Journal of Computers Vol. 30 No. 5, 2019, pp. 252-267
doi:10.3966/199115992019103005020

Superpixel Segmentation Using Improved Lazy Random Walk Framework Based on Texture Complexities

Yi-Xuan Zhan, Chin-Han Shen, Hsu-Feng Hsiao*

Department of Computer Science, National Chiao Tung University, Hsinchu 300, Taiwan, ROC
rlixaoi@gmail.com, fionaschs@nctu.edu.tw, hillhsiao@cs.nctu.edu.tw

Received 15 March 2019; Revised 30 June 2019; Accepted 4 September 2019

Abstract. Superpixel segmentation has been a very important pre-processing step of many computer vision applications. By grouping the pixels with similar data properties, the computation complexity can be reduced since the scale of data processing has been transformed from pixel level to region level. In this paper, an improved superpixel segmentation approach using a lazy random walk algorithm is proposed. Two major improvements are applied to obtain the better visualization results: center relocation and splitting strategy modification. The improved performance is confirmed with the subjective and objective performance comparison. In particular, the sizes of the produced superpixels depend on texture complexities of different regions which can be more appreciated

Keywords: image segmentation, lazy random walk, superpixel

1 Introduction

An image or a video frame is composed of many pixels. With the development of technologies, the required resolution of multimedia applications escalates, and the amount of data requiring image processing has also increased significantly. In order to efficiently process large amounts of data, an efficient approach is to integrate homogenous units into a smaller number of new units. A technique called superpixel segmentation has been developed for the classification of image pixels based on homogeneity.

A superpixel is a collection of a few pixels of the same or similar characteristics, such as color and intensity. It remains a challenge to have an excellent superpixel segmentation method. However, there are some principles that are generally agreed on when designing a superpixel segmentation algorithm: [1-3]

- A superpixel segmentation method should assign each pixel just to a single superpixel; therefore, there are no overlapped superpixels.
- Each superpixel should represent a connected set of pixels.
- Superpixels should adhere well to object boundaries.
- Superpixels should be generated as efficiently as possible.

Superpixel segmentation has been wildly used in the applications of computer vision and image processing. The major advantage of superpixel representation is to increase computational efficiency. Using superpixels can significantly reduce the number of units of an image, compared to using pixels. Currently, the mainstream of researches using deep learning networks focuses on visual contents. The huge amount of required processing has been one of the bottlenecks in such areas. One of the temporary solutions is to limit the spatial resolution of an input image. Another favorable approach is to digest an input image into an array of superpixels to reduce the dimension of an input data.

Due to the effectiveness in representation based on mid-level cues [4], computer vision tasks like object segmentation and recognition [5-9], background subtraction [10], and multi-target tracking [11-12] have adopted the superpixel segmentation. Visual tracking techniques often employ superpixels to handle non-rigid and deformable targets [13-16]. Wang et al. [13] use superpixels for constructing the

* Corresponding Author

discriminative appearance model to distinguish foreground objects from cluttered backgrounds. Particle filtering is used to find the optimal target state. A dynamic Bayesian network tracking [14] adopts a superpixel-based constellation model to deal with non-rigid deformations. Xiao et al. [15] and Hong et al. [16] have developed the tracking techniques with more information levels to overcome the limitation of flat representations. Three levels of features—pixel, superpixel, and bounding box have been used to avoid the situation where the semantic relations between superpixels are not well-considered for segmentation.

In this paper, an improved superpixel segmentation method, based on lazy random walk (LRW) algorithm, is proposed. With the better performance of the proposed method, it can be a great benefit to other researches of visual contents where computation efficiency matters. Also, the sizes of the produced superpixels depend on texture complexities of different regions which can be more appreciated for the applications where superpixels are used as delegates of an image.

The rest of the paper is organized as follows. We will review several superpixel segmentation algorithms in Section 2. The details of our proposed method are in Section 3. The evaluation of the proposed method and the comparison will be described in Section 4, followed by the conclusion in Section 5.

2 Related Works

There are many superpixel segmentation algorithms in the literature. First, algorithms for generating superpixels can be categorized as graph-based and gradient-based approaches. With the emersion of Simple linear iterative clustering (SLIC), a new category called k-means based algorithms is getting popular in the research of superpixel segmentation. Different methods have their advantages and drawbacks, depending on the requirement of an application. For example, the graph-based methods are appealing for generating superpixels adherent to object boundaries. However, if the superpixels are used to build a graph, a method that produces a more regular lattice will be a better choice [1]. In the following, some popular works of superpixel segmentation will be reviewed. The methodologies to evaluate a superpixel segmentation algorithm will be depicted.

2.1 Graph-based Algorithms

As the title implies, graph-based approaches of superpixel segmentation take an input image as a graph and each pixel is taken as a node in the graph. Connections between nodes pairs are called edges, which contain the weights to measure similarity between two nodes. The objective of a typical graph-based algorithm is to minimize the cost function defined over the graph to create the superpixels.

A classical graph partitioning algorithm called Normalized Cuts (NCut) [17] has been commonly used in image segmentation. The authors suggested that image partitioning should be done from the big picture downward and proposed the approach which focuses on extracting the global impressions of an image to the perceptual grouping problems. In order to avoid the unequal partition result of an image, the partitioning criterion is used to measure the cut cost as a fraction of the total edge connections to all the nodes in the graph. Visually, the segmentation result is regular and pleasant but the boundary adherence of the generated superpixels are not ideal. Moreover, since the measurement of cut cost involves the nodes in the whole graph, the computational cost becomes quite expensive when the number of superpixels increases. An alternative graph-based approach was proposed by Felzenszwalb and Huttenlocher [18]. First, the weights of this approach are measured by the region dissimilarity. Then the graph is performed with agglomerative hierarchical clustering to classify the nodes in the graph. The segmentation results are presented in several minimum spanning trees (MST).

Moore et al. [19] proposed a superpixel lattices method where an image is segmented into several regions at the initialization step. The input used in this approach is a boundary map, which is a 2D array measuring the probability of existence of a semantically meaningful boundary between two pixels. To process the input more efficiently, the boundary map is normalized to boundary cost map. The superpixel segmentation is to use the boundary cost map to find optimal paths along with the boundaries. To make sure the grid segmentation is regular, two rules are employed to find the optimal paths: (1) the horizontal and vertical paths only have single cross point, (2) the optimal paths with same direction must not cross each other.

Veksler et al. [20] take the superpixel segmentation as an energy minimization framework and optimize it with graph cut. In this approach, the basic energy function is defined as a trade-off. It depends on whether the user needs a well boundary adherence result or a faster calculation. Shen et al. [21] proposed a graph-based method which uses lazy random walk (LRW) as the mechanism of segmentation. Two main steps are included in the approach: (1) the superpixel seeds initialization and (2) the energy optimization to segment the superpixels. The method is able to segment the complicated texture data result and preserve the weak boundary.

