Chapter 7

Texture

Texture is another feature that can help to segment images into regions of interest and to classify those regions. In some images, it can be the defining characteristic of regions and critical in obtaining a correct analysis. The image of Figure 7.1 has three very distinct textures: the texture of the tiger, the texture of the jungle, and the texture of the water. These textures can be quantified and used to identify the object classes they represent.

Figure 7.1: An image containing several different regions, each having a distinct texture (licensed from Corel Stock Photos).

Texture gives us information about the *spatial arrangement* of the colors or intensities in an image. Suppose that the histogram of a region tells us that it has 50% white pixels and 50% black pixels. Figure 7.2 shows three different images of regions with this intensity distribution that would be considered three different textures. The leftmost image has two big blocks: one white and one black. The center image has 18 small white blocks and 18

small black blocks forming a checkerboard pattern. The rightmost image has six long blocks, three white and three black, in a striped pattern.

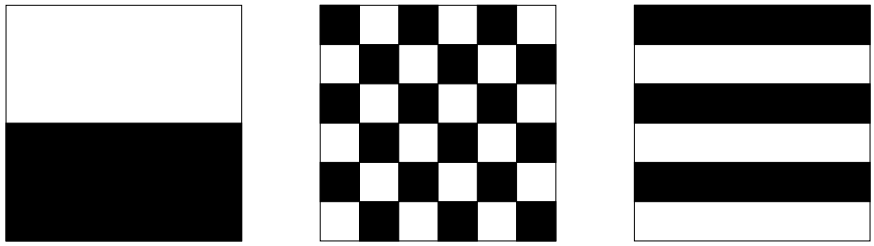


Figure 7.2: Three different textures with the same distribution of black and white.

The images of Figure 7.2 were artificially created and contain geometric patterns constructed from black and white rectangles. Texture is commonly found in natural scenes, particularly in outdoor scenes containing both natural and man-made objects. Sand, stones, grass, leaves, bricks, and many more objects create a textured appearance in images. Figure 7.3 illustrates some of these natural textures. Note that the two different brick textures and two different leaf textures shown are quite different. Thus, textures must be described by more than just their object classifications. This chapter discusses what texture is, how it can be represented and computed, and what it can be used for in image analysis.

7.1 Texture, Texels, and Statistics

The artificial textures of Figure 7.2 are made up of primitive rectangular regions in white or black. In the checkerboard, the regions are small squares arranged in a 2D grid of alternating colors. In the striped pattern, the regions are long stripes arranged in a vertical sequence of alternating colors. It is easy to segment out these single-color regions and to recognize these simple patterns.

Now, consider the two leaf textures of Figure 7.3. The first has a large number of small, round leaves, while the second has a smaller number of larger, pointed leaves. It is difficult to describe the spatial arrangements of the leaves in words; the arrangements are not regular, but there is some quality of the image that would make one argue that there *is* a noticeable arrangement in each image.

Part of the problem in texture analysis is defining exactly what texture is. There are two main approaches:

1. **structural approach:** Texture is a set of primitive *texels* in some regular or repeated relationship.

