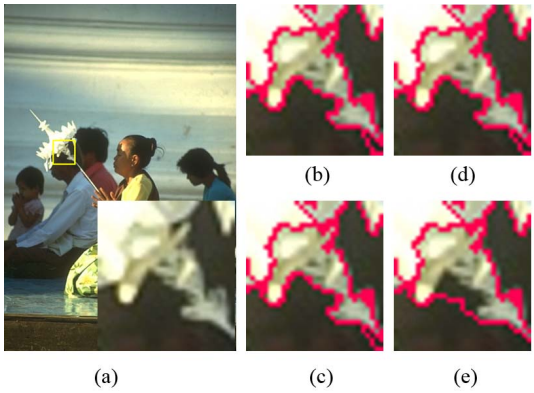


Fig. 7. Results of superpixels with different strategies to update seeds. (a) Original image. (b) and (c) Results with seeds updated by (8) based on our strategy. (d) and (e) Results of using all of pixels contained in a superpixel to update its seed.

$$\tilde{C}_{l,i} = \frac{\sum_{x \in \Phi_{l,i-1}} q(x)}{|\Phi_{l,i-1}|}, \quad i \geq 1$$
$$\Phi_{l,i-1} = \{x | L(x) = l \text{ and } |p(x) - \tilde{C}_{l,i-1}| \leq \xi_{l,i-1}\} \quad (8)$$

where $\tilde{C}_{l,i}$ and $\tilde{C}_{l,i-1}$ are the seeds of the lth superpixel in the ith and the $(i-1)$th iterations, respectively. $q(x)$ is the intensity vector (L_x, A_x, B_x) or position vector $(r(x), c(x))$ of pixel x. $p(x)$ is the intensity of pixel x. $\xi_{l,i-1}$ is the threshold that controls which pixels are used to update the seed in the $(i-1)$th iteration. The larger the value of $\xi_{l,i-1}$ is, the more pixels are employed.

In statistics, the three-sigma rule is that 99.73% of the data lies within three standard deviations of the mean in a normal distribution, which is one of the most widespread distribution in the nature. To make use of a sufficient number of pixels for updating superpixel seeds, we choose reliable pixels according to the three-sigma rule and set $\xi_{l,i-1} = 3\delta_{l,i-1}$, where $\delta_{l,i-1}$ is the standard deviation of colors of the lth superpixel at the $(i-1)$th iteration. Fig. 7(b) and (c) shows two examples of applying our strategy on updating superpixel seeds after two consecutive iterations, which produces comparable results compared with each iteration and better than those of continuous iterations using all of the pixels contained in a superpixel to update its seed, as shown in Fig. 7(d) and (e).

C. Superpixel Segmentation Algorithm

The scheme of our algorithm is similar to the framework of k-means based superpixel generation approaches, which performs initialization and refinement till some termination conditions are met. Taking the VCells algorithm as an example, our approach is distinct from them mainly in the following aspects: 1) our algorithm defines a distance function to balance among boundary adherence, intensity, and COM characteristics of

Algorithm 1 Proposed Superpixel Generation Algorithm

input : image I, expected number of superpixel K
output: the labels of pixels
1) Place K superpixel seeds in a hexagonal pattern
2) Initialize labels of all of the pixels by 9
3) Update the superpixel seeds using 8
4) Update the labels of all pixels by 1
5) Repeat 3) and 4) till the termination condition is reached
6) Enforce connectivity, according to the closest color guideline

superpixels and 2) our approach uses only reliable pixels belonging to one superpixel to update its seed, while for the other superpixel segmentation methods based on k-means, i.e., VCells, all the pixels within one superpixel are employed. As a summary, Algorithm 1 gives a complete description of our method.

1) Seeds Initialization: In our approach, we place K initial seeds in a hexagonal pattern, which is a user-defined parameter. Similar to other algorithms, to avoid centering a superpixel on an edge, we initialize the seeds with positions at the local lowest gradient locations compared with their neighborhoods.

2) Labeling Pixels: A superpixel segmentation method aims to assign a unique label to each pixel of the input image. In our algorithm, we label pixels based on the distance measurement between pixels and superpixel seeds as introduced in (1).

In the beginning, since no pixels are labeled except for the initialized seeds, n_{lx} in (3) cannot be calculated. Here, we use a simplified distance measurement defined in (9) to initialize the labels of pixels

$$D(x,l) = I(x,l) + C(x,l). \tag{9}$$

After the initialization step, we use (1) to measure the distance between pixels and superpixel seeds and update the label of each pixel with the superpixel whose seed has the shortest distance to it.

