Adaptive Growing and Merging Algorithm for Image Segmentation

Hsuan-Yi Ko and Jian-Jiun Ding *
Graduate Institute of Communication Engineering, National Taiwan University
E-mail: r03942056@ntu.edu.tw, jjding@ntu.edu.tw
Tel: +886-2-33669652

***Abstract*— Image Segmentation plays an important role in image processing as it is at the foundation of many high-level computer vision tasks, such as scene understanding and object recognition. In this paper, an adaptive growing and merging algorithm is proposed to segment an image accurately. First, mean shift is applied to produce superpixels, and then superpixels grow according to their lab histograms and textures under the constraint of the edge's intensity. In adaptive region merging, we use the proposed dissimilarity measures, which are based on colors, textures, region sizes and multi-scale contour maps with non-constant weights that are adaptive to the region features. Furthermore, we take account of the contact rate of two adjacent regions to avoid over-merging. We also exploit the saliency map to maintain the main objects when the number of regions is small. The simulations on Berkeley segmentation database show that our proposed method outperforms state-of-the-art methods.**

I. INTRODUCTION

Image segmentation is a process of partitioning an image into a small number of disjointed homogeneous regions, with the goal of minimizing intra-variance and maximizing inter-variance among regions. There are many different ways to perform image segmentation. Traditional image segmentation is pixel-based and it clusters pixels into regions by different approaches such as seeded region growing [1, 2], statistical region merging [3] and pyramidal structures [4].

A recent trend in segmentation is to start the computation from superpixels instead of pixels [5-7]. Superpixels are perceptually meaningful atomic regions which are clusters of connected pixels with similar features. There are several algorithms of generating superpixels like mean shift (MS) [8], normalized cut (Ncut) [9], simple linear iterative clustering (SLIC) [10] and entropy rate superpixel (ERS) [11]. Since the number of superpixels is smaller than that of pixels, using superpixels can reduce complexity. Moreover, superpixels provide local low-level features and can improve the quality of the final result.

In this paper, a superpixel-based algorithm is introduced. Besides the widely used features in segmentation: colors and textures, to make the results contain accurate object boundaries, we use the edge detection and generate multi-scale contour maps. We regard the edge information on the common boundaries of two adjacent superpixels as a form of similarity between the two. Then, we combine colors, textures and the multi-scale edge information, and weight them according to the region size, texture distance and color difference in our proposed dissimilarity measures. In our framework, we first grow superpixels based on their Lab color histogram and the texture features which have been generated by Log-Gabor filters, and then merge the adjacent regions in the order of the dissimilarity which is adaptive to the number of regions. To avoid over-merging, the contact rate and the contour rate of two adjacent regions are adopted to limit growing or merging. We also apply saliency maps to avoid merging the background and main objects. An overview of our framework is shown in Fig. 1.

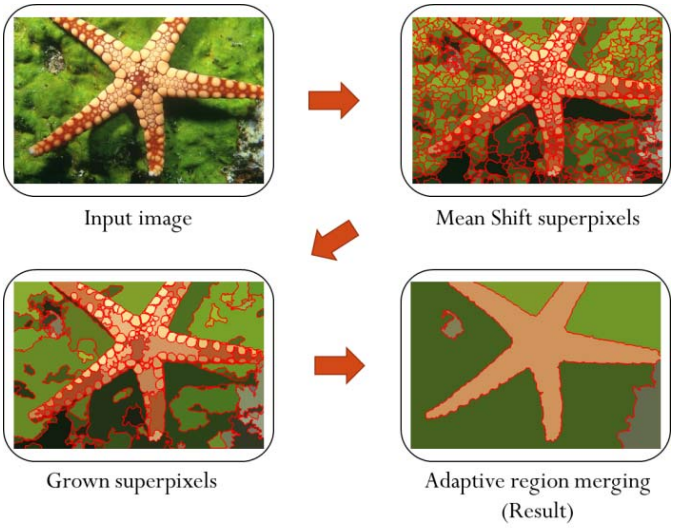


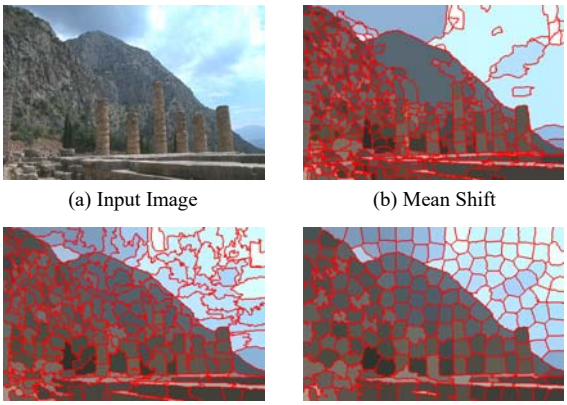
Fig. 1 An overview of the proposed framework.

The rest of the paper is organized as follows. In Section 2, we give a brief introduction about the techniques used in our method including superpixels, the texture distance, the contour rate, the edge strength, the color histogram similarity, CIEDE2000 color difference, and saliency maps. In Section 3, we give a detailed description of our proposed algorithm which consists of two stages: superpixel growing and adaptive region merging. In Section 4, the simulations on Berkeley Segmentation Database are presented. Finally, the paper is concluded in Section 5.

II. ADOPTED TECHNIQUES

A. Superpixel Generation

We choose mean shift [8] to generate superpixels because the generated superpixels adhere to object boundaries well and the segmented result is between over-segmentation and over-merging, which means the simple regions are segmented



(a) Input Image (b) Mean Shift

(c) ERS (d) SLIC

Fig. 2 A comparison of different types of superpixels.

into fewer superpixels and the complicated regions are segmented into more superpixels. This property is useful in our algorithm since we can omit a lot of steps of merging redundant superpixels in the case of simple regions. Furthermore, the large superpixels are helpful for growing and merging regions in our algorithm by providing more accurate local image features such as mean color and texture. In Fig. 2, compared with mean shift, ERS and SLIC segment the simple region like the sky into many superpixels, which causes the redundancy.