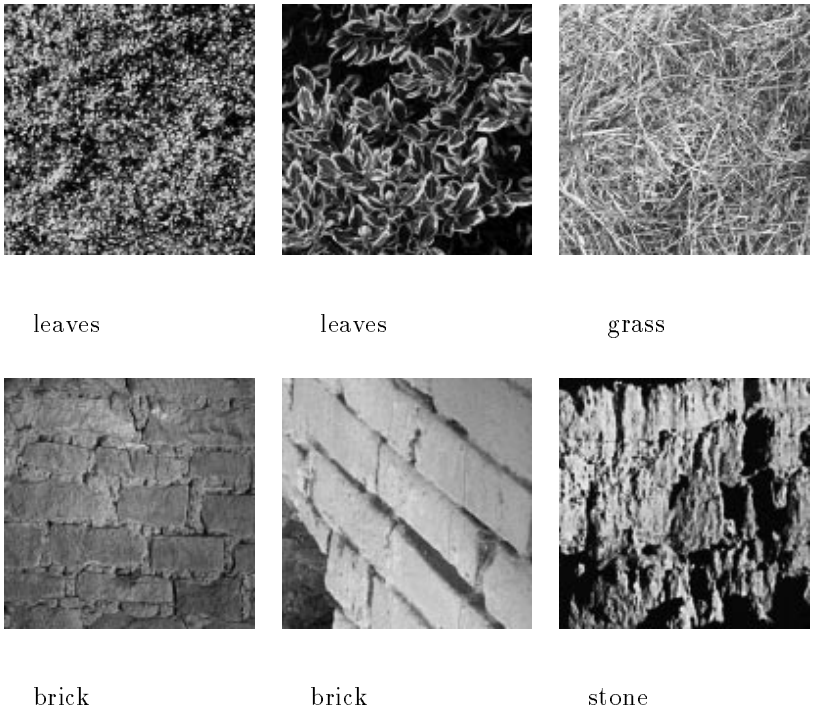


Figure 7.3: Natural textures (from the MIT Media Lab VisTex database: http://vismod.www.media.mit.edu/vismod/imagery/VisionTexture/vistex.html).

2. **statistical approach:** Texture is a quantitative measure of the arrangement of intensities in a region.

While the first approach is appealing and can work well for man-made, regular patterns, the second approach is more general and easier to compute and is used more often in practice.

7.2 Texel-Based Texture Descriptions

A texture can be thought of as a set of primitive texels in a particular spatial relationship. A structural description of a texture would then include a description of the texels and a specification of the spatial relationship. Of course, the texels must be segmentable and the relationship must be efficiently computable. One very nice geometry-based description was proposed by Tuceryan and Jain. The texels are image regions that can be extracted through some simple procedure such as thresholding. The characterization of their spatial relationships is obtained from a Voronoi tesselation of the texels as explained below.

Suppose that we have a set of already-extracted texels, and that we can represent each one by a meaningful point, such as its centroid. Let S be the set of these points. For any pair of points P and Q in S, we can construct the perpendicular bisector of the line joining them. This perpendicular bisector divides the plane into two half planes, one of which is the

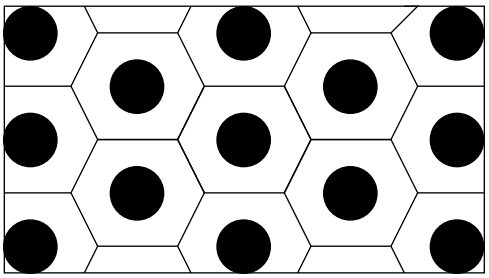


Figure 7.4: The Voronoi tesselation of a set of circular texels.

set of points that are closer to P and the other of which is the set of points that are closer to Q. Let $H^Q(P)$ be the half plane that is closer to P with respect to the perpendicular bisector of P and Q. We can repeat this process for each point Q in S. The *Voronoi polygon* of P is the polygonal region consisting of all points that are closer to P than to any other point of S and is defined by

$$V(P) = \bigcap_{Q \in S, Q \neq P} H^Q(P)$$

Figure 7.4 illustrates the Voronoi polygons for a set of circular texels. This pattern produces hexagonal polygons for internal texels; texels that border the image boundary have various other shapes.

Once the texels have been extracted from an image and their Voronoi tesselation computed, shape features of the polygons are calculated and used to group the polygons into clusters that define uniformly-textured regions. The type of pattern shown in Figure 7.4 extended to a large image would produce a single region of uniform texture characterized by the shape features of the regular hexagons.

Exercise 1 Texel-Based Descriptions

Find or create a set of 5 images showing textures that have obvious texels that can be detected via a simple procedure such as thresholding based on gray-tone or color ranges. Try to find at least one texture that has more than one kind of texel. Draw the Voronoi tesselation for a small area on this image.

7.3 Quantitative Texture Measures

For the most part, segmenting out the texels is more difficult in real images than in artificially generated patterns. Instead, numeric quantities or statistics that describe a texture can be computed from the gray tones (or colors) themselves. This approach is less intuitive, but is computationally efficient and can work well for both segmentation and classification of textures.

7.3.1 Edge Density and Direction

Since edge detection is a well-known and simple-to-apply feature detection scheme, it is natural to try to use an edge detector as the first step in texture analysis. The number of edge pixels in a given fixed-size region gives some indication of the busyness of that region. The directions of these edges, which are usually available as a biproduct of the edge-detection process, can also be useful in characterizing the texture pattern.

