1986

Characterization and classification of textures in digitized images.

Hukam C. Gupta
University of Windsor

CHARACTERIZATION AND CLASSIFICATION OF TEXTURES IN DIGITIZED IMAGES

by

Hukam C. Gupta

A thesis
presented to the University of Windsor
in partial fulfillment of the
thesis requirement for the degree of
Master of Applied Sciences
in
Electrical Engineering

Windsor, Ontario, 1986

ISBN 0-315-32007-9

ABSTRACT

Texture analysis is an important aspect of many image analysis tasks. The texture of a surface is characterized by properties such as fine, coarse, smooth, granulated, rippled, irregular, random, etc.

This thesis deals with three main problems of texture analysis, namely, texture description (Given a textured region, how can it be described?), texture discrimination or classification (Given a textured region or sample, to which of a finite number of classes does the sample belong?), and texture based segmentation (Given a scene, how can it be segmented into different texturally homogeneous regions?). A critical analysis of the texture discrimination technique proposed by Raafat, 1985 is performed and the results are compared with the spatial gray level dependence method given by Haralick et al., 1973. The results obtained indicate that the scheme proposed by Raafat, 1985 could be as accurate as the method proposed by Haralick et al., 1973. The final part of the thesis introduces a new segmentation algorithm based on a region growing technique. Various advantages and limitations of this method are also discussed.

ACKNOWLEDGEMENTS

I gratefully acknowledge Prof. M. Shridhar for his invaluable guidance and encouragement at every stage of this work. I would also like to acknowledge Prof. M. Ismail, my co-supervisor, for his generous help and assistance. I extend my sincerest thanks to the committee members, Prof. M. Ahmadi and Prof. P. Alexander for their kind suggestions and help.

I wish to express my deep appreciation to my wife, Aruna Gupta, for her help and understanding throughout the pursuit of this degree. Many thanks are also due to my family members and friends for their concerns and inspirations.

CONTENTS

CHAPTER I

INTRODUCTION

With the advent of digital computers, it has become possible to simulate some aspects of human perception by machines. For the last two decades a significant amount of work has been done in the area of image understanding and image analysis. The main interest in this area is to extract image information for human interpretation or for machine processing. In the latter case the extracted data must be in a form acceptable to digital computers.

The problem of texture characterization and texture based segmentation has become critically important to many researchers in various fields of image processing. Several approaches to this problem have been reported in the literature (e.g. Haralick, 1979; Weszka, 1976; Gupta, 1975). The areas of applications include scene analysis, remote sensing (e.g. multispectral image analysis, classification and monitoring of urban areas etc.), industrial automation (e.g. parts identification on assembly lines, defects and faults inspection), medical imaging (e.g. radiographs, photographs of the ocular fundus and blood samples etc.), and military applications.

It has long been recognized that texture plays an important role in establishing and classifying objects and regions in a given image (Ehrich et al., 1977; Wechsler et al., 1980). Texture is important because it covers the entire area of an object. For instance, consider the contour detection problem, where a distinct rectangular contour is given in an aerial photograph (Figure 1). The shape of this rectangular region does not give sufficient information about the object; it may be grassland, forest, and so on. In such cases, the properties like color and gray level are not sufficient to describe the object and we must consider more complicated properties. For example, in Figure 2, the areas shown are not homogeneous on the basis of pointwise gray level. It is clear that they exhibit homogeneous properties on a large basis. These complicated properties which are required in order to describe the interior of an object are referred to as textural properties (Niemann, 1981). Also according to Sklansky, "Local properties of a region may be constant, slowly varying, or approximately periodic, a discontinuity in the textural properties between different adjacent regions can be detected and hence be used to segment a given image". In spite of its importance in image anlysis problems there is no rigid description of texture and there is no agreement on how to measure it. Texture has been defined in different forms by different authors. A few of these definitions are listed below.

"Texture is the term used to characterize the surface of a given material", (Wechsler and Citron, 1980). According to Sklarsky, 1978, "A region in an image has a constant texture if a set of local statistics or other local properties of the picture function are constant, slowly varying, or approximately periodic". In general, we can say that texture is a regional property of an image function which refers to the spatial distribution of intensity variations. And it is observed in the structural patterns of surfaces of objects such as wood, grain, grass, cloth, sand and so on. Figure 2 shows some examples of textures like beach pebbles, lizard skin, handwoven oriental rattan, and plastic pellets.

Visual texture is a difficult concept to define, but it is commonly considered as the repetitive occurrence of local patterns in the given region. Thus one can describe a texture by describing its local patterns and their placement rule. A texture element is a visual primitive with certain invariant properties which occur repeatedly in different positions, deformations, and orientations inside a given textured region. The elements which are smaller in size (having a diameter of a few pixels) form a micro-texture whereas elements which are bigger in size form a macro-texture.

In this topic, there are three main problems of interest, namely:.

1. Texture Description

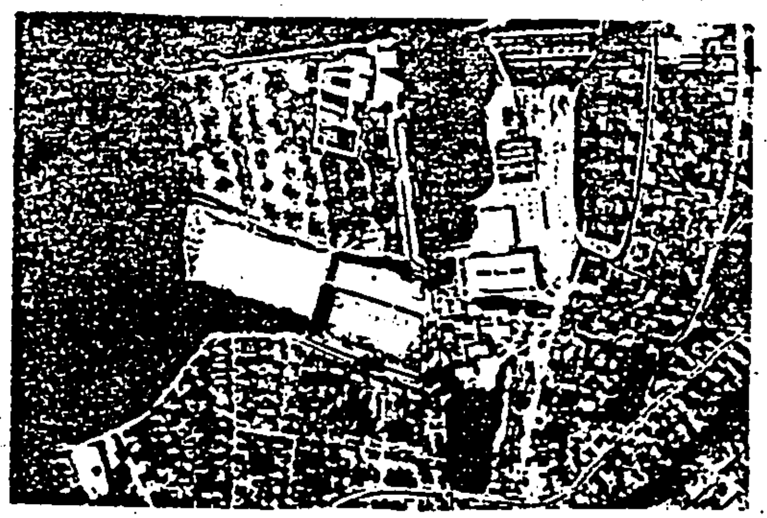
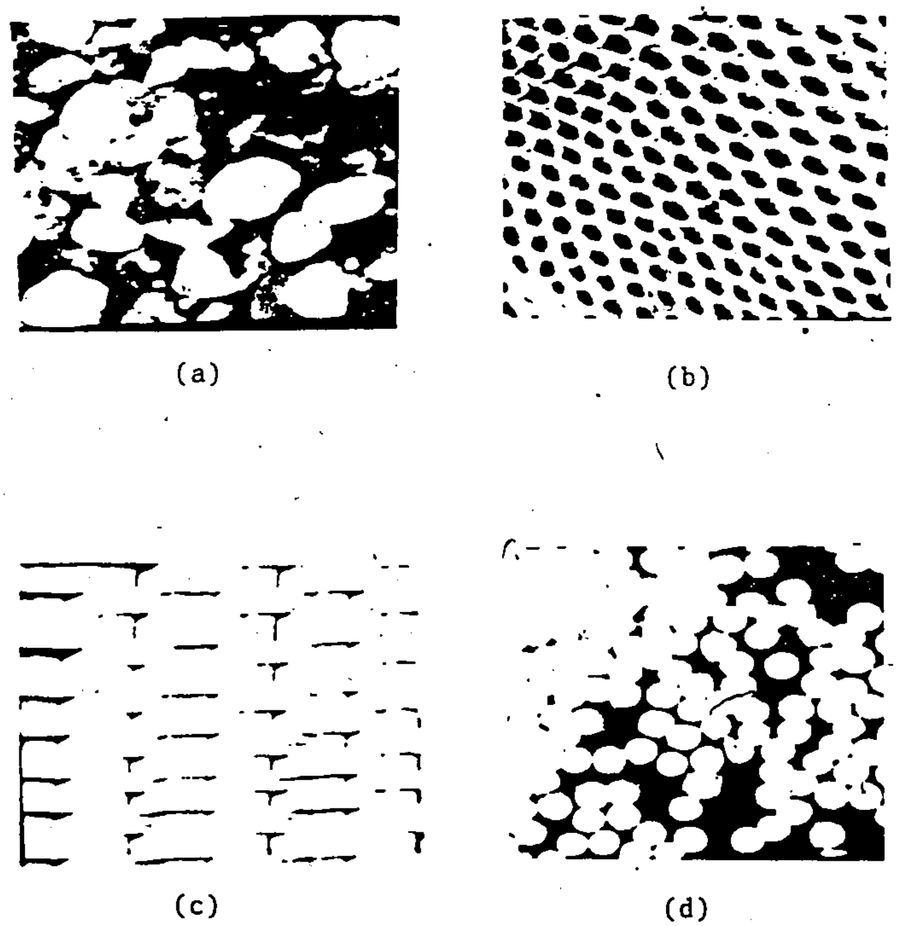


Fig. 1 An Aerial Photograph



(a) (b) (c) (d)

Fig. 2 Textures (a) Beach Pebbles, (b) Lizard Skin, (c) Handwoven Oriental Rattan; (d) Plastic Pellets

2. Texture Discrimination

3. Segmentation of a Given Textured Image

1.1 Texture Description:

The problem of texture description can be defined as:Given a textured region, how can it be represented?

In this case one usually assumes the given sample to be texturally homogeneous (a single texture element). According to Shen and Wong, 1980, an ideal texture representation should enhance various aspects of image textures at different resolution levels and should allow for small or no variations across different samples taken from the same texture.

1.2 Texture Discrimination:

Here we define the problem as: Given a textured region, to which of a finite number of classes does the region belong?

The main objective is to develop an effective procedure so that classification of different classes of textures can be done by automation. The procedure should be as general as possible so that one does not have to develop a new procedure or rule each time a new texture class is defined.

1.3 Segmentation of a Given Textured Image:

The final problem can be defined as: Given a textured image or a scene, how can it be segmented into different texturally homogeneous regions?

This is the most difficult problem for a number of reasons. For example, textural properties may undergo slow spatial variations that are difficult to detect or quantify. Several techniques that use image texture information have been proposed. Frequently used texturally based approaches to image segmentation are:

1. Region Growing
2. Region Merging
3. Region Splitting

Region growing and region merging are processes in which adjacent regions having similar textures are combined or merged. Region splitting is a process where the detection of dissimilarity of texture among two or more neighbourhoods within a region results in splitting the region into regions of greater homogeneity.

A brief discussion on texture analysis algorithms is given in Chapter II. This chapter also covers a critical review of some of the existing segmentation techniques.

The problem of texture description is covered in Chapter III. Texture can be considered as a global pattern arising from the repetition of one or more local subpatterns. So in order to characterize a given texture, it is important to consider both local and global aspects. The importance of multi-resolution levels for an ideal texture representation is reported in the first section of this chapter. Then the local features (the gray level feature and the gradient feature) are extracted at different resolution levels using a feature extraction scheme proposed by Raafat, 1985. The spatial organization of these features is given by the gray level histogram and the gradient histogram. So, a texture block can be described by a set of histograms of various primitive features at different resolution levels. An experiment is demonstrated at the end of this chapter to show little or no variations in the shape of histograms if a sample is taken from a homogeneous textured image.

The problem of texture discrimination or classification (Given a texture sample, to which of a finite number of classes does the sample belong?) is addressed in Chapter IV. Texture within an image block is characterized by its gray level and its gradient histograms. Thus in order to compare two blocks, effective methods are needed to compare histograms for a specific feature at different resolution levels. A traditional approach of comparing two histograms is the symmetric difference method. But this method does not take into account the event distance (i.e. the distance between the events of the two histograms), which is important in order to evaluate the exact similarity between two histograms (Shen and Wong, 1980). The texture distance between two image blocks is calculated by treating it as a transportation problem that can be solved by the simplex method. This method is then compared with the gray level dependence method (Haralick et al., 1973).

In Chapter V, a computationally simple and efficient technique for segmentation of a given textured image is proposed. This method uses texture as the basic information for segmentation. The image is first divided into cells or blocks of equal size. Each block is then represented by its gradient feature (i.e. gradient vector histogram). Next, the gradient vector histogram of the first image block is compared to its neighbouring blocks by applying an appropriate histogram comparison technique (Chapter IV). All neighbouring blocks that meet the similarity criteria become part of the region being grown. The neighbours of the newly accepted blocks are then examined for further growth. The process is repeated recursively 'till no more blocks can be added to the region under consideration. Then a new region is grown around a block which has not been previously selected. When all the blocks have been assigned appropriate labels, the process terminates.

Chapter VI, presents a brief summary of the work done and the future research possible.

CHAPTER II

A REVIEW OF TEXTURE ANALYSIS SCHEMES

Texture is a regional property of an image which refers to the spatial distribution of intensity variations. In many cases a picture can be considered as a composition of uniformly textured regions, or as containing objects on a background, where objects and background differ in texture. Thus texture features, measuring specific texture characteristics, are often of importance for picture description and analysis. It is generally agreed that texture plays a fundamental role in classifying objects and outlining regions. A critical review of the various approaches to texture analysis is given in Haralick, 1979. A brief description of the frequently used techniques is given in the next section.

Generally texture analysis techniques fall into the following two categories:

1. Statistical Techniques
2. Modelling Techniques

2.1 Statistical Techniques:

Some of the various statistical methods which are frequently used to analyse a given textured image are given below. For more detailed algorithms see Haralick, 1979, Weszka et al., 1976; and Conners and Harlow, 1980.

1. The Spatial Gray Level Dependence Method
2. The Gray Level Run Length Method
3. The Gray Level Difference Method
4. The Power Spectral Method

2.1.1 The Spatial Gray Level Dependence Method:

The spatial gray level dependence approach computes an intermediate matrix of measures from the digitized image data, and then defines features as functions on this intermediate matrix. Given a picture function F with a set of discrete gray level values G; an intermediate matrix or gray level co-occurrence matrix is computed by estimating the probability $P(i,j)$ of a pair of gray levels (i,j) occurring at a separation 'd' and an angle 'θ'. A set of statistical features proposed by Haralick et al., 1973 (Appendix A) can be used to extract textural information from these matrices. Weszka et al., 1976 used some of these features for terrain classifications.

2.1.2 The Gray Level Run Length Method:

A gray level run is a set of consecutive, collinear picture points having the same gray level value. This method is based on an estimation of the number of times an image contains a run of length 'j' for gray level 'i' in a given direction 'θ', where the length of the run is the number of picture elements within the run. From these matrices, statistical features like gray level distribution, run length distribution, short run emphasis etc. (Appendix A) are extracted which represent the underlying texture in the given image.

2.1.3. The Gray Level Difference Method:

The gray level difference method is based on computing the first order statistics of local property values. Consider a digital image $F(i,j)$. For any given displacement 'd'$=(d_1,d_2)$, where d_1 and d_2 are integers, let $G_d(i,j)$ be the difference of gray levels of two points separated by the distance 'd' i.e.

$$G_d(i,j) = | F(i,j) - F(i+d_1, j+d_2) |$$

Then the gray level difference density function $D(k|d)$, associated with the possible values of G can be defined as:

$$D(k|d) = P(G_d(i,j) = k)$$

Commonly used values of 'd' are $(0,s)$, $(-s,0)$, $(s,0)$, or $(-s,s)$, where 's' is the inter-sample spacing distance. From each of these density functions, a set of statistical features such as contrast, angular second-moment, entropy, mean, and inverse difference moment (refer to Appendix A) are computed to characterize a given textured image.

2.1.4 The Power Spectral Method:

The power spectral method is based on computing the two-dimensional Fourier transform of a given image. The power spectrum (magnitude squared of the Fourier transform) of the given image is then evaluated. Thus if a texture is at all periodic or directional, the power spectrum tends to have peaks for corresponding spatial frequencies. Then radial and angular features (Appendix A) are measured using the power spectrum.

Weszka et al., 1976 performed a comparative study to determine the relative abilities of the spatial gray level dependence method, the gray level run length method, the gray level difference method and the power spectral method. Conners et al., 1980 also developed a theoretical evaluation methodology to compare the relative loss of texture context information on going from digital images to intermediate matrices (e.g. gray level co-occurrence matrices, gray level run length matrices, gray level density functions and power spectrum). They concluded that the gray level dependence method is the most powerful method of the above four. However the power of this method depends on how many and which values are used for inter-sample spacing distance 'd'. Multiple values of 'd' and 'θ' must be used to completely embody the whole information about the spatial distribution of intensity variations in some texture images. It was also found that the gray level difference method was more powerful than the power spectral method, and the gray level run length method was a poor method for texture analysis.

The above methods work well if an image of uniform texture is provided. However if an image contains several regions of different textures, it would not be meaningful to compute these matrices for the entire image (Niemann, 1979). For such images, the first step will be to segment them into texturally homogeneous regions. Then these matrices can be computed for uniformly textured regions.

2.2 Modelling Techniques:

In this approach, we have to define models which discriminate or characterize a variety of textures. Most of the existing texture models use the statistical approach. There are two broad classes of texture models:

1. Statistical Models
2. Structural Models

2.2.1 Statistical Models:

Statistical models are based upon an underlying generative process. The set of parameters which characterize the models are used as features required for texture discrimination. By studying the statistical properties of the given textured image (e.g. auto-correlation function) McCormick et al., 1974 made a best fitting time series analysis model. These models are useful for texture synthesis purposes.

In a theoretical study done by Conners and Harlow, Markov random fields were used to compare different texture analysis approaches. Although statistical models have shown some success in discriminating sets of textures, they are less powerful than the structural models that use probabilistic subpattern selection and placement.

2.2.2 Structural Models:

Structural models are based on the notion that texture is composed of texture primitives and the spatial arrangement of these primitives. Serra et al., 1973 proposed a model that views a binary texture as produced by a set of translations of a structural element. Textural properties can be obtained by appropriately parameterizing the structural elements. Serra et al., 1973 also pointed out that generalized covarience functions can be used to obtain various texture features.

Zucker, 1976 proposed a model in which he viewed a real texture as being a distortion of an ideal texture. Certain transformations were applied to the primitives to distort them to provide a realistic texture. Carlucci, 1972 used primitives such as line segments and open or close polygons. Lu et al., 1978 developed a grammar syntactic approach to texture analysis. Actually there is no unique grammar for any given texture. In fact there are an infinite set of choices for rules and symbols. Few of the various grammars used for grammatical models are shape grammar, tree grammar and array grammar.

One of the main objectives of texture analysis problems is to segment a given image into texturally homogeneous regions (regions having a uniform texture). Image segmentation is the division of an image into different regions each having certain properties. A detailed survey of image segmentation is given in Fu and Mui, 1981. It was observed that almost all image segmentation techniques are application dependent. The reason for not having a general purpose image segmentation algorithm is that a two-dimensional image function can represent a potentially infinite number of possibilities. Segmentation techniques can fall into one of the two general categories:

1. Point Dependent Techniques
2. Region Dependent Techniques

A summary of some of the techniques is provided in the next section.

Image thresholding is one of the simplest and reasonably effective techniques. It thresholds the given image into different regions according to some appropriately chosen threshold values (e.g. if an image has a bimodal gray level histogram where one mode represents the object and the other one represents the background, then the valley between these modes can be used as a threshold value). This method works reasonably well when an appropriate threshold selection is possible. But it is difficult to apply to the situations where an object consists of textures made up of pixels of various gray levels.

