The Visual Computer (2021) 37:1061–1074
https://doi.org/10.1007/s00371-020-01852-2

ORIGINAL ARTICLE

Simple and fast image superpixels generation with color and boundary probability

Yongxia Zhang[1,2] · Qiang Guo[1,2] · Caiming Zhang[3]

Published online: 13 May 2020
© Springer-Verlag GmbH Germany, part of Springer Nature 2020

Abstract
As one of the most popular preprocessing steps in computer vision fields, superpixel generation algorithm has been extensively studied in recent years. Researchers have to find a way to produce superpixels with both accuracy and computationally efficiency. Inspired by the real-time superpixel segmentation method using density-based spatial clustering of applications with noise (DBSCAN), we propose a two-stage, non-iterative superpixel segmentation approach. In the first stage, we produce the initial regions. To make the superpixels attach to most object boundaries well, we define an adaptive parameter based on the boundary probability map in the distance measurement. At the same time, we adopt the averaging colors of region to represent the cluster center feature. In the second stage, we merge small regions to produce superpixels. To make them have uniform sizes, we take the initial region size into consideration and define a new distance measurement between the two neighboring regions. In the whole framework, we process all the pixels only once. We test the proposed method on the public data sets. The experimental results show that our proposed algorithm outperforms the most compared approaches with accuracy and has competitive speed with the real-time methods (e.g., DBSCAN).

Keywords Superpixel · Image segmentaion · Efficient · Real time · Accuracy

1 Introduction

The definition of superpixel was first put forward by Ren and Malik [34] in 2003. Superpixels group similar pixels into perceptually meaningful atomic regions and capture structure information in an image. Superpixels are usually used to replace the rigid pixel grids in an image to reduce the number of primitive in computer vision tasks, such as image segmentation [35,38], matching [10], semantic segmentation [36], object detection and tracking [19,21,44,49,52], classification [6,22] and image-based rendering [47], and improve their efficiency. These tasks require a high accuracy, i.e., good boundary adherence, for superpixel segmentation. Meanwhile, as a preprocessing step, a desired superpixel segmentation method should be computationally efficient.

In recent years, a great deal of progress has been achieved on this crucial topic. David Stutz et al. [42,43] have made a review of the existing superpixel generation algorithms. Most of them are categorized into cluster-based methods [1,5,13, 14,17,18,24,25,33,40,51], graph-based approaches [3,4,23, 26,27,30,32,39] and learning-based algorithms [15,16,45]. Among them, the cluster-based method is attractive for its simplicity and efficiency, such as the simple linear iterative clustering approach (SLIC) [1]. However, it often takes more time to obtain high-accuracy superpixels. For example, superpixels with contour adherence using linear path (SCALP) [13,14], the linear spectral clustering (LSC) [5] methods generate superpixels with higher boundary recall, while much slower than SLIC. Besides, some algorithms improve their efficiency by optimizing the algorithm framework. For example, simple non-iterative cluster (SNIC) [2], DBSCAN [40], adopt a non-iterative framework to generate

This work was supported by the National Natural Science Foundation of China (61802229, 61873145), NSFC Joint Fund with Zhejiang under Key Project (U1609218), Natural Science Foundation of Shandong Province (ZR2018BF007, ZR2017JL029), and Fostering Project of Dominant Discipline and Talent Team of Shandong Province Higher Education Institutions.

✉ Yongxia Zhang
sdu_zyx@hotmail.com

1 School of Computer Science and Technology, Shandong University of Finance and Economics, Jinan 250014, China

2 Digital Media Technology Key Lab of Shandong Province, Jinan 250014, China

3 School of Software, Shandong University, Jinan 250010, China

superpixels. Seeds [8] reduces the computational time by processing only boundary pixels in each iteration. Nevertheless, the accuracy of superpixels produced by these approaches is not enough good for computer vision tasks. As a preprocessing step of image processing applications, there is still a challenge to develop a real-time superpixel segmentation approach with accuracy, including good boundary adherence.

In this paper, we propose a new efficient approach, an improvement of DBSCAN [40], adopting a two-stage non-iterative framework to generate superpixels with more accuracy and competitive computational time. The main works of this paper relative to DBSCAN algorithm are as follows.

(1) In the first stage, we calculate the average color of pixels in a region once a pixel is included, which is more accurate to represent the region features than the chosen pixel seed color used in DBSCAN.

(2) We define an adaptive parameter for each pixel based on boundary probabilities and use it to calculate the distance measurement from a pixel to a seed in the first stage. As a result, compared with DBSCAN, where the parameter is fixed for all pixels, our superpixels boundaries adhere to image contours better.

(3) In the second stage, we merge small initial regions and define a new distance measurement between two adjacent regions combining with their sizes. In this case, the result superpixels have uniform sizes.

The remainder of this paper is organized as follows. We review the related works of superpixel generation in Sect. 2. In Sect. 3, we elaborate on the proposed approach in detail. Section 4 shows the experiments comparing several existing algorithms, and we summarize this work in Sect. 5.

2 Related works

In recent years, numerous methods have been presented to generate superpixels. Generally, each of them can be categorized into cluster-based methods, graph-based approaches or learning-based algorithms. In the following, we review these superpixel generation methods briefly.

Cluster-based approach produces superpixels using the clustering method. One of the most popular and applicable superpixel segmentation algorithms is SLIC [1]. SLIC adopts k-means clustering method iteratively in a combined five-dimensional color and spatial space. However, during the iterative process, some edges preserved at the beginning may be lost, which is a common drawback of iterative methods. Anyway, SLIC is fast and generates superpixels with regular size and shapes. Inspired by the merits of SLIC, so many algorithms are developed to improve the performance either on accuracy or efficiency. In order to get better contour adherence, Peng et al. [32] used k-means clustering method to get the initial superpixels and then refined them by optimizing an energy function, which includes a first-order data term, a second-order smoothness term and a higher order term. Giraud, Remi et al. [13,14] computed superpixels with contour adherence using linear path (SCALP). SCALP enhances the distance from a pixel to a superpixel center by considering the linear path to the barycenter. As a result, SCALP produces regular superpixels with accuracy. Zhang et al. [51] developed a new distance measurement with three items to restrain the boundary adherence. At the same time, they updated the superpixel seeds only by the most reliable pixels in each iteration. Hence, the update of superpixel center is stable and accurate, and the number of iterations is reduced. The resultant superpixels preserve more object boundaries and obtain a higher segmentation accuracy. Yet, the computational cost is higher than SLIC. Lee et al. [17] proposed a new contour-constrained superpixel algorithm (CCS) to improve the accuracy. It employs the holistically nested edge detection (HED) scheme [50] to obtain a boundary probability map at first. Based on this, CCS formulates an objective function with contour constraint and generates superpixels with good boundary adherence, whereas, even if ignoring the run time of HED, CCS is much slower than SLIC. Liu et al. [24,25] presented a manifold SLIC (MSLIC) method to compute content-sensitive superpixels. MSLIC projects the five-dimensional feature in $\mathbb{R}^5$ used by SLIC into a two-dimensional manifold $\mathcal{M} \subset \mathbb{R}^5$. Although MSLIC makes a quantitative improvement in superpixel quality, it also needs multiple iterations. Li and Chen et al. [5] proposed a linear spectral clustering (LSC) method. Each pixel is mapped to a point in a ten-dimensional feature space, in which weighted K-means is applied for segmentation. It captures perceptually important global image properties. LSC has been one of the most accurate superpixel segmentation approaches. All of the methods listed above are more accurate than SLIC. However, they are iterative and slower than SLIC.

From another perspective, the desired superpixel methods should be completed in limit time. Den et al. [8] proposed a fast method, Seeds, to generate superpixels via energy-driven sampling. It starts from an initialized partition and refines the initialization continuously by only modifying its boundary pixels. It is fast, whereas the shapes of superpixels are irregular. Choi, Kangsun and Oh, Kiwon [7,31] proposed two new methods to speed up SLIC. They use the Cauchy–Schwarz inequality to find a formula with simple computations, which is a lower bound of the original distance measurement used in SLIC [31]. In each iteration, the lower bound of the distance is calculated at first and compared with the minimum distance in the last iteration. If the lower bound is smaller, the original distance measurement should be calculated and the pixel may be relabeled; otherwise, the pixel label remains unchanged without any other computation. In this way, the computational cost of SLIC is reduced to some extent. Besides, Choi, Kangsun and Oh, Kiwon [7] used the spatial correlation between adjacent pixels to reduce the number of the candidate clusters. These two accelerating methods are slightly faster than SLIC and have similar performance with SLIC. Achanta et al. [2] designed a new scheme to implement SLIC by using a priority queue, named as simple non-iterative cluster (SNIC). The smaller the distance is, the higher the priority of the pixel is. SNIC generates superpixels by labeling the pixel with the shortest distance one by one without iteration. SNIC is faster than SLIC and has comparable performance. Shen et al. [40] proposed a real-time image superpixel segmentation approach, DBSCAN, which is one of the fastest state-of-the-art methods. It initials superpixels by clustering pixels only by using color restrictions and refines superpixels by merging small ones. All the pixels in an image are processed only once, which makes the algorithm speed up to 50 frames per second. Yet, the result superpixels miss some object boundaries and have irregular size.