Consider a region of N pixels. Suppose that a gradient-based edge detector is applied to this region producing two outputs for each pixel p: 1) the gradient magnitude $Mag(p)$ and 2) the gradient direction $Dir(p)$, as defined in Chapter 5. One very simple texture feature is *edgeness per unit area* which is defined by

$$F_{edgeness} = \frac{|\{p \mid Mag(p) \geq T\}|}{N} \tag{7.1}$$

for some threshold T. Edgeness per unit area measures the busyness, but not the orientation of the texture.

This measure can be extended to include both busyness and orientation by employing histograms for both the gradient magnitude and the gradient direction. Let $H_{mag}(R)$ denote the normalized histogram of gradient magnitudes of region R, and let H_{dir} denote the normalized histogram of gradient orientations of region R. Both of these histograms are over a small, fixed number (such as 10) of bins representing groups of magnitudes and groups of orientations. Both are normalized according to the size N_R of region R. Then

$$F_{magdir} = (H_{mag}(R), H_{dir}(R)) \tag{7.2}$$

is a quantitative texture description of region R.

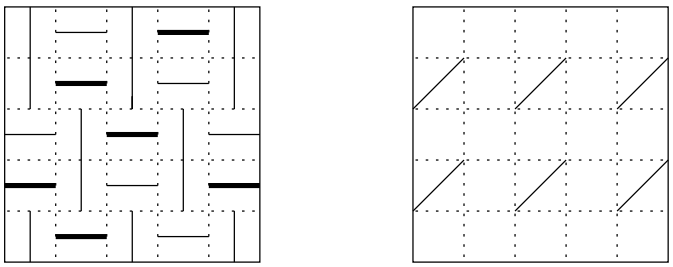


Figure 7.5: Two images with different edgeness and edge-direction statistics.

Consider the two 5×5 images shown in Figure 7.5. The image on the left is busier than the image on the right. It has an edge in every one of its 25 pixels, so its edgeness per unit area is 1.0. The image on the right has 6 edges out of its 25 pixels, so its edgeness per unit area is only 0.24. For the gradient-magnitude histograms, we will assume there are two bins representing dark edges and light edges. For the gradient-direction histograms, we will use three bins for horizontal, vertical, and diagonal edges. The image on the left has 6

dark edges and 19 light edges, so its normalized gradient-magnitude histogram is (0.24,0.76), meaning that 24% of the edges are dark and 76% are light. It also has 12 horizontal edges, 13 vertical edges, and no diagnonal edges, so its normalized gradient-direction histogram is (0.48,0.52,0.0), meaning that 48% of the edges are horizontal, 52% are vertical and 0% are diagonal. The image on the right has no dark edges and 6 light edges, so its normalized gradient-magnitude histogram is (0.0,0.24). It also has no horizontal edges, no vertical edges, and 6 diagonal edges, so its normalized gradient-direction histogram is (0.0,0.0,0.24). In the case of these two images, the edgeness-per-unit-area measure is sufficient to distinguish them, but the histogram measure provides a more powerful descriptive mechanism in general. Two n-bin histograms H_1 and H_2 can be compared by computing their L_1 *distance*

$$L_1(H_1, H_2) = \sum_{i=1}^{n} \mid H_1[i] - H_2[i] \mid \tag{7.3}$$

Exercise 2 Edge-Based Texture Measures

Obtain a set of images that have lots of man-made structures with clearly defined edges. Write a program to compute the texture measure F_{magdir} of equation 7.2 for each of these images, and compare them using the L_1 distance of equation 7.3.

7.3.2 Local Binary Partition

Another very simple, but useful texture measure is the local binary partition measure. For each pixel p in the image, the eight neighbors are examined to see if their intensity is greater than that of p. The results from the eight neighbors are used to construct an eight-digit binary number $b_1b_2b_3b_4b_5b_6b_7b_8$ where $b_i = 0$ if the intensity of the ith neighbor is less than or equal to that of p and 1 otherwise. A histogram of these numbers is used to represent the texture of the image. Two images or regions are compared by computing the L_1 distance between their histograms as defined above.

Exercise 3 LPB

Using the images from the previous exercise, write another program to compute the histogram representing the LBP texture measure of each image. Again compute the L_1 distances between pairs of images using this measure. Compare to your previous results.