Muerle and Allen, 1968 proposed another technique which uses the statistical properties of an image block or image subregion. Here, the image is first divided into blocks of size 2x2 or 4x4 and so on. The first block is labelled as '1'. The statistical features of this block are then compared with the statistical features of the neighbouring blocks. If they are similar then the blocks are merged together and a new set of statistics are computed. If no more blocks can be merged then the next unlabelled block is given label '2' and the procedure is repeated until all the blocks are labelled. In this method, the image is segmented into regions of homogeneous gray level statistics.

In another approach given by Niemann, 1979, one starts with the entire image and splits it into homogeneous regions, where a region is homogeneous if the mean gray level of any one of its subregions is equal to the mean gray level of the region.

Most of the region growing techniques are strictly based on image properties such as gray level, border strength, and region shape. So basically the merging criteria is influenced by the interpretation of the region.

Muerle and Allen used a regional neighbour search method to merge regions of similar properties. Pavlidis, 1972 partitioned the image into a collection of one dimensional strips. These strips were approximated by a linear combinaton of some known simple functions. Then the approximated functions were merged together if the approximating coefficients were close enough.

Feldman et al., 1973 proposed another method based on semantic interpretation for region merging. They emphasized maximizing the probability that all regions and borders are correctly interpreted.

Gupta et al., 1974 used a minimum distance classifier in which the initial segementation was done by usirg the scheme proposed by Muerle and Allen. Then each initial region was interpreted as one of a small predetermined number of different classes such as water, lizard skin, wood, etc. Neighbouring regions were then merged on the basis of their class membership.

A clustering method can be used to group the points in the characteristic feature space into clusters. These clusters are then mapped back to the spatial domain to produce the segmentation of an image. The characteristic features that may be used are gray level; texture measures etc. It is desirable to use two or more features to perform image segmentation because there may be problems which cannot be resolved using one feature. Swain et al., 1968, Haralick et al., 1975, Schachter et al., 1979, and Aggarwal et al., 1977 used clustering techniques for image segmentation.

The method proposed in this thesis uses texture as the basic information for segentation. The image is first partitioned into blocks or subregions of equal size. Each block is represented by its local features and global features. The global features of each block are compared with the global features of its neighbouring blocks using a distance measure technique proposed by Baafat, 1985. The regions are then merged according to a thresholding criteria. A detailed explanation of this method is given in chapter V.

CHAPTER III

TEXTURE DESCRIPTION

Texture can be described as being generated by one or more basic local patterns which are replicated over an image region. Texture has generally been represented through the qualitative texture descriptors such as coarseness and directionality. Coarseness is one of the important texture features which depicts the size of the texture elements. For example, if there are two textured images which differ only in scale, then the magnified one is coarser. Also, the larger the element size and/or the less its elements are repeated, the coarser it is. Directionality on the other hand involves the shape of the texture elements and their placement rule.

According to Haralick, 1979, texture can be viewed as a two layer process. The first layer is for describing the tonal primitives or local properties which form the image texture and the second layer is for describing the spatial relationship of these tonal primitives. Tonal primitives are regions with tonal properties and are described in terms of average gray level. Consequently, for texture characterization, the tonal primitives or the local properties as well as the spatial relationship between these tonal primitives must be taken into consideration.

A survey of most of the approaches for texture characterization is given in Haralick, 1979. Most of these schemes tend to emphasize one or the other aspect, but not both. The method used in this thesis for texture description is the resolution dependent texture representation technique proposed by Raafat, 1985, and it considers both the local and the global aspects of texture. The concept of resolution dependency for an ideal texture representation was devised by Wong and Shen, 1980. According to this scheme an ideal texture representation should be schematically general. It should also allow small variations across different samples taken from the same texture and it should contain all essential texture characteristics with little redundant information. The details of this method are given in the next section.

According to Wong et al., 1980 and Rosenfeld, 1984, an ideal texture representation should provide information at multiple resolution levels. Resolution is related to the ability to discriminate fine details in the field of view. A multiple resolution scheme is important because a gray level histogram may remain unchanged under the rearrangement of pixels at resolution level one (original image). However when higher resolution levels or low resolution images are used, these histograms will be different. The resolution levels are varied by using observation windows of different sizes. The resolution can be reduced within a given observation window by replacing the gray level of each pixel with the average gray level of that pixel and its neighbours.

The primitive texture features chosen are:

1. The Gray Level Feature
2. The Gradient Feature

Both features are used to represent a texture at different resolution levels. The choice of these features were made on the basis of a study done by Baafat, 1985. If resolution is taken into account, the most primitive features to be observed in a window are the gray level feature and the gradient feature, where gradient is a two-dimensional vector (gradient magnitude and direction). Some of the important definitions concerning the feature extraction scheme are given below.

3.1 Definitions:

3.1.1 Observation Window:

According to Wong and Vogel, 1977, an observation window W_m can be defined as a square, of dimension $S(m) \times S(m)$ pixels, such that $S(m) > S(m-1)$ where 'm' represents a specific resolution level.

In order to enable the centre of the window to be placed on a pixel, the number of pixels in $S(m)$ should be an odd number.

$$S(m) = \beta^{m-1}$$

where β (>1) is the basic window size parameter.
So if $\beta = 3$ and $m = 1$ then window size = 1 x 1;
if $\beta = 3$ and $m = 2$ then window size = 3 x 3;
and so on.

3.1.2 Neighbourhood:

Let $X_m(i,j)$ be the neighbourhood specified by W_m with centre positioned at (i,j) for resolution level 'm'.

3.1.3 Feature Observation Scheme:

Let $E_t = \{ k \mid k=0, 1, 2, \ldots, K_t - 1 \}$ be a set of discretized values, where 't' represents a specific texture feature and k_t is the total number of events in the set. The feature extraction scheme is then defined as the mapping of the feature contents inside the neighbournocd onto the set E_t.

3.2 Feature Extraction:

3.2.1 Gray Level Feature Extraction:

The images have been obtained by directly photographing them from the pictures in Brodatz, 1966. Lower resolution level images were obtained by replacing the gray level of each pixel within an observation window with the average gray level of that pixel and its neighbours.

Let $E_g = \{ k \mid k=0, 1, 2, \ldots, K_g-1 \}$

where K_g is the number of discretized gray levels.

The gray level feature extraction is thus defined as the mapping of the gray level values at different resolution levels onto the set E_g.

3.2.2 Gradient Feature Extraction:

The gradient feature at resolution level 'm' is calculated by applying a Sobel operator to a 3 x 3 image region. Consider a 3 x 3 image region as shown in Figure 3, where a,b,c,...,i represent the gray level values. By applying the Sobel operator (Fig. 4) to this image region, the gradient in the x-direction can be defined as:

$$G_x(i,j) = (c + 2f + i) - (a + 2d + g)$$

and the gradient in the y-direction can be defined as:

$$G_y(i,j) = (a + 2b + c) - (g + 2h + i)$$

From this, the gradient magnitude G(i,j) at point 'e' can be given as:

$$G(i,j) = \sqrt{(G_x^2 (i,j) + G_y^2 (i,j))}$$

$$\Theta(i,j) = \tan^{-1} (G_y/G_x)$$

For computer implementation of the gradient magnitude, the following first order approximation is computationally more efficient:

$$|G(i,j)| \simeq |G_x(i,j)| + |G_y(i,j)|$$

A two-dimensional discretization scheme given in fig. 5, has been used to represent the two-dimensional gradient vector (gradient magnitude and gradient directionality). The gradient magnitude is discretized into five intervals, whereas the gradient directionality is discretized into eight directions. It is assumed that the central event bears no directionality. Thus if the gradient is less than '5', irrespective of the direction, it will be mapped onto event '0' (Figure 5). Thus there are forty-one events in total. The outer ring has the maximum gradient magnitude possible.

Let $E_v = \{ k \mid k = 0, 1, 2, \ldots, K_v - 1 \}$
where K_v is the number of gradient events (=41).

Then gradient feature extraction can be defined as the mapping of the gradient magnitude and the directionality onto one of the events in set E_v. For example, If $10 \leq G(i,j) < 15$

and $\frac{\Pi}{8} \leq \Theta(i,j) < \frac{3\Pi}{8}$

then the mapped event will be '10'.

a	b	c
d	e	f
g	h	i

Figure 3 A 3 x 3 Image Region

-1	0	1
-2	0	2
-1	0	1

(a)

1	2	1
0	0	0
-1	-2	-1

(b)

Figure 4 Sobel Operator

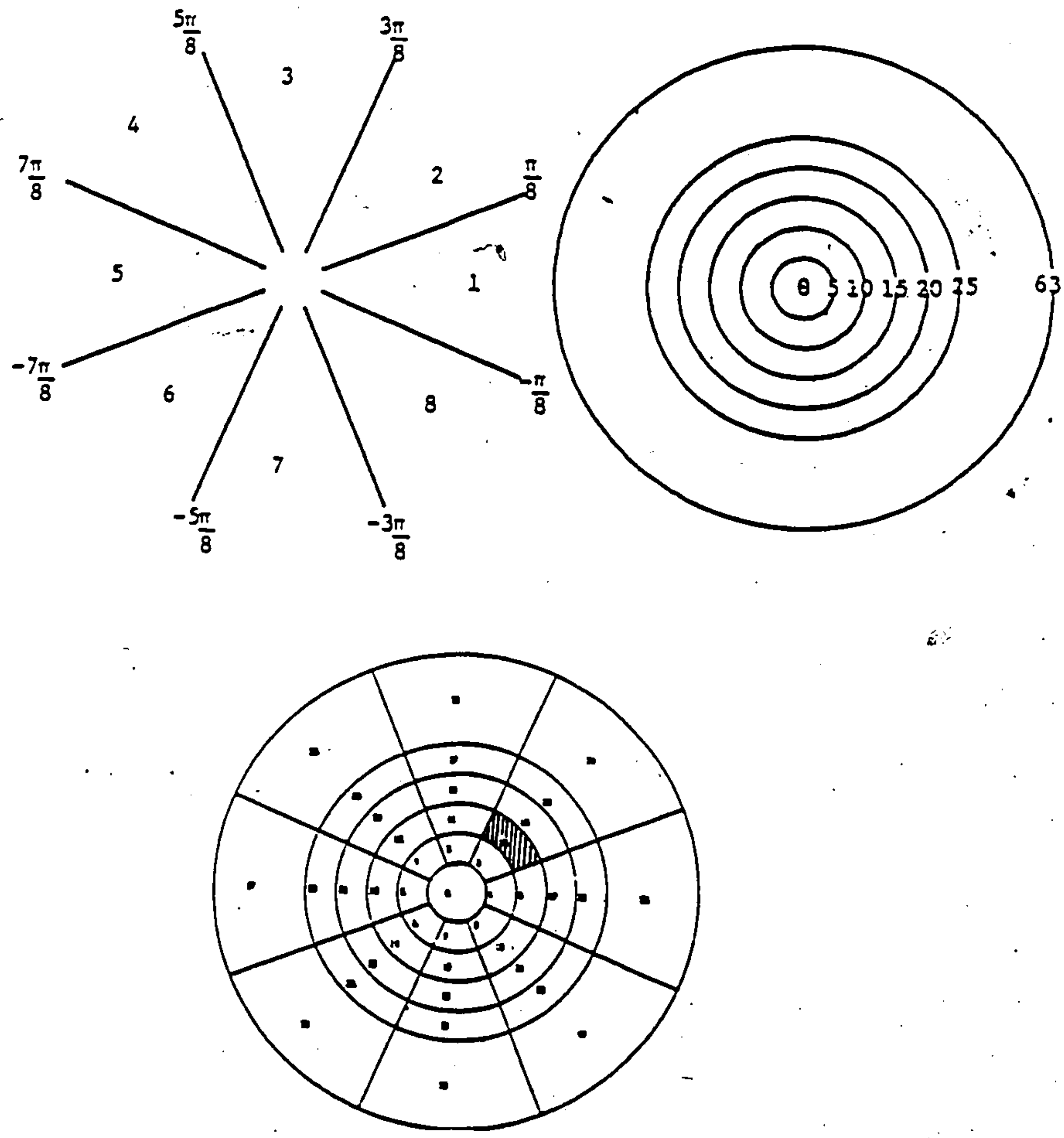


Fig. 5 A two-dimensional discretization scheme

3.3 Spatial Relationship of Primitive Features:

The histogram representation technique is used to describe the spatial relationship of the features used. A histogram is a set of values where each value represents the frequency of occurrence of a particular event. We define a gray level histogram for the gray level feature and a gradient vector histogram for the gradient feature.

3.3.1 Gray Level Histogram:

Consider an image block 'a' of size n x n. The gray level feature is extracted using the feature extraction scheme and is mapped to the set E_g. The gray level histogram over this image block can then be defined as:

$$H_g^a = \{ h(k) \mid k=0, 1, 2, ..., K_g-1 \}$$

where h(k) is the frequency of occurrence of event k.

3.3.2 Gradient Histogram:

For an image block 'a' of size n x n the gradient magnitude and directionality can be mapped onto the set E_v using the two dimensional discretization scheme. The gradient vector histogram is then defined as:

$$H_v^a = \{ h(k) \mid k=0, 1, 2, ..., K_v-1 \}$$

where h(k) is the frequency of occurrence of event k.

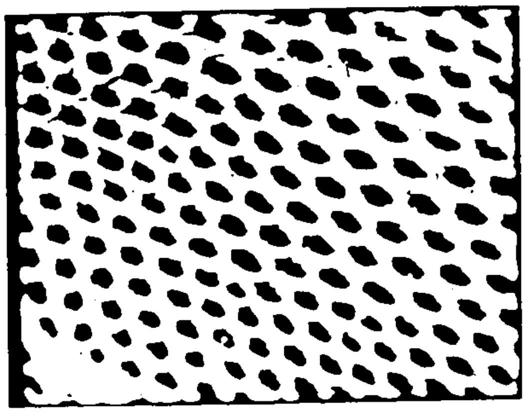
Thus given an image block 'a' of size n x n, texture representation can be defined as a set of histograms of various primitive texture features at different resolution levels. Also texture representation of a texturally homogeneous region is defined as a set of the mean of the histograms of various primitive features obtained from all image subregions covering that region at different resolution levels (Wong et al., 1980).

3.4 Results:

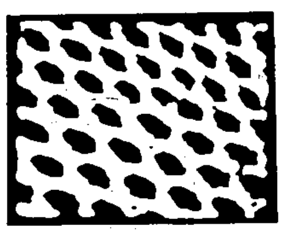
A set of digital image data is selected from Brodatz, 1966 to show how the histogram representation characterizes the underlying texture and also to show little or no variations across different samples taken from the same texture. The images selected for this experiment are lizard skin, and sand. Both images, of size 128x128, are digitized into 256 gray levels, ranging 0-255, '0' being the darkest. Lower resolution images or higher resolution levels can be obtained by replacing the gray level of each pixel within an observation window with the average gray level of that pixel and its neighbours.

Figures 6(a)-6(b) show the images of the lizard skin and its sample (64x64). The corresponding gray level histograms at different resolution levels, with the gray levels mapped to 32 values, are shown in figs. 6(c)-6(d). Figures 6(e)-6(f) show the corresponding gradient vector histograms at different resolution levels, with the gradient mapped to forty one events. Similarly figs. 7(a)-7(f) show the images and their corresponding histograms for the sand.

The gray level histogram shown in fig. 6(c) represents the black spots on the white background. It is noticed that the two peaks are more distinct at resolution level two. As the resolution decreases, the curve becomes more smooth with extremal points moving inward. Figure 6(d) clearly shows that there are very little variations if samples are taken from the same texture. The wide range of gray level values and the high values of the gradient are also noticed. As expected, when the resolution decreases, the gradient tends to shift toward the centre, although the shift is slow due to the wide range of gray level values (fig. 6(e)-6(f)). The general behaviour of the gradient vector histogram remained the same for its sample. The underlying characteristics of the textured image and slight variations if a sample is taken from a homogeneous texture, are also clear from figs. 7(a)-7(f).

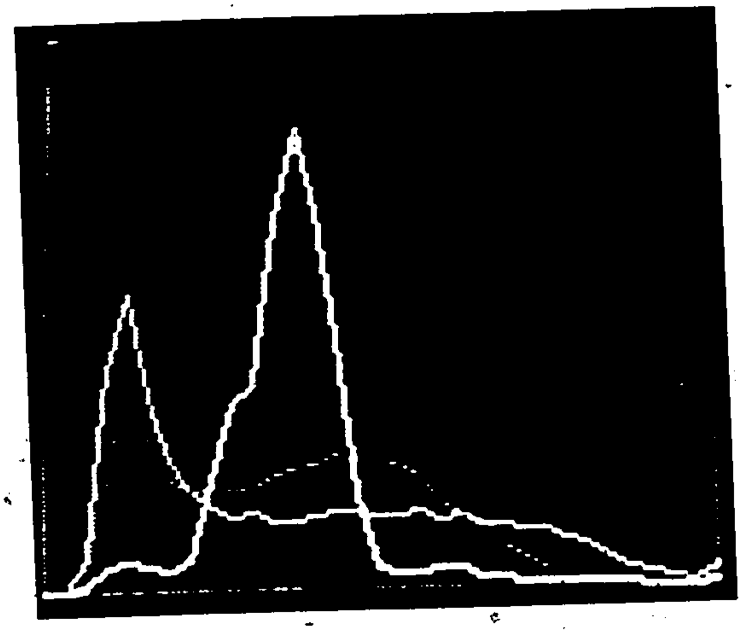


(a)

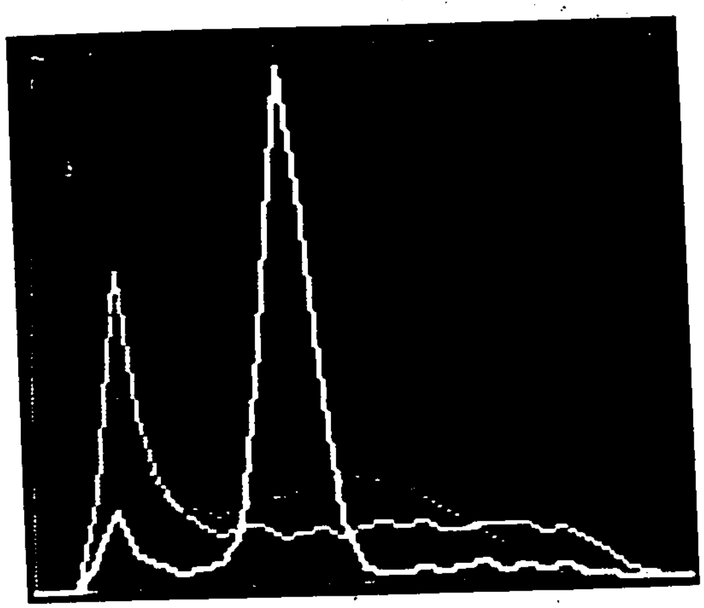


(b)

Fig. 6 (a) Lizard Skin; (b) Sample of (a) (64 x 64)



(c)



(d)

Fig. 6 (c) Gray level relative frequency histogram of Fig. 6 (a)

(d) Gray level relative frequency histogram of Fig. 6 (b)