Similar to the SLIC algorithm, at the end of the labeling procedure, some orphaned pixels do not belong to the same connected superpixel. To enforce the connectivity, we relabel small disjoint components with the label of the neighboring superpixel that has the closest color to it.

3) Implementation:

a) Parameters: Five parameters should be decided in the proposed algorithm. According to experiments on 5% images in the test subset of Berkeley segmentation database (BSD) [32], we set $\lambda_1 \in [0.01, 0.08]$ and $\lambda_2 \in [0.1, 0.3]$ in (3). The smaller the value of λ_1 is, the larger proportion of n_{lx} is. It is the same as the value of λ_2. In this paper, we set $\lambda_1 = 0.05$ and $\lambda_2 = 0.2$. We set $\lambda_3 = 6$ in the intensity term (6). The results of superpixels with different values of λ_3 are shown in Fig. 8. Most of object boundaries are detected when $\lambda_3 = 6$ in Fig. 8(c). We set $\lambda_c = 0.57$ in (7). $\alpha \in [0, +\infty]$ is the parameter to balance COM and boundary adherence. The larger α is, the more compact superpixels are.

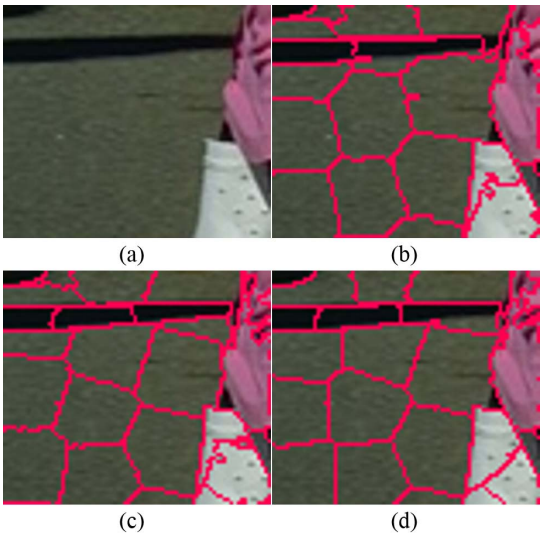


Fig. 8. Result of superpixels with different values of λ_3 in (6). (a) Original image. (b) $\lambda_3 = 4.5$. (c) $\lambda_3 = 6$. (d) $\lambda_3 = 7.5$.

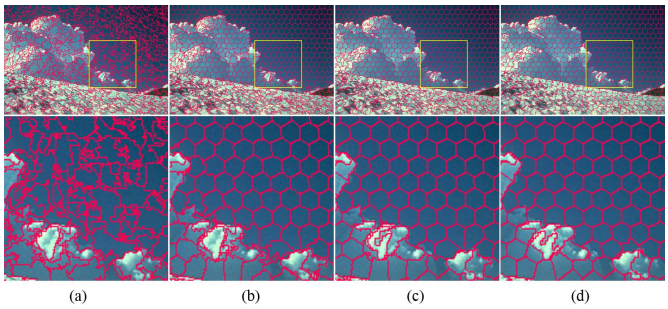


Fig. 9. Illustration of superpixels on BSD. All images are computed with $\alpha = 1$ unless otherwise specified. Images in the second row are zoomed-in views of the ones in the first row. (a) $\alpha = 0$. (b) $\alpha = 1$. (c) $\alpha = 10$. (d) $\alpha = 50$.

b) Termination condition: Based on the residual error E of location and intensity between the new superpixel seeds and the previous ones, for all the experimented test cases, our algorithm finishes within 20 iterations.

c) Complexity: If an image contains N pixels and the number of superpixels is K, then each superpixel should contain approximately N/K pixels. In the first step, we place seeds in a way that is independent of the image size, whose computational cost is negligible. The most time consuming part is the superpixels refinement step. For each pixel, three terms in (1) should be calculated. Both the computational cost of COM and intensity terms are $O(1)$. The complexity of the computing boundary term is $O(\omega^2)$, where $\omega \times \omega$ is the local window size decided by the user. Therefore, the computational cost of calculating the distance between one pixel and a seed at the refinement step is $O(1+1+\omega^2)$. In this paper, the local window is set to be as 3×3. By keeping ω as a constant, the complexity of the basic calculation is $O(1)$. Therefore, the computational cost for each iteration is $O(N)$. For n iterations, the computational cost is $O(n \cdot N)$. The probabilities of pixels on object boundaries in an image are calculated only once and the computational cost

Fig. 10. Results of image superpixels and magnified regions. Top to bottom: the results are processed by our method ($\alpha = 1$), Turbopixels, SLIC, VCells, FCC, and LRW, respectively.

is $O(N)$. Thus, the total complexity of the proposed algorithm is $O((n+1) \cdot N)$.

