Texture Segmentation by Windowed Projection

[1, 2]Fan-Chen Tseng, [2]Ching-Chi Hsu, [2]Chiou-Shann Fuh
[1]Department of Electronic Engineering
National I-Lan Institute of Technology
e-mail : fctseng@ccmail.ilantech.edu.tw
[2]Department of Computer Science and Information Engineering,
National Taiwan University
e-mail: fuh@csie.ntu.edu.tw

Abstract

This paper investigates a novel approach to texture segmentation. Windowed projection is used to generate the feature vectors that capture the texture characteristics. The similarity between vectors is defined, and according to the similarity, pixel clustering is performed to segment the image into regions. The experimental results are given and the features of this approach are discussed.

Keywords: windowed projection, projection window, projection vector , texture segmentation

1. Introduction

Image segmentation is the important step toward image understanding and pattern recognition. After image segmentation, image objects can be extracted for further analysis. There are several approaches to image segmentation using information such as gray levels, edges and textures. While the former two approaches have inherent shortcomings [7], image segmentation based on textures, known as *texture segmentation* [5], provides another choice.

There is no official definition for the texture of an image. Still, textures can be associated with the attributes of fineness, coarseness, directionality, roughness, and regularity in an image [2].

Textures consist of two elements: gray levels (or colors) and their spatial distribution in a neighborhood. We can regard textures as repeating patterns of local variations in image [6]. It should be noted that the resolution at which an image is observed determines the scale at which the texture is perceived. Hence, textures should be discussed in a specified scale [1,2,5].

Texture segmentation is concerned with determining the boundaries between differently textured regions in an image. That is, the image is segmented into regions, each of which is differently textured. Each region should have a homogeneous texture, while each pair of adjacent regions are differently textured [2]. Usually, texture segmentation starts with statistical measures within a window around a pixel that represent the texture in the neighborhood of the pixel. Texture segmentation is then done by thresholding the image into regions according to the statistical measures [8]. However, this approach does not utilize the spatial patterns of the texture. Gray level co-occurrence matrix [2,8] tries to convey information about the variation patterns of gray levels over a neighborhood, but the amount of computation makes it impractical. Multi-channel filtering is another approach [5,9]. For all its good performance on artificially textured images, the approach remains computationally expensive [1].

In this paper, we propose a novel approach to texture segmentation. First, *windowed projection* is taken to generate the feature vector that captures the texture characteristics in the neighborhood of a pixel. Then, the similarity between vectors is defined. Moreover, according to the similarity, a region growing process is performed to segment the image into regions. Finally, border detection operator is used to mark the borders between differently textured regions. These segmentation steps are depicted in Figure 1. Most of the computations are integer arithmetic operations, thus our approach provides an efficient and effective paradigm for texture segmentation.

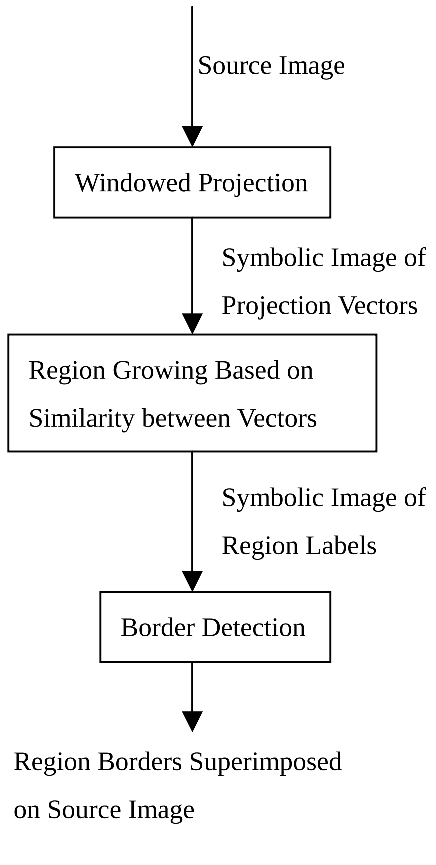
Source Image
Windowed Projection
Symbolic Image of Projection Vectors
Region Growing Based on Similarity between Vectors
Symbolic Image of Region Labels
Border Detection
Region Borders Superimposed on Source Image

Figure 1. Steps of texture segmentation.

