AN ADAPTIVE CLUSTERING AND CHROMINANCE-BASED MERGING APPROACH FOR IMAGE SEGMENTATION AND ABSTRACTION

Lulu He, Thrasyvoulos N. Pappas

EECS Department, Northwestern University
2145 Sheridan Rd, Evanston, IL 60208, USA
lulu-he@northwestern.edu, pappas@eecs.northwestern.edu

ABSTRACT

We present a novel, computationally efficient approach for natural image segmentation that uses the adaptive clustering algorithm (ACA) to obtain an initial segmentation and chrominance-based region merging to consolidate regions of perceptually uniform texture. The combination of ACA and chrominance-based merging preserves salient edges and smooths out noise and edges within textured regions. It can thus be used for image abstraction. Experimental results with natural images indicate the effectiveness of the proposed approach.

Index Terms— Adaptive clustering algorithm, region merging, bilateral filtering.

1. INTRODUCTION

The segmentation of natural images into perceptually uniform regions remains a challenging task [1]. There is a substantial literature on color image segmentation (e.g., [2, 3]) but there has been relatively little work on texture (e.g., [4, 5]). In [6], Chen *et al.* proposed a perceptually-based color-texture segmentation algorithm, with good performance on a wide variety of natural images. However, the computational requirements are quite substantial. The challenge, and primary goal of this paper, is to develop computationally efficient algorithms without significant sacrifices in performance.

One of the key observations that led to the development of the color-texture segmentation algorithm proposed in [6] was that the adaptive clustering algorithm (ACA) [2] is quite successful at segmenting images with regions of smoothly varying intensity but oversegments images with texture. Thus, there is a need to consolidate oversegmented areas into regions of uniform texture. Chen *et al.* [6] did this by introducing two types of features, color composition features that consist of the dominant colors and associated percentages in the neighborhood of each pixel, and spatial texture features that are based on a multiscale frequency decomposition and describe the spatial characteristics of the grayscale component of the texture. However, the computational requirements associated with these features are quite substantial, requiring median filtering to obtain spatial texture, and an elaborate metric for comparing color compositions [7].

In this paper, we propose an alternative approach for consolidating oversegmented textured regions that is a lot less computationally demanding. It is based on the observation that natural textures consist mainly of intensity variations [8] of a single hue (e.g., green leaves, gray building, blue mountains), and that as a result, significant changes in hue occur mainly at region boundaries. This is illustrated in Figure 1, which shows chrominance (hue and saturation) dominant edges on the left and luminance dominant edges on the right, superimposed on the original image. Note that chrominance edges occur mostly at region boundaries, while luminance edges can occur both within regions and at boundaries. This provides a strong clue as to the location of region boundaries, but by itself cannot lead to a good segmentation because of the noise in the data. Note that some edges are completely missing, some are too thick, etc. On the other hand, as we will see, ACA [2] segments the image into regions of uniform color with precise boundary localization. All we have to do, then, is merge the uniform color blobs of the textured areas into one region, utilizing the fact that the boundaries within perceptually uniform areas have very low chrominance contrast. In addition to chrominance, we can also utilize other statistics of textured areas, for example, the size of the uniform color blobs, or equivalently, the length of the edges.

Image segmentation is a key step for many image analysis applications, including content-based retrieval, object recognition, event detection, etc. Segmentation can make images easier to interpret by getting rid of redundant information, emphasizing the most salient image features. As such, one can think of segmentation as providing image abstraction [9, 10]. In addition to image understanding, image abstraction can contribute to efficient visual communication, and of course, it can be used for artistic purposes. In the following, we evaluate the proposed image segmentation techniques in terms of the image abstraction that they can provide.

This paper is organized as follows. In Section 2, we review the adaptive clustering algorithm and compare it to bilateral filtering. The region merging algorithm is presented in Section 3. The experimental results and conclusions are presented in Section 4.

(a) (b)