2.2 Gradient-based Algorithms

From a rough clustering of pixels, a typical gradient-based method iteratively refines the clusters until a convergence criterion is met to form superpixels [2]. A geometric-flow-based algorithm was proposed by Levinshtein et al. [3] for computing a dense over-segmentation of an image. The goal of the method is to distribute the superpixels evenly in the whole image. The superpixels are constrained to have the uniform size, compactness, and boundary adherence. The watershed approach proposed by Vincent and Soille [22] fulfills the superpixel segmentation using gradient ascent approach from the local minima in the image. The calculation speed of this method is relatively fast, but it does not provide the external control of desired superpixel number and compactness. The completed superpixels are often with the irregular shapes and sizes, and their lack of well boundary adherence can be a problem.

2.3 K-means-based Algorithms

K-means-based methods share the same conceptual process as the gradient-based algorithms. The major difference between two types of algorithms is that K-means-based methods replace the gradient-ascent calculation with K-means clustering method to generate the superpixels.

Achanta et al. [1] presented a well-known algorithm called simple linear iterative clustering (SLIC), adopting the k-means clustering approach to generate superpixels. There are two major points: (1) the number of distance calculations in the optimization is dramatically reduced by limiting the search space to a region proportional to the superpixel size, (2) The measurement of the weighted distance combines color similarity and spatial proximity while simultaneously providing external control to adjust the size and compactness of the superpixels. Zeng et al. [23] proposed a structure sensitive over-segmentation technique which adopts geometric flows to compute the geodesic distances among the pixels. The generated superpixel size is verified with the variation of intensity or color. The superpixels with smaller size are generated in structure dense regions with high intensity or color variation. On the other hand, the superpixels with larger size are generated in structure-sparse regions.

A superpixels extracted via energy-driven sampling (SEEDS) approach based on a simple hill-climbing optimization was proposed by Bergh et al [24]. After the initialization of superpixel, the segmentation refinement is finished by iteratively modifying the boundaries. This algorithm defines an energy function based on enforcing color similarity between the boundaries and superpixel color histogram [26]. The method has the faster computational speed [26] but the generated shapes of superpixels are irregular. The flooding based method with color, compactness and smoothness constraints (FCCS) algorithm was proposed by Pan et al. [25]. Two main improvements were proposed in FCCS: a new distance metric is defined for estimating pixels and seeds' similarity with color, compactness and smoothness constraints and a seeds update strategy based on Lloyd's algorithm is adopted for optimizing seeds location and superpixels' contour regions [25]. The method can generate better boundary adherence perceptually, compared with some previously mentioned works [1, 23].

2.4 Methodologies for Segmentation Results

To evaluate the superpixel segmentation performance, some commonly used benchmarks metrics from the extended Berkeley segmentation benchmark were implemented by Stutz et al. [27]. In this subsection, we will review the related works and definitions of the evaluation metrics for superpixel segmentation.

2.4.1 Boundary Recall (BR)

The Boundary Recall (BR) introduced in [28] is part of the Precision-Recall framework to accurately detect boundaries in natural scenes. It was originally designed for evaluating a boundary detection algorithm. Higher BR value means that superpixels adhere well to the object boundaries in an image. For the principle that the superpixel segmentation result shall match the objects boundary well, it is one of the important objective indexes to measure the accuracy of segmentation.

Let sp_j be a superpixel segmentation and g_i be a ground truth segmentation, the BR is defined as below:

$$BR(sp_j, g_i) = \frac{|TP(sp_j, g_i)|}{TP(sp_j, g_i) + FN(sp_j, g_i)} \tag{1}$$

where $TP(sp, g)$ means true positives for the boundary pixels in g when there is a pixel in range r of the boundaries of sp. $FN(sp, g)$ means false negatives for the boundary pixels in g when there is no pixel in range γ of the boundaries of sp.

2.4.2 Undersegmentation Error (UE)

The scenario of an undersegmentation error is shown in Fig. 1 where several superpixels are covered by a region in the ground truth. There are various implementations of undersegmentation error metrics. To avoid a serious penalty for large superpixels that have only a small overlap with the ground truth segment, the undersegmentation error (UE) proposed by Neubert and Protzel [29] in the extended Berkeley segmentation benchmark is defined as follows.

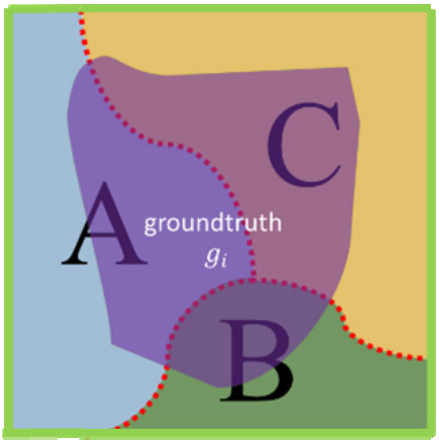


Fig. 1. Illustration of undersegmentation error

Three superpixels sp_j (A, B, C) are covered with the groundtruth g_i (center covered area)

$$UE(Sp, g) = \frac{1}{N} \Sigma_{g_i \in G} \Sigma_{Sp_j \cap g_i \neq \phi} \min\{|Sp_j \cap g_i|, |Sp_j - g_i|\} \tag{2}$$

where N denotes the total number of pixels on the image, $|sp_j \cap g_i|$ denotes the number of the pixels which both belong to sp_j and g_i, and $|sp_j - g_i|$ denotes the number of the pixels which belong to sp_j but not g_i.

2.4.3 Achievable Segmentation Accuracy (ASA)

The Achievable Segmentation Accuracy (ASA) used by the extended Berkeley Segmentation Benchmark is defined by Liu et al. [2]. It measures the highest accuracy achievable for object segmentation based on superpixel units. Each superpixel is labeled with the ground truth segment which has the largest overlap. Mathematically, the ASA is defined as the performance upper-bound of the segmentation results with the equation written below:

$$ASA(Sp, g) = \frac{1}{N} \Sigma_{Sp_j \in Sp} \max_{g_i} \{| Sp_j \cap g_i |\} \quad \textbf{(3)}$$

3 The Proposed Superpixel Segmentation

As mentioned in the previous section, lazy random walk (LRW) algorithm can be quite helpful for the problem of superpixel segmentation [21]. Inspired by it, we improve the LRW for better superpixel segmentation.

3.1 Foundation of Lazy Random Walk-based Superpixel Segmentation

The flow of the original LRW approach is shown in Fig. 2. First, the seeds initialization is fulfilled by placing the central pixel averagely in the image plane. Then, each pixel is assigned to a unique initial seed to form the superpixel using LRW algorithm. In order to make the superpixels have better performance on edge preserving and compactness, an energy optimization algorithm is used to relocate the seed positions and split the large superpixels into two small new superpixels in each iteration. The process of the LRW for superpixel optimization will be terminated only when the number of the segmented superpixels or the maximum number of the iterations reaches its threshold.

