Content-Adaptive Superpixel Segmentation

Xiaolin Xiao, Yicong Zhou, *Senior Member, IEEE*, and Yue-Jiao Gong, *Member, IEEE*

***Abstract*— Superpixel segmentation targets at grouping pixels in an image into atomic regions whose boundaries align well with the natural object boundaries. This paper first proposes a new feature representation for superpixel segmentation that holistically embraces color, contour, texture, and spatial features. Then, we introduce a clustering-based discriminability measure to iteratively evaluate the importance of different features. Integrating the feature representation and the discriminability measure, we propose a novel content-adaptive superpixel (CAS) segmentation algorithm. CAS is able to automatically and iteratively adjust the weights of different features to fit various properties of image instances. Experiments on several challenging datasets demonstrate that the proposed CAS outperforms the state-of-the-art methods and has a low computational cost.**

***Index Terms*— Content-adaption, discriminability, preprocessing, superpixel.**

I. Introduction

SUPERPIXEL segmentation produces atomic regions of pixels (namely, the "superpixel"s) that are perceptually consistent. The concept of superpixel was firstly introduced in [1], in which the idea of over-segmentation was adopted as a pre-processing step for image segmentation. Illustrated in Fig. 1, unlike the traditional rigid pixel representation of images, superpixels provide visually meaningful entities that can be utilized as atomic units for image processing and computer vision tasks. The prominent advantage of using superpixels instead of pixel representation is the reduction in computational cost for subsequent processing. Some typical applications of superpixel segmentation include image segmentation [2], [3], object recognition [4], object tracking [5]–[7], image parsing [8], [9], 3D reconstruction [10], [11], depth prediction [12], and salient object detection [13]–[15].

For different applications, the generation of superpixels needs to satisfy specific requirements. Although it is hard to define universal criteria for all superpixel segmentation algorithms, the generally accepted performance metrics are:

- The boundaries of superpixels should adhere well to the object boundaries such that pixels on the object boundaries are recalled as many as possible.
- The boundaries of superpixels should not wing across different objects in the image.
- Superpixels should share similar sizes and regular shapes. This metric guarantees perceptual consistency and facilitates the subsequent process, e.g., feature extraction.
- Computational efficiency is also a crucial issue since superpixel segmentation is usually used for preprocessing.

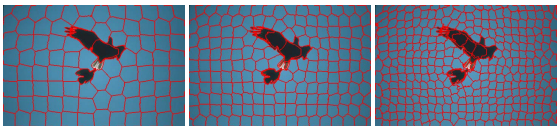

Fig. 1. Images segmented into 100/200/400 superpixels using the proposed CAS algorithm.

Existing superpixel segmentation algorithms have different priorities. Examples place emphasis on boundary adherence [16]–[18], time efficiency [19], [20], superpixel regularity [21], topology preserving [22]–[24], and many others. To the best of our knowledge, the existing methods do not have consistently good performance in handling different types of image instances considering the following observations:

First, they suffer from inadequate feature representation. The state-of-the-art methods always rely critically on color feature to distinguish different pixels. However, the color feature is inadequate to represent the characteristics of color images in some cases. This will significantly affect the performance of the superpixel segmentation algorithms. Fig. 2 provides an example. Measuring only color difference in the low color contrast regions impose difficulty in separating the petals. Therefore, it would be useful to design a new feature representation that can embed more local characteristics of images such as contour and texture features.

Second, they do not include a reliable feature discriminability measure. Natural images vary significantly in their contents, such as the untextured or textured images shown in Fig. 3 (a) and (d). For these two images, we extract their color features in CIELAB color space and their texture features using the differential excitation part of Weber Local Descriptor (WLD) [25]. The histograms of these two features on Fig. 3 (a) are plotted in Fig. 3 (b) and (c), respectively. The eagles and

Manuscript received March 29, 2017; revised November 19, 2017 and January 5, 2018; accepted February 16, 2018. Date of publication February 28, 2018; date of current version March 21, 2018. This work was supported in part by the National Natural Science Foundation of China under Grant 61502542, in part by the Macau Science and Technology Development Fund under Grant FDCT/016/2015/A1, and in part by the Research Committee at the University of Macau under Grant MYRG2016-00123-FST. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Junsong Yuan. *(Corresponding author: Yue-Jiao Gong.)*

X. Xiao and Y. Zhou are with the Department of Computer and Information Science, University of Macau, Macau 999078, China (e-mail: shellyxiaolin@gmail.com; yicongzhou@umac.mo).

Y.-J. Gong is with the School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, China, and also with the Department of Computer and Information Science, University of Macau, Macau 999078, China (e-mail: gongyuejiao@gmail.com).

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

Digital Object Identifier 10.1109/TIP.2018.2810541

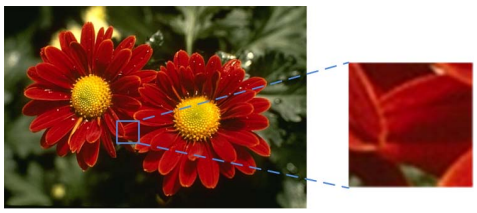

Fig. 2. Region that is hard to cluster based only on color distance.

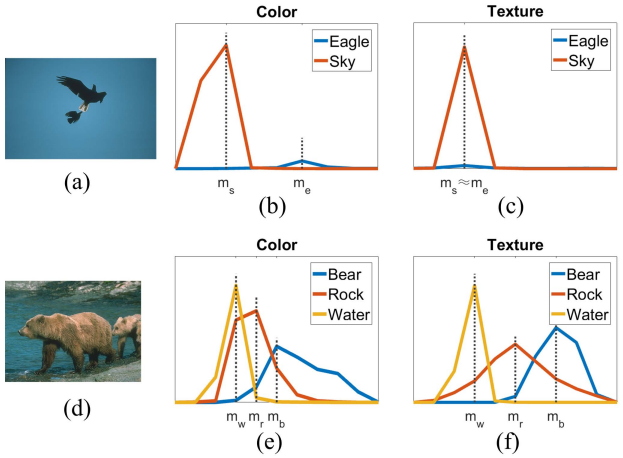

Fig. 3. The histograms of different features considering different image contents. (a) Eagle image (untextured). (b) Color histograms for different objects of the eagle image. (c) Texture histograms for different objects of the eagle image. (d) Bear image (textured). (e) Color histograms for different objects of the bear image. (f) Texture histograms for different objects of the bear image.

the sky can be well separated using the color feature, because their color histograms possess different modes (mode refers to the most frequent value in the data distribution). However, considering the histograms of the texture feature, their modes are similar, namely, $m_s \approx m_e$. It means that the values of the texture feature for different objects are close to each other in most cases. The texture feature hence provides little discriminative information for separating these two objects. As another example, the color and texture histograms for different objects of Fig. 3 (d) are depicted in Fig. 3 (e) and (f). In this example, the color feature is inadequate to separate the bears, the rocks, and the water because there is a large overlapping region on the color histograms for different objects. On the other hand, the texture feature is helpful to separate the bears, the rocks, and the water, since the modes of their texture histograms differ a lot. From the above comparison, we can notice that, for different images or objects, the histograms of features vary significantly. Therefore, the contributions of different features to the clustering operation differ a lot. Considering this, we propose a clustering-based feature discriminability measure to iteratively evaluate the importance of different features so that we can always rely on features with good discriminability, while reducing the effects of features with poor discriminability.

In addition, existing methods use fixed settings of parameters and operations for different input images during the segmentation processes. Several superpixel segmentation algorithms select their parameters manually to balance different performance metrics [17], [19]. Therefore, they have difficulty in obtaining the optimal performance for images with various contents.

This work aims to solve the above problems, and our contributions are listed as follows:

- We propose a new feature representation for superpixel segmentation that utilizes more robust local characteristics of images.
- We introduce a clustering-based discriminability measure to iteratively evaluate the importance of different features for each specific input image.
- Using the proposed feature representation and discriminability measure, we further design a content-adaptive superpixel (CAS) segmentation algorithm.
- Extensive experiments on several challenging datasets demonstrate that CAS outperforms the state-of-the-art methods.

In the rest of this paper, Section II reviews the state-of-the-art superpixel segmentation algorithms. Section III and Section IV pose a new feature representation and a reliable feature discriminability measure, respectively. Based on that, in Section V, we elaborate the implementation of CAS. In Section VI, we compare CAS with several state-of-the-art methods. Finally, conclusions are drawn in Section VII.[1]

II. Related Work

Superpixel segmentation algorithms can be generally classified into two categories [19], namely, graph-based methods and gradient-based methods.

A. Graph-Based Superpixel Segmentation Methods

For the graph-based methods, an image is represented by a graph containing vertices and edges. Each vertex corresponds to a pixel on the image, while an edge defines a connected pair of vertices. The graph is finally partitioned into a few disjoint sub-graphs (superpixels). These methods can be further divided into the following classes:

(1) Graph-cut-based: Superpixel segmentation can be achieved through minimizing the cost of cuts on the image graph. The most famous graph-cut-based method is the Normalized Cuts (NC) algorithm [27]. NC is well known for its capability of producing superpixels with good boundary adherence and regular shapes. However, the computational complexity for NC is extremely high. Entropy Rate Superpixel (ERS) [16] calculates the entropy rate from the cut costs on the graph, together with a balance term, to generate superpixels. Its drawback lies in the irregular shapes of superpixels.

(2) Boundary evolution: Instead of assigning optimal labels to pixels, superpixel Lattices (Lattice) [28] and Superpixels Extracted via Energy-Driven Sampling (SEEDS) [20] solve the segmentation problem by seeking the optimal boundaries

[1]The proposed CAS algorithm has a preliminary conference version [26]. In this fully developed journal paper, we have made significant improvements in the experimental validation and also refined some algorithmic components, which will be summarized in the main body of this paper.

of superpixels. Lattice generates superpixels by adding vertical and horizontal boundaries incrementally, while SEEDS iteratively evolves superpixel boundaries using a hill-climbing-based method.