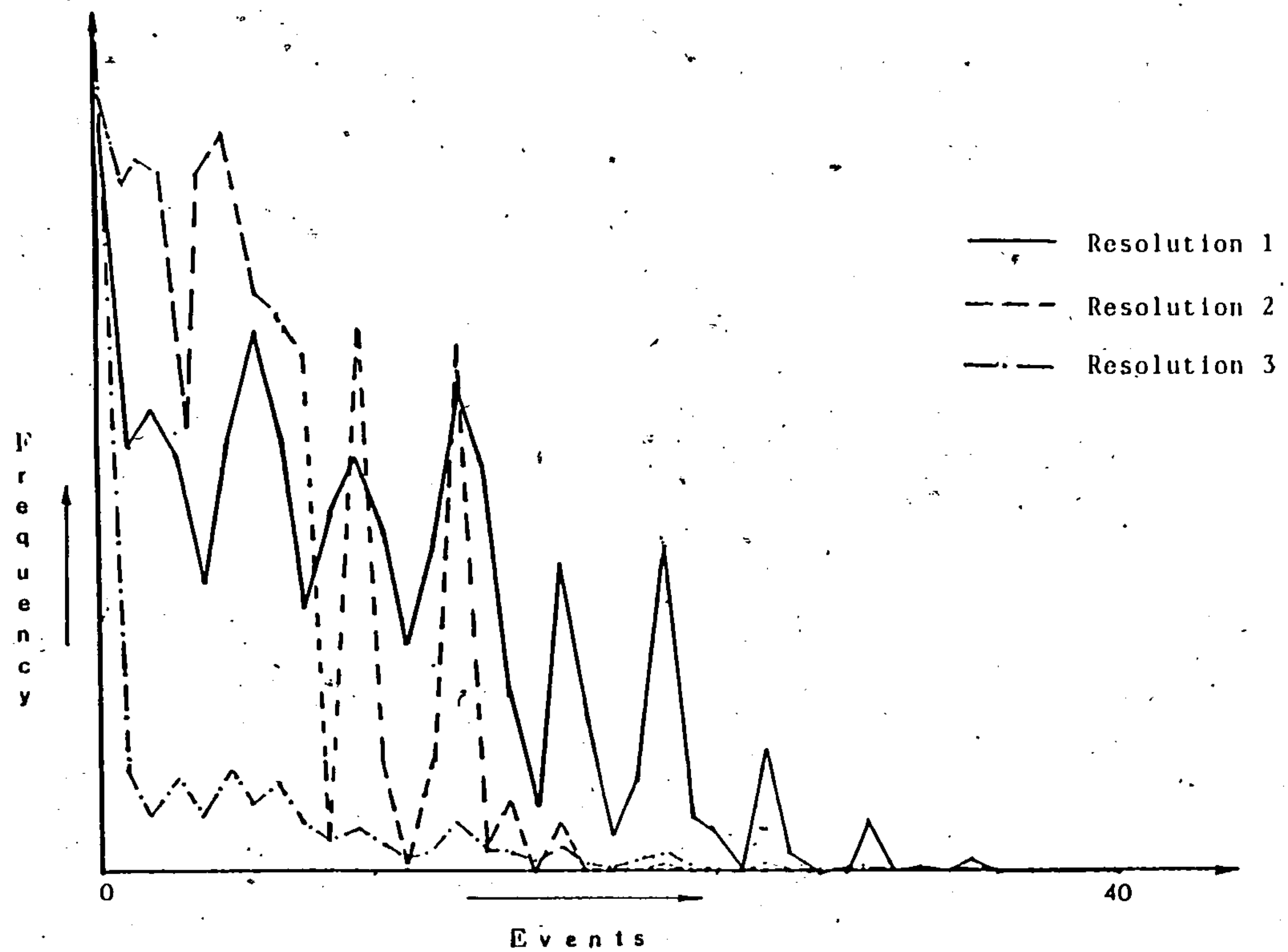


Fig. 6(e) Gradient vector histogram of Fig. 6(a)

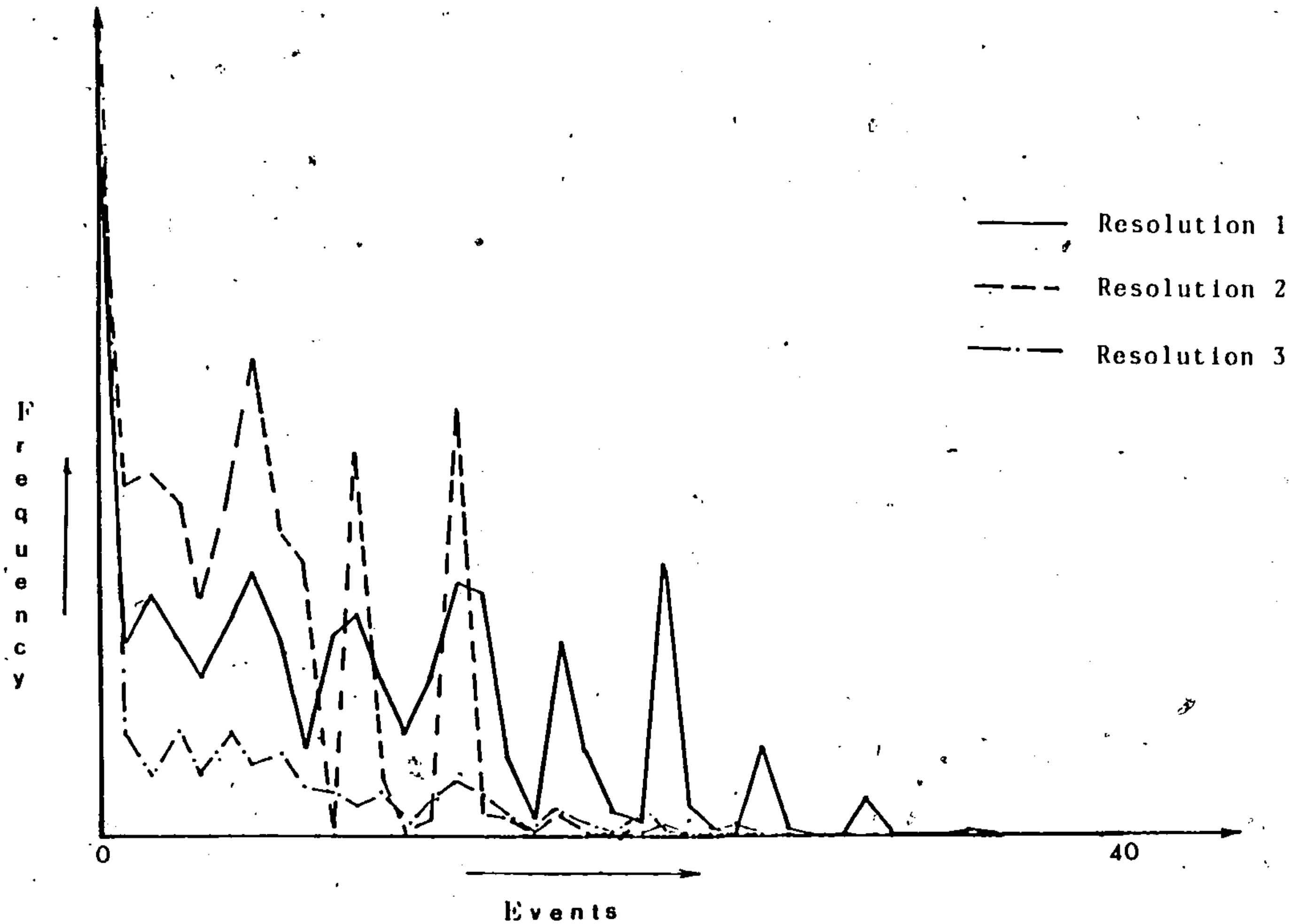
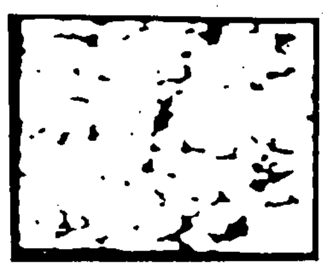


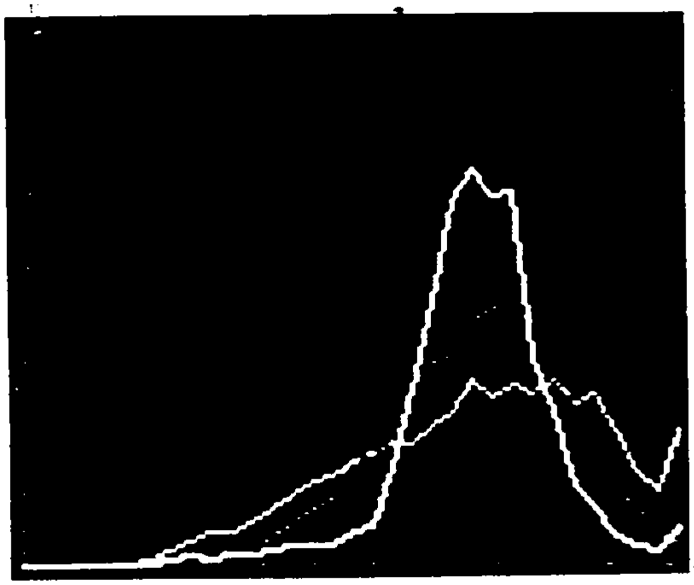
Fig. 6(f) Gradient Vector histogram of Fig. 6(b)

(a)

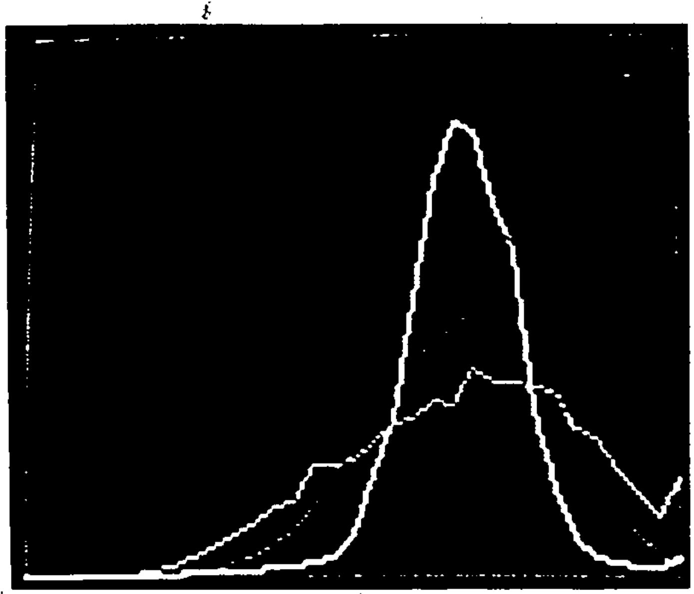


(b)

Fig. 7 (a) Sand; (b) Sample of (a) (64 x 64)



(c)



(d)

Fig.7 (c) Gray level relative frequency histogram of Fig.7 (a)

(d) Gray level relative frequency histogram of Fig.7 (b)

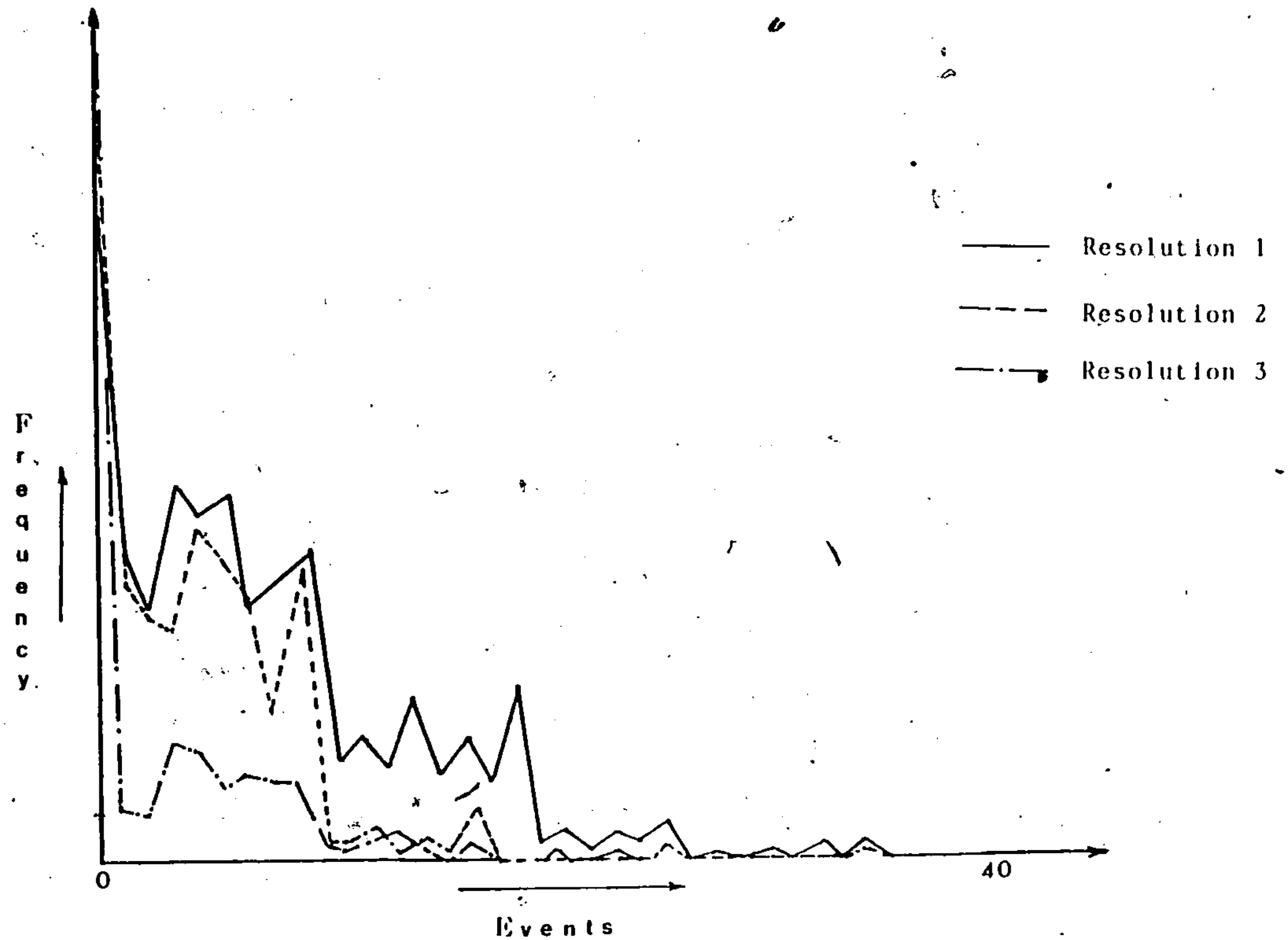


Fig. 7(e) Gradient vector histogram of Fig. 7(a)

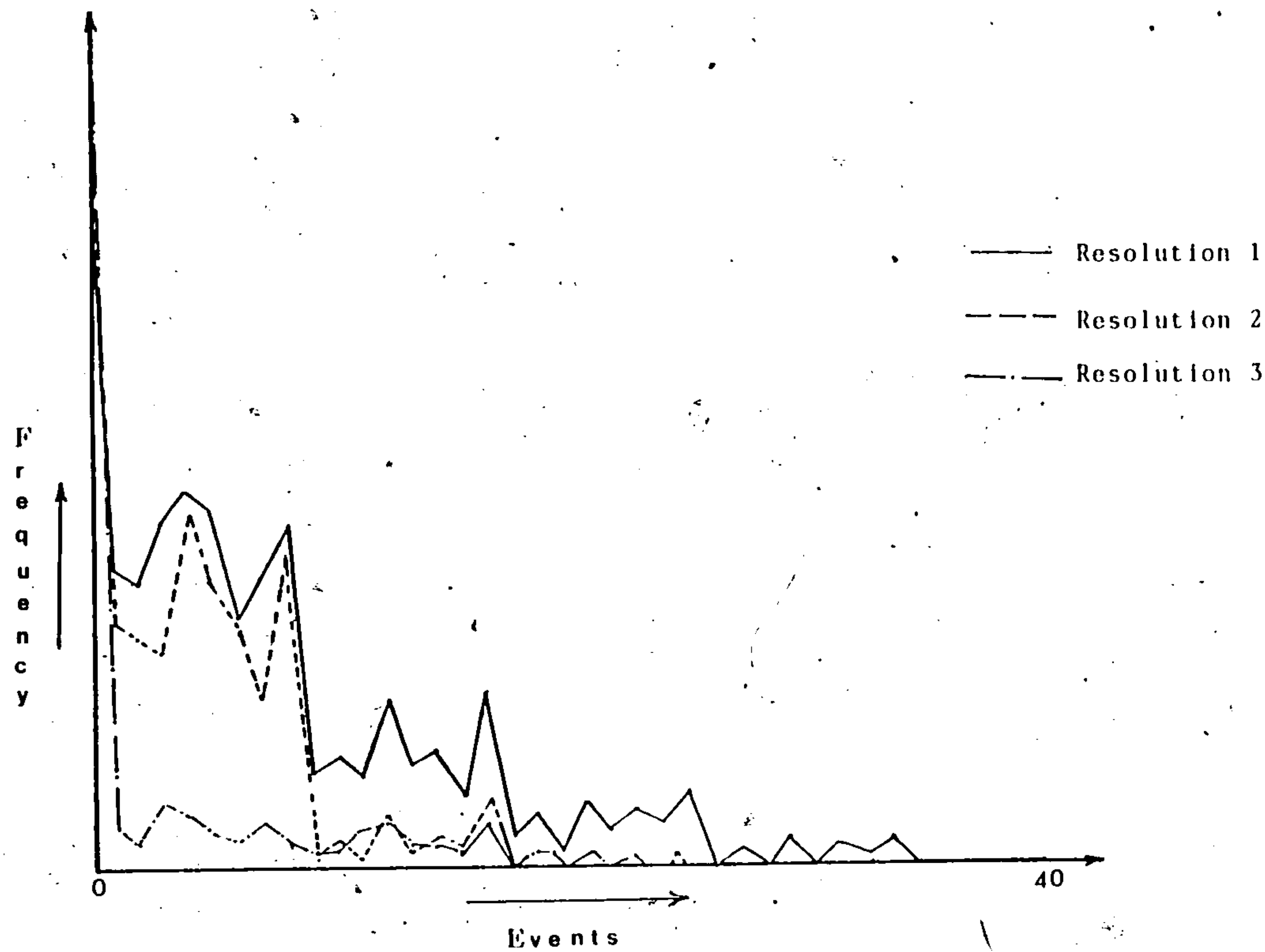


Fig. 7(f) Gradient vector histogram of Fig. 7(b)

CHAPTER IV

TEXTURE DISCRIMINATION

Texture discrimination or classification is an important aspect of many image analysis problems. In this type of problem, an image is known to contain data from one of a finite number of texture classes, and the aim is to find out the proper texture class to which that image belongs. In this chapter, a critical analysis of a texture classification technique proposed by Raafat, 1985 is presented. The first section discusses the resolution dependent texture classification technique (Raafat, 1985); the second section of this chapter describes the gray level dependence method (Haralick et al., 1973). And then the performance of both methods are compared.

4.1 Resolution Dependent Texture Classification:

In Chapter III, it was shown that texture within an image block can be represented by its histograms (the gray level histogram and the gradient histogram). So in order to compute the similarity between two image blocks, we need to compare the two histograms for a specific feature.