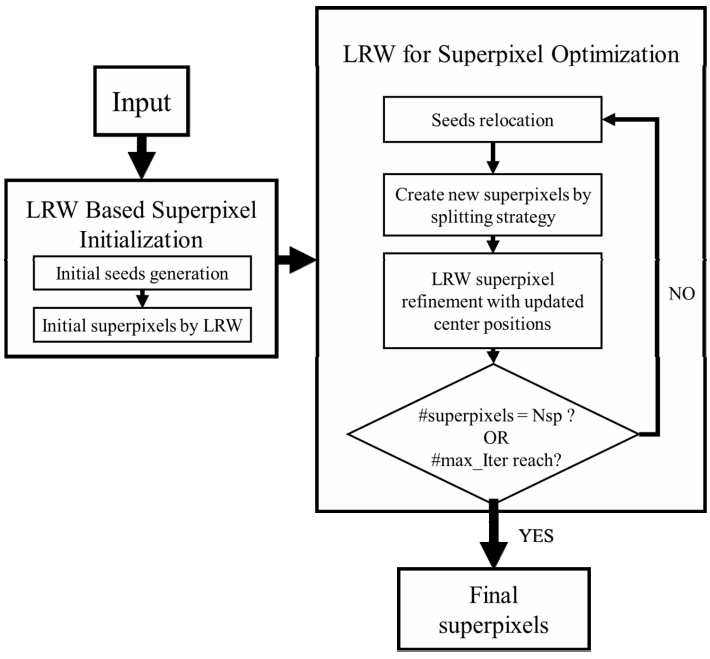


Fig. 2. The work flow of LRW for superpixel segmentation

The LRW algorithm is one of graph-based algorithms mentioned in the Section 2.1. The input image $I(p_i)$ is taken as an undirected graph $G = (V, E)$. The input image is presented as a weighted graph containing a set of nodes V and edges $E \subseteq V \times V$. Each pixel p_i on the image is identified by a unique node $v_i \in V$. The degree of each vertex is computed as $d_i = \Sigma_j w_{ij}$ for all the edges which connect to v_i.

The weight w_{ij} is defined by the similarity between two neighbor nodes v_i and v_j, and usually the

pixel intensity is used in the form of Gaussian function. Many graph-based image segmentation methods [30-32] have adopted the Gaussian weight as the following:

$$W_{ij} = \exp(-\frac{\| Y_i - Y_j \|^2}{2\sigma^2}) \tag{4}$$

where Y_i and Y_j denote the intensity values of the two nodes v_i and v_j. σ is the user defined parameter.

The LRW graph has the property that a lazy random walk remains at the same node by adding a self-loop to each vertex aiming to solve the segmentation problem in weak boundary and complex texture regions [21]. The input graph could be turned into an adjacency matrix w_{ij} defined as

$$W_{ij} = \begin{cases} 1-\alpha, \ if \ i = j, \\ \alpha \cdot w_{ij}, if \ v_i \ and \ v_j \ are \ neighbors, \\ 0, otherwise \end{cases} \tag{5}$$

where α is a control parameter of self-loop in the range (0, 1). The adjacency matrix W is a sparse and symmetric matrix whose nonzero elements are positive. From equation (5), the adjacency matrix W_{ij} is transformed to Laplacian matrix $L = D - \alpha W$, where D is a diagonal matrix and D_{ii} is the degree of the i-th node v_i. In LRW algorithm, the commute time CT_{ij} [33-34] denotes the expected quantities of steps that starts at node v_i to reach node v_j and then goes back to node v_i. From the definition above, both the Laplacian matrix L and its inverse matrix L^{-1} are symmetric matrices and we can express it with $L_{ij}^{-1} = L_{ji}^{-1}$. Therefore, the normalized Laplacian $L = (I - \alpha D^{-1/2} W D^{-1/2})$ is obtained and the normalized commute time CJ is defined as the following:

$$CT_{ij} = \begin{cases} 1 - L_{ij}^{-1}, \ if \ i \neq j, \\ 1 \quad if \ i = j, \end{cases} \tag{6}$$

where l is the identity matrix. Due to the property that the probability is inversely proportion to the commute time, the likelihood probabilities of superpixel label l can be defined as the following equation:

$$Prob_l = 1 - CT_{ij} = L_{ij}^{-1} \tag{7}$$

From the equation (7), a defined matrix $S = D^{-1/2} W^{-1/2}$ is used to rewrite the likelihood probabilities of superpixel label l as:

$$Prob_l = (I - \alpha S)^{-1} f_l \tag{8}$$

where $Prob_l$ is a $N \times 1$ vector and the probabilities of the nodes are assigned the superpixel label l. Then f_l is a $N \times 1$ column vector whose all the elements are zero except that the seed pixels are one. When $f_l(p_i) = 1$ means p_i is labeled with l, and if $f_l(p_i) = 0$, otherwise.

3.1.1 LRW based Superpixel Initialization

During the phase of LRW-based superpixel initialization in Fig. 2, the first step is to place the superpixel seeds distributed over the image as even as possible. However, if there is any seed located near a strong edge, it can be the false basis of subsequent segmentation process. Hence, the positions of the initial seeds are moved along the gradient direction of their intensity values after being placed evenly over the image.

After the generation of seed position, the next step is using LRW algorithm to assign each pixel to a unique superpixel seed. The LRW algorithm will converge at a pixel p_i with the likelihood probabilities $Prob_{l_k}(p_i)$ from the equation (8). As a result, the set of the pixels which belong to the l-th superpixel can be obtained from the commute time as the following equation:

$$R(p_i) = arg\ \min_{l_k} CT(c_{l_k}, p_i) = arg\ \max_{l_k} Prob_{l_k}(p_i) \tag{9}$$

where c_{l_k} denotes the central pixel of the l-th superpixel. Finally, the initial superpixels are obtained by $Sp_{l_k} = \{p_i \mid R(p_i) = l_k\}$ where $\{i = 1, 2, \ldots, N\}$ and $\{k = 1, 2, \ldots, M\}$. And N is the number of the pixels in an image and M is the number of the initial superpixels.

