Appeared in *Pattern Recognition Letters*, vol. 15, pp. 659-668, July 1994.

Moment Based Texture Segmentation[1]

Mihran Tuceryan
European Computer-Industry Research Centre (ECRC)
Arabellastraße 17
8000 München 81, Germany

Abstract

Texture segmentation is one of the early steps towards identifying surfaces and objects in an image. In this paper a moment based texture segmentation algorithm is presented. The moments in small windows of the image are used as texture features which are then used to segment the textures. The algorithm has successfully segmented binary images containing textures with iso-second order statistics as well as a number of gray level texture images.

1. INTRODUCTION

The natural world abounds with textured surfaces. Any realistic vision system that is expected to work successfully, therefore, must be able to handle such input. The process of identifying regions with similar texture and separating regions with different texture is one of the early steps towards identifying surfaces and objects. This process is called texture segmentation and is the major focus of this paper.

Texture analysis has been studied for a long time using various approaches. Various methods perform texture analysis directly upon the gray levels in an image. These include gray level co-occurrence matrix (GLCM) [11], autocorrelation function analysis [11], generalized co-occurrence matrices (GCM) [8], second order spatial averages [10], and two-dimensional filtering in the spatial and frequency domain [6,5,26,27]. Other approaches operate at a symbolic level where a textured image is organized or represented in terms of primitives. Examples of this can be seen in Julesz's work [17,18] and in syntactic texture analysis. Some texture analysis methods, for example, Beck *et al.* have examined the role of spatial frequency

[1.] This research was supported in part by the US National Science Foundation Grant no. CDA 8806599. The facilities of the Pattern Recognition and Image Processing Laboratory at Michigan State University are also gratefully acknowledged.

channels (signal processing level) and perceptual grouping (symbolic level) in texture segregation [1,2]. Model based analysis of textures is another approach that has often been utilized. These methods include statistical modeling such as Markov random fields (MRF) [7,21] and fractal based modeling [23]. For a more detailed review of the various methods in texture analysis see [25].

Computing features that capture textural properties is at the heart of most of these approaches. What is meant by textural properties often depends on perceptual and psychophysical considerations. That is, the success of a particular feature is in its ability to describe textures that agree with human perception. The perceptual task can be both classification and segmentation.

Julesz had conjectured [16,19] that texture pairs with identical second-order statistics cannot be preattentively discriminated by humans, but he later gave counterexamples to this conjecture [17]. An important factor in the preattentive discrimination of such texture pairs appears to be certain shape features of the texture primitives. These features include closure, linearity, terminations, etc. which are called *textons* [17]. The important questions that need to be answered in this theory are whether the list of textons is complete and how to extract textons from images. One way to answer these is to formulate a general framework for feature detection so that the detected features capture the texton information. This is the main motivation of this paper.

In this paper we propose a method of obtaining texture features directly from the gray-level image by computing the moments of the image in local regions. Section 2 defines the moments of a two-dimensional function, describes the computation of texture features from the moments, and presents an algorithm that uses these features to segment the texture images. Section 3 gives experimental results, and Section 4 makes some concluding remarks.

2. TEXTURE SEGMENTATION

Our texture segmentation algorithm is given in the flowchart of Figure 1 and consists of the following steps: (a) Compute the image moments within a small window around each pixel, (b) compute the texture features from these moments by applying a nonlinear transformation followed by an averaging operation, (c) perform an unsupervised clustering of a randomly selected points in the image, and (d) classify every pixel in the image according to the results of step (c).

2.1. Moments

Our algorithm uses the moments of an image to compute texture features. The $(p+q)^{th}$ order moments of a function of two variables $f(x,y)$ with respect to the origin (0,0) are defined as [12]:

$$_{pq} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x, y)x^p y^q dxdy \tag{1}$$

where $p + q = 0, 1, 2, \ldots$. The infinite set of moments $\{m_{p, q}, p + q = 0, 1, \ldots\}$ uniquely determine $f(x, y)$ and vice versa by the following equation[13]:

$$f(x, y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} exp(-j2\pi(ux + vy))\left[\sum_{p = 0}^{\infty}\sum_{q = 0}^{\infty} m_{p, q}\frac{(j2\pi)^{p + q}}{p!q!}u^p v^q\right]dudv \tag{2}$$

Normally the moments are computed over some bounded region R. If the function is equal to one within the region and zero outside the region, the lower order moments (small values of p and q) have well defined geometric interpretations. For example, m_{00} is the area of the region,

