ANALYSIS METHODS

FOR

COST EFFICIENT SCENE INTERPRETATION

by

Moses Amadasun, B.Sc.

January, 1988

A thesis submitted for the degree of
Doctor of Philosophy of the University of London
and for the Diploma of Imperial College

Electrical Engineering Department
Imperial College
London, SW7 2BT

ABSTRACT

This thesis investigates automatic scene interpretation in relation to computational cost. The scenes of interest are essentially two-dimensional. Three specific areas of scene interpretation are considered:

1) texture characterization
2) image classifier design
3) image segmentation

Computationally-efficient and generally-applicable methods are developed to minimize cost.

Five properties of visual texture, namely: coarseness, contrast, busyness, complexity, and strength of texture, are approximated in computational forms, to produce five textural features for texture-based image classification. The cost involved in the computation of the features is very low, as they are easily computable and require little CPU process time, and the memory requirement is small. The features correlate well with human perceptual measurements in the rank ordering of a set of natural textures, and fairly well in indicating similarity between different textural patterns. The features produce better classification accuracy in two applications, compared with features from two classical texture analysis techniques [32,80].

With regard to texture-based partitioning of images, two additional features were developed. The application of these features in the segmentation of some test images produced satisfactory results.

In the area of classifier design, a distribution-free scheme was developed which is based essentially upon Euclidean distance. In the design, features are normalized such that their values are constrained to lie between zero and one inclusive. The contribution of each feature in classification decision making depends on its relative ability to separate the classes. The classifier was employed in three classification problems; it obtained classification accuracy comparable with that of the maximum likelihood classifier, but in terms of speed, it proved to be faster than the latter.

In respect of segmentation - a technique was developed which combines the region growing concept of seeking uniform areas in an image with the concept of agglomerative clustering. On the basis of a defined criterion, uniform neighbourhoods are located in an image, and their mean feature values are computed. These feature values are agglomeratively clustered to produce the mean feature vectors for the different categories present in the image. The mean vectors are in turn used to classify the image pixels.

In terms of implementation, two algorithms were designed for the segmentation scheme. One algorithm uses fixed neighbourhood size in seeking uniform areas in the image, while the criterion for uniformity is varied subject to some constraints. In the second algorithm, the uniformity criterion is fixed, while a quad-tree approach is used to vary the size of neighbourhood from one part of the image to another, depending upon the relative level of uniformity.

The segmentation results obtained for different kinds of test image confirm the feasibility of the approach. The method is fast, and requires only a small amount of memory; hence it can be used in real time.

DEDICATION

This piece of work is dedicated to the following people:

My kids

Osazuwamen and Osarenadoru

for distracting me with their lively games during very dull moments in the course of this work;

My wife

Lucy

for her understanding and encouragement through very difficult circumstances during the period of this work;

and my parents

for bringing me to the world.

ACKNOWLEDGEMENTS

It is my pleasure to record my gratitude to my supervisor, Dr. R.A. King, whose help and encouragement were a source of inspiration during this work. His patient guidance and direction has enabled me to develop my interest in this subject. The useful criticisms and suggestions from members of Dr. King's research group were very helpful. I acknowledge the interest of the members of this group.

I would like to thank the Imperial College Centre for Remote Sensing for allowing me to use their facilities for most of the work. In particular, I am grateful to the Centre's System Manager, Dr. C. England, for his help throughout my stay in the Centre. The cooperation of other researchers in the Centre is acknowledged.

I appreciate the contributions of the following individuals in their provision of some of the images used in the experiments. They are: Mr. T.G. Tekie, for the Landsat multispectral scanner image and the corresponding "ground truth" information; Mr. J. Brunner, for the satellite thematic mapper image; Mr. E. Amiaz, for the aerial thematic mapper image and the corresponding "ground truth" map; and Mr. A. Blakeman, Ministry of Agriculture, Cambridge, for the black-and-white aerial picture of agricultural fields and the corresponding "ground truth" information.

I am especially indebted to Miss C.F. Lane, for her help in typing the script.

This work was sponsored by the Federal Government of Nigeria through a postgraduate scholarship which was awarded to me, and I wish to thank the Nigerian taxpayers in this respect.

Finally, I am grateful to all my colleagues at Imperial College who in one way or another contributed to the successful completion of this research work.

TABLE OF CONTENTS

LIST OF FIGURES

PAGE

LIST OF TABLES

PAGE

CHAPTER ONE

INTRODUCTION

1.1 Introduction to Scene Interpretation

Scene interpretation or analysis is a major problem area in the image processing field. It is sometimes referred to as image analysis, image recognition, or image understanding. Scene analysis deals with the automatic interpretation of the image of a scene in order to make a decision. The interpretation is based on characterization knowledge, and this knowledge in turn requires the analysis of the basic properties, characteristics or attributes of the contents of the scene. The output of a scene interpretation system is essentially a description of the contents of the image of the scene, or an assignment of the image, or part(s) of it, to particular class(es).

Scene analysis techniques have diverse applications. Some of the application areas are enumerated below:-

1) Remote sensing - vegetation mapping, land-use classification, and monitoring of the environment from aerial and satellite images of the earth surface

2) Photogeology in mineral exploration

3) In agriculture, for example

   (i) Mapping and classification of crop types in aerial photographs for the purpose of agricultural yield estimation

(ii) Mapping and classification of different forest plants in remotely sensed images for the effective management of forest resources

4) Biology and medicine; for example, tumour detection and blood cell counting

5) Robotics and industrial automation

6) Military problems - target detection and recognition in aerial surveillance

1.1.1 Scene Analysis System

The analysis of the image of any scene generally consists of two aspects. The first is the partitioning of the image into its component parts - that is, into regions corresponding to the objects or categories present in the scene. It is essentially the grouping together of pixels having similar properties. The second part is to classify or identify each of these component regions, or at least the region(s) of interest. The first part in the analysis process is generally called "segmentation", while the second is more specifically referred to as "identification". A block diagram of a basic scene analysis system is shown in Fig. 1.1.

The segmentation aspect generally involves working at local level (i.e. at pixel level or over small neighbourhoods) in the digital image, while identification is carried out on subimages or images. In segmentation, no information external to the image may be

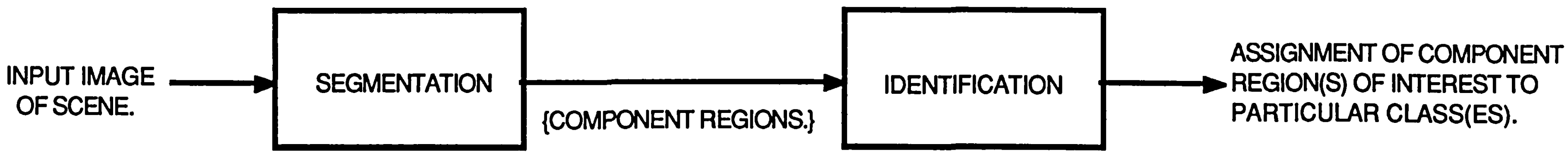


Fig. 1.1 A Basic Scene Analysis System.

required, but in identification, external information may be needed. This is often the case in remote sensing and agricultural applications - the so-called "ground truth information".

In general, there are three distinct, but not necessarily independent, phases that are associated with scene analysis; namely, characterization, abstraction and generalization [59].

(i) Characterization

This involves observing the attributes, or characteristics, of the content of the scene, quantifying these attributes, and extracting or selecting useful features from the set of observations.

(ii) Abstraction

This is the formulation of decision rules for classification or segmentation. It is basically a decision-making phase.

(iii) Generalization

This is essentially the evaluation of the proposed solution. It is the phase for system performance evaluation.

Almost all the problems encountered in scene analysis are associated with the first two phases; namely, the extraction of useful features, and the formulation of decision rules.

1.1.2 Feature Extraction

One important issue in the extraction of features for the automatic analysis of a scene is the problem of texture characterization. In the analysis of scenes, humans use three types of information. These are: spectral, textural and contextual information. However, in a machine environment, context is usually difficult to implement. The two important factors are spectral information and texture [32]. Spectral information includes brightness and colour. It represents brightness in the cases of black-and-white pictures and monochrome images; and in the case of a multiband image set, it represents the pixel gray levels in an individual band. Brightness is directly conveyed by the gray levels of the image pixels; while in the case of coloured image, spectral information is depicted by the tristimuli of red, green and blue.

Texture, on the other hand, is a much more difficult concept. Literally, texture refers to the arrangement of the constituent components of a material. In image analysis, however, one is concerned with visual texture. This is a function of the spatial distribution of tonal values (gray tones). The properties of texture which humans use in distinguishing between textures are well known. The most important ones are: coarseness, contrast, busyness, complexity, shape, directionality, and strength of the texture.

Perceptually, coarseness relates to the sizes of the constituent elements - the so-called basic patterns, or texture primitives. In coarse textures, the primitives have relatively large sizes, and consequently, there is a relatively high level of spatial uniformity in intensity or tonal values. Contrast is dependent on the dynamic range of tonal values, as well as the amount of local variations in the values. Busyness depends on the spatial frequency of change from one intensity level to another, while complexity is related to the amount of visual information present in a texture. Directionality refers to the angular orientation of the texture primitives. Shape refers to the geometrical shape of the primitives, while the strength of a texture depends on the degree of distinguishability of the texture primitives from one another.

Texture, unlike brightness, is not directly conveyed by individual image pixels. It is a neighbourhood property. The difficult problem is to derive features from the gray tones of the image pixels that would convey information about the above mentioned textural properties - that is, to characterize texture from the gray tones of the image pixels.

1.1.3 Formulation of Decision Rules

There are generally two sets of decision rules: those for identification or "pure" classification, and those for segmentation; but identification rules can also be incorporated into the segmentation process. For the purpose of identification, decision rule formulation is essentially the problem of classifier design. Various designs exist in the literature, but the majority of them use the approach of statistical decision theory, employing classical criteria such as the maximum likelihood rule, min-max rule, Bayesian rule, and a host of linear discriminant functions.

Some other designs use normalized distance measures, or even simple distance metrics, such as the Euclidean distance. There are also various approaches to decision rule formulations in relation to segmentation, such as rules based upon edge detection, clustering, feature histograms, region growing, and so on.

1.2 Scene Analysis and Computational Cost

The computational cost associated with any technique in the scene analysis field may be measured in terms of one or more of the following considerations:-

(i) The Amount of Computation Performed

This may be measured quantitatively by the computation time taken, i.e., the CPU process time. If a large amount of computation is involved, the process is likely to take a long time, resulting in high cost.

(ii) Memory Requirement of the Method

Large memory requirement may demand the use of special hardware for storage purposes, increasing the cost of the analysis system. Alternatively, virtual memory may be used. However, the use of virtual memory generally results in the machine spending more time in paging than in actual computation, and thereby increasing the overall process time.

(iii) Complexity of the Technique

Complex techniques are generally difficult to implement, and such methods, in most cases, are likely to involve a large amount of computation and/or memory.

(iv) General Applicability of the Method

Problem-dependent methods are generally ad hoc approaches, which are hardly useful for real-time applications. This is because, for each application, a new method may be required, or a substantial modification to the existing technique may need to be made. This results in increased complexity of the systems that have been designed, and an attendant rise in overall cost.

1.3 Scope of Work

The work reported in this thesis is concerned with the development of cost-effective techniques in three areas of scene analysis. The scenes of interest are two-dimensional: for example, X-ray images, radiographs, photomicrographs, aerial and satellite images of terrains, etc. Specifically, the work is in the areas of textural feature extraction, image classifier design, and image segmentation.

1.3.1 Textural Feature Extraction

In the area of textural feature extraction, a set of five features was developed for texture-based image classification. The features, though statistical, were developed from the conceptual relationship of some textural properties to spatial changes in intensity. These textural properties are: coarseness, contrast, busyness, complexity, and strength of texture.

The extent to which the features approximate the properties, and the extent to which certain combinations of the features approximate, or relate to, human perception of textures, was investigated in two experiments, also involving human perceptual measurements. The features were also applied in two image-classification problems, and the results were compared with those obtained using features from two classical texture analysis techniques; namely, the spatial gray level dependence method [32], and the gray level difference method [80].

Two features were also developed for the textural segmentation of images. The segmentations of some textured images using the features are shown.

1.3.2 Design of Image Classifier

For the classification of images, a distribution-free scheme called a "weighted-feature minimum distance classifier" was developed. The design is based essentially upon the Euclidean distance metric, but the features are normalized in such a way that they are constrained to have values between zero and one inclusive. Furthermore, in the design, each feature is weighted such that its effectiveness in the classification decision making depends on its relative ability to separate the classes. A measure of separability used for weighting is the distance between mean values of features for the classes - the so-called "contrast criterion" [43]. The performance of the classifier was compared with the maximum likelihood and Euclidean-distance classifiers in three applications, two of which involved texture classification, and the other, classification of agricultural land-cover types using spectral signatures.

1.3.3 Development of Image Segmentation Scheme

A pixel-classification based segmentation technique has been developed. This is a hybrid scheme, which combines the concepts of region growing and clustering to partition an image into a given number of categories or regions. Uniform neighbourhoods are first located in an image. The mean feature values of these neighbourhoods are then agglomeratively clustered to produce the mean feature vectors for the categories. The feature vectors are in turn used to classify the image pixels.

There are two variants of the scheme with regard to implementation. In the first case, i.e. Algorithm I, a fixed neighbourhood size was used in seeking areas of uniformity in the image; whilst in the second case, Algorithm II, an arrangement in the form of a quad-tree was used to vary the size of neighbourhoods from one part of an image to another, depending on the degree of uniformity.

The segmentation results obtained from both algorithms were not only similar, but also very satisfactory. However, Algorithm II has been found to be more accurate and takes less computation time. Hence, it is the better algorithm.

The test images include a human passport photograph, an X-ray image of a human wrist, a composite textured image, an outdoor scene, and two satellite images of terrains. In the segmentations, either spectral features, textural features, or a combination of both, were used, depending on the image. The segmentation technique is fast, requires small memory, and can partition an image into any given number of categories or regions depending upon the desired level of detail.

1.4 Outline of Thesis

The thesis is divided into seven chapters and five appendices. In Chapter One, a general introduction to the subject of scene interpretation is given. Chapter Two presents a review of some of the methods that have been developed or suggested in the literature, specifically in the areas of texture analysis, image segmentation and image classifier design. Chapter Three describes the perception-related texture features that have been developed. Also presented in this chapter are the results of the experiments performed to assess the extent to which the features approximate or relate to human perception of textures.

The application of the features developed here in image classification, and a comparison with two classical texture analysis methods, are reported in Chapter Four. The features developed for textural segmentation, and their application in the supervised segmentation of three textured images, are also presented in this chapter. The design of a distribution-free classifier (the weighted-feature minimum distance classifier), and some applications, are given in Chapter Five. In Chapter Six are descriptions of the segmentation scheme that has been developed and the two variants of the algorithm. The results of segmentations are also included. Overall conclusions, and some suggestions for further work, are discussed in Chapter Seven.

In Appendix A-1, tables showing the frequencies of ranks and similarity assignments, obtained from the human perceptual measurements, are presented. A description of the minimum error-rate (maximum likelihood) classifier is given in Appendix A-2. The spatial gray level dependence method and the gray level difference technique of texture analysis are discussed in Appendix A-3, while in Appendix A-4 a proof is given for the variance updating formula given in Chapter Five. Finally, in Appendix A-5, six Fortran programs are shown. The first one is for the computation of the textural features that have been developed; and the second is for the implementation of the classifier that has been designed. The remaining four are for the two segmentation algorithms. For each algorithm, there are two programs. One is for segmentation of a three-band multispectral image using the pixel gray levels in the three bands as features. The other is for the segmentation of a black-and-white or monochrome image. Segmentation may be carried out on the basis of texture (in which case the two features developed for segmentation are used); or on the basis of brightness (i.e. using pixel gray levels as features); or on the basis of a combination of both texture and brightness.

CHAPTER TWO

REVIEW OF SCENE ANALYSIS TECHNIQUES

2.1 Introduction

The design of a scene analysis system generally depends upon the intended application and/or the form of the result that is expected from the system. A complete system usually consists of two aspects: segmentation and identification. Segmentation is the partitioning of the image of the scene into component regions corresponding to the objects or categories present in the scene; while identification is the assignment of the region(s) of interest to particular class(es). This assignment is carried out by a decision-making process called a classifier.

Both segmentation and identification require the observation of the basic properties of the contents of the scene, and the extraction of useful information from them - a process referred to as "feature extraction".

Various methods have been suggested, or developed, in the areas of feature extraction, image segmentation, and design of classifiers. In the area of feature extraction, the focus is on texture characterization; that is, the derivation of textural features from the gray levels of image pixels. Some of the attempts already made in this regard are reviewed in section 2.2. Subsection 2.2.1 describes some statistical techniques; subsection 2.2.2 discusses some structural methods; while statistical-structural approaches are reviewed in subsection 2.2.3.

In section 2.3, some approaches to image classifier design are discussed. Their relative merits and demerits are also mentioned. The review of some image segmentation schemes is contained in section 2.4. Subsections 2.4.1 and 2.4.2 describe edge detection based approaches and non-edge detection based techniques respectively.

2.2 Texture Characterization

Texture is an important property that humans use in analysing a scene, or distinguishing one scene from another. It is of particular importance in the analysis of natural scenes, as most natural environments consist of textured surfaces. The development of computational measures for the automatic discrimination between different textural patterns is called texture analysis or characterization. It is a subject that has received considerable research effort. Various techniques have been developed.

Haralick [30] groups the various methods into three categories: statistical approaches; structural methods; and statistical-structural techniques. While the statistical methods are generally applicable, the structural approaches, in most cases, can only be applied if the constituent elements (called the texture primitives or basic patterns), and also some placement rules, can be extracted. In general, statistical techniques are more suitable for microtextures, while structural methods are more relevant in the case of macro-textures.

2.2.1 Statistical Techniques

Statistical approaches may be subdivided into two classes: model based and non-model based methods.

(i) Non-Model Based Statistical Methods

These are statistical techniques in which no stochastic or probabilistic model is assumed for the texture field. Some methods belonging to this class are the "spatial gray level dependence method" (SGLDM) [32]; the "gray level run length method" (GLRLM) [23]; and the "gray level difference method" (GLDM) [80]. Other methods in this category are the "neighbouring gray level dependency method" (NGLDM) [72]; the "textural edgeness technique" [67]; and the frequency domain based approaches of Fourier power spectrum and autocorrelation.

In the SGLDM method, a matrix called a gray level co-occurrence matrix is computed, in which an entry $p(i,j)/d,\theta$ is the probability of finding two gray levels i and j in the image, separated by distance d and in angular direction θ. Four matrices are produced, one for each value of θ of 0°, 45°, 90°, and 135°. A number of textural features are derived from these matrices, out of which four are considered to be most useful.

The GLRLM uses the run lengths of gray levels. A gray level run length primitive is a maximal collinear connected set of pixels, all having the same gray level. Four matrices are produced, one for a given angular direction in which an element $p(k,\ell)$ is the number of times there is a run of length ℓ, having a gray level of value k.

The gray level difference method makes use of the differences between the gray levels of pixels. A matrix is formed in which an element $p(i)/d,\theta$ is the probability of obtaining a difference of value i between the gray levels of two pixels separated by distance d from one another and in angular direction θ. Four matrices are obtained, one each for θ value of 0°, 45°, 90°, and 135°. From these matrices, five textural features are derived.

Instead of finding co-occurrences of gray levels in four directions, as in [32], in [72] co-occurrences were found in a neighbourhood. In this approach, a neighbourhood is centred on a pixel and a count is made of the pixels having the same gray level as the centre pixel. This count gives the NGLDM number, denoted as s. A matrix of gray levels and NGLDM numbers is formed, and some features derived from the matrix.

The method of textural edgeness characterizes texture in terms of the number of edges present per unit area, where we define an edge (the so-called microedge) as restricted to the edges within a texture field, rather than the edges separating different texture fields. A similar approach was used in [73] for pulmonary disease identification.

The Fourier power spectrum and autocorrelation techniques essentially use spatial frequency to characterize texture. The spatial frequency spectrum contains information about the texture of an image, because fine textures are rich in high frequency components, whereas coarse textures are rich in low frequency components. Some frequency domain techniques are reported in [10,19,27,36]. In [10], autocorrelation was used, while the remaining three approaches used Fourier methods. A discussion of autocorrelation methods is also given in [61,Chapter 17].

(ii) Model Based Statistical Techniques

In these approaches, a model is assumed for the texture field. The classical examples of this class of technique are the autoregression (AR) models of different kinds. Essentially, the AR methods work by making use of the degree to which a pixel gray level can be estimated given the gray levels of the neighbouring pixels. A set of parameters are estimated from the image data. These estimates are then used for texture classification, segmentation or synthesis. McCormick et al [46] first used this idea in texture synthesis. They used a 1-D time series model. Deguchi and Morishita [17] developed a 2-D autoregression model for texture classification and/or segmentation. Mitrakos et al [49,50] developed a technique called the "composite source model" for image partitioning and coding. In their method, two components called the C and E components are derived from a Gaussian-Markov model of order p. Using maximum likelihood estimates of p parameters and the variance of the residuals, they successfully performed image segmentation and image coding. Other model based techniques are given in [40,51,62].

The methods of textural edgeness, autocorrelation and Fourier power spectrum essentially characterize one aspect of texture; namely, coarseness. An added disadvantage of the frequency domain methods is the asssumption that the image function is periodic, which certainly is not true. The spatial domain approaches are generally better, but some of them may require large memory - for example, the SGLDM - due to the need to store four matrices. The main advantage of the model based techniques is that they can also be used for texture synthesis, but, as is pointed out in [30], their effectiveness is mainly restricted to microtextures.

2.2.2 Structural Approaches

A fundamental technique in almost all structural approaches is the extraction of texture primitives. A primitive may be defined as a connected set of pixels characterized by some predefined properties, e.g. shape. The pixel, with its gray level attribute, is the simplest primitive of a texture field. After defining the primitives, they are extracted from the image using suitable procedures. The spatial interactions between the primitives are then examined to characterize the texture.

Structural analysis methods include the texture model of Carlucci [8], which uses primitives of line segments, open and closed polygons, in combination with some rules that are given syntactically in a graph-like language. Another example is the tree grammar syntactic method of Lu and Fu [45]. They view rules of spatial placement of texture primitives as production rules of a specific grammar. Classification of a given texture then reduces to the determination of whether the texture field exhibits a pattern which belongs to a given language. Zucker [83] also developed a method similar to that in [45].

Other structural methods are: the structural element technique of Serra et al [68]; the structural analysis approach of Tsuji and Tomita [76]; and the technique proposed by Vilrotter et al [78]. In the approach of Vilrotter, matrices that are in some respects similar to gray tone co-occurrence matrices and called "edge repetition arrays" (ERAs) are first defined, and then computed from the image. The computed ERAs give an initial and partial description of texture elements (texels). From the ERAs, texture primitives, as well as their spatial interrelationships, are determined. Then, on the basis of the extracted primitives and determined interrelationships, texture classification is achieved.

Structural approaches in general have the advantage of being able to capture the shape aspect of texture, but the detection or extraction of primitives in real textures can be quite a problem. Furthermore, structural techniques are not only computationally expensive, but are also very complex in terms of implementation.

2.2.3 Statistical-Structural Methods

These are methods that tend to combine both statistical and structural approaches. As stated in [30], they are structural in the sense that they also involve the extraction of primitives, and statistical in the sense that spatial interactions between the primitives are measured by probabilities. The generalized co-occurrence matrices method of Davies et al [16] is a classical example of this group of techniques. Other examples are [48,71]. They share in the relative merits and demerits of statistical and structural techniques.

The various texture analysis methods, and their relative advantages and limitations, are discussed in [26,30], while information about the relative performance of some of the methods is given in [13,80].

2.3 Image Classifier Design

The assignment of an image or part(s) of it to a particular category, or categories, requires a good decision making process. A major consideration in the design of a classifier is the accuracy of decision making. The other considerations are the complexity of the design, and the cost of computation. Approaches to image classifier design may be divided into two groups. There is the group of classifiers in which classification decisions are based on statistical decision theory. The second group of methods makes use of simple similarity measures or distance metrics.

2.3.1 Methods Using the Approach of Statistical Decision Theory

The majority of classifiers belong to this group. These classifiers employ the classical criteria used in statistical decision making. Such criteria include the Bayesian rule, the maximum likelihood rule, the min-max rule, the Neyman-Pearson rule, and a host of linear discriminant functions - for example, the Fisher's linear discriminant. These criteria, and various types of discriminant functions, are discussed in [18,22]. The design of classifiers employing statistical decision making is also described in [14]. The criterion to be used in a given situation may depend on the risk of, or cost involved in misclassification. The particular circumstances in which one criterion may be employed in preference to another are explained in [22].

Statistical classifiers have two main disadvantages:-

(i) The statistical properties of the image are not closely approximated by those of the model; for example, the classifiers generally assume a probability distribution (usually normal) for the image data. Another assumption inherent in the use of statistical criteria is that the samples in the data set are independent. It is well documented that image data in most cases are not normally distributed [15]. Therefore, the assumption of normality in these designs is inappropriate, because their normality assumption is violated by the data, and this introduces a substantial amount of error. Thus, their accuracy would be poor in applications where the image data deviates substantially from the normal distribution model. For instance, better results were obtained in [20] in the classification of agricultural crop types from aerial photographs using distribution-free methods than by using a linear discriminant function approach; this is because the latter method assumes normality, although the data set were not normal. Furthermore, the inherent assumption of independence between image samples is not true. There is some degree of dependency in images. Therefore, because of the normality and independence assumptions made in these classifiers, they may not have general applicability.

(ii) This group of classifiers is generally complex in design, and also computationally expensive. The high cost of computation may be due to the inversions and multiplications of matrices that are involved in the decision making process.

2.3.2 Methods Employing Simple Distance Metrics

For this group of techniques, the most commonly used measure of similarity is the Euclidean distance. Some other measures of similarity (e.g. the Tanimoto coefficient) that these classifiers may use are also described in [18]. The Euclidean-distance classifier is simple in design, has comparatively high speed, and is generally applicable. However, it has the following disadvantages:-

(i) The accuracy is comparatively poor. This is due to the dominance of certain features in distance calculations merely because of their large numerical values. A relatively large difference in value between two classes for features that generally have large numerical values may not convey as much difference (say perceptually) between the classes, as small differences for features that have low numerical values.

(ii) A Euclidean-distance classifier does not take into consideration the abilities of the individual features to discriminate between the classes. However, in most cases, there is a higher degree of separability between the classes using some features than others. For instance, in a multispectral (multiband) satellite image of a terrain, two regions or categories may be highly distinguishable from one another in one band, while such distinction may not be possible in another band.

The first disadvantage is generally minimized by a process of feature normalization, while for the second, a weighting of the features dependent upon their relative abilities to discriminate between the classes is needed. A combination of the two processes would lead to improved classification accuracy. However, it is desirable that this improvement in performance does not result in significant increase in computational cost.

2.4 Image Segmentation Methods

There are many approaches to image segmentation in the literature. However, most of the techniques are essentially ad hoc, as there is no general theory of segmentation. As stated in [33], "the methods essentially differ from one another precisely in the way they emphasize one or more of the desired properties, and in the way they balance or compromise one desired property against another." Nevertheless, image segmentation techniques may be divided into two broad groups: edge detection based schemes and non-edge detection based approaches.

2.4.1 Edge Detection Based Methods

Segmentation techniques in this group seek in an image for points of significant change or discontinuity in feature activity; the feature generally used is the gray level of the image pixels. An edge is defined as a significant change in intensity (gray) level. A connected set of the edges gives the boundaries between an object and its background and/or between the different objects or regions in the scene.

There are two main classes of edge detection methods: the enhancement/thresholding techniques; and the methods of edge fitting. A third group consists of those methods which use some other kind of criterion for determining edge points.

(i) Enhancement/Thresholding Methods

In these methods, discontinuities in feature activity are enhanced or accentuated by some spatial processing involving the use of differential operators. Such operators include the Prewitt, Roberts, and Sobel operators, or their modified forms. These operators are used to perform discrete differentiation of the image array to produce a gradient field. An edge is deemed to exist at an image point if the gradient or magnitude of change in feature activity at the point is sufficiently large, and greater than some specified threshold.

(ii) Edge Fitting Techniques

These methods employ template matching operators. These are sets of masks representing discrete approximations to ideal edges of various orientations. Some operators of this kind are the compass gradient introduced by Prewitt [63] and the Kirsch operator [41]. An edge of the particular type or orientation defined by the mask is deemed to exist at a given image point if there is a sufficiently high degree of "fit" between the image and the mask centred on the point.

(iii) Other Edge Detection Schemes

These are methods which do not involve spatial differentiation or edge fitting, but rather use some other criterion for determining edge points. Some segmentation approaches belonging to this category are the equal-means and equal-variances hypothesis testing technique of Yakismovsky [82], and the zero-slope hypothesis testing method of Haralick [29]. The method in [82] assumes a normal distribution for regions. Statistical hypothesis testing is used to locate edge points. Edges are declared to exist between pairs of contiguous and exclusive neighbourhoods if the hypothesis of equal means and equal variances between them has to be rejected. Haralick's method involves fitting a plane to the neighbourhood centred on a pixel and then testing the hypothesis that the slope of the plane is zero. Edge pixels are the ones between neighbourhoods for which the zero-slope hypothesis has to be rejected.

A discussion of some edge detection approaches to segmentation and their relative performances is given in [1]. In general, edge detection methods are only good for brightness or gray tone dominated images in which there are clear differences in brightness between objects and background. They are poor performers on textured images, and for complex scenes in which boundaries are best established using a combination of features instead of only intensity. Segmentation schemes employing edge detection are also described in [25,66].

2.4.2 Non-Edge Detection Techniques

Methods in this class group together pixels having similar attributes or properties. Essentially, they assign or classify image pixels to one category/region or another in the image. These segmentation approaches include histogram methods, clustering methods, and various region growing schemes. A general advantage of these approaches is that they are capable of using more than one feature in the segmentation process.

(i) Histogram Methods

These techniques involve the construction of a feature histogram. Thresholds are selected in the histogram to partition the image. A particular category or object in the image would correspond to all pixels in the image having feature values between any two sets of thresholds. They are essentially methods in which clustering is done in measurement space, and this is then mapped on to the image domain to produce segmentation. As in any clustering approach, an inherent assumption is that feature values of pixels belonging to one category would be similar to one another, but significantly different from those of pixels belonging to other categories. Hence, the accuracy of histogram techniques depends on how well the objects or categories of interest in the image separate into distinct measurement space clusters; that is, on the modality of the histogram.

If there is no clear distinction between clusters in measurement space, the histogram may be unimodal or even flat, and one may not be able to partition the image because of the difficulty in setting thresholds. Some approaches for the construction of histograms with enhanced modality are described in [11,42,47,57,58,81]. Histogram methods are also described in [25,66] for the partitioning of an image into regions of different average brightness, or separating an object from its background. The use of more than one feature in histogram-based techniques requires the construction of multi-dimensional histograms. Histograms of this kind are difficult to construct, and generally require considerable memory; and the selection of thresholds is a more difficult problem. Some multi-dimensional histogram schemes are described in [24,55].

(ii) Clustering Approaches

In general, clustering procedures involve the iterative grouping and/or regrouping of the image pixels subject to a minimization or maximization of a given criterion function. In some clustering methods, the number of groups (clusters) into which the image data is to be partitioned is specified, while in some others, the iterative process is stopped when the criterion function being maximized or minimized reaches a critical value.

The most commonly used clustering techniques in the partitioning of image data are of the ISODATA [2] type. The criterion function may be the minimization of the least square or mean square errors between the samples in a cluster and the cluster mean. Some clustering approaches to image segmentation are [6,12,31]. In [12], the criterion function that was maximized was the ratio of the inter-cluster scatter to intra-cluster scatter.

The subject of clustering, different kinds of clustering procedures, and the various criteria that may be used in clustering algorithms, are discussed in [18,Chapter 6]. Some investigators [3,7,75] have also introduced the fuzzy sets concept into the clustering process.

Clustering techniques are generally very expensive in terms of computation. Another disadvantage is that they may also require considerable memory. Hence, their use for real time applications is rather limited.

(iii) Region Growing Schemes

The techniques in this group may be divided into two types. In one set of methods, the general concept is to identify or locate uniform areas in an image. Such areas, or core points, are considered to belong to particular objects or categories in the image. Regions are then grown from them. The second type of region growing approach consists of those techniques in which the underlying principle is graph theory. Their implementation is usually based upon hierarchical or pyramidal data structure.

In the first kind of approach, pixels spatially adjacent to a core region that are similar enough to it are merged with the region, and its mean feature values are then updated. This process is continued until all pixels have been assigned to one region or another. The process may also involve the merging of contiguous regions that are similar enough.

One main disadvantage of this kind of scheme is the production of a large number of regions in the segmentation; also, areas corresponding to identical objects or to the same category at different locations in the scene may be labelled differently. Another difficulty is the determination of the criterion or criteria by which one judges similarity. The criteria vary from one scheme to another [4,35,44,53,54,60].

The graph-theoretic methods generally involve the mapping of image points on to nodes in a graph. Narenda and Goldberg [56] used directed graphs to define regions after an edge detection operation. Morris and Constantinides [52] mapped an image on to a weighted graph, and a minimum spanning tree of this graph was used to define regions or edges in the image. Spann and Wilson [70] combined a quad-tree representation of an image with a parametric classifier in a clustering framework to produce segmentations. The boundary following algorithm [71], and the split-and-merge technique of Horowitz and Pavlidis [34], may also be considered as belonging to this group of methods.

In the split-and-merge method, an image is divided into a number of square blocks. Blocks that are considered uniform, spatially connected, and are similar enough, are merged together to form regions. The mean feature values of the regions are updated in the process. Non-uniform blocks are split and their component parts merged with the nearest appropriate region.

In addition to the large memory requirement of graph-theroetic techniques, the split-and-merge approach also produces jagged or squarish boundaries. Graph-theoretic schemes, and various data structures that may be used for their implementation, are described in [59].

Image segmentation in general, and the various approaches to the problem, are discussed in [21,33,64,65].

CHAPTER THREE

TEXTURAL FEATURES CORRESPONDING TO TEXTURAL PROPERTIES

3.1 Introduction

Texture is an important item of information which humans use in analysing a scene. It is particularly useful in the analysis of natural environments, as most natural scenes consist of textured surfaces. Literally, texture refers to the arrangement of the basic constituents of a material. In a digital image, texture is depicted by spatial interrelationships between, and/or spatial arrangement of, the image pixels. Visually, these spatial interrelationships, or arrangement of image pixels, are seen as changes in the intensity patterns, or gray tones. Thus, in automatic analysis, information about texture has to be derived from the gray tones of the image pixels.

A number of texture analysis methods have been proposed, some of which [23,32,36,46,67,80] are frequently referred to in the literature. These and other methods are discussed in the review. A major disadvantage of almost all of these approaches is that they do not have general applicability - that is, they cannot be applied to different classes of textures with reasonable success. For instance, while the statistical techniques are generally good for microtextures and are poor performers on macrotextures, the reverse is the case for the structural techniques. Another disadvantage of some of the existing methods is the computational cost involved, either in terms of memory requirement, computation time or implementational complexity.

The human perception mechanism, in comparison, seems to work well for almost all types of textures. The properties which humans use to discriminate between different textural patterns include coarseness, contrast, complexity, "busyness" or fineness, shape, directionality and strength of the texture. Therefore, for general applicability of developed texture measures, and also for improved performance in automatic texture classification, it is relevant that measures reflect or represent to some extent some of the above mentioned textural properties. Tamura et al [74] did some work in this direction, but they used already developed features, only modifying a given feature or combining some features in one way or another to have a close relationship to a specific property.

Furthermore, the extraction of the features may be computationally expensive, as diverse analysis techniques are involved in their derivation.

Some other investigations carried out in the study of human perception of textures are reported by Julesz [37-39]. However, in these investigations, the aim was not the development of texture measures, but rather the study of the extent to which one can just perceive differences in artificially generated textures when all familiar cues are removed. He concluded that the discrimination of textures depends mostly on the difference in second-order statistics.

In the present approach, an attempt is made to develop completely new computational measures corresponding to some textural properties, so as to ensure general applicability, while at the same time minimize the cost of computation. Five textural properties, namely: coarseness, contrast, busyness, complexity and strength of texture, were approximated in computational forms. The computational form for each property was derived by expressing a perceptual description of the property in terms of spatial changes in intensity or gray tones. In a digital image, information about spatial changes in intensity can be obtained by looking at the differences between the gray tone of each image pixel and the gray tones of its surrounding neighbours. Therefore, central to the development of the reported features is the computation of a one-dimensional matrix for an image, in which the ith entry is a summation of the differences between the gray level of all pixels with gray level i, and the average gray level of their surrounding neighbours. The computational measures are derived from this matrix.

A discussion of the five textural properties and their conceptualized relationships to changes in gray tones is presented in Section 3.2. A description of the matrix, which shall be referred to as the "Neighbourhood Gray Tone Difference Matrix" (NGTDM) follows in Section 3.3. The computational approximations to the textural properties are developed in Section 3.4. In Section 3.5, the approximations of the texture measures to textural properties are experimentally evaluated.

3.2 Description of Textural Properties and their Relationship to Changes in Gray Tones

(a) Coarseness

Coarseness is the most fundamental property of texture, and in a narrow sense, it is used to imply texture. It is the size of the basic patterns or primitives making up a texture that determines the degree of coarseness of the texture. In coarse textures, the texture primitives are relatively large in size. As a result, coarse textures tend to possess a high degree of uniformity in intensity even over fairly large areas. Therefore, for such textures, the difference between the gray tone of a pixel and the average gray tone in its neighbourhood, even for fairly large neighbourhood size, would generally be small.

(b) Contrast

Perceptually, an image is said to have a high level of contrast if areas of different intensity levels are clearly visible, such as black and white patches. If, in an image, the differences between the different intensity levels are made smaller - as would happen in gray scale shrinking - the less distinguishable are areas corresponding to different levels of intensity, and hence the less is the contrast. Conversely, the contrast would be increased if the gray scale is stretched. In such a situation, the change in intensity between areas of different intensities would appear more abrupt, resulting in the perception of sharp edges.

However, apart from dynamic range of gray scale, the amount of local variations in intensity may also influence the contrast of an image. For example, consider two checkerboard patterns in which the patterns consist of equal sizes of black and white patches, but the size of the patches in one checkerboard is half that in the other board. For these two patterns, the dynamic gray scale range is the same, yet the pattern with the smaller sizes of patches would tend to give the illusion of higher contrast. This is because the spatial rate of change in intensity, and consequently the amount of local variations in intensity, is higher. In this kind of situation, the gray level of an image pixel may be substantially different from those of its neighbours.

(c) Busyness

A busy texture is one in which there is rapid spatial change from one intensity level to another. The spatial rate of change in intensity in an image depends primarily upon two factors. One is the spatial frequency of change from one intensity level to another, while the second is the magnitude of these changes. If the changes are very small in magnitude, they may not be visually noticeable and a high level of local uniformity may be perceived. Similarly, if the spatial frequency of change is low, a high degree of local uniformity in intensity may still be perceived, even if the magnitude of the changes is large. While the spatial frequency of change from one intensity level to another reflects the level of busyness, the magnitude of these changes depends upon the dynamic range of gray scale, and thus relates to contrast. Therefore, a suppression of the contrast aspect from the information about spatial rate of change in intensity may convey information about texture busyness.

(d) Complexity

The complexity of an image relates to its visual information content. A texture is considered complex if the information content is high, and this is generally the case when the texture comprises many patches of different average intensities. In textures that are made up of large primitives, the number of patches with visually noticeable different average intensity would tend to be few compared with textures having small sizes of primitives. Also, a texture with a large number of sharp edges tends to be complex, compared to one with few edges.

The number of patches, as well as the number of edges, depends upon the spatial period of repeating patterns, while the sharpness of the edges depends upon the dynamic gray scale range. Thus, complexity of a texture has some relationship to its level of busyness as well as to the contrast.

(e) Texture Strength

The term "strength of texture" is a difficult concept to define concisely. It appears that a texture is referred to as being strong when the texture primitives, i.e. the basic patterns making up the texture, are clearly visible or identifiable. Such textures generally tend to look either very attractive, or rough. For instance, given three photographs of different textures - say, one of fossilized seafan, one of soap bubbles, and the other of still water (see [5]) - one is at first sight involuntarily attracted to the one of seafan. This is because it presents the strongest "visual feel" amongst the three, as a result of the fact that the constituent components are very discernible to the eye. In fact, the photograph of still water would be the least attractive, because there are virtually no identifiable components.

The degree of distinguishability between the primitives making up a texture may depend upon two factors: the size of the primitives, and the differences between the average intensities of the primitives. It may be possible to distinguish between large primitives, even though differences between their average intensities are small. However, for such distinctions to be made between primitives of small sizes, there must be wide differences between their intensities and/or sharp edges between them.

From the above descriptions of the textural properties, the two most important factors that determine the degree in which a texture possesses a given property are:

(i) spatial changes in intensity levels (gray tones), and

(ii) dynamic range of gray scale

3.3. Neighbourhood Gray Tone Difference Matrix (NGTDM)

This is a column matrix in which the ith element, s(i), is the summation of the differences between all pixels having gray tone of value i and the average gray tones of their neighbourhoods (as defined below). The size of neighbourhood is specified by a distance parameter d.

Definition

Suppose the gray tone of the pixel at the point (k,ℓ), is denoted as $f(k,\ell)$, then the average gray tone in the neighbourhood of this pixel is defined as

$$\bar{A}(k,\ell) = \frac{1}{W-1}\left\{\left[\sum_{m=-d}^{d} \sum_{n=-d}^{d} f(k+m, \ell+n)\right] - f(k,\ell)\right\} \tag{3.1}$$

where d specifies the neighbourhood size, given by

$$W = (2d+1)(2d+1), \quad d = 1,2,3 \ldots..$$

Again, over all pixels of intensity i, namely $f(k,\ell) = i$, we define

$$\bar{A}_i = \bar{A}(k,\ell)$$

An entry in the matrix for the gray tone of value i is given by

$$s(i) = \sum_{i \in N_i} \left| i - \bar{A}_i \right| \tag{3.2}$$

where $\{N_i\}$ is the set of all pixels in the image (except those in the periphery) having gray tone = i, and s(i) is necessarily zero if no pixel within the appropriate part of the image has a gray tone = i.

Illustration

Consider the 5x5 sample image shown in Fig. 3.1(a). Specifying a distance, d=1, results in a 3x3 neighbourhood. This neighbourhood can only be centred on pixels within the indicated square. The other pixels are considered as being in the periphery of the image.

1	1	4	3	1
3	4	0	1	1
5	4	2	2	2
2	1	1	4	4
0	2	2	5	1

(a)

i	s(i)
0	2.750
1	4.125
2	0.250
3	0.000
4	4.875

(b)

Fig. 3.1 (a) Sample Image (b) NGTDM for Sample Image

There are two pixels within the indicated square with gray tone = 2.

Thus for this image

$$s(2) = \left| 2 - \frac{17}{8} \right| + \left| 2 - \frac{15}{8} \right| = 0.250$$

In similar fashion, we have

$$s(0) = 2.750$$

$$s(1) = 4.125$$

$$s(4) = 4.875$$

and s(3) is necessarily zero. The NGTDM for this sample image is as shown in Fig. 3.1(b).

3.4 Computational Measures for Textural Properties

(a) Coarseness

In coarse textures, there is slight spatial rate of change in intensity. For such textures, therefore, the difference between a pixel gray tone and the average gray tone of its neighbourhood would tend to be small. Thus, the result of the summation of such differences computed over all image pixels would give an indication of the degree of spatial rate of change in intensity. This is the same as the summation of the entries in the NGTDM. However, in the summation of the entries, each entry is weighted by the probability of occurrence of the corresponding gray tone value. The result of this summation, denoted as T, is given by

$$T = \sum_{i=0}^{G_h} p_i s(i) \tag{3.3}$$

where G_h is the highest gray tone value present in the image. For coarse textures, the value of T would be low. In order to give a measure that increases with the degree of coarseness, the following is derived:

$$f_{cos} = [\ \varepsilon + T\ ]^{-1}$$

$$= [\ \varepsilon + \sum_{i=0}^{G_h} p_i s(i)\ ]^{-1} \qquad (3.4)$$

where ε is a very small number, just to ensure a non-zero value, and p_i is the probability of gray tone value i. For an NxN image, and NGTDM computed using distance d, $p_i = N_i/n^2$, where $n = N-2d$.