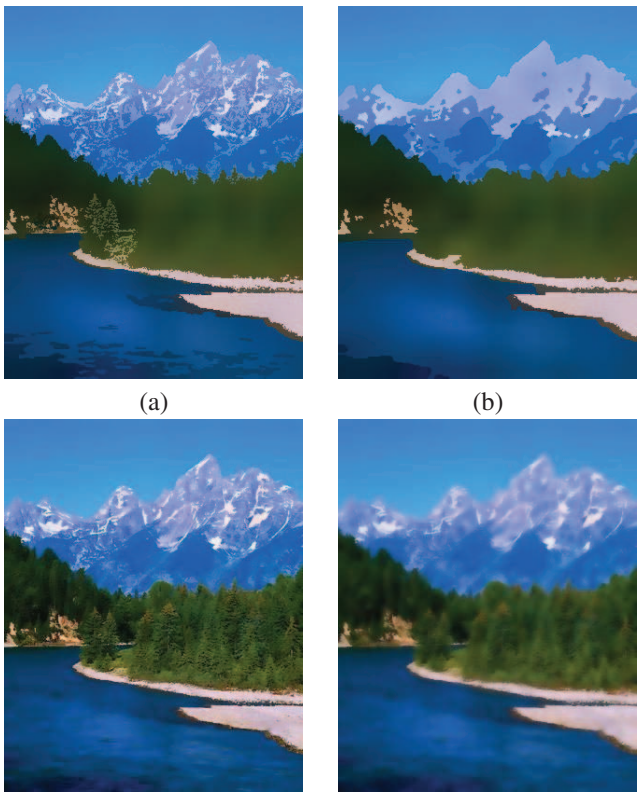
Fig. 1. Classification of edges. (a) Chrominance dominant edges. (b) Luminance dominant edges.

2. REVIEW OF ADAPTIVE CLUSTERING

The Adaptive Clustering Algorithm (ACA) [2] is an iterative algorithm that generalizes K-means clustering by introducing spatial constraints and spatially varying characteristic functions. ACA segments the image into K classes, each of which is characterized by a spatially varying function $\mu^k(s)$, instead of the spatially constant cluster centers of the K-means algorithm; here k denotes the region label and s denotes the pixel location. Given the $\mu^k(s)$ functions, ACA finds the MAP estimate for the segmentation given the observed image. The algorithm alternates between estimating the $\mu^k(s)$ functions and updating the segmentation. At each pixel location s, $\mu^k(s)$ is computed as the average color of the pixels in the neighborhood of s that belong to class k [2]. The initial estimate is obtained by the K-means algorithm. The key to obtaining accurate estimates for the local image characteristics is that the ACA estimates the characteristic functions $\mu^k(s)$ by averaging over a sliding window with progressively decreasing size, starting form global estimates and slowly adapting to the local characteristics of each region.

Once the algorithm converges, we can reconstruct an "abstracted" version of the image, by using the corresponding characteristic function to represent each segment. We refer to this as the *local average* image. This is shown in Figure 2 (a) and (b) for two parameter settings of the algorithm. Note that ACA smooths over noise and small details, while preserving the dominant edges of the scene. By varying the strength of the spatial constraints, or equivalently the estimate of the noise variance [2], we can vary the amount of detail in the reconstructed image. Note, however, that in contrast to other scale-space techniques, the location of the edges does not change. The only changes are in the smoothness of the edges and the fact that some edges disappear.

It is interesting to compare the performance of ACA with *bilateral filtering,* a class of algorithms designed to smooth images while preserving strong edges [11]. Thus, both ACA and bilateral filters can be used to obtain image abstractions.



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

Fig. 2. First row: ACA with different noise variances: (a) $\sigma = 8$; (b) $\sigma = 32$. Second row: bilateral filtering using different sets of parameters: (c) $\sigma_d = 3$ and $\sigma_r = 0.1$. (d) $\sigma_d = 3$ and $\sigma_r = 0.3$.

Bilateral filtering is similar to Gaussian filtering except that it also considers the variation of image intensities. Smoothing is conducted only when two pixels are both spatially close and similar in the photometric range. The definition of the bilateral filter is as follows:

$$BF\left[I\right]_{\mathbf{p}} = \frac{1}{W_{\mathbf{p}}} \sum_{\mathbf{q} \in S} G_{\sigma_s}\left(||\mathbf{p}-\mathbf{q}||\right) G_{\sigma_r}\left(|I_{\mathbf{p}} - I_{\mathbf{q}}|\right) I_{\mathbf{q}} \quad (1)$$

where $W_{\mathbf{p}}$ is a normalization factor:

$$W_{\mathbf{p}} = \sum_{\mathbf{q} \in S} G_{\sigma_s}\left(||\mathbf{p}-\mathbf{q}||\right) G_{\sigma_r}\left(|I_{\mathbf{p}} - I_{\mathbf{q}}|\right) \quad (2)$$

(1) shows that bilateral filtering is the combination of two Gaussian filters: a spatial one G_{σ_s} and a range one G_{σ_r}. The spatial Gaussian acts like the normal Gaussian filter in assigning larger weights to nearby pixels and smaller weights to distant pixels. The effect of G_{σ_r} is to include only pixels with intensity values similar to $I_{\mathbf{p}}$. Thus strong edges with high contrast image intensities on each side are preserved.