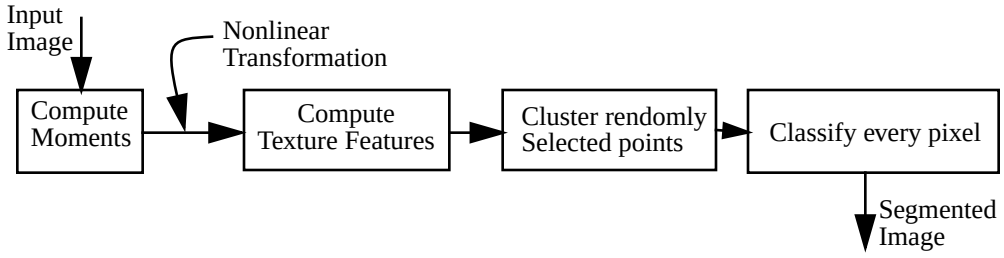


FIGURE 1. The flowchart of the segmentation algorithm.

m_{10} / m_{00} and m_{01} / m_{00} give the x and y coordinates of the centroid for the region, respectively. The m_{20}, m_{11}, and m_{02} can be used to derive the amount of elongation of the region, and the orientation of its major axis. The higher order moments give even more detailed shape characteristics of the polygons such as symmetry, etc. The image moments have been used before in other contexts. Hu [12] has used the moments for character recognition. Dudani et al. [9] have used them for aircraft identification and Zusne has used them for shape description [28]. Moments have also been used previously for characterizing texture [22,24].

In this paper, we regard the intensity image as a function of two variables, $f(x,y)$. We compute a fixed number of the lower order moments for each pixel in the image (we use $p+q \leq 2$). The moments are computed within small local windows around each pixel. Given a window size, the coordinates are normalized to the range [-1,1], the pixel being at the origin. The moments are then computed with respect to this normalized coordinate system. This permits us to compare the set of moments computed for each pixel. The geometry is shown in Figure 2. Let W be the window width. We always choose the window width W to be odd so that the pixel (i,j) is centered on a grid point. Let (i,j) be the pixel coordinates for which the moments are computed. For a pixel with coordinates (m,n) which falls within the window, the normalized coordinates (x_m,y_n) are given by:

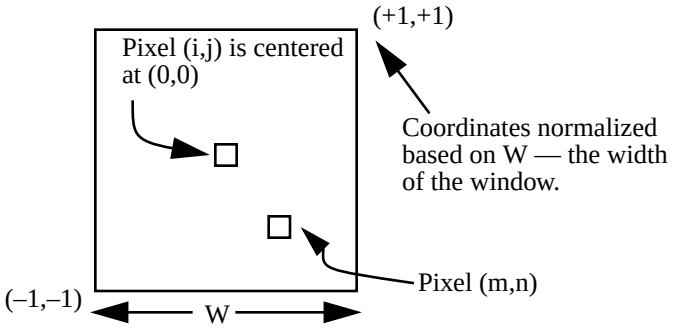


FIGURE 2. The coordinates of the region over which the moments are computed are normalized between -1 and 1. The pixel (i,j) is considered to be at the origin (0,0) of this coordinate system.

$$x_m = \frac{m-i}{W/2} \qquad y_m = \frac{n-j}{W/2} \tag{3}$$

Then the moments within a window centered at pixel (i,j) are computed by a discrete sum approximation of Equation (1) that uses the normalized coordinates (x_m, y_n):

$$_{pq} = \sum_{-W/2}^{W/2} \sum_{-W/2}^{W/2} f(m,n) x_m^p y_n^q \tag{4}$$

This discrete computation of the set of moments for a given pixel over a finite rectangular window corresponds to a neighborhood operation, and, therefore, it can be interpreted as a convolution of the image with a mask. Figure 3 shows the masks corresponding to the moments up to order two with a window size of three. The following important points are noted:

- When we examine these masks, we see that they can be interpreted as local feature detectors. For example, the mask for m_{00} corresponds to a box averaging window and thus can be interpreted as computing the total energy within that box. The masks for m_{10} and m_{01} have the form of edge detectors or contrast detectors. They would respond to sudden intensity changes in the x and y directions, respectively. The second order moments are not as easy to interpret; the only exception being m_{11} which looks like a cross de-

$$m_{00} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \qquad m_{10} = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} \qquad m_{01} = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$$

$$m_{20} = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} \qquad m_{11} = \begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix} \qquad m_{02} = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \end{bmatrix}$$

FIGURE 3. The coordinates of the region over which the moments are computed are normalized between -1 and 1. The pixel (i,j) is considered to be at the origin (0,0) of this coordinate system.

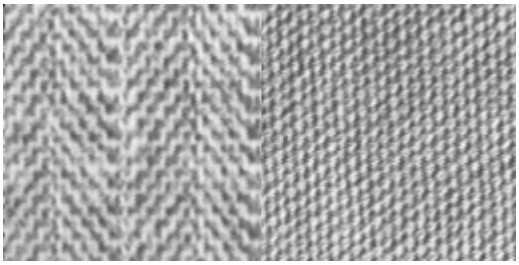


FIGURE 4. A texture pair taken from the Brodatz album: D17 (herringbone weave) and D77 (cotton canvas).

tector.