The commonly used method for comparing histograms is to compute the area of symmetric difference between the two histograms and this is defined as the sum of the absolute values of the difference in the frequencies between corresponding events.

$$\text{Area} = | H^a - H^b |$$

$$= \text{Difference between histograms } H^a \text{ and } H^b$$

According to Wong et al., 1980, this method fails in certain cases because it does not take into consideration the distance effect between the events. For example, consider the following three gray level histograms:

$$H^a = \{ 5, 0, 0, 0, 0, 0, 0 \}$$
$$H^b = \{ 0, 5, 0, 0, 0, 0, 0 \}$$
$$H^c = \{ 0, 0, 0, 0, 0, 0, 5 \}$$

The gray level contrast between any two of the above histograms is the same according to the symmetric difference method, although it should not be. So it is important to consider the distance effect between events in order to compare two histograms. The method proposed by Raafat, 1985 provides an effective way for texture classification or discrimination and takes the event distances into consideration. A formal exposure to this technique is given below.

4.1.1 Problem Formulation:

If H_t^a and H_t^b are the two histograms representing the distribution of feature events within image blocks 'a' and 'b' respectively, then for a given feature 't' and a resolution level 'm', find a texture distance between them.

Definition 4.1: Let H_t^a and H_t^b be the two histograms representing the feature distribution within image blocks a and b, for a specific feature 't', and are defined as:

$$H_t^a = \{ h_p^a \mid p = 0, 1, 2, 3, \ldots, K_t - 1 \}$$

and

$$H_t^b = \{ h_q^b \mid q = 0, 1, 2, 3, \ldots, K_t - 1 \}$$

where h_p^a is the frequency of occurrence of event p in image block 'a' and h_q^b is the frequency of occurrence of event q in image block 'b' for a specific feature 't'.

Definition 4.2: The event distance set C between any two events p and q for a specific feature 't' is defined as:

$$C_t = \{C_t(p,q) \mid p = 0, 1, 2, \ldots, K_t - 1, \; q = 0, 1, 2, \ldots, K_t - 1\}$$

where $C_t(p,q)$ is the distance function between p and q.

For the gray level feature the distance function between p and q can be given by computing the absolute difference of the two events.

$$C_g(p,q) = |p - q| \quad \forall \quad p,q$$

For the gradient feature, the distance function can be computed by finding the radial distance and the angular distance. In Chapter III, it was assumed that the central event bears no direction. So if one or both of the events is 'C' that is the central event, then only the radial components of the distance between two events will be taken into consideration.

The radial distance between two events p and q is given by their absolute difference.

The angular distance between events p and q is given by their smallest angular separation. Thus the event distance between two events p and q for the gradient vector feature is defined as:

If $p \neq 0$ and $q = 0$

then $C_v(p,q) = K \bullet R_p$.

If $p = 0$ and $q \neq C$

then $C_v(p,q) = K \bullet R_q$.

If $p = 0$ and $q = C$

then $C(p,q) = 0$.

else $C_v(p,q) = K \bullet |R_p - R_q| + \min(|A_p - A_q|, N - |A_p - A_q|)$.

where K is the Normalizing Factor (=0.5 in this thesis);

R_p is the Discrete Radial Value for event p;

R_q is the Discrete Radial Value for event q;

A_p is the Discrete Angular Value for p;

A_q is the Discrete Angular Value for q;

and N is the number of distinct directions.

The event distance set is illustrated in figure 8.

Definition 4.3: Let $d_t(a,b)$ be the distance between two histograms H_t^a and H_t^b and be defined as:

$$d_t(a,b) = \sum_p \sum_q C_t(p,q) * X(p,q)$$

where $C_t(p,q)$ is the event distance set and

$X(p,q)$ are the weights associated with the events p and q.

Definition 4.4: The texture distance between the two histograms H_t^a and H_t^b at a resolution level 'm' can then be defined as the cost of minimal matching between the two histograms. Mathematically,

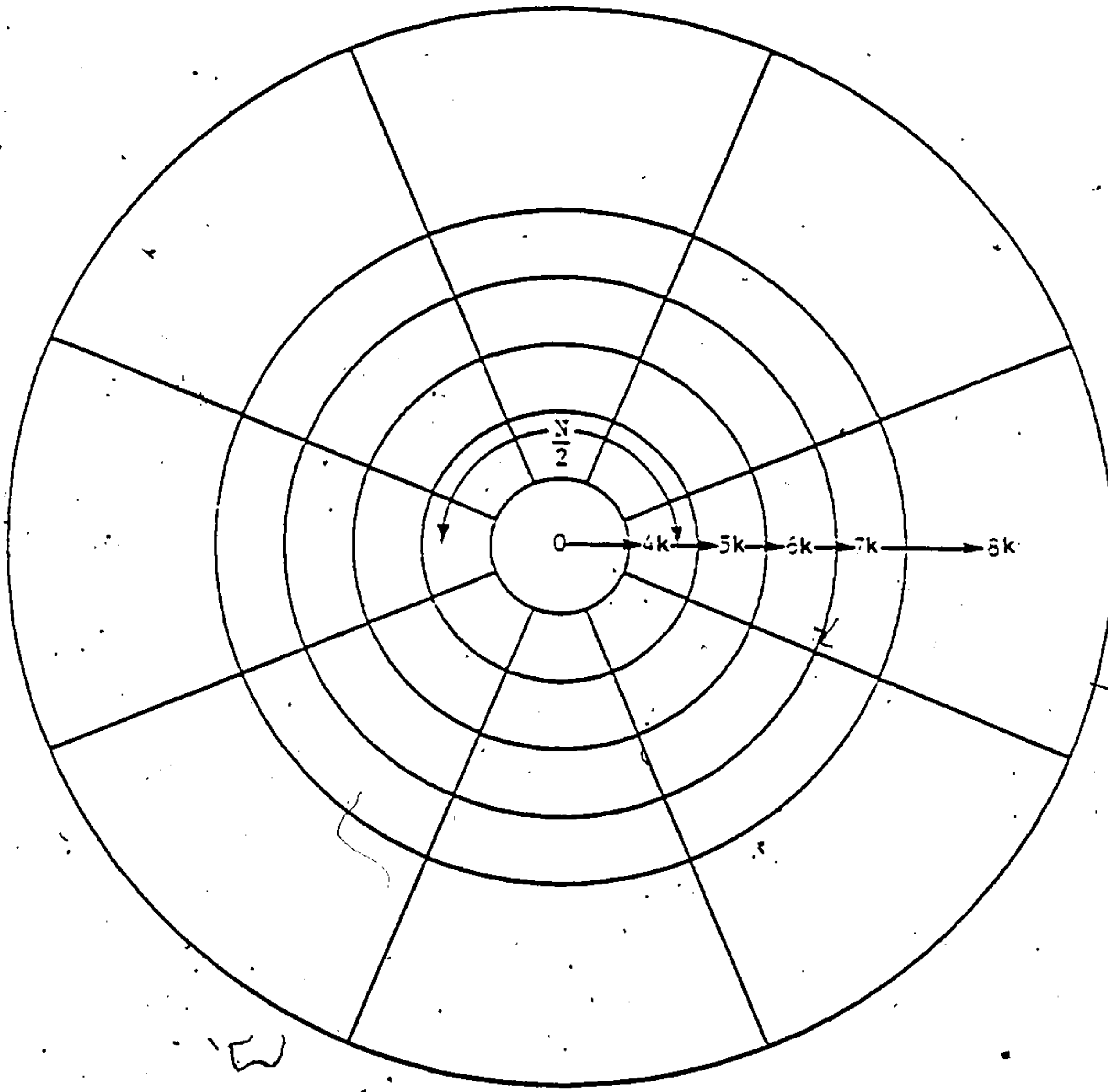


Fig. 8 Event Distance Set

$$\text{Minimize } d_t(a,b) = \sum_{p=0}^{K_t - 1} \sum_{q=0}^{K_t - 1} C_t(p,q) \cdot X(p,q)$$

with respect to $X(p,q)$ and subject to constraints:

$$(1) \quad \sum_{q=0}^{K_t - 1} X(p,q) = h_p^a \quad | \ p=0, 1, 2, \ldots, K_t - 1$$

$$(2) \quad \sum_{p=0}^{K_t - 1} X(p,q) = h_q^b \quad | \ q=0, 1, 2, \ldots, K_t - 1$$

$$(3) \quad \text{and} \quad X(p,q) \geq 0 \ \forall \ p,q$$

According to Hiller and Lieberman 1980, any linear programming problem that fits this special formulation is of the transportation problem type regardless of its physical context.

Let us assume in general that there are m sources S_1, S_2, S_3,..., S_m with capacities a_1, a_2, a_3,..., a_m and n destinations D_1, D_2, D_3,..., D_n with requirements b_1, b_2, b_3,..., b_n respectively. The shipping cost from the ith source to the jth destination is C_{ij} and the amount shipped is X_{ij}. If the total capacity of all sources is equal to the total requirements of all destinations, then the aim is to find out the value of X_{ij} with i=1, 2, 3,..., m and j=1, 2, 3,...., n for the total shipping cost to be a minimum. So the objective function can be defined as:

$$f(X) = \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij} \cdot x_{ij}$$

The conditions to be imposed on the above objective function can be stated as:

(a) The total capacity is equal to the total requirement i.e. $\sum_i a_i = \sum_i b_i$.

(b) The individual capacity of each source must be utilized and the individual requirement of each destination must likewise be fully satisfied. From this condition it is clear that we have m capacity constraints and n requirement constraints. The capacity constraints impose on the solution the condition that the total shipment to all destinations from any source must be equal to the capacity of that source i.e.

$$X_{i1} + X_{i2} + \ldots + X_{in} = a_i \quad | \; i=1, 2, 3, \ldots, m.$$

The requirement constraints require that the demand of every destination be fully satisfied by the total shipment from all sources i.e.

$$X_{1j} + X_{2j} + \ldots + X_{mj} = b_j \quad | \; j=1, 2, 3, \ldots, n.$$

(c) There is always another constraint called non-negativity constraint i.e, $X_{ij} \geq 0 \quad \forall \; i,j$

So finally the transportation problem can be summed up in the following formulation.

$$\text{Minimize} \quad f(X) = \sum_{i=1}^{m}\sum_{j=1}^{n} C_{ij} \cdot X_{ij}$$

subject to:

$$(1) \quad \sum_{j=1}^{n} X_{ij} = a_i \quad | \; i = 1, 2, 3, \ldots, m$$

$$(2) \quad \sum_{i=1}^{m} X_{ij} = b_j \quad | \; j = 1, 2, 3, \ldots, n$$

$$(3) \quad \text{and } X_{ij} \geq 0 \quad \forall \; i,j$$

The above model implies that the original problem will have a feasible solution only if the sum of the frquencies of the two histograms is equal, i.e.

$$\sum_{p} h_p^a = \sum_{q} h_q^b \, .$$

That is either blocks of the same size should be used or scaling of histograms should be done so that the total sum of frequencies of the histograms to be compared is equal. A general form of cost and the requirement table is given in Fig. 9.

Figure 10 shows the texture distance problem as a transportation problem, where the demand is represented by the frequencies of histogram H_t^a of image block 'a', and supply is represented by the frequencies of histogram H_t^b of image image block 'b'. The cost of a shipment is replaced by the event distance set C_t.

Definition 4.5: The total texture distance D(a,b), between two image blocks 'a' and 'b' at a resolution level 'm' is given as:

$$D(a,b) = \frac{(D_g(a,b) + W \cdot D_v(a,b))}{(1 + W)}$$

where

- g - represents the gray level feature;
- v - represents the gradient feature;
- W - is the weighting factor.

Source	Shipping cost per unit distributed					Capacity
	1	2	3	...	n	
1	C(1,1)	C(1,2)	C(1,3)	...	C(1,n)	A_1
2	C(2,1)	C(2,2)	C(2,3)	...	C(2,n)	A_2
3	C(3,1)	C(3,2)	C(3,3)	...	C(3,n)	A_3
⋮	⋮	⋮	⋮	⋮	⋮	⋮
m	C(m,1)	C(m,2)	C(m,3)	...	C(m,n)	A_m
Demand	B_1	B_2	B_3	...	B_n	

Figure 9 A General Form of Transportation Problem

Source (event)	Shipping cost per unit distributed 0	1	...	K_t'	Capacity H_t^a
0	$C(0,0)$	$C(0,1)$	...	$C(0,K_t')$	h_0^a
1	$C(1,0)$	$C(1,1)$	...	$C(1,K_t')$	h_1^a
2	$C(2,0)$	$C(2,1)$	...	$C(2,K_t')$	h_2^a
.	.	.	...	.	.
.	.	.	...	.	.
.	.	.	...	.	.
K_t'	$C(K_t',0)$	$C(K_t',1)$	...	$C(K_t',K_t')$	$h_{K_t}^a$
H_t^b	h_0^b	h_1^b	...	$h_{K_t}^b$	Demand

H_t^a - Histogram representing feature distribution of feature events within image block 'a'

H_t^b - Histogram representing feature distribution of feature events within image block 'b'

$C(m,n)$ - Distance function between events m in H_t^a & n in H_t^b

Figure 10 Problem Formulation

4.1.2 Experimental Results:

A set of four texture classes, namely, beach pebbles, handwoven oriental rattan, lizard skin and sand is chosen from Brodatz, 1966. Each texture is divided into different samples of size 64x64 pixels. Figure 11 shows the sample set selected for the classification test. Figure 12 shows the texture distance matrix whose (i,j)th entry gives the range of texture distances for the gradient vector feature between classes 'i' and 'j'. The gray level feature is not considered for this analysis because it does not contain sufficient information for discrimination. The low diagonal values in the matrix in Fig. 12 are due to the comparisons made between the samples from the same class. Also it is obvious from this figure that the gradient vector feature alone is sufficient for the discrimination purposes and it gives a classification rate of 100%. The classification rate is calculated as the average correct classification rate of all samples.

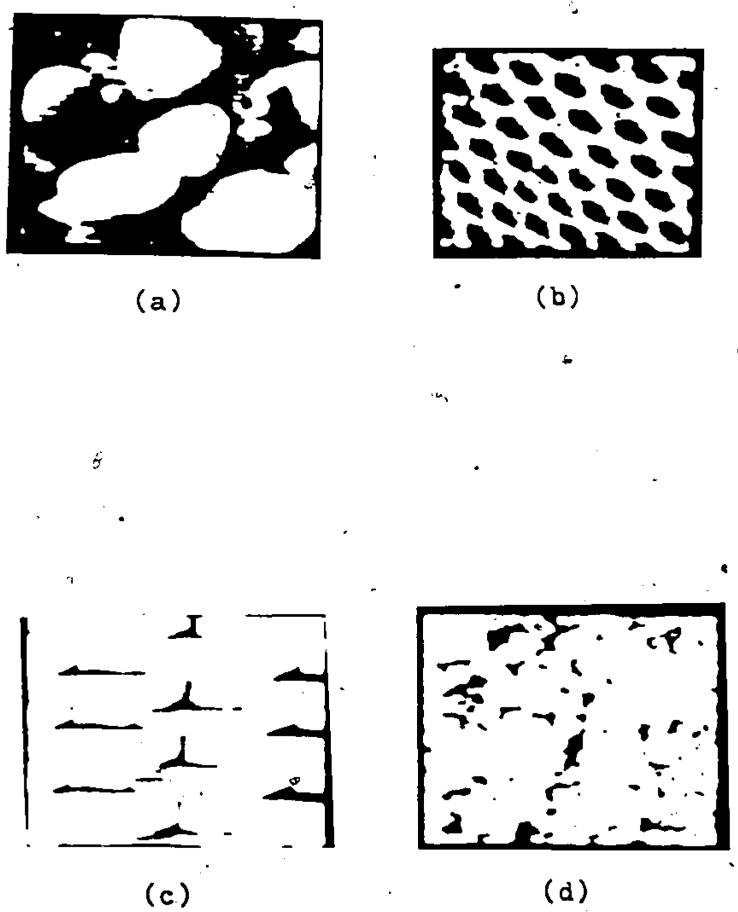


Fig.11 Textures (a) Beach Pebbles, (b) Lizard Skin
(c) Handwoven Oriental Rattan
(d) Sand

CLASSES

CLASSES	T1	T2	T3	T4
T1	0.236-0.683	2.531-2.994	3.328-2.834	2.413-2.974
T2		0.224-0.931	2.613-3.100	3.102-3.436
T3			0.102-0.279	2.930-3.864
T4				0.276-0.492

T1 - Beach Pebbles

T2 - Handoven Oriental Basket

T3 - Lizard Skin

T4 - Sand

Figure 12 Texture distance matrix (Range shown)

4.2 The Spatial Gray Level Dependence Method:

The spatial gray level co-occurrence matrices are one of the most powerful sources of feature extraction (Haralick et al., 1973) A gray level co-occurrence matrix is formed by specifying the probability $P(i,j)$ of a pair of gray levels (i,j) occurring at a separation 'd' and angle 'θ'. Given an image F, with a set of discrete gray levels G, we define for each, a set of discrete values of 'd' and 'θ', the gray level co-occurrence matrix $P(d,\theta)$ as follow:

$$P(d,\theta) = [P(i,j|d,\theta)]$$

where $P(i,j|d,\theta)$, an element of the matrix, is defined as the probability of going from gray level i to gray level j for a given inter-sample spaacing 'd' and direction 'θ'.

To illustrate the gray level co-occurrence matrices, consider a 4 x 4 image (Fig. 13(a)) with four gray level values, ranging 0-3. A general form of a spatial gray level dependence matrix is given in Figure 13(b). Fig. 14(a)-14(b) gives the four spatial gray level dependence matrices for 'd'=1 and 'θ' = 0^0, 45^0, 90^0 135^0. To determine the elements of these matrices we count the number of times gray levels 'i' and 'j' have been neighbours. For example the element in the (0,2) position of the horizontal $P(d,0^0)$ matrix is the total number of times two gray levels of value '0' and '2' occurred horizontally adjacent to each other.

A set of measures has been proposed by Haralick et al. 1973, which can be used to extract textural information from co-occurence matrices. The following are a few of the features used to characterize a given texture:

1. Energy

$$E(d,\Theta) = \sum_{i=0}^{Ng-1} \sum_{j=0}^{Ng-1} [P(i,j|d,\Theta)]^2$$

2. Entropy

$$En(d,\Theta) = \sum_{i=0}^{Ng-1} \sum_{j=0}^{Ng-1} P(i,j|d,\Theta) \log P(i,j|d,\Theta)$$

3. Local Homogeneity

$$Hm(d,\Theta) = \sum_{i=0}^{Ng-1} \sum_{j=0}^{Ng-1} \frac{1}{1 + (i-j)^2} P(i,j|d,\Theta)$$

4. Inertia

$$In(d,\Theta) = \sum_{i=0}^{Ng-1} \sum_{j=0}^{Ng-1} (i-j)^2 \cdot P(i,j|d,\Theta)$$

These measures relate to specific texture characteristics of the image such as homogeneity, contrast, and nature of gray level transitions which occur in the image. Other textural features are defined in Appendix A.

