E7.5 10011

CR-140655

Original photography may be purchased from:
EROS Data Center
10th and Dakota Avenue
Sioux Falls, SD 57198

"Made available under NASA sponsorship in the interest of early and wide dissemination of Earth Resources Survey Program information and without liability for any use made thereof."

(E75-10011) USE OF FEATURE EXTRACTION TECHNIQUES FOR THE TEXTURE AND CONTEXT INFORMATION IN ERTS IMAGERY: SPECTRAL AND (Kansas Univ. Center for Research, Inc.) 273 p HC $8.50 CSCL 05B

N75-13335

Unclas G3/43 00011

1060A

THE UNIVERSITY OF KANSAS CENTER FOR RESEARCH, INC.
Space Technology Center—Nichols Hall
2291 Irving Hill Drive—Campus West

TECHNICAL REPORT STANDARD TITLE PAGE

1. Report No. 2261-10	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle Kansas Environmental and Resource Study: A Great Plains Model-Use of Feature Extraction Techniques for the Texture and Context Information in ERTS Imagery		5. Report Date March 17, 1974
		6. Performing Organization Code
7. Author(s) R. M. Haralick and Robert Bosley		8. Performing Organization Report No.
9. Performing Organization Name and Address University of Kansas Center for Research, Inc. c/o Space Technology Center 2291 Irving Hill Dr.-Campus West Lawrence, Kansas 66045		10. Work Unit No.
		11. Contract or Grant No. NAS5-21822, Task 1
12. Sponsoring Agency Name and Address Goddard Space Flight Center Greenbelt, Maryland 20771 GSFC Technical Monitor: G. R. Stonesifer, Code 430		13. Type of Report and Period Covered Type III Report August 1, 1972-March 17, 1974
		14. Sponsoring Agency Code
15. Supplementary Notes		

16. Abstract

A procedure is developed to extract cross-band textural features from ERTS multi-spectral scanner imagery. Evolving from a single-image texture extraction procedure which uses spatial dependence matrices to measure relative co-occurrence of nearest neighbor grey tones, the cross-band texture procedure uses the distribution of neighboring grey tone N-tuple differences to measure the spatial inter-relationships, or co-occurrences, of the grey tone N-tuples present in a texture pattern. In both procedures, texture is characterized in such a way as to be invariant under linear grey tone transformations. However, the cross-band procedure compliments the single-image procedure by extracting texture information and spectral information contained in ERTS multi-images. Classification experiments show that when used alone, without spectral processing, the cross-band texture procedure extracts more information than the single-image texture analysis. Results show an improvement in average correct classification from 86.2% to 88.8% for ERTS image no. 1021-16333 with the cross-band texture procedure. However, when used together with spectral features, the single-image texture plus spectral features perform better than the cross-band texture plus spectral features, with an average correct classification of 93.8& and 91.6% respectively.

17. Key Words (Selected by Author(s)) Texture, land-use classification, spatial and spectral processing		18. Distribution Statement	
19. Security Classif. (of this report) UNCLASSIFIED	20. Security Classif. (of this page) UNCLASSIFIED	21. No. of Pages 249	22. Price*

PRECEDING PAGE BLANK NOT FILMED

KANSAS ENVIRONMENTAL AND RESOURCE STUDY:
A GREAT PLAINS MODEL

SPECTRAL AND TEXTURAL PROCESSING OF ERTS IMAGERY

PREFACE

A procedure is developed to extract cross-band textural features from ERTS multi-spectral scanner imagery. Evolving from a single-image texture extraction procedure which uses spatial dependence matrices to measure relative co-occurrence of nearest neighbor grey tones, the cross-band texture procedure uses the distribution of neighboring grey tone N-tuple differences to measure the spatial interrelationships, or co-occurrences, of the grey tone N-tuples present in a texture pattern. In both procedures, texture is characterized in such a way as to be invariant under linear grey tone transformations. However, the cross-band procedure compliments the single-image procedure by extracting texture information and spectral information contained in ERTS multi-images. Classification experiments show that when used alone, without spectral processing, the cross-band texture procedure extracts more information than the single-image texture analysis. Results show an improvement in average correct classification from 86.2% to 88.8% for ERTS image no. 1021-16333 with the cross-band texture procedure. However, when used together with spectral features, the single-image texture plus spectral features perform better than the cross-band texture plus spectral features, with an average correct classification of 93.8% and 91.6%, respectively.

STIF

KANSAS ENVIRONMENTAL AND RESOURCE STUDY: A GREAT PLAINS MODEL

MARCH 1974

Type III Final Report for the Period
August 1, 1972 - March 17, 1974

Prepared for:

National Aeronautics and Space Administration
Goddard Spaceflight Center
Greenbelt, Maryland 20771

Contract No. NAS 5-21822

THE UNIVERSITY OF KANSAS CENTER FOR RESEARCH, INC.
2385 Irving Hill Rd.—West Campus Lawrence, Kansas 66044

KANSAS ENVIRONMENTAL AND RESOURCE STUDY:
A GREAT PLAINS MODEL

Use of Feature Extraction Techniques for the Texture
and Context Information in ERTS Imagery:

Spectral and Textural Processing of ERTS Imagery

R. M. Haralick, Principal Investigator
University of Kansas Center for Research, Inc.
Remote Sensing Laboratory
c/o Space Technology Center
Nichols Hall
2291 Irving Hill Drive-Campus West
Lawrence, Kansas 66045

March 17, 1974
Type III Final Report for the Period August 1, 1972 - March 17, 1974

Prepared for:

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
GODDARD SPACEFLIGHT CENTER
GREENBELT, MARYLAND 20771

Type III Final Report

for the period: August 1, 1972 - March 17, 1974

Title of Investigation: Use of Feature Extraction Techniques for Texture Context Information in ERTS Imagery

ERTS-A Proposal Number: 60-1

Task Number: 1

Co-Investigators: R. M. Haralick and G. L. Kelly

NASA-GSFC PI ID Number: UN 094

Report Prepared by:

Robert J. Bosley
Research Assistant

Report Approved by:

R. M. Haralick
Co-Principal Investigator

TABLE OF CONTENTS

PRECEDING PAGE BLANK NOT FILMED

LIST OF FIGURES

LIST OF TABLES

LIST OF TABLES

I. INTRODUCTION

A procedure is developed to extract textural features for automatic analysis of ERTS multi-spectral scanner imagery. Previous work (Haralick, 1973; Haralick, 1972) indicates that useful textural features can be computed from the co-occurrence matrices for grey tones in specific spatial relationships on an image. The performance of the land-use classification algorithm using these textural features from only one band is encouraging; up to 75 per cent of the images were correctly classified (Haralick, 1973). Since textural features and spectral features of ERTS multi-images provide different kinds of information, a significant increase in identification accuracy will occur when both features are used together.

Adoption of the texture procedure for multi-images leads to excessive amounts of storage for the grey tone N-tuple co-occurrence matrices. Therefore, to solve the storage problem we measure grey tone N-tuple differences instead of grey tone N-tuples and assume an ellipsoidally symmetric functional form for the co-occurrence distribution of multi-image grey tone N-tuple differences.

It is expected that the estimated parameters of the ellipsoidally symmetric distribution will lead to textural features that can distinguish between texturally distinct categories on ERTS MSS images over Kansas. In order to obtain more texture information, the dimensionality of the grey tone N-tuples was increased from the original four MSS bands by the addition of cross-band product terms for higher order components. This procedure for cross-band texture analysis of multi-images provides a natural extension of the single-image texture analysis while retaining its advantages: invariance under translating and scaling transformations, low storage requirements, and direct proportionality between the number of operations required to process an image and the number of resolution cells present in an image.

II. TEXTURE

Texture and tone are two fundamental pattern elements used in the interpretation of image data. The concept of tone is concerned with the whiteness, greyness, or blackness of resolution cells of the image. The concept of texture is concerned with the spatial distribution of the grey tones. Tone is based upon the varying shades of grey of the resolution cells in the image, while texture is based upon the spatial distribution of grey tones. However, texture and tone are not independent concepts but are intrinsically related to one another. Although either property can dominate the other depending upon the image context, texture and tone are always present.

When one attempts to objectively use tone and texture pattern elements, the texture-tone concept must be explicitly defined. This can be visualized as follows. When a small area patch of an image has little variation of features of discrete grey tone, then that area is dominated by tonal properties. As the number of distinguishable features of discrete grey tone increases within the patch, then the texture properties will dominate. The size of the small area patch, the relative sizes of the discrete features and the number of distinguishable discrete features are all crucial in this distinction. When the size of the small area patch is reduced to one resolution cell, the only property present is tone. When there is no spatial pattern in the tonal features and the grey tone variation between features is wide, a fine texture results. And as the spatial pattern becomes more defined using more and more resolution cells, then a coarser texture results.

Texture can be termed as being fine, coarse, smooth, rippled, mottled irregular, or lineated. Texture is a property of nearly all surfaces, the grain of wood, the weave of fabric, the pattern of crops in a field, etc. Although texture is quite easy for humans to recognize and describe, it is quite subjective by its nature and is extremely difficult to precisely define and analyze by digital computers. Since the texture of images contains important information for discrimination purposes, textural features could be very useful.

III. REVIEW OF PAST WORK ON TEXTURE

To date there has been at least six different approaches to the problems of measuring and characterizing texture of images: autocorrelation functions, optical transforms, digital transforms, edgeness, structural elements, and spatial grey tone co-occurrence probabilities. The first three approaches all measure spatial frequency either directly or indirectly. Spatial frequency is related to texture because fine textures are rich in high frequencies while coarse textures are rich in low frequencies.

One alternative approach to viewing texture as spatial frequency distribution is to view texture as the amount of edgeness per unit area. Fine textures have a high number of edges per unit area whereas coarse textures have a small number of edges per unit area.

The structural element approach uses a matching procedure to detect the spatial regularity of shapes called structural elements in a binary image. When the structural elements themselves are single resolution cells, the information provided by this approach is the autocorrelation function of the binary image. By using larger and more complex shapes, a more generalized autocorrelation can be computed.

The grey tone spatial dependence approach characterizes texture by the spatial distribution of its grey tones. In coarse textures the distribution changes only slightly with distance, but for fine textures it changes rapidly with distance.

Because of our familiarity with the concepts of spatial frequency and edgeness, these approaches to texture characterizations are readily employed. However, an inherent problem exists with these approaches in regard to grey tone calibration of the image and they are not invariant under even a linear grey tone translation. And the price paid for invariance by compensating with quantization is a loss of grey tone precision in the quantized image.

The power of the structural element approach is that it emphasizes the shape aspects of the discrete tonal features. Weakness of this approach lies in that it can only do so for binary images.

The power of the spatial grey tone co-occurrence approach lies in characterizing the spatial inter-relationships of the grey tones in a texture pattern in such a way that is invariant under monotonic grey tone transformations. Weakness of the approach lies in failure to capture the shape aspects of the discrete tonal features.

IV. TEXTURAL FEATURES

The above description of texture is an idealization of what actually occurs, a gross simplification. Discrete tonal features are actually quite subjective in that they do not necessarily stand out as entities by themselves. Therefore, the texture analysis presented here is concerned with more general or macroscopic concepts rather than discrete tonal features. The procedure developed by Haralick (Haralick, 1972) for obtaining the textural features of an image is based on the assumption that the texture information on an image I is contained in the overall spatial co-occurrence relationship which the greytones in the image I have to one another. More specifically, we assume that this texture information is adequately specified by a set of spatial grey tone dependence matrices, which are computed for various angular relationships and distances between neighboring resolution cell pairs on the image. All of the textural features are then derived from these angular nearest neighbor spatial grey tone dependence matrices.

IV.1 Spatial Grey Tone Dependence Matrices

Let G = {0, 1,...,Ng} be the set of possible grey tones that each resolution cell can take on after image normalization by equal probability quantizing to Ng levels. It can be shown that this quantization quarantees that images which are a monotonic transformation of one another, such as lighter or darker images due to variations in film, lighting, or development, will produce the same results. Let Nx be the number of resolution cells in the horizontal direction and Ny the number of resolution cells in the vertical direction in the image to be analyzed so that Lx = {1, 2,...,Nx} and Ly = {1, 2,...,Ny} are the horizontal and vertical spatial domains. Then Ly x Lx will be the set of resolution cells of the image. And the image I can be represented as a function which assigns some grey tone in G to each resolution cell or pair of coordinates in Ly x Lx; I:Ly x Lx→G.

Essential to our conceptual framework of texture are four closely related measures called angular nearest neighbor grey tone spatial dependence matrices. The concept of angular nearest neighbor for a resolution cell is the adjacent resolution cell for a given angle, as shown in Figure 1.

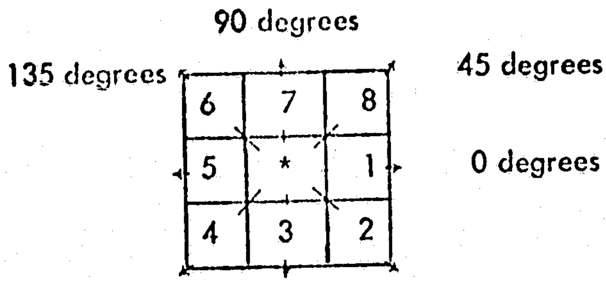


FIGURE 1. Eight nearest neighbor resolution cells of cell '*'. Resolution cells 1 and 5 are the 0-degree nearest neighbors to resolution cell '*', resolution cells 2 and and 6 are the 135-degree nearest neighbors, etc. Note that this information is purely spatial, having nothing to do with grey tone values.

We assume that the texture information in our image I is contained in the overall or "average" spatial relationship which the grey tones in image I have to one another. Specifically, we shall assume that this information is adequately specified by the matrix of relative frequencies Pij with which two neighboring resolution cells separated by distance d occur on the image, one with grey tone i and the other with grey tone j. These matrices of spatial grey tone dependence frequencies are a function of the angular relationship between the neighboring resolution cells as well as a function of the distance between them. Figure 2 illustrates the set of all horizontal neighboring resolution cells separated by distance 1. This set along with the image grey tones would be used to calculate a distance i horizontal spatial grey tone co-occurrence matrix Formally, for angles quantized to 45° intervals the unnormalized frequencies are defined by:

$$P(i,j,d,0^{\circ}) = \#\{((k,l),(m,n)) \in (L_y \times L_x) \times (L_y \times L_x) \mid k-m=0,\ |l-n|=d,\ I(k,l)=i,\ I(m,n)=j\}$$

$$P(i,j,d,45^{\circ}) = \#\{((k,l),(m,n)) \in (L_y \times L_x) \times (L_y \times L_x) \mid (k-m=d,\ l-n=-d) \text{ or } (k-m=-d,\ l-n=d),\ I(k,l)=i,\ I(m,n)=j\}$$

$$P(i,j,d,90^{\circ}) = \#\{((k,l),(m,n)) \in (L_y \times L_x) \times (L_y \times L_x) \mid |k-m|=d,\ l-n=0,\ I(k,l)=i,\ I(m,n)=j\}$$

$$P(i,j,d,135^{\circ}) = \#\{((k,l),(m,n)) \in (L_y \times L_x) \times (L_y \times L_x) \mid (k-m=d,\ l-n=d) \text{ or } (k-m=-d,\ l-n=-d),\ I(k,l)=i,\ I(m,n)=j\}$$

Note that these matrices are symmetric; $P(i, j; d, a) = P(j, i; d, a)$. The distance metric ρ implicit in the above equations can be explicitly defined by $\rho((k, l), (m, n)) = \max \{|k-m|, |l-n|\}$.

(1,1)	(1,2)	(1,3)	(1,4)
(2,1)	(2,2)	(2,3)	(2,4)
(3,1)	(3,2)	(3,3)	(3,4)
(4,1)	(4,2)	(4,3)	(4,4)

$$L_y = \{1,2,3,4\}$$
$$L_x = \{1,2,3,4\}$$

$$
\begin{aligned}
R_H = & \{((k,l),(m,n)) \in (L_y \times L_x) \times (L_y \times L_x) \mid k-m = 0,\ |l-n| = 1\} \\
= & \{((1,1),(1,2)),\ ((1,2),(1,1)),\ ((1,2),(1,3)),\ ((1,3),(1,2)), \\
& ((1,3),(1,4)),\ ((1,4),(1,3)),\ ((2,1),(2,2)),\ ((2,2),(2,1)), \\
& ((2,2),(2,3)),\ ((2,3),(2,2)),\ ((2,3),(2,4)),\ ((2,4),(2,3)), \\
& ((3,1),(3,2)),\ ((3,2),(3,1)),\ ((3,2),(3,3)),\ ((3,3),(3,2)), \\
& ((3,3),(3,4)),\ ((3,4),(3,3)),\ ((4,1),(4,2)),\ ((4,2),(4,1)), \\
& ((4,2),(4,3)),\ ((4,3),(4,2)),\ ((4,3),(4,4)),\ ((4,4),(4,3))\}
\end{aligned}
$$

FIGURE 2. Illustrates the set of all Distance 1 Horizontal Neighboring Resolution Cells on a 4 x 4 Image.

For an example of the four distance 1 grey tone spatial dependence matrices, consider Figure 3. Figure 3-a represents a 4 x 4 image with four grey tones, ranging from 0 to 3. Figure 3-b shows the general form for any grey tone spatial dependence matrix. For example, the element in the (2,1)-st position of the distance 1 horizontal P_H matrix is the total number of times two grey tones of value 2 and 1 occurred horizontally adjacent to each other. To determine this number, we count the number of pairs of resolution cells in R_H such that the first resolution cell of the pair has grey tone 2 and the second resolution cell of the pair has grey tone 1. Figure 3-c through 3-f shows all four distance 1 grey tone spatial dependence matrices.

From the grey tone dependence matrices a set of 17 textural features is derived. The equations defining these 17 features are given in Appendix I. To illustrate the significance of these features, three are defined as follows:

$$f_1 = \sum_{i=1}^{N_g} \sum_{j=1}^{N_g} \left(\frac{P(i,j)}{\#R} \right)^2 ,$$

$$f_{12} = \sum_{n=0}^{N_g - 1} n^2 \left\{ \sum_{|i-j|=n} \left(\frac{P(i,j)}{\#R} \right) \right\}$$

$$f_3 = \frac{\sum_{i=1}^{N_g} \sum_{j=1}^{N_g} \frac{ij\, P(i,j)}{\#R} - \mu_x \mu_y}{\sigma_x \sigma_y}$$

where

$\#R$ = number of resolution cells pairs, and μ_x and σ_x are the mean and standard deviation of the marginal distribution P_x defined by

$$P_x(i) = \sum_{j=1}^{N_g} \frac{P(i,j)}{\#R}$$

and μ_y and σ_y are the mean and standard deviation of the marginal distribution P_y defined by:

$$P_y(j) = \sum_{i=1}^{N_g} \frac{P(i,j)}{\#R}$$

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

Figure 3-a.

		Grey Tone 0	1	2	3
	0	#(0,0)	#(0,1)	#(0,2)	#(0,3)
	1	#(1,0)	#(1,1)	#(1,2)	#(1,3)
Grey Tone	2	#(2,0)	#(2,1)	#(2,2)	#(2,3)
	3	#(3,0)	#(3,1)	#(3,2)	#(3,3)

Figure 3-b. This shows the general form of any grey tone spatial dependence matrix for an image with integer grey tone values 0 to 3. #(i,j) stands for number of times grey tones i and j have been neighbors.

0°

$$P_H = \begin{pmatrix} 4 & 2 & 1 & 0 \\ 2 & 4 & 0 & 0 \\ 1 & 0 & 6 & 1 \\ 0 & 0 & 1 & 2 \end{pmatrix}$$

Figure 3-c.

90°

$$P_V = \begin{pmatrix} 6 & 0 & 2 & 0 \\ 0 & 4 & 2 & 0 \\ 2 & 2 & 2 & 2 \\ 0 & 0 & 2 & 0 \end{pmatrix}$$

Figure 3-d.

135°

$$P_{LD} = \begin{pmatrix} 2 & 1 & 3 & 0 \\ 1 & 2 & 1 & 0 \\ 3 & 1 & 0 & 2 \\ 0 & 0 & 2 & 0 \end{pmatrix}$$

Figure 3-e.

45°

$$P_{RD} = \begin{pmatrix} 4 & 1 & 0 & 0 \\ 1 & 2 & 2 & 0 \\ 0 & 2 & 4 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

Figure 3-f.

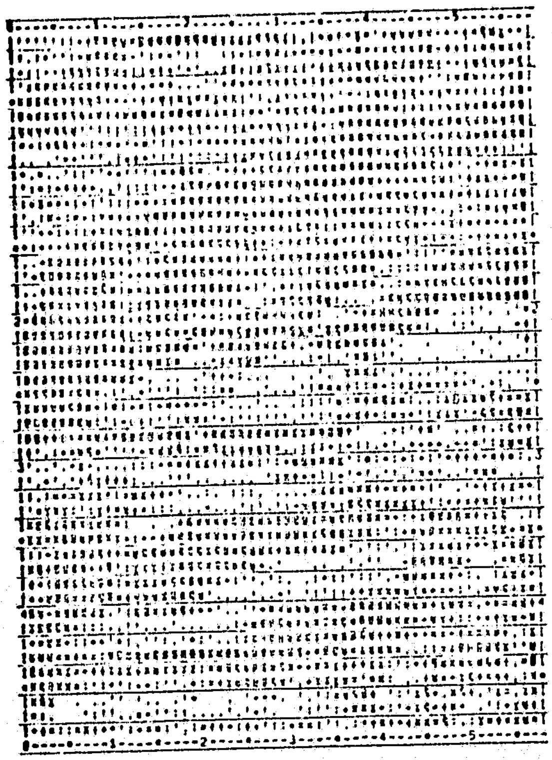
To explain the significance of these features, let us consider the kind of values they take on two different land use category images. Figure 4 shows the digital printout of two sub-images of size 64 x 64 resolution cells (approximately 8.5 square mile area) from MSS band 5 of 1002-18134, see Figure 10. The image shown in 4 (a) belongs to the grass land category and the image in Figure 4(b) is mostly water. Values of the features f_1, f_{12}, and f_3 are also shown for these images in Figure 4.

The angular second moment feature (ASM), f_1, is a measure of homogeneity of the image. In a homogeneous image, such as shown in 4(b), there are very few dominant grey tone transitions. Hence, the P matrix for this image will have fewer entries of large magnitude. For an image like the one shown in Figure 4(a), the P matrix will have a large number of small entries and hence the ASM feature which is the sum of squares of the entries in the P matrix will be smaller. A comparison of the ASM values given below the images in Figure 4 shows the usefulness of the ASM feature as a measure of the homogeneity of the image.

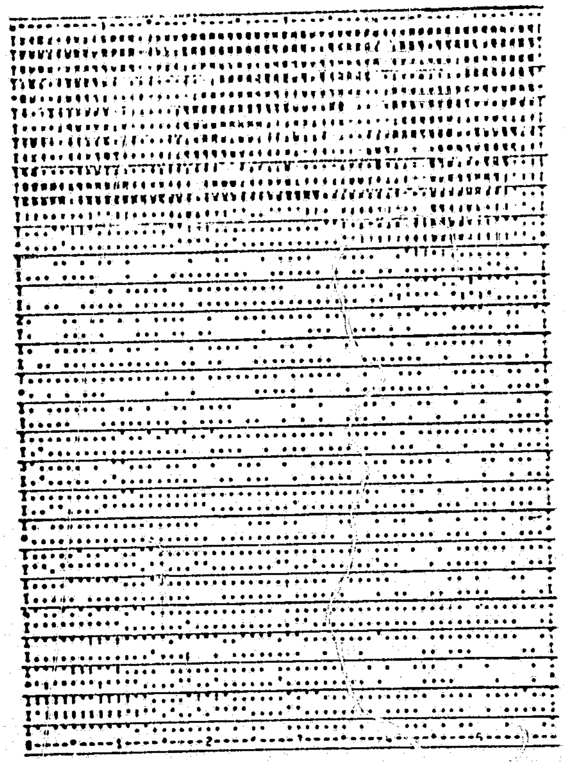
The contrast feature, f_{12}, is obtained as a difference moment of the P matrix and is a measure of the contrast or the amount of boundaries present in an image. Since there is a large amount of boundaries present in the image 4(a) compared to the image shown in 4(b), the contrast feature for the grassland image has consistently higher values compared to the water body image.

The correlation feature, f_3, is a measure of linear grey tone dependencies in the image. For both the images shown in Figure 4, the correlation feature is somewhat higher in the horizontal (0°) direction, along the line of scan. The water body image consists mostly of a constant grey tone value for the water plus some additive noise. Since the noise samples are mostly uncorrelated, the correlation features for the water body image have lower correlation values compared to the grassland image. Also the grassland image has a considerable amount of linear structure along 45° lines across the image and hence the value of the correlation feature is higher along this direction compared to the values for 90° and 135° directions.

The various features presented here are all functions of distance and angle. The angular dependencies present a special problem. Suppose image A has features a, b, c, d for angles 0°, 45°, 90°, and 135° and image B is identical to A except that B is rotated 90° with respect to A. Then B will have features c, d, a, b, for angles 0°, 45°, 90°, and 135° respectively. Since the texture



a. Grassland



b. Water Body

Angle	ASM	Contrast	Correlation	ASM	Contrast	Correlation
0°	.0128	3.048	.8075	.1016	2.153	.7254
45°	.0080	4.011	.6366	.0771	3.057	.4768
90°	.0077	4.014	.5987	.0762	3.113	.4646
135°	.0064	4.709	.4610	.0741	3.129	.4650
Avg.	.0087	3.945	.6259	.0822	2.863	.5327

Figure 4. Textural Features for Two Different Land Use Category Images.

context of A is the same as the texture context of B, any decision rule using the angular features a, b, c, d must produce the same results for c, d, a, b, or for that matter b, c, d, a (45° rotation) and d, a, b, c, (135° rotation). To guarantee this, we do not use the angularly dependent features directly. Instead, we merge the four arrays by summing corresponding elements. The merged array is then used for computing the 17 texture features defined in Appendix I.

IV.2 Textural Features for Multi-Images

Results of previous work in texture using the spatial grey tone dependence matrices as the basis from which all textural features are extracted has been very encouraging (Haralick, 1973). The good performance of these texture features can be seen by the results of the classification experiments. The computational aspects of the procedure are also notable. The number of operations required to process an image using the spatial grey tone dependence matrices is directly proportional to the number of resolution cells, N, present in an image. In comparison, the number of operations needed to use Fourier or Hadamard transforms to extract texture information are of the order of N log N. And, to compute the entries in the spatial grey tone dependence matrices, one needs to keep only two lines of image data in core at a time, keeping storage requirements to a minimum.

Even with these advantages, however, the extraction of texture information from multiimages, as in the case of ERTS MSS data, forces a new approach to the measurement of grey tone N-tuple co-occurrences. The use of the spatial dependence matrices requires that they be stored in the computer. For multi-images containing grey tone N-tuples, we have too many possible grey tone N-tuples which can neighbor each other and as a result, the dependence matrices will be very large. For example, for four MSS bands in which each grey tone can range through 64 levels, each matrix would have $64^4 \times 64^4$ elements. Even using the symmetry of the matrices to reduce the number of entries does not help since there would be on the order of 10^{15} entries.

The spatial dependence matrices, however, provide a way of escape. In using these matrices, it was observed that they are heavily weighted along the diagonal with decreasing entries farther from the diagonal. Figure 5 gives an example of one of these matrices. Note that the number of entries decreases as we move away from the diagonal. This suggests that neighboring resolution cells are similar. Choosing any resolution cell in an image at random, we are very likely to find nearly identical neighbors to the cell in all directions and less likely to find dissimilar neighbors. Clearly, a measure which indicates how similar the neighboring N-tuples are and how fast the similarity drops off with distance must contain textural information about the object imaged.

It is therefore reasonable to measure the difference between neighboring grey tone N-tuples and observe this distribution instead of computing the number of times each N-tuple neighbors every other N-tuple. In both cases we measure the co-occurrence of nearest neighbor grey tone N-tuples.

Since the textural features are based on the spatial dependence of grey tone N-tuples, our first step must be to define a binary relation between neighboring resolution cells on which the co-occurrence of grey tone N-tuples can be counted. As above, let $Lx = \{1, 2, \ldots, Nx\}$ and $Ly = \{1, 2, \ldots, Ny\}$ be the set of column and row indexes, respectively, so that $Ly \times Lx$ is the set of resolution cells in the image. Let $G = \{0, 1, \ldots, Ng\}$ be the set of possible grey tones that each component of every grey tone N-tuple can be assigned. Then, the image I can be defined by $I: Ly \times Lx \rightarrow G \times G \times \ldots \times G$.

Let R be the set of all pairs of resolution cells in a specified spatial relation. Then R is a binary relation on the set $Ly \times Lx$; $R \subseteq (Ly \times Lx) \times (Ly \times Lx)$. For example, the set of all distance 1 horizontally neighboring pairs of neighboring resolution cells would be defined by:

$$R = \Big\{\big((k, l), (m, n)\big) \in (Ly \times Lx) \times (Ly \times Lx) \mid k-m = 0, \ |l-n| = 1\Big\}.$$

The co-occurrence frequency of grey tone N-tuples $(i_1, i_2, \ldots, i_N)$ and $(j_1, j_2, \ldots, j_N)$ in spatial relation defined by R is

$$P\big((i_1, \ldots, i_N), (j_1, \ldots, j_N)\big) = \frac{\# \Big\{\big((k, l), (m, n)\big) \in R \mid I(k, l) = (i_1, \ldots, i_N), I(m,n) = (j_1, \ldots, j_N)\Big\}}{\# R}$$

where # denotes the number of elements in the set.

Note that this R is symmetric. Assume that $\big((k, l), (m, n)\big)$ is in R. Then $k-m = 0$, and $|l-n| = 1$ from the definitions of R. But $|l-n| = 1$ when $|n-l| = 1$.

30 DEGREES

24	26																														
15	23	28																													
6	16	19	8																												
15	7	11	11	8																											
5	6	7	7	6	6																										
1	3	6	6	3	7	3																									
2	5	8	2	8	2	3	8																								
1	6	4	1	4	2	3	4	2																							
1	1	5	4	2	3	4	3	1	2																						
3	1	6	5	1	6	5	1	1	1	2																					
2	3	5	3	2	2	0	2	3	2	0	0																				
0	1	4	6	4	5	5	2	3	0	4	2	0																			
0	2	1	6	2	4	2	2	3	3	2	2	3	0																		
1	2	6	3	5	1	3	4	0	0	5	1	2	1	0																	
0	2	3	5	2	4	2	2	5	1	1	2	6	1	0	2																
0	1	0	3	3	1	2	2	1	2	4	3	1	1	1	4	4															
2	0	3	4	2	2	1	6	1	1	2	0	2	2	2	2	0	6														
0	1	1	1	2	1	0	0	3	1	0	0	3	2	0	4	0	0	0													
0	0	1	0	3	1	1	1	0	0	1	4	1	1	0	0	1	3	0	0												
0	0	1	0	2	0	3	0	2	2	0	2	3	2	1	0	1	0	0	2	2											
0	0	0	0	1	0	4	1	1	2	1	1	2	3	0	3	1	1	2	2	2	2										
0	0	0	1	0	0	0	1	1	0	2	3	2	2	0	1	3	0	1	0	0	1	2									
0	0	1	0	1	0	0	0	1	1	0	4	1	0	2	1	4	2	1	0	0	1	0	0								
0	1	0	0	1	3	0	0	2	1	2	4	0	2	4	7	4	0	2	2	2	0	1	1	2							
0	0	0	0	2	1	0	3	1	1	1	2	1	1	1	1	1	3	1	1	1	0	2	2	2	2						
0	0	0	0	1	0	0	1	0	1	2	2	1	1	2	3	0	3	1	0	1	1	1	1	3	1	2					
0	0	0	0	1	1	1	0	0	0	1	1	2	2	1	3	1	6	3	0	0	1	4	0	2	3	1	0				
0	0	0	1	0	0	0	0	1	0	0	0	2	1	2	2	0	4	0	2	1	8	2	2	8	3	3	0	6			
0	0	0	0	0	0	0	0	0	0	1	0	1	1	0	1	3	2	2	2	0	0	0	1	7	3	4	4	3	8		
0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	0	1	1	1	0	0	1	1	1	0	3	8	4	2	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	7	6	10

FIGURE 5. Example of Nearest Neighbor Grey Tone Dependence Matrices, Taken from Processing ERTS Data.

And if $|n-l| = 1$ and $k-m = 0$, then $((m, n), (k, l))$ is in R. Thus, R is symmetric. In fact, by the symmetry of any distance function, R, in general, must be symmetric. And since R is symmetric, P is also symmetric.

IV.3 Textural Feature Extraction Procedure

Let R be a symmetric binary relation pairing nearby neighboring resolution cells. We define the frequency of grey tone N-tuple differences co-occurring in the spatial configuration defined by R as

$$P(x_1, x_2, \ldots, x_N) = \frac{\# \left\{ \left((i, j), (m, n)\right) \in R \mid I(i, j) - I(m, n) = \begin{pmatrix} x_1 \\ \vdots \\ x_N \end{pmatrix} \right\}}{\# R}.$$

Note that P is an even function since

$$P(x_1, x_2, \ldots, x_N) = \# \left\{ \left((i, j), (m, n)\right) \in R \mid I(i, j) - I(m, n) = \begin{pmatrix} x_1 \\ \vdots \\ x_N \end{pmatrix} \right\} / \# R$$

$$= \# \left\{ \left((i, j), (m, n)\right) \in R^{-1} \mid I(i, j) - I(m, n) = \begin{pmatrix} x_1 \\ \vdots \\ x_N \end{pmatrix} \right\} / \# R$$

$$= \# \left\{ \left((m, n), (i, j)\right) \in R \mid I(i, j) - I(m, n) = \begin{pmatrix} x_1 \\ \vdots \\ x_N \end{pmatrix} \right\} / \# R$$

$$= \# \left\{ \left((m, n), (i, j)\right) \in R \mid I(m, n) - I(i, j) = \begin{pmatrix} -x_1 \\ \vdots \\ -x_N \end{pmatrix} \right\} / \# R$$

$$= P(-x_1, -x_2, \ldots, -x_N).$$

Referring to the monotonic behavior of nearly every column in the matrices of Figure 5, and assuming that this behavior occurs on every band of the ERTS multi-images, it is reasonable to assume that the even frequency distribution $P(x_1, \ldots, x_N)$ of the nearby grey tone N-tuple differences can be adequately approximated using an ellipsoidally symmetric distribution; thus we may write

$$P(x_1, x_2, \ldots, x_N) = f(x' A x)$$

for some monotonically decreasing function f.

This implies that only the function f and the matrix A need to be determined. We take f to be one of the two forms $e^{-1/2\mu^2}$, $(1+\mu^2)^{-m}$. Figure 6 is a scattergram of the differences for the first two bands of distance 1 horizontally neighboring resolution cells of a 64x64 sample image. Figures 6 and 7 clearly show the ellipsoidally symmetric functional form of the distribution of neighboring differences. In Appendix II N-dimensional spherical coordinate systems and ellipsoidally symmetric distributions are discussed and it is shown that the matrix A is proportional to the inverse covariance matrix of the N-tuple differences. Thus, we estimate A by a matrix proportional to the inverse of the estimate for the covariance matrix.

Therefore, if the image is blocked into subimages of small area so that each subimage is essentially of one category, we can expect the distribution of grey tone N-tuple differences over each subimage to be a function only of the assumed form of the function f and the covariance matrix of the difference vectors for grey tone N-tuples in a specified spatial relationship within the subimage. This leads us to consider textural features for multi-images based upon the elements of this spatial-spectral covariance matrix.

Consider each covariance matrix as a vector. Consider the distribution of the set of covariance matrices from the blocked image. Since the entries of the covariance matrix are the parameters of the distribution, we would like to have these entries invariant with respect to scale changes on the grey tone N-tuple differences. In order to do this, we scale the grey tone N-tuple differences so that all components have variance 1. The covariance matrix of these normalized differences is equivalent to the correlation matrix. Appendix III shows that this normalization procedure makes the covariance matrix invariant with respect to translating and scaling transformations on the grey tone N-tuples. The normalized covariance matrix can be considered as an extracted texture feature vector in an N(N-1)/2 dimensional hyperspace.

Initial classification experiments indicated a need for more textural information and in order to provide this the dimensionality, N, of each resolution cell was increased from the four provided by the four MSS bands to eight by appending higher order terms and cross-band product terms for each cell. The resultant increase in correct classification accuracy can be seen in section VII.

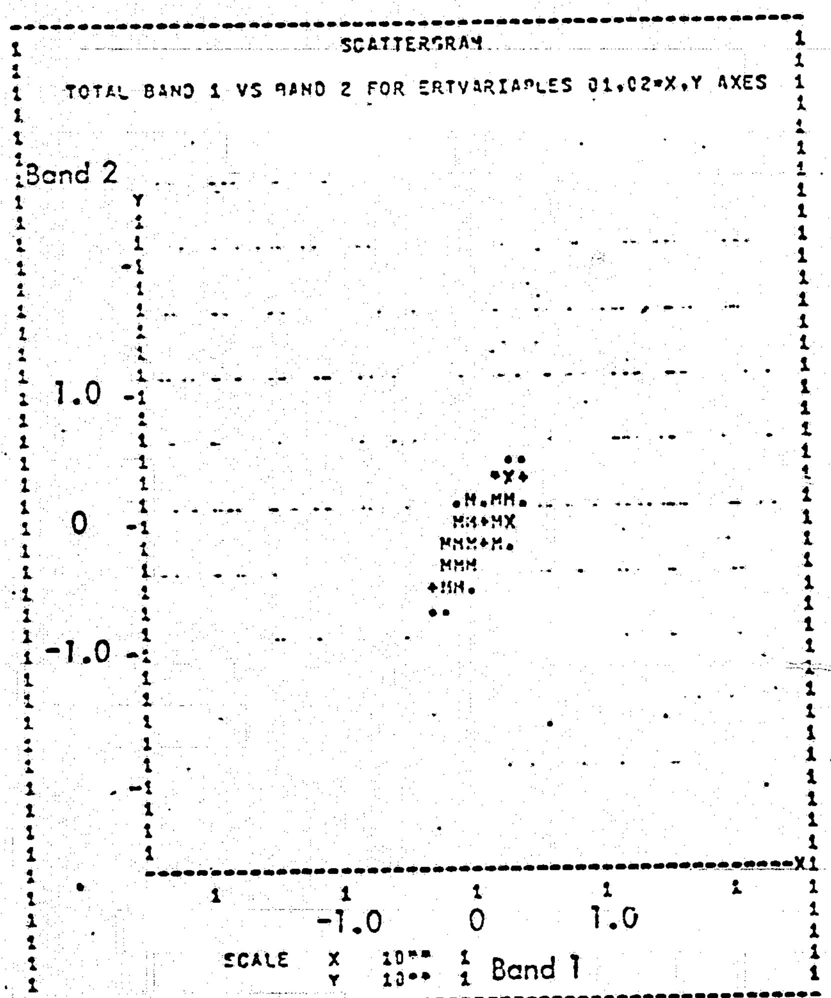


FIGURE 6. Scattergram showing ellipsoidally symmetric distribution of differences for Bands 4 and 5 over a sample 64 x 64 ERTS image.

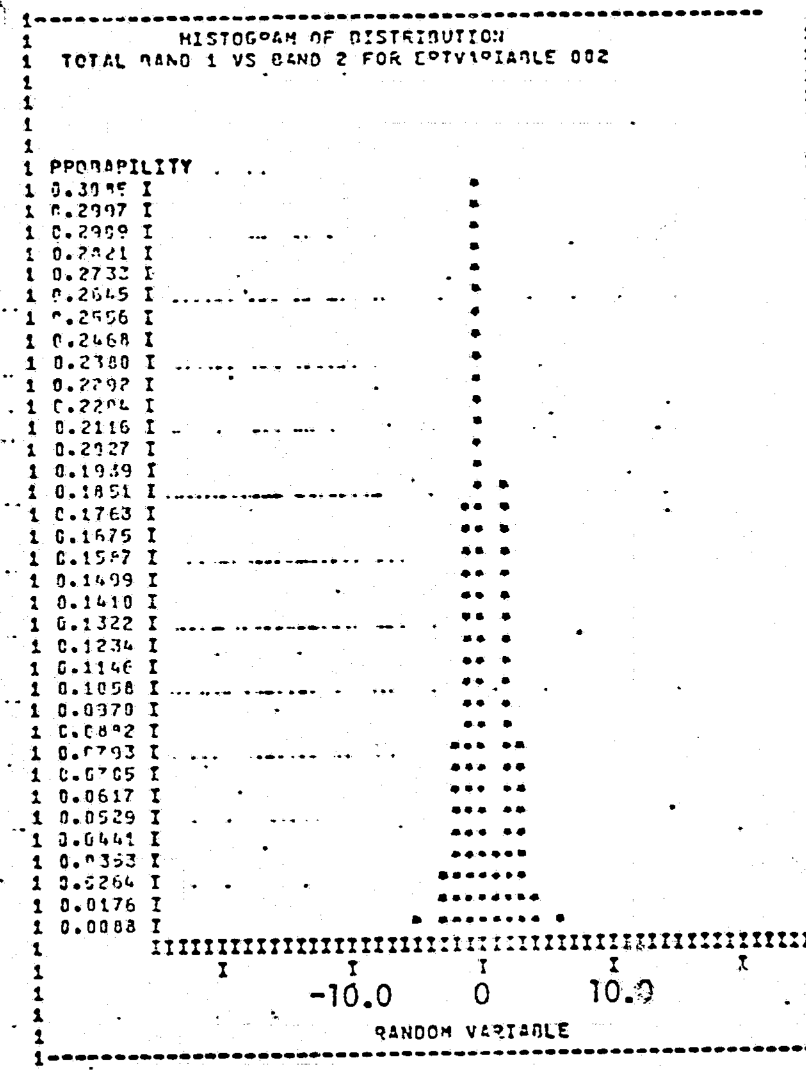


FIGURE 7. Histogram of the distribution of differences on Band 5 for distance 1 horizontally neighboring resolution cells.

V. GROUND TRUTH ASSIGNMENT PROCEDURE

For all classification experiments, the only method available to determine ground truth was based upon the 9 inch by 9 inch ERTS image transparency. Initially, the land-use category names were determined with the aid of two photo interpreters and, when available, contour maps published by the U. S. Geological Survey.

After the categories to be used were decided upon, each ERTS image was processed. The image was blocked into 64 x 64 subimages for texture analysis on MSS band 5 and 32 by 32 subimages for cross-band texture analysis of MSS bands 4 thru 7. The ERTS Retrieval Program was then used to printout a picture of the same area that was processed. This was subdivided into 64 x 64 subimages. In this way, the location of each subimage was known and could be fairly accurately determined on a 7 inch by 7 inch print. Then, on the basis of its location, the subimage is assigned a ground truth category. Figure 8 shows a picture printout by the ERTS Retrieval Program for a portion of image 1021-16333 (see Figure 12) over Kansas City. The picture has been blocked into 64 x 64 subimages and by refering to the original image in Figure 12, each subimage was assigned a ground truth category.

Figures 8 and 12 also illustrate the major problem with ground truth assignment. This is the determination of ground truth when a subimage covers more than one ground truth category. The irregular boundary around the urban area of Kansas City makes ground truth assignment difficult. The deciding factor was the amount of area within the subimage from each category. The subimage was assigned to the category which had the largest area within its boundary. For this reason the accuracy with which the location of every subimage is determined becomes very important, and the picture printouts from the retrieval programs aided in this.