Graph-based method represents an image as a graph by taking pixels as nodes. Shen et al. [39] proposed a method using lazy random walk (LRW) to get superpixels. The LRW algorithm calculates the probabilities of each pixel from the input image and utilizes them with the commute time to get the initial superpixels. Then, they introduce an energy function based on texture measurement and the commute time, to optimize the initial superpixels iteratively. It can preserve weak boundaries, and segment complicated texture regions very well. However, the LRW algorithm is very expensive with the computational cost $\mathrm{O}\left(nN^2\right)$. ($n$ is the number of iterations, and N is the number of pixels in an image.) Dong et al. [11] proposed a sub-Markov random walk (subRW) algorithm and provided a new idea for designing RW methods to solve the image segmentation problems. With label prior, the designed subRW approach outperforms previous RW methods for image segmentation. Wang et al. [48] proposed an adaptive nonlocal random walk (ANRW) to generate superpixels. They use ANRW to get the initial superpixels and merge small ones to get the resultant compact and regular superpixels. Liang et al. [20] applied partially absorbing random walks to get more accurate video supervoxels in regions with complex textures or weak boundaries. Liu et al. [23] presented another graph-based approach (ERS) that connects subgraphs by maximizing an objective function based on the entropy rate of a random walk. ERS is fast and able to preserve jagged object boundaries. However, the result superpixels are in irregular size and shapes. Shen et al. [41] proposed a novel minimization method to reduce any order of a general higher-order energy function, without auxiliary variables and produce accurate solutions. Image segmentation with entropy optimized by this method can achieve better results. To solve computer vision problems, Shen et al. [37] proposed a framework of maximizing quadratic submodular energy with a knapsack constraint approximately. The experimental results of image segmentation prove the effectiveness of the new energy function. Mooer et al. [30] proposed a method named as Lattices, to generate superpixels preserving the topology of a regular lattice. However, the performances on superpixel quality and efficiency rely heavily on the precomputed boundary map. Ban et al. [3] presented an alternative superpixel segmentation algorithm based on the Gaussian mixture model (GMMSP). It assumes that each superpixel corresponds to a Gaussian distribution and each pixel can be represented by a Gaussian mixture model. It proposes a new method to estimate the parameters of Gaussian distributions and assigns a unique label for pixels based on a posterior probability. GMMSP outperforms the state-of-the-art methods in accuracy and speed up by using OpenMP. Dong et al. [12] utilized the user scribbles to build the Gaussian mixture model (GMM) and proposed a novel interactive co-segmentation method using global and local energy optimization. Beucher and Meyer [4] segmented an image by using the watershed method and achieved a good performance. To speed up the watershed approach, some linear-complexity algorithms [29,46] have been proposed. The method [46] performs a gradient ascent step by starting from local minima to produce watersheds, lines, which separate catchment basins. This method is fast with the complexity of $O(N \log N)$. However, the result superpixels are often highly irregular in size and shapes, and the boundary adherence is relatively poor. To overcome these problems, Machairas et al. [26,27] introduced a spatially regularized gradient algorithm to generate superpixels and achieved a tunable trade-off between the superpixel regularity and the boundary adherence. In the meantime, the complexity of this method is also linear with the number of pixels in an image.

Learning-based method generates superpixels with machine learning, which is one of the hottest technologies currently. Most classical superpixel segmentation approaches rely on color and coordinate features that limit the performance on low-contrast regions and the applicability to infrared or medical images with a wide range of boundary changes. At present, learning-based methods use a neural network to get more image features and generate superpixels based on them by applying the traditional approach. Verelst et al. [45] used scattering networks to generate 81 features maps of size 64×64 per image channel and applied SLIC to get the superpixel map. Jampani et al. [16] proposed a new superpixel sampling networks (SSN). The input image is firstly passed onto a deep network that extracts features of each pixel, which are then used by differentiable SLIC to produce the superpixels.

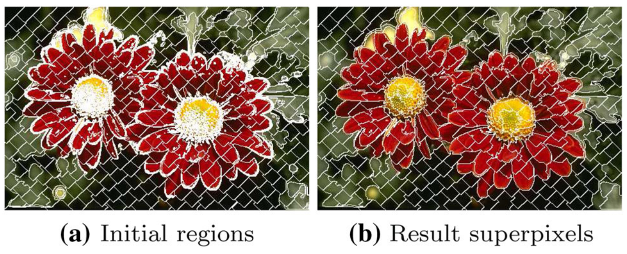

Fig. 1 Illustrating superpixel segmentation results in different stages. **a** The initial regions after the first stage and **b** the final result after the second stage

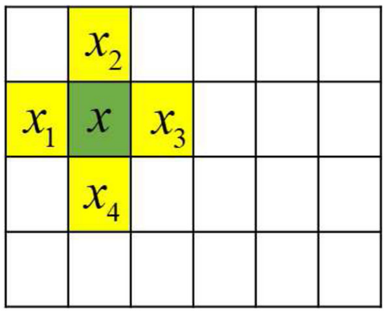

Fig. 2 Illustration of four-neighbor search. Each grid represents a pixel. Pixel x in green is chosen as the seed. The pixels in yellow are the candidate pixels to be labeled the same as x

3 The proposed algorithm

Most of the cluster-based superpixel generation algorithms are iterative. As a result, the computational cost is expensive. Furthermore, a post-processing step is usually adopted to enhance the connectivity. A real-time superpixel generation method with accuracy is always worth pursuing.

Inspired by the idea of DBSCAN [40], we adopt a simple two-stage framework without iteration to produce superpixels in a high speed with better boundary adherence than DBSCAN. In the first clustering stage, we apply neighbor search and group pixels with similar intensities as initial regions, as shown in Fig. 1a. In the second merging stage, we merge small regions to produce the result superpixels as shown in Fig. 1b.

In the whole procedure, all the pixels in an image are processed only once. Due to the neighbor search scheme in the first stage, the initial regions and the result superpixels are interconnected without any post-processing step. The proposed approach can generate superpixels faster than many existing algorithms (such as, SLIC [1], SNIC [2], Seeds [8]) and has comparable efficiency with the current real-time methods (e.g., DBSCAN). Our major works, which also are the differences from DBSCAN lie in three aspects: (1) We update the cluster centers dynamically in the first stage, rather than fixed on the chosen seeds in DBSCAN. In this way, the resultant regions may be more homogeneous and adhere to object boundaries better. Section. 3.1.1 gives more details. (2) In the clustering stage, to preserve more object boundaries accurately, we use a boundary probability map to define an adaptive parameter in the distance measurement between pixels, which are presented in Sect. 3.1.2. (3) In the merging stage, we define a new distance measurement between two neighboring regions combining their size as explained in Sect. 3.2. Merge small ones based on the closest distance and generate final superpixels with uniform size.

Notations: The superpixel segmentation of an image I containing N pixels is to divide it into a collection of subregions, namely $I = \{S_k | k = 1, 2, \cdots, K\}$ by assigning a unique label for each pixel in an image. $L(x) = k$ represents the pixel x belongs to the kth superpixel with the center $\tilde{C}_k$.

3.1 Clustering stage

In the clustering stage, we process the pixels in an image in a conventional order from left to right and from top to bottom and form initial regions one by one. All the pixels are processed only once. To achieve this, we define a matrix $\mathbf{L}$ to identify whether a pixel is labeled or not. Only unlabeled pixels are likely to be selected as a new seed and form new regions.

Supposing pixel x is chosen as a new seed and labeled as k, we produce the kth region by four-neighbor searching as shown in Fig. 2. Only those pixels in yellow around of x are computed. Compared with searching in eight-neighbor and circle regions, four-neighbor search can reduce computation efficiently. Meanwhile, it can ensure the connectivity in the generated regions.