参考：超像素边缘

B. Log-Gabor Texture Distance

We use Log-Gabor filter proposed by Field [12] to extract the texture features. It is an alternative to the Gabor function. The Fourier transform of the log-Gabor filter is:

$$G(f_x, f_y) = \exp\left(-\frac{\left(\log(\frac{f_x \cos\phi + f_y \sin\phi}{f_0}) \right)^2}{2\left(\log(\frac{\sigma}{f_0}) \right)^2} \right) \quad (1)$$

which is a Gaussian function on the log frequency axis. By varying the values of σ and ϕ, one can extract the texture feature with different scales and orientations.

Field suggests that natural images are better coded by filters that have Gaussian transfer functions when viewed on the *logarithmic* frequency scale. (Gabor functions have Gaussian transfer functions when viewed on the *linear* frequency scale.) Therefore, Log-Gabor filter is a better model for suiting visual system than Gabor filter.

In our algorithm, we use the Log-Gabor function with 2 scales and 4 orientations to extract texture features (eight dimensions in total). For each superpixel i and its adjacent superpixel j with the mean texture $T_k(i)$ and $T_k(j)$ respectively, we define the difference of the texture as

$$dTex(i,j) = \sqrt{\sum_{k=1}^{8} (T_k(i) - T_k(j))^2} \quad (2)$$

where k means the kth texture feature.

C. Edge Detection

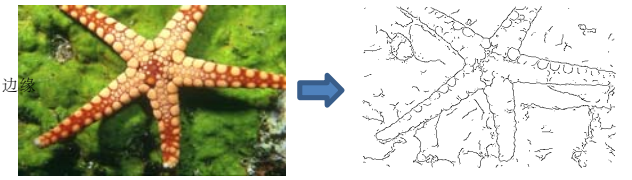


Fig. 3 An example of a contour map using the structured edge (SE) detector with the *threshold* = 0.1.

Fig. 4 An example of the saliency map generated by the saliency detection [20].

Edge detection [13, 14] is a critical component of many vision systems. In computer vision and image processing, edge detection concerns the localization of significant variations of the grey level image. We apply the structured edge (SE) detection method proposed by Dollar *et al.* [15] to the input image and set a *threshold* on the edge map to obtain a binary contour map as shown in Fig. 3. Then long contours are selected from the contour map. Here we define the contour rate as

$$ContourRate(i,j) = \frac{\#\ of\ \ pixels\ of\ \ long\ contours\ on\ Bnd(i,j)}{\#\ of\ \ pixels\ of\ \ Bnd(i,j)} \quad (3)$$

where the long contours are the connected components of the contour map whose number of pixels are higher than a threshold *contourLength* and $Bnd(i, j)$ is the common boundary of adjacent superpixel i and superpixel j.

Since the long contour is usually not closed, to compensate for a lack of long contours, we consider the edge strength defined as

$$ES(i,j) = \frac{\sum_{p \in Bnd(i,j)} edge\ value\ of\ \ p}{\#\ of\ \ pixels\ of\ \ Bnd(i,j)} \quad (4)$$

where the edge value is from the edge map generated by SE detector and p is the pixel on $Bnd(i, j)$.

D. Color Histogram Similarity

We first quantize each color channel to reduce computation and noises and then generate the color histogram with the quantized image. Denote the normalized histogram of a superpixel R by $Hist_R$. We define the similarity measure $\rho(R,Q)$ between two superpixels R and Q as

$$\rho(R,Q) = \sum_{u=1}^{U} \sqrt{\mathrm{Hist}_R^u \cdot \mathrm{Hist}_Q^u} \quad (5)$$

where U is the number of bins based on the quantization levels, and the superscript u represents the uth element of the normalized histogram. Bhattacharyya coefficient ρ is the cosine of the angle between the unit vectors:

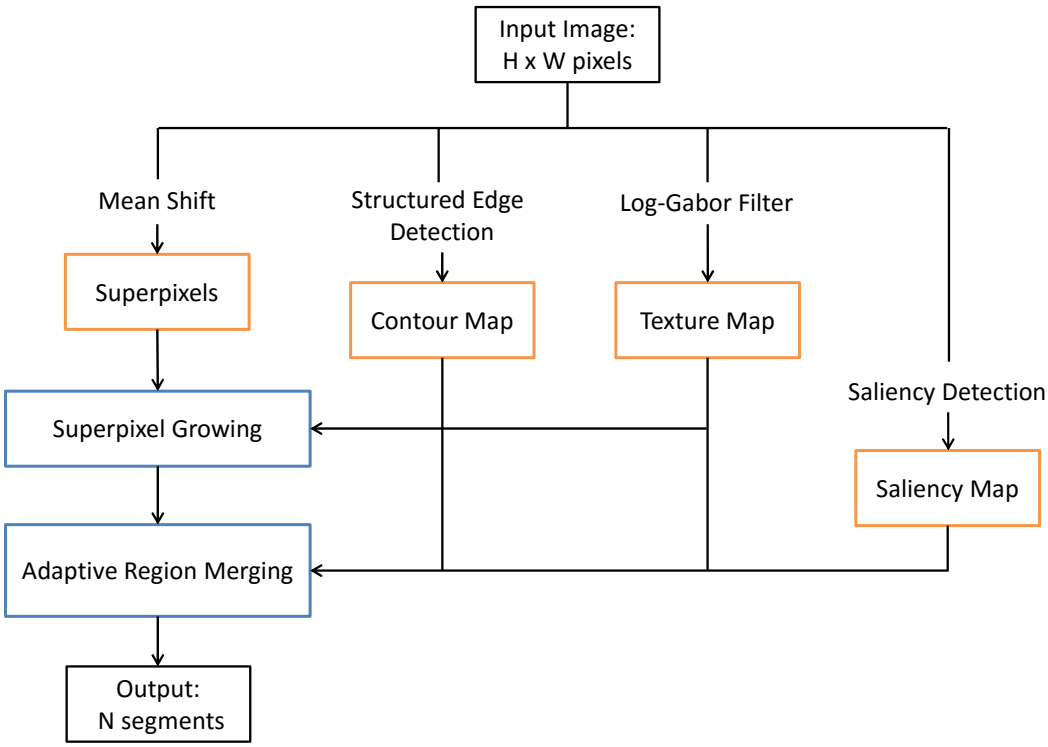


Fig. 5 The block diagram of our method.

$$\left(\sqrt{\text{Hist}_R^1}....\sqrt{\text{Hist}_R^U}\right)^T \text{ and } \left(\sqrt{\text{Hist}_Q^1}....\sqrt{\text{Hist}_Q^U}\right)^T$$

The higher the Bhattacharyya coefficient between R and Q is, the higher their similarity will become, and ρ is within [0, 1].