3.1.2 Superpixel Optimization

In order to make the superpixels with better performance both on edge preserving and compactness, the proposed energy optimization function composed of data term and smooth term is expressed as:

$$E = \Sigma_l (Area(Sp_l) - Area(\overline{Sp}))^2 + \Sigma_l \widetilde{W}_p CT(c_l^n, p)^2 \tag{10}$$

The data term locates at the front part of equation (10) is used to make the texture information of a superpixel more homogeneous. $Area(Sp_l)$ denotes the area of superpixel Sp_l and $Area(\overline{Sp})$ denotes the average area of all superpixels. The smooth term in the later part of equation (10) is used to make the superpixels adhere well to object/region boundaries. $\widetilde{W}_p$ is defined as $\widetilde{W}_p = e^{-CT(c_l, p)/\beta}$ with a normalization factor β, and it is a penalty function to measure the inconsistency under the specific superpixel label.

Since the equation (10) is a nonconvex function, the energy optimization function can be solved iteratively. First, the smooth term in equation (10) is minimized by computing the first derivative on the variable c_l^n as the following.

$$\begin{aligned} \frac{\partial E}{\partial c_l^n} &= 2\Sigma_l \widetilde{W}_p CT(c_l^n, p) \nabla CT(c_l^n, p) \\ &\approx 2\sum_l \widetilde{W}_p CT(c_l^{n-1}, p) \frac{p - c_l^n}{\| p - c_l^{n-1} \|} = 0 \end{aligned} \tag{11}$$

where n is the number of iterations and c_l^0 is the initial central position of superpixel Sp_l. The new central position of superpixel Sp_l can be relocated according to the following equation:

$$c_l^n = \frac{\Sigma_l \widetilde{W}_p \frac{CT(c_l^{n-1}, p)}{\| p - c_l^{n-1} \|} p}{\Sigma_l \widetilde{W}_p \frac{CT(c_l^{n-1}, p)}{\| p - c_l^{n-1} \|}} \tag{12}$$

Second, according to equation (10), when $Area(Sp_l)$ is equal to $Area(\overline{Sp})$, the data item will have the minimum value. The value of $Area(Sp_l)$ presents the texture complexity of the specific superpixel Sp_l; therefore, the local binary pattern (LBP) [35] can be used to compute the texture simplicity of a superpixel. The LBP of a pixel p is define as below:

$$LBP^{Q,R}(p) = \sum_{q=0}^{Q-1} s(L_q - L_p) 2^q \tag{13}$$

where

$$s(x) = \begin{cases} 1, x \geq 0 \\ 1, x < 0 \end{cases} \tag{14}$$

where 2^q is a binomial factor, and R is the radius of the circle that forms a circularly symmetric neighbor set for the pixel p. The parameters $Q = 8$ and $R = 1$ are set to compute the constrained LBP value.

According to the texture feature presented by the LBP value, the area of a superpixel Sp_l and the average area of superpixels are defined as the following equations (15) and (16):

$$Area(Sp_l) = \Sigma_{p \in Sp_l} LBP(p) \tag{15}$$

$$Area(\overline{Sp}) = \frac{\Sigma_{p \in Sp_l} LBP(p)}{N_{sp}} \tag{16}$$

where N_{sp} is the user defined number of superpixels to achieve. When the value of $Area(Sp_l)$ is larger, the texture information of Sp_l is more complicated. Hence, a threshold TH_{LBP} is adopted for checking the LBP value if the superpixel is diverse enough to be divided into two new superpixels. The splitting condition is achieved when $Area(Sp_l) / Area(\overline{Sp}) \geq TH_{LBP}$ to split the current superpixel into two parts.

Next, the splitting method chooses the principal components analysis (PCA) to divide the superpixel. The superpixel Sp_l is divided along the direction of the eigenvector S with the largest eigenvalue of the covariance matrix which consists of the commute time and the shape of superpixel Sp_l. The direction is determined by $(p - c_l) \cdot S = 0$ and the covariance matrix of superpixel Sp_i is defined as the following equation:

$$\Sigma_{\{p | p \in Sp_l,\, p \notin c_l\}} \frac{CT(c_l, p)^2}{\| p - c_l \|^2} (p - c_l)(p - c_l)^T \tag{17}$$

As a result, after splitting superpixel Sp_l with the covariance matrix $Cov(Sp_l)$ into two small superpixels, the two new central positions of the small superpixels can be obtained as below:

$$\begin{aligned} c_{l_{new,1}} &= \frac{\Sigma_{\{p | p \in Sp_l, (p - c_l) \cdot S > 0\}} \widetilde{W}_p \frac{CT(c_l, p)}{\| p - c_l \|} p}{\Sigma_{\{p | p \in Sp_l, (p - c_l) \cdot S > 0\}} \widetilde{W}_p \frac{CT(c_l, p)}{\| p - c_l \|}} \\ c_{l_{new,2}} &= \frac{\Sigma_{\{p | p \in Sp_l, (p - c_l) \cdot S < 0\}} \widetilde{W}_p \frac{CT(c_l, p)}{\| p - c_l \|} p}{\Sigma_{\{p | p \in Sp_l, (p - c_l) \cdot S < 0\}} \widetilde{W}_p \frac{CT(c_l, p)}{\| p - c_l \|}} \end{aligned} \tag{18}$$

3.2 The Proposed Modification of LRW for Superpixel Segmentation

The proposed superpixel segmentation framework with our improvements is showed in Fig. 3. First, we generate the initial seeds by evenly placing the positions of superpixel seeds over the image. Then, according to the gradient of intensity values, we move the seeds to the locations which have the local minimum variety of gradient to avoid locating the seeds close to the object boundaries. Before using LRW to initialize the superpixels with the initial seeds, we use the equation (5) to compute the edge weight w_{ij} between two neighbor pixels p_i and p_j. Then we construct the sparse adjacency matrix W_{ij} in equation (6) and compute the commute time using the equation (7). After that, the LRW algorithm is used to assign each pixel to a unique superpixel labeled by the equation (8). It can approach the optimization of the superpixel segmentation in each iteration. Two main improvements for superpixel optimization are proposed: adjusting the position after superpixel relocation and the new proposed splitting condition in splitting strategy, which will be described in details as follows.

3.2.1 The Quad-segments Relocation

At the beginning of each iteration, each pixel is assigned to a unique superpixel label with the maximum likelihood probability using the LRW method. Then we relocate each superpixel center by the equation

(12). At this step, the original mechanism of center relocation sometimes wrongly assigns the central pixel for the superpixel. When the shape of this superpixel is concave, it may not locate the superpixel center inside the superpixel. This situation will be a bad foundation in the next iteration. Therefore, we adjust the location of superpixel center if it does not belong to this superpixel with the proposed quad-segments relocation method.