(3) Energy optimization: Algorithms in this category obtain their optimum results by designing different energy functions. An energy minimization framework is formulated in [29], and further extended in [30]–[32], emphasizing either on regular shapes (EOpt0) or on better boundary adherence (EOpt1). Nevertheless, these algorithms have no explicit control over the number of superpixels, and their computational costs are relatively high. A real-time Coarse-to-Fine Topology-Preserving (CFTP) superpixel segmentation method is proposed in [23]. The energy function of CFTP is formulated using Markov random field. Besides, in [33], lazy random walk is exploited to construct the energy function for superpixel segmentation.

B. Gradient-Based Superpixel Segmentation Methods

For the gradient-based methods, pixels are iteratively clustered along the directions that the gradients change most quickly, and finally they are grouped into superpixels. These schemes are generally grouped into the following types:

(1) Mode seeking: Mean shift [34], [35], medoid shift [36], and Quick Shift (QS) [37] are examples of mode seeking methods. They generate superpixels via seeking the modes of the underlying superpixel densities. QS has relatively better performance and runs much faster than other two methods. The drawback of these mode seeking methods is the lack of compactness, resulting in irregular shapes of superpixels. Besides, they also have no explicit control over the superpixel numbers.

(2) Geodesic superpixel: Methods in this class are proposed considering the geodesic structures of images. Among them, TurboPixel (TP) [21] dilates superpixel seeds using a geometric-flow-based curve constraint, generating superpixels with regular shapes. However, TP endures low boundary adherence while it is time-consuming. The centroidal voronoi tessellation is computed to produce superpixels in (Mslic) [24].

(3) Clustering-based: VCells [38], Simple Linear Iterative Clustering (SLIC) [19] and Linear Spectral Clustering (LSC) [17] are three representatives for clustering-based methods. They can produce superpixels with similar sizes and regular shapes. VCells makes use of the edge-weighted centroidal Voronoi tessellations to produce superpixels. The shapes of superpixels deform so that they can fit local structures of images. SLIC is time-efficient. LSC makes use of spectral clustering technique to attain a global optimal solution, resulting in further improvement on boundary adherence. Nevertheless, these three algorithms utilize only color and spatial features, and hence their performance is decreased when the color feature is inadequate.

III. Feature Representation

Existing superpixel segmentation methods depend critically on color and spatial features to guarantee perceptually consistent superpixels with compact shapes. As depicted in Fig. 2, it is hard to get an accurate segmentation result merely using these two features, especially in regions with low color contrast. In order to improve the separating ability, we propose a new feature representation for superpixel segmentation that incorporates more robust local characteristics of images. The proposed feature representation inherently embraces color, spatial, contour, and texture features to improve its processing ability in dealing with different image instances. Specifically, each pixel is represented by a seven dimension feature vector $p = [l, a, b, x, y, g, u]^T$, where $[l, a, b]^T$ measures the color property of a pixel, $[x, y]^T$ is the spatial feature, g and u are the contour and texture features, respectively.

A. Color and Spatial Features

Many superpixel segmentation algorithms [17], [19] calculate color difference in CIELAB color space. This color space provides a distance measure to characterize the uniform changes of the human perceived colors using the Euclidean distance between two color pixels. Following this idea, we also exploit the CIELAB color space. The color feature in CIELAB space is represented by vector $[l, a, b]^T$, where l stands for lightness and a and b are the color dimensions. Given two feature vectors $\boldsymbol{p}_q = [l_q, a_q, b_q, x_q, y_q, g_q, u_q]^T$ and $\boldsymbol{p}_r = [l_r, a_r, b_r, x_r, y_r, g_r, u_r]^T$, the differences of $\boldsymbol{p}_q$ and $\boldsymbol{p}_r$ considering feature l, a, and b are calculated as follows:

$$d_l(\boldsymbol{p}_q, \boldsymbol{p}_r) = |l_q - l_r|. \tag{1}$$

$$d_a(\boldsymbol{p}_q, \boldsymbol{p}_r) = |a_q - a_r|. \tag{2}$$

$$d_b(\boldsymbol{p}_q, \boldsymbol{p}_r) = |b_q - b_r|. \tag{3}$$

In addition to the color feature, spatial feature is used to enforce compactness of superpixels. The spatial feature is represented by $[x, y]^T$, where x and y are the vertical and horizontal coordinates of a pixel on the image. The spatial distance between $\boldsymbol{p}_q$ and $\boldsymbol{p}_r$ is defined as:

$$d_s(\boldsymbol{p}_q, \boldsymbol{p}_r) = \sqrt{(x_q - x_r)^2 + (y_q - y_r)^2}. \tag{4}$$

B. Local Contour Feature

In particular, when pixels share extremely similar colors, the color and spatial features are inadequate to distinguish them. To solve this problem, our new feature representation exploits more robust local characteristics of images. An intuitive idea is to make use of the local contour feature to differentiate the pixels which are unable to be correctly clustered using merely the color and spatial features. In this paper, we exploit the image gradient to produce the contour feature g.

Image gradient measures the directional intensity changes of an image, and its magnitude is achieved by the square root of the sum of the squared directional signal changes. We represent image gradient by $\nabla\hat{g}$, and calculate it in lightness domain since human eyes are very sensitive to lightness changes. $\nabla\hat{g}$ and its magnitude are defined as:

$$\nabla\hat{g} = (\hat{g}_v, \hat{g}_h), \tag{5}$$

$$\|\nabla\hat{g}\| = \sqrt{\hat{g}_v^2 + \hat{g}_h^2}, \tag{6}$$

where the subscripts v and h denote vertical and horizontal directions, respectively, $\hat{g}_v$ and $\hat{g}_h$ denote the directional intensity changes, and $\|\cdot\|$ represents the magnitude. The approximation of $\|\nabla\hat{g}\|$ can be calculated using image filters, depicted in Fig. 4(a). Two thresholds, T_1 and T_2 ($T_1 < T_2$), are adopted to stratify the gradient values. Defined in Eq. (7), for $\|\nabla\hat{g}\|$ that is lower than T_1, the stratified gradient value is set to T_1; for $\|\nabla\hat{g}\|$ which is larger than T_2, we consider it to have equal effect on reflecting the contour feature and set the stratified gradient value to T_2; we leave the other gradient values as they are. Afterwards, T_1 is subtracted from the stratified gradient values to make the contour feature g start from zero. In our experiments, the two thresholds are set as $T_1 = 5$ and $T_2 = 15$.

$$g = -T_1 + \begin{cases} T_1, & \|\nabla\hat{g}\| < T_1 \\ T_2, & \|\nabla\hat{g}\| > T_2 \\ \|\nabla\hat{g}\|, & \text{otherwise} \end{cases} \tag{7}$$

The gradient difference between $\boldsymbol{p}_q$ and $\boldsymbol{p}_r$ is set to:

$$d_g(\boldsymbol{p}_q, \boldsymbol{p}_r) = |g_q - g_r|. \tag{8}$$

C. Local Texture Feature

Natural images may be either untextured or textured. For the untextured image, contour feature is more useful, whereas for the textured image, texture feature is more discriminating. Now, we use WLD to construct the texture feature u.

WLD [25] is a robust texture descriptor based on a psychology theorem called Weber's law:

$$\frac{\Delta S}{S} = \rho, \tag{9}$$

where ΔS denotes the minimum change of a signal that can be noticed by human eyes, and S is the intensity of the original signal. The Weber's law indicates that the ratio of ΔS over S is a constant ρ.

For image processing applications, WLD is designed to extract the local salient patterns in an image. We exploit the differential excitation (DE) part of WLD, which is denoted as u, to represent the texture feature. The DE part of WLD is computed as the ratio over two terms: the relative intensity difference of a current pixel against its neighbors, and the intensity of the current pixel. An arctan mapping is used to prevent u from changing rapidly as the input changes:

$$u = \arctan\left(\frac{v_0}{v_1}\right) = \arctan\left[\frac{\sum_{i=0}^{7}(x_i - x_c)}{x_c + \varepsilon}\right], \tag{10}$$

where x_c denotes the intensity of the current pixel; x_i ($i = 0, 1, \ldots 7$) represents the intensity of the ith neighbor around the current pixel, and thus v_0 and v_1 are the relative intensity difference and the current intensity, respectively. A small constant is ε added to x_c so as to avoid v_0 from being divided by zero. The texture distance between $\boldsymbol{p}_q$ and $\boldsymbol{p}_r$ is then formulated as:

$$d_u(\boldsymbol{p}_q, \boldsymbol{p}_r) = |u_q - u_r|. \tag{11}$$

Similarly to the magnitude of image gradient, the DE part of WLD can also be computed using filters, as shown in Fig. 4(b).

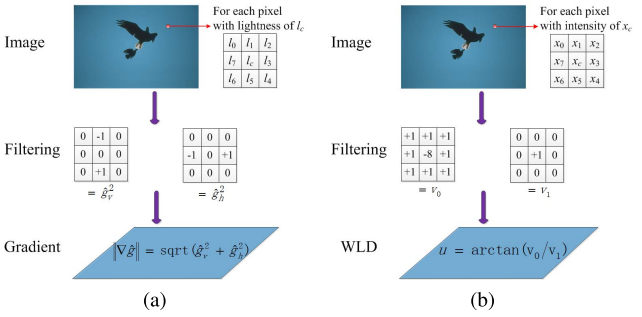

Fig. 4. Filters to compute $\|\nabla\hat{g}\|$ and u. (a) Gradient filters. (b) WLD filters.