Since the starting and ending image row and column coordinates were known, the Retrieval Program could use the same coordinates for the picture printout. This printout could then be accurately divided into 64 by 64 subimages. The actual location of these subimages could then be determined fairly well, although it is sometimes difficult to locate objects in the printout that are on the original image. To aid in this, a few of the well defined objects were used to construct a 64 by 64 grid on the 7 inch by 7 inch

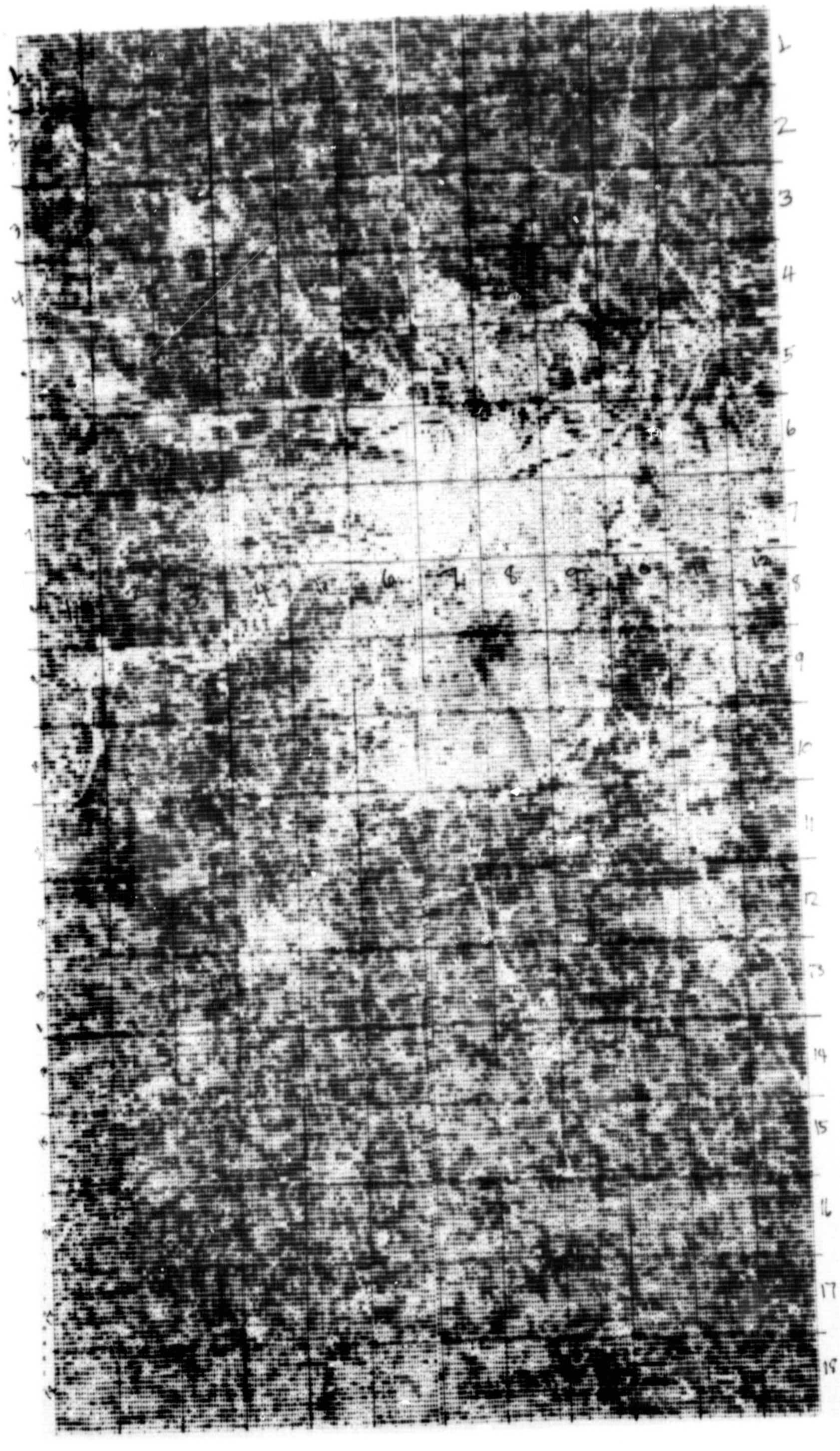


Figure 8. Printout of a Portion of Image 1021-16333 Over Kansas City by the ERTS Retrieval Program for Use in Ground Truth Assignment.

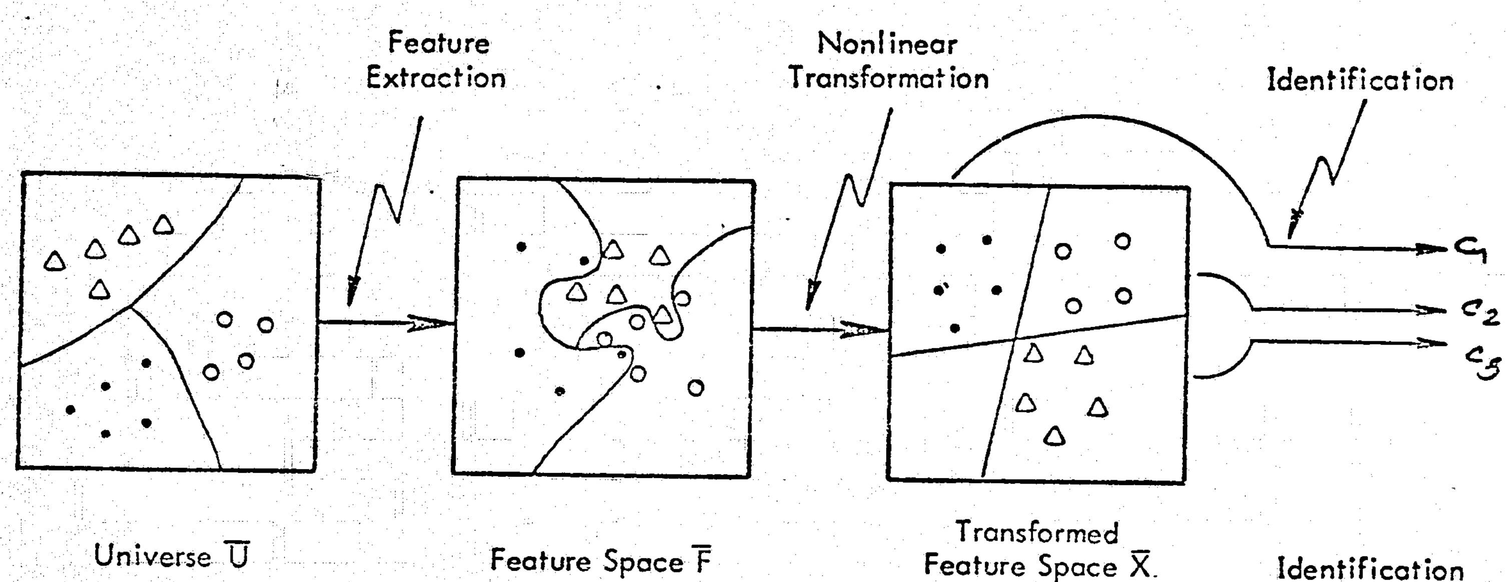
print. When there was a question in the ground truth, the printout could be referred to for the more accurate subimage location. The size of the subimage also relates directly to the accuracy with which its location on the print can be determined. It was found that 32 lines by 32 points per line was about as small as a subimage can be for good accuracy in ground truth assignment.

VI. IDENTIFICATION PROCEDURE

In the classification experiments two different classification algorithms were used, a Bayes classifier and a piecewise linear classifier. The Bayes classifier assumes a multivariate normal distribution and randomly choses pattern vectors for the training set where the piecewise linear classifier does not.

The problem of developing procedures for categorizing environmental units consists of the following.

With reference to Figure 9, the Universe $\bar{U}$ consists of environmental units (for example rocks) $U_1, U_2, \ldots, U_T$ which belongs to one of R possible categories $C_1, C_2, \ldots, C_R$(different land use categories). Of the large number of environmental units present in the universe, we observe a smaller subset of units $U_1, U_2, \ldots, U_N$. Our observations consist of a set of measured values of n features $f_1, f_2, \ldots, f_n$ for each unit U sampled. Based on the information contained in the feature vectors $F_1, F_2, \ldots, F_N$, the categories of the environmental units which produce these measurements being known, we want to develop an algorithm to identify the categories of new units based on the measurements they produce.



$$\overline{U} = \left\{ U_1, U_2, \ldots, U_T \right\}$$

$$\overline{F} = \left\{ F_1, F_2, \ldots, F_T \right\}; \quad F = [f_1 \; f_2 \; \ldots \; f_n]^T$$

$$\overline{X} = \left\{ X_1, X_2, \ldots, X_T \right\}; \quad X = [x_1 \; x_2 \; \ldots \; x_m]^T$$

The vectors F_i and X_i are usually referred to as feature vector and pattern vector respectively.

Figure 9. Identification Scheme.

The decision rule which assigns categories based on the values of features may be implemented in the feature space $\bar{F}$ by partitioning $\bar{F}$ into various regions and assigning categories to new units based on the regions to which their feature vectors belong. Efficient partitioning of the feature space may require complicated nonlinear decision boundaries (discriminant functions). Instead of deriving a decision rule in the feature space $\bar{F}$, we may transform the feature vectors into a new space $\bar{X}$ and implement a decision rule in the new space $\bar{X}$. By using appropriate nonlinear transformations, we may be able to implement nonlinear decision boundaries in $\bar{F}$ as linear decision boundaries in $\bar{X}$. Several procedures are available for deriving linear decision boundaries for partitioning $\bar{X}$ into various regions, based on the information contained in a set of sample patterns $X_1, X_2, \ldots, X_N$ whose categories are known.

<u>Identification Algorithms</u>: In a widely used algorithm (Fukunaga 1972, Fu and Mendel 1970, Miesel 1972), the pattern space $\bar{X}$ is separated into a number of regions using a set of hyperplanes (decision boundaries) whose locations are determined by the sample patterns. Each region is dominated by sample patterns of a particular category. When a new pattern is presented for identification, it is assigned a category depending on the region in which it belongs. If the new pattern X is located in a region dominated by sample patterns of category c_j, then X is classified as coming from category c_j.

To illustrate the procedure for obtaining the hyperplanes, consider the problem of separating the sample patterns $X_1, X_2, \ldots, X_{n_i}$ belonging to category c_i and $X_{n_i+1}, X_{n_i+2}, \ldots, X_{n_i+n_j}$ belonging to category c_j. We can write the linear discriminant function (hyperplane) which separates the patterns belonging to categories c_i and c_j as

$$h_{ij}(X) = V_{ij}^T X + v_{ij}^o \geq 0 \text{ for } X \epsilon c_i,$$

$$h_{ij}(X) = V_{ij}^T X + v_{ij}^o < 0 \text{ for } X \epsilon c_j$$

The vector V_{ij} and the scalar v_{ij}^o are to be determined from the information contained on the sample patterns.

If we introduce a new form to express the pattern vectors as

$$Z = [+1 \; x_1 \; x_2 \; \ldots \; x_n]^T \text{ for } X \epsilon c_i$$

$$Z = [-1 \; -x_1 \; -x_2 \; \ldots \; -x_n]^T \text{ for } X \epsilon c_j$$

then the discriminant function can be written as

$$h_{ij}(Z) = W_{ij}^T Z > 0 \qquad (1)$$

where W_{ij} is referred to as a weight vector and

$$h_{ij}(Z) = W_{ij}^T Z = 0$$

is the equation of a hyperplane in the transformed feature space.

The weight vector W_{ij} is chosen so as to satisfy equation 1 for as many training patterns as possible. Usually we do not know the precise form of h_{ij}. But, given our knowledge of the categories of the training patterns, we can postulate reasonable values $g_{ij}(Z_k)$ for $h_{ij}(Z_k)$ and choose W_{ij} to minimize the mean square error given by

$$\epsilon^2 = \frac{1}{n_i + n_j} \sum_{k=1}^{n_i + n_j} (W_{ij}^T Z_k - g_{ij}(Z_k))^2 \text{ .}$$

Usually $g_{ij}(Z_k)$ is taken to be +1 for $k = 1,2,\ldots,n_i + n_j$. We can rewrite ϵ^2 as,

$$\epsilon^2 = \frac{1}{(n_i + n_j)} [W_{ij}^T Y - G_{ij}^T][Y^T W_{ij} - I] \qquad (2)$$

where $\quad Y = [Z_1 \; Z_2 \; \ldots \; Z_{n_i + n_j}]$, and

$$G_{ij} = [g_{ij}(Z_1) \;\; g_{ij}(Z_2) \;\ldots\; g_{ij}(Z_{n_i + n_j})] \text{ .}$$

The weight vector which minimizes ϵ^2 given in equation 2 is given by

$$W_{ij} = (YY^T)^{-1} Y G_{ij}$$

which is the well-known normal equation set from linear least square theory.

For the multicategory problem involving N_R categories, a total of $N_R(N_R-1)/2$ hyperplanes must be determined using the procedure described above. After the hyperplanes are determined, the classification of new patterns is done as follows. For each category c_i, the number of hyperplanes, V_i, which give a positive response when the new pattern X is presented are determined using

$$V_i = \sum_{\substack{j=1 \\ j \neq i}}^{N_R} \frac{|W_{ij}^T Z| + W_{ij}^T Z}{2|W_{ij}^T Z|} \; ; \; i = 1,2,\ldots,N_R$$

where $Z = \begin{bmatrix} 1 \\ X \end{bmatrix}$.

X is assigned to category c_j if

$$V_j = \max_i \{V_i\}$$

If there is a tie between categories c_m and c_n, then X is assigned to c_m if $W_{mn}{}^T Z \geq 0$ or to c_n if $W_{mn}{}^T Z < 0$. Several modifications of the linear discriminant function method and a multitude of other classification procedures may be found in the references cited.

VII. RESULTS OF CLASSIFICATION EXPERIMENTS

Tables 1 thru 24 show the results of the classification experiments. Unless otherwise stated, the contingency tables are determined using the piecewise linear classification programs (RCLASS) given in Appendix IV. In each contingency table the number of errors (#ERR) and percent error (%ERR) is shown for both errors of commission and errors of ommission. The final entry in the percent error column is the average of the percent error for each category. The average correct classification is simply the number of correct classifications divided by the number of incorrect classifications. The final column in each contingency table (%SD) is an estimate for the standard deviation of the probability for correct classification. This was taken from Afarani (Afarani, 1972) where he gives an estimate for the variance of the probability of correct classification when the sample size is fixed as:

$$\hat{\sigma}^2 = \frac{\left(\sum_i n_{ii}\right)\left(\sum_{\substack{i=j \\ i \neq j}} n_{ij}\right)}{n^3} = \frac{\begin{pmatrix}\text{no.} \\ \text{correct}\end{pmatrix}\begin{pmatrix}\text{no.} \\ \text{errors}\end{pmatrix}}{(\text{total no.})^3}$$

where n is the fixed sample size and n_{ij} is the number of classifications of units assigned to category j whose true category is i. Used directly, this gives an estimate over all categories for the entire contingency table. This is the last entry in the percent standard deviation (%SD) column. The remaining entries were determined by fixing i and estimating the standard deviation for each category i. This estimate says that as the number of samples tested increases and the number of correct classifications increase, the variance of the probability of correct classification decreases, as one would expect.

In order to obtain an initial estimate of performance of the multi-image texture features an experiment was performed on ERTS satellite imagery over Monterey Bay, California, image number 1002-18134 (see Figure 10) taken on July 25, 1972. Using a small set of 64 sampled 32 x 32 subimages and training on 34 of these, 80 per cent of the remaining 30 test samples were correctly classified according to four land-use categories: coastal forest, annual grassland, urban area, and water, as shown in Table 1. This is encouraging since previous accuracy using spatial dependence matrices on band 5 with 64 by 64 subimages over the same general area was only 70.5 per cent as shown in Table 2 (Haralick, 1973).

The ability to obtain good ground truth and several distinct categories in the California data was not the case for an ERTS image over Finney County, Kansas, which was used in later experiments. Approximately a 40 mile by 60 mile section near Garden City, Kansas, on image number 1330-16515 (see Figure 11), taken on June 18, 1973, was processed with initially four categories: grassland, large fields, small fields, and water. Both texture procedures, using the multi-image texture features with 32 by 32 subimages and the single-image texture analysis on MSS band 5 with 64 by 64 subimages, were used on the image. Tables 3 and 4 show the results of classification for distance 1 resolution cells while Tables 5 and 6 show distance 8 results. In both cases the single image classification is higher. However, when both distances 1 and 8 are used together, classification accuracy for both procedures is nearly identical, as shown in Tables 7 and 8, about 70 per cent. Tables 9 and 10 show results using the Bayes Classifier.

This implies that more information is contained in the single-band texture features than the multi-image texture features. In order to add more texture information, a measure of entropy (Kullbach, 1959), given by

$$E = -\tfrac{1}{2} \log |P|$$

where P is the correlation matrix, was added to the cross-band texture feature set. Also, higher order components were appended to each grey tone N-tuple by squaring the grey tones and getting cross-band product terms. Only a few of these were added, increasing the grey tone N-tuple dimensionality from 4 to 8, which results in an increase in the number of feature vector components (elements in the correlation matrix) from 6 to 28 plus the entropy measure. The eight components in each grey tone N-tuple are: MSS Band 5, Band 6, Band 7, $(\text{Band } 5)^2$, $(\text{Band } 6)^2$, $(\text{Band } 7)^2$, (Band 5) x (Band 6), (Band 5) x (Band 7). Figure 13 gives an illustration of a correlation matrix with the feature vector component designation to be used in indicating the components selected for input to the classification programs.

Table 11 shows the contingency table for the cross-band texture features using 9 of the 29 components (1,2,5,6,9,12, 20, 24, and 27 of figure 13) for a portion of the Garden City, Kansas, data. The increase in identification accuracy between the large and small fields results in an increase in overall correct classification, up to 87.1%.

The final classification experiment to test the cross-band texture analysis was made on ERTS image 1021-16333 (see Figure 12) taken on August 13, 1973, over Kansas City. Four land-use categories were chosen: cropland (directly north and south of Kansas City), urban area (Kansas City, Topeka), grassland (southwestern corner of the image), and water (Perry reservoir plus several small lakes). These areas were processed three separate ways:

1) spectrally, 32 by 32 subimages
2) texturally, 64 by 64 subimages on band 5
and 3) cross-band texturally, 32 by 32 subimages.

Because of problems with ground truth and a small data set for water, that category was later dropped. The spectral processing involved obtaining the average grey tone over the subimage for each spectral band, giving 4 components for each feature vector. The textural processing was over a larger subimage than either the spectral or the cross-band textural processing. This was chosen because the 64 by 64 subimages have performed better in the past than the 32 by 32 subimages for the single-image texture analysis. The smaller subimage size was chosen for the cross-band texture processing so that the subimage would more likely be from only one category. The cross-band texture method uses spectral information which the single-image texture procedure does not have available. The smaller subimage size was also chosen for the spectral processing in order to provide an estimate of the amount of spectral information contained in the four MSS bands that is available to the cross-band texture analysis.

Single-image texture analysis was done for distance 1 nearest neighbors at all angles and all of the 17 texture features defined in Appendix I were computed for each subimage. The cross-band texture processing was also done for distance 1 using horizontally adjacent nearest neighbor grey tone N-tuples.

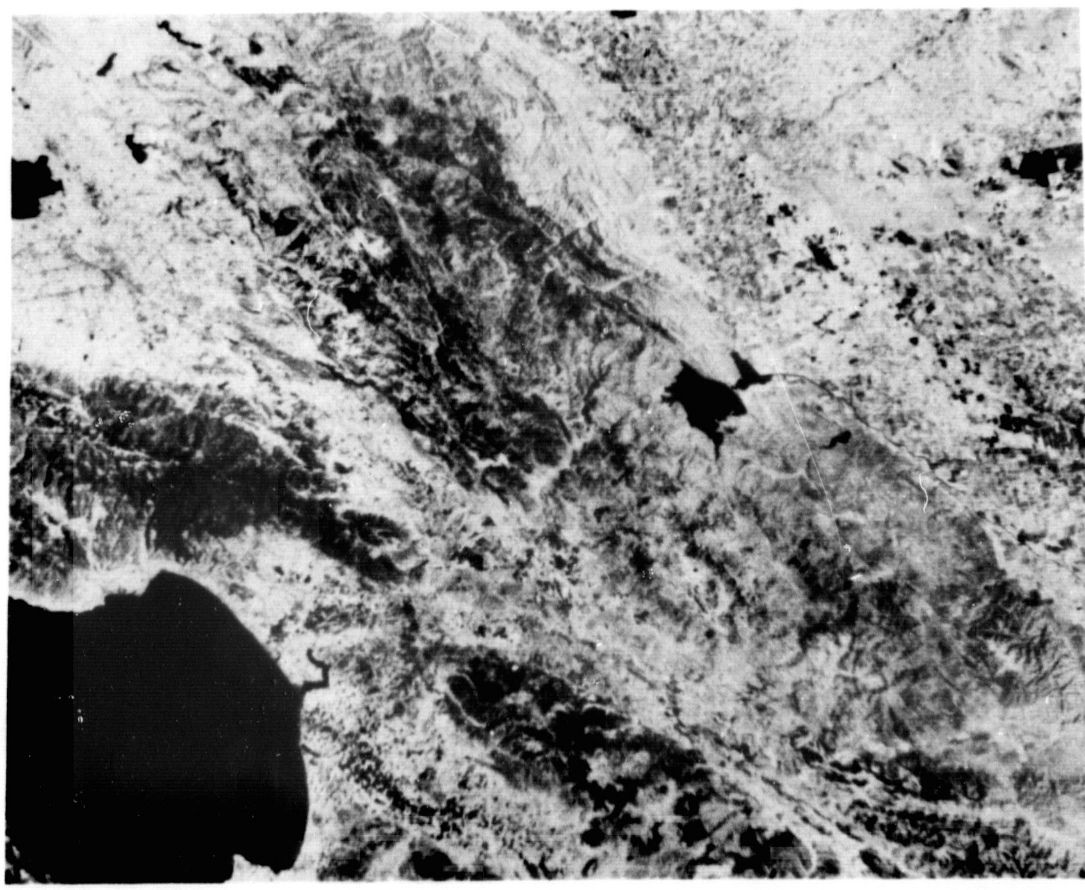


Figure 10. Part of ERTS Image No. 1002-18134 (MSS Band 5) Over Monterey Bay, California, Taken on July 25, 1972.

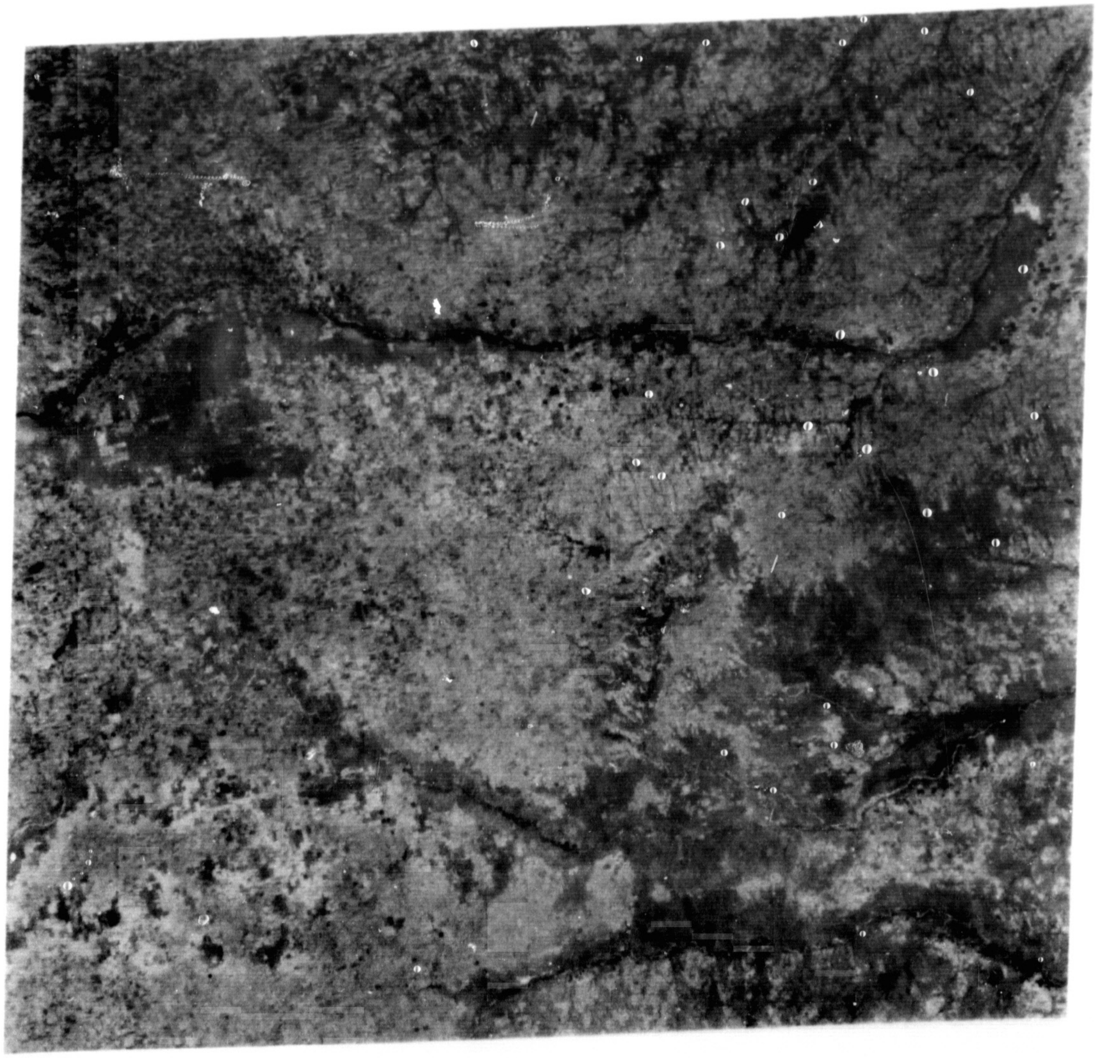


Figure 11. ERTS Image No. 1330-16515 (MSS Band 5) Over Garden City, Kansas, Taken on June 18, 1973.

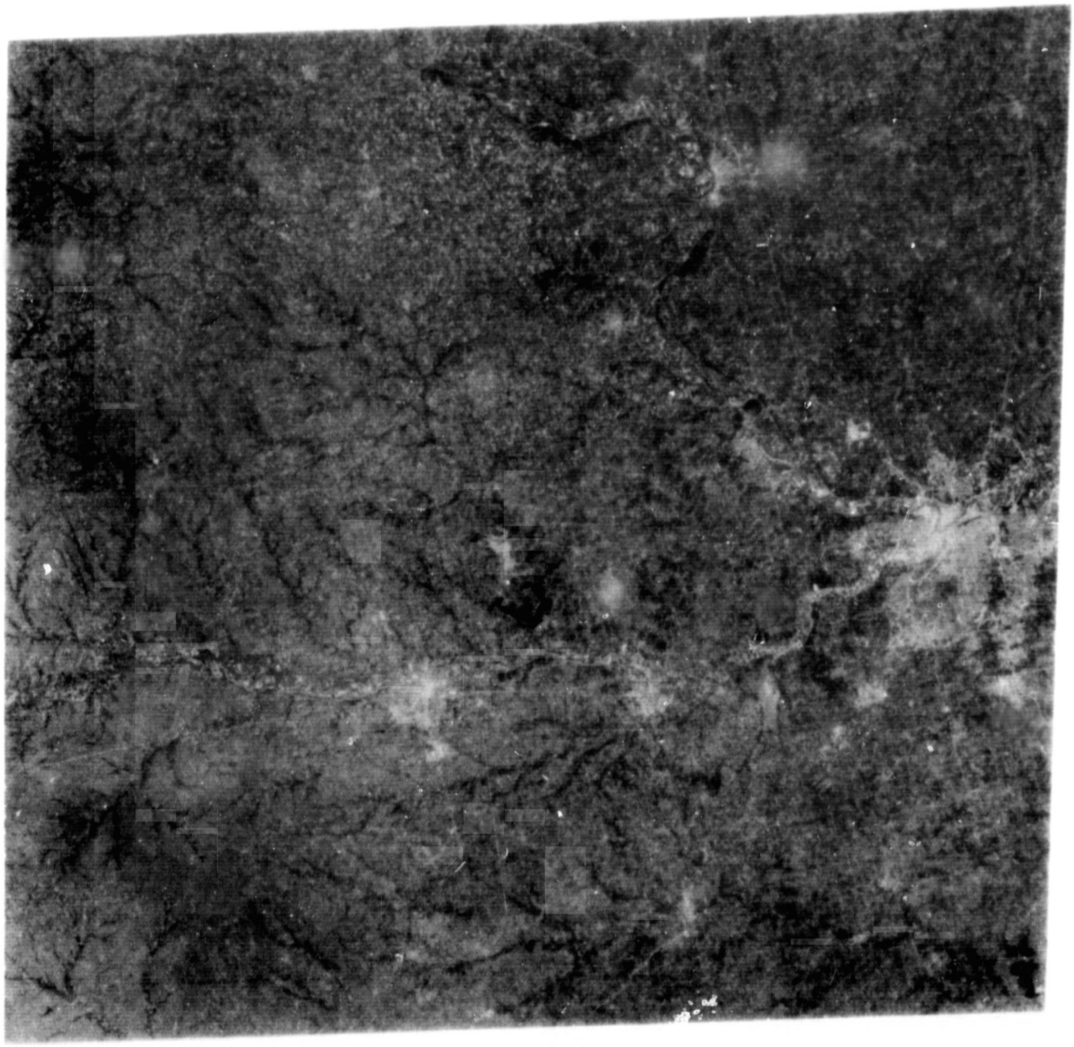


Figure 12. ERTS Image No. 1021-16333 Over (MSS Band 5) Kansas City, Taken on August 13, 1972.

	Band 5	Band 6	Band 7	(Band 5)^2	(Band 6)^2	(Band 7)^2	(Band 5) x(Band 6)	(Band 5) x(Band 7)
Band 5	1							
Band 6	F(2)	1						
Band 7	F(3)	F(9)	1					
(Band 5)^2	F(4)	F(10)	F(15)	1				
(band 6)^2	F(5)	F(11)	F(16)	F(20)	1			
(Band 7)^2	F(6)	F(12)	F(17)	F(21)	F(24)	1		
(Band 5)(Band 6)	F(7)	F(13)	F(18)	F(22)	F(25)	F(27)	1	
(Band 5)(Band 7)	F(8)	F(14)	F(19)	F(23)	F(26)	F(28)	F(29)	1

Entropy = F(1)

Figure 13. Designation of the entropy measure and elements of the correlation matrix as feature vector components for cross-band texture analysis of multi-images.

Table 12 shows the contingency table for all 17 single-image texture features over image 1021-16333 with an average correct classification of 86.2%. Table 13 shows the resulting contingency table over the same image for the 29 cross-band texture features with an increased correct classification of 88.8%. The spectral processing on the four MSS bands for the same image gave four spectral features-the mean grey tone over the subimage for each spectral band. The resulting contingency table using these four spectral features is shown in Table 14. The relatively good performance, 73.9% correct, shows a significant amount of land-use information is contained in the four spectral bands, accounting for the better performance of the cross-band texture analysis over the single-image texture analysis.

It is interesting to see the classification accuracy of the single-image texture greatly improve to 93.8% with the addition of the four spectral features to the original 17 texture features, as shown in Table 15. As expected, the addition of the four spectral features to the first 26 of the 29 cross-band texture features does not improve the classification accuracy as well as with the single-image texture. Table 16 shows the resulting contingency table with an average correct classification of 91.6%. Note the higher estimate for the standard deviation of the probability for correct classification with the textural plus spectral features.

These tables show that the single-image texture procedure does well in extracting texture information, but for this data set, the cross-band texture procedure by itself performs slightly better by extracting more information texturally and spectrally.

The remaining tables show the effect upon classification accuracy of using fewer features. In each case it can be seen that the fewer the number of features used, the lower the average correct classification. Also, it can be seen that the Bayes classifier performs slightly better than the piecewise linear classifier.

Tables 20 thru 24 show that as the number of components is increased, the spectral plus cross-band texture reaches a limiting accuracy of approximately 92%. Ground truth assignment errors could easily account for the remaining 8% error.

VII.1 Summary of Classification Results

It is apparent from these classification experiments that both texture extraction procedures complement each other in that they extract different kinds of texture information. When used without spectral features, the cross-band texture procedure performed better. However, when the spectral features were added, the single-image texture plus spectral features performed better than the cross-band texture plus spectral features. This indicates that the cross-band texture procedure does well in extracting more information texturally and spectrally than the single-image texture procedure. However, when the spectral information is made available to the single-image texture procedure, it performs better than the cross-band texture plus spectral features.

Programs used in these experiments can be found in Appendix IV. The single-image texture programs are under the Texture Analysis Programs with mainline MAINLN. And the cross-band texture programs are in the Cross-Band Texture Analysis Programs with mainline SPECTR. RCLASS is the mainline program for the piecewise linear classifier. Some of the cross-band texture feature components can be seen in Figure 14. Figure 14a shows the ground truth assignment for the image 1021-16333 (Figure 12) for the area near Kansas City. At the bottom of the image, grassland was inserted to give an idea of how well the features separate the three categories: cropland, urban area, and grassland.

VIII. CONCLUSION

The procedure developed here for the extraction of texture information from ERTS multi-images gives encouraging results. The classification experiments show that the cross-band texture procedure can be used successfully in automatic land-use classification of multi-images over Kansas.

The cross-band texture procedure is a natural extension of the previous single-image texture extraction procedure based upon angular nearest neighbor grey tone spatial dependence matrices. It retains the power of the previous approach to texture by characterizing the spatial inter-relationships, or co-occurrences, of the grey tone N-tuples present in a texture pattern in such a way as to be invariant under linear grey tone transformations. And both procedures are simple to employ, economical, and require a minimum of core-storage (see Figure 15).

Both procedures complement each other by extracting different kinds of textural information with the cross-band texture procedure using the cross-band spectral information contained in ERTS multi-images. Results indicate that the cross-band texture procedure does well by extracting more information texturally and spectrally than the single-image texture procedure when used alone. However, when the spectral information is made available to the single-image texture procedure, it performs better than the cross-band texture plus spectral features in classifying texturally distinct land-use categories from ERTS multi-images over Kansas.

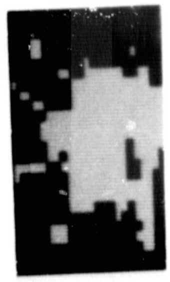


Figure 14-a. Ground Truth Map

Figure 14-b. F(16)

Figure 14-e. F(29)

Figure 14-d. F(24)

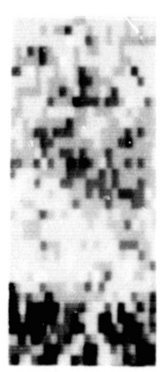


Figure 14-c. F(20)

Figure 14. Examples of Cross-Band Texture Feature Vector Components; Resolution Cell Size is 32 x 32

	Core Required	Total Processing Time	Total Job Cost	Cost Per Resolution Cell
Single-Image Texture (64×64)	43K	0.788HR	$223.	$0.000252
Cross-Band Texture (32×32)	25K	0.884HR	$216.	$0.000244

Figure 15. Comparison of the Performance of the Single-Image Texture Analysis Programs on MSS Band 5 with the Cross-Band Texture Analysis Programs on MSS Bands 4 thru 7 in Processing the Same Area, Near Kansas City, on ERTS Image No. 1021-16333 using a Honeywell 635 Computer.

		Assigned Category					Errors of Commission		
		Coastal Forest	Annual Grassland	Urban Area	Water	Total	#Err	%Err	%SD
True Category	Coastal Forest	5	0	5	0	10	5	50.	15.8
	Annual Grassland	0	7	0	0	7	0	0.	0.
	Urban Area	0	1	7	0	8	1	14.3	11.7
	Water	0	0	0	5	5	0	0.	0.
	Total	5	8	12	5	30		21.4	7.3
Errors of Commission	#Errors	0	1	7	0				
	%Error	0.	12.5	58.3	0.	17.7			

TABLE 1. CONTINGENCY TABLE FOR IMAGE NO. 1002-18134 USING MULTI-IMAGE TEXTURAL FEATURES, 6 COMPONENTS, AVERAGE CORRECT CLASSIFICATION ON TEST SET=80.0%*

FEATURES	NO. OF SAMPLES IN TRAINING SET	NO. OF SAMPLES IN TEST SET	OVERALL ACCURACY OF TEST SET
MULTI-IMAGE TEXTURAL	34	30	80.0%
SINGLE-IMAGE TEXTURAL	260	172	70.5%

TABLE 2. RESULTS OF LAND-USE CLASSIFICATION EXPERIMENTS FROM ERTS IMAGE NO. 1002-18134 OVER MONTEREY BAY, CALIFORNIA.

*UNLESS STATED OTHERWISE, ALL CONTINGENCY TABLES ARE DETERMINED USING THE PIECEWISE LINEAR CLASSIFICATION PROGRAMS (RCLASS).

		ASSIGNED CATEGORY					ERRORS OF COMMISSION		
		GRASSLAND	LARGE FIELDS	SMALL FIELDS	WATER	TOTAL	#ERR	%ERR	%SD
TRUE CATEGORY	GRASSLAND	47	11	5	0	63	16	25.4	5.5
	LARGE FIELDS	3	142	32	1	178	36	20.2	3.0
	SMALL FIELDS	17	46	52	0	115	63	31.3	4.6
	WATER	0	1	6	1	8	2	25.0	11.7
	TOTAL	67	200	95	2	364		25.5	2.5
ERRORS OF COMMISSION	#ERRORS	20	58	43	1				
	%ERROR	29.9	29.0	45.3	50.	38.6			

TABLE 3. CONTINGENCY TABLE FOR IMAGE NO. 1330-16515 USING MULTI-IMAGE TEXTURAL FEATURES, DISTANCE 1, 6 COMPONENTS. AVERAGE CORRECT CLASSIFICATION = 66.5%

EEATURES	NO. OF SAMPLES IN TRAINING SET	NO. OF SAMPLES IN TEST SET	OVERALL ACCURACY OF TEST SET
MULTI-IMAGE TEXTURAL	548	364	66.5%
SINGLE-IMAGE TEXTURAL	140	88	76%

TABLE 4. RESULTS OF LAND-USE CLASSIFICATION EXPERIMENTS FOR ERTS IMAGE NO. 1330-16515 AT DISTANCE 1.

		ASSIGNED CATEGORY					ERRORS OF COMMISSION		
		GRASSLAND	LARGE FIELDS	SMALL FIELDS	WATER	TOTAL	#ERR	%ERR	%SD
TRUE CATEGORY	GRASSLAND	32	9	22	0	63	31	49.2	6.3
	LARGE FIELDS	7	147	24	0	178	31	17.4	2.8
	SMALL FIELDS	18	44	53	0	115	62	53.9	4.6
	WATER	0	4	4	0	8	4	50.0	0.
	TOTAL	57	204	103	0	364		42.6	2.5
ERRORS OF COMMISSION	#ERRORS	25	57	50	0				
	%ERROR	43.9	27.9	48.5	0.	30.1			

TABLE 5. CONTINGENCY TABLE FOR IMAGE NO. 1330-16515 USING MULTI-IMAGE TEXTURAL FEATURES, DISTANCE 8, 6 COMPONENTS. AVERAGE CORRECT CLASSIFICATION = 63.7%

EEATURES	NO. OF SAMPLES IN TRAINING SET	NO. OF SAMPLES IN TEST SET	OVERALL ACCURACY OF TEST SET
MULTI-IMAGE TEXTURAL	548	364	63.7%
SINGLE-IMAGE TEXTURAL	140	88	76%

TABLE 6. RESULTS OF LAND-USE CLASSIFICATION EXPERIMENTS FOR ERTS IMAGE NO. 1330-16515 AT DISTANCE 8.

		ASSIGNED CATEGORY					ERRORS OF COMMISSION		
		GRASSLAND	LARGE FIELDS	SMALL FIELDS	WATER	TOTAL	#ERR	%ERR	%SD
TRUE CATEGORY	GRASSLAND	52	7	4	0	63	11	17.5	4.8
	LARGE FIELDS	2	145	31	0	178	33	18.5	2.9
	SMALL FIELDS	12	42	61	0	115	54	47.0	4.7
	WATER	0	1	6	1	8	7	87.5	11.7
	TOTAL	66	195	102	1	364		42.6	2.4
ERRORS OF COMMISSION	#ERRORS	14	50	41	0				
	%ERROR	21.2	25.6	40.2	0.	21.8			

TABLE 7. CONTINGENCY TABLE FOR IMAGE NO. 1330-16515 USING MULTI-IMAGE TEXTURAL FEATURES, DISTANCES 1 AND 8, 12 COMPONENTS. AVERAGE CORRECT CLASSIFICATION =71%

EEATURES	NO. OF SAMPLES IN TRAINING SET	NO. OF SAMPLES IN TEST SET	OVERALL ACCURACY OF TEST SET
MULTI-IMAGE TEXTURAL	548	364	71%
SINGLE-IMAGE TEXTURAL	140	88	73%

TABLE 8. RESULTS OF LAND-USE CLASSIFICATION EXPERIMENTS FOR ERTS IMAGE NO. 1330-16515 USING BOTH DISTANCES 1 AND 8.

		ASSIGNED CATEGORY					ERRORS OF COMMISSION		
		GRASSLAND	LARGE FIELDS	SMALL FIELDS	WATER	TOTAL	#ERR	%ERR	%SD
TRUE CATEGORY	GRASSLAND	49	4	6	1	60	11	18.3	5.0
	LARGE FIELDS	6	150	27	1	184	34	18.5	2.9
	SMALL FIELDS	10	34	67	2	113	46	40.7	4.6
	WATER	0	4	0	4	8	4	31.9	17.7
	TOTAL	65	192	100	8	365		31.9	2.3
ERRORS OF COMMISSION	#ERRORS	16	42	33	4				
	%ERROR	24.6	21.9	33.0	50.0	32.4			

TABLE 9. CONTINGENCY TABLE FOR IMAGE NO. 1330-16515 USING MULTI-IMAGE TEXTURAL FEATURES, DISTANCE 1,6 COMPONENTS. AVERAGE CORRECT CLASSIFICATION = 74.0% USING BAYES.

		ASSIGNED CATEGORY					ERRORS OF COMMISSION		
		GRASSLAND	LARGE FIELDS	SMALL FIELDS	WATER	TOTAL	#ERR	%ERR	%SD
TRUE CATEGORY	GRASSLAND	31	10	19	0	60	29	48.3	6.5
	LARGE FIELDS	5	140	38	1	184	44	23.9	3.1
	SMALL FIELDS	12	46	51	4	113	62	54.9	4.7
	WATER	1	4	2	1	8	7	87.5	11.7
	TOTAL	49	200	110	6	365		53.7	2.6
ERRORS OF COMMISSION	#ERRORS	18	50	59	5				
	%ERROR	36.7	25.0	53.6	83.3	49.7			

TABLE 10. CONTINGENCY TABLE FOR IMAGE NO.1330-16515 USING MULTI-IMAGE TEXTURAL FEATURES, DISTANCE 8,6 COMPONENTS. AVERAGE CORRECT CLASSIFICATION =61.1% USING BAYES.

		ASSIGNED CATEGORY				ERRORS OF COMMISSION		
		GRASSLAND	LARGE FIELDS	SMALL FIELDS	TOTAL	#ERR	%ERR	%SD
TRUE CATEGORY	GRASSLAND	35	3	0	38	3	7.9	4.4
	LARGE FIELDS	2	12	2	16	4	25.0	10.8
	SMALL FIELDS	0	2	14	16	2	12.5	8.3
	TOTAL	37	17	16	70		15.1	4.0
ERRORS OF COMMISSION	#ERRORS	2	5	2				
	%ERROR	5.4	29.4	12.5	15.8			

TABLE 11. CONTINGENCY TABLE FOR IMAGE NO. 1330-16515 USING 9 OF THE 29 CROSS-BAND TEXTURAL FEATURES, DISTANCE 1. #TRAIN=103, #TEST=70, AVERAGE CORRECT CLASSIFICATION = 87.1% USING THE BAYES CLASSIFIER. THE 9 FEATURE COMPONENTS USED WERE: 1,2,5,6,9,12,20,24,27.

