An Increased Modularity based Contour Detection

Sonam Verma
M.tech student
Mittal Group of Institutions Bhopal
India

Achint Chugh
Assistant Professor
Mittal Group of Institutions Bhopal
India

ABSTRACT
This paper proposes an increased modularity created contour detection algorithm. Given an over segmented image that entails of many small regions, our algorithm automatically combines those neighboring regions that produce the largest increase in modularity index. When the modularity of the segmented image is increased, the method stops merging and produces the final segmented image. To preserve the repetitive patterns in a homogeneous region, we propose a feature on the basis of the histogram of states of image gradients and use it together with the color feature to characterize the similarity of two regions. By building the similarity matrix in an adaptive manner, the over segmentation problem can be successfully avoided.

Keywords
Clustering, community detection, image Contouring, modularity, contourdetection

1. INTRODUCTION
As a very important step for these high-level image analysis tasks, image segmentation is an initial step in processing group image pixels into some sizable homogeneous regions so that the complexity of further analysis can be substantially reduced. Image segmentation has received considerable attention since the problem was proposed [1]–[8], yet it still remains to be a challenging problem because of the following reasons: 1) image segmentation is an ill-defined problem and the optimal segmentation is user or application dependent and 2) image segmentation is time consuming in that each image includes a large number of pixels, especially for high-resolution images, and this prevents image segmentation from being applied to real-time applications. In fact, Gestalt principles [9] and some cognition and psychological studies [10] have noted that several key factors affect perceptual grouping a lot; for example, proximity, similarity, regularity, that is, the repetitive patterns, relative size, and so on. In this paper, we will consider all these factors and develop a computational efficient algorithm. Moreover, comprehensive evaluations of the segmentation performance under various metrics are also presented.

The mean shift algorithm [3] treats image segmentation as a problem of clustering by detecting the modes of the probability density function in the feature space. Each pixel in the image is transformed to the joint spatial-range feature space by concatenating the pixel color value and its spatial coordinates into a single vector. Then the mean shift procedure is applied in this feature space to yield aconvergence point for each pixel. All the pixels whose convergence points are closer than the spatial bandwidth hs and the range bandwidth h_r are claimed

to be in the same segment. In addition, minimum segment size is enforced to guarantee sizable segmentation. Although this method is usually fast, it is very sensitive to the bandwidth parameter h_r and h_s, and often results in over segmentation. Because it is based on the density estimation of the color feature for all the pixels, some smooth changes in brightness and texture or the regularities of different colors will converge to different modes, though they belong to the same segment visually. Another line of work view the segmentation problem from the perspective of graph partition. In this framework, the image is regarded as an undirected weighted graph, while each pixel is treated as a node in the graph and the edge weights measure the similarity or dissimilarity between nodes. Felzenszwalb and Huttenlocher (F&H) [4] consider each pixel as a single component in the initial stage, and arrange the edge weights of dissimilarity in a Non-decreasing order. For each iteration, the algorithm merges component C1 and C2 connected by the current edge, if the corresponding edge weight is less than

$$\min(\mathrm{Int}(C1) + \tau(C1), \mathrm{Int}(C2) + \tau(C2)) \quad (1)$$

Where Int (C) is the internal difference of component C, defined as the largest weight in the minimum spanning tree of component C, $\tau(C) = k/|C|$, and k is a constant parameter to control the minimized size of the segment. This algorithm gives nearly linear time complexity; however, it is very difficult to tune the parameter k for optimal segmentation. Normalized cut [2], as another popular graph partition based approach, incorporates the global information of the image into the segmentation process by studying the spectral characteristics of the graph. Given an affinity matrix W with each entry representing the similarity of two pixels, normalized cut tries to solve the generalized eigenvector problem

$$(D - W)y = \lambda Dy \quad (2)$$

Where D is the diagonal matrix with its diagonal entry $ii = \_j W_{ij}$. Then the segmentation is achieved by clustering the eigenvectors. Because of the high computational cost, it can only deal with images of relatively small size. A variant is the multiscale normalized cut approach [11], which allows us to deal with larger images by compressing large images into multiple scales. However, it has the same problem with normalized cut, specifically, it: 1) often breaks uniform or smooth regions where the eigenvectors have smooth gradients; 2) has a high time complexity; and 3) needs a predefined number of segments, which itself is a challenging problem to deal with. More graph-based segmentation can be found in [12]–[14]. However, they all need human intervention, that is, they need a user to specify the number of regions resulting from image segmentation.