- The size of the window is important. As the window size gets larger, more global features are detected. This suggests that the choice of window size could possibly be tied to the contents of the image. The images with larger texture tokens would require larger window sizes whereas finer textures would require smaller windows. It is also possible to regard the window size as a space parameter and use a multi-scale filtering approach. In this work, we have done experiments in both aspects.

The set of values for each moment over the entire image can be regarded as a new feature image. Let M_k be the k^{th} such image. If we use n moments, then there will be n such moment images. In our experiments, we used up to second order moments. That is, we used m_{00}, m_{10}, m_{01}, m_{20}, m_{11}, and m_{02} which result in the images M_1, M_2, M_3, M_4, M_5, and M_6, respectively.

The moments alone are not sufficient to obtain good texture features in certain images. Some iso-second order texture pairs which are preattentively discriminable by humans, would have the same average energy over finite regions. However, their distribution would be different for the different textures. One solution suggested by Caelli is to introduce a nonlinear transducer that maps moments to texture features [4]. Caelli suggests that the nonlinear transducer is "usually logistic, sigmoidal, or power function in form." Coggins and Jain [6], on the other hand, use the absolute deviation of their feature vectors from the mean.We have chosen to use

the hyperbolic tangent function as our nonlinear transducer which is logistic in shape. This is followed by an averaging step. We obtain the texture feature image F_k corresponding to the moment image M_k with mean $\overline{M}$ using the following transformation:

$$F_k(i, j) = \frac{1}{L^2} \sum_{(a, b) \in W_{ij}} \left| \tanh(\sigma(M_k(a, b) - \overline{M})) \right| \tag{5}$$

where W_{ij} is an $L \times L$ averaging window centered at location (i,j). σ controls the shape of the logistic function. We have determined by trial and error the value of σ to be 0.01. We have, however, used the same value of σ for all the images in our experiments. Figure 4 shows a texture pair from the Brodatz album [3]. Figure 5 shows its moment images and Figure 6 shows the computed texture feature images after the nonlinear transducer stage. One can see in Figure 6 that the two texture regions have different responses in the third and sixth feature images. Therefore, for this texture pair, these features will play an important role in their segmentation.

2.2. Segmentation Algorithm

If n moments are computed over the image, then each pixel will have n feature values associated with it. For a pixel at (i,j), we define a textural feature vector $\boldsymbol{T}_{ij} = \langle F_1(i,j), \ldots, F_n(i,j) \rangle$ which is a point in an n-dimensional feature space. We perform the texture segmentation by applying a general purpose clustering algorithm to the texture features $\boldsymbol{T}_{ij}$. Because the number of pixels and hence the number of feature points is too large, we use a two step process to produce the segmentation and still have reasonable computational efficiency. This approach is the same as that used by Coggins and Jain in [6]. The two steps of the segmentation process are:

1. Randomly subsample the feature image. In our experiments, we subsampled approximately 1000 pixels in a 256×256 image (which is about 6% of the image). This number is picked solely the performance and computational efficiency of the clustering algorithm in mind and is not an inherent property of the texure features. These 1000 samples

are then clustered using a partitional clustering algorithm called CLUSTER which uses a squared error criterion. This algorithm is described in detail by Jain and Dubes in [14]. Using other clustering methods is also possible, but we have not explored this avenue since the segmentation is working with the CLUSTER algorithm. The result of this step is a segmentation of the feature vectors into a number of clusters (texture classes). Along with this segmentation, the algorithm also provides statistics about each of these classes such as cluster centers and variances within each cluster. This algorithm requires that the user provide the number of clusters. In our experiments, we usually request the correct number of clusters. In some cases, however, we request an