IV. Clustering-Based Feature Discriminability Measure

It is inadequate to directly sum up the above feature distances without considering their importance. Previous clustering-based superpixel segmentation methods [17], [19] determine the importance of different features by manually setting fixed weight parameters for the involved features. However, on the one hand, it is a nontrivial work to set reasonable weights for different features (always tedious and time-consuming). On the other hand, these weights are rigid while the image contents vary greatly, and hence using unified weight parameters for all images may impose limitation on their performance. To overcome this limitation, we provide a clustering-based feature discriminability measure to exploit the property of specific image, then to adjust the operational focus accordingly.

A. Distance Measure

Combining the specific feature distances defined in Section III, the final distance $D(\boldsymbol{p}_q, \boldsymbol{p}_r)$ between two feature vectors $\boldsymbol{p}_q = [l_q, a_q, b_q, x_q, y_q, g_q, u_q]^T$ and $\boldsymbol{p}_r = [l_r, a_r, b_r, x_r, y_r, g_r, u_r]^T$ is calculated as:

$$D(\boldsymbol{p}_q, \boldsymbol{p}_r) = \sqrt{\sum_{j\in\delta} w_j^{\beta} d_j^2(\boldsymbol{p}_q, \boldsymbol{p}_r)}, \tag{12}$$

where $\delta = \{l, a, b, s, g, u\}$ represents the set of features, namely, lightness, color components a and b, spatial feature, contour feature, and texture feature; the spatial feature s is formulated from $[x, y]^T$; w_j is the weight parameter for feature j; β is an amplification factor for the weight parameters, and $\beta \geq 0$ is required so that the influence of w_j^{β} coincides with w_j; $d_j(\boldsymbol{p}_q, \boldsymbol{p}_r)$ is the distance between $\boldsymbol{p}_q$ and $\boldsymbol{p}_r$ considering feature j, and it can be calculated using Eqs. (1), (2), (3), (4), (8), (11), respectively.

B. Clustering-Based Feature Discriminability Measure

We propose a measure to evaluate the discriminability of different features based on the current partition of pixels. The principle is that the features with smaller sum of within-cluster variances are more discriminative, while the features with larger sum of within-cluster variances are less discriminative.

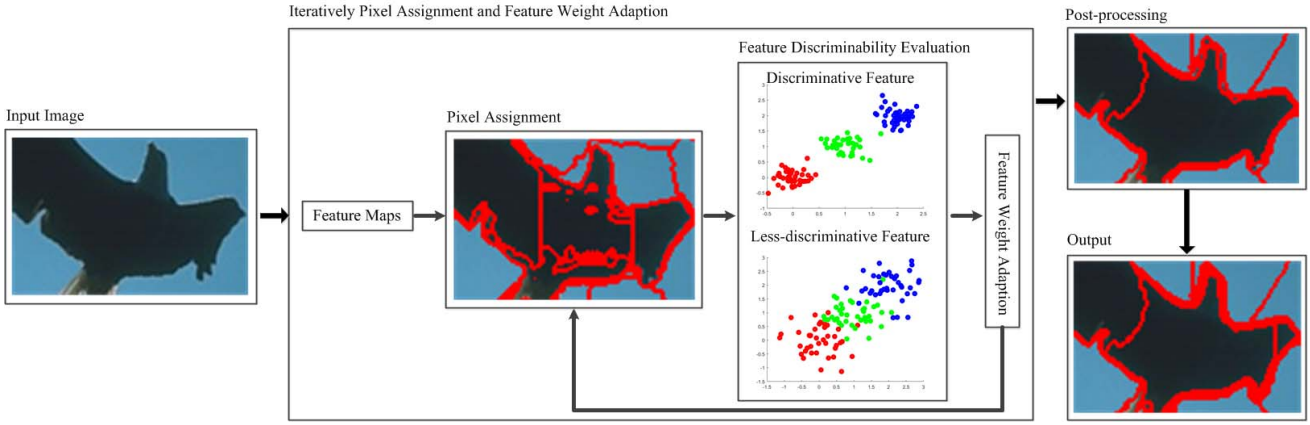

Fig. 5. Flow-chart of the proposed CAS algorithm.

We adopt SW_j to represent the sum of the within-cluster variance for feature j:

$$SW_j = \sum_{i=1}^{K}\sum_{q=1}^{N} \mathbb{1}(\boldsymbol{p}_q \in C_i) d_j^2(\boldsymbol{p}_q, \boldsymbol{c}_i), \tag{13}$$

where $j \in \{l, a, b, s, g, u\}$; K is the number of superpixels; N stands for the number of pixels; $\mathbb{1}()$ is a binary indicator, and $\mathbb{1}(\boldsymbol{p}_q \in C_i) = 1$ indicates that pixel $\boldsymbol{p}_q (q = 1, 2, \ldots, N)$ belongs to cluster $C_i (i = 1, 2, \ldots, K)$; $\boldsymbol{c}_i$ is the centroid for cluster C_i; and $d_j(\boldsymbol{p}_q, \boldsymbol{c}_i)$ measures the distance between pixel $\boldsymbol{p}_q$ and cluster center $\boldsymbol{c}_i$ on feature j.

As we can see from [17], [19], [21], [24], [34], [35], and [37], superpixels are required to be compact since they are used as atomic units to represent the image. Thus, in the clustering-based methods [17], [19], a pre-defined searching window is exploited to enforce local clustering. This way, the spatial feature is overwhelmed. Besides, the sum of within-cluster variance (SW_j) would be affected by the data range of different feature j, while the discriminability only concerns its clustering ability. To overcome these problems, we calculate the actual data range (R_j) of feature j on each specific image. Then we use R_j to normalize SW_j, resulting in the normalized within-cluster variance nSW_j:

$$nSW_j = \begin{cases} \dfrac{SW_j}{R_j^2}, & R_j \neq 0 \\ 0, & R_j = 0 \end{cases} \tag{14}$$

Note that $\{nSW_j\}_{j\in\delta}$ provides a reliable discriminability measure for different features considering their clustering ability.

V. Content-Adaptive Superpixel (CAS) Segmentation

Integrating the new feature representation and discriminability measure, we elaborate a content-adaptive superpixel (CAS) segmentation algorithm. Generally speaking, CAS is based on a modified linear clustering algorithm. It incorporates color, spatial, contour, and texture features into a holistic distance measure. And it is able to adapt the weights of different features considering the specific image contents.

A. Algorithmic Structure

The framework of CAS is illustrated in Fig. 5. Given an input image I, first, the color, spatial, contour, and texture features of each pixel are extracted. More particularly, we use the image gradient to compute the contour feature and WLD to generate the texture feature. Then, the discriminability of different features are evaluated using the proposed discriminability measure. The weights of different features are automatically adjusted based on their discriminability on the current partition of pixels.

The proposed CAS needs several iterations to obtain the final segmentation result. In each iteration, we first assign pixels to their nearest centers, generating a partition of the image instance. Then, based on the current partition, we calculate the discriminability of different features, and reset their weights accordingly. These weights will be updated in next iteration. The number of clusters is identical to the specified number of superpixels (K), and the initial cluster centers are uniformly sampled from a grid division of the image. Finally, the clustering process stops after a fixed number ($Iter$) of iterations. Another difference with the traditional K-means clustering is that the pixel assignment task in CAS is restricted in a pre-defined searching window, but not over the entire image. The purpose of such a constraint is to reduce the computational cost as well as to improve the compactness of superpixels.

Since the pixel-wise clustering cannot guarantee the connectivity inside superpixels, a post-processing step is needed. In CAS, we adopt the post-processing method introduced in [17] to get the final segmentation result.

B. Feature Weight Adaptation

In this section, we apply the proposed discriminabilty measure to automatically adapt the weights of different features in each iteration. The same symbols used in Section IV are adopted to formulate our problem. The objective is to segment images into K disjoint superpixels that each unit has consistent property inside itself. It can be fulfilled by minimizing energy E:

$$E = \sum_{i=1}^{K}\sum_{q=1}^{N}\sum_{j\in\delta} \mathbb{1}(\boldsymbol{p}_q \in C_i) w_j^{\beta} d_j^2(\boldsymbol{p}_q, \boldsymbol{c}_i) \tag{15}$$

subject to

$$\begin{cases} \sum_{i=1}^{K} \mathbb{1}(\boldsymbol{p}_q \in C_i) = 1, \ 1 \le q \le N & (16) \\ \sum_{j\in\delta} w_j^{\beta} = 1, \ 0 < w_j < 1 & (17) \end{cases}$$

The energy E can be optimized iteratively by solving three sub-problems:

- Pixel assignment problem (P_1): Fixing the cluster centers $\{\boldsymbol{c}_i\}_{i=1}^{K}$ and the feature weights $\{w_j\}_{j\in\delta}$, find the optimal pixel labels;
- Centroid adjustment problem (P_2): Fixing the feature weights $\{w_j\}_{j\in\delta}$ and the pixel labels, identify the optimal cluster centers $\{\boldsymbol{c}_i\}_{i=1}^{K}$;
- Weight adaption problem (P_3): Fixing the cluster centers $\{\boldsymbol{c}_i\}_{i=1}^{K}$ and the pixel labels, adapt the feature weights $\{w_j\}_{j\in\delta}$.