(b) Contrast

Considering the description of contrast, and the increase in the level of contrast with gray scale range and local variation in intensity, the following computational form is proposed:

$$f_{con} = \left[\frac{1}{N_g(N_g-1)} \sum_{i=0}^{G_h} \sum_{j=0}^{G_h} p_i p_j (i-j)^2 \right] \left[\frac{1}{n^2} \sum_{i=0}^{G_h} s(i) \right] \qquad (3.5)$$

where N_g is the total number of different gray tone values (i.e. different gray levels) present in the image. It is given by

$$N_g = \sum_{i=0}^{G_h} Q_i \tag{3.6}$$

where
$$Q_i = \begin{cases} 1 & \text{if } p_i \neq 0 \\ 0 & \text{otherwise} \end{cases} \tag{3.7}$$

f_{con} is the product of two terms. The first quantity is the average weighted squared difference between the different gray tone values taken in pairs, and is used to reflect the dynamic range of gray scale; the weighting factor is a product of the probabilities of the two gray tone values under consideration. The second term is the average difference between pixel gray tones and the average gray tone of their neighbourhoods; this quantity increases with the amount of local variation in intensity.

(c) Busyness

The following computational measure is given for this property:

$$f_{bus} = \left[\sum_{i=0}^{G_h} p_i s(i) \right] \Big/ \left[\sum_{i=0}^{G_h} \sum_{j=0}^{G_h} ip_i - jp_j \right] \tag{3.8}$$

$$p_i \neq 0, \quad p_j \neq 0$$

The numerator is essentially a measure of the spatial rate of change in intensity, while the denominator is a summation of the magnitude of differences between the different gray tone values. Each value is weighted by its probability of occurrence. The denominator results in the suppression of the effect of contrast variations. Hence, the expression would tend to emphasize the frequency of spatial changes in intensity values.

(d) Complexity

As already mentioned, a texture is considered to be complex if the visual information content is high. The amount of visual information depends upon the number of patches, lines and edges that are noticed by the eye. These in turn depend on the rapidity or otherwise with which spatial changes in intensity occur, as well as the magnitude of the changes. Textures in which the spatial rate of change in intensity is slight generally tend to have few different values of gray tones; but there is a high probability of each value occurring. Consequently, in these textures, there are not many patches that have different average intensity levels, but the patches are large; hence these textures tend to have low degree of complexity.

On the other hand, textures with many different gray tone values tend to consist of many patches, and also many edges, due to rapid spatial changes in intensity. These patches are more noticeable, and the edges sharper, when the dynamic range of gray scale is large. A proposed computational measure for complexity is as follows:

$$f_{com} = \sum_{i=0}^{G_h} \sum_{j=0}^{G_h} \{(|i-j|)\ (p_i s(i) + p_j s(j))\} / \{n^2 (p_i + p_j)\} \qquad \textbf{(3.9)}$$

$$P_i \neq 0, \quad P_j \neq 0$$

(e) Texture Strength

Following the discussion in Part (e) of Section 3.2, we may define texture strength as

$$f_{str} = \left[\sum_{i=0}^{G_h} \sum_{j=0}^{G_h} (p_i+p_j)\ (i-j)^2 \right] \Bigg/ \left[\varepsilon + \sum_{i=0}^{G_h} s(i) \right] \qquad \textbf{(3.10)}$$

$$p_i \neq 0, \quad p_j \neq 0$$

This expression involves two terms. The numerator is a factor stressing the variation in intensity levels, and therefore may reflect intensity differences between adjacent primitives; while the denominator conveys information about the size of texture primitives, as it is essentially a measure of the spatial rate of change in intensity.

3.5. Approximation of Features to Textural Properties

Two sets of experiments were performed to determine the extent to which the texture features correspond to the properties, and therefore to human perception of textures. In each set of experiments, human subjects performed perceptual measurements on a set of natural textures, and the computer also performed corresponding tasks using the features that have been developed. The experiments were carried out with the following aims:-

(i) to investigate the degree to which each of the five textural features relates to each of the five textural properties, and consequently to determine whether the theoretically conceptualized textural property - textural feature relationship agrees with the practical case

(ii) to investigate the extent to which the features relate to each other, and also how the properties are correlated with one another

(iii) to investigate the extent, if any, to which certain combinations of the features can indicate similarity between different textural patterns, and therefore to determine the extent to which the features approximate human perception of textures

In all, 88 subjects performed the experiments; 48 men and 40 women. Ten natural textures taken from Brodatz's album [5] were used in the experiments. They were:

crushed rose <u>quartz</u> (D98)
depressed <u>cork</u> (D4)
straw <u>matting</u> (D55)
herringbone <u>weave</u> (D16)
<u>beach</u> pebbles and sand (D27)
<u>grass</u> lawn (D9)
beach <u>pebbles</u> (D23)
oriental glass <u>fibre</u> cloth (D79)
<u>pigskin</u> (D92)
fur of unborn <u>calf</u> (D93)

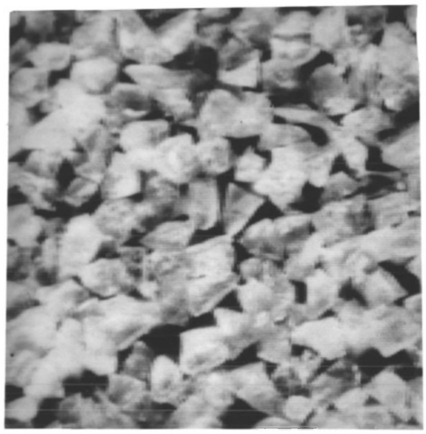
They were labelled samples A - J respectively. From henceforth, the textures will simply be referred to as quartz, cork, matting, weave, beach, grass, pebbles, fibre, pigskin and calf. A part of each of the original picture from the album was photographed on a 35 mm negative film and digitized into a 384 x 384 digital image with 256 gray levels. There was no other operation performed on the digital images (e.g. gray-scale contraction or histogram flattening).

However, since a comparison was to be made between the results of the perceptual measurements and those produced by the texture features, it was only natural that both processes used the same pictures. Thus, in the perceptual measurements, the original pictures from the album were not used, but rather the printed copies of the digital versions. In this regard, the digital pictures were displayed on a monitor and photographed. They are shown in Fig. 3.2 (A-J).

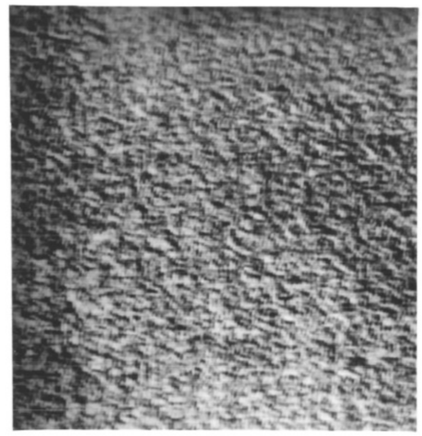
3.5.1 Ranking Experiments

This was the first set of experiments. Subjects were told to rank the ten textures using each of the five properties - coarseness, contrast, busyness, complexity and texture strength. Prior to performing the experiment, the subject was given a brief explanation of the concept of texture and each of the five textural properties.

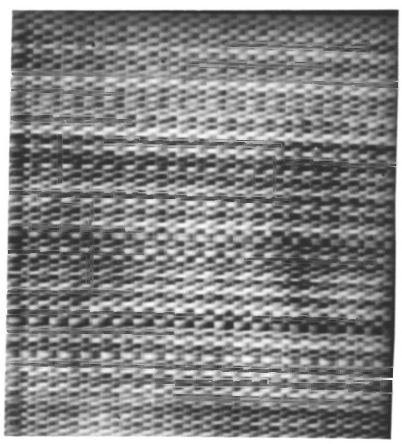
In the case of the computer, each of the 384 x 384 digital images was divided into sixteen subimages, each of size 96 x 96. It is reasonable to assume that a subimage of this size is large enough to capture the desired textural properties satisfactorily. The five features were computed for each of the subimages and the average over the sixteen was determined. Two distances, d = 1 and d = 2, were



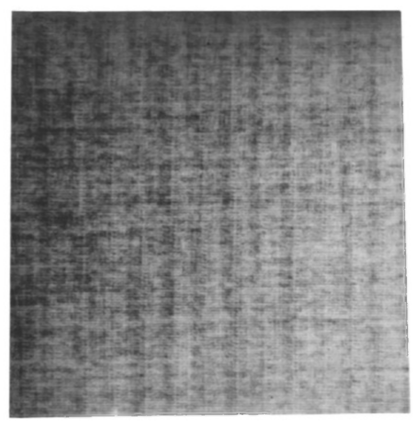
A. Quartz



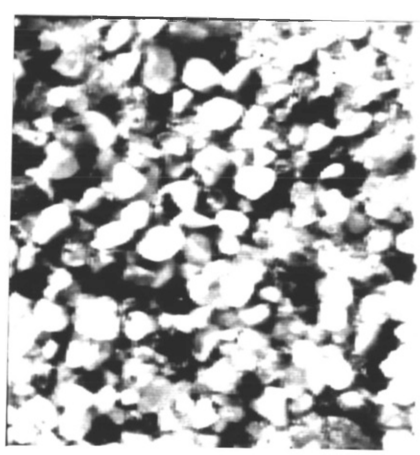
B. Cork



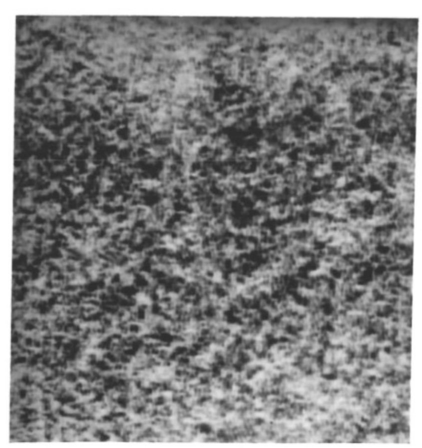
C. Matting



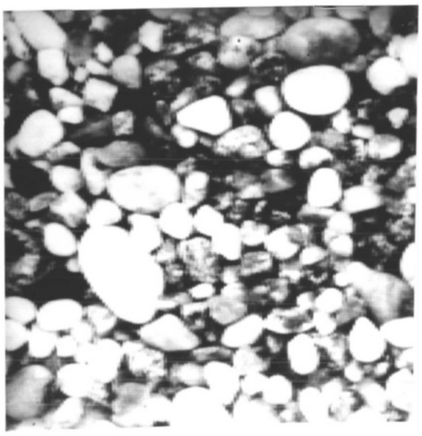
D. Weave



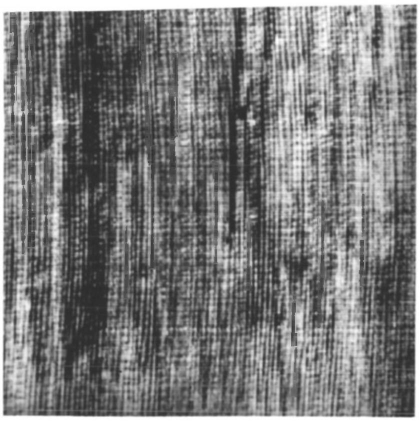
E. Beach



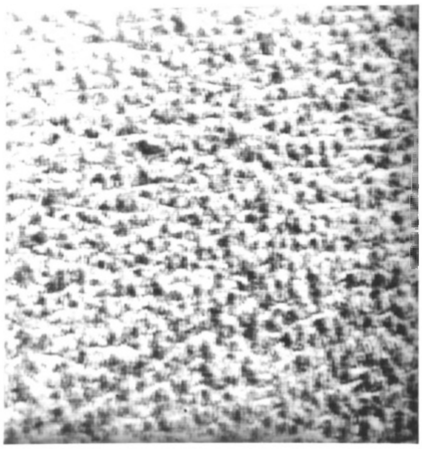
F. Grass



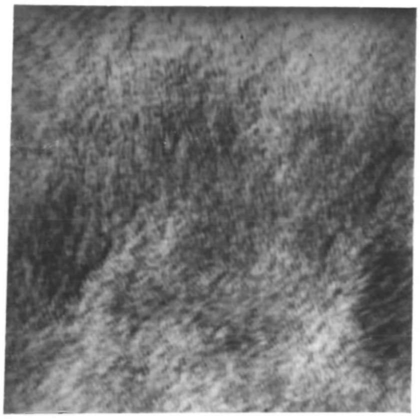
G. Pebbles



H. Fibre



I. Pigskin



J. Calf

Fig. 3.2 Natural Textures Used in Ranking and Texture Similarity Measurements used in feature computation, corresponding to neighbourhood sizes of 3 x 3 and 5 x 5 respectively. These average values of features were used to rank the textures. The texture having the highest average value for a given feature was given a rank of 1 with respect to that feature, and the one with the least value a rank of 10. In the computation of the features f_{cos} and f_{str}, the value of ε was put at 10^{-7}. The rankings are presented in Table T3.1.

3.5.2 Comparison of Human and Feature Rankings

In order to make a comparison between the rankings produced by humans and those by features, it was first of all necessary to determine a representative human ranking for each texture property from the rankings produced by the 88 subjects. The Psychometric Method of Rank Order, discussed in [28], was adapted to determine these representative, or composite, rankings.

The technique involves the computation of a quantity called the sum of rank values. Assuming that n objects are ranked, the sum of rank values for the jth object is given by

$$Z_j = \sum_{k=1}^{n} f_{jk} R_k \qquad (3.11)$$

where f_{jk} is the frequency of giving the rank k (k = 1,2,....,n) to the jth object. This is the same as the number of subjects that give the jth object the rank k.

The frequencies of ranks for the ten textures are presented in Table T-A1.1 in Appendix A-1.

Ranks (k)	f_{cos} d=1	f_{cos} d=2	f_{con} d=1	f_{con} d=2	f_{bus} d=1	f_{bus} d=2	f_{com} d=1	f_{com} d=2	f_{str} d=1	f_{str} d=2
1	G	E	G	G	C	H	B	B	A	A
2	E	G	B	B	H	C	F	F	G	E
3	A	A	F	F	D	B	C	C	E	G
4	I	I	E	C	B	D	G	H	F	F
5	J	J	C	E	I	F	I	I	B	B
6	F	F	I	H	F	I	E	G	I	I
7	C	B	H	I	J	J	H	E	C	J
8	B	D	A	A	G	G	A	A	J	C
9	H	C	D	D	E	E	J	J	H	H
10	D	H	J	J	A	A	D	D	D	D

Table T3.1

Ranking of Textures Using Features

Computed at Two Distances: d=1 and d=2

Ranks (k)	Coarseness	Contrast	Busyness	Complexity	T.Strength
1	G	G	D	C	G
2	A	E	H	F	E
3	E	C	F	H	C
4	I	H	B	I	A
5	C	F	C	G	H
6	F	I	J	E	I
7	B	B	I	B	F
8	H	J	E	D	B
9	J	A	G	J	D
10	D	D	A	A	J

Table T3.2

Representative Human Rankings of Textures

using Textural Properties

R_k is a series made up of rank values. These values are in exact reverse order to the rank k. R_k is related to k by the equation

$$R_k = n - k + 1 \tag{3.12}$$

The sums of rank values are then used to obtain the representative ranking. The object (and in the present case the texture) whose sum of rank values is highest is assigned a rank of 1, that with the second highest a rank of 2, and so on. The resulting representative human rankings for the five textural properties are shown in Table T3.2.

The comparison of human and feature rankings involved the determination of the degree of correspondences between them. In this regard, the well-known Spearman's coefficient of rank correlation was used. This coefficient is given as

$$r_s = 1 - \frac{6D}{N^3-N} \tag{3.13}$$

where D, called the summed squared difference, is given by

$$D = \sum_{k=1}^{N} (r_{ik} - r_{jk})^2 \tag{3.14}$$

and r_{ik} and r_{jk} are the ranks given to the kth object in the ith and jth ranking respectively. N is the number of objects ranked; in the present case, N = 10.

The value of r_s is between -1 and 1. The value -1 corresponds to complete disagreement between the two rankings, and the value 1 indicates complete agreement. Equation (3.13) assumes that, as in the present case, there are no ties in ranks, i.e. no two or more objects are given the same rank in any of the rankings. A more complex expression exists for situations where there are ties - see [69] for details.

Using equations (3.13) and (3.14), the coefficients of rank correlation were determined for the following:

(i) between each feature ranking and the representative human ranking for each textural property

(ii) between each feature ranking and every other feature ranking

(iii) between the representative ranking for each property and the representative ranking for every other property

The results are presented in Table T3.3(a-e).

The results in Table T3.3(a) and (b) show that each feature is more correlated with the appropriate texture property than the other properties, except for the feature f_{str}. There is a stronger correlation of this feature with coarseness than texture strength. A strong correlation also exists between the features f_{cos} and f_{str}, and between f_{cos} and texture strength. It is very likely that the two features, and perhaps the two properties as well (as they are also very correlated) convey essentially the same information about a texture.

Textural Features	Textural Properties Coarseness	Contrast	Busyness	Complexity	T.Strength
f_{cos}	0.856	0.442	-0.927	-0.152	0.612
f_{con}	0.527	0.685	-0.176	0.467	0.515
f_{bus}	-0.600	-0.018	0.782	0.552	-0.272
f_{com}	0.321	0.503	-0.006	0.600	0.261
f_{str}	0.879	0.321	-0.794	-0.139	0.600

(a) Between Human and Feature Rankings (Features Computed at d=1)

Textural Features	Textural Properties Coarseness	Contrast	Busyness	Complexity	T.Strength
f_{cos}	0.721	0.224	-0.842	-0.382	0.418
f_{con}	0.455	0.697	-0.079	0.539	0.515
f_{bus}	-0.624	0.018	0.830	0.564	-0.297
f_{com}	0.091	0.406	0.236	0.685	0.127
f_{str}	0.806	0.248	-0.794	-0.248	0.503

(b) Between Human and Feature Rankings (Features Computed at d=2)

	f_{str}	f_{com}	f_{bus}	f_{con}
f_{cos}	0.806	0.079	-0.830	0.152
f_{con}	0.552	0.867	-0.079	
f_{bus}	-0.782	0.176		
f_{com}	0.370			

(c) Between Feature Rankings (Features Computed at d=1)

Continued over page ....

	f_{str}	f_{com}	f_{bus}	f_{con}
f_{cos}	0.830	-0.345	-0.939	0.079
f_{con}	0.345	0.745	0.164	
f_{bus}	-0.794	0.539		
f_{com}	0.042			

(d) Between Feature Rankings (Features Computed at d=2)

	T.Strength	Complexity	Busyness	Contrast
Coarseness	0.842	0.091	-0.855	0.539
Contrast	0.782	0.661	-0.261	
Busyness	-0.588	0.309		
Complexity	0.345			

(e) Between Representative Human Rankings

Table T3.3

Coefficients of Rank Correlations

There is also a strong correlation between the features f_{con} and f_{com}, and between the properties of contrast and complexity, though not as strong as that between f_{cos} and f_{str}, and coarseness and texture strength respectively. The feature f_{bus} is shown to be the most independent feature.

3.5.3 Measurement of Texture Similarity

In this experiment, subjects were told to find a most similar, and a second most similar, texture to each of the ten textures; similarity need not be reciprocal. For instance, if B was considered to be most similar to A, this did not necessarily mean that A was most similar to B; C might be more similar to A than B. The number of subjects that considered a given texture as the most similar, or second most similar, to a reference texture constitutes the frequency of assignment of the given texture as the most similar or second most similar one to the reference texture. These frequencies were used to obtain representative human similarity assignments. The texture which had the highest frequency as being the most similar to a reference texture was considered to be the representative most similar texture. The same applied for the second most similar case. The human representative similarity assignments obtained are given in Table T3.4. The frequencies of similarity assignments are shown in Table T-A1.2 in Appendix A-1.

For the automatic case, five different combinations of features were used, and two distance criteria were employed to measure similarity. The first criterion finds for each texture the one having the maximum likelihood from amongst the other nine or eight textures. This corresponds to finding, from amongst the other textures, the one with the minimum (squared) Mahalanobis distance to

Reference Texture	Most Similar Texture	Second Most Similar Texture
A	E	G
B	F	I
C	H	D
D	H	C
E	G	A
F	B	I
G	E	A
H	C	D
I	F	B and F
J	F	B

Table T3.4

Representative Human Similarity Assignments

Reference Texture	Feature Combination (d=1)										Feature Combination (d=2)									
	f_{cos} f_{con} f_{bus}		f_{cos} f_{con} f_{com}		f_{bus} f_{com} f_{str}		f_{cos} f_{con} f_{bus} f_{com}		f_{con} f_{bus} f_{com} f_{str}		f_{cos} f_{con} f_{bus}		f_{cos} f_{con} f_{com}		f_{bus} f_{com} f_{str}		f_{cos} f_{con} f_{bus} f_{com}		f_{con} f_{bus} f_{com} f_{str}	
A	**E**	**G**	**E**	**G**	G	E	**E**	**G**	G	E	**E**	**G**	**E**	**G**	G	E	**E**	**G**	G	E
B	**F**	**I**	**F**	C	**F**	**I**	**F**	**I**	**F**	**I**	**F**	C	**F**	C	**F**	H	**F**	C	**F**	C
C	I	F	B	F	**H**	G	**H**	F	G	H	F	B	**B**	F	**H**	G	F	B	G	F
D	J	H	J	**C**	J	**C**	J	**C**	J	G	J	**C**	J	B	J	**C**	J	**C**	J	**C**
E	**G**	**A**	**G**	**A**	**G**	**A**	**G**	**A**	**G**	**A**	**G**	**A**	**G**	**A**	**G**	**A**	**G**	**A**	**G**	**A**
F	**B**	**I**	**B**	C	**B**	**I**	**B**	**I**	**B**	**I**	**B**	C	**B**	C	**B**	C	**B**	C	**B**	C
G	**E**	**A**	**E**	**A**	**E**	J	**E**	**A**	**E**	**A**	**E**	**A**	**E**	**A**	**E**	J	**E**	**A**	**E**	**A**
H	**C**	**D**	**C**	B	**C**	G	**C**	F	**C**	G	**C**	B	**C**	**D**	**C**	G	**C**	B	**C**	G
I	C	J	C	F	C	H	C	D	C	F	J	D	J	D	**F**	G	D	F	**F**	G
J	D	H	D	**B**	D	C	D	C	C	G	D	F	D	**B**	D	F	D	C	D	G
No. of Agreements	6	6	6	5	6	4	7	6	5	4	6	4	7	5	7	2	6	4	6	3

(a)

Reference Texture	Feature Combination (d=1)										Feature Combination (d=2)									
	f_{cos} f_{con} f_{bus}		f_{cos} f_{con} f_{com}		f_{bus} f_{com} f_{str}		f_{cos} f_{con} f_{bus} f_{com}		f_{con} f_{bus} f_{com} f_{str}		f_{cos} f_{con} f_{bus}		f_{cos} f_{con} f_{com}		f_{bus} f_{com} f_{str}		f_{cos} f_{con} f_{bus} f_{com}		f_{con} f_{bus} f_{com} f_{str}	
A	**E**	**G**	**E**	I	**E**	**G**	**E**	**G**	**E**	**G**	**E**	**G**	**E**	**G**	**E**	**G**	**E**	**G**	**E**	**G**
B	**F**	C	**F**	C	**F**	C	**F**	C	**F**	C	**F**	C	**F**	C	**F**	C	**F**	C	**F**	C
C	**H**	B	I	H	**H**	I	**H**	I	**H**	I	**H**	D	**H**	F	**H**	I	**H**	B	**H**	B
D	J	I	J	H	J	H	J	H	J	H	J	I	J	I	J	I	J	I	J	I
E	A	G	**G**	**A**	A	G	A	G	A	G	A	G	A	G	A	G	A	G	A	G
F	**B**	C	**B**	C	**B**	C	**B**	C	**B**	C	**B**	**I**	**B**	C	**B**	C	**B**	C	**B**	C
G	**E**	**A**	**E**	**A**	**E**	**A**	**E**	**A**	**E**	**A**	**E**	**A**	**E**	**A**	**E**	**A**	**E**	**A**	**E**	**A**
H	**C**	I	I	C	**C**	I	**C**	I	**C**	I	**C**	I	I	C	**C**	I	**C**	I	**C**	I
I	H	D	H	C	C	H	H	C	H	C	D	J	H	C	J	D	J	D	J	H
J	D	I	D	H	D	I	D	I	D	I	D	I	D	I	D	I	D	I	D	I
No. of Agree-ments	6	2	5	2	6	2	6	2	6	2	6	3	5	2	6	2	6	2	6	2

(b)

Table T3.5
Computer Similarity Assignments
(a) using Maximum Likelihood Criterion
(b) using Euclidean Distance Criterion

the mean of the reference texture. The second distance criterion is a normalized Euclidean distance, where the result of normalization is such as to constrain all feature values to lie between zero and one. This kind of normalization procedure is described in Chapter Five.

The similarity assignments for the two distance criteria and for the five combinations of features are shown in Table T3.5(a) and (b). Under each feature combination, there are two columns, the one on the left being for the most similar assignment, and the one on the right for the second most similar. A letter in bold type corresponds to an agreement with the representative human similarity assignment. The total number of such agreements is written under each column.

The results show that, for the most-similar assignment category, there is agreement between the human and computer similarity assignments for at least half of the number of textures for the two distance criteria. For the second most similar assignments, the results are not as good. Overall, however, the results are very encouraging. They indicate that the features, to some extent, approximate visual perception.

3.6 Conclusion

An attempt has been made to develop measures that correspond to some textural properties, and therefore to visual perception of textures. Five basic properties of texture; namely, coarseness, contrast, busyness, complexity, and the strength of texture, were conceptually defined or expressed in terms of spatial changes in intensity. The conceptual expressions were then put into computational forms. In this approach, a one-dimensional matrix, called a Neighbourhood Gray Tone Difference Matrix (NGTDM), was computed for a given image, and from this matrix the features were derived. The method is computationally efficient, as the features are quickly computable and the memory requirement is very small.

The measures were used in two experiments which also involved perceptual measurements by human subjects. One experiment involved the ranking, by humans, of a set of natural textures according to the degree to which they possessed a given textural property; the computer performed a similar task using the features. The second experiment was the measurement of texture similarity, both by humans and by the computer, the latter using certain combinations of the features.

With respect to ranking, very successful results were obtained. The results show not only that there are very high levels of correspondences between computational and perceptual measurements, but also that each feature relates more to the appropriate textural property, except for the feature f_{str}. This feature is found to be slightly more correlated to coarseness than to texture strength for the textures used in the experiments. In any case, the two features f_{cos} and f_{str} are very correlated with each other. There is a high likelihood that they convey essentially the same information about a texture.

For the experiments designed to indicate similarity between different textural patterns, the results were also encouraging. The most similar pattern was correctly identified by the computer for at least five of the ten test textures for each of the five feature combinations used. The results for the second most similar textures varied more widely. Only in two to six cases did the computer results agree with the representative human similarity assignments. However, it should be realised that the representative human similarity assignments were derived using a "vote-type" count (i.e. a simple majority), and not taking into consideration the variations between the assignments of the individual subjects. If the variations are taken into account, then the results obtained could be considered as very successful, especially if one also considers the results of similar experiments in [74]. It is also probable that the differences between human and computer similarity assignments arise from the fact that the mechanism of human usage of multiple cues may be much more complex than the maximum likelihood and Euclidean distance criteria used by machine.

The results also show that the features computed at the two distances used produced similar rankings and similarity measurements. However, the feature computation at d = 1 involved a smaller amount of computation. Therefore, one may consider this distance as optimum for the computation of these features.

Finally, based upon the results obtained, it is hoped that any combination of three or four features would produce satisfactory results in image classification problems. For three-feature combination, the feature f_{bus} can be combined with either f_{con} or f_{com} and either f_{cos} or f_{str}. In the case of four features, any combination not inclusive of both f_{cos} and f_{str} is recommended.

CHAPTER FOUR

TEXTURE-BASED IMAGE CLASSIFICATION AND SEGMENTATION

4.1 Introduction

As mentioned in Chapter Three, texture analysis is an important aspect of scene interpretation. It is the characterization or description of a texture, such that either or both of the following problems may be solved:-

(a) A sample image can be classified or identified on the basis of its textural pattern.

(b) Given an image with differently textured regions, the image can be partitioned into component areas corresponding to the textured regions.

These two problems are generally referred to as "texture classification" and "textural segmentation" respectively.

4.1.1 Texture Classification

In texture classification, the interest is in the extraction of a set of texture measures for the automatic discrimination between different texture classes. The perception-related features developed in Chapter Three, and almost all the texture analysis methods mentioned in Chapter Two, are attempts made in this direction. The performance criterion in any texture classification problem, (and as a result, the evaluation of any texture analysis technique for image classification), is the accuracy of classification. The general procedure is to consider groups of images belonging to different texture classes, where images within any one group belong to a single class. For each class, the images are arbitrarily divided into two sets; a training set and a testing set. The images in the training set are called the training samples, while those in the testing set are the testing samples.

Features are computed on images in the training set, and used to train a classifier. Features are also computed for each image in the testing set. After training, each testing sample is presented to the classifier to identify. This approach to the division of the images in a class into training and testing sets is followed when there are many sample images. When there are few samples available for a class, the method of leaving-one-out is used. In this method, also called the method of training on the data [18], a sample is left out for that class, while the classifier is trained on the remaining samples. The sample that was omitted is then presented to the classifier for identification. The process is repeated, each sample in turn being the one left out. The accuracy of classification in either approach is: the ratio of the number of correctly identified samples to the total number of samples presented for identification, expressed in percentage. That is,

$$\text{Classification Accuracy} = \frac{\text{No. of Correctly Identified Samples}}{\text{Total No. of Samples Presented for Identification}} \times 100\% \quad (4.1)$$

In section 4.2, the perception-related features developed in Chapter Three are applied in two image classification tasks; one involving twelve natural textures (subsection 4.2.1), and the other, agricultural land-use categories (subsection 4.2.2). The performances of the texture features developed in this work are then compared with two existing texture analysis methods for the same classification problems mentioned above.

4.1.2 Textural Segmentation

There are two approaches to textural segmentation. One approach is to assign a unique label to all image points belonging to the same textural pattern in the image, as in supervised segmentation. The other is to locate or trace the boundaries between areas of different textures - a process also known as "texture edge detection". The performance criterion in the first approach may also be percentage classification accuracy, particularly in situations where the number of image points belonging to each textural pattern is known. For the second approach, the criterion is edge identification accuracy. In either approach, the task is to associate each pixel with the texture region in the image to which it belongs. This necessitates the extraction of features at local level. However, texture is a neighbourhood property. An image point on its own possesses no texture, and this also applies to a very small neighbourhood. Thus, there exists a contradiction between texture as a neighbourhood property and the local-level requirement for segmentation.

Texture characterization with respect to segmentation has received only little attention compared with texture classification. The use of the existing texture analysis approaches for segmentation would result in great computational burden. In section 4.3, two features are developed specifically for texture-based image segmentation, with the additional aim of minimizing computational cost. The application of the features in the segmentation of some textured images is presented. The concluding part of the Chapter is in section 4.4.

4.2 Application of Perception-Related Texture Features in Image Classification

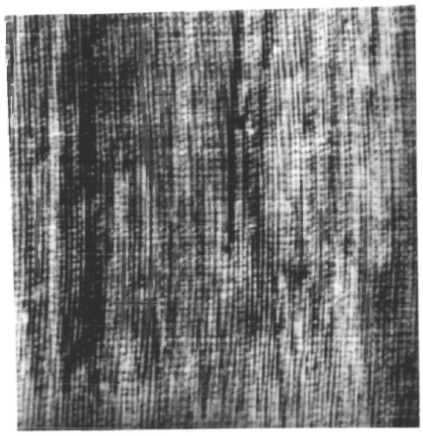
In this section, the results of image classification experiments using the texture features that have been developed are presented. The corresponding results using two classical texture analysis methods are also given, and a comparison is made between the present approach and these classical techniques. In all the classification experiments, the so-called "minimum error-rate classifier", described in [18,Chapter Two], was used. For this classifier, the training process involves the computation of the mean feature vectors and feature covariance matrices for each of the classes. Once these are obtained, the feature space is partitioned into n number of regions separated by hyperplanes, where n is the number of classes. By obtaining the feature vector of a testing sample, and determining into which of the n regions in the feature space it falls, the testing sample is classified. Further discussion of the classifier is given in Appendix A-1. In the experiments, a 3x3 neighbourhood (i.e. distance, d = 1) was used in the computation of the NGTDM, and for features f_{cos} and f_{str}, the value of ε was put at 10^{-7}.

4.2.1 Classification of Natural Textures

Twelve classes of natural textures (also taken from Brodatz's album [5]) were used in the experiments. Nine of the textures were also used in the previous experiments of ranking and texture similarity measurements. The twelve textures are:

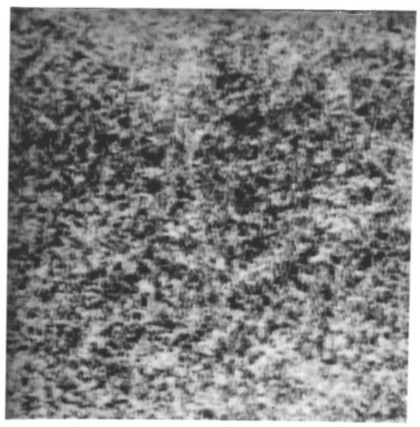
oriental glass fibre cloth (D79)
grass lawn (D9)
straw matting (D55)
beach pebbles (D23)
pigskin (D92)
crushed rose quartz (D98)
seafan fossilized with coral covering (D87)
herringbone weave (D16)
straw - North Beach, Long Island (D15)
fur of unborn calf (D93)
handwoven oriental rattan (D65)
depressed cork (D4)

The textures will simply be referred to as fibre, grass, matting, pebbles, pigskin, quartz, seafan, weave, straw, calf, rattan and cork; and they correspond to classes 1 to 12 respectively. The textures cover a wide range of different types. Among them, cork, calf, grass and weave can be regarded as fine, while quartz and pebbles belong to a very coarse category. Matting, pebbles and seafan display high contrast, while calf, straw, seafan and rattan are rich in directionality. From a subjective point of view,

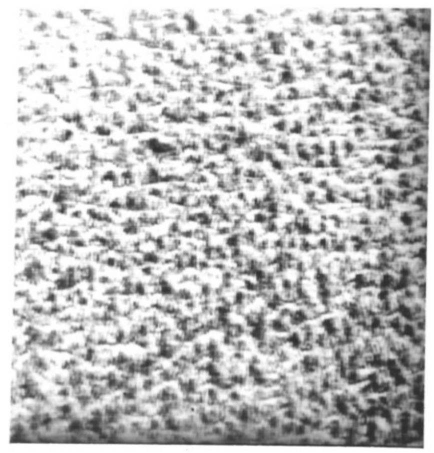


Class 1. Fibre

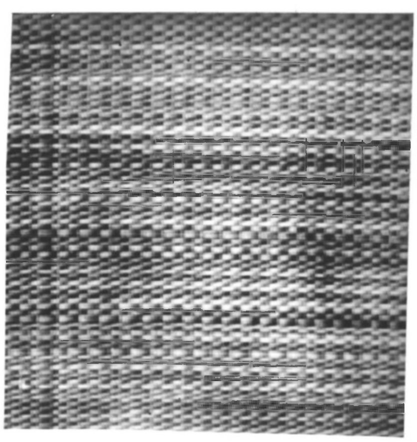
Class 4. Pebbles



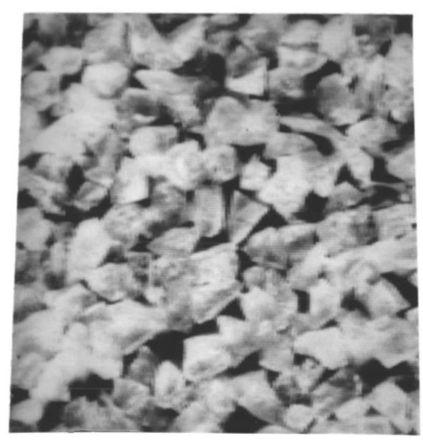
Class 2. Grass



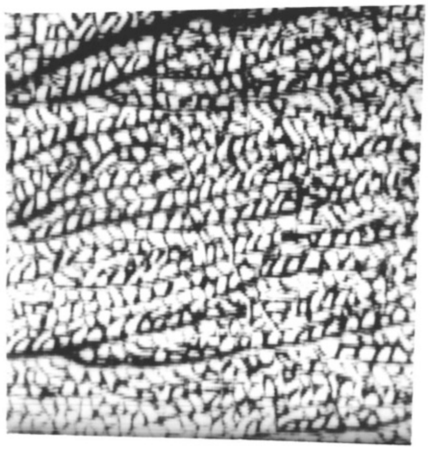
Class 5. Pigskin



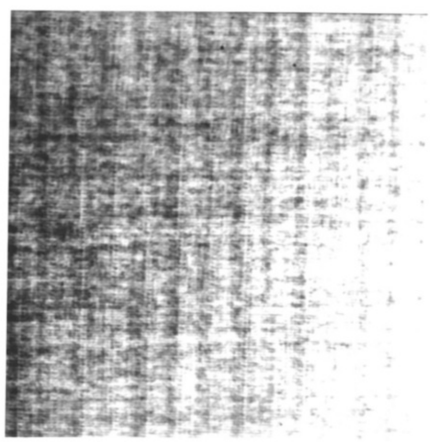
Class 3. Matting



Class 6. Quartz

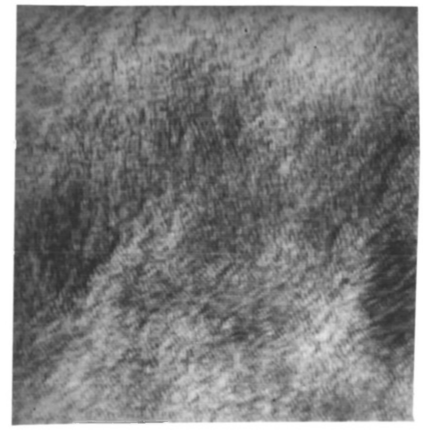


Class 7. Seafan

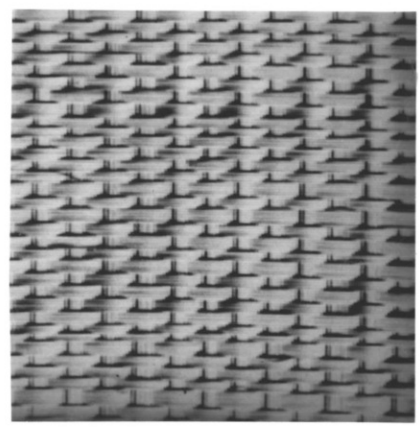


Class 8. Weave

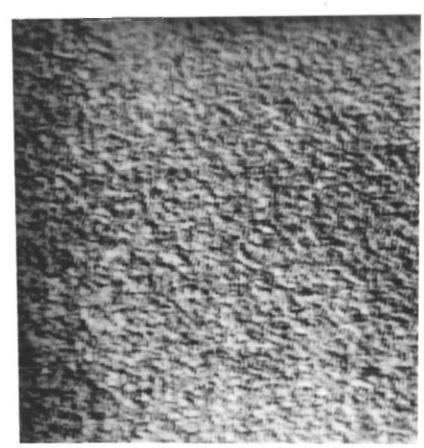
Class 9. Straw



Class 10. Calf



Class 11. Rattan



Class 12. Cork

Fig. 4.1 Twelve Classes of Natural Textures Used in Classification

matting, seafan and grass may be considered as very busy textures, while pebbles and seafan would be considered very attractive in appearance.

Each texture class was a 384 x 384 digital image. Prints of the images are shown in Fig. 4.1. Each image was divided into thirty-six 64 x 64 subimages. For each class, twenty-four subimages were randomly selected and used for training the classifier, while the remaining twelve were used as testing samples. Thus, there was a total of 288 training samples and 144 testing samples in all.

The five features - f_{cos}, f_{con}, f_{bus}, f_{com} and f_{str} - were computed for each of the training samples and also for each of the testing samples. Ten different combinations of the features were used in classification; six combinations of three features, three combinations of four features, and the five features together. For each feature combination, the features computed for the training samples were used to train the classifier, after which the feature set for each of the 144 testing samples was presented for identification. Table T4.1 shows the mean values of the features for the classes. The number of correctly identified samples per class; the total number of correctly identified samples; and the percentage classification accuracy, are shown in Table T4.2 for the given feature combination. The range plots for the five features and for the twelve classes are given in Fig. 4.2(a-e).