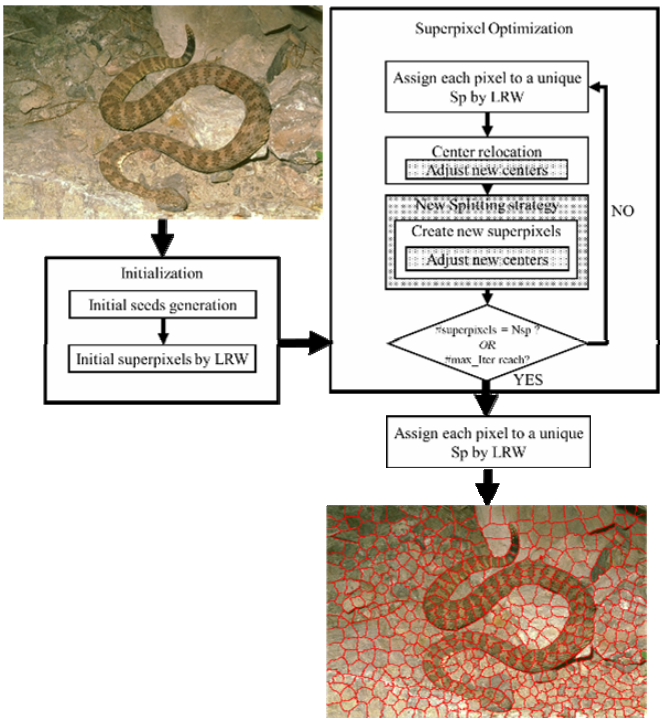


Fig. 3. The framework of the proposed LRW for superpixel segmentation

First, the gravity center $C_g(Sp_l)$ of the superpixel Sp_l is computed using the following equation:

$$C_g(Sp_l) = \frac{1}{N} \sum_{p_i \in Sp_l}^{n} (x_i, y_i) \tag{19}$$

where N is the number of pixels belong to Sp_l, and (x_i, y_i) denote the x and y coordinates of pixel p_i. If the pixel which locates at $C_g(Sp_l)$ belongs to Sp_i, then the center of Sp_i is $C_g(Sp_l)$; otherwise, we use the proposed *quad-segments relocation* method to obtain the appropriate superpixel center.

In the first step, we obtain the rectangular bounding box of the superpixel region, and the position of middle center C_m in the bounding box. The superpixel is divided into four segments by a vertical line and a horizontal line crossed at the position C_m, which is illustrated in Fig. 4. The points marked in different colors denote the pixels from four segments. Each segment is presented in s_k, and has its own gravity center C_{gk} computation according to the member pixels.

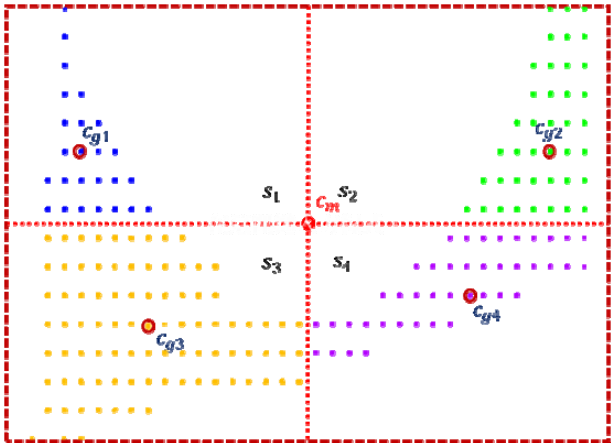


Fig. 4. Illustration of the bounding box and the four parts of the superpixel Sp_l (marked in area with colored dots)

$$C_{g_k}(s_k) = \frac{1}{N_k} \sum_{p_i \in s_k}^{N_k} (x_i, y_i) \tag{20}$$

k =1, …, 4 and N_k is the number of pixels in segment s_k.

In the second step, we pick the four calculated C_{gk} locate inside Sp_i to compute the sum of geometric distances from all the pixels belong to Sp_l to C_{gk} as below:

$$dist(C_{g_k}) = \sum_{p_i \in Sp_l}^{N} \sqrt{(x_i - c_{x_k})^2 + (y_i - c_{x_k})^2} \tag{21}$$

where (c_{xk}, c_{yk}) denotes the x and y coordinates of the gravity center C_{gk}.

In the final step, check if the gravity center $c_{gk} = (c_{xk}, c_{yk})$ with the minimum $dist(C_{gk})$ is inside the range of superpixel Sp_l or not. If the selected gravity center C_{gk} is not inside the superpixel range, the quad-segments relocation is applied in each segment to find the center candidates in minor level until the suitable center of Sp_l is found. Otherwise, the gravity center C_{gk} with the minimum $dist(C_{gk})$ is selected to be the new reallocation center. This quad-segments relocation will be applied whenever dividing a large superpixel in the superpixel splitting strategy.

3.2.2 The Splitting Strategy of a Superpixel

After adjusting the relocation of the current superpixel Sp_l, another critical issue of image segmentation is the splitting strategy. Using LBP as the splitting strategy is not a good option. According to the equation (13) and (14), when $s(x)$ is equal to 1, a large LBP value will be generated and wrongly estimate the member pixels that have the same intensity values in Sp_l. The superpixels with sufficient homogeneity may still be split into two smaller superpixels.

Therefore, we propose the splitting condition to decide whether or not Sp_l shall be split. There are two kinds of measurement involved in the proposed splitting condition: a simple metric to measure the texture complexity called the sum of local binary difference (SLBD) and the intensity variance of a superpixel. We design the splitting condition such that the sizes of the produced superpixels depend on texture complexities of different regions.

The SLBD value of a pixel p is defined as below:

$$SLBD(p) = \sum_{q=0}^{Q-1} s(|L_q - L_p|) \tag{22}$$

and

$$s(x)=\begin{cases}1, x>0\\ 0, x=0\end{cases} \tag{23}$$

where L_p and L_q are the intensity values of pixel p and pixel q.

We set $Q=8$ to denote the 8-neighbors of the pixel p. We define the ratio of *SLBD* value of the superpixel Sp_l as the following:

$$Ratio_{SLBD}(Sp_l)=\frac{\Sigma_{p\in Sp_l} SLBD(p)}{\sum_{j=1}^{N} SLBD(p_j)/N_{sp}} \tag{24}$$

where N is the total number of pixels and N_{sp} is the user defined number of the expected superpixels.

We then use a dynamic threshold TH_{SLBD} to control the process of splitting. First, we set a TH_{SLBD} to a strict value to make the superpixels with larger texture complexity have the higher priority to be split in the first few iterations. When there are no superpixels divided in two consecutive iterations, a more relaxed TH_{SLBD} is used to split more superpixels.