7.3.3 Co-occurrence Matrices and Features

A *co-occurrence* matrix is a two-dimensional array C in which both the rows and the columns represent a set of possible image values V. For example, for gray-tone images V can be the set of possible gray tones and for color images V can be the set of possible colors. The value of $C(i, j)$ indicates how many times value i co-occurs with value j in some designated spatial relationship. For example, the spatial relationship might be that value i occurs immediately to the right of value j. To be more precise, we will look specifically at the case where the set V is a set of gray tones and the spatial relationship is given by a vector d that specifies the displacement between the pixel having value i and the pixel having value j.

Let d be a displacement vector (dr, dc) where dr is a displacement in rows (downward) and dc is a displacement in columns (to the right). Let V be a set of gray tones. The gray-tone co-occurrence matrix C_d for image I is defined by

$$C_d(i,j) = \mid \{(r,c) \mid I(r,c) = i \; and \; I(r+dr, c+dc) = j\} \mid \tag{7.4}$$

Figure 7.6 illustrates this concept with a 4×4 image I and three different co-occurrence matrices for I: $C_{(0,1)}$, $C_{(1,0)}$, and $C_{(1,1)}$.

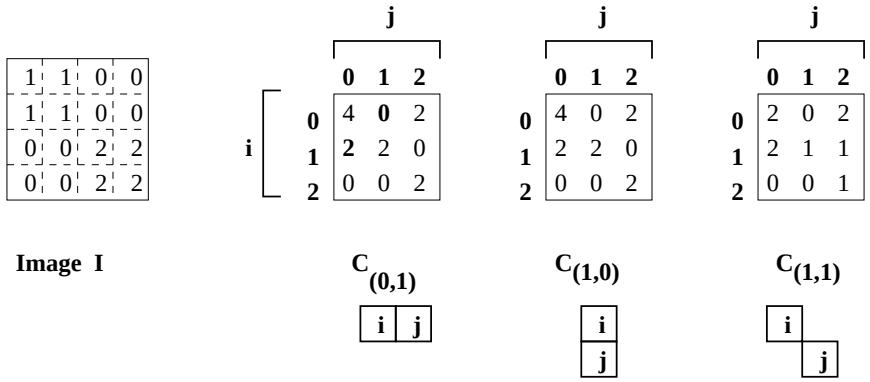


Figure 7.6: Three different co-occurrence matrices for a gray-tone image.

In $C_{(0,1)}$ note that position (1,0) has a value of 2, indicating that $j = 0$ appears directly to the right of $i = 1$ two times in the image. However, position (0,1) has a value of 0, indicating that $j = 1$ never appears directly to the right of $i = 0$ in the image. The largest co-occurence value of 4 is in position (0,0), indicating that a 0 appears directly to the right of another 0 four times in the image.

Exercise 4 Co-occurence Matrices

Construct the gray-tone co-occurrence matrices $C_{(1,2)}$, $C_{(2,2)}$, and $C_{(2,3)}$ for the image of Figure 7.6.

There are two important variations of the standard gray-tone co-occurrence matrix. The first is the *normalized* gray-tone co-occurrence matrix N_d defined by

$$N_d(i,j) = \frac{C_d(i,j)}{\sum_i \sum_j C_d(i,j)} \tag{7.5}$$

which normalizes the co-occurrence values to lie between zero and one and allows them to be thought of as probabilities in a large matrix. The second is the *symmetric* gray-tone co-occurence matrix $S_d(i,j)$ defined by

$$S_d(i,j) = C_d(i,j) + C_{-d}(i,j) \tag{7.6}$$

which groups pairs of symmetric adjacencies.

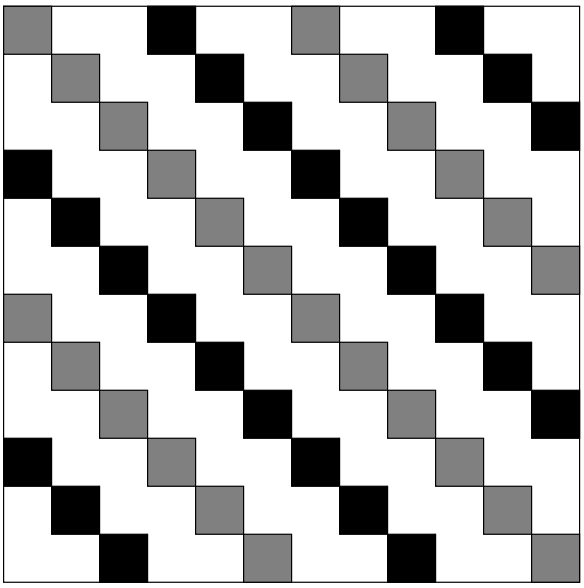


Figure 7.7: An image with a diagonal texture pattern.