The Watershed segmentation method regards the gradient magnitude of an image as a topographic surface. The pixelswhere a water drop starts from would drain to the same local intensity minimum are in one segment, which is called catchment basins. The lines separating the catchment basins are the watersheds, namely, the boundaries. Various improved methods are available [15]–[18], but these methods are generally sensitive to noise and easily lead to over

segmentation. The compression-based texture merging (CTM) method [19] fits the image textures by using the Gaussian mixture model, and employs the principle of minimum description length to find the optimal segmentation, which gives the minimum coding length under a certain distortion ratio. Later, the texture and boundary encoding-based segmentation algorithm [6] improves the CTM by considering a hierarchy of multiple window sizes and offering more precise coding length computation. To be specific, it only encodes the texture information inside the non over lapping windows in each region and also encodes the boundary with the adaptive chain code. However, this greatly increases the computational time cost; besides, the texture feature used in these two algorithms is essentially the pure color information from the cu-toff windows and neglects the regularities inside the image, thus it may split the object with some regularities of different color..

2. ALGORITHM DESCRIPTION

Image segmentation is connected to community detection to some level. Comparable to nodes in the same community, the pixels inside the same segment as well share some properties in common, similar to pixel color value. In this sense, we can give each homogeneous image segment as a community, and consider of image segmentation as a community detection problem. However, because of the inherent properties of images, segmentation is not accurately a community detection problem, and directly applying community detection algorithms in image segmentation will lead to awful performance. The alterations between image segmentation and community detection can be revealed from the following aspects: 1) different from single node in a community, single pixel cannot capture these regularities in each visually homogeneous segment; 2) the pixels within the same segment possibly have completely different properties, like color; whereas for communities, a community is a group of nodes sharing exactly similar properties.

Take the face image as a trial; the full face should be treated as one segment for the purpose of image segmentation. In contrast, for community finding, the eye pixels would be served as a discrete community, while other parts of the face would be served as another community, because the pixel color value property of the eyes is totally dissimilar from that of supplementary parts of the face; 3) compared with communities, images share some a priori information, say, nearby regions are more likely to belong to the same segment; and 4) as the aggregation process goes on, more pixels are contained within one region and the texture inside the region keeps updating, but the properties of the collected communities do not change much. To address the abovementioned problems, we propose an effectual agglomerative image segmentation algorithm, taking benefit of the efficient calculation of the modularity optimization in community detection and the inherent properties of images. The algorithm beginnings from a set of over-segmented regions, thus, runs very fast, and produces sizable segmentation with the regularities inside the same object unspoiled. The overview of the proposed segmentation algorithm is summarized in Algorithm 1. And the detailed presentation of some technical points for our algorithm is as follows.

A. Superpixels

The agglomerative algorithm can start the aggregation process by treating each single pixel as a community; however, it turns out that this will be too much time consuming, especially for the first Louvain iteration. Fortunately, this is indeed not necessary, because no texture information is included for single pixel. Therefore, instead, we start with superpixels, which can reduce the computational cost as well as capture the regularities. Superpixels are a set of very small and standardized regions of pixels. Initializing with superpixels can greatly reduce the time complexity without affecting the segmentation performance. Hence, we first employ a preprocessing step to over-segment the image into a set of superpixels. This preprocessing step can be accomplished by simple K-means bunching algorithm (K is set to be a comparative large value, e.g., 200 or more) or other superpixels creating algorithms. In our implementation, we use a publicly available code [31] to get the superpixel initialization. The superpixel generation step usually gives more than 200 over-segmented regions on average. This step can greatly reduce the complexity to only consider about 200 nodes in the first iteration for our algorithm. The segmentation result given by our proposed algorithm, where only around ten homogeneous regions with similar regular patterns inside are left. This fact demonstrates that the segmentation results are indeed the effects of our proposed algorithm rather than the superpixel generation algorithm.