		ASSIGNED CATEGORY				ERRORS OF COMMISSION		
		CROPLAND	URBAN	GRASSLAND	TOTAL	#ERR	%ERR	%SD
TRUE CATEGORY	CROPLAND	61	2	1	64	3	4.7	2.6
	URBAN	6	18	0	24	6	25.0	8.8
	GRASSLAND	6	1	21	28	7	25.0	8.2
	TOTAL	73	21	22	116		18.2	3.2
ERRORS OF OMMISSION	#ERRORS	12	3	1				
	%ERROR	16.4	14.2	4.5	11.7			

TABLE 12. CONTINGENCY TABLE FOR ERTS IMAGE 1021-16333 USING ALL 17 SINGLE-BAND TEXTURE FEATURES AT DISTANCE 1. #TRAIN=178, #TEST=116, AVERAGE CORRECT CLASSIFICATION = 86.2%

		ASSIGNED CATEGORY				ERRORS OF COMMISSION		
		CROPLAND	URBAN	GRASSLAND	TOTAL	#ERR	%ERR	%SD
TRUE CATEGORY	CROPLAND	150	15	3	168	18	10.7	2.4
	URBAN	11	81	0	92	11	12.0	3.4
	GRASSLAND	11	2	103	116	13	11.2	2.9
	TOTAL	172	98	106	376		11.3	1.6
ERRORS OF OMMISSION	#ERRORS	22	17	3				
	%ERROR	12.8	17.3	2.8	11.0			

TABLE 13. CONTINGENCY TABLE FOR ERTS IMAGE 1021-16333 USING ALL 29 CROSS-BAND TEXTURE FEATURES AT DISTANCE 1. #TRAIN = 569, #TEST = 376, AVERAGE CORRECT CLASSIFICATION = 88.8%

		CROPLAND	URBAN	GRASSLAND	TOTAL	#ERR	%ERR	%SD
		ASSIGNED CATEGORY				ERRORS OF COMMISSION		
TRUE CATEGORY	CROPLAND	138	9	22	169	37	21.9	3.0
	URBAN	8	53	30	91	38	41.8	5.2
	GRASSLAND	20	8	84	112	28	25.0	4.1
	TOTAL	166	70	136	372		29.6	2.3
ERRORS OF OMMISSION	#ERRORS	28	17	52				
	%ERROR	16.9	24.3	38.2	26.5			

TABLE 14. CONTINGENCY TABLE FOR ERTS IMAGE 1021-16333 USING SPECTRAL FEATURES ONLY. #TRAIN=564, #TEST=372 AVERAGE CORRECT CLASSIFICATION = 73.9%

		CROPLAND	URBAN	GRASSLAND	TOTAL	#ERR	%ERR	%SD
		ASSIGNED CATEGORY				ERRORS OF COMMISSION		
TRUE CATEGORY	CROPLAND	62	2	0	64	2	3.1	2.2
	URBAN	4	19	1	24	5	20.8	8.3
	GRASSLAND	0	0	24	24	0	0.0	0.0
	TOTAL	66	21	25	112		8.0	2.3
ERRORS OF OMMISSION	#ERRORS	4	2	1				
	%ERROR	6.1	9.5	4.0	6.5			

TABLE 15. CONTINGENCY TABLE FOR ERTS IMAGE 1021-16333 USING ALL 17 SINGLE-IMAGE TEXTURE PLUS THE 4 SPECTRAL FEATURES, 21 COMPONENTS. #TRAIN=175, #TEST=112 AVERAGE CORRECT CLASSIFICATION = 93.8%

		ASSIGNED CATEGORY				ERRORS OF COMMISSION		
		CROPLAND	URBAN	GRASSLAND	TOTAL	#ERR	%ERR	%SD
TRUE CATEGORY	CROPLAND	155	12	1	168	13	7.7	2.1
	URBAN	12	80	0	92	12	13.0	3.5
	GRASSLAND	4	2	103	109	6	5.5	2.2
	TOTAL	171	94	104	369		8.7	1.4
ERRORS OF OMMISSION	#ERRORS	16	12	1				
	%ERROR	9.4	12.8	1.0	7.7			

TABLE 16. CONTINGENCY TABLE FOR ERTS IMAGE 1021-16333 USING FIRST 26 CROSS-BAND TEXTURE PLUS THE 4 SPECTRAL FEATURES, 30 COMPONENTS. #TRAIN=567,#TEST=369
AVERAGE CORRECT CLASSIFICATION = 91.6%

		ASSIGNED CATEGORY				ERRORS OF COMMISSION		
		CROPLAND	URBAN	GRASSLAND	TOTAL	#ERR	%ERR	%SD
TRUE CATEGORY	CROPLAND	60	1	3	64	4	6.2	3.0
	URBAN	8	16	0	24	8	33.3	9.6
	GRASSLAND	5	1	22	28	6	21.4	7.8
	TOTAL	73	18	25	116		20.3	3.4
ERRORS OF OMMISSION	#ERRORS	13	2	3				
	%ERROR	17.8	11.1	12.0	13.6			

TABLE 17. CONTINGENCY TABLE FOR ERTS IMAGE 1021-16333 USING FIRST 8 OF THE 17 SINGLE-IMAGE TEXTURE FEATURES. #TRAIN=178,#TEST=116, AVERAGE CORRECT CLASSIFICATION = 84.5%

		Assigned Category				Errors of Commission		
		CROPLAND	URBAN	GRASSLAND	TOTAL	#ERR	%ERR	%SD
TRUE CATEGORY	CROPLAND	138	22	8	168	30	17.9	3.0
	URBAN	21	68	3	92	24	26.1	4.6
	GRASSLAND	20	5	91	116	25	21.6	3.8
	TOTAL	179	95	102	376		21.9	2.1
ERRORS OF OMMISSION	#ERRORS	41	27	11				
	%ERROR	22.9	28.4	10.8	20.7			

TABLE 18. CONTINGENCY TABLE FOR ERTS IMAGE 1021-16333 USING FIRST 9 OF THE 29 CROSSBAND TEXTURE FEATURES. #TRAIN=569, #TEST=376, AVERAGE CORRECT CLASSIFICATION = 79.0%

		Assigned Category				Errors of Commission		
		CROPLAND	URBAN	GRASSLAND	TOTAL	#ERR	%ERR	%SD
TRUE CATEGORY	CROPLAND	142	25	4	171	29	17.0	2.9
	URBAN	20	66	5	91	25	27.5	4.7
	GRASSLAND	12	5	95	112	17	15.2	3.4
	TOTAL	174	96	104	374		19.9	2.0
ERRORS OF OMMISSION	#ERRORS	32	30	9				
	%ERROR	18.4	31.3	8.7	19.5			

TABLE 19. CONTINGENCY TABLE FOR ERTS IMAGE 1021-16333 USING FIRST 9 OF THE 29 CROSS-BAND TEXTURE FEATURES. #TRAIN=562, #TEST=374, AVERAGE CORRECT CLASSIFICATION = 81.0% USING THE BAYES CLASSIFIER.

		CROPLAND	URBAN	GRASSLAND	TOTAL	#ERR	%ERR	%SD
		ASSIGNED CATEGORY				ERRORS OF COMMISSION		
TRUE CATEGORY	CROPLAND	158	6	2	166	8	4.8	1.7
	URBAN	13	58	18	89	31	34.8	5.1
	GRASSLAND	9	10	93	112	19	17.0	3.5
	TOTAL	180	74	113	367		18.9	1.9
ERRORS OF OMMISSION	#ERRORS	22	16	20				
	%ERROR	12.2	21.6	17.7	17.2			

TABLE 20. CONTINGENCY TABLE FOR ERTS IMAGE 1021-16333 USING FIRST 6 OF THE 29 CROSS-BAND TEXTURE PLUS THE 4 SPECTRAL FEATURES, #TRAIN=569, #TEST=367, AVERAGE CORRECT CLASSIFICATION = 84.2%

		CROPLAND	URBAN	GRASSLAND	TOTAL	#ERR	%ERR	%SD
		ASSIGNED CATEGORY				ERRORS OF COMMISSION		
TRUE CATEGORY	CROPLAND	61	2	1	64	3	4.7	2.6
	URBAN	3	19	2	24	5	20.8	8.3
	GRASSLAND	0	0	24	24	0	0.0	0.0
	TOTAL	64	21	27	112		8.5	2.4
ERRORS OF OMMISSION	#ERRORS	3	2	3				
	%ERROR	4.7	9.5	11.1	8.4			

TABLE 21. CONTINGENCY TABLE FOR ERTS IMAGE 1021-16333 USING FIRST 9 OF THE 17SINGLE-IMAGE TEXTURE PLUS THE 4 SPECTRAL FEATURES, #TRAIN=175,#TEST=112, AVERAGE CORRECT CLASSIFICATION = 92.9%

		ASSIGNED CATEGORY				ERRORS OF COMMISSION		
		CROPLAND	URBAN	GRASSLAND	TOTAL	#ERR	%ERR	%SD
TRUE CATEGORY	CROPLAND	161	7	0	168	7	4.2	1.5
	URBAN	15	77	0	92	15	16.3	3.9
	GRASSLAND	9	6	97	112	15	13.4	3.2
	TOTAL	185	90	97	372		11.3	1.6
ERRORS OF OMMISSION	#ERRORS	24	13	0				
	%ERROR	13.0	14.4	0.0	9.1			

TABLE 22. CONTINGENCY TABLE FOR ERTS IMAGE 1021-16333 USING FIRST 11 OF THE 29 CROSS-BAND TEXTURE PLUS THE 4 SPECTRAL FEATURES, #TRAIN=564, #TEST=372, AVERAGE CORRECT CLASSIFICATION = 90.1%

		ASSIGNED CATEGORY				ERRORS OF COMMISSION		
		CROPLAND	URBAN	GRASSLAND	TOTAL	#ERR	%ERR	%SD
TRUE CATEGORY	CROPLAND	159	8	0	167	8	4.8	1.7
	URBAN	15	76	1	92	16	17.4	4.0
	GRASSLAND	7	5	100	112	12	10.7	2.9
	TOTAL	181	89	101	371		11.0	1.5
ERRORS OF OMMISSION	#ERRORS	22	13	1				
	%ERROR	12.2	14.6	1.0	9.3			

TABLE 23. CONTINGENCY TABLE FOR ERTS IMAGE 1021-16333 USING FIRST 16 OF THE 29 CROSS-BAND TEXTURE PLUS THE 4 SPECTRAL FEATURES, #TRAIN=565, #TEST=371, AVERAGE CORRECT CLASSIFICATION = 90.3%

		ASSIGNED CATEGORY				ERRORS OF COMMISSION		
		CROPLAND	URBAN	GRASSLAND	TOTAL	#ERR	%ERR	%SD
TRUE CATEGORY	CROPLAND	156	9	1	166	10	6.0	1.8
	URBAN	13	79	0	92	13	14.1	3.6
	GRASSLAND	6	1	103	110	7	6.4	2.3
	TOTAL	175	89	104	368	.	8.8	1.4
ERRORS OF OMMISSION	#ERRORS	19	10	1				
	%ERROR	10.9	11.2	1.0	7.7			

TABLE 24. CONTINGENCY TABLE FOR ERTS IMAGE 1021-16333 USING FIRST 21 OF THE 29 CROSS-BAND TEXTURE PLUS THE 4 SPECTRAL FEATURES, #TRAIN=568, #TEST=368, AVERAGE CORRECT CLASSIFICATION = 91.9%

APPENDIX I

TEXTURAL FEATURES OBTAINED FROM THE GREY TONE DEPENDENCE MATRIX

In this appendix, we define 17 textural features which are computed for each of the four angular grey tone dependence matrices.

The following notation will be used in defining the 17 textural features.

$P(i,j)$ - $(i,j)^{th}$ entry in a particular grey tone dependence matrix.

$\left.\begin{array}{l}\frac{P_x(i)}{\#R} \\ \frac{P_y(i)}{\#R}\end{array}\right\}$ - i^{th} entry in the marginal distributions of $P(i,j)$ obtained by summing rows and columns of $P(i,j)$ respectively.

$\#R$ - number of resolution cell pairs which were considered in computing the entries in $P(i,j)$.

N_g - number of distinct grey tone values in the image.

μ - mean of $P(i,j)/\#R$.

$\frac{P_{x+y}}{\#R}(i)$ - i^{th} entry in the distribution of the sum of grey tones of neighboring resolution cells.

$\frac{P_{x-y}}{\#R}(i)$ - i^{th} entry in the distribution of the absolute differences in the grey tones of neighboring resolution cells.

TEXTURAL FEATURES

1. Angular Second Moment:

$$f_1 = \sum_{i=1}^{N_g} \sum_{j=1}^{N_g} \left\{ \frac{P(i,j)}{\#R} \right\}^2$$

2. Entropy:

$$f_2 = \sum_{i=1}^{N_g} \sum_{j=1}^{N_g} - \left(\frac{P(i,j)}{\#R} \right) \log \left(\frac{P(i,j)}{\#R} \right)$$

3. Correlation:

$$f_3 = \frac{\sum_{i=1}^{N_g} \sum_{j=1}^{N_g} ij \left\{ \frac{P(i,j)}{\#R} \right\} - \mu_x \mu_y}{\sigma_x, \sigma_y}$$

where μ_x and σ_x are the mean and standard deviation of P_x, and μ_y and σ_y are the mean and standard deviation of P_y.

4. Sum of Squares on x:

$$f_4 = \sum_{i=1}^{N_g} \sum_{j=1}^{N_g} (i-\mu)^2 \left\{ \frac{P(i,j)}{\#R} \right\}$$

5. Product Moment:

$$f_5 = \sum_{i=1}^{N_g} \sum_{j=1}^{N_g} (i-\mu)(j-\mu) \left\{ \frac{P(i,j)}{\#R} \right\}$$

6. Inverse Moment:

$$f_6 = \sum_{i=1}^{N_g} \sum_{j=1}^{N_g} \frac{1}{1+(i-j)^2} \left\{ \frac{P(i,j)}{\# R} \right\}$$

7. Difference Moment:

$$f_7 = \sum_{i=1}^{N_g} \sum_{j=1}^{N_g} (i-j)^2 \left\{ \frac{P(i,j)}{\# R} \right\}$$

8. Sum Average:

$$f_8 = \sum_{i=1}^{2N_g} i \left\{ \frac{P_{x+y}(i)}{\# R} \right\}$$

9. Mean:

$$f_9 = \frac{1}{N_g} \sum_{i=1}^{N_g} \left\{ \frac{P(i,j)}{\# R} \right\}$$

10. Sum Variance:

$$f_{10} = \text{variance of } P_{x+y} / \# R$$

11. Sum Entropy:

$$f_{11} = \sum_{i=1}^{2N_g} - \left\{ \frac{P_{x+y}(i)}{\# R} \right\} \log \left\{ \frac{P_{x+y}(i)}{\# R} \right\}$$

12. Contrast:

$$f_{12} = \sum_{n=0}^{N_g-1} n^2 \left\{ \sum_{\substack{i=1 \\ |i-j|=n}}^{N_g} \sum_{j=1}^{N_g} \left(\frac{P(i,j)}{\# R} \right) \right\}$$

13. Difference Variance:

$$f_{13} = \text{variance of } \frac{P_{x-y}}{\#R}$$

14. Difference Entropy:

$$f_{14} = \sum_{i=0}^{N_g-1} - \left\{\frac{P_{x-y}(i)}{\#R}\right\} \log \left\{\frac{P_{x-y}(i)}{\#R}\right\}$$

15, 16, 17. Additional Measures of Correlation:

$$f_{15} = \frac{HXY - HXY1}{\max HX, HY}$$

$$f_{16} = \sqrt{1 - \exp[-2.0(HXY2-HXY)]}$$

$$f_{17} = \sqrt{\text{second largest eigenvalue of } QQ^T}\ ^*$$

where HX and HY are the entropies of the marginals of the transition matrix before quantization, HXY is the entropy of the transition matrix, and HXY2 is the entropy of the product distribution of the marginals before quantization;

$$Q(i,j) = P(i,j)/\sqrt{P_x(i)P_y(j)} \,.$$

---

*f_{17} is the maximal correlation coefficient.

APPENDIX II

N-DIMENSIONAL SPHERICAL COORDINATE SYSTEMS AND ELLIPSOIDALLY SYMMETRIC DISTRIBUTIONS

We illustrate the N-dimensional spherical coordinate system in the calculation of the volume of the N-dimensional hypersphere. Next we show how suitable functions can be used to define ellipsoidally symmetric density functions and we determine the normalizing constant for any function. Finally, we show that for any ellipsoidally symmetric density f ($\sqrt{x' A x}$), the matrix A is proportional to the inverse covariance matrix of x and we determine the constant of proportionality.

II.1 Volume of an N-dimensional Hypersphere

Let V be the volume of a N-dimensional hypersphere of radius r_o. By definition

$$v = \underset{\sqrt{\sum_{i=1}^{N} x_i^2} \leq r_o}{\int \int \cdots \int} dx_1\, dx_2 \ldots dx_N$$

To evaluate this N-fold integral, we transform to spherical coordinates.

$$x_1 = r \cos \theta_1 \cos \theta_2 \ldots \cos \theta_{N-3} \cos \theta_{N-2} \cos \theta_{N-1}$$

$$x_2 = r \cos \theta_1 \cos \theta_2 \ldots \cos \theta_{N-3} \cos \theta_{N-2} \sin \theta_{N-1}$$

$$x_3 = r \cos \theta_1 \cos \theta_2 \ldots \cos \theta_{N-3} \sin \theta_{N-2}$$

$$\vdots$$

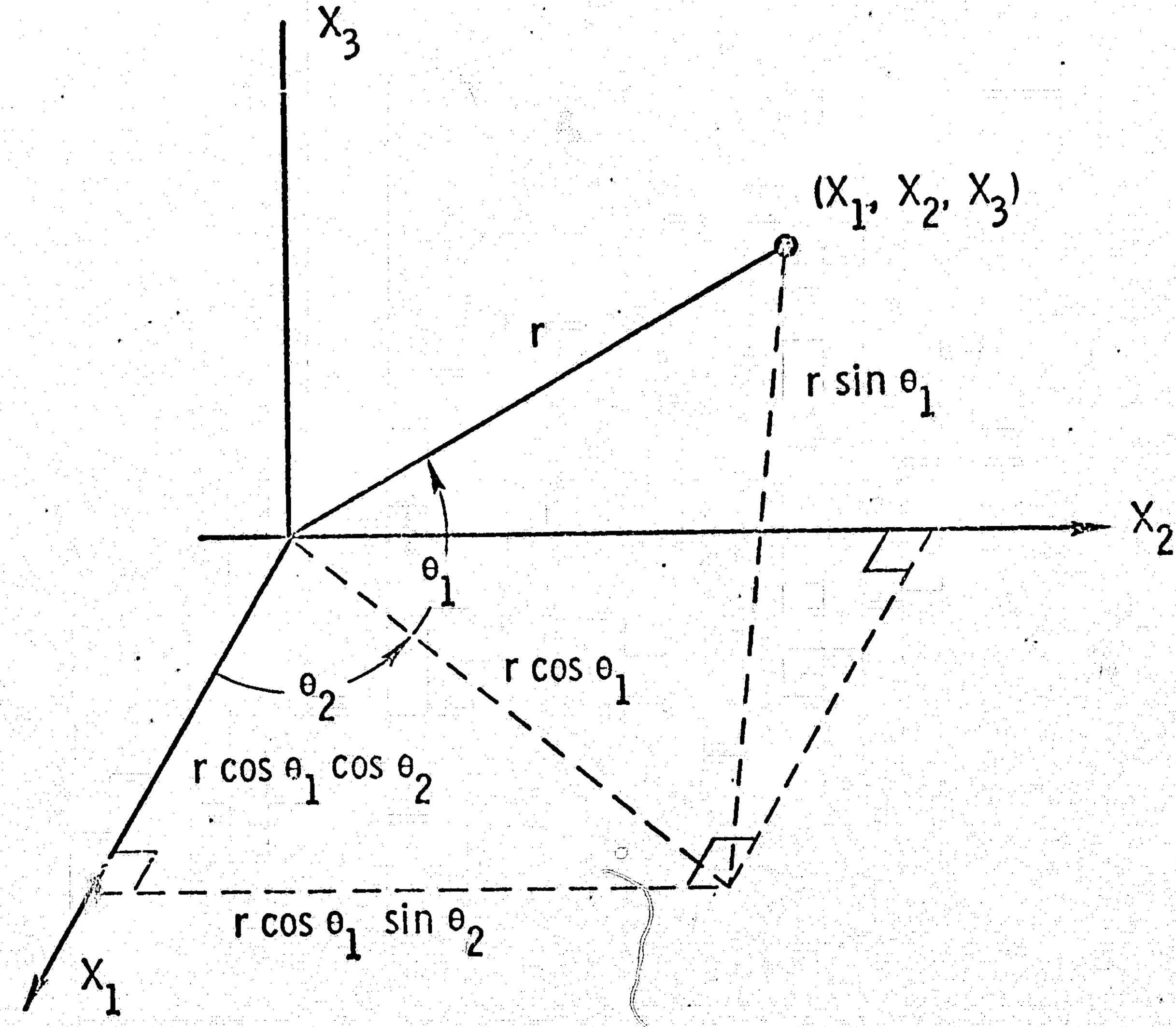
$$x_j = r \cos \theta_1 \cos \theta_2 \ldots \cos \theta_{N-j} \sin \theta_{N-j}$$

$$\vdots$$

$$x_N = r \sin \theta_1$$

Figure 16 illustrates the geometry of the spherical coordinate system we use for a 3-dimensional system.

Three-Dimensional Spherical Coordinate System



$$\left.\begin{aligned} x_1 &= r \cos \theta_1 \cos \theta_2 \\ x_2 &= r \cos \theta_1 \sin \theta_2 \\ x_3 &= r \sin \theta_1 \end{aligned}\right\}$$

Transformation between rectangular coordinate system and spherical coordinate system.

Figure 16 Three-Dimensional Spherical Coordinate System

The Jacobian J of this transformation is defined by the determinant J.

$$J = \begin{vmatrix} \dfrac{\partial x_1}{\partial r} & \dfrac{\partial x_2}{\partial r} & \cdots & \dfrac{\partial x_N}{\partial r} \\ \dfrac{\partial x_1}{\partial \theta_1} & \dfrac{\partial x_2}{\partial \theta_1} & \cdots & \dfrac{\partial x_N}{\partial \theta_1} \\ \vdots & & & \\ \dfrac{\partial x_1}{\partial \theta_{N-1}} & \dfrac{\partial x_2}{\partial \theta_{N-1}} & \cdots & \dfrac{\partial x_N}{\partial \theta_{N-1}} \end{vmatrix}$$

$$= \begin{vmatrix} \cos\theta_1 \cos\theta_2 \cdots \cos\theta_{N-1} & \cos\theta_1 \cos\theta_2 \cdots \cos\theta_{N-2} \sin\theta_{N-1} & \cdots & \sin\theta_1 \\ -r \sin\theta_1 \cos\theta_2 \cdots \cos\theta_{N-1} & -r \sin\theta_1 \cos\theta_2 \cdots \cos\theta_{N-2} \sin\theta_{N-1} & \cdots & r\cos\theta_1 \\ -r \cos\theta_1 \sin\theta_1 \cdots \cos\theta_{N-1} & -r \cos\theta_1 \sin\theta_2 \cdots \cos\theta_{N-2} \sin\theta_{N-1} & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ -r \cos\theta_1 \cos\theta_2 \cdots \sin\theta_{N-1} & r \cos\theta_1 \cos\theta_2 \cdots \cos\theta_{N-2} \cos\theta_{N-1} & \cdots & 0 \end{vmatrix}$$

To find the value of the Jacobian, factor r out of the last (N-1) rows and from each column factor out its first entry.

$$J = r^{N-1} \cos^{N-1}\theta_1 \cos^{N-2}\theta_2 \cdots \cos\theta_{N-1} \sin\theta_1 \sin\theta_2 \cdots \sin\theta_{N-1}$$

$$\begin{vmatrix}
1 & 1 & 1 & \cdots & 0 \\
-\tan\theta_1 & -\tan\theta_1 & -\tan\theta_1 & \cdots & \cot\theta_1 \\
-\tan\theta_2 & -\tan\theta_2 & -\tan\theta_2 & \cdots & 0 \\
\vdots & \vdots & \vdots & \cdots & \vdots \\
\vdots & \vdots & -\tan\theta_{N-3} & \cdots & 0 \\
\cdot\cdot & -\tan\theta_{N-2} & \cot\theta_{N-2} & \cdots & 0 \\
-\tan\theta_{N-1} & \cot\theta_{N-1} & 0 & \cdots & 0
\end{vmatrix}$$

Subtracting column 2 from column 1, column 3 from column 2, ... column N from column N-1 there results

$$\begin{vmatrix}
0 & 0 & 0 & \cdots & 0 & 1 \\
0 & 0 & 0 & \cdots & -\tan\theta_1 - \cot\theta_1 & \cot\theta_1 \\
\vdots & \vdots & \vdots & & \cot\theta_2 & 0 \\
\vdots & \vdots & \vdots & & 0 & \vdots \\
 & \vdots & -\tan\theta_{N-3} - \cot\theta_{N-3} & & \vdots & \vdots \\
 & -\tan\theta_{N-2} - \cot\theta_{N-2} & \cot\theta_{N-2} & & \vdots & \vdots \\
-\tan\theta_{N-1} - \cot\theta_{N-1} & \cot\theta_{N-1} & 0 & \cdots & 0 & 0
\end{vmatrix}$$

Since all entries in the upper left triangle are zero, the value of the determinant is easily found as minus one times the product of entries on the lower left to upper right diagonal.

$$J = r^{N-1}\cos^{N-1}\theta_1 \cos^{N-2}\theta_2 \ldots \cos\theta_{N-1} \sin\theta_1 \sin\theta_2 \ldots \sin\theta_{N-1}(-1) \prod_{N=1}^{N-1}(-\tan\theta_N - \cot\theta_N)$$

Notice that $\tan\theta + \cot\theta = \frac{1}{\sin\theta \cos\theta}$. Now upon simplifying we obtain

$$J = (-1)^N r^{N-1} \cos^{N-2}\theta_1 \cos^{N-3}\theta_2 \ldots \cos\theta_{N-2}$$

and $|J| = r^{N-1} \cos^{N-2}\theta_1 \cos^{N-3}\theta_2 \ldots \cos\theta_{N-2}$ since

$\cos\theta_i > 0$ for $-\Pi/2 \leq \theta_i \leq \Pi/2$, $i=1, 2, \ldots N-2$.

In spherical coordinates the volume V of the N-dimensional hypersphere of radius r_o is readily evaluated.

$$V = \int_{r \leq r_o} \int_{\theta_1 = -\frac{\Pi}{2}}^{\Pi/2} \ldots \int_{\theta_{N-2} = -\frac{\Pi}{2}}^{\Pi/2} \int_{\theta_{N-1}=0}^{2\Pi} r^{N-1}\cos^{N-2}\theta_1 \cos^{N-3}\theta_2 \ldots \cos\theta_{N-2}\, dr d\theta_1 \ldots d\theta_{N-1}$$

Separating the integrations,

$$V = \int_{r=0}^{r_o} r^{N-1} dr \int_{\theta_1 = -\frac{\Pi}{2}}^{\Pi/2} \cos^{N-2}\theta_1\, d\theta_1 \ldots \int_{\theta_{N-2} = -\frac{\Pi}{2}}^{\Pi/2} \cos\theta_{N-2}\, d\theta_{N-2} \int_{\theta_{N-1}=0}^{2\Pi} d\theta_{N-1}$$

Since $\displaystyle\int_{\theta = -\frac{\Pi}{2}}^{\Pi/2} \cos^N\theta d\theta = \frac{\Gamma\left(\frac{N+1}{2}\right)\Gamma\left(\frac{1}{2}\right)}{\Gamma\left(\frac{N+2}{2}\right)}$,

$$V = \frac{r_o^N}{N} \left[\frac{\Gamma\left(\frac{N-1}{2}\right)\Gamma\left(\frac{1}{2}\right)}{\Gamma\left(\frac{N}{2}\right)} \right] \left[\frac{\Gamma\left(\frac{N-2}{2}\right)\Gamma\left(\frac{1}{2}\right)}{\Gamma\left(\frac{N-1}{2}\right)} \right] \cdots\cdots \left[\frac{\Gamma\left(\frac{2}{2}\right)\Gamma\left(\frac{1}{2}\right)}{\Gamma\left(\frac{3}{2}\right)} \right] 2\pi$$

$$= \frac{r_o^N}{N} \, 2\pi \, \frac{\Gamma\left(\frac{1}{2}\right)^{N-2} \Gamma\left(1\right)}{\Gamma\left(\frac{N}{2}\right)}$$

But $\Gamma\left(\frac{1}{2}\right) = \pi^{1/2}$ and $\Gamma(1) = 1$

$$V = \frac{r_o^N}{N} \, \frac{2\pi\,\pi^{\frac{N-2}{2}}}{\Gamma\left(\frac{N}{2}\right)} = \frac{2\,r_o^N}{N} \, \frac{\pi^{\frac{N}{2}}}{\Gamma\left(\frac{N}{2}\right)} .$$

II.2 Suitable Functions for Ellipsoidally Symmetric Distribution.

Suppose f is a real function, defined on domain R, a subset of $[0,\infty]$, and satisfying $f(\mu) \geq 0$ for all μ in R and $\mu^k f(\mu) d$ is finite for $k \leq N+1$. We show that f is suitable for defining a ellipsoidally symmetric density function and we determine the constant c so that of $(\sqrt{x' Ax})$ is an ellipsoidally symmetric density.

Let A be a NXN symmetric positive definite matrix and X an NX1 vector. Consider the ellipsoidally symmetric function $f(\sqrt{x' Ax})$. We wish to determine a constant C such that of $(\sqrt{x' Ax})$ is a density function.

It is clear that $C = \dfrac{1}{\displaystyle\int\cdots\int_{\sqrt{x' Ax} \in R} f\sqrt{x' Ax} \quad dx_1 \ldots dx_N}$.

To determine the value of the integral, we will make a transformation which rotates and scales. Let T be an orthonormal matrix such that $T'AT = D$, where D is a diagonal matrix. Make the change of variables

$$X = TD^{\frac{-1}{2}} z.$$

The Jacobian J of this transformation is

$$J = \begin{vmatrix} \frac{\partial x_1}{\partial z_1} & \frac{\partial x_2}{\partial z_1} & \cdots & \frac{\partial x_N}{\partial z_1} \\ \frac{\partial x_1}{\partial z_2} & \frac{\partial x_2}{\partial z_2} & \cdots & \frac{\partial x_N}{\partial z_2} \\ \vdots & & & \\ \frac{\partial x_1}{\partial z_N} & \frac{\partial x_2}{\partial z_N} & & \frac{\partial x_N}{\partial z_N} \end{vmatrix} = |TD^{-1/2}| = |T| \; |D|^{-1/2}$$

Since T is an orthornormal matrix, $|T| = 1$ and

$$|D| = |T'AT| = |T'|\ |A|\ |T| = |A|.$$

So the Jacobian is the determinant $|A|^{-1/2}$ which is positive since A is positive definite.

$$I = \underset{\sqrt{x'Ax} \in R}{\int \cdots \int} f\left(\sqrt{x'Ax}\right) dx_1 \ldots dx_N$$

$$= |A|^{-1/2} \underset{\sqrt{z'D^{-1/2}T'ATD^{-1/2}z} \in R}{\int \cdots \int} f\left(\sqrt{z'D^{-1/2}T'ATD^{-1/2}z}\right) dz_1 \ldots dz_N$$

$$= |A|^{-1/2} \underset{\sqrt{z'z} \in R}{\int \cdots \int} f(z'z)\, dz_1 \ldots dz_N$$

Now change to spherical coordinates.

$$z_1 = r \cos\theta_1 \cos\theta_2 \cdots \cos\theta_{N-1}$$

$$z_2 = r \cos\theta_1 \cos\theta_2 \cdots \cos\theta_{N-2} \sin\theta_{N-1}$$

$$\vdots$$

$$z_j = r \cos\theta_1 \cdots \cos\theta_{N-j} \sin\theta_{N-j+1}$$

$$\vdots$$

$$z_N = r \sin\theta_1$$

The Jacobian of this transformation is $(-1)^{N-1} r^{N-1} \cos^{N-2}\theta_1 \cos^{N-2}\theta_2 \ldots \cos\theta_{N-2}$.

$$I = |A|^{-1/2} \int_{r\varepsilon R} \int_{\theta_1=-\pi/2}^{\pi/2} \cdots \int_{\theta_{N-2}=-\pi/2}^{\pi/2} \int_{\theta_{N-1}=0}^{2\pi} f(r) r^{N-1} \cos^{N-2}\theta_1 \cos^{N-3}\theta_2 \cdots \cos\theta_{N-2}\, dr\, d\theta_1 \ldots d\theta_{N-1}$$

$$= |A|^{-1/2} \int_{r\varepsilon R} r^{N-1} f(r)\, dr \int_{\theta_1=-\pi/2}^{\pi/2} \cos^{N-2}\theta_1\, d\theta_1 \int_{\theta_2=-\pi/2}^{\pi/2} \cos^{N-3}\theta_2\, d\theta_2 \cdots$$

$$\int_{\theta_{N-2}=-\pi/2}^{\pi/2} \cos\theta_{N-2} \int_{\theta_{N-1}=0}^{2\pi} d\theta_{N-1}$$

Since $\int_{\theta=-\pi/2}^{\pi/2} \cos^N \theta\, d\theta = \dfrac{\Gamma\left(\frac{N+1}{2}\right) \Gamma\left(\frac{1}{2}\right)}{\Gamma\left(\frac{N+2}{2}\right)}$, the integrals are readily evaluated.

$$I = |A|^{-1/2} \int_{r \in R} r^{N-1} f(r)dr \left[\frac{\Gamma\left(\frac{N-1}{2}\right) \Gamma\left(\frac{1}{2}\right)}{\Gamma\left(\frac{N}{2}\right)} \; \frac{\Gamma\left(\frac{N-2}{2}\right)\Gamma\left(\frac{1}{2}\right)}{\Gamma\left(\frac{N-1}{2}\right)} \cdots \frac{\Gamma\left(\frac{2}{2}\right)\Gamma\left(\frac{1}{2}\right)}{\Gamma\left(\frac{3}{2}\right)} \right] 2\pi$$

$$= |A|^{-1/2} \int_{r \in R} r^{N-1} f(r)dr \; \frac{\Gamma\left(\frac{1}{2}\right)^{N-2}}{\Gamma\left(\frac{n}{2}\right)} \; 2\pi$$

$$= \frac{2(\pi)^{N/2}}{|A|^{1/2}\,\Gamma\left(\frac{N}{2}\right)} \int_{r \in R} r^{N-1} f(r)dr$$

Therefore, the constant c is

$$c = \frac{|A|^{+1/2}\,\Gamma\left(\frac{N}{2}\right)}{2\,(\pi)^{N/2} \int_{r \in R} r^{N-1} f(r)dr} .$$

Next we determine the normalizing constant c for the forms $e^{-\frac{1}{2}u^2}$ and $(1+u^2)^{-m}$.

Case 1. Multivariate Normal

The density function for the multivariate normal distribution is of the form

$$f(\sqrt{x'Ax}) = e^{-\frac{1}{2}x'Ax}, \quad o \le x'Ax \le \infty$$

$$\text{or,} \quad f(r) = e^{-\frac{1}{2}r^2}, \quad o \le r \le \infty .$$

Since $\int_{r\in R} r^{N-1} f(r)dr = \int_o^\infty r^{N-1} e^{-\frac{1}{2}r^2} dr = \int_o^\infty (2u)^{\frac{N-2}{2}} e^{-u} du = 2^{\frac{N-2}{2}} \Gamma\left(\frac{N}{2}\right)$

then the normalizing constant is

$$c = \frac{\Gamma\left(\frac{N}{2}\right)}{2\pi^{\frac{N}{2}} |A|^{-\frac{1}{2}} \int_{r\in R} r^{N-1} f(r)dr} = \frac{\Gamma\left(\frac{N}{2}\right)}{2\pi^{\frac{N}{2}} |A|^{-\frac{1}{2}} 2^{\frac{N}{2}-1} \Gamma\left(\frac{N}{2}\right)}$$

$$= \frac{1}{(2\pi)^{\frac{N}{2}} |A|^{-\frac{1}{2}}}$$

and,

$$f(\sqrt{x'Ax}) = \frac{1}{(2\pi)^{\frac{N}{2}} |A|^{-\frac{1}{2}}} e^{-\frac{1}{2}x'Ax}, \quad o \le x'Ax \le \infty .$$

Case 2. Multivariate Pearson Type VII.

Let $f(\sqrt{x'Ax}) = (1+x'Ax)^{-m}, \quad o \le x'Ax \le \infty$.

then,

$$\int_{r\in R} r^{N-1} f(r)dr = \int_o^\infty r^{N-1}(1+r^2)^{-m} dr = \int_o^1 \left(\frac{1-u}{u}\right)^{\frac{N-2}{2}} u^m \frac{du}{2u^2}$$

$$= \frac{1}{2} \frac{\Gamma\left(\frac{N}{2}\right)\Gamma\left(m-\frac{N}{2}\right)}{\Gamma(m)}, \quad m > \frac{N}{2} .$$

And the normalizing constant is

$$c = \frac{\Gamma\left(\frac{N}{2}\right)\Gamma(m)}{2\pi^{\frac{N}{2}} |A|^{-\frac{1}{2}} \frac{1}{2}\Gamma\left(\frac{N}{2}\right)\Gamma\left(m-\frac{N}{2}\right)} = \frac{\Gamma(m)}{\pi^{\frac{N}{2}} |A|^{-\frac{1}{2}} \Gamma\left(m-\frac{N}{2}\right)}$$

and,

$$f(\sqrt{x'Ax}) = \frac{\Gamma(m)|A|^{\frac{1}{2}}}{\pi^{\frac{N}{2}} \Gamma\left(m-\frac{N}{2}\right)} (1+x'Ax)^{-m}, \quad m > \frac{N}{2}, \quad o \le x'Ax \le \infty .$$

II.3 Covariance Matrix For Multivariate Distributions.

Given the density function $f(\sqrt{x'Ax})$ we want to find the covariance matrix Σ,

$$\Sigma = E(xx') = c \int \cdots \int_{\sqrt{x'Ax}\,\in\, R} xx' f(\sqrt{x'Ax})dx_1 \ldots dx_N$$

where c is a normalizing constant and N is the dimension of x. Using the orthonormal transformation T'AT = D, where D is a diagonal matrix, and scaling with $x=TD^{-\frac{1}{2}}z$,

we have
$$\Sigma = c\int\cdots\int_{\sqrt{z'z}\,\epsilon\,R} (TD^{-\frac{1}{2}}z)(z'D^{-\frac{1}{2}}T')\, f(\sqrt{z'z})\, |A|^{-\frac{1}{2}} dz_1 \cdots dz_N$$

since
$$x'AX = z'D^{-\frac{1}{2}}T'\,ATD^{-\frac{1}{2}}\,z = z'\,D^{-\frac{1}{2}}D\,D^{\frac{1}{2}}z = z'z$$

and
$$J = \begin{vmatrix} \frac{\partial x_1}{\partial z_1} & \cdots & \frac{\partial x_N}{\partial z_1} \\ \frac{\partial x_1}{\partial z_N} & \cdots & \frac{\partial x_N}{\partial z_N} \end{vmatrix} = |T\,D^{-\frac{1}{2}}| = |T|\,|D|^{-\frac{1}{2}} = |D|^{-\frac{1}{2}} = |T'AT|^{-\frac{1}{2}} = |A|^{-\frac{1}{2}}\,.$$

Rearranging,
$$\Sigma = c\,|A|^{-\frac{1}{2}}\,T\,D^{-\frac{1}{2}} \int\cdots\int_{\sqrt{z'z}\,\epsilon\,R} z\,z'\; f(\sqrt{z'z})\; dz_1 \cdots dz_N\; D^{-\frac{1}{2}}\,T',$$

where z'z is an N x N matrix. Looking at the off diagonal terms, for $i \neq j$,
$$\int\cdots\int_{\sqrt{z'z}\,\epsilon\,R} z_i\, z_j\, f(\sqrt{z_i z_j})\; dz_1 \cdots dz_N = 0$$
since we are integrating an odd function over even limits.

For terms of Σ along the diagonal, for i=j,
$$\int\cdots\int_{\sqrt{z'z}\,\epsilon\,R} z_i^2\; f(\sqrt{z'z})\; dz_1 \cdots dz_N = \int\cdots\int_{\sqrt{z'z}\,\epsilon\,R} z_1^2\quad f(\sqrt{z'z})\, dz_1 \cdots dz_N$$

and changing to spherical coordinates,
$$= \int_{r\epsilon R}\int_{\theta_1=-\pi/2}^{\pi/2}\cdots\int_{\theta_{N-2}=-\pi/2}^{\pi/2}\int_{\theta_{N-1}=0}^{2\pi} r^2\cos^2\theta_1\cdots\cos^2\theta_{N-1} f(r) r^{N-1}\cos^{N-2}\theta_1\cdots\cos\theta_{N-2}\,dr\,d\theta_1\cdots d$$

$$= \int_{r\,\epsilon\,R} r^{N+1} f(r)dr \int_{\theta_1=-\pi/2}^{\pi/2}\cos^N\theta_1 d\theta_1 \cdots \int_{\theta_{N-2}=-\pi/2}^{\pi/2}\cos^3\theta_{N-2}\,d\theta_{N-2}\; 2\int_{\theta_{N-1}=-\pi/2}^{\pi/2}\cos^2\theta_{N-1}\,d\theta_{N-1}.$$

Since
$$\int_{-\pi/2}^{\pi/2}\cos^k\theta\,d\theta = \frac{\Gamma\left(\frac{k+1}{2}\right)\Gamma\left(\frac{1}{2}\right)}{\Gamma\left(\frac{k+2}{2}\right)},$$

then

$$\underset{\sqrt{z'z}\,\epsilon\,R}{\int\cdots\int} z_i^{\,2}\, f(\sqrt{z'z})\, dz_1 \cdots dz_N = \underset{r\,\epsilon\,R}{\int} r^{N+1} f(r)dr \; \frac{\Gamma\left(\frac{N+1}{2}\right)\Gamma\left(\frac{1}{2}\right)}{\Gamma\left(\frac{k+2}{2}\right)} \; \frac{\Gamma\left(\frac{N}{2}\right)\Gamma\left(\frac{1}{2}\right)}{\Gamma\left(\frac{N+1}{2}\right)} \cdots \frac{\Gamma\left(\frac{4}{2}\right)\Gamma\left(\frac{1}{2}\right)}{\Gamma\left(\frac{5}{2}\right)} \; 2\,\frac{\Gamma\left(\frac{3}{2}\right)\Gamma\left(\frac{1}{2}\right)}{\Gamma\left(\frac{4}{2}\right)}$$

$$= \frac{2\pi^{\frac{N}{2}}}{N\,\Gamma\left(\frac{N}{2}\right)} \underset{r\,\epsilon\,R}{\int} r^{N+1} f(r)dr.$$

From II.2,

$$c = \frac{\Gamma\left(\frac{N}{2}\right) |A|^{\frac{1}{2}}}{2\,\pi^{\frac{N}{2}} \underset{r\,\epsilon\,R}{\int} r^{N-1} f(r)\, dr}.$$

so that

$$\Sigma = \frac{\Gamma\left(\frac{N}{2}\right) |A|^{\frac{1}{2}}}{2\,\pi^{\frac{N}{2}} \underset{r\,\epsilon\,R}{\int} r^{N-1} f(r)\, dr} \; |A|^{-\frac{1}{2}} \; T D^{-\frac{1}{2}} \; \frac{2\,\pi^{\frac{N}{2}}}{N\,\Gamma\left(\frac{N}{2}\right)} \underset{r\,\epsilon\,R}{\int} r^{N+1} f(r)\, dr \;\; D^{-\frac{1}{2}}\, T'$$

and, $$\Sigma = T\, D^{-1} T' \quad \frac{\int r^{N+1} f(r)\, dr}{N \int r^{N-1} f(r)\, dr}$$

where $T'\,AT = D$, or $D^{-1} = T'\, A^{-1} T$.

So that

$$\Sigma = A^{-1} \quad \frac{\int r^{N+1} f(r)\, dr}{N \int r^{N-1} f(r)\, dr}$$

Since the integrals are constants for any f, the covariance matrix is directly proportional to A^{-1}. We determine the constant of proportionality for the multivariate normal and Pearson Type VII distributions.

Case 1. Multivariate Normal

For the multivariate normal, the density is of the form $f(r) = ce^{-\frac{1}{2}r^2}$ where $r^2 = x'Ax$, $o \leq r \leq \infty$.

Since, $\int_o^\infty r^k e^{-\frac{1}{2}r^2} dr = 2^{\frac{k-1}{2}} \Gamma\left(\frac{k+1}{2}\right)$

then $\int r^{N+1} f(r)\, dr = c\, 2^{\frac{N}{2}} \Gamma\left(\frac{N+2}{2}\right)$.

and $\int r^{N-1} f(r)\, dr = c2^{\frac{N-2}{2}} \Gamma\left(\frac{N}{2}\right)$

so that $\Sigma = A^{-1} \dfrac{2^{\frac{N}{2}} \frac{N}{2} \Gamma\left(\frac{N}{2}\right)}{N\, 2^{\frac{N}{2}-1} \Gamma\left(\frac{N}{2}\right)}$

and thus, $\Sigma = A^{-1}$

or, $A = \Sigma^{-1}$.

Case 2 Multivariate Pearson Type VII.

$f(r) = c(1 + r^2)^{-m}$, $o \leq r \leq \infty$.

