UNSUPERVISED SEGMENTATION OF TEXTURED IMAGES

Alireza Khotanzad and Jong-Yih Chen

Image Processing and Analysis Laboratory
Department of Electrical Engineering
Southern Methodist University
Dallas, Texas 75275
U.S.A
Phone: (214) 692-3197

Abstract:
This paper describes a procedure for segmenting an image consisting of several regions with different texture into regions of similar texture. The technique does not require training prototypes but operates in an "unsupervised" mode. In addition, no knowledge about the number of regions in the underlying image is needed. Segmentation is achieved by finding "textural edges", i.e. boundaries of homogeneously textured regions. Textural edges are defined to be at positions where abrupt changes in textural features of small neighboring regions occur. To detect them, the image is scanned by a small size sliding window. Six features are extracted from the encompassed region by each window. These features are the estimated parameters of two two-dimensional, non-causal random field models which capture region characteristics in [horizontal, vertical] and [diagonal, off-diagonal] directional pairs. Sobel edge enhancement operator is used to accentuate the changes in individual features over local blocks of three windows wide by three windows high. A new image is then created in which a pixel value corresponds to a measure of "textural change" constructed from the normalized outputs of the Sobel operator and integrating the effects of all the features. This "textural change image" is thresholded to get a binary edge/no edge image. The location of the edge pixels are mapped onto the original image in the spatial domain. Finally, an edge thinning algorithm is applied to obtain one-pixel wide edges. A novel approach for automatic selection of the size of the scanning window is presented. Two window sizes are used instead of one and the common resulting edges are picked. The goodness of the method is demonstrated through experimental studies involving images containing two and three different natural textures.

Keywords:
Segmentation, texture, random field models, textural edge detection.

I. Introduction

One of the most difficult problems in image segmentation is to divide an image into distinct regions, each of which are more or less homogeneously textured, and to establish the boundaries among them. In general, the problem is an unsupervised pattern classification task in the sense that no a priori information about the number of regions and the types of textures in the underlying image is available. This is a missing feature in the majority of previously developed texture segmentation algorithms. They usually make unrealistic assumptions regarding the availability of some a priori knowledge which makes the problem easier to solve.

Addressing the problem in the stated framework presents three major issues to be resolved. They are: (1) selection of *m good* features to represent texture, (2) selection of an appropriate size for the scanning window, (3) devising a way to combine the contributions of individual features to create a measure of "total change".

There has been quite a number of different approaches toward development of textural features over the years [1],[2]. The segmentation algorithm which is presented here can employ any set of m features provided that they satisfy the following three requirements. (1) Within-class invariance: this means that features extracted from several images all having a similar texture should have numerically close values. (2) Between-class separation: implies that features extracted from images with different textures be quite different. (3) Low sensitivity to small sample size: means that satisfying requirements (1) and (2) should not be contingent upon availability of large size texture fields. The features extracted from small size regions should satisfy the above requirements as well. The reason for this requirement will become clear later when the algorithm is outlined.

Any set of features with strong class discrimination power will do well. In this study, our selected features are derived from a class of spatial interaction random field models called "Simultaneous Autoregressive (SAR)" model [3]. The estimates of the parameters of the SAR models fitted to the image are taken as textural features. Specifically, six parameters estimated from two different SAR models which capture region characteristics in [horizontal, vertical] and [diagonal, off-diagonal] directional pairs are used in this study. The effectiveness of such features in supervised texture classification problems as shown in [4] provided the motivation for their use in here.

A second and very important problem is how to select an appropriate size for the scanning window over

which local characteristics are observed and measured. Texture is a regional property. Hence, the window should be large enough for the region enclosed by it to exhibit similar textural characteristics as that of the underlying region that it masks. At the same time, it should be as small as possible to enable accurate detection of the edges. Operating in unsupervised mode prevents a priori selection of a good window size. In fact, a major shortcoming of the previously developed edge-based algorithms is the arbitrary preselection of this parameter. In this study, a systematic method is presented which enables the algorithm to automatically select an appropriate window size.

The final issue is how to integrate the effects of the selected m features to construct a measure which reflects total change in local characteristics. Note that usually not all of the m features but only a subset of them is useful for discrimination between different textures. A subset that might work well for texture types A and B may not be as powerful for textures B and C and a different subset should be considered. Therefore, it is necessary to develop a mechanism which picks a good subset of features in a given case and combines their contributions.