IV. Experimental Results and Discussion

In order to evaluate the proposed method, we apply it on BSD [32] and the Microsoft MSRC GrabCut dataset [33] and benchmarked and compare it with the state-of-the-art methods. BSD contains 500 images with approximately six human-annotated ground-truth segmentations for each image [34]. MSRC contains 30 images with ground-truth segmentation masks [35]. In order to obtain an objective and intuitive comparison, we compare our algorithm with other three well-known k-means-based algorithms: SLIC, FCC, and VCells. At the same time, Turbopixels and LRW as the representative of gradient-ascent-based approach and graph-based method are also being compared. In the comparison, the superpixel results of Turbopixels, SLIC, VCells, and LRW are generated using the implementations downloaded from their websites with parameter settings for the best performances.

A. Visual Comparison

Fig. 9 shows an image from BSD and its superpixels to demonstrate the influence of the parameter α (0, 1, 10, 50) for homogeneous (blue sky) and textured regions (clouds). As expected, when $\alpha = 0$, the superpixels are in a mess for there is no COM constraint, while when $\alpha \to +\infty$, superpixels tend toward regular grid of hexagonal cells. Both of them show good adherence to object boundaries in images, as shown in Fig. 9.

Figs. 10 and 11 show some images with superpixels generated by our method, Turbopixels, SLIC, VCells, FCC,