From the above analysis, it is clear that the bilateral filter is controlled by two parameters: the spatial domain standard deviation σ_s and the intensity domain standard deviation σ_r. The larger the σ_r, the closer it is to Gaussian blur and the less edges are preserved. Also, increasing σ_s can result in smoothing over larger features. Figure 2 (c) and (d) shows the result

1. Use ACA to segment original image; obtain local average image
2. Do connected components labeling
3. Extract all the edges between regions
4. **For** each pair of regions i and j
   Compute the edge length $L_{i,j}$
   Compute the normalized edge strength $S_{i,j}$
   End
5. Compute the histogram of S_{ij}
6. Calculate thresholds T_{low} and T_{high}
7. Do region merging based on T_{low}
8. Recalculate characteristic functions and update $S_{i,j}$'s
9. Do region merging based on T_{high}

Fig. 3. The proposed algorithm.

of bilateral filtering for two parameter settings. Note that, like ACA, the parameters can be chosen to vary the amount of detail. In Figure 2 (c) the range parameter ($\sigma_r = 0.1$) is small enough for the range Gaussian to play the dominant role, and thus preserve most of the edges. In Figure 2 (d) the range parameter is bigger ($\sigma_r = 0.3$), while the domain parameter remains the same as in (c) ($\sigma_d = 3$). Now the domain Gaussian filter has a more dominant effect, and thus fewer edges are preserved. Note that for $\sigma_r = 0.3$, edges with amplitude changes less than 30% of the full intensity span are blurred.

When we compare the results of bilateral filtering to those of ACA, we observe that, as we discussed above, ACA extracts precise and crisp edges for all parameter settings, while in bilateral filtering the edge blurring varies with the amount of detail. In addition, bilateral filtering may introduce artifacts near the edges, especially in smoother settings. One possible solution to this problem is to do multiple iterations [11, 12]. However, this may introduce ringing or halo artifacts. Overall, the superiority of ACA is obvious with cleanly smoothed regions and well preserved edges without artifacts.

3. CHROMINANCE-BASED MERGING

As we demonstrated in the previous section, ACA is good at preserving accurate edges while smoothing out details. However, it oversegments textured regions. For example, there are small color blobs in the forest, mountain, and water areas of Figure 2(a). Increasing the amount of smoothing in Figure 2(b) eliminate many of the blobs, but also results in substantial smoothing of the edges. Another approach to consolidate the regions of perceptually uniform texture is to do region merging based on chrominance contrast.

We now outline the basic steps of the region merging algorithm. We start with the local average image (see Section2) and the corresponding region specification. First, we perform connected component analysis to identify and label all the connected regions. We then compute the normalized edge strength $S_{i,j}$ between each pair of neighboring regions i and j. That is, for each pixel of region i along the edge, we calculate the chrominance biased *L*a*b* color difference with its 4-connected neighbors that belong to region j. In a typical setting, the luminance is suppressed to half of its original value so that the ratio of chrominance and luminance is 2:1. Then we sum the color differences for all pixels and divide the sum by the number of pixels $L_{i,j}$ on that edge. If there is a big difference, then the edge is strong and should be preserved. Conversely, if the difference is small, then the edge is not significant and the regions should be merged. The luminance suppression ensures that luminance dominant edges within texture regions are more likely to be eliminated. Our merging criterion also takes into account the edge length; typically two regions are merged only when the length of the edge between them is smaller than a threshold. This also favors merging of within-texture boundaries, based on the fact that edges within texture regions are usually short, while edges between different objects/material are much longer.

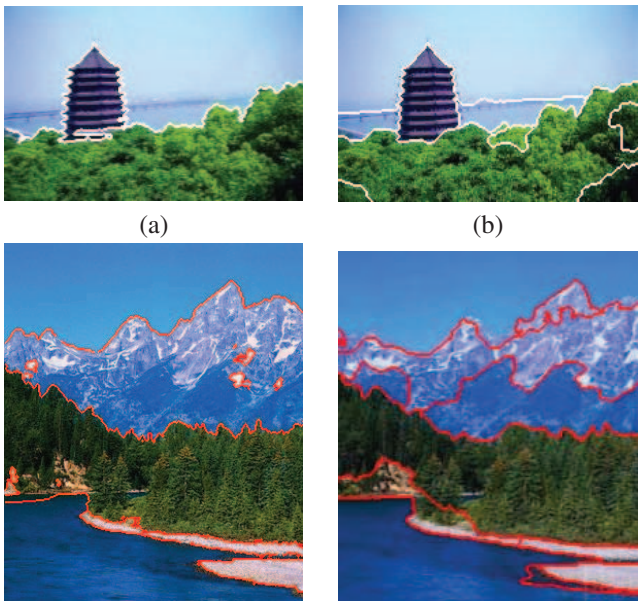


Fig. 4. Left column: final segmentation using proposed approach. Right column: result from [6].