The general idea of extracting textural edges by finding discontinuities in local texture properties has been pursued by a number of other authors. Zobrist and Thompson [5] constructed a textural distance function which is a weighted sum of a collection of elementary distance functions, each corresponding to a difference in a textural property between two regions. This measure constructed from 43 features along with a modified Roberts gradient operator applied to local neighborhoods consisting of four nonoverlapping 16×16 blocks was used in [6] to detect textural boundaries.

Grinaker [7] used a similar type measure but tried to achieve a better accuracy by following a coarse-to-fine segmentation approach. A coarse edge-zone was first established by considering large blocks. This zone is subsequently explored in more detail by using smaller blocks. The features used to describe texture were mean and variance of gray-levels.

Triendl and Henderson [8] approach the problem in a different order. Instead of detecting edges in measurement constructed from multiple textural cues, they first found edges in each of the 22 utilized features individually and then combined the results. Only edge pixels located in the same spatial locations in at least two features were retained. An arbitrary 11×11 neighborhood around each pixel was used to extract the corresponding features.

II. Selected Textural Features

Texture is viewed as two-dimensional local variations of gray-levels. These variations are considered random in nature with varying degrees of randomness associated with them. Regularly structured, periodic textures have a low degree of randomness (corresponding to a small noise variance associated with the model), and vice versa. One can quantify such characterization by modeling the spatial interaction among pixels using a class of two-dimensional, non-causal, stochastic random field models called "Simultaneous Autoregressive (SAR)" models [3].

Let $\{g(x,y);\ x,y=0,1,\cdots,M-1\}$ be a two-dimensional array representing the gray level values of a discrete $M\times M$ image with the top left of the image at $(x=0,y=0)$ and the bottom right at $(x=M-1,y=M-1)$. If $g(x,y)$ obeys a SAR model defined over a $M\times M$ toroidal lattice, then

$$g(x,y) = \sum_{(i,j)\in N} \theta_{(i,j)}\, g(x\oplus i, y\oplus j) + \sqrt{\rho_N}\,\omega(x,y),$$

$$x,y = 0,1,\cdots,M-1,$$

where

- N = a neighbor set defined in the spatial domain.
- $\theta_{(i,j)}$ = coefficients of the model characterizing the dependence of a pixel to its neighboring pixels. If both (i,j) and $(-i,-j)\in N$, then $\theta_{(i,j)} = \theta_{(-i,-j)}$.
- $\oplus$ = addition modulo M. Acts like an ordinary addition except at those (x,y) points located on or near the edges of image for which a complete neighbor set cannot be found. In this situation, $\oplus$ operation causes a wrap around (torus lattice) effect and creates a full neighbor set for each pixel.
- $\omega(x,y)$= independent identically distributed (i.i.d.) Gaussian random variable with zero mean and unit variance, characterizing fluctuations.
- ρ_N = overall variance of the noise which is a measure of randomness of texture.

N, which defines a neighborhood around each pixel, consists of a set integer pairs excluding (0,0). These pairs are the relative grid locations of the neighboring pixels in the sense that the spatial locations of a pixel's neighbors are found by adding these pairs to the coordinates of the respective pixel. The labeling convention for N and a simple set $N_1 = \{(1,0),(0,1),(-1,0),(0,-1)\}$ are shown in Fig. 1. A SAR model with neighbor set N_1 assumes that the intensity level of each pixel in the image depends on the gray levels of the pixels above, below, to the left, and to the right of it. The additive noise accounts for fluctuations in such a relation throughout the image.

The extracted features from SAR models are the least squares estimates of the model parameters [9] denoted by $\{\hat{\theta}_{(i,j)};\ (i,j)\in N,\ \hat{\rho}_N\}$. The details of the estimation will be derived in the next section. In this study, six such features derived from two SAR models are utilized. They are $\{\hat{\theta}_{(0,1)},\ \hat{\theta}_{(1,0)},\ \hat{\rho}_{N_1},\ \hat{\theta}_{(1,1)},\ \hat{\theta}_{(-1,1)},\ \hat{\rho}_{N_2}\}$. The first three are from a SAR model with neighbor set $N_1 = \{(1,0),(0,1),(-1,0),(0,-1)\}$ and the remaining three are calculated using $N_2 = \{(1,1),(-1,1),(-1,-1),(1,-1)\}$. These six features were found to have a powerful discriminative ability for texture [10] and thus were utilized here.