4.2.2 Agricultural Land-Use Classification

In this application, a black-and-white aerial picture of an agricultural area was obtained from the Ministry of Agriculture in Cambridge. The area consists mainly of agricultural fields (wheat, potato and winter barley), and forests (coniferous trees under

CLASS		FEATURES				
		f_{cos}	f_{con}	f_{bus}	f_{com}	f_{str}
1.	Fibre	0.00165	0.77004	0.05241	14.54081	6.04796
2.	Grass	0.00174	1.15642	0.03710	26.12795	9.77911
3.	Matting	0.00173	1.01351	0.05177	18.63201	8.04669
4.	Pebbles	0.00309	1.38273	0.02671	17.60530	21.94452
5.	Pigskin	0.00189	0.81238	0.04042	15.73748	8.39689
6.	Quartz	0.00311	0.68346	0.01626	14.11043	22.72552
7.	Seafan	0.00127	3.95989	0.06132	41.52526	10.82443
8.	Weave	0.00159	0.47815	0.04185	9.08838	4.53459
9.	Straw	0.00150	1.69606	0.06092	30.17628	10.61665
10.	Calf	0.00172	0.49481	0.03937	9.90405	6.11509
11.	Rattan	0.00190	1.30975	0.02217	25.17763	18.58795
12.	Cork	0.00171	1.21351	0.04138	24.95346	8.74924

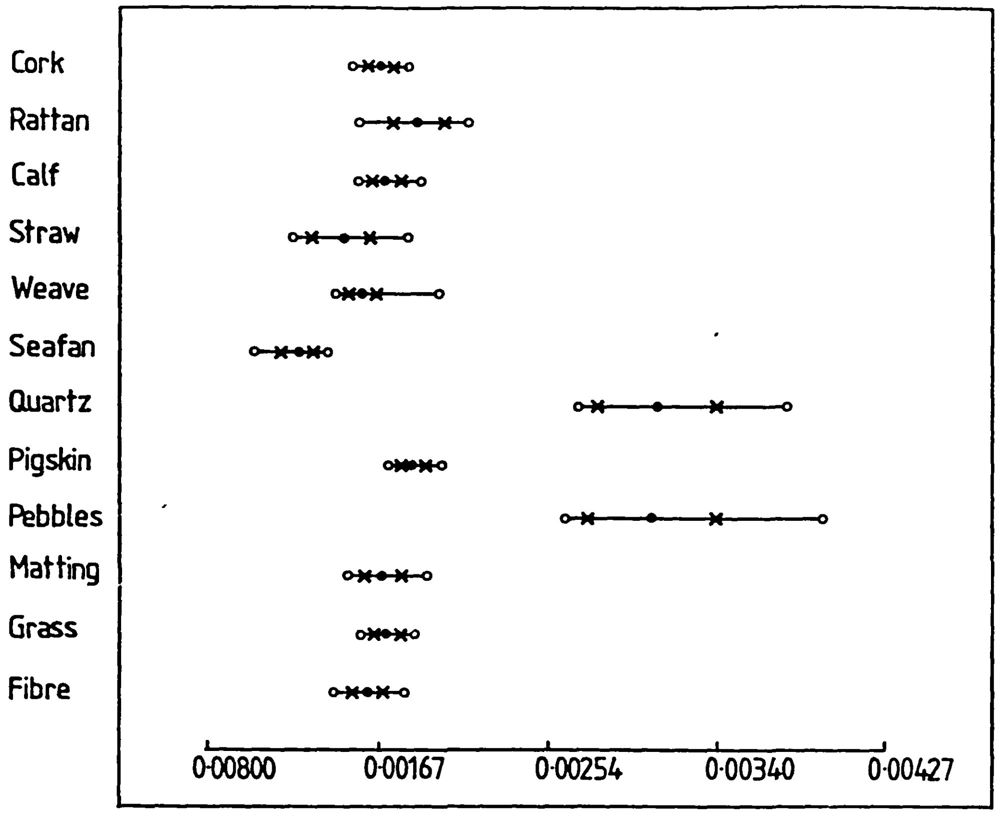
Table T4.1

Mean Values of Features

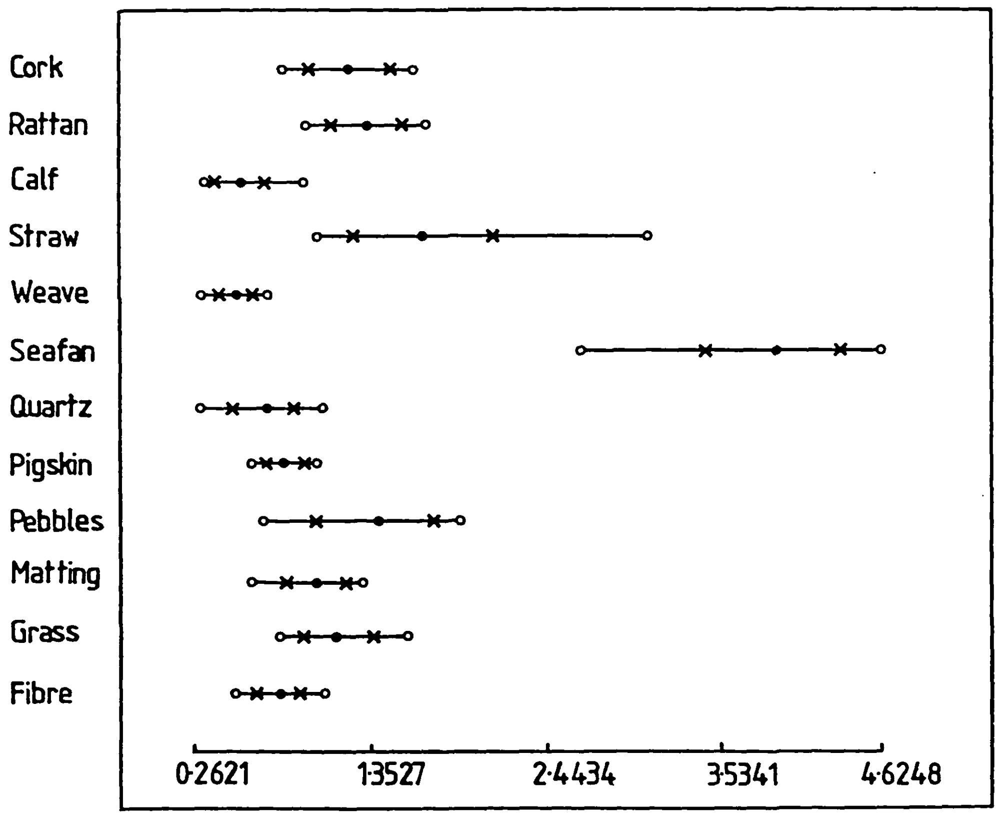
for Natural Textures

FEATURES	NUMBER OF CORRRECTLY CLASSIFIED SAMPLES PER CLASS												TOTAL NO of CORRECTLY CLASSIFIED SAMPLES	ACCURACY IN (%)
	1	2	3	4	5	6	7	8	9	10	11	12		
f_{con}, f_{com}, f_{str}	8	7	4	8	6	11	12	11	10	12	12	9	110	76.39
f_{cos}, f_{con}, f_{com}	6	5	5	9	7	12	12	11	12	8	9	7	103	71.53
f_{cos}, f_{com}, f_{str}	6	6	6	10	6	11	10	11	10	9	12	9	106	73.61
f_{bus}, f_{com}, f_{str}	7	10	5	11	7	9	9	11	11	11	12	9	112	77.78
f_{con}, f_{bus}, f_{com}	7	7	9	9	6	10	12	8	12	10	12	4	106	73.61
f_{cos}, f_{con}, f_{bus}	8	6	10	11	8	12	12	11	12	9	12	8	119	82.64
f_{cos}, f_{con}, f_{com}, f_{str}	8	8	6	9	7	12	12	11	12	10	12	11	118	81.94
f_{con}, f_{bus}, f_{com}, f_{str}	8	10	8	11	7	10	12	12	12	12	12	9	123	85.42
f_{cos}, f_{con}, f_{bus}, f_{com}	6	7	11	11	9	11	12	12	12	10	12	8	121	84.03
f_{cos}, f_{con}, f_{bus}, f_{com}, f_{str}	9	8	12	11	8	11	12	12	11	10	12	10	126	87.50

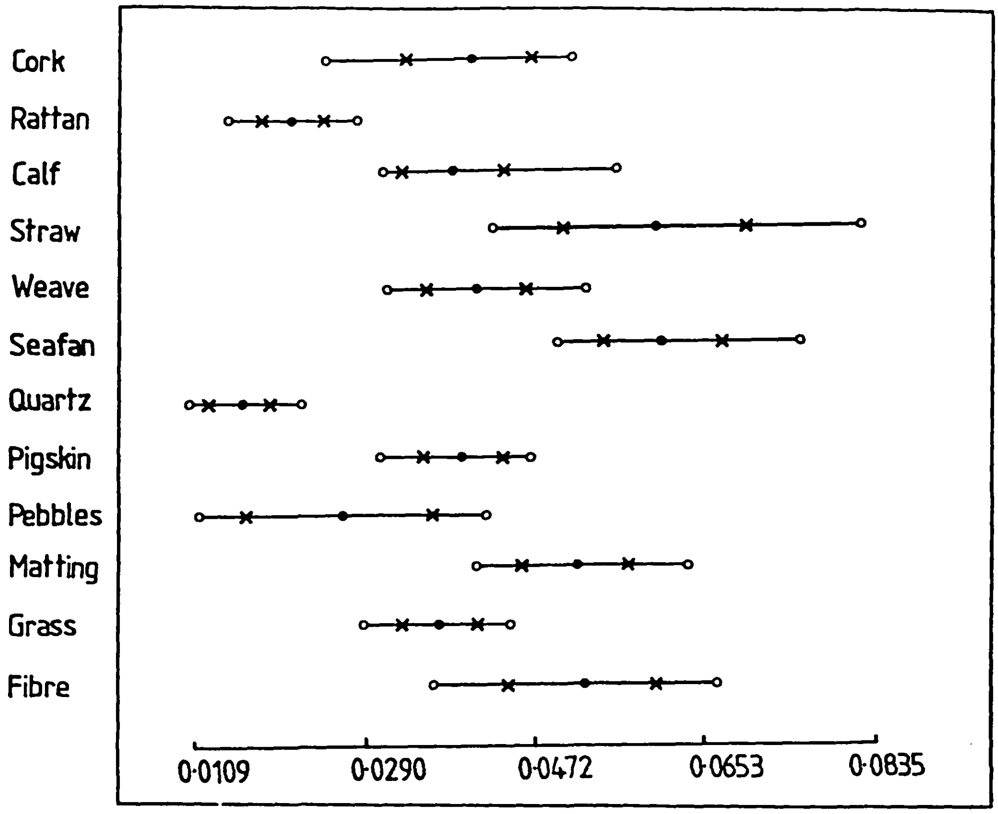
Table T4.2
Classification Results for Natural Textures Using Features Developed in this Work


Cork
Rattan
Calf
Straw
Weave
Seafan
Quartz
Pigskin
Pebbles
Matting
Grass
Fibre
0·00800
0·00167
0·00254
0·00340
0·00427

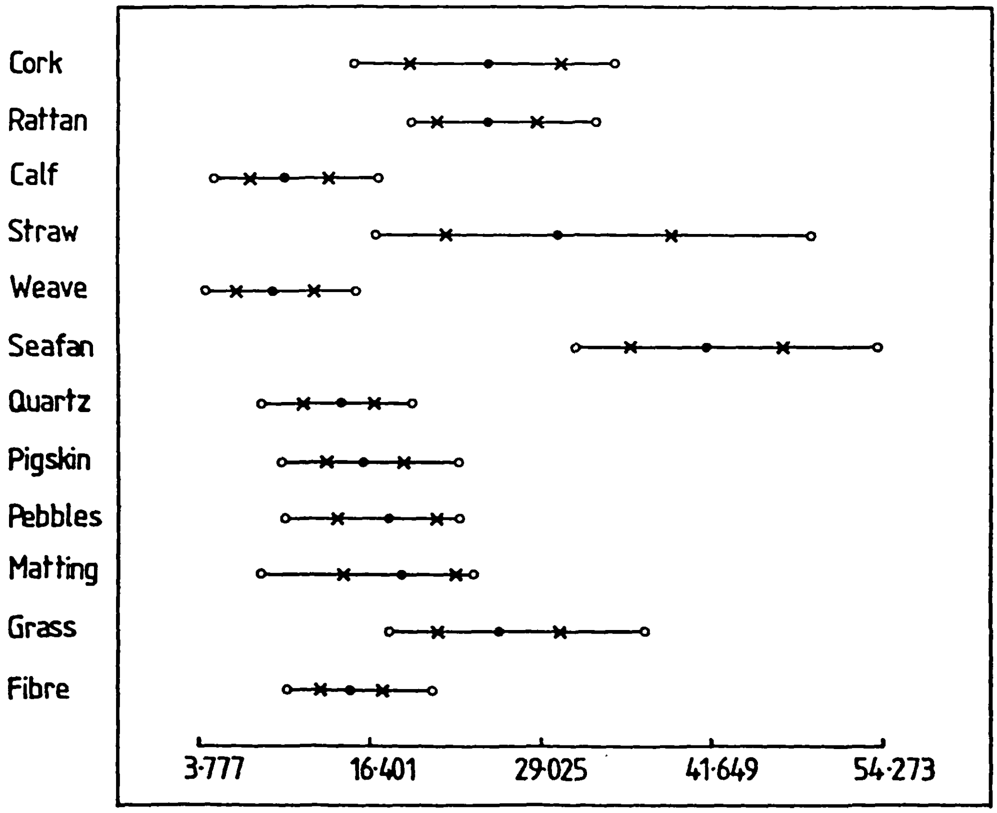
a, f_{COS}


Cork
Rattan
Calf
Straw
Weave
Seafan
Quartz
Pigskin
Pebbles
Matting
Grass
Fibre
0·2621
1·3527
2·4434
3·5341
4·6248

b, f_{CON}


Cork
Rattan
Calf
Straw
Weave
Seafan
Quartz
Pigskin
Pebbles
Matting
Grass
Fibre
0·0109
0·0290
0·0472
0·0653
0·0835

c, f_{bus}


Cork
Rattan
Calf
Straw
Weave
Seafan
Quartz
Pigskin
Pebbles
Matting
Grass
Fibre
3·777
16·401
29·025
41·649
54·273

d, f_{COM}

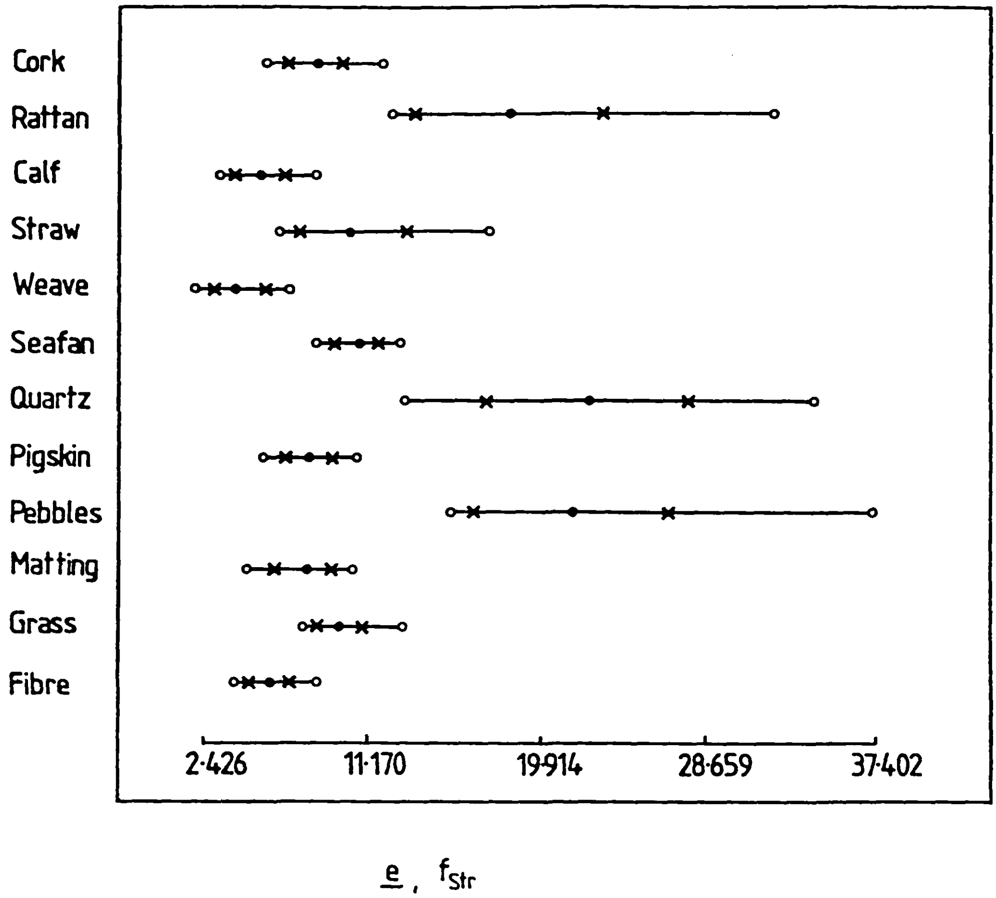


Fig. 4.2 Range Plots of Features for Twelve Natural Textures

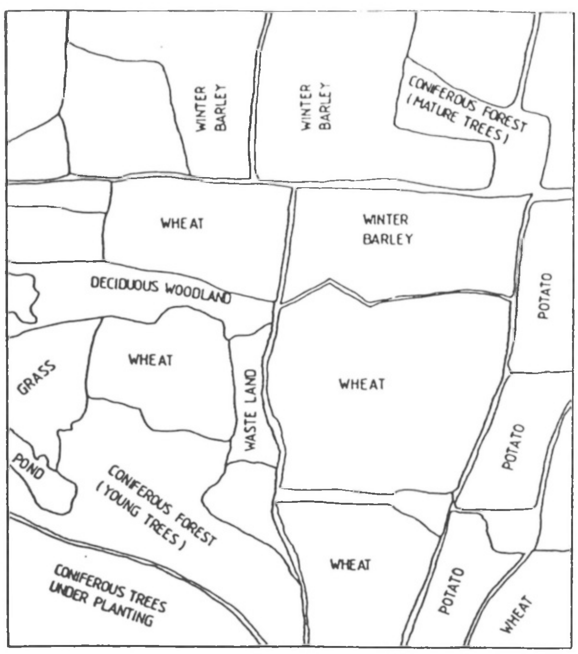
•:mean value o:range extrema x:mean ± σ

planting, young coniferous trees, mature coniferous trees and deciduous woodland). As a result of the ground survey that they carried out, specific parts of the image were associated with particular crop or forest types. This provided the ground truth information.

The picture was rephotographed and digitized into a 1024 x 1024 digital image. A print of the digital image and a sketch of the ground truth information provided are shown in Fig. 4.3. Five plant-cover types were chosen for the classification experiment. They were: wheat, potato, winter barley, coniferous trees under planting, and young coniferous trees. Twenty subimages, each of size 54 x 54, were obtained for each of the five agricultural classes. Thus, there were 100 sample subimages in all, and the five features developed in this work were computed for each sample.

In the classification, the method of training on the data was used, leaving out four samples each time for each of the classes, and training the classifier on the remaining sixteen. After this, the four left out for each class were presented for identification. Therefore, in all, there were five runs of training and testing, and for each run there was a total of eighty training samples and twenty testing samples. The mean values of the features for the five agricultural land-use classes are given in Table T4.3, while Table T4.4 shows the classification results for the different feature combinations. Fig. 4.4 gives the range plots of the features.

(a)



(b)

Fig. 4.3 (a) Aerial Photograph of Agricultural Area
(b) "Ground Truth" Map of Agricultural Area

CLASS	FEATURES				
	f_{cos}	f_{con}	f_{bus}	f_{com}	f_{str}
Wheat	0.00455	0.08118	0.06781	0.80708	2.42349
Potato	0.00442	0.19956	0.05063	1.84235	2.71362
Winter Barley	0.00473	0.29700	0.03348	4.86470	6.08927
Young Coniferous Trees	0.00275	1.59757	0.06393	70.74030	19.87066
Coniferous Trees Under Planting	0.00383	0.81911	0.02968	40.21661	16.23721

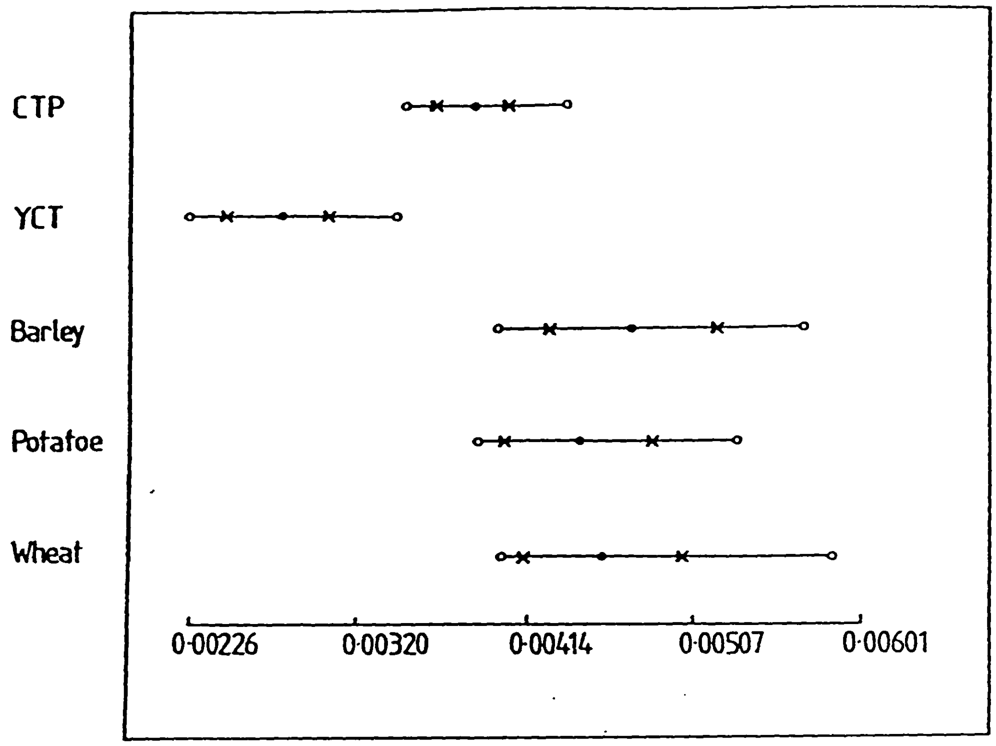
Table T4.3

Mean Values of Features

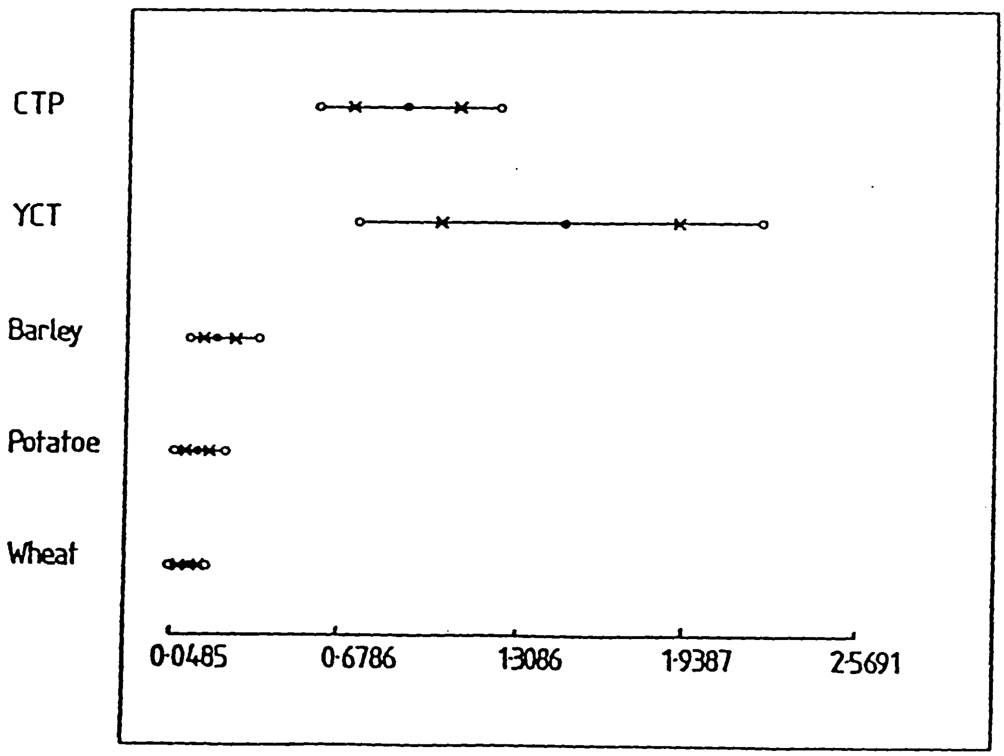
for Agricultural Land-Use Classes

FEATURES	NUMBER OF CORRECTLY CLASSIFIED SAMPLES PER CLASS					TOTAL NUMBER OF CORRECTLY CLASSIFIED SAMPLES	ACCURACY IN (%)
	WH	POT	WB	YCT	CTP		
f_{con}, f_{bus}, f_{str}	13	12	14	19	19	77	77
f_{cos}, f_{con}, f_{bus}	16	13	17	19	17	82	82
f_{cos}, f_{bus}, f_{str}	14	12	16	20	19	81	81
f_{bus}, f_{com}, f_{str}	15	11	16	19	20	81	81
f_{con}, f_{bus}, f_{com}	11	15	16	20	17	79	79
f_{cos}, f_{con}, f_{com}	13	8	19	20	17	77	77
f_{cos}, f_{con}, f_{bus}, f_{str}	16	12	17	20	18	83	83
f_{con}, f_{bus}, f_{com}, f_{str}	14	13	17	19	18	81	81
f_{cos}, f_{con}, f_{bus}, f_{com}	16	13	17	20	18	84	84
f_{cos}, f_{con}, f_{bus}, f_{com}, f_{str}	15	13	16	20	18	82	82

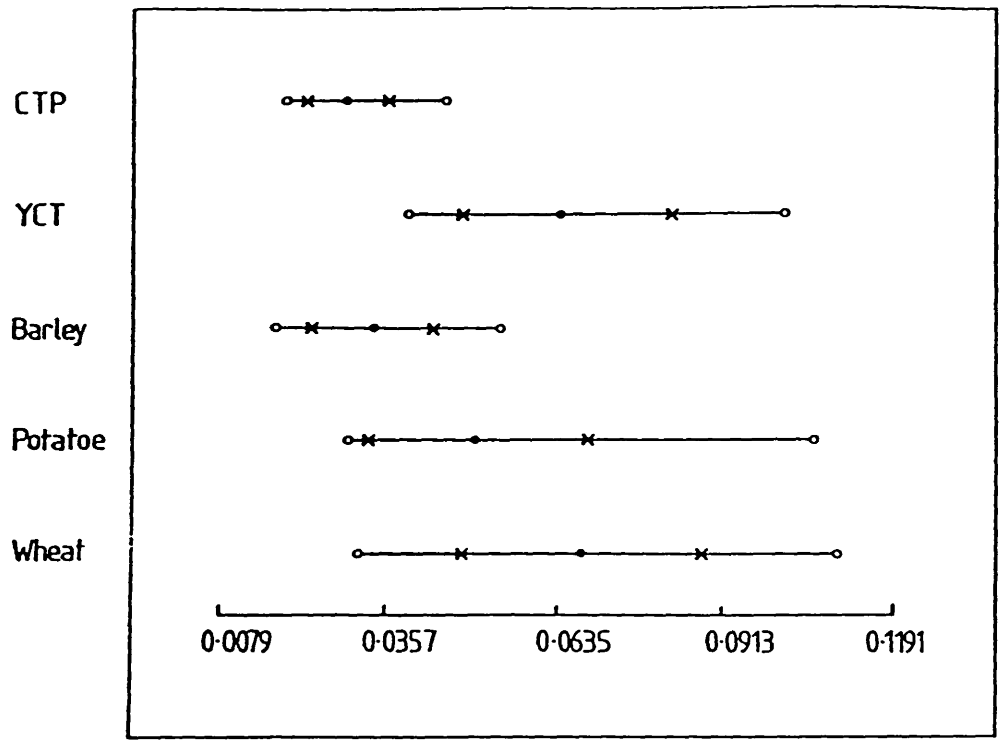
Table T4.4
Classification Results for Agricultural Land-Use Classes Using Features Developed in this Work
(WH=Wheat; POT=Potato; WB=Winter Barley; YCT=Young Coniferous Trees; CTP=Coniferous Trees Under Planting)


CTP
YCT
Barley
Potatoe
Wheat
0·00226
0·00320
0·00414
0·00507
0·00601

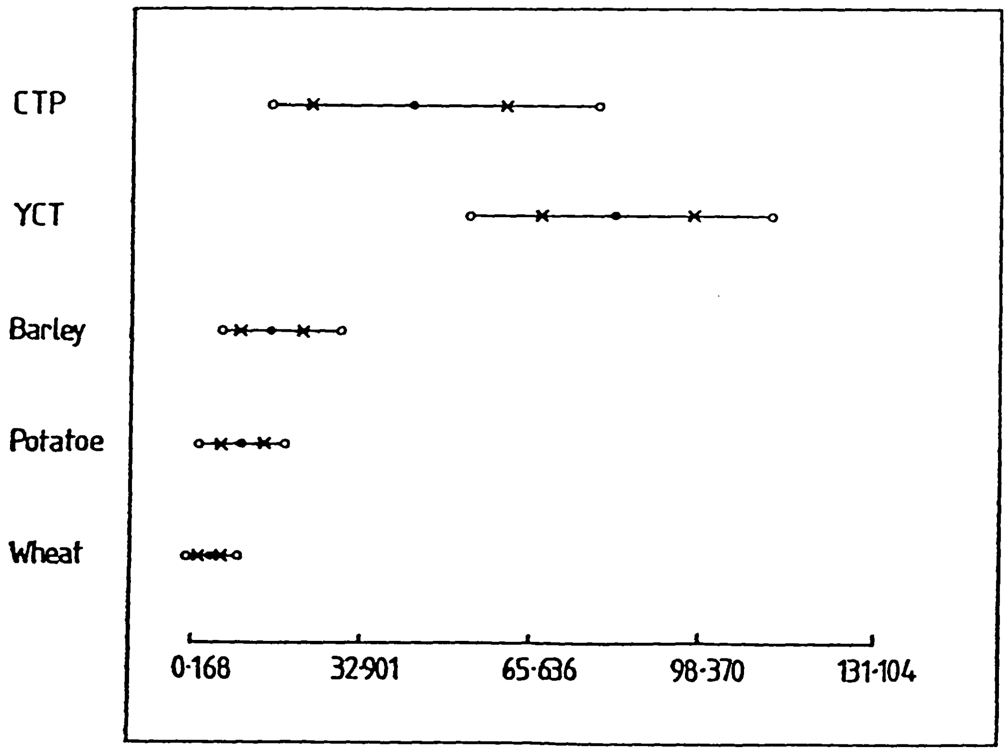
g. f_{cos}


CTP
YCT
Barley
Potatoe
Wheat
0·0485
0·6786
1·3086
1·9387
2·5691

b. f_{con}


CTP
YCT
Barley
Potatoe
Wheat
0·0079
0·0357
0·0635
0·0913
0·1191

c, f_{bus}


CTP
YCT
Barley
Potatoe
Wheat
0·168
32·901
65·636
98·370
131·104

d, f_{com}

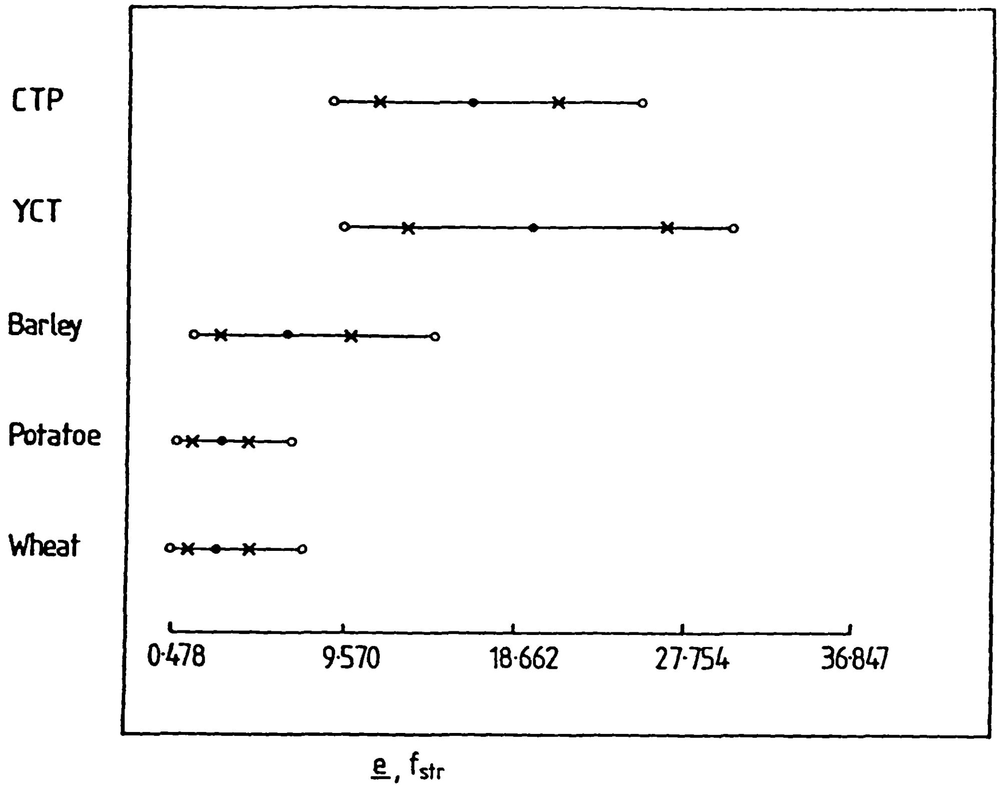


Fig. 4.4 Range Plots of Features for Agricultural Land-Use Classes

CTP = Coniferous Trees Under Planting
YCT = Young Coniferous Trees

4.2.3 Comparison of Perception-Related Features with Classical Texture Analysis Methods

For the purpose of comparison of performance, the two classification experiments described above were repeated using features from two existing analysis techniques:

(i) the Spatial Gray Level Dependence Method (SGLDM), developed by Haralick et al [32], and

(ii) the Gray Level Difference Method (GLDM), of Weszka et al [80]

These two methods are considered in the literature to be the best texture analysis techniques [13,80].

Four features have been used in each of the methods. They are: Angular Second Moment (ASM), Contrast (CON), Correlation (COR), and Entropy (ENT), for the SGLDM. For the GLDM, the features are: contrast (con), angular second moment (asm), entropy (ent), and Mean (MN). The first three features are abbreviated in small letters in order to distinguish them from features of the same name in the SGLDM. The SGLDM and GLDM are discussed in Appendix A-3, where the computational expressions for the above features are also given. For both methods, the distance used in feature computation was d=1; that is, an intersample spacing of 1. The classification results for the two methods, employing the same minimum error-rate classifier, are given in Table T4.5.

Considering a combination of four features in Tables T4.2 and T4.4, there is improved performance in terms of accuracy using the features developed in this thesis over the two classical techniques. Furthermore, the method that has been developed here is

CLASS	NUMBER OF CORRECTLY CLASSIFIED SAMPLES PER CLASS	
	ASM, CON, ENT, COR (SGLDM)	asm, con, ent, MN (GLDM)
1	9	12
2	10	3
3	10	11
4	12	12
5	7	9
6	8	7
7	12	10
8	9	11
9	11	12
10	2	8
11	12	11
12	6	11
TOTAL	108	117
ACCURACY IN (%)	75	81.25

(a) Natural Textures

CLASS	NUMBER OF CORRECTLY CLASSIFIED SAMPLES PER CLASS	
	ASM, CON, ENT, COR (SGLDM)	asm, con, ent, MN (GLDM)
Wheat	17	19
Potato	11	13
Winter Barley	17	19
Young Coniferous Trees	19	12
Coniferous Trees Under Planting	19	19
TOTAL	83	82
ACCURACY IN (%)	83	82

(b) Agricultural Land-Use

Table T4.5
Classification Results using Features from SGLDM and GLDM

computationally more efficient. For instance, the SGLDM and GLDM require about four times as much computation in extracting features from their respective matrices than does the method presented here. This is because, in the SGLDM and the GLDM, the features have to be computed over four matrices; whereas, for the approach presented in this work, there is only one matrix. Moreover, compared with the two earlier methods, the present technique requires less memory. For an image with gray level range 0 - 255, only one matrix of size 256 x 1 need be stored, whereas the SGLDM would require the storage of four matrices, each of size 256 x 256. The corresponding storage for the GLDM would be four 256 x 1 matrices. Thus, the computational cost involved in using the features developed here is small compared with the two classical techniques.

4.3 Textural Features for Image Partitioning

Two textural features are developed for texture-based partitioning of an image.

Let the average gray level in a window of size W centred on a pixel at point (i,j) be as defined in equation (3.1); that is

$$\bar{A}_{ij}(d) = \frac{1}{W-1}\left\{\left[\sum_{m=-d}^{d}\sum_{n=-d}^{d} F(i+m,\ j+n)\right] - F(i,j)\right\} \qquad \textbf{(4.2)}$$

where $\bar{A}_{ij}(d)$ is the average gray level of the window.

The size W is specified by a distance parameter d, and is given by $W = (2d+1)(2d+1)$. F(i,j) is the gray level of the pixel at the point (i,j).

Also, let the difference between the pixel gray level F(i,j) and the mean $\bar{A}_{ij}(d)$ be denoted as $S_{ij}(d)$. That is

$$S_{ij}(d) = F(i,j) - \bar{A}_{ij}(d) \tag{4.3}$$

Then, the following two features are defined for the pixel at the point (i,j)

(i) $$f_1(i,j) = \sum_{d=1}^{L} | S_{ij}(d) | \tag{4.4}$$

(ii) $$f_2(i,j) = \sum_{d_1=1}^{L} \sum_{d_2=1}^{L} | S_{ij}(d_1) - S_{ij}(d_2) | \tag{4.5}$$

The distance L specifies the maximum window size for computing the matrix S_{ij}. This window shall henceforth be referred to as the feature window (W_f).

Equation (4.4) is a summation of the differences between a pixel gray level and the average gray level of its neighbourhood over different neighbourhood sizes. In coarse texture, the gray level of a pixel would be similar to that of its neighbours, while there tend to be differences in the case of fine textures. Therefore, f_1 would take higher values for fine textures than for coarse textures. The feature f_2 is essentially a summation of the differences between the average gray levels of different neighbourhood sizes centred on a pixel. For a very busy and/or fine texture, the spatial rate of change in intensity is high. Consequently, the average gray levels of neighbourhoods of different sizes would tend to be significantly different, and the value of f_2 would be high. Therefore, one may refer to f_1 and f_2 as the "local-level equivalents" of the features

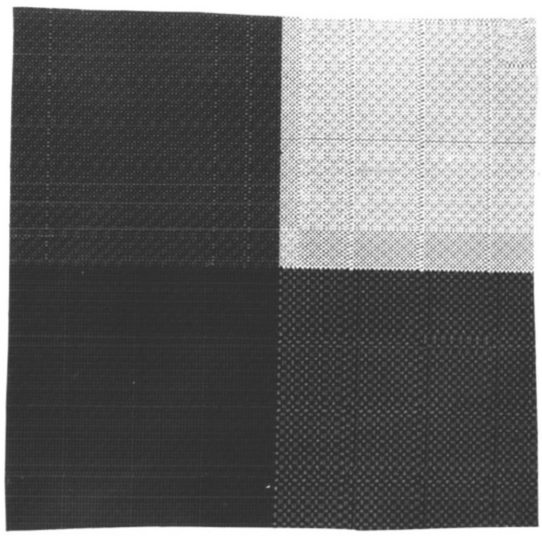
f_{cos} and f_{bus} (defined in section 3.4). Thus, using the two features, an image could be partitioned into component areas corresponding to textures with different levels of coarseness.

4.4 Image Partition Experiments and Results

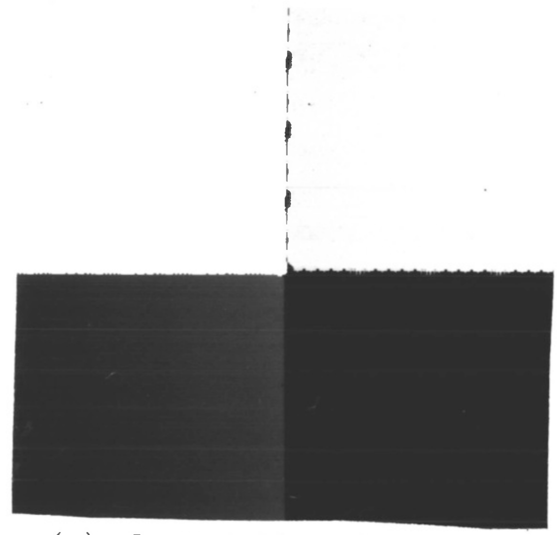
In the experiments, two test images were used. One is an artificially generated textured image, and the other a satellite image of a terrain. The images are shown in Fig. 4.5(a-b). Each image is of size 512 x 512 and consists of four different textured regions. The partitioning was performed by supervised classification of the pixels; that is, supervised segmentation. An area was specified for each texture region for training a classifier, and the classifier used was the minimum error-rate classifier. The two features were computed for each image pixel. However, the values actually used in the classification were the averages of the computed features in a much larger window centred on each pixel. This window shall be referred to as the characterization window W_c. The averaging was performed to minimize intra-region variation in pixel feature values. After averaging, the feature values of the pixels in the specified areas were used to train the classifier; subsequently, each pixel was classified. The size of W_f was fixed at 7 x 7, corresponding to L = 3, and that of W_c was 19 x 19. The resulting segmentations are shown in Fig. 4.5(c-d).

4.4.1 Segmentation Accuracy using Features

A further experiment was performed to evaluate the performance of the two features, and also to investigate the effect of varying sizes of W_f and W_c on the segmentation result. A composite image was created from the images of four natural textures (Fig. 4.6(a)), and

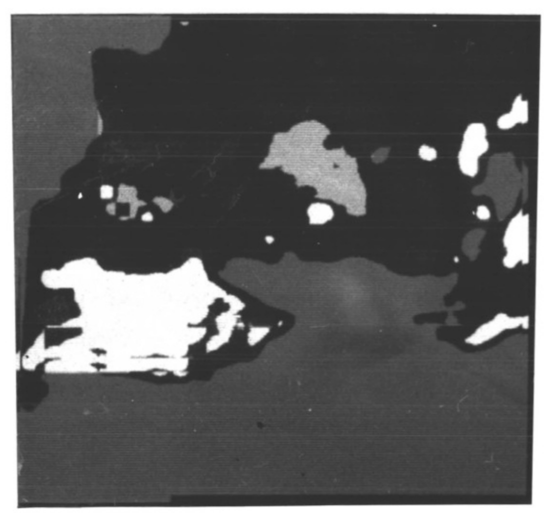


(a) Artificial Image



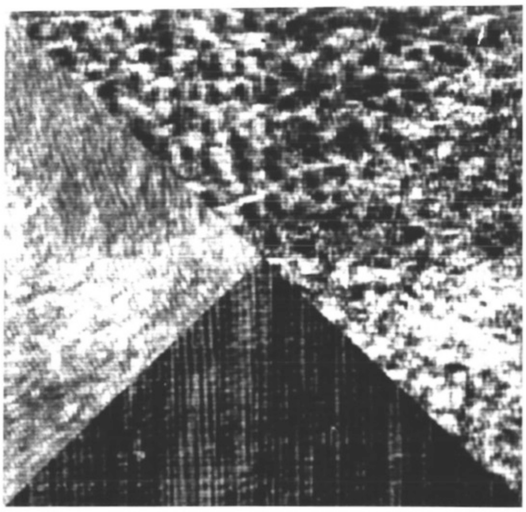
(c) Segmentation of Artificial Image

(b) Satellite Image of a Terrain

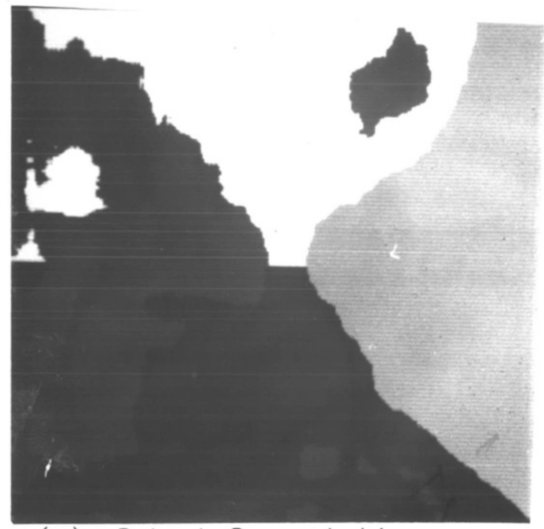


(d) Segmentation of Terrain Image

Fig. 4.5 Test Images for Textural Segmentation and Results

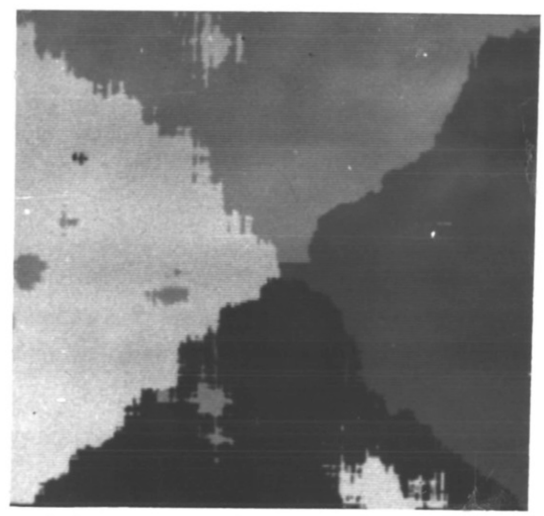


(a) Test Image for Segmentation Performance Evaluation



(c) Output Segmentation $W_f = 7x7$; $W_c = 25x25$

(b) Output Segmentation $W_f = 7x7$; $W_c = 19x19$



(d) Output Segmentation $W_f = 7x7$; $W_c = 33x33$

Fig. 4.6 Test Image for Evaluation of Segmentation Performance and Output Segmentations for Different Combinations of W_f and W_c

taken as a test image on which the accuracy of segmentation was evaluated. It was a 256 x 256 image, and each of the four texture regions consisted of (128 x 128) pixels. The segmentation accuracy was determined as a percentage ratio of the total number of pixels correctly labelled for their respective texture region to the total number of image pixels; that is

$$\text{Segmentation Accuracy} = \frac{\text{Total No of Pixels Correctly Classified in the Four Texture Regions}}{\text{Total Number of Image Pixels}} \times 100\%$$

In the experiments, three sizes of W_f - 5x5, 7x7 and 9x9 - corresponding to L = 2, 3 and 4 respectively; and four sizes of W_c - 15x15, 19x19, 25x25 and 33x33 - were used. The accuracy of segmentation using the two features, and for different combinations of W_f and W_c, is shown in Table T4.6.

Feature Window, W_f	Characterization Window, W_c			
	15 x 15	19 x 19	25 x 25	33 x 33
5 x 5	74.43	83.33	87.75	92.83
7 x 7	76.01	83.47	88.07	92.89
9 x 9	75.08	82.98	88.05	90.46

Table T4.6

Accuracy of Segmentation for Combinations of Different Sizes of W_f and W_c (in Per Cent)

The corresponding segmentations for W_f and W_c combinations of (7x7, 19x19), (7x7, 25x25), and (7x7, 33x33) are presented in Fig. 4.6(b-d) respectively.

From the results in Table 4.6, it is clear that there is no practical difference in accuracy for the three sizes of W_f for the same size of W_c. This indicates that the ability of the features to discriminate between the texture regions is not significantly dependent upon the choice of feature window size. On the other hand, the segmentation accuracy shows improvement when the size of W_c is increased. Thus, the use of a large characterization window is desirable, as this improves the quality of segmentation. Fortunately, as the partitioning is classification-based, the use of a large characterization window did not lead to what has been referred to in the literature as the "window problem" [64]. This is because the classifier is able to assign pixels to the correct region, even if the window overlaps two or more regions of different textures.

4.5 Conclusion

A set of images covering a wide range of texture classes was classified on the basis of their textures. The perception-related features developed in this work, and also features from two classical texture analysis approaches (the spatial gray level dependence method and the gray level difference method), were used in two image classification experiments. Two features for texture-based image segmentation were also developed and applied in the supervised segmentation of three images consisting of different textured regions.