E. CIEDE2000 Color Difference

To better match human color perception, we use CIELAB color space and the standard CIEDE2000 color difference [16]. CIEDE2000 color difference formula is proposed based on the situation that every person has a different sensitivity of color difference under various values of the brightness, color saturation, or hue.

Given two regions R_1 and R_2, we compute their mean values of the Lab components $M_{R1} = (L_1, a_1, b_1)$ and $M_{R2} = (L_2, a_2, b_2)$, and define the distance *de00* as

$$de00(R_1, R_2) = \Delta E_{00}(M_{R1}, M_{R2}) \tag{6}$$

where ΔE_{00} is the CIEDE2000 color difference.

F. Saliency Detection

Saliency detection [17-19] is a process of detecting the importance of each pixel in an image. The degree of the importance is called saliency value, which has larger value when the location captures more attention. Extracting saliency maps are widely used in many computer vision applications, including object-of-interest image segmentation, object recognition, content-aware image resizing and image retrieval. In our segmentation framework, we use the saliency detection method proposed by Zhu *et al.* [20] (see Fig. 4) and compute the difference of the saliency values between regions R_1 and R_2:

$$dSV(R_1, R_2) = |SV(R_1) - SV(R_2)| \tag{7}$$

where $SV(R_1)$ is the mean saliency value of region R_1.

III. Proposed Algorithm

The block diagram of the proposed method is illustrated in Fig. 5. The proposed algorithm consists of two stages: superpixel growing and adaptive region merging. The detailed procedure is introduced below.

A. Superpixel Growing

With the initial segments from mean shift, we first merge any superpixel R with its adjacent superpixel Q if the two are the most similar adjacent superpixels to each other. We uniformly quantize each Lab color channel into 32 levels and then the histogram of each superpixel is calculated in the feature space of 32×32×32 = 32768 bins. Denote the set of Q's adjacent superpixels as $\{S_i^Q\}_{i=1,2,...,q}$. Obviously, R is a member of $\{S_i^Q\}_{i=1,2,...,q}$ and Q is a member of $\{S_i^R\}_{i=1,2,...,r}$. We will merge R and Q when the following conditions are satisfied.

$$\begin{aligned} \rho(R,Q) &= \max_{i=1,2,...r} \rho(R, S_i^R) \\ \rho(R,Q) &= \max_{i=1,2,...q} \rho(Q, S_i^Q) \end{aligned} \tag{8}$$

Although condition (8) is strict, over-merging still occurs in some situations. Therefore, we set limitations to solve this problem. If $dTex(R,Q)$ or $ContourRate(R,Q)$ is higher than a threshold, we will not merge them.

After merging each superpixel, we select the superpixels whose merging times are higher than or equal to three to be the seeds and the seeds will grow according to the distance:

$$Dist(i, j) = a \cdot dTex(i, j) + b \cdot (1 - \rho(i, j)) \quad (9)$$

where region i is a seed and region j is one of its adjacent regions. The seeds will merge with the neighbor who has the smallest *Dist*. In our experiment, we set the weights (a, b) = (3, 0.5) and only use the quantized a* and b* color channels, each containing 32 quantization levels, to merge the regions with shadows easily. In order to avoid over-merging, we also set a limitation of *ContourRate*(i, j) in our growing process. Sometimes, the seed is entirely different from all adjacent regions so we need to ensure that the color difference and texture distance are small enough to grow seeds.

B. Adaptive Region Merging

In our adaptive region merging, we take account of the whole image instead of a region and its neighbors so we generate an $N \times N$ dissimilarity table for N regions and merge regions in the order of the dissimilarity.

We use multi-scale contour maps in our adaptive region merging to generate *ContourRate* in different degree and define *CR25*, *CR80*, and *CR30* as *ContourRate* from contour maps with (*contourLength*, *threshold*)= (25,0.1), (80,0.1) and (30, 0.15) respectively.

When the number of regions becomes smaller, we want to remain the main object, which is usually large, rather than the meaningless small regions. Therefore, we try to merge small regions first. For the two regions R_1 and R_2, we define

$$minArea = \min(area(R_1), area(R_2)) \quad (10)$$

where *area*(R_1) means the size of the area of R_1 and min(A, B) outputs the smaller value taken from A or B.

We propose three kinds of dissimilarity measures:

$$\begin{aligned} &score1(R_1, R_2) \\ &= \left(a \cdot ES(R_1, R_2) + b \cdot CR25(R_1, R_2) + c \cdot CR80(R_1, R_2)\right) \\ &\times \min\left(\frac{minArea}{W}, K\right) - \rho(R_1, R_2) + 2dTex(R_1, R_2) \end{aligned} \quad (11)$$

$$\begin{aligned} &score2(R_1, R_2) \\ &= (ES(R_1, R_2) + CR25(R_1, R_2)) \times \frac{minArea}{W} + CR30(R_1, R_2) \\ &-(1-\beta) \cdot \rho(R_1, R_2) + \beta \cdot de00(R_1, R_2) + \alpha \cdot dTex(R_1, R_2) \end{aligned}$$