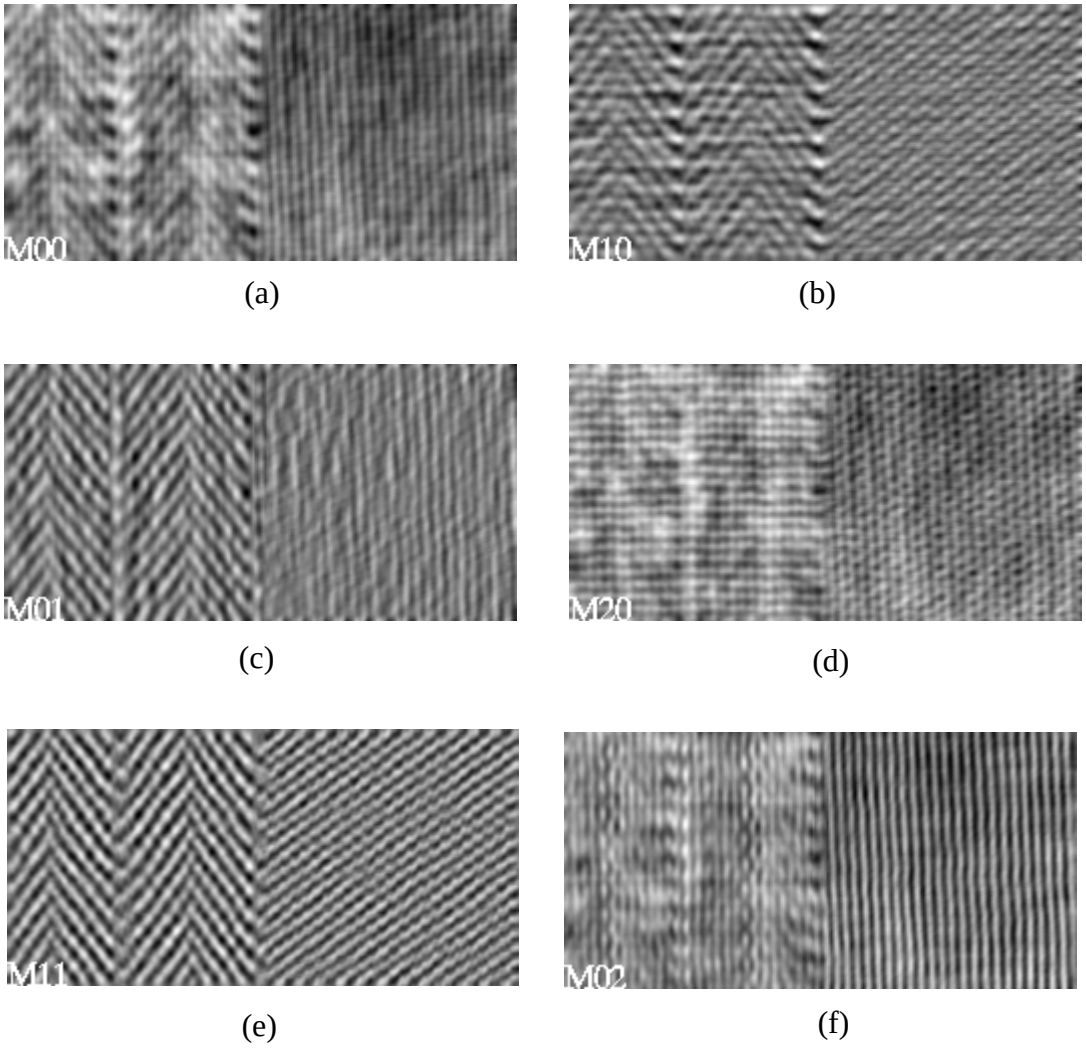


FIGURE 5. The moment images computed for the texture pair in Figure 4.