The choice of the merging threshold is crucial. We use two thresholds, a low T_{low} and a high T_{high}. The lower threshold is applied first. This removes all the weak edges. Next, we recompute the characteristic functions of ACA based on the rearranged regions, recompute the normalized edge strengths, and apply the second threshold. The flow of the algorithm is given in Figure 3. Since regions with similar colors have already been merged in the first iteration, the new characteristic function calculations are more robust, and hence more representative of the underlying regions. In addition, the difference between distinct regions becomes larger. Therefore, we need to use a higher merging threshold to get more coherent regions for texture while maintaining important region boundaries. Using two thresholds instead

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

Fig. 5. Abstraction and segmentation results. (a) Original image. (b) ACA local average image. (c) Abstraction result, based on chrominance-oriented region merging. (d) Scene segmentation, with edges superimposed on the original image.

of one gives us more leeway when setting the threshold values. If only one threshold is used, close regions may not be merged if the threshold is too low, while distinct regions may be merged if the threshold is too high. The choice of the two thresholds is image adaptive, and is based on the histogram of edge strengths. T_{high} is usually set at 80% of the edge strengths below it, and T_{low} is set at 40% of the edge strengths below it.

4. EXPERIMENTAL RESULTS

We now consider the performance of the proposed algorithm. Figure 5 shows the abstraction and segmentation result. We can see that the algorithm makes clear distinction among different objects while maintaining coherent regions within each of the texture areas. Additional segmentation results on photographic images of natural scenes are shown on the left column of Figure 4. For comparison, on the right column, we show the segmentation results obtained by Chen *et al.* [6]. One can argue that the proposed algorithm gets more accurate segmentations in terms of texture boundaries and uniformity within each texture region. Note that in Figure 4 (a) the forest is consolidated into one region, while (b) has a couple of oversegmented blobs. On the other hand, there is a problem in the segmentation at the bottom tower. Note also that the water and the sky have been merged due to the small luminance (and no chrominance) difference. The performance of the two algorithms appears to be comparable in the bottom two images. Note, that the proposed algorithm achieves a similar degree of merging with less smoothing of the region boundaries. However, the proposed algorithm is a lot faster than the one in [6].

In summary, we presented a novel computationally efficient algorithm for segmenting natural images into perceptually uniform texture regions. The algorithm combines ACA segmentation with chrominance-based region merging. Experimental results show that the performance of our algorithm is comparable to the state of the art at a much lower computational cost. In future work, we plan to further improve the algorithm performance by incorporating local constraints in the region merging process in the form of spatially adaptive chrominance biasing.

5. REFERENCES

[1] T. N. Pappas, *et al.*, "Perceptually based techniques for image segmentation and semantic classification," *IEEE Commun. Mag.*, vol. 45, no. 1, pp. 44–51, Jan. 2007.

[2] T. N. Pappas, "An adaptive clustering algorithm for image segmentation," *IEEE Tr. Sig. Proc.*, vol. SP-40, no. 4, pp. 901–914, Apr. 1992.

[3] D. Comaniciu and P. Meer, "Robust analysis of feature spaces: Color image segmentation," *CVPR*, San Juan, Puerto Rico, June 1997, pp. 750–755.

[4] Y. Deng and B. S. Manjunath, "Unsupervised segmentation of color-texture regions in images and video," *IEEE Tr. PAMI*, vol. 23, no. 8, pp. 800–810, Aug. 2001.

[5] L. Garcia Ugarriza, *et al.*, "Automatic image segmentation by dynamic region growth and multiresolution merging," *IEEE Tr. Im. Proc.*, vol. 18, no. 10, pp. 2275–2288, Oct. 2009.

[6] J. Chen, *et al.*, "Adaptive perceptual color-texture image segmentation," *IEEE Tr. Im. Proc.*, vol. 14, no. 10, pp. 1524–1536, Oct. 2005.

[7] A. Mojsilović, *et al.*, "Extraction of perceptually important colors and similarity measurement for image matching, retrieval, and analysis," *IEEE Tr. Im. Proc.*, vol. 11, no. 11, pp. 1238–1248, Nov. 2002.

[8] I. Omer and M. Werman, "Color lines: Image specific color representation," *CVPR*, 2004, vol. 2, pp. 946–953.

[9] D. Decarlo and A. Santella, "Stylization and abstraction of photographs," *ACM Tr. Graph.*, vol. 21, pp. 769–776, July 2002.

[10] B. Gooch, *et al.*, "Human facial illustrations: Creation and psychophysical evaluation," *ACM Tr. Graph.*, vol. 23, pp. 27–44, Jan. 2004.

[11] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," *ICCV*, 1998, pp. 839–846.

[12] H. Winnemoller, *et al.*, "Real-time video abstraction," *ACM SIGGRAPH*, July 2006, vol. 25, pp. 1221–1226.