

# ADAPTIVE IMAGE SEGMENTATION BASED ON COLOR AND TEXTURE

Junqing Chen\*, Thrasyvoulos N. Pappas†

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

Aleksandra Mojsilovic, Bernice Rogowitz

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

## ABSTRACT

We propose an image segmentation algorithm that is based on spatially adaptive color and texture features. The features are first developed independently, and then combined to obtain an overall segmentation. Texture feature estimation requires a finite neighborhood which limits the spatial resolution of texture segmentation, while color segmentation provides accurate and precise edge localization. We combine a previously proposed adaptive clustering algorithm for color segmentation with a simple but effective texture segmentation approach to obtain an overall image segmentation. Our focus is in the domain of photographic images with an essentially unlimited range of topics. The images are assumed to be of relatively low resolution and may be degraded or compressed.

## 1. INTRODUCTION

The field of Content-Based Image Retrieval (CBIR) has made significant advances during the past decade [1, 2]. A number of practical systems have been proposed, and the MPEG-7 standard specifies descriptors for visual content [3]. One of the most challenging problems is image segmentation. While significant progress has been made in texture segmentation (*e.g.*, [4–7]) and color segmentation (*e.g.*, [8–10]) separately, the combined texture and color segmentation problem is considerably more challenging [11–13].

We propose an image segmentation algorithm that is based on spatially adaptive color and texture features. These features are based on perceptual models and principles about the processing of texture and color information. The color and texture features are first developed independently, and then combined to obtain an overall segmentation. One of the most significant differences between the color and texture features is that texture requires a finite neighborhood to be defined. This limits the resolution of texture segmentation. Color segmentation, on the other hand, can provide very accurate and precise edge localization. To accomplish this goal, our color segmentation is based on the adaptive clustering algorithm [8]. For texture analysis, we use an estimate of the energy of the coefficients of a wavelet decomposition [14]. The key to the proposed approach is the use of the median operator for estimating the texture energy in a window around each pixel. The advantage of the median is that it responds to texture within uniform regions and suppresses textures associated with transitions between regions. Finally, in order to combine the texture and color information, we start with the texture segmentation and use the color segmentation to refine it.

Our focus is in the domain of photographic images, but with an essentially unlimited range of topics (people, nature, buildings, textures, objects, indoor scenes, etc.). The images are assumed to

be of relatively low resolution (*e.g.*,  $200 \times 200$ ) and occasionally degraded or compressed. This has the additional advantage that image access and processing time is significantly reduced.

The image segmentation results can be used to derive region-wide color and texture features, which in turn, together with the segment location, boundary shape, and region size, can be used 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, it is enough to identify a few features (color composition, presence of key segments such as “sky,” “mountains,” “people,” etc.) to be able to classify it in a given category [15]. Thus, some of the regions will be classified as “complex” or “none of the above,” and as such will still play a significant role in scene analysis.

In Section 2, we discuss color feature extraction. Our approach for texture feature extraction is presented in Section 3. Section 4 discusses the combination of texture and color features to obtain an overall segmentation.

## 2. COLOR FEATURE EXTRACTION

Color has been used extensively as a low-level feature for image retrieval [1, 3]. Many of the existing techniques are based on the color image histogram. Even though such techniques have been quite successful in given settings, they have some notable shortcomings. First, the histogram does not incorporate any spatial information. Second, the color histogram is too finely quantized in color space, and hence, does not take into consideration the fact that the human visual system can only perceive a few colors at a time. The proposed approach attempts to obtain low-level color features that incorporate knowledge of human perception.

One of the most important characteristics of human color perception is that the human eye cannot simultaneously perceive a large number of colors [16]. Moreover, the number of colors that can be internally represented and identified in cognitive space is about 30 [17]. In addition to providing a very efficient representation, the compact set of color categories also makes it easier to capture invariant properties in object appearance [18].

Ma *et al.* [11] were the first to propose a compact color representation in terms of dominant colors for image segmentation and retrieval. The representation they proposed consists of the dominant colors and the corresponding percentage of occurrence of each color:

$$f_c = \{(c_i, p_i), i = 1, \dots, N, p_i \in [0, 1]\} \quad (1)$$

where each of the dominant colors,  $c_i$ , is a three dimensional vector in *RGB* space, and  $p_i$  are the corresponding percentages.