Exercise 5 Normalized Co-occurrence

Compute the normalized co-occurrence matrix $N_{(1,1)}$ for the image of Figure 7.7 assuming that the black pixels have gray tone 0, the gray pixels have gray tone 1, and the white pixels have gray tone 2. How does it represent the texture pattern of the image?

Co-occurrence matrices capture properties of a texture, but they are not directly useful for further analysis, such as comparing two textures. Instead, numeric features are computed from the co-occurrence matrix that can be used to represent the texture more compactly. The following are standard features derivable from a normalized co-occurence matrix.

$$Energy = \sum_i \sum_j N_d^2(i,j) \tag{7.7}$$

$$Entropy = -\sum_i \sum_j N_d(i,j) log_2 N_d(i,j) \tag{7.8}$$

$$Contrast = \sum_i \sum_j (i-j)^2 N_d(i,j) \tag{7.9}$$

$$Homogeneity = \sum_i \sum_j \frac{N_d(i,j)}{1+\mid i-j \mid} \tag{7.10}$$

$$Correlation = \frac{\sum_i \sum_j (i-\mu_i)(j-\mu_j) N_d(i,j)}{\sigma_i \sigma_j} \tag{7.11}$$

where μ_i, μ_j are the means and σ_i, σ_j are the standard deviations of the row and column

sums $N_d(i)$ and $N_d(j)$ defined by

$$N_d(i) = \sum_j N_d(i,j)$$
$$N_d(j) = \sum_i N_d(i,j)$$

One problem with deriving texture measures from co-occurrence matrices is how to choose the displacement vector d. A solution suggested by Zucker and Terzopoulos is to use a χ^2 statistical test to select the value(s) of d that have the most structure; that is, to maximize the value:

$$\chi^2(d) = (\sum_i \sum_j \frac{N_d^2(i,j)}{N_d(i)N_d(j)} - 1)$$

7.3.4 Laws' Texture Energy Measures

Another approach to generating texture features is to use local masks to detect various types of texture. Laws developed a texture-energy approach that measures the amount of variation within a fixed-size window. A set of nine 5 x 5 convolution masks is used to compute texture energy, which is then represented by a vector of nine numbers for each pixel of the image being analyzed. The masks are computed from the following vectors, which are similar to those studied in Chapter 5.

$$\begin{array}{llll} \text{L5} & \text{(Level)} & = & [\ \ 1\ \ 4\ \ 6\ \ 4\ \ 1\ ] \\ \text{E5} & \text{(Edge)} & = & [\ -1\ -2\ \ 0\ \ 2\ \ 1\ ] \\ \text{S5} & \text{(Spot)} & = & [\ -1\ \ 0\ \ 2\ \ 0\ -1\ ] \\ \text{R5} & \text{(Ripple)} & = & [\ \ 1\ -4\ \ 6\ -4\ \ 1\ ] \end{array}$$

The names of the vectors describe their purposes. The L5 vector gives a center-weighted local average. The E5 vector detects edges, the S5 vector detects spots, and the R5 vector detects ripples. The 2D convolution masks are obtained by computing outer products of pairs of vectors. For example, the mask E5L5 is computed as the product of E5 and L5 as follows.

$$\begin{bmatrix} -1 \\ -2 \\ 0 \\ 2 \\ 1 \end{bmatrix} \times \begin{bmatrix} 1 & 4 & 6 & 4 & 1 \end{bmatrix} = \begin{bmatrix} -1 & -4 & -6 & -4 & -1 \\ -2 & -8 & -12 & -8 & -1 \\ 0 & 0 & 0 & 0 & 0 \\ 2 & 8 & 12 & 8 & 2 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$$

The first step in Laws' procedure is to remove effects of illumination by moving a small window around the image, and subtracting the local average from each pixel, to produce a preprocessed image, in which the average intensity of each neighborhood is near to zero. The size of the window depends on the class of imagery; a 15 X 15 window was used for natural scenes. After the preprocessing, each of the sixteen 5 X 5 masks are applied to the preprocessed image, producing sixteen filtered images. Let $F_k[i,j]$ be the result of filtering

Table 7.1: Laws texture energy measures for the images of Figure 7.3.

