IMAGE SEGMENTATION BY SPATIALLY ADAPTIVE COLOR AND TEXTURE FEATURES

Junqing Chen, Thrasyvoulos N. Pappas

Electrical and Computer Engineering Dept.
Northwestern University, Evanston, IL 60208

Aleksandra Mojsilovic, Bernice E. Rogowitz

IBM T. J. Watson Research Center
Hawthorn, NY 10532

ABSTRACT

We present an image segmentation algorithm that is based on spatially adaptive color and texture features. The proposed algorithm is based on a previously proposed algorithm but introduces a number of new elements. We use a new set of texture features based on a steerable filter decomposition. The steerable filters combined with a new spatial texture segmentation scheme provide a finer and more robust segmentation into texture classes. The proposed algorithm includes an elaborate border estimation procedure, which extends the idea of Pappas' adaptive clustering segmentation algorithm to color texture. The performance of the proposed algorithm is demonstrated in the domain of photographic images, including low resolution compressed images.

1. INTRODUCTION

The field of Content-Based Image Retrieval (CBIR) has made significant advances during the past decade [1, 2]. Many CBIR systems rely on scene segmentation. However, image segmentation remains one of the most challenging problems. While significant progress has been made in texture segmentation (*e.g.,* [3–6]) and color segmentation (*e.g.,* [7–9]) separately, the combined spatial texture and color segmentation problem remains quite challenging [10, 11].

In [12], we presented an image segmentation algorithm that is based on spatially adaptive color and spatial texture features. The perceptual aspects of this algorithm were further developed in [13], including the use of a steerable filter decomposition instead of the discrete wavelet transform. As we saw in [12], the resolution of the spatial texture segmentation is limited because it is defined on a finite neighborhood, while color segmentation can provide accurate and precise edge localization. In this paper, we improve and refine the algorithm presented in [12, 13]. The main structure of the algorithm remains the same, *i.e.,* the color and spatial texture features are first developed independently, and then combined to obtain an overall segmentation. While the color features also remain the same, we use a new set of spatial texture features based on the steerable filter decomposition. The steerable filters combined with a new texture segmentation scheme provide a finer and more robust segmentation into different texture classes (*smooth, horizontal, vertical,* $+45^o$, -45^o, and *complex*). A key to the proposed method is the use of the "max" operator to account for the fact that there is significant overlap between the filters that correspond to the different orientations. This avoids misclassification problems associated with the previously proposed texture extraction technique [13]. The "max" operation is followed by a median type of operation which, as we pointed out in [12], responds to texture within uniform regions and suppresses textures associated with transitions between regions. Finally, we use an elaborate border refinement procedure [13], which extends the idea of the adaptive clustering algorithm [7] to color texture, and results in accurate border locations.

The focus of this work is in the domain of photographic images. The range of topics is essentially unlimited (people, nature, buildings, textures, indoor scenes, etc.). An important assumption is that the images are of relatively low resolution (*e.g.,* 200×200) and occasionally degraded or compressed. The image segmentation results can be used to derive region-wide color and texture features. These can be combined with other segment information, such as location, boundary shape, and size, in order to extract semantic information. A key to the success of the proposed approach is the recognition of the fact that it is not necessary to obtain a complete understanding of a given image: In many cases, the identification of a few key segments (such as "sky," "mountains," "people," etc.) may be enough to classify the image in a given category [14]. In addition, regions that are classified as complex or "none of the above," can also play a significant role in scene analysis.

In Section 2, we review the color feature extraction. Our new approach for spatial texture feature extraction is presented in Section 3. Section 4 discusses the proposed algorithm for combining the texture and color features to obtain an overall segmentation.

2. COLOR FEATURE EXTRACTION

In this section, we review the color feature extraction algorithm that was used in [12]. The main idea is to use spatially adaptive dominant colors as features that incorporate knowledge of human perception [12]. The color feature representation consists of a limited number of locally adapted dominant colors and the corresponding percentage of occurrence of each color within a certain neighborhood:

$$f_c(x, y, N_{x,y}) = \{(c_i, p_i), i = 1, \ldots, M, p_i \in [0, 1]\} \qquad (1)$$

where each of the dominant colors, c_i, is a three dimensional vector in *Lab* space, and p_i are the corresponding percentages. $N_{x,y}$ represents the neighborhood around the pixel at location (x,y). M is the total number of colors in the neighborhood $N_{x,y}$.