B. Choice of Color Space

To capture dissimilar characteristics of the color, various color spaces are proposed in [32], such as $\mathrm{RGB}, \mathrm{L} * \mathrm{a} * \mathrm{b}, \mathrm{YUV}, \mathrm{HSV}, \text{and } \mathrm{XYZ}$. To achieve good segmentation performance,

Algorithm 1 Modularity-Based Image Segmentation

Input: Given a color image I and its oversegmented initialization with a set of superpixles $\mathrm{R} = \{\mathrm{R1}, \ldots, \mathrm{Rn}\}$

1: while Pixel labels still change do

2: Reconstruct the neighborhood system for each region in R.

3: Recompute the HoS texture feature and estimate the distribution of the color feature for each region.

4: Adaptively update the similarity matrix W according to Equation (10), $W_{ij} \neq 0$ only if R_i and R_j are adjacent regions in .

5: while modularity increase still exists by merging any two adjacent regions do

6: for each region $R_i \in$ R do

7: Compute the modularity increase caused by merging region R_i with any of its neighboring regions according to (4)

and find the neighboring region R_j, which gives the largest modularity increase among all of the neighboring regions of R_i .

8: Merge regions R_i and R_j by setting the labels of pixels in these two regions to be of the same label

9: end for

10: end while

11: Update the region labels to get a new set of regions $\mathrm{R} = \{\mathrm{R1}, \ldots, \mathrm{Rm}\}$, where m is current number of regions;

12: end while

Output: The set of image segments R.

The select of color space is very significant. Among all the color spaces, the L ∗ a ∗ b color space is identified to be in accord with human visual system and perceptually uniform; hence the image representation in this color space has been widely used in the field of image processing and computer vision. Because of this facet, all of our considerations of thealgorithm are in the L ∗ a ∗ b color space. Later, in the experimental calculation section, we have also validated that the segmentation performance in L ∗a ∗b color space is much better than that in the RGB color space.

C. Neighborhood System Construction

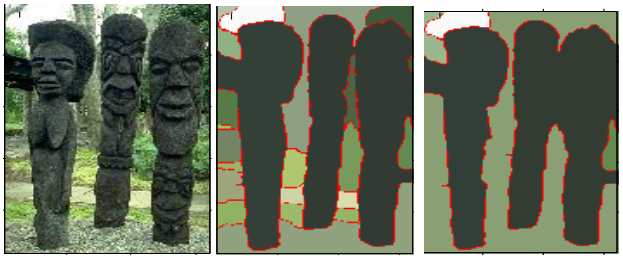
Dissimilar from normal networks, such as social networks or citation networks, images have self-contained spatial a priori information, that is, spatial coherent regions are more expected to be observed as a single segment, while regions far away from each other are more likely to go to different segments. Hence, different from Louvain method where two regions are deliberated to be neighbors as long as the similarity weight between them are nonzero, we have as an alternative constructed a different neighborhood system by incorporating this spatial a earlier information of images. To be definite, we only consider the possibility of merging neighboring regions in the image all through each aggregation method. To accomplish this, for each region in the image, we only contemplate the adjacent regions of this region to be its neighbors and accumulation its neighboring regions using an adjacent list. The adjacent regions are defined to be the regions that share at least one pixel with the current region. In the following methods for the likeness matrix construction and aggregation, we only consider the current region and the regions in its neighborhood system.