Image	**E5E5**	**S5S5**	**R5R5**	**E5L5**	**S5L5**	**R5L5**	**S5E5**	**R5E5**	**R5S5**
Leaves1	250.9	140.0	1309.2	703.6	512.2	1516.2	187.5	568.8	430.0
Leaves2	257.7	121.4	988.7	820.6	510.1	1186.4	172.9	439.6	328.0
Grass	197.8	107.2	1076.9	586.9	410.5	1208.5	144.0	444.8	338.1
Brick1	128.1	60.2	512.7	442.1	273.8	724.8	86.6	248.1	176.3
Brick2	72.4	28.6	214.2	263.6	130.9	271.5	43.2	93.3	68.5
Stone	224.6	103.2	766.8	812.8	506.4	1311.0	150.4	413.5	281.1

with the kth mask at pixel $[i, j]$. Then the *texture energy map* E_k for filter k is defined by

$$E_k[r, c] = \sum_{j=c-7}^{c+7} \sum_{i=r-7}^{r+7} \mid F_k[i, j] \mid \tag{7.12}$$

Each texture energy map is a full image, representing the application of the kth mask to the input image.

Once the sixteen energy maps are produced, certain symmetric pairs are combined to produce the nine final maps, replacing each pair with its average. For example, E5L5 measures horizontal edge content, and L5E5 measures vertical edge content. The average of these two maps measures total edge content. The nine resultant energy maps are

L5E5/E5L5	L5S5/S5L5
L5R5/R5L5	E5E5
E5S5/S5E5	E5R5/R5E5
S5S5	S5R5/R5S5
R5R5	

The result of all the processing gives nine energy map images or, conceptually, a single image with a vector of nine texture attributes at each pixel. Table 7.1 shows the nine texture attributes for the main texture of each of the grass/stones/brick images of Figure 7.3. These texture attributes can be used to cluster an image into regions of uniform texture. Figure 7.8 illustrates the segmentation of several multi-texture images into clusters.

7.3.5 Autocorrelation and Power Spectrum

The autocorrelation function of an image can be used to detect repetitive patterns of texture elements and also describes the fineness/coarseness of the texture. The autocorrelation function $\rho(dr, dc)$ of an $N + 1 \times N + 1$ image for displacement $d = (dr, dc)$ is given by

$$\rho(dr, dc) = \frac{\sum_{r=0}^{N} \sum_{c=0}^{N} I[r, c] I(r+dr, c+dc]}{\sum_{r=0}^{N} \sum_{c=0}^{N} I^2[r, c]} \tag{7.13}$$

$$= \frac{I[r, c] \circ I_d[r, c]}{I[r, c] \circ I[r, c]} \tag{7.14}$$

using the ideas of Chapter 5.

Exercise 6 Laws Texture Energy Measures

Write a program to compute the Laws texture energy measures that inputs a gray-scale image and outputs a set of nine images, one for each of the texture energy measures. Obtain a set of images of both man-made and natural textures and perform a sequence of tests on them. For each test, make one of the images the *test image* and call the others the *database images*. Write an interactive front end that allows a user to select a pixel of the test image and then looks for those database images that have a texture similar to that of the selected pixel anywhere in that database image using the L_1 distance on the set of nine texture energy measures available for each pixel. The brute force way to do this is to merely compare the nine values for the test image pixel with the nine values for each pixel of each database image and select an image as soon as any of its pixels has similar enough texture energy measure values. How might you do this more efficiently?

If the texture is coarse, then the autocorrelation function drops off slowly; otherwise, it will drop off very rapidly. For regular textures, the autocorrelation function will have peaks and valleys. Since $I[r + dr, c + dc]$ is undefined at the boundaries of the image, a method for computing these virtual image values must be defined.