Since $\int r^{N+1} f(r)\, dr = \dfrac{c}{2} \quad \dfrac{\Gamma\left(\frac{N}{2} + 1\right) \Gamma\left(m - \frac{N}{2} - 1\right)}{\Gamma(m)}$

and $\int r^{N-1} f(r)\, dr = \dfrac{c}{2} \quad \dfrac{\Gamma\left(\frac{N}{2}\right) \Gamma\left(M - \frac{N}{2}\right)}{\Gamma(m)}$

then, $\Sigma = A^{-1} \dfrac{\frac{cN}{2} \Gamma\left(\frac{N}{2}\right) \Gamma\left(m - \frac{N}{2} - 1\right)}{2N\, c\, \Gamma(m)} \qquad \dfrac{2\, \Gamma(m)}{c\Gamma\left(\frac{N}{2}\right)\left(m - \frac{N}{2} - 1\right) \Gamma\left(m - \frac{N}{2} - 1\right)}$

$= A^{-1} \dfrac{1}{2\left(m - \frac{N}{2} - 1\right)}$

APPENDIX III

NORMALIZATION PROCEDURE TO MAKE COVARIANCE MATRIX INVARIANT UNDER TRANSLATING AND SCALING TRANSFORMATIONS

Let $\ddagger_x$ be a covariance matrix for the difference vectors of grey tone N-tuples in a specified spatial relationship within a subimage. We transform the covariance matrix to obtain the normalized covariance matrix $\ddagger_y$ using $y = Dx$, where x is the difference vector and D is diagonal. Thus, assuming zero mean,

$$\begin{aligned} \ddagger_y &= E(yy') = E\ (Dxx'D') \\ &= D\ E\ (xx')\ D' \\ &= D\ \ddagger_x\ D \quad \text{since } D' = D. \end{aligned}$$

For normalization, we have

$$d_{ii} = \frac{1}{\sqrt{\sigma_{ii}}}$$

where σ_{ii} is the ii^{th} element of $\ddagger_x$ and is the variance σ_i^2 of the i^{th} component of x.

Assume that all grey tone N-tuples have a scale factor a and an additive factor c so that for N-tuples $ax_1 + c$ and $ax_2 + c$, the difference becomes

$$y = (ax_1 + c) - (ax_2 + c)$$

$$y = a(x_1 - x_2).$$

Hence, translational effects due to bias terms are cancelled but scaling effects are marked by the diagonal transformation $y = Ax$ so that the elements of the covariance matrix become

$$\ddagger = E\ (yy') = E\ (Axx'A')$$

$$\ddagger = A\ \ddagger_x\ A$$

where A is a diagonal matrix. We must show that Σ_N, the normalized covariance matrix of Σ, is identical to Σ_y. Normalizing Σ we have

$$\Sigma_N = D \Sigma D$$

$$= D\,(A\,\Sigma_x A)\,D$$

where D is again diagonal but in this case,

$$d_{ii} = \frac{1}{\sqrt{\sigma_{ii}\, a_{ii}^{2}}}$$

with a_{ii} the ii^{th} element of diagonal matrix A. For the ij^{th} element of Σ_N we have

$$\sigma_{Nij} = d_{ii}\, a_{ii}\, \sigma_{ij}\, a_{jj}\, d_{jj}$$

$$= \frac{a_{ii}\, \sigma_{ij}\, a_{jj}}{\sqrt{\sigma_{ii}\, a_{ii}^{2}}\sqrt{\sigma_{jj}\, a_{jj}^{2}}}$$

$$= \sigma_{ij},$$

Thus, this procedure of normalization makes the entries of the normalized covariance matrix invariant with respect to translating and scaling transformations on the grey tone N-tuples.

APPENDIX IV

COMPUTER PROGRAM DOCUMENTATION & LISTINGS

IV.1 PROGRAM DOCUMENTATION

- IV.1-a ERTS Retrieval Programs
- IV.1-b Texture Analysis Programs
- IV.1-c Cross-Band Texture Analysis Programs
- IV.1-d Piecewise Linear Classification Programs

IV.2 PROGRAM LISTINGS

- IV.2-a ERTS Retrieval Programs
- IV.2-b Texture Analysis Programs
- IV.2-c Cross-Band Texture Analysis Programs
- IV.2-d Piecewise Linear Classification Programs

IV.1-a ERTS Retrieval Programs - Documentation

- RETV
  - ERTS
  - PIXEY
    - ZEQUAN
    - PITCHR
  - WRTDSK
  - RDDSK1
    - KEQUAN
  - RDDSK2

ERTS RETRIEVAL PROGRAMS

PROGRAM TITLE: RETV
VERSION: I
DATE: January, 1974
UPDATE: January, 1974
AUTHOR: Robert J. Bosley
DOCUMENTED BY: Robert J. Bosley
PROGRAM LANGUAGE: FORTRAN IV
IMPLEMENTED ON: HW635
PURPOSE:

This is the mainline for the ERTS Retrieval Programs which retrieve ERTS MSS data from standard NASA digital tapes and outputs the image data in a picture, or greytone listing, or copies it onto an output tape.

INPUT PARAMETERS under NAMELIST 'PARAM':

NBAND	Band number, 1 through 4 for MSS band 4 through 7, to be selected. Set to 5 for all 4 bands, assumed to be 2.
IRSTRT, ICSTRT	Row, column starting coordinates.
IRSTOP, ICSTOP	Row, column stopping coordinates.
TITLE	80 Column title for output list.
MILLI	TRUE if coordinates are in millimeters; assumed to be FALSE, coordinates are row, column points in ERTS image.
SMALL	TRUE if millimeter coordinates are from a small 70 mm by 70 mm negative; assumed to be FALSE, coordinates from a 7 inch by 7 inch print.
PRNT	TRUE for grey tone listing; assumed FALSE.
TAPE	TRUE for tape output; assumed FALSE.

IFIL	Output file code for tape; assumed to be 02.
PIC	TRUE for picture output; assumed TRUE.
QUAN	TRUE for equals probability quantizing of the image; assumed to be TRUE.
SPIC	TRUE for special picture run-see PIXEY; assumed to be FALSE.

REQUIREMENTS:

1. ERTS input tape must be on file code 'ES'.
2. Four disc files must be on files 11, 12, 13 and 14.
3. Any output tape must be positioned on file code 'IFIL'.
4. All coordinates must be determined relative to the input tape rather than the print -- that is, ICSTOP must not exceed 824 points or 46 mm.
5. Core - 22 k
6. Subprograms required:

   RETV
   ERTS
   PIXEY
   ZEQUAN
   ERTS
   PITCHR
   WRTDSK
   RDDSKI
   KEQUAN
   PITCHR
   RDDSK2

COMMENTS:

For efficiency, data is read by RETV in blocks of 41 lines by 41 points. One ERTS tape (one-fourth of an image) will then be covered by 20 horizontal blocks, leaving 4 points left at the end of each line. Note --- sometimes on the first tape of an image, the first four points are greytones of 255 and can adversely affect a picture printout. If so, set ICSTRT = 5.

Also note that if the point ICSTRT is not a multiple of 41 from the end of the line, then the last points may not necessarily be listed since blocks are determined starting from ICSTRT.

A special picture run can be made to print out and reduce any image over the entire tape by setting SPIC to TRUE and specifying under namelist PARAM the following parameters: IRSTRT, IRSTOP, ICSTRT, ICSTOP, NBAND, QUAN. Then under namelist PICTUR, parameters for PITCHR are specified. See PIXEY for details.

ERTS RETRIEVAL PROGRAM

SUBPROGRAM TITLE: ERTS
VERSION: II
DATE: September, 1972
UPDATE: November, 1973
AUTHOR: G. Gunnels
DOCUMENTED BY: R. Bosley
PROGRAM LANGUAGE: GMAP
IMPLEMENTED ON: HW635

PURPOSE:

To read 7-track ERTS MSS data tapes.

ENTRY POINTS:

```
CALL EINIT (NOLS)
CALL ESKIP (NOSK)
CALL EREAD (I, LN)
CALL EREWND
```


ARGUMENTS:

NOLS	Number of words per scan line; returned by EINIT.
NOSK	The number of records to skip.
I	The array into which the NOLS words of data from a line of ERTS data is placed.
LN	Returned by EREAD giving the line number of the line of data returned. If LN =0, the end of file was reached on the ERTS tape.

ERROR FLAGS:

MB	EREAD buffer is not large enough for a block of ERTS data.
AI	EINIT was called twice.
NI	EINIT was not called before calling EREAD, ESKIP, or EREWND.

UE	EOF encountered while reading ID or annotation blocks on ERTS tape.
EF	EOF encountered while trying to skip records in ESKIP.

COMMENTS:

EINIT initializes the ERTS tape so that data may be read, and must be called first. ESKIP skips over NOSK records (scan lines). EREWND rewinds the ERTS tape. EINIT must not be called twice. The data placed into array I by EREAD is in standard corresponding point forms. Since the ERTS MSS data has four channels, there are actually NDLS/4 points or cells per scan line.

REQUIREMENTS:

ERTS tape must be on file code 'ES'.

SUBPROGRAM TITLE:	PIXEY
VERSION:	1
DATE:	January, 1974
UPDATE:	January, 1974
AUTHOR:	Robert J. Bosley
DOCUMENTED BY:	Robert J. Bosley
PROGRAM LANGUAGE:	FORTRAN IV
IMPLEMENTED ON:	HW635

PURPOSE:

To provide a flexible option for printing a picture of ERTS MSS data using user specified parameters to PITCHR.

ENTRY POINT:

CALL PIXEY (ILINE, IMAGE, IRSTRT, IRSTOP, ICSTRT, ICSTOP, QUAN, NBAND)

INPUT ARGUMENTS:

ILINE	Array the ERTS line is read into.
IMAGE	Array containing one line of the image.
IRSTRT, IRSTOP	Starting, stopping row in the image.
ICSTRT, ICSTOP	Starting, stopping column in the image.
QUAN	TRUE for equal probability quantizing of the image.
NBAND	Band number to be processed; set to 5 for all 4 MSS bands.

INPUT PARAMETERS: under NAMELIST 'PICTUR':

LNSKIP, KOLSKP	Line and column increment for ERTS data; assumed to be 4,3.

ICELL, JCELL	Number of rows, columns in image to be printed; assumed to be 1,256.
INIT	Number of times entry to PITCHR is made at SNAP: must be greater than 1; assumed to be 304.
IMIN, IMAX	Minimum, maximum greytones in image; assumed to be 0,75 or 12 if QUAN is true .
NROW	Number of rows to be printed; equal to ICELL.
NFILES	Number of output files available to PITCHR; set to 0 for all output on file code 06, set to 2 for files 06 and 42; assumed to be 2.
IFIL(10)	Array containing output file codes; assumed to be 06 and 42.
NULW, NULD	Number of columns, rows per output page; assumed to be 129, 60.
AMAG, DMAG	Width, length magnification for output picture; assumed to be 1.

SAMPLE RUN:

$PARAM SPIC=T, QUAN=T, IRSTRT=1, IRSTOP=1216, ICSTRT=5, ICSTOP=772, NBAND=2$END/$PICTUR INIT=304, NFILES=2, LNSKIP=4, KOLSKP=3, JCELL=256$END.

This run will print out on files 06 and 42 a picture 256 points wide by 304 lines long. Note that 1216/4 = 304 and 786/3 = 256 gives the values for LNSKIP=4, KOLSKP=3, JCELL=256, INIT=304. Also, using these values for LNSKIP and KOLSKP will result in a picture in proportion to the same area on an ERTS image print, approximately twelve 64 by 64 subimages across by 19 subimages down the tape.

COMMENTS:

If $PIC=T in the $PARAM card, then a $PICTUR card must follow. Note that the parameters IRSTRT, IRSTOP, ICSTRT, ICSTOP, QUAN, NBAND are supplied on the $PARAM card while LNSKIP, KOLSKP and all PITCHR parameters are supplied on the $PICTUR card.

The program will output a negative picture but a positive picture can be made by setting IMAX=0 and IMIN=75 (or 12 if quantization is used). The parameters are initialized to output an area approximately twelve 64 by 64 subimages wide by 19 down. For a complete description of PITCHR parameters, see PITCHR.

REQUIREMENTS:

1. Processor time for one band with parameters as shown in the sample run is 0.075 hr with 3k lines of output.
2. Subprograms required are ZEQUAN, PITCHR.

CALLED BY:

RETV

SUBPROGRAM TITLE: ZEQUAN
VERSION: I
DATE: September, 1973
UPDATE: September, 1973
AUTHOR: Z. Dinstein
DOCUMENTED BY: Robert J. Bosley
PROGRAM LANGUAGE: FORTRAN IV
IMPLEMENTED ON: HW635
PURPOSE:

To equal probability quantize a large image on disc to NQ levels.

ENTRY POINT:

CALL ZEQUAN (LINE, NUMLIN, NUMPPL, NCOMP, ICOMP, LEFT, NQ, INFILE, IOUTFL)

INPUT ARGUMENTS:

LINE	Array to store one line of the image.
NUMLIN	Number of lines in the image.
NUMPPL	Number of columns in the image.
NCOMP	Number of components in the image.
ICOMP	The component to be quantized.
LEFT	Left-most cell desired in the line.
NQ	Number of quantized levels.
INFIL	File code of disc containing the image to be quantized.

OUTPUT ARGUMENTS:

IOUTFL	Output file code for the quantized image.

COMMENTS:

Processing is done line by line after an initial pass through the image is made to determine the number of grey tone levels in the image. The minimum and maximum grey tones are printed. The number of grey tone levels should not exceed 512. Input data on disc INFILE must be in binary.

CALLED BY:

PIXEY

SUBPROGRAM TITLE:	PITCHR
VERSION:	II
DATE:	July, 1969
UPDATE:	November, 1970
AUTHOR:	R. Cowles
DOCUMENTED BY:	G. Elliott
PROGRAM LANGUAGE:	GMAP
IMPLEMENTED ON:	HW635

PURPOSE:

To print out images in 13 grey levels.

ENTRY POINTS:

CALL PITCHR (IRRAY, ICELL, JCELL, INIT, IT, IMIN, IMAX, NROW, NFIL, IFIL, NULW, NULD, AMAG, DMAG,*)
CALL SNAP
CALL SNAPA (IARRAY)
CALL ENDBNR

ARGUMENTS:

IRRAY	Array to be printed, either integer or floating point.
ICELL	Number of rows in array. (row dimension)
JCELL	Number of columns in array. (column dimension)
INIT	=0 if all of image to be printed out is in core at time of call. Output will be done before return to calling program. =1 for reinitialization entry. Any of the arguments previously specified with INIT = 0 with the exception of INIT may now be changed. Return will be made to the calling program without any output. This is especially useful if the image is read into core in pieces and the last piece does not completely fill the array. >1 for initialization

	entry. INIT will reflect the number of times entry is made at SNAP before final border is to be printed. Return is to the calling routine without any output. INIT =0 is assumed.
II	=0 if array is floating point =1 if array is integer II =1 is assumed.
IMIN	Minimum brightness level in array. Type of IMIN should correspond to that indicated by II. IMIN =0 is assumed.
IMAX	Maximum brightness level in array. Type of IMAX should correspond to that indicated by II. IMAX =12 is assumed.
NROW	Number of rows of array to be printed if full array is not to be printed. This allows for partial printing. NROW = ICELL is assumed.
NFIL	Number of output files available if image is to be output in strips that are NULW lines wide. =0 for all output on file code 06 =1 for all output on file code IFIL 1 for outputs in strips, on file codes specified in array IFIL. NFIL=0 is assumed.
IFIL	Array containing output file codes. Ignored unless NFIL = 0. IFIL must be a variable rather than a literal, since SNAP alters the value(s) of IFIL to contain the location of the relevant file control block in the upper half of the word.
NULW	Number columns per output page. MAX =129 NULW =120 unless otherwise specified.
NULD	Number of rows to be printed before a slew on the top of the next page is given. NULD = ICELL*DMAG +1.

AMAG	Floating point magnification in width. AMAG =1 is assumed.
DMAG	Floating point magnification in length, DMAG =1 is assumed.
*	Error return if not enough output files are available. Return is made with NARG =-10. If not specified, a message will be printed out on the accounring report and an NF report will terminate execution.

SUBPROGRAM TITLE:	WRTDSK
VERSION:	II
DATE:	September, 1972
UPDATE:	January, 1974
AUTHOR:	Robert J. Bosley
DOCUMENTED BY:	Robert J. Bosley
PROGRAM LANGUAGE:	FORTRAN IV
IMPLEMENTED ON:	HW635

PURPOSE:

To write one line of ERTS data onto disc files, one file for each band.

ENTRY POINT:

CALL WRTDSK (ILINE, NHOR, IPSTR, IPEND, NBAND)

INPUT ARGUMENTS:

ILINE	Array containing one line of ERTS data
NHOR	Number of horizontal blocks of 41 columns in the line.
IPSTR, IPEND	Starting and ending points in the ERTS line.
NBAND	The desired band; set to 5 for all 4 MSS bands.

COMMENTS:

After reading a line of data into ILINE, RETV calls WRTDSK to pick out the segment of NHOR blocks in the line and write it onto disc. If only one band is desired, only that band is put onto disc. Disc files 11, 12, 13, and 14 must be present.

CALLED BY:

RETV

ERTS RETRIEVAL PROGRAMS

SUBPROGRAM TITLE: RDDSKL

VERSION: II

DATE: September, 1972

UPDATE: January, 1974

AUTHOR: Robert J. Bosley

DOCUMENTED BY: Robert J. Bosley

PROGRAM LANGUAGE: FORTRAN IV

IMPLEMENTED ON: HW635

PURPOSE:

To read the ERTS data from disc and print out a picture of the image, proceeding vertically and then from left to right.

ENTRY POINT:

CALL RDDSKL (IMAGE, QUAN, NHOR, NVERT, NDSK)

INPUT ARGUMENTS:

IMAGE	Array used to store two 41 x 41 blocks.
QUAN	TRUE for equal probability quantization of the image.
NHOR	Number of horizontal 41 x 41 blocks.
NVERT	Number of vertical 41 x 41 blocks.
NDSK	File code of disc to be processed.

ERROR FLAGS:

Same as for PITCHR.

COMMENTS:

Since blocks are 41 columns wide, then two horizontal blocks are processed together, except for the final strip of blocks when NHOR is an odd number. It is recommended that QUAN be set to true since transmission errors result in very high and very low greytones along the line which will make the true image features indistinguishable unless equal probability quantizing is used.

SUBPROGRAMS CALLED:

KEQUAN
PITCHR

CALLED BY:

RETV

SUBPROGRAM TITLE: KEQUAN

VERSION: II

DATE: September, 1971

UPDATE: June, 1973

AUTHOR: G. Elliot

DOCUMENTED BY: Robert J. Bosley

PROGRAM LANGUAGE: FORTRAN IV

IMPLEMENTED ON: HW635

PURPOSE:

To equal probability quantize the input array to NQ levels.

ENTRY POINT:

CALL KEQUAN (IA, NGL, NQ, IASIZE)

ARGUMENTS:

IA	Input array which is returned quantized.
NGL	Number of greytone levels in IA.
NQ	Number of quantizing levels.
IASIZE	Size of array IA.

ERROR FLAGS:

If the number of greytone levels exceeds 512, an error message is printed.

CALLED BY:

RDDSK1

SUBPROGRAM TITLE: RDDSK2
VERSION: II
DATE: September, 1972
UPDATE: January, 1974
AUTHOR: Robert J. Bosley
DOCUMENTED BY: Robert J. Bosley
PROGRAM LANGUAGE: FORTRAN IV
IMPLEMENTED ON: HW635

PURPOSE:

To read ERTS data from disc and print out the grey tones and/or copy the data onto an output tape.

ENTRY POINT:

CALL RDDSK2 (IMAGE, IRSTRT, IRSTOP, NHOR, NDSK, PRNT, TAPE, IFIL)

INPUT ARGUMENTS:

IMAGE	Array data is read into.
IRSTRT, IRSTOP	Starting, stopping lines of data.
NHOR	Number of horizontal blocks of 41 columns.
NDSK	File code of the disc to be read.
PRNT	TRUE for greytone listing.
TAPE	TRUE for tape output
IFIL	Output tape file code.

COMMENTS:

If neither the grey tone listing nor the output tape is desired, execution is returned to the calling program, RETV.

CALLED BY:

RETV

IV.1-b Texture Analysis Programs - Documentation

MAINLN
- ERTS
- MAING
  - KEQUAN
  - PITCHR
  - FPLXIT
    - INDEX
  - IMOMTR
    - COR
    - IEQPQ1
  - RITOWT

ERTS TEXTURE ANALYSIS

C-2

PROGRAM TITLE: MAINLN

VERSION: II

DATE: September, 1972

UPDATE: June, 1973

AUTHOR: Robert J. Bosley

DOCUMENTED BY: Craig Paul

PROGRAM LANGUAGE: FORTRAN IV

IMPLEMENTED ON: HW 635

PURPOSE:

To read the Texture Analysis parameters, read in subimages from ERTS input tape, and send these subimages to MAING for processing.

INPUT PARAMETERS:

1. Title Card: Up to 80 columns for use as identification information.

2. Parameters under NAMELIST /PARAM/:

IBAND — Band used for processing-
= 1 for MSS Band 4
= 2 for MSS Band 5
= 3 for MSS Band 6
= 4 for MSS Band 7
assumed to be 2.

NUMPPL — Number of points per line of each subimage; assumed to be 64.

NUMLIN — Number of lines of each subimage; assumed to be 64.

NBVERT — Number of the last vertical subimage to be processed; assumed to be 36.

NUMIM — Number of subimages taken horizontally in one run; assumed to be 3. Note: NUMPPL x NUMIM must not exceed 192 points.

N11	The upper-left column coordinate for the vertical strip (of NUMIM horizontal subimages) being processed in this run.
NUMSTR	The number of the vertical strip being processed; assumed to be 1.
NBSKIP	The number of vertical rows of subimages to skip before beginning processing; assumed to be 0.
PNCH	Specifies the output option: = 1HY for card output = 1HT for tape output on file code IF = 1HN for neither card nor tape but a listing of the LEX arrays; assumed to be tape (1HT). Note: PNCH must be denoted as a Hollerith constant in the $PARAM card.
IF	Output tape file code which is assumed to be in position (it is not rewound); assumed to be '03'.
NRED	The base used for image reduction; assumed to be 1.
NSTART, NTIMES	Each subimage is processed NSTART through NTIMES times, each time with a new reduction factor NFT = NRED ** NLAYER where NLAYER goes from NSTART -1 through NTIMES -1; both assumed to be 1.
PICTUR	TRUE for a picture of each subimage; assumed FALSE.
NQUANT	Number of quantization levels of the probability function in IMOMTR; assumed to be 16.
MERGE	TRUE for all four LEX arrays to be merged into one array; assumed to be TRUE.

REQUIREMENTS:

1. ERTS input tape must be on file code 'ES'.
2. Core - 43k.
3. Disc file must be on file code '02'.
4. Subprograms required:

MAINLN
ERTS
MAING
KEQUAN
PITCHR
FPLXIT
INDEX
IMOMTR
INDEX
COR
IEQPQ1
RITOWT

CARD SETUP FOR SAMPLE RUN:

```
$   IDENT      9999, ANYNAME
$   LIBRARY    LB
$   OBJECT     ERTS TEXTURE ANALYSIS PROGRAMS
$   DKEND
$   EXECUTE
$   PRMFL      LB, R, S, PATTERN/GEE/LIB
$   TAPE       ES, A5DD, 60500, ERTS00, INPUT
$   FILE       02, A2R, 2L
$   LIMITS     20, 43k,,10k
$   INCODE     IBMF
TEST-SETUP FOR TEXTURE ANALYSIS PROGRAMS
$   PARAM      N11=1, PNCH = 1HN, PICTUR = T$END
$   ENDJOB
```

This run of the texture analysis programs will process the ERTS image in 64 x 64 subimages, giving only printed output plus a picture of each subimage.

COMMENTS:

This is the mainline of the texture analysis programs. Each ERTS image is divided into 4 vertical strips, each put onto a 7-track digital tape. This image is divided up into subimages for processing. For example, if the subimages are 64 columns by 64 lines, then the first tape contains subimages 1 through 12, the second contains 13 through 24, and so on up to 48 horizontal subimages. And, each tape contains 36 vertical subimages. Often, the first 8 points at the beginning of each line have grey tones of 255 and will adversely affect processing. Therefore, the first eight points of each line are skipped.

Each tape contains usually 3296 points of 4 bands, or 824 points per line for one band. Skipping the first eight points leaves 816 points per line. If subimages are each 64 columns, then 12 will occupy 768 columns, leaving 48 points at edge of each tape unused.

Due to core limitations, each input tape is processed in vertical strips of up to 192 horizontal points. This is 3, 64 by 64 subimages, or 6, 32 by 32 subimages in one run. For example, if NUMPPL = 64 and NUMIM =3, then the tape is processed as follows: Run 1 - (1, 1), (1, 2), (1, 3), (2, 1), (2, 2), (2, 3), (3, 1),...,(36, 1) (36, 2), (36, 3). Run 2 - (1, 4), (1, 5), (1, 6), (2, 4), (2, 5), (2, 6), (3, 4),...,(36, 4), (36, 5), (36, 6). Run 3 - (1, 7), (1, 8), (1, 9), (2, 7), (2, 8), (2, 9), (3, 7),..., (36, 7), (36, 8), (36, 9). Run 4 - (1, 10), (1,11) (1, 12),...,(36, 10), (36, 11), (36, 12). This completes the processing of the first tape. Note that N11 is determined relative to the entire image while NUMSTR is relative to the input tape. That is, for tape 1, N11 and NUMSTR are both 1 for Run 1. But for Run 2, N11 is 4 while NUMSTR is 2 and for Run 3, N11 is 7 while NUMSTR is 3, etc.

Continuing with the example, tape 2 will be processed as follows: Run 1 (N11= 13, NUMSTR=1) - (1, 13), (1, 14), (1, 15),...,(36, 13), (36, 14), (36, 15). Run 2 (N11 =16, NUMSTR=2) - (1, 16), (1, 17), (1, 18),..., (36, 16), (36, 17), (36, 18), etc.

See Figure 1 for an illustration of an ERTS image divided into 64 by 64 subimages. Note - the size of each subimage must not exceed 4096 points.

ERTS IMAGE

4 Tapes for Each ERTS Image

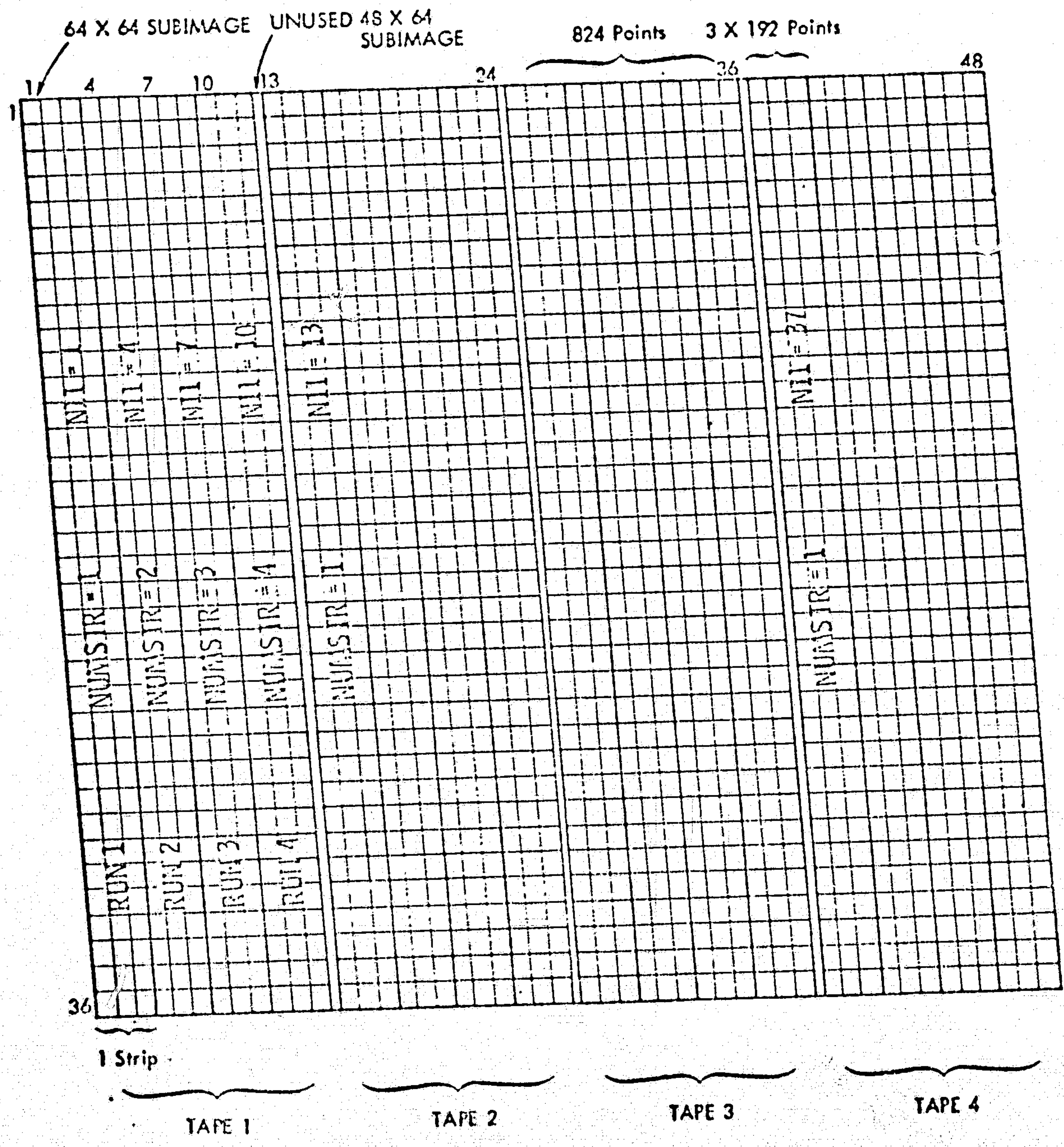


Figure 1.

SUBPROGRAM TITLE:	ERTS
VERSION:	II
DATE:	September, 1972
UPDATE:	November, 1973
AUTHOR:	G. Gunnels
DOCUMENTED BY:	R. Bosley
PROGRAM LANGUAGE:	GMAP
IMPLEMENTED ON:	HW635

PURPOSE:

To read 7-track ERTS MSS data tapes.

ENTRY POINTS:

CALL EINIT (NOLS)
CALL ESKIP (NOSK)
CALL EREAD (I, LN)
CALL EREWND

ARGUMENTS:

NOLS	Number of words per scan line; returned by EINIT.
NOSK	The number of records to skip.
I	The array into which the NOLS words of data from a line of ERTS data is placed.
LN	Returned by EREAD giving the line number of the line of data returned. If LN =0, the end of file was reached on the ERTS tape.

ERROR FLAGS:

MB	EREAD buffer is not large enough for a block of ERTS data.
AI	EINIT was called twice.
NI	EINIT was not called before calling EREAD, ESKIP, or EREWND.

UE	EOF encountered while reading ID or annotation blocks on ERTS tape.
EF	EOF encountered while trying to skip records in ESKIP.

COMMENTS:

EINIT initializes the ERTS tape so that data may be read, and must be called first. ESKIP skips over NOSK records (scan lines). EREWND rewinds the ERTS tape. EINIT must not be called twice. The data placed into array I by EREAD is in standard corresponding point forms. Since the ERTS MSS data has four channels, there are actually NDLS/4 points or cells per scan line.

REQUIREMENTS:

ERTS tape must be on file code 'ES'.

SUBPROGRAM TITLE:	MAING
VERSION:	III
DATE:	October, 1971
UPDATE:	June, 1973
AUTHOR:	R. M. Haralick
DOCUMENTED BY:	Robert J. Bosley
PROGRAM LANGUAGE:	FORTRAN IV
IMPLEMENTED ON:	HW 635

PURPOSE:

To print out a picture of the subimage, copy it to file 2, and then process it through the texture subroutines FPLXIT to get the LEX arrays, IMOMTR to calculate the texture features, and RITOWT to output the results.

ENTRY POINT:

CALL MAING (IWORK, MERR, MERGE, PICTUR, IF)

ARGUMENTS:

IWORK	NUMLIN by NUMPPL subimage array.
MERR	Set to 1 if an error occurs in fitting the LEX arrays into IWORK.
MERGE	TRUE to merge the four LEX arrays into one.
PICTUR	TRUE for a picture of the subimage.
IF	File code for the output tape.

COMMENTS:

The subimage sent in IWORK is scaled to fill a page for the picture printout, and at the same time it is copied to a scratch disc on file code '02'. If the size of the LEX arrays is greater than NUMPPL x NUMLIN, then an error message is printed and processing is terminated by MAINLN, after putting EOF marks on the output tape. Each subimage is quantized by KEQUAN to 32 levels.

CALLED BY:

MAINLN

SUBPROGRAMS REQUIRED:

KEQUAN
PITCHR
FPLXIT
IMOMTR
RITOWT

SUBPROGRAM TITLE:	KEQUAN
VERSION:	II
DATE:	September, 1971
UPDATE:	June, 1973
AUTHOR:	G. Elliot
DOCUMENTED BY:	Robert J. Bosley
PROGRAM LANGUAGE:	FORTRAN IV
IMPLEMENTED ON:	HW635

PURPOSE:

To equal probability quantize the input array to NQ levels.

ENTRY POINT:

CALL KEQUAN (IA, NGL, NQ, IASIZE)

ARGUMENTS:

IA	Input array which is returned quantized.
NGL	Number of greytone levels in IA.
NQ	Number of quantizing levels.
IASIZE	Size of array IA.

ERROR FLAGS:

If the number of greytone levels exceeds 512, an error message is printed.

SUBPROGRAM TITLE: PITCHR

VERSION: II

DATE: July, 1969

UPDATE: November, 1970

AUTHOR: R. Cowles

DOCUMENTED BY: G. Elliott

PROGRAM LANGUAGE: GMAP

IMPLEMENTED ON: HW635

PURPOSE:

To print out images in 13 grey levels.

ENTRY POINTS:

CALL PITCHR (IRRAY, ICELL, JCELL, INIT, IT, IMIN, IMAX, NROW, NFIL, IFIL, NULW, NULD, AMAG, DMAG,*)
CALL SNAP
CALL SNAPA (IARRAY)
CALL ENDBNR

ARGUMENTS:

IRRAY	Array to be printed, either integer or floating point.
ICELL	Number of rows in array. (row dimension)
JCELL	Number of columns in array. (column dimension)
INIT	=0 if all of image to be printed out is in core at time of call. Output will be done before return to calling program. =1 for reinitialization entry. Any of the arguments previously specified with INIT =0 with the exception of INIT may now be changed. Return will be made to the calling program without any output. This is especially useful if the image is read into core in pieces and the last piece does not completely fill the array. >1 for initialization

	entry. INIT will reflect the number of times entry is made at SNAP before final border is to be printed. Return is to the calling routine without any output. INIT =0 is assumed.
II	=0 if array is floating point =1 if array is integer II =1 is assumed.
IMIN	Minimum brightness level in array. Type of IMIN should correspond to that indicated by II. IMIN =0 is assumed.
IMAX	Maximum brightness level in array. Type of IMAX should correspond to that indicated by II. IMAX =12 is assumed.
NROW	Number of rows of array to be printed if full array is not to be printed. This allows for partial printing. NROW = ICELL is assumed.
NFIL	Number of output files available if image is to be output in strips that are NULW lines wide. =0 for all output on file code 06 =1 for all output on file code IFIL >1 for outputs in strips, on file codes specified in array IFIL. NFIL=0 is assumed.
IFIL	Array containing output file codes. Ignored unless NFIL= 0. IFIL must be a variable rather than a literal, since SNAP alters the value(s) of IFIL to contain the location of the relevant file control block in the upper half of the word.
NULW	Number columns per output page. MAX =129 NULW =120 unless otherwise specified.
NULD	Number of rows to be printed before a slew on the top of the next page is given. NULD = ICELL*DMAG +1.

AMAG	Floating point magnification in width. AMAG =1 is assumed.
DMAG	Floating point magnification in length, DMAG =1 is assumed.
*	Error return if not enough output files are available. Return is made with NARG =-10. If not specified, a message will be printed out on the accounring report and an NF report will terminate execution.

SUBPROGRAM TITLE: FPLXIT
VERSION: II
DATE: September, 1971
UPDATE: June, 1973
AUTHOR: R. M. Haralick
DOCUMENTED BY: R. J. Bosley
PROGRAM LANGUAGE: FORTRAN IV
IMPLEMENTED ON: HW 635

PURPOSE:

To compute the four nearest neighbor greytone matrices LEX1, LEX2, LEX3, LEX4, for angles of 90, 0, 135, and 45 degrees.

ENTRY POINT:

CALL FPLXIT (IDATA, LEX1, LEX2, LEX3, LEX4, NUMPPL, MERGE)

ARGUMENTS:

IDATA	Scratch array holding two lines of the subimage.
LEX1 - LEX 4	Address indices for the four LEX arrays.
NUMPPL	Number of points in each line of the subimage.
MERGE	TRUE to merge the four LEX arrays into one array.

COMMENTS:

This subroutine reads two lines at a time from the subimage being processed, which is now on scratch disc file 02. After all the LEX arrays are created, they can be merged into one by setting MERGE to TRUE in MAINLN. The merge is performed by adding each array term by term and putting the total into LEX1.

SUBPROGRAMS REQUIRED:

INDEX

CALLED BY:

MAING

SUBPROGRAM TITLE: INDEX
VERSION: I
DATE: September, 1971
UPDATE: September, 1971
AUTHOR: R. M. Haralick
DOCUMENTED BY: R. J. Bosley
PROGRAM LANGUAGE: FORTRAN IV
IMPLEMENTED ON: HW 635

PURPOSE:

To return to the calling program the single subscript for the LEX array that indicates where element (I, L) can be found, given its row and column subscripts I and L.

ENTRY POINT:

INDEX (I, J)

ARGUMENTS:

I	Row subscript for an element in the LEX array.
J	Column subscript for an element in the LEX array.

COMMENTS:

This subprogram is a FUNCTION.

CALLED BY:

FPLXIT
IMOMTR

SUBPROGRAM TITLE: IMOMTR

VERSION: II

DATE: September, 1971

UPDATE: June, 1973

AUTHOR: R. M. Haralick

DOCUMENTED BY: R. J. Bosley

PROGRAM LANGUAGE: FORTRAN IV

IMPLEMENTED ON: HW 635

PURPOSE:

To calculate the moment texture statistics.

ENTRY POINT:

CALL IMOMTR (LEX1, LEX2, LEX3, LEX4, F, IQ, MERGE)

ARGUMENTS:

LEX1 - LEX4	Address indices for the four LEX arrays.
F	Cumulative distribution function.
IQ	Quantized output array of IEQPQ1,
MERGE	TRUE indicates the four LEX arrays have been merged into one.

TEXTURE FEATURES:

1. Angular Second Moment

$$\text{ANGMOM} = \sum_{i=1}^{N_g} \sum_{j=1}^{N_g} P^2(i, j)$$

where N_g is the number of grey tone levels, and $P(i, j)$ is the array of joint probabilities.

2. Entropy

$$\text{ENTROP} = -\sum_{i=1}^{N_g} \sum_{j=1}^{N_g} P(i, j) \log P(i, j)$$

3. Mean

$$\text{AMEAN} = \sum_{i=1}^{N_g} \sum_{j=1}^{N_g} i \cdot P(i, j)$$

4. Variance

$$\text{SGMASQ} = \sum_{i=1}^{N_g} \sum_{j=1}^{N_g} (i - \text{AMEAN})^2 P(i, j)$$

5. Covariance

$$\text{SGMAXY} = \sum_{i=1}^{N_g} \sum_{j=1}^{N_g} (i - \text{AMEAN}) (j - \text{AMEAN}) P(i, j)$$

6. Correlation

$$\text{RATIO} = \frac{\text{SGMAXY}}{\text{SGMASQ}}$$

7. Inverse Moment

$$\text{IVDMOM} = \sum_{i=1}^{N_g} \sum_{j=1}^{N_g} \frac{P(i, j)}{1 + (i-j)^2}$$

8. Average Contrast

$$\text{DIFAVE} = \sum_{k=1}^{N_{g-1}} k \cdot \text{DIF}(k)$$

$$\text{where DIF}(k) = \sum_{|i-j|=k} P(i, j)$$

9. Variance of DIF

$$\text{DIFVAR} = \sum_{k=1}^{N_g-1} k^2 \cdot \text{DIF}(k) - \left[\sum_{k=1}^{N_g-1} k \cdot \text{DIF}(k)\right]^2$$

10. Entrophy of DIF

$$\text{DIFENT} = -\sum_{k=1}^{N_g-1} \text{DIF}(k) \cdot \log(\text{DIF}(k))$$

11. Average of Intensity

$$\text{SUMAVE} = \sum_{k=2}^{2N_g} k \cdot \text{SUM}(k)$$

$$\text{where SUM} = \sum_{i+j=k} P(i, j)$$

12. Variance of SUM

$$\text{SUMVAR} = \sum_{k=2}^{2N_g} k^2 \cdot \text{SUM}(k) - \left[\sum_{k=2}^{2N_g} k \cdot \text{SUM}(k)\right]^2$$

13. Entropy of SUM

$$\text{SUMENT} = -\sum_{k=2}^{2N_g} \text{SUM}(k) \cdot \log(\text{SUM}(k))$$

14. True mean of probability function

$$\text{TMEAN} = \frac{1}{N_g} \sum_{i=1}^{N_g} F(i)$$

COMMENTS:

The three remaining texture features are computed in subroutine COR: CORINF, CORMUT, and CORMAX. If MERGE is TRUE, then these features are computed for only the merged array, LEX1. Otherwise they are computed for each LEX array, corresponding to each of four angles.

SUBPROGRAMS REQUIRED:

INDEX
IEQPQ1
COR

CALLED BY:

MAING

SUBPROGRAM TITLE:	COR
VERSION:	II
DATE:	November, 1972
UPDATE:	June, 1973
AUTHOR:	Sam Shanmugam
DOCUMENTED BY:	R. J. Bosley
PROGRAM LANGUAGE:	FORTRAN IV
IMPLEMENTED ON:	HW 635

PURPOSE:

To calculate three measures of correlation between two discrete random variables X and Y whose joint probabilities of occurrence are known.

ENTRY POINT:

CALL COR(PXY, N, IOPT, Q, COR1, COR2, COR3)

INPUT ARGUMENTS:

PXY	Array of joint probabilities.
N	Size of the array PXY
IOPT	Option flag - if IOPT =0, then only COR1 and COR2 will be calculated; if IOPT =1, then COR3 will also be calculated.
Q	Scratch array of size N x N. This array is used only if IOPT is non-zero.

OUTPUT ARGUMENTS:

COR1	Maximal correlation measure.
COR2	Information measure of correlation.
COR3	Second type of maximal measure.

COMMENTS:

These three correlation measures are the last three texture features. For details of the measures see "Mutual Information and Maximal Correlation As Measure of Dependence," by C. B. Bell, in the Annals of Mathematical Statistics, vol. 43, 1962.

CALCULATIONS:

1. $$\text{COR1} = \frac{H(x, y) - H_1(x, y)}{\max\left(H(x), H(y)\right)}$$

where $$H(x, y) = \sum_i \sum_j \log\left(P_{xy}^2(i, j)\right)$$

$$H_1(x, y) = \sum_i \sum_j \left(\log\left(P_x(i)\, P_y(j)\right)\right) P_{xy}(i, j)$$

$$H(x) = \sum_i \left(\log P_x(i)\right) P_x(i)$$

and $$H(y) = \sum_j \left(\log P_y(j)\right) P_y(j)$$

2. $$\text{COR2} = \sqrt{1 - e^{-2R}}$$

where $R = H_2(x, y) - H(x, y)$

$$H_2(x, y) = \sum_i \sum_j \left(\log P_x(i)\, P_y(j)\right) P_x(i)\, P_y(j)$$

and $P_x(i) = \sum_i P_{xy}(i, j)$, $P_y(j) = \sum_j P_{xy}(i, j)$.

3. COR3 is computed using the eigenvector corresponding to the second largest eigenvalue of QQ^T, where

$$Q(I, J) = \frac{P_{xy}(i, j)}{\sqrt{P_x(i)\, P_y(j)}}.$$

CALLED BY:

IMOMTR

SUBPROGRAM TITLE:	IEQPQ1
VERSION:	I
DATE:	September, 1971
UPDATE:	September, 1971
AUTHOR:	D. Goel
DOCUMENTED BY:	R. J. Bosley
PROGRAM LANGUAGE:	FORTRAN IV
IMPLEMENTED ON:	HW 635

PURPOSE:

To determine k levels of equal probability quantization for an array for which the cumulative distribution function is known for all elements.

ENTRY POINT:

CALL IEQPQ1 (N, K, F, IQ, IMIN)

ARGUMENTS:

N	Number of items in array F to be equal probability quantized.
K	Number of quantizing levels.
F	Input array to be quantized.
IQ	Output array of quantized levels.
IMIN	The lowest possible level in the input data.

CALLED BY:

IMOMTR

SUBPROGRAM TITLE:	RITOWT
VERSION:	II
DATE:	September, 1971
UPDATE:	June, 1973
AUTHOR:	R. M. Haralick
DOCUMENTED BY:	R. J. Bosley
PROGRAM LANGUAGE:	FORTRAN IV
IMPLEMENTED ON:	HW 635

PURPOSE:

To output onto printer, cards, or tape the texture features.