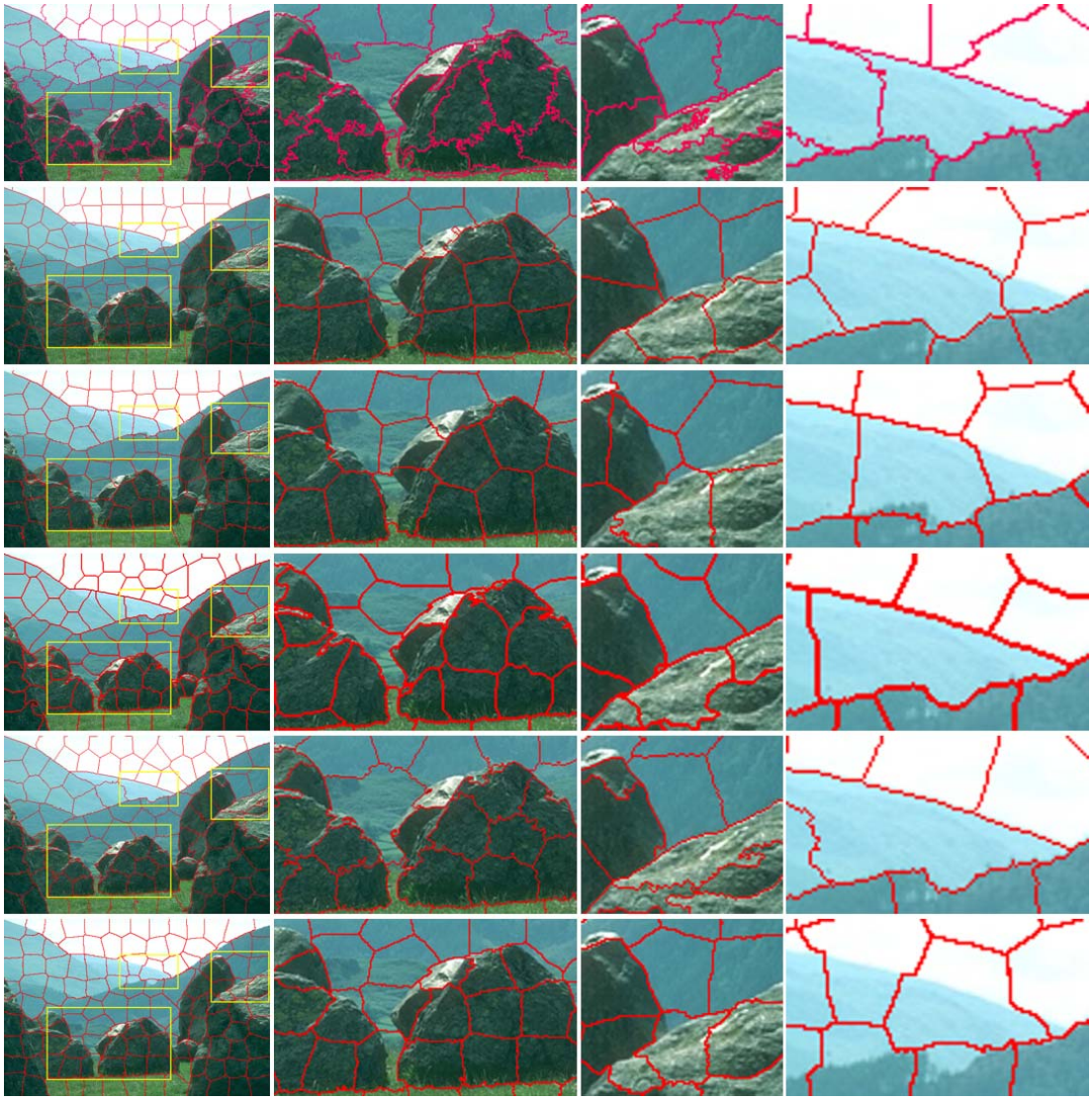


Fig. 11. Results of image superpixels and magnified regions. From top to bottom: the results are processed by our method ($\alpha = 1$), Turbopixels, SLIC, VCells, FCC, and LRW, respectively.

and LRW from top to bottom, respectively. The expected number of superpixels is 1100 and 100 in Figs. 10 and 11, respectively. The first column displays the whole image superpixels generated by each method and the other columns are their corresponding magnified regions. It is clear that the edges of our resulting superpixels better adhere to object boundaries in images especially in the yellow rectangular regions. For instance, the leg boundaries of this tiger in Fig. 10 are kept tightly in the resulting superpixels generated by our proposed method, while have a certain extent of contraction in the results of comparative algorithms, as shown in the second column of Fig. 10. The reason is that in our algorithm, image boundaries are considered additionally and the probabilities of pixels lying on image boundaries are estimated more accurately by (2).

Figs. 12 and 13 show various images from BSD, MSRC with their corresponding superpixels generated by our proposed algorithm, Turbopixels, SLIC, VCells, FCC, and LRW. It is clear that our algorithm has the best performance.

B. Quantitative Comparison

Boundary recall (BR), contour density (CD), undersegmentation error (USE), COM [36], and achievable segmentation accuracy (ASA) [17] are five commonly used metrics to evaluate the performance of superpixel generation algorithms. We adopt all these measures to compare our approach with those existing superpixel methods. The results for BR, CD, USE, and COM on BSD and MSRC are shown in Figs. 14 and 15, respectively.

1) BR: BR is an important metric for evaluating boundary adherence of superpixels. It measures what fraction of the ground-truth edges falls within at least two pixels of superpixel boundaries. A high BR indicates that very few true edges are missed. BR of each considered algorithm and our proposed method with different values of α are plotted against the number of superpixels in Figs. 14(a) and 15(a). All of these plots are produced by averaging the values of BR across all the images in each dataset. As shown in Figs. 14(a) and 15(a), the larger the value of α is, the lower the BR is. This is because the negative correlation between BR and COM.

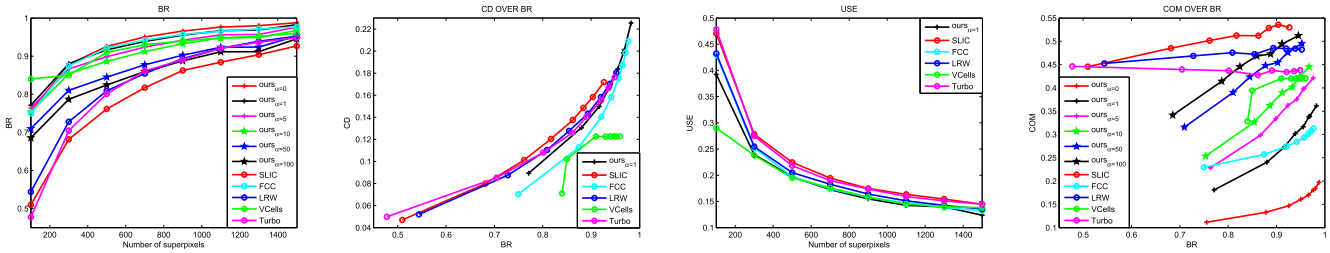
Fig. 12. Comparison results of image superpixels in BSD generated by our method and compared algorithms. Left to right: the results are processed by our method ($\alpha = 1$), Turbopixels, SLIC, VCells, FCC, and LRW. The number of superpixels in all the results is 300 except the first row with 100 superpixels.

In Fig. 14(a), when $\alpha = 1$, our algorithm achieves a comparable performance with the best one among the comparative methods. Given various values of α, Fig. 15(a) demonstrates that our algorithm achieves the best performance among the comparative methods. The reason is that we introduce a boundary term in our algorithm that may make the edges of superpixels consistent with object boundaries in an image. As a result, we compute CD, USE, and ASA with $\alpha = 1$ in this paper.