$$\alpha = \max\left(2,\ 2 \times \frac{W}{minArea}\right) \quad (12)$$

$$\begin{aligned} score3(R_1, R_2) &= a \cdot ES(R_1, R_2) + b \cdot CR25(R_1, R_2) \\ &+ c \cdot CR80(R_1, R_2) + d \cdot CR30(R_1, R_2) \\ &- \max\left(\frac{W}{minArea}, K\right) - \rho(R_1, R_2) + 2dTex(R_1, R_2) \end{aligned} \quad (13)$$

where a, b, c, and d are weights which can be constants, $dTex(R_1, R_2)$, $1 - \rho(R_1, R_2)$ or $de00(R_1, R_2)$, W and K are constants, and max(A, B) outputs the largest value taken from A or B. K and W are the weights for adjusting the influence of small regions. In order to avoid the difference of the scores between the large region and the small region being too large to merge the similar lager regions, we set K to restrict the weights. In addition, here we use the similarity measure ρ defined in (5) with the quantization levels=16, 32, 32 for L*, a* and b* color channels respectively in order to reduce the effect of shadows.

Since the shape of superpixels is irregular, it is highly probable that two superpixels contact each other by few pixels and then be regarded as neighbors. In this case, *ContourRate* and *ES* may be useless to separate the different adjacent superpixels. To avoid this problem, we propose

$$ContactRate(R_1, R_2) = \frac{\#\ of\ \ pixels\ of\ \ Bnd(R_1, R_2)}{\min(BL(R_1), BL(R_2))} \quad (14)$$

where $BL(R_1)$ means the perimeter of region R_1. If $ContactRate(R_1, R_2)$ is lower than *ContactTh*, then region R_1 and region R_2 will not be merged. When the number of segments N is large, we would rather over-segmentation than over-merging so large *ContactTh* is set for large N.

In addition, when N decreases, how to maintain the main objects is important. Therefore, in our experiment, we consider *dSV* when the number of regions is smaller than 44, and the regions with large *dSV* will not be merged.

C. Analysis of Our Algorithm

We analyze the techniques we used and summarize the advantages of our method compared with the state-of-the-art methods as follows.

(1) Superpixel-based growing and merging

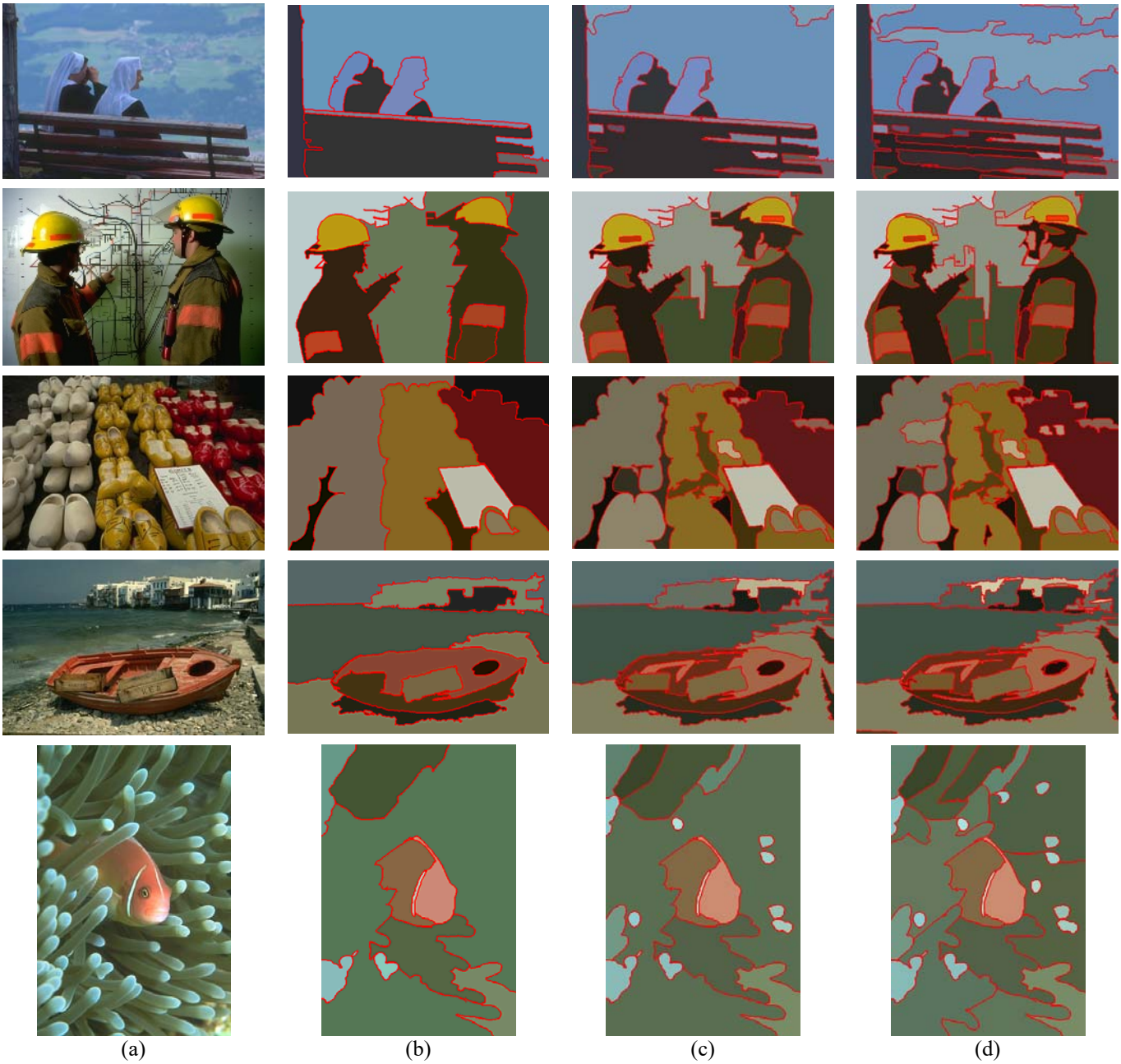
Traditional seeds growing algorithms are based on pixels [21, 22]. They grow pixels into regions. However, our algorithm is based on superpixels, which contains regional information that pixels can't provide. Furthermore, the computation of superpixel-based segmentation is much lower than pixel-based segmentation since the number of superpixels is much smaller than that of pixels. In addition, the superpixel method [8] used in our framework can preserve accurate object boundaries and reduce redundant superpixels without losing details. Therefore, the results of our segmentation algorithm are better than that of the algorithms not based on superpixels.