2. Windowed projection and vector similarity

Windowed projection takes projections of the pixel values within a window. For each pixel, a square window centered about that pixel is used to limit the range of projection. In our system, a 9-by-9 window is chosen as the *projection window*. Horizontal projections along rows and vertical projections along columns are taken. These projections constitute the components of the *projection vector*. Since adjacent rows or columns are usually similar, the resulting projection vectors will contain adjacent components that are almost equal if each row (or column) is used for projection. To eliminate this redundancy of information, every three rows (or columns) are used as a projection unit to produce one component of the projection vector.

Therefore, there are three components from horizontal projections (denoted as *h1*, *h2* and *h3*), and three components from vertical projections (denoted as *v1*, *v2* and *v3*). Each component is normalized to be integers within the interval [0, 255] by the assumption that the gray levels range from 0 to 255. Figure 2 shows the details.

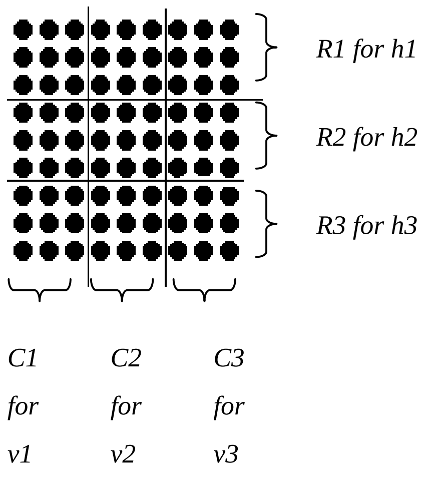

Figure 2. Windowed projection.

In formula, the projection vector ***V*** for a pixel *p* at *x*-column *y*-row, whose gray level denoted as *p(x,y)*, is:

V *(x,y)=(h1,h2,h3,v1,v2,v3)*, where, for *i*=1,2,3

$$hi = \left(\sum_{(c,r)\in Ri} q(c,r) \right) / |Ri|$$
$$vi = \left(\sum_{(c,r)\in Ci} q(c,r) \right) / |Ci|$$

,where *q(c,r)* is the gray level of the pixel at (c,r);R_1 is the top three rows within the projection window; R_2 is the middle three rows within the projection window; R_3 is the bottom three rows within the projection window; C_1 is the left three columns within the projection window; C_2 is the middle three columns within the projection window; C_3 is the right three columns within the projection window.

There are several ways to measure the similarity between vectors. Inner product of normalized vectors is quite popular [4]. However, it only cares about the angle between vectors and ignores the vector length. In images, the vector length is important since it represents the gray level of textures. Euclidean distance is also popular [7], but it needs a lot of computations. Remember that the projection vectors are only "feature vectors" for textures and not actual vectors in space. Hence, any reasonable metric [3,10] can be defined to measure the distance between projection vectors. In our system, we define two distances: *horizontal projection distance* (denoted as *DH*) and *vertical projection distance* (denoted as *DV*). From these two distances, we define two similarities: *horizontal similarity, simH*, and *vertical similarity, simV*. The similarity between two projection vectors, *sim(**A**,**B**)*, is defined to be the product of *simH* and *simV*.

Formally, let ***A**=(ah1,ah2,ah3,av1,av2,av3)* and ***B**=(bh1,bh2,bh3,bv1,bv2,bv3)* be the projection vectors for pixel a and pixel b, respectively. We have:

$DH = |ah1-bh1| + |ah2-bh2| + |ah3-bh3|$

$DV = |av1-bv1| + |av2-bv2| + |av3-bv3|$

The range of *DH* and *DV* is from 0 to 765 (255x3=765). Hence we define *simH* and *simV* as follows:

$SimH = 1 - DH / 765$

$SimV = 1 - DV / 765$

The similarity between projection vectors ***A*** and ***B*** is then: *Sim(**A**,**B**)= SimH* SimV* .

From the above formulas, it is clear that smaller distances result in higher similarities, and larger distances result in lower similarities.

3. Region growing and border detection

The essence of texture segmentation is a clustering problem of how to partition an image into regions of homogeneous texture features. However, the traditional clustering techniques [4] are not necessary for image segmentation. This saving of computations is due to the *principle of spatial locality: adjacent pixels tend to have similar texture features except for the pixels on the borders of differently textured regions*. That is, we have to consider only the neighbors of each pixel for clustering instead of considering pixels that are far apart. Therefore, region growing [2] is appropriate for this application.