ENTRY POINT:

CALL RITOWT (LEX1, LEX2, LEX3, LEX4, G, IQ, MERGE, IF, PICTUR)

ARGUMENTS:

LEX1 -LEX4	Address indices for the LEX arrays.
G	CDF for the image data
IQ	Quantized output of IEQPQ1 of NQUANT levels.
MERGE	TRUE indicates that the four LEX arrays have been merged into one array.
IF	File code for output tape.
PICTUR	TRUE indicates that a picture of the subimage has been printed.

COMMENTS:

The output format for the listing is slightly different depending upon the PICTUR and merge options. The PNCH option determines whether cards or tape or neither are used to output the 17 texture features for each subimage. In any case, the texture features are listed on the printer. If PNCH = 1HY in the $PARAM card in MAINLN, then cards are punch according to the following formats:

1. for MERGE = TRUE: M1, N1, NFT, ANGMOM, ENTROP, RATIO, SGMASQ, SGMAXY, AMEAN, VIDMOM, KOUNT/ TMEAN, DIFENT, DIFAVE, DIFVAR, SUMENT, SUMAVE, SUMVAR, KOUNT+1/CORINF, CORMUT, CORMAX, KOUNT+2.

   FORMAT (I2, 1X, 2I2, 1X, 7F9.5, I9/8X, 7F9.5, I9/8X, 3F9.5, 36X, I9), where (M1, N1) is the subimage row, column coordinate.

2. for MERGE = FALSE: M1, N1, NFT/ANGMOM(4), ENTROP(4), KOUNT/RATIO(4), SGMASQ(4), KOUNT+1/SGMAXY(4), AMEAN(4), KOUNT+2/VIDMOM(4), TMEAN(4), KOUNT+3/ DIFENT(4), DIFAVE(4), KOUNT+4/DIFVAR(4), SUMENT(4), KOUNT+5/SUMAVE(4), SUMVAR(4), KOUNT+6/CORINF(4), CORMUT(4), KOUNT+7/CORMAX(4), KOUNT+8, where (4) denotes four values, one for each angle.

   FORMAT (/'THE SCENE(', I2, 1,', I2,') HAS BEEN REDUCED BY', I5/8(1X, 8F9.5, I7/)/1X, 4F9.4, 38X, I5).

If tape output on file code 'IF' is selected by PNCH = 1HT, then the texture features are written in binary as follows:
WRITE(IF) M1, N1, NFT, ANGMOM(K), ENTROP(K), RATIO(K), SGMASQ(K) , SGMAXY(K), AMEAN(K), VIDMOM(K), TMEAN(K), DIFENT(K), DIFAVE(K), DIFVAR(K), SUMENT(K), SUMAVE(K), SUMVAR(K), CORINF(K), CORMUT(K), CORMAX(K), where K is one for MERGE = TRUE and is 4 for MERGE = FALSE denoting the number of values for each measure.
If neither cards nor tape output is selected, then the LEX arrays are listed after the texture features.

CALLED BY:

MAING

IV.1-c Cross-Band Texture Analysis Programs - Documentation

SPECTR
- GETIM / GETIT
  - ERTS
- DIFFER
- COVAR
  - MNCVIN / MNCV
- CORREL

CROSS-BAND TEXTURE ANALYSIS

PROGRAM TITLE: SPECTR

VERSION: I

DATE: January, 1974

UPDATE: January, 1974

AUTHOR: Robert J. Bosley

DOCUMENTED BY: Robert J. Bosley

PROGRAM LANGUAGE: FORTY

IMPLEMENTED ON: HW635

PURPOSE:

This program is the mainline of the spectral-textural analysis which obtain spectral-textural features for land-use classification of ERTS MSS data.

ENTRY POINT:

CALL SPECTR (IMAGE, X, ILINE, IXDIM, IYDIM, NDIN)

INPUT ARGUMENTS:

IMAGE (IYDIM, IXDIM, IDIN)	Array containing a subimage.
X (IYDIM, IXDIM, IDIN)	Array containing the difference image.
ILINE	Array one ERTS line is read into.
IXDIM	Column dimension of IMAGE, X.
IYDIM	Row dimension of IMAGE, X.
NDIN	Number of components, bands, in IMAGE, X.

INPUT PARAMETERS: under NAMELIST 'PARAM':

NDIM	Number of components desired in IMAGE: Assumed to be NDIN.
NUMLIN	Number of lines in subimage; Assumed to be IYDIM.

NUMPPL	Number of columns in subimage; Assumed to be IYDIM.
FMT	Format used to output elements of covariance matrix; assumed to be 'E11.4'.
TITLE	80 column title for run.
OPT	TRUE to print covariance matrix; assumed FALSE.
IDIST	Distance between neighboring cells for difference array; assumed to be 1.
IRSTRT	Starting row in ERTS image; assumed to be 1.
IRSTOP	Stopping row in ERTS image; assumed as last row.
LAPHOR	Number of horizontal points that subimages overlap; assumed to be 0.
LAPVER	Number of vertical points that subimages overlap; assumed to be 0.
PNCH	TRUE for output on cards, FALSE for output to file code 01 for tape or disc.

EXAMPLE OF DRIVER:

```
DIMENSION IMAGE (16, 17, 8), X (16, 17, 8), ILINE (3300)
EQUIVALENCE (IMAGE, X, ILINE(130))
IXDIM =17
IYDIM =16
NDIN =8
CALL SPECTR (IMAGE, X, ILINE, IXDIM, IYDIM, NDIN)
STOP
END
```

This driver will set up the spectral-textural analysis mainline SPECTR to process 16 x 16 subimages over 8 components with IDIST =1. Note IXDIM must include NUMPPL plus IDIST points, and array ILINE must have at least NUMPPL × NDIM points outside of any other array. These points form array XLINE in COVAR.

REQUIREMENTS:

1. Core - 25k for IMAGE (32, 33, 8)
2. ERTS input tape must be on file code 'ES'
3. Random access disc file on file 11, eg. $ FILE II AIIR, OR.
4. Subprograms required:
   - DRIVER
   - SPECTR
   - GETIM
   - SETDIM (Fortran callable program to initialize HEMP package)
   - GETIT
   - ERTS
   - DIFFER
   - COVAR
   - MNCVIN
   - MNCV
   - CORREL
   - SFA07F
   - HEMDET (Fortran callable program from HEMP library to solve for determinant of matrix).

COMMENTS:

These analysis programs obtain a series of NUMLIM by NUMPPL by NDIM subimages from the ERTS input tape and outputs a feature vector with (1 + NDIM (NDIM-1)/2) components for each subimage. The input data is processed in horizontal rows of subimages that may overlap both horizontally and vertically. The distance between neighboring resolution cells used to get the difference array is variable. Note that IXDIM must include NUMPPL plus IDIST points, and that the array ILINE must have at least NUMPPL × NDIM points outside of any other array because these are used for array XLINE in COVAR. Other than this, arrays IMAGE, X, and ILINE may be equivalenced to conserve core, as in the example for a DRIVER. The first feature component on the output file is the entropy measure, and the remaining NDIM (NDIM-1)/2 components are elements of the correlation matrix. See the GETIM subprogram for a listing of all 8 possible components for a subimage.

CALLED BY:

DRIVER

CROSS-BAND TEXTURE ANALYSIS

SUBPROGRAM TITLE: GETIM
VERSION: I
DATE: January, 1974
UPDATE: January, 1974
AUTHOR: Robert J. Bosley
DOCUMENTED BY: Robert J. Bosley
PROGRAM LANGUAGE: FORTY
IMPLEMENTED ON: HW635

PURPOSE:

To get a row of subimages from the ERTS input tape and copy them onto a random access disc file.

ENTRY POINTS:

CALL GETIM (ILINE, IDIST, NDIM, IRSTRT, IRSTOP, NUMLIN, NUMPPL, LAPVER, LAPHOR, NHOR, INCR, IPEND)

CALL GETIT

INPUT ARGUMENTS:

ILINE	Array into which one ERTS line is read.
IDIST	Distance between neighboring resolution cells whose differences form the difference array, X.
NDIM	Number of components in each resolution cell.
IRSTRT	Starting line of ERTS data.
IRSTOP	Stopping line of ERTS data.
NUMLIN	Number of lines in each subimage.
NUMPPL	Number of points per line (columns) in each subimage.
LAPVER	Number of lines that subimages overlap.
LAPHOR	Number of columns that subimages overlap.

OUTPUT ARGUMENTS:

NHOR	Number of horizontal overlapping subimages in image.
INCR	Horizontal increment to the first column of the next subimage in the row.
IPEND	Last cell in the row.

REQUIREMENTS:

Random access disc file on file code II: and FILE II, AIIR, IOR.

COMMENTS:

This program is initialized by calling GETIM which initializes the ERTS input tape and sets up the disc on file II for random access with fixed length records of NDIM words, up to a maximum of NDIM =8. The eight possible greytone components are:

1. Band 2
2. Band 3
3. Band 4
4. Band 2 x Band 2
5. Band 3 x Band 3
6. Band 4 x Band 4
7. Band 2 x Band 3
8. Band 2 x Band 4, where band 1 through 4 is MSS band 4 through 7.

It is suggested that all eight components be used and the best of these be selected for feature vector components. This gives a total of 29 components, including the entropy measure.

After calling GETIM, all further calls are made to GETIT which goes down the input data file line by line copying to the random access disc an entire row of NHOR subimages of NDIM components. Note that MSS band 4 (band 1 here) has been deleted because of its high correlation with MSS band 5 (band 2 here).

CALLED BY:

SPECTR

SUBPROGRAM TITLE:	ERTS
VERSION:	II
DATE:	September, 1972
UPDATE:	November, 1973
AUTHOR:	G. Funnels
DOCUMENTED BY:	R. Bosley
PROGRAM LANGUAGE:	GMAP
IMPLEMENTED ON:	HW635

PURPOSE:

To read 7-track ERTS MSS data tapes.

ENTRY POINTS:

CALL EINIT (NOLS)
CALL ESKIP (NOSK)
CALL EREAD (I, LN)
CALL EREWND

ARGUMENTS:

NOLS	Number of words per scan line; returned by EINIT.
NOSK	The number of records to skip.
I	The array into which the NOLS words of data from a line of ERTS data is placed.
LN	Returned by EREAD giving the line number of the line of data returned. If LN =0, the end of file was reached on the ERTS tape.

ERROR FLAGS:

MB	EREAD buffer is not large enough for a block of ERTS data.
AI	EINIT was called twice.
NI	EINIT was not called before calling EREAD, ESKIP, or EREWND.

UE	EOF encountered while reading ID or annotation blocks on ERTS tape.
EF	EOF encountered while trying to skip records in ESKIP.

COMMENTS:

EINIT initializes the ERTS tape so that data may be read, and must be called first. ESKIP skips over NOSK records (scan lines). EREWND rewinds the ERTS tape. EINIT must not be called twice. The data placed into array I by EREAD is in standard corresponding point forms. Since the ERTS MSS data has four channels, there are actually NDLS/4 points or cells per scan line.

REQUIREMENTS:

ERTS tape must be on file code 'ES'.

CROSS-BAND TEXTURE ANALYSIS

SUBPROGRAM TITLE: DIFFER
VERSION: 1
DATE: January, 1974
UPDATE: January, 1974
AUTHOR: Robert J. Bosley
DOCUMENTED BY: Robert J. Bosley
PROGRAM LANGUAGE: FORTY
IMPLEMENTED ON: HW635
PURPOSE:

To calculate the nearest neighbor difference array.

ENTRY POINT:

CALL DIFFER (IA, X, IXDIM, IYDIM, NDIM, IDIST, NUMPPL, NUMLIN)

INPUT ARGUMENTS:

IA(IYDIM, IXDIM, NDIM)	Array containing the subimage being processed.
IXDIM	Column dimension of IA and X.
IYDIM	Row dimension of IA and X.
NDIM	Number of components of each resolution cell.
NUMPPL	Number of columns in the subimage.
NUMLIN	Number of lines in the subimage.

OUTPUT ARGUMENTS:

X(IYDIM, IXDIM, NDIM)	Array of nearest neighbor differences.

COMMENTS:

This subroutine will replace the original subimage in array IA with the nearest neighbor horizontal difference: (I1 - J1, I2 - J2, . . . , IN - JN) where I and J are N-dimensional horizontally neighboring resolution cells separated by distance IDIST. Arrays IA and X may be equivalenced to occupy the same area of core. Note that the absolute value is used to get the differences. This gives only the positive half of the distribution of differences I-J and J-I. This shifts the mean of the distribution from the origin and must be accounted for in COVAR when the covariance matrix of the difference array is calculated.

CALLED BY:

SPECTR

CROSS-BAND TEXTURE ANALYSIS

SUBPROGRAM TITLE:	COVAR
VERSION:	I
DATE:	January, 1974
UPDATE:	January, 1974
AUTHOR:	Robert J. Bosley
DOCUMENTED BY:	Robert J. Bosley
PROGRAM LANGUAGE:	FORTY
IMPLEMENTED ON:	HW635

PURPOSE:

To calculate the covariance matrix of the difference array.

ENTRY POINT:

CALL COVAR (XLINE, NDIM, NUMPPL, X, IXDIM, IYDIM, NUMLIN, NDIN, COV)

INPUT ARGUMENTS:

XLINE(NDIM, NUMPPL)	Array used to send one line of the difference array to MNCVIN.
NDIM	Number of components of each vector in X.
NUMPPL	Number of columns of vectors in X.
X	Nearest neighbor difference array.
IXDIM, IYDIM	Column, row dimensions of array X.
NUMLIN	Number of rows of vectors in X.
NDIN	Dimension of COV array.

OUTPUT ARGUMENTS:

COV (NDIN, NDIN)	Covariance matrix of the difference array X.

REQUIREMENTS:

Subroutine MNCVIN.

COMMENTS:

Array XLINE is formed from the first NDIM x NUMPPL points of array ILINE in SPECTR. Hence at least the first NDIM x NUMPPL words of ILINE must not be equivalenced into array X. Since only the positive differences were used to make array X by DIFFER, the mean is reset to zero for each component .

CALLED BY:

SPECTR

CROSS-BAND TEXTURE ANALYSIS

SUBPROGRAM TITLE: MNCVIN
VERSION: 1
DATE: August, 1973
UPDATE: August, 1973
AUTHOR: James D. Young
DOCUMENTED BY: James D. Young
PROGRAM LANGUAGE: FORTRAN IV or FORTY
IMPLEMENTED ON: HW635

PURPOSE:

To calculate the mean vector and covariance matrix for each category of a set of vectors, based on a specified percentage of the vectors randomly chosen within the set.

ENTRY POINTS:

CALL MNCVIN (NVPCAL, NDIM, NCALL, PERCNT, NCAT, X, NTRUTH, COV, XMEAN, SCTMEN, SAMSZ, IERROR, JERROR)

CALL MNCV

INPUT ARGUMENTS:

NVPCAL	Number of vectors per call.
NDIM	Dimension of data vectors.
NCALL	Number of calls.
PERCNT	Percentage of total number of vectors from which the mean and covariance matrices will be calculated.
NCAT	Number of categories considered; set to 1 if only one set of statistics will be calculated for all data, set to the number of categories in data set if one set of statistics will be calculated for each category.

X (NDIM, NVPCAL)	Matrix containing input data vectors in its columns.
NTRUTH (NVPCAL)	Vector containing the ground truth integers, 1 through NCAT, associated with the data vectors of X; if NCAT is 1 this vector is ignored.

OUTPUT ARGUMENTS:

COV(NDIM, NDIM, NCAT)	Matrix containing covariance matrices of the data.
XMEAN (NDIM, NCAT)	Matrix containing mean vectors of the data.
SCTMEN(NDIM, NCAT)	Scratch matrix.
SAMSZ (NCAT)	Vector with the number of vectors used to calculate the statistics for each category.
IERROR	Error flag when returned non-zero: 1. if NVPCAL .LE. 0 2. if NDIM .LE. 0 3. If NCAL .LE. 0 4. if PERCNT .GT. 100. or so small that less than 2 vectors will be used to calculate all the statistics. 5. if NCAT .LE. 0
JERROR	Error flag when returned non-zero: 1. if an illegal ground truth label is formed.

COMMENTS:

One call to MNCVIN Initializes this routine. Calls to MNCV should be performed NCALL times, each with the next line of vectors in X. After MNCV has been called NCALL times, the mean vector and covariance matrix for each category is completed. For use in the Spectral-Textural Analysis programs, PERCNT is set to 100 and NCAT is set to 1.

CROSS-BAND TEXTURE ANALYSIS

SUBPROGRAM TITLE:	CORREL
VERSION:	I
DATE:	January, 1974
UPDATE:	January, 1974
AUTHOR:	Robert J. Bosley
DOCUMENTED BY:	Robert J. Bosley
PROGRAM LANGUAGE:	FORTY
IMPLEMENTED ON:	HW 635

PURPOSE:

To calculate the correlation matrix given the covariance matrix of the difference array.

ENTRY POINT:

CALL CORREL (COV, NDIM, COR)

INPUT ARGUMENTS:

COV	Covariance matrix of the difference array
NDIM	Order of matrix COV.

OUTPUT ARGUMENTS:

COR	Correlation matrix of COV.

CALLED BY:

SPECTR

IV.1-d Piecewise Linear Classification Programs - Documentation

RCLASS
XIN
LINEAR
WEIGHT

LINEAR DISCRIMINANT CLASSIFIER

SUBPROGRAM TITLE:	RCLASS
VERSION:	I
DATE:	September, 1972
UPDATE:	November, 1972
AUTHOR:	Sam Shanmugam
DOCUMENTED BY:	Sam Shanmugam
PROGRAM LANGUAGE:	FORTRAN IV
IMPLEMENTED ON:	HW 635

PURPOSE:

This subroutine is the mainline for routines which implement a decision rule using piecewise linear discriminant functions. It calls subroutine XIN to read in the pattern vectors, and calls LINEAR to get the decision rule and classify the pattern vectors.

ENTRY POINT:

CALL RCLASS (WORK, ISIZE)

ARGUMENTS:

WORK	Scratch array of size ISIZE which holds the training vectors and the weight vectors.
ISIZE	Dimension of array WORK.

INPUT PARAMETERS:

Card 1. Format (4I1)

NOPT1	Set to 1 to print out training patterns; otherwise set to zero.
NOPT2	Set to 1 to print out test patterns; otherwise set to zero.
NOPT3	Set to 0 to list only the contingency table for the training patterns; otherwise set to 1 and the classification of each training pattern is listed as well as the contingency table.

NOPT4	Set to 0 for only the contingency table of the test set; otherwise set to 1 for the classification of each test pattern as well.

Card 2. Format (515)

NTOT	The total number of pattern vectors in the data set.
NPART	NPART out of every ten pattern vectors in the data set will be used for training. The remaining will be used as test patterns.
NDIM	The number of measurements per vector plus two.
NC	The number of ground truth categories.
NPAIR	Twice the maximum number of training patterns in any one category.

REQUIREMENTS:

1. Maximum number of categories is 15.
2. Maximum number of components, NDIM, is 100.
3. Pattern vectors must be sorted by category.
4. A scratch disc file must be on 02.
5. ISIZE must be at least

   NDIM (NTRAIN + 10) + 1000
   + (NC (NC + 1)/2) ND + NPAIR · ND

   where ND = NDIM -1

   and NTRAIN = number of training patterns.
6. Pattern vectors must be written in binary to disc file 01 as follows:

   WRITE (01) IGT, MI, NI, NFT, (FEAT(I), I =1, NMEAS)

   where

   IGT Is the ground truth category.

   M1, N1, NFT are not used - may be used as ID tags

   FEAT Is the feature vector

   NMEAS is the number of measurements per feature vector.

7. Subprograms required

DRIVER
RCLASS
XIN
LINEAR
WEIGHT

ERROR FLAGS:

If ISIZE is too small, processing is terminated and an error message is listed.

THEORY:

Using a regression type algorithm the program obtains a set of hyperplanes for separating the training patterns of different category pairs. A total of NC(NC-1)/2 hyperplanes are determined. Test patterns are identified by taking a majority vote on this set of hyperplanes. For complete details, see "Introduction to Statistical Pattern Recognition" by Y. Fukunaga, Academic Press, 1972.

COMMENTS:

The input data file on 01 should have a total of NTOT logical records in binary. Each logical record must be of length NDIM + 2 words, where word 1 is the ground truth category and words set is sorted into training and test sets according to NPART. Training vectors are stored in WORK and test vectors are copied to disc 02.

CALLED BY:

DRIVER

EXAMPLE OF DRIVER:

```
DIMENSION WORK (10000)
ISIZE = 10000
CALL RCLASS (WORK, ISIZE)
STOP
END
```

SUBPROGRAM TITLE: XIN
VERSION: II
DATE: September, 1972
UPDATE: December, 1973
AUTHOR: Sam Shanmugam
DOCUMENTED BY: Robert J. Bosley
PROGRAM LANGUAGE: FORTRAN IV
IMPLEMENTED ON: HW 635

PURPOSE:

To read the parameter cards and the Input data set, copying the test vectors to disc.

ENTRY POINT:

CALL XIN (WORK, U)

ARGUMENTS:

WORK	Array training vectors are read into
U	Scratch array

CALLED BY:

RCLASS

COMMENTS:

This version does not use subroutine POST to position the input file 01. Hence the input data file must be the first file on 01.

LINEAR DISCRIMINANT CLASSIFIER

SUBPROGRAM TITLE:	LINEAR
VERSION:	I
DATE:	September, 1972
UPDATE:	November, 1972
AUTHOR:	Sam Shanmugam
DOCUMENTED BY:	Sam Shanmugam
PROGRAM LANGUAGE:	FORTRAN IV
IMPLEMENTED ON:	HW 635

PURPOSE:

Using the set of training patterns, this program obtains a set of hyperplanes for pairwise separation of training patterns of different categories. The program also identifies the test patterns on a majority vote on the hyperplanes and outputs contingency table.

ENTRY POINT:

CALL LINEAR (XTRAIN, XTEST, W, U, DUMMY)

ARGUMENTS:

XTRAIN	Matrix containing training patterns.
XTEST	Matrix containing test patterns.
W	Array of weight vectors.
U	Matrix used for calculating the boundary between category pairs.
DUMMY	Scratch array.

CALLED BY:

RCLASS

SUBPROGRAMS REQUIRED:

WEIGHT

SUBPROGRAM TITLE: WEIGHT
VERSION: I
DATE: September, 1972
UPDATE: November, 1972
AUTHOR: Sam Shanmugam
DOCUMENTED BY: Sam Shanmugam
PROGRAM LANGUAGE: FORTRAN IV
IMPLEMENTED ON: HW 635

PURPOSE:

To find the minimum mean square fit hyperplane for separating the training patterns of two different categories.

ENTRY POINT:

CALL WEIGHT (U, DUMMY, WT, ND, NIJ)

INPUT ARGUMENTS:

U	Array containing the patterns of category I and category J.
DUMMY	Scratch array.
ND	ND+1 is the dimension of the pattern vector.
NIJ	Number of vectors in U.

OUTPUT ARGUMENTS:

WT	Weight vector which defines the hyperplane separating categories I and J.

CALLED BY:

LINEAR

SUBPROGRAMS REQUIRED:

MINV	Matrix inversion program from the IBM Scientific Subroutine Package.

IV.2-a ERTS Retrieval Program Listings

RETV
- ERTS
- PIXEY
  - ZEQUAN
  - PITCHR
- WRTDSK
- RDDSK1
  - KEQUAN
- RDDSK2

```
CRETV          ERTS RETRIEVAL  PROGRAM
C
C      VERSION 1     WRITTEN BY RJ BOSLEY                          JAN 1974
C
C      DESCRIPTION OF PROGRAM.
C         THIS IS THE MAINLINE FOR ERTS RETRIEVAL PROGRAMS WHICH WILL
C      RETRIEVE DATA FROM STANDARD NASA ERTS DATA TAPES AND OUTPUT  THE
C      IMAGE DATA IN A PICTURE, OR GREYTONE LISTING, OR COPY IT ONTO AN
C      OUTPUT TAPE ON FILE CODE 'IFIL'.  COORDINATES  MAY BE EITHER IN
C      MILLIMETERS OR IN ROW, COLUMN COUNT.  FOR REASONABLE EFFICIENCY
C      DATA IS READ IN BLOCKS OF 41 LINES BY 41 POINTS.  ONE ERTS TAPE
C      WILL THEN BE COVERED BY 21 HORIZONTAL BLOCKS. LEAVING 4 POINTS
C      LEFT AT THE END OF EACH LINE.  NOTE--SOMETIMES ON THE FIRST TAPE
C      OF AN IMAGE, THE FIRST FOUR POINTS ARE 255 AND WILL ADVERSELY
C      AFFECT A PICTURE PRINTOUT.  IF SO, SKIP THE FIRST FOUR POINTS BY
C      SETTING ICSTRT=5.
C         NOTE---IF THE POINT ICSTRT IS NOT A MULTIPLE OF 41 FROM THE
C      END OF THE LINE, THEN THE LAST POINTS MAY NOT NECESSARILY BE
C      OUTPUTTED SINCE THE BLOCKS ARE DETERMINED BY THE ICSTRT POSITION.
C      THEREFORE IT IS SUGGESTED THAT IF THE DATA OUT TO THE END OF EACH
C      LINE IS DESIRED, THAT THE COORDINATES BE SPECIFIED  IN ROW,COLUMN
C      FORMAT AND THAT THEY BE PREDETERMINED SO AS TO OUTPUT THE DESIRED
C      POINTS IN THE ERTS LINE.
C         A SPECIAL PICTURE RUN TO PRINT OUT AND REDUCE ANY IMAGE OVER
C      THE ENTIRE TAPE CAN BE MADE BY SETTING SPIC TO .TRUE. IN $PARAM
C      AND THEN SPECIFYING PICTURE PARAMETERS UNDER NAMELIST $PICTUR.
C      SEE THE PIXEY PROGRAM FOR AN EXAMPLE OF A SAMPLE RUN.
C
C      INPUT PARAMETERS UNDER NAMELIST /PARAM/.
C             NBAND                BAND NUMBER TO BE SELECTED
C                                   SET=5 FOR ALL BANDS
C             IRSTRT,ICSTRT        ROW,COLUMN STARTING COORDINATES
C             IRSTOP,ICSTOP        ROW,COLUMN STOPING COORDINATES
C             NHOR                 NUMBER OF HORIZONTAL 41 X 41 BLOCKS
C                                    THAT COVERS AREA SELECTED
C             NVERT                NUMBER OF VERTICAL BLOCKS COVERING
C                                    AREA SELECTED
C             IPSTR,IPEND          STARTING,ENDING POINTS IN ERTS LINE
C             TITLE                TITLE FOR THIS RUN
C             IFIL                 OUTPUT FILE FOR TAPE, SET TO 02
C             ILINE                ARRAY WHERE ERTS LINE IS READ INTO
C             IMAGE                ARRAY TO STORE 41 X 41 SUBIMAGE BLOCK
C             PIC                  TRUE FOR PICTURE OUTPUT
C             SPIC                 TRUE FOR SPECIAL PICTURE RUN
C             QUAN                 TRUE FOR QUANTIZED PICTURE OUTPUT
C             PPNT                 TRUE FOR GREY-TONE LISTING
C             TAPE                 TRUE FOR TAPE OUTPUT
C             MILLI                TRUE IF COORD ARE SPECIFIED IN MM.
C             SMALL                TRUE IF MM. COORD COME FROM A SMALL
C                                    70 X 70 NEGATIVE---OTHERWISE ASSUMED
```

```
                                      TO BE FALSE--FROM A 7 X 7 INCH PRINT
C
C
C     RESTRICTIONS.
C        1. ERTS INPUT TAPE MUST BE ON FILE CODE 'ES'.
C        2. FOUR DISC FILES MUST BE ON FILES 11,12,13,14.
C        3. ANY OUTPUT TAPE MUST BE POSITIONED ON FILE CODE 'IFIL'.
C        4.  ALL COORDINATES MUST BE DETERMINED RELATIVE TO THE INPUT
C           TAPE---THAT IS, ICSTOP MUST NOT EXCEED 824 COLUMNS OR
C           46MM.  STARTING COORDINATES MUST BE AT LEAST 1.
C
C     SUBPROGRAMS REQUIRED.
C           RETV
C              ERTS (WITH EREWND)
C              PIXEY
C                 ZEQUAN
C                 ERTS
C                 PITCHR
C              WRTDSK
C              RDDSK1
C                 KEQUAN
C                 PITCHR
C              RDDSK2
C
C
      DIMENSION IMAGE(41,82),ILINE(3300),TITLE(14)
      EQUIVALENCE (IMAGE(1,1),ILINE(1))
      LOGICAL EOF,PIC,SPIC,QUAN,PRNT,TAPE,SMALL,MILLI
      DATA BLANK/'        '/
      NAMELIST /PARAM/NBAND,PIC,QUAN,PRNT,TAPE,IRSTRT,ICSTRT,IRSTOP,
     1                  ICSTOP,IFIL,TITLE,MILLI,SPIC
C
C
C     ****** SECTION I --- SET UP PARAMETERS FOR PROCESSING ******
C
C                                   INITIALIZE ERTS AND EOF FLAG
      CALL EINIT(LENGTH)
      CALL FLGEOF(05,EOF)
C                                   INITIALIZE PARAMETERS
1001  PIC=.TRUE.
      SPIC=.FALSE.
      QUAN=.TRUE.
      PRNT=.FALSE.
      TAPE=.FALSE.
      SMALL=.FALSE.
      MILLI=.FALSE.
      DO 7 I=1,14
7     TITLE(I)=BLANK
      IFIL=2
      NBAND=2
      IRSTRT=1
```

```
      ICSTRT=1
      IPSTOP=0
      ICSTOP=0
C                                READ IN PARAMETERS
      READ(05,PARAM)
C                                CHECK FOR END OF CARDS
      IF(EOF)   STOP
C                                WRITE OUT THE PARAMETERS
      WRITE(6,1)  (TITLE(I),I=1,14),NBAND
      IF(NBAND.EQ.5)  WRITE(6,2)
      IF(TAPE)  WRITE(6,3) IFIL
      IF(PIC)  WRITE(6,4) QUAN
      WRITE(6,5)  IRSTRT,IRSTOP,ICSTRT,ICSTOP
1     FORMAT(20X,'ERTS RETRIEVAL PROGRAM     VERSION II'///1X,14A6////
     1        ' BAND NUMBER IS ',I3)
2     FORMAT(' ALL FOUR ERTS BANDS WILL BE PROCESSED')
3     FORMAT(' AN OUTPUT TAPE WILL BE CREATED ON FILE ',I3)
4     FORMAT(' PICTURE QUANTIZATION IS ',L1)
5     FORMAT(' STARTING ROW IS      ',I6,10X,'ENDING ROW IS     ',I6/
     1         ' STARTING COLUMN IS ',I6,10X,'ENDING COLUMN IS ',I6)
C                                CHANGE MM COORD INTO ROW AND COL
      IF(.NOT.MILLI) GO TO 19
      WRITE(6,6)
6     FORMAT(' COORDINATES ARE IN MILLIMETERS')
C                                GET THE CORRECT IMAGE SIZE
      IF(SMALL)  GO TO 10
C                                ASSUME THAT A PRINT HAS 180MM VERT
C                                AND 184 MM ACROSS OR 2336 ROWS VERT
C                                AND 3296 POINTS ACROSS.
C                                HOWEVER--SINCE THE TAPE ONLY COVERS
C                                ONE FOURTH ACROSS AN IMAGE, THEN USE
C                                46MM ACROSS AND 824 COLUMNS ACROSS.
      IRSTRT=IRSTRT*2336/180-12
      IRSTOP=IRSTOP*2336/180
      ICSTRT=ICSTRT*824/46-17
      ICSTOP=ICSTOP*824/46
      GO TO 15
C                                DO SMALL NEGATIVE COORDINATES
C                                A NEGATIVE IS 70MM BY 70MM
10    IRSTRT=IRSTRT*2336/70-31
      IRSTOP=IRSTOP*2336/70
      ICSTRT=ICSTRT*824*2/35-46
      ICSTOP=ICSTOP*824*2/35
C                                WRITE OUT THE NEW COORDINATES
15    WRITE(6,16)  IRSTRT,IRSTOP,ICSTRT,ICSTOP
16    FORMAT('    COORDINATES IN ROW AND COLUMN FORMAT ARE---'/
     1     '    STARTING ROW IS     ',I6,10X,'ENDING ROW IS     ',I6/
     1     '    STARTING COLUMN IS ',I6,10X,'ENDING COLUMN IS ',I6)
C                                CHECK COORDINATES SO THEY FALL IN
C                                  ONE TAPE
```

```
                                                                  RETV0151
19    IF(ICSTOP.GT.824)  GO TO 20                                 RETV0152
C                                  CHECK PARAMETERS               RETV0153
      IF(NBAND.GT.5)  GO TO 20                                    RETV0154
      IF(NBAND.LT.1)  GO TO 20                                    RETV0155
      IF(IRSTRT.LT.1)  GO TO 20                                   RETV0156
      IF(ICSTRT.LT.1)  GO TO 20                                   RETV0157
      IF(IRSTOP.LT.IRSTRT)  GO TO 20                              RETV0158
      IF(ICSTOP.GT.ICSTRT)  GO TO 22                              RETV0159
20    WRITE(6,21)                                                 RETV0160
21    FORMAT(///' ***EXECUTION TERMINATED FOR THIS RUN--ERROR IN PARAMETRETV0161
     1ERS'/1H1)                                                   RETV0162
      GO TO 1001                                                  RETV0163
C                                  CHECK FOR SPECIAL PICTURE RUN  RETV0164
22    IF(.NOT.SPIC)  GO TO 18                                     RETV0165
      CALL PIXEY(ILINE,ILINE,IRSTRT,IRSTOP,ICSTRT,ICSTOP,QUAN,NBAND)  RETV0166
      GO TO 1001                                                  RETV0167
C                                  POSITION THE INPUT TAPE        RETV0168
18    NOSK=IRSTRT-1                                               RETV0169
      CALL EREWND                                                 RETV0170
      IF(NOSK.NE.0)  CALL ESKIP(NOSK)                             RETV0171
      WRITE(6,23)  LENGTH                                         RETV0172
23    FORMAT(' LENGTH OF ONE ERTS LINE IS',I6)                    RETV0173
      IF(LENGTH.LE.3300) GO TO 25                                 RETV0174
      WRITE(6,24)                                                 RETV0175
24    FORMAT(///' LENGTH OF LINE ON ERTS EXCEEDS DIMENSION OF ARRAY ILINRETV0176
     1E---EXECUTION TERMINATED')                                  RETV0177
      STOP                                                        RETV0178
C                                  DETERMINE THE NUMBER OF BLOCKS RETV0179
C                                  ALLOW 15 PTS TO BE CUT OFF BEFORE  RETV0180
C                                  STARTING A NEW STRIP OF BLOCKS RETV0181
25    NHOR=((ICSTOP-ICSTRT-15)/41)+1                              RETV0182
C                                  RESET ENDING COLUMN            RETV0183
27    ICSTOP=NHOR*41+ICSTRT-1                                     RETV0184
      IF(ICSTOP.LE.824)  GO TO 26                                 RETV0185
      NHOR=NHOR-1                                                 RETV0186
      GO TO 27                                                    RETV0187
26    NVERT=((IRSTOP-IRSTRT-15)/41)+1                             RETV0188
      IRSTOP=41*NVERT+IRSTRT-1                                    RETV0189
C                                  SET STARTING AND ENDING PTS IN ILINE  RETV0190
      IPSTR=(ICSTRT-1)*4                                          RETV0191
      IPEND=ICSTOP*4                                              RETV0192
C                                  WRITE OUT NO OF BLOCKS AND STOPS   RETV019
      WRITE(6,28)  NHOR,NVERT,IRSTOP,ICSTOP                       RETV019
28    FORMAT(' NUMBER OF HORIZONTAL BLOCKS IS',I5/                RETV019
     1       ' NUMBER OF VERTICAL   BLOCKS IS ',I5/               RETV019
     1       ' REVISED STOPPING ROW     IS NOW ',I6/              RETV019
     1       ' REVISED STOPPING COLUMN IS NOW ',I6)               RETV019
C                                                                 RETV019
C                                                                 RETV020
C     ****** SECTION II --- PROCESS THE ERTS DATA TAPE ******
```

```
C
C                                    PART 1--READ THE DATA
C                                    GO THRU EACH VERTICAL BLOCK
      DO 29 ID=11,14
29    REWIND ID
      DO 100 ILINE=IRSTRT,IRSTOP
      CALL EREAD(ILINE,LN)
C                                    CHECK FOR ERROR
      IF(LN.NE.0)  GO TO 90
      WRITE(6,80)LN
80    FORMAT(///' ***EXECUTION TERMINATED---EOF DETECTED ON ERTS TAPE')
      GO TO 301
C                                    WRITE NHOR RECORDS ON DISC,ONE DISC
C                                       PER BAND
90    CALL WRTDSK(ILINE,NHOR,IPSTR,IPEND,NBAND)
100   CONTINUE
C                                    PART 2--OUTPUT THE DATA
C                                    SET UP NO OF TIMES AND DISC FILE CODE
      NTIMES=1
      NDSK=10+NBAND-1
      IF(NBAND.NE.5)  GO TO 250
      NTIMES=4
      NDSK=10
C                                    GO THRU ONE TIME FOR EACH BAND
250   DO 300 ITIME=1,NTIMES
      NDSK=NDSK+1
C                                    OUTPUT THE PICTURE FOR THIS BAND
      IF(PIC)  CALL RODSK1(IMAGE,QUAN,NHOR,NVERT,NDSK)
C                                    WRITE OUT THE GREY-TONES
      CALL RODSK2(IMAGE,IRSTRT,IRSTOP,NHOR,NDSK,PRNT,TAPE,IFIL)
300   CONTINUE
C                                    GO BACK FOR ANOTHER RUN
301   IF(TAPE)  ENDFILE IFIL
      WRITE(6,302)
302   FORMAT(////' END OF THIS RUN'/1H1)
      GO TO 1001
      END
```

```
         LBL      ERTS,ERTS TAPE READING PROGRAM
         TTL      ERTS READING PROGRAM-EREAD,ESKIP,EINIT
         TTLS     FOR CENTER FOR RESEARCH, INC.
                  DATE 9/11/72
*
*  CALLING SEQUENCES ARE---
*
*  TO INITIALIZE--
*
*        CALL     EINIT(NOLS)
*
*        EINIT INITIALIZES THE ERTS TAPE SO THAT DATA MAY
*        BE READ.  IT RETURNS THE NUMBER OF WORDS PER
*        SCAN LINE IN THE VARIABLE NOLS.
*
*  TO SKIP A NUMBER OF RECORDS
*
*        CALL     ESKIP(NOSK)
*
*        NOSK IS THE NUMBER OF RECORDS TO SKIP.
*        IF THE END OF FILE IS ENCOUNTERED BEFORE NOSK RECORDS
*        ARE SKIPPED, EREAD IS ABORTED.
*
*  TO READ A LINE
*
*        CALL     EREAD(I,LN)
*
*        I        THIS IS THE ARRAY INTO WHICH THE NOLS WORDS OF DATA
*        FROM A LINE OF ERTS DATA IS PLACED.  THE DATA IS PLACED INTO
*        THIS ARRAY IN STANDARD CORRESPONDING POINT FORM.
*        THE ERTS DATA HAS FOUR CHANNELS, SO THERE ARE
*        ACTUALLY NOLS/4 DATA POINTS PER SCAN LINE.
*        LN       THIS IS RETURNED BY EREAD.  IF LN IS RETURNED AS ZERO, THE
*        END OF FILE WAS REACHED ON THE ERTS TAPE.  IF IT IS
*        RETURNED NON-ZERO, THEN IT IS THE LINE NUMBER
*        OF THE LINE OF DATA RETURNED.
*
*        NOTE--  THE ERTS TAPE MUST HAVE FILE CODE 'ES' FOR
*        THIS PROGRAM...
*
*        ABORT CODES POSSIBLE FROM THIS SUBROUTINE ARE--
*        MB       EREAD BUFFER 'DATA' IS NOT LARGE ENOUGH
*                 FOR A BLOCK OF ERTS DATA. IT MUST BE INCREASED
*                 IN SIZE.
*        AI       EINIT WAS CALLED TWICE.
*        NI       EINIT WAS NOT CALLED BEFORE CALLING EREAD OR ESKIP
*        UE       END OF FILE ENCOUNTERED WHILE READING
*                 ID OR ANNOTATION BLOCKS ON ERTS TAPE
*        EF       END OF FILE ENCOUNTERED WHILE TRYING TO SKIP
*                 RECORDS ON ERTS TAPE IN ESKIP.
```

STORAGE

```
.EINIT  DEC     0                             THIS IS NONZERO IF EINIT HAS BEEN CALLED ERTS0053
RECNO.  DEC     0                             THIS IS THE CURRENT LINE NO.             ERTS0054
FDW     VFD     18/FCB,1/0,1/1,2/0,1/0,1/0,1/1                                         ERTS0055
        FILCB   FCB,ES,,,1024,,,1,,1                                                   ERTS0056
CNT1    IOTD    DATA,9                        DCW FOR READING ID                       ERTS0057
CNT2    IOTD    DATA,139                      DCW FOR READING ANNOTATION BLOCK         ERTS0058
CONT    IOTD    DATA,**                       DCW FOR READING DATA BLOCK               ERTS0059
EOFA    LDQ     =3HOUE,DL                                                              ERTS0060
        MME     GEBORT                                                                 ERTS0061
DATA    BSS     1024                          DATA BUFFER AREA                         ERTS0062
OT      BSS     1                             OUTPUT TALLY                             ERTS0063
IT      BSS     1                             INPUT TALLY                              ERTS0064

STS     TALLY   STK,4                         STACK TALLY                              ERTS0065
ST      BSS     1                             STACK TALLY                              ERTS0066
STK     BSS     4                             STACK AREA                               ERTS0067
```

EINIT

```
EINIT  SAVE                                                        ERTS0069

       SZN    .EINIT         HAVE WE BEEN CALLED BEFORE            ERTS0070
       TNZ    EABTA          YES                                   ERTS0071
       STC1   .EINIT         MARK THAT WE'VE BEEN HERE ONCE        ERTS0072
       EAX0   2,1*                                                 ERTS0073
       STX0   EIP            SAVE ADDRESS                          ERTS0074
       CALL   OPEN(FDW,1)  OPEN FILE                               ERTS0075

       CALL   READ(FCB,CNT1)  READ ID RECORD                       ERTS0076

       CALL   WAIT(FCB,EOFA)  AND WAIT FOR IT TO GET DONE          ERTS0077

       LDQ    DATA+3         GET RECORD LENGTH                     ERTS0078
       ANQ    =0177777,DL    ISOLATE IT                            ERTS0079
EIP    STQ    **             GIVE IT TO CALLER                     ERTS0080
       QLS    6+18                                                 ERTS0081
       STCQ   ADDCNT,70      SAVE TALLY COUNT                      ERTS0082
       QRL    6+18-1         MULTIPLY BY 2                         ERTS0083
       DIV    9,DL           AND DIVIDE BY 9                       ERTS0084
       ARL    0              IS REMAINDER ZERO                     ERTS0085
       TZE    *+2            YES                                   ERTS0086
       ADQ    1,DL           NO, SO INCREMENT QUOTIENT             ERTS0087
       CMPQ   1025,DL        IS IT TOO BIG                         ERTS0088
       TNC    *+3            NO                                    ERTS0089

       LDQ    =3HOMB,DL                                            ERTS0090
       MME    GEBORT         YES, SO ABORT                         ERTS0091
       STCQ   CONT,07        SAVE IT IN DCW                        ERTS0092
       MME    GESNAP                                               ERTS0093
       ZERO   DATA,9         SNAP OUT ID BLOCK                     ERTS0094
       CALL   READ(FCB,CNT2) READ ANNOTATION BLOCK                 ERTS0095

       CALL   WAIT(FCB,EOFA) WAIT ON IT                            ERTS0096
```