II.1 Least Squares Estimation Of SAR Parameters

If the image obeys a SAR model, the intensity

errors between the estimated model and the actual one can be formulated in terms of the noise sequence $\omega(\bullet)$. The corresponding sum of squares of errors are expressed as

$$q = \sum_{x=0}^{M-1}\sum_{y=0}^{M-1}\left[\, g(x,y) - \theta^T z(x,y)\,\right]^2$$

$$= \sum_{x=0}^{M-1}\sum_{y=0}^{M-1}\left[g^2(x,y) - 2\theta^T z(x,y) g(x,y) + \theta^T z(x,y) z^T(x,y)\theta\right]$$

where

$$z(x,y) = \text{col.}\left[\, g(x\oplus i, y\oplus j);\ (i,j)\in N\,\right]$$

The LS estimates of θ and ρ_N can be obtained by minimizing the sum of squares with respect to θ. This is done by setting

$$\frac{\partial q}{\partial \theta} = 0$$

which results in

$$\hat{\theta} = \left[\sum_{x=0}^{M-1}\sum_{y=0}^{M-1} z(x,y) z^T(x,y)\right]^{-1}\left[\sum_{x=0}^{M-1}\sum_{y=0}^{M-1} z(x,y) g(x,y)\right]$$

Since $\omega(\bullet)$ is assumed to be zero mean, its variance can be estimated as

$$\hat{\rho}_N = \frac{1}{M^2}\sum_{x=0}^{M-1}\sum_{y=0}^{M-1}\left[\, g(x,y) - \hat{\theta}^T z(x,y)\,\right]^2$$

Note that $\omega(\bullet)$ need not be Gaussian and can have an arbitrary distribution.

III. Segmentation Procedure

The image is scanned from top to bottom in a row by row manner by a L×L window (L ≪ image size). Discussions regarding selection of L are presented later. The window slides over the image in D-pixel wide in both horizontal and vertical directions. Therefore, successive windows overlap. The six discussed SAR features are extracted from each window. Textural edges could then be detected by finding those locations where a drastic change in the features of neighboring windows occurs.

This is done in a manner similar to gray level edge detection. Many operators for enhancement of gray level edges are available. Sobel operator is one of those. It operates on 3×3 neighborhood of a pixel and accentuates edges by outputting high values elsewhere. The same operator can be used to enhance changes in th ith feature, f_i, i=1,...,6, (i.e., $f_1=\hat{\theta}_{(0,1)}$, $f_2=\hat{\theta}_{(1,0)}$, etc.) throughout the image. This is accomplished by replacing brightness level with f_i value, and a 3×3 neighborhood with a block of nine adjacent windows (three windows wide by three windows high). For each f_i, Sobel operator yields two outputs which can be expressed as

$$S_{\bar{f}_i}(x,y) = \left[\, S^2_{H-\bar{f}_i}(x,y) + S^2_{V-\bar{f}_i}(x,y)\,\right]^{1/2}$$

where

$$S_{H-\bar{f}_i}(x,y) = [\bar{f}_i(w(x-D,y+D)) - \bar{f}_i(w(x-D,y-D))] + 2\,[\bar{f}_i(w(x,y+D)) - \bar{f}_i(w(x,y-D))] + [\bar{f}_i(w(x+D,y+D)) - \bar{f}_i(w(x+D,y-D))]$$

$$S_{V-\bar{f}_i}(x,y) = [\bar{f}_i(w(x+D,y-D)) - \bar{f}_i(w(x-D,y-D))] + 2\,[\bar{f}_i(w(x+D,y)) - \bar{f}_i(w(x-D,y))] + [\bar{f}_i(w(x+D,y+D)) - \bar{f}_i(w(x-D,y+D))]$$

with $S_{H-\bar{f}_i}$ and $S_{V-\bar{f}_i}$ representing enhancement in horizontal and vertical directions respectively, $\bar{f}_i$ standing for normalized ith features, w(x,y) standing for an L×L window centered on (x,y) pixel, and D representing the displacement in pixels between the centers of two adjacent windows in either horizontal or vertical direction. f_i is normalized to have zero sample mean and unit sample variance by

$$\bar{f}_i(w(x,y)) = \frac{f_i(w(x,y)) - \mu_{f_i}}{\sigma_{f_i}}$$

$$\mu_{f_i} = \frac{1}{J}\sum_{k=1}^{J} f_i^{(k)}$$

$$\sigma_{f_i}^2 = \frac{1}{J-1}\sum_{k=1}^{J}\left(f_i^{(k)} - \mu_{f_i}\right)^2$$

where J is the total number of windows in the process of scanning the image, and $f_i^{(k)}$ refers to the ith feature of the kth window, k=1,...,J.