The region growing technique we use is a combination of *hybrid-linkage* and *centroid-linkage* [2]. The projection vector associated with each pixel is derived from the texture characteristic within a window around the pixel, and this is the principle of *hybrid-linkage*. The region growing is determined by the similarity between *the average projection vector of a region* (the *region centroid*) and the pixel's projection vector, and this is the principle of *centroid-linkage*.

The region growing process proceeds in a top-down, left-right fashion. For each pixel, the 8-connected neighboring pixels are examined. If there are neighbors that have been assigned to regions, the similarities between the pixel's projection vector and the centroids of these regions are evaluated. The pixel will be assigned to the region (denoted as ***R***) that maximizes the similarity if that similarity is above the *region-growing threshold*. Otherwise, a new region is created for this pixel. Besides, during region growing process, if there are more than one regions (denoted as a region set $\{ \Gamma_i \}$) whose similarities with the pixel exceed the region-growing threshold, the similarity between the centroids of these regions and the centroid of ***R*** is evaluated. The region Γ_i will be merged into ***R*** if its similarity with ***R*** is higher than the *region-merging threshold*. The region growing process stops when every pixel in the image is assigned to a region.

The border detection is a standard process [2]. The mark-interior/border-pixel operator is used in 4-connected mode to detect the border between textured regions.

4. Experimental results and discussions

4.1. Experimental setups

The system is implemented in Visual Basic running on Pentium II 200 MHz with 64 MB RAM. The execution time is counted from the time when the image is loaded into memory to the time when the resulting image is displayed on the screen. There are four images for test: the artificial two-textured image in Figure 3 (size 245 x 118 pixels), the clothes in Figure 4 (size 259 x 160 pixels), the four rectangular areas of two textures in Figure 5 (size 215x215 pixels), and the scene in Figure 6 (size 320x200 pixels).

4.2. Results

For Figure 3, the execution time is 5768 ms, the region-growing threshold is 0.6, and the region-merging threshold is 0.66. For Figure 4, the execution time is 8241 ms, the region-growing threshold is 0.6, and the region-merging threshold is 0.84. For Figure 5, the execution time is 9339 ms, the region-growing threshold is 0.6, and the region-merging threshold is 0.72. For Figure 6, the execution time is 12896 ms, the region-growing threshold is 0.6, and the region-merging threshold is 0.96.

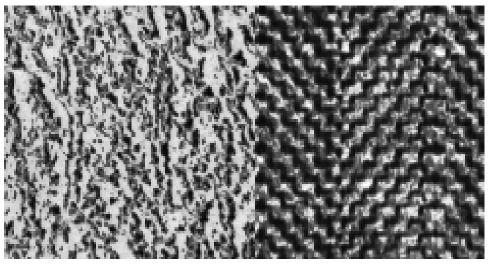
Figure 3. (a) Source image.

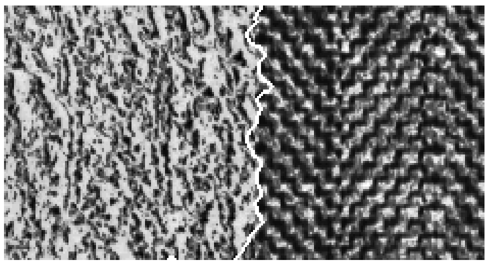
Figure 3. (b) Resulting image.

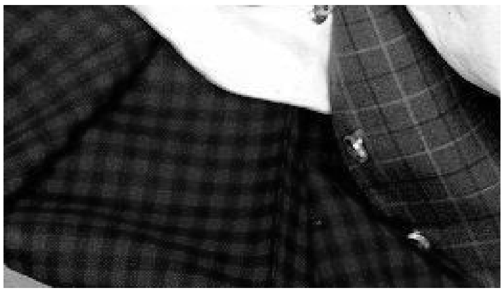
Figure 4. (a) Source image.

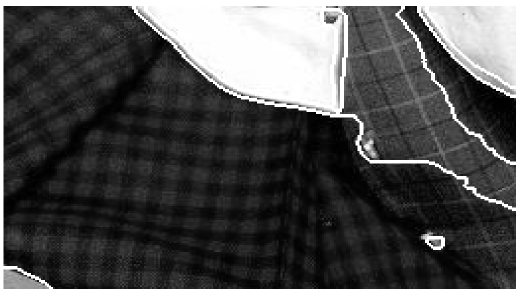
Figure 4. (b) Resulting image.