(2) Emphasis on the small region merging

Reference [7] uses *minArea* in their boundary-focused region merging but they only subtract *W*/*minArea* with a constant W in their dissimilarity measure. However, we regard *minArea*/W as a weight of the edge information to enforce merging small regions and W changes according to the number of regions. Fig. 6 shows the results of our algorithm when the number of segments *Nseg* decreases. We can remain the main object with small *Nseg* due to the emphasis on the small region merging.

(3) Multi-scale contour maps and non-constant weights

We use different *contourLength* and *threshold* to construct the multi-scale contour maps. Then we apply multi-scale



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

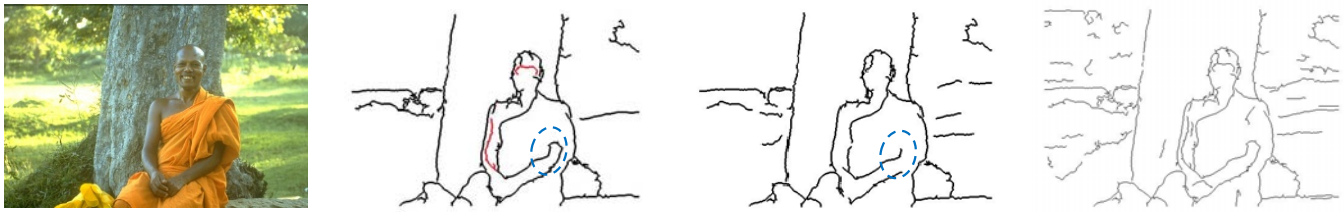
Fig. 6 Our segmentation results with decreasing number of regions *Nseg*.
(a) Input images. (b) *Nseg*=10. (c) *Nseg*=20. (d) *Nseg*=30.

contour maps in our adaptive region merging to generate *ContourRate* defined in (3) in different degree. In Fig. 7, the multi-scale contour maps used in our algorithm are shown. It can be seen that due to the low threshold, Fig. 7 (b) contains some redundant long contours which marked in red color, and these make the regions with the redundant long contours merged hard. However, we can use Fig. 7 (c) to improve the situation. On the other hand, some important long contours which Fig. 7 (b) contains, are absent in Fig. 7 (c) such as the place that the blue dotted line circles. Fig. 7 (b), (c) and (d) all have the main long contours such as the shape of the body. Therefore, the multi-scale contour maps can compensate for the shortage of each other and strengthen the main contours.

Moreover, for *ContourRate* and *ES*, we use non-constant weights which are determined by the texture distance or color difference of the two adjacent regions.

In consequence, the multi-scale contour maps and non-constant weights make our dissimilarity measures more adaptive to the features of the regions (e.g. color, texture, edge).

(4) Avoidance of over-merging by *ContactRate* and saliency maps



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

Fig. 7 The long contour maps in distinct degree. (a) Input Image. (b) The dilated long contour map with *contourLength* = 80 and *threshold*=0.1 for the edge map. The redundant long contours are marked in red. (c) The dilated long contour map with *contourLength* = 30 and *threshold*=0.15 for the edge map. Compared with (b), the blue dotted line circles the place where important contours are absent. (d) The long contour map with *contourLength* = 25 and *threshold*=0.1 for the edge map. (Note: the edge map is generated by the structured edge detector [15])

In our observation, the edge information on $Bnd(R_1, R_2)$ is usually inadequate to segment well due to a small number of pixels of $Bnd(R_1, R_2)$. Therefore, we calculate the *ContactRate* defined in (14) and we do not merge the regions with low *ContactRate* to avoid over-merging.

The saliency detection extracts the important parts in an image so exploiting the saliency maps in our method makes the segmentation results maintain the main objects, especially with a small number region.

(5) Adaptive region merging

We propose three dissimilarity measures and the parameter setting are adaptive to the current number of regions *N*. The details are described in Table 1. For simplification, the large region mentioned below is the region whose *minArea* is larger than *W*, and the small region mentioned below is the region whose *minArea* is smaller than *W*.

When *N* is large, the influence of the edge information is greater than that of color and texture in *score1* defined in (11). Since the multi-scale edge information of the large regions can be multiplied by at most five, the difference of the scores of a large region and a small region is dominated by the edge information.

When *N* is between 74 and 44, we use *score2* defined in (12) and for large regions, the edge information is more important than texture. Furthermore, for the purpose of avoiding a lack of edge information due to the very small *minArea*, we keep *CR30* alone.

When *N* is small, we use *score3* defined in (13), which is the weighted summation of all information about edge, texture, color and region size. We merge regions more carefully and therefore the parameter setting changes frequently.

D. Uniqueness of Our Method

We summarize the methods we proposed and the uniqueness of our algorithm as follows.

(1) Lab histogram similarity

Most segmentation methods only calculate the mean color difference. Even though there are methods using color histogram, they are based on RGB color space. We apply the Lab color space to match human visual perception and compute the Lab histogram similarity by (5). We also reduce the quantization level of L* color channel to reduce the effect of the changes in luminance and make the regions with shadows merged easily.

(2) The *ContourRate* based on multi-scale contour maps and the *ES* from structured edge detection

In addition to color and texture, we also apply the edge information on the superpixel boundaries and propose *ContourRate* and *ES* defined in (3) and (4) respectively. They can be regarded as a kind of similarity between two adjacent superpixels. Higher *ContourRate* or *ES* means the superpixels are more dissimilar and it is harder to merge them. Furthermore, we use the multi-scale contour maps, which make the long contours more robust, to compute the different *ContourRate* in the adaptive region merging stage.