As regards classification, the features developed in this work produced higher classification accuracy compared with the features from the two classical techniques. The method presented here also involved less computational cost compared with the two classical methods, as it required less computation and less memory. The classification results showed that the combination of f_{cos}, f_{con} and f_{bus} was the best of the six combinations of three features. For the four-feature category, the combinations of f_{cos}, f_{con}, f_{bus} and f_{com}; and f_{con}, f_{bus}, f_{com} and f_{str} produced comparable results, and these results were about the same as those produced using all five features. Thus, either of these two combinations can be considered as adequate for image classification problems.

In the case of segmentation, the two features developed in this work produced satisfactory results. Further experiments, carried out to evaluate the performance of the features, and also to investigate the effect of variations in the size of windows used in feature computation, produced high levels of segmentation accuracy. The results indicated that the choice of feature window size does not significantly affect the output segmentation; while there was general improvement with increase in the size of the characterization window. However, the larger the sizes of windows, the greater is the cost of computation. In any case, the choice of characterization window size may be a function of the degree of coarseness of the textures involved. For instance, for very fine textures, small sizes may be used, while for coarse textures, large sizes are recommended. This is to ensure that averaging is done over a number of texture primitives.

Lastly, it is pertinent to mention that the two features may not be able to distinguish between two regions of different textures with comparable levels of coarseness but different contrast. A third feature that conveys information about contrast would be necessary in such situations. Such a feature might be the variance of the gray level values of pixels within the characterization window centred on each pixel. The results obtained using this additional feature in similar segmentation experiments are presented in Chapter Six.

CHAPTER FIVE

WEIGHTED-FEATURE MINIMUM DISTANCE CLASSIFIER

5.1 Introduction

There are two kinds of approaches which are generally used in the design of classifiers. One set of techniques employs statistical criteria in classification decision making, while methods in the second group are distribution-free schemes which make use of simple measures of similarity or distance metrics. Some statistical methods are the Bayesian rule, the maximum likelihood rule, the min-max rule, the Neyman-Pearson rule, and a host of discriminant functions. Statistical classifiers are described in [18,22,66].

In the use of statistical criteria, two inherent assumptions are made about the data; normal density as the underlying probability distribution, and independence between samples in the data set. However, for images, there is some degree of dependency between samples. Secondly, the assumption of normality is not valid in many applications, as the image data are not normal [15]. Therefore, for general applicability, it is necessary that distribution-free techniques are considered in the design of image classifiers. Moreover, techniques which assume a distribution are generally complex in design and computationally expensive, as compared with techniques which are distribution-free and employ simple distance metrics.

The most commonly used, and perhaps also the simplest distance metric is the Euclidean distance. However, a Euclidean-distance classifier has two major disadvantages. One is the dominance of distance calculations by features having large numerical values. The second is the equal weighting of the features in the classification decision making. In most situations, however, the ability of some features to discriminate between the classes is greater than that of others. It is desirable that the contribution of such features to the decision-making process be increased, so that the effectiveness of these features is enhanced.

The usual approach for eliminating the first disadvantage is to normalize the features, such that each has zero mean and unit variance [18]. However, such a normalization process takes a fair amount of computation time, as it involves the calculation of feature values and the standard deviations in values.

In the approach developed here, the design is still based on the Euclidean distance metric, but a different kind of normalization procedure is used. The features are normalized in such a way that they are constrained to take values between zero and one inclusive. The method requires only the computation of average feature values. Furthermore, in this scheme, the features are weighted so that the contribution of each feature to the classification decision depends on its relative ability to discriminate between the classes. Feature normalization is described in subsection 5.2.1, while the derivation of weighting factors for the features is presented in subsection 5.2.2. In section 5.3, the performance of the classifier developed here is compared with that of the maximum likelihood and Euclidean-distance classifiers.

In this regard, three experiments were performed. Both the classifier designed here, and the Euclidean-distance classifier, were applied to the image classification problems described in subsections 4.2.1 and 4.2.2; namely, identification of natural textures, and texture-based classification of agricultural land-use categories. In the third experiment, the three classifiers were applied in the classification of image blocks belonging to five agricultural land-cover types using spectral features.

5.2 Design of Classifier

Let there be c number of classes with the ith class having a mean feature vector $\bar{\mathbf{X}}_i$, where $\bar{\mathbf{X}}_i$ is a column vector having m components. That is,

$$\bar{\mathbf{X}}_i = \begin{bmatrix} \bar{x}_{i1} \\ \bar{x}_{i2} \\ \cdot \\ \cdot \\ \cdot \\ \bar{x}_{im} \end{bmatrix} = \begin{bmatrix} \bar{x}_{ik} \end{bmatrix} \quad , k = 1,2,\ldots, m$$

Also, let $\mathbf{Y}$ be an unknown sample vector given by

$$\mathbf{Y} = \begin{bmatrix} y_1 \\ y_2 \\ \cdot \\ \cdot \\ \cdot \\ y_m \end{bmatrix} = \begin{bmatrix} y_k \end{bmatrix}$$

The Euclidean distance between the class i and sample $\mathbf{Y}$ is given by

$$d(i,Y) = \| \bar{X}_i - Y \|$$

$$= \sum_{k=1}^{m} (\bar{x}_{ik} - y_k)^2 \tag{5.1}$$

5.2.1 Feature Normalization

Now, let $\hat{x}_{ik}$ be a normalized version of $\bar{x}_{ik}$, given by

$$\hat{x}_{ik} = \beta \bar{x}_{ik} \tag{5.2}$$

subject to the condition that

$$\sum_{i=1}^{c} \hat{x}_{ik} = 1 \tag{5.3}$$

where β is a normalizing factor.

Therefore

$$\sum_{i=1}^{c} \beta \bar{x}_{ik} = 1 .$$

from which

$$\beta = \left[\sum_{i=1}^{c} \bar{x}_{ik} \right]^{-1} \tag{5.4}$$

Hence, from equation (5.2)

$$\hat{x}_{ik} = \bar{x}_{ik} \Big/ \sum_{i=1}^{c} \bar{x}_{ik} \tag{5.5}$$

This ensures that

$$0 \leq \hat{x}_{ik} \leq 1, \qquad \text{for all i and k}$$

The normalized Euclidean distance between the ith class and sample **Y** is given by

$$\hat{d}(i,\mathbf{Y}) = \sum_{k=1}^{m} (\hat{x}_{ik} - \hat{y}_k)^2 \tag{5.6}$$

where $$\hat{y}_k = \beta y_k = y_k \Big/ \sum_{i=1}^{c} \bar{x}_{ik} \tag{5.7}$$

5.2.2 Feature Weighting Factors

One of the simplest indicators of the degree of difference between two regions or categories (say i and j) in terms of feature k is the "contrast" between them in terms of that feature. Levine and Nazif [43] give an expression for this as:

$$d_{ijk} = \frac{|\bar{x}_{ik} - \bar{x}_{jk}|}{\bar{x}_{ik} + \bar{x}_{jk}} \tag{5.8}$$

where $\bar{x}_{ik}$ and $\bar{x}_{jk}$ are the mean values of the feature k for the two categories. It is reasonable to take the quantity d_{ijk} as a measure of the ability of the kth feature to separate between the classes i and j. A large value for d_{ijk} would indicate a high measure of separability.

For c number of categories, one can define a total measure of separability for the kth feature as the summation of the pairwise contrast measures between the classes. Denoting this as D_k, it is given by

$$D_k = \sum_{i=1}^{c-1} \sum_{j=i+1}^{c} d_{ijk} \tag{5.9}$$

$$= \sum_{i=1}^{c-1} \sum_{j=i+1}^{c} \frac{|\bar{x}_{ik} - \bar{x}_{jk}|}{\bar{x}_{ik} + \bar{x}_{jk}} \tag{5.10}$$

Then, the feature whose value of k, (k = 1,2,...., m), for which D_k is largest, has the greatest ability to discriminate between the classes and would have the largest weight in the classification decision making. However, as the classification is based on minimum distance, the weighting factor for each feature is such that the value of the weight is smallest for the feature with the highest measure of separability.

Therefore, the weighting factor denoted as W_k is then given by

$$W_k = \frac{D_n}{D_k} \tag{5.11}$$

where $D_n = \max_{\forall k \in m} \{D_k\}$

Thus the feature with the highest measure of separability has its weight always equal to 1, and the weights for the other features are greater than 1.

The normalized and weighted-feature Euclidean distance between the unknown vector $\mathbf{Y}$ and the ith class is therefore given (from equation 5.6) by

$$\tilde{d}(i,\mathbf{Y}) = \sum_{k=1}^{m} w_k \, (\hat{x}_{ik} - \hat{y}_k)^2 \tag{5.12}$$

The decision rule is that $\mathbf{Y}$ be assigned to the nth class if

$$\tilde{d}(n,\mathbf{Y}) = \min_{\forall i \in c} \{\tilde{d}(i,\mathbf{Y})\}$$

5.3 Application in Classification and Comparison with Maximum Likelihood and Euclidean-Distance Classifiers

The classifier designed here was applied in three classification problems, in order to evaluate its performance vis-a-vis the maximum likelihood classifier and the ordinary Euclidean-distance classifier (i.e. of the type given in equation 5.1). Three image classification experiments were performed. In two of them, textural features were employed, while in the third experiment, the input features were spectral.

5.3.1 Texture-Based Image Identification

The classifier developed in this work and the Euclidean-distance classifier were applied to the texture classification problems described in Chapter 4; identification of images of natural textures, and classification of agricultural land-use classes. Six combinations of features were used in the experiments. Four of the combinations consisted of the texture features that have been developed here, while the fifth and sixth combinations were the four features from the SGLDM and the GLDM respectively.

The training process for the classifier designed here involved the computation of the mean values of features for the classes and the determination of feature weighting factors. The training and testing of the classifiers were performed in exactly the same way as in subsections 4.2.1 and 4.2.2. That is, for the natural textures, the same training and testing samples as in the previous experiment were used, while for the agricultural land-use identification, the method of leaving-four-out was also used. The classification results are shown in Tables T5.1 and T5.2. The corresponding results for the maximum likelihood classifier for the given feature combinations (extracted from Tables T4.2 and T4.3) are also included. The results show that, given the same feature set, the accuracy of the classifier developed in this work is much better than that of the Euclidean-distance classifier, and is not far below that of the maximum likelihood classifier.

The mean and normalized mean values of features for the five agricultural land-use categories, obtained in one of the five classification runs and for the feature combination of f_{con}, f_{bus}, f_{com} and f_{str}, are given in Table T5.3 as an example. The corresponding weighting factors for each of the features are also

Features	Classification Accuracy of Classifiers (in Per Cent)		
	Maximum Likelihood Classifier	Weighted-Feature Minimum-Distance Classifier	Euclidean-Distance Classifier
f_{cos}, f_{con} f_{bus}	82.64	78.47	31.25
f_{cos}, f_{con} f_{com}, f_{str}	81.94	71.53	56.25
f_{con}, f_{bus} f_{com}, f_{str}	85.42	72.22	56.25
f_{cos}, f_{con} f_{bus}, f_{com}	84.03	74.31	31.25
ASM, CON ENT, COR (SGLDM)	75.00	64.58	35.42
asm, con, ent, MN (GLDM)	81.25	71.53	45.83

Table T5.1

Accuracy of Classifiers in the Identification of Natural Textures

Features	Classifiction Accuracy of Classifiers (in Per Cent)		
	Maximum Likelihood Classifier	Weighted-Feature Minimum-Distance Classifier	Euclidean-Distance Classifier
f_{cos}, f_{con} f_{bus}	77	76	74
f_{cos}, f_{con} f_{bus}, f_{str}	83	78	54
f_{con}, f_{bus} f_{com}, f_{str}	81	78	68
f_{cos}, f_{con} f_{bus}, f_{com}	84	78	74
ASM, CON ENT, COR (SGLDM)	83	72	47
asm, con ent, MN (GLDM)	82	73	64

Table T5.2
Accuracy of Classifiers in Agricultural Land-Use Classification

Classes	Mean Values of Features				Normalized Mean Values of Features			
	f_{con}	f_{bus}	f_{com}	f_{str}	f_{con}	f_{bus}	f_{com}	f_{str}
Wheat	0.08559	0.07088	0.76037	2.20420	0.02841	0.29245	0.00634	0.04544
Potato	0.19735	0.04669	1.91229	2.77607	0.06551	0.19265	0.01594	0.05723
Winter Barley	0.28854	0.03279	5.19142	6.65104	0.09578	0.13530	0.04328	0.13710
Young Coniferous Trees	1.62469	0.06058	74.74767	22.18845	0.53929	0.24996	0.62313	0.45739
Coniferous Trees under Planting	0.81648	0.03142	37.34303	14.69147	0.27102	0.12965	0.31131	0.30285

Feature Weighting Factors (W_k): f_{con} (1.289); f_{bus} (3.408); f_{com} (1.000); f_{str} (1.431)

Table T5.3
Mean Values, Normalized Mean Values, and Weighting Factors of Features for Agricultural Land-Use Categories

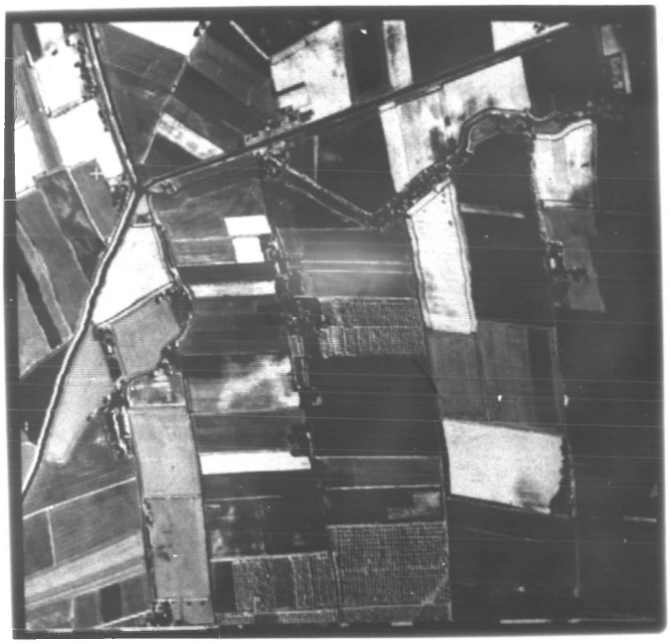
shown. For that particular run, the most effective feature in the classification decision making is f_{com}. Its effectivenes is about 3.4 times that of f_{bus}, 1.43 times that of f_{str}, and 1.29 times that of f_{con}; while the effectiveness of f_{con} is 2.65 times (i.e. 3.40/1.29) that of f_{bus}.

5.3.2 Spectral Classification of Agricultural Land-Cover Types

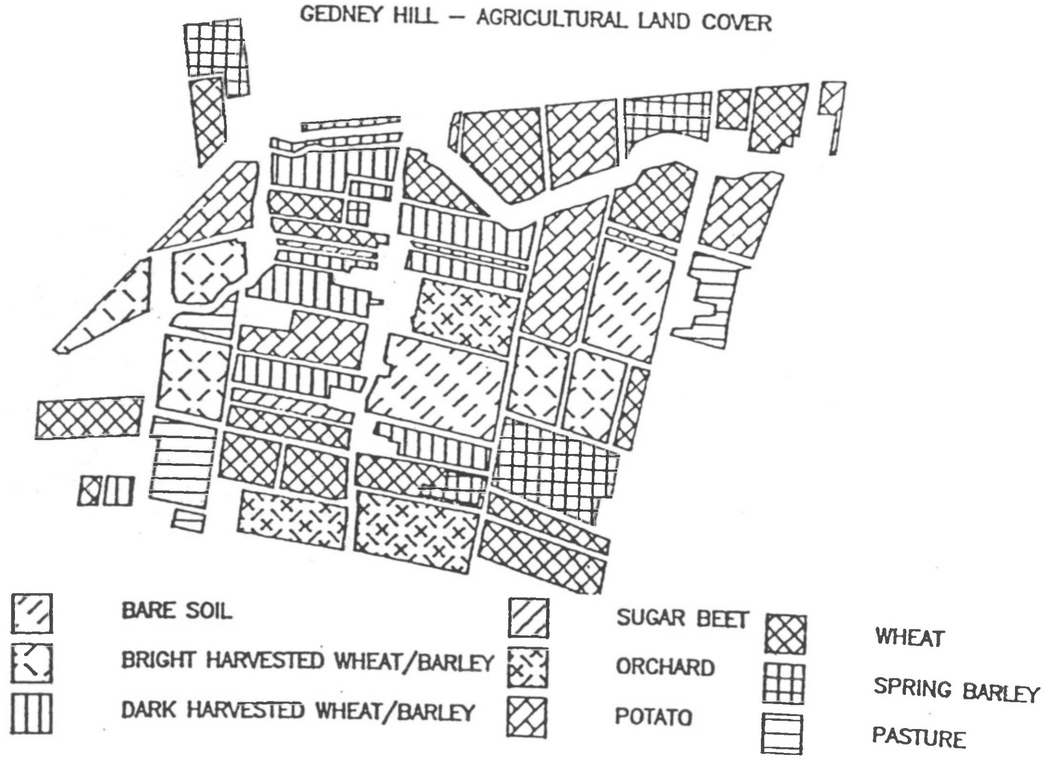
For this application, ninety image blocks, each of size 32 x 32, were obtained from a multispectral image of an agricultural area, situated near Gedney Hill, Lincolnshire, England. It is a 7-band Aerial Thematic Mapper (ATM) image. The ninety blocks belong to five agricultural land-cover classes: Orchard, Wheat, Potato, Spring Barley and Bare Soil. There were eighteen image blocks per class. The mean spectral responses (gray levels) of the blocks in three of the bands (bands 3,5,7) were used as input features to the classifiers. These bands were considered because they were found in earlier work[1] to be the three most decorrelated bands. A false colour composite[2] of the three bands taken in black-and-white, and the ground truth map for the image, are shown in Fig. 5.1. The mean spectral responses in each of the three bands were computed for each of the blocks. The mean values for a block therefore constitute a sample feature vector. Thus, there are eighteen sample vectors for

----

[1]Work done by the Remote Sensing Group at Imperial College, in the classification of agricultural crop types for the given image using spectral signatures.
[2]The three bands were displayed as a colour image on a colour monitor by feeding the bands 7, 5 and 3 images into the red, green and blue channels of the monitor respectively.



(a)



(b)

Fig. 5.1 (a) ATM Image of Agricultural Area
(b) "Ground Truth" Map for Image

each category. Employing the classification technique of training on the data, and in this case leaving three out at a time, all the samples were classified in six runs of training and identification.

The classification results using the classifier developed here, the maximum likelihood classifier, and the Euclidean-distance classifier, are given in Table T5.4. The results also show that the accuracy of the classifier designed here is comparable with that of the maximum likelihood classifier. The mean spectral responses for the five agricultural land-cover types, their corresponding normalized values, and the weighting factors for each of the three bands obtained in one of the classification runs, are given in Table T5.5. The weighting factors show that, using spectral responses, the five categories are most separable in band 7. The effectiveness of this band in classification decision is about 3.98 times that of band 3.

5.4 Conclusion

A minimum distance classifier based essentially on the Euclidean distance metric has been developed. In this scheme, the features are normalized such that they are constrained to take values betwen zero and one inclusive. The features are also weighted such that the contribution made by a feature in classification decision depends on its relative ability to discriminate between the classes. The classifier was used in three classification tasks in which the Euclidean-distance and maximum likelihood classifiers were also employed. In terms of accuracy, the classifier developed here is considerably better than the Euclidean-distance classifier, and

Classifier Type	Number of Correctly Classified Samples per Class: Orchard	Wheat	Potato	Spring Barley	Bare Soil	Total Number of Correctly Classified Samples	Accuracy (in Per Cent)
Maximum Likelihood	18	18	16	13	18	83	92.22
Feature-Weighted Minimum Distance	18	16	10	17	18	79	87.78
Euclidean-Distance	15	15	10	15	16	71	78.89

Table T5.4
Accuracy of Classifiers in Spectral Classification of Agricultural Land-Cover Types

Class	Mean Spectral Responses			Normalized Mean Spectral Responses		
	Band 3	Band 5	Band 7	Band 3	Band 5	Band 7
Orchard	59.87	51.45	67.12	0.1727	0.1671	0.1986
Wheat	77.74	74.12	52.98	0.2242	0.2407	0.1568
Potato	67.94	59.29	87.67	0.1959	0.1925	0.2594
Spring Barley	69.85	60.24	93.22	0.2014	0.1956	0.2758
Bare Soil	71.39	62.87	37.00	0.2059	0.2042	0.1095

Weighting Factors: Band 3 (3.982); Band 5 (2.893); Band 7 (1.000)

Table T5.5
Mean and Normalized Mean Values of Spectral Responses for Agricultural Land-Cover Types

comparable to the maximum likelihood classifier. However, in terms of implementation and speed, the classifier designed here is better than the maximum likelihood classsifier. For example, for an m-dimensional feature space, the amount of computation performed by the maximum likelihood classifier is proportional to m^2 (as the covariance and inverse covariance matrices used in decision making are m x m). For the classifier developed in this work, the amount of computation is, as for the Euclidean-distance classifier, proportional to m. Furthermore, the classifier presented here is simple to implement and requires no storage of matrices. Thus, this classifier has a high degree of accuracy and low computational cost.

CHAPTER SIX

IMAGE SEGMENTATION VIA AGGLOMERATIVE CLUSTERING OF UNIFORM NEIGHBOURHOODS

6.1 Introduction

A number of approaches have been developed for the segmentation of images. Some techniques seek within an image for points of abrupt changes or discontinuities in feature activity. Other techniques group together pixels which have sufficient degree of similarity in feature values, to form regions. In the development of a segmentation scheme, one may consider three conditions as being necessary for the good performance of the scheme. These conditions are particularly important for images of large scenes; for example, remotely sensed images of terrains.

(i) The method should be able to produce segmentation in which all areas corresponding to identical objects or to the same category, even if they are at different locations in the scene, appear the same in the segmented image.

(ii) The scheme should be able to use more than one feature simultaneously, as this enhances the characteristic differences between the different categories or objects.

(iii) The approach should be able to partition a scene into a given number of categories or regions depending on the level of detail desired - the hierarchical order of importance referred to by Morris and Constantinides [52].

The segmentation methods with the ability to meet all the above stated conditions are the clustering techniques and the region growing schemes. Clustering methods, in general, are computationally expensive, and may also require considerable memory. Region growing schemes of the graph-theoretic type [34,52,56,70,71] are also very demanding as regards memory requirement, and could be computationally expensive as well. On the other hand, those region growing methods which first identify uniform areas in an image and then grow regions from them [4,35,44,53,54,60] are generally less expensive, both in terms of computation and memory requirement. However, this latter type of method has two major drawbacks. One is the production of many regions; thus pixels belonging to identical objects, or to the same category at different locations in the scene, may be labelled differently. A second problem is the determination of "similar enough" criteria.

However, one can combine the region growing concept of seeking uniform areas with clustering techniques, to produce a segmentation scheme in which an image can be partitioned into a specified number of categories or regions; at the same time, the cost of computation, and the memory requirement, are minimized. The present method follows this approach. A description of the method is given in section 6.2. An important parameter used in the scheme, and called the uniformity criterion, is discussed in subsection 6.2.1. There are two variants of the algorithm, which are described in subsections

6.3.1 and 6.3.2. In section 6.4, the experimental results of segmentation are presented. Six different images were used as test images. These include: a human passport photograph, an X-ray image of part of a human hand, an outdoor scene, two satellite multi-spectral images of terrains, and a composite image consisting of parts belonging to three different texture classes. In the segmentations, spectral features, textural features, or a combination of both, were used, depending on the particular image. For the black-and-white and the monochrome images (passport photograph and X-ray), only the pixel gray levels were used as measures of brightness. Spectral features were employed in the case of the multispectral images, textural features for the composite image, and a combination of texture and brightness for the outdoor scene.

The conclusion to this chapter is given in section 6.5.

6.2 Segmentation Method

The segmentation method that has been developed in this work is a pixel classification based scheme employing clustering and region growing techniques. In this approach, the number of categories into which an image is to be partitioned is specified. The image is first divided into a number of non-overlapping neighbourhoods (square blocks). On the basis of a defined criterion (described in subsection 6.2.1), those neighbourhoods that could be considered uniform in terms of all features are located in an image. The mean feature values of such neighbourhoods constitute feature vectors, which are agglomeratively clustered to produce the mean feature vectors for the different categories present in the image. These mean feature vectors are then used to classify the image pixels.

Two assumptions are made in this development. One is that there is at least one uniform neighbourhood representative of each of the categories present in the image. The second is that the feature vectors of neighbourhoods representative of a particular category are similar to each other, and different from those of neighbourhoods belonging to other categories.

Suppose an image is to be partitioned into n number of categories, and N number of uniform neighbourhoods are found in the image ($N > n$). Therefore, there are N feature vectors to be clustered. Using a normalized Euclidean distance as a measure of similarity, the two most similar feature vectors are determined. These two vectors are considered to be from neighbourhoods belonging to the same category. The two vectors are "merged" together, and the number of vectors is reduced by one. This merging process is peformed iteratively until the number of mean feature vectors equals the specified number of categories. Thus, the merging process is the same as agglomerative clustering of the uniform neighbourhoods. At any stage in the iteration, a "resultant" mean feature vector, and the number of image pixels used in its determination, is as follows:-

Consider $\bar{\mathbf{X}}_i$ and $\bar{\mathbf{X}}_j$ to be the two most similar mean vectors. Let N_i be the number of pixels (already) used in determining $\bar{\mathbf{X}}_i$, and N_j the corresponding one for $\bar{\mathbf{X}}_j$.

Given that the vectors are m-dimensional, defined by

$$\bar{\mathbf{X}}_i = \begin{bmatrix} \bar{x}_{i1} \\ \bar{x}_{i2} \\ \\ \cdot \\ \cdot \\ \cdot \\ \bar{x}_{im} \end{bmatrix} = \begin{bmatrix} \bar{x}_{ik} \end{bmatrix} \quad \text{and} \quad \bar{\mathbf{X}}_j = \begin{bmatrix} \bar{x}_{j1} \\ \bar{x}_{j2} \\ \\ \cdot \\ \cdot \\ \cdot \\ \bar{x}_{jm} \end{bmatrix} = \begin{bmatrix} \bar{x}_{jk} \end{bmatrix}$$

$$k = 1,2,\ldots,m$$

the resultant mean value for the kth feature (component) resulting from the merging of $\bar{\mathbf{X}}_i$ and $\bar{\mathbf{X}}_j$ is given by

$$\bar{x}_{ijk} = \frac{N_i\bar{x}_{ik} + N_j\bar{x}_{jk}}{N_i + N_j} \tag{6.1}$$

and the corrresponding resultant mean feature vector is

$$\bar{\mathbf{X}}_{ij} = \begin{bmatrix} \bar{x}_{ij1} \\ \bar{x}_{ij2} \\ \\ \cdot \\ \cdot \\ \cdot \\ \bar{x}_{ijm} \end{bmatrix} = \begin{bmatrix} \bar{x}_{ijk} \end{bmatrix} \tag{6.2}$$

The total number of pixels used in determining $\bar{\mathbf{X}}_{ij}$ is given by

$$N_{ij} = N_i + N_j \tag{6.3}$$

In the experiments that were performed, the feature normalization procedure described in subsection 5.2.2 was used in the determination of the most similar vectors during the agglomerative clustering process. For the classification of the image pixels, the weighted-feature minimum distance classifier was employed. However, if a different kind of normalization and/or classifier is employed which requires the use of variances in feature values, the variance for the kth feature can be updated during the clustering iteration using the following expression:

$$s_{ijk}^2 = \frac{N_i(\bar{x}_{ik}^2 + s_{ik}^2) + N_j(\bar{x}_{jk}^2 + s_{jk}^2)}{N_i + N_j} - \bar{x}_{ijk}^2 \tag{6.4}$$

where $\bar{x}_{ik}$ and $\bar{x}_{jk}$ are the kth components of the ith and jth vectors (that is, the mean values of the kth feature for the ith and jth clusters); and s_{ik}^2 and s_{jk}^2 are the corresponding variances. The proof of equation (6.4) is given in Appendix A-4.

6.2.1 Uniformity Criterion

The uniformity criterion as used in the experiments is defined as follows:

A neighbourhood (block) is considered to be uniform provided that the ratio of the minimum of the mean feature value for the neighbourhood and the corresponding value for a quarter of the neighbourhood to the maximum of the two mean values be no smaller than a certain threshold, α, for all features and for each of the quarters, i.e.

$$\frac{\min\{\bar{f}_{neighbourhood}, \ \bar{f}_{quarter}\}}{\max\{\bar{f}_{neighbourhood}, \ \bar{f}_{quarter}\}} \geq \alpha \quad \text{for all features and for each of the quarters}$$

where $\bar{f}_{neighbourhood}$ = mean feature value for neighbourhood

$\bar{f}_{quarter}$ = mean feature value for a quarter

α is the threshold

The criterion implies that, for a neighbourhood to be considered uniform, there should be no significant difference between the mean values of features for the neighbourhood and for each of its quarters. Chen and Pavlidis [9] also suggest this type of criterion for determining region uniformity.

6.3 Segmentation Algorithms

A satisfactory peformance of the segmentation technique described depends to a considerable extent upon the two parameters, uniformity criterion and neighbourhood size. Either or both of them may be varied. In one implementation of the scheme, the sizes of neighbourhood were the same and fixed, while the uniformity criterion was varied. This is Algorithm I. In a second approach, Algorithm II, the uniformity criterion was fixed and the size of neighbourhood varied from one part of the image to another.

6.3.1 Algorithm I: Fixed Neighbourhood Size and Variable Uniformity Criterion

The choice of a value for the uniformity criterion is very important. A satisfactory value would lead to reasonably good segmentation results, as well as a reasonable cost of computation. Too relaxed a criterion may lead to the detection of many uniform neighbourhoods; consequently, many feature vectors would be used in the clustering iteration, and the process would take a long time. On the other hand, if the criterion is too strict, only a few neighbourhoods may be considered uniform, and all may well belong to the same category, or to a number of categories less than that desired. This would lead to poor results. Thus, a compromise has to be reached between a large number of detected uniform neighbourhoods as a result of too relaxed a criterion, and a few uniform neighbourhoods due to too strict a criterion.

A constraint was therefore introduced specifying the allowable maximum and minimum number of neighbourhoods that could be considered uniform. (In practice, it is desirable that this minimum is greater than the number of categories.) Thus, on the basis of these constraints, the initially stated value of α then becomes only a starting value, to be referred to as the starting uniformity criterion. Depending on the situation, it is either increased or decreased. An increase corresponds to making the criterion stricter, and a decrease represents a relaxation of the criterion. A flow chart of the algorithm is shown in Fig. 6.1.

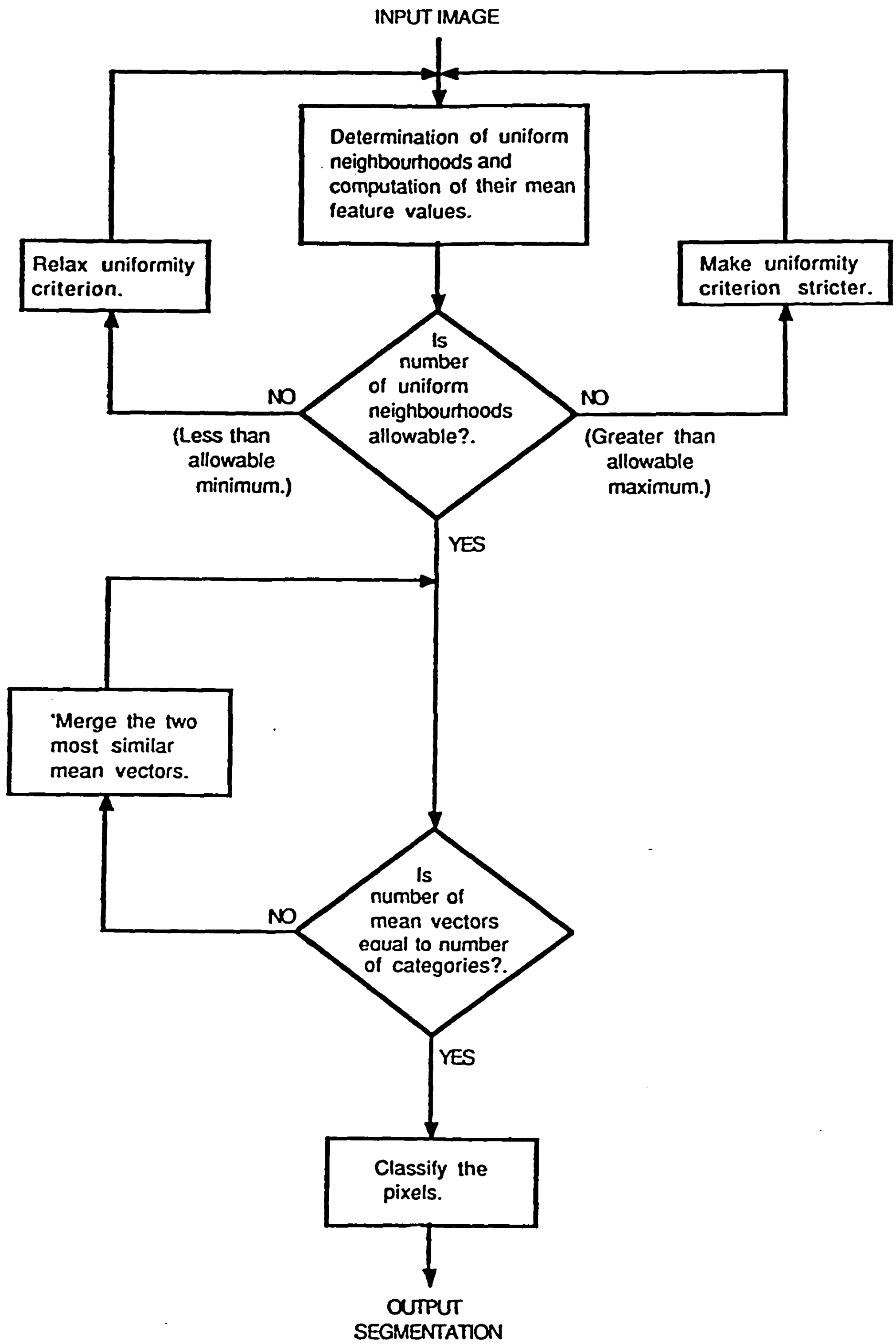


Fig. 6.1 Flow Chart of Segmentation Algorithm I.

6.3.2 Algorithm II: Variable Neighbourhood Size and Fixed Uniformity Criterion

The size of neighbourhoods used in seeking uniform areas is another parameter which is of fundamental importance in the segmentation scheme described here. Too large a size of neighbourhood may result in a small number of mutually exclusive neighbourhoods in the image, and many of these may well consist of parts belonging to different categories. Hence, they will be non-uniform. On the other hand, very small neighbourhoods are likely to be uniform. A large number of uniform neighbourhoods would be detected, resulting in a large number of vectors being clustered. This would be computationally expensive. In Algorithm I, the uniformity criterion would be made stricter in such situations under the imposed constraint of the allowable maximum number of uniform neighbourhoods.

However, in images where areas belonging to some categories are very uniform and large, and those belonging to other categories are not so uniform, no uniform neighbourhoods may be found for the latter categories. This is because, as the criterion is made stricter and stricter, it may become so stringent that uniform neighbourhoods are not found for some categories. As a result, these categories are missed out, and pixels belonging to them would be misclassified, leading to very poor results. Moreover, a feature vector obtained from a very small neighbourhood may not be a good representative of the particular category to which the neighbourhood belongs, especially if this is the only neighbourhood found for that category.

Thus, it is possible that the neighbourhood size which produces reasonably good results in one application may perform poorly in another. One solution to the problem is to use varying sizes of neighbourhoods for different parts of the image, depending on the degree of uniformity. For areas with a high level of uniformity, large neighbourhoods can be used, and smaller sizes for the not-so-uniform areas. This is done in Algorithm II. Varying sizes of neighbourhoods are obtained using a quad-tree approach.

The image is first divided into mutually exclusive neighbourhoods of size $N_1 \times N_1$. Each is tested for uniformity. Non-uniform ones are subdivided into four blocks, each of size $N_2 \times N_2$ ($N_2 = N_1/2$). The four blocks are tested for uniformity. Any one not found to be uniform is again split into four parts, each of size $N_3 \times N_3$ ($N_3 = N_2/2 = N_1/2^2$), and each part is tested. The splitting and testing for uniformity is continued until the desired smallest neighbourhood size $N_n \times N_n$ is reached, where $N_n = N_1/2^{n-1}$. The $N_1 \times N_1$ neighbourhood constitutes the largest search block for seeking uniform areas in the image, and corresponds to the level 1 neighbourhood of the quad-tree. The $N_n \times N_n$ neighbourhood is the smallest search block, and corresponds to the level n neighbourhood of the quad-tree. A flow chart of the algorithm is shown in Fig. 6.2.

6.4 Segmentation Experiments and Results

The feasibility of the segmentation techniques developed here was evaluated using six different images as test images. These include: two satellite multispectral images of terrains; a human passport photograph; an X-ray image of part of a human hand; an outdoor scene; and a composite image of three texture types. The passport photograph, X-ray image and composite texture image are

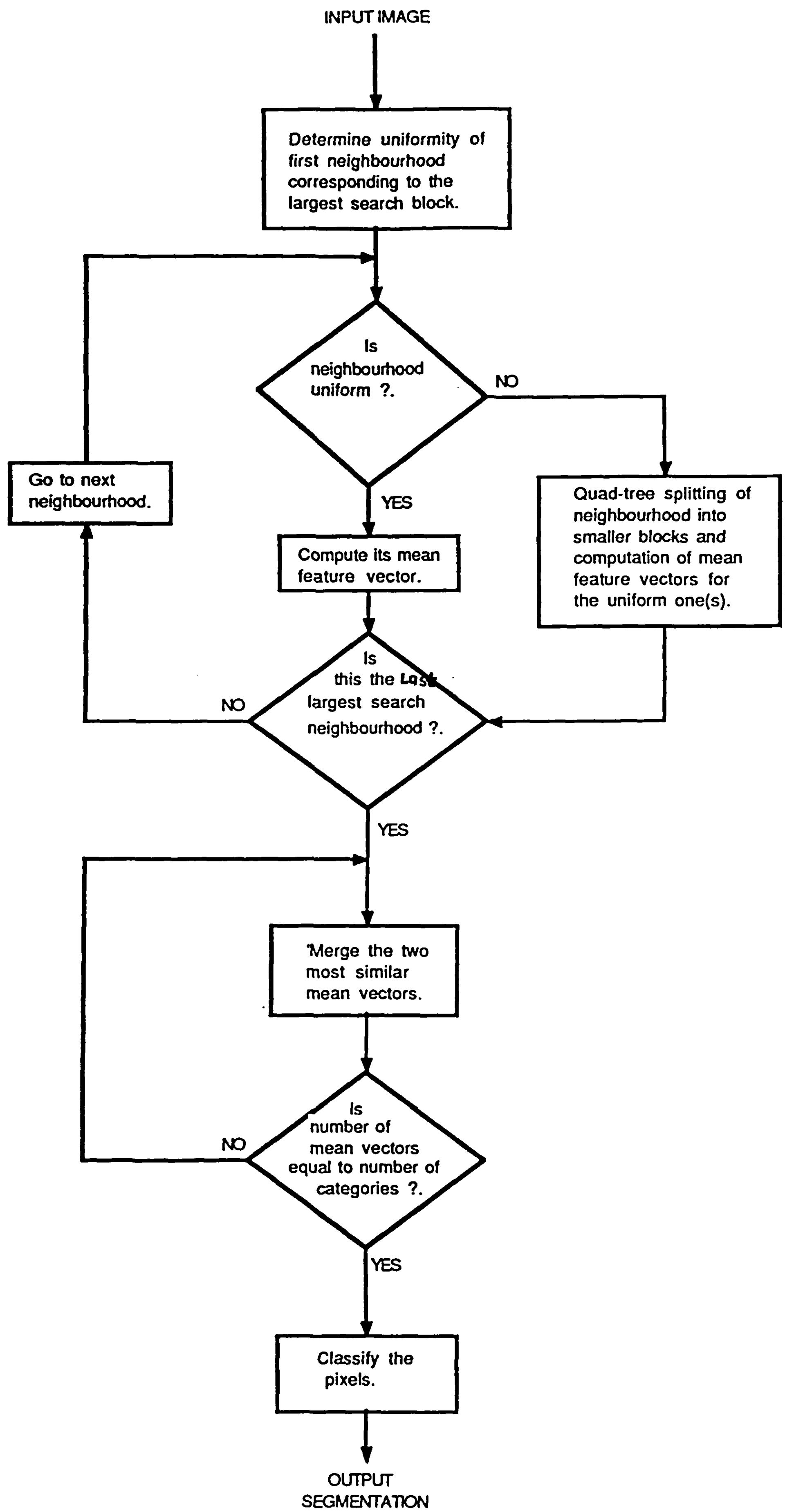


Fig. 6.2 Flow Chart of Segmentation Algorithm II.

256 x 256 images, while the outdoor scene and multispectral images are of size 512 x 512. One of the multispectral images is a Landsat multispectral scanner (MSS) image, while the other is a thematic mapper (TM) image. The passport photograph and X-ray image are gray tone dominated images; that is, they consist of parts that are essentially different only in their levels of brightness.