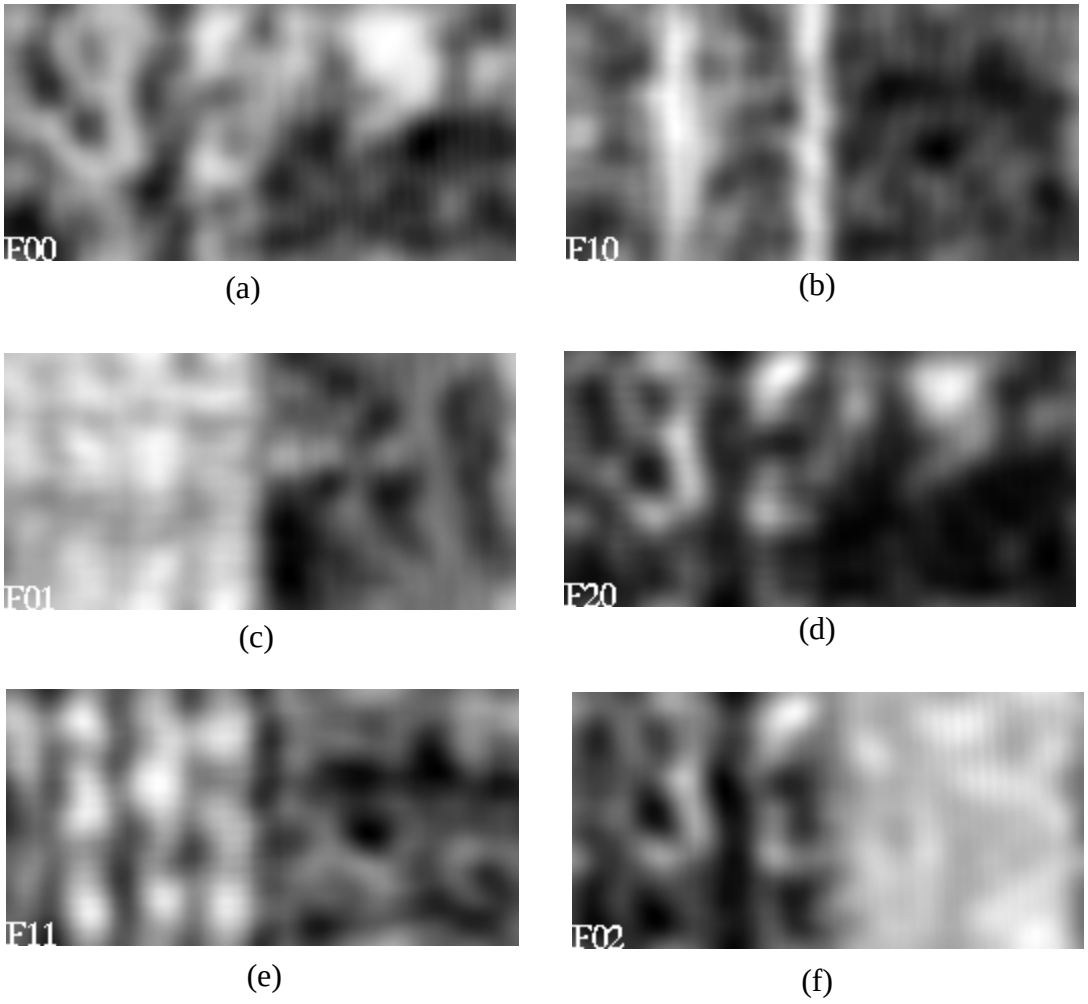


FIGURE 6. The feature images computed for the texture pair in Figure 4.

oversegmentation to overcome border effects (See Figure 8).

2. Classify each pixel in the image using a supervised clustering algorithm. The results of the clustering from the step (a) are used as training points and a minimum distance classifier is used.

The resulting segmentation for the texture pair in Figure 4 is given in Figure 7.

3. EXPERIMENTAL RESULTS

We have tested our segmentation algorithm on both synthetic and natural textures. The reason

for testing our algorithm on synthetic images was to ensure that the features were able to discriminate difficult texture pairs which the humans can discriminate preattentively. This would demonstrate that our texture features were perceptually significant.Figure 8(a) gives a popular example of such textures seen in the literature [17]. In this example, the texture elements which are used to construct the textures have identical second-order statistics. The natural textures were taken from the texture album of Brodatz [3]. Several gray level images of textures were put together in pairs to test our segmentation algorithm. Figures 10(a)– 12(a) are examples of such texture pairs. These are 128x256 images (each Brodatz texture is 128x128) and they have a range of 256 gray levels.

FIGURE 7. The segmentation obtained for the texture pair in Figure 4. The different texture regions are shown as different gray levels.

Our algorithm successfully segments the “corner-closure” texture pattern (Figure 8(a)). In Figure 8(b), we are able to label the two texture regions correctly if we select a three cluster solution. The border effects in this example are substantiaql. The border between the two textures, however, is reasonably localized. This localized border is especially impressive considering that no spatial locality constraints are imposed during processing. We have also run the segmentation algorithm on patterns that consist of textures made up of L’s and crosses, L’s and T’s, etc. [1]. In each of these cases, the algorithm finds the correct segmentation.

The algorithm also successfully segments most of the texture pairs we used from the Brodatz texture album [3]. Figures 10(a)–12(a) give the input gray level images. Figures 10(b)–12(b) give the segmentation results. We have also obtained oversegmentations of these examples. In

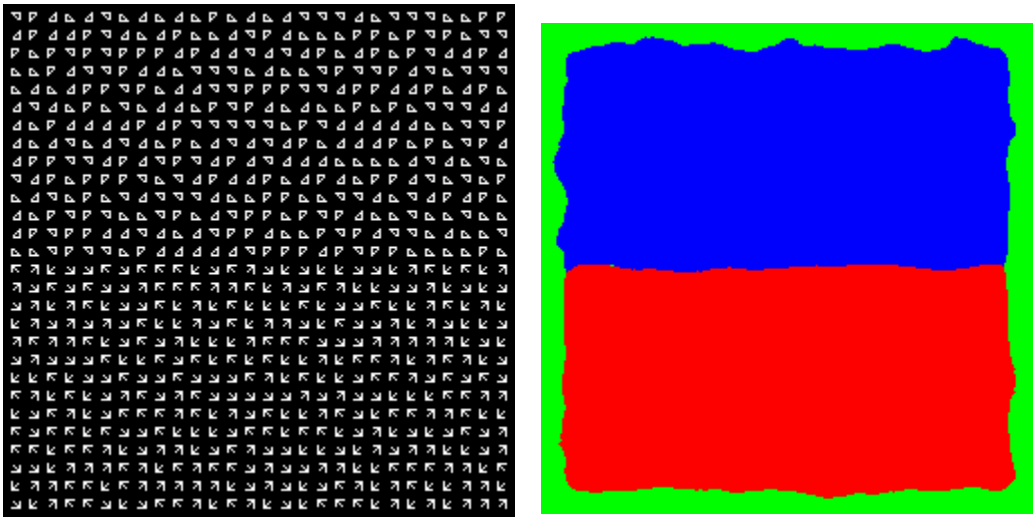
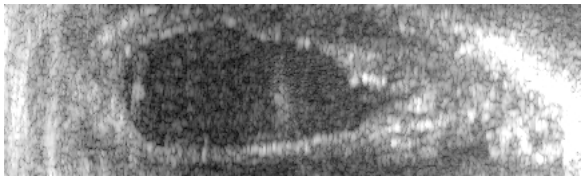