0	1	2	0
1	3	0	2
2	1	2	0
1	0	1	3

Figure 13(a) [4 x 4 Image with four gray levels ranging 0-3]

Gray Level		0	1	2	3
Gray Level	0	#(0,0)	#(0,1)	#(0,2)	#(0,3)
	1	#(1,0)	#(1,1)	#(1,2)	#(1,3)
	2	#(2,0)	#(2,1)	#(2,2)	#(2,3)
	3	#(3,0)	#(3,1)	#(3,2)	#(3,3)

Figure 13(b) General form of any gray level co-occurrence matrix for image data given in 13(a). #(i,j) denotes the number of times gray level i and j have been neighbours.

$$P(d,\theta) = \frac{1}{R} \times \begin{bmatrix} 0 & 2 & 1 & 0 \\ 1 & 0 & 2 & 2 \\ 2 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \quad \text{For} \quad \begin{array}{l} d = 1 \\ \theta = 0^0 \\ R = 12 \end{array}$$

(a)

$$P(d,\theta) = \frac{1}{R} \times \begin{bmatrix} 1 & 0 & 1 & 0 \\ 2 & 2 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad \text{For} \quad \begin{array}{l} d = 1 \\ \theta = 45^0 \\ R = 9 \end{array}$$

(b)

Figure 14(a) Gray level co-occurrence matrices for intersampling distance 'd' = 1 and 'θ' = 0^0, 45^0. R is the total number of entries in a matrix

$$P(d,\theta) = \frac{1}{R} \times \begin{bmatrix} 0 & 1 & 2 & 0 \\ 1 & 0 & 2 & 1 \\ 2 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{bmatrix} \quad \text{For} \quad \begin{array}{l} d = 1 \\ \theta = 90^0 \\ R = 12 \end{array}$$

(c)

$$P(d,\theta) = \frac{1}{R} \times \begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix} \quad \text{For} \quad \begin{array}{l} d = 1 \\ \theta = 135^0 \\ R = 9 \end{array}$$

(d)

Figure 14(b) Gray level co-occurrence matrices for intersampling distance 'd' = 1 and 'θ' = 90^0, 135^0. R is the total number of entries in a matrix

4.2.1 Experimental Results:

Four classes, namely, sand, lizard skin, handwoven oriental rattan, and beach pebbles were selected for the classification test. Ten samples of size 64x64 were taken from each class. The gray levels of each sample were mapped to 16 values. The gray level co-occurrence matrices were then computed for each sample for an inter-sample spacing d='1' and directions 'θ' = 0^0, 45^0, 90^0, 135^0. To implement this method, eight features (Appendix A) were extracted from each of these co-occurrence matrices.

If T1, T2, T3, and T4 represent the four texture classes, then let T_{i1} and T_{i2} be the samples of each class T_i (i=1, 2, 3, 4). Figure 15(a)-(d) show typical texture feature values for each T_{i1} and T_{i2} (i=1, 2, 3, 4) for 'θ'=0^0, 135^0. Figure 15(e)-(h) show the normalized inter-sample textural distances for 'd'=1 and 'θ'=0^0, 90^0, 45^0 and 135^0 respectively. The results obtained indicated that the inter-class distances were much greater than the intera-class distances even though some problems did arise for samples T_{12} and T_{32} at 'θ'=135^0. It was also noted that the method was sensitive to change in direction.

4.3 Conclusion:

On the basis of the study done in this section it seems that the gradient vector histogram method is more computationally efficient than the gray level dependence method and performs as accurately as the gray level dependence method.

FEATURES

SAMPLES	Energy (10E-02)	Entropy	Local Homogeneity (10E-01)	Inertia
T11	1.541	1.920	4.6015	4.551
T21	4.594	1.854	4.6623	6.794
T31	4.283	1.612	7.2647	1.683
T41	3.912	1.623	7.1234	1.347

FEATURES

SAMPLES	Sum Average (10E+01)	Sum Entropy	Sum Varience (10E+02)	Difference Entropy (10E-01)
T11	2.071	1.313	4.0625	7.103
T21	1.152	1.312	1.5380	7.845
T31	1.983	1.302	3.7760	4.895
T41	1.243	1.312	1.6637	4.812

Figure 15(a) Texture features computed for inter-sampling space 'd'=1 and 'θ'=0°; T1(Sand); T2(Lizard skin); T3(Handwoven oriental rattan); T4(Beach Pebbles).

FEATURES

SAMPLES	Energy (10E-02)	Entropy	Local Homogeneity (10E-01)	Inertia
T12	1.582	1.902	4.573	4.453
T22	4.662	1.843	4.662	6.764
T32	4.541	1.592	7.339	1.602
T42	3.687	1.631	7.094	1.340

FEATURES

SAMPLES	Sum Average (10E+01)	Sum Entropy	Sum Varience (10E+02)	Difference Entropy (10E-01)
T12	2.080	1.290	4.0801	7.053
T22	1.140	1.302	1.5185	7.832
T32	1.950	1.300	3.6702	4.770
T42	1.260	1.324	1.6980	4.831

Figure 15(b) Texture features computed for inter-sampling space 'd'=1 and 'θ'=0°; T1(Sand); T2(Lizard skin); T3(Handwoven oriental rattan); T4(Beach Pebbles).

FEATURES

SAMPLES	Energy (10E-02)	Entropy	Local Homogeneity (10E-01)	Inertia
T11	1.290	1.990	4.0403	7.002
T21	3.230	1.970	3.5510	1.701
T31	2.011	1.900	4.5310	6.100
T41	3.620	1.661	6.8700	1.781

FEATURES

SAMPLES	Sum Average (10E+01)	Sum Entropy	Sum Varience (10E+02)	Difference Entropy (10E-01)
T11	2.070	1.290	4.0421	7.900
T21	1.150	1.290	1.4400	9.532
T31	1.980	1.300	3.7310	7.512
T41	1.240	1.310	1.6608	5.201

Figure 15(c) Texture features computed for inter-sampling space 'd'=1 and 'θ'=135°; T1(Sand); T2(Lizard skin); T3(Handwoven oriental rattan); T4(Beach Pebbles).

FEATURES

SAMPLES	Energy (10E-02)	Entropy	Local Homogeneity (10E-01)	Inertia
T12	1.370	1.960	4.1453	6.510
T22	3.270	1.970	3.6120	1.685
T32	2.100	1.890	4.4600	6.362
T42	3.290	1.690	6.7356	2.020

FEATURES

SAMPLES	Sum Average (10E+01)	Sum Entropy	Sum Varience (10E+02)	Difference Entropy (10E-01)
T12	2.080	1.280	4.0700	7.780
T22	1.140	1.290	1.4220	9.500
T32	1.950	1.290	3.6400	7.580
T42	1.260	1.320	1.7043	5.380

Figure 15(d) Texture features computed for inter-sampling space 'd'=1 and 'θ'=135°; T1(Sand); T2(Lizard skin); T3(Handwoven oriental rattan); T4(Beach Pebbles).

	T11	T12	T21	T22	T31	T32	T41	T42
T11	0	.022	25.0	25.3	30.6	28.2	37.6	36.1
T12		0	25.7	26.0	29.5	27.1	36.5	35.1
T21			0	.007	57.9	56.6	58.3	58.5
T22				0	57.5	56.3	57.9	58.1
T31					0	.091	5.79	5.55
T32						0	6.40	6.36
T41							0	.052
T42								0

Figure 15(e) Texture distances for 'd'=1 and 'θ'=0°;
T11,T12 ∈ T1; T21,T22 ∈ T2;
T31,T32 ∈ T3; T41,T42 ∈ T4.

	T11	T12	T21	T22	T31	T32	T41	T42
T11	0	.207	35.8	36.3	2.30	1.87	35.7	31.4
T12		0	39.6	40.0	3.41	3.00	31.8	27.9
T21			0	.005	21.6	23.1	78.8	74.8
T22				0	22.0	23.5	78.9	74.8
T31					0	.082	38.9	34.7
T32						0	40.9	36.6
T41							0	.188
T42								0

Figure 15(f) Texture distances for 'd'=1 and 'θ'=90^0; T11,T12 $\in$ T1; T21,T22 $\in$ T2; T31,T32 $\in$ T3; T41,T42 $\in$ T4.

	T11	T12	T21	T22	T31	T32	T41	T42
T11	0	.152	18.4	18.7	2.74	1.67	51.9	49.0
T12		0	18.1	18.5	2.10	1.25	47.5	44.7
T21			0	.011	11.6	12.9	37.1	36.2
T22				0	12.0	13.3	38.3	37.4
T31					0	.162	33.3	31.2
T32						0	38.0	35.7
T41							0	.090
T42								0

Figure 15(g) Texture distances for 'd'=1 and 'θ'=45^0; T11,T12 $\in$ T1; T21,T22 $\in$ T2; T31,T32 $\in$ T3; T41,T42 $\in$ T4.

	T11	T12	T21	T22	T31	T32	T41	T42
T11	0	.406	100.5	98.9	1.63	1.40	65.0	58.1
T12		0	101.5	99.6	.895	.561	56.5	50.0
T21			0	.023	95.8	97.8	148.6	142.4
T22				0	93.9	95.9	145.2	139.0
T31					0	.078	47.8	42.0
T32						0	48.4	42.6
T41							0	.253
T42								0

<u>Figure</u> 15(h) Texture distances for 'd'=1 and 'θ'=135°; T11,T12 ∈ T1; T21,T22 ∈ T2; T31,T32 ∈ T3; T41,T42 ∈ T4.

CHAPTER V

SEGMENTATION OF TEXTURED IMAGES

Image segmentation is defined as the partitioning of an image into different regions each having certain properties. In the previous chapters, methods for characterizing and classifying a texturally homogeneous region are given. In many applications, images are composed of several regions, each possessing different characteristics, and it is important to isolate them. The problem of locating the boundaries between regions of different textures is known as a texture segmentation problem.

Frequently used texturally based approaches to image segmentation are: region growing, region merging, and region splitting. Several fairly extensive surveys are given in (Zucker, 1976; Fu and Mui, 1981; and Rosenfeld and Kak, 1982).

The concept of region based segmentation, makes use of the regional properties. Region growing and region merging are processes in which adjacent regions having similar texture characteristics are combined or merged. Region splitting is a process where the detection of dissimilarity of texture among two or more neighbourhoods within a region results in splitting the region into regions of greater homogeneity. The segmentation process is initiated by dividing the image into blocks of equal size. Each block is represented by its textural features. Then the textural features of the first image block is compared to its neighbouring blocks by applying an appropriate feature comparison technique. All neighbouring blocks that meet the similarity criteria become part of the region being grown. The neighbours of the newly accepted blocks are then examined for further growth. The process is repeated recursively until no more blocks can be added to the region under consideration. Then, a new region is grown around a block which has not been previously selected. When all the blocks have been assigned to regions, the algorithm terminates.

Muerely and Allen, 1968, used this approach by comparing the statistics of an image block with its neighbouring blocks. The regions found were homogeneous according to gray level statistics. This approach was subsequently refined by Brice and Femmema, 1974 who used multi-regional heuristics.

This thesis employs a region growing technique to establish the boundaries between different textured regions. A detailed explanation of this method can be found in the next section.

5.1 The Segmentation Algorithm:

Given a textured image, how can it be segmented into different texturally homogeneous regions?

The proposed algorithm is based on both local and global textural information. The process begins by dividing the image into blocks, of size NRxNC pixels, where NR and NC are the number of rows and columns respectively in the image block. The size of the image block for a homogeneous texture can be determined by finding the textural distance between two blocks of size 2x2, 4x4, 8x8, and so on. The size at which the textural distance starts to stablize can be chosen as the size of the image block. Each block is then represented by its gradient vector histogram. The gradient is a two-dimensional vector representing gradient magnitude and directionality.

The region growing concept starts by setting up a labelled map of the same size as the divided image. For example, consider an image of size 128x128 pixels. If the block size is 32x32 pixels , then the labelled map will be a 4x4 array. Labelling is initiated by assigning the image block at (1,1) the label '1'. By the application of a suitable similarity criteria it is then determined if this label can be assigned to the unlabelled neighbours of the current image block. If any of the neighbours are assigned the current label, then the addresses of these blocks are stored in a stack and all relevent arrays are updated. The addresses are then sequentially retrieved from the stack on a "first-in, first-out" basis and the neighbours of these blocks are then examined for label assignment. After the first set of entries in the stack have been examined, these are eliminated from the stack by a simple reassignment.

If, at any stage of the label propagation, no new entries are added to the stack, the propagation of that specific label is terminated since further region growth is not possible. At this stage, the stack is reset and the growth of another region is initiated by assigning a new label to the next unlabelled block. This process is repeated until all the blocks in the labelled map have been assigned appropriate labels.

At the end of this step, all the image blocks have been labelled and the segmentation is completed, although the boundaries between different textured regions are not well defined due to the large size of the image blocks. In order to achieve a reasonably high resolution, or equivalently, detect small regions of homogeneous textures, the image block should be smaller in size, especially when the block contains the boundary between different textured regions. In order to define the boundaries properly, the image blocks in the vicinity of the boundary must be very small. On the other hand the image block must be large enough to be a valid texture sample. If a region of uniform texture is sub-divided such that the image blocks do not adequately reflect the texture characteristics, the uniform region will be labelled as being nonuniform. Thus, in order to avoid the loss of texture information, an overlapping scheme can be used where the comparisons are made on the basis of large blocks while the labelling is done for the smaller blocks.

A block diagram of the segmentation algorithm is given in fig. 16. Figure 17 schematically depicts the labelling process for a non-overlapping case. In order to define the boundaries properly, an overlapping scheme must be used. Figure 18(a) shows a 4x4 labelled map for the non-overlapping case. The labelled map of 7x7, obtained after doing an overlapping of half the size of the image block in the horizontal and vertical directions, is given in fig. 18(b). Figure 19 illustrates the process of labelling in an overlapping case. Here the labelling is done to smaller blocks while the comparisons are made on the basis of larger blocks. The essential steps in the segmentation process are given next.

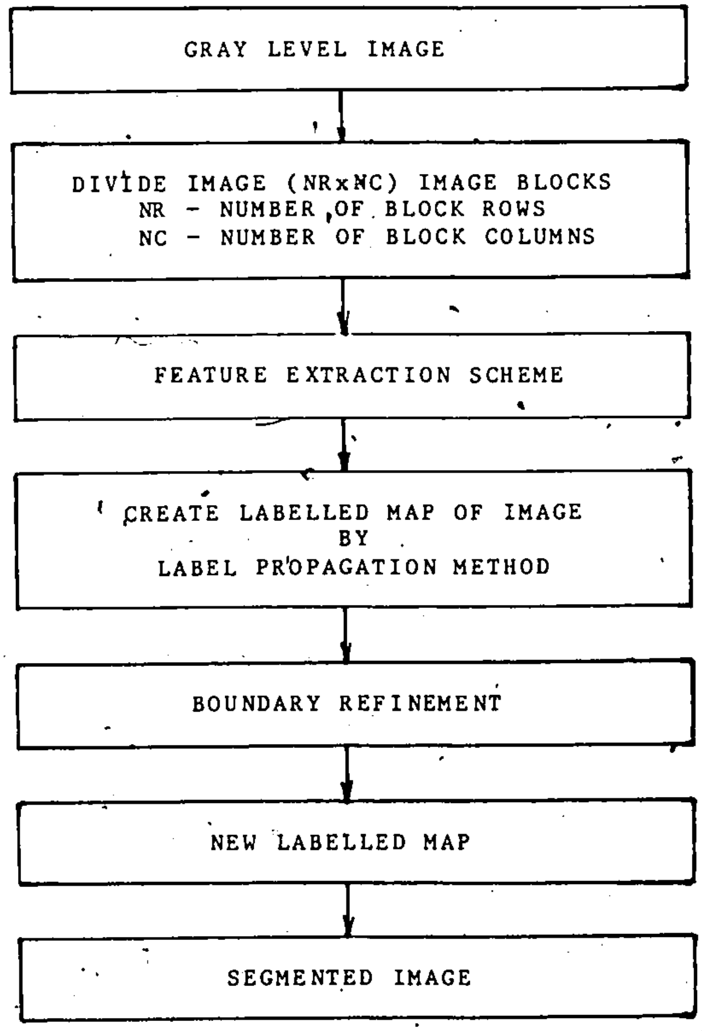


Figure 16 Block Diagram of The Segmentation Algorithm

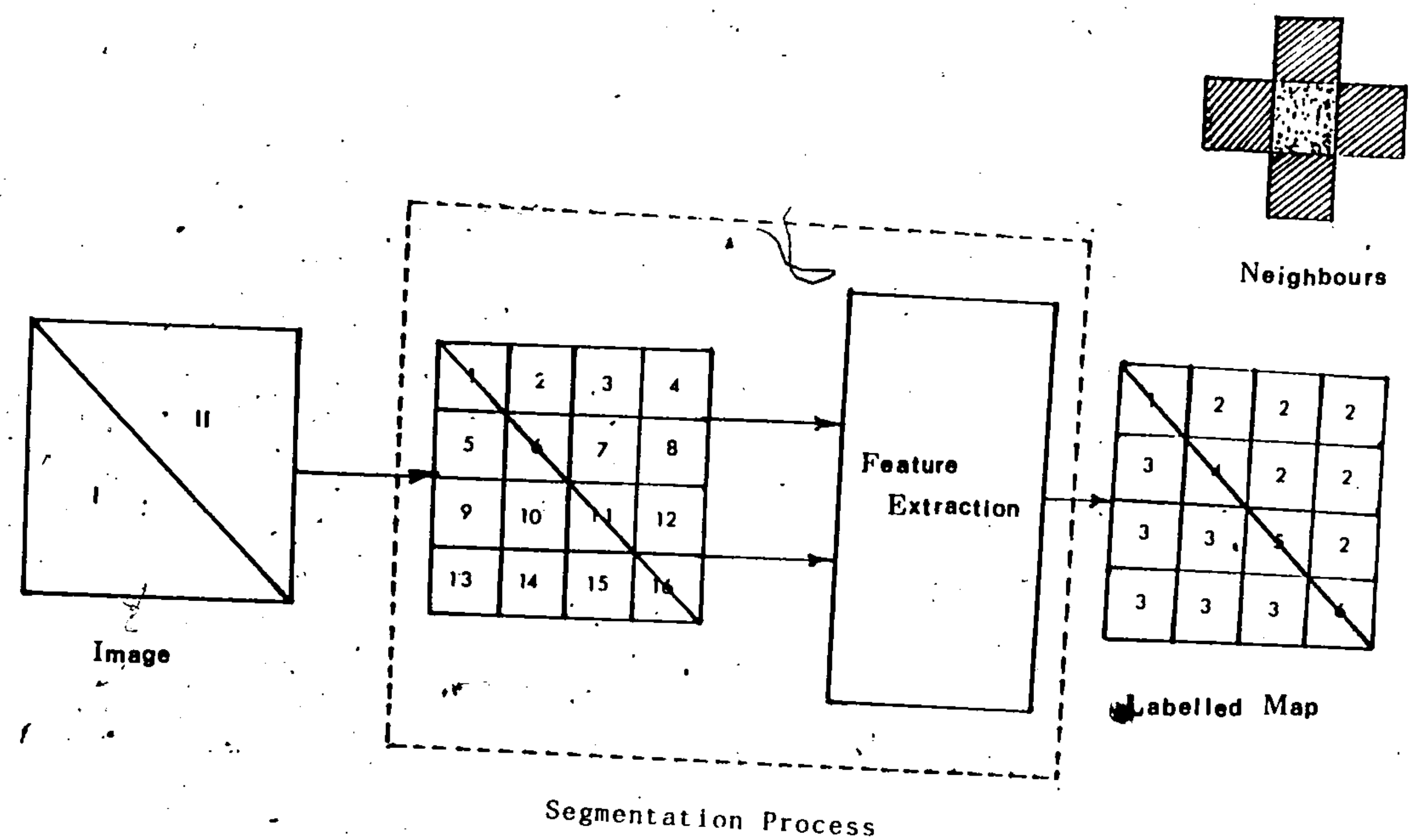
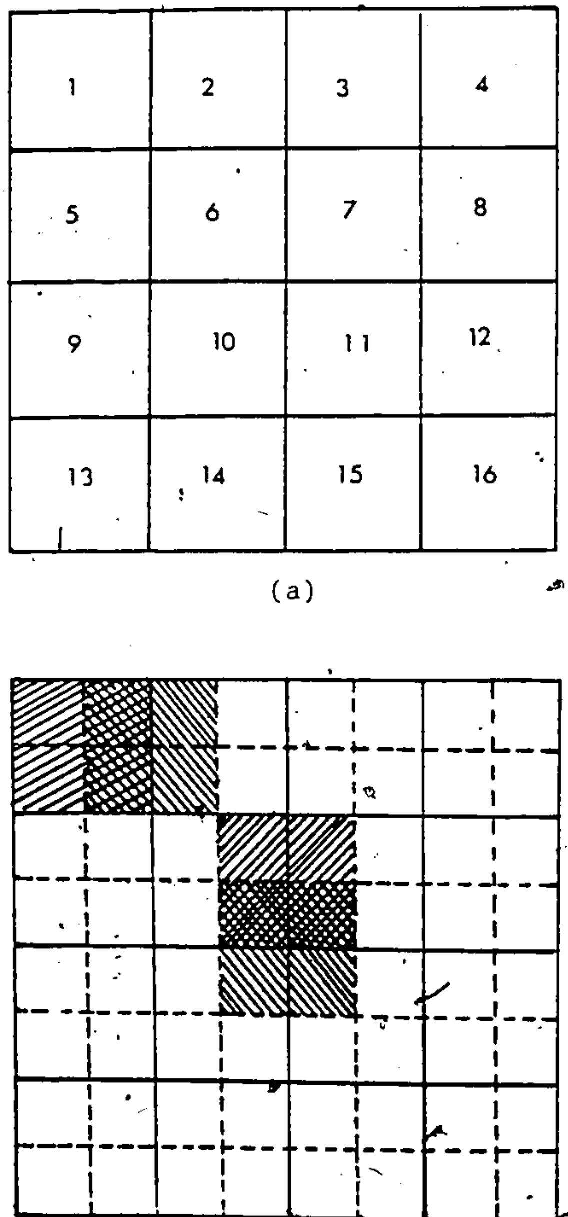