If the similarity between the candidate pixel x_i, $(i = 1, 2, 3, 4)$ and the seed x is larger than the predefined threshold, set the pixel $x_i \in S_k$, label it as k and insert it into a waiting queue Q, in which four-neighborhood pixels of each element are candidates to form the region S_k. The growth process of the four neighborhoods is shown in Fig. 3.

Repeat this process to produce the initial region from seed x until the terminal condition is satisfied. To constrain the initial regions to have homogeneous intensities and uniform sizes, there are two stopping criteria, respectively. The first one is that the waiting queue Q is empty, which means there is no neighboring pixel which has similar intensities enough with the current seed. As a result, the initial regions are homogeneous. The second one is that the number of pixels in S_k becomes larger than expected, which controls the size of regions based on the anticipated number of superpixels.

Select the first unlabeled pixel in the conventional order as a new seed and generate regions by performing the steps

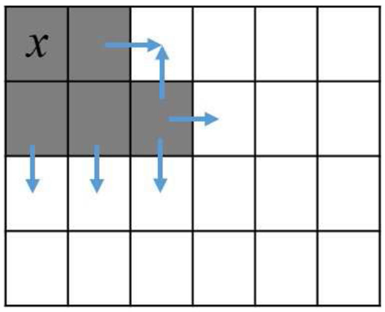

Fig. 3 Illustration of growth process in the clustering stage. Each grid represents a pixel. Pixel x is the seed that has expanded the region marked in dark gray. The pixels that blue arrows point to are the candidate pixels to be labeled the same as that of x

described above. Repeat the process till all pixels in the image are labeled and produce the initial regions.

We use the formula defined as Eq. 1 to measure the dissimilarities between pixel i and seed x conditioned by the pixel j.

$$D_c(i, j, x) = \beta \times d_c(i, j) + \alpha \times d_c(i, x), \tag{1}$$

where j represents the pixel in the waiting queue Q labeled with k as the same as x. The unlabeled pixel i which is adjacent to the pixel j in four neighborhoods is a candidate to be labeled as k. $d_c(i, j)$ represents the RGB color distance between the pixel i and j, which is defined as Eq. 2. $d_c(i, x)$ carries the similar meaning.

$$d_c(i, j) = \sqrt{\left(R_i - R_j\right)^2 + \left(G_i - G_j\right)^2 + \left(B_i - B_j\right)^2} \tag{2}$$

The first item in Eq. 1 measures the color difference between two adjacent pixels, which is sensitive to the local features and can detect little changes in an image. The second item $d_c(i, x)$ represents the color dissimilarity between pixel i and the seed x, which is benefit for the intensity consistency inside the regions. α and β are weight functions that are designed to vary the focus on different characteristics with the constraint $\alpha + \beta = 1$.

In order to maintain the intensity consistency better within the region and detect more image boundaries accurately, we use a new strategy to calculate $d_c(i, x)$, which will be illustrated in detail in Sect. 3.1.1. At the same time, we do effort on the formula definition of α and β as the function of image edge probability as described in Sect. 3.1.2.

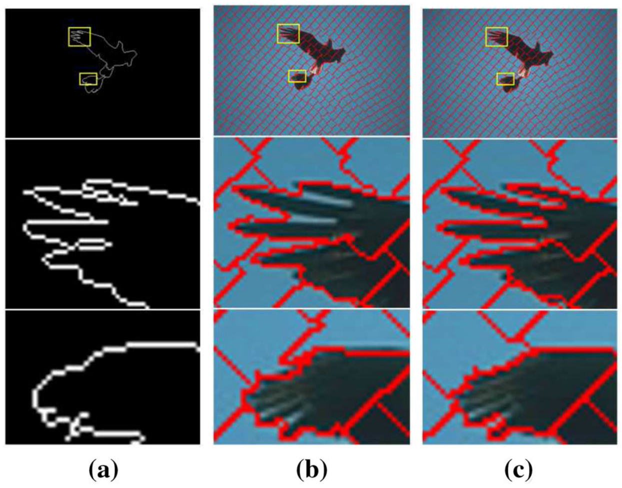

Fig. 4 Visual comparison of the superpixel segmentation results of different calculations of seeds color. The number of superpixel is about 500. **a** Ground truth; **b** seed is fixed as the pixel x; and **c** seed is updated by average all labeled pixels

3.1.1 Calculations of seeds color

Due to the strategy of selecting seeds in the conventional order, the seeds are usually located in the upper left margin of the regions. The first seed, the top-left pixel in the image, is a prime example. Besides, the seeds are most likely near the image boundaries, which is not a good representation of a region. To get homogeneous regions, we use the color of center $\tilde{C}_k$, the average color (R_k, G_k, B_k) of pixels in the region S_k to calculate $d_c\left(i, \tilde{C}_k\right)$ instead of the constant color of the selected seed x. Equation 1 can be reformulated as:

$$D_c\left(i, j, \tilde{C}_k\right) = \beta \times d_c(i, j) + \alpha \times d_c\left(i, \tilde{C}_k\right). \tag{3}$$

Most cluster-based superpixel generation approaches, such as SLIC [1], calculate the average color of superpixels in the last iteration and set it as the new seeds intensity in the next iteration. However, the seed color is fixed in each iteration. The biggest difference from this is that once a new pixel marked as k, we recalculate the color of $\tilde{C}_k$ immediately and use it to measure the distance to other candidate pixels.

To further illustrate the effectiveness of this work, Fig. 4 shows the superpixel results using different calculations of seeds color. It is clear that the superpixels generated by Eq. 3, as shown in Fig. 4c, adhere more object boundaries tightly and the intensities within the superpixels are more homogeneous.

3.1.2 The formula definition of α

In DBSCAN [40] method, α and β are set as constants, which means that all the pixels play the same role in preserving the image contour. However, for one pixel that lies on object boundaries, the color distance between it with the center of a region should be of more importance for detecting image edges, that is to say, α should be larger.

To maintain more edges, especially weak ones in an image, we set α based on the possibility of pixels on boundaries, in a simple and direct manner as Eq. 4.

$$\alpha = \frac{g(i) + g(j)}{2}, \tag{4}$$

where $g(i)$ and $g(j)$ state the possibilities that pixel i and j are located on the boundaries, respectively. If pixel i is on or near the edge of the image (the adjacent pixel j is on edges), whether to include it into the region is more relying on the dissimilarity between pixel x and C_k, namely the second item $d_c\left(i, \tilde{C}_k\right)$. That is to say, as long as $g(i)$ or $g(j)$ is large, α should be large. Thus, we define that the value of α is proportional to $g(i)$ and $g(j)$. In this way, the resultant regions may adhere to boundaries well.

Existing edge detection approaches, such as Sobel, Canny operators, methods in [9,50] can be used to calculate the boundary possibilities. In this paper, in order to show the validity of the definition of α, we adopt the simple Sobel operator to calculate the gradient values as the probabilities of pixels on object boundaries.

Figure 5 shows the result superpixels obtained by the same procedure only with different settings of α and β. The second row in Fig. 5 is partial enlargement of the yellow rectangle regions. Figure 5a shows the results obtained with $\alpha = 0.62, \beta = 0.38$, the same as the value used in DBSCAN [40]. α defined as Eq. 4 is used to get the results in Fig. 5b. It is clear that our proposed method can produce superpixels with detecting object boundaries, especially that the yellow arrow points to, more accurately.

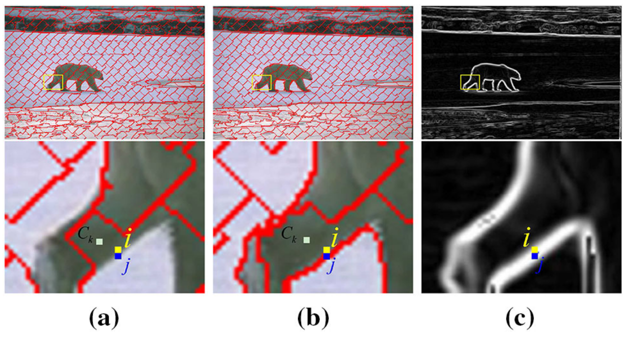

Fig. 5 Superpixel results by different α settings. The number of superpixel is about 500. **a** Result generated using constant α; **b** result generated by α defined as Eq. 4; **c** boundary probability map

To further illustrate the effect of alpha on the result superpixels, we take the pixels in yellow and blue in Fig. 5 as an example. Figure 5c shows the boundary probability map of the input images calculated by Sobel operator. For grouping pixels in the conventional order, pixels that have been labeled are located in the top left corner, where the intensities are homogeneous. $\tilde{C}_k$, the average color of the labeled pixels can be represented by the white pixel in Fig. 5a. The yellow pixel has been labeled as k, and the blue pixel is calculated based on Eq. 1 to decide whether to be labeled as k or not. As shown in Fig. 5c, the blue pixels are most likely on object boundaries. In this case, the probability of blue pixel labeled as k is more relative to the dissimilarity between $\tilde{C}_k$. That is to say, the second item $d_c\left(i, \tilde{C}_k\right)$ in Eq. 3 matters more. The value of α should be large. As a result, α defined in Eq. 4 is helpful to keep more object boundaries as shown in Fig. 5b.