FIGURE 8. (a) A texture pair with identical second order statistics. This texture is preattentively discriminable by humans. (b) The three region segmentation obtained by our algorithm with moment mask size 7 and an averaging window of size 49. Note that we needed a three region segmentation in this case because of the border effects of the various filtering operations. The different texture regions are shown as different gray levels.

all cases, the border between the two textures remains intact. The oversegmentation of the examples in Figures 10 and 12 results in the texture D68 (wood grain) being split along the shadow edge in the middle of the region. We have also tried the segmentation on texture images with more than two textures. Figure 13 shows such a texture image and the segmentation computed by our algorithm.

We have also applied our texture segmentation algorithm to some real world problems including the analysis of ultrasound images of the blood flow in a heart. The ultrasound images in this study are time sequence images of the left ventricle of the heart. Figure 9(a) shows one frame in such a sequence. Texture in this context is defined to be the amount of randomness in the image of the blood flow. The randomness is expected to be higher on an average in blood than in tissue due to the noise and backscatter characteristics of the blood which is more disordered than that of solid tissue. Figure 9(b) shows the result of the segmentation using the moment based texture segmentation.

(a)



(b)

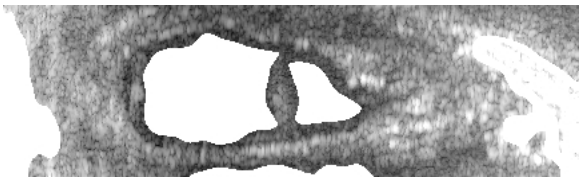


FIGURE 9. (a) The original ultrasound image of the left ventricle of a heart. (b) The result of the segmentation and the region identified as blood (white region) superposed on the original image using the texture segmentation method described in this paper.

4. CONCLUSION

In this paper we have developed a texture segmentation algorithm based on the moments of an image. The algorithm first computes moments within localized regions of the image around each pixel, then it computes a feature vector for each pixel based on these moments. Finally it segments these feature vectors (hence the texture regions) using a partitional clustering algorithm.

The results of the segmentation algorithm show that the image moments computed over local regions provide a powerful set of features that reflect certain textural properties in images. Particularly, the results on the “corner-closure” type texture patterns of Figure 8 show that the important aspects of the textural properties have been captured by this representation.

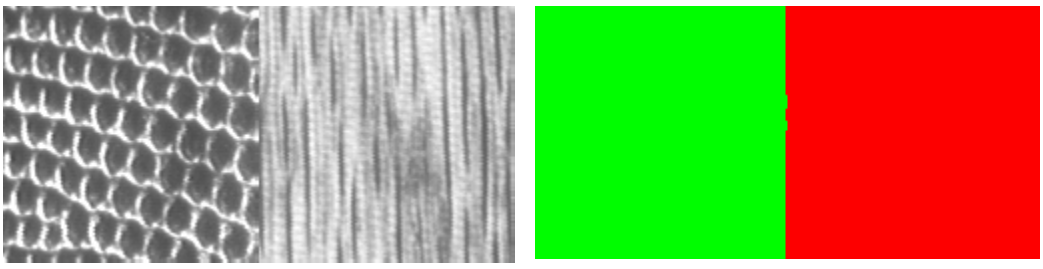
Certain aspects of our algorithm need to be studied more carefully. First, the size of the window within which the moments are computed can be regarded as a scale parameter. This can be seen easily with the zero-order moments where the moment window size corresponds to the size of a box averaging window which is obviously a scale parameter. This window size depends on the content of the image: finer textures (i.e. textures which contain higher spatial frequencies) require a smaller window size in order to detect smaller features, whereas coarser textures (i.e. those with lower spatial frequencies) require larger windows. The window size

may be tied to the frequency content of image. We have done experiments with both hand selected, finely tuned window sizes and with using multiple sized windows simultaneously. In bothe cases the results are comparable.

Second, how many moments need to be computed? In our experiments we used only up to second order moments which seemed to do a good job in segmenting most texture pairs we tried. However, the usefulness of higher order moments needs to be studied more carefully.