A one-dimensional measure of "textural change", denoted by TC, is constructed as a function of these individual changes. In general, if m features are available, the TC measure is expressed as

$$TC(w(x,y)) = \left\{\sum_{k=1}^{n} Y_{(k)}^2(x,y)\right\}^{1/2},\quad n=1,2,\cdots,m$$

where $Y_{(k)}$ is the reverse order statistics of $S_{\bar{f}_i}$, i.e. $S_{\bar{f}_i}$ ordered from largest to smallest. Note that

$$Y_{(1)}(x,y) = \text{Max}\,[S_{\bar{f}_i}(x,y)],\qquad i=1,\cdots,m$$

$$Y_{(m)}(x,y) = \text{Min}\,[S_{\bar{f}_i}(x,y)],\qquad i=1,\cdots,m$$

and

$$Y_{(1)}(x,y) \geq Y_{(2)}(x,y) \geq \cdots \geq Y_{(m)}(x,y)$$

The structure of this measure explains the reason for normalizing the features. The dynamic ranges of the features are different and if not normalized, one or a subgroup of them will dominate TC.

Using the above measure, a new two-dimensional picture is generated in which the value of TC(w(x,y)) represents the pixel value at location (x,y). Let us call this image "TC image". Note that each (x,y) pixel in TC image represents the D×D central block of a window in the original image. Hence the size of the TC image will be $[\frac{M-L}{D}-1]$. From this point on, the operation takes the form of classical edge detection in the TC image.

To detect edges, the TC image needs to be thresholded to create a binary edge/no-edge picture which will be called "TC image edge mask"(TCE). The threshold t is automatically selected to be at the midpoint between 95th and 5th percentiles of the compute TC values.

$$t = \frac{1}{2}\left\{ \text{95th percentile } [TC(w(x,y))] \right\} - \frac{1}{2}\left\{ \text{5th percentile } [TC(w(x,y))] \right\}$$

The reason for using percentiles is to suppress any large swings at both ends and preventing them from dominating the mapping. Such a threshold clearly creates very thick edges as well as isolated noisy edge pixels. The isolated edge pixels with no 8-connected neighbors are deleted next.

A reverse mapping from the feature space back to the spatial domain costitutes the next stage. This is done through enlarging the TCE image by growing each 0 or 1 pixel to a D×D block of 0s or 1s. The process is followed by an edge thinning and skeletonization procedure [11] to get the final one pixel wide edges.

III.1 Window Size Selection

It is apparent that L, the size of the sliding window used to scan the image, is an important factor in the success of the discussed strategy. L has to be as small as possible to enable accurate location of the borders between regions without a blocky outlook. At the same time, to avoid detecting micro-edges within a uniformly textured region, L has to be large enough with respect to element size of the underlying texture. In other words, the resolution of texture plays an important role in selection of L.

Availability of some a priori knowledge about the type and resolution of textures to appear in the image solves the problem to a great extent. However, in a totally unsupervised case, such information is not available and other means have to be deployed.

If the selected L is smaller than the element size of one or more textures in the image, the 1-pixels in the resulting TCE image will come from two sources, micro-edges of the texture whose element size is large and the textural edges. In that case, using a slightly larger L should eliminate some of the edge pixels of TCE image which correspond to micro-edges. However, those edge pixels that are due to textural edges will not be affected. Based on this assertion, the arrangement and spread of 1-pixels in the TCE image are utilized for selection of a pair of window sizes for the suggested operation. The following ratio is used as a measure for the spread of 1-pixels in a TCE image obtained by using window size L,

$$SPRD[TCE(L)] = \frac{\mu_{20}+\mu_{02}}{M_{00}}$$

where μ_{ij} and M_{ij} stand for ijth central and regular moments of the image respectively. They are defined as follows.

$$M_{ij} = \sum_{x=0}^{M-1}\sum_{y=0}^{M-1} x^i y^j \, g(x,y)$$

$$\mu_{ij} = \sum_{x=0}^{M-1}\sum_{y=0}^{M-1} (x-\bar{x})^i \, (y-\bar{y})^j \, g(x,y)$$

$$\bar{x} = \frac{M_{10}}{M_{00}} \quad , \quad \bar{y} = \frac{M_{01}}{M_{00}}$$

The suggested window size selection process starts by specifying a range for L, $L \in [L_{min}, L_{max}]$. Note that merely selecting a range for L does not violate the requirement of operating in an unsupervised mode. The algorithm then proceeds in the following steps.