3.2 Merging stage

In the clustering stage, we produce initial regions, preserving the most boundaries. Consequently, some regions may have much smaller sizes than expected and the number of regions is greater than K. To enforce the uniformity of the result superpixels size and control the number of superpixels, we merge small regions in the second stage considering the initial region size and the expected number of superpixels. We define a new distance measurement between two adjacent regions labeled as k_1 and k_2 as Eq. 5 to decide whether to merge them or not.

$$D(k_1, k_2) = s(k_1, k_2) \times d_c(k_1, k_2) + \lambda \times d_s(k_1, k_2), \tag{5}$$

where $d_c(k_1, k_2)$ indicates the color distance that evaluates the intensity homogeneity of the resultant superpixels. $d_s(k_1, k_2)$ represents the spatial distance defined as in Eq. 6, which has a strong impact on the regular shapes of superpixels. $s(k_1, k_2)$ is the term to constrain the uniform size of the result superpixels. λ is a parameter.

$$d_s(k_1, k_2) = \sqrt{\left(x_{k_1} - x_{k_2}\right)^2 + \left(y_{k_1} - y_{k_2}\right)^2}. \tag{6}$$

In order to make the superpixels have uniform sizes, the smaller neighbor regions compared to the expected size are, the more preferentially it should be merged into. So, we define $s(k_1, k_2)$ based on the initial region size as follows.

$$s(k_1, k_2) = \frac{\left|S_{k_1}\right| + \left|S_{k_2}\right|}{2\tilde{n}}, \tag{7}$$

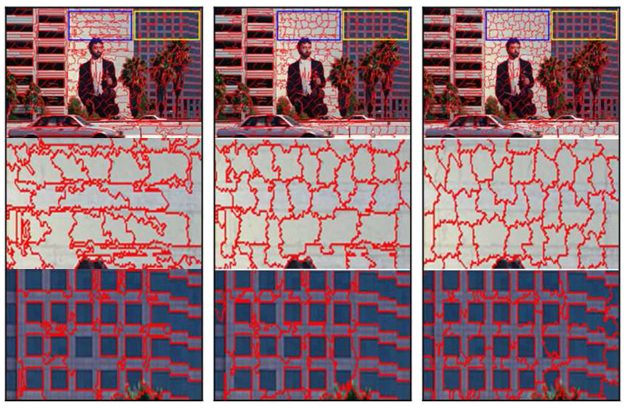

Fig. 6 Visual comparison of superpixel results with different merging schemes. **a** Superpixels merged according to $d_c(k_1, k_2)$; **b** superpixels merged without considering superpixel size ($s\,(k_1, k_2) = 1$ in Eq. 5); **c** superpixels merged based on Eqs. 5 and 7

where $|S_{k_1}|$ and $|S_{k_2}|$ represent the number of pixels in regions with the label k_1 and k_2, respectively. $\tilde{n} = \frac{N}{K}$ is the expected size of superpixels. S_{k_1} is the detected small region which should be merged. S_{k_2}, next to S_{k_1}, is the candidate one, with which combine S_{k_1}. The average color values and the coordinates of the centroids in the initial regions are used to measure the distance. We merge the small ones into its neighbors with the shortest distance. The smaller $|S_{k_2}|$ is, the value of $d_s\,(k_1, k_2)$ decreases generally. The possibility of merging S_{k_1} and S_{k_2} depends on the color similarity of them. In this way, the process is inclined to generate superpixels with more intensity homogeneous. Inversely, if $|S_{k_2}|$ is large, $s\,(k_1, k_2)$ and $d_s\,(k_1, k_2)$ increase and the superpixel S_{k_1} is not reliable to be combined with S_{k_2}. Superpixels with similar size are pleased to be formed. λ is a positive constant. The larger the value of λ is, the more regular shapes of superpixels have. In this paper, we set $\lambda = 2$.

To demonstrate that our rule in the merging stage is effective, Fig. 6 shows the superpixel results by different merging schemes. The blue and yellow rectangular areas are magnified and displayed in the second and third rows. Merging superpixels into their neighbors only with the closest color distance $d_c\,(k_1, k_2)$ may cause the resultant superpixels in a mess as shown in Fig. 6a. Comparatively speaking, considering the spatial distance in the merging stage may bring regular superpixels relatively, as shown in Fig. 6b, c. It is obvious that the superpixels in Fig. 6c obtained by merging regions according to the closest distance defined as Eq. 5 perform the best. They adhere to the most object boundaries and have uniform sizes as much as possible, especially in the yellow rectangular areas. The reasons may rely on the weighting sum of color and spatial distance between two neighboring regions and the consideration of region size in the definition of $s\,(k_1, k_2)$.

Besides, we make a test on 100 images from Berkeley segmentation database (BSD) [28] to prove the effectiveness of our scheme. Denote the method with fixed seed, constant α in Eq. 3, $s\,(k_1, k_2) = 1$ in Eq. 7 as f1, f2, f3, respectively. We use three common metrics, boundary recall (BR), under-segmentation error(USE) and achievable segmentation accuracy (ASA) to evaluate the performance of f1, f2, f3 and our proposed method. The higher the BR and ASA are, lower USE is, the superpixels are more accurate. More about these metrics are described in 4.2. At the same time, we calculate the average standard deviation of superpixels size to measure the uniformity of superpixel size. We plot them against the number of superpixels in Fig. 7. It is clear that our proposed approach has the best performance. In Fig. 7d, it is clear that the superpixels generated by our proposed method have more regular sizes with more than 20 percent increase than f3, which uses $s\,(k_1, k_2) = 1$ in Eq. 7.

As a summary, Algorithm 1 gives a complete description of the proposed method.

Algorithm 1: Proposed superpixel generation algorithm

```
Data: image $I$, expected number of superpixel $K$
Result: the labels of pixels
1  Set initial label as 0 for each pixel in image $I$;
2  Calculate the probabilities of pixels on object boundaries ;
3  while each pixel has a new label do
4      set $Q = \emptyset$;
5      find a new seed $x$ labeled as $k$ and insert $x$ into $Q$;
6      while $Q \neq \emptyset$ and the number of pixels labeled as k is smaller than $\frac{N}{K}$ do
7          for each pixel $j$ in the queue $Q$ do
8              set $L(j) = k$;
9              $|S_k| + +$;
10             compute the color of $\tilde{C}_k$ by averaging the color of pixels in $S_k$;
11             for each pixel $i$ around the pixel $j$ do
12                 compute $alpha$ using Equation. 4;
13                 compute the distance $D_c\left(i, j, \tilde{C}_k\right)$ using Equation. 3;
14                 if $D_c\left(i, j, \tilde{C}_k\right) \leq \varphi$ then
15                     insert $i$ into $Q$;
16                 end
17             end
18         end
19     end
20 end
21 Merging small regions according to the closest guideline based on Equation. 5;
```

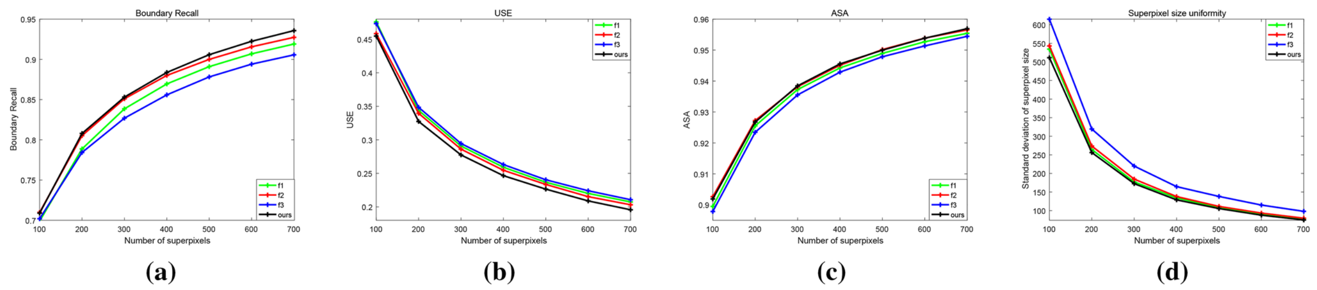

Fig. 7 Quantitative evaluation on BSD data Val set. f1 denotes the proposed method without calculation of seed color in Sect. 3.1.1; f2 represents the method with constant α in Eq. 3; f3 denotes the method with $s\,(k_1, k_2) = 1$ in Eq. 7. **a** Boundary recall; **b** under-segmentation error; **c** achievable segmentation accuracy; and **d** uniformity of superpixel size