The spatially adaptive dominant colors are obtained by the adaptive clustering algorithm (ACA) proposed in [7] and extended to color in [8]. The ACA is an iterative algorithm that uses spatial constraints in the form of Markov random fields (MRF). The algorithm starts with global estimates (obtained using the K-means algorithm) and slowly adapts to the local characteristics of each region. We found that a good choice for the number of (locally adapted) dominant colors is $M = 4$.

This material is based upon work supported by the National Science Foundation (NSF) under Grant No.CCR-0209006. Any opinions, findings and conclusions or recomendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF.

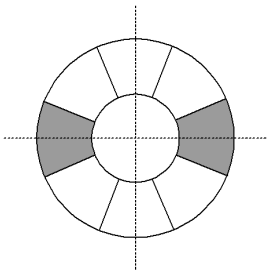


Fig. 1. One-Level Steerable Filter Decomposition

Note that the ACA was developed for images of objects with smooth surfaces and no texture. In textured regions, the ACA oversegments the image, but the segments do correspond to actual texture details. Thus, some other mechanism is needed to consolidate these small segments into regions. Such a mechanism is provided by the "local histograms" we describe next and the texture classes we present in the next section.

Color Features: The ACA provides a segmentation of the image into classes. In the example of Fig. 5(b), each pixel is painted with the average color of the pixels in its neighborhood that belong to the same class [7]. Assuming that the dominant colors are slowly varying, we can assume that they are approximately constant in the immediate vicinity of a pixel. We can then count the number of pixels in each class within a given window, and average their color values to obtain a feature vector that consists of a few (up to four) dominant colors and the associated percentages. Thus, the color feature vector at each pixel is essentially a crude "local histogram" of the image.

Color Metric: To measure the perceptual similarity of two color feature vectors, we use the "Optimal Color Composition Distance (OCCD)" proposed by Mojsilovic *et al.* [15]. The OCCD was designed to provide the optimal mapping between the dominant colors of two images, and thus obtain a better measure of their similarity. Since we are using OCCD to compare local histograms that contain only four bins, its implementation can be simplified considerably [13].

3. SPATIAL TEXTURE FEATURE EXTRACTION

As in [12], the spatial texture feature extraction is independent from that of color. Thus, we use the gray-scale component of the image to obtain the spatial texture features, based on which we obtain an intermediate segmentation, which is then used together with the color features described in the previous section to produce the final image segmentation.

As indicated in [13], the steerable filter decomposition [16] provides a finer frequency decomposition that more closely corresponds to human visual processing. We use a one-level steerable filter decomposition with four orientations which provide four texture classes: horizontal, vertical and two diagonal directions ($+45^o$ and -45^o), as shown in Fig. 1.

As noted in our earlier work [12], many textures are not directional; thus, it is necessary to include a complex or "none of above" category. Even though such a category does not provide much information about a given region, it nevertheless plays an important role in the overall image classification. Note that even with the addition of the diagonal texture categories, the texture description is still quite crude. However, unlike the texture synthesis problem that requires a very precise model in order to accurately synthesize a wide range of textures, a crude model can be quite adequate for segmentation. Such a simple model is actually the key to obtaining good segmentations from low-resolution compressed images (*e.g.*, 200×200 pixels).

Fig. 2 shows a circular cross-section from the steerable filter responses. The x-coordinate denotes spatial orientation in degrees. Thus, the filter with peak at 0^o represents the horizontal subband s_0, the filter with peak at 90^o represents the vertical subband s_2, and so on. Note that there is a large overlap between neighboring filters. In [13], we account such overlap by comparing the 1st and 2nd maximum among the four subband coefficients. However, this method could misclassify, as complex, textures with orientations that fall between the main orientations of the steerable filters. That's because for such textures the responses of the two filters are close. The complex category should instead be reserved for textures with many different orientations. Note that using sharper orientation filters will narrow the range of misclassified orientations, but will not entirely eliminate the problem. As we will see below, we solve this problem by introducing a "max" operator, and using the local histogram of orientations to determine the texture orientation.