Fig.17 Schematic Representation of the Segmentation Algorithm



(a)

(b)

Fig. 18 (a) Labelled Map for Non-overlapping Case
(b) Horizontal and Vertical Overlapping Case

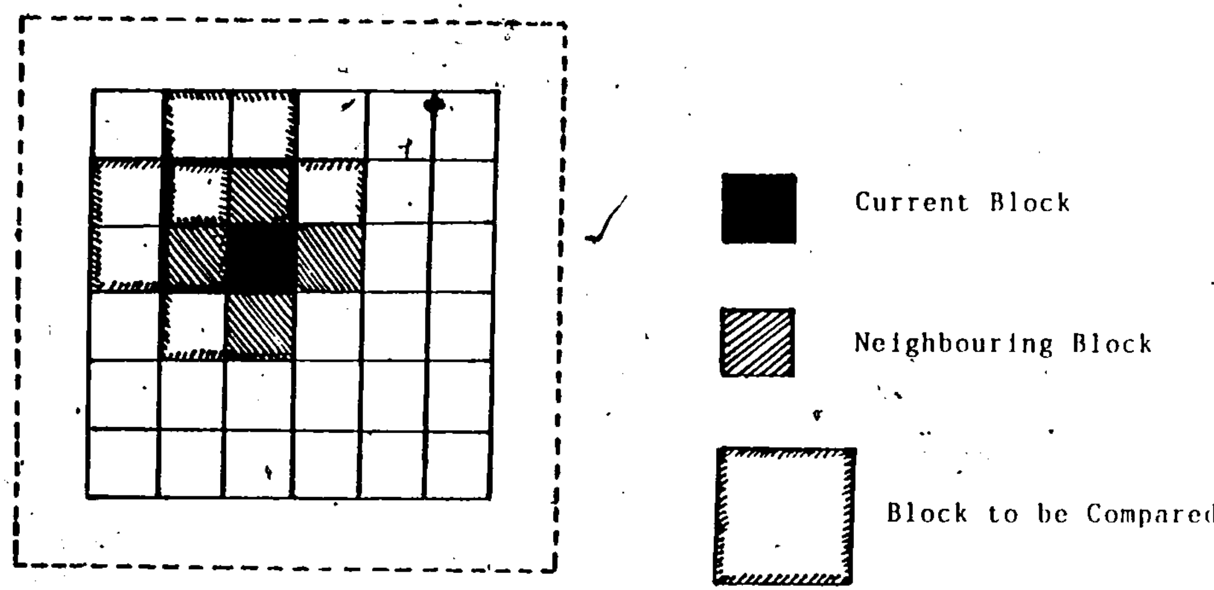


Fig. 19 Labelling Process (Overlapping)

ALGORITHM

Input: (a) Gray level array for digitized image
{Img(I,J), J=1, 2,...., N; I=1, 2,...., N}

(b) Threshold value (TH)

Output: (a) Texturally segmented image

(b) Number of regions in the image.

Procedure:

1. Divide the image into NR x NC image blocks.
   NR - Number of block rows
   NC - Number of block columns

2. Set up a labelled array IL(.,.) of same size as the divided image
   i.e. array {IL(I,J), J=1, 2,...., NM1; I=1, 2,...., NM1}
   NM1=(Size of the image/Size of each block)

3. Set up arrays {LX(.),LY(.)} for storing the co-ordinates of the four neighbours of an image block and arrays {LXT(.),LYT(.)} for storing block co-ordinates.

4. Initialize:
   IL(.,.) <-- 0; NL <-- 0; ILL <-- 1

5. Start Labelling {I, J represent locations in the labelled array: 1 <= I, J <= NM1}.

   5(a). If array IL(.,.) at location (I,J) has label '0' then assign it with label ILL otherwise increment J and go to step 5.

   5(b). Compute the gradient vector histogram (H1) for image block at location (I,J).

   5(c). Compute four neighbouring blocks of image block (I,J)
   {LX(K),LY(K); K=1, 2, 3, 4}.

   5(d). Compare each neighbour with block at (I,J)

(For 1 <= K <= 4; K is the neighbour being compared).

5(e). If the neighbour at location (LX(K),LY(K)) is unlabelled then find its gradient vector histogram (H2) otherwise increment K and go to step 5(d).

5(f). Evaluate the texture distance between H1 and H2 and denote it by D(H1,H2).

5(g). If D(H1,H2) is less than the specified threshold (TH) then perform the following three steps:
(IL(LX(K),LY(K))) <-- ILL
LXT(NL) <-- LX(K); LYT(NL) <-- LY(K)

5(h). Increment K and go to step 5(d).

5(i). If no more growth is possible (NL=0) then go to step 5(s) otherwise perform the following:
NL1 <-- 1; NL2 <-- NL

5(j). Grow regions around blocks whose addresses are stored on the stack (For NL1 <= I1 <= NL2: where I1 points to the element of the stack under examination).

5(k). Compute H1 for image block (LXT(I1),LYT(I1)).

5(l). Compute four neighbouring blocks of image block (LXT(I1),LYT(I1)) (LX(K),LY(K); K=1, 2, 3, 4).

5(m). Compare each neighbour with block at (LXT(I1),LYT(I1)) (For 1 <= K <= 4; K is the neighbour being compared).

5(n). If the neighbour at location (LX(K),LY(K)) is unlabelled then find its gradient vector histogram (H2) otherwise increment K and go to step 5(m).

5(o). Evaluate the texture distance between H1 and H2 and denote it by D(H1,H2).

5(p). If D(H1,H2) is less than the specified threshold (TH) then perform the following:
(IL(LX(K),LY(K))) <-- ILL
NL <-- NL+1; LXT(NL) <-- LX(K); LYT(NL) <-- LY(K)

5(q). Increment K and go to step 5(m).

5(r). If all the elements on the stack have been examined (NL2=NL) then go to step 5(s) otherwise perform the following:
NL1 <-- NL2+1; NL2 <-- NL
go to step 5(j).

5(s). ILL <-- ILL+1

5(t). Increment J and go to step 5.

5(u). Increment I and go to step 5.

6. Divide boundary blocks into four sub-blocks.

7. Connect each sub-block to the one with minimum texture distance from it.

8. Merge all the isolated regions which may not be more than one block size with the largest adjacent region.

9. Create a segmented image from the labelled map.

10. Stop; End.

1.1 A Threshold Selection Technique:

The success of the segmentation algorithm given in the previous section depends upon the threshold value used. The method proposed to evaluate the threshold value makes use of texture distance statistics. The following steps are required to calculate the threshold value:

+ The image is first divided into NRxNC blocks where NR and NC are the number of block rows and columns respectively.

Each block is then represented by its gradient feature histogram. The process initiates by choosing block (1,1). The histogram of this block is compared with the histogram of its neighbouring blocks and the results are stored in an array D(.). The process is repeated recursively until all the blocks have been considered. At the end of this step, we have an array containing all the possible texture distances. The mean and the standard deviation of these distances is then computed. These results are used to calculate the threshold value.

1. Mean:

$$\mu = \frac{1}{N} \sum_{I=1}^{N} D(I)$$

where

N - is the number of entries in D(.)

2. Standard Deviation:

$$\sigma = \sqrt{\frac{1}{N} \sum_{I=1}^{N} (D(I)-\mu)^2}$$

The threshold value (TH) is given as:

$$TH = \mu - \epsilon \cdot \sigma$$

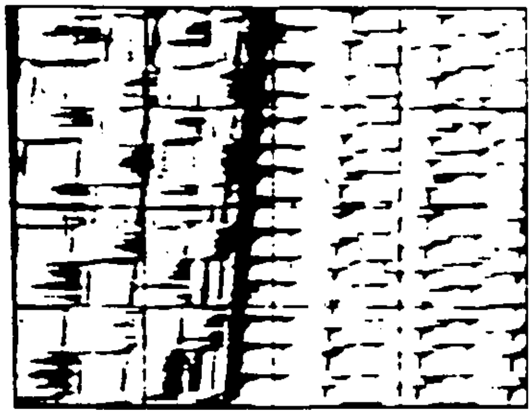
where ϵ is a tolerance factor.

5.3 Discussions and Results:

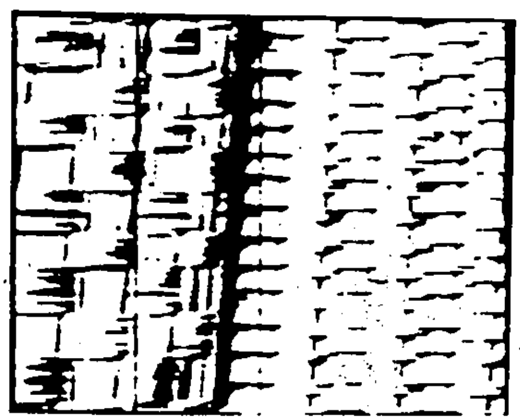
This algorithm is computationally simple and easy to implement. It has been tested on a variety of textured images and was shown to work very efficiently if a proper block size is chosen. The size of the image block for a homogeneous texture can be determined by finding the textural distance between two blocks of size 2x2, 4x4, 8x8, and so on. The size at which the textural distance starts to stablize can be chosen as the size of the image block.

The algorithm fails however, to provide an exact boundary location between different textured regions. In order to achieve a reasonably high resolution, the image block must be as small as possible, especially when the block contains the boundary between different regions. On the other hand the block size must be large enough to be considered as a valid texture sample. Thus in order to maintain the textural information, an overlapping scheme can be used, where the comparisons are made on the basis of the larger blocks and the labelling is done to the smaller blocks. A more detailed explanation of overlapping is illustrated in fig. 19.

The images are directly photographed from Brcdatz, 1966. In this experiment, three textures, namely, handwoven oriental rattan, lizard skin and water, and plastic pellets are chosen. Each image is digitized into 128x128 pixels with 64 gray levels. Figure 20(a) shows the original image of the handwoven oriental rattan (128x128) after dividing it into blocks of size 32x32. The segmented image for the non-overlapping scheme is given in fig. 20(b). Fig. 21(a) shows the image of the lizard skin and water and fig. 21(b) is the segmented image. Fig. 22(a)-(b) shows the original image and the segmented image of the lizard skin and water, using an overlapping of half the size of the image block in the horizontal and vertical directions. The segmented and the original image of the plastic pellets using the non-overlapping and overlapping schemes are given in fig. 23(a)-(b) and fig. 24(a)-(b) respectively. From the results it is obvious that overlapping works better than the non-overlapping method for defining the boundaries between different textured regions. Also it can be observed that there are some extra regions in fig. 24(b) which are due to some small intensity variations. These can be avoided by normalizing the gradient vector histogram with respect to the gray level values.

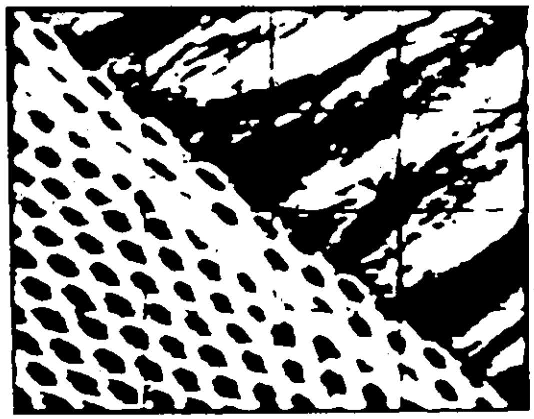


(a)

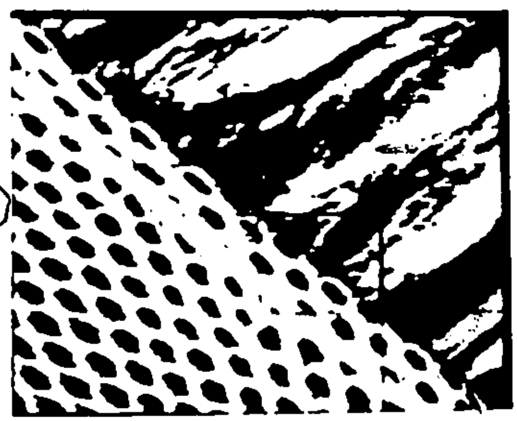


(b)

Fig.20 (a) Handwoven Oriental Rattan
(b) Segmented Image (Non-overlapping)

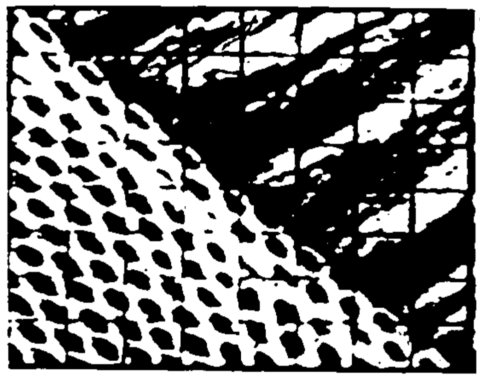


(a)

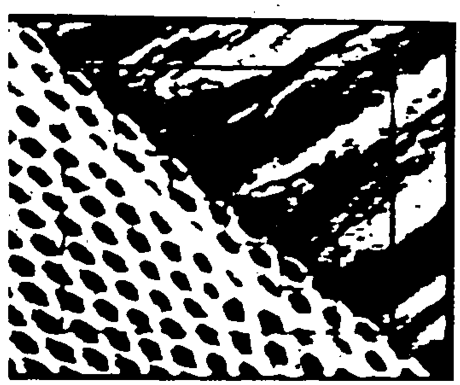


(b)

Fig. 21 (a) Lizard skin and water
(b) Segmented image (non-overlapping)

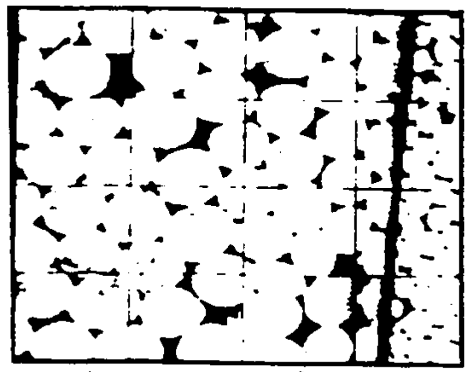


(a)

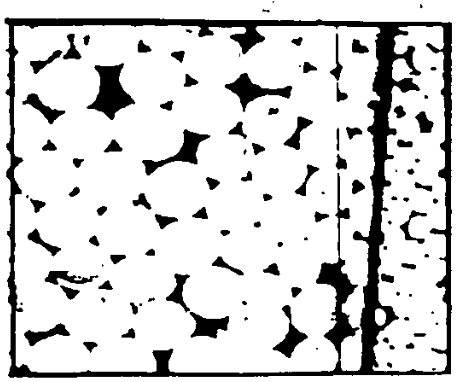


(b)

Fig. 22 (a) Lizard skin and water
(b) Segmented image (overlapping)

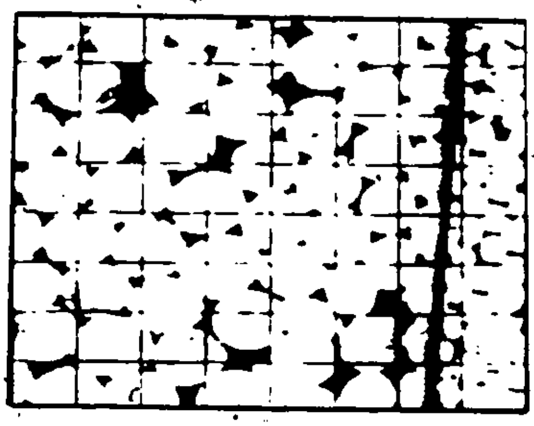


(a)

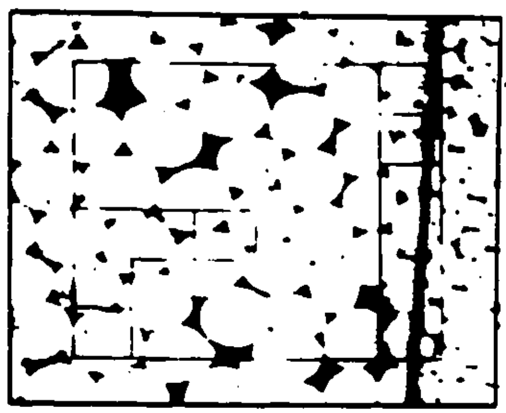


(b)

Fig. 23 (a) Plastic Pellets
(b) Segmented image (non-overlapping)



(a)



(b)

Fig. 24 (a) Plastic Pellets
(b) Segmented image (overlapping)

CHAPTER VI

SUMMARY AND CONCLUSIONS

Texture has become critically important to researchers in the various areas of image processing because almost all natural surfaces are characterized by local intensity variations. This thesis has dealt with three main problems of texture analysis, namely, texture description, texture discrimination or classification, and segmentation of textured images.