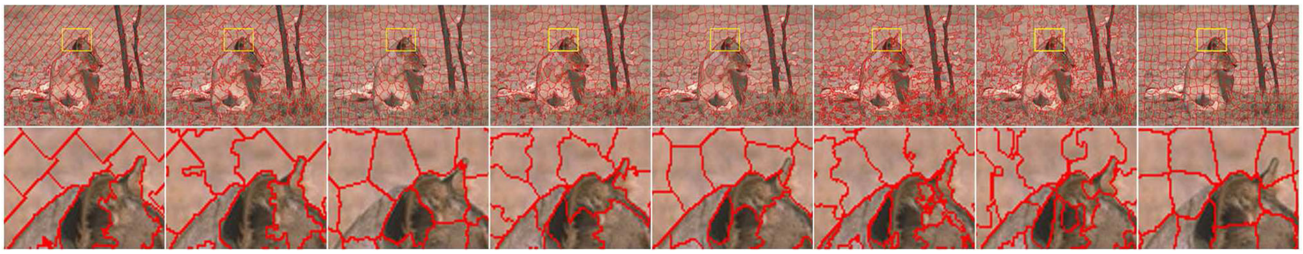

Fig. 8 Visual comparison of superpixel segmentation results by our proposed method, DBSCAN [40], SLIC [1], SNIC [2], SCALP [14], LSC [5], Seeds [8] and LRW [39] from left to right. The number of superpixel is about 300. Images are provided in high resolution for zoom-in examination

4 Results and analysis

In this section, we evaluate the performance of our proposed approach by comparing with several state-of-the-art methods, including DBSCAN [40], SLIC [1], SNIC [2], SCALP [14], LSC [5]. Seeds [8], as one of the fastest method, is used to make comparisons. We also compare with LRW [39], as a representative of graph-based algorithm. All the results of the compared methods are obtained by running the publicly available codes provided by the original authors with parameter settings recommended by them. All of the algorithms are tested on BSD [28] data set, which contains 500 images with approximately 6 human-annotated ground truth segmentation for each image.

4.1 Visual comparison

Figures 8 and 9 give the representative visual superpixel results generated by the proposed method, DBSCAN [40], SLIC [1], SNIC [2], SCALP [14], LSC [5], Seeds [8] and LRW [39] from left to right, respectively. Some local segmentation results are enlarged to facilitate close visual inspection and displayed at the bottom. It is clear that the superpixels generated by our proposed method as shown in the left columns in Figs. 8 and 9 are among the best performers on the adherence to object boundaries in images, especially in the yellow rectangular regions. For instance, the right ear boundaries of the lion in Fig. 8 are kept tightly in the result superpixels generated by our proposed method, while having a certain extent of contraction in the results of some compared algorithms as shown in the second row in Fig. 8. On the one hand, the reason is that the iterative methods such as SLIC, some edges may be lost in the iterations. On the other hand, in our algorithm, image boundaries are considered additionally in the clustering stage by calculating the color distance between two neighboring pixels. At the same time, we use a boundary probability map to define the function of α in Eq. 1 for detecting more object boundaries accurately.

Figure 10 shows various images from BSD with their corresponding superpixel results. It is clear that our algorithm performance is among the best ones.

4.2 Quantitative comparison

Superpixels segmentation usually serves as a fundamental process in the computer vision field. Thus, the accuracy, the ability to preserve object boundaries, is one of the most crucial merits of a superpixel generation algorithm. We adopt three commonly used metrics on evaluation boundary adherence for quantitative comparison, including boundary recall (BR), under-segmentation error (USE) and achievable segmentation accuracy (ASA). BR 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

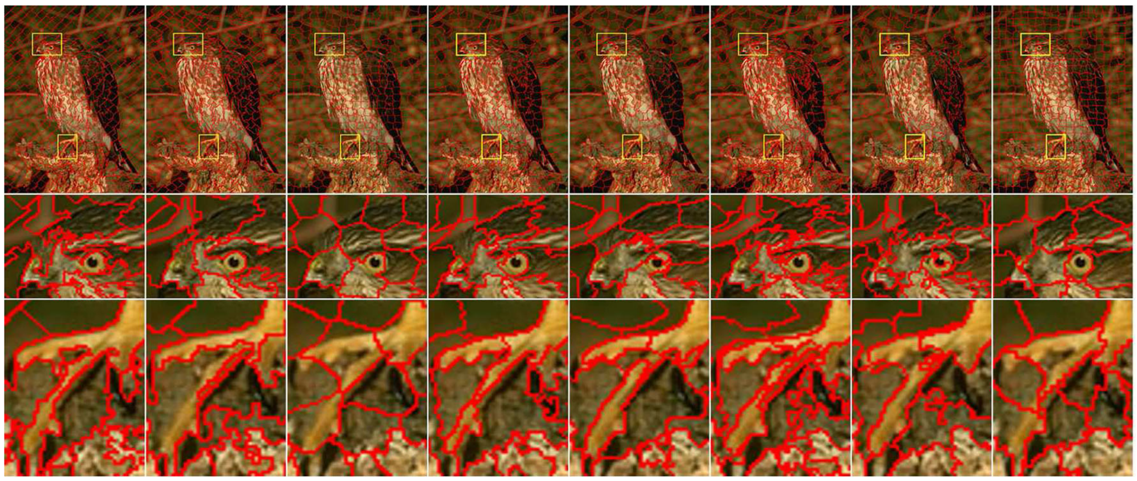

Fig. 9 Visual comparison of superpixel segmentation results by our proposed method, DBSCAN [40], SLIC [1], SNIC [2], SCALP [14], LSC [5], Seeds [8] and LRW [39] from left to right. The number of superpixel is about 500. Images are provided in high resolution for zoom-in examination

edges are missed. USE essentially measures the error that an algorithm makes in segmenting an image with respect to the known ground truth (human-segmented images in this case). This error is computed in terms of the 'bleeding' of the segment output by an algorithm when placed over the ground truth segments. This measure thus penalizes superpixels that do not tightly fit the ground truth segment boundaries. A low USE indicates that fewer superpixels straddle multiple objects. ASA 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 the ground truth segmentation that has the biggest overlap area. A higher ASA indicates that the superpixels match objects in the image better. We calculate all the metrics of each considered approach by averaging the values of them across all of the 500 images in BSD and plot them against the number of superpixels in Fig. 11.

As shown in Fig. 11a, it is obvious that the BR of our proposed method is larger than that of DBSCAN [40] from the lower superpixel densities to the higher one. The reason mainly relies on the boundary probability map which is used in Eq. 4 to define the distance measurement in Eq. 3. Meanwhile, our algorithm outperforms SLIC [1], SNIC [2], SCALP [14] and LRW [39] and achieves competitive performance with Seeds and becomes better with increase in superpixel number. It is clear that LSC [5] has the best performance on BR, USE and ASA, as shown in Fig. 11. The main reason is that LSC maps pixels to a ten-dimensional feature space and preserves global image structures by optimizing the cost function of the normalized cuts using K-means approach. It has the advantage of these two approaches.

SNIC and SCALP are slightly better than the proposed method on the metrics USE when the number of superpixels is less than 300. The metric ASA of SCALP is somewhat larger than the proposed method when the number of superpixels is less than 500. Nonetheless, the BR values of them are much smaller than the proposed approach. Furthermore, as the number of superpixels increases, the proposed algorithm outperforms them.

We also evaluate the uniformity of the result superpixels size. We plot the averages of the standard deviation of superpixel size of all the compared methods against the number of superpixels in Fig. 12. It is clear that, as the number of superpixel increases, our algorithm has comparable performance as SLIC and LRW. Furthermore, superpixels generated by the proposed method are always have more regular sizes than those obtained by LSC, SNIC, Seeds, SCALP and DBSCAN. The reason is that in the merging stage, we merge small superpixels into its neighbors with the closest distance considering the initial region sizes.