The pixel assignment problem (P_1) can be solved as:

$$\begin{cases} \mathbb{1}(\boldsymbol{p}_q \in C_i) = 1, & \text{if } \sum_{j\in\delta} w_j^{\beta} d_j^2(\boldsymbol{p}_q, \boldsymbol{c}_i) \le \sum_{j\in\delta} w_j^{\beta} d_j^2(\boldsymbol{p}_q, \boldsymbol{c}_\chi) \\ & \quad \text{for } \forall\chi, \quad 1 \le \chi \le K \\ \mathbb{1}(\boldsymbol{p}_q \in C_i) = 0, & \text{otherwise} \end{cases} \tag{18}$$

In the application of superpixel segmentation, we assign pixel labels locally to generate compact superpixels. This way, the assignment is not processed over the whole image, but in pre-defined local regions around current cluster centers. Hence, the solution of P_1 is slightly different from Eq. (18):

$$\begin{cases} \mathbb{1}(\boldsymbol{p}_q \in C_i) = 1, & \text{if } \ d_s^2(\boldsymbol{p}_q, \boldsymbol{c}_i) \le \tau^2 \ \text{ and} \\ & \quad \sum_{j\in\delta} w_j^{\beta} d_j^2(\boldsymbol{p}_q, \boldsymbol{c}_i) \le \sum_{j\in\delta} w_j^{\beta} d_j^2(\boldsymbol{p}_q, \boldsymbol{c}_\chi) \\ & \quad \text{for } \forall\chi, \quad 1 \le \chi \le K \\ \mathbb{1}(\boldsymbol{p}_q \in C_i) = 0, & \text{otherwise} \end{cases} \tag{19}$$

where $[\tau, \tau]$ is a pre-defined searching window. In our experiments, the size of the searching window, following [19], is set to four times of the averaged size of superpixel.

Following P_1, the centroid adjustment problem (P_2) is optimized to:

$$c_{i,j} = \frac{\sum_{i=1}^{K}\sum_{q=1}^{N} \mathbb{1}(\boldsymbol{p}_q \in C_i) F_{\boldsymbol{p}_q,j}}{\sum_{i=1}^{K}\sum_{q=1}^{N} \mathbb{1}(\boldsymbol{p}_q \in C_i)}, \tag{20}$$

where $F_{\boldsymbol{p}_q,j}$ represents the value of feature j for pixel $\boldsymbol{p}_q$; vector $\boldsymbol{c}_i = [c_{i,l}, c_{i,a}, c_{i,b}, c_{i,x}, c_{i,y}, c_{i,g}, c_{i,u}]$ represents the centroid of cluster C_i.

Finally, based on the solutions of P_1 and P_2, the weight adaption problem (P_3) can be optimized by adopting the proposed feature discriminability measure (Eqs. (13), (14)). Recall that, during the clustering process, the normalized within-cluster variance ($\{nSW_j\}_{j\in\delta}$) provides a reliable discriminability measure for different features. The features with smaller normalized within-cluster variances have similar values inside superpixels, and they are more helpful to group pixels with compact properties into superpixels. Our principle is that we always rely on the features with smaller normalized within-cluster variances, while alleviating the effects of the features with larger normalized within-cluster variances. Thus, w_j is set as:

$$w_j = \frac{1}{\sum_{t\in\delta} \left[\frac{nSW_j}{nSW_t}\right]^{\frac{1}{\beta-1}}} \tag{21}$$

When $\beta > 1$, the smaller nSW_j, the larger w_j. This way, the influence of features with small normalized within-cluster variances increases, and this phenomenon coincides with our principle for feature weight adaption. Therefore, $\beta > 1$ can be chosen for our task, and this parameter is empirically set to 7 in our experiments.

The detailed implementation of CAS is summarized in Algorithm 1. Given the initial weights of features, pixels are assigned to their nearest centers. Then, the discriminability of features is evaluated and the weights of features are adapted accordingly. In each iteration, the three sub-problems, namely, the pixel assignment problem, the centroid adjustment problem, and the weight adaption problem, are solved independently. The stopping criterion is set as a fixed number ($Iter$) of iteration. In our experiments, $Iter$ is set to 10. Once the stopping criterion is satisfied, the labels of pixels are further processed to enforce the connectivity inside superpixels.

C. Discussion

Our idea of generating content-adaptive superpixels has been published in a conference paper [26]. In this journal extension, we made some algorithmic refinements to improve the efficiency of the clustering distance calculation and the justifiability of the clustering-based discriminability measure.

First, instead of using the CIELCH color space in our conference paper, this manuscript uses the CIELAB color space to reduce the computational cost. This is because, when measuring the color difference of two pixels, some parameters should be pre-calculated according to the current color values in the CIELCH color space, while in the CIELAB color space, we directly use the weighted Euclidean distance. Moreover, in [26], the sum of within-cluster variances (SW_j, $j \in \{l, a, b, s, g, u\}$) are used to evaluate the discriminability of different features. However, the sum of within-cluster variances would be affected by the data ranges of different features, while the discriminability only concerns the clustering ability. Thus, we normalize the sum of within-cluster variances of different features using the square of the corresponding data ranges, generating the normalized within-cluster variances (nSW_j, $j \in \{l, a, b, s, g, u\}$). Then, we exploit the normalized within-cluster variances to adjust the weights of different features.

VI. Experiments

A. Datasets

The following four different datasets are used to evaluate the superpixel segmentation algorithms: (1) Berkeley

Algorithm 1 Content-Adaptive Superpixel (CAS) Segmentation

Input : Input image $\boldsymbol{I}$, Superpixel number K, and Iteration number $Iter$.
Output: Label matrix $\boldsymbol{L}$ of the input image.

```
Represent each pixel p by [l, a, b, x, y, g, u]^T.
Select K initial centers on the image grid.
Initialize pixel label L(p) = 0 for each pixel.
Initialize distance η(p) = ∞ recording the difference between each pixel and its nearest center.
Initialize equal weight parameters w_j = 1/6, j ∈ {l, a, b, s, g, u}.
repeat
    for each center c_i (i = 1, 2, ..., K) do
        for each pixel p that located inside the local window ([τ, τ]) of c_i do
            Calculate the distance D(p, c_i) between p and c_i using Eq. (12).
            if D(p, c_i) < η(p) then
                η(p) = D(p, c_i)
                L(p) = i
            end
        end
    end
    Update the cluster centers using Eq. (20).
    for each feature j do
        Calculate the normalized within-cluster variance nSW_j using Eqs. (13) and (14).
    end
    for each feature j do
        Adapt the weight parameter w_j for feature j using Eq. (21).
    end
until stopping criterion: iteration number=Iter;
Merge unconnected superpixels to their most similar neighbors.
```

Segmentation Dataset 500 (BSDS500): BSDS500 is the most widely used evaluation dataset for superpixel segmentation methods. It consists of 500 images with the contents ranging from outdoor scenes, landscapes, buildings, animals to humans; (2) PASCAL Visual Object Classes 2007 (VOC2007): VOC2007 is a standard image dataset for object recognition, classification, and segmentation tasks. We use the segmentation subset which contains 632 images of different objects like person, animals, vehicles, and indoor objects; (3) PASCAL-S: This dataset includes a subset of 850 images selected from PASCAL 2010; (4) Weizmann Segmentation Dataset (WSD): WSD provides an empirical and scientific basis for image segmentation. The dataset is specially designed to avoid potential ambiguities by only including images that clearly depict one or two object/s in the foreground.

B. Evaluation Metrics

To formulate the evaluation metrics, in the following statement, s_i $(i = 1, 2, \ldots, m)$ represents the ith superpixel, g_j $(j = 1, 2, \ldots, n)$ represents the jth grountruth segment, $|\cdot|$ stands for the size of a pixel set or a segment, and $\|\cdot\|$ measures the Euclidean distance. For a comprehensive evaluation, the common metrics [17], [19] for superpixel segmentation are used in this work:

Boundary Recall (BR) evaluates boundary adherence using the percentage of ground truth boundaries that are recalled by the superpixel boundaries:

$$BR = \frac{\sum_{p \in B(g_j)} \mathbb{1}\big(\min_{q \in B(s_i)} \|p - q\| \leq \epsilon\big)}{\sum_{j=1}^{n} |B(g_j)|}, \tag{22}$$

where $B(s_i)$ and $B(g_j)$ represent the sets of pixels for the boundaries of superpixel and the boundaries of ground truth segment, respectively, $\mathbb{1}()$ is an indicator function to check whether the nearest pixels among $B(s_i)$ and $B(g_j)$ are within ϵ-distance of pixels and ϵ is set to 2.

Undersegmentation Error (UE) is another evaluation metric for boundary adherence. It is based on the requirement that each superpixel belongs to only one object. UE measures the percentage of pixels inside superpixels that are leaked from the ground truth segments. If a superpixel has valid overlap with more than one ground truth segment, UE will increase accordingly.

$$UE = \frac{\sum_{j=1}^{n} \sum_{i=1}^{m} \mathbb{1}\big(|s_i \cap g_j| > \kappa \cdot |s_i|\big) \cdot |s_i| - \sum_{j=1}^{n} |g_j|}{\sum_{j=1}^{n} |g_j|}, \tag{23}$$

where the indicator function $\mathbb{1}()$ equals one for a valid overlap, and $\kappa = 0.05$ is the threshold.

Achievable Segmentation Accuracy (ASA) measures the maximum segmentation performance that can be achieved using superpixels as atomic units. To fulfill this goal, we label each superpixel with the label of the ground truth segment that has the largest overlap with the current superpixel. Then, ASA is set as the ratio between the total number of correctly labeled pixels and the number of all image pixels:

$$ASA = \frac{\sum_{i=1}^{m} \max_{j} |s_i \cap g_j|}{\sum_{j=1}^{n} |g_j|}. \tag{24}$$

The above three metrics are commonly used in the literature of superpixel segmentation. In addition, this paper presents a method to test the Intersection over Union metric, which is defined as follows.

Intersection over Union (IoU), also known as Jaccard index, compares the similarity and diversity between the ground truth segments and the superpixel segments. As an effective approach for pre-processing, the superpixel segments should be highly consistent with the ground truth segments. Again, we label each superpixel with the label of the ground truth segment that has the largest overlap with the current superpixel. Then, we merge all superpixel segments that have

the same labels. The tth $(t = 1, 2, \ldots, n)$ merged segment is denoted by:

$$M_t = \bigcup_{i=1}^{m} \mathbb{1}\left(t = \arg\max_j |s_i \cap g_j|\right) \cdot s_i, \tag{25}$$

where the indicator function $\mathbb{1}()$ equals one when the ground truth segment that has the largest overlap with s_i is g_t.

The IoU score for the tth merged segment is defined as the size of the intersection over the size of the union between M_t and g_t:

$$IoU_t = \frac{|M_t \cap g_t|}{|M_t \cup g_t|}. \tag{26}$$

Finally, the IoU score for an image is calculated as the weighted average of IoU_t according to the size of g_t:

$$IoU = \frac{\sum_{t=1}^{n} |g_t| \cdot IoU_t}{\sum_{t=1}^{n} |g_t|}. \tag{27}$$

Note that, if the image has only one ground truth segment, BR, ASA, and IoU will always equal to one, while UE equals to zero. Therefore, to evaluate the performance of different algorithms for superpixel segmentation, we assume that there are at least two ground truth regions per image, namely, $n \geq 2$.

Runtime is an important evaluation metric because superpixel segmentation is usually used as a pre-processing step so that the algorithm should be time-efficient.

C. Performance of Feature Discriminability Measure

Firstly, we compare CAS with its degenerative version (DCAS) that does not exploit the feature discriminability measure. We present both quantitative and visual comparisons to demonstrate the effectiveness of the proposed feature discriminability measure.

1) Quantitative Comparisons: Fig. 6 illustrates the statistical comparisons between CAS and DCAS on the BSDS500 dataset. The number of superpixels, K, in each test image, is set to 100, 200, ... 1000, respectively. It can be observed that BR, UE, and ASA are all well improved, meanwhile, the computational cost of CAS is slightly higher than DCAS. The quantitative results are greatly improved by adopting the feature discriminability measure when K is small, and finally the performance of CAS and DCAS are getting closer as K increases.

2) Visual Comparisons: Fig. 7 depicts some visual comparisons between CAS and DCAS. For different image contents, DCAS does not consider the importance of different features, resulting in very similar boundaries on various image instances. Meanwhile, CAS takes the discriminability of different features into consideration, thus, it can always rely on the features that show good discriminability based on the current image partition, while reducing the unwelcome influence of using the features with relative poor discriminability. As a result, CAS produces almost straight superpixel boundaries on the first column, moderate twisting superpixel boundaries on the second column, and pretty twisting superpixel boundaries on the third column. The first image is quite clean without fine-grained structure, thus, both CAS and DCAS produce very straight superpixel boundaries. The second image has many slender branches such that the superpixel boundaries from CAS are twisting along these branches. In the meantime, the third image has rich structure and gradually color changes among different objects, and CAS also obtains better boundary adherence on the boundaries of the leaf and the grasshopper.

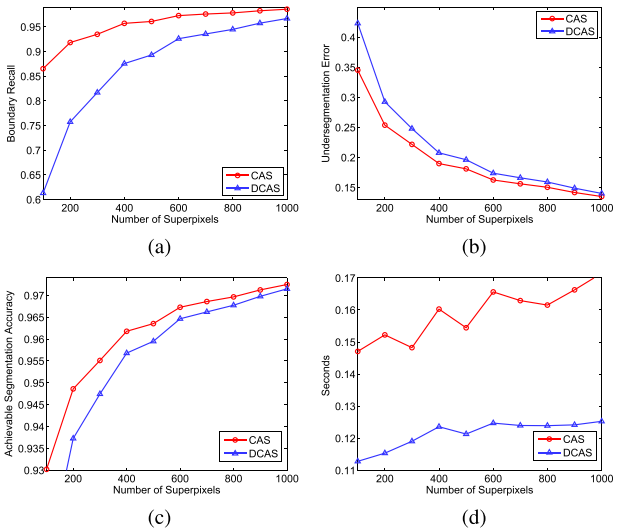

Fig. 6. Quantitative evaluation of CAS and DCAS on BSDS500. (a) Boundary Recall (BR). (b) Undersegmentation Error (UE). (c) Achievable Segmentation Accuracy (ASA). (d) Runtime.

Fig. 7. Visualization of CAS and DCAS on BSDS500 when $K = 100$. First column: image without fine-grained structure, straight superpixel boundaries; second column: image with slender branches, moderate twisting superpixel boundaries; third column: image with rich structure and gradually color changes, pretty twisting superpixel boundaries.

3) Exploration on Image Contents and the Adapted Clustering Weights: In Fig. 8, we provide examples of images and their final adapted clustering weights. For the ease of observation, we plot the relative weight values, namely, the proportion of weights to their corresponding data ranges. Generally speaking, for different images, the final clustering weights of different features produced by CAS vary significantly. This validates the effectiveness of the proposed method for feature weight adaptation: CAS is able to well adapt the weights of different features according to image contents and the clustering tasks. Particularly, we can see that Image 1 has simple structures and compact colors in local regions. Hence the superpixel segmentation of Image 1 is dominated by the

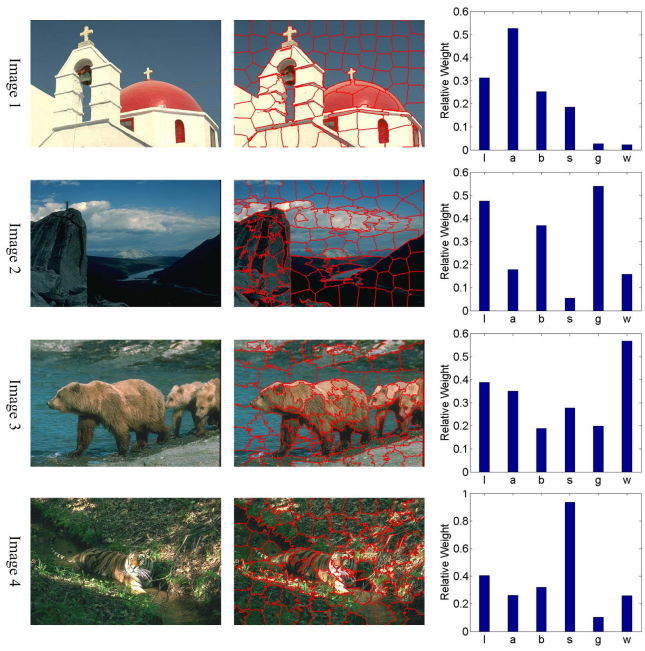

Fig. 8. Relative final clustering weights of different features on different images.

color features, while the other three features have very small weights. Considering Image 2, the scenery contains many mountains, rivers and clouds, and their boundaries should benefit the superpixel segmentation. As a result, the contour feature shows good discriminability and gains a large weight. The texture feature is helpful for Image 3, especially for correctly separating bears and rocks. Sometimes, none of the color, contour or texture features helps in generating compact superpixels, e.g., Image 4. In this case, the relative importance of the spatial feature increases.

D. Performance

In this section, CAS is compared with eleven state-of-the-art superpixel segmentation methods considering three aspects: (1) quantitative analysis in Section VI-D.1; (2) visual comparison in Section VI-D.2; and (3) pre-processing ability analysis in Section VI-D.3. The competing algorithms include CFTP [23], EOpt0 and EOpt1 [32], ERS [16], Lattice [28], LSC [17], Mslic [24], QS [37], SLIC [19], TP [21], and VCells [38]. For CAS, the parameters w_l, w_a, w_b, w_s, w_g, and w_u are initialized to $\frac{1}{6}$, and then adapted in each iteration. The parameters in the compared schemes are set according to the recommendations in the related references. All algorithms are tested in the same or equivalent computational environment. The following experimental results provide comprehensive comparisons to verify that CAS has better or equal performance with the state-of-the-art superpixel segmentation algorithms.

1) Quantitative Comparisons: Figs. 9-13 illustrate the quantitative comparisons of all compared superpixel segmentation algorithms on the four datasets introduced in Section VI-A in terms of BR, UE, ASA, ASA, IoU, and Runtime, respectively. Considering BR (Fig. 9) and UE (Fig. 10), CAS gains the best results, especially on the WSD dataset. CAS also achieves better results than the other competing algorithms in terms of ASA (Fig. 11) and IoU (Fig. 12). As for the Runtime indicator (Fig. 13), CAS is also competitive. It ranks as the second fastest methods, following SLIC.

In summary, CAS performs comparable or better than the other state-of-the-art superpixel segmentation algorithms in terms of BR, UE, ASA, and IoU, and it is the second fastest methods in our comparisons. For clarity, Table I shows a numerical comparison of different algorithms on the BSDS500 dataset when $K = 400$.

2) Visual Comparisons: We also provide visual comparisons of the superpixels generated from CAS, CFTP, ERS, Lattice, LSC, Mslic, QS, and SLIC from the four datasets when $K = 200$ since these algorithms have relatively better quantitative results.

It can be seen from Fig. 14 that CFTP, Mslic, and SLIC generate superpixels with the most regular shapes. The shapes of superpixels generated from LSC, QS, SLIC and CAS are less regular compared to the previous three algorithms, nevertheless, their shapes are much better than that generated from ERS. As shown in the flower picture, CFTP, Lattice, LSC, Mslic, SLIC, and VCells are, partly, unable to capture the boundaries of the flower petals. ERS, QS, and CAS have relatively good performance in preserving the boundaries of the flower petals. However, ERS and QS suffer from much twisting boundaries on the stamens of the flowers. For the cyclists picture, only CAS almost correctly segments the two guys on the background. In spite of the fact that the two guys have relative small sizes compared to the riders in the foreground, and relative low color contrast against the mountains in the background. For the family picture, CAS is the only one that correctly preserves the whole right hand of the daddy. Finally, for the handbag picture, CFTP, LSC, and CAS preserve the boundaries of the handbag from the background desk almost correctly, while the other algorithms mistake the boundaries of the black spots for the superpixel boundaries.

3) Pre-Processing Performance: As aforementioned, superpixel segmentation is commonly used as a pre-processing step in image segmentation applications and other related fields such as saliency detection and image parsing. Note that the use of superpixels instead of pixels should not decrease the performance of subsequent processing. In order to test the effectiveness of using superpixel segmentation as a pre-processing step, we assume using an ideal classifier on the superpixels. The ideal classifier can assign each superpixel to the ground truth segment that has the largest overlap with the current superpixel. This way, the obtained ideal segmentation results using superpixel representation are visualized in Fig. 15. We compare the performance of CAS with CFTP, ERS, Lattice, LSC, Mslic, QS, and SLIC when $K = 400$.

As shown in Fig. 15, for the stone statue image, CFTP, QS, SLIC, and CAS correctly preserve the boundaries of leaves; for the bird picture, CAS is the only one that well preserves the beak, the wings, and the paws; for the cat picture, SLIC and CAS are the best two in capturing the palms of the cat;

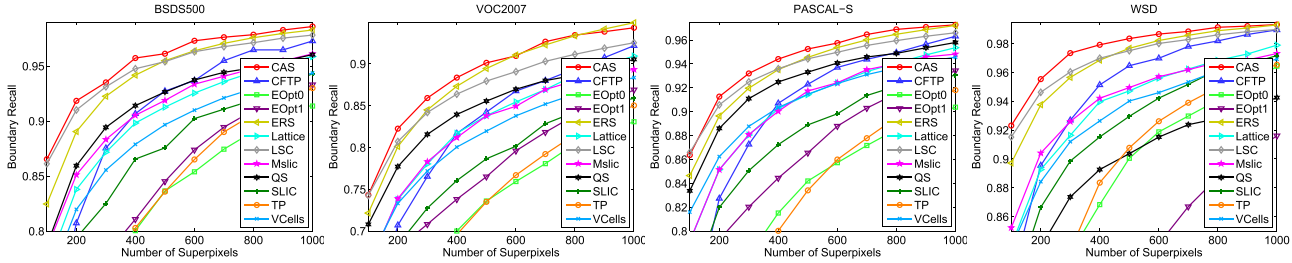

Fig. 9. Boundary Recall (BR) of different superpixel segmentation algorithms on four datasets.

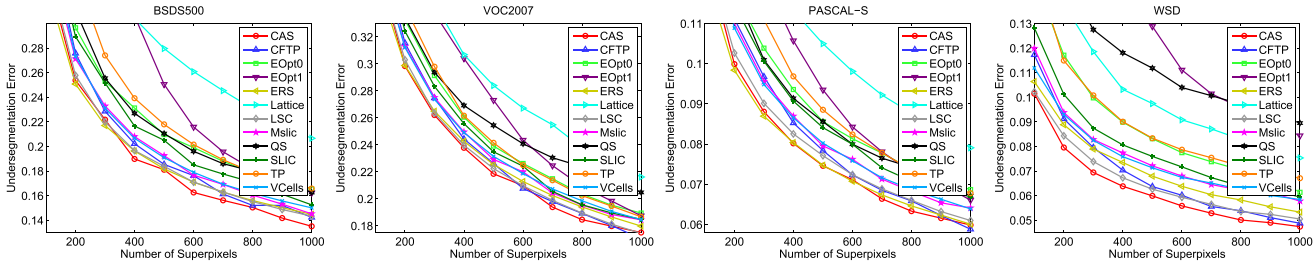

Fig. 10. Undersegmentation Error (UE) of different superpixel segmentation algorithms on four datasets.

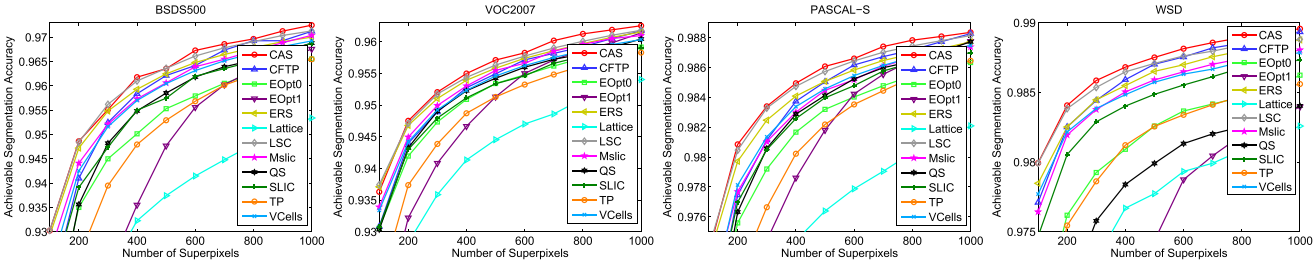

Fig. 11. Achievable Segmentation Accuracy (ASA) of different superpixel segmentation algorithms on four datasets.

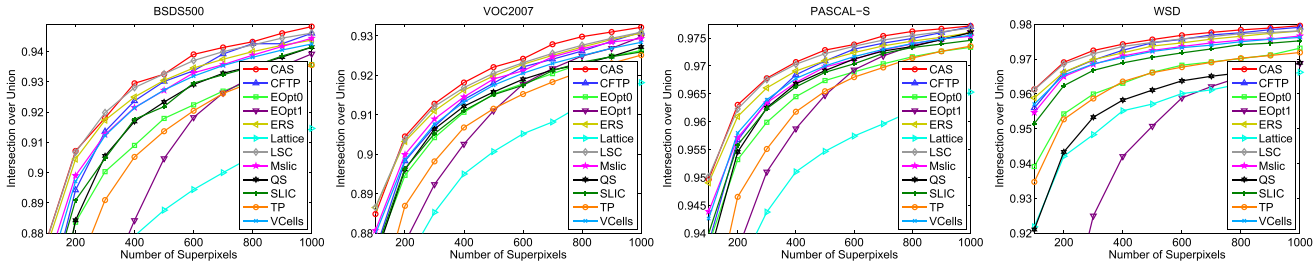

Fig. 12. Intersection over Union (IoU) of different superpixel segmentation algorithms on four datasets.

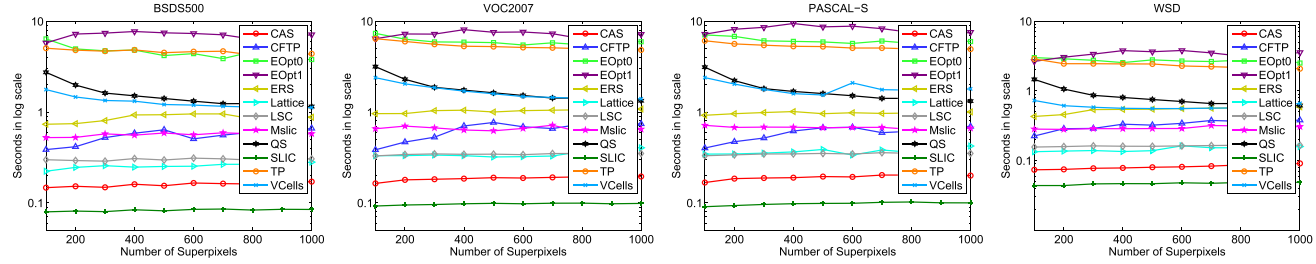

Fig. 13. Runtime of different superpixel segmentation algorithms on four datasets (plotted in log scale).

for the leaf picture, CAS has relatively good performance in preserving the long petiole.

4) Real-World Segmentation Application: To further validate the effectiveness of CAS, we employ a real-world cosegmentation application [39]. It adopts superpixel segmentation as a pre-processing step. We replace the original superpixel segmentation algorithm by CAS, ERS, LSC, and SLIC, individually. The reason of choosing ERS, LSC, and SLIC for comparison is that ERS and LSC have better quantitative results, while SLIC is the most widely used superpixel segmentation algorithm in practical applications. Following [39], a subset of the MSRC database [40] containing

TABLE I

AVERAGE PERFORMANCE METRICS OF DIFFERENT SUPERPIXEL SEGMENTATION ALGORITHMS ON BSDS500 WHEN $K = 400$

	CFTP	EOpt0	EOpt1	ERS	Lattice	LSC	Mslic	QS	SLIC	TP	VCells	CAS
BR	0.9067	0.8004	0.8108	0.9417	0.8982	0.9476	0.9052	0.9140	0.8655	0.8028	0.8789	**0.9580**
UE	0.2023	0.2313	0.3062	0.1964	0.3042	0.1967	0.2080	0.2269	0.2166	0.2393	0.2067	**0.1869**
ASA	0.9584	0.9501	0.9355	0.9593	0.9323	0.9610	0.9573	0.9548	0.9550	0.9479	0.9570	**0.9624**
IoU	0.9237	0.9090	0.8842	0.9249	0.9295	0.9281	0.9215	0.9169	0.9176	0.9052	0.9214	**0.9295**
Runtime	0.5847	4.8081	7.7012	0.9300	0.2477	0.3076	0.5572	1.5087	**0.0840**	4.8247	1.3180	0.1667

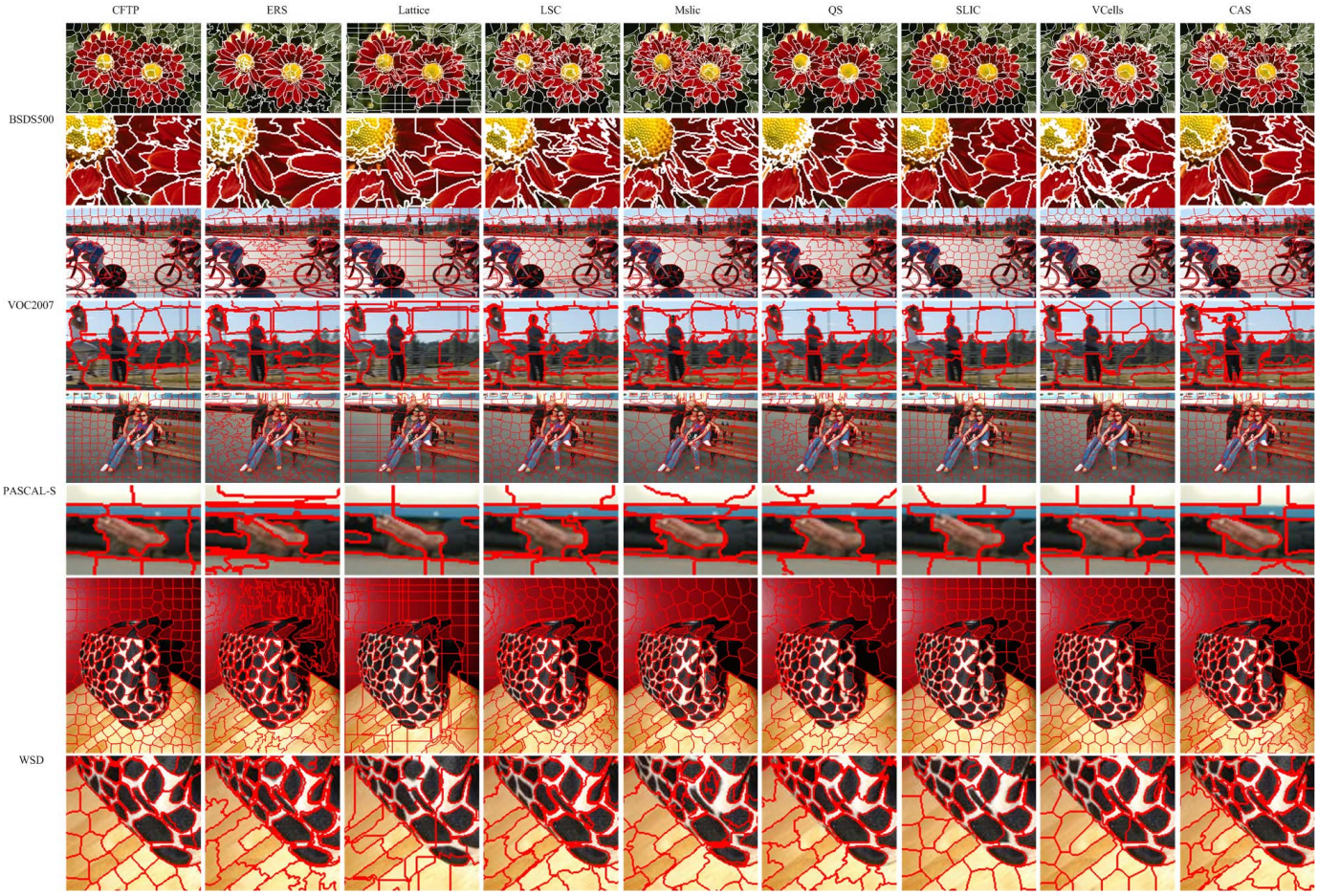

Fig. 14. Visualization of different superpixel segmentation algorithms on different datasets when $K = 200$.

TABLE II

SEGMENTATION ACCURACIES ON THE MSRC DATABASE BY ADOPTING DIFFERENT SUPERPIXEL SEGMENTATION ALGORITHMS FOR PRE-PROCESSING

	bike	bird	car	cat	chair	cow	dog	flower	house	sheep	Average
Method in [39]	0.8408	0.933	0.9108	0.9078	0.9166	**0.9767**	0.9354	0.9217	0.919	0.9417	0.9204
CAS	**0.8606**	**0.9559**	0.9198	**0.9393**	**0.9257**	0.9749	**0.9532**	0.9276	**0.9329**	**0.9487**	**0.9339**
ERS	0.8334	0.9518	0.9146	0.9281	0.91	0.9657	0.9486	0.9211	0.9179	0.9423	0.9233
LSC	0.8538	0.9484	0.9136	0.9292	0.9193	0.9654	0.9524	0.9253	0.924	0.9426	0.9274
SLIC	0.8484	0.9507	**0.9315**	0.9318	0.894	0.9604	0.9399	**0.9286**	0.92	0.9394	0.9245

10 classes is employed for our experiment. For each class, we use 10 images with scribbles for cosegmentation. The original superpixel segmentation algorithm in [39] is the mean shift algorithm [35], which is also included in our comparison. Given the default parameter settings in [39], the mean shift algorithm produces about 200-400 superpixels for images in the MSRC database. Considering CAS, ERS, LSC, and SLIC, we test their segmentation performance by setting the number of superpixels to 200, 300, and 400, respectively. The best average segmentation accuracies on different classes and the total average accuracies are reported in Table II. Examples of the segmented objects by adopting different superpixel

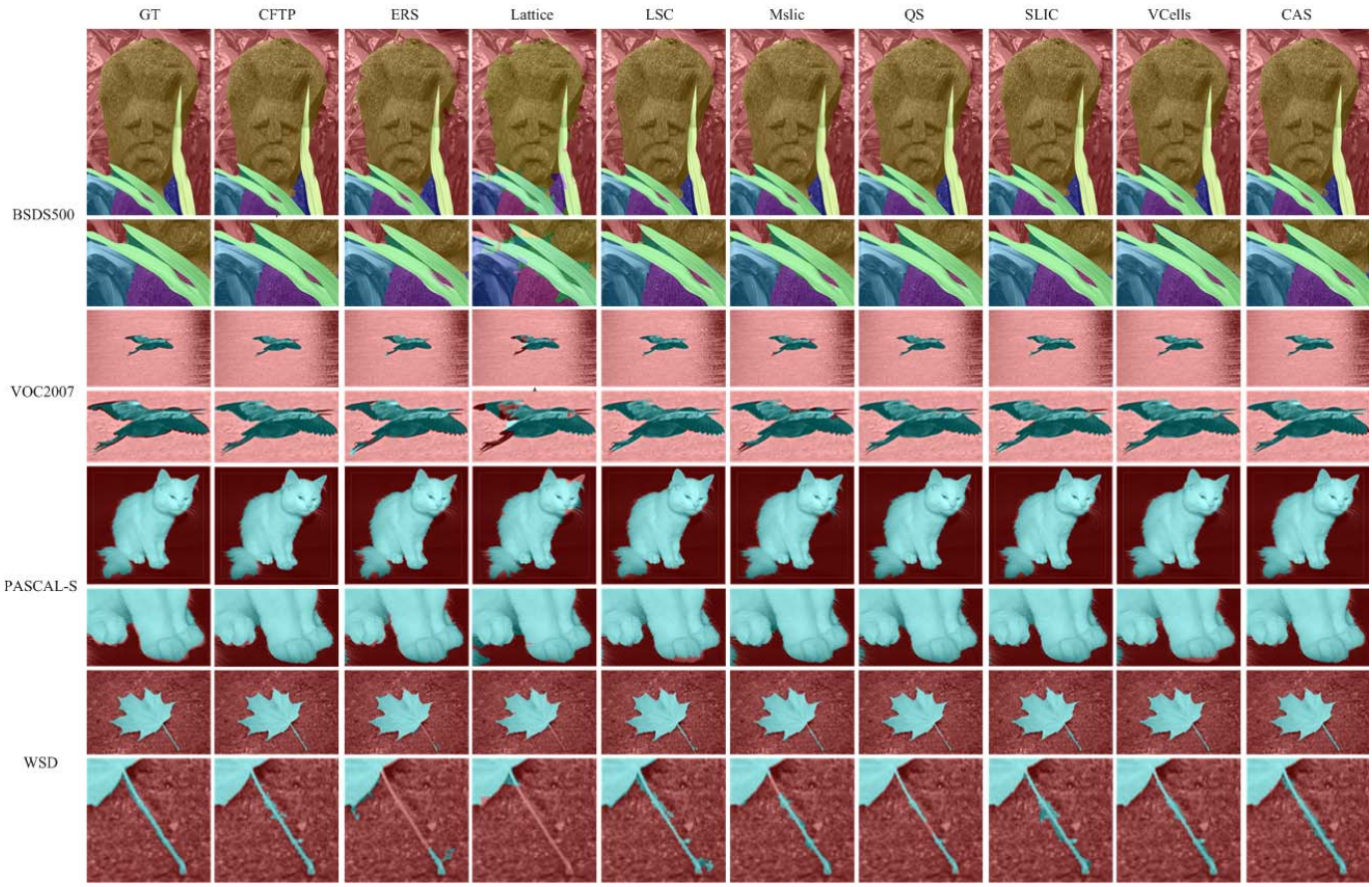

Fig. 15. Visualization of the maximum pre-processing performance using different superpixel segmentation algorithms on four datasets when $K = 400$.

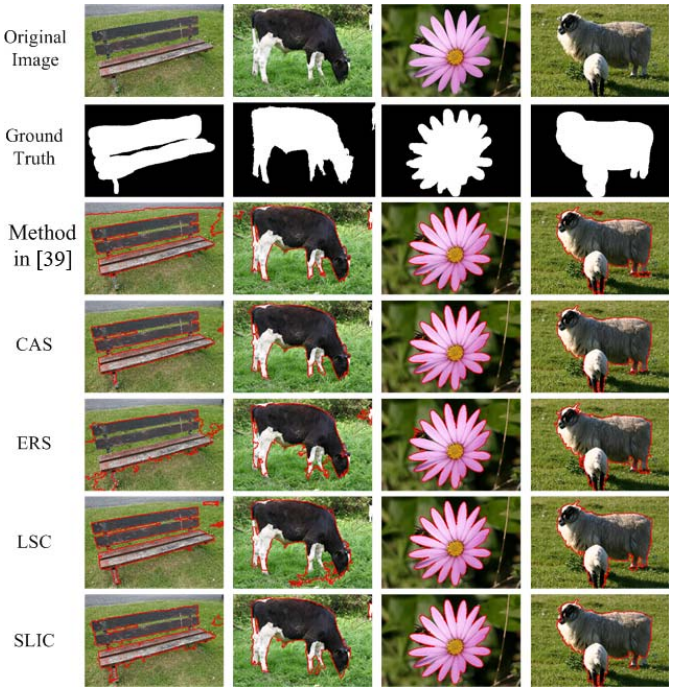

Fig. 16. Examples of the segmented objects on the MSRC database by adopting different superpixel segmentation algorithms for pre-processing.

segmentation algorithms for pre-processing are given in Fig. 16. Both the segmentation accuracies and the visual results show that CAS outperforms all competitors.

VII. Conclusion

In this paper, we developed a content-adaptive superpixel (CAS) segmentation method. The method adopts a new feature representation that utilizes color, contour, texture, and spatial features to present a robust characterization of the input image. We proposed a feature discriminability measure to evaluate the segmentation capability of each type of features for the image. Based on the feature representation and discriminability measure, a content-adaptive clustering algorithm is further designed for accomplishing the superpixel segmentation task. Particularly, we adjust the weights of different features in an iterative and adaptive way according to their discriminability on the current partition of pixels for different image instances, resulting in a more accurate and reasonable distance measure to discriminate pixels on natural images. Experimental results demonstrated the advantages of our algorithm over other state-of-the-art superpixel segmentation algorithms.

In the future, owing to the powerfulness of CAS, it is rather appealing to apply it to pre-process images in various computer vision tasks such that the effectiveness of applications can be improved. Besides, there are also opportunities to further improve the performance of CAS. For example, since images have different regional properties, it would be interesting to develop locally adaptive methods that assign different weights of features on different local regions of an image to fit the requirements from specific applications.

REFERENCES

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

[2] J. Malik, S. Belongie, T. Leung, and J. Shi, "Contour and texture analysis for image segmentation," *Int. J. Comput. Vis.*, vol. 43, no. 1, pp. 7–27, 2001.

[3] X. Wang, Y. Tang, S. Masnou, and L. Chen, "A global/local affinity graph for image segmentation," *IEEE Trans. Image Process.*, vol. 24, no. 4, pp. 1399–1411, Apr. 2015.

[4] C. Wang, Z. Liu, and S.-C. Chan, "Superpixel-based hand gesture recognition with Kinect depth camera," *IEEE Trans. Multimedia*, vol. 17, no. 1, pp. 29–39, Jan. 2015.

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

[6] S. Oron, A. Bar-Hillel, D. Levi, and S. Avidan, "Locally orderless tracking," *Int. J. Comput. Vis.*, vol. 111, no. 2, pp. 213–228, Jan. 2015.

[7] J. Xiao, R. Stolkin, and A. Leonardis, "Single target tracking using adaptive clustered decision trees and dynamic multi-level appearance models," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 4978–4987.

[8] J. Tighe and S. Lazebnik, "Superparsing," *Int. J. Comput. Vis.*, vol. 101, no. 2, pp. 329–349, Jan. 2013.

[9] S. Gupta, P. Arbeláez, R. Girshick, and J. Malik, "Indoor scene understanding with RGB-D images: Bottom-up segmentation, object detection and semantic segmentation," *Int. J. Comput. Vis.*, vol. 112, no. 2, pp. 133–149, 2015.

[10] A. Bódis-Szomorú, H. Riemenschneider, and L. Van Gool, "Superpixel meshes for fast edge-preserving surface reconstruction," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 2011–2020.

[11] T.-K. Huang, Y.-H. Wang, T.-K. Lin, and Y.-Y. Chuang, "A robust automatic object segmentation method for 3D printing," in *Proc. IEEE Int. Conf. Multimedia Expo*, Jul. 2016, pp. 1–6.

[12] P. Wang, X. Shen, Z. Lin, S. Cohen, B. Price, and A. Yuille, "Towards unified depth and semantic prediction from a single image," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 2800–2809.

[13] L. Zhu, D. A. Klein, S. Frintrop, Z. Cao, and A. B. Cremers, "A multisize superpixel approach for salient object detection based on multivariate normal distribution estimation," *IEEE Trans. Image Process.*, vol. 23, no. 12, pp. 5094–5107, Dec. 2014.

[14] J. Kim, D. Han, Y.-W. Tai, and J. Kim, "Salient region detection via high-dimensional color transform and local spatial support," *IEEE Trans. Image Process.*, vol. 25, no. 1, pp. 9–23, Jan. 2016.

[15] S. Li, H. Lu, Z. Lin, X. Shen, and B. Price, "Adaptive metric learning for saliency detection," *IEEE Trans. Image Process.*, vol. 24, no. 11, pp. 3321–3331, Nov. 2015.

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

[17] Z. Li and J. Chen, "Superpixel segmentation using linear spectral clustering," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 1356–1363.

[18] Y.-J. Gong and Y. Zhou, "Differential evolutionary superpixel segmentation," *IEEE Trans. Image Process.*, vol. 27, no. 3, pp. 1390–1404, Mar. 2018.

[19] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, "SLIC superpixels compared to state-of-the-art superpixel methods," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 11, pp. 2274–2282, Nov. 2012.

[20] M. Van den Bergh, X. Boix, G. Roig, B. de Capitani, and L. Van Gool, "SEEDS: Superpixels extracted via energy-driven sampling," in *Proc. Eur. Conf. Comput. Vis.*, 2012, pp. 13–26.

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

[22] Y. Zhou, L. Ju, and S. Wang, "Multiscale superpixels and supervoxels based on hierarchical edge-weighted centroidal Voronoi tessellation," *IEEE Trans. Image Process.*, vol. 24, no. 11, pp. 3834–3845, Nov. 2015.

[23] J. Yao, M. Boben, S. Fidler, and R. Urtasun, "Real-time coarse-to-fine topologically preserving segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 2947–2955.

[24] Y.-J. Liu, C.-C. Yu, M.-J. Yu, and Y. He, "Manifold SLIC: A fast method to compute content-sensitive superpixels," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 651–659.

[25] J. Chen *et al.*, "WLD: A robust local image descriptor," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 9, pp. 1705–1720, Sep. 2010.

[26] X. Xiao, Y.-J. Gong, and Y. Zhou, "Adaptive superpixel segmentation aggregating local contour and texture features," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, Mar. 2017, pp. 1902–1906.

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

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

[29] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 11, pp. 1222–1239, Nov. 2001.

[30] Y. Boykov and V. Kolmogorov, "An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 9, pp. 1124–1137, Sep. 2004.

[31] V. Kolmogorov and R. Zabin, "What energy functions can be minimized via graph cuts?" *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 2, pp. 147–159, Feb. 2004.

[32] O. Veksler, Y. Boykov, and P. Mehrani, "Superpixels and supervoxels in an energy optimization framework," in *Proc. Eur. Conf. Comput. Vis.*, 2010, pp. 211–224.

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

[34] Y. Cheng, "Mean shift, mode seeking, and clustering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 17, no. 8, pp. 790–799, Aug. 1995.

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

[36] Y. A. Sheikh, E. A. Khan, and T. Kanade, "Mode-seeking by medoidshifts," in *Proc. IEEE Int. Conf. Comput. Vis.*, Jun. 2007, pp. 1–8.

[37] A. Vedaldi and S. Soatto, "Quick shift and kernel methods for mode seeking," in *Proc. Eur. Conf. Comput. Vis.*, 2008, pp. 705–718.

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

[39] X. Dong, J. Shen, L. Shao, and M.-H. Yang, "Interactive cosegmentation using global and local energy optimization," *IEEE Trans. Image Process.*, vol. 24, no. 11, pp. 3966–3977, Nov. 2015.

[40] J. Shotton, J. Winn, C. Rother, and A. Criminisi, "TextonBoost for image understanding: Multi-class object recognition and segmentation by jointly modeling texture, layout, and context," *Int. J. Comput. Vis.*, vol. 81, no. 1, pp. 2–23, Dec. 2007.

Xiaolin Xiao received the B.E. degree in software engineering from Wuhan University, China, in 2013. She is currently pursuing the Ph.D. degree with the Department of Computer and Information Science, University of Macau, Macau, China. Her research interests include superpixel segmentation, saliency detection, and color image processing and understanding.

Yicong Zhou (M'07–SM'14) received the B.S. degree in electrical engineering from Hunan University, Changsha, China, and the M.S. and Ph.D. degrees in electrical engineering from Tufts University, Medford, MA, USA. He is currently an Associate Professor and the Director of the Vision and Image Processing Laboratory, Department of Computer and Information Science, University of Macau, Macau, China. His research interests include chaotic systems, multimedia security, image processing and understanding, and machine learning.

Dr. Zhou was a recipient of the Third Price of the Macau Natural Science Award in 2014. He serves as an Associate Editor for the *Neurocomputing*, the textitJournal of Visual Communication and Image Representation, and the textitSignal Processing: Image Communication. He is the Co-Chair of Technical Committee on Cognitive Computing in the IEEE Systems, Man, and Cybernetics Society.

Yue-Jiao Gong (M'15) received the B.S. and Ph.D. degrees in computer science from Sun Yat-sen University, China, in 2010 and 2014, respectively. During 2015–2016, she was a Post-Doctoral Research Fellow with the Department of Computer and Information Science, University of Macau, Macau. She is currently an Associate Professor with the School of Computer Science and Engineering, South China University of Technology, China. Her research interests include evolutionary computation and machine learning methods, and their applications to image processing. She has authored or co-authored over 50 papers in these research areas. She currently serves as a Reviewer for the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, the IEEE TRANSACTIONS ON NEURAL NETWORK AND LEARNING SYSTEMS, and the IEEE TRANSACTIONS ON IMAGE PROCESSING.