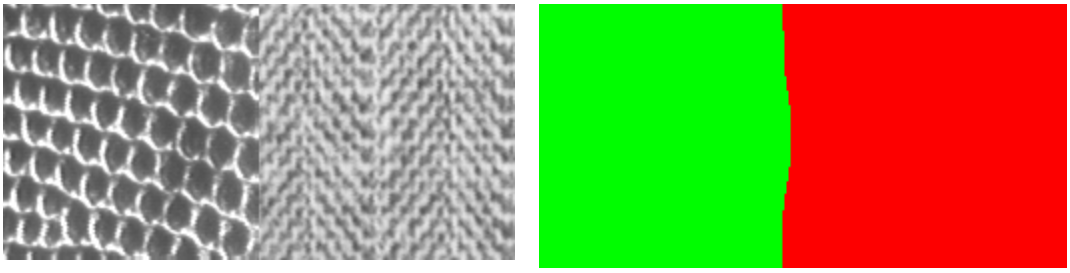
Third, the selection of the averaging window size in obtaining texture features from the moments is also not done automatically. In this case, the window should cover enough texture elements for the features to be meaningful. This window size could also be tied to the spatial frequency content of the texture image.

Finally, the general purpose clustering algorithm requires that we provide the number of clusters to be detected. The possibility of spatial algorithms which utilize the moment based textural features need to be further studied. The algorithm given in [24] may be one candidate. Local spatial continuity constraints such as border smoothness can also be enforced.



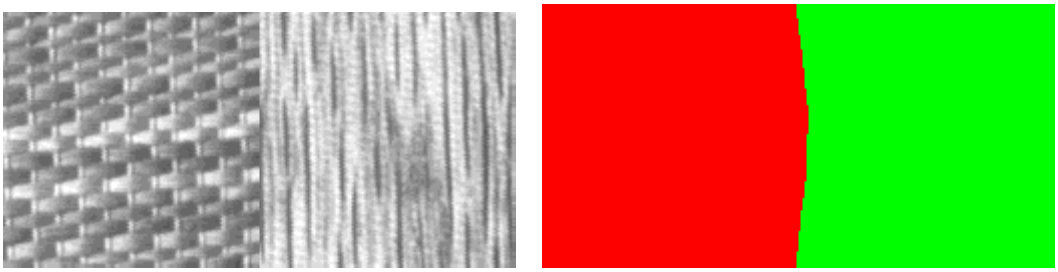
(a) (b)

FIGURE 10. (a) A natural texture pair constructed from the Brodatz album: D3 (reptile skin) and D68 (wood grain). (b) The two region segmentation obtained by our algorithm with moment mask size 9 and an averaging window of size 49.



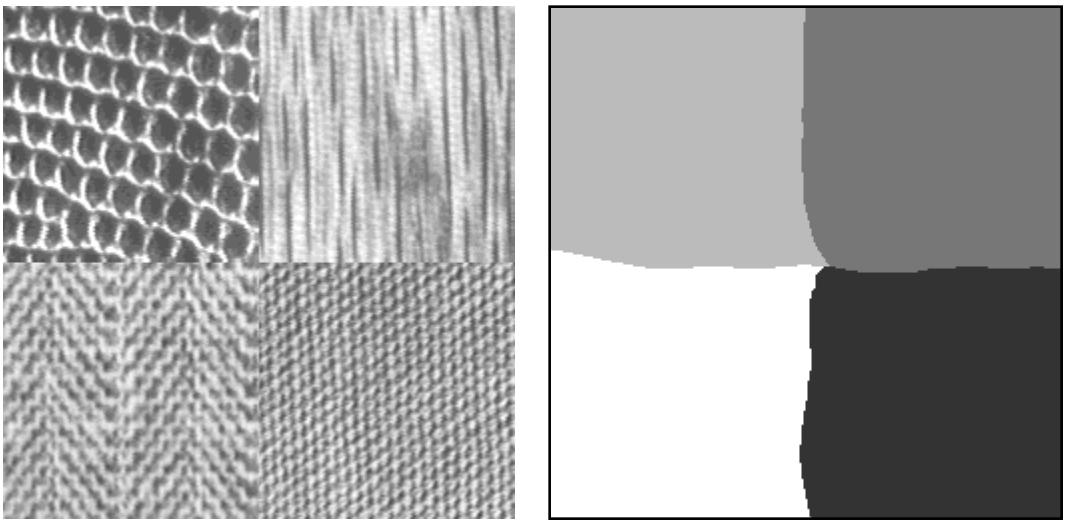
(a) (b)

FIGURE 11. (a) A natural texture pair constructed from the Brodatz album: D3 (reptile skin) and D17 (Herringbone weave). (b) The two region segmentation obtained by our algorithm with moment mask size 9 and an averaging window of size 49.