In addition to the *SLBD* value, we also compute the variance of a superpixel as another guidance. Because the SLBD value only considers the neighbor information of a pixel in local, we use another fast measurement, which is the variance of the superpixel, to measure the intensity diversity within the superpixel Sp_l. The variance of the superpixel Sp_l is defined as the following equation:

$$Var(Sp_l)=\frac{1}{N}\sum_{p_i\in Sp_l}^{n}(L_{p_i}-Mean(L_{Sp_l}))^2 \tag{25}$$

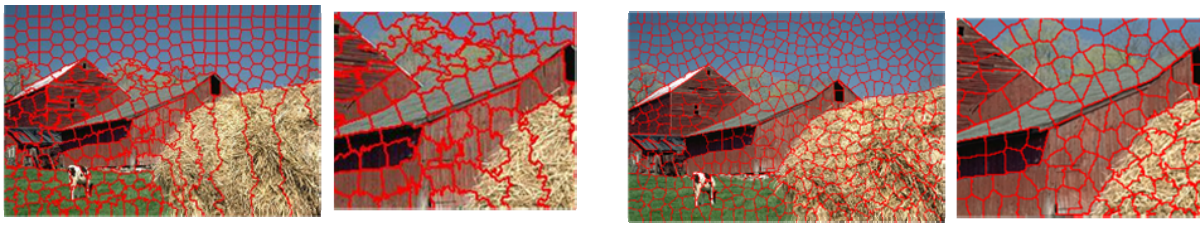
where L_{pi} denotes the intensity value of pixel p_i. $Mean(L_{Sp_l})$ denotes the mean of the pixel intensity values from Sp_l, and N is the number of pixels belong to Sp_l. When $Var(Sp_l)$ is large, It means the intensity consistency is not unanimous enough among the member pixels in current superpixel Sp_l. Hence, the superpixel with large enough variance should be divided too. Two thresholds TH_{var} and TH_{SLBD} are used to control the splitting condition. When both $Ratio_{SLBD}(Sp_l)\geq TH_{SLBD}$ and $Var(Sp_l)\geq TH_{var}$ conditions are satisfied, the superpixel is split by the equation (17). In our empiricism, we set TH_{SLBD} to 1.35 as initial value and set TH_{var} to 0.005 for good superpixel segmentation results. The new split superpixels will be assigned to the new superpixel centers according the equation (18)

4 Experimental Evaluation and Discussion

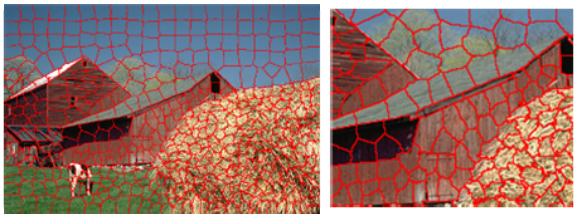
In this section, we are going to examine the segmentation results of the proposed method. From the design of the proposed method, we expect the results to be more homogeneous in each segment and the size of a superpixel reflect the complexity of a region. The subjective and objective performance of the proposed method will be compared with: (a) the well-known simple linear iterative clustering (SLIC) [1] and (b) the original lazy random walks (Original-LRW) algorithm [21].

4.1 Results of Subjective Evaluation

We have visualized the segmentation results with several image examples from the Berkeley database (BSD500) [36]. In the Fig. 5 and Fig. 6, our proposed methods have the best object boundary adherence performance. For the segmentation accuracy, our proposed method can generate more regular superpixels with highly homogeneous texture.

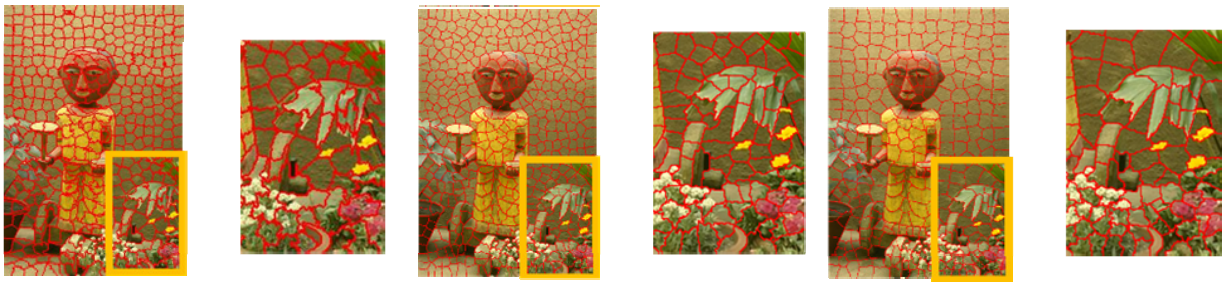


(a) Results by SLIC (b) Results by Original-LRW



(c) Results by the proposed-LRW

Fig. 5. Visual comparison (Boundary adherence)



(a) Results by SLIC (b) Results by Original-LRW (c) Results by Proposed-LRW

Fig. 6. Visual comparison (Segmentation accuracy)

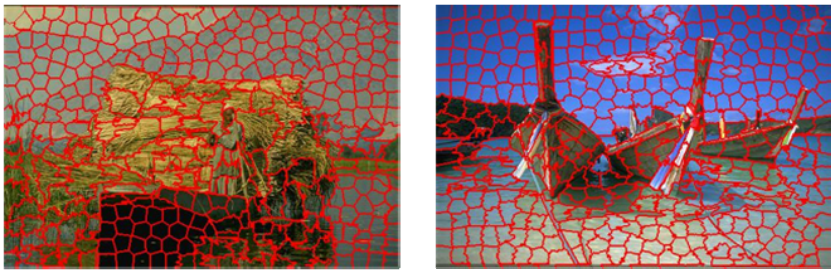
On the other hand, the shapes of superpixels using the SLIC method are much irregular, and the segmentation results using the original LRW method shows less object boundary adherence, compared with our proposed method.

In addition, it is clear to see that the sizes of superpixels using our method can match the region complexity much better. In other words, the sizes of superpixels can be larger for regions with less details, and the sizes get smaller for regions with richer texture details. An ideal situation of segmentation is that the size of segments is adapted to the actual texture complexity of the image while fulfilling the homogeneousness and superpixel quantity requirement by an application. As a pre-processing step of other applications, the segmentation results are able to represent regional homogeneity with appropriate shape and size, which may improve the computation efficiency in the subsequent missions.

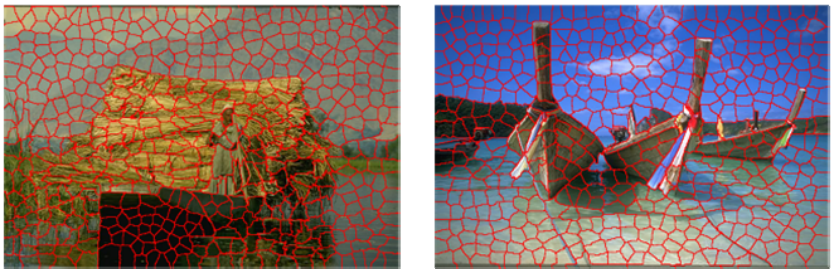
4.2 Results of Objective Evaluation

In order to evaluate the objective quality, we have applied three benchmark metrics mentioned in the section 2.4 to our proposed method and the compared methods: the boundary recall (BR), the undersegmentation error (UE) and the achievable segmentation accuracy (ASA).