2) CD: To avoid overestimating artificially BR, we also evaluate boundary adherence by CD, which is defined as the number of pixels used to describe superpixels' edges normalized by the total number of pixels in an image. The average CDs on BSD and MSRC are plotted against BR in Figs. 14(b) and 15(b), respectively. For a constant BR value, the lower the CD is, the better result is. The VCells method has the best performance on the BSD, as shown in Fig. 14(b), and our approach has better performance than SLIC, Turbopixels, and LRW. Furthermore, our algorithm has the best performance on MSRC, as shown in Fig. 15(b).

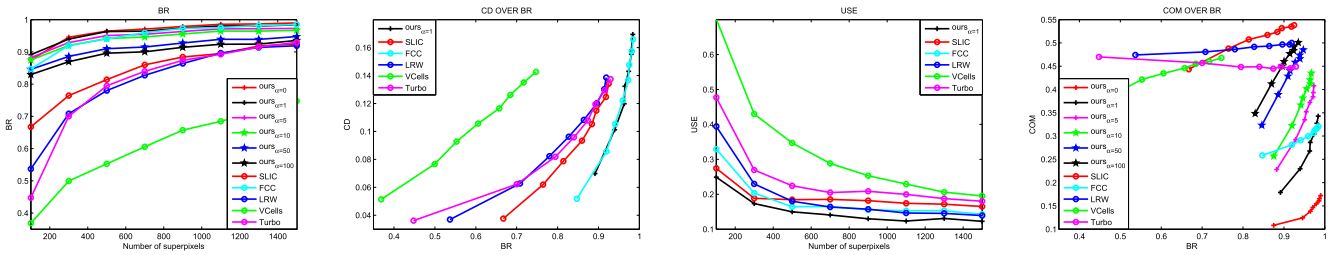
3) USE: USE essentially measures the error that an algorithm makes in segmenting an image with respect to a known ground truth (human segmented images in this case). This error

Fig. 13. Comparison results of image superpixels in MSRC by our method and compared algorithms. Left to right: the results are processed by our method ($\alpha = 1$), Turbopixels, SLIC, VCells, FCC, and LRW. Top to bottom: the number of superpixel is 500, 500, 300, and 100, respectively.



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

Fig. 14. Quantitative evaluation on the BSD benchmark. (a) BR. (b) CD over BR. (c) USE. (d) COM over BR.



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

Fig. 15. Quantitative evaluation on the MSRC dataset. (a) BR. (b) CD over BR. (c) USE. (d) COM over BR.

is computed in terms of the bleeding of the segment output by an algorithm when placed over ground-truth segments. This measure thus penalizes superpixels that do not tightly fit the ground-truth segment boundary. Given a ground-truth segmentation of an image into regions $\{G_i|, i = 1, 2, \ldots, M\}$ and a superpixel segmentation into superpixels $\{S_l|l = 1, 2, \ldots, K\}$,

the USE of the whole image is calculated by

$$\text{USE} = \frac{1}{N}\left[\sum_{i=1}^{M}\left(\sum_{\{S_l||S_l - G_i|>B\}} \text{Area}(S_l)\right) - N\right]$$

where N is the total number of pixels, $\text{Area}(S_l)$ is the area of the superpixel S_l, and B is the minimum area of overlapping. In our experiment, B is set to be 5% of $\text{Area}(S_l)$. A lower USE indicates that more objects in an image are recognized. We average the values of USE across all the images in BSD and MSRC and plot the curves of considered algorithms for increasing numbers of superpixels in Figs. 14(c) and 15(c). As shown in Fig. 14(c), our algorithm has a comparable performance with VCells and a better performance than the other methods. At the same time, our algorithm achieves the best performance on MSRC, as shown in Fig. 15(c).

4) COM: The larger the area of a shape for a given boundary length is, the better its COM is. That is to say, compact superpixels correspond to the ones whose shape tends to be a disc. Let A_{S_l} be the area and L_{S_l} be the perimeter of a shape, e.g., of a superpixel S_l. The isoperimetric quotient of this superpixel is

$$Q_{S_l} = \frac{4\pi A_{S_l}}{L_{S_l}^2}.$$

For a given segmentation, we compute the sum over the isoperimetric quotients of each superpixel normalized by the fraction of its size $|S_l|$ compared with the whole image. Let $\aleph$ be the set of all superpixels of a segmentation of the image I. The COM of the segmentation is

$$\text{COM} = \sum_{S_l \in \aleph} Q_{S_l} \cdot \frac{|S_l|}{|I|}.$$

The larger the value is, the more compact superpixels are. However, pursuing a high value of COM blindly is not desirable, because of the negative correlation between BR and COM [36]. The reason is that for a higher BR, the superpixel boundaries have to adjust more to the image content, which makes them less compact. In other words, the tradeoff between BR and COM is more considerable than higher COM unilateral.