In all the experiments, the values of the required segmentation parameters were fixed as follows:-

Algorithm I

Starting Uniformity Criterion, α: 0.95

Incremental/Decremental Value : 0.001

Neighbourhood Size: 16 x 16

Allowable Maximum Number of
Uniform Neighbourhoods:

(i) Images of Size 512 x 512 -
One-third of the number of mutually exclusive neighbourhoods in image (i.e. one-third of 1024 = 341)

(ii) Images of Size 256 x 256 -
Three-quarters of the number of mutually exclusive neighbourhoods in image (i.e. three-quarters of 256 = 192)

Allowable Minimum Number of
Uniform Neighbourhoods: One-third of the allowable maximum

Algorithm II

Uniformity Criterion, α: 0.97

Largest Search Neighbourhood (N_1 x N_1): 64 x 64

Smallest Search Neighbourhood (N_n x N_n): 16 x 16

6.4.1 Segmentation of Multispectral Images

(a) Multispectral Scanner (MSS) Image

The MSS image is a scene in West Central Nigeria - Kainji Lake and its surrounding lands. The images in three of the spectral bands (bands 4, 5 and 7) were used in the experiment. These images are shown in Fig. 6.3(a-c). A human expert who is familiar with the area, and who is also a remote sensing scientist, supplied the "ground truth information". For this image, the expert categorized the area into four main land-cover types. These are:

(1) water body (i.e. Lake Kainji and parts of the river Niger)
(2) areas of good vegetation (tree cover)
(3) areas corresponding to burnt grassland
(4) farmlands

The expert also identified areas in other portions of the Landsat image (not part of the test image) that are representative of each of the four categories. The mean spectral responses from these areas are as follows:

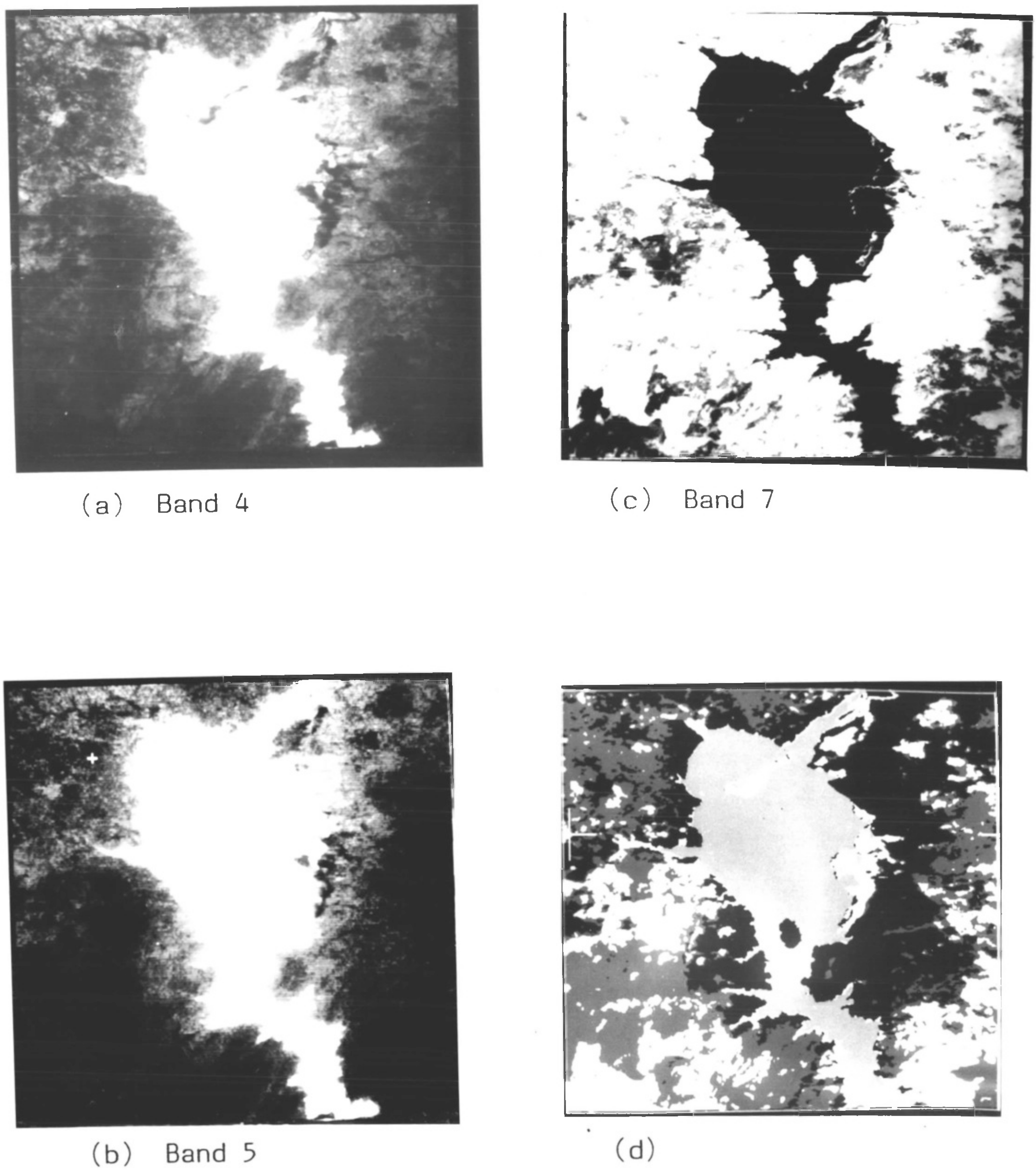


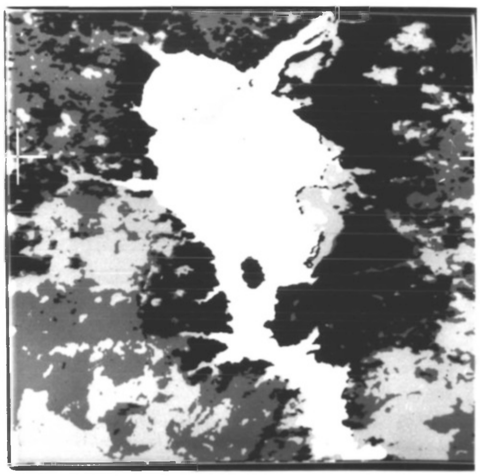
Fig. 6.3 (a-c) Test Landsat MSS Image
(d) Four-Category Partition of MSS Image by Supervised Classification using Training Areas Provided by Expert

Band 4	Band 5	Band 7	Category
85.75	115.32	46.13	Water Body
61.36	79.73	82.74	Tree Cover
60.92	78.52	70.01	Burnt Area
65.05	87.06	88.67	Farmland

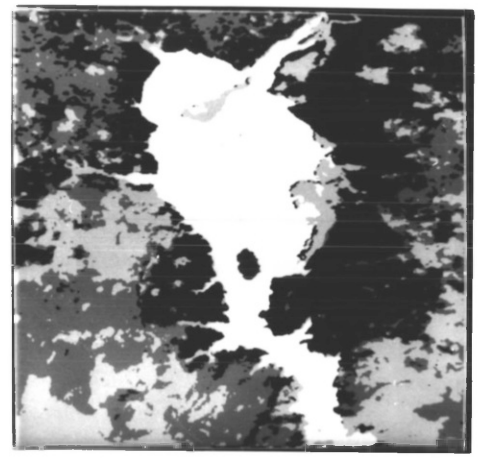
Table T6.1
Mean Spectral Responses of Representative Area of Each Category

These mean values were used in the classification of the pixels. The resulting segmentation is shown in Fig. 6.3(d). The image was partitioned using the two algorithms. For Algorithm I, two other neighbourhood sizes, 12 x 12 and 14 x 14, were also used, in addition to the 16 x 16 size, to investigate the effect of different neighbourhood sizes on the results. Thus, the image being of size 512 x 512, the allowable maximum number of uniform neighbourhoods for these two neighbourhood sizes corresponds to 588 and 432 respectively.

The mean spectral responses obtained from the algorithms for the four categories are given in Table T6.2. The corresponding segmentations are shown in Fig. 6.4(a-d). The CPU process times for each segmentation are also indicated for a VAX 11/780, Version VMS 4.1 computer. It is seen, from Table T6.2, that the mean spectral responses obtained from both algorithms are very similar, and close to those obtained from the representative areas provided by the expert (Table T6.1). In Table T6.3, the number of neighbourhoods considered uniform in Algorithm I at the initial value of α (i.e. 0.95), and for the three neighbourhood sizes, is presented. The

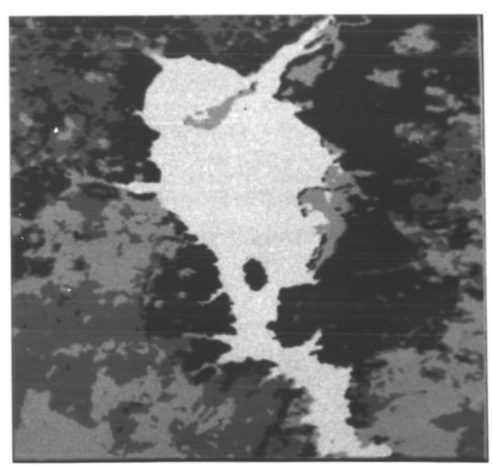


(a) 12 x 12
(CPU Time: 22 mins 46.09 secs)



(b) 14 x 14
(CPU Time: 12 mins 56.15 secs)

(c) 16 x 16
(CPU Time: 8 mins 29.67 secs)



(d)
(CPU Time: 7 mins 33.35 secs)

Fig. 6.4 Four-Category Partitions of MSS Image Generated by Algorithms:
(a-c) Algorithm I for the three neighbourhood sizes
(d) Algorithm II

NEIGHBOURHOOD SIZE	MEAN SPECTRAL RESPONSES BAND 4	BAND 5	BAND 7	CATEGORY TYPE
	85.66	119.33	46.13	Water Body
	61.67	80.24	83.49	Tree Cover
12 x 12	62.16	79.23	69.72	Burnt Area
	65.38	87.89	89.57	Farmland
	85.54	119.06	46.17	Water Body
	62.59	81.86	84.65	Tree Cover
14 x 14	60.76	78.40	69.24	Burnt Area
	66.07	89.55	90.41	Farmland
	85.73	119.27	46.12	Water Body
	61.50	79.87	82.45	Tree Cover
16 x 16	61.03	78.84	70.47	Burnt Area
	65.06	87.08	88.57	Farmland

(a)

BAND 4	BAND 5	BAND 7	CATEGORY TYPE
85.72	119.29	46.22	Water Body
62.73	81.96	83.72	Tree Cover
60.95	78.46	72.09	Burnt Area
65.79	89.01	87.72	Farmland

(b)

Table T6.2

Mean Spectral Responses Obtained for the Four Categories

(a) from Algorithm I for the Three Neighbourhood Sizes

(b) from Algorithm II

NEIGHBOURHOOD SIZE	NUMBER OF UNIFORM NEIGHBOURHOODS AT α = 0.95	FINAL NUMBER OF NEIGHBOURHOODS CONSIDERED UNIFORM UNDER THE ALLOWABLE MAXIMUM CONSTRAINT	FINAL VALUE OF α
12 x 12	1085	581	0.982
14 x 14	773	417	0.976
16 x 16	574	319	0.969

Table T6.3
Initial and Final Number of Neighbourhoods
for the Three Neighbourhood Sizes

final number of neighbourhoods considered uniform under the allowable maximum constraint, and the corresponding final value of α, are also included in this table. In Algorithm II, 221 feature vectors were clustered.

In one other investigation, two experiments were performed to determine how well the algorithm can partition an image depending upon the desired level of detail. In this regard, the following question was posed to the expert:

"If you were to partition this test image into

(i) three categories and

(ii) five categories

with each category being a significant proportion of the image, what would be the partitions?"

His answer is given below.

(i) For three-category partition:-

(a) Water body remains the same

(b) Burnt area remains the same

{The same means as in the four-category partition}

(c) Areas with plant cover, comprising the two categories of farmland and tree cover, because one would consider these two to be most similar in physical terms, as well as in reflectance values

(ii) For five-category partition:-

(a) Tree cover area remains the same

(b) Burnt area remains the same

(c) Farmland remains the same

(d) Silt water (a split of the category

(e) Clear Water of water body)

	MEAN SPECTRAL RESPONSES			CATEGORY TYPE
	BAND 4	BAND 5	BAND 7	
	85.76	119.33	46.13	Water Body
Algorithm I	60.92	78.52	70.01	Burnt Area
	63.76	84.46	86.58	Areas with Plant Cover
	85.72	119.29	46.22	Water Body
Algorithm II	60.95	78.46	72.09	Burnt Area
	63.68	84.14	85.81	Areas with Plant Cover

(a)

	MEAN SPECTRAL RESPONSES			CATEGORY TYPE
	BAND 4	BAND 5	BAND 7	
	83.12	113.68	44.43	Silt Water
	86.25	120.39	46.45	Clear Water
Algorithm I	61.38	79.73	82.74	Tree Cover
	65.06	87.06	88.67	Farmland
	60.92	78.52	70.01	Burnt Area
	82.79	113.41	43.98	Silt Water
	86.32	120.84	47.03	Clear Water
Algorithm II	62.58	81.93	83.70	Tree Cover
	65.78	89.05	87.97	Farmland
	61.02	78.49	72.13	Burnt Area

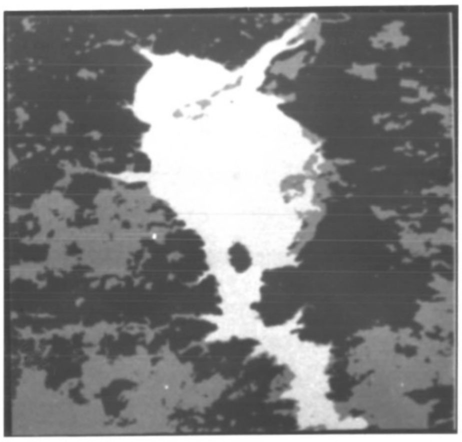
(b)

Table T6.4

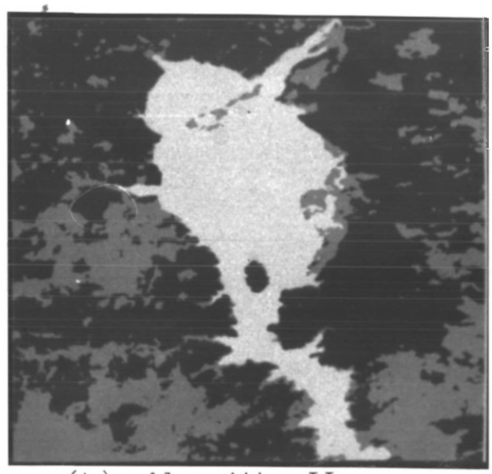
Mean Spectral Responses Obtained from Algorithms

(a) Three-Category Partitions and

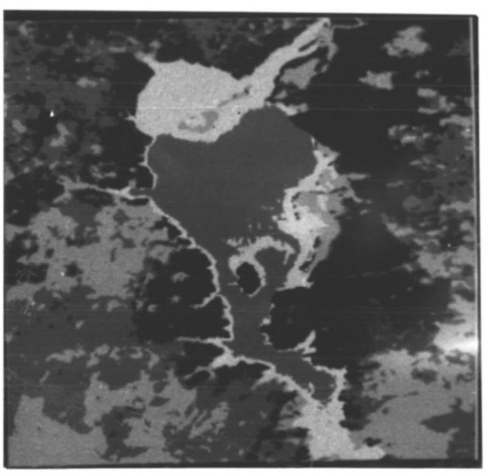
(b) Five-Category Partitions



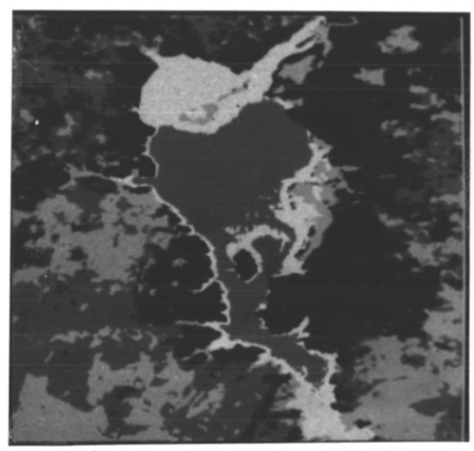
(a) Algorithm I
(CPU Time: 7 mins 36.13 secs)



(b) Algorithm II
(CPU Time: 7 mins 18.13 secs)



(c) Algorithm I
(CPU Time: 9 mins 44.05 secs)



(d) Algorithm II
(CPU Time: 8 mins 19.63 secs)

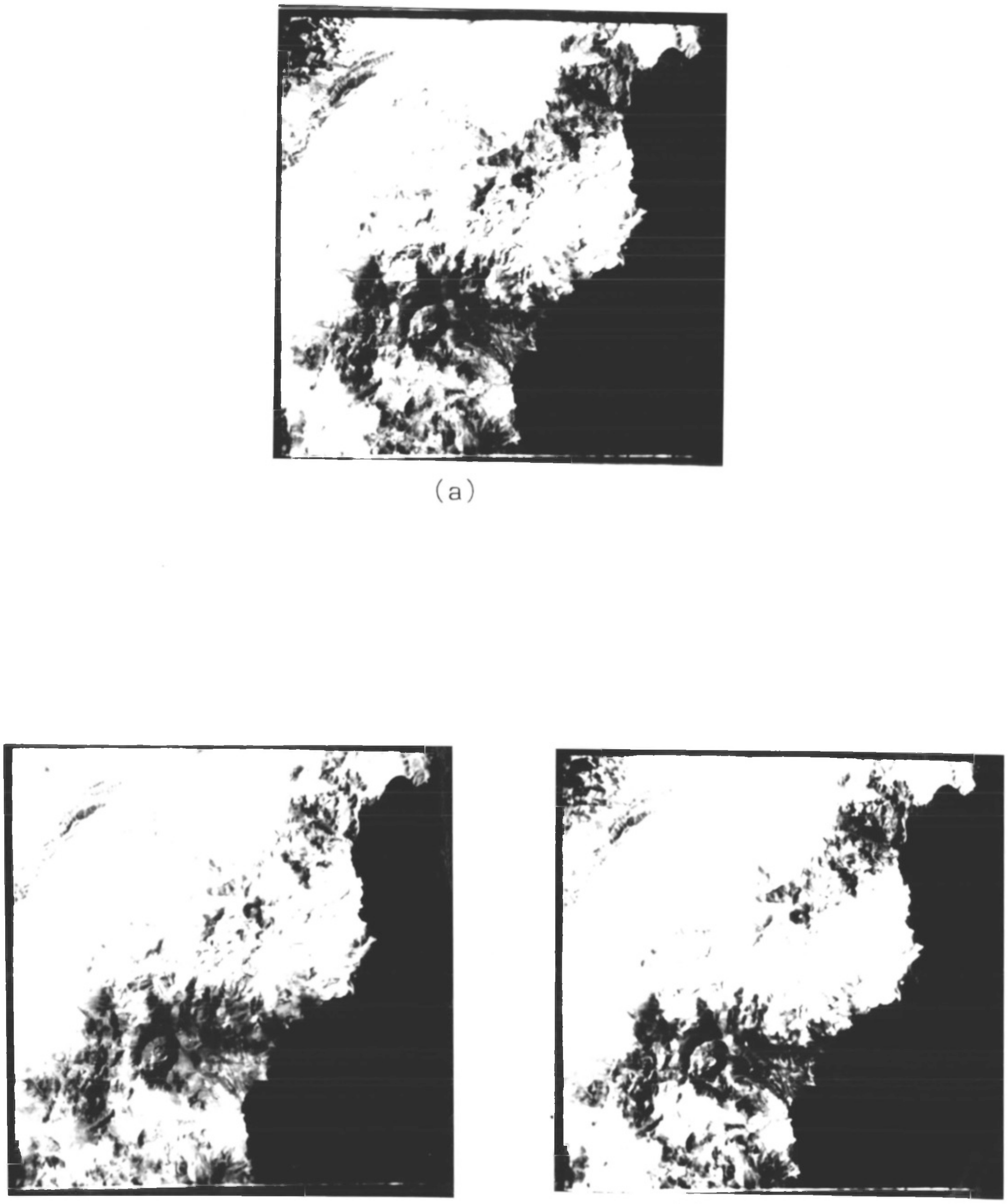
Fig. 6.5 (a and b) Three-Category Partitions of MSS Image
(c and d) Five-Category Partitions of MSS Image

In the first experiment, the number of categories present in the image was put at three. The mean spectral responses for the three categories are given in Table T6.4(a) for the two algorithms, and the corresponding segmentations are shown in Fig. 6.5(a) and (b). It is seen that the areas corresponding to the farmland and tree cover categories have been compounded into one, with the boundaries of the areas corresponding to water body and burnt area remaining more or less the same.

In the second experiment, the number of categories was put at five. The mean spectral responses for the categories are shown in Table T6.4(b). The segmentations (Fig. 6.5(c-d)) show the partitioning of the lake into two categories, while the boundaries of the categories of tree cover, burnt area and farmland remain more or less unchanged (compare Fig. 6.5(a-d) with Fig. 6.4(a-d)).

(b) Thematic Mapper Image

This is an image of a part of the east coast of Spain. The gray levels in three of the spectral bands (bands 4, 5 and 7) were also used as input features for the segmentation. The three bands are shown in Fig. 6.6(a-c). The sea is clearly identifiable in the images. A false colour image of the scene showed a dominance of four colours in the land portion. This indicated the presence of four major land-cover types in this part of the image. Thus, including the sea, there are five categories in all. However, as there was no ground truth information available for this image, these land-cover types could not be identified.

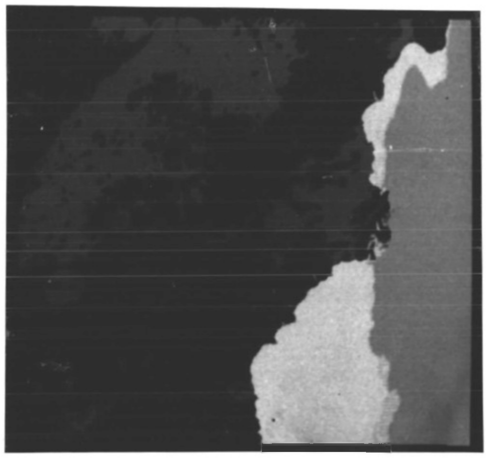


(a)

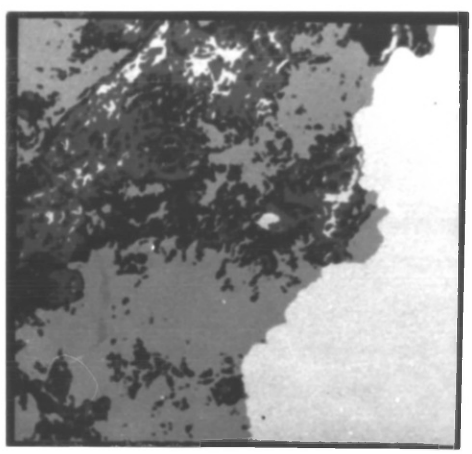
(b)

(c)

Fig. 6.6 Test Thematic Mapper Image
(a) Band 4 (b) Band 5 (c) Band 7



(a) (CPU Time: 10 mins 34.80 secs)



(b) (CPU Time: 6 mins 17.64 secs)

Fig. 6.7 **Five-Category Partitions of TM Image Generated by:**
(a) Algorithm I
(b) Algorithm II

The two algorithms were applied to the image and the number of categories specified was five. For Algorithm I, the number of uniform neighbourhoods found in the image at the starting α value (0.95) was 480. The final number under the allowable maximum constraint was 278; while for Algorithm II, the number of vectors that was clustered was 173. The segmentations are shown in Fig. 6.7(a) and (b). The respective CPU times are also indicated. Clearly, the segmentation result from Algorithm I for this image is poor. The reason for this is that no uniform neighbourhood was found for one of the categories in the land portion of the image, as the uniformity criterion became very stringent under the allowable maximum constraint. The sea was instead split into two categories.

6.4.2 Segmentation of Gray Tone Dominated Images

In these images, the only information available for segmentation is brightness. Thus, the input features for segmentation were the gray levels of the image pixels.

(a) Passport Photograph

The photograph is shown in Fig. 6.8(a). In this image, the number of categories easily identified as having different average brightness levels depends upon the level of detail that is desired, but there are about five brightness levels that are dominant. The two algorithms were applied to the picture, with the number of categories put at four. However, for this kind of image, a segmentation showing outlines of boundaries is more desirable. In this regard, an edge detection operation was performed on the outputs

(a)

(b) Algorithm I
(CPU Time: 50.01 secs)

(c) Algorithm II
(CPU Time: 42.34 secs)

Fig. 6.8 (a) Passport Photograph
(b and c) Four-Category Segmentations of Passport Photograph

(a) Algorithm I
(CPU Time: 51.06 secs)

(b) Algorithm II
(CPU Time: 43.28 secs)

(c) Algorithm I
(CPU Time: 54.36 secs)

(d) Algorithm II
(CPU Time: 45.22 secs)

Fig. 6.9 (a and b) Five-Category Segmentations of Passport Photograph
(c and d) Six-Category Segmentations of Passport Photograph

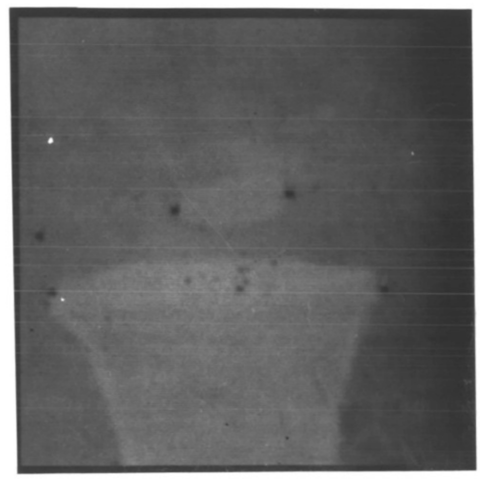
from the algorithms. These results are shown in Fig. 6.8(b) and (c). The results from both algorithms are similar. The indicated CPU times are the "actual" segmentation times; that is, they do not include those for the edge detection operations.

The experiments were repeated putting the number of categories at five and six respectively, so as to increase the level of detail. The corresponding segmentations are shown in Fig. 6.9(a-d). A comparison of the four-category partition and the five-category case for each algorithm (i.e. Fig. 6.8(b) and Fig. 6.9(a); Fig. 6.8(c) and Fig. 6.9(b)) shows that there is an increase in the number of boundaries in the five-category partitions, but with the boundaries produced in the four-category segmentations remaining relatively unchanged. The same trend is shown for the five-category and six-category cases.

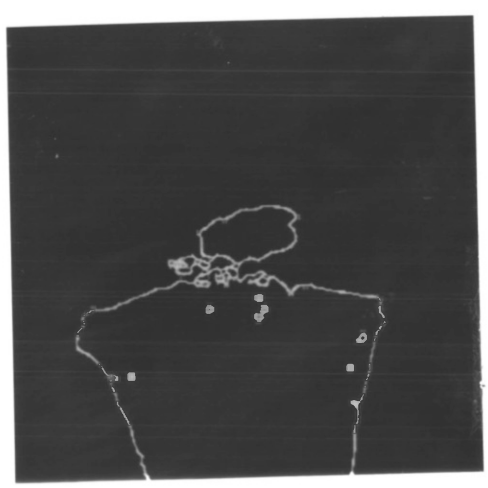
(b) X-ray Image

This is an image of a human wrist. For this image, shown in Fig. 6.10(a), the main objective was to identify the outline between the bones and their fleshy background. The image consists of two bones, which have essentially the same level of brightness. In the background, two regions of different average brightness can be noticed. Thus, in all, there are three categories in the image with different average brightness levels, with the bones being the brightest regions.

In the segmentation experiments, therefore, the number of categories was put at three. However, as the interest was to distinguish the bones from the background, a condition was imposed on the algorithms such that, after clustering, only the two highest average gray level values were used in classifying the pixels. These



(a)



(b) Algorithm I
(CPU Time: 51.47 secs)

(c) Algorithm II
(CPU Time: 46.14 secs)

Fig. 6.10 (a) X-ray Image
(b and c) Segmentations of X-ray Image

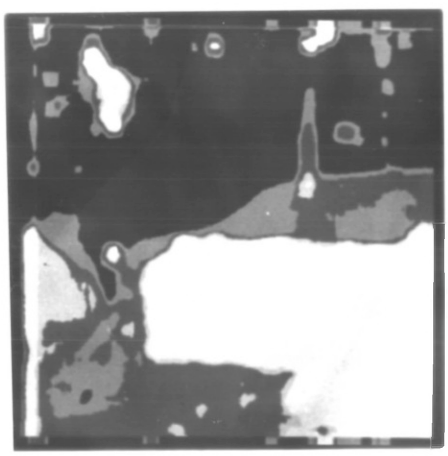
two values therefore correspond to the bones and the brightest part of the background. An edge detection operation was then performed on the output of the algorithms to produce the boundary outlines. The overall segmentation results are shown in Fig. 6.10(b) and (c). The CPU times for the overall segmentations are also indicated.

6.4.3 Segmentation of Outdoor Scene

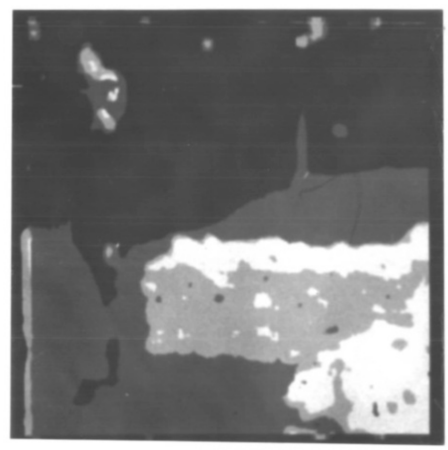
The outdoor scene, shown in Fig. 6.11(a), is a picture of a man walking across a garden. The image is a fairly complex one, consisting of many component regions. However, the number of regions of interest depends upon the level of description that is desired. Most of the regions differ in texture as well as in average brightness. Therefore, the image was segmented on the basis of both texture and brightness. The two textural features described in section 4.3 were employed. Feature window and characterization window sizes of 5 x 5 and 25 x 25 respectively were used. The average gray level in the characterization window was used to represent the level of brightness at each image point.

The image was first partitioned into five categories. These partitions, using both algorithms, are shown in Fig. 6.11(b) and (c). The corresponding boundary outlines are in Fig. 6.11(d) and (e). The image was also partitioned into six and seven categories in order to increase the level of description. The boundary outlines of the resulting segmentations are shown in Fig. 6.12(a-d). The indicated CPU times are the times for the "actual" segmentation operations; that is, for the textural feature computation and application of algorithms. They do not include those for the outlining of boundaries.

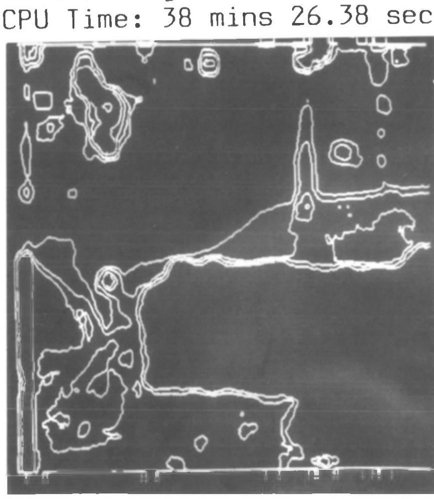
(a)



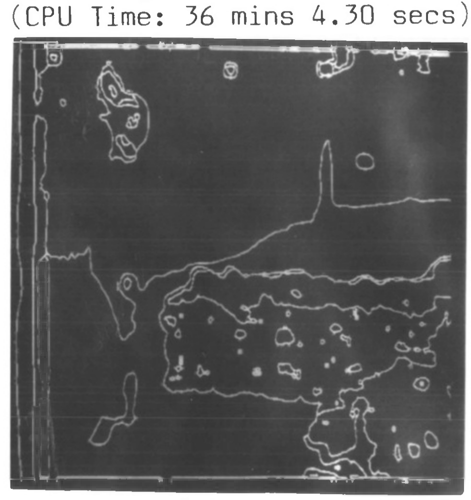
(b) Algorithm I
(CPU Time: 38 mins 26.38 secs)



(c) Algorithm II
(CPU Time: 36 mins 4.30 secs)

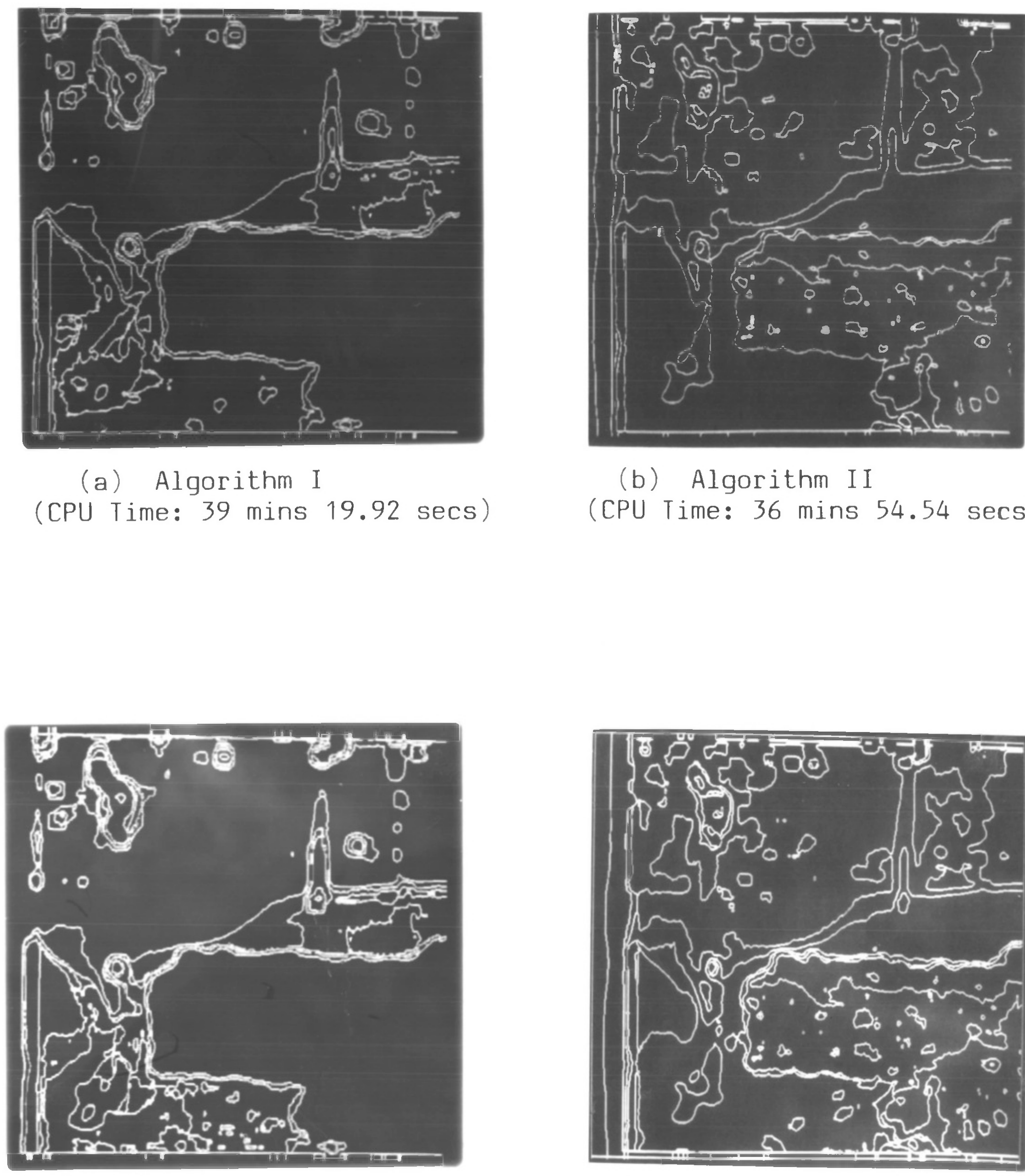


(d)



(e)

Fig. 6.11 (a) Outdoor Scene
(b and c) Five-Category Partitions of Outdoor Scene
(d and e) Boundary Outlines of Five-Category Partitions



(a) Algorithm I
(CPU Time: 39 mins 19.92 secs)

(b) Algorithm II
(CPU Time: 36 mins 54.54 secs)

(c) Algorithm I
(CPU Time: 40 mins 34.30 secs)

(d) Algorithm II
(CPU Time: 38 mins 4.38 secs)

Fig. 6.12 Boundary Outlines of Six-Category and Seven-Category Partitions of Outdoor Scene
(a and b) Six-Category Partitions
(c and d) Seven-Category Partitions

6.4.4 Segmentation of Composite Textured Image

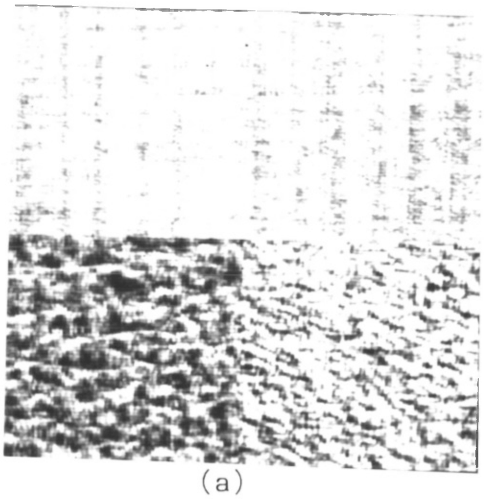
This composite image, shown in Fig. 6.13(a), consists of three regions of different textures. Two of the regions are very similar in terms of coarseness, but fairly different in their levels of contrast. As mentioned in section 4.5, the two textural features developed in this work for segmentation may not be able to discriminate between the two regions satisfactorily. A feature that is strongly related to contrast is needed. The variance in gray level values as a feature was also suggested in the above mentioned section.

In this regard, the variance in gray level values in the characterization window centred on each pixel was used as an additional feature. As in the previous experiments, the feature and characterization window sizes were 5 x 5 and 25 x 25 respectively. The segmentation results for the image are shown in Fig. 6.13(b) and (c).

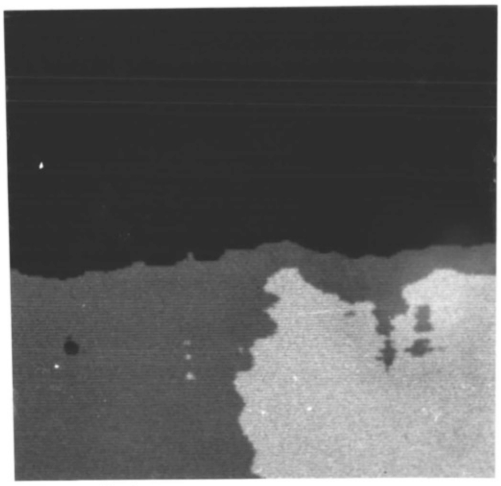
6.5 Conclusion

A segmentation method is presented which combines clustering with the region growing concept of locating uniform areas in an image. Essentially, the technique involves the computation of the mean feature values of uniform neighbourhoods in an image. These mean feature values are agglomeratively clustered to produce the mean feature vectors for the different categories present in the image, and these vectors are then used to classify the pixels. The clustering process introduces the notion of hierarchy.

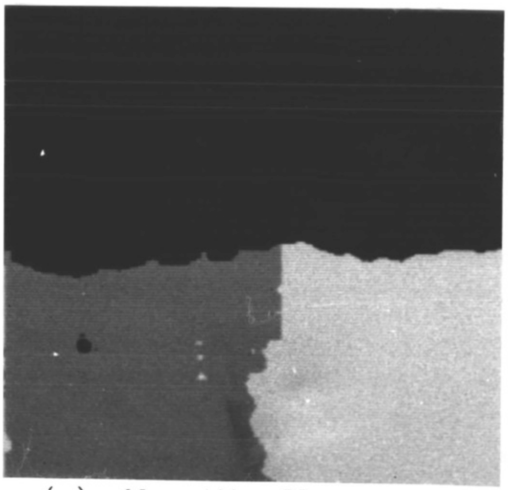
The method has been applied to the partitioning of six different images, including three at different levels of description (i.e. number of categories specified), with considerable success. The



(a)



(b) Algorithm I
(CPU Time: 13 mins 14.74 secs)



(c) Algorithm II
(CPU Time: 10 mins 52.07 secs)

Fig. 6.13 (a) Composite Textured Image
(b and c) Segmentations of Composite Textured Image

choice of the two parameters used in the scheme - neighbourhood size and uniformity criterion - is of paramount importance, particularly in relation to accuracy and cost of computation. Either or both of these parameters can be varied.

In this regard, two algorithms were designed for the implementation of the scheme. In Algorithm I, a fixed neighbourhood size was used for all parts of the image, while the uniformity criterion was varied subject to some constraints. In Algorithm II, the uniformity criterion was fixed, while the size of neighbourhood was varied from one region of the image to another, depending on the degree of uniformity. This variation in neighbourhood size was accomplished using a quad-tree approach.

The results obtained in the segmentation of the different images used in the experiments confirm the general applicability of the approach presented in this work. The CPU process times for the segmentations indicate the feasibility of the method for real-time applications. Algorithm II, in addition to being very fast, also produces more accurate segmentations than Algorithm I. This is therefore the recommended algorithm for the implementation of the technique described here.

The choice of the largest and smallest search neighbourhoods depends upon the degree of uniformity, as well as the anticipated sizes of the categories in the image. The sizes of the categories may in turn be dependent upon the level of description desired. For instance, if the desired level of detail is high, the areas corresponding to some of the categories may be small. In such situations, it is only natural that the size of the smallest search neighbourhood should be small. On the other hand, the size of the largest search neighbourhood should be large if areas corresponding to some categories are very uniform and make up a significant proportion of the image. This would ensure that the number of feature vectors obtained from such areas is small, and consequently minimize the cost of computation.

Moreover, the value of the uniformity criterion may be dependent upon the desired level of splitting of non-uniform neighbourhoods - in other words, upon the size of the smallest search block. In general, a very small neighbourhood is likely to be uniform. Therefore, the smaller the size of the smallest search neighbourhood, the stricter could be the criterion. Alternatively, different values of the uniformity parameter may be used at different levels in the quad-tree splitting of non-uniform neighbourhoods, with the value increasing (i.e. making the uniformity criterion stricter) with increased level of splitting. With careful selection of the parameters; namely, sizes of the largest and smallest search neighbourhoods, and uniformity criterion, it is expected that the scheme can be applied to segment any kind of image at reasonable cost and with satisfactory accuracy.

CHAPTER SEVEN

CONCLUSIONS

The problem of scene interpretation at low cost has been investigated. Attention has been focused on three aspects of scene interpretation: namely, texture characterization, design of image classifier, and image segmentation. Attempts have been made to develop computationally-efficient and generally-applicable methods in order to minimize cost, at the same time ensuring that the methods are satisfactory with respect to accuracy of analysis.

In the area of texture characterization, the two issues of texture classification and textural segmentation were considered. With regard to texture classification, five features were developed. These features, though statistical, were derived from the conceptual relationship of some textural properties to spatial changes in intensity or gray tones. These properties are: coarseness, contrast, busyness, complexity, and strength of texture. Each property was conceptually defined in terms of spatial changes in image gray tones, and the conceptual definition was approximated in computational form to produce the related textural feature. The features are quickly computable, and their computation requires much less memory than other systems.

The features were applied in two sets of experiments that also involved human perceptual measurements. One set of experiments involved the rank ordering of ten natural textures by human subjects using each of the five textural properties. The computer performed the same task using the features that have been developed. The second set of experiments was the measurment of similarity between different textural patterns by humans, and also by the computer, the latter using certain combinations of the features. With respect to ranking, there was a high level of correspondence between the perceptual and computational measurements. For the texture similarity measurements, the most similar pattern was correctly identified by the computer for five or more of the ten textures. This degree of correspondence, while not as good as in the case of ranking, is nevertheless very encouraging, particularly with regard to the result of similar experiments described in [74]. Better classification accuracy was also obtained using the features developed here, as compared with the methods of Haralick et al [32] and Weszka et al [80]. Moreover, in terms of cost, the computation of these features is considerably less expensive when compared with the other two methods.

For textural segmentation, two features were also developed. The application of these features in the segmentation of a number of images produced satisfactory results. A distribution-free classifier based upon the Euclidean distance metric was also designed. In this design, features are constrained to take values between zero and one inclusive. Features are also weighted such that the effectiveness of each feature in classification decision depends upon its relative ability to discriminate between the classes. The classifier has a level of accuracy which is comparable with that of the maximum likelihood classifier. The design is simple, and it is considerably faster than the maximum likelihood classifier in terms of speed. Thus, the classifier which has been designed here has a high level of accuracy, at low computational cost.

In the area of image segmentation, a method was developed which combines the concept of clustering with the region growing concept of locating uniform areas in an image. The technique involves the computation of the mean feature values of uniform neighbourhoods in an image. These mean feature values are agglomeratively clustered to produce the mean feature vectors for the different categories present in the image, and these vectors are then used to classify the image pixels. The clustering process introduces the notion of hierarchy. The method was successfully applied to segment six different images, three of them at different levels of description. Two algorithms were designed for the implementation of the segmentation scheme, varying either one of the two parameters used in the scheme, i.e. uniformity criterion and neighbourhood size. In Algorithm I, the same neighbourhood size is used for all parts of the image, while the uniformity criterion is varied subject to some constraints. In Algorithm II, the uniformity criterion is fixed at a given value, and the size of neighbourhood varied from one part of the image to another, depending upon the degree of uniformity. This variation in neighbourhood size is accomplished using the approach of a quad-tree. The CPU process times for different applications indicate the feasibility of the approach for use in real time, in particular with regard to Algorithm II. This algorithm is very fast, and also has better accuracy; therefore it is the recommended algorithm for the implementation of the segmentation method.

Suggestions for Further Work

As indicated by the better classification accuracy obtained using the features developed in this thesis, it is highly desirable that further efforts are directed towards the development of perception-related textural features. Perhaps, in this regard, more work needs to be done in the psychological field, to provide a better understanding of the visual perception mechanism. Also, the problem of texture characterization with respect to segmentation needs much more attention than it has hitherto been given.

In the area of classifier design, further research is needed to investigate the use of other simple measures of similarity (other than the Euclidean distance metric) in a framework similar to the classifier developed here. This is necessary, as the results obtained using this classifier show that high levels of accuracy can be achieved using simple designs, and at minimal computational cost. In this respect, the use of other kinds of criteria for feature weighting also deserves investigation, particularly those criteria which take into consideration the overlaps or variances in feature values; for example, the Fisher's distance.

With regard to segmentation, the use of high-level information or knowledge for the improvement of results needs investigation. In other words, effort is needed to develop segmentation algorithms that are self-tuning; that is, algorithms that are capable of employing high-level knowledge to modify the segmentation process in one way or another until acceptable or reasonable results are obtained. High-level knowledge may consist of topological descriptors [25], or other kinds of information that can be adequately described. Such information may include: the expected sizes of the categories/ regions, their shapes, the angular position of one desired region with respect to another, or the distance between any two regions.

Furthermore, such information should be made to have some relationship to segmentation parameters; therefore, depending upon the results obtained, these parameters are modified and the relevant stage(s) of the segmentation process repeated. Systems using such algorithms would then be capable of asking the question: "Is this segmentation result reasonable, and if not, why is it not reasonable, and which parameter and/or stage of the process should be modified?" With an adequate knowledge base, greater accuracy would be realized, and more complex problems, such as the analysis of dynamic scenes, would be made easier.