EINIT

```
         MME      GESNAP                                       ERTS009
         ZERO     DATA,139          SNAP OUT ANNOATION BLOCK   ERTS009
         RETURN   EINIT             AND RETURN                 ERTS009
ESKIP    SAVE                                                  ERTS010

         SZN      .EINIT            ARE WE INITIALIZED         ERTS010
         TZE      EABTB             NO                         ERTS010
         LDQ      2,1*              GET NUMBER RECORDS TO SKIP ERTS010
ECOMP    TZE      ESKR              NONE TO DO, SO RETURN      ERTS010
         CMPQ     64,DL             IS IT > 63                 ERTS010
         TRC      ESK64             YES                        ERTS010
         ASQ      RECNO.            INCREMENT RECORD COUNT     ERTS010
         EAX1     0,QL                                         ERTS010
         STX1     *+5                                          ERTS010
         CALL     FSREC(FCB,.ESKIP,EOFS)                       ERTS011
```



```
ESKR     RETURN   ESKIP                                        ERTS011
.ESKIP   BSS      1                 NO. OF RECORDS TO SKIP     ERTS011

EOFS     LDQ      =3H0EF,DL         UNEXPECTED EOF             ERTS011
         MME      GEBORT                                       ERTS011
* TRYING TO SKIP 64 OR MORE RECORDS                            ERTS011
ESK64    LDA      63,DL                                        ERTS011
         ASA      RECNO.            INCREMENT RECORD COUNT     ERTS011
         SBQ      63,DL                                        ERTS011
         STQ      .ESKIP            SAVE FOR LATER             0ERTS012
         CALL     FSREC(FCB,63,EOFS)

         LDQ      .ESKIP                                       ERTS012
         TRA      ECOMP                                        ERTS012
*                                                              ERTS012
EABTA    LDQ      =3H0AI,DL         TRIED TO CALL EINIT TWICE  ERTS012
         MME      GEBORT                                       ERTS012
EABTB    LDQ      =3H0NI,DL         DIDN'T CALL EINIT          ERTS012
```

```
     EINIT
  127         MME      GEBORT                                        ERTS0127
```

EREAD

```
EREAD   SAVE                                                   ERTS0129

        SZN     .EINIT          ARE WE INITIALIZED             ERTS0130
        TZE     EABT3           NO                             ERTS0131
        EAQ     2,1*            GET ARRAY ADDR.                ERTS0132
ADDCNT  ORQ     **,DL           ADD TALLY COUNT                ERTS0133
        STQ     OT              AND SAVE IT                    ERTS0134
        EAQ     DATA                                           ERTS0135
        STQ     IT              SET UP INPUT TALLY             ERTS0136
        EAX0    3,1*            INSERT ADDRESS OF COUNT WD     ERTS0137
        STX0    ERP                                            ERTS0138
        CALL    READ(FCB,CONT)  READ NEXT LINE OF DATA         ERTS0139

        CALL    WAIT(FCB,EOF)   WAIT ON IT                     ERTS0140

        AOS     RECNO.          INCREMENT LINE NO.             ERTS0141
        LDQ     RECNO.          GET LINE NO.                   ERTS0142
ERP     STQ     **              RETURN TO CALLER               ERTS0143
        STZ     LINIT           INITIALIZE EXPANDER            ERTS0144
NEXT    LDQ     STS                                            ERTS0145
        STQ     ST              INITIZE STACK TALLY            ERTS0146
NXTONE  TSX1    L               GET POINT                      ERTS0147
        STA     OT,ID           AND PUT IT IN ARRAY            ERTS0148
        TTF     *+2                                            ERTS0149
        DRL                     ERROR                          ERTS0150
        TSX1    L               GET NEXT POINT                 ERTS0151
        STA     ST,ID           PUT INTO STACK TALLY           ERTS0152
        TTF     NXTONE          GO PROCESS NEXT                ERTS0153
        LDQ     STS             TALLY RUNOUT                   ERTS0154
        STQ     ST              SO REINITIALIZE TALLY          ERTS0155
AGAIN   LDQ     ST,ID           PICK UP POINTS                 ERTS0156
        TTF     MORE            GOT ONE                        ERTS0157
        STQ     OT,ID           TALLY RUNOUT                   ERTS0158
        TTF     NEXT            GO PROCESS NEXT EIGHT POINTS.  ERTS0159
        RETURN  EREAD           TALLY RUNOUT, SO WE'RE DONE.   ERTS0160
MORE    STQ     OT,ID           SAVE IT                        ERTS0161
        TTF     AGAIN           GO GET NEXT ONE STACKED        ERTS0162
        DRL                     ERROR                          ERTS0163
```

POINT FETCHING ROUTINE L

```
LSAVE   BSS     1                QR SAVED FROM LAST TSX1 L          ERTS0165
LINIT   BSS     1                SHIFT INDICATOR FOR L              ERTS0166
*                                                                   ERTS0167
L       SZN     LINIT            ARE WE READY FOR NEXT PAIR         ERTS0168
        TNZ     LCONT            YES                                ERTS0169
        LDQ     IT,ID            NO, SO GET NEXT WORD               ERTS0170
        AOS     LINIT            INCREMENT LINIT                    ERTS0171
        TRA     LRET             DONE FOR THIS ONE                  ERTS0172
LCONT   AOS     LINIT            INCREMENT AGAIN.                   ERTS0173
        LDQ     LINIT                                               ERTS0174
        CMPQ    5,DL             IS THIS 5TH TIME                   ERTS0175
        TZE     L5               YES                                ERTS0176
        CMPQ    9,DL             OR 9TH TIME                        ERTS0177
        TNZ     *+2              NO                                 ERTS0178
        STZ     LINIT            REINITIALIZE                       ERTS0179
        LDQ     LSAVE            GET SAVED QR                       ERTS0180
LRET    LDA     0,DL                                                ERTS0181
        LLS     8                SHIFT OVER                         ERTS0182
        STQ     LSAVE            SAVE QR FOR NEXT TIME              ERTS0183
        TRA     0,1              AND RETURN                         ERTS0184
L5      LDA     0,DL                                                ERTS0185
        LDQ     LSAVE            RESTORE QR                         ERTS0186
        LLS     4                THIS ONE IS DIVIDED OVER           ERTS0187
        LDQ     IT,ID            WORD BOUNDARIES                    ERTS0188
        LLS     4                                                   ERTS0189
        STQ     LSAVE                                               ERTS0190
        TRA     0,1              RETURN                             ERTS0191
*                                                                   ERTS0192
*  EOF PROCESSOR FOR EREAD                                          ERTS0193
EOF     STZ     ERP,I            MARK EOF TO CALLER                 ERTS0194
        RETURN  EREAD            AND RETURN                         ERTS0195

        END                                                         ERTS0196
TION.
PA 091572/091472       JMPB 053172/070872       JMPC 072772/072772
IN THE ABOVE ASSEMBLY
```

```
CPIXEY          SPECIAL PICTURE ROUTINE
C
C     WRITTEN BY RJ BOSLEY                                  JAN 1974
C
C     DESCRIPTION OF PROGRAM.
C        THIS SUBPROGRAM OUTPUTS A PICTURE OF THE ERTS DATA FROM USER
C     SPECIFIED PARAMETERS TO PITCHR.  INPUT PARAMETERS FOR PITCHR PLUS
C     THE PARAMETERS LNSKIP AND KOLSKP ALL MUST BE SUPPLIED TO THIS
C     SUBPROGRAM UNDER THE NAMELIST $PICTUR FORMAT.  FOR A COMPLETE
C     DESCRIPTION OF INPUT ARGUMENTS TO PITCHR, SEE THE PITCHR WRITEUP.
C     BY THE USER SETTING IMAX=0 AND IMIN= MAX GREY TONE LEVEL, IT IS
C     POSSIBLE TO OBTAIN A POSITIVE IMAGE INSTEAD OF A NEGATIVE.
C        NOTE--DISC FILES MUST BE ON FILE CODES 13 AND 14.
C
C     INTERNAL PARAMETERS.
C             IBAND               BAND NUMBER BEING PROCESSED
C             IS,IE               STARTING,ENDING BAND NUMBERS
C             LINES               NUMBER OF LINES IN IMAGE PRINTED
C             NOSK                NUMBER OF LINES TO SKIP IN ERTS TAPE
C             IPSTR,IPEND         STARTING,ENDING POINTS IN ERTS LINE
C             LNSKIP,KOLSKP       LINE AND COLUMN INCREMENT
C             ISKIP               COLUMN INCREMENT FOR ERTS LINE
C             KP                  NUMBER OF COLUMNS IN IMAGE
C
C     INPUT ARGUMENTS TO PITCHR.
C             IMAGE               ARRAY TO BE PRINTED
C             ICELL,JCELL         NUMBER OF ROWS,COLS IN ARRAY
C             INIT                NUMBER OF TIMES ENTRY IS MADE AT SNAP
C                                   MUST BE GREATER THAN 1
C             IMIN,IMAX           MINIMUM,MAXIMUM GREY TONES IN ARRAY
C             NROW                NUMBER OF ROWS TO BE PRINTED=ICELL
C             NFILES              NUMBER OF OUTPUT FILES AVAILABLE,
C                                   SET=0 FOR ALL OUTPUT ON FILE CODE 6
C                                   SET=2 FOR FILES 06 AND 42
C                                   IF .GT.2, USER MUST SUPPLY FILES
C             IFIL                ARRAY CONTAINING OUTPUT FILE CODES
C             NULW,NULD           NUMBER OF COLS,ROWS PER OUTPUT PAGE
C             AMAG,DMAG           WIDTH,LENGTH DOWN MAGNIFICATION
C
C     ENTRY POINT.
C        CALL PIXEY(ILINE,IMAGE,IRSTRT,IRSTOP,ICSTRT,ICSTOP,QUAN,NBAND)
C
C     INPUT ARGUMENTS.
C             ILINE               ARRAY ERTS LINE IS READ INTO
C             IMAGE               IMAGE ARRAY FOR PITCHR,HOLDS ONE LINE
C             IRSTRT,IRSTOP       STARTING,STOPING ROW IN IMAGE
C             ICSTRT,ICSTOP       STARTING,STOPING COL IN IMAGE
C             QUAN                TRUE FOR EQUAL PROBABILITY QUANTIZING
C                                   OF IMAGE
C             NBAND               BAND NUMBER TO BE PROCESSED
```

```
C                                 SET=5 FOR ALL BANDS
C
C     EXAMPLE OF SAMPLE RUN.
C        CONTROL CARDS--
C          $PARAM SPIC=T,QUAN=T,IRSTRT=1,IRSTOP=1216,ICSTRT=5,
C           ICSTOP=772,NBAND=2$END
C         $PICTUR INIT=304,NFILES=2,LNSKIP=4,KOLSKP=3,JCELL=256$END
C       THIS RUN WILL PRINT OUT ON FILES 06 AND 42 A PICTURE 256 PTS
C     WIDE BY 304 LINES.  NOTE THAT 1216/4=304 AND 768/3=256, GIVING THE
C     VALUES FOR LNSKIP=4,KOLSKP=3,JCELL=256,INIT=304.  ALSO, USING
C     THESE VALUES FOR LNSKIP AND KOLSKP, THE PICTURE WILL BE IN
C     PROPORTION TO THE ERTS PRINT.
C
C
      SUBROUTINE PIXEY(ILINE,IMAGE,IRSTRT,IRSTOP,ICSTRT,ICSTOP,QUAN,
     1                        NBAND)
C
      DIMENSION ILINE(1),IFIL(10),IMAGE(1)
      LOGICAL QUAN
      NAMELIST /PICTUR/NFILES,IFIL,ICELL,JCELL,INIT,IMIN,IMAX,NROW,
     1                 NULW,NULD,AMAG,DMAG,LNSKIP,KOLSKP
C                                 INITIALIZE PARAMETERS
      ICELL=1
      JCELL=256
      INIT=304
      IMIN=0
      IMAX=75
      IF(QUAN) IMAX=12
      NFILES=2
      IFIL(1)=6
      IFIL(2)=42
      NROW=ICELL
      NULW=129
      NULD=60
      AMAG=1
      DMAG=1
      LNSKIP=4
      KOLSKP=3
C                                 READ PARAMETERS
      READ(5,PICTUR)
C                                 WRITE OUT PARAMETERS
      WRITE(6,1)
      WRITE(6,PICTUR)
    1 FORMAT(////20X,'SPECIAL PICTURE ROUTINE'///)
    2 FORMAT(1H1,'BAND NUMBER ',I1)
C                                 SET UP FOR THE DESIRED BAND
      IS=1
      IE=4
      ISKIP=4*KOLSKP
      IF(NBAND.EQ.5)  GO TO 5
```

```
      IS=NBAND
      IE=IS
C                                   GO THRU EACH BAND
5     DO 100 IBAND=IS,IE
      REWIND 13
      REWIND 14
C                                   WRITE SUBTITLE
      WRITE(6,2) IBAND
      IPSTR=(ICSTRT-1)*4+IBAND
      IPEND=ICSTOP*4
C                                   INITIALIZE SNAP IN PITCHR
      CALL PITCHR(IMAGE,ICELL,JCELL,INIT,1,IMAX,IMIN,NROW,NFILES,IFIL,
     1                  NULW,NULD,AMAG,DMAG,)
C                                   POSITION THE INPUT TAPE
      NOSK=IRSTRT-1
      CALL EREWND
      IF(NOSK.NE.0)  CALL ESKIP(NOSK)
C                                   GO THRU EACH LINE
      DO 50 LINE=IRSTRT,IRSTOP,LNSKIP
C                                   READ LNSKIP LINES
      DO 10 LIN=1,LNSKIP
      CALL EREAD(ILINE,LN)
      IF(LN.NE.0)  GO TO 10
      L=LINE+LIN
      WRITE(6,9)  L
9     FORMAT(///' ***EXECUTION TERMINATED--EOF DETECTED ON ERTS TAPE,
     1LINE NUMBER IS ',I6/1H1)
      RETURN
10    CONTINUE
C                                   GET THIS LINE FOR THE IMAGE
      KP=0
      DO 20 IP=IPSTR,IPEND,ISKIP
      KP=KP+1
      IMAGE(KP)=ILINE(IP)
20    CONTINUE
C                                   WRITE THIS LINE TO SCRATCH FILE 14
      WRITE(14) (IMAGE(IP),IP=1,JCELL)
50    CONTINUE
      ENDFILE 14
      REWIND 14
C                                   QUANTIZE FILE 14 AND PUT OUT ON 13
      IF(QUAN)  CALL ZEQUAN(IMAGE,INIT,JCELL,1,1,1,13,14,13)
      REWIND 13
C                                   READ THE QUANTIZED IMAGE
      DO 90 I=1,INIT
      READ(13) (IMAGE(K),K=1,JCELL)
C                                   SNAP OUT THIS LINE
90    CALL SNAP
100   CONTINUE
      RETURN
      END
```

```
CZEQUAN                           Z E Q U A N
C
C     MODIFIED FROM KEQUAN BY Z DINSTEIN                      SEPT 1973
C
C     DESCRIPTION OF PROGRAM.
C        THIS SUBROUTINE QUANTIZES AN IMAGE ON FILE 'INFILE' BY EQUAL
C     PROBABILITY TO NQ LEVELS AND OUTPUTS IT TO FILE 'IOUTFL'.  THIS
C     PROCESSING IS DONE LINE BY LINE AFTER A FIRST PASS IS MADE THRU
C     THE IMAGE TO DETERMINE THE MINIMUM AND MAXIMUM GREY TONES.
C
C     ENTRY POINT.
C        CALL ZEQUAN(LINE,NUMLIN,NUMPPL,NCOMP,ICOMP,LEFT,NQ,INFILE,
C                    IOUTFL)
C
C     INPUT ARGUMENTS.
C             LINE   = ARRAY ONE LINE OF IMAGE IS READ INTO
C             NUMLIN= NUMBER OF LINES IN IMAGE
C             NUMPPL= NUMBER OF COLUMNS IN IMAGE
C             NCOMP = NUMBER OF COMPONENTS IN ORIGINAL IMAGE
C             ICOMP = THE COMPONENT TO BE QUANTIZED
C             LEFT  = LEFT-MOST CELL IN LINE DESIRED
C             NQ    = NUMBER OF QUANTIZED LEVELS
C             INFILE= INPUT FILE CONTAINING IMAGE TO BE QUANTIZED
C
C     OUTPUT ARGUMENTS.
C             IOUTFL= OUTPUT FILE CONTAINING QUANTIZED IMAGE
C
C
      SUBROUTINE ZEQUAN(LINE,NUMLIN,NUMPPL,NCOMP,ICOMP,LEFT,NQ,INFILE,
     1          IOUTFL)
C
      DIMENSION LINE(NCOMP,1),KN(512)
C                                    INITIALIZE HISTOGRAM TO ZERO
      DO 1 I=1,512
      KN(I)=0
    1 CONTINUE
      MIN=10000
      MAX=-10000
      REWIND INFILE
C                                    GO THRU EACH LINE
      DO 2 II=1,NUMLIN
      READ(INFILE) ((LINE(J,L),J=1,NCOMP),L=1,NUMPPL)
C                                    SET STARTING,ENDING PTS IN LINE
      IRIGHT=LEFT+NUMPPL-1
      DO 2 I=LEFT,IRIGHT
      J=LINE(ICOMP,I)
C                                    GET MIN,MAX AND HISTOGRAM
      IF(MAX.LT.J)  MAX=J
      IF(MIN.GT.J) MIN=J
      KN(J+1)=KN(J+1)+1
```

```
2       CONTINUE
        NGL=MAX
        WRITE(6,100) MIN,MAX
100     FORMAT(1X,'MIN, MAX ARE ',2I4)
C                                      GET NO OF PTS IN IMAGE
        NP=NUMLIN*NUMPPL
        J=1
        MQ=NQ
C                                      DETERMINE LEVELS
        DO 3 I=1,NQ
        NL=NP
C                                      GET NEW LEVEL
4       NL=NL-MQ*KN(J)
        NP=NP-KN(J)
        KN(J)=I-1
        J=J+1
C                                      IF LAST LEVEL,SKIP
         IF(J.GT.NGL) GO TO 6
C                                      INCREMENT THE LEVEL AGAIN
        IF(MQ*KN(J).LE.NL*2)GO TO 4
C                                      DECREASE THE NO. OF LEVELS LEFT
3       MQ=MQ-1
         DO 5 I=J,NGL
5       KN(I)=NQ-1
        GO TO 8
C                                      RESET THE LAST LEVEL
6       N=(NQ-I)/2
        IF(N.LT.1)GO TO 8
        DO 7 I=1,NGL
7       KN(I)=KN(I)+N
8       CONTINUE
C                                      REWIND DISC FILES
        REWIND INFILE
        REWIND IOUTFL
C                                      ASSIGN QUANTIZED LEVELS LINE BY LINE
        DO 11 II=1,NUMLIN
        READ(INFILE) ((LINE(J,L),J=1,NCOMP),L=1,NUMPPL)
        DO 9 I=LEFT,IRIGHT
        J=LINE(ICOMP,I)
9       LINE(ICOMP,I)=KN(J+1)
C                                      WRITE OUT QUANTIZED FILE
        WRITE(IOUTFL) (LINE(ICOMP,K),K=LEFT,IRIGHT)
11      CONTINUE
        ENDFILE IOUTFL
        REWIND IOUTFL
        RETURN
        END
```

```
R7544 01   03-05-71   10.192        PTCHRG      GMPEX EXECUTION REPORT
                                                                       PTCHRG
$        GMAP      NDECK,COMDK                                          PTCHRG01
                   TTL       P I T C H R     G M A P                     PTCHRG02
                   LBL       PTCHRG,       P I T C H R   G M A P         PTCHRG03
                   REM       ENTRY POINTS ARGSET & MARG BY G ELLIOTT,    PTCHRG04
                   REM                                                   PTCHRG05
                   SYMDEF    ARGSET          TO SET BIT 35 & INITIALIZE  PTCHRG06
         ARGSET    SAVE                      FOR MARG.                   PTCHRG07
                   LDQ       1,DL            SET BIT 35                  PTCHRG08
                   MME       GESETS                                      PTCHRG09
                   LDX0      1,1             TRACE BACK TO GET           PTCHRG10
                   LDX1      0,0             LOCATION OF LAST CALL       PTCHRG11
                   EAX0      1,0             TO PITCHR                   PTCHRG12
                   STX0      LOCAT           PUT IT WHERE MARG CAN USE IT. PTCHRG13
                   LDX0      -1,0            GET NUMBER OF ARGUMENTS PROVIDED, PTCHRG14
                   SBLX0     LOCAT                                       PTCHRG15
                   STX0      MAXNUM          ALSO FOR MARG.              PTCHRG16
                   RETURN    ARGSET                                      PTCHRG17
                   SYMDEF    MARG            ENTRY POINT TO CHECK FOR THE PTCHRG18
         MARG      SAVE                      EXISTENCE OF NON-NULL ARGUMENTS, PTCHRG19
                   LXL0      2,1*            WHICH ARGUMENT AREWE CHECKING ? PTCHRG20
                   EAQ       0                                           PTCHRG21
         MAXNUM    CMPX0     **,DU           DO WE HAVE THAT MANY ?      PTCHRG22
                   TMI       *+3                                         PTCHRG23
                   SZN       0,DU            NO,                         PTCHRG24
                   TRA       .E.L..,I        GO HOME.                    PTCHRG25
         LOCAT     EAX0      **,0                                        PTCHRG26
                   LDX0      0,0             IS IT A NULL POINTER ?      PTCHRG27
                   TNZ       *+2                                         PTCHRG28
                   TRA       .E.L..,I        YES                         PTCHRG29
                   LDQ       1,DL            NO                          PTCHRG30
                   TRA       .E.L..,I                                    PTCHRG31
         *         SYMDEF    ABORT                                       PTCHRG32
         E         BSS       1                                           PTCHRG33
         ABORT     NULL                                                  PTCHRG34
                   LDQ       =3HOPT,DL                                   PTCHRG35
                   MME       GEBORT                                      PTCHRG36
                   SYMDEF    BORED                                       PTCHRG37
         BORED     EAX0      3,1*            GET ADDRESS OF ARRAY        PTCHRG38
                   LDQ       4,1*            NO. ELEMENTS IN ARRAY       PTCHRG39
                   QLS       6               USE FOR TALLY COUNT         PTCHRG40
                   STQ       TALI                                        PTCHRG41
                   STX0      TALI                                        PTCHRG42
                   EAX0      =6H-            CHARACTER THAT GOES INTO ARRAY PTCHRG43
                   STX0      STR-1                                       PTCHRG44
                   EAX0      TW              XED WILL USE TALLY MODIFICATION PTCHRG45
                   STX0      STR                                         PTCHRG46
                   TRA       BOR             START FILLING ARRAY         PTCHRG47
         TT        ADS       2,1*            BUMP BORDER POSITION        PTCHRG48
                   TTF       BOR                                         PTCHRG49
                   TRA       0,1*                                        PTCHRG50
         TALI      BSS       1                                           PTCHRG51
                   SYMDEF    BORDER
```

```
                                                                          PTCHRG52
BORDER  EAX0    DS          WE STORE ON EITHER SIDE OF IMAGE DATA         PTCHRG53
        STX0    STR         CHARACTER FOR VERTICAL BORDER                 PTCHRG54
        EAX0    =6HI                                                      PTCHRG55
        STX0    STR-1                                                     PTCHRG56
BOR     LDQ     2,1*        CURRENT LINE COUNT                            PTCHRG57
        DIV     10,DL       IS IT DIVISIBLE BY TEN                        PTCHRG58
        CMPA    0,DL                                                      PTCHRG59
        TNZ     NTEN                                                      PTCHRG60
        DIV     10,DL                                                     PTCHRG61
        LDQ     BLNK        PUT BLANKS IN QR                              PTCHRG62
        LLS     30          MAKE NO. LEFT JUSTIFIED, BLANK FILLED         PTCHRG63
        TRA     STR                                                       PTCHRG64
NTEN    CMPA    5,DL        WAS THE NUMBER DIVISIBLE BY FIVE              PTCHRG65
        TZE     FIVE                                                      PTCHRG66
        LDA     **                                                        PTCHRG67
STR     XED     **                                                        PTCHRG68
        TRA     0,1         RETURN IF WE ENTERED AT BORDER                PTCHRG69
FIVE    LDA     =6H*                                                      PTCHRG70
        TRA     STR                                                       PTCHRG71
TH      ESTA    TALI,ID     PUT IN HOR BORDER ARRAY                       PTCHRG72
        TRA     TT                                                        PTCHRG73
DS      STA     3,1*                                                      PTCHRG74
        STA     4,1*                                                      PTCHRG75
        SYMDEF  LSHIFT                                                    PTCHRG76
LSHIFT  EAX0    2,1*        ADDRESS OF ARRAY                              PTCHRG77
        LDQ     3,1*        NO. ELEMENTS IN ARRAY                         PTCHRG78
        QLS     6           USE FOR TALLY COUNT                           PTCHRG79
        STQ     TALI                                                      PTCHRG80
        STX0    TALI                                                      PTCHRG81
LD      LDQ     TALI,I                                                    PTCHRG82
        QLS     6           SHIFT UP NEXT CHARACTER                       PTCHRG83
        STQ     TALI,ID                                                   PTCHRG84
        TTF     LD                                                        PTCHRG85
        TRA     0,1*                                                      PTCHRG86
        SYMDEF  IFETCH                                                    PTCHRG87
S1      BSS     1                                                         PTCHRG88
S2      BSS     1                                                         PTCHRG89
S3      BSS     1                                                         PTCHRG90
IFETCH  SAVE    2,7                                                       PTCHRG91
        LXL7    4,1*        =1 FOR INTEGER, 0 FOR FLOATING PT             PTCHRG92
        EAX2    2,1*                                                      PTCHRG93
        TSX0    GETT                                                      PTCHRG94
SS1     FSTR    S1                                                        PTCHRG95
        LXL0    7,1*                                                      PTCHRG96
        TNZ     *+3                                                       PTCHRG97
        ADX2    1,DU                                                      PTCHRG98
        TSX0    GETT                                                      PTCHRG99
        FSTR    S2                                                        PTCHRG00
        LXL0    6,1*                                                      PTCHRG01
        TZE     *+3                                                       PTCHRG02
        FLD     S1                                                        PTCHRG03
        TRA     FS
```

```
                                        PTCHRG04
         LXL2    3,1*                   PTCHRG05
         ADX2    2,1                    PTCHRG06
         TSX0    GETT                   PTCHRG07
         FSB     S1                     PTCHRG08
         FMP     6,1*                   PTCHRG09
         FAD     S1                     PTCHRG10
FS       FSTR    S3                     PTCHRG11
         FLD     S2                     PTCHRG12
         FSB     S1                     PTCHRG13
         FMP     5,1*                   PTCHRG14
         FAD     S1                     PTCHRG15
         FAD     S3                     PTCHRG16
         FDV     =2.                    PTCHRG17
FM       FSB     FMIN                   PTCHRG18
IE       EQU     FM-SS1                 PTCHRG19
         FDV     FDIF                   PTCHRG20
         FMP     E                      PTCHRG21
         FAD     =0.,DU                 PTCHRG22
         TPL     *+2                    PTCHRG23
         UFA     =0777777777            PTCHRG24
         UFA     71*1024,DU             PTCHRG25
         ADQ     1,DL                   PTCHRG26
         CMPQ    2,DL                   PTCHRG27
         TPL     *+3                    PTCHRG28
         LDQ     1,DL                   PTCHRG29
         RETURN  IFETCH                 PTCHRG30
ID       CMPQ    **,DL                  PTCHRG31
         TMI     *+2                    PTCHRG32
IDD      LDQ     **,DL                  PTCHRG33
         RETURN  IFETCH                 PTCHRG34
GETT     CMPX7   1,DU                   PTCHRG35
         TZE     3,IC                   PTCHRG36
         FLD     0,2                    PTCHRG37
01       TRA     **,0                   PTCHRG38
         LDA     0,2                    PTCHRG39
         EAQ     0                      PTCHRG40
         LDE     35*1024,DU             PTCHRG41
         FAD     =0.,DU                 PTCHRG42
02       TRA     **,0                   PTCHRG43
         SYMDEF  FI                     PTCHRG44
FI       FLD     2,1*                   PTCHRG45
         FSB     3,1*                   PTCHRG46
         FST     FDIF                   PTCHRG47
         FLD     3,1*                   PTCHRG48
         FST     FMIN                   PTCHRG49
         LXL0    4,1*                   PTCHRG50
         STX0    IDD                    PTCHRG51
         ADX0    1,DU                   PTCHRG52
         STX0    ID                     PTCHRG53
         EAA     0                      PTCHRG54
         EAQ     -1,0                   PTCHRG55
         LLS     18
```

```
              LDE     35*1024,DU                         PTCHRG56
              FAD     =0,,DU                             PTCHRG57
              FSTR    E                                  PTCHRG58
              LDQ     0,2                                PTCHRG59
              LLS     36                                 PTCHRG60
              LXL0    5,1*                               PTCHRG61
              TNZ     ZZ                                 PTCHRG62
              EAX0    IE                                 PTCHRG63
     SX       STX0    01                                 PTCHRG64
              STX0    02                                 PTCHRG65
              TRA     0,1*                               PTCHRG66
     ZZ       EAX0    0                                  PTCHRG67
              TRA     SX                                 PTCHRG68
     FDIF     BSS     1                                  PTCHRG69
     FMIN     BSS     1                                  PTCHRG70
     BLNK     BCI     1,                                 PTCHRG71
              END                                        PTCHRG72
```

```
CWRTDSK             WRITE ERTS DATA ONTO DISC                                WRTDSK01
C                                                                            WRTDSK02
C     VERSION II WRITTEN BY RJ BOSLEY                         JAN 1974       WRTDSK03
C                                                                            WRTDSK04
C                                                                            WRTDSK05
C     DESCRIPTION OF PROGRAM.                                                WRTDSK06
C        THIS SUBROUTINE WRITES ONE LINE OF ERTS DATA FROM ARRAY ILINE       WRTDSK07
C     ONTO DISC FILES, ONE FILE FOR EACH BAND.  DATA IS WRITTEN IN NHOR     WRTDSK08
C     BLOCKS.                                                                WRTDSK09
C                                                                            WRTDSK10
C     INTERNAL PARAMETERS.                                                   WRTDSK11
C              NDSK              DISC FILE CODE, 11 THRU 14                  WRTDSK12
C              IP                POINT INDEX FOR ERTS LINE                   WRTDSK13
C              IPSTRT,IPSTOP     STARTING,STOPING POINTS IN LINE,            WRTDSK14
C                                  DEPENDENT UPON THE BAND                   WRTDSK15
C                                                                            WRTDSK16
C     ENTRY POINT.                                                           WRTDSK17
C              CALL WRTDSK(ILINE,NHOR,IPSTR,IPEND,NBAND)                     WRTDSK18
C                                                                            WRTDSK19
C     INPUT ARGUMENTS.                                                       WRTDSK20
C              ILINE             ARRAY CONTAINING ERTS LINE OF DATA          WRTDSK21
C              NHOR              NUMBER OF HORIZONTAL BLOCKS OF 41 COL       WRTDSK22
C              IPSTR,IPEND       START AND END POINTS IN ERTS LINE           WRTDSK23
C              NBAND             THE BAND DESIRED, SET=5 FOR ALL FOUR        WRTDSK24
C                                                                            WRTDSK25
C                                                                            WRTDSK26
      SUBROUTINE  WRTDSK(ILINE,NHOR,IPSTR,IPEND,NBAND)                       WRTDSK27
C                                                                            WRTDSK28
      DIMENSION ILINE(1)                                                     WRTDSK29
C                                IF ALL BANDS,SKIP DOWN                      WRTDSK30
      IF(NBAND.EQ.5)  GO TO 100                                              WRTDSK31
C                                WRITE ONE BAND ONLY                         WRTDSK32
      NDSK=10+NBAND                                                          WRTDSK33
C                                WRITE NHOR BLOCKS                           WRTDSK34
      DO 10 J=1,NHOR                                                         WRTDSK35
      IPSTRT=(J-1)*41*4+NBAND+IPSTR                                          WRTDSK36
      IPSTOP=IPSTRT+40*4                                                     WRTDSK37
      WRITE(NDSK)  (ILINE(IP),IP=IPSTRT,IPSTOP,4)                            WRTDSK38
10    CONTINUE                                                               WRTDSK39
      RETURN                                                                 WRTDSK40
100   CONTINUE                                                               WRTDSK41
C                                WRITE ALL FOUR BANDS                        WRTDSK42
      DO 101 I=1,4                                                           WRTDSK43
      NDSK=10+I                                                              WRTDSK44
C                                WRITE NHOR BLOCKS                           WRTDSK45
      DO 101 J=1,NHOR                                                        WRTDSK46
      IPSTRT=(J-1)*41*4+I+IPSTR                                              WRTDSK47
      IPSTOP=IPSTRT+40*4                                                     WRTDSK48
      WRITE(NDSK)  (ILINE(IP),IP=IPSTRT,IPSTOP,4)                            WRTDSK49
101   CONTINUE                                                               WRTDSK50
      RETURN                                                                 WRTDSK51
      END
```

```
CRDDSK1             READ DISC AND OUTPUT PICTURE
C
C      VERSION I. WRITTEN BY RJ BOSLEY                           JAN 1974
C
C      DESCRIPTION OF PROGRAM.
C         THIS SUBROUTINE READS THE ERTS DATA FROM THE DISC FILES AND
C      THEN PRINTS OUT A PICTURE OF THE DATA, PROCEEDING VERTICALLY AND
C      THEN FROM LEFT TO RIGHT.   SINCE THE BLOCKS ARE 41 COLS THEN TWO
C      WILL FIT ON ONE PAGE, ELIMINATING ONE HALF OF THE TOTAL LINES.
C
C      INTERNAL PARAMETERS.
C              NGL                  NUMBER OF GREY LEVELS IN IMAGE
C              NQ                   NUMBER OF QUANTIZING LEVELS
C              IASIZE               SIZE OF ARRAY TO BE QUANTIZED
C              MHOR                 HALF OF NHOR
C              NCOL                 NUMBER OF COLUMNS USED IN IMAGE
C              NSKIP                NUMBER OF RECORDS TO SKIP TO STAY IN
C                                     THE SAME STRIP
C              LAST                 TRUE INDICATES THE LAST STRIP
C
C      ENTRY POINT.
C              CALL RDDSK1(IMAGE,QUAN,NHOR,NVERT,NDSK)
C      INPUT ARGUMENTS.
C              IMAGE                ARRAY TO STORE TWO 41 X 41 BLOCKS
C              QUAN                 TRUE FOR EQUAL PROBABILITY QUANTI-
C                                     ZATION OF THE IMAGE
C              MAX                  MAXIMUM GREY TONE IN IMAGE
C              MIN                  MINIMUM GREY TONE IN IMAGE
C              NHOR                 NUMBER OF HORIZONTAL BLOCKS
C              NVERT                NUMBER OF VERTICAL BLOCKS
C              NDSK                 FILE CODE OF DISC TO BE PROCESSED
C
C
C
       SUBROUTINE RDDSK1(IMAGE,QUAN,NHOR,NVERT,NDSK)
C
       DIMENSION IMAGE(41,82)
       LOGICAL EOF,QUAN,LAST
       EOF=.FALSE.
       LAST=.FALSE.
       CALL FLGEOF(NDSK,EOF)
C                                   DETERMINE THE NUMBER OF LEVELS
       MAX=63
       MIN=0
       IF(QUAN)   MAX=11
C                                   SET UP FOR QUANTIZATION
       NGL=75
       NQ=12
       MHOR=NHOR/2
       NCOL=82
```

```
      IF(NHOR.NE.1)  GO TO 2
3     NCOL=41
      MHOR=1
2     IASIZE=41*NCOL
C                             SET THE NUMBER OF RECORDS TO SKIP
C                               TO REMAIN IN THE SAME STRIP
      NSKIP=NHOR-(NCOL/41)
C                             GO THRU HORIZONTALLY
      DO 100 J=1,MHOR
      REWIND NDSK
      K=J
      IF(LAST)  K=(NHOR/2)+1
C                             IF FIRST BLOCK, DO NOT SKIP OVER
      IF(K.EQ.1)  GO TO 5
      DO 1 I=2,K
      IF(NHOR.GT.1)  READ(NDSK)
1     READ(NDSK)
C                             IF ONLY ONE BLOCK VERTICALLY, DO NOT
C                             INITIALIZE SNAP IN PITCHR
5     IF(NVERT.NE.1)
     1        CALL PITCHR(IMAGE,41,NCOL,NVERT,,MAX,MIN,,,,,,1.21,,)
C                             READ IN TWO 41 BY 41 BLOCKS
      DO 50 II=1,NVERT
C                             READ 41 ROWS
      DO 20 JJ=1,41
C                             READ NCOL COLUMNS
      READ(NDSK)  (IMAGE(JJ,KOL),KOL=1,41)
      IF(NCOL.EQ.82)  READ(NDSK)  (IMAGE(JJ,KOL),KOL=42,82)
C                             SKIP OVER RECORDS NOT WANTED
      IF(NSKIP.EQ.0)  GO TO 20
      DO 16 N=1,NSKIP
      READ(NDSK)
      IF(EOF)  GO TO 20
16    CONTINUE
20    CONTINUE
      EOF=.FALSE.
C                             QUANTIZE THE IMAGE
      IF(QUAN)  CALL KEQUAN(IMAGE,NGL,NQ,IASIZE)
C                             SNAP OUT A 41 BY 82 BLOCK
      IF(NVERT.NE.1)  CALL SNAP
      IF(NVERT.EQ.1)  CALL PITCHR(IMAGE,41,NCOL,1,,MAX,MIN,,,,,,1.21,,)
50    CONTINUE
100   CONTINUE
C                             CHECK TO SEE IF NHOR IS EVEN
      IF(NHOR.EQ.1)  GO TO 101
      IF(NHOR.EQ.(MHOR*2))  GO TO 101
C                             IF NOT, DO THE LAST STRIP OF 41 COLS
      IF(LAST)  GO TO 101
      LAST=.TRUE.
      GO TO 3
```

```
C                                    WRITE THE BAND NUMBER                RDDSK10
101   IBAND=NDSK-10                                                        RDDSK10
      WRITE(6,102)  IBAND                                                  RDDSK10
102   FORMAT(6X,'PICTURE FROM BAND NUMBER',I2)                             RDDSK10
      RETURN                                                               RDDSK10
      END                                                                  RDDSK10
```

```
CKEQUAN                         K-E-Q-U-A-N
C
C     WRITTEN BY G. ELLIOT                                          SEPT 1971
C     VERSION II BY RJ BOSLEY FOR IASIZE                            JUNE 1973
C
C     DESCRIPTION OF PROGRAM.
C            THIS PROGRAM WILL QUANTIZE BY EQUAL PROBABILITY THE INPUT
C            ARRAY IA TO NQ LEVELS.
C
C     ENTRY POINT.
C            CALL KEQUAN(IA,NGL,NQ,IASIZE)
C
C     ARGUMENTS.
C            IA                    INPUT ARRAY TO BE CONVERTED TO
C                                     QUANTIZED ARRAY
C            NGL                   NUMBER OF GREY TONE LEVELS IN IA
C            NQ                    NUMBER OF QUANTIZING LEVELS
C            IASIZE                SIZE OF ARRAY IA
C
      SUBROUTINE KEQUAN(IA,NGL,NQ,IASIZE)
C
      DIMENSION IA(1),KN(512)
       IF(NGL.GT.512) WRITE(6,10)
 10   FORMAT(5X,'***********NUMBER OF GREY LEVELS TOO LARGE***********')
       DO 1 I=1,NGL
 1    KN(I)=0
C                                  COUNT EACH GREY LEVEL
      DO 2 I=1,IASIZE
      J=IA(I)
 2    KN(J+1)=KN(J+1)+1
      NP=IASIZE
      J=1
      MQ=NQ
C                                  GO THRU NQ LEVELS
      DO 3 I=1,NQ
      NL=NP
C                                  GET NEW LEVEL
 4    NL=NL-MQ*KN(J)
      NP=NP-KN(J)
      KN(J)=I-1
      J=J+1
C                                  CHECK FOR LAST LEVEL
       IF(J.GT.NGL) GO TO 6
C                                  INCREMENT AGAIN FOR LEVEL
      IF(MQ*KN(J).LE.NL*2)GO TO 4
C                                  DECREASE NO. OF LEVELS LEFT
 3    MQ=MQ-1
       DO 5 I=J,NGL
 5    KN(I)=NQ-1
      GO TO 8
```

```
C                                     SET LAST LEVEL
C
6       N=(NQ-I)/2
        IF(N.LT.1)GO TO 8
        DO 7 I=1,NGL
7       KN(I)=KN(I)+N
C                                     ASSIGN ELEMENTS TO A LEVEL AND RETURN
C
8       DO 9 I=1,LASIZE
        J=IA(I)
9       IA(I)=KN(J+1)
        RETURN
        END
```

```
CRDDSK2              READ DISC AND LIST GREY TONES
C
C       VERSION II WRITTEN BY RJ BOSLEY                        JAN 1974
C
C       DESCRIPTION OF PROGRAM.
C          THIS SUBROUTINE READS DATA FROM DISC FILE NDSK AND PRINTS OUT
C       THE GREY TONES AND WRITES THEM OUT ONTO AN OUTPUT TAPE.
C
C       INTERNAL PARAMETERS.
C               K                        BAND BEING PROCESSED
C               LINE                     ERTS LINE BEING READ
C
C       ENTRY POINT.
C               CALL RDDSK2(IMAGE,IRSTRT,IRSTOP,NHOR,NDSK,PRNT,TAPE,IFIL)
C
C       INPUT ARGUMENTS.
C               IMAGE                    ARRAY TO READ DATA INTO
C               IRSTRT,IRSTOP            STARTING,STOPING LINES OF DATA
C
C               NHOR                     NUMBER OF HORIZONTAL BLOCKS
C               NDSK                     FILE CODE OF DISC WITH DATA
C               PRNT                     TRUE FOR GREY-TONE LISTING
C               TAPE                     TRUE FOR TAPE OUTPUT
C               IFIL                     OUTPUT TAPE FILE CODE
C
        SUBROUTINE RDDSK2(IMAGE,IRSTRT,IRSTOP,NHOR,NDSK,PRNT,TAPE,IFIL)
C
        DIMENSION IMAGE(41,41)
        LOGICAL PRNT,TAPE
        REWIND NDSK
        K=NDSK-10
C                                        IF NEITHER TAPE NOR PRINT,RETURN
        IF(PRNT) GO TO 2
        IF(TAPE)  GO TO 3
        RETURN
C                                        WRITE HEADING FOR LIST
2       WRITE(6,1) K
1       FORMAT(1H1,'LINE STRIP',20X,'BAND NUMBER IS',I2)
C                                        GO THRU EACH LINE OF DATA
3       DO 50 LINE=IRSTRT,IRSTOP
C                                        GO THRU EACH BLOCK ACROSS THE IMAGE
        DO 50 J=1,NHOR
C                                        READ ONE LINE
        READ(NDSK)  (IMAGE(1,KOL),KOL=1,41)
C                                        WRITE IT OUT
        IF(PRNT)  WRITE(6,100) LINE,J,(IMAGE(1,KOL),KOL=1,41)
        IF(TAPE)  WRITE(IFIL)  LINE,J,(IMAGE(1,KOL),KOL=1,41)
50      CONTINUE
100     FORMAT(1X,I5,I3,41I3)
        RETURN
        END
```