Accordingly, the COM of superpixels generated by each considered algorithm, which is calculated by averaging the values of COM across all the images in BSD and MSRC, is plotted against BR in Figs. 14(d) and 15(d). To demonstrate the impact of the parameter α on the balance between boundary adherence and COM, Figs. 14(d) and 15(d) display the results of our method with different values of α. The larger α is, the higher COM is and the lower BR is. The results with $\alpha = 0$ have the poorest performance both on BSD and MSRC, as shown in Figs. 14(d) and 15(d), as there is no constraint on COM. The superpixels generated by our method with $\alpha = 1$ are more compact than FCC when the number of superpixel is large. Furthermore, the results with $\alpha = 100$ have a comparable performance with LRW and a better performance than FCC, VCells, and Turbopixels. The performance of our algorithm becomes better for the increasing value of α. Remarkably, the BR of our algorithm is always higher than those of LRW, SLIC, and Turbopixels both on MSRC and BSD, as shown in Figs. 14(a) and 15(a).

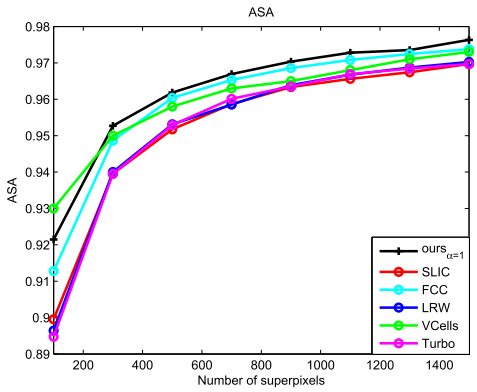


Fig. 16. ASA curves of our method and compared approaches on BSD.

5) ASA: This metric measures whether objects in images are correctly recognized. In other words, ASA computes the highest achievable accuracy by labeling each superpixel with the label of ground-truth segmentation that has the biggest overlap area. The metric is defined as

$$\text{ASA} = \frac{\sum_l \arg\max_i |S_l \cap G_i|}{\sum_i |G_i|}.$$

The higher ASA is, the more the objects recognized correctly in an image are. The ASA of each considered algorithm, which is calculated by averaging the values of ASA across all the images in BSD, is plotted against the number of superpixels in Fig. 16. Our algorithm outperforms the other methods. For only one object has been labeled in the ground-truth segmentation mask of each image in MSRC, the ASAs of all the algorithms are similar and approximated to 1. ASA plots against the number of superpixels on MSRC are not presented in this paper.

C. Computation Time

In this section, we compute the runtime of our algorithm, Turbopixels, SLIC, VCells, and FCC on a personal computer based on Intel Core i5 central processing units, operating at 3.10 GHz. The image size is kept as 321×481. We disregard LRW since it is computationally expensive compared with the other five methods. Fig. 17 shows the plots of runtime versus superpixel density of these five methods. Our algorithm is faster than Turbopixels, FCC, and VCells, while slower than SLIC. The reason is that in each iteration, on the one hand, we relabel all pixels in an image by calculating (1), which contains a boundary term based on the pixels in the neighborhood. On the other hand, our algorithm uses highly reliable pixels to update superpixel seeds based on the three-sigma rule. The threshold in (8) should be calculated for each superpixels in each iteration.

D. Limitation

Although superpixels generated by our proposed algorithm keep most of object boundaries in images and achieve a tunable tradeoff between boundary adherence and COM,

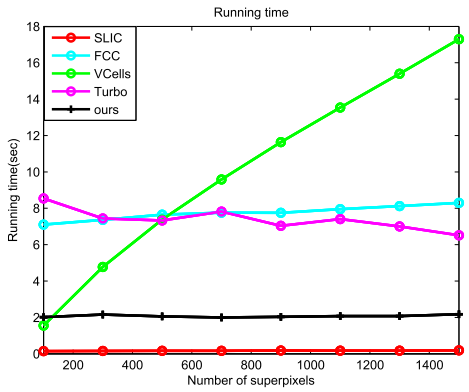


Fig. 17. Computation time comparison.

a number of limitations exist in the current approach. First, some weak boundaries in images are missing (e.g., the head boundary under the left ear and the beard of the tiger in the first image of Fig. 10). The reasons are that the colors of this object and its neighbors are similar and the boundary terms of pixels in this object are small. However, this may be addressed by a preprocessing step especially on image boundaries or combining pixel colors with gradients to evaluate the object boundaries in the future. Second, the efficiency of the current approach should be improved further.