(3) Small region merging

Few methods consider merging small regions but our dissimilarity measures take region size into account. In *score1* and *score2*, the edge information is multiplied by a weight about the region size to enforce merging small regions. We also set *K* to restrict the weight to avoid the difference of scores between the large region and the small region being too large to merge the similar lager regions.

(4) The contact rate of two adjacent regions

We propose the *ContactRate* defined in (14) and if $ContactRate(R_1, R_2)$ is lower than *ContactTh*, then region R_1 and region R_2 will not be merged to avoid over-merging, especially with a large number of regions.

(5) Region merging is adaptive to the number of regions and local features.

Our dissimilarity measures are adaptive to the number of regions. For example, when the current number of regions *N* decreases, the small regions will be merged into middle and then large regions so the weight *W* of *minArea* should increase to achieve the goal of merging small regions. The threshold of the *ContactRate* is also adaptive to *N*. When *N* is large, the segmentation result can contain more details and we prefer over-segmentation rather than over-merging. Therefore, *ContactTh* is large when *N* is large.

TABLE I
THE PARAMETER SETTING AND DISSIMILARITY MEASURES USED FOR DIFFERENT NUMBER OF REGIONS N

$N \geq 75$	*score1* (11)	$N \geq 100$	$(a,b,c) = (1, 1, 0.5)$, W=200, K=5
		$100 > N \geq 75$	$(a,b,c) = (5, 0.5, 0)$, W=400, K=2
$75 > N \geq 44$	*score2* (12)	$75 > N \geq 44$	$\beta = 0.2$, W=600
$44 > N$	*score3* (13)	$44 > N \geq 24$	$(a,b,c,d) = (de00, 1, (1-\rho)\times dTex, 0)$, W=600, K=1
		$24 > N > 15$	$(a,b,c,d) = (de00, 1, (1-\rho)\times dTex, 0)$, W=1200, K=0
		$15 \geq N \geq 10$	$(a,b,c,d) = (de00, 1-\rho, 0.4, 0)$, W=1200, K=0
		$10 > N$	$(a,b,c,d) = (2, 0, 0, 1)$, W=0, K=0

In addition, the weights of the edge information in our dissimilarity measures can be the texture distance or the color difference of two adjacent regions. This makes our region merging adaptive to the local image features.

IV. SIMULATIONS

A. Parameter Setting

The superpixels are generated by Mean Shift with (h_s, h_r, M) = (5, 7, 100) where h_s is the spatial bandwidth, h_r is the range bandwidth and M is the minimum size of final output regions.

In adaptive region merging stage, three different dissimilarity measures are applied for different number of regions. We use *score1* defined by (11) when the current number of regions N is more than 74, use *score2* defined by (12) when N is between 74 and 44 and use *score3* defined by (13) when N is less than 44. The particularized parameter setting in our algorithm is illustrated in Table 1.

The parameters, a, b, c, and d, are weights of $ES(R_1, R_2)$, $CR25(R_1, R_2)$, $CR80(R_1, R_2)$ and $CR30(R_1, R_2)$ respectively. They can be not only constants but also texture distance or color difference; that is, the influence of the edge information can be adjusted by the similarity of the texture or the color of two adjacent regions. The larger W is set when the number of regions in an image is smaller in order to enforce merging small regions. When N is less than 10, we will not put emphasis on the small regions any more so K and W are set to be zero.

B. Database and Evaluation Metrics

All experiments are carried out on the Berkeley Segmentation Database (BSD) [23], which consists of 300 color images with the size of 481x321. For each image, a set of ground-truth data, which includes at least 4 human annotations, is provided. Four standard measurements are used for quantitative evaluation: the Probabilistic Rand Index (PRI) [24], the Variation of Information (VoI) [25], the Global Consistency Error (GCE) [26], and the Boundary Displacement Error (BDE) [27]. The segmentation is better if PRI is larger and the other three are smaller.

The Probabilistic Rand Index (PRI) counts the fraction of pairs of pixels whose labels are consistent between the computed segmentation and the ground truth. The value of PRI ranges from 0 (when there is no intersection at all between test and ground truth segmentation) to 1 (when the two segmentations are actually the same).

The Volume of Information (VoI) measures the distance between a segmentation S and a ground-truth segmentation T in terms of their average conditional entropy given by

$$\mathrm{VoI}(S,T) = H(S) + H(T) - 2I(S;T) \tag{15}$$

where H and I represent, respectively, the entropies and mutual information between the two segmentations. Therefore, VoI measures the amount of information of one result not contained in the other, with lower values representing greater similarity.

The Global Consistency Error (GCE) measures the extent to which one segmentation can be viewed as a refinement of the other. If subsets of regions in one segmentation consistently merge into some region in the other segmentation the consistency error should be low.

In order to compute the consistency error for a pair of images, we first define a measure of the non-symmetric local consistency error at each pixel p_i as

$$E(S_1, S_2, p_i) = \frac{|R(S_1, p_i) \setminus R(S_2, p_i)|}{|R(S_1, p_i)|} \tag{16}$$

where $R(S_j, p_i)$ is the subregion of segmentation j that contains pixel p_i, $\setminus$ denotes set difference, and $|\cdot|$ denotes set cardinality. This measure evaluates to 0 if all the pixels in S_1 are also contained in S_2 thus achieving the tolerance to refinement. It is important to note that this measure is not symmetric, so for every pixel it must be computed twice, once in each direction.

The global consistency error is obtained by

$$GCE(S_1, S_2) = \frac{1}{n}\min\left(\sum_i E(S_1, S_2, p_i), \sum_i E(S_2, S_1, p_i)\right) \tag{17}$$

where n is the total number of pixels of the image. GCE is valued in [0, 1], and the null value indicates of course that both segmentations are equivalent.