A critical analysis of the work done by Raafat, 1985, has been given in Chapter III and Chapter IV. The problem of texture representation is examined by extracting features through observation windows at different resolution levels. Texture is considered as a two layer process. The first layer is for describing the tonal primitives or local properties and the second is for describing the spatial organization of these tonal primitives. The primitive texture features used are the gray level feature and the gradient vector feature at different resolution levels. The spatial organization of these features are given by their histograms namely, the gray level histogram and the gradient vector histogram. Thus texture description of a given image block is given by a set of histograms of various primitive texture features at different resolution levels.

The problem of texture classification involves the comparison of two texture samples. Here each sample is represented by its gradient and gray level histograms. The problem of histogram comparison is then formulated as a transportation problem. This provides a texture distance between two image blocks which is used for classification and segmentation purposes.

A new methodology for segmentation of a textured image onto texturally homogeneous regions is given in Chapter V. The algorithm is initiated by dividing the image into blocks of equal size. Each block is then represented by its textural features. Next, the textural features of the first image block are compared to its neighbouring blocks by applying an appropriate feature comparison technique (Chapter IV). All neighbouring blocks that meet the similarity criteria become part of the region being grown. The neighbours of the newly accepted blocks are then examined for further growth. The process is repeated recursively until no more blocks can be added to the region under consideration. Then a new region is grown around a block which has not been previously labelled. When all the blocks have been assigned to regions, the algorithm terminates. The following conclusions are drawn on the basis of the study and the research done in this project.

It was found that the representation of a texture sample by its histograms shows little variations across different samples taken from the same texture. Also it contains essential texture characteristics which define the textural differences between two samples. For the classification purposes, it was found that the performance of the method proposed by Raafat, 1985 was computationally better than the spatial gray level dependence method (Haralick, 1973). It was also noticed that the gray level feature in the first scheme did not give sufficient information for discrimination purposes; however the gradient feature alone was quite adequate for discrimination and segmentation.

The algorithm proposed in Chapter V does not provide well defined boundaries between different textured regions due to the large size of the image blocks. This problem can be resolved by using image blocks that are smaller in size, especially for the blocks containing the boundaries. However the image block size must be large enough to be a valid texture sample. Thus in order to achieve a reasonably high resolution, an overlapping scheme should be used.

Future research is needed for an appropriate selection of the image block size for segmentation purposes. Also other textural features can be introduced for a better texture description. Research is also needed for better location of boundaries between different textured regions.

Appendix A

TEXTURAL FEATURES

The following is a reference summary of the features used for the extraction of textural information. from the gray level co-occurrence matrices, gray level run length matrices, gray level difference density functions, and power spectrum (Haralick et al., 1973; Conners et al., 1980).

The textural features which can be extracted from each of the gray level co-occurrence matrices are given below:

Notations:

$P(I,J)$ - (I,J)th entry in a gray level co-occurrence matrix

$P(I)$ - Ith entry in the marginal-probability matrix and is given as:

$$= \sum_{J=1}^{Ng} P(I,J) \; ; \; I = 1, 2, 3, \ldots, Ng;$$

Ng - Number of distinct gray levels

1. Angular Second-Moment:

$$ASM = \sum_{I} \sum_{J} [P(I,J)]^2$$

2. Contrast:

$$CON = \sum_{N} N^2 \sum_{I=0}^{Ng-1} \sum_{\substack{J=0 \\ |I-J|=N}}^{Ng-1} P(I,J)$$

3. Correlation:

$$COR = \frac{\sum_{I=0}^{Ng-1} \sum_{J=0}^{Ng-1} (I-\mu_x)(J-\mu_y) P(I,J)}{\sigma_x \cdot \sigma_y}$$

where

$$\mu_x = \sum_{I=0}^{Ng-1} I \sum_{J=0}^{Ng-1} P(I,J)$$

$$\mu y = \sum_{J=0}^{Ng-1} J \sum_{I=0}^{Ng-1} P(I,J)$$

$$\sigma x = \sum_{J=0}^{Ng-1} (I-\mu x)^2 \sum_{I=0}^{Ng-1} P(I,J)$$

$$\sigma y = \sum_{J=0}^{Ng-1} (J-\mu y)^2 \sum_{I=0}^{Ng-1} P(I,J)$$

3. Local Homogeneity:

$$Lh = \sum_{I=0}^{Ng-1} \sum_{J=0}^{Ng-1} \frac{1}{1+(I-J)^2} P(I,J)$$

5. Entropy:

$$Ep = -\sum_{I=0}^{Ng-1} \sum_{J=0}^{Ng-1} P(I,J) \log P(I,J)$$

6. Sum Entropy:

$$Es = -\sum_{I=0}^{2Ng-2} Ps(I) \log\{Ps(I)\}$$

where

$$Ps(K) = \sum_{\substack{I=0 \\ K=I+J}}^{Ng-1} \sum_{J=0}^{Ng-1} P(I,J) \;;\; K=0,\ 1,\ 2,\ldots,\ 2Ng-2$$

7. Sum Varience:

$$Sv = \sum_{I=0}^{2Ng-2} (I - Es)^2 Ps(I)$$

8. Sum Average:

$$Sa = \sum_{I=0}^{2Ng-2} I\ Ps(I)$$

9. Difference Entropy:

$$De = - \sum_{I=0}^{Ng-1} Pd(I) \log\{Pd(I)\}$$

where

$$Pd(K) = \sum_{\substack{I=0 \\ |I-J|=K}}^{Ng-1} \sum_{J=0}^{Ng-1} P(I,J) \quad ; K=0,\ 1, \ldots, Ng-1$$

The textural features extracted from the gray level run length matrices are:

Notations:

$R(I,J)$ - (I,J)th entry of gray level run length matrices

Tr - $\sum_{I=0}^{Ng-1} \sum_{J=1}^{Nr} R(I,J)$

Tp - Number of points in the picture

Nr - Number of run lengths in the matrix

1. Short Run Emphasis:

$$SRE = \frac{1}{Tr} \sum_{I=0}^{Ng-1} \sum_{J=1}^{Nr} \frac{R(I,J)}{J^2}$$

2. Long Run Emphasis:

$$LRE = \frac{1}{Tr} \sum_{I=0}^{Ng-1} \sum_{J=1}^{Nr} J^2 \, R(I,J)$$

3. Gray Level Distribution:

$$GLD = \frac{1}{Tr} \sum_{I=0}^{Ng-1} \sum_{J=1}^{Nr} R(I,J)^2$$

4. Run Length Distribution:

$$RLD = \frac{1}{Tr} \sum_{J=1}^{Nr} \sum_{I=0}^{Ng-1} R(I,J)^2$$

5. Run Percentages:

$$RP = \frac{1}{Tp} \sum_{I=0}^{Ng-1} \sum_{J=1}^{Nr} R(I,J)$$

The statistical features used to extract textural information from the gray level difference density functions are:

Notations:

F(I|d) - Estimated probability associated with 'I'

Ng - Number of gray levels

d - any displacement = (d1,d2) where d1 and d2 are integers

1. Contrast:

$$CON = \sum_{I=0}^{Ng-1} I^2 \; F(I|d)$$

2. Angular Second-Moment:

$$ASM = \sum_{I=0}^{Ng-1} F(I|d)^2$$

3. Entropy:

$$ENT = \sum_{I=0}^{Ng-1} F(I|d) \log(F(I|d))$$

4. Mean:

$$MUE = \sum_{I=0}^{Ng-1} I \; F(I|d)$$

5. Inverse Difference Moment:

$$IDM = \sum_{I=0}^{Ng-1} \frac{F(I|d)}{(I^2 + 1)}$$

The features commonly used with the power spectrum method are given below:

Notations:

$\Phi'(u,v)$ - is the sampling power spectrum

Ma - Number of annular rings

Mw - Number of wedges

Ms - Number of slits

1. Annular-ring Sampling Geometry:

$$ASG = \int_{0}^{\pi}\int_{\rho}^{\rho_j+\Delta\rho} \Phi'(a,\theta)\,a\,da\,d\theta; \quad j=1,\ 2,\dots,\ Ma$$

where

$$a = \sqrt{(u^2 + v^2)}$$

$$\theta = \tan^{-1}(u/v)$$

2. Wedge Sampling Geometry:

$$WSG = \int_{\rho_{min}}^{\rho_{max}}\int_{\theta_j}^{\theta_j+\Delta\theta_j} \Phi'(a,\theta)\,a\,da\,d\theta \quad ; \; j=1,\ 2,\dots,\ Mw$$

3. Slit Sampling Geometry:

$$SSG = \int_{-v_{max}}^{v_{max}}\int_{u_j}^{u_j+\Delta u} \Phi'(u,v)\,du\,dv \quad ; \; j=1,\ 2,\dots,\ Ms$$

Appendix B

PROGRAM LISTING

```
{
*****                 SEGMENTATION ALGORITHM                    *****

Purpose :
~~~~~~~
     This algorithm establishes boundaries between different
     textured regions.