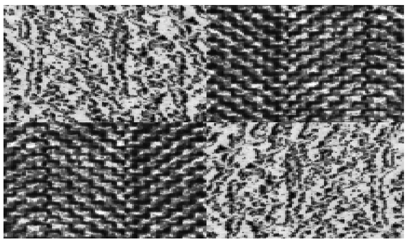

Figure 5. (a) Source Image.

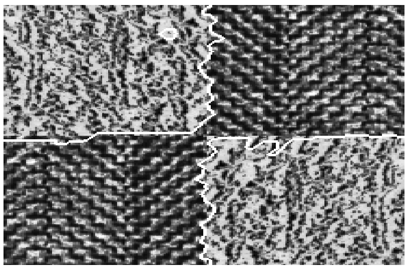

Figure 5. (b) Resulting Image.

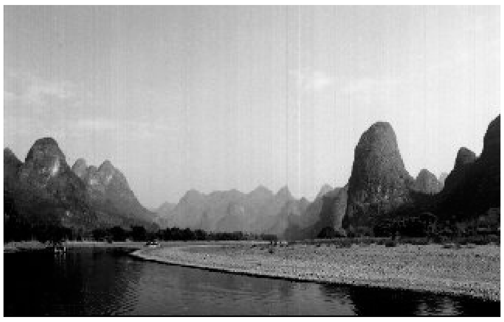

Figure 6. (a) Source Image.

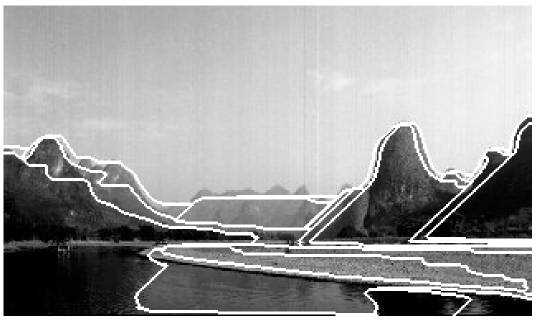

Figure 6. (b) Resulting Image.

4.3. Discussion

For Figure 3, the region boundary is actually a straight line. However, even human eyes may perceive it as a curved boundary. The result is acceptable. For Figure 4, although the textured regions undergo deformations due to the folding of clothes, our system can still recognize the textured regions. In Figure 5, there are four rectangular regions of two textures. However, there exists a “false bridge” between the two darker regions. Unfortunately, the width of this “bridge” happens to be about nine-pixel wide, the dimension of our projection window. Therefore, our system takes the two darker regions as one region, and merges them together. For Figure 6, the sky is lowly textured; that is, homogeneous in texture. Hence, it is regarded as a single region despite its large area.

As is true of other window-based operations, in our experiments, the scale of textures that are detected depends on the size of projection window. Larger windows tend to under-segment the image, while smaller windows tend to over-segment the image.

The basic assumption of windowed projection is that the gray levels are uniformly distributed over the range [0, 255]. Therefore, it performs well for image with good contrast. However, for poorly contrasted images whose histogram concentrates within a narrow range, the segmentation results are not satisfactory. To cure this problem, the images may be histogram equalized before segmentation.

Also, the results of texture segmentation are affected by the policy of region growing. Especially, the thresholds for region growing and for region merging will affect the number and the shape of textured regions.

4.3. Comparison with other methods

For comparison with conventional filtering approaches to texture segmentation, we use two filtering methods: Laws energy filter [11] and Discrete Cosine Transform (DCT) filter [12] to segment the same set of test images. These two filtering approaches are chosen because, just as our *Windowed Projection* does, they use filter masks of fixed sizes: 5 by 5 for Laws energy filter and 3 by 3 for DCT filter. The results of Laws energy filter and DCT filter are shown in Figure 7 and Figure 8, respectively.

For Figure 7(a), the region-growing threshold is 0.8, and the region-merging threshold is 0.96. For Figures 7(b) and 7(c), the region-growing threshold is 0.8, and the region-merging threshold is 1.2, which is greater than one to inhibit region merging. For Figure 7(d), the region-growing threshold is 0.8, and the region-merging threshold is 0.96.