IV.2-b Texture Analysis Program Listings

```
MAINLN
  ERTS (see IV.2-a)
  MAING
    KEQUAN (see IV.2-a)
    PITCHR (see IV.2-a)
    FPLXIT
      INDEX
    IMOMTR
      COR
      IEQPQ1
    RITOWT
```

```
CMAINLN                  ERTS TEXTURE ANALYSIS              JUNE 1973
C
C      WRITTEN BY RJ BOSLEY FOR PROCESSING LAWRENCE DATA    SEPT 1972
C      VERSION 1 BY RJ BOSLEY FOR PROCESSING ERTS DATA      NOV 1972
C      VERSION 2 BY RJ BOSLEY FOR MERGE OPTION              JUNE 1973
C
C*********************START OF DOCUMENTATION COMMENT CARDS**************
*
*
*     PROGRAM IDENTIFICATION.
*        PROGRAM TITLE  - TEXTURE ANALYSIS
*        WRITTEN BY     - RJ BOSLEY
*        DATE WRITTEN   - JUNE 1973
*        SITE NAME      - CRINC, UNIV OF KANSAS
*        SITE ADDRESS   - LAWRENCE, KANSAS
*        PHONE NUMBER   - 1-913-864-4832
*
*     DATE  OF MODIFICATION.
*
*        JUNE 1973 -- MODIFIED FOR MERGE OPTION.
*
*     HARDWARE / SOFTWARE SUMMARY.
*
*        COMPUTER REQUIRED  - HONEYWELL 635
*        SYSTEM EXECUTIVE   - GECOS III
*        MEMORY REQUIRED    - 42K
*        PROGRAM LANGUAGE   - FORTRAN IV AND GMAP
*        PERIPHERALS        - ONE SCRATCH DISK, TWO TAPE DRIVES
*
*     PURPOSE.
*          THIS TEXTURE ANALYSIS PACKAGE WAS WRITTEN IN ORDER TO
*        PROCESS ERTS IMAGERY DATA USING PATTERN RECOGNITION TECHNIQUES.
*
*     METHOD.
*        SEE 'LAND USE CLASSIFICATION',TECH. REPORT NO.2262-1,JANUARY 1973
*
*     INPUT.
*
*        PARAMETER CARDS.
*           1.   TITLE CARD.  THIS CARD IS USED FOR TITLE INFORMATION
*                           AND IS LISTED ON THE OUTPUT LISTING.
*           2.   PARAMETERS ACCORDING TO THE FORTRAN NAMELIST FORMAT
*                           UNDER THE NAME PARAM, SEE BELOW.
*
*     OTHER INPUT.
*
*        ERTS IMAGE   DATA TAPE ON INPUT FILE CODE 'ES'.
*
*     ABORTS.
*
```

```
*        SPECIFIED IN OUTPUT LISTING.
*
*     OUTPUT.
*        PRINTER OUTPUT.
*           1.  PARAMETER VALUES.
*           2.  PICTURE OF IMAGE BEING PROCESSED.
*           3.  COORDINATES OF THE IMAGE.
*           4.  VALUES OF THE EXTRACTED FEATURES.
*           5.  LISTING OF LFX ARRAYS.
*
*        CARD OUTPUT.
*           1.  COORDINATES OF THE IMAGE BEING PROCESSED.
*           2.  VALUES OF THE FEATURES EXTRACTED.
*           3.  CARD COUNT.
*
*        TAPE OUTPUT.
*           1.  COORDINATES OF THE IMAGE BEING PROCESSED.
*           2.  VALUES OF THE FEATURES EXTRACTED.
*
*     RESTRICTIONS.
*        1.  DATA SET IS STANDARD ERTS DATA TAPE FROM NASA.
*        2.  THE NUMBER OF POINTS IN ONE LINE OF THE STRIP BEING
*               PROCESSED MUST NOT EXCEED 192 POINTS.
*        3.  THE SIZE OF EACH IMAGE MUST NOT EXCEED 4096 POINTS.
*        4.  QUANTIZATION MUST BE 32 LEVELS OR LESS.
*        5.  THE ERTS INPUT TAPE MUST HAVE A FILE CODE 'ES'.
*        6.  IBAND MUST NOT EXCEED FOUR.
*
*     SUBPROGRAMS REQUIRED.
*           MAINLN
*              ERTS  READ PROGRAM
*              MAING
*                 KEQUAN
*                 PICTUR
*                 FPLXIT
*                    INDEX
*                 IMOMTR
*                    INDEX
*                    COR
*                    IEQPQ1
*                 RITOWT
*
*     CARD SETUP FOR SAMPLE RUN.
*
*       $          IDENT    9999,ANYNAME
*       $          LIBRARY LB
*       $          OBJECT       M-A-I-N-L-N
*       $             ...  ERTS TEXTURE ANALYSIS PROGRAMS ...
*       $          DKEND
*       $          EXECUTE
```

```
        $      PRMFL    LB,R,S,PATTERN/GEE/LIB
*       $      TAPE     ES,A1DD,,99999,,NNAME,INPUT
*       $      DSPK     02,APR,5L
*       $      LIMITS   32,43K,,23K
*       $      INCODE   IBMF
*       XXXXX  TEST OF TEXTURE ANALYSIS PROGRAMS    XXXXX
*        $PARAM  N11=1,PNCH=1HN,PICTUR=T$END
*       $      ENDJOB
*
*       THIS RUN OF THE TEXTURE ANALYSIS PROGRAMS WILL PROCESS THE ERTS
*       DATA IN 64 X 64 IMAGES, GIVING ONLY PRINTED OUTPUT, PLUS A
*       PICTURE OF THE IMAGES.
*
*
*    BIBLIOGRAPHY.
*       NASA TECHNICAL REPORT NO.2262-1. JAN. 1973, 'LAND USE
*       CLASSIFICATION USING TEXTURE INFORMATION IN ERTS-A MSS IMAGERY'
*       BY K. SHANMUGAM,R. HARALICK, R. BOSLEY.
*
***********************END OF DOCUMENTATION COMMENT CARDS*****************
C
C     DESCRIPTION OF PROGRAM.
C        THIS IS THE MAINLINE OF THE ERTS TEXTURE ANALYSIS PROGRAMS.
C        EACH ERTS IMAGE IS DIVIDED INTO 4 STRIPS,EACH PUT ONTO ONE
C        TAPE.  FOR EXAMPLE, IF THE IMAGES ARE 64 X 64, THEN THE FIRST
C        ERTS DATA TAPE CONTAINS SUBIMAGES 1 TO 12, THE SECOND FROM 13
C        TO 24, ETC, UP TO 48 HORIZONTAL IMAGES.
C          DUE TO CORE LIMITATIONS, EACH INPUT TAPE IS PROCESSED IN
C        STRIPS WITH EACH STRIP TOTALING UP TO 192 POINTS, HORIZONTALLY.
C        IF THE IMAGES ARE 64 X 64, EACH STRIP WILL CONTAIN 3 SUBIMAGES.
C        IMAGES WILL BE PROCESSED AS FOLLOWS, RUN1-(1,1),(1,2),(1,3),
C        (2,1),(2,2),(2,3),....,(36,1),(33,2),(36,3).  RUN 2-(1,4),(1,5),
C        (1,6),(2,4),....,(36,4),(36,5),(36,6).  ETC.
C          NOTE THAT FOR RUN1, N11 IS 1 AND NUMSTR IS 1. FOR RUN 2, N11
C        IS 4 AND NUMSTR IS 2...   AND FOR THE SECOND TAPE FOR RUN 1,
C        N11 IS 13 AND NUMSTR IS 1.  N11 IS THE UPPER-LEFT COLUMN
C        COORDINATE AND IS RELATIVE TO THE ENTIRE IMAGE, WHERE  NUMSTR
C        IS THE STRIP NUMBER RELATIVE TO THE DATA TAPE.
C          ALSO NOTE THAT THE FIRST 8 POINTS AT THE BEGINNING OF EACH
C        LINE ARE LEFT OUT.  IT IS POSSIBLE TO HAVE LESS THAN ONE FULL
C        IMAGE AT THE END OF EACH TAPE THAT CANNOT BE PROCESSED.
C
C
C     DESCRIPTION OF INPUT PARAMETERS UNDER NAMELIST /PARAM/.
C           NQUANT    NO. OF QUANTIZING LEVELS IN IEQPQ1,SET TO 16
C           NUMIM     THE NUMBER OF IMAGES TAKEN HORIZONTALLY IN ONE
C                               PASS
C           NUMSTR    THE STRIP NUMBER OF THE RUN IN RELATION TO THE
C                               ERTS INPUT TAPE
C           NOVERT    THE NUMBER OF IMAGES IN A VERT COL OF THE STRIP
```

```
C             NUMLIN    THE NUMBER OF LINES IN EACH NUMLIN X NUMPPL
C                                SUB-IMAGE
C             NUMPPL    THE NUMBER OF PTS PER LINE IN EACH SUB-IMAGE
C                       ***NOTE***NUMPPL*NUMIM MUST NOT EXCEED 192
C                       ***EXAMPLE***IF NUMPPL=64,THEN NUMIM=3
C                       ***EXAMPLE***IF NUMPPL=32,THEN NUMIM=6
C             IBAND     THE SPECTRUM BAND TO BE PROCESSED,FROM 1 TO 4
C             PNCH      SPECIFIES THE OUTPUT OPTION---Y FOR CARDS, T
C                                FOR TAPE, AND N FOR PRINTER ONLY
C                       ***NOTE***PNCH MUST BE DENOTED AS A HOLLERITH
C                                CONSTANT IN THE DATA CARD
C             N11       THE UPPER LEFT COLUMN COORDINATE FOR THE STRIP
C                                BEING PROCESSED
C             IF        THE FILE CODE OF THE OUTPUT TAPE--ASSUMED TO BE
C                                IN POSITION
C             NBSKIP    THE NUMBER OF VERTICAL ROWS OF SUB-IMAGES TO
C                                BE SKIPPED PRIOR TO EXECUTION
C             MERGF     IF .TRUE., THE LEX ARRAYS WILL BE MERGED TO ONE
C             PICTUR    IF .TRUE., A PICTURE OF EACH SUBIMAGE WILL BE
C                                PRINTED
C                       ***NOTE***PROCESSING IS, APPROXIMATELY--
C                                100 PERCENT= MERGE-OFF,PICTUR-ON
C                                125 PERCENT= MERGE-ON ,PICTUR-ON
C                                158 PERCENT= MERGE-ON ,PICTUR-OFF
C
C
      DIMENSION ILINE(4096),TITLE(14)
      COMMON /Q/ NQUANT
      COMMON M1,N1,F(15),IMAX,IMIN,NUMPPL,NUMLIN,NBUBL,IR1,IR2,IR3,IR4,
     1 DUMMY(29),LEAST1,NRED,NLAYER,NSTART,NTIMES,ND,PNCH
      COMMON /EC/ENTROP(4),DIFENT(4),DIFAVE(4),DIFVAR(4),SUMENT(4),
     1SUMAVE(4),SUMVAR(4)
      COMMON /CORREL/CORINF(4),CORMUT(4),CORMAX(4)
      COMMON IMAGE(64,192)
      LOGICAL MERGE,PICTUR
      NAMELIST/PARAM/NUMIM,NUMSTR,NBVERT,IBAND,N11,MERGE,PICTUR,NBSKIP,
     1 PNCH,NUMPPL,NUMLIN,NPED,NLAYER,NSTART,NTIMES,NQUANT,IF
      DATA IBAND,NUMIM,NBVERT,NBSKIP,MERGE,PICTUR,N11/2,3,36,0,.TRUE.,
     1 .FALSE.,1/,TAPE/1HT/,NUMSTR/1/,IF/03/,Y/1HY/,N/1HN/
      PNCH=TAPE
      NUMPPL=64
      NUMLIN=64
      NPED=1
      NLAYER=1
      NSTART=1
      NTIMES=1
      NQUANT=16
C
C     ************SECTION I--PREPARE FOR ERTS READING AND PROCESSING*****
C
```

```
C
C      READ TITLE AND WRITE IT OUT
C
      READ(5,6) (TITLE(I),I=1,14)
      WRITE(6,21)
      WRITE(6,7)(TITLE(I),I=1,14)
21    FORMAT(///////40X,'ERTS TEXTURE ANALYSIS'////)
6     FORMAT(13A6,A2)
7     FORMAT(20X,13A6,A2////)
C
C      READ ERTS TEXTURE PARAMETERS
C
      READ(5,PARAM)
      M=NUMIM*NUMPPL
      IF(M.LE.192)  GO TO 2
      WRITE(6,1)
1     FORMAT(' NUMBER OF IMAGES TIMES THE NUMBER OF POINTS IN EACH LINE
     1MUST NOT EXCEED 192 ')
      STOP
C
C      INITIALIZE THE ERTS READ PROGRAM AND WRITE OUT PARAMETERS
C
2     CALL EINIT(LENGTH)
C
      WRITE(6,11) LENGTH,NUMIM,NUMSTR,NBVERT,IBAND,NBSKIP
      WRITE(6,101) PNCH,NUMPPL,NUMLIN,NRED,NSTART,NTIMES,NQUANT
11    FORMAT(10X,'LENGTH OF ERTS LINE IS ',
     1  I5,' POINTS'/10X,'NUMBER OF HORIZONTAL IMAGES,NUMIM,IS ',I2,/
     210X,'THIS STRIP IS NUMBER ',I2,/10X,'NUMBER OF VERTICAL IMAGES CON
     3TAINED IN STRIP IS ',I2,/10X,'PROCESSING WILL BE ON BAND ',I2,/
     410X,'SKIPPED DOWN ',I3,' VERTICAL IMAGES BEFORE STARTING'////)
101   FORMAT(10X,'PUNCH=',A3,'  NUMPPL=',I4,'NUMLIN=',I4,'  NRED=',I3,
     1'  NSTART=',I3,'  NTIMES=',I3,'  NQUANT=',I4)
      IF(MERGE) WRITE(6,3)
3     FORMAT(//10X,'THE FOUR TEX ARRAYS HAVE BEEN MERGED INTO ONE ARRAY'
     1 //)
      IF(.NOT.PICTUR)  WRITE(6,4)
4     FORMAT(' THE PICTURE OPTION IS OFF '///)
C
C      SKIP THE FIRST NBSKIP ROWS OF IMAGES
C
      NOSK=NUMLIN*NBSKIP
      CALL ESKIP(NOSK)
C
C      GO DOWN THE STRIP
C      IMAGE COORDINATES (M1,N1) ARE TRANSFERRED IN COMMON
C      M1 GIVES THE ROW COUNT GOING DOWN THE STRIP
C
      IBEGIN=((NUMSTR-1)*(192*4))+IBAND
C
```

```
C     MOVE THE IMAGE TO  THE RIGHT BY EIGHT POINTS
C
      IBEGIN=IBEGIN+32
      JSTOP=IBEGIN+((NUMIM*NUMPPL)*4)-IBAND
      NB=NBSKIP+1
      DO 99 M1=NB,NBVERT
C
C      **********SECTION II--READ ERTS AND MOVE DATA INTO IMAGE*********
C
C     ILINE IS THE ARRAY INTO WHICH THE ERTS DATA IS READ--ILINE AND
C      IWORK USE THE SAME STORAGE SPACE
C     MDOWN GIVES THE ROW COUNT IN IMAGE FROM 1 DOWN TO NUMLIN
C
      DO 90 MDOWN=1,NUMLIN
C
C     READ ERTS LINE BY LINE
C
      CALL EREAD(ILINE,LN)
C
C     LN,RETURNED BY EREAD, GIVES THE LINE NUMBER, OR ERROR INDICATION
C
      IF(LN.EQ.0) GO TO 996
C
C     MOVE EVERY POINT IN THE LINE  THAT BELONGS TO IBAND, INTO IMAGE
C     JSTOP GIVES THE STOPPING POINT IN ILINE FOR THE TRANSFER
C     THE TRANSFER IS INCREMENTED BY 4, THE NUMBER OF BANDS
C     LCOUNT GOES FROM 1 TO NUMPPL*NUMIM, GIVING THE LENGTH COUNT
C
      LCOUNT=0
      DO 80 IPOINT=IBEGIN,JSTOP,4
      LCOUNT=LCOUNT+1
      IMAGE(MDOWN,LCOUNT)=ILINE(IPOINT)
80    CONTINUE
90    CONTINUE
C
C     IMAGE IS FULL, START TEXTURE ANALYSIS
C
C      **********SECTION III--PROCESS IMAGE, BLOCK BY BLOCK**********
C
C      BLOCK IMAGE INTO NUMLIN X NUMPPL BLOCKS FOR PROCESSING
C
      KE=0
      DO 60 JBLOCK=1,NUMIM
      KS=KE+1
      KE=KS+NUMPPL-1
      KL=0
      DO 59 KLINE=1,NUMLIN
C
C     KL GOES FROM 1 TO NBR OF PTS WHILE KCOL GIVES THE COLUMN COUNT
C
```

```
      DO 58 KCOL=KS,KE
      KL=KL+1
      ILINE(KL)=IMAGE(KLINE,KCOL)
 58   CONTINUE
 59   CONTINUE
      N1=(N11-1)+JBLOCK
C
C     USE ILINE AS A DUMMY ARRAY TO SEND IWORK TO MAING
C
      CALL MAING(ILINE,MERR,MERGE,PICTUR,IF)
C
C     CHECK FOR ERROR CONDITION IN MAING
C
      IF(MERR.EQ.1) GO TO 992
 60   CONTINUE
 99    CONTINUE
C
C      **********SECTION IV--END OF STRIP, FINISH UP**********
C
C     PUT AN EOF MARK ON OUTPUT FILE IF AND WRITE ANOTHER RECORD
C
      ENDFILE IF
      WRITE(IF) (ILINE(K),K=1,10)
      STOP
C
C     ERROR DETECTED---WRITE  FILE MARK AND ANOTHER RECORD
C
 992  WRITE(6,993) M1,N1
 993  FORMAT(10X,'ERROR IN SUBROUTINE MAING, LAST IMAGE WAS (',I2,',',
     1I2,') ')
      ENDFILE IF
      WRITE(IF) (ILINE(K),K=1,10)
      STOP
C
C     ERROR DETECTED---WRITE  FILE MARK AND ANOTHER RECORD
C
 996  WRITE(6,997) M1
 997  FORMAT(10X,'UNEXPECTED EOF ON ERTS, LAST ROW COMPLETED WAS ',I3)
      ENDFILE IF
      WRITE(IF) (ILINE(K),K=1,10)
      STOP
      END
```

```
CMAING             ERTS M-A-I-N-G                              JUNE 1973
C                                                          DATE OCT  1971
C       WRITTEN BY    RMH                                       SEPT 1973
C       VERSION 1 BY RJ BOSLEY FOR LAWRENCE DATA                NOV  1973
C       VERSION 2 BY RJ BOSLEY FOR ERTS DATA PROCESSING         JUNE 1973
C       VERSION 3 BY RJ BOSLEY FOR MERGE OPTION
C
C       DESCRIPTION OF PROGRAM.
C          THIS SUBROUTINE  PREPARES THE IMAGE IN ARRAY IWORK FOR PITCHR
C          AND THEN PROCESSES IT, ACCORDING TO THE MERGE OPTION, CALLING
C          SUBROUTINES FOR THE LEX ARRAYS,CALCULATING THE TEXTURE FEATURES
C          AND THEN WRITTING OUT THE RESULTS.
C
C
C       ENTRY POINT.
C                CALL MAING(IWORK,MERR,MERGE,PICTUR,IF)
C       ARGUMENTS.
C                          IWORK      THE NUMLIN*NUMPPL IMAGE ARRAY
C                          MERR       ERROR FLAG FOR LEX ARRAY SIZE
C                          MERGE      OPTION TO MERGE THE FOUR LEX ARRAYS
C                          PICTUR     OPTION TO PRINT PICTURE OF THE IMAGE
C                          IF         FILE CODE FOR OUTPUT TAPE IN RITOWT
C
C       INTERNAL PARAMETERS.
C                          NUMLIN     THE NUMBER OF LINES IN THE IMAGE
C                          NUMPPL     THE NUMBER OF POINTS PER IMAGE LINE
C                          NDIM       MAXIMUM ALLOWABLE SIZE OF THE LEX
C                                     ARRAYS, NUMPPL*NUMLIN
C                          NQ         NUMBER OF QUANTIZING LEVELS FOR KFQUAN
C                          IMAX       MAXIMUM GREY TONE LEVEL IN IWORK
C                          IMIN       MINIMUM GREY TONE LEVEL IN IWORK
C                          IS         STARTING PT OF A ROW OF IWORK
C                          IE         ENDING POINT OF THE ROW IN IWORK
C                          NGL        NUMBER OF GREY TONE LEVELS IN IWORK
C                          IASIZE     SIZE OF IWORK ARRAY
C                          HORZ       HORIZONTAL SCALE FACTOR FOR PITCHR
C                          VERT       VERTICAL SCALE FACTOR FOR PITCHR
C                          LEAST1     ONE GREY TONE LEVEL BELOW IMIN
C                          NOBL       NUMBER OF GREY TONE LEVELS IN IWORK
C                          NBURL      THE NUMBER OF LEVELS IN THE TRIANGULAR
C                                        LEX ARRAY
C                          L1...L6    ADDRESS INDEXS FOR THE LEX ARRAYS
C
C
C
C
        SUBROUTINE MAING(IWORK,MERR,MERGE,PICTUR,IF)
C
        LOGICAL MERGE,PICTUR
        DIMENSION IWORK(4096)
```

```
      DIMENSION G(64),IQ(64)
      COMMON /Q/ NQUANT
      COMMON M1,N1,F(15),IMAX,IMIN,NUMPPL,NUMLIN,NDUBL,IR1,IR2,IR3,IR4,
     1 DUMMY(29),LEAST1,NRED,NLAYER,NSTART,NTIMES,NJ,PNCH
      COMMON /EN/ENTROP(4),DIFENT(4),DIFAVE(4),DIFVAR(4),SUMENT(4),
     1SUMAVE(4),SUMVAR(4)
      COMMON /CORREL/CORINF(4),CORMUT(4),CORMAX(4)
      COMMON IMAGE(64,192)
      NDIM=NUMPPL*NUMLIN
      NQ=32
C
C      REWIND SCRATCH FILE, COPY IMAGE IN A LINE BY LINE FASHION ONTO
C      THE SCRATCH FILE, AND DETERMINE THE MINIMUM AND MAXIMUM GREY TONE
C
      REWIND 2
C
C      FIRST, QUANTIZE THE ARRAY TO NQ LEVELS FOR PICTURE AND EFFICIENCY
C
      IMAX=-10000
      IMIN=10000
      DO 13 J=1,NUMLIN
      IS=NUMPPL*(J-1)+1
      IE=NUMPPL*J
      DO 12 K=IS,IE
      IF(IMIN.GT.IWORK(K)) IMIN=IWORK(K)
      IF(IMAX.LT.IWORK(K)) IMAX=IWORK(K)
   12 CONTINUE
   13 CONTINUE
      NGL=IMAX+1
      IASIZE=NUMPPL*NUMLIN
      CALL KEQUAN(IWORK,NGL,NQ,IASIZE)
C
C      COPY IMAGE ON SCRATCH FILE AFTER QUANTIZATION
C
      DO 20 I=1,NUMLIN
      IS=NUMPPL*(I-1)+1
      IE=NUMPPL*I
      WRITE(2)   (IWORK(K),K=IS,IE)
   20 CONTINUE
C
C      THE MAXIMUM AFTER KEQUAN QUANTIZES TO NQ LEVELS IS NQ-1,MINIMUM=0
C
      IMAX=NQ-1
      IMIN=0
C
C      TEST FOR PICTURE OPTION
C
      IF(.NOT.PICTUR) GO TO 16
C
C      TRANSPOSE IWORK FOR PITCHR
```

```
C
      L=0
      DO 14 I=1,NUMLIN
      L=L+1
      IS=NUMPPL*(I-1)+1
      IE=NUMPPL*I
      K=0
      DO 15 J=IS,IE
      K=K+1
      IMAGE(L,K)=IWORK(J)
   15 CONTINUE
   14 CONTINUE
C
C     PRINT OUT PICTURE OF THE IMAGE
C
      HORZ=(64.0*0.90)/FLOAT(NUMPPL)
      VERT=(64.0*0.75)/FLOAT(NUMLIN)
      CALL PITCHR(IMAGE,NUMLIN,NUMPPL,0,1,0,IMAX,,0,,,,HORZ,VERT, )
   16 LEAST1=IMIN-1
C
C      NOBL IS THE NUMBER OF BRIGHTENESS LEVELS
C
      NOBL=IMAX-LEAST1
C
C      NBUBL IS THE NUMBER OF LEVELS IN THE TRIANGULAR LEX ARRAY
C
      NBUBL=NOBL*(NOBL+1)/2
C
C      SET UP THE INDEXS FOR THE LEX ARRAYS
C
      L1=1
      L2=L1+NUMPPL*2
      L3=L2+NBUBL
      L4=L3+NBUBL
      L5=L4+NBUBL
      L6=L5+NBUBL-1
C
C      CHECK THE SIZE
C
      IF(L6.GT.NDIM) GO TO 78
      DO 4 NN=NSTART,NTIMES
      REWIND 2
      NLAYER=NN-1
C
C      GET THE LEX ARRAYS
C
      CALL FPLXIT(IWORK(L1),IWORK(L2),IWORK(L3),IWORK(L4),IWORK(L5),
     1  NUMPPL,MERGE)
C
C      CALCULATE THE TEXTURE FEATURES
```

```
C                                                                        MAING15
      CALL IMOMTR(IWORK(L2),IWORK(L3),IWORK(L4),IWORK(L5),G,I1,MERGE)    MAING15
C                                                                        MAING15
C       OUTPUT THE TEXTURE DATA                                          MAING15
C                                                                        MAING15
      CALL RITOWT(IWORK(L2),IWORK(L3),IWORK(L4),IWORK(L5),G,I1,MERGE,IF,MAING15
     1 PICTUR)                                                           MAING15
    4 CONTINUE                                                           MAING15
C                                                                        MAING15
C       SET ERROR INDICATOR TO NO ERRORS                                 MAING16
C                                                                        MAING16
      MERR=0                                                             MAING16
      RETURN                                                             MAING16
   78 WRITE(6,104) NDIM,L6                                               MAING16
  104 FORMAT(6H NDIM=,I5,16H NDIM MUST BE = ,I7)                         MAING16
      MERR=1                                                             MAING16
      RETURN                                                             MAING16
      END                                                                MAING16
```

```
CFPLXIT                        F-P-L-X-I-T                    JUNE 1973
C
C      WRITTEN BY RMH                                         SEPT 1971
C      VERSION 1 BY RJ BOSLEY FOR MERGE               JUNE 1973
C
C      DESCRIPTION OF PROGRAM.
C       THIS SUBROUTINE COMPUTES FOUR NEAREST NEIGHBOR GREY TONE MA-
C       TRICES, LEX1, LEX2, LEX3, AND LEX4 FOR ANGLES OF 90-DEGREES,
C       0-DEGREES, 135-DEGREES, AND 45-DEGREES RESPECTIVELY.
C      INCLUDED IN THIS SUBROUTINE IS AN OPTION TO MERGE THE FOUR LEX
C      ARRAYS INTO ONE, LEX1
C
C
C      ENTRY POINT.
C         CALL FPLXIT(IDATA,LEX1,LEX2,LEX3,LEX4,NUMPPL,MERGE)
C
C      ARGUMENTS.
C              IDATA               WORKING ARRAY FOR TWO LINES OF IMAGE
C              LEX1-LEX4           ADDRESS INDEXS FOR LEX ARRAYS
C              NUMPPL              NUMBER OF PTS PER IMAGE LINE
C              MERGE               OPTION TO MERGE THE FOUR LEX ARRAYS
C                                     INTO ONE ARRAY, LEX1
C
C      INTERNAL PARAMETERS.
C              NBUBL               NUMBER OF LEVELS IN THE TRIANGULAR
C                                     LEX ARRAYS
C              IST                 POINTER TO FIRST LINE
C              NND                 POINTER TO SECOND LINE
C              NRED                BASE FOR IMAGE REDUCTION
C              NLAYER              THE POWER TO WHICH NRED IS RAISED
C              MM                  AMOUNT OF REDUCTION OF THE IMAGE
C              FILE 2              SCRATCH FILE CONTAINING THE IMAGE
C              I,J,L,K             GREY TONE VALUES OF NEIGHBORING
C                                     RESOLUTION CELLS, ONE TO EACH ANGLE
C              INDEX(I,J)          FUNCTION USED TO RETURN A SINGLE
C                                     SUBSCRIPT FOR THE LEX ARRAY
C                                     INDICATING WHERE ELEMENT (I,J) CAN
C                                     BE FOUND
C
      SUBROUTINE FPLXIT(IDATA,LEX1,LEX2,LEX3,LEX4,NUMPPL,MERGE)
C
      DIMENSION IDATA(NUMPPL,2),LEX1(1),LEX2(1),LEX3(1),LEX4(1)
      COMMON M1,N1,TYPE,F(14)
       COMMON I1,I2,NNMPPL,NUMLIN,NBUBL,IR1,IR2,IR3,IR4,DUMMY(29)
       COMMON LEAST1,NRED,NLAYER,NSTART,NTIMES
      LOGICAL MERGE
C
C      INITIALIZE LEX1, LEX2, LEX3, AND LEX4 ARRAYS TO ZERO
C
       DO 10 I = 1,NBUBL
```

```
      LEX1(I) = 0
      LEX2(I) = 0
      LEX3(I) = 0
   10 LEX4(I) = 0
C
      IR1 = 0
      IR2 = 0
      IR3 = 0
      IR4 = 0
C
C     IST POINTS TO FIRST LINE. NND POINTS TO SECOND LINE
C
      IST = 1
      NND = 2
C
C     NLAYER INDICATES BY HOW MUCH THE IMAGE WILL BE REDUCED.
C     NRED IS THE FACTOR BY WHICH THE IMAGE WILL BE REDUCED. (IT IS THE
C     BASE WHICH IS RAISED TO THE POWER NLAYER.) THEN, BY DEFINING THE
C     QUANTITY MM, WHERE MM = NRED**NLAYER, WE HAVE A SINGLE FACTOR
C     THAT DETERMINES THE REDUCTION BASE AND THE AMOUNT OF THE REDUC-
C     ION. IF, FOR EXAMPLE, NRED = 2, AND NLAYER RANGES FROM 0 TO 3 --
C     THIS RANGE IS DETERMINED BY THE PARAMETER NTIMES (SEE 'MAIN'),
C     THE RESULTANT PROCESSING WILL YIELD FOUR IMAGES THAT WILL BE SUC-
C     CESSIVELY REDUCED BY 1, 1/2, 1/4, AND 1/8 RESPECTIVELY.
C
      MM = NRED**NLAYER
      NUMPL2 = NUMPPL/MM
      DO 111 KK1=1,MM
      DO 111 KK2=1,MM
C
C     GET THE FIRST LINE OF DATA FROM DISC FILE 02
C
      DO 8 LL=1,KK1
    8 READ(2) (IDATA(L,IST),L=1,NUMPPL)
      N = 0
      DO 29 J=KK2,NUMPPL,MM
      N = N + 1
   29 IDATA(N,IST) = IDATA(J,IST)
C
      MMM=MM+KK1
      DO 1 LCNT = MMM,NUMLIN,MM
C
C     GET THE SECOND LINE OF DATA. AFTER EACH ITERATION, THE OLD SEC-
C     OND LINE BECOMES THE NEW FIRST LINE.
C
      DO 18 LL=1,MM
   18 READ(2) (IDATA(L,NND),L=1,NUMPPL)
      N = 0
      DO 19 J=KK2,NUMPPL,MM
      N = N + 1
```

```
   19 IDATA(N,NND) = IDATA(J,NND)
C
C
C     SET I, L, J, AND K EQUAL TO THE (NORMALIZED) VALUES OF GREY TONES
C     OF RESOLUTION CELLS IN POSITIONS (1,IST), (1,NND), (2,IST), AND
C     (2,NND) RESPECTIVELY.
C
      I = IDATA(1,IST) - LEAST1
      J = IDATA(2,IST) - LEAST1
      L = IDATA(1,NND) - LEAST1
      K = IDATA(2,NND) - LEAST1
C
C     PUT TWO DIMENSIONAL INFORMATION INTO ONE DIMENSIONAL FORM. THE
C     FUNCTION NEEDED TO CONVERT A DOUBLE SUBSCRIPTED ARRAY, IMM(X,Y)
C     INTO A SINGLE SUBSCRIPTED ARRAY, IMM(Z), IS OF THE FORM G(X) +
C     F(Y), WHERE G(X) = (X-1)*X/2, AND F(Y) = Y. THEREFORE,
C                Z = (X-1)*X/2 + Y.
C     THIS IS DONE IN THE PROGRAM BY THE EXTERNAL FUNCTION, INDEX(X,Y).
C
C     SINCE THE ORDER OF OCCURRENCE OF THE GREY TONES BELONGING TO A
C     RESOLUTION CELL PAIR IS IMMATERIAL, THE ARRAYS ARE SYMMETRIC.
C     WE LET THE LARGER OF THE TWO HAVE THE FIRST SUBSCRIPT, I.E.,
C     THE ARRAY IS STORED IN LOWER TRIANGULAR FORM. THE ORDER OF SUB-
C     SCRIPTING IS AS FOLLOWS,
C                            IMM(1,1) = IMM(1),
C                            IMM(2,1) = IMM(2),
C                            IMM(2,2) = IMM(3),
C                            IMM(3,1) = IMM(4),
C                                     .
C                                     .
C                                     .
C                            IMM(NOBL,NOBL) = IMM(NBUBL), WHERE
C     NBUBL = NOBL*(NOBL + 1)/2, AND NOBL IS THE TOTAL NUMBER OF GREY
C     TONES IN THE ARRRAY.
C
C
C     THE SCANNING PROCEDURE, THAT IS, THE METHOD BY WHICH THE PAIR-
C     WISE COMPARISONS ARE MADE, IS DESCRIBED BELOW FOR THE GENERAL
C     CASE.
C     CONSIDER A RESOLUTION CELL WITH SPATIAL COORDINATES (M,N), AND
C     CALL THIS CELL 'I'. THE SCANNING OPERATION BEGINS IN THE UPPER
C     LEFT HAND CORNER OF THE IMAGE (THE FIRST POSITION OF'I' IS IN
C     THAT OF RESOLUTION CELL (1,1) AND IT THEN PROCEEDS BY COMPARING
C     THE GREY TONE OF 'I' WITH, AT MOST, FOUR GREY TONES OF ITS NEIGH-
C     BORING RESOLUTION CELLS.
C     THAT 'I' NEVER NEEDS TO CONSIDER MORE THAN FOUR NEAREST NEIGHBORS
C     CAN BE SEEN FROM THE DIAGRAM OF THE SEARCH PATTERN SHOWN BELOW
C
C                I  J
C             M  L  K
```

```
C
C     ON A GIVEN ITERATION, 'I' WILL LOOK FIRST AT ITS NEAREST VERTICAL
C     NEIGHBOR ('L'), NEXT AT ITS NEAREST HORIZONTAL NEIGHBOR ('J'),
C     THIRD AT ITS LOWER RIGHT NEIGHBOR ('K'), AND FOURTH AT ITS LOWER
C     LEFT DIAGONAL NEIGHBOR ('M').  'I' THE MOVES INTO THE POSITION OF
C     THE LEFT-MOST RESOLUTION CELL OF THE PREVIOUSLY SCANNED SECOND
C     ROW (THE POSITION OCCUPIED BY 'M'). THE OPERATION IS REPEATED UN-
C     TIL ALL NEIGHBORING PAIRS OF RESOLUTION CELLS HAVE BEEN EXAMINED.
C
C
C     MAKE COUNT FOR THE FIRST TWO COLUMNS.
C
C
      IL = INDEX(I,L)
C
C     COUNT VERTICALLY ADJACENT (90-DEGREE) NEAREST NEIGHBORS FOR FIRST
C     TWO COLUMNS.
C
      LEX1(IL) = LEX1(IL) + 1
      IR1 = IR1 + 1
C
      IJ = INDEX(I,J)
C
C     COUNT HORIZONTALLY ADJACENT (0-DEGREE) NEAREST NEIGHBORS FOR THE
C     FIRST TWO COLUMNS.
C
      LEX2(IJ) = LEX2(IJ) + 1
      IR2 = IR2 + 1
C
      IK = INDEX(I,K)
C
C     COUNT 'LEFT DIAGONALLY' ADJACENT (135-DEGREE) NEAREST NEIGHBORS
C     FOR FIRST TWO COLUMNS.
C
      LEX3(IK) = LEX3(IK) + 1
      IR3 = IR3 + 1
C
C                 NOW SHIFT ONE COLUMN TO THE RIGHT AND CONTINUE THE
C     PROCEDURE FOR GENERAL CASE IN WHICH A RESOLUTION CELL (I) HAS
C     ONE VERTICAL NEAREST NEIGHBOR (L), ONE HORIZONTAL (J), ONE LOW-
C     ER RIGHT DIAGONAL (K), AND ONE LOWER LEFT DIAGONAL (M). ITERATE
C     UP TO NEXT TO LAST COLUMN.
C
      DO 2 N = 3,NUMPL2
      I = J
      M = L
      L = K
      J = IDATA(N,IST) - LEAST1
      K = IDATA(N,NND) - LEAST1
C
```

```
      IL = INDEX(I,L)
C
C     COUNT VERTICALLY ADJACENT (90-DEGREE) NEAREST NEIGHBORS.
C
      LEX1(IL) = LEX1(IL) + 1
      IR1 = IR1 + 1
C
      IJ = INDEX(I,J)
C
C     COUNT HORIZONTALLY ADJACENT (0-DEGREE) NEAREST NEIGHBORS.
C
      LEX2(IJ)= LEX2(IJ) + 1
      IR2 = IR2 + 1
C
      IK = INDEX(I,K)
C
C     COUNT 'LEFT DIAGONALLY' ADJACENT (135-DEGREE) NEAREST NEIGHBORS.
C
      LEX3(IK) = LEX3(IK) + 1
      IR3 = IR3 + 1
C
      IM = INDEX(I,M)
C
C     COUNT 'RIGHT DIAGONALLY' (45-DEGREE) ADJACENT NEAREST NEIGHBORS.
C
      LEX4(IM) = LEX4(IM) + 1
      IR4 = IR4 + 1
C
    2 CONTINUE
C
C     MAKE COUNT FOR LAST COLUMN.
C
      I = J
      M = L
      L = K
C
      IL = INDEX(I,L)
C
C     COUNT VERTICALLY   ADJACENT NEAREST NEIGHBORS FOR LAST COLUMN.
C
      LEX1(IL) = LEX1(IL) + 1
      IR1 = IR1 + 1
C
      IM = INDEX(I,M)
C
C     COUNT 'RIGHT DIAGONALLY' ADJACENT NEAREST NEIGHBORS FOR THE LAST
C     COLUMN.
C
      LEX4(IM) = LEX4(IM) + 1
      IR4 = IR4 + 1
```

```
C
C     INTERCHANGE THE LINE POINTERS.
C
      MN = IST
      IST = NND
      NND = MN
C
    1 CONTINUE
C
C     MAKE COUNT FOR LAST ROW.
C
      I=IDATA(1,IST) - LEAST1
      J=IDATA(2,IST) - LEAST1
C
      IJ = INDEX(I,J)
C
C     COUNT HORIZONTALLY ADJACENT NEAREST NEIGHBORS FOR FIRST TWO COL-
C     UMNS OF LAST ROW.
C
      LEX2(IJ) = LEX2(IJ) + 1
      IR2 = IR2 + 1
C
C     COMPLETE COUNT FOR LAST ROW.
C
      DO 12 N = 3,NUMPL2
      I = J
      J = IDATA(N,IST) - LEAST1
C
      IJ = INDEX(I,J)
C
C     COUNT HORIZONTALLY ADJACENT NEAREST NEIGHBORS FOR REMAINDER OF
C     LAST ROW.
C
      LEX2(IJ) = LEX2(IJ) + 1
      IR2 = IR2 + 1
   12 CONTINUE
      REWIND 02
  111 CONTINUE
C
C
C      NOW DOUBLE THE DIAGONAL TO MAKE EVERYTHING COME OUT RIGHT
C
      NOBL=I1-I2+1
      DO 100 I=1,NOBL
      II=INDEX(I,I)
      LEX1(II)=2*LEX1(II)
      LEX2(II)=2*LEX2(II)
      LEX3(II)=LEX3(II)*2
      LEX4(II)=LEX4(II)*2
  100 CONTINUE
```

```
      IF(.NOT.MERGE) RETURN                                            FPLXITO
C                                                                      FPLXITO
C      IF MERGE IS TRUE, SUM ALL ARRAYS INTO LEX1                      FPLXITO
C                                                                      FPLXITO
      DO 112 I=1,NBUBL                                                 FPLXITO
      LEX1(I)=LEX1(I)+LEX2(I)+LEX3(I)+LEX4(I)                          FPLXITO
  112 CONTINUE                                                         FPLXITO
       RETURN                                                          FPLXITO
       END                                                             FPLXITO

769 WORDS OF MEMORY USED BY THIS COMPILATION
```

```
CINDEX               I-N-D-E-X
C
C     WRITTEN BY RMH                                    SEPT 1971
C
C      GIVEN THE ROW AND COLUMN SUBSCRIPTS I AND L, INDEX RETURNS
C      THE SINGLE SUBSCRIPT FOR THE LEX ARRAY INDICATING WHERE
C      ELEMENT (I,L) CAN BE FOUND.
C
C
C        1
C        2    3
C        4    5    6
C        7    8    9    10
C
      FUNCTION INDEX(I,L)
      INDEX1(I,L) = (I-1)*I/2 + L
      IF(I.GT.L) GO TO 1
      INDEX = INDEX1(L,I)
      RETURN
    1 INDEX = INDEX1(I,L)
      RETURN
      END
```

```
CIMOMTR                        I-M-O-M-T-R                    JUNE 1973
C
C     WRITTEN BY RMH                                          SEPT 1971
C     VERSION 1 BY SAM SHANMUGAM FOR LAWRENCE DATA            JUNE 1972
C     VERSION 2 BY RJ BOSLEY FOR MERGE OPTION                 JUNE 1973
C
C     DESCRIPTION OF PROGRAM.
C        THIS PROGRAM CALCULATES THE MOMENT TEXTURE STATISTICS (AS
C     DEFINED BELOW UNDER TEXTURAL FEATURES)  FROM THE LEX ARRAYS,
C      ACCORDING TO THE MERGE OPTION.
C
C
C     ENTRY POINT.
C        CALL IMOMTR(LEX1,LEX2,LEX3,LEX4,F,IQ,MERGE)
C
C     ARGUMENTS.
C              LEX1-LEX4           ADDRESS INDEXS FOR LEX ARRAYS
C              F                   CUMULATIVE DISTRIBUTION FUNCTION
C              IQ                  QUANTIZED OUTPUT ARRAY OF IEQPQ1
C              MERGE               OPTION TO MERGE THE FOUR LEX ARRAYS
C                                     INTO ONE ARRAY
C
C
C     INTERNAL PARAMETERS.
C              ANGMOM...CORMAX     TEXTURAL FEATURES--SEE BELOW
C              NQUANT              NUMBER OF QUANTIZING LEVELS FOR IEQPQ1
C              NQUAN2              TWICE NQUANT
C              IMAX                MAXIMUM NUMBER OF GREY TONE LEVELS
C              IMIN                MINIMUM NUMBER OF GREY TONE LEVELS
C              IR1-IR4             THE NUMBER OF RESOLUTION CELL PAIRS
C              R1-R4               INVERSE OF IR1-IR4
C                                     COUNTED IN EACH LEX ARRAY
C                                  SCRATCH ARRAY USED BY SUBROUTINE COR
C              QD                  ARRAY OF JOINT PROBABILITIES
C              P(XY)               NUMBER OF GREY TONE LEVELS
C              NOBL                SUM OF ELEMENTS OF THE LEX ARRAY
C              NAD01-NAD04
C
C      TEXTURAL FEATURES.
C       ANGMOM= SUM SUM P(I,J)*P(I,J)
C                I   J
C
C
C       AMEAN= SUM SUM I*P(I,J)
C               I   J
C
C
C       AMEAN= SUM SUM J*P(I,J)
C               I   J
C
C
C       SGMASQ= SUM SUM ((I-AMEAN)**2)*P(I,J)
```

```
C            I    J
C
C
C     SGMASQ= SUM SUM ((J-AMEAN)**2)*P(I,J)
C             I    J
C
C
C     SGMAXY= SUM SUM (I-AMEAN)*J-AMEAN)P(I,J)
C             I    J
C
C
C     IVDMOM= SUM SUM (P(I,J)/(1+(I J)**2)
C             I    J
C
C
C     RATIO=SGMAXY/SMGASQ
C
C
C     ENTROP=-SUM SUM P(I,J)LOG(P(I,J))
C             I    J
C
C
C
C     DIF(K)= SUM SUM P(I,J)
C          ABS(I-J)=K
C
C     SUM(K) = SUM SUM P(I,J)
C              I+J=K
C
C     DIFENT = SUM DIF(K)*LOG(DIF(K)) *(-1)
C              K
C
C     DIFAVE = SUM K*DIF(K)
C               K
C
C     DIFVAR IS THE VARIANCE OF THE DISTRIBUTION DIF
C
C
C      NOTE THAT
C            DIFMOM=2*(SGMASQ-SGMAXY)
C     THE INTEGERS 1,2,3,4 FOLLOWING THE VARIABLE NAMES CORRESPONDE TO
C     THE FOUR ANGLES--
C
C                SUBSCRIPT                  ANGLE
C            LEX ARRAY  MOMENT ARRAY
C
C                1            3            90 DEGREES
C                2            1             0 DEGREES
C                3            4           135 DEGREES
C                4            2            45 DEGREES
```

```
C
C     IF MERGE IS .TRUE., THE FOUR LEX ARRAYS HAVE BEEN MERGED INTO
C     LEX1.
C
C
      SUBROUTINE IMOMTR(LEX1,LEX2,LEX3,LEX4,F,IQ,MERGE)
C
      REAL IVDMOM
      LOGICAL MERGE
      DIMENSION LEX1(1),LEX2(1),LEX3(1),LEX4(1)
      DIMENSION F(64),IQ(64)
      DIMENSION DIF1(64),DIF2(64),DIF3(64),DIF4(64)
      DIMENSION SUM1(128),SUM2(128),SUM3(128),SUM4(128)
      DIMENSION P(3600),QQ(300)
      EQUIVALENCE(P(1),QQ(1))
      COMMON /Q/ NQUANT
      COMMON M1,N1,TYPE,G(14)
      COMMON IMAX,IMIN,NUMPPL,NUMLIN,NBUBL,IR1,IR2,IR3,IR4,ANGMOM(4),
     1 AMEAN(4),SGMASQ(4),SGMAXY(4),DIFMOM(4),RATIO(4),IVDMOM(4),TMEAN
      COMMON /EC/ENTROP(4),DIFENT(4),DIFAVE(4),DIFVAR(4),SUMENT(4),
     1SUMAVE(4),SUMVAR(4)
      COMMON /CORREL/CORINF(4),CORMUT(4),CORMAX(4)
C
C      INITIALIZE ARRAYS TO ZERO
C
      DO 1 I=1,4
      IVDMOM(I)=0
      ANGMOM(I)=0
      AMEAN(I)=0
      SGMASQ(I)=0
      SGMAXY(I)=0
      ENTROP(I)=0.0
      DIFENT(I)=0.0
      DIFAVE(I)=0.0
      DIFVAR(I)=0.0
       SUMENT(I)=0.0
       SUMAVE(I)=0.0
      SUMVAR(I)=0.0
    1 RATIO(I)=0
      DO 86 K=1,NQUANT
      DIF1(K)=0.0
      DIF2(K)=0.0
      DIF3(K)=0.0
   86 DIF4(K)=0.0
      NQUAN2=2*NQUANT
      DO 87 KS=1,NQUAN2
      SUM1(KS)=0.0
      SUM2(KS)=0.0
      SUM3(KS)=0.0
   87 SUM4(KS)=0.0
```

```
C
C      GET THE NUMBER OF BRIGHTNESS LEVELS,NOBL
C      IF THE LEX ARRAY WERE SQUARE AND NOT COMPACTED, IT WOULD BE
C      DIMENSIONED NOBL BY NOBL
C
      NOBL=IMAX-IMIN+1
C
C      NOW DETERMINE THE TOTAL NUMBER OF RESOLUTION CELL PAIRS
C      COUNTED IN EACH OF THE LEX ARRAYS
C
      IR1=0
      IR2=0
      IR3=0
      IR4=0
      IF(.NOT.MERGE) GO TO 40
      DO 42 I=1,NOBL
      DO 42 J=1,NOBL
      IJ=INDEX(I,J)
      IR1=IR1+LEX1(IJ)
42    CONTINUE
      R1=1./FLOAT(IR1)
      GO TO 41
40    DO 5 I=1,NOBL
      DO 5 J=1,NOBL
      IJ=INDEX(I,J)
      IR1=IR1+LEX1(IJ)
      IR2=IR2+LEX2(IJ)
      IR3=IR3+LEX3(IJ)
    5 IR4=IR4+LEX4(IJ)
C
C      GET R1,R2,R3,R4 TO SAVE DIVISIONS
C
      R1=1./FLOAT(IR1)
      R2=1./FLOAT(IR2)
      R3=1./FLOAT(IR3)
      R4=1./FLOAT(IR4)
C
C      FIND THE CORRELATION MEASURES
C      PUT THE LEX ARRAYS IN P MATRIX AND CALL CORRELATION ROUTINE
C
C       DO LEX2 ARRAY
C
      JJ=0
      DO 201 I=1,NOBL
      DO 201 J=1,NOBL
      IJ=INDEX(I,J)
      JJ=JJ+1
201   P(JJ)=FLOAT(LEX2(IJ))*R1
      CALL COR(P,NOBL,1,QD,COR1,COR2,COR3)
      CORINF(1)=COR1
```

```
      CORMUT(1)=COR2
      CORMAX(1)=COR3
      JJ=0
C
C      DO LEX4 ARRAY
C
      DO 211 I=1,NOBL
      DO 211 J=1,NOBL
      IJ=INDEX(I,J)
      JJ=JJ+1
  211 P(JJ)=FLOAT(LEX4(IJ))*R2
      CALL COR(P,NOBL,1,QD,COR1,COR2,COR3)
      CORINF(2)=COR1
      CORMUT(2)=COR2
      CORMAX(2)=COR3
41    JJ=0
C
C      DO LEX1 ARRAY
C
      DO 221 I=1,NOBL
      DO 221 J=1,NOBL
      IJ=INDEX(I,J)
      JJ=JJ+1
  221 P(JJ)=FLOAT(LEX1(IJ))*R1
      CALL COR(P,NOBL,1,QD,COR1,COR2,COR3)
      CORINF(3)=COR1
      CORMUT(3)=COR2
      CORMAX(3)=COR3
      IF(MERGE) GO TO 43
      JJ=0
C
C      DO LEX3 ARRAY
C
      DO 231 I=1,NOBL
      DO 231 J=1,NOBL
      IJ=INDEX(I,J)
      JJ=JJ+1
  231 P(JJ)=FLOAT(LEX3(IJ))*R4
      CALL COR(P,NOBL,1,QD,COR1,COR2,COR3)
      CORINF(4)=COR1
      CORMUT(4)=COR2
      CORMAX(4)=COR3
C
C      GET THE PROBABILITY FUNCITON IN F
C
43    DO 379 I=1,64
      IO(I)=0
  379 F(I)=0
C
C      IF MERGE, GO TO SECTION II TO MAKE THE COMPUTATIONS
```

```
C
      IF(MERGE) GO TO 911
      DO 6 I=1,NOBL
      IA=0
      DO 7 J=1,NOBL
      IJ=INDEX(I,J)
    7 IA=IA+LEX1(IJ)+LEX2(IJ)+LEX3(IJ)+LEX4(IJ)
    6 F(I)=FLOAT(IA)/FLOAT(IR1+IR2+IR3+IR4)
C
C       FIRST COMPUTE THE TRUE MEAN
C
      TMEAN=0
      DO 10 I=1,NOBL
   10 TMEAN=TMEAN+F(I)*FLOAT(I)
      TMEAN=TMEAN+FLOAT(IMIN-1)
C
C       GET CUMULATIVE DISTRIBUTION FUNCTION IN F
C
      DO 8 I=2,NOBL
    8 F(I)=F(I)+F(I-1)
C
C       DETERMINE THE QUANTIZING FUNCTION
C
      CALL IEQP01(NOBL,NQUANT,F,IQ,IMIN)
C
C       NEXT COMPUTE THE QUANTIZED TRANSLATED MEAN FOR EACH ARRAY
C
      DO 2 I=1,NQUANT
      DO 2 J=1,NQUANT
      NSI=1
      IF(I.NE.1) NSI=IQ(I-1)+2-IMIN
      NEI=IQ(I)-IMIN +1
      NSJ=1
      IF(J.NE.1) NSJ=IQ(J-1)+2-IMIN
      NEJ=IQ(J) -IMIN  +1
      IF(NSI.GT.NEI) GO TO 2
      IF(NSJ.GT.NEJ) GO TO 2
      NADD1=0
      NADD2=0
      NADD3=0
      NADD4=0
      DO 9 NI=NSI,NEI
      DO 9 NJ=NSJ,NEJ
      IJ=INDEX(NI,NJ)
      NADD1=NADD1+LEX1(IJ)
      NADD2=NADD2+LEX2(IJ)
      NADD3=NADD3+LEX3(IJ)
    9 NADD4=NADD4+LEX4(IJ)
      AMEAN(1)=AMEAN(1)+FLOAT(NADD2* I)
      AMEAN(4)=AMEAN(4)+FLOAT(NADD3*I)
```

```
      AMEAN(2)=AMEAN(2)+FLOAT(NADD4* I)
      AMEAN(3)=AMEAN(3)+FLOAT(NADD1* I)
    2 CONTINUE
C
C      NOW NORMALIZE TO GET THE MEANS
C
      AMEAN(1)=AMEAN(1)*R2
      AMEAN(2)=AMEAN(2)*R4
      AMEAN(3)=AMEAN(3)*R1
      AMEAN(4)=AMEAN(4)*R3
C
C      NOW DO MOMENT CALCULATIONS
C
      DO 3 I=1,NQUANT
      DO 3 J=1,NQUANT
      NSI=1
      IF(I.NE.1) NSI=IQ(I-1)+2-IMIN
      NEI=IQ(I)   -IMIN   +1
      NSJ=1
      IF(J.NE.1) NSJ=IQ(J-1)+2 -IMIN
      NEJ=IQ(J)    -IMIN   +1
      IF(NSI.GT.NEI) GO TO 3
      IF(NSJ.GT.NEJ) GO TO 3
      NADD1=0
      NADD2=0
      NADD3=0
      NADD4=0
      DO 13 NI=NSI,NEI
      DO 13 NJ=NSJ,NEJ
      NINJ=INDEX(NI,NJ)
C
C      SUM UP THE ELEMENTS IN EACH LEX ARRAY
C
      NADD1=NADD1+LEX1(NINJ)
      NADD2=NADD2+LEX2(NINJ)
      NADD3=NADD3+LEX3(NINJ)
   13 NADD4=NADD4+LEX4(NINJ)
C
C      NORMALIZE
C
      RL1=FLOAT(NADD1)*R1
      RL2=FLOAT(NADD2)*R2
      RL3=FLOAT(NADD3)*R3
      RL4=FLOAT(NADD4)*R4
C
C      CALCULATE THE MOMENTS
C
      ANGMOM(1)=ANGMOM(1)+RL2**2
      ANGMOM(2)=ANGMOM(2)+RL4**2
      ANGMOM(3)=ANGMOM(3)+RL1**2
```

```
      ANGMOM(4)=ANGMOM(4)+RL3**2
      SGMASQ(1)=SGMASQ(1)+((FLOAT(I)-AMEAN(1))**2)*RL2
      SGMASQ(2)=SGMASQ(2)+((FLOAT(J)-AMEAN(2))**2)*RL4
      SGMASQ(3)=SGMASQ(3)+((FLOAT(I)-AMEAN(3))**2)*RL1
      SGMASQ(4)=SGMASQ(4)+((FLOAT(I)-AMEAN(4))**2)*RL3
      SGMAXY(1)=SGMAXY(1)+(FLOAT(I)-AMEAN(1))*(FLOAT(J)-AMEAN(1))*RL2
      SGMAXY(2)=SGMAXY(2)+(FLOAT(I)-AMEAN(2))*(FLOAT(J)-AMEAN(2))*RL4
      SGMAXY(3)=SGMAXY(3)+(FLOAT(I)-AMEAN(3))*(FLOAT(J)-AMEAN(3))*RL1
      SGMAXY(4)=SGMAXY(4)+(FLOAT(I)-AMEAN(4))*(FLOAT(J)-AMEAN(4))*RL3
      IVDMOM(1)=IVDMOM(1)+RL2/(1.+FLOAT((I-J)**2))
      IVDMOM(2)=IVDMOM(2)+RL4/(1.+FLOAT((I-J)**2))
      IVDMOM(3)=IVDMOM(3)+RL1/(1.+FLOAT((I-J)**2))
      IVDMOM(4)=IVDMOM(4)+RL3/(1.+FLOAT((I-J)**2))
      IF(RL2.LT.0.000001)GO TO 50
      ENTROP(1)=ENTROP(1)-RL2*ALOG(RL2)
   50 IF(RL4.LT.0.000001)GO TO 51
      ENTROP(2)=ENTROP(2)-RL4*ALOG(RL4)
   51 IF(RL1.LT.0.000001)GO TO 52
      ENTROP(3)=ENTROP(3)-RL1*ALOG(RL1)
   52 IF(RL3.LT.0.000001)GO TO 53
      ENTROP(4)=ENTROP(4)-RL3*ALOG(RL3)
   53 CONTINUE
C
C     SET UP THE SUM ARRAY
C
      K=IABS(I-J)+1
C
C     SET UP THE DIFFERENCE ARRAY
C
      KS=IABS(I+J)+1
      DIF1(K)=DIF1(K)+RL2
      DIF2(K)=DIF2(K)+RL4
      DIF3(K)=DIF3(K)+RL1
      DIF4(K)=DIF4(K)+RL3
      SUM1(KS)=SUM1(KS)+RL2
      SUM2(KS)=SUM2(KS)+RL4
      SUM3(KS)=SUM3(KS)+RL1
      SUM4(KS)=SUM4(KS)+RL3
    3 CONTINUE
      DO 4 I=1,4
    4 RATIO(I)=SGMAXY(I)/SGMASQ(I)
C
C     CALCULATE THE ENTROPY,AVERAGE,AND THE VARIANCE OF THE DIFFERENCE
C     ARRAY
C
      DO 31 K=1,NQUANT
      IF(DIF1(K).LT.0.000001) GO TO 54
      DIFENT(1)=DIFENT(1)-DIF1(K)*ALOG(DIF1(K))
   54 IF(DIF2(K).LT.0.000001) GO TO 55
      DIFENT(2)=DIFENT(2)-DIF2(K)*ALOG(DIF2(K))
```

```
55    IF(DIF3(K).LT.0.000001) GO TO 56
      DIFENT(3)=DIFENT(3)-DIF3(K)*ALOG(DIF3(K))
56    IF(DIF4(K).LT.0.000001) GO TO 57
      DIFENT(4)=DIFENT(4)-DIF4(K)*ALOG(DIF4(K))
57    CONTINUE
      G=FLOAT(K)
      DIFAVE(1)=DIFAVE(1)+(G*DIF1(K))
      DIFAVE(2)=DIFAVE(2)+(G*DIF2(K))
      DIFAVE(3)=DIFAVE(3)+(G*DIF3(K))
      DIFAVE(4)=DIFAVE(4)+(G*DIF4(K))
      DIFVAR(1)=DIFVAR(1)+(G*G)*DIF1(K)
      DIFVAR(2)=DIFVAR(2)+(G*G)*DIF2(K)
      DIFVAR(3)=DIFVAR(3)+(G*G)*DIF3(K)
31    DIFVAR(4)=DIFVAR(4)+(G*G)*DIF4(K)
      DO 32 KK=1,4
32    DIFVAR(KK)=DIFVAR(KK)-(DIFAVE(KK)*DIFAVE(KK))
C
C     CALCULATE THE ENTROPY,AVERAGE,AND THE VARIANCE OF THE SUM
C     ARRAY
C
      DO 33 K=1,NQUAN2
      IF(SUM1(K).LT.0.000001) GO TO 58
      SUMENT(1)=SUMENT(1)-SUM1(K)*ALOG(SUM1(K))
58    IF(SUM2(K).LT.0.000001) GO TO 59
      SUMENT(2)=SUMENT(2)-SUM2(K)*ALOG(SUM2(K))
59    IF(SUM3(K).LT.0.000001) GO TO 60
      SUMENT(3)=SUMENT(3)-SUM3(K)*ALOG(SUM3(K))
60    IF(SUM4(K).LT.0.000001) GO TO 61
      SUMENT(4)=SUMENT(4)-SUM4(K)*ALOG(SUM4(K))
61    CONTINUE
      G=FLOAT(K)
      SUMAVE(1)=SUMAVE(1)+(G*SUM1(K))
      SUMAVE(2)=SUMAVE(2)+(G*SUM2(K))
      SUMAVE(3)=SUMAVE(3)+(G*SUM3(K))
      SUMAVE(4)=SUMAVE(4)+(G*SUM4(K))
      SUMVAR(1)=SUMVAR(1)+(G*G)*SUM1(K)
      SUMVAR(2)=SUMVAR(2)+(G*G)*SUM2(K)
      SUMVAR(3)=SUMVAR(3)+(G*G)*SUM3(K)
33    SUMVAR(4)=SUMVAR(4)+(G*G)*SUM4(K)
      DO 34 KK=1,4
34    SUMVAR(KK)=SUMVAR(KK)-(SUMAVE(KK)*SUMAVE(KK))
      RETURN
C
C
C      SECTION II      IMOMTR FOR THE MERGED LEX ARRAY
C
C      GET THE PROBABILITY FUNCTION IN F FOR MERGE OPTION
C
911   DO 16 I=1,NOBL
      IA=0
```

```
      DO 17 J=1,NOBL
      IJ=INDEX(I,J)
   17 IA=IA+LEX1(IJ)
   16 F(I)=FLOAT(IA)/FLOAT(IR1)
C
C      FIRST COMPUTE THE TRUE MEAN
C
      TMEAN=0
      DO 90 I=1,NOBL
   90 TMEAN=TMEAN+F(I)*FLOAT(I)
      TMEAN=TMEAN+FLOAT(IMIN-1)
C
C      SET CUMULATIVE DISTRIBUTION FUNCTION IN F
C
      DO 91 I=2,NOBL
   91 F(I)=F(I)+F(I-1)
C
C      DETERMINE THE QUANTIZING FUNCTION
C
      CALL IEQPQ1(NOBL,NQUANT,F,IQ,IMIN)
C
C      NEXT COMPUTE THE QUANTIZED TRANSLATED MEAN
C
      DO 92 I=1,NQUANT
      DO 92 J=1,NQUANT
      NSI=1
      IF(I.NE.1) NSI=IQ(I-1)+2-IMIN
      NEI=IQ(I)-IMIN +1
      NSJ=1
      IF(J.NE.1) NSJ=IQ(J-1)+2-IMIN
      NEJ=IQ(J) -IMIN  +1
      IF(NSI.GT.NEI) GO TO 92
      IF(NSJ.GT.NEJ) GO TO 92
      NADD1=0
      DO 93 NI=NSI,NEI
      DO 93 NJ=NSJ,NEJ
      IJ=INDEX(NI,NJ)
   93 NADD1=NADD1+LEX1(IJ)
      AMEAN(3)=AMEAN(3)+FLOAT(NADD1* I)
   92 CONTINUE
C
C      NOW NORMALIZE TO GET THE MEANS
C
      AMEAN(3)=AMEAN(3)*R1
C
C      NOW DO MOMENT CALCULATIONS
C
      DO 95 I=1,NQUANT
      DO 95 J=1,NQUANT
      NSI=1
```

```
      IF(I.NE.1) NSI=IQ(I-1)+2-IMIN
      NEI=IQ(I)   -IMIN  +1
      NSJ=1
      IF(J.NE.1) NSJ=IQ(J-1)+2 -IMIN
      NEJ=IQ(J)    -IMIN  +1
      IF(NSI.GT.NEI) GO TO 95
      IF(NSJ.GT.NEJ) GO TO 95
      NADD1=0
      DO 96 NI=NSI,NEI
      DO 96 NJ=NSJ,NEJ
      NINJ=INDEX(NI,NJ)
C
C      SUM UP THE ARRAY
C
96    NADD1=NADD1+LEX1(NINJ)
C
C      NORMALIZE
C
      RL1=FLOAT(NADD1)*R1
C
C      COMPUTE MOMENTS
C
      ANGMOM(3)=ANGMOM(3)+RL1**2
      SGMASQ(3)=SGMASQ(3)+((FLOAT(I)-AMEAN(3))**2)*RL1
      SGMAXY(3)=SGMAXY(3)+(FLOAT(I)-AMEAN(3))*(FLOAT(J)-AMEAN(3))*RL1
      IVDMOM(3)=IVDMOM(3)+RL1/(1.+FLOAT((I-J)**2))
      IF(RL1.LT.0.000001)GO TO 533
      ENTROP(3)=ENTROP(3)-RL1*ALOG(RL1)
533   CONTINUE
C
C     SET UP THE SUM ARRAY
C
      K=IABS(I-J)+1
C
C     SET UP THE DIFFERENCE ARRAY
C
      KS=IABS(I+J)+1
      DIF3(K)=DIF3(K)+RL1
      SUM3(KS)=SUM3(KS)+RL1
95    CONTINUE
      RATIO(3)=SGMAXY(3)/SGMASQ(3)
C
C     CALCULATE THE ENTROPY,AVERAGE,AND THE VARIANCE OF THE DIFFERENCE
C     ARRAY
C
      DO 97 K=1,NQUANT
      IF(DIF3(K).LT.0.000001) GO TO 577
      DIFFNT(3)=DIFENT(3)-DIF3(K)*ALOG(DIF3(K))
577   CONTINUE
      G=FLOAT(K)
```

```
      DIFAVE(3)=DIFAVE(3)+(G*DIF3(K))                                IMOMTR5
   97 DIFVAR(3)=DIFVAR(3)+(G*G)*DIF3(K)                              IMOMTR5
      DIFVAR(3)=DIFVAR(3)-(DIFAVE(3)*DIFAVE(3))                      IMOMTR5
C                                                                    IMOMTR5
C     CALCULATE THE ENTROPY,AVERAGE,AND THE VARIANCE OF THE SUM      IMOMTR5
C     ARRAY                                                          IMOMTR5
C                                                                    IMOMTR5
      DO 98 K=1,NQUAN2                                               IMOMTR5
      IF(SUM3(K).LT.0.000001) GO TO 99                               IMOMTR5
      SUMENT(3)=SUMENT(3)-SUM3(K)*ALOG(SUM3(K))                      IMOMTR6
   99 CONTINUE                                                       IMOMTR6
      G=FLOAT(K)                                                     IMOMTR6
      SUMAVE(3)=SUMAVE(3)+(G*SUM3(K))                                IMOMTR6
   98 SUMVAR(3)=SUMVAR(3)+(G*G)*SUM3(K)                              IMOMTR6
      SUMVAR(3)=SUMVAR(3)-(SUMAVE(3)*SUMAVE(3))                      IMOMTR6
      RETURN                                                         IMOMTR6
      END                                                            IMOMTR6
```

```
CCOR         SUBROUTINE COR
C
C     WRITTEN BY SAM SHANMUGAM                                NOV 1972
C
C   ***********************************************************************
C
C                    OBJECTIVE.
C                    -------------
C
C                              THIS PROGRAM CALCULATES THREE MEASURES OF
C                              CORRELATION COR1,COR2,COR3, BETWEEN TWO
C                              DISCRETE RANDOM VARIABLES X AND Y WHOSE
C                              JOINT PROBABILITIES OF OCCURANCE ARE
C                              STORED IN THE ARRAY PXY
C
C                    ENTRY POINT.
C                    ------------
C
C                              CALL COR(PXY,N,IOPT,Q,COR1,COR2,COR3)
C
C                    INPUT ARGUMENTS.
C                    ----------------
C
C                        PXY      ARRAY OF JOINT PROBABILITIES
C                        N        SIZE OF THE ARRAY PXY
C                        IOPT     OPTION FLAG--IF IOPT=0 THEN COR1 AND
C                                    COR2 ONLY WILL BE COMPUTED.  IF
C                                    IOPT=1, THEN COR3 WILL BE COMPUTED
C                        Q        SCRATCH ARRAY OF SIZE N X N. THIS
C                                    ARRAY IS NEEDED ONLY IF IOPT IS
C                                 NON-ZERO.  IF IOPT IS ZERO THEN A
C                                 DUMMY VARIABLE MAY BE SUBSTITUTED FOR
C                                 THE ARGUMENT Q
C
C                    OUTPUT  ARGUMENTS.
C                    ------------------
C
C                        COR1     MAXIMAL CORRELATION MEASURE
C                        COR2     INFORMATION MEASURE OF CORRELATION
C                        COR3     SECOND TYPE OF MAXIMAL MEASURE
C
C                    BIBLIOGRAPHY.
C                    -------------
C
C                          ANNALS OF MATHEMATICAL STATISTICS, VOL.43, 1962,
C                             P.587,'MUTUAL INFORMATION AND MAXIMAL
C                             CORRELATION AS MEASURE OF DEPENDENCE' BY
C                             C.B. BELL.
C
C                    CAUTION.
```

```
C               ----------
C                                1. THE ARRAY PXY MUST HAVE A DIMENSION OF
C                                N X N IN THE CALLING PROGM. OR PXY SHOULD
C                                BE A ONE DIMENSIONAL VECTOR,CONTAINING TH
C                                JOINT PROBABILITIES IN A COLUMN BY COLUMN
C                                ARRANGEMENT.
C                                2. IF N IS LARGE ,THE COMPUTATIONS FOR
C                                COR3 WILL TAKE CONSIDERABLE TIME. HENCE
C                                THE USE OF THIS ROUTINE IS RESTRICTED TO
C                                N LESS THAN OR EQUAL TO 32
C
C               COMPUTATIONS.
C                -------------
C
C                                PX(I)= SUM PXY(I,J)
C                                        I
C                                PY(J)= SUM PXY(I,J)
C                                        J
C                                HXY  = SUM SUM LOG(PXY(I,J) *PXY(I,J)
C                                        I   J
C                                HXY1 = SUM SUM LOG((PX(I)*PY(J)) *PXY(I,J
C                                        I   J
C                                HXY2 =SUM SUM (LOG(PX(I)*PY(J)) *PX(I)*
C                                        I   J                    PY(J)
C                                HX   = SUM (LOG(PX(I))* PX(I)
C                                        I
C                                HY   = SUM (LOG(PY(J)) * PY(J)
C                                        J
C                                R=HXY2-HXY .  EMAX= MAX(HX,HY)
C                                COR1=(HXY-HXY1)/EMAX
C                                COR2=SQRT(1.0-EXP(-2.0*R))
C                                COR3- IS COMPUTED USING THE EIGEN VECTOR
C                                CORRESPONDING TO THE SECOND LARGEST
C                                    EIGEN VALUE OF Q*QTRANSPOSE, WHERE
C                                    Q(I,J)=PXY(I,J)/SQRT(PX(I)*PY(J))
C
C*************************************************************************
C
      SUBROUTINE COR(PXY,N,IOPT,Q,COR1,COR2,COR3)
C
      DIMENSION PXY(1),Q(1)
      DIMENSION PX(64),PY(64),E(64),V(128),B(64),C(64),D(64),F(64)
      DIMENSION IZERO(32)
C*************************************************************************
C
C                                 INITIALIZE PX,PY,HX,HY,HXY,HXY1,AND HXY2
C
      DO 80 I=1,N
      PY(I)=0.0
   80 PY(I)=0.0
```

```
      HX=0.0
      HY=0.0
      HXY=0.0
      HXY1=0.0
      HXY2=0.0
C
C                                  COMPUTE THE MARGINALS AND THEIR ENTROPY
C
      DO 82 I=1,N
      DO 81 J=1,N
      IJ=(J-1)*N+I
   81 PX(I)=PX(I)+PXY(IJ)
      IF(PX(I).LT.0.000001) GO TO 82
      HX=HX-(ALOG(PX(I)))*(PX(I))
   82 CONTINUE
      DO 84 J=1,N
      DO 83 I=1,N
      IJ=(J-1)*N+I
   83 PY(J)=PY(J)+PXY(IJ)
      IF(PY(J).LT.0.000001) GO TO 84
      HY=HY-ALOG(PY(J))*PY(J)
   84 CONTINUE
C
C                                  COMPUTE THE ENTROPY OF THE JOINT DISTR.
C
      DO 69 I=1,N
      DO 69 J=1,N
      IJ=(J-1)*N+I
      IF(PXY(IJ).LT.0.000001) GO TO 68
      HXY=HXY-(ALOG(PXY(IJ)))*(PXY(IJ))
   68 PXPY=PX(I)*PY(J)
      IF(PXPY.LT.0.000001) GO TO 69
      HXY1=HXY1-(ALOG(PXPY))*PXY(IJ)
      HXY2=HXY2-(ALOG(PXPY))*(PXPY)
   69 CONTINUE
C
C                                  COMPUTE COR1 AND COR2
C
      EMAX=HX
      IF(HX.LT.HY) EMAX=HY
      COR1=(HXY-HXY1)/EMAX
      R=HXY2-HXY
      COR2 = SQRT(1.0-EXP(-2.0*R))
C
C                                  IF COR3 NOT ASKED FOR RETURN
C
      IF(IOPT.EQ.0) RETURN
C                                  SCAN PXY AND DELETE ROWS OF ZEROS
C                                  AND COLUMNS OF ZEROS
      DO 599 INDEX=1,32
```

```
 599  IZERO(INDEX)=0
      NZERO=0
      DO 600 I=1,N
      II=I
      IF(PX(I).GT.0.000001) GO TO 601
      NZERO=NZERO+1
      IZERO(NZERO)=II
 601  CONTINUE
 600  CONTINUE
      IF(NZERO.EQ.0) GO TO 651
      JJ=0
      DO 650 J=1,N
      DO 650 I=1,N
      DO 640 KK=1,NZERO
      NDEX=IZERO(KK)
      IF((I.EQ.NDEX).OR.(J.EQ.NDEX)) GO TO 649
 640  CONTINUE
      JJ=JJ+1
      IJ=(J-1)*N+I
      PXY(JJ)=PXY(IJ)
 649  CONTINUE
 650  CONTINUE
 651  CONTINUE
C                              REMOVE ZERO ENTRIES IN THE MARGINALS
      JJ=0
      DO 661 I=1,N
      IF(PX(I).LT.0.000001) GO TO 662
       JJ=JJ+1
       PX(JJ)=PX(I)
 662  CONTINUE
 661  CONTINUE
      JJ=0
       DO 671 I=1,N
      IF(PY(I).LT.0.000001) GO TO 672
      JJ=JJ+1
       PY(JJ)=PY(I)
 672  CONTINUE
 671  CONTINUE
      NNNN=N
      N=N-NZERO
C
C
C                              NORMALIZE PXY AND STORE IN Q. SAVE PXY
C
C
      DO 58 I=1,N
      DO 58 J=1,N
      IJ=(J-1)*N+I
      CONS=SQRT(PY(J))
 58   Q(IJ)=PXY(IJ)/CONS
```

```
C
C                                       COMPUTE THE UPPER/DIAG ELEMENTS OF Q*QT
C                                       STORE IN Q
C
      DO 49 I=1,N
      DO 51 J=I,N
      B(J)=0.0
      DO 52 K=1,N
      IK=(K-1)*N+I
      JK=(K-1)*N+J
   52 B(J)=B(J)+Q(IK)*Q(JK)
   51 CONTINUE
      DO 50 J=1,N
      IJ=(J-1)*N+I
   50 Q(IJ)=B(J)
   49 CONTINUE
C
C                                       FILL IN THE BELOW DIAG ELEMENTS OF Q*QT
C
      DO 48 J=1,N
      DO 48 I=J,N
      IJ=(J-1)*N+I
      JI=(I-1)*N+J
   48 Q(IJ)=Q(JI)
C                                       FORM SQRT(PX) * Q*QT * SQRT(PY)
CC                                      STORE IN Q
      DO 91 I=1,N
      DO 91 J=1,N
      IJ=(J-1)*N+I
   91 Q(IJ)=Q(IJ)/(SQRT(PX(I)*PX(J)))
C
C                                       GET THE EIGEN VECTORS AND EIGEN VALUES
C                                       OF Q*QT
C
C
C                                       CALL THE SUBROUTINE TO GET THE EIGENVALS
C                                       GET A MAX OF 5 EIGENVALUES.  IF ALL FIVE
C                                       ARE NEAR UNITY,SET COR3=0.9999.  RETURN
C                                       IF ALL OF THEM (OTHER THAN THE FIRST ONE)
C                                       ARE LESS THAN 0.001 , SET COR3=0.0001 AND
C                                       RETURN. THE EIGEN VALUES ARE CALCULATED
C                                       WITH AN ACCURACY OF 0.0001.
C
C
C
      MAX=5
      IF(N.LT.MAX) MAX=N
      CR=0.0001
      EPS=0.00001
      CALL SFA02D(Q,N,N,CR,EPS,MAX,NE,E,V,B,C,D,F,IE)
C
```

```
      IF(NE.GT.1) GO TO 60
C
C                                       SET COR3=0.0001 AND RETURN
C
      COR3=0.01
      WRITE(6,200)
  200 FORMAT(1H1,10X,' ALL EIGEN VALUES OTHER THAN THE FIRST ARE LESS TH
     1AN 0.001. COR3 IS SET=0.0001)'/10X,'ABORT AA')
      N=NNNN
      RETURN
   60 CONTINUE
C     FIND THE EIGEN VALUE CLOSEST TO 1.0
      DO 63 I=1,NE
      DIF=ABS(E(I)-1.0)
      JJE=I
      IF(DIF.GT.0.00003) GO TO 64
   63 CONTINUE
C
C                                       IF PROGM IS AT THIS POINT, THEN THE FIRST
C                                       FIVE EIGEN VALUES ARE CLOSE TO 1.0. SET
C                                       COR3=0.9999 AND RETURN
C
C
      COR3=0.9999
      WRITE(6,202)
  202 FORMAT(1H1,10X,' THE FIRST 5 EIGEN VALUES ARE NEAR UNITY. '/10X,'
     1 COR3 IS SET =0.999.....ABORT BB')
      N=NNNN
      RETURN
C
C                                       FOUND A PROPER EIGEN VALUE
   64 IFOUND=JJE
      COR3=SORT(E(IFOUND))
      N=NNNN
      RETURN
      END
```

```
CIEQPQ1                          I-E-Q-P-Q-1
C                                                             SEPT 1971
C       WRITTEN BY   DENISH GOEL
C
C       DESCRIPTION OF PROGRAM.
C       THIS SUBROUTINE DETERMINES K LEVELS OF QUANTIZING FOR AN ARRAY
C       FOR WHICH THE CUMMULATIVE DISTRIBUTION FUNCTION OF ALL THE EL-
C       MENTS HAVE ALREADY BEEN OBTAINED.
C
C
C       ENTRY POINT.
C            CALL IEQPQ1(N,K,F,IQ,IMIN)
C
C       ARGUMENTS.
C
C               N                    NUMBER OF ITEMS TO BE QUANTIZED, THE
C                                       DIMENSION OF THE F ARRAY.
C               K                    THE NUMBER OF QUANTIZING LEVELS
C               F                    INPUT ARRAY OF CUMULATIVE DISTRIBUTION
C                                      FUNCTION.
C               IQ                   OUTPUT ARRAY OF QUANTIZING LEVELS.
C               IMIN                 THE LOWEST POSSIBLE LEVEL IN
C                                       THE INPUT DATA.
C
C
C
        SUBROUTINE IEQPQ1(N,K,F,IQ,IMIN)
C
        DIMENSION F(1),IQ(1)
        DIF=10.**6
C
C       OBTAIN THE FIRST QUANTIZING LEVEL.
C       GO THRU THE WHOLE ARRAY OF C.D.F,S
C
        DO 1 J=1,N
C
C       FIND PERCENTAGE OF DISTRIBUTION FOR FIRST QUANTIZING LEVEL AND
C       CHECK FOR THE NEAREST C.D.F.
C
        X=ABS(1./FLOAT(K)-F(J))
        IF(DIF.LE.X)GO TO 1
        DIF=X
        ISAVE=J
      1 CONTINUE
C
C       FIRST QUANTIZING LEVEL
C
        IQ(1)=ISAVE+IMIN-1
C
C       TO GO FOR NEXT LEVEL
C
```

```
      LFTOFF=ISAVE
      DO 2 I=2,K
C
C     DECIDE FOR OTHER QUANTIZING LEVELS IN THE SIMILAR WAY.
C
      DIF=10.**6
      DO 3 J=LFTOFF,N
C
C     THE PERCENTAGE OF DISTRIBUTION FOR NEXT QUANTIZING LEVEL WILL
C     BE DECIDED AMONG REST OF ELEMENTS.
C
      X=ABS(((1.-F(LFTOFF))/FLOAT(K-I+1))+F(LFTOFF)-F(J))
      IF(DIF.LE.X)GO TO 3
      DIF=X
      ISAVE=J
C
C     GET THE NEXT QUANTIZING LEVEL
C
    3 CONTINUE
      IQ(I)=ISAVE+IMIN-1
      LFTOFF=ISAVE
    2 CONTINUE
      RETURN
      END
```

```
CRITOWT                               R-I-T-O-W-T               JUNE 1973
C
C     WRITTEN BY RMH                                             SEPT 1971
C     VERSION 1 BY RJ BOSLEY FOR TAPE OUTPUT OPTION              NOV  1972
C     VERSION 2 BY RJ BOSLEY FOR MERGE                           JUNE 1973
C
C     DESCRIPTION OF PROGRAM.
C     THIS SUBROUTINE PUNCHES,PRINTS,OR WRITES TO FILE(IF) THE TEXTURE
C      DATA ACCORDING TO THE PNCH / MERGE OPTIONS
C
C     ENTRY POINT.
C       CALL RITOWT(LEX1,LEX2,LEX3,LEX4,G,IQ,MERGE,MERGE,IF,PICTUR)
C
C     ARGUMENTS.
C             LEX1-LEX4            ADDRESS INDEXS FOR THE LEX ARRAYS
C             G                    CDF FOR THE IMAGE DATA
C             IQ                   QUANTIZED OUTPUT OF IEQPQ1 OF  NQUANT
C                                     LEVELS
C             MERGE                OPTION TO MERGE THE FOUR LEX ARRAYS
C                                     INTO ONE ARRAY
C             IF                   FILE CODE FOR OUTPUT TAPE
C             PICTUR               OPTION TO PRINT A PICTURE OF THE
C                                     IMAGE,  USED TO VARY SPACING
C
C     INTERNAL PARAMETERS.
C             PNCH                 =TAPE FOR TAPE OUTPUT ON FILE IF
C             PNCH                 =Y  FOR PUNCH OUTPUT
C             PNCH                 =N  FOR PRINTER OUTPUT ONLY
C             NRED                 REDUCTION FACTOR FROM FPLXIT
C             NLAYER               THE POWER TO WHICH NRED IS RAISED
C             NET                  AMOUNT OF REDUCTION IN FPLXIT
C             N                    THE NUMBER OF ANGLES USED IN LEX
C                                     ARRAYS
C             M                    THE LEX ARRAY CONTAINING ALL THE
C                                     MERGED ARRAYS
C             M1                   IMAGE ROW INDEX
C             N1                   IMAGE COLUMN INDEX
C             KOUNT                CARD COUNTER
C             ANGMOM...CORMAX      FEATURES
C             NOBL                 NUMBER OF BRIGHTNESS LEVELS IN IMAGE
C             IMT                  LINE COUNTER FOR MERGE OPTION
C
      SUBROUTINE RITOWT(LEX1,LEX2,LEX3,LEX4,G,IQ,MERGE,IF,PICTUR)
C
      DIMENSION LEX1(1),LEX2(1),LEX3(1),LEX4(1),G(64),IQ(64),3(4)
      COMMON M1,N1,TYPE,F(14),IJD(9),ANGMOM(4),AMEAN(4),SGMASQ(4),
     1 SGMAXY(4),DIFMOM(4),RATIO(4),VIDMOM(4),TMEAN,LEAST1,NRED
      COMMON NLAYER,NSTART,NTIMES,ND,PNCH
      COMMON /EL/ENTROP(4),DIFENT(4),DIFAVE(4),DIFVAR(4),SUMENT(4),
     1SUMAVE(4),SUMVAR(4)
```

```
      COMMON /CORREL/CORINF(4),CORMUT(4),CORMAX(4)                     RITOWT51
      LOGICAL MERGE,PICTUR                                             RITOWT52
      DATA B(1),B(2),B(3),B(4)/0.,45.,90.,135./                        RITOWT53
      DATA TAPE/1HT/                                                   RITOWT54
      DATA Y/1HY/                                                      RITOWT55
      DATA KOUNT/0/,IMT/0/                                             RITOWT56
      NFT=NRED**NLAYER                                                 RITOWT57
      N=4                                                              RITOWT58
      KKJ=50                                                           RITOWT59
      M=3                                                              RITOWT60
C                                                                      RITOWT61
C     CHECK FOR THE MERGE OPTION                                       RITOWT62
C                                                                      RITOWT63
      IF(MERGE) GO TO 22                                               RITOWT64
C                                                                      RITOWT65
C      PUNCH OR NO PUNCH                                               RITOWT66
C                                                                      RITOWT67
      IF(PNCH.NE.Y) GO TO 400                                          RITOWT68
C                                                                      RITOWT69
C      PUNCH TEXTURE FEATURES FOR ALL FOUR ANGLES                      RITOWT70
C                                                                      RITOWT71
      WRITE(43,60) M1,N1,NFT                                           RITOWT72
  106 FORMAT(1X,2A5,12HIS COMPLETED)                                   RITOWT73
      KOUNT=KOUNT+1                                                    RITOWT74
      WRITE(43,600 )(ANGMOM(K),K=1,4),(ENTROP(K),K=1,4),KOUNT          RITOWT75
      KOUNT=KOUNT+1                                                    RITOWT76
      WRITE(43,600)  (RATIO(K),K=1,4),(SGMASQ(K),K=1,4),KOUNT          RITOWT77
      KOUNT=KOUNT+1                                                    RITOWT78
      WRITE(43,600)  (SGMAXY(K),K=1,4),(AMEAN(K),K=1,4),KOUNT          RITOWT79
      KOUNT=KOUNT+1                                                    RITOWT80
      WRITE(43,600)  (VIDMOM(K),K=1,4),(TMEAN,K=1,4),KOUNT             RITOWT81
      KOUNT=KOUNT+1                                                    RITOWT82
      WRITE(43,600)  (DIFENT(K),K=1,4),(DIFAVE(K),K=1,4),KOUNT         RITOWT83
      KOUNT=KOUNT+1                                                    RITOWT84
      WRITE(43,600)  (DIFVAR(K),K=1,4),(SUMENT(K),K=1,4),KOUNT         RITOWT85
      KOUNT=KOUNT+1                                                    RITOWT86
      WRITE(43,600)  (SUMAVE(K),K=1,4),(SUMVAR(K),K=1,4),KOUNT         RITOWT87
      KOUNT=KOUNT+1                                                    RITOWT88
      WRITE(43,600)(CORINF(K),K=1,4),(CORMUT(K),K=1,4),KOUNT           RITOWT89
      KOUNT=KOUNT+1                                                    RITOWT9
      WRITE(43,601)  (CORMAX(K),K=1,4),KOUNT                           RITOWT9
  600 FORMAT(1X,8F9.5,I7)                                              RITOWT9
  601 FORMAT(1X,4F9.4,38X,I5)                                          RITOWT9
  661 FORMAT(1X,4F9.5,36X,I7)                                          RITOWT9
  400 CONTINUE                                                         RITOWT9
C                                                                      RITOWT9
C      WRITE TEXTURE FEATURES TO TAPE FILE 'IF'                        RITOWT9
C                                                                      RITOWT9
      IF(PNCH.NE.TAPE) GO TO 500                                       RITOWT9
      WRITE(IF)M1,N1,NFT,(ANGMOM(K),K=1,N),(ENTROP(K),K=1,N),(RATIO(K), RITOWT0
```