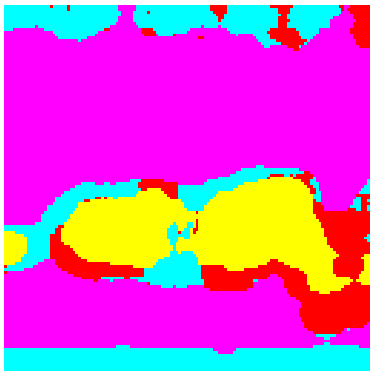
The autocorrelation function is related to the power spectrum of the Fourier transform. If $I(r, c)$ is the image function and $F(u, v)$ is its Fourier transform, the quantity $| F(u, v) |^2$ is defined as the power spectrum where $| \cdot |$ is the modulus of a complex number. The frequency domain can be divided into n_r regions bounded by circular rings (for frequency content) and n_d wedges (for orientation content) and the total energy in each region is computed to produce a set of texture features, as introduced in Chapter 5.

7.4 Texture Segmentation

Any texture measure that provides a value or vector of values at each pixel, describing the texture in a neighborhood of that pixel, can be used to segment the image into regions of similar textures. Like any other segmentation algorithm, texture segmentation algorithms are of two major types: region-based approaches and boundary-based approaches. Region-based approaches attempt to group or cluster pixels with similar texture properties. Boundary-based approaches attempt to find "texture edges" between pixels that come from different texture distributions. We leave the discussion of segmentation algorithms to Chapter 10 on Image Segmentation. Figure 7.8 shows the segmentation of several images using the Laws texture energy measures and a clustering algorithm to group pixels into regions.

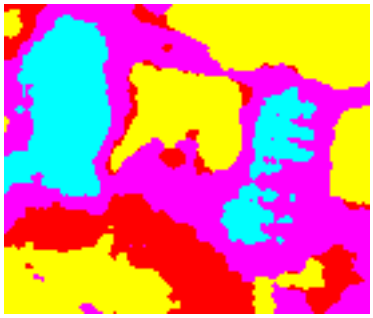
In Figure 7.8(a) and (b), the tiger image has been segmented into regions representing tiger, water, and some other miscellaneous areas of the image. In Figure 7.8(c) and (d) a multi-object image has been segmented into regions that roughly correspond to the grass, the two flags, the black mesh fence and some background. In Figure 7.8(e) and (f), the sunflower image has been segmented into three types of texture: dark borders found at the top and bottom of the image, small, far-away sunflowers at the back of the field and large, close-up sunflowers at the front of the field. Table 7.2 shows the mean Laws texture energy measures for the main regions of each of these images. Table 7.3 gives a comparison of the Laws measures for tiger regions in several different images.

(a) Original image

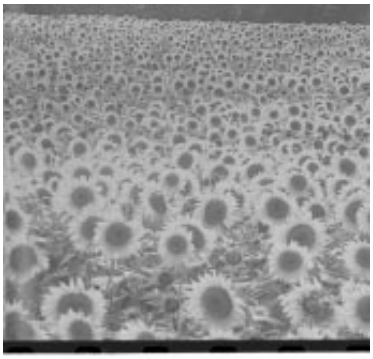


(b) Segmentation into 4 clusters

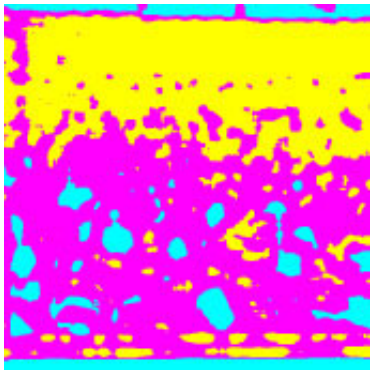
(c) Original image



(d) Segmentation into 4 clusters



(e) Original image



(f) Segmentation into 3 clusters

Figure 7.8: Examples of image segmentation using the Laws texture energy measures (original images are from Corel Stock Photos and from the MIT Media Lab VisTex database).

Table 7.2: Laws texture energy measures for major regions of the images of Figure 7.8.

Region	**E5E5**	**S5S5**	**R5R5**	**E5L5**	**S5L5**	**R5L5**	**S5E5**	**R5E5**	**R5S5**
Tiger	168.1	84.0	807.7	553.7	354.4	910.6	116.3	339.2	257.4
Water	68.5	36.9	366.8	218.7	149.3	459.4	49.6	159.1	117.3
Flags	258.1	113.0	787.7	1057.6	702.2	2056.3	182.4	611.5	350.8
Fence	189.5	80.7	624.3	701.7	377.5	803.1	120.6	297.5	215.0
Grass	206.5	103.6	1031.7	625.2	428.3	1153.6	146.0	427.5	323.6
Small flowers	114.9	48.6	289.1	402.6	241.3	484.3	73.6	158.2	109.3
Big flowers	76.7	28.8	177.1	301.5	158.4	270.0	45.6	89.7	62.9
Borders	15.3	6.4	64.4	92.3	36.3	74.5	9.3	26.1	19.5

Table 7.3: Laws texture energy measures for tiger regions of several different images.