D. Features for Similarity

Color is the most straightforward and important feature for segmentation, so we use the pixel value in the L ∗ a ∗ b color space as one of the features for computing the similarity. However, the color feature alone cannot achieve good segmentation performance, because it does not consider the repetitive patterns of different colors in some homogeneous object. The black and white stripe regularities on zebra would be treated as a whole part according to human's perception. Simply using color feature will break down these regularities into different segments. To address this problem, we not only employ the color feature, but we have also proposed a novel texture feature to capture the regularities in the image. Our proposed texture depicter is motivated by the histogram of oriented gradients (HoG) [33] for pedestrian detection; however, instead of constructing a histogram of gradients.

3. EXPERIMENTS

Full results present three measures. The optimal image scale (OIS) is the F-Measure score obtained using the optimal threshold on each image. The last measure is the average precision (AP) and corresponds to the area under the precision-recall curves of Fig.



(a) (b) (c)

Fig. 1. Experimental results (a) The contour detector by previous method (b) contour detector by proposed method

The user manually defines a rough contour and the algorithm aims at extracting an accurate contour path from it. First, all the pixels that do not belong to the rough contour are considered forbidden. Then, taking advantage of the user interaction, we consider that the rough contour path is ordered and follows the true contour. Thus, intermediate control points regions are automatically spread at regular intervals within the rough contour. Using the rough contour tool has several advantages. First, the extraction process is efficient since the space problem is constrained. Second, it is certainly more convenient for the user since, unlike the first two tools that often result in a trial-and-error procedure; this one only needs one simple fast interaction.

(a) (b) (c)

Fig. 2. Experimental results (a) The contour detector by previous method (b) contour detector by proposed method

4. PERFORMANCE PARAMETER

We also calculate the performance using the standard boundary-based methodology established in [39]. This structure first gets the optimal boundary corresponding between the testing segmentation and the ground-truth, and then calculates the results from two phases: Precision and Recall. Given the testing segmentation C_{test} and the ground truth segmentationC_{gt}, the Precision measures the fraction of identified boundary pixels that match the ground-truth boundaries, and is well-defined as

$$\text{Precision} = \frac{|C_{test} \cap |C_{gt}|}{|C_{test}|} \qquad (12)$$

Where $|C|$ is the number of boundary pixels in the segmentationC. Similarly, the Recall is defined as

$$\text{Recall} = \frac{|C_{test} \cap |C_{gt}|}{|C_{gt}|} \qquad (13)$$

Which measures the fraction of ground-truth boundary pixels that are detected To encapsulate these two indexes, the global Fα-measure, defined in (14), is used to measure the harmonic mean of the Precision and Recall. We set $\alpha = 0.5$ and use it for all the experiments

$$F\alpha = \frac{\text{Precision} \cdot \text{Recall}}{(1-\alpha)\cdot \text{Recall} + \alpha \cdot \text{Precision}} \qquad . (14)$$

Table II lists the Precision and Recall values for different algorithms under the same settings as the region level evaluation. The table 1 shows the comparison parameters in which the precision is improved recall rate is approx. same.

Table 1 Parameter comparison for stone image

Methods	Precision	Recall	F_α Measure
Previous Method	0.733	0.508	0.60
Proposed Method	0.785	0.505	0.63

Table 1 Parameter comparison for man image

Methods	Precision	Recall	F_α Measure
Previous Method	0.813	0.518	0.61
Proposed Method	0.819	0.512	0.62

5. CONCLUSIONS

This paper shows an efficient image contour detection algorithm taking advantages of the scalability of modularity optimization and the inherent properties of images. Adopting the bottom-up framework, the proposed algorithm automatically detects the number of segments in the image, and by employing the color feature as well as the proposed HoS texture feature; it adaptively constructs the similarity matrix among different regions, optimizes the modularity, and aggregates the neighboring regions iteratively. The optimal contour detection is achieved when no modularity increase occurs by aggregating any neighboring regions. Results of extensive experiments have validated that the proposed algorithm gives impressive qualitative contour detection results. Because the algorithm aims to avoid over-contour detection, it produces low Recall value. In addition, it is demonstrated that the new algorithm can preserve regularities in the object and achieve the best performance from the semantic level on SSDS. In upcoming, the more refined design of region/contour cues could benefit to extract contours of complex objects.