4.3 Analysis of computational complexity and comparison

For superpixel segmentation commonly serves as a preprocessing step in computer vision tasks, the computational efficiency matters a great deal in addition to accuracy. Thus, we analyze the complexity of the proposed method and compare the computation times with all the compared approaches.

Supposing an image I contains N pixels, and the expected number of superpixels is K. Then, each superpixel should contain approximately $\frac{N}{K}$ pixels. In the clustering stage, our algorithm searches the whole image in the conventional order to find the unlabeled pixels as new seeds and groups the unlabeled neighboring pixels to generate initial regions. Each of the pixels in the image is processed only once, and the computational cost is $\mathrm{O}(N)$. In the merging stage, initialized regions

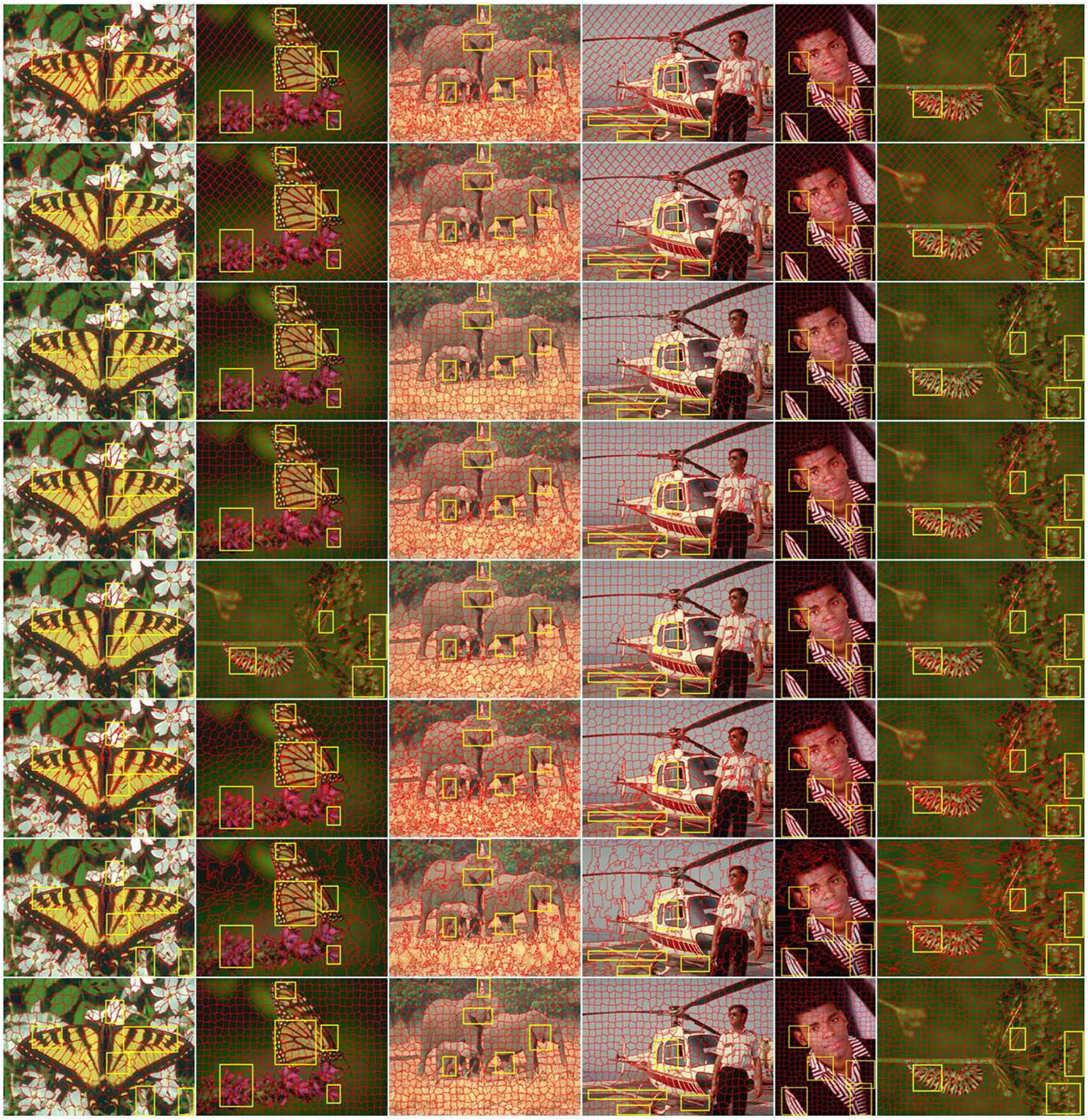

Fig. 10 Visual comparison of superpixel segmentation results by our proposed method, DBSCAN [40], SLIC [1], SNIC [2], SCALP [14], LSC [5], Seeds [8] and LRW [39] from up to down

in the first stage are processed. Supposing the number of initial regions is M, the second stage can be completed with the complexity of $\mathrm{O}(M)$. M is much less than N and has a lot to do with the thresholds in the cluster step. The smaller the threshold is, more object boundaries can be detected accurately and M increases. In the merging stage, only small regions should be merged, whose number is much smaller than M and N. That makes the merging stage can be finished with the complexity of $\mathrm{O}(1)$. Thus, the total computational cost of the proposed algorithm is $\mathrm{O}(N)$.

Figure 13 shows the plots of average run time for all the compared methods versus the number of superpixels. All the experiments are performed on a personal computer based on Intel(R) Core(TM) i7-6700 central processing units, operating at 3.10 GHz. The size of all tested images is kept as 321×481. The time consumption of LRW is consider-

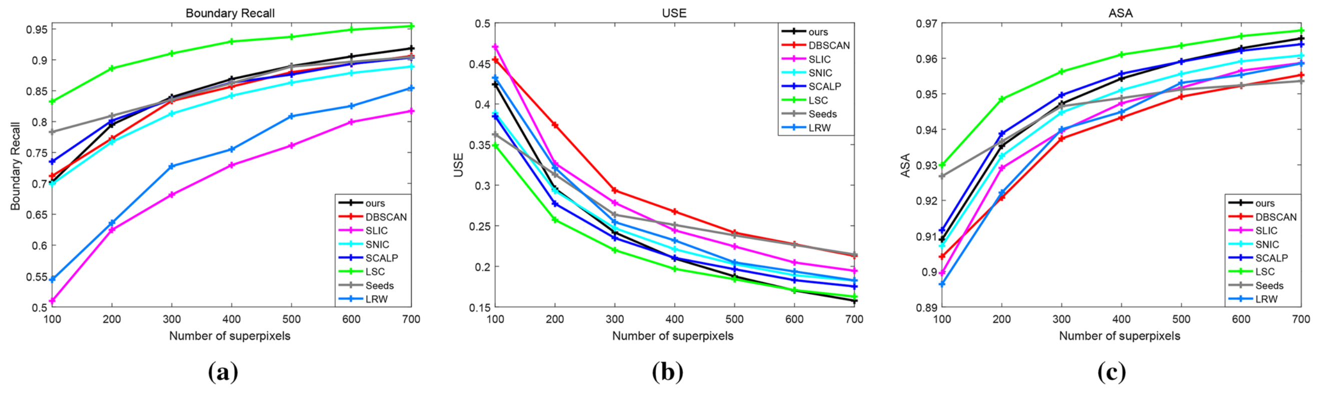

Fig. 11 Quantitative evaluation on BSD data set. **a** Boundary recall; **b** Under-segmentation error; **c** achievable segmentation accuracy