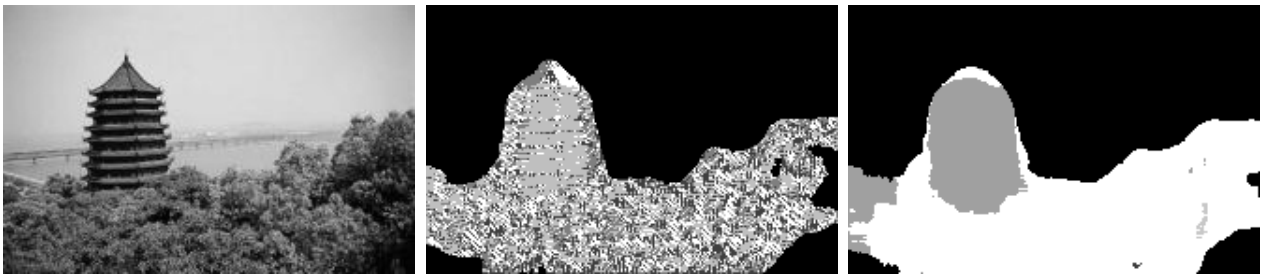
The first step in the texture classification is to identify and locate the smooth regions in the image. We use $s_0(x,y)$, $s_1(x,y)$, $s_2(x,y)$, $s_3(x,y)$ to represent the steerable subband coefficients at location (x,y) that correspond to the horizontal (0^o), diagonal with positive slope ($+45^o$), vertical (90^o), and diagonal with negative slope (-45^o) directions, respectively. For each image location (x,y), we find the maximum of the four coefficients, denoted by $s_{\max}(x,y)$. The subband index $s_i(x,y)$ that corresponds to that maximum is also stored for use in the next step. Then, a median operation is performed on $s_{\max}(x,y)$. Recall that the values in $s_{\max}(x,y)$ come from four different subbands; thus, the cross-subband median can only help in determining whether a pixel belongs to a smooth region, not which texture class it belongs to. Finally, a two-level K-means algorithm, segments the image into smooth and non-smooth regions.

A cluster validation step is necessary at this point. If the clusters are too close, then the image may contain only smooth or non-smooth regions, depending on the actual value of the cluster center.

We have also experimented with alternative ways to obtain smooth vs. non-smooth classification. For example, we tried an approach similar to the one described in [12], whereby a median is applied to each subband followed by a 2-level K-means. A pixel is then classified as smooth if all subbands are classified in the lower class. This leads to similar results yet involves much more computation. Another approach is to apply a median to each subband, followed by K-means applied to the vector of the four subband coefficients. We found that the proposed algorithm has the best performance in terms of accuracy and robustness.

The next stage is to further classify the pixels in the non-smooth regions. As we discussed above, there is significant overlap between neighboring directional filters. Thus, even in a texture of a single orientation (e.g., horizontal), the responses of the two neighboring filters will still be significant. Thus, the maximum of the four coefficients is the one that carries significant information about the texture orientation. Based on this observation, we use the index $s_i(x,y)$ of the subband with the maximum value in order to determine the texture orientation for each pixel location.

We then consider a window, and find the percentage of indices for each orientation. Only non-smooth pixels within the window are considered. If the maximum of the percentages is higher than a given threshold (e.g., 36%) and the difference between 1st and 2nd maximum is significant (e.g., greater than 15%), we conclude that



(a) Original image (b) Max index (c) Texture Classes

Fig. 3. Texture Map Extraction

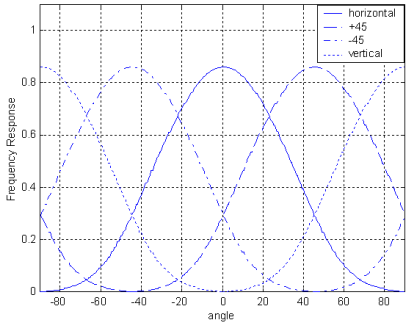


Fig. 2. Steerable Filter Frequency Response

there is a dominant orientation in the window and the pixel is classified accordingly. Otherwise, there is no dominant orientation, and the pixel is classified as complex. Thus, our texture classification is based on the local histogram of the indices corresponding to maximum subband values. This is essentially a "median" type of operation, which is necessary, as we saw in [12], for boosting the response to texture within uniform regions and to suppress the response due to textures associated with transitions between regions. An example is given in Fig. 3. Fig. 3 (a) shows the grey-level component of the original color image. Fig. 3(b) shows the matrix s_i of indices indicating the maxima. The smooth regions are shown in black, while the other 4 orientations are shown in shades of gray. Fig. 3 (c) shows the resulting texture classes, where black denotes smooth, white denotes complex, and light grey denotes horizontal textures. The window used in this example was 23×23.

The window size for median operation should be reasonably big to obtain an accurate estimate of the histogram and to suppress texture edges. On the other hand, a very big window will result in texture classes too crude to be useful. Our experiments indicate that a window size in the range of 17×17 to 29×29 works well. Since the texture classes are obtained through window operations, we know the texture boundaries are not accurate. Thus, we have to rely on the color texture features to obtain a more accurate segmentation.