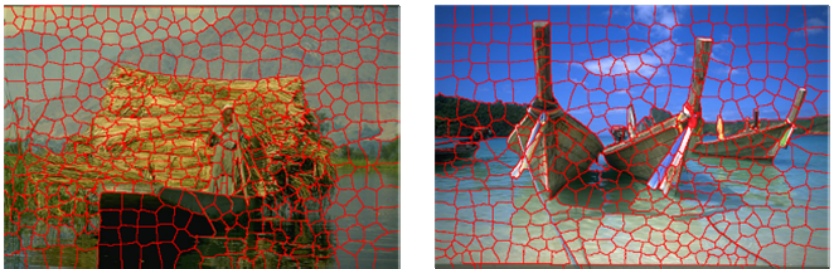
We firstly evaluate the boundary adherence objectively in the Fig. 8. SLIC has the best BR value since the boundary has adapted the texture complexity with more irregular segmentation contours. As for our proposed method, we have outperformed the original-LRW on the objective evaluation of boundary adherence.



(a) Results by SLIC



(b) Results by Original-LRW



(c) Results by Proposed-LRW

Fig. 7. Visual comparison (texture adaption)

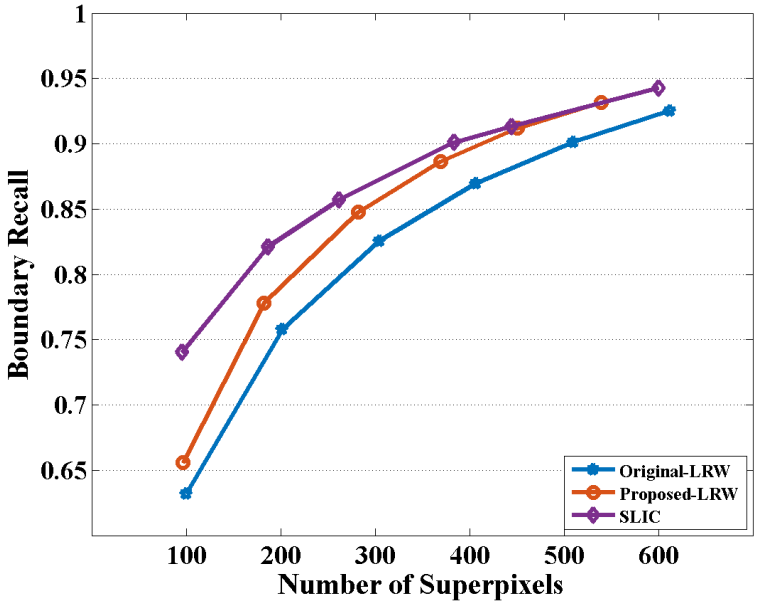


Fig. 8. Benchmark evaluation by Boundary Recall (BR)

In Fig. 9, our method has the lowest UE values compared with SLIC and the original LRW method. It means that our method has the lowest possible segmentation error. The segmentation accuracy has been improved under our proposed strategies, especially for larger number of superpixels. In the Fig. 10, the ASA values of our method outperform the others when the needed quantity of superpixels grows. For both of evaluations, our results have better outcome when the segmentation quantity increases.

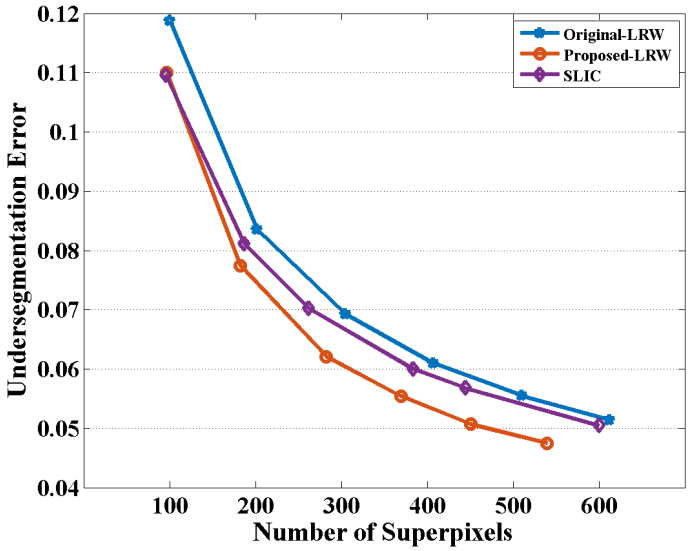


Fig. 9. Benchmark evaluation by Undersegmentation Error (UE)

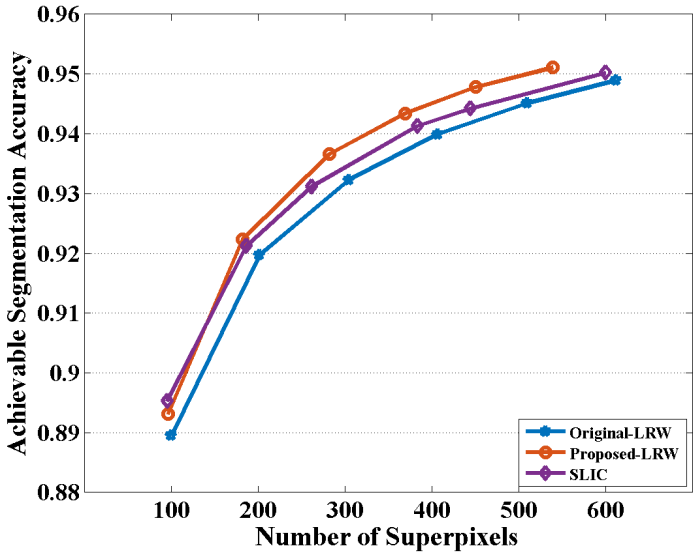


Fig. 10. Benchmark evaluation by Achievable Segmentation Accuracy (ASA)

Currently, our method is implemented on Matlab. To finish superpixel segmentation of an image to produce 500 superpixels, it takes around several hundred seconds on a personal computer with CPU i6700 and 16G RAM to reach convergence. We noticed that our method requires large memory space for part of the calculation and it takes time for context switching when it is out of memory. Therefore, efficient programming implementation and larger memory shall be quite helpful in the future.

5 Conclusion

In this paper, we propose an improved superpixel segmentation approach using a lazy random walk algorithm. Through the better center relocation and improved splitting strategy, the sizes of the produced

superpixels can adapt to texture complexities of different regions while preserving the boundary adherence and better segmentation precision. The improved results are verified both subjectively and objectively. For a long-term vision, the temporal concept of video data is also a considerable feature to extend the improvements into the supervoxel segmentation.