V. Conclusion

To well align the computed superpixels to object boundaries in an image, we define a new measurement to calculate the probabilities of pixels lying on image boundaries. Based on this term, we define a new distance measurement to evaluate the similarities of pixels and superpixel seeds combining image intensity, COM, and boundary terms. At the same time, we use only reliable pixels based on three-sigma rule to update superpixel seeds. Our algorithm is simple, effective, and accurate. The proposed algorithm generates superpixels with good adherence to object boundaries in an image and achieves a tunable tradeoff between COM and boundary adherence.

References

[1] X. Ren and J. Malik, "Learning a classification model for segmentation," in *Proc. 9th IEEE Int. Conf. Comput.*, Oct. 2003, pp. 10–17.

[2] B. Peng, L. Zhang, and D. Zhang, "Automatic image segmentation by dynamic region merging," *IEEE Trans. Image Process.*, vol. 20, no. 12, pp. 3592–3605, Dec. 2011.

[3] J. Shen, Y. Du, and X. Li, "Interactive segmentation using constrained Laplacian optimization," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 24, no. 7, pp. 1088–1100, Jul. 2014.

[4] G. Mori, X. Ren, A. A. Efros, and J. Malik, "Recovering human body configurations: Combining segmentation and recognition," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, vol. 2. Jun./Jul. 2004, pp. II-326–II-333.

[5] Z. Li, X.-M. Wu, and S.-F. Chang, "Segmentation using superpixels: A bipartite graph partitioning approach," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2012, pp. 789–796.

[6] J. Cheng *et al.*, "Superpixel classification based optic disc and optic cup segmentation for glaucoma screening," *IEEE Trans. Med. Imag.*, vol. 32, no. 6, pp. 1019–1032, Jun. 2013.

[7] B. Liu, H. Hu, H. Wang, K. Wang, X. Liu, and W. Yu, "Superpixel-based classification with an adaptive number of classes for polarimetric SAR images," *IEEE Trans. Geosci. Remote Sens.*, vol. 51, no. 2, pp. 907–924, Feb. 2013.

[8] F. Yang, H. Lu, and M.-H. Yang, "Robust superpixel tracking," *IEEE Trans. Image Process.*, vol. 23, no. 4, pp. 1639–1651, Apr. 2014.

[9] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 5, pp. 603–619, May 2002.

[10] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph-based image segmentation," *Int. J. Comput. Vis.*, vol. 59, no. 2, pp. 167–181, Sep. 2004.

[11] A. Levinshtein, A. Stere, K. N. Kutulakos, D. J. Fleet, S. J. Dickinson, and K. Siddiqi, "TurboPixels: Fast superpixels using geometric flows," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 12, pp. 2290–2297, Dec. 2009.

[12] 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.*, vol. 34, no. 11, pp. 2274–2282, Nov. 2012.

[13] O. Veksler, Y. Boykov, and P. Mehrani, "Superpixels and supervoxels in an energy optimization framework," in *European Conference on Computer Vision*. Crete, Greece: Springer, 2010, pp. 211–224.

[14] J. Peng, J. Shen, A. Yao, and X. Li, "Superpixel optimization using higher-order energy," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 5, pp. 917–927, May. 2016.

[15] J. Wang and X. Wang, "VCells: Simple and efficient superpixels using edge-weighted centroidal Voronoi tessellations," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 6, pp. 1241–1247, Jun. 2012.

[16] X. Pan, Y. Zhou, C. Zhang, and Q. Liu, "Flooding based superpixels generation with color, compactness and smoothness constraints," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Oct. 2014, pp. 4432–4436.

[17] J. Shen, Y. Du, W. Wang, and X. Li, "Lazy random walks for superpixel segmentation," *IEEE Trans. Image Process.*, vol. 23, no. 4, pp. 1451–1462, Apr. 2014.

[18] Y. Liang, J. Shen, X. Dong, H. Sun, and X. Li, "Video supervoxels using partially absorbing random walks," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 5, pp. 928–938, May. 2016.

[19] J. Lu, H. Yang, D. Min, and M. N. Do, "PatchMatch filter: Efficient edge-aware filtering meets randomized search for fast correspondence field estimation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2013, pp. 1854–1861.

[20] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 888–905, Aug. 2000.

[21] A. P. Moore, S. J. D. Prince, J. Warrell, U. Mohammed, and G. Jones, "Superpixel lattices," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2008, pp. 1–8.