There are a number of approaches for extracting the dominant colors. One approach is to use vector quantization (VQ) to obtain

\*Summer intern at IBM T. J. Watson Research Center, 2001

†Consultant at IBM T. J. Watson Research Center, September 2001

a set of colors which minimize the mean-square quantization error for all the pixels of the images in a given database [11, 16]. This requires a large training set, which dramatically increases the computational burden for the codebook design. An alternative technique that avoids this problem was proposed by Mojsilovic *et al.* [19]. Rather than trying to find a set of colors that is representative of all images or a particular image database, one can obtain the dominant colors of a given image. For example, one can use VQ on a single image. While the resulting colors may be useful in characterizing the image as a whole, the resulting segmentation could be quite inadequate due to lack of spatial constraints and spatial adaptation [8].

The above dominant color extraction techniques depend on the assumption that the characteristic colors of an image are relatively constant, *i.e.*, they do not change due to variations in illumination, perspective, etc. This is true for images of fabrics, carpets, interior design patterns, and other pure textures that were the primary focus of the work presented in [16]. The class of images that we are considering, however, is more general and includes indoor and outdoor scenes, such as landscapes, cityscapes, plants, animals, people, and man-made objects. To handle such images, one has to account for color and lighting variations in the scene.

A relatively simple and quite effective algorithm that one can use for obtaining the dominant colors of an image in this wider class of images is the color segmentation algorithm proposed by Comaniciu and Meer [10]. It is based on the “mean shift” algorithm for estimating density gradients, and essentially works with the image histogram, even though it also attempts to incorporate spatial constraints by imposing constraints on the connectivity of the detected regions. However, it does not take into consideration that the colors in an image (and hence the dominant colors) may be slowly varying across the image.

Instead, we propose to use **spatially adaptive dominant colors**. This is necessitated by the spatially varying image characteristics and the adaptive nature of the human visual system. For example, an observer’s notion of a blue or brown or green color is highly dependent on the surrounding colors; moreover, it varies with the lighting conditions and the colors of the display device. The spatially adaptive dominant colors can be obtained by the adaptive clustering algorithm (ACA) proposed in [8] and extended to color in [9]. The ACA is an iterative algorithm that uses spatial constraints in the form of Markov random fields (MRF). The initial estimate is obtained by the  $K$ -means algorithm, which estimates the cluster centers (*i.e.*, the dominant colors) by averaging the colors of the pixels in each class over the whole image. As the algorithm progresses, the dominant colors are updated by averaging over a sliding window whose size progressively decreases. Thus, the algorithm starts with global estimates and slowly adapts to the local characteristics of each region.

While the local color adaptation is not suited for comparing whole images, we show that it significantly improves image segmentation, as it virtually eliminates false contours and provides precise boundary location. Moreover, in contrast to the other approaches, the ACA is quite robust to the number of classes. This is because the characteristic levels of each class adapt to the local characteristics of the image, and thus, regions of entirely different intensities can belong to the same class, as long as they are separated in space. We found that the best choice is four classes [8].

Fig. 1 compares the Comaniciu-Meer algorithm [10] (using the “oversegmentation” setting for determining the number of dominant colors) to the ACA. The image resolution is  $250 \times 214$  pix-

els. Note the false contours in the Comaniciu-Meer algorithm in the water and the sky. Also, while there are color variations in the forest region, the segment boundaries do not appear to correspond to any true color boundaries. The ACA on the other hand, smooths over the water, sky, and forest regions, while capturing the dominant edges of the scene. Note that the ACA was developed for images of objects with smooth surfaces and no texture. For some textured regions, like the mountain area, the ACA over segments the image, but the segments do correspond to actual texture details. Thus, we need some other mechanism to consolidate such small segments into regions.

**Color Features:** The ACA provides a segmentation of the image into classes. In the example of Fig. 1(c), each pixel was painted with the average color of the pixels in its neighborhood that belong to the same class [8]. Assuming that the dominant colors are slowly varying, we can assume that they are approximately constant in the immediate vicinity of a pixel. Thus, we can 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.

**Color Metric:** Once we have a (spatially varying) representation of the form (1), we need a metric that measures the perceptual similarity between the feature vectors. Based on human perception, the color composition of two images or image segments is similar, if two conditions are satisfied [16, 19]: The colors are similar, and the corresponding area percentages are similar. The definition of a metric that takes into account both the color and area differences, depends on the mapping between the dominant colors of the two images [19]. Various suboptimal solutions have been proposed [11, 16]. Mojsilovic *et al.* [19] defined a metric that finds the optimal mapping between the dominant colors of the images and call it the “optimal color composition distance (OCCD).” In general, this metric requires more computation than simpler metrics. However, since we are primarily interested in comparing image segments that contain only a few colors (at most four), the additional overhead for the OCCD is reasonable. Moreover, we use a more efficient implementation than the one described in [19].