Image	**E5E5**	**S5S5**	**R5R5**	**E5L5**	**S5L5**	**R5L5**	**S5E5**	**R5E5**	**R5S5**
Tiger1	171.2	96.8	1156.8	599.4	378.9	1162.6	124.5	423.8	332.3
Tiger2a	146.3	79.4	801.1	441.8	302.8	996.9	106.5	345.6	256.7
Tiger2b	177.8	96.8	1177.8	531.6	358.1	1080.3	128.2	421.3	334.2
Tiger3	168.8	92.2	966.3	527.2	354.1	1072.3	124.0	389.0	289.8
Tiger4	168.1	84.0	807.7	553.7	354.4	910.6	116.3	339.2	257.4
Tiger5	146.9	80.7	868.7	474.8	326.2	1011.3	108.2	355.5	266.7
Tiger6	170.1	86.8	913.4	551.1	351.3	1180.0	119.5	412.5	295.2
Tiger7	156.3	84.8	954.0	461.8	323.8	1017.7	114.0	372.3	278.6

In the sunflower image, some of the larger flowers, some of the dark flower centers are grouped with the dark border textures, because the mask used to compute the texture is smaller than some of the larger flowers. In general, these segmentation results are imperfect; they can do no better than the operators that define them. Segmentation based on both color *and* texture can do better, but segmentation of natural scenes is an unsolved problem. See Chapter 10 for a more comprehensive treatment of segmentation in general.

7.5 References

1. G. R. Cross and A. K. Jain, "Markov Random Field Texture Models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-5, 1983, pp. 25-39.

2. R. M. Haralick, K. Shanmugam, and I. Dijnstein, "Textural features for image classification," *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 3, 1973, pp. 610-621.

3. B. Julesz, "Experiments in the Visual Perception of Texture," *Scientific American*, 1975, pp. 34-43.

Exercise 7 Texture Segmentation

Use your program that computes the Laws texture energy measures for an image to investigate how well they would perform for texture segmentation. Write another interactive front end that allows the user to draw boxes around sections of the image that contains a single category of texture such as flowers or grass or sky. For each box, compute the the average values for the nine texture features. Produce a table that lists each texture category by name and prints the nine averages for that category. Compare the results on several different categories.

4. K. Laws, "Rapid Texture Identification," in *SPIE Vol. 238: Image Processing for Missile Guidance*, 1980, pp. 376-380.

5. H. Tamura, S. Mori and T. Yamawaki, "Textural Features Corresponding to Visual Perception," *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 8, NO. 6, 1978, pp. 460-473.

6. F. Tomita, Y. Shirai, and S. Tsuji, "Description of Textures by a Structural Analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-4, 1982, pp. 183-191.

7. M. M. Trivedi, "Object Detection Based on Gray Level Cooccurrence," *Computer Vision, Graphics, and Image Processing*, Vol. 28, 1984, pp. 199-219.

8. M. Tuceryan and A. K. Jain, "Texture Analysis," in *Handbook of Pattern Recognition and Computer Vision*, Eds. C. H. Chen, L. F. Pau, and P.S. P. Wang, World Scientific Publishing Co., 1994, pp. 235-276.

9. M. Tuceryan and A. K. Jain, "Texture Segmentation Using Voronoi Polygons," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, VOl. 12, No. 2, 1990, pp. 211-216.

10. L. Wang and D. C. He, "Texture classification Using Texture Spectrum," *Pattern Recognition Letters*, Vol. 13, 1990, pp. 905-910.

11. J. Weszka, C. R. Dyer, and A. Rosenfeld, "A Comparative Study of Texture Measures for Terrain Classification," *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. SMC-6, 1976, pp. 269-285.

12. S. W. Zucker and D. Terzopoulos, "Finding Structure in Co-occurrence Matrices for Texture Analysis," *Computer Graphics and Image Processing*, Vol. 12, 1980, pp. 286-308.