[22] M.-Y. Liu, O. Tuzel, S. Ramalingam, and R. Chellappa, "Entropy rate superpixel segmentation," in *Proc. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2011, pp. 2097–2104.

[23] S. Beucher and C. Lantuéjoul, "Use of watersheds in contour detection," in *Proc. Int. Workshop Image Process., Real-Time Edge Motion Detection*, 1979, pp. 391–396.

[24] S. Beucher and F. Meyer, "The morphological approach to segmentation: The watershed transformation. Mathematical morphology in image processing," *Opt. Eng.*, vol. 34, pp. 433–481, 1993.

[25] F. Meyer, "Un algorithme optimal pour la ligne de partage deseaux," *Congr. Reconnaissance Formes Intell. Artif.*, vol. 2, pp. 847–857, Nov. 1991.

[26] L. Vincent and P. Soille, "Watersheds in digital spaces: An efficient algorithm based on immersion simulations," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 13, no. 6, pp. 583–598, Jun. 1991.

[27] V. Machairas, M. Faessel, D. Cárdenas-Peña, T. Chabardes, T. Walter, and E. Decencière, "Waterpixels," *IEEE Trans. Image Process.*, vol. 24, no. 11, pp. 3707–3716, Nov. 2015.

[28] V. Machairas, E. Decencière, and T. Walter, "Spatial repulsion between markers improves watershed performance," in *Mathematical Morphology and Its Applications to Signal and Image Processing*. Reykjavik, Iceland: Springer, 2015, pp. 194–202.

[29] A. Vedaldi and S. Soatto, "Quick shift and kernel methods for mode seeking," in *Computer Vision*. Marseille, France: Springer, 2008, pp. 705–718.

[30] P. Wang, G. Zeng, R. Gan, J. Wang, and H. Zha, "Structure-sensitive superpixels via geodesic distance," *Int. J. Comput. Vis.*, vol. 103, no. 1, pp. 1–21, 2013.

[31] M. Van den Bergh, X. Boix, G. Roig, B. de Capitani, and L. Van Gool, "SEEDS: Superpixels extracted via energy-driven sampling," in *Computer Vision*. Florence, Italy: Springer, 2012, pp. 13–26.

[32] (2007). *The Berkeley Segmentation Dataset and Benchmark*. [Online]. Available: http://www.cs.berkeley.edu/projects/vision/grouping/segbench/

[33] (2004). *GrabCut*. [Online]. Available: http://research.microsoft.com/en-us/um/cambridge/projects/visionimagevideoediting/segmentation/grabcut.htm
[34] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *Proc. 8th IEEE Int. Conf. Comput. Vis. (ICCV)*, vol. 2. Jul. 2001, pp. 416–423.
[35] C. Rother, V. Kolmogorov, and A. Blake, "'GrabCut': Interactive foreground extraction using iterated graph cuts," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 309–314, 2004.
[36] A. Schick, M. Fischer, and R. Stiefelhagen, "Measuring and evaluating the compactness of superpixels," in *Proc. 21st Int. Conf. Pattern Recognit. (ICPR)*, 2012, pp. 930–934.

Yongxia Zhang received the B.S. degree from the Qilu University of Technology, Jinan, China, in 2011. She is currently pursuing the Ph.D. degree with the School of Computer Science and Technology, Shandong University, Jinan.

Her current research interests include computer vision, image processing, and machine learning, in particular, the problem of image and video denoising and image segmentation.

Xuemei Li received the master's and Ph.D. degrees from Shandong University, Jinan, China, in 2004 and 2010, respectively.

She is currently an Associate Professor with the School of Computer Science and Technology, Shandong University, where she is also a member of the Geometric Design and Information Visualization Laboratory. Her current research interests include geometric modeling, computer aided geometric design, medical image processing, and information visualization.

Xifeng Gao received the B.S. and M.S. degrees in computer science from Shandong University, Jinan, China, in 2008 and 2011, respectively. He is currently pursuing the Ph.D. degree in computer science with the University of Houston, Houston, TX, USA.

His current research interests include computer graphics, geometry processing, medical imaging, and multimedia security.

Caiming Zhang received the B.S. and M.E. degrees from Shandong University, Jinan, China, in 1982 and 1984, respectively, and the D.Eng. degree from the Tokyo Institute of Technology, Tokyo, Japan, in 1994, all in computer science.

He held a visiting position with the University of Kentucky, Lexington, KY, USA, from 1997 to 2000. He is currently a Professor and Doctoral Supervisor with the School of Computer Science and Technology, Shandong University. His current research interests include CAGD, computer graphics, information visualization, and medical image processing.