The proposed color feature vector and associated metric incorporate knowledge of the human visual characteristics. The most important new element of the proposed approach is the fact that the feature vector consists of a small number of spatially varying dominant colors. This variation reflects the spatially varying image characteristics and the adaptive nature of the human visual system.

### 3. TEXTURE FEATURE EXTRACTION

In this section, we try to isolate the textural feature extraction from that of color. We thus consider only the grayscale component of the image to obtain the texture features that can be used to obtain an intermediate segmentation, which can then be combined with the color features to produce the final segmentation. This is in contrast to some other approaches [13, 16] where the color quantization/segmentation is used to obtain an achromatic pattern map which is the basis for texture feature extraction.

Like many of the existing algorithms for texture analysis and synthesis (*e.g.*, [6, 20]), our approach is based on a multiscale frequency decomposition. Such decompositions have been widely used as descriptions of early visual processing in mammals and have also been used as the basis for texture classification and segmentation (*e.g.*, [21–23]).

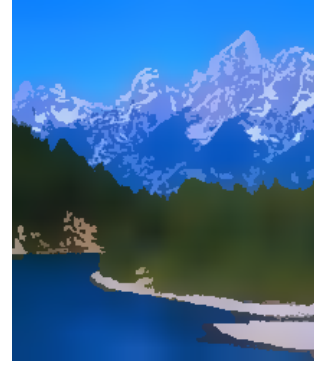
While the texture synthesis problem requires a precise model in order to accurately synthesize a wide range of textures, the



(a) Original Color Image



(b) Comaniciu-Meer Algorithm



(c) Adaptive Clustering Algorithm (ACA)

**Fig. 1.** Color Image Segmentation (all images shown in color)

model for the segmentation problem can be quite crude. Moreover, we want to capture such information from relatively small images, *e.g.*,  $200 \times 200$  pixels, which may have been compressed or degraded. Thus, we expect our texture models to be a lot simpler than those required for texture synthesis.

One of the key ideas in the proposed approach is that we do not need to recognize every region in an image. Large areas of an image may be classified as “too complex,” or “non of the above,” and as such play a role in overall image classification. In this paper, we use simple texture features to obtain a segmentation in a limited number of texture classes (“smooth,” “horizontal,” “vertical,” and “complex”). In the next section, we show that this simple texture classification can lead to impressive results.

For our texture analysis, we use the 9/7 biorthogonal wavelet decomposition [14] which is separable and computationally efficient. Since the images are fairly small, we obtain a one-level wavelet decomposition, and use the HL and LH bands, where H and L stand for the high-pass and low-pass band in each of the horizontal and vertical orientations. We found that discarding the HH band does not result in any significant loss of visual quality, and hence it should not be critical for texture analysis. We also tried the steerable pyramid [24] but have not found any significant performance differences.

The most commonly used feature for texture analysis in the wavelet domain is the energy of the subband coefficients [4, 5]. Since the coefficients are quite sparse, it is necessary to perform some type of window operation to obtain a more uniform characterization of texture. In [4, 5], the average of the energy of the coefficients in a small window was used. In [11, 20] both the mean and standard deviation of the magnitude of the Gabor transform coefficients were used as texture features. In this paper, we use the **median** of the energy in a window. The advantage of the median is that it tends to filter out textures associated with transitions between regions. In such cases, the increase in wavelet coefficients due to the region boundary is concentrated along the edge and is not picked up by the median operator. Extensive experimentation with the average, median, and maximum operators indicates that the average results in significant smoothing and false textures along the region boundaries, while the maximum provides unreliable results. The median leads to significantly superior results. The size of the window must be large enough to capture the local texture characteristics, but not too large to avoid border effects. We found that for the image resolution and viewing distance under consideration, a  $9 \times 9$  window gives the best results.