For Figures 8(a) and 8(d), the region-growing threshold is 0.6. For Figures 8(b) and 8(c), the region-growing threshold is 0.5. There is no region merging because the results will be worse.

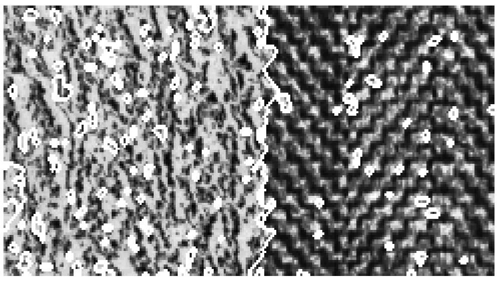

Figure 7(a). Execution time: 11536 ms

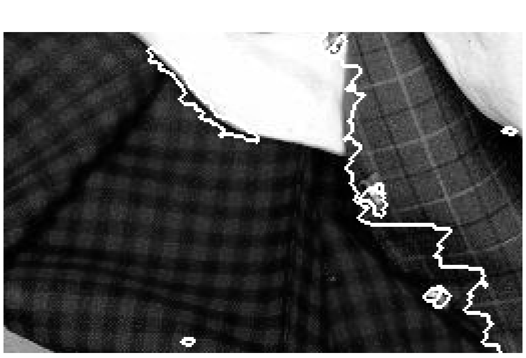

Figure 7(b). Execution time: 16301 ms

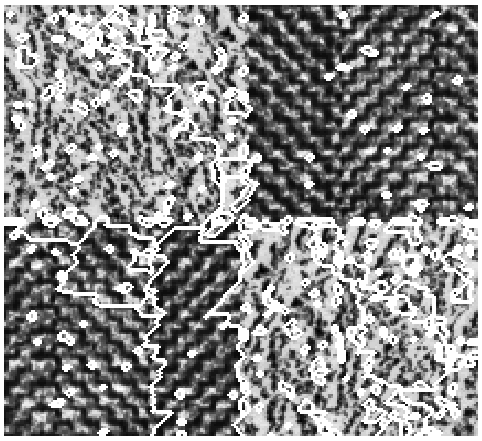

Figure 7(c). Execution time: 18732 ms

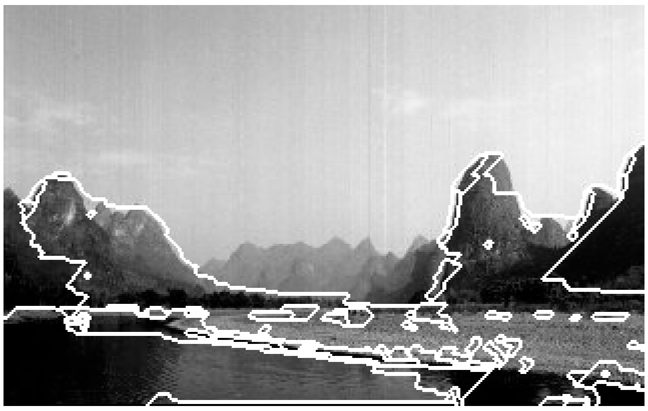

Figure 7(d). Execution time: 26506 ms

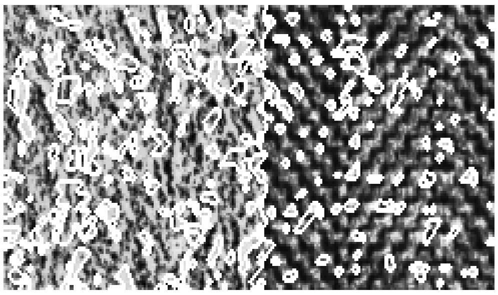

Figure 8(a). Execution time:6592 ms

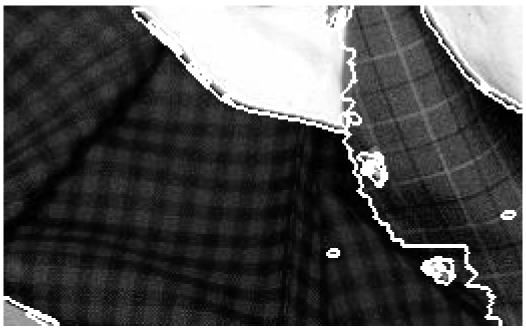

Figure 8(b). Execution time: 9160 ms

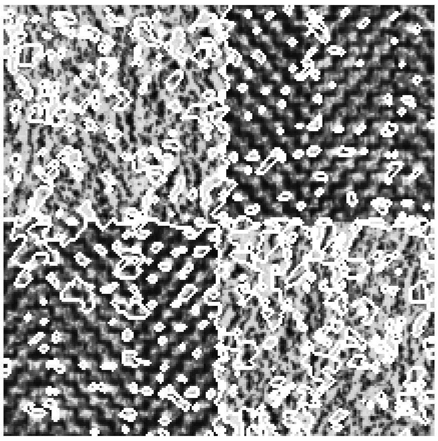

Figure 8(c). Execution time: 11041 ms

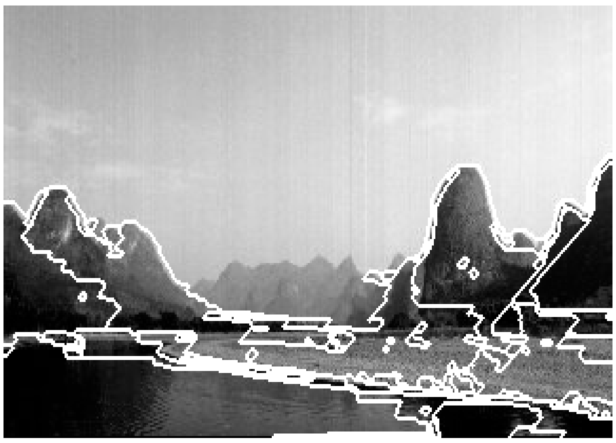

Figure 8(d). Execution time: 15225 ms

5. Conclusions

In this paper, we proposed a simple operation, called *Windowed Projection*, which can effectively capture the texture characteristics of the image. The similarity between *projection vectors* is also defined as a basis for region growing to segment the image into textured regions. As is demonstrated by the experimental results, *windowed projection* is able to reflect the spatial distribution of gray levels in an image. To improve our system, various strategies for windowed projection and region growing should be investigated. The relation between filtering approach and windowed projection is also worth studying.

Reference

[1] B.Y. Chitre and A. P. Dhawan, "M-band wavelet discrimination of natural textures, " *Pattern Recognition*, Vol. 32, pp. 773-789, 1999.