(1) $L_1 = L_{min}$
(2) Compute $TCE(L_1)$ and $SPRD[TCE(L_1)]$
(3) $L_2 = L_1 + \Delta L$
(4) If $L_2 + \Delta L > L_{max}$, stop. Flag the need for increase of L_{max} or ϵ.
(5) Compute $TCE(L_2)$ and $SPRD[TCE(L_2)]$
(6) If $|\, SPRD[TCE(L_1)] - SPRD[TCE(L_2)] \,| < \epsilon$, go to step 9
(7) $L_1 = L_2$
(8) go to step 3
(9) $TCE = TCE(L_1)$ .AND. $TCE(L_2)$

In summary, L is incremented by ΔL at each step and the resulting change in the spread of 1-pixels in the corresponding TCE images is checked. If the change is below a preselected threshold, the process stops. The output is a TCE image which is the result of a logical AND operation between two TCE images corresponding to the present and previous values of L, i.e. L_2 and L_1.

III.2 Selection Of Number Of Features

Another parameter of the algorithm to be selected is the number of features to be used in construction of TC measure. The simplest choice is $n=m$. In fact, selection of L_1 and L_2 are carried out by using $n=m$. However, this may not lead to the best segmentation result. The reason being that some of the utilized features may not have a strong discrimination power for the considered textures. Integration of the effect of these weak features in the TC measure will tend to smooth out sharp changes in the measure which are due to the strong features. Hence, it will be helpful to decide upon an n which will give the best result.

The criterion used for selection of n is similar to the one used to select L, namely the spread of 1-pixels in the resulting TCE image. After L_1 and L_2 are decided upon by using $n=m$, n is varied from 1 to m and the resulting TCE images and their corresponding spread function is calculated. In other words, if TCE image using p features is denoted by TCE_p,

$$TCE_p = TCE(L_1) \text{ .AND. } TCE(L_2) \;|_{n=p} \qquad p=1,\cdots,m$$

Then $n=p^*$ where $p^* = \text{Min}_p\ SPRD(TCE_p)$. In other words, that n is selected which results in a TCE image whose spread function is minimum among all the other TCE images.

IV. Experimental Study

The performance of the proposed segmentation algorithm is examined by experiments on four images containing different regions of natural textures. These images are shown in Fig. 2. The actual textural edges are outlined on the pictures. Each image is 128×128 with 256 gray levels. All of the textures are digitized from [12]. The numerical values of some of the algorithm parameters that need to be selected before hand

are presented below:

- Minimum value for window size, $L_{min}=8$
- Maximum value for window size, $L_{max}=24$
- Increment for window size enlargement in the window size selection algorithm, $\Delta L=4$ (Allowable windows are superimposed on the original pictures in Fig. 2 for comparison purpose)
- Step size for sliding of the observation window, $D=5$
- Threshold value for binarizing TC image into TCE image, $t=128$
- Stopping criterion for window size selection algorithm, $\epsilon=.02$
- The range considered for n, $n=3, \cdots, 6$.

In the first experiment, a digitized picture made up of three different regions of natural texture was considered. The picture is shown in Fig. 2(a). Wood on top, leather to the left, pig skin to the right make up this image. The location of the boundaries looks like a "Y" with irregular lines.

Fig. 3 shows the resulting 19×19 TC and TCE images using different window sizes. Note that the spread and irregularity of 1-pixels in the TCE images decreases as L increases. Table I(a) provides the computed statistics used for selection for L_1 and L_2. Since $\epsilon=.02$, $L_1=12$ and $L_2=16$ are selected and there is no need to examine larger windows. Table I(b) suggests that three features are better than all six, i.e., $n=3$. The final detected boundaries are superimposed on the original picture and are shown in Fig. 4(a).

In the second experiment image of Fig. 2(b) is considered. The textures are the same as those in Fig. 2(a) but the resolution and orientation of wood and leather are changed. The algorithm selects $L_1=16$, $L_2=20$, and $n=3$. The final result is shown in Fig. 4(b)

The third image (Fig. 3(c)) involves two textures whose textural element sizes are quite different. $L_1=20$, $L_2=24$,and $n=3$ are selected resulting in detected edges shown in Fig. 4(c).