Finally, we tried different clustering approaches to obtain the

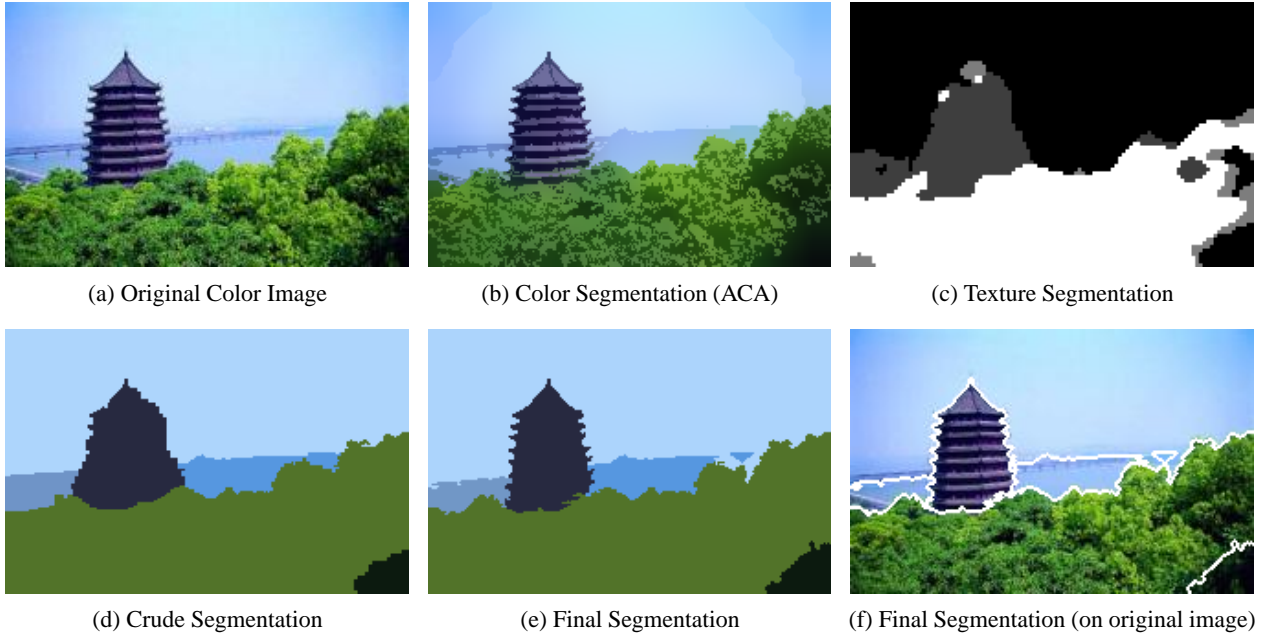
(intermediate) texture segmentation. The simplest and most effective was to apply two-level K-means to each of the horizontal and vertical components separately. One of the cluster centers was always fixed at 0 (smooth texture) and the other was determined by the K-means algorithm. The added advantage of this approach is that we obtain four texture classes with obvious interpretations: smooth texture (0,0), vertical texture (1,0), horizontal texture (0,1), and complex texture (1,1). Fig. 2(c) shows the results, with smooth texture represented by black, vertical by light gray, horizontal by dark gray, and complex by white. We also experimented with VQ (K-means applied to the vector of the two coefficients), but found very little difference in performance for most images.

#### 4. COMBINATION OF TEXTURE AND COLOR

We now discuss the combination of the texture and color segmentations to obtain the final image segmentation. We start with the texture segmentation and use the color segmentation to refine it.

First, we consider the “smooth” texture regions. We rely on the color segmentation to determine if they should be further subdivided into segments of different color. For each connected smooth region, we find all the connected segments that belong to different color classes and compute the average color of each segment. Recall that the ACA provides slowly varying color. We then merge the segments whose colors are similar. The comparison is performed in *Lab* color space; the threshold is set at 10% of the distance range. For the remaining segments, we compute the average color difference across the common border, and merge them if it is below a given threshold. Finally, any remaining small regions neighboring non-smooth texture regions are considered together with the textured regions in the next step.

All the other textured regions (“horizontal,” “vertical,” and “complex” regions, as well small border regions from the previous step) are then considered together. We apply a region growing algorithm based on two features: the grayscale texture classification obtained in the previous section and the color features presented in Section 2. The latter consist of the dominant colors and their frequencies computed in a  $21 \times 21$  window. For each pair of pixels, we compute the color feature difference according to the OCCD criterion described in Section 2, and merge them if it is below a given threshold. The threshold is higher if the pixels belong to the same texture class, and lower if they belong to different texture classes. The region growing starts from the horizontal and vertical texture regions; the complex regions are considered last. Results are shown in Fig. 2(d). Each region in the figure is painted with a color close to its average.



**Fig. 2.** Color and Texture Image Segmentation (a,b,d,e,f shown in color)

Finally, we adjust the segment borders using color information. That is, for each pixel along the border of the crude segmentation (Fig. 2(d)), we compare the pixel color provided by the ACA algorithm with the local average (over a larger window) of each dominant color of each region on either side of the border, and move it accordingly. If there are two close matches, then we compute the local average across the dominant colors of each region, and use that for the comparison. This process is repeated iteratively until convergence. The final result is shown in Fig. 2(e) and (f).