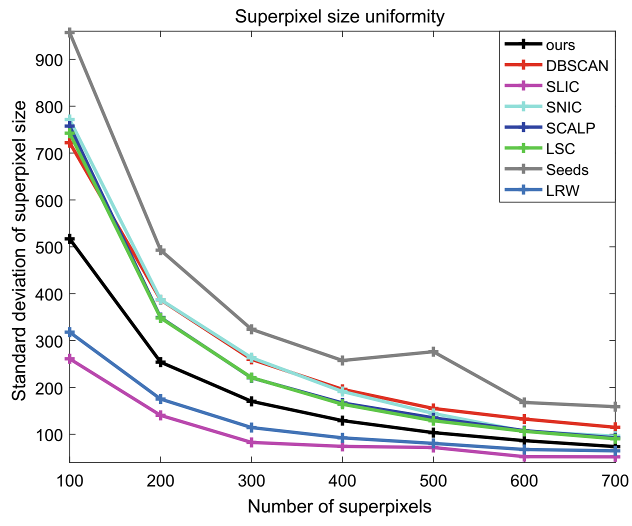

Fig. 12 Quantitative evaluation on superpixel size uniformity

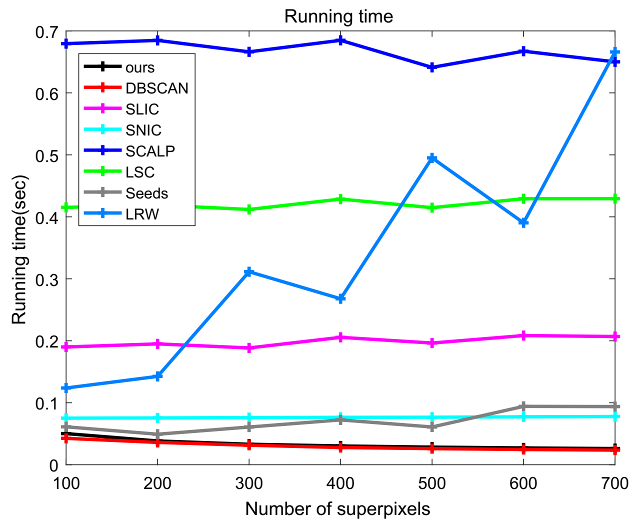

Fig. 13 Computation time comparison

ably higher than that of the other algorithms and divided by 100 for comparison. For one thing, it is obvious that our method and DBSCAN are relatively much faster than the other algorithms. The reason is that those slower methods, except SNIC, are iterative, while our proposed approach and DBSCAN process all pixels in an image only once. SNIC, another non-iterative approach, spends more times than DBSCAN and our proposed method. This is mainly due to the sort operation in priority queues, which is time-consuming. It is clear from Fig. 13 that our algorithm is slightly slower than DBSCAN. It may rely on the consideration of the boundary possibilities of pixels and the initial region sizes in clustering and merging stages, respectively. For another, the computation times for some algorithms (LSC, LRW, SLIC, Seeds, to name a few) increase slightly with the number of superpixels, while our proposed method is reversing that owing to the bottom to the top scheme, which means that the larger K is, the less the regions that need to be merged in the second stage.

As supplementary, Table 1 lists the computational statistics of the compared approaches and the proposed method, where the number of superpixels is fixed on 700. The computational complexity of each algorithm is also listed. n is the number of iterations. It is clear that the results are consistent with the description above.

4.4 Limitation

Although the proposed approach has competitive performance with the most compared algorithms, state of the art on accuracy and efficiency, room for improvement still exists. First, the boundary adherence should be improved further. Second, the uniformity of the resultant superpixel sizes has some room for increase. Although small initial regions are merged into its neighbors with the constraint of size, the size of the resultant superpixels still varies greatly as shown in Fig. 12. The reason may fall in two aspects. On the one hand, it is caused by the image intrinsic properties. In order to seg-

Table 1 Performance metrics of superpixel segmentation algorithms as the superpixel number is 700

	DBSCAN [40]	SLIC [1]	SNIC [2]	SCALP [14]	LSC [5]	Seeds [8]	LRW [39]	Ours
Adherence to boundaries								
Boundary recall (BR)	0.9062	0.8172	0.8891	0.9037	0.9546	0.9045	0.8545	0.9185
Under-segmentation error (USE)	0.2129	0.1945	0.1826	0.1753	0.1628	0.2146	0.1827	0.1577
Achievable segmentation accuracy (ASA)	0.9553	0.9587	0.9608	0.9639	0.9678	0.9536	0.9585	0.9656
Segmentation speed								
Computational complexity	$\mathrm{O}(N)$	$\mathrm{O}(nN)$	$\mathrm{O}(N)$	$\mathrm{O}(nN)$	$\mathrm{O}(nN)$	$\mathrm{O}(nN)$	$\mathrm{O}\left(nN^2 \lg N\right)$	$\mathrm{O}(N)$
Average time per image (s)	0.0236	0.2069	0.0778	0.6501	0.4294	0.0939	66.6045	0.0261

ment an image accurately, smaller superpixels are presented in the detailed regions and larger ones, whose size is controlled by the desired number of superpixels, present on the flat areas. On the other hand, the proposed algorithm uses a constant threshold in the clustering stage and merges regions whose size are smaller than $\frac{1}{4} \times \frac{N}{K}$ only.

5 Conclusion

We develop a fast superpixel segmentation algorithm with more accuracy compared with several existing superpixel generation methods. Our proposed approach generates superpixels in two steps. In the first stage, we use a dynamic seeding strategy and a boundary probability map of an image to define a new distance measurement and produce the initial regions attaching to the object boundaries. In the second stage, we merge small regions obtained in the first step into their nearest neighbors by considering their size, color and spatial information. Experiments on BSD show that our proposed algorithm achieves competitive performance. In future work, we will incorporate the deep networks to get some image representation features and generate superpixels with better boundary adherence and more regular size. Besides, we will design a new network to generate superpixels end-to-end by taking superpixel segmentation as image smoothing problem. We also plan to extend the proposed method to video supervoxels segmentation and take superpixels/supervoxels as the input of deep networks of computer vision tasks.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., Susstrunk, S.: Slic superpixels compared to state-of-the-art superpixel methods. IEEE Trans. Pattern Anal. Mach. Intell. **34**(11), 2274–2282 (2012)
2. Achanta, R., Susstrunk, S.: Superpixels and polygons using simple non-iterative clustering. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 4895–4904 (2017)
3. Ban, Z., Liu, J., Cao, L.: Superpixel segmentation using gaussian mixture model. IEEE Trans. Image Process. **27**(8), 4105–4117 (2018)
4. Beucher, S., Meyer, F.: The morphological approach to segmentation: the watershed transformation. Math. Morphol. Image Process. **34**, 433–481 (1993)
5. Chen, J., Li, Z., Huang, B.: Linear spectral clustering superpixel. IEEE Trans. Image Process. **26**(7), 3317–3330 (2017)
6. Cheng, J., Liu, J., Xu, Y., Yin, F., Wong, D.W.K., Tan, N.M., Tao, D., Cheng, C.Y., Aung, T., Wong, T.Y.: Superpixel classification based optic disc and optic cup segmentation for glaucoma screening. IEEE Trans. Med. Imaging **32**(6), 1019–1032 (2013)
7. Choi, K., Oh, K.: Subsampling-based acceleration of simple linear iterative clustering for superpixel segmentation. Comput. Vis. Image Underst. **146**, 1–8 (2016)
8. Den Bergh, M.V., Boix, X., Roig, G., Van Gool, L.: Seeds: Superpixels extracted via energy-driven sampling. Int. J. Comput. Vis. **111**(3), 298–314 (2015)
9. Dollár, P., Zitnick, C.L.: Fast edge detection using structured forests. IEEE Trans. Pattern Anal. Mach. Intell. **37**(8), 1558–1570 (2014)
10. Dong, X., Shen, J., Shao, L.: Hierarchical superpixel-to-pixel dense matching. IEEE Trans. Circuits Syst. Video Technol. **27**(12), 2518–2526 (2017)
11. Dong, X., Shen, J., Shao, L., Van Gool, L.: sub-markov random walk for image segmentation. IEEE Trans. Image Process. **25**(2), 516–527 (2016)
12. Dong, X., Shen, J., Shao, L., Yang, M.H.: interactive cosegmentation using global and local energy optimization. IEEE Trans. Image Process. **24**(11), 3966–3977 (2015)
13. Giraud, R., Ta, V., Papadakis, N.: Scalp: Superpixels with contour adherence using linear path. In: International Conference on Pattern Recognition pp. 2374–2379 (2016)
14. Giraud, R., Ta, V., Papadakis, N.: Robust superpixels using color and contour features along linear path. Comput. Vis. Image Underst. **170**, 1–13 (2018)
15. Hartley, T., Sidorov, K., Willis, C.J., Marshall, A.D.: Gradient weighted superpixels for interpretability in cnns. Computer Vision and Pattern Recognition (2019)