APPENDIX A-1

FREQUENCY OF RANKS AND TEXTURE SIMILARITY ASSIGNMENTS

A) Frequency of Ranks

In Table T-A1.1(a-e), the frequencies of ranks for the ten textures are presented. These are the same as the number of subjects who gave a particular rank to each of the textures. A blank in the table indicates zero frequency.

Ranks	Frequency of Ranks for Each Texture									
(k)	A	B	C	D	E	F	G	H	I	J
1	40				1		47			
2	26			1	24		36		1	
3	18	1	2		57		4	1	5	
4	2	2	14	3	2	5		3	50	7
5	2	5	26	1	1	27		7	11	7
6		24	12	1		22		18	5	5
7		26	14	2	1	22		21	3	
8		21	10	3		8		26	6	13
9		7	8	15		3	1	11	4	39
10		2	2	62		1		1	3	17

(a) Coarseness

Ranks (k)	A	B	C	D	E	F	G	H	I	J
	Frequency of Ranks for Each Texture									
1		1	11	2	6	2	58	5		3
2		4	10	1	41	5	11	9	3	4
3	8	5	21		11	15	1	18	8	1
4	7	5	15		6	12	3	20	15	5
5	3	6	8	4	6	21	2	20	11	7
6	3	10	11	4	8	20	4	7	13	8
7	12	27	3	2	5	7	2	6	10	14
8	10	23	5	3	2	5	2	2	13	23
9	26	6	2	28	1	1	3	1	4	16
10	19	1	2	44	2		2		11	7

(b) Contrast

Ranks (k)	A	B	C	D	E	F	G	H	I	J
	Frequency of Ranks for Each Texture									
1		2	10	39	4	9	3	9	2	10
2		10	9	6	2	15	5	20	2	19
3		16	13	9	3	15	2	15	7	8
4	2	21	11	6	1	11	1	15	11	9
5	1	18	6	2	4	20	2	12	16	7
6	2	9	21	6	7	14	4	4	16	5
7	8	4	10	4	4	4	7	10	25	12
8	12	6	5	2	39		15	3	1	5
9	19		3	5	21		29		2	9
10	44	2		9	3		20		6	4

(c) Busyness

Ranks (k)	A	B	C	D	E	F	G	H	I	J
	Frequency of Ranks for Each Texture									
1		7	23	7	3	14	18	4	5	7
2	1	10	9	6	14	11	8	19	8	2
3	4	10	15	9	7	7	8	12	11	5
4	9	8	7	5	14	15	4	10	9	7
5	6	5	8	7	7	15	8	9	21	2
6	12	6	10	6	6	8	7	15	8	10
7	9	16	7	4	9	5	4	11	13	10
8	19	9	4	6	13	10	9	6	5	7
9	16	11	3	7	11	3	11	2	2	22
10	12	6	2	31	4		11		6	16

(d) Complexity

Ranks (k)	Frequency of Ranks for Each Texture A	B	C	D	E	F	G	H	I	J
1	8		23	1	6		48		2	
2	8		10	3	37	1	16	10	3	
3	33	2	11	5	14	1	9	7	5	1
4	10	6	18		12	8	5	11	10	8
5	11	11	5	6	6	15	1	21	9	3
6	8	6	4	13	5	16		12	21	3
7	5	25	6	8	4	14	3	6	14	3
8	4	20	7	9	4	14	1	10	10	9
9	1	15		21		15	5	9	3	19
10		3	4	22		4		2	11	42

(e) Texture Strength

Table T-A1.1
Frequency of Ranks for Textures
Using Texture Properties

B) Frequency of Similarity Assignments

The frequency of assignment of a given texture (i.e. the number of subjects who considered a given texture as most similar to, or second most similar to, a reference texture) is shown in Table T-A1.2. The frequencies are in two columns for each texture. The first column is for the assignment as a most similar texture to the reference one, while the second column is for the assignment as the second most similar one. For instance, 58 subjects considered texture F to be most similar to texture B, while 25 subjects considered it as the second most similar. Again, a blank indicates zero assignment.

Reference Texture	A	B	C	D	E	F	G	H	I	J
A		2 2		1	50 32	2 3	26 46		6 3	1 2
B			5	2	2	58 25	1	2	27 42	1 11
C	1	4 9		10 56		4	1	66 8	6 11	
D	1 1	2 6	12 41			1 1	1	54 23	3 4	15 11
E	14 64	1 1				2 7	71 14		1	1
F		46 36	1 4						37 39	4 10
G	3 73	2 1	1	2	81 3	5		1	1 2	1
H		3 9	44 30	32 37		3 5			2	6 5
I	1 1	38 36	2 4	1 4	2 1	43 36		1		6
J		26 28	2	19 8		36 26	2	4 20	1 4	

Table T-A1.2
Frequency of Assignments of Textures as Most Similar and Second Most Similar to Reference Texture

APPENDIX A-2

THE MINIMUM ERROR-RATE (MAXIMUM LIKELIHOOD) CLASSIFIER

The classifier design is such as to obtain minimum rate of misclassification.

Let $\mathbf{X}$ be a d-dimensional column vector representing the features of a sample. The d-dimensional conditional Gaussian density function for $\mathbf{X}$, given class i, with mean feature vector $\mathbf{M}_i$ and covariance matrix Σ_i, is given by

$$g_i(\mathbf{X}) = (2\pi)^{-d/2}|\Sigma_i|^{-1/2} \exp[-1/2(\mathbf{X} - \mathbf{M}_i)^t\Sigma_i^{-1}(\mathbf{X} - \mathbf{M}_i)] \qquad \textbf{(A2.1)}$$

where Σ_i^{-1} is the inverse of the matrix Σ_i. It is assumed that the matrix is non-singular.

$|\Sigma_i|$ is the determinant of Σ_i, and the superscript t denotes the transpose of a matrix.

It is shown in [18] that minimum error-rate classification can be achieved by the use of the discriminant function

$$G_i(\mathbf{X}) = \ell n\ g_i(\mathbf{X}) + \ell n\ p_i \qquad \textbf{(A2.2)}$$

where p_i is the a priori probability that $\mathbf{X}$ belongs to class i.

Assuming that there are c number of classes, to one of which $\mathbf{X}$ belongs, then the decision rule is: decide class i if

$$G_i(\mathbf{X}) = \max_{j = 1,2,\ldots,c} \{G_j(\mathbf{X})\} \qquad \textbf{(A2.3)}$$

Substituting (A2.1) into (A2.2), we have

$$G_i(\mathbf{X}) = -1/2\ (\mathbf{X} - \mathbf{M}_i)^t\ \Sigma_i^{-1}\ (\mathbf{X} - \mathbf{M}_i) - d/2\ \ell n\ 2\pi - 1/2\ \ell n\ |\Sigma_i| + \ell n\ p_i \qquad \textbf{(A2.4)}$$

If we define a new term $\hat{G}_i(\mathbf{X})$ given by

$$\hat{G}_i(\mathbf{X}) = - G_i(\mathbf{X})$$

$$= 1/2 \, (\mathbf{X} - \mathbf{M}_i)^t \, \Sigma_i^{-1} \, (\mathbf{X} - \mathbf{M}_i) + d/2 \, \ell n \, 2\pi + 1/2 \, \ell n |\Sigma_i| - \ell n \, p_i \qquad \textbf{(A2.5)}$$

the decision rule then becomes: decide class i if

$$\hat{G}_i(\mathbf{X}) = \min \{\hat{G}_j(\mathbf{X})\}, \quad j = 1,2,\ldots,c$$

The term $d/2 \, \ell n \, 2\pi$ is a constant and is common to all the classes, and hence can be dropped from equation (A2.5). We then have

$$\hat{G}_i(\mathbf{X}) = 1/2 \, (\mathbf{X} - \mathbf{M}_i)^t \, \Sigma_i^{-1} \, (\mathbf{X} - \mathbf{M}_i) + 1/2 \, \ell n |\Sigma_i| - \ell n \, p_i \qquad \textbf{(A2.6)}$$

The last two terms on the right hand side of equation (A2.6) do not involve the vector **X**. They are simply constants that represent a certain bias towards class i, and in practice, eliminating them from the equation hardly affects the result of classification.

Therefore, equation (A2.6) can be written as

$$\tilde{G}_i(\mathbf{X}) = 1/2 \, (\mathbf{X} - \mathbf{M}_i)^t \, \Sigma_i^{-1} \, (\mathbf{X} - \mathbf{M}_i) \qquad \textbf{(A2.7)}$$

$\tilde{G}_i(\mathbf{X})$ is actually the squared Mahalanobis distance from the vector **X** to $\mathbf{M}_i$.

We would then use the decision rule and assign $\mathbf{X}$ to class i if

$$\tilde{G}_i(\mathbf{X}) = \min \{\tilde{G}_j(\mathbf{X})\}, \quad j = 1,2,\ldots,c$$

In order to use the decision algorithm, a training set of data is required to obtain the mean vector and covariance matrix for each class.

If the number of the representative samples of category i is N_i, then the mean feature vector and feature covariance matrix for the ith category are given by

$$\mathbf{M}_i = \frac{1}{N_i} \sum_{n=1}^{N_i} \mathbf{X}_n \tag{A2.8}$$

$$\Sigma_i = \frac{1}{N_i} \sum_{n=1}^{N_i} (\mathbf{X}_n - \mathbf{M}_i)(\mathbf{X}_n - \mathbf{M}_i)^t \tag{A2.9}$$

where $\mathbf{X}_n$ is the feature vector of the nth sample in the ith category.

It is desirable that the number of training samples, N_i, (for each category) is large, for the following reasons:-

(i) to ensure non-singularity of the covariance matrix, as the matrix will be singular if $N_i < d$.

(ii) for a training set to be representative of a category, the training set must include a variety of the samples in the category.

APPENDIX A-3

TEXTURE CHARACTERIZATION TECHNIQUES: SPATIAL GRAY LEVEL DEPENDENCE METHOD AND GRAY LEVEL DIFFERENCE METHOD

A. Spatial Gray Level Dependence Method (SGLDM)

This technique, suggested by Haralick et al [32], assumes that the texture information in an image is contained in the overall or "average" spatial relationship which the gray tones in the image have to one another. Mathematically, it is assumed that the texture information is adequately specified by a set of gray-tone spatial-dependence matrices.

A matrix is computed for an image in which an entry, $(p(i,j)/d,\theta)$, is the probability of finding two gray tones i and j in the image separated by distance d and in angular direction θ. Thus, the entry $(p(i,j)/d,\theta)$ is a second-order joint probability density of gray tones i and j, given that the intersample spacing is d, and the angular direction is θ. From henceforth, it will simply be written as $p(i,j)$ for a given d and specified θ.

If a texture is coarse, and d is small compared to the sizes of the texture primitives, the pairs of points at separation d would tend to have similar gray levels. This results in the concentration of high-value entries in the matrix along its main diagonal; while the values in the matrix should spread out more uniformly in the case of fine texture for the same value of d. Some features for texture can be derived by computing some measures of the scatter of the entries around the main diagonal. Four of these features are considered to be most useful [80], and were used in the experiments. They are:-

(i) Angular Second Moment (ASM)

$$ASM = \sum_{i=0}^{N_G-1} \sum_{j=0}^{N_G-1} [p(i,j)]^2 \qquad \textbf{(A3.1)}$$

where N_G is the number of gray levels in the picture from which the matrix was extracted. The ASM is a measure of homogeneity; it has small value when the matrix elements are evenly spread out and high value when the elements cluster around the main diagonal.

(ii) Contrast (CON)

This is given by

$$CON = \sum_{i=0}^{N_G-1} \sum_{j=0}^{N_G-1} (i-j)^2 \, p(i,j) \qquad \textbf{(A3.2)}$$

This feature gives the moment of inertia of the matrix around its main diagonal; i.e. it is a measure of spread of matrix values.

(iii) Entropy (ENT)

$$ENT = - \sum_{i=0}^{N_G-1} \sum_{j=0}^{N_G-1} p(i,j) \log p(i,j) \qquad \textbf{(A3.3)}$$

This measure is largest for equal p(i,j) and small when they are very unequal. The matrix values tend to be equal and evenly spread out when there are many gray levels in the image and the image has some measure of complexity.

(iv) Correlation (COR)

$$\mathrm{COR} = \sum_{i=0}^{N_G-1} \sum_{j=0}^{N_G-1} \left[\{ijp(i,j)\} - \mu_x \mu_y \right] \Big/ (\sigma_x \sigma_y) \qquad \textbf{(A3.4)}$$

where μ_x and σ_x are the mean and standard deviation of the row sums of the matrix, and μ_y and σ_y are the analogous statistics of the column sums. They are given by:

$$\mu_x = \sum_{i=0}^{N_G-1} i \sum_{j=0}^{N_G-1} p(i,j)$$

$$\mu_y = \sum_{j=0}^{N_G-1} j \sum_{i=0}^{N_G-1} p(i,j)$$

$$\sigma_x^2 = \sum_{i=0}^{N_G-1} (i-\mu_x)^2 \sum_{j=0}^{N_G-1} p(i,j)$$

$$\sigma_y^2 = \sum_{j=0}^{N_G-1} (j-\mu_y)^2 \sum_{i=0}^{N_G-1} p(i,j)$$

The COR is a measure of the degree to which the rows (or columns) of the matrix resemble each other. It has a high value when the entries in the matrix are uniformly distributed, and a low value otherwise.

The features are all functions of distance and angle. For a specified distance d, matrices are usually computed for four θ values: 0°, 45°, 90° and 135°, and features are derived from each matrix. The value of each feature that is actually used in classification is the average of the features from the four matrices. This ensures that classification results are invariant to the angular orientation of an image. The study in [80] also showed that better results are obtained using small values of d; say, d = 1 or 2. Hence d = 1 was used in the experiments.

B. Gray Level Difference Method (GLDM)

The gray level difference method, suggested by Weszka et al [80], considers the absolute differences between pairs of gray levels at a given distance from one another and in a specified angular direction.

For any displacement $\delta = (\Delta_x, \Delta_y)$, let

$$f_\delta(x,y) = \left| f(x,y) - f(x + \Delta_x, y + \Delta_y) \right|$$

and p_δ be the probability density of $f_\delta(x,y)$, where $f(x,y)$ is the gray level of the pixel at the point (x,y). If the number of gray levels in the image is N_G, p_δ has the form of an N_G-dimensional column vector whose ith component is the probability that $f_\delta(x,y)$ will have value i.

For a coarse texture with δ small compared with the texture element size, the pairs of points at separation δ should usually have similar gray levels, so that $f_\delta(x,y)$ would be small, and the values of p_δ would be concentrated near i=0. Conversely, values of p_δ should be concentrated away from i=0 for fine textures. Thus, the measure of the spread of values in p_δ away from the origin is a good way of analysing texture. Five features can be extracted from the matrix, of which the following four were used in the experiments. Three of the features are abbreviated below in small letters, in order to distinguish them from features of the same name in the SGLDM. For a given angular direction θ, the features are given by:

(i) Contrast (con)

$$\text{con} = \sum_{i=0}^{N_G-1} i^2 p_\delta(i) \tag{A3.5}$$

This feature gives the moment of inertia about the origin.

(ii) Angular Second Moment (asm)

$$\text{asm} = \sum_{i=0}^{N_G-1} [p_\delta(i)]^2 \tag{A3.6}$$

It is a measure of homogeneity, and generally takes low values for coarse textures, while the values are high for fine textures.

(iii) Entropy (ent)

$$\text{ent} = - \sum_{i=0}^{N_G-1} p_\delta(i) \log p_\delta(i) \tag{A3.7}$$

This is largest for equal $p_\delta(i)$, and small when they are very unequal. The entries in p_δ tend to be equal when there are many gray levels; hence ent tend to reflect the level of complexity.

(iv) Mean (MN)

$$\text{MN} = \frac{1}{N_G} \sum_{i=0}^{N_G-1} i\, p_\delta(i) \tag{A3.8}$$

The value of MN is small when $p_\delta(i)$ are concentrated near the origin and large when they are far from the origin.

Again, the features are all functions of distance and angle, and, as in the SGLDM, the $p_\delta(i)$ matrix is computed for four θ values: 0°, 45°, 90° and 135°. The averages of the features over the four angular directions are used for classification.

APPENDIX A-4

PROOF OF VARIANCE UPDATING FORMULA

Consider two subpopulations with means m_1 and m_2 and variances s_1^2 and s_2^2, and suppose that the number of elements in the subpopulations are N_1 and N_2 respectively.

Let us denote the ith element in subpopulation 1 as a_i and that in subpopulation 2 as b_i.

Then clearly

$$\sum_{i=1}^{N_1} a_i^2 = N_1(s_1^2 + m_1^2) \tag{A4.1}$$

and

$$\sum_{i=1}^{N_2} b_i^2 = N_2(s_2^2 + m_2^2) \tag{A4.2}$$

If the two subpopulations are merged into one population, whose kth element is denoted as c_k, with mean m_3, variance s_3^2 and number of elements N_3, then

$$N_3 = N_1 + N_2$$

and

$$\sum_{k=1}^{N_3} c_k^2 = N_3(s_3^2 + m_3^2) \tag{A4.3}$$

Therefore

$$s_3^2 = \frac{1}{N_3} \sum_{k=1}^{N_3} c_k^2 - m_3^2 \qquad \textbf{(A4.4)}$$

But

$$\sum_{k=1}^{N_3} c_k^2 = \sum_{i=1}^{N_1} a_i^2 + \sum_{i=1}^{N_2} b_i^2 , \quad \text{and} \quad N_3 = N_1 + N_2$$

Hence

$$s_3^2 = \frac{1}{N_1 + N_2} \sum_{i=1}^{N_1} a_i^2 + \sum_{i=1}^{N_2} b_i^2 - m_3^2 \qquad \textbf{(A4.5)}$$

Putting equations (A4.1) and (A4.2) into (A4.5), we have

$$s_3^2 = \frac{1}{N_1 + N_2} [N_1(s_1^2 + m_1^2) + N_2(s_2^2 + m_2^2)] - m_3^2$$

APPENDIX A-5

LISTING OF COMPUTER PROGRAMS DEVELOPED FOR THE SIMULATION OF ALGORITHMS

The simulations for this research were done on the VAX11/780 computer belonging to the Imperial College Centre for Remote Sensing Image Processing Laboratory. Several programs were developed to enhance the investigations, and the relevant ones are included in this appendix. Apart from the system routines which are used for reading and writing out images, the programs have been written in FORTRAN 77.

All the programs were developed by the investigator (M. Amadasun) during the period of the research. Great care has been taken in preparing the programs in their present form, and there should not be any typographical error. However, if there is any such error, the original programs are available on a magnetic tape deposited with the Digital Communications Section of the Imperial College Electrical Engineering Department.

The programs are self-explanatory, and the comment statements inserted at the relevant places should make them meaningful. The programs have been collated in the order in which they appear in the thesis.