## 5. REFERENCES

- [1] Y. Rui, T.S. Huang, S.-F. Chang, "Image retrieval: Current techniques, promising directions and open issues," *J. Vis. Comm. Im. Repr.*, v. 10, p. 39–62, Mar. 1999.
- [2] W.M. Smeulders, *et al.*, "Content-based image retrieval at the end of the early years," *IEEE Tr. PAMI*, v. 22, p. 1349–1379, Dec. 2000.
- [3] B.S. Manjunath, *et al.*, "Color and texture descriptors," *IEEE Tr. CSVT*, p. 703–715, June 2001.
- [4] A. Kundu, J.-L. Chen, "Texture classification using QMF bank-based subband decomposition," *CVGIP, GMIP*, v. 54, p. 369–384, Sept. 1992.
- [5] T. Chang, C.-C.J. Kuo, "Texture analysis and classification with tree-structured wavelet transform," *IEEE Tr. IP*, v. 2, p. 429–441, Oct. 1993.
- [6] M. Unser, "Texture classification and segmentation using wavelet frames," *IEEE Tr. IP*, v. 4, p. 1549–1560, Nov. 1995.
- [7] T. Randen, J.H. Husoy, "Texture segmentation using filters with optimized energy separation," *IEEE Tr. IP*, v. 8, p. 571–582, Apr. 1999.
- [8] T.N. Pappas, "An adaptive clustering algorithm for image segmentation," *IEEE Tr. SP*, v. 40, p. 901–914, Apr. 1992.
- [9] M.M. Chang, M.I. Sezan, A.M. Tekalp, "Adaptive Bayesian segmentation of color images," *JEI*, p. 404–414, Oct. 1994.
- [10] D. Comaniciu, P. Meer, "Robust analysis of feature spaces: Color image segmentation," *CVPR*, June 1997, p. 750–755.
- [11] W.Y. Ma, Y.Deng, B.S. Manjunath, "Tools for texture/color based search of images," *Human Vision and Electronic Imaging II*, Feb. 1997, Proc. SPIE, Vol. 3016, p. 496–507.
- [12] S. Belongie, *et al.*, "Color- and texture-based image segmentation using EM and its application to content based image retrieval," *ICCV*, 1998, p. 675–682.
- [13] Y. Deng, B.S. Manjunath, "Unsupervised segmentation of color-texture regions in images and video," *IEEE Tr. PAMI*, v. 23, p. 800–810, Aug. 2001.
- [14] A. Cohen, I. Daubechies, J. C. Feauveau, "Biorthogonal bases of compactly supported wavelets," *Commun. Pure Appl. Math.*, v. 45, p. 485–560, 1992.
- [15] A. Mojsilovic, B. Rogowitz, "Capturing image semantics with low-level descriptors," *ICIP*, Oct. 2001, p. 18–21.
- [16] A. Mojsilović, *et al.*, "Matching and retrieval based on the vocabulary and grammar of color patterns," *IEEE Tr. IP*, v. 1, p. 38–54, Jan. 2000.
- [17] G. Derefeldt, T. Swartling, "Color concept retrieval by free color naming," *Displays*, v. 16, p. 69–77, 1995.
- [18] S.N. Yendrikhovskij, "Computing color categories," *Human Vision and Electronic Imaging V*, Jan. 2000, Proc. SPIE Vol. 3959.
- [19] A. Mojsilović, J. Hu, E. Soljanin, "Extraction of perceptually important colors and similarity measurement for image matching, retrieval, and analysis," *IEEE Tr. IP*, To appear.
- [20] B.S. Manjunath, W.Y. Ma, "Texture features for browsing and retrieval of image data," *IEEE Tr. PAMI*, v. 18, p. 837–842, Aug. 1996.
- [21] J.R. Bergen, E.H. Adelson, "Visual texture segmentation based on energy measures," *JOSA A*, v. 3, p. 99–, 1986.
- [22] M. R. Turner, "Texture discrimination by Gabor functions," *Biol. Cybern.*, v. 55, p. 71–82, 1986.
- [23] J. Malik, P. Perona, "Preattentive texture discrimination with early vision mechanisms," *JOSA A*, v. 7, p. 923–932, 1990.
- [24] E.P. Simoncelli, W.T. Freeman, "The steerable pyramid: A flexible architecture for multi-scale derivative computation," *ICIP, vol. III*, Oct. 1995, p. 444–447.