(a) (b)

FIGURE 12. (a) A natural texture pair constructed from the Brodatz album: D55 (straw matting) and D68 (wood grain). (b) The two region segmentation obtained by our algorithm with moment mask size 9 and an averaging window of size 49.



(a) (b)

FIGURE 13. (a) An image containing four textured regions. (b) The resulting segmentation computed by our algorithm. Each gray level represents a different textured region.

References

1. J. Beck, K. Prazdny, and A. Rosenfeld. A theory of textural segmentation. In Jacob Beck, Barbara Hope, and Azriel Rosenfeld, editors, *Human and Machine Vision*, pages 1–38. Academic Press, New York, 1983.

2. J. Beck, A. Sutter, and R. Ivry. Spatial frequency channels and perceptual grouping in texture segregation. *Computer Vision, Graphics, and Image Processing*, 37:299–325, 1987.

3. P. Brodatz. *Textures: a photographic album for artists and designers*. Dover Publications, Inc., New York, 1966.

4. T. Caelli and M. N. Oguztoreli. Some tasks and signal dependent rules for spatial vision. *Spatial Vision*, 2:295 – 315, 1987.

5. M. Clark and A. C. Bovik. Texture segmentation using Gabor modulation/demodulation. *Pattern Recognition Letters*, 6:261–267, Sept. 1987.

6. J. M. Coggins and A. K. Jain. A spatial filtering approach to texture analysis. *Pattern Recognition Letters*, 3:195–203, 1985.

7. G. C. Cross and A. K. Jain. Markov random field texture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-5:25–39, 1983.

8. L. S. Davis, M. Clearman, and J. K. Aggarwal. An empirical evaluation of generalized cooccurrence matrices. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-3:214–221, 1981.

9. S. A. Dudani, K. J. Breeding, and R. B. McGhee. Aircraft identification by moment invariants. *IEEE Transactions on Computers*, C-26(1):39–45, 1977.

10. A. Gagalowicz. Blind texture segmentation. In *Proc. Nineth International Conference on Pattern Recognition*, pages 46–50, Rome, Italy, November 1988.

11. R. M. Haralick. Statistical and structural approaches to texture. *Proceedings of the IEEE*, 67:786–804, 1979.

12. M. K. Hu. Visual pattern recognition by moment invariants. *IRE Trans. on Information Theory*, IT-8:179–187, 1962.

13. A. K. Jain, *Fundamentals of Digital Image Processing*, Prentice Hall, Englewood Cliffs, New Jersey, 1989.

14. A. K. Jain and R. C. Dubes. *Algorithms for Clustering Data*. Prentice Hall, Englewood Cliffs, New Jersey, 1988.

15. A. K. Jain and F. Farrokhnia. Unsupervised texture segmentation using Gabor filters. *Pattern Recognition*, 24: 1167–1186, 1991.

16. B. Julesz. Visual pattern discrimination. *IRE Trans. on Information Theory*, IT-8:84–92, 1962.

17. B. Julesz. Textons, the elements of texture perception, and their interactions. *Nature*, 290:91–97, 1981.

18. B. Julesz. Texton gradients: The texton theory revisited. *Biological Cybernetics*, 54:245–251, 1986.

19. B. Julesz, E. N. Gilbert, L. A. Shepp, and H. L. Frisch. Inability of humans to discriminate between visual textures that agree in second-order statistics - revisited. *Perception*, 2:391–405, 1973.

20. R. L. Kashyap and R. Chellappa. Estimation and choice of neighbors in spatial interaction models of images. *IEEE Trans. on Information Theory*, 29:60–72, 1983.

21. R. L. Kashyap, R. Chellappa, and N. Ahuja. Decision rules for the choice of neighbors in random field models of images. *Computer Graphics and Image Processing*, 15:301–318, 1981.

22. K. I. Laws, *Textured Image Segmentation*. Ph.D. Thesis, University of Southern California, 1980.

23. A. P. Pentland. Fractal-based description of natural scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6(6):661–674, November 1984.

24. M. Tuceryan and A. K. Jain. Texture segmentation using Voronoi polygons. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-12:211 – 216, February 1990.

25. M. Tuceryan and A. K. Jain, Texture Analysis, In *The Handbook of Pattern Recognition and Computer Vision*, L. F. Pau and P.S.P. Wang (eds.), World Scientific Publishing Co., 1993.

26. M. R. Turner. Texture discrimination by Gabor functions. *Biological Cybernetics*, 55:71–82, 1986.

27. H. Voorhees and T. Poggio. Detecting textons and texture boundaries in natural images. In *Proc. First International Conference on Computer Vision*, pages 250–258, London, 1987.

28. L. Zusne. Moments of area and of the perimeter of visual form as predictors of discrimination preformance. *Journal of Experimental Psychology*, 69:213 – 220, March 1965.