M. Amadasun

January 1988

```
      PROGRAM ANET
C -----------------------------------------------------
C Program to compute the developed texures features;
C fcos,fcon,fbus,fcom and fstr for an image of
C size 64 X 64.
C -----------------------------------------------------
      PARAMETER(NYY=64,NXX=64,NG=255)
      INTEGER*2 IMAGE(NYY,NXX),IC
      INTEGER*2 I,J,K,L,MQ,NQ,K2
      REAL S(0:NG),P(0:NG),IBUS,IVI,TSI,MC,Z3
      REAL FCOS,FCON,FBUS,FCOM,FSTR,DAT,QT,QC
      REAL PP,AB,QP,AC,AD,Z1,Z2,TI,TJ,PMU,PBUS
      INTEGER*2 STATUSFLAG,LINENUMBER,NAT
      LOGICAL*1 FILE(30)
      COMMON IMAGE,S,P,MQ,NQ
C
C Supply the distance for specifying neighbourhood size.
      WRITE(6,*)'INPUT THE DISTANCE FOR SPECIFYING'
      WRITE(6,*)'NEIGHBOURHOOD SIZE,IC,.GE.1 AND.LE.4'
      READ(5,*)IC
C Specify the name of the file into which the computed
C feature values are to be written.
      WRITE(6,*)'DEFINE FILE NAME'
      READ(5,1)NCH,(FILE(I),I=1,NCH)
 1    FORMAT(Q,30A1)
      OPEN(UNIT=70,NAME=FILE,TYPE='NEW')
C Read in the texture image
      CALL VICINIT('ANET')
      CALL OPENV(STATUSFLAG,2,0,0,0,0)
      DO LINENUMBER=1,NYY
      CALL READ(STATUSFLAG,2,0,1,0,NXX,IMAGE(1,LINENUMBER),0)
      ENDDO
C
      DAT=FLOAT((NYY-2*IC)*(NXX-2*IC))
C Call subroutine to compute Neighbourhood Gray Tone
C Difference Matrix for image.
      CALL COMP(IC)
C Compute the texture measures.
C Compute fcos.
      PMU=0.0
      PBUS=0.0
      DO K=MQ,NQ
       PBUS=PBUS+S(K)
       PMU=PMU+P(K)*S(K)
      ENDDO
      FCOS=1.0/(0.0000001+PMU)
C Compute fcon
      IVI=0.0
      Z3=0.0
      DO I=MQ,NQ
       IF(P(I).NE.0.0)THEN
        Z3=Z3+1.0
         DO J=MQ,NQ
          IF(P(J).NE.0.0)THEN
           Z1=FLOAT(I)
```

```
         Z2=FLOAT(J)
         IVI=IVI+(P(I)*P(J))*(Z1-Z2)**2
        ENDIF
       ENDDO
      ENDIF
     ENDDO
     Z3=Z3*(Z3-1.0)
     IF(Z3.EQ.0.0)Z3=1.0
     FCON=(IVI/Z3)*(PBUS/DAT)
C Compute fbus
     IBUS=0.0
     DO I=MQ,NQ
      IF(P(I).NE.0.0)THEN
       DO J=MQ,NQ
        IF(P(J).NE.0.0)THEN
         Z1=FLOAT(I)
         Z2=FLOAT(J)
         AC=ABS((P(I)*Z1)-(P(J)*Z2))
         IBUS=IBUS+AC
        ENDIF
       ENDDO
      ENDIF
     ENDDO
     IF(IBUS.EQ.0.0)IBUS=1.0
     FBUS=(PMU/IBUS)
C Compute fstr.
     TSI=0.0
      DO I=MQ,NQ
       IF(P(I).NE.0.0)THEN
        DO J=MQ,NQ
         IF(P(J).NE.0.0)THEN
          Z1=FLOAT(I)
          Z2=FLOAT(J)
          AB=(Z1-Z2)**2
          TSI=TSI+AB*(P(I)+P(J))
         ENDIF
        ENDDO
       ENDIF
      ENDDO
      FSTR=TSI/(0.0000001+PBUS)
C Compute fcom.
      MC=0.0
      DO I=MQ,NQ
       IF(P(I).NE.0.0)THEN
        DO J=MQ,NQ
         IF(P(J).NE.0.0)THEN
          Z1=FLOAT(I)
          Z2=FLOAT(J)
          AB=ABS(Z1-Z2)
          TI=P(I)*DAT
          TJ=P(J)*DAT
          MC=MC+((AB*(P(I)*S(I)+P(J)*S(J)))/(TI+TJ))
         ENDIF
        ENDDO
       ENDIF
```

```
         ENDDO
         FCOM=MC/DAT
         WRITE(70,70)FCOS,FCON,FBUS,FCOM,FSTR
 70        FORMAT(5F15.6)
         END
C ***********************************************************
         SUBROUTINE COMP(IC)
C Subroutine to compute Neighbourhood Gray Tone
C Difference Matrix (NGTDM)
         PARAMETER(NYY=64,NXX=64,NG=255)
         INTEGER*2 IMAGE(NYY,NXX),IC,NQ
         INTEGER*2 I,J,K,L,MQ,M,N,K2
         REAL S(0:NG),P(0:NG)
         REAL Q1,Q2,DAT,QT,QC,SUMG
         REAL PP
         COMMON IMAGE,S,P,MQ,NQ
C
         DAT=FLOAT((NYY-2*IC)*(NXX-2*IC))
         QC=FLOAT((2*IC)+1)
         QT=(QC**2)-1.0
         DO K=0,NG
          S(K)=0.0
         ENDDO
         NQ=0
         MQ=255
         DO J=IC+1,NXX-IC
          DO I=IC+1,NYY-IC
           Q1=FLOAT(IMAGE(I,J))
           K2=IMAGE(I,J)
           IF(NQ.LT.K2)NQ=K2
           IF(MQ.GT.K2)MQ=K2
           SUMG=0.0
           DO L=-IC,IC
            DO K=-IC,IC
             Q2=FLOAT(IMAGE(I+K,J+L))
             SUMG=SUMG+Q2
            ENDDO
           ENDDO
           SUMG=(SUMG-Q1)/QT
           P(K2)=P(K2)+1.0
           S(K2)=S(K2)+ABS(SUMG-Q1)
          ENDDO
         ENDDO
         DO K=MQ,NQ
          P(K)=P(K)/DAT
         ENDDO
         RETURN
         END
```

```
              PROGRAM ACLASS
C -----------------------------------------------------------
C Program to implement the weighted-feature minimum distance
C classifier.
C
C INPUT  : The mean feature  vectors of each class and
C        : the feature vector(s) of the unknown sample(s).
C
C OUTPUT : The class to which the unknown sample(s) is/are
C        :  assigned.
C
C        :  In the program NZ/NT stands for the number of
C        :  classes, NX/NF for the number of features to be
C        :  used in classification, and NSAMP/ISAMP for the
C        :  total number of unknown samples to be classified.
C
C -----------------------------------------------------------
        PARAMETER(NZ=50,NX=10,NSAMP=150)
        INTEGER*2 NT,NF,I,J,K,L
        INTEGER*2 ICHAN,NS,ISAMP,NAT
        REAL AB,DB,KM(NSAMP,NX),TH,ZA,ZC
        REAL REF(NZ,NX),AZ(NX),QT(NX),ZB
        REAL INCOV(NZ,NX,NX),VAR(NX)
        COMMON REF,KM,QT
C
        CALL VICINIT('ACLASS2')
C
        WRITE(6,*)'INPUT THE NO. OF LIKELY CLASSES,'
        WRITE(6,*)'less or equal to 50'
        READ(5,*)NT
C
        WRITE(6,*)'INPUT THE NO. OF FEATURES TO BE USED IN'
        WRITE(6,*)'CLASSIFICATION,less or equal to 10'
        READ(5,*)NF
C
        WRITE(6,*)'INPUT THE TOTAL NO. OF UNKNOWN SAMPLE(S)'
        WRITE(6,*)'TO BE CLASSIFIED ,MAXIMUM NO. OF UNKNOWN'
        WRITE(6,*)'SAMPLES THAT CAN BE CLASSIFIED IN ONE'
        WRITE(6,*)'RUN IS 150'
        READ(5,*)ISAMP
C
        WRITE(6,*)'INPUT THE CHANNEL NUMBER FOR WRITING FILE'
        READ(5,*)ICHAN
C Read in the mean feature vectors for the classes
        DO I=1,NT
         READ(9,70)(REF(I,J),J=1,NF)
 70      FORMAT(<NF>F15.6)
        ENDDO
C Read in the feature vector(s) of the unknown sample(s).
        DO I=1,ISAMP
         READ(9,70)(KM(I,J),J=1,NF)
        ENDDO
C -----------------------------------------------------------
C Determine weighting factors for features.
        DO L=1,NF
```

```
          QT(L)=0.0
          DO I=1,NT-1
           DO J=I+1,NT
            ZA=ABS(REF(I,L)-REF(J,L))
            ZC=(REF(I,L)+REF(J,L))/2.0
            QT(L)=QT(L)+(ZA/ZC)
           ENDDO
          ENDDO
         ENDDO
C
         ZC=QT(1)
         DO L=1,NF
          IF(ZC.LT.QT(L))ZC=QT(L)
         ENDDO
         DO L=1,NF
          QT(L)=ZC/QT(L)
         ENDDO
         WRITE(ICHAN,*)'THESE ARE THE WEIGHTING FACTORS FOR'
         WRITE(6,*)'FEATURES'
         WRITE(ICHAN,85)(QT(L),L=1,NF)
C -----------------------------------------------------------
C Determine normalizing factors for features.
         DO L=1,NF
          AZ(L)=0.0
          DO K=1,NT
           AZ(L)=AZ(L)+REF(K,L)
          ENDDO
         ENDDO
C -----------------------------------------------------------
C Normalize mean feature values for classes
         DO L=1,NF
          DO K=1,NT
           REF(K,L)=REF(K,L)/AZ(L)
          ENDDO
         ENDDO
C -----------------------------------------------------------
C Normalize feature values of unknown sample(s)
         DO L=1,NF
          DO K=1,ISAMP
           KM(K,L)=KM(K,L)/AZ(L)
          ENDDO
         ENDDO
C -----------------------------------------------------------
 85      FORMAT(<NF>F12.5)
C
         CALL ASSIGN(NT,NF,ISAMP,ICHAN)
C
         END
C ***********************************************************
          SUBROUTINE ASSIGN(NT,NF,ISAMP,ICHAN)
C Subroutine to classify unknown sample(s)
         PARAMETER(NZ=50,NX=10,NSAMP=150)
         INTEGER*2 NT,NF,N2,ISAMP,ICHAN,KK
         INTEGER*2 LEF(NZ),NAT,MM,J
         REAL REF(NZ,NX),KM(NSAMP,NX)
```

```
      REAL D(NX),AB,PP,QQ,G(NZ),QT(NX)
      REAL H(NZ,NX),SUM,PAX,P,Q
      COMMON REF,KM,QT
C
      MM=NF
      DO J=1,ISAMP
       DO L=1,MM
        D(L)=KM(J,L)
       ENDDO
       DO K=1,NT
        G(K)=0.0
        DO L=1,MM
         PAX=(D(L)-REF(K,L))**2
         G(K)=G(K)+(QT(L)*PAX)
        ENDDO
       ENDDO
       PAX=G(1)
       KK=1
       DO K=1,NT
        IF(PAX.GT.G(K))THEN
         PAX=G(K)
         KK=K
        ENDIF
       ENDDO
       IF(ISAMP.GT.1)THEN
        WRITE(ICHAN,*)'THE UNKNOWN SAMPLE',J
        WRITE(ICHAN,*)'BELONGS TO CLASS',KK
        WRITE(ICHAN,*)
        WRITE(ICHAN,*)
       ELSE
        WRITE(ICHAN,*)'THE UNKNOWN SAMPLE BELONGS'
        WRITE(ICHAN,*)'TO CLASS',KK
       ENDIF
      ENDDO
      RETURN
      END
```

```
              PROGRAM SPEG1
C ----------------------------------------------------------
C  Program for segmentation ALGORITHM I for the
C  segmentation of any 3-band multispectral image.
C
C     INPUT    :    ANY 3-BAND MULTISPECTRAL IMAGE
C
C     OUTPUT   :    SEGMENTED VERSION OF THE INPUT IMAGE
C
C -----------------------------------------------------------
      PARAMETER(NYY=512,NXX=512,MM=3,NZ=150)
      INTEGER*2 IMAGE(NYY,NXX),KC(NYY,NXX)
      INTEGER*2 ICHAN,NY,NX,IC,ID,IB,NT,ITEST
      REAL KM(NYY,NXX,MM),KD(NYY,NXX),TH
      REAL REF(NZ,MM),AY(MM),QT(MM),THR
      INTEGER*4 STATUSFLAG,LINENUMBER
      COMMON IMAGE,KC,KD,KM,REF,AY
C -----------------------------------------------------------
C Read in the images in the three bands.
      CALL VICINIT('SPEG1')
      CALL OPENV(STATUSFLAG,2,0,0,0,0)
      CALL OPENV(STATISFLAG,3,0,0,0,0)
      CALL OPENV(STATUSFLAG,4,0,0,0,0)
      DO LINENUMBER=1,NYY
       CALL READ(STATUSFLAG,2,0,1,0,NXX,KC(1,LINENUMBER),0)
       CALL READ(STATUSFLAG,3,0,1,0,NXX,KD(1,LINENUMBER),0)
       CALL READ(STATUSFLAG,4,0,1,0,NXX,IMAGE(1,LINENUMBER),0)
      ENDDO
C ----------------------------------------------------------------
C Supply the required segmentation parameters.
      WRITE(6,*)'DO YOU WANT TO USE THE GRAY LEVELS OF'
      WRITE(6,*)'PIXELS DIRECTLY OR THE AVERAGE GRAY LEVEL'
      WRITE(6,*)'IN A SMALL WINDOW CENTERED ON PIXEL FOR'
      WRITE(6,*)'SEGMENTATION?. IF PIXEL GRAY LEVELS INPUT'
      WRITE(6,*)'0, OTHERWISE INPUT 1'
      READ(5,*)ITEST
      IF(ITEST.EQ.1)THEN
       WRITE(6,*)'INPUT THE DISTANCE FOR SPECIFYING THIS'
       WRITE(6,*)'WINDOW SIZE; IB'
       WRITE(6,*)'NOTE W=(2*IB+1)*(2*IB+1)'
       READ(5,*)IB
      ELSE
       IB=1
      ENDIF
C
      WRITE(6,*)'INPUT THE NUMBER OF CATEGORIES; NT'
      READ(5,*)NT
C
      WRITE(6,*)'INPUT THE DISTANCE FOR SPECIFYING DIMENSION'
      WRITE(6,*)'OF SEARCH BLOCKS; NX. NOTE: DIMENSION=NX*NX.'
      READ(5,*)NX
C
      WRITE(6,*)'INPUT THE INITIAL VALUE OF UNIFORMITY'
      WRITE(6,*)'CRITERION; THR, AND THE INCREMENTAL/'
      WRITE(6,*)'DECREMENTAL FACTOR; TH.BOTH ARE'
```

```
      WRITE(6,*)'REAL NUMBERS.'
      READ(5,*)THR,TH
C -------------------------------------------------------------
      WRITE(6,*)'INPUT THE CHANNEL NUMBER FOR WRITING;ICHAN,'
      WRITE(6,*)'AN INTEGER'
      READ(5,*)ICHAN
C
      CALL FEATURE(IB,ITEST)
      CALL CLAVECT(NX,NT,QT,ICHAN,THR,TH)
      CALL ASSIGN(NT,QT)
C
      CALL OPENV(STATUSFLAG,1,1,0,0,0)
      CALL ADJUST(1,NYY,NXX)
      DO I=1,NYY
       CALL WRITE(STATUSFLAG,1,0,1,0,NXX,IMAGE(1,I),0)
      ENDDO
      CALL RELAB2(1,NYY,NXX)
      END
C ************************************************************
              SUBROUTINE FEATURE(IB,ITEST)
C Subroutine to compute features.
      PARAMETER(NYY=512,NXX=512,MM=3,NG=255,LL=10,NZ=150)
      INTEGER*2 IMAGE(NYY,NXX),KC(NYY,NXX),KD(NYY,NXX)
      INTEGER*2 IP1,IP2,IQ1,IQ2,ITEST,IB,ID
      INTEGER*2 M,N,I,J,I1,J1
      REAL KM(NYY,NXX,MM),REF(NZ,MM),QT(MM)
      REAL DNAT,DM,DB,DG,AY(MM)
      COMMON IMAGE,KC,KD,KM,REF,AY
C -------------------------------------------------------------
C If desired, replace the gray levels of each pixel by the
C average in a window centered on it.
      IF(ITEST.EQ.1)THEN
       DO J1=1,NXX
        J=J1
        DO I1=1,NYY
         I=I1
         KM(I1,J1,1)=0.0
         KM(I1,J1,2)=0.0
         KM(I1,J1,3)=0.0
         IP1=-IB
         IP2=IB
         IQ1=-IB
         IQ2=IB
         IF(I.LE.IB)THEN
          IP1=IB+1
          IP2=IP1+IB+I
          I=0
         ENDIF
         IF(I.GT.(NYY-IB))THEN
          IP2=NYY-IB
          IP1=IP2-(IB+(NYY-I))
          I=0
         ENDIF
         IF(J.LE.IB)THEN
          IQ1=IB+1
```

```
      IQ2=IQ1+IB+J
      J=0
     ENDIF
     IF(J.GT.(NXX-IB))THEN
      IQ2=NXX-IB
      IQ1=IQ2-(IB+(NXX-J))
      J=0
     ENDIF
     DNAT=0.0
     DM=0.0
     DB=0.0
     DG=0.0
     DO L=IQ1,IQ2
      DO K=IP1,IP2
       DNAT=DNAT+1.0
       DM=DM+FLOAT(KC(I+K,J+L))
       DB=DB+FLOAT(KD(I+K,J+L))
       DG=DG+FLOAT(IMAGE(I+K,J+L))
      ENDDO
     ENDDO
     KM(I1,J1,1)=DM/DNAT
     KM(I1,J1,2)=DB/DNAT
     KM(I1,J1,3)=DG/DNAT
    ENDDO
   ENDDO
C -------------------------------------------------------
      ELSE
       DO J=1,NXX
        DO I=1,NYY
         KM(I,J,1)=FLOAT(KC(I,J))
         KM(I,J,2)=FLOAT(KD(I,J))
         KM(I,J,3)=FLOAT(IMAGE(I,J))
        ENDDO
       ENDDO
      ENDIF
      RETURN
      END
C *******************************************************
         SUBROUTINE ASSIGN(NT,QT)
C Subroutine to classify pixels.
      PARAMETER(NYY=512,NXX=512,MM=3,NZ=150)
      INTEGER*2 IMAGE(NYY,NXX),IMA(NZ),NT,KK
      INTEGER*2 PEF(NZ),GL,GINC,IFEAT,N1,N2
      INTEGER*2 KC(NYY,NXX),KD(NYY,NXX)
      REAL KM(NYY,NXX,MM)
      REAL REF(NZ,MM),AY(MM),KR1,KR2,PP,QQ
      REAL RK(NZ),RK2,D(MM),SUM,P,Q,PAX,QT(MM)
      COMMON IMAGE,KC,KD,KM,REF,AY
C
      DO L=1,MM
       IF(AY(L).EQ.0.0)AY(L)=1.0
      ENDDO
      N1=1
      N2=MM
      GL=40
```

```
      GINC=15
      DO K=1,NT
       IF(K.EQ.1)THEN
        IMA(K)=GL
       ELSE
        IMA(K)=IMA(K-1)+GINC
       ENDIF
      ENDDO
      DO J=1,NXX
       DO I=1,NYY
        DO L=N1,N2
         D(L)=KM(I,J,L)/AY(L)
        ENDDO
        DO K=1,NT
         RK(K)=0.0
          DO L=N1,N2
           PAX=(D(L)-REF(K,L))**2
           RK(K)=RK(K)+(QT(L)*PAX)
          ENDDO
         ENDDO
         P=RK(1)
         KK=1
         DO K=1,NT
          IF(P.GT.RK(K))THEN
           P=RK(K)
           KK=K
          ENDIF
         ENDDO
         IMAGE(I,J)=IMA(KK)
        ENDDO
       ENDDO
       RETURN
       END
C ***********************************************************
        SUBROUTINE CLAVECT(NX,NT,QT,ICHAN,THR,TH)
C Subroutine to determine mean feature vectors of classes
C or categories and also performs normalisation of features.
        PARAMETER(NYY=512,NXX=512,MM=3,NZ=150,NB=64,NL=4096)
        INTEGER*2 IMAGE(NYY,NXX),IC,K,IFAT,NUT,IE,LL1,IW
        INTEGER*2 NAT,L1,L2,NT,JQ,ID,I1,I2,J1,J2,I,J
        INTEGER*2 ICON(NL),NR,NC,NY,NX,M,N,NN1,NN2,ICHAN
        INTEGER*2 KC(NYY,NXX),KD(NYY,NXX),NY1,NX1,LL2,KOUNT
        REAL KM(NYY,NXX,MM),BG,QT(MM)
        REAL FM(MM),FM1(MM),FM2(MM),FM3(MM),FM4(MM)
        REAL TN1,TN2,TM1,TM2,TN,THR,TH,PT,DAT,AZ(MM),Z2
        REAL EF(NB,NB,MM),EF1(NB,NB,MM),EF2(NB,NB,MM)
        REAL EF3(NB,NB,MM),EF4(NB,NB,MM),DB,UG,ZTT,Z1
        REAL FF(NL,MM),ZA,ZC,DNAT,BF(MM),THD,DET,UB,UM
        REAL REF(NZ,MM),AY(MM),BH,BM,BB,FG1,FG2,FG3,FG4
        COMMON IMAGE,KC,KD,KM,REF,AY
C
        NY=NX
        NR=NYY/NY
        NC=NXX/NX
        DAT=FLOAT(NY*NX)
```

```
      NY1=NY/2
      NX1=NX/2
      DET=FLOAT(NY1*NX1)
C Determine the allowable maximum and minimum number of
C neighbourhoods that can be considered uniform.
      LL1=(NR*NC)/3
      LL2=LL1/3
C
      L1=1
      L2=MM
      DB=FLOAT(MM)
      KOUNT=0
C -----------------------------------------------------
C Divide the image into blocks and compute the mean feature
C values for block and for each of its quarters.
      DO J1=1,NC
       DO I1=1,NR
        DO L=L1,L2
         FM(L)=0.0
         FM1(L)=0.0
         FM2(L)=0.0
         FM3(L)=0.0
         FM4(L)=0.0
        ENDDO
        DO N=1,NX
         J=(J1-1)*NX+N
         DO M=1,NY
          I=(I1-1)*NY+M
          DO L=L1,L2
           FM(L)=FM(L)+KM(I,J,L)
             IF((M.LE.NY1).AND.(N.LE.NX1))THEN
            FM1(L)=FM1(L)+KM(I,J,L)
           ENDIF
           IF((M.LE.NY1).AND.(N.GT.NX1))THEN
            FM2(L)=FM2(L)+KM(I,J,L)
           ENDIF
           IF((M.GT.NY1).AND.(N.LE.NX1))THEN
            FM3(L)=FM3(L)+KM(I,J,L)
           ENDIF
           IF((M.GT.NY1).AND.(N.GT.NX1))THEN
            FM4(L)=FM4(L)+KM(I,J,L)
           ENDIF
          ENDDO
         ENDDO
        ENDDO
        DO L=L1,L2
         EF(I1,J1,L)=FM(L)/DAT
         EF1(I1,J1,L)=FM1(L)/DET
         EF2(I1,J1,L)=FM2(L)/DET
         EF3(I1,J1,L)=FM3(L)/DET
         EF4(I1,J1,L)=FM4(L)/DET
        ENDDO
       ENDDO
      ENDDO
C -----------------------------------------------------
```

```
C Determine those blocks that can be considered uniform in
C terms of all the features. IE counts the number of
C such blocks.
 200        IE=0
        DO J1=1,NC
         DO I1=1,NR
           BM=0.0
           DO L=L1,L2
            BB=0.0
            IF(EF(I1,J1,L).GE.EF1(I1,J1,L))THEN
             Z1=EF(I1,J1,L)
             Z2=EF1(I1,J1,L)
            ELSE
             Z1=EF1(I1,J1,L)
             Z2=EF(I1,J1,L)
            ENDIF
            THD=THR*Z1
            IF(Z2.GE.THD)BB=BB+1.0
            IF(EF(I1,J1,L).GE.EF2(I1,J1,L))THEN
             Z1=EF(I1,J1,L)
             Z2=EF2(I1,J1,L)
            ELSE
             Z1=EF2(I1,J1,L)
             Z2=EF(I1,J1,L)
            ENDIF
            THD=THR*Z1
            IF(Z2.GE.THD)BB=BB+1.0
            IF(EF(I1,J1,L).GE.EF3(I1,J1,L))THEN
             Z1=EF(I1,J1,L)
             Z2=EF3(I1,J1,L)
            ELSE
             Z1=EF3(I1,J1,L)
             Z2=EF(I1,J1,L)
            ENDIF
            THD=THR*Z1
            IF(Z2.GE.THD)BB=BB+1.0
            IF(EF(I1,J1,L).GE.EF4(I1,J1,L))THEN
             Z1=EF(I1,J1,L)
             Z2=EF4(I1,J1,L)
            ELSE
             Z1=EF4(I1,J1,L)
             Z2=EF(I1,J1,L)
            ENDIF
            THD=THR*Z1
            IF(Z2.GE.THD)BB=BB+1.0
C
            IF(BB.EQ.4.0)BM=BM+1.0
           ENDDO
           IF(BM.EQ.DB)THEN
            IE=IE+1
            DO L=L1,L2
             FF(IE,L)=EF(I1,J1,L)
            ENDDO
            ICON(IE)=1
           ENDIF
```

```
            ENDDO
           ENDDO
           IF(KOUNT.EQ.0)THEN
            WRITE(ICHAN,*)'NUMBER OF NEIGHBOURHOODS CONSIDERED'
            WRITE(ICHAN,*)'UNIFORM AT THE INITIAL VALUE OF'
            WRITE(ICHAN,*)'UNIFORMITY CRITERION=',IE
            KOUNT=KOUNT+1
           ENDIF
C %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
C If the number of uniform blocks is greater than the
C allowable maximum, make the criterion stricter and
C determine the blocks considered uniform using the
C new criterion.
          IF(IE.GT.LL1)THEN
           THR=THR+TH
           GO TO 200
          ENDIF
C -------------------------------------------------------------
C If the number of uniform blocks is less than the allowable
C minimum, relax the criterion and determine the
C blocks considered uniform using the new criterion.
          IF(IE.LT.LL2)THEN
           THR=THR-TH
           GO TO 200
          ENDIF
C -------------------------------------------------------------
C If the number of uniform neighbourhoods is allowable, then
C cluster the mean feature vectors agglomeratively.
 202       IC=IE
 203       ZTT=10000000.0
           DO L=L1,L2
            AZ(L)=0.0
            DO K=1,IE
             AZ(L)=AZ(L)+FF(K,L)
            ENDDO
            IF(AZ(L).EQ.0.0)AZ(L)=1.0
           ENDDO
C
            DO I=1,IE
            IF(ICON(I).NE.0)THEN
             DO J=I+1,IE
             IF(ICON(J).NE.0)THEN
              DNAT=0.0
              DO L=L1,L2
               DNAT=DNAT+((FF(I,L)-FF(J,L))/AZ(L))**2
              ENDDO
              IF(ZTT.GT.DNAT)THEN
               ZTT=DNAT
               NN1=I
               NN2=J
              ENDIF
             ENDIF
            ENDDO
           ENDIF
          ENDDO
```

```
      TN1=FLOAT(ICON(NN1))
      TN2=FLOAT(ICON(NN2))
      TN=(TN1+TN2)*DAT
      TM1=TN1*DAT
      TM2=TN2*DAT
      DO L=L1,L2
       BF(L)=(TM1*FF(NN1,L)+TM2*FF(NN2,L))/TN
      ENDDO
      DO L=L1,L2
       FF(NN1,L)=BF(L)
       FF(NN2,L)=0.0
      ENDDO
      ICON(NN1)=ICON(NN1)+ICON(NN2)
      ICON(NN2)=0
      IC=IC-1
      IF(IC.GT.NT)GO TO 203
C %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
      WRITE(ICHAN,*)'THE FINAL VALUE OF UNIFORMITY'
      WRITE(ICHAN,*)'CRITERION. =',THR
      WRITE(ICHAN,*)'NO. OF MEAN VECTORS clustered =',IE
      NUT=0
      DO I=1,IE
       IF(ICON(I).NE.0)THEN
        NUT=NUT+1
        DO L=L1,L2
         REF(NUT,L)=FF(I,L)
        ENDDO
       ENDIF
      ENDDO
      WRITE(ICHAN,*)'THESE ARE AVERAGE VALUES OF FEATURES'
      WRITE(ICHAN,*)'FOR CATEGORIES'
      WRITE(ICHAN,70)((REF(I,J),J=L1,L2),I=1,NUT)
 70   FORMAT(<MM>F13.6)
C -------------------------------------------------------------
C Determine the feature normalizing factors and normalize
C the mean feature values.
      DO L=L1,L2
       AY(L)=0.0
       DO K=1,NT
        AY(L)=AY(L)+REF(K,L)
       ENDDO
       IF(AY(L).EQ.0.0)AY(L)=1.0
      ENDDO
      DO L=L1,L2
       DO K=1,NT
        REF(K,L)=REF(K,L)/AY(L)
       ENDDO
      ENDDO
C Determine the feature weighting factors using distance
C between means (contrast) criterion.
      DO L=L1,L2
       QT(L)=0.0
       DO I=1,NT-1
        DO J=I+1,NT
         ZA=ABS(REF(I,L)-REF(J,L))
```

```
        ZC=REF(I,L)+REF(J,L)
        IF(ZC.EQ.0.0)ZC=1.0
        QT(L)=QT(L)+(ZA/ZC)
       ENDDO
      ENDDO
     ENDDO
     ZC=0.0
     DO L=L1,L2
      IF(ZC.LT.QT(L))ZC=QT(L)
     ENDDO
     DO L=L1,L2
      IF(QT(L).NE.0.0)QT(L)=ZC/QT(L)
     ENDDO
     WRITE(ICHAN,*)'WEIGHTING FACTORS USING DISTANCE'
     WRITE(ICHAN,*)'BETWEEN MEANS CRITERION'
C
     WRITE(ICHAN,85)(QT(L),L=1,MM)
 85  FORMAT(<MM>F10.5)
C -------------------------------------------------------------
     RETURN
     END
```

```
           PROGRAM SPEG2
C -----------------------------------------------------------
C This a program for segmentation ALGORITHM II for
C the segmentation of any 3-band multispectral image.
C
C     INPUT    :    ANY 3-BAND MULTISPECTRAL IMAGE.
C
C     OUTPUT   :    SEGMENTED VERSION OF THE INPUT IMAGE
C
C -----------------------------------------------------------
      PARAMETER(NYY=512,NXX=512,MM=3,NZ=150)
      INTEGER*2 IMAGE(NYY,NXX),KC(NYY,NXX),IC
      INTEGER*2 KD(NYY,NXX),IB,NT,ID,NX,ICHAN
      REAL KM(NYY,NXX,MM),QT(MM)
      REAL REF(NZ,MM),THR,AY(MM)
      INTEGER*4 STATUSFLAG,LINENUMBER
      COMMON IMAGE,KM,KC,KD,REF,AY
C
      CALL VICINIT('SPEG2')
      CALL OPENV(STATUSFLAG,2,0,0,0,0)
      CALL OPENV(STATUSFLAG,3,0,0,0,0)
      CALL OPENV(STATUSFLAG,4,0,0,0,0)
      DO LINENUMBER=1,NYY
       CALL READ(STATUSFLAG,2,0,1,0,NXX,KC(1,LINENUMBER),0)
       CALL READ(STATUSFLAG,3,0,1,0,NXX,KD(1,LINENUMBER),0)
       CALL READ(STATUSFLAG,4,0,1,0,NXX,IMAGE(1,LINENUMBER),0)
      ENDDO
C
      WRITE(6,*)'DO YOU WANT TO USE THE GRAY LEVELS OF'
      WRITE(6,*)'PIXELS DIRECTLY OR THE AVERAGE GRAY LEVEL'
      WRITE(6,*)'IN A SMALL WINDOW CENTERED ON A PIXEL, FOR'
      WRITE(6,*)'SEGMENTATION?. IF PIXEL GRAY LEVELS INPUT'
      WRITE(6,*)'0, OTHERWISE INPUT 1'
      READ(5,*)ITEST
      IF(ITEST.EQ.1)THEN
       WRITE(6,*)'INPUT THE DISTANCE FOR SPECIFYING THE SIZE OF'
       WRITE(6,*)'THIS WINDOW; IB. NOTE W=(2*IB+1)*(2*IB+1)'
       READ(5,*)IB
      ELSE
       IB=1
      ENDIF
C
      WRITE(6,*)'INPUT THE NUMBER OF CATEGORIES; NT'
      READ(5,*)NT
C
      WRITE(6,*)'INPUT THE DISTANCE FOR SPECIFYING DIMENSION'
      WRITE(6,*)'OF LARGEST SEARCH BLOCKS; NX.'
      WRITE(6,*)'NOTE: DIMENSION=NX*NX.'
      READ(5,*)NX
C
      WRITE(6,*)'INPUT THE VALUE OF THE UNIFORMITY'
      WRITE(6,*)'CRITERION; THR, A REAL NUMBER.'
      READ(5,*)THR
C ----------------------------------------------------------------
      WRITE(6,*)'INPUT THE CHANNEL NUMBER FOR WRITING;ICHAN'
```

```
      READ(5,*)ICHAN
C
      CALL FEATURE(IB,ITEST)
      CALL CLAVECT(NX,NT,QT,ICHAN,THR)
      CALL ASSIGN(NT,QT)
C
C Write out the segmented image
      CALL OPENV(STATUSFLAG,1,1,0,0,0)
      CALL ADJUST(1,NYY,NXX)
      DO I=1,NYY
       CALL WRITE(STATUSFLAG,1,0,1,0,NXX,IMAGE(1,I),0)
      ENDDO
      CALL RELAB2(1,NYY,NXX)
      END
C ***********************************************************
              SUBROUTINE FEATURE(IB,ITEST)
C Subroutine to compute features.
      PARAMETER(NYY=512,NXX=512,MM=3,NG=255,LL=10,NZ=150)
      INTEGER*2 IMAGE(NYY,NXX),KC(NYY,NXX)
      INTEGER*2 IP1,IP2,IQ1,IQ2,ITEST,IB,ID
      INTEGER*2 M,N,I,J,I1,J1,KD(NYY,NXX)
      REAL REF(NZ,MM),QT(MM),DNAT
      REAL DM,DB,DG,AY(MM),KM(NYY,NXX,MM)
      COMMON IMAGE,KM,KC,KD,REF,AY
C -----------------------------------------------------------
C If desired, replace the gray level of each pixel by the
C average gray level in a window centered on it.
      IF(ITEST.EQ.1)THEN
       DO J1=1,NXX
        J=J1
        DO I1=1,NYY
         I=I1
         KM(I1,J1,1)=0.0
         KM(I1,J1,2)=0.0
         KM(I1,J1,3)=0.0
         IP1=-IB
         IP2=IB
         IQ1=-IB
         IQ2=IB
         IF(I.LE.IB)THEN
          IP1=IB+1
          IP2=IP1+IB+I
          I=0
         ENDIF
         IF(I.GT.(NYY-IB))THEN
          IP2=NYY-IB
          IP1=IP2-(IB+(NYY-I))
          I=0
         ENDIF
         IF(J.LE.IB)THEN
          IQ1=IB+1
          IQ2=IQ1+IB+J
          J=0
         ENDIF
         IF(J.GT.(NXX-IB))THEN
```

```
        IQ2=NXX-IB
        IQ1=IQ2-(IB+(NXX-J))
        J=0
       ENDIF
       DNAT=0.0
       DM=0.0
       DB=0.0
       DG=0.0
       DO L=IQ1,IQ2
        DO K=IP1,IP2
         DNAT=DNAT+1.0
         DM=DM+FLOAT(KC(I+K,J+L))
         DB=DB+FLOAT(KD(I+K,J+L))
         DG=DG+FLOAT(IMAGE(I+K,J+L))
        ENDDO
       ENDDO
       KM(I1,J1,1)=DM/DNAT
       KM(I1,J1,2)=DB/DNAT
       KM(I1,J1,3)=DG/DNAT
      ENDDO
     ENDDO
C -------------------------------------------------------
      ELSE
       DO J=1,NXX
        DO I=1,NYY
         KM(I,J,1)=FLOAT(KC(I,J))
         KM(I,J,2)=FLOAT(KD(I,J))
         KM(I,J,3)=FLOAT(IMAGE(I,J))
        ENDDO
       ENDDO
      ENDIF
      RETURN
      END
C ***********************************************************
         SUBROUTINE ASSIGN(NT,QT)
C Subroutine to classify pixels.
      PARAMETER(NYY=512,NXX=512,MM=3,NZ=150)
      INTEGER*2 IMAGE(NYY,NXX),IMA(NZ),NT,KK
      INTEGER*2 PEF(NZ),GL,GINC,IFEAT,N1,N2
      INTEGER*2 KC(NYY,NXX),KD(NYY,NXX)
      REAL KM(NYY,NXX,MM),QT(MM)
      REAL REF(NZ,MM),AY(MM),KR1,KR2,PP,QQ
      REAL RK(NZ),RK2,D(MM),SUM,P,Q,PAX
      COMMON IMAGE,KM,KC,KD,REF,AY
C
      DO L=1,MM
       IF(AY(L).EQ.0.0)AY(L)=1.0
      ENDDO
      N1=1
      N2=MM
      GL=40
      GINC=15
      DO K=1,NT
       IF(K.EQ.1)THEN
        IMA(K)=GL
```

```
          ELSE
           IMA(K)=IMA(K-1)+GINC
          ENDIF
         ENDDO
         DO J=1,NXX
          DO I=1,NYY
           DO L=N1,N2
            D(L)=KM(I,J,L)/AY(L)
           ENDDO
           DO K=1,NT
            RK(K)=0.0
            DO L=N1,N2
             PAX=(D(L)-REF(K,L))**2
             RK(K)=RK(K)+(QT(L)*PAX)
            ENDDO
           ENDDO
           P=RK(1)
           KK=1
             DO K=1,NT
            IF(P.GT.RK(K))THEN
             P=RK(K)
             KK=K
            ENDIF
           ENDDO
           IMAGE(I,J)=IMA(KK)
          ENDDO
         ENDDO
         RETURN
         END
C *********************************************************
            SUBROUTINE CLAVECT(NX,NT,QT,ICHAN,THR)
C Subroutine to determine mean feature vectors for classes
C or categories and also performs normalisation of features.
C There are presently only two levels of splitting of non-
C uniform neighbourhoods i.e from NX X NX  to NX/n X NX/n,
C where n=2**2
         PARAMETER(NYY=512,NXX=512,MM=3,NZ=150,NB=64,NL=4096)
         INTEGER*2 IMAGE(NYY,NXX),IC,K,IFEAT,NUT,KOUNT
         INTEGER*2 L1,L2,NT,JQ,ID,I1,I2,J1,J2,I,ED2(2,2)
         INTEGER*2 NX,M,N,NN1,NN2,ICHAN,IE,J,ITEST,IFAT,NC
         INTEGER*2 KC(NYY,NXX),KD(NYY,NXX),MX1,MX2,MX3,NR
         REAL KM(NYY,NXX,MM),REF(NZ,MM),BG,QT(MM),ZTT
         REAL TN1,TN2,TM1,TM2,TN,DF(MM),AZ(MM)
         REAL FF(NL,MM),ZA,ZC,DNAT,BF(MM),THR,DET
         REAL ICAN(NL),UB,UM,UG,AY(MM),IMAG3(NB,NB,MM)
         REAL IMAG1(NB,NB,MM),IMAG2(NB,NB,MM)
         COMMON IMAGE,KM,KC,KD,REF,AY
C
         MX1=NX
         NR=NYY/NX
         NC=NXX/NX
         IE=0
         L1=1
         L2=MM
         IFAT=(L2-L1)+1
```

```
C %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
C Divide the image into blocks of size NX*NX and test
C for uniformity of blocks.compute the average feaure
C values for blocks considered uniform.
      DO JJ=1,NC
       DO II=1,NR
        DO N=1,MX1
         J=(JJ-1)*MX1+N
         DO M=1,MX1
          I=(II-1)*MX1+M
          DO L=L1,L2
           IMAG1(M,N,L)=KM(I,J,L)
          ENDDO
         ENDDO
        ENDDO
        CALL UNIFORM(IMAG1,MX1,ITEST,DF,L1,L2,THR)
        IF(ITEST.EQ.1)THEN
         IE=IE+1
         ICAN(IE)=FLOAT(MX1**2)
         DO L=L1,L2
          FF(IE,L)=DF(L)
         ENDDO
         GO TO 100
        ENDIF
C -----------------------------------------------------------
C If block of size NX*NX is not uniform, split it into four
C four subblocks of size MX2*MX2 (where MX2=NX/2).Test for
C uniformity of each subblock and compute the average
C feature values for those considered uniform.
       MX2=MX1/2
       KOUNT=0
       DO J1=1,2
        DO I1=1,2
         ED2(I1,J1)=0
         DO N=1,MX2
          J=(J1-1)*MX2+N
          DO M=1,MX2
           I=(I1-1)*MX2+M
           DO L=L1,L2
            IMAG2(M,N,L)=IMAG1(I,J,L)
           ENDDO
          ENDDO
         ENDDO
         CALL UNIFORM(IMAG2,MX2,ITEST,DF,L1,L2,THR)
         IF(ITEST.EQ.1)THEN
          IE=IE+1
          ICAN(IE)=FLOAT(MX2**2)
          DO L=L1,L2
           FF(IE,L)=DF(L)
          ENDDO
          ED2(I1,J1)=1
          KOUNT=KOUNT+1
         ENDIF
        ENDDO
         ENDDO
```

```
      IF(KOUNT.EQ.4)GO TO 100
C -------------------------------------------------------
C If any of the subblock of size MX2*MX2 is not uniform,
C split the subblock further into four portions,
C each of size MX3*MX3 (where MX3=MX2/2). Test for
C uniformity of each portion and compute the average
C feature values for those considered uniform.
       MX3=MX2/2
       DO J1=1,2
        DO I1=1,2
         IF(ED2(I1,J1).NE.1)THEN
          DO N=1,MX2
           J=(J1-1)*MX2+N
           DO M=1,MX2
            I=(I1-1)*MX2+M
            DO L=L1,L2
             IMAG2(M,N,L)=IMAG1(I,J,L)
            ENDDO
           ENDDO
          ENDDO
          DO J2=1,2
           DO I2=1,2
            DO N=1,MX3
             J=(J2-1)*MX3+N
             DO M=1,MX3
              I=(I2-1)*MX3+M
              DO L=L1,L2
               IMAG3(M,N,L)=IMAG2(I,J,L)
              ENDDO
             ENDDO
            ENDDO
            CALL UNIFORM(IMAG3,MX3,ITEST,DF,L1,L2,THR)
            IF(ITEST.EQ.1)THEN
             IE=IE+1
             ICAN(IE)=FLOAT(MX3**2)
             DO L=L1,L2
              FF(IE,L)=DF(L)
             ENDDO
            ENDIF
           ENDDO
          ENDDO
         ENDIF
        ENDDO
       ENDDO
C -------------------------------------------------------
 100   CONTINUE
      ENDDO
     ENDDO
C %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
C Cluster the mean vectors agglomeratively.
 202 IC=IE
 203 ZTT=10000000.0
     DO L=L1,L2
      AZ(L)=0.0
      DO K=1,IE
```

```
          AZ(L)=AZ(L)+FF(K,L)
         ENDDO
        ENDDO
         DO I=1,IE
         IF(ICAN(I).NE.0.0)THEN
          DO J=I+1,IE
           IF(ICAN(J).NE.0.0)THEN
            DNAT=0.0
            DO L=L1,L2
             DNAT=DNAT+((FF(I,L)-FF(J,L))/AZ(L))**2
            ENDDO
            IF(ZTT.GT.DNAT)THEN
             ZTT=DNAT
             NN1=I
             NN2=J
            ENDIF
           ENDIF
          ENDDO
         ENDIF
        ENDDO
        TM1=ICAN(NN1)
        TM2=ICAN(NN2)
        TN=TM1+TM2
        DO L=L1,L2
         BF(L)=(TM1*FF(NN1,L)+TM2*FF(NN2,L))/TN
        ENDDO
        DO L=L1,L2
         FF(NN1,L)=BF(L)
         FF(NN2,L)=0.0
        ENDDO
        ICAN(NN1)=ICAN(NN1)+ICAN(NN2)
        ICAN(NN2)=0.0
        IC=IC-1
        IF(IC.GT.NT)GO TO 203
C %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        WRITE(ICHAN,*)'NO. OF VECTORS CLUSTERED =',IE
        NUT=0
        DO I=1,IE
         IF(ICAN(I).NE.0.0)THEN
          NUT=NUT+1
          DO L=L1,L2
           REF(NUT,L)=FF(I,L)
          ENDDO
         ENDIF
        ENDDO
        WRITE(ICHAN,*)'THESE ARE AVERAGE VALUES OF FEATURES'
        WRITE(6,*)'FOR CATEGORIES'
        WRITE(ICHAN,70)((REF(I,J),J=L1,L2),I=1,NUT)
 70     FORMAT(<IFAT>F13.6)
C
C ------------------------------------------------------------
C Determine the feature normalizing factors and normalize
C the mean feature values.
         DO L=L1,L2
         AY(L)=0.0
```

```
         DO K=1,NT
          AY(L)=AY(L)+REF(K,L)
         ENDDO
         IF(AY(L).EQ.0.0)AY(L)=1.0
        ENDDO
        DO L=L1,L2
         DO K=1,NT
          REF(K,L)=REF(K,L)/AY(L)
         ENDDO
        ENDDO
C Determine the feature weighting factors using distance
C between means criterion.
        DO L=L1,L2
         QT(L)=0.0
         DO I=1,NT-1
          DO J=I+1,NT
           ZA=ABS(REF(I,L)-REF(J,L))
           ZC=REF(I,L)+REF(J,L)
           IF(ZC.EQ.0.0)ZC=1.0
           QT(L)=QT(L)+(ZA/ZC)
          ENDDO
         ENDDO
        ENDDO
        ZC=0.0
        DO L=L1,L2
         IF(ZC.LT.QT(L))ZC=QT(L)
        ENDDO
        DO L=L1,L2
         IF(QT(L).NE.0.0)QT(L)=ZC/QT(L)
        ENDDO
        WRITE(ICHAN,*)'WEIGHTING FACTORS USING DISTANCE'
        WRITE(ICHAN,*)'BETWEEN MEANS CRITERION'
C
        WRITE(ICHAN,85)(QT(L),L=1,MM)
 85     FORMAT(<IFAT>F10.5)
C ------------------------------------------------------------
        RETURN
        END
C ************************************************************
           SUBROUTINE UNIFORM(IMAG,MX,ITEST,DF,L1,L2,THR)
C Subroutine to determine uniformity of neighbourhoods.
        PARAMETER(NB=64,MM=3)
        INTEGER*2 ITEST,MX,LX,L1,L2
        REAL Z1,Z2,THR,BM,EF(MM),EF1(MM)
        REAL IMAG(NB,NB,MM),DF(MM),EF2(MM),BB
        REAL DAT,DET,EF3(MM),EF4(MM),DB
C
        LX=MX/2
        DAT=FLOAT(MX**2)
        DET=FLOAT(LX**2)
        DB=FLOAT(L2-L1)+1.0
        ITEST=0
C
        DO L=L1,L2
         EF(L)=0.0
```

```
          EF1(L)=0.0
          EF2(L)=0.0
          EF3(L)=0.0
          EF4(L)=0.0
          DF(L)=0.0
         ENDDO
         DO N=1,MX
          DO M=1,MX
           DO L=L1,L2
            EF(L)=EF(L)+IMAG(M,N,L)
            IF((M.LE.LX).AND.(N.LE.LX))THEN
             EF1(L)=EF1(L)+IMAG(M,N,L)
            ENDIF
            IF((M.LE.LX).AND.(N.GT.LX))THEN
             EF2(L)=EF2(L)+IMAG(M,N,L)
            ENDIF
            IF((M.GT.LX).AND.(N.LE.LX))THEN
             EF3(L)=EF3(L)+IMAG(M,N,L)
            ENDIF
            IF((M.GT.LX).AND.(N.GT.LX))THEN
             EF4(L)=EF4(L)+IMAG(M,N,L)
            ENDIF
           ENDDO
          ENDDO
         ENDDO
         DO L=L1,L2
          EF(L)=EF(L)/DAT
          EF1(L)=EF1(L)/DET
          EF2(L)=EF2(L)/DET
          EF3(L)=EF3(L)/DET
          EF4(L)=EF4(L)/DET
         ENDDO
C
         BM=0.0
         DO L=L1,L2
          BB=0.0
          IF(EF(L).GE.EF1(L))THEN
           Z1=EF(L)
           Z2=EF1(L)
          ELSE
           Z1=EF1(L)
           Z2=EF(L)
          ENDIF
          THD=THR*Z1
          IF(Z2.GE.THD)BB=BB+1.0
          IF(EF(L).GE.EF2(L))THEN
           Z1=EF(L)
           Z2=EF2(L)
          ELSE
           Z1=EF2(L)
           Z2=EF(L)
          ENDIF
          THD=THR*Z1
          IF(Z2.GE.THD)BB=BB+1.0
          IF(EF(L).GE.EF3(L))THEN
```

```
            Z1=EF(L)
            Z2=EF3(L)
           ELSE
            Z1=EF3(L)
            Z2=EF(L)
           ENDIF
           THD=THR*Z1
           IF(Z2.GE.THD)BB=BB+1.0
           IF(EF(L).GE.EF4(L))THEN
            Z1=EF(L)
            Z2=EF4(L)
           ELSE
            Z1=EF4(L)
            Z2=EF(L)
           ENDIF
           THD=THR*Z1
           IF(Z2.GE.THD)BB=BB+1.0
           IF(BB.EQ.4.0)BM=BM+1.0
          ENDDO
C
          IF(BM.EQ.DB)THEN
           ITEST=1
           DO L=L1,L2
            DF(L)=EF(L)
           ENDDO
          ENDIF
C
          RETURN
          END
```

```
          PROGRAM SEG1
C --------------------------------------------------------
C Program for segmentation ALGORITHM II for the
C segmentation of black-and-white/monochrome image.
C
C INPUT      : ANY BLACK-AND-WHITE OR MONOCHROME IMAGE
C
C OUTPUT     : SEGMENTED VERSION OF THE INPUT IMAGE
C
C COMMENTS   : Depending on the image, the user may
C            : segment on the basis of texture, or
C            : brightness ,or on the basis of both.
C --------------------------------------------------------
    PARAMETER(NYY=256,NXX=256,MM=3,NZ=150)
    INTEGER*2 IMAGE(NYY,NXX),IC,ID,NT,IB,IFEAT,IA
    INTEGER*2 NY,NX,ISL,ISS,NL,NS,GL,GINC,ICHAN
    REAL KM(NYY,NXX,MM),THR,TH
    REAL REF(NZ,MM),AY(MM),QT(MM)
    INTEGER*4 STATUSFLAG,LINENUMBER
    COMMON IMAGE,KM,REF,AY
C
C Read in the image.
    CALL VICINIT('SEG1')
    CALL OPENV(STATUSFLAG,2,0,0,0,0)
    DO LINENUMBER=1,NYY
     CALL READ(STATUSFLAG,2,0,1,0,NXX,IMAGE(1,LINENUMBER),0)
    ENDDO
C ----------------------------------------------------------------
C Supply the required segmentation parameters.
    WRITE(6,*)'WHAT TYPE OF FEATURES ARE TO BE USED FOR'
    WRITE(6,*)'SEGMENTATION; BRIGHTNESS, TEXTURE OR A'
    WRITE(6,*)'COMBINATION OF BOTH?. IF ONLY BRIGHTNESS'
    WRITE(6,*)'INPUT 1, IF ONLY TEXTURE INPUT 2 OR IF'
    WRITE(6,*)'A COMBINATION OF BOTH INPUT 3'
    READ(5,*)IFEAT
C
    IF(IFEAT.GT.1)THEN
     ID=3
     WRITE(6,*)'INPUT THE DISTANCE FOR SPECIFYING THE'
     WRITE(6,*)'CHARACTREIZATION WINDOW SIZE FOR THE'
     WRITE(6,*)'COMPUTATION OF TEXTURAL FEATURES; IB'
     READ(5,*)IB
    ENDIF
    IF(IFEAT.EQ.1)THEN
     ID=1
     WRITE(6,*)'NOW, THAT YOU ARE USING ONLY BRIGHTNESS,'
     WRITE(6,*)'DO YOU WANT TO USE THE GRAY LEVELS OF THE'
     WRITE(6,*)'PIXELS DIRECTLY OR THE AVERAGE GRAY LEVEL'
     WRITE(6,*)'IN SMALL WINDOWS CENTERED ON PIXELS?.'
     WRITE(6,*)'IF AVERAGE INPUT 1, OTHERWISE INPUT 0'
     READ(5,*)IA
     IF(IA.EQ.1)THEN
      WRITE(6,*)'THEN, INPUT THE DISTANCE FOR SPECIFYING'
      WRITE(6,*)'THE SIZE OF THIS WINDOW; IB'
      READ(5,*)IB
```

```
        ELSE
         IB=0
        ENDIF
       ENDIF
C
       WRITE(6,*)'INPUT THE DISTANCE FOR SPECIFYING'
       WRITE(6,*)'DIMENSIONS OF SEARCH BLOCKS; NX'
       WRITE(6,*)'NOTE: DIMENSION= NX x NX'
       READ(5,*)NX
C
       WRITE(6,*)'INPUT THE NUMBER OF CATEGORIES; NT'
       READ(5,*)NT
       WRITE(6,*)'INPUT THE INITIAL VALUE OF UNIFORMITY'
       WRITE(6,*)'CRITERION; THR AND THE INCREMENTAL/'
       WRITE(6,*)'DECREMENTAL FACTOR; TH, BOTH ARE'
       WRITE(6,*)'REAL NUMBERS.'
       READ(5,*)THR,TH
C -----------------------------------------------------
       WRITE(6,*)'INPUT THE CHANNEL NUMBER FOR WRITING'
       WRITE(6,*)'AN INTEGER'
       READ(5,*)ICHAN
C
       CALL FEATURE(IB,ID,IFEAT)
       CALL CLAVECT(NX,NT,IFEAT,QT,ICHAN,THR,TH)
       CALL ASSIGN(NT,IFEAT,QT)
C
       CALL OPENV(STATUSFLAG,1,1,0,0,0)
       CALL ADJUST(1,NYY,NXX)
       DO I=1,NYY
        CALL WRITE(STATUSFLAG,1,0,1,0,NXX,IMAGE(1,I),0)
       ENDDO
       CALL RELAB2(1,NYY,NXX)
       END
C ***************************************************
          SUBROUTINE ASSIGN(NT,IFEAT,QT)
C Subroutine to classify pixels.
       PARAMETER(NYY=256,NXX=256,MM=3,NZ=150)
       INTEGER*2 IMAGE(NYY,NXX),IMA(NZ),NT,KK
       INTEGER*2 PEF(NZ),GL,GINC,IFEAT,N1,N2
       REAL KM(NYY,NXX,MM)
       REAL REF(NZ,MM),AY(MM),KR1,KR2,PP,QQ
       REAL RK(NZ),RK2,D(MM),SUM,P,Q,PAX,QT(MM)
       COMMON IMAGE,KM,REF,AY
C
       IF(IFEAT.EQ.1)THEN
        N1=3
        N2=3
       ENDIF
       IF(IFEAT.EQ.2)THEN
        N1=1
        N2=2
       ENDIF
       IF(IFEAT.EQ.3)THEN
        N1=1
        N2=3
```

```
      ENDIF
      GL=40
      GINC=15
C
      DO L=1,MM
       IF(AY(L).EQ.0.0)AY(L)=1.0
      ENDDO
C
      DO K=1,NT
       IF(K.EQ.1)THEN
        IMA(K)=GL
       ELSE
        IMA(K)=IMA(K-1)+GINC
       ENDIF
      ENDDO
      DO J=1,NXX
       DO I=1,NYY
        DO L=N1,N2
         D(L)=KM(I,J,L)/AY(L)
        ENDDO
        DO K=1,NT
         RK(K)=0.0
         DO L=N1,N2
          PAX=(D(L)-REF(K,L))**2
          RK(K)=RK(K)+(QT(L)*PAX)
         ENDDO
        ENDDO
        P=RK(1)
        KK=1
        DO K=1,NT
         IF(P.GT.RK(K))THEN
          P=RK(K)
          KK=K
         ENDIF
        ENDDO
        IMAGE(I,J)=IMA(KK)
       ENDDO
      ENDDO
      RETURN
      END
C ******************************************************
       SUBROUTINE CLAVECT(NX,NT,IFEAT,QT,ICHAN,THR,TH)
C Subroutine to determine mean feature vectors for classes
C or categories and also performs normalisation of features.
      PARAMETER(NYY=256,NXX=256,MM=3,NZ=150,NB=64,NL=4096)
      INTEGER*2 IMAGE(NYY,NXX),IC,K,IFEAT,NUT,IE,IW
      INTEGER*2 NAT,L1,L2,NT,JQ,ID,I1,I2,J1,J2,I,J
      INTEGER*2 LL1,LL2,NY1,NX1,NN1,NN2,ICHAN
      INTEGER*2 ICON(NL),NR,NC,NY,NX,M,N,KOUNT
      REAL FM(MM),FM1(MM),FM2(MM),FM3(MM),FM4(MM)
      REAL KM(NYY,NXX,MM),THD,UM,UB,UG,ZTT,BG,QT(MM)
      REAL TN1,TN2,TM1,TM2,TN,THR,TH,PT,DAT,AZ(MM)
      REAL EF(NB,NB,MM),EF1(NB,NB,MM),EF2(NB,NB,MM)
      REAL EF3(NB,NB,MM),EF4(NB,NB,MM)
      REAL FF(NL,MM),ZA,ZC,DNAT,BF(MM),DET
```

```
      REAL REF(NZ,MM),AY(MM),BH,BM,BB,DB
      COMMON IMAGE,KM,REF,AY
C
      NY=NX
      NR=NYY/NY
      NC=NXX/NX
      DAT=FLOAT(NY*NX)
      NY1=NY/2
      NX1=NX/2
      DET=FLOAT(NY1*NX1)
C ---------------------------------------------------
C Determine allowable maximum and minimum number
C of neighbourhoods that can be considered uniform.
      IF(NYY.EQ.512)THEN
       LL1=(NR*NC)/3
      ENDIF
      IF(NYY.EQ.256)THEN
       LL1=(3*(NR*NC))/4
      ENDIF
      LL2=LL1/3
C ---------------------------------------------------
      IF(IFEAT.EQ.1)THEN
       L1=3
       L2=3
      ENDIF
      IF(IFEAT.EQ.2)THEN
       L1=1
       L2=2
      ENDIF
      IF(IFEAT.EQ.3)THEN
       L1=1
       L2=3
      ENDIF
      DB=FLOAT(L2-L1)+1.0
      KOUNT=0
C -------------------------------------------------------
C Divide the image into small blocks and compute the
C mean feature values for block and for each of its
C quarters.

      DO J1=1,NC
       DO I1=1,NR
        DO L=L1,L2
         FM(L)=0.0
         FM1(L)=0.0
         FM2(L)=0.0
         FM3(L)=0.0
         FM4(L)=0.0
        ENDDO
        DO N=1,NX
         J=(J1-1)*NX+N
         DO M=1,NY
          I=(I1-1)*NY+M
          DO L=L1,L2
           FM(L)=FM(L)+KM(I,J,L)
```

```
            IF((M.LE.NY1).AND.(N.LE.NX1))THEN
             FM1(L)=FM1(L)+KM(I,J,L)
            ENDIF
            IF((M.LE.NY1).AND.(N.GT.NX1))THEN
             FM2(L)=FM2(L)+KM(I,J,L)
            ENDIF
            IF((M.GT.NY1).AND.(N.LE.NX1))THEN
             FM3(L)=FM3(L)+KM(I,J,L)
            ENDIF
            IF((M.GT.NY1).AND.(N.GT.NX1))THEN
             FM4(L)=FM4(L)+KM(I,J,L)
            ENDIF
           ENDDO
          ENDDO
         ENDDO
         DO L=L1,L2
          EF(I1,J1,L)=FM(L)/DAT
          EF1(I1,J1,L)=FM1(L)/DET
          EF2(I1,J1,L)=FM2(L)/DET
          EF3(I1,J1,L)=FM3(L)/DET
          EF4(I1,J1,L)=FM4(L)/DET
         ENDDO
C
        ENDDO
       ENDDO
C -----------------------------------------------------------
C Determine those blocks that can be considered uniform
C in terms of all the features. IE counts the number of
C such blocks.
 200   IE=0
       DO J1=1,NC
        DO I1=1,NR
         BM=0.0
         DO L=L1,L2
          BB=0.0
          IF(EF(I1,J1,L).GE.EF1(I1,J1,L))THEN
           Z1=EF(I1,J1,L)
           Z2=EF1(I1,J1,L)
          ELSE
           Z1=EF1(I1,J1,L)
           Z2=EF(I1,J1,L)
          ENDIF
          THD=THR*Z1
          IF(Z2.GE.THD)BB=BB+1.0
          IF(EF(I1,J1,L).GE.EF2(I1,J1,L))THEN
           Z1=EF(I1,J1,L)
           Z2=EF2(I1,J1,L)
          ELSE
           Z1=EF2(I1,J1,L)
           Z2=EF(I1,J1,L)
          ENDIF
          THD=THR*Z1
          IF(Z2.GE.THD)BB=BB+1.0
          IF(EF(I1,J1,L).GE.EF3(I1,J1,L))THEN
           Z1=EF(I1,J1,L)
```

```
          Z2=EF3(I1,J1,L)
         ELSE
          Z1=EF3(I1,J1,L)
          Z2=EF(I1,J1,L)
         ENDIF
         THD=THR*Z1
         IF(Z2.GE.THD)BB=BB+1.0
         IF(EF(I1,J1,L).GE.EF4(I1,J1,L))THEN
          Z1=EF(I1,J1,L)
          Z2=EF4(I1,J1,L)
         ELSE
          Z1=EF4(I1,J1,L)
          Z2=EF(I1,J1,L)
         ENDIF
         THD=THR*Z1
         IF(Z2.GE.THD)BB=BB+1.0
C
         IF(BB.EQ.4.0)BM=BM+1.0
        ENDDO
        IF(BM.EQ.DB)THEN
         IE=IE+1
         DO L=L1,L2
          FF(IE,L)=EF(I1,J1,L)
         ENDDO
         ICON(IE)=1
        ENDIF
       ENDDO
      ENDDO
      IF(KOUNT.EQ.0)THEN
       WRITE(ICHAN,*)'NUMBER OF NEIGHBOURHOODS CONSIDERED'
       WRITE(ICHAN,*)'UNIFORM AT THE INITIAL VALUE OF'
       WRITE(ICHAN,*)'UNIFORMITY CRITERION =',IE
      ENDIF
C %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
C If the number of uniform blocks is greater than the
C allowable maximum, make the criterion stricter and
C determine the blocks considered uniform using the
C new criterion.
      IF(IE.GT.LL1)THEN
       THR=THR+TH
       GO TO 200
      ENDIF
C ____________________________________________________________
C If the number of uniform blocks is less than the
C allowable minimum, relax the criterion and determine
C the blocks considered uniform using the new criterion.
      IF(IE.LT.LL2)THEN
       THR=THR-TH
       GO TO 200
      ENDIF
C ______________________________________________________________
C If the number of uniform neighbourhoods is allowable,
C cluster the mean vectors agglomeratively.
  202 IC=IE
  203 ZTT=10000000.0
```

```
      DO L=L1,L2
       AZ(L)=0.0
       DO K=1,IE
        AZ(L)=AZ(L)+FF(K,L)
       ENDDO
      ENDDO
C
      DO I=1,IE
       IF(ICON(I).NE.0)THEN
        DO J=I+1,IE
         IF(ICON(J).NE.0)THEN
          DNAT=0.0
          DO L=L1,L2
           DNAT=DNAT+((FF(I,L)-FF(J,L))/AZ(L))**2
          ENDDO
          IF(ZTT.GT.DNAT)THEN
           ZTT=DNAT
           NN1=I
           NN2=J
          ENDIF
         ENDIF
        ENDDO
       ENDIF
      ENDDO
      TN1=FLOAT(ICON(NN1))
      TN2=FLOAT(ICON(NN2))
      TN=(TN1+TN2)*DAT
      TM1=TN1*DAT
      TM2=TN2*DAT
      DO L=L1,L2
       BF(L)=(TM1*FF(NN1,L)+TM2*FF(NN2,L))/TN
      ENDDO
      DO L=L1,L2
       FF(NN1,L)=BF(L)
       FF(NN2,L)=0.0
      ENDDO
      ICON(NN1)=ICON(NN1)+ICON(NN2)
      ICON(NN2)=0
      IC=IC-1
      IF(IC.GT.NT)GO TO 203
C %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
      WRITE(ICHAN,*)'THE FINAL VALUE OF UNIFORMITY'
      WRITE(ICHAN,*)'CRITERION =',THR
      WRITE(ICHAN,*)'NO. OF MEAN VECTORS CLUSTERED =',IE
      NUT=0
      DO I=1,IE
       IF(ICON(I).NE.0)THEN
        NUT=NUT+1
        DO L=L1,L2
         REF(NUT,L)=FF(I,L)
        ENDDO
       ENDIF
      ENDDO
      I2=(L2-L1)+1
      WRITE(ICHAN,*)'THESE ARE AVERAGE VALUES OF FEATURES'
```

```
      WRITE(ICHAN,*)'FOR CATEGORIES'
      WRITE(ICHAN,70)((REF(I,J),J=L1,L2),I=1,NUT)
 70   FORMAT(<I2>F13.6)
C
C -----------------------------------------------------------
C Determine the feature normalizing factors and normalize
C the mean feature values.
       DO L=L1,L2
       AY(L)=0.0
       DO K=1,NT
        AY(L)=AY(L)+REF(K,L)
       ENDDO
       IF(AY(L).EQ.0.0)AY(L)=1.0
      ENDDO
      DO L=L1,L2
       DO K=1,NT
        REF(K,L)=REF(K,L)/AY(L)
       ENDDO
      ENDDO
C Determine the feature weighting factors using distance
C between means (contrast) criterion.
      DO L=L1,L2
       QT(L)=0.0
       DO I=1,NT-1
        DO J=I+1,NT
         ZA=ABS(REF(I,L)-REF(J,L))
         ZC=REF(I,L)+REF(J,L)
         IF(ZC.EQ.0.0)ZC=1.0
         QT(L)=QT(L)+(ZA/ZC)
        ENDDO
       ENDDO
      ENDDO
      ZC=0.0
      DO L=L1,L2
       IF(ZC.LT.QT(L))ZC=QT(L)
      ENDDO
      DO L=L1,L2
       IF(QT(L).NE.0.0)QT(L)=ZC/QT(L)
      ENDDO
      WRITE(ICHAN,*)'WEIGHTING FACTORS USING DISTANCE'
      WRITE(ICHAN,*)'BETWEEN MEANS CRITERION'
C
      WRITE(ICHAN,85)(QT(L),L=L1,L2)
 85   FORMAT(<I2>F10.5)
C -----------------------------------------------------------
      RETURN
      END
C ***********************************************************
              SUBROUTINE FEATURE(IB,ID,IFEAT)
C Subroutine to compute features.
      PARAMETER(NYY=256,NXX=256,MM=3,NG=255,LL=10,NZ=150)
      INTEGER*2 IMAGE(NYY,NXX),IB,ID,M,N,I,J,I1,J1
      INTEGER*2 IP1,IP2,IQ1,IQ2,IFEAT
      REAL BH,BM,BB,S(LL),SUMG3,Q1,Q2,DAT
      REAL KG(NYY,NXX),KM(NYY,NXX,MM),KB(NYY,NXX)
```

```
      REAL KC(NYY,NXX),KD(NYY,NXX),DNAT,DH,DM,DB
      REAL REF(NZ,MM),AY(MM),SUMG1,SUMG2
      REAL DNAT1,DNAT2,DNAT3,DNAT4,SUMG4,DG
      COMMON IMAGE,KM,REF,AY
C ------------------------------------------------------------
C If texture is to be used in segmentation,compute the
C textural features.
      IF(IFEAT.GT.1)THEN
       DO J=1,NXX
        DO I=1,NYY
         KC(I,J)=0.0
         KD(I,J)=0.0
         IP1=-ID
         IP2=ID
         IQ1=-ID
         IQ2=ID
         IF(I.LE.ID)IP1=0
          IF(I.GT.(NYY-ID))IP2=0
         IF(J.LE.ID)IQ1=0
         IF(J.GT.(NXX-ID))IQ2=0
          Q1=FLOAT(IMAGE(I,J))
          DO L=1,5
           S(L)=0.0
          ENDDO
          SUMG1=0.0
          SUMG2=0.0
          SUMG3=0.0
          SUMG4=0.0
          DNAT1=-1.0
          DNAT2=-1.0
          DNAT3=-1.0
C
          DO L=IQ1,IQ2
           N=ABS(L)
           DO K=IP1,IP2
            M=ABS(K)
            Q2=FLOAT(IMAGE(I+K,J+L))
            IF((M.LE.1).AND.(N.LE.1))THEN
             SUMG1=SUMG1+Q2
             DNAT1=DNAT1+1.0
            ENDIF
            IF((M.LE.2).AND.(N.LE.2))THEN
             SUMG2=SUMG2+Q2
             DNAT2=DNAT2+1.0
            ENDIF
            IF((M.LE.3).AND.(N.LE.3))THEN
             SUMG3=SUMG3+Q2
             DNAT3=DNAT3+1.0
            ENDIF
           ENDDO
          ENDDO
          SUMG1=(SUMG1-Q1)/DNAT1
          SUMG2=(SUMG2-Q1)/DNAT2
          SUMG3=(SUMG3-Q1)/DNAT3
```

```
       S(1)=S(1)+(SUMG1-Q1)
       S(2)=S(2)+(SUMG2-Q1)
       S(3)=S(3)+(SUMG3-Q1)
       DO L=1,ID
        KC(I,J)=KC(I,J)+ABS(S(L))
       ENDDO
C
       DO L=1,ID-1
        DO K=L+1,ID
         KD(I,J)=KD(I,J)+ABS(S(L)-S(K))
        ENDDO
       ENDDO
       KD(I,J)=2.0*KD(I,J)
      ENDDO
     ENDDO
C
     DO J1=1,NXX
      J=J1
      DO I1=1,NYY
       I=I1
       KM(I1,J1,1)=0.0
       KM(I1,J1,2)=0.0
       KM(I1,J1,3)=0.0
       IP1=-IB
       IP2=IB
       IQ1=-IB
       IQ2=IB
       IF(I.LE.IB)THEN
        IP1=IB+1
        IP2=IP1+IB+I
        I=0
       ENDIF
       IF(I.GT.(NYY-IB))THEN
        IP2=NYY-IB
        IP1=IP2-(IB+(NYY-I))
        I=0
       ENDIF
       IF(J.LE.IB)THEN
        IQ1=IB+1
        IQ2=IQ1+IB+J
        J=0
       ENDIF
       IF(J.GT.(NXX-IB))THEN
        IQ2=NXX-IB
        IQ1=IQ2-(IB+(NXX-J))
        J=0
       ENDIF
       DNAT=0.0
       DM=0.0
       DB=0.0
       DG=0.0
       DO L=IQ1,IQ2
        DO K=IP1,IP2
         DNAT=DNAT+1.0
         DM=DM+KC(I+K,J+L)
```

```
          DB=DB+KD(I+K,J+L)
          IF(IFEAT.EQ.3)THEN
           DG=DG+IMAGE(I+K,J+L)
          ENDIF
         ENDDO
        ENDDO
        KM(I1,J1,1)=DM/DNAT
        KM(I1,J1,2)=DB/DNAT
        KM(I1,J1,3)=DG/DNAT
       ENDDO
      ENDDO
     ENDIF
C ---------------------------------------------------------
     IF(IFEAT.EQ.1)THEN
      IF(IB.NE.0)THEN
      DO J1=1,NXX
       J=J1
       DO I1=1,NYY
        I=I1
        KM(I1,J1,1)=0.0
        KM(I1,J1,2)=0.0
        KM(I1,J1,3)=0.0
        IP1=-IB
        IP2=IB
        IQ1=-IB
        IQ2=IB
        IF(I.LE.IB)THEN
        IP1=IB+1
        IP2=IP1+IB+I
        I=0
        ENDIF
        IF(I.GT.(NYY-IB))THEN
        IP2=NYY-IB
        IP1=IP2-(IB+(NYY-I))
        I=0
        ENDIF
        IF(J.LE.IB)THEN
        IQ1=IB+1
        IQ2=IQ1+IB+J
        J=0
        ENDIF
        IF(J.GT.(NXX-IB))THEN
        IQ2=NXX-IB
        IQ1=IQ2-(IB+(NXX-J))
        J=0
        ENDIF
        DNAT=0.0
        DG=0.0
        DO L=IQ1,IQ2
         DO K=IP1,IP2
          DNAT=DNAT+1.0
          DG=DG+FLOAT(IMAGE(I+K,J+L))
         ENDDO
        ENDDO
        KM(I1,J1,3)=DG/DNAT
```

```
           ENDDO
          ENDDO
         ELSE
          DO J=1,NXX
           DO I=1,NYY
            KM(I,J,3)=FLOAT(IMAGE(I,J))
           ENDDO
          ENDDO
         ENDIF
        ENDIF
C
        RETURN
        END
```

```
                PROGRAM SEG2
C -----------------------------------------------------------------
C  Program for segmentation ALGORITHM II for  the
C  segmentation of a black-and-white/monochrome image.
C
C  INPUT    :  ANY BLACK-AND-WHITE OR MONOCHROME IMAGE
C
C  OUTPUT   :  SEGMENTED VERSION OF THE INPUT IMAGE
C
C  COMMENTS :  Depending on the image, the user may segment
C           :  on the basis of texture, or brightness or on
C           :  the basis of both.
C -----------------------------------------------------------------
      PARAMETER(NYY=256,NXX=256,MM=3,NZ=150)
      INTEGER*2 IMAGE(NYY,NXX),IC,ID,NT,IB,IFEAT
      INTEGER*2 NY,NX,ISL,ISS,NL,NS,ICHAN,IA
      REAL REF(NZ,MM),AY(MM),KM(NYY,NXX,MM)
      REAL QT(MM),THR
      INTEGER*4 STATUSFLAG,LINENUMBER
      COMMON IMAGE,KM,REF,AY
C
C Read in the image.
      CALL VICINIT('SEG2')
      CALL OPENV(STATUSFLAG,2,0,0,0,0)
      DO LINENUMBER=1,NYY
       CALL READ(STATUSFLAG,2,0,1,0,NXX,IMAGE(1,LINENUMBER),0)
      ENDDO
c
C Supply the required segmentation parameters.
C -------------------------------------------------------------------
      WRITE(6,*)'WHAT TYPE OF FEATURES ARE TO BE USED FOR
      WRITE(6,*)'SEGMENTATION, BRIGHTNESS, TEXTURE OR A '
      WRITE(6,*)'COMBINATION OF BOTH?. IF ONLY BRIGHTNESS'
      WRITE(6,*)'INPUT 1, IF ONLY TEXTURE INPUT 2 OR IF'
      WRITE(6,*)'A COMBINATION OF BOTH INPUT 3'
      READ(5,*)IFEAT
C
      IF(IFEAT.GT.1)THEN
      ID=3
      WRITE(6,*)'INPUT THE DISTANCE FOR SPECIFYING THE'
      WRITE(6,*)'CHARACTERIZATION WINDOW SIZE FOR THE'
      WRITE(6,*)'COMPUTATION OF TEXTURAL FEATURES; IB'
      READ(5,*)IB
      ENDIF
      IF(IFEAT.EQ.1)THEN
       ID=1
       WRITE(6,*)'NOW, THAT YOU ARE USING ONLY BRIGHTNESS'
       WRITE(6,*)'DO YOU WANT TO USE THE GRAY LEVELS OF THE'
       WRITE(6,*)'PIXELS DIRECTLY OR THE AVERAGE GRAY LEVEL'
       WRITE(6,*)'IN SMALL WINDOWS CENTERED ON PIXELS?.'
       WRITE(6,*)'IF AVERAGE INPUT 1, OTHERWISE INPUT O'
       READ(5,*)IA
       IF(IA.EQ.1)THEN
        WRITE(6,*)'THEN, INPUT THE DISTANCE FOR SPECIFYING'
        WRITE(6,*)'THE SIZE OF THIS WINDOW; IB'
```

```
          READ(5,*)IB
         ELSE
          IB=0
         ENDIF
        ENDIF
C
        WRITE(6,*)'INPUT THE DISTANCE FOR SPECIFYING'
        WRITE(6,*)'DIMENSION OF LARGEST SEARCH BLOCKS; NX.'
        WRITE(6,*)'NOTE DIMENSION=NX*NX '
        READ(5,*)NX
C
        WRITE(6,*)'INPUT THE NUMBER OF CATEGORIES; NT'
        READ(5,*)NT
        WRITE(6,*)'INPUT THE VALUE OF THE UNIFORMITY'
        WRITE(6,*)'CRITERION; THR, A REAL NUMBER'
        READ(5,*)THR
C ----------------------------------------------------------
        WRITE(6,*)'INPUT THE CHANNEL NUMBER FOR WRITING FILE'
        READ(5,*)ICHAN
C
        CALL FEATURE(IB,ID,IFEAT)
        CALL CLAVECT(NX,NT,IFEAT,QT,ICHAN,THR)
        CALL ASSIGN(NT,IFEAT,QT)
C
C Write out the segmented image
        CALL OPENV(STATUSFLAG,1,1,0,0,0)
        CALL ADJUST(1,NYY,NXX)
        DO I=1,NYY
         CALL WRITE(STATUSFLAG,1,0,1,0,NXX,IMAGE(1,I),0)
        ENDDO
        CALL RELAB2(1,NYY,NXX)
        END
C **********************************************************
                SUBROUTINE FEATURE(IB,ID,IFEAT)
C Subroutine to compute features.
        PARAMETER(NYY=256,NXX=256,MM=3,NG=255,LL=10,NZ=150)
        INTEGER*2 IMAGE(NYY,NXX),IB,ID,M,N,I,J,I1,J1
        INTEGER*2 IP1,IP2,IQ1,IQ2,IFEAT
        REAL BH,BM,BB,S(LL),SUMG3,Q1,Q2,DAT
        REAL KG(NYY,NXX),KM(NYY,NXX,MM),KB(NYY,NXX)
        REAL KC(NYY,NXX),KD(NYY,NXX),DNAT,DH,DM,DB
        REAL REF(NZ,MM),AY(MM),SUMG1,SUMG2
        REAL DNAT1,DNAT2,DNAT3,DNAT4,SUMG4,DG
        COMMON IMAGE,KM,REF,AY
C -------------------------------------------------------
C If texture is to be used for segmentation, compute the
C textural features.
        IF(IFEAT.GT.1)THEN
         DO J=1,NXX
          DO I=1,NYY
           KC(I,J)=0.0
           KD(I,J)=0.0
           IP1=-ID
           IP2=ID
           IQ1=-ID
```

```
      IQ2=ID
      IF(I.LE.ID)IP1=0
      IF(I.GT.(NYY-ID))IP2=0
      IF(J.LE.ID)IQ1=0
      IF(J.GT.(NXX-ID))IQ2=0
      Q1=FLOAT(IMAGE(I,J))
      DO L=1,5
       S(L)=0.0
      ENDDO
      SUMG1=0.0
      SUMG2=0.0
      SUMG3=0.0
      SUMG4=0.0
      DNAT1=-1.0
      DNAT2=-1.0
      DNAT3=-1.0
C
      DO L=IQ1,IQ2
       N=ABS(L)
       DO K=IP1,IP2
        M=ABS(K)
        Q2=FLOAT(IMAGE(I+K,J+L))
        IF((M.LE.1).AND.(N.LE.1))THEN
         SUMG1=SUMG1+Q2
         DNAT1=DNAT1+1.0
        ENDIF
        IF((M.LE.2).AND.(N.LE.2))THEN
         SUMG2=SUMG2+Q2
         DNAT2=DNAT2+1.0
        ENDIF
        IF((M.LE.3).AND.(N.LE.3))THEN
         SUMG3=SUMG3+Q2
         DNAT3=DNAT3+1.0
        ENDIF
       ENDDO
      ENDDO
      SUMG1=(SUMG1-Q1)/DNAT1
      SUMG2=(SUMG2-Q1)/DNAT2
      SUMG3=(SUMG3-Q1)/DNAT3
      SUMG4=(SUMG4-Q1)/DNAT4
      S(1)=S(1)+(SUMG1-Q1)
      S(2)=S(2)+(SUMG2-Q1)
      S(3)=S(3)+(SUMG3-Q1)
      DO L=1,ID
       KC(I,J)=KC(I,J)+ABS(S(L))
      ENDDO
C
      DO L=1,ID-1
       DO K=L+1,ID
        KD(I,J)=KD(I,J)+ABS(S(L)-S(K))
       ENDDO
      ENDDO
      KD(I,J)=2.0*KD(I,J)
     ENDDO
    ENDDO
```

```
C
       DO J1=1,NXX
        J=J1
        DO I1=1,NYY
         I=I1
         KM(I1,J1,1)=0.0
         KM(I1,J1,2)=0.0
         KM(I1,J1,3)=0.0
         IP1=-IB
         IP2=IB
         IQ1=-IB
         IQ2=IB
         IF(I.LE.IB)THEN
          IP1=IB+1
          IP2=IP1+IB+I
          I=0
         ENDIF
         IF(I.GT.(NYY-IB))THEN
          IP2=NYY-IB
          IP1=IP2-(IB+(NYY-I))
          I=0
         ENDIF
         IF(J.LE.IB)THEN
          IQ1=IB+1
          IQ2=IQ1+IB+J
          J=0
         ENDIF
         IF(J.GT.(NXX-IB))THEN
          IQ2=NXX-IB
          IQ1=IQ2-(IB+(NXX-J))
          J=0
         ENDIF
         DNAT=0.0
         DM=0.0
         DB=0.0
         DG=0.0
         DO L=IQ1,IQ2
          DO K=IP1,IP2
           DNAT=DNAT+1.0
           DM=DM+KC(I+K,J+L)
           DB=DB+KD(I+K,J+L)
           IF(IFEAT.EQ.3)THEN
            DG=DG+IMAGE(I+K,J+L)
           ENDIF
          ENDDO
         ENDDO
         KM(I1,J1,1)=DM/DNAT
         KM(I1,J1,2)=DB/DNAT
         KM(I1,J1,3)=DG/DNAT
         ENDDO
        ENDDO
       ENDIF
C
       IF(IFEAT.EQ.1)THEN
        IF(IB.NE.0)THEN
```

```
            DO J1=1,NXX
             J=J1
             DO I1=1,NYY
              I=I1
              KM(I1,J1,1)=0.0
              KM(I1,J1,2)=0.0
              KM(I1,J1,3)=0.0
              IP1=-IB
              IP2=IB
              IQ1=-IB
              IQ2=IB
              IF(I.LE.IB)THEN
              IP1=IB+1
              IP2=IP1+IB+I
              I=0
              ENDIF
              IF(I.GT.(NYY-IB))THEN
               IP2=NYY-IB
               IP1=IP2-(IB+(NYY-I))
               I=0
              ENDIF
              IF(J.LE.IB)THEN
               IQ1=IB+1
               IQ2=IQ1+IB+J
               J=0
              ENDIF
              IF(J.GT.(NXX-IB))THEN
               IQ2=NXX-IB
               IQ1=IQ2-(IB+(NXX-J))
               J=0
              ENDIF
              DNAT=0.0
              DG=0.0
              DO L=IQ1,IQ2
               DO K=IP1,IP2
                DNAT=DNAT+1.0
                DG=DG+FLOAT(IMAGE(I+K,J+L))
               ENDDO
              ENDDO
              KM(I1,J1,3)=DG/DNAT
             ENDDO
            ENDDO
           ELSE
            DO J=1,NXX
             DO I=1,NYY
              KM(I,J,3)=FLOAT(IMAGE(I,J))
             ENDDO
            ENDDO
           ENDIF
          ENDIF
C
          RETURN
          END
C ***************************************************
            SUBROUTINE ASSIGN(NT,IFEAT,QT)
```

```
C Subroutine to classify pixels.
      PARAMETER(NYY=256,NXX=256,MM=3,NZ=150)
      INTEGER*2 IMAGE(NYY,NXX),IMA(NZ),NT,KK
      INTEGER*2 PEF(NZ),GL,GINC,IFEAT,N1,N2
      REAL KM(NYY,NXX,MM),QT(MM)
      REAL REF(NZ,MM),AY(MM),KR1,KR2,PP,QQ
      REAL RK(NZ),RK2,D(MM),SUM,P,Q,PAX
      COMMON IMAGE,KM,REF,AY
C
      IF(IFEAT.EQ.1)THEN
       N1=3
       N2=3
      ENDIF
      IF(IFEAT.EQ.2)THEN
       N1=1
       N2=2
      ENDIF
      IF(IFEAT.EQ.3)THEN
       N1=1
       N2=3
      ENDIF
      GL=40
      GINC=15
C
      DO L=1,MM
       IF(AY(L).EQ.0.0)AY(L)=1.0
      ENDDO
C
      DO K=1,NT
       IF(K.EQ.1)THEN
        IMA(K)=GL
       ELSE
        IMA(K)=IMA(K-1)+GINC
       ENDIF
      ENDDO
      DO J=1,NXX
       DO I=1,NYY
        DO L=N1,N2
         D(L)=KM(I,J,L)/AY(L)
        ENDDO
        DO K=1,NT
         RK(K)=0.0
         DO L=N1,N2
          PAX=(D(L)-REF(K,L))**2
          RK(K)=RK(K)+(QT(L)*PAX)
         ENDDO
        ENDDO
        P=RK(1)
        KK=1
        DO K=1,NT
         IF(P.GT.RK(K))THEN
          P=RK(K)
          KK=K
         ENDIF
        ENDDO
```

```
          IMAGE(I,J)=IMA(KK)
         ENDDO
        ENDDO
        RETURN
        END
C *******************************************************
         SUBROUTINE CLAVECT(NX,NT,IFEAT,QT,ICHAN,THR)
C Subroutine to determine mean feature vectors for
C classes or categories and also performs normalisation
C of features.
       PARAMETER(NYY=256,NXX=256,MM=3,NZ=150,NB=64,NL=4096)
       INTEGER*2 IMAGE(NYY,NXX),IC,K,IFEAT,NUT
       INTEGER*2 L1,L2,NT,JQ,ID,I1,I2,J1,J2,I,KOUNT
       INTEGER*2 NR,NC,NX,M,N,NN1,NN2,ICHAN,IE,J
       INTEGER*2 IFAT,MX1,MX2,MX3,ITEST,ED2(2,2)
       REAL IMAG3(NB,NB,MM),ICAN(NL),UM
       REAL KM(NYY,NXX,MM),REF(NZ,MM),BG,QT(MM),ZTT
       REAL TN1,TN2,TM1,TM2,TN,DF(MM),AZ(MM)
       REAL FF(NL,MM),ZA,ZC,DNAT,BF(MM),THR,DET,UB
       REAL IMAG1(NB,NB,MM),IMAG2(NB,NB,MM),UG,AY(MM)
       COMMON IMAGE,KM,REF,AY
C
       MX1=NX
       NR=NYY/NX
       NC=NXX/NX
       IF(IFEAT.EQ.1)THEN
        L1=3
        L2=3
       ENDIF
       IF(IFEAT.EQ.2)THEN
        L1=1
        L2=2
       ENDIF
       IF(IFEAT.EQ.3)THEN
        L1=1
        L2=3
       ENDIF
       IE=0
       IFAT=(L2-L1)+1
C %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
C Divide the image into blocks of size NX*NX and test
C for uniformity of blocks.compute the average
C feature values of blocks considered uniform.
C
       DO JJ=1,NC
        DO II=1,NR
         DO N=1,MX1
          J=(JJ-1)*MX1+N
          DO M=1,MX1
           I=(II-1)*MX1+M
           DO L=L1,L2
            IMAG1(M,N,L)=KM(I,J,L)
           ENDDO
          ENDDO
         ENDDO
```

```
      CALL UNIFORM(IMAG1,MX1,ITEST,DF,L1,L2,THR)
      IF(ITEST.EQ.1)THEN
       IE=IE+1
       ICAN(IE)=FLOAT(MX1**2)
       DO L=L1,L2
        FF(IE,L)=DF(L)
       ENDDO
       GO TO 100
      ENDIF
C ---------------------------------------------------
C If block of size NX*NX is not uniform, split it into
C four subblocks of size MX2*MX2 (where MX2=NX/2).Test
C for uniformity of each subblock and compute the
C average feature values for those considered uniform.
      MX2=MX1/2
      KOUNT=0
      DO J1=1,2
       DO I1=1,2
        ED2(I1,J1)=0
        DO N=1,MX2
         J=(J1-1)*MX2+N
         DO M=1,MX2
          I=(I1-1)*MX2+M
          DO L=L1,L2
           IMAG2(M,N,L)=IMAG1(I,J,L)
          ENDDO
         ENDDO
        ENDDO
        CALL UNIFORM(IMAG2,MX2,ITEST,DF,L1,L2,THR)
        IF(ITEST.EQ.1)THEN
         IE=IE+1
         ICAN(IE)=FLOAT(MX2**2)
         DO L=L1,L2
          FF(IE,L)=DF(L)
         ENDDO
         ED2(I1,J1)=1
         KOUNT=KOUNT+1
        ENDIF
       ENDDO
      ENDDO
      IF(KOUNT.EQ.4)GO TO 100
C ---------------------------------------------------
C If any of the subblock of size MX2*MX2 is not uniform,
C split the subblock further into four portions, each of
C size MX3*MX3 (where MX3=MX2/2). Test for uniformity
C of each portion and compute the average feature
C values for those considered uniform.
      MX3=MX2/2
      DO J1=1,2
       DO I1=1,2
        IF(ED2(I1,J1).NE.1)THEN
         DO N=1,MX2
          J=(J1-1)*MX2+N
          DO M=1,MX2
           I=(I1-1)*MX2+M
```

```
              DO L=L1,L2
               IMAG2(M,N,L)=IMAG1(I,J,L)
              ENDDO
             ENDDO
            ENDDO
            DO J2=1,2
             DO I2=1,2
              DO N=1,MX3
               J=(J2-1)*MX3+N
               DO M=1,MX3
                I=(I2-1)*MX3+M
                DO L=L1,L2
                 IMAG3(M,N,L)=IMAG2(I,J,L)
                ENDDO
               ENDDO
              ENDDO
              CALL UNIFORM(IMAG3,MX3,ITEST,DF,L1,L2,THR)
              IF(ITEST.EQ.1)THEN
               IE=IE+1
               ICAN(IE)=FLOAT(MX3**2)
               DO L=L1,L2
                FF(IE,L)=DF(L)
               ENDDO
              ENDIF
             ENDDO
            ENDDO
           ENDIF
          ENDDO
         ENDDO
C -----------------------------------------------------------
 100     CONTINUE
        ENDDO
       ENDDO
C %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
C Cluster the mean vectors agglomeratively.
       WRITE(ICHAN,*)'NO.OF VECTORS CLUSTERED =',IE
 202   IC=IE
 203   ZTT=10000000.0
       DO L=L1,L2
        AZ(L)=0.0
        DO K=1,IE
         AZ(L)=AZ(L)+FF(K,L)
        ENDDO
       ENDDO
        DO I=1,IE
        IF(ICAN(I).NE.0.0)THEN
         DO J=I+1,IE
          IF(ICAN(J).NE.0.0)THEN
           DNAT=0.0
           DO L=L1,L2
            DNAT=DNAT+((FF(I,L)-FF(J,L))/AZ(L))**2
           ENDDO
           IF(ZTT.GT.DNAT)THEN
            ZTT=DNAT
            NN1=I
```

```
            NN2=J
           ENDIF
          ENDIF
         ENDDO
        ENDIF
       ENDDO
       TM1=ICAN(NN1)
       TM2=ICAN(NN2)
       TN=TM1+TM2
       DO L=L1,L2
        BF(L)=(TM1*FF(NN1,L)+TM2*FF(NN2,L))/TN
       ENDDO
       DO L=L1,L2
        FF(NN1,L)=BF(L)
        FF(NN2,L)=0.0
       ENDDO
       ICAN(NN1)=ICAN(NN1)+ICAN(NN2)
       ICAN(NN2)=0.0
       IC=IC-1
       IF(IC.GT.NT)GO TO 203
C %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
       NUT=0
       DO I=1,IE
        IF(ICAN(I).NE.0.0)THEN
         NUT=NUT+1
         DO L=L1,L2
          REF(NUT,L)=FF(I,L)
         ENDDO
        ENDIF
       ENDDO
       WRITE(ICHAN,*)'THESE ARE AVERAGE VALUES OF FEATURES'
       WRITE(ICHAN,*)'FOR CATEGORIES'
       WRITE(ICHAN,70)((REF(I,J),J=L1,L2),I=1,NUT)
 70    FORMAT(<IFAT>F13.6)
C
C ---------------------------------------------------------
C Determine the feature normalizing factors and normalize
C the mean feature values.
        DO L=L1,L2
        AY(L)=0.0
        DO K=1,NT
         AY(L)=AY(L)+REF(K,L)
        ENDDO
        IF(AY(L).EQ.0.0)AY(L)=1.0
       ENDDO
       DO L=L1,L2
        DO K=1,NT
         REF(K,L)=REF(K,L)/AY(L)
        ENDDO
       ENDDO
C Determine the feature weighting factors using distance
C between means criterion.
       DO L=L1,L2
        QT(L)=0.0
        DO I=1,NT-1
```

```
           DO J=I+1,NT
            ZA=ABS(REF(I,L)-REF(J,L))
            ZC=REF(I,L)+REF(J,L)
            IF(ZC.EQ.0.0)ZC=1.0
            QT(L)=QT(L)+(ZA/ZC)
           ENDDO
          ENDDO
         ENDDO
         ZC=0.0
         DO L=L1,L2
          IF(ZC.LT.QT(L))ZC=QT(L)
         ENDDO
         DO L=L1,L2
          IF(QT(L).NE.0.0)QT(L)=ZC/QT(L)
         ENDDO
         WRITE(ICHAN,*)'WEIGHTING FACTORS USING DISTANCE'
         WRITE(ICHAN,*)'BETWEEN MEANS CRITERION'
C
         WRITE(ICHAN,85)(QT(L),L=L1,L2)
 85      FORMAT(<IFAT>F10.5)
C -----------------------------------------------------
         RETURN
         END
C ******************************************************
          SUBROUTINE UNIFORM(IMAG,MX,ITEST,DF,L1,L2,THR)
C Subroutine to determine uniformity of neighbourhoods.
         PARAMETER(NB=64,MM=3)
         INTEGER*2 ITEST,MX,LX,L1,L2
         REAL Z1,Z2,THR,THD,BM,EF(MM),EF1(MM)
         REAL IMAG(NB,NB,MM),DF(MM),EF2(MM),BB
         REAL DAT,DET,EF3(MM),EF4(MM),DB
C
         LX=MX/2
         DAT=FLOAT(MX**2)
         DET=FLOAT(LX**2)
         DB=FLOAT(L2-L1)+1.0
         ITEST=0
C
         DO L=L1,L2
          EF(L)=0.0
          EF1(L)=0.0
          EF2(L)=0.0
          EF3(L)=0.0
          EF4(L)=0.0
          DF(L)=0.0
         ENDDO
         DO N=1,MX
          DO M=1,MX
           DO L=L1,L2
            EF(L)=EF(L)+IMAG(M,N,L)
            IF((M.LE.LX).AND.(N.LE.LX))THEN
             EF1(L)=EF1(L)+IMAG(M,N,L)
            ENDIF
            IF((M.LE.LX).AND.(N.GT.LX))THEN
             EF2(L)=EF2(L)+IMAG(M,N,L)
```

```
        ENDIF
        IF((M.GT.LX).AND.(N.LE.LX))THEN
         EF3(L)=EF3(L)+IMAG(M,N,L)
        ENDIF
        IF((M.GT.LX).AND.(N.GT.LX))THEN
         EF4(L)=EF4(L)+IMAG(M,N,L)
        ENDIF
       ENDDO
      ENDDO
     ENDDO
     DO L=L1,L2
      EF(L)=EF(L)/DAT
      EF1(L)=EF1(L)/DET
      EF2(L)=EF2(L)/DET
      EF3(L)=EF3(L)/DET
      EF4(L)=EF4(L)/DET
     ENDDO
C
     BM=0.0
     DO L=L1,L2
      BB=0.0
      IF(EF(L).GE.EF1(L))THEN
       Z1=EF(L)
       Z2=EF1(L)
      ELSE
       Z1=EF1(L)
       Z2=EF(L)
      ENDIF
      THD=THR*Z1
      IF(Z2.GE.THD)BB=BB+1.0
      IF(EF(L).GE.EF2(L))THEN
       Z1=EF(L)
       Z2=EF2(L)
      ELSE
       Z1=EF2(L)
       Z2=EF(L)
      ENDIF
      THD=THR*Z1
      IF(Z2.GE.THD)BB=BB+1.0
      IF(EF(L).GE.EF3(L))THEN
       Z1=EF(L)
       Z2=EF3(L)
      ELSE
       Z1=EF3(L)
       Z2=EF(L)
      ENDIF
      THD=THR*Z1
      IF(Z2.GE.THD)BB=BB+1.0
      IF(EF(L).GE.EF4(L))THEN
       Z1=EF(L)
       Z2=EF4(L)
      ELSE
       Z1=EF4(L)
       Z2=EF(L)
      ENDIF
```

```
       THD=THR*Z1
       IF(Z2.GE.THD)BB=BB+1.0
       IF(BB.EQ.4.0)BM=BM+1.0
      ENDDO
C
      IF(BM.EQ.DB)THEN
       ITEST=1
       DO L=L1,L2
        DF(L)=EF(L)
       ENDDO
      ENDIF
C
      RETURN
      END
```