4. FINAL SEGMENTATION

We now discuss the combination of texture and color features to obtain the final segmentation. First, we consider the smooth texture regions. As in [12], we rely on the color segmentation which provides regions of different uniform colors. Recall that the ACA provides slowly varying color. To avoid oversegmentation, we find all the connected segments that belong to different color classes, and then merge neighboring segments if the average color difference across the common border is below a given threshold. Finally,

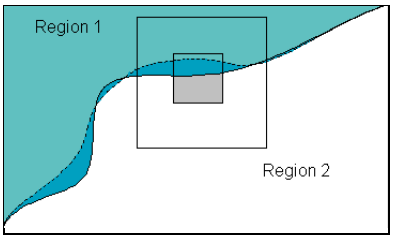
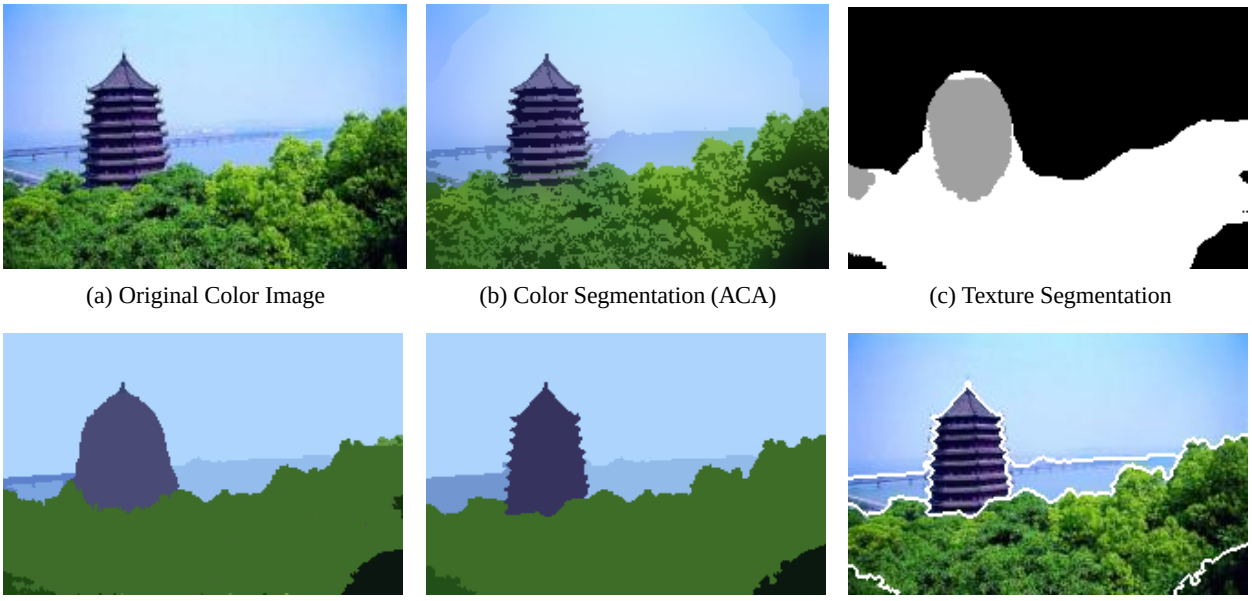


Fig. 4. Illustration of border refinement.

any remaining small regions neighboring non-smooth texture regions are relabeled as complex so that they can be considered in the next step.

Next, we consider the *non-smooth* texture regions. First, we use a region growing approach to obtain an initial "crude" segmentation that is based on the grayscale texture classification and the color features presented in Section 2. Since both of these features are slowly varying, we use a multi-grid approach. We start from pixels located on a coarse grid. We set the window size for the color feature equal to twice the grid spacing, i.e., there is a 50% window overlap. A pair of pixels belong to the same region if the color features are similar in the OCCD sense. The threshold is higher for pixels that belong to the same texture class (i.e., easier to merge), and lower for pixels in different texture classes. In addition, we use MRF-type spatial constraints (as in [13]). That is, a pixel is more likely to belong to a region if many of its neighbors belong to the same region. The symmetric MRF constraint makes it necessary to iterate a few times for a given grid spacing. The grid spacing is then reduced, and the procedure repeated until the grid spacing is equal to one pixel.