References

[1] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, S. Süsstrunk, SLIC Superpixels compared to state-of-the-art Superpixel methods, IEEE Transactions on Pattern Analysis and Machine Intelligence 34(11)(2012) 2274-2282.

[2] M. Liu, O. Tuzel, S. Ramalingam, R. Chellappa, Entropy rate superpixel segmentation, in: Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2011.

[3] A. Levinshtein, A. Stere, K.N. Kutulakos, D.J. Fleet, S.J. Dickinson, K. Siddiqi, TurboPixels: fast Superpixels using geometric flows, IEEE Transactions on Pattern Analysis and Machine Intelligence 31(12)(2009) 2290-2297.

[4] D. Yeo, J. Son, B. Han, J.H. Han, Superpixel-based tracking-by-segmentation using Markov chains, in: Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.

[5] T. Cour, J. Shi, Recognizing objects by piecing together the segmentation puzzle, in: Proc. IEEE Conference on Computer Vision and Pattern Recognition, 2007.

[6] B. Fulkerson, A. Vedaldi, S. Soatto, Class segmentation and object localization with superpixel neighborhoods, in: IEEE 12th International Conference on Computer Vision, 2009.

[7] A. Lucchi, Y. Li, K. Smith, P. Fua, Structured image segmentation using kernelized features, in: Proc. European Conference on Computer Vision, 2012.

[8] X. Ren, J. Malik, Learning a classification model for segmentation, in: Proc. Ninth IEEE International Conference on Computer Vision, 2003.

[9] A. Rosenfeld, D. Weinshall, Extracting foreground masks towards object recognition, in: Proc. International Conference on Computer Vision, Barcelona, 2011.

[10] J. Lim, B. Han, Generalized background subtraction using superpixels with label integrated motion estimation, in: Proc. European Conference on Computer Vision, 2014.

[11] L. Liu, J. Xing, H. Ai, S. Lao, Semantic superpixel based vehicle tracking, in: Proc. 21st International Conference on Pattern Recognition (ICPR), 2012.

[12] A. Milan, L. Leal-Taixé, K. Schindler, I. Reid, Joint tracking and segmentation of multiple targets, in: Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015.

[13] S. Wang, H. Lu, F. Yang, M.-H. Yang, Superpixel tracking, in: Proc. International Conference on Computer Vision, 2011.

[14] W. Wang, R. Nevatia, Robust object tracking using constellation model with superpixel, in: Proc. Asian Conference on Computer Vision, 2013.

[15] J. Xiao, R. Stolkin, A. Leonardis, Single target tracking using adaptive clustered decision trees and dynamic multilevel appearance models, in: Proc. IEEE Conference on Computer Vision and Pattern Recognition, 2015.

[16] Z. Hong, C. Wang, X. Mei, D. Prokhorov, D. Tao, Tracking using multilevel quantizations, in: Proc. European Conference on Computer Vision, 2014.

[17] J Shi, J. Malik, Normalized cuts and image segmentation, in: IEEE Transactions on Pattern Analysis and Machine Intelligence 22(8)(2000) 888-905.

[18] P.F. Felzenszwalb, D.P. Huttenlocher, Efficient Graph-Based Image Segmentation, Int. J. Comput. Vision 59(2)(2004) 167-181.

[19] A.P. Moore, S.J.D. Prince, J. Warrell, U. Mohammed, G. Jones, Superpixel lattices, in: Proc. IEEE Conference on Computer Vision and Pattern Recognition, Anchorage, 2008.

[20] O. Veksler, Y. Boykov, P. Mehrani, Superpixels and supervoxels in an energy optimization framework, in: Proc. 11th European conference on Computer vision: Part V (ECCV), 2010.

[21] J. Shen, Y. Du, W. Wang, X. Li, Lazy random walks for Superpixel segmentation, IEEE Transactions on Image Processing 23(4)(2014) 1451-1462.

[22] L. Vincent, P. Soille, Watersheds in digital spaces: an efficient algorithm based on immersion simulations, IEEE Transactions on Pattern Analysis & Machine Intelligence 13(6)(1991) 583-598.

[23] G. Zeng, P. Wang, J. Wang, R. Gan, H. Zha, Structure-sensitive superpixels via geodesic distance, in: Proc. International Conference on Computer Vision, 2011.

[24] M. Van den Bergh, X. Boix, G. Roig, B. de Capitani, L. Van Gool, SEEDS: Superpixels extracted via energy-driven sampling, in: Proc. European Conference on Computer Vision, 2012.

[25] X. Pan, Y. Zhou, C. Zhang, Q. Liu, Flooding based superpixels generation with color, compactness and smoothness constraints, in: Proc. IEEE International Conference on Image Processing (ICIP), 2014.

[26] Y. Zhang, X. Li, X. Gao, C. Zhang, A simple algorithm of superpixel segmentation with boundary constraint, IEEE Transactions on Circuits and Systems for Video Technology 27(7)(2017) 1502-1514.

[27] D. Stutz, A. Hermans, B. Leibe, Superpixel segmentation using depth information, [dissertation] Aachen, Germany: RWTH Aachen University, 2014.

[28] D.R. Martin, C.C. Fowlkes, J. Malik, Learning to detect natural image boundaries using local brightness, color, and texture cues, IEEE Transactions on Pattern Analysis & Machine Intelligence 26(5)(2004) 530-549.

[29] P. Neubert, P. Protzel, Superpixel benchmark and comparison, in: Proc. Forum Bildverarbeitung, 2012.

[30] X. Ren, J. Malik, Learning a classification model for segmentation, in: Proc. Ninth IEEE International Conference on Computer Vision, 2003.

[31] L. Grady, Random walks for image segmentation, IEEE Transactions on Pattern Analysis & Machine Intelligence 28(11)(2006) 1768-1783.

[32] V. Gopalakrishnan, Y. Hu, D. Rajan, Random walks on graphs for salient object detection in images, IEEE Transactions on Image Processing 19(12)(2010) 3232-3242.

[33] J. Ham, D.D. Lee, S. Mika, B. Schölkopf, A kernel view of the dimensionality reduction of manifolds, in: Proc. Twenty-First International Conference On Machine Learning, 2004.

[34] D. Zhou, B. Schölkopf, Learning from labeled and unlabeled data using random walks, in: Proc. Joint Pattern Recognition Symposium, 2004.

[35] T. Ojala, M. Pietikainen, T. Maenpaa, Multiresolution gray-scale and rotation invariant texture classification with local binary patterns, IEEE Transactions on Pattern Analysis & Machine Intelligence 24(7)(2002) 971-987.

[36] P. Arbelaez, M. Maire, C. Fowlkes, J. Malik, Contour detection and hierarchical image segmentation, IEEE Transactions on Pattern Analysis and Machine Intelligence 33(5)(2010) 898-916.