REFERENCES

[1] **Abdou, I.E.,** and Pratt, W.K. (1979): "Quantitative Design and Evaluation of Enhancement/Thresholding Edge Detectors", Proc. IEEE, Vol. 67, No. 5, pp. 753-763.

[2] **Ball, G.H.** and Hall, D.J. (1965): "ISODATA, an Iterative Method of Multivariate Analysis and Pattern Classification", Proc. IFIPS Congress.

[3] **Bezdek, J.C.** (1980): "A Convergence Theorem for the Fuzzy ISODATA Algorithms", IEEE Trans. Pattern Anal. Mach. Intell., Vol. PAMI-2, No. 1, pp. 1-8.

[4] **Brice C.** and Fennema, C. (1970): "Scene Analysis Using Regions", Artificial Intelligence-1, pp. 205-226.

[5] **Brodatz, P.** (1966): "Textures: A Photographic Album for Artists and Designers", Dover, New York

[6] **Bryant, J.** (1979): "On the Clustering of Multidimensional Pictorial Data", Pattern Recognitiion, Vol. 11, pp. 115-125.

[7] **Cannon, R.L.,** Dave, J.V., Bezdek, J.C. et al (1986): "Segmentation of a Thematic Mapper Image Using the Fuzzy c-Means Clustering Algorithm", IEEE Trans. Geosc. and Remote Sen., Vol. GE-24, No. 3, pp. 400-408.

[8] **Carlucci, L.** (1972): "A Formal System for Texture Languages", Pattern Recognition, Vol. 4, pp. 53-72.

[9] **Chen, P.C.** and Pavlidis, T. (1980): "Image Segmentation as an Estimation Problem", Computer Graphics and Image Processing-12, pp. 153-172.

[10] **Chen, P.C.** and Pavlidis, T. (1983): "Segmentation by Texture Using Correlation", IEEE Trans. Pattern Anal. Mach. Intell., Vol. PAMI-5, pp. 64-69.

[11] **Chow, C.K.** and Kaneko, T. (1972): "Boundary Detection of Radiographic Images by a Thresholding Method", in Frontiers of Pattern Recognition, ed. by S. Watanabe, Academic Press, New York, pp. 61-82.

[12] **Coleman, G.B.** and Andrews, H.C. (1979): "Image Segmentation by Clustering", Proc. IEEE, Vol. 67, pp. 773-785.

[13] **Connors, R.W.** and Harlow, C.A. (1980): "A Theoretical Comparison of Texture Algorithms", IEEE Trans. Pattern Anal. Mach. Intell., Vol. PAMI-2, pp. 204-222.

[14] **Cooper, P.W.** (1969): "Fundamentals of Statistical Decisions and Learning", in Automatic Interpretation and Classification of Images - A NATO Advanced Study Institute, ed. by A. Grasselli, pp. 97-129.

[15] **Crane, R.B.,** Malina, N.A. and Richardson, W. (1972): "Suitability of the Normal Density Assumption for Processing Multispectral Scene Data", IEEE Trans. Geoscience Electronics, Vol. GE-10, pp. 158-167.

[16] **Davies, L.S.,** Clearman, M. and Aggarwal, J.K. (1981): "An Empirical Evaluation of Generalized Cooccurrence Matrices", IEEE Trans. Pattern Anal. Mach. Intell., Vol. PAMI-3, No. 2, pp. 214-221.

[17] **Deguchi, K.** and Morishita, I. (1978): "Texture Characterization and Texture-Based Image Partitioning Using 2-D Linear Estimation Techniques", IEEE Trans. Computers, Vol. C-27, pp. 739-745.

[18] **Duda, R.O.** and Hart, P.E. (1973): "Pattern Classification and Scene Analysis", A Wiley Interscience Publication, New York.

[19] **Ekhundh, J.O.** (1979): "On the Use of Fourier Phase Features for Texture Discriminations", Computer Graphics and Image Processing, Vol. 9, pp. 199-201.

[20] **Fu, K.S.,** Landgrebe, D.A. and Philips, T.L. (1969): "Information Processing of Remotely Sensed Agricultural Data", Proc. IEEE, Vol. 57, No. 4, pp. 639-653.

[21] **Fu, K.S.** and Mui, J.K. (1981): "A Survey on Image Segmentation", Pattern Recognition, Vol. 13, pp. 3-16.

[22] **Fukunaga, K.** (1972): "Introduction to Statistical Pattern Recognition", Academic Press, New York, pp. 50-119.

[23] **Galloway, M.** (1974): "Texture Analysis Using Gray Level Run Lengths", Computer Graphics and Image Processing, Vol 4, pp. 172-199.

[24] **Goldberg, M.** and Shilien, S. (1978): "A Clustering Scheme for Multispectral Images", IEEE Trans. Syst., Man, Cybern., Vol. SMC-8, pp. 80-92.

[25] **Gonzalez, R.C.** and Wintz, P. (1977): "Digital Image Processing", Addison-Wesley Publishing Co. Inc.

[26] **Gool, L.V.,** Dewaele, P. and Oosterlinck, A. (1983): "SURVEY: Texture Analysis Anno", Computer Vision, Graphics and Image Processing, Vol. 9, pp. 336-357.

[27] **Gramenopoulos, N.** (1973): "Terrain Type Recognition Using ERTS-1 MSS Images", in Record Symposium, Significant Results Obtained from the Earth Resources Technol. Satellite, NASA SP-327, pp. 1229-1241.

[28] **Guildford, J.P.** (1954): "Psychometric Methods", McGraw-Hill Book Company Inc., New York, Chapter 8.

[29] **Haralick, R.M.** (1980): "Edge and Region Analysis for Digital Image Data", Computer Graphics and Image Processing, Vol. 12, pp. 60-73.

[30] ——————— (1979): "Statistical and Structural Approaches to Texture", Proc. IEEE, Vol. 67, pp. 786-804.

[31] **Haralick, R.M.** and Dinstein, I. (1975): "A Spatial Clustering Procedure for Multi-Image Data", IEEE Trans. Circuits and Systems, Vol. CAS-22, pp. 440-450.

[32] **Haralick, R.M.**, Shanmugam, K. and Dinstein, I. (1973): "Textural Features for Image Classification", IEEE Trans. Syst., Man, Cybern., Vol. SMC-3, No. 6, pp. 610-621.

[33] **Haralick, R.M.** and Shapiro, L.G. (1985): "SURVEY: Image Segmentation Techniques", Computer Vision, Graphics and Image Processing, Vol. 29, pp. 100-132.

[34] **Horowitz, S.L.** and Pavlidis, T. (1976): "Picture Segmentation by a Tree Transversal Algorithm", J. Assoc. Comput. Mach., Vol. 23, pp. 368-388.

[35] **Jarvis, R.A.** and Patrick, E.A. (1973): "Clustering Using Similarity Measure Based on Shared Near Neighbours, IEEE Trans. Comput., Vol. C-22, pp. 1025-1034.

[36] **Jernigan, M.E.** and D'Astous, F.D. (1984): "Entropy-Based Texture Analysis in the Spatial Domain", IEEE Trans. Pattern Anal. Mach. Intell., Vol. PAMI-6, pp. 237-243.

[37] **Julesz, B.** (1962): "Texture and Visual Perception", IRE Trans. Info. Theory, Vol. IT-8, No. 2, pp. 84-92.

[38] ————— (1965): "Texture and Visual Perception", Scientific American, Vol. 212, pp. 38-54.

[39] **Julesz, B.**, Gilbert, E.N., Shepp, L.A. et al (1973): "Inability of Humans to Discriminate between Visual Textures that Agree in Second-Order Statistics - Revisited", Perception, Vol. 2, pp. 391-405.

[40] **Kashyap, R.L.** and Khotanzad, A. (1986): "A Model-Based Method for Rotation Invariant Texture Classification", IEEE Trans. Pattern Anal. Mach. Intell., Vol. PAMI-8, pp. 472-482.

[41] **Kirsch, R.** (1971): "Computer Determination of the Constituent Structure of Biological Images", Comput. Biomed. Res., Vol. 4, No. 3, pp. 315-328.

[42] **Kohler, R.** (1981): "A Segmentation System Based on Thresholding", Computer Graphics and Image Processing-15, pp. 319-338.

[43] **Levine, M.D.** and Nazif, A.M. (1985): "Dynamic Measurement of Computer Generated Image Segmentations", IEEE Trans. Pattern Anal. Mach. Intell., Vol. PAMI-7, No. 2, pp. 155-164.

[44] **Levine, M.D.** and Shasheen, S. (1981): "A Modular Computer Vision System for Picture Segmentation and Interpretation", IEEE Trans. Pattern Anal. Mach. Intell., Vol. PAMI-3, pp. 540-556.

[45] **Lu, S.Y.** and Fu, K.S. (1978): "A Syntactic Approach to Texture Analysis", Computer Graphics and Image Processing, Vol. 7, pp. 303-330.

[46] **McCormick, B.H.** and Jayaramurthy, S.N. (1974): "Time Series Model for Texture Synthesis", J. Computing and Information Science, Vol. 3, No. 4, pp. 329-343.

[47] **Milgram, D.L.** and Herman, M. (1979): "Clustering Edge Values for Threshold Selection", Computer Graphics and Image Processing, Vol. 10, pp. 272-280.

[48] **Mitchell, O.,** Myer, C.R. and Boyne, W. (1977): "A Max-Min Measure for Image Texture Analysis", IEEE Trans. Comput., Vol. C-26, pp. 408-414.

[49] **Mitrakos, D.K.** and Constantinides, A.G. (1982): "Composite Source Coding Techniques for Image Bandwidth Compression", Proc. IEEE Inter. Conf. Acoustics, Speech and Signal Processing, Paris, Vol. 2, pp. 1223-1226.

[50] **Mitrakos, D.K.** and Constantinides A.G. (1984): "Graph-Theoretic Approach to Composite-Source-Model Estimation for Image Coding", IEE Proc., Vol. 131, Part F, pp. 71-79.

[51] **Modestino, J.W.,** Fries, R.W. and Vickers, A.L. (1981): "Texture Discrimination Based on an Assumed Stochastic Texture Model", IEEE Trans. Pattern Anal. Mach. Intell., Vol. PAMI-3, pp. 557-580.

[52] **Morris, O.J.** and Constantinides, A.G. (1986): "Graph Theory for Image Analysis: an Approach Based on the Shortest Spanning Tree", IEE Proc., Vol. 133, Part F, No. 2, pp. 146-152.

[53] **Muerle, J.** and Allen, D. (1986): "Experimental Evaluation of Techniques for Automatic Segmentation of Objects in a Complex Scene", in Pictorial Pattern Recognition, ed. by G. Cheng, Thompson, Washington D.C., pp. 3-13.

[54] **Nagy, G.** and Tolaba, J. (1972): "Nonsupervised Crop Classification through Airborne Multispectral Observations", IBM J. Res. Dev., Vol. 16, pp. 138-153.

[55] **Narenda, P.M.** and Goldberg, M. (1977): "A Non-Parametric Clustering Scheme for LANDSAT", Pattern Recognition, Vol. 9, pp. 207-215.

[56] **Narenda, P.M.** and Goldberg, M. (1980): "Image Segmentation with Directed Trees", IEEE Trans. Pattern Anal. Mach. Intell., Vol. PAMI-2, pp. 185-191.

[57] **Ohlander, R.,** Price, K. and Reddy, D.R. (1978): "Picture Segmentation Using a Recursive Region Splitting Method", Computer Graphics and Image Processing, Vol. 8, pp. 313-333.

[58] **Panda, D.P.** and Rosenfeld, A. (1978): "Image Segmentation by Pixel Classification in (Gray Level Edge Value) Space", IEEE Trans. Comput., Vol. C-27, pp. 875-879.

[59] **Pavlidis, T.** (1982): "Algorithms for Graphics and Image Processing", Springer-Verlag, New York.

[60] ———————— (1972): "Segmentation of Pictures and Maps through Functional Approximations", Computer Graphics and Image Processing, Vol. 1, pp. 360-372.

[61] **Pratt, W.K.** (1978): "Digital Image Processing", Wiley Interscience Publication, New York.

[62] **Pratt, W.K.,** Faugeras, O.D. and Gagalowicz, A. (1981): "Applications of Stochastic Texture Field Models to Image Processing", Proc. IEEE, Vol. 69, pp. 542-551.

[63] **Prewitt, J.M.S.** (1970): "Object Enhancement and Extraction", in Picture Processing and Psychopictorics, ed. by B.S. Lipkin and A. Rosenfeld, Academic Press, New York.

[64] **Riseman, E.M.** and Arbib, M.A. (1977): "SURVEY: Computational Techniques in the Visual Segmentation of Static Scenes", Computer Graphics and Image Processing, Vol. 6, pp. 221-276.

[65] **Rosenfeld, A.** and Davies, L.S. (1979): "Image Segmentation and Image Models", Proc. IEEE, Vol. 67, No. 5, pp. 764-772.

[66] **Rosenfeld, A.** and Kak, A.C. (1982): "Digital Picture Processing", Vol 2, Academic Press Inc.

[67] **Rosenfeld, A.** and Thurston, M. (1971): "Edge and Curve Detection for Visual Scene Analysis", IEEE Trans. Comput., Vol. C-20, pp. 562-569.

[68] **Serra, J.** and Verchery, G. (1973): "Mathematical Morphology Applied to Fibre Composite Materials", Film Science Tech., Vol. 6, pp. 141-158.

[69] **Siegel, S.** (1956): "Nonparametric Statistics for the Behavioural Sciences", McGraw-Hill, Chapter 9.

[70] **Spann, M.** and Wilson, R. (1975): "A Quad-Tree Approach to Image Segmentation which combines Statistical and Spatial Information", Pattern Recognition, Vol. 18, No. 3/4, pp. 257-269.

[71] **Suk, M.** and Song, O. (1984): "Curvilinear Feature Extraction Using Minimum Spanning Trees", Computer Vision, Graphics and Image Processing, Vol. 29, pp. 400-411.

[72] **Sun, C.** and Wee, W.G. (1983): "Neighbouring Gray Level Dependence Matrix for Texture Classification", Computer Graphics and Image Processing, Vol. 23, pp. 341-352.

[73] **Sutton R.** and Hall, E. (1972): "Texture Measures for Automatic Classification of Pulmonary Diseases", IEEE Trans. Comput., Vol. C-21, pp. 667-672.

[74] **Tamura, H.,** Mori, S. and Yamawaki, T. (1978): "Textural Features Corresponding to Visual Perception", IEEE Trans. Syst., Man, Cybern., Vol. SMC-8, pp. 460-473.

[75] **Trivedi, M.M.** and Bezdek, J.C. (1986): "Low-Level Segmentation of Aerial Images with Fuzzy Clustering", IEEE Trans. Syst., Man, Cybern., Vol. SMC-16, No. 4, pp. 589-598.

[76] **Tsuji, S.** and Tomita, F. (1973): "A Structural Analyzer for a Class of Textures", Computer Graphics and Image Processing, Vol. 2, p. 216-231.

[77] **Vickers, A.L.** and Modestino, J.W. (1982): "A Maximum Likelihood Approach to Texture Classification", IEEE Trans. Pattern Anal. Mach. Intell., Vol. PAMI-4, pp. 61-68.

[78] **Vilrotter, F.M.,** Nevatia, R. and Price, K.E. (1986): "Structural Analysis of Natural Textures", IEEE Trans. Pattern Anal. Mach. Intell., Vol. PAMI-8, pp. 76-90.

[79] **Watanabe, S.** and the CYBEST Group (1974): "An Automated Apparatus for Cancer PreScreening", CYBEST, Computer Graphics and Image Processing, Vol. 3, pp. 350-358.

[80] **Weszka, J.S.,** Dyer, C.R. and Rosenfeld, A. (1976): "A Comparative Study of Texture Measures for Terrain Classification", IEEE Trans. Syst., Man, Cybern., Vol. SMC-6, pp. 269-285.

[81] **Weszka, J.S.,** Nagel, R.N. and Rosenfeld, A. (1974): "A Threshold Selection Technique", IEEE Trans. Comput., Vol. C-23, p. 1322-1326.

[82] **Yakismovsky, Y.** (1976): "Boundary and Object Detection in Real World Images", J. Assoc. Comput. Mach., Vol. 23, pp. 599-618.

[83] **Zucker, S.W.,** Rosenfeld, A. and Davies, L.S. (1975): "Picture Segmentation by Texture Discrimination, IEEE Trans. Comput., Vol. C-24, pp. 1228-1233.