[2] R. M. Haralick and L. G. Shapiro, *Computer and Robot Vision*, Vol. I, Addison-Wesley, Reading, MA, 1992.

[3] R. M. Haralick and L. G. Shapiro, *Computer and Robot Vision,* Vol. II, Addison-Wesley, Reading, MA, 1993.

[4] C. C. Hsu, F. C. Tseng, J. M. Chen, and C. H. Chang, "Constructing personal digital library by multi-search and customized category," *Proceedings of Tenth International Conference on Tools with Artificial Intelligence,* Taipei, Taiwan, pp. 148-157,1998.

[5] A. K. Jain and F. Farrokhnia, "Unsupervised texture segmentation using gabor filters, " *Pattern Recognition,* Vol. 24, pp. 1167-1186, 1991.

[6] R. Jain, R. Kasturi, and B. G. Schunck, *Machine Vision*, McGraw-Hill, New York, 1995

[7] A. Moghaddamzadeh and N. Bourbakis, "A fuzzy region growing approach for segmentation of color images, " *Pattern Recognition*, vol. 30, pp. 867-881, 1997.

[8] J. K. Parker, *Algorithms for Image Processing and Computer Vision*, John Wiley & Sons, New York, 1997.

[9] T. Randen and J. H. Husoy, "Filtering for texture classification: a comparative study", *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 21, pp. 291-310, 1999.

[10] J. Setubal and J. Meidanis, *Introduction to Computational Molecular Biology*, PWS Publishing, Boston, 1997.

[11] K.I. Laws, “Rapid Texture Identification,” *Proc. SPIE Conf. Image Processing for Missile Guidance*, pp. 376-380, 1980.

[12]I. Ng, T. Tan, and J. Kitter, “On Local Linear Transform and Gabor Filter Representation of Texture,” *Proc. Int’l Conf. Pattern Recognition*, pp. 627-631, 1992.