Array Description :
~~~~~ ~~~~~~~~~~~
       Fr  [.,.]                  - Digitized Image
       I1  [.,.]                  - Labelled Array
       Cv  [.,.]                  - Event Distance Set
       Ar1 [.,.], Ar2 [.,.] - Texture Distance Arrays

*************************************************************
                                                            }

Program Labover ;

Label
  Five ;

Type
  Image  = Packed Array [1..128, 1..128] of Byte ;
  Store1 = Packed Array [0..40, 0..40] of Integer ;
  Store2 = Packed Array [1..8, 1..8] of Real ;
  Store3 = Packed Array [1..8, 1..8] of Real ;
  Store6 = Packed Array [1..4]  of Integer ;
  Store  = Packed Array [0..40] of Integer ;

{**  Declaration of Global Variables  **}

Var
  Infile      : File ;
  Ian         : File of Byte ;
  Inname      : String [15] ;
  Fr          : Image ;
  I1          : Packed Array [1..8,1..8] of Integer ;
  Cv          : Store1 ;
  Ar1         : Store2 ;
  Ar2         : Store3 ;
  Count       : Integer ;
  Rows, Cols  : Integer ;
  Nr, Nc      : Integer ;
  Nm1, I, J   : Integer ;
  Decision    : Integer ;
  Limit       : Integer ;
  Ans         : Char ;
  Threshold   : Real ;

{****   This procedure reads in the input image file ****}
```

```
Procedure Readfile ;

Label
  ReadAgain ;

Var
  I, J : Integer ;
  Okay : Char ;

Begin
  ReadAgain : ClrScr ;

  Gotoxy (15, 10) ;
  TextColor (12) ;
  Write (' ':5 ,'  Enter name of image file    ') ;
  Readln (Inname) ;

  Gotoxy (10, 13) ;
  Write ('   Enter Number of rows      ') ;
  Readln (Rows) ;

  Gotoxy (10, 15) ;
  Write ('   Enter Number of Columns ') ;
  Readln (Cols) ;

  Gotoxy (10, 17) ;
  Write ('   Enter Number of block rows   ') ;
  Readln (Nr) ;

  Gotoxy (10, 19) ;
  Write ('   Enter Number of block cols   ') ;
  Readln (Nc) ;
  Nml := Round (Rows / Nr) ;

  Gotoxy (10, 21) ;
  Write ('   Overlapping Yes(1), No(0)   ') ;
  Readln (Decision) ;
  Writeln ;
  ClrScr ;
  TextColor (14) ;

  Gotoxy (15, 10) ;
  Write ('  Name of image file is ', Inname : 10) ;
  Gotoxy (15, 12) ;
  Write ('  Number of Rows is ', Rows :3) ;
  Gotoxy (15, 13) ;
  Write ('  Number of Cols is ', Cols :3) ;
  Gotoxy (15, 15) ;
  Write ('  Would you like to Change any Parameters   ') ;
  Readln (Okay) ;

  If (Okay = 'y') or (Okay = 'Y')
```

```
    Then goto ReadAgain ;

  ClrScr ;
  Gotoxy (20,14) ;
  Textcolor (10) ;
  Writeln ('  READING OF IMAGE FILE IN PROGRESS...') ;
  If (Cols * Rows = 16384) Then
  Begin
    Assign (InFile, Inname) ;
    Reset (InFile) ;
    BlockRead (InFile, Fr, 128) ;
    Close (InFile) ;
    For i:=1 to rows do
      For j:=1 to cols do
        Fr[i,j]:=round(Fr[i,j]/4.0+0.5);
  End

  Else
  Begin
    TextMode ;
    Assign (Ian, Inname) ;
    Reset (Ian) ;
    For I := 1 to Rows do
      For J := 1 to Cols do
        Read(Ian, Fr[I,J]) ;
    Close (Ian) ;
  End ;
End ;

{****  This procedure writes the segmented image  ****}

Procedure Writefile;

Begin
  ClrScr ;
  Gotoxy (20,14) ;
  Write (  'Enter the name of the output file ') ;
  Readln (Inname) ;
  Assign (InFile, Inname) ;
  Rewrite (InFile) ;
  Blockwrite (InFile, Fr, 128) ;
  Close (InFile) ;
End ;

{****  This procedure finds the neighbouring blocks   ****}
{****  of the image block under growth                ****}

Procedure Nb4 (I, J : Integer ; Var Lx, Ly : Store6) ;

Begin
  Lx[1] := I ;     Ly[1] := J - 1 ;
  Lx[2] := I - 1 ; Ly[2] := J ;
```

```
  Lx[3] := I ;       Ly[3] := J + 1 ;
  Lx[4] := I + 1 ; Ly[4] := J ;
End ;

{****  This procedure calculates the threshold value    ****}
{****  for labelling purposes .                         ****}

Procedure Thresh ;

Var
  Sd, Mean, Th : Real ;
  Sum, Count   : Real ;
  I, J         : Integer ;

Begin
  Sum := 0 ;
  For I := 1 to Limit do
    For J := 1 to (Limit-1) do
      Sum := Sum + Ar1[I,J] ;

  For I := 1 to (Limit-1) do
    For J := 1 to Limit do
      Sum := Sum + Ar2[I,J] ;

  Mean := Sum / (2.0 * Limit * (Limit - 1)) ;
  Sum := 0 ;

  For I := 1 to Limit do
    For J := 1 to (Limit-1) do
      Sum := Sum + Sqr (Ar1[I,J] - Mean) ;

  For I := 1 to (Limit-1) do
    For J := 1 to Limit do
      Sum := Sum + Sqr (Ar2[I,J] - Mean) ;

  Sd := Sqrt (Sum / (2.0 * Limit * (Limit - 1))) ;

  ClrScr ;
  Gotoxy (20,14) ;
  TextColor (12) ;
  Write ('  Enter the fractional value ') ;
  Readln (Th) ;

  Threshold := Mean - Th * Sd ;
  Writeln (' ':3, 'Threshold is = ', Threshold) ;
End ;

{****  This procedure creates a labelled matrix for   ****}
{****  the segmented image                            ****}

Procedure Labelling (var Fr : image);
```

```
Var
  Lxt, Lyt : Packed Array [1..1024] of Integer ;
  Ill, Nl  : Integer ;
  I, Il    : Integer ;
  J, K     : Integer ;
  Nll, Nl2 : Integer ;
  Limitl   : Integer ;
  Di       : Real ;
  Lx, Ly   : Store6 ;
  Rep      : Boolean ;

Begin
  ClrScr ;
  Gotoxy (20,14) ;
  Writeln ('  SEGMENTATION IN PROGRESS... ') ;

  Ill := 1 ;        {**  Starting Label  **}
  If (Decision = 1) Then
    Limitl := Limit - 1

  Else Limitl := Limit ;

  {**  Initialization of labelled matrix  **}

  For I := 1 to Limitl do
    For J := 1 to Limitl do
      Il[I,J] := 0 ;

  For I := 1 to Limitl do
    For J := 1 to Limitl do
    Begin
      Nl := 0 ;     {**  Initialization of Stack Pointer  **}
      If (Il[I,J] = 0) Then
      Begin
        Il[I,J] := Ill ;
        Nb4 (I, J, Lx, Ly) ;

        For K := 1 to 4 do
        Begin

          {**  Check for previosly labelled block or block
               outside of labelling array  **}

          If ((Lx[K] > 0) And
             (Ly[K] > 0) And (Lx[K] <= Limitl)
             And (Ly[K] <= Limitl) And
             (Il[Lx[K],Ly[K]] = 0)) Then

             Begin

               {**  Fetch distance between current block and
                 its neighbours  **}
```

```
          If (K = 1) Then
            Di := Ar1[I,J-1]

          Else If (K = 2) Then
            Di := Ar2[I-1,J]

          Else If (K = 3) Then
            Di := Ar1[I,J]

          Else Di := Ar2[I,J] ;

          If (Di < Threshold) Then
          Begin
            I1[Lx[K],Ly[K]] := I11 ;
            N1 := N1 + 1 ;
            Lxt[N1] := Lx[K] ;
            Lyt[N1] := Ly[K] ;
          End ;
        End ;
End ;

If (N1 <> 0) Then   {**  Grow regions around new  **}
Begin               {**  blocks of same label  **}
  Rep := True ;
  N11 := 1 ;
  N12 := N1 ;

  While (Rep = True) do
  Begin
    For I1 := N11 to N12 do
    Begin

      {**  Fetch neighbours  **}

      Nb4 (Lxt[I1],Lyt[I1],Lx,Ly) ;
      For K := 1 to 4 do
      Begin
        If ((Lx[K] > 0) And
            (Ly[K] > 0) And
            (Lx[K] <= Limit1) And
            (Ly[K] <= Limit1) And
            (I1[Lx[K],Ly[K]] = 0)) Then
        Begin
          If (K = 1) Then
            Di := Ar1[Lxt[I1],Lyt[I1] - 1]

          Else If (K = 2) Then
            Di := Ar2[Lxt[I1] - 1,Lyt[I1]]

          Else If (K = 3) Then
            Di := Ar1[Lxt[I1],Lyt[I1]]
```

```
                    Else Di := Ar2[Lxt[I1],Lyt[I1]] ;

                    If (Di < Threshold) Then
                    Begin
                      I1[Lx[K],Ly[K]] := I11 ;
                      N1 := N1 + 1 ;
                      Lxt[N1] := Lx[K] ;
                      Lyt[N1] := Ly[K] ;
                    End ;
                  End ;
                End ;
              End ;

              If (N12 <> N1) Then
              Begin
                N11 := N12 + 1 ;
                N12 := N1 ;
              End

              Else
              Begin
                I11 := I11 + 1 ;
                Rep := False ;
              End ;
            End ;
          End

          Else I11 := I11 + 1 ; {** Increment label **}
        End ;
      End ;

    ClrScr ;   {** Display labelled array **}
    For I := 1 to Limit1 do
    Begin
      For J := 1 to Limit1 do
      Write (I1[I,J], ' ') ;
      Writeln ;
    End ;

    {** Create segmented image from labelled array **}

    For I := 1 to Rows do
    Begin
      Fr[I,1] := 0 ;
      Fr[1,I] := 0 ;
      Fr[I,Rows] := 0 ;
      Fr[Rows,I] := 0 ;
    End ;

    If (Decision = 0) Then
    Begin
      For I := 1 to (Limit-1) do
```

```
Begin
  If (I1[1,I] <> I1[1,I+1]) Then
    For J := 1 to Nc do
      Fr[J,I * Nr] := 0 ;

  If (I1[I,Limit] <> I1[I+1,Limit]) Then
    For J := ((Rows + 1) - Nr) to Rows do
      Fr[I * Nr,J] := 0 ;
End ;

For I := 2 to Limit do
  For J := 1 to (Limit-1)do
  Begin
    If (I1[I,J] <> I1[I - 1,J]) Then
      For K := (J - 1) * Nr to (J-1) * Nr + Nr do
        Fr[(I - 1) * Nr,K] := 0 ;

     If (I1[I,J] <> I1[I,J + 1]) Then
       For K := (I - 1) * Nc to (I - 1) * Nc + Nc do
         Fr[K,J * Nc] := 0 ;
  End ;
End

Else
Begin
  For I := 17 to 112 do
  Begin
    Fr[17,I] := 0 ;
    Fr[112,I] := 0 ;
    Fr[I,17] := 0 ;
    Fr[I,112] := 0 ;
  End ;

  Nc := Nr Div 2 ;
  Nr := Nc ;

  For I := 1 to (Limit1 - 1) do
  Begin
    If (I1[1,I] <> I1[1,I + 1]) Then
      For J := 1 to Nc do
         Fr[J + 16,I * Nr + 16] := 0 ;

    If (I1[I,Limit1] <> I1[I + 1,Limit1]) Then
      For J := ((Rows + 1) - Nr) to Rows do
        Fr[I * Nr + 1 + 16,J - 16] := 0 ;
  End ;

  For I := 2 to Limit1 do
    For J := 1 to (Limit1 - 1) do
    Begin
      If (I1[I,J] <> I1[I - 1,J]) Then
       For K := ((J-1)*Nr+16) to ((J-1)*Nr+Nr+16) do
```

```
        Fr[(I-1) * Nr + 16,K] := 0 ;

      If (Il[I,J] <> Il[I,J + 1]) Then
        For K := ((I-1)*Nc+16) to ((I-1)*Nc+Nc+16) do
          Fr[K,J * Nc + 16] := 0 ;
    End ;
  End ;
End ;

{****  This procedure computes the gradient vector   ****}
{****  histogram of a given image block              ****}

Procedure Histblk (A, B : Integer ; Var Cc : Store) ;

Var
  Ir               : Packed Array [1..32,1..32] of Byte ;
  G                : Packed Array [1..32,1..32] of Integer ;
  Theta            : Packed Array [1..32,1..32] of Real ;
  Ml, Nl, Mr, Is   : Integer ;
  Xy, K, L, I, J   : Integer ;
  M, N, Count0     : Integer ;
  Count            : Integer ;
  Part1, Part2     : Integer ;
  Gy, Gx, Xx       : Real ;
  Pi, R, S         : Real ;

Begin
  For M := A to (A + Nr - 1) do
    For N := B to (B + Nr - 1) do
      Ir[M - A + 1,N - B + 1] := Fr[M,N] ;

  Pi := 3.1415927 ;

  {**  Initialize gradient and directionality arrays  **}

  For I := 1 to Nr do
    For J := 1 to Nc do
    Begin
      Theta[I,J] := 0.0 ;
      G[I,J] := 0 ;
    End ;

  {**  Compute gradient and directionality  **}

  For I := 2 to (Nr - 1) do
    For J := 2 to (Nc-1) do
    Begin
      Part1 := Ir[I+1,J-1]+ 2 *Ir[I+1,J] + Ir[I+1,J+1] ;
      Part2 := Ir[I-1,J-1]+ 2 *Ir[I-1,J] + Ir[I-1,J+1] ;
      Gy := Part2 - Part1 ;
      Part1 := Ir[I-1,J+1]+ 2 *Ir[I,J+1] + Ir[I+1,J+1] ;
      Part2 := Ir[I-1,J-1]+ 2 *Ir[I,J-1] + Ir[I+1,J-1] ;
```

```
      Gx := Part1 - Part2 ;
      Xx := Abs (Gy) + Abs (Gx) ;
      G[I,J] := Round ((Xx / 6.0) + 0.5) ;   {**  Normalize  **}
                                              {**  gradient   **}
      If (Gx = 0) Then
        Theta[I,J] := Pi / 2.0

      Else
      Begin
        Theta[I,J] := Arctan (Gy / Gx) ;
        If ((Gx < 0.0) And (Gy > 0.0) ) Then
          Theta[I,J] := Theta[I,J] + Pi ;

        If ((Gy < 0) And (Gx < 0)) Then
           Theta[I,J] := Theta[I,J] - Pi ;
      End ;
   End ;

  {**  Evaluate gradient vector histogram using 2-D
       discretization scheme  **}

  Count0 := 0 ;
  For I := 1 to Nr do
  For J := 1 to Nc do
  If (G[I,J] < 5) Then
    Count0 := Count0 + 1 ;

    Cc[0] := Count0 ;
    Mr := 1 ;
    Is := 8 ;
    M  := 5 ;
    N  := 10 ;
    Xy := 4 ;

    For K := 1 to 5 do
    Begin
      R := -Pi / 8.0 ;
      S := Pi / 8.0 ;

      For L := Mr to Is do
      Begin
        Count := 0 ;
        For I := 2 to (Nr-1) do
          For J := 2 to (Nc-1) do
          Begin
            If ((G[I,J] >= M) And (G[I,J] < n)) Then
            Begin
              If (L <> (Xy + 1)) Then
              Begin
                If ((Theta[I,J]>=R)And(Theta[I,J]<s)) Then
                  Count := Count + 1 ;
              End
```

```
                Else
                Begin
                  If ((Theta[I,J]>=R)Or(Theta[I,J]<S)) Then
                    Count := Count + 1 ;
                End ;
              End ;
            End ;

          Cc[L] := Count ;
          If (L = xy) Then
          Begin
            R := S ;
            S := -S ;
          End

          Else
          Begin
            R := S ;
            S := S + Pi / 4.0 ;
          End ;

        End ;

        Mr := Mr + 8 ;
        Is := Is + 8 ;
        M := M + 5 ;
        N := N + 5 ;
          Xy := Xy + 8 ;

        If (K = 4)
          Then N := 255 ;
      End ;
End;

{****  This procedure evaluates the event distances ****}

Procedure Cost ;

Var
  Dist, Angle : Store ;
  First, Last : Integer ;
  Icount, k   : Integer ;
  I, J, F, Ia : Integer ;
  Ib, Ic      : Integer ;

Begin
  ClrScr ;
  TextMode ;
  TextColor (10) ;
  Gotoxy (20,14) ;
  Writeln (' EVENT DISTANCE CALCULATIONS IN PROGRESS...') ;
  Icount := 0 ;
```

```
  First := 1 ;
  Last := 8 ;

  For K := 1 to 5 do
  Begin
    For I := First to Last do
    Begin
      J := I - Icount ;
      Dist[I] := K + 3 ;
      Angle[I] := J ;
    End ;

    First := First + 8 ;
    Last := Last + 8 ;
    Icount := Icount + 8 ;
  End ;

  Dist[0] := 0 ;
  Angle[0] := 0 ;
  F := 2 ;

  For I := 0 to 40 do
    For J := 0 to 40 do
    Begin
      If ((I = 0) And (J = 0))
        Then Cv[I,J] := 0

      Else If ((I = 0) And (J > 0))
        Then Cv[I,J] := F * Dist[J]

      Else If ((I > 0) And (J = 0))
        Then Cv[I,J] := F * Dist[I]

      Else
      Begin
        Ia := Abs (Angle[I] - Angle[J]) ;
        Ib := 8 - Ia ;
        If (Ia < Ib)
          Then Ic := Ia
        Else Ic := Ib ;
          Cv[I,J] := F * Abs (Dist[I] - Dist[J]) + Ic ;
      End ;
    End ;
End ;

{****  This procedure computes the texture distance ****}
{****  between two image blocks using the trans-     ****}
{****  portation algorithm                           ****}

Procedure Dist (Yy,Yy1:Store;L,M:Integer;Arrays:Char) ;

Var
```

```
  Cpr, X         : Storel ;
  Nr, Nc, N      : Integer ;
  Nm, Zr, Zc     : Integer ;
  Icnt, Imax     : Integer ;
  I, J, Minr     : Integer ;
  Minc, Cmin     : Integer ;
  Costl          : Real ;

Begin
  Nr := 40 ;
  Nc := 40 ;
  If (Nr < Nc) Then
  Begin
    N := Nr ;
    Nm := Nc ;
  End

  Else
  Begin
    N := Nc ;
    Nm := Nr ;
  End ;

  Zr := 0 ;
  Zc := 0 ;

  For I := 0 to Nr do
    For J := 0 to Nc do
    Begin
      Cpr[I,J] := Cv[I,J] ;
      X[I,J] := 0 ;
    End ;

  Icnt := 0 ;
  Imax := Nr + Nc + 1 ;

  While (Icnt < Imax) do
  Begin
    Minr := 0 ;
    Minc := 0 ;
    Cmin := Cpr[0,0] ;

    For I := 0 to Nr do
      For J := 0 to Nc do
      Begin
        If (Cmin > Cpr[I,J]) Then
        Begin
          Cmin := Cpr[I,J] ;
          Minr := I ;
          Minc := J ;
        End ;
     End ;
```

```
    Icnt := Icnt + 1 ;
    If (Yy[Minr] < Yyl[minc]) Then
      X[Minr,Minc] := Yy[Minr]
    Else X[Minr,Minc] := Yyl[Minc] ;

    If (Yy[Minr] < Yyl[Minc]) Then
    Begin
      Zr := Zr + 1 ;
      Yyl[Minc] := Yyl[Minc] - Yy[Minr] ;
      Yy[Minr] := 0 ;

      For J := 0 to Nc
         Do Cpr[Minr,J] := Maxint ;
    End

    Else If (Yy[Minr] > Yyl[Minc]) Then
    Begin
      Zc := Zc + 1 ;
      Yy[Minr] := Yy[Minr] - Yyl[Minc] ;
      Yyl[Minc] := 0 ;

      For I := 0 to Nr do
        Cpr[I,Minc] := Maxint ;
    End

    Else
    Begin
      Yy[Minr] := 0 ;
      Yyl[Minc] := 0 ;
      If ((Nr - Zr) >= (Nc - Zc)) Then
      Begin
        For J := 0 to Nc do
          Cpr[Minr,J] := Maxint ;
          Zr := Zr + 1 ;
      End

      Else
      Begin
        For I := 0 to Nr do
          Cpr[I,Minc] := Maxint ;
          Zc := Zc + 1 ;
      End ;
    End ;
  End ;

  Costl := 0 ;
  For I := 0 to Nr do
    For J := 0 to Nc do
      Costl := Costl + (Cv[I,J] * X[I,J] / 4.0) ;

  If (Arrays = 'y') Then
    Arl[1,M] := Costl / Sqr (Nr)
```

```
  Else Ar2[1,M] := Costl / Sqr (Nr) ;
End ;

Procedure Hist ;

Var
  Yy, Yyl     : Store ;
  A, B, C, D : Integer ;
  I, J, K, M : Integer ;
  Sh          : Packed Array [1..8,1..8,0..40] of Integer ;

Begin
  Clrscr ;
  TextMode ;
  TextColor (10) ;
  Gotoxy (20,14) ;
  Writeln ('  HISTOGRAM EVALUATION IN PROGRESS...') ;

  For I := 1 to Limit do
    For J := 1 to Limit do
    Begin
      If (Decision = 1) Then
      Begin
        C := Round (Nr / 2 * (I - 1) + 1) ;
        D := Round (Nc / 2 * (J - 1) + 1) ;
      End

      Else
      Begin
        C := Nr * (I - 1) + 1 ;
        D := Nc * (J - 1) + 1 ;
      End;

      Histblk (C, D, Yy) ;
      For K := 0 to 40 do
        Sh[I,J,K] := Yy[K] ;
    End ;

  ClrScr ;
  Gotoxy (20,14) ;
  Writeln ('  TEXTURE DISTANCE CALCULATIONS IN PROGRESS...');
  For I := 1 to Limit do
    For J := 1 to Limit do
    Begin
      If (J <> Limit) Then
      Begin
        For K := 0 to 40 do
        Begin
          Yy[K] := Sh[I,J,K] ;
          Yyl[K] := Sh[I,J + 1,K] ;
        End ;
        Dist (Yy, Yyl,I, J, 'y') ;
```

```
        End ;

        If (I <> Limit) Then
        Begin
          For K := 0 to 40 do
          Begin
            Yy[K] := Sh[I,J,K] ;
            Yy1[K] := Sh[I + 1,J,K] ;
          End ;
          Dist (Yy, Yy1, I, J, 'n') ;
        End ;
      End ;
End ;

Begin  {** Main Program **}
  ReadFile ;
  If (Decision = 1) Then
    Limit := (2 * Nm1) - 1
  Else Limit := Nm1 ;
  Cost ;
  Hist ;
  Thresh ;
  Labelling(Fr) ;
  Five :  Writeln ;
  Writeln ;
  Writeln ;
  Write ('  Do you wish to change threshold ') ;
  Readln (Ans) ;
  If ((Ans = 'y') Or (Ans = 'Y')) Then
  Begin
    Thresh ;
    Labelling(Fr) ;
    Goto Five ;
  End ;
  WriteFile ;
End .
```

BIBLIOGRAPHY

1. Aggarwal R. and Bacus J., "A multi-spectral approach for scene analysis of cervical cytology smears", J. Histochem. Cytochem. 25, 1977, pp. 668-680.
2. Ahuja N. and Rosenfeld A., "Mosaic models for textures", IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. PAMI-3, Jan. 1981, pp. 1-11.
3. Brodatz P., A photographic album for artists and designers, Dover Publication Inc., New York, 1966.
4. Carlucci L., "A formal system for texture languages", Pattern Recognition, Vol. 4, 1972, pp. 53-72.
5. Conners R. and Harlow C., "A theoretical comparison of texture algorithms", IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. PAMI-2, May 1980, pp. 204-222.
6. D'Astous F., Textural feature extraction in the spatial frequency domain, Ph.D Dissertation, Dept. of Systems and Design Eng., Univ. of Waterloo, 1983.
7. Ehrich R. and Foith J., "A view of texture topology and texture description", Computer Graphics and Image Processing, Vol. 8, Oct. 1977, pp. 174-202.
8. Feldman J. and Yakimovsky Y., Decision theory and artificial intelligence, Vol. 5, 1974, pp. 349-371.

9. Fu K. and Mui J., "A survey on image segmentation", Pattern Recognition, Vol. 13, 1981, PP. 3-16.

10. Gonzalez R. and Wintz P., Digital image processing, Addison-Wesley Publishing Company, London, Ontario, 1977.

11. Grasselli, A., Automatic interpretation and classification of images, Academic Press, New York, London, 1969.

12. Gupta J. and Wintz P., Multi-image modelling, TR-EE-74-24, School of E.E. Engg., Purdue University, 1974.

13. Gupta J. and Wintz P., "A boundary finding algorithm and its applications", IEEE Trans. on Circuits Syst., Vol. CAS-22, 1975, PP. 351-362.

14. Haralick, R., "Statistical and structural approaches to texture", IEEE Proc. Vol. 67, May 1979, PP. 786-804.

15. Haralick R. and Dinstein I., "A spatial clustering procedure for multi-image data", IEEE Trans. on Circuits Systems, CAS-22, 1975, PP. 440-450.

16. Haralick R., Shanmugan K., and Dinstein I., "Textural features for image classification", IEEE Trans. on Systems, Man, and Cybernetics, Vol. SMC-3, Nov. 1973, PP. 610-621.

17. Hiller F. and Lieberman G., Introduction to operation research, Holden-Day, San Fransisco, 1980.

18. Lu S. and Fu K., "A syntactic approach to texture analysis", Computer Graphics and Image Processing, Vol. 7, 1978, PP.303-330.

19. McCormick B. and Jayaramamurthy S., "Time series model for texture synthesis", International Journal of Computer and Information Sciences, Vol. 3, 1974, PP. 329-343.

20. Muerle J. and Allen D., "Experimental evaluation of techniques for automatic segmentation of objects in a complex scene", Pictorial Pattern Recognition, C.C. Cheng et al., , (Eds), Thompson, Washington, 1968, PP. 3-13.

21. Niemann H., Digital image analysis, Plenum Press, New York, 1979, PP. 77-122.

22. Niemann, H., Pattern analysis, Springer-Verlay Berlin Heidelberg, New York, 1981.

23. Pavlidis T., "Segmentation of pictures and maps through functional approximation", Computer Graphics and Image Processing, Vol. 1, 1972, PP. 360-372.

24. Raafat H., Textured image representation, classification, and segmentation, Ph.D Dissertation, Dept. of Systems and Design Eng., Univ. of Waterloo, 1985.

25. Rosenfeld A., "Image analysis: problems, progress and prospects", Pattern Recognition, Vol. 17, 1984, PP. 1-12.

26. Schachter B., Davis L., and Rosenfeld A., Scene segmentation by cluster detection in color space, TR-424, Computer Science Center, University of Maryland, Nov. 1975.

27. Serra J. and Verchery G., "Mathematical morphology applied to fiber composite materials", Film Science Technology, Vol. 6, 1973, PP. 141-158.

28. Shen H. and Wong A., "Generalized texture representation and metric", Proc. of the International Conference on Cybernetics and Society, 1980, PP. 695-703.

29. Shridhar M. and Rao M., "Segmentation of digitized images through label propagation method", Can. Elec. Eng. Journal, Vol. 10, 1985, PP. 61-68.

30. Sklansky J., "Image segmentation and feature extraction", IEEE Trans. on Systems, Man, and Cybernetics, Vol. SMC-8, April 1978, PP. 237-275.

31. Swain P. and Fu K., "On the applications of nonparametric techniques to crop classification problems", National Electronics Conf., Proc. 24, 1968, PP. 14-19.

32. Wechsler H. and Citron T., "Feature extraction for texture classification", Pattern Recognition, Vol. 12, 1980, PP. 301-311.

33. Weszka J., Dyer C., and Rosenfeld A., "A comparative study of texture measures for terrain classification", IEEE Trans. on Systems, Man, and Cybernetics, Vol. SMC-6, April 1976, PP. 269-285.

34. Wong A. and Vogel M., "Resolution dependent information measures for image analysis", IEEE Trans. on Systems, Man, and Cybernetics, Vol. SMC-7, Jan. 1977, PP. 49-61.

35. Zucker S., "Towards a model of texture", Computer Graphics and Image Processing, Vol. 5, 1976, pp. 190-202.

VITA AUCTORIS

1961	Born August 28, Moga, Panjab, India.
1977	Higher Secondary Part-II S.D. Higher Secondary School, Moga, Panjab, India.
1982	B.Sc. Engineering (Electronics and Communication) Kurukshetra University, Kurukshetra, Haryana, India.
1982	Design Engineer Hindustan Aeronautics Limited, Hyderabad, India.
1983	Teaching Assistant Panjab Engineering College, Chandigarh, Panjab, India.
1986	Master of Applied Science (Electrical Engineering) University of Windsor, Windsor, Ontario, Canada.