In the last experiment, image of Fig. 2(d) which consists of two regions of leather texture with an orientation difference of 90° is considered. The appropriate parameters are found to be $L_1=12$, $L_2=16$, and $n=3$. Fig. 4(d) shows the detected edges.

In all of the above experiments, the detected boundaries were very close to the actual ones signaling the success of the developed algorithm.

V. Conclusion

An algorithm for finding the boundaries among differently textured regions in an image is presented. It is based on scanning the image with a small window and locating positions where a drastic change in textural features of neighboring windows happen. This is done by constructing a measure of "total change" from changes in individual elements of the multidimensional feature vector. The individual changes are computed in a manner similar to gray-level edge detection using a Sobel operator with a modified region of operation. Instead of using a single window, two windows whose sizes vary by a few pixels are used and the common resulting edges are taken. A novel method based on the spread of edge pixels is presented for selection of the appropriate pair of window sizes. Six features derived from two different random field models are used in this study. The good performance of the algorithm is demonstrated through experimental studies.

References

[1] L. V. Gool, P. Dewaele, and A. Oosterlinck, "Survey: Texture analysis anno 1983," *Computer Vision, Graphics, and Image Processing*, vol. 29, pp. 336-357, 1985.

[2] R. M. Haralick, "Statistical and structural approaches," *Proceedings of IEEE*, vol. 67, pp. 786-804, 1979.

[3] R. L. Kashyap, "Analysis and synthesis of image patterns by spatial interaction models," *Progress in Pattern Recognition*, Edited by L. N. Kanal, and A. Rosenfeld, vol. 1, North-Holland, New York, pp. 149-186, 1981.

[4] A. Khotanzad and R. L. Kashyap, "Feature selection for texture recognition using image synthesis," to appear in *IEEE Trans. on Syst. Man and Cybern.* vol. 17, No. 6, Nov. 1987.

[5] A. I. Zobrist and W. B. Thompson, "Building a distance function for Gestalt grouping," *IEEE Trans. on Computers*, vol. C-24, No. 7, pp. 718-728, 1975.

[6] W. B. Thompson, "Textural boundaries analysis," *IEEE Trans. on Computers*, vol. C-26, pp. 272-276, March 1977.

[7] S. Grinaker, "Edge based segmentation and texture separation," in *IEEE Proc. 5th Int. Confer. on Pattern Recognition*, Miami Beach, Florida, Dec. 1-4, pp. 544-557, 1980.

[8] E. Triendl and T. Henderson, "A model for texture edges," in *IEEE Proc. 5th Int. Confer. on Pattern Recognition*, Miami Beach, Florida, Dec. 1-4, pp. 1100-1102, 1980.

[9] R. L. Kashyap and R. Chellappa, "Estimation and choice of neighbors in spatial interaction models of images," *IEEE Trans. on Information Theory*, vol. IT-29, pp. 60-72, 1983.

[10] R. L. Kashyap and A. Khotanzad, "A stochastic model based technique for texture segmentation," in *IEEE Proc. 7th Int. Conf. on Pattern Recognition*, Montreal, Canada, July 30-August 2, pp. 1202-1205, 1984.

[11] A. Rosenfeld and A. C. Kak, *Digital Picture Processing*, vol. 2, Academic Press, New York, N.Y., 1982, pp. 232-240.

[12] P. Brodatz, *Texture: A Photographic Album for Artists and Designers*, Dover, New York, N.Y., 1956.

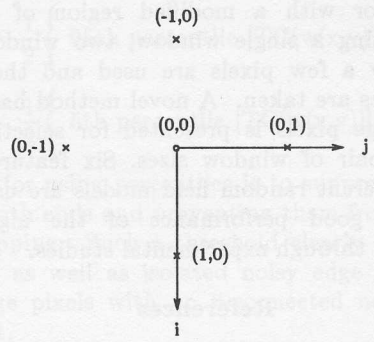

Fig. 1. Relative lattice locations of symmetric neighbor set $N_1 = \{ (1,0), (0,1), (-1,0), (0,-1) \}$.