6. REFERENCES

[1] S. Li and D. Oliver Wu "Modularity-Based Image Segmentation" IEEE Transactions on Circuits And Systems For Video Technology, Vol. 25, No. 4, April 2015

[2] B. Bhanu and J. Peng, "Adaptive integrated image segmentation and object recognition," IEEE Trans. Syst., Man, Cybern. C, Appl. Rev., vol. 30, no. 4, pp. 427–441, Nov. 2000.

[3] 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.

[4] 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.

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

[6] P. Arbeláez, M. Maire, C. Fowlkes, and J. Malik, "From contours to regions: An empirical evaluation," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2009, pp. 2294–2301.

[7] S. Rao, H. Mobahi, A. Yang, S. Sastry, and Y. Ma, "Natural image segmentation with adaptive texture and boundary encoding," in Proc. Asian Conf. Comput. Vis. (ACCV), 2010, pp. 135–146.

[8] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," IEEE Trans. Pattern Anal. Mach. Intell., vol. 33, no. 5, pp. 898–916, May 2011.

[9] H. Zhu, J. Zheng, J. Cai, and N. M. Thalmann, "Object-level image segmentation using low level cues," IEEE Trans. Image Process., vol. 22, no. 10, pp. 4019–4027, Oct. 2013.

[10] M. Wertheimer, "Laws of organization in perceptual forms," in A Source Book of Gestalt Psychology. Evanston, IL, USA: Routledge, 1938, pp. 71–88.

[11] D. D. Hoffman and M. Singh, "Salience of visual parts," Cognition, vol. 63, no. 1, pp. 29–78, 1997.

[12] T. Cour, F. Benezit, and J. Shi, "Spectral segmentation with multiscale graph decomposition," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), vol. 2. Jun. 2005, pp. 1124–1131.

[13] J. Wang, Y. Jia, X.-S. Hua, C. Zhang, and L. Quan, "Normalized tree partitioning for image segmentation," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2008, pp. 1–8.

[14] C. Couprie, L. Grady, L. Najman, and H. Talbot, "Power watershed: A unifying graph-based optimization framework," IEEE Trans. Pattern Anal. Mach. Intell., vol. 33, no. 7, pp. 1384–1399, Jul. 2011.

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

[16] V. Grau, A. U. Mewes, M. Alcaniz, R. Kikinis, and S. K. Warfield, "Improved watershed transform for medical image segmentation using prior information," IEEE Trans. Med. Imag., vol. 23, no. 4, pp. 447–458, Apr. 2004.

[17] X.-C. Tai, E. Hodneland, J. Weickert, N. V. Bukoreshtliev, A. Lundervold, and H.-H. Gerdes, "Level set methods for watershed image segmentation," in Scale Space and Variational Methods in Computer Vision. Berlin, Germany: Springer-Verlag, 2007, pp. 178–190.

[18] V. Osma-Ruiz, J. I. Godino-Llorente, N. Sáenz-Lechón, and P. Gómez-Vilda, "An improved watershed algorithm based on efficient computation of shortest paths," Pattern Recognit., vol. 40, no. 3, pp. 1078–1090, 2007.

[19] G. Mori, "Guiding model search using segmentation," in Proc. 10th IEEE Int. Conf. Comput. Vis. (ICCV), vol. 2. Oct. 2005, pp. 1417–1423.

[20] K. N. Plataniotis and A. N. Venetsanopoulos, Color Image Processing and Applications. New York, NY, USA: Springer-Verlag, 2000.

[21] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in Proc. 8th IEEE Int. Conf. Comput. Vis. (ICCV), vol. 2. Jul. 2001, pp. 416–423.