REPRODUCIBILITY OF THE ORIGINAL PAGE IS POOR

```
     1K=1,N),(SGMASQ(K),K=1,N),(SGMAXY(K),K=1,N),(AMEAN(K),K=1,N),
     1(VIDMOM(K),K=1,N),(TMEAN,K=1,N),(DIFENT(K),K=1,N),
     1(DIFAVE(K),K=1,N),(DIFVAR(K),K=1,N),(SUMENT(K),K=1,N),
     1(SUMAVE(K),K=1,N),(SUMVAR(K),K=1,N),(CORINF(K),K=1,N),(CORMUT(K),
     1K=1,N),(CORMAX(K),K=1,N)
  500 CONTINUE
C
C     PRINT TEXTURE FEATURES FOR EACH ANGLE AND TITLE
C
      WRITE(6,60) M1,N1,NFT
   60 FORMAT(/' THE SCENE (',I2,',',I2,') HAS BEEN REDUCED BY ',I5)
      WRITE(6,303)
303   FORMAT(6H ANGLE ,9H  ANGMOM  ,8H ENTROP ,8H  RATIO ,8H SGMASQ  ,
     18H SGMAXY ,8H IVDMOM ,8H DIFENT ,8H DIFAVE ,8H DIFVAR ,8H SUMENT ,
     18H SUMAVE ,8H SUMVAR ,8H CORINF ,7HCORMUT ,8H CORMAX )
      WRITE(6,300)(B(K),ANGMOM(K),ENTROP(K),RATIO(K),SGMASQ(K),SGMAXY(K)
     1,VIDMOM(K),DIFENT(K),DIFAVE(K),DIFVAR(K),SUMENT(K),SUMAVE(K),
     2SUMVAR(K),CORINF(K),CORMUT(K),CORMAX(K),K=1,N)
  300 FORMAT(1X,F5.1,15F8.4)
      WRITE(6,600) TMEAN
  100 CONTINUE
C
C     IF NEITHER PNCH NOR TAPE, PRINT LEX ARRAYS
C
      IF((PNCH.EQ.Y).OR.(PNCH.EQ.TAPE)) RETURN
      WRITE(6,30) G
      WRITE(6,31) IQ
   30 FORMAT(2H F/(1X,16F7.3))
   31 FORMAT(3H IQ/(1X,16I7))
      NOBL=IDD(1)-IDD(2)+1
C
C     IF MERGE, JUST DO LEX1 AND RETURN
C
      IF(MERGE) GO TO 54
      WRITE(6,566)
  566 FORMAT(//10X,9H0 DEGREES)
      DO 50 I=1,NOBL
      NS=I*(I-1)/2+1
      NE=(I+1)*I/2
C
C     PRINT LEX2 FOR 0 DEGREES
C
   50 WRITE(6,700) (LEX2(J),J=NS,NE)
  700 FORMAT(1X,26I5)
      WRITE(6,567)
  567 FORMAT(//10X,10H45 DEGREES)
      DO 51 I=1,NOBL
      NS=I*(I-1)/2+1
      NE=(I+1)*I/2
C
```

REPRODUCIBILITY OF THE ORIGINAL PAGE IS POOR

```
C
C      PRINT LEX4 FOR 45 DEGREES
C
   51 WRITE(6,700) (LEX4(J),J=NS,NE