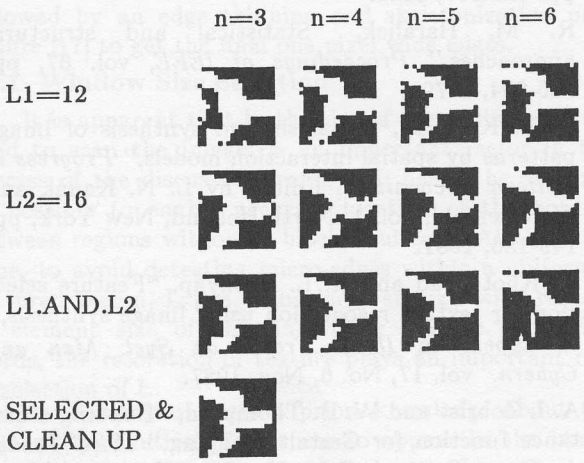

Fig. 3. TCE images resulting from the selected L_1 and L_2 and n=3, $\cdots$, 6. n=3 is selected.

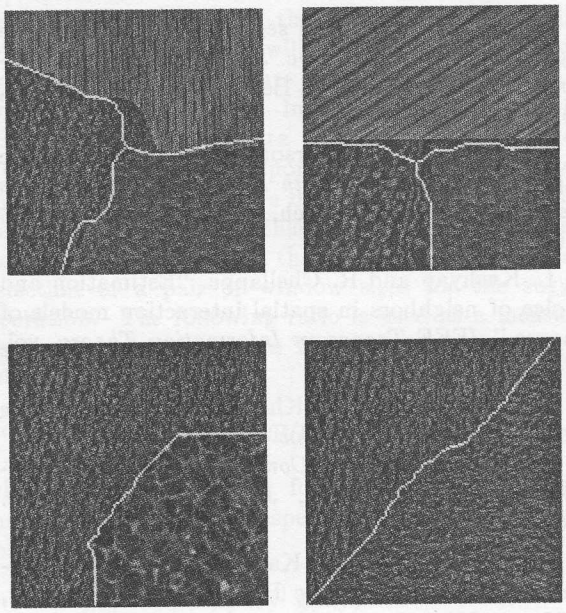

Fig. 4. The detected edges superimposed on the original images.

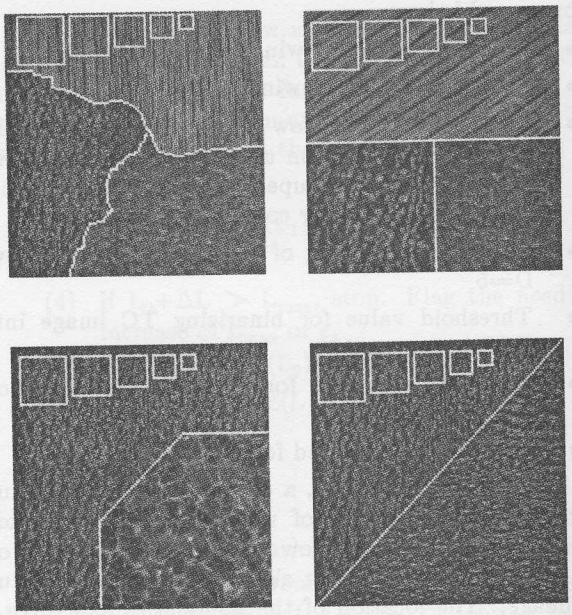

Fig. 2. Images to be segmented. Actual boundaries are outlined. The allowable window sizes are superimposed.

(a) Wood on top, pigskin to the right, and leather to the left.

(b) Same textures as in (a). Resolution and orientation of wood and leather are different from (a).

(c) Leather to the left of the border, plastic bubble to its right.

(d) Two regions of leather texture with 90° difference in orientation.

Table I

Statistics for image of Figure 2(a).

(a) Selection of L_1 and L_2. The selected ones are shown by *.
(b) Selection of n. The selected one is shown by *.

L	μ_{20}	μ_{02}	M_{00}	SPRD[TCE(L)]	Change in SPRD
8	50.852	51.432	162	.63	NA
12 *	35.988	35.827	120	**.60**	.03
16 *	31.037	36.407	113	**.60**	.00
20	31.370	31.161	109	.57	.03
24	42.877	43.827	147	.59	.02

(a)

TCE_n	μ_{20}	μ_{02}	M_{00}	SPRD[TCE_n]
n=3 *	31.383	32.839	126	**.5097**
n=4	27.185	31.012	112	.5196
n=5	24.506	27.210	95	.5444
n=6	24.383	28.926	93	.5732

(b)