16. Jampani, V., Sun, D., Liu, M.Y., Yang, M.H., Kautz, J.: Superpixel sampling networks. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 352–368 (2018)
17. Lee, S.H., Jang, W.D., Kim, C.S.: Contour-constrained superpixels for image and video processing. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 5863–5871 (2017)
18. Levinshtein, A., Stere, A., Kutulakos, K.N., Fleet, D.J., Dickinson, S.J., Siddiqi, K.: Turbopixels: fast superpixels using geometric flows. IEEE Trans. Pattern Anal. Mach. Intell. **31**(12), 2290–2297 (2009)
19. Li, Z., Wu, X.M., Chang, S.F.: Segmentation using superpixels: A bipartite graph partitioning approach. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2012, pp. 789–796. IEEE (2012)
20. Liang, Y., Shen, J., Dong, X., Sun, H., Li, X.: video supervoxels using partially absorbing random walks. IEEE Trans. Circuits Syst. Video Technol. **26**(5), 928–938 (2016)
21. Liang, Z., Shen, J.: local semantic siamese networks for fast tracking. IEEE Trans. Image Process. (2019). https://doi.org/10.1109/TIP.2019.2959256
22. Liu, B., Hu, H., Wang, H., Wang, K., Liu, X., Yu, W.: Superpixel-based classification with an adaptive number of classes for polarimetric sar images. IEEE Trans. Geosci. Remote Sens. **51**(2), 907–924 (2013)
23. Liu, M.Y., Tuzel, O., Ramalingam, S., Chellappa, R.: Entropy rate superpixel segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2011, pp. 2097–2104. IEEE (2011)
24. Liu, Y., Yu, M., Li, B., He, Y.: Intrinsic manifold slic: a simple and efficient method for computing content-sensitive superpixels. IEEE Trans. Pattern Anal. Mach. Intell. **40**(3), 653–666 (2018)
25. Liu, Y.J., Yu, C.C., Yu, M.J., He, Y.: Manifold slic: A fast method to compute content-sensitive superpixels. In: Computer vision and pattern recognition, pp. 651–659 (2016)
26. Machairas, V., Decencière, E., Walter, T.: Spatial repulsion between markers improves watershed performance. In: Mathematical Morphology and Its Applications to Signal and Image Processing, pp. 194–202. Springer (2015)
27. Machairas, V., Faessel, M., Cardenas-Pena, D., Chabardes, T., Walter, T., Decenciere, E.: Waterpixels. IEEE Trans. Image Process. **24**, 3707–3716 (2015)
28. Martin, D., Fowlkes, C., Tal, D., Malik, J.: A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In: Proceedings of the 8th IEEE International Conference on Computer Vision, 2001. ICCV 2001, vol. 2, pp. 416–423. IEEE (2001)
29. Meyer, F.: Un algorithme optimal pour la ligne de partage deseaux. Congrés Reconnaissance Formes. Intell. Artif. **2**, 847–857 (1991)
30. Moore, A.P., Prince, J., Warrell, J., Mohammed, U., Jones, G.: Superpixel lattices. In: IEEE Conference on Computer Vision and Pattern Recognition, 2008. CVPR 2008, pp. 1–8. IEEE (2008)
31. Oh, K., Choi, K.: Acceleration of simple linear iterative clustering using early candidate cluster exclusion. J Real-Time Image Proc. **16**, 945–956 (2016)
32. Peng, J., Shen, J., Yao, A., Li, X.: Superpixel optimization using higher order energy. IEEE Trans. Circuits Syst. Video Technol. **26**(5), 917–927 (2016)
33. Qian, X., Li, X., Zhang, C.: weighted superpixel segmentation. Vis. Comput. **35**, 985–996 (2019)
34. Ren, X., Malik, J.: Learning a classification model for segmentation. In: Proceedings of the 9th IEEE International Conference onComputer Vision, 2003, pp. 10–17. IEEE (2003)
35. Reyes, A., Rincón, M.E.R., García, M.O.M., Santana, E.R.A., Cadena, F.A.A.: Robust image segmentation based on superpixels and Gauss–Markov measure fields. In: 16th Mexican International Conference on Artificial Intelligence (MICAI), pp. 46–52. IEEE (2017)
36. Schuurmans, M., Berman, M., Blaschko, M.B.: Efficient semantic image segmentation with superpixel pooling. Computer Vision and Pattern Recognition (2018)
37. Shen, J., Dong, X., Peng, J., Jin, X., Shao, L., Porikli, F.: submodular function optimization for motion clustering and image segmentation. IEEE Trans. Neural Netw. **30**(9), 2637–2649 (2019)
38. Shen, J., Du, Y., Li, X.: Interactive segmentation using constrained laplacian optimization. IEEE Trans. Circuits Syst. Video Technol. **24**(7), 1088–1100 (2014)
39. Shen, J., Du, Y., Wang, W., Li, X.: Lazy random walks for superpixel segmentation. IEEE Trans. Image Process. **23**(4), 1451–1462 (2014)
40. Shen, J., Hao, X., Liang, Z., Liu, Y., Wang, W., Shao, L.: Real-time superpixel segmentation by dbscan clustering algorithm. IEEE Trans. Image Process. **25**(12), 5933–5942 (2016)
41. Shen, J., Peng, J., Dong, X., Shao, L., Porikli, F.: higher order energies for image segmentation. IEEE Trans. Image Process. **26**(10), 4911–4922 (2017)
42. Stutz, D.: Superpixel segmentation: an evaluation. Pattern Recogn. **9358**, 555–562 (2015)
43. Stutz, D., Hermans, A., Leibe, B.: Superpixels: an evaluation of the state-of-the-art. Comput. Vis. Image Underst. **166**, 1–27 (2018)
44. Tian, X., Jiao, L., Zheng, X., Zhang, X.: inter-frame constrained coding based on superpixel for tracking. Vis. Comput. **31**(5), 701–715 (2015)
45. Verelst, T., Blaschko, M.B., Berman, M.: Generating superpixels using deep image representations. arXiv: Computer Vision and Pattern Recognition (2019)
46. Vincent, L., Soille, P.: Watersheds in digital spaces: an efficient algorithm based on immersion simulations. IEEE Trans. Pattern Anal. Mach. Intell. **13**(6), 583–598 (1991)
47. Wang, C., Chan, S., Zhu, Z.Y., Zhang, L., Shum, H.: superpixel-based color-depth restoration and dynamic environment modeling for kinect-assisted image-based rendering systems. Vis. Comput. **34**(1), 67–81 (2018)
48. Wang, H., Shen, J., Yin, J., Dong, X., Sun, H., Shao, L.: adaptive nonlocal random walks for image superpixel segmentation. IEEE Trans. Circuits Syst. Video Technol. (2019). https://doi.org/10.1109/TCSVT.2019.2896438
49. Wang, W., Shen, J.: deep visual attention prediction. IEEE Trans. Image Process. **27**(5), 2368–2378 (2018)
50. Xie, S., Tu, Z.: Holistically-nested edge detection. In: Proceedings of the IEEE international conference on computer vision, pp. 1395–1403 (2015)
51. Zhang, Y., Li, X., Gao, X., Zhang, C.: A simple algorithm of superpixel segmentation with boundary constraint. IEEE Trans. Circuits Syst. Video Technol. **27**(7), 1502–1514 (2017)
52. Zhou, X., Wang, Y., Zhu, Q., Xiao, C., Lu, X.: SSG: superpixel segmentation and grabcut-based salient object segmentation. The Visual Computer **35**(3), 385–398 (2019)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Yongxia Zhang received the B.S. degree from the Qilu University of Technology, Jinan, China, in 2011, the Ph.D. degree in the School of Computer Science and Technology, Shandong University, Jinan, China in 2017. She is currently a lecturer in the School of Computer Science and Technology, Shandong University of Finance and Economics, Jinan, China. Her research interests include computer vision, image processing and machine learning. In particular, she focused on the problem of image and video denoising and image segmentation.

Qiang Guo received the B.S. degree from Shandong University of Technology, Zibo, China, in 2002, the M.S. and Ph.D. degrees from Shanghai University, Shanghai, China, in 2005 and 2010, respectively. From 2012 to 2015, he was a Post-Doctoral Fellow with Shandong University, Jinan, China. He is currently an associate professor with the School of Computer Science and Technology, Shandong University of Finance and Economics, Jinan, China. His research interests include image restoration, sparse representation, and object detection. He is a member of the IEEE.

Caiming Zhang received the B.S. and M.S. degrees from Shandong University, Jinan, China, in 1982 and 1984, respectively, and the Ph.D. degree from the Tokyo Institute of Technology, Tokyo, Japan, in 1994. From 1998 to 1999, he was a postdoctoral fellow with the University of Kentucky, Lexington. He is currently a professor with Shandong University, and a distinguished professor with Shandong University of Finance and Economics. His research interests include computer aided geometric design, computer graphics, information visualization, and medical image processing.