Finally, the crude segmentation is refined using an adaptive algorithm similar in nature to the ACA [7]. Fig. 4 illustrates the idea. The dotted line in Fig. 4 represents the real boundary and the solid line denotes the boundary given by our algorithm in current iteration or an initial segmentation obtained in the previous step. Given a segmentation, we use two windows to update the classification of each pixel. The larger window provides a localized estimate of texture characteristics of each region that overlaps the window. For each texture, within the segmentation boundaries, we find the average color and the corresponding percentage for each of the dominant colors. The smaller window provides an estimate of the pixel texture. This consists of the dominant colors corresponding percentages within the smaller window ignoring the current boundary. Then the texture of the pixel is compared with the textures of the different regions using the OCCD criterion. The procedure is repeated for each pixel in a raster scan. As in [7],



(a) Original Color Image (b) Color Segmentation (ACA) (c) Texture Segmentation

(d) Crude Segmentation (e) Final Segmentation (f) Final Segmentation (on original image)

Fig. 5. Color and Texture Image Segmentation (a,b,d,e,f shown in color)

an MRF-type constraint is necessary to insure region smoothness. A few iterations are necessary for convergence. An iteration has converged when the number of pixels that change class is below a given threshold. The overall procedure must then be iterated for a series of window pairs starting from 35/5 and ending with 11/3.

One of the important details in the above procedure is that each of the candidate regions in the larger window must be large enough in order to obtain a reliable estimate of its texture attributes. Otherwise, the region is not a valid candidate. A reasonable choice for the threshold for deciding whether a region should be a valid candidate is to use the product of the two window sizes divided by 2. The crude and final segmentation results are shown in Figs. 5(e) and (f).

The use of both color and texture information to estimate region boundaries finds further justification in psychophysical experiments [17], which showed that the perceived edge location is a combination of the position signaled by texture and by other cues (motion, luminance, color etc).

5. REFERENCES

[1] Y. Rui, T.S. Huang, and S.-F. Chang, "Image retrieval: Current techniques, promising directions and open issues," *J. Vis. Comm. Im. Repr.*, vol. 10, pp. 39–62, Mar. 1999.

[2] W.M. Smeulders, *et al.*, "Content-based image retrieval at the end of the early years," *IEEE Tr. PAMI*, vol. 22, pp. 1349–1379, Dec. 2000.

[3] A. Kundu and J.-L. Chen, "Texture classification using QMF bank-based subband decomposition," *CVGIP, GMIP*, vol. 54, pp. 369–384, Sept. 1992.

[4] T. Chang and C.-C.J. Kuo, "Texture analysis and classification with tree-structured wavelet transform," *IEEE Tr. Im. Pr.*, vol. 2, pp. 429–441, Oct. 1993.

[5] M. Unser, "Texture classification and segmentation using wavelet frames," *IEEE Tr. Im. Pr.*, vol. 4, pp. 1549–1560, Nov. 1995.

[6] T. Randen and J. H. Husoy, "Texture segmentation using filters with optimized energy separation," *IEEE Tr on Im. Pr.*, vol. 8, pp. 571–582, Apr. 1999.

[7] T.N. Pappas, "An adaptive clustering algorithm for image segmentation," *IEEE Tr. SP*, vol. 40, pp. 901–914, Apr. 1992.

[8] M.M. Chang, M.I. Sezan, and A.M. Tekalp, "Adaptive Bayesian segmentation of color images," *JEI*, vol. 3, pp. 404–414, Oct. 1994.

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

[10] S. Belongie, *et al.*, "Color- and texture-based image segmentation using EM and its application to content based image retrieval," *ICCV*, 1998, pp. 675–682.

[11] Y. Deng and B.S. Manjunath, "Unsupervised segmentation of color-texture regions in images and video," *IEEE Tr. PAMI*, vol. 23, pp. 800–810, Aug. 2001.

[12] J. Chen, T. N. Pappas, A. Mojsilovic, and B. Rogowitz, "Adaptive image segmentation based on color and texture," *Proc. ICIP 2002*, Rochester, New York, Sept. 2002.

[13] J. Chen, T. N. Pappas, A. Mojsilovic, and B. Rogowitz, "Perceptual color and texture features for segmentation," *Human Vision and Electronic Imaging VIII*, Jan. 2003, Proc. SPIE Vol. 5007.

[14] A. Mojsilovic and B. Rogowitz, "Capturing image semantics with low-level descriptors," *Proc. ICIP*, Oct. 2001, pp. 18–21.

[15] A. Mojsilović, J. Hu, and E. Soljanin, "Extraction of perceptually important colors and similarity measurement for image matching, retrieval, and analysis," *IEEE Tr. Im. Pr.*, vol. 11, no. 11, pp. 1238–1248, Nov. 2002.

[16] E.P. Simoncelli and W.T. Freeman, "The steerable pyramid: A flexible architecture for multi-scale derivative computation," *Proc. ICIP, vol. III*, Oct. 1995, pp. 444–447.

[17] J. Rivest and P. Cavanagh, "Localizing contours defined by more than one attibute," *Vision Research*, vol. 36, pp. 53–66, 1996.