The Boundary Displacement Error (BDE) computes the average displacement between the boundaries of two segmentations. More precisely, it defines the error of one boundary pixel as the distance between the pixel and its closest boundary pixel in the other image. Denoting

$$d(p_i, B_2) = \min_{p \in B_2} \| p_i - p \| \quad (18)$$

the distance of a boundary point $p_i \in B_1$ to the boundary set B_2, and N_1, N_2 the total number of points in the boundary sets B_1 and B_2, BDE is defined as

$$BDE(B_1, B_2) = \frac{\sum_i^{N_1} d(p_i, B_2)/N_1 + \sum_i^{N_2} d(p_i, B_1)/N_2}{2} \quad (19)$$

The smaller BDE indicates the two segmentations are more similar.

C. Comparison with the State-the-art-of Methods

We compare our algorithm with the other methods including Normalized cut (Ncut) [9], JSEG [28], Multi-scale Ncut (MNcut) [29], Normalized Tree Partitioning (NTP) [30], Saliency Driven Total Variation (SDTV) [31], Texture and Boundary Encoding-based Segmentation (TBES) [32], Ultrametric Contour Map (UCM) [33], Multi-Layer Spectral Segmentation (MLSS) [5], Segmentation by Aggregating Superpixels (SAS) [6], 5-D spectral clustering with boundary-focused region merging [7], Superpixel-based image segmentation with color covariance matrix manifolds [34], A global/local affinity graph for image segmentation (GL-graph) [35], SCS [36].

The scores are shown in Table 2, and the best performance is marked in red bold type. Obviously, our algorithm outperforms all the others in all evaluation metrics. We also compare our method with MLSS, SAS and 5-D spectral clustering with boundary-focused region merging [7] when the number of segments N_{seg} is fixed for all images. Table 3 shows the average scores by varying N_{seg} from 10 to 30 in the step of ten.

Some segmentation results can be visualized in Fig. 8. The results of Ncut are manually tuned using the authors' software. The results of TBES available on the authors' website are the segmentations which have the best PRI. The results of MLSS and SAS are the best segmentations provided by the authors and the results of ours are also the best segmentation with the optimal number of regions. It can be seen that the segmentations of our method are perceptually more satisfactory and contain more accurate object boundaries than the others.

V. Conclusion

In this paper, we proposed a novel superpixel growing and merging algorithm, which consists of two stages: superpixel growing and adaptive region merging. In stage 1, we grow superpixels according to their mean texture and Lab histogram under the constraint of the edge's intensity. In stage 2, we merge regions in the order of the dissimilarity, which is adaptive to the number of regions.

The techniques used in our framework include the Euclidean distance of Log-Gabor texture features, the Bhattacharyya distance of color histogram, CIEDE2000 color difference, *ContourRate* based on the multi-scale contour maps, *Edge Strength* from the edge map generated by the structured edge detection, *ContactRate*, saliency detection, and region size. Constructing the initial segments by Mean Shift simplifies merging process and makes our algorithm efficient. Enforcing merging small regions and applying saliency detection maintain the main shapes of objects when the number of regions decreases. Moreover, the contour-based merging and accurate superpixel boundaries make our results more adherent to object boundaries

The experimental results on the Berkeley Segmentation Database have demonstrated that the proposed image segmentation method outperforms the state-of-the-art methods in terms of the quantitative and perceptual criteria.

TABLE II
PERFORMANCE EVALUATION OF THE PROPOSED METHOD AGAINST OTHER METHODS OVER THE BERKELEY SEGMENTATION DATABASE

Method	PRI	VoI	GCE	BDE
Ncut	0.7242	2.9061	0.2232	17.15
JSEG	0.7756	2.3217	0.1989	14.40
MNcut	0.7559	2.4701	0.1925	15.10
NTP	0.7521	2.4954	0.2373	16.30
SDTV	0.7758	1.8165	0.1768	16.24
TBES	0.80	1.76	N/A	N/A
UCM	0.81	1.68	N/A	N/A
MLSS	0.8146	1.8545	0.1809	12.21
SAS	0.8319	1.6849	0.1779	11.29
[7]	0.8474	1.5542	0.1834	11.15
[26] (using W_{HP})	0.8495	1.6260	0.1785	12.3034
GL-graph	0.8384	1.8012	0.1934	10.6633
SCS	0.8414	1.6573	0.1795	10.8783
Proposed	**0.8681**	**1.3905**	**0.1443**	**8.99**

TABLE III
COMPARISON WITH THE STATE-OF-THE-ART METHODS WITH THE FIXED NUMBER OF SEGMENTS N_{SEG}={10,20,30}

Nseg=10				
Method	**PRI**	**VoI**	**GCE**	**BDE**
MLSS	0.7927	1.9478	0.2137	13.832
SAS	0.8043	1.8275	0.2043	13.521
[7]	0.8083	1.7034	0.2039	14.090
Proposed	**0.8157**	**1.6011**	**0.1609**	**10.9588**

Nseg=20				
Method	**PRI**	**VoI**	**GCE**	**BDE**
MLSS	0.7861	2.2201	0.1759	12.557
SAS	0.8015	2.0612	0.1839	12.4869
[7]	0.8178	1.7558	0.2060	12.5114
Proposed	**0.8282**	**1.7031**	**0.1590**	**10.3593**

Nseg=30				
Method	**PRI**	**VoI**	**GCE**	**BDE**
MLSS	0.7758	2.4874	**0.151**	12.497
SAS	0.7974	2.2743	0.1647	12.3961
[7]	0.8200	1.8114	0.2090	12.2438
Proposed	**0.8298**	**1.7839**	0.1524	**10.4655**

(a) (b) (c) (d) (e) (f)

Fig. 8 Segmentation examples on Berkeley Segmentation Database.
(a) Input images. (b) Ncut. (c) TBES. (d) MLSS (e) SAS. (f) Ours.

REFERENCES

[1] F. Y. Shih, and S. Cheng, "Automatic seeded region growing for color image segmentation," *Image and vision computing*, vol. 23, no. 10, pp. 877-886, 2005.

[2] C. C. Kang, W. J. Wang, and C. H. Kang, "Image segmentation with complicated background by using seeded region growing," *AEU-International Journal of Electronics and Communications*, vol. 66, no. 9, pp. 767-771, 2012.

[3] R. Nock, and F. Nielsen, "Statistical region merging," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 26, no. 11, pp. 1452-1458, 2004.

[4] R. Marfil, L. Molina-Tanco, A. Bandera, J. A. Rodríguez, and F. Sandoval, "Pyramid segmentation algorithms revisited," *Pattern Recognition*, vol. 39, no. 8, pp. 1430-1451, 2006.

[5] T. Kim and K. Lee, "Learning full pairwise affinities for spectral segmentation," in *CVPR*, pp. 2101-2108, 2010.

[6] Z. Li, X. M. Wu, and S. F. Chang, "Segmentation using superpixels: A bipartite graph partitioning approach," in *CVPR*, pp. 789-796, 2012.

[7] C. Y. Hsu and J. J. Ding, "Efficient image segmentation algorithm using SLIC superpixels and boundary-focused region merging," in *ICICS*, pp. 1-5, 2013.

[8] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 24, no. 5, pp. 603-619, May 2002.

[9] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 888-905, Aug. 2000.

[10] 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 Analysis and Machine Intelligence*, vol. 34, no. 11, pp. 2274-2282, May 2012.

[11] Y. M. Liu, O. Tuzel, S. Ramalingam, and R. Chellappa, "Entropy rate superpixel segmentation," in *CVPR*, pp. 2097-2104, 2011.

[12] David J. Field, "Relations between the statistics of natural images and the response properties of cortical cells," *JOSA A*, vol. 4, no. 12, pp. 2379-2394, 1987.

[13] P. Acharjya, R. Das, and D. Ghoshal, "A Study on Image Edge Detection Using the Gradients," *International Journal of Scientific and Research Publications*, vol. 2, no. 12, Dec. 2012.

[14] D. Ziou and S. Tabbone, "Edge detection techniques-an overview," *Int. J. Pattern Recognit. Image Anal.*, vol. 8, no.4, pp. 537-559, 1998.

[15] P. Dollar and C. L. Zitnick, "Structured forests for fast edge detection," in *ICCV*, pp. 1841–1848, 2013.

[16] G. Sharma, W. Wu and E. Dalal, "The CIEDE2000 color-difference formula: Implementation notes, supplementary test data, and mathematical observations," *Color Research & Application*, vol. 30, no. 1, pp. 21-30, 2005.

[17] F. Perazzi, P. Krähenbühl, Y. Pritch, and A. Hornung, "Saliency filters: Contrast based filtering for salient region detection," in *CVPR,* pp. 733-740, June 2012.

[18] C. Yang, L. Zhang, H. Lu, X. Ruan, and M. H. Yang, "Saliency detection via graph-based manifold ranking," in *CVPR*, pp. 3166-3173, 2013.

[19] Q. Yan, L. Xu, J. Shi, and J. Jia, "Hierarchical saliency detection." in *CVPR*, pp. 1155-1162, 2013.

[20] W. Zhu, S. Liang, Y. Wei and J. Sun, "Saliency optimization from robust background detection," in *CVPR*, pp. 2814-2821, 2014.

[21] P. R. Narkhede, and A. V. Gokhale, "Color image segmentation using edge detection and seeded region growing approach for CIELab and HSV color spaces," in *ICIC*, pp. 1214-1218, May 2015.

[22] H. Chen, H. Ding, X. He, and H. Zhuang, "Color image segmentation based on seeded region growing with Canny edge detection," in *Signal Processing (ICSP),* pp. 683-686, Oct. 2014.

[23] D. Martin, C. Fowlkes, D. Tal, and J. Malik. "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *ICCV*, pp. 416–423, 2001.

[24] R. Unnikrishnan, C. Pantofaru, and M. Hebert, "Toward objective evaluation of image segmentation algorithms," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, pp. 929-944, 2007.

[25] M. Meilă, "Comparing clusterings: an axiomatic view," in *ICML*, pp. 577-584, 2005.

[26] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *ICCV*, vol. 2, pp. 416-423, 2001.

[27] J. Freixenet, X. Muñoz, D. Raba, J. Martí, and X. Cufí, "Yet another survey on image segmentation: Region and boundary information integration," in *ECCV*, pp. 408-422, 2002.

[28] Y. Deng and B. S. Manjunath, "Unsupervised segmentation of color-texture regions in images and video," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 23, no. 8, pp. 800-810, 2001.

[29] T. Cour, F. Benezit, and J. Shi, "Spectral segmentation with multiscale graph decomposition," in *CVPR*, vol. 2, pp. 1124-1131, 2005.

[30] J. Wang, Y. Jia, X. S. Hua, C. Zhang, and L. Quan, "Normalized tree partitioning for image segmentation," in *CVPR*, pp. 1-8, 2008.

[31] M. Donoser, M. Urschler, M. Hirzer, and H. Bischof, "Saliency driven total variation segmentation," in *ICCV*, pp. 817-824, 2009.

[32] S. R. Rao, H. Mobahi, A. Y. Yang, S. S. Sastry, and Y. Ma, "Natural image segmentation with adaptive texture and boundary encoding," in *ACCV*, pp. 135-146, 2009.

[33] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 33, no. 5, pp. 898-916, May 2011.

[34] X. Gu, J. D. Deng, and M. K. Purvis, "Improving superpixel-based image segmentation by incorporating color covariance matrix manifolds," in *ICIP*, pp. 4403-4406, Oct. 2014.

[35] X. Wang, Y. Tang, S. Masnou, and L. Chen, "A Global/Local Affinity Graph for Image Segmentation," *IEEE Trans. Image Processing*, vol. 24, no. 4, pp. 1399-1411, 2015.

[36] Y. Yang, Y. Wang, and X. Xue, "A novel spectral clustering method with superpixels for image segmentation," *Optik-International Journal for Light and Electron Optics*, vol. 127, no. 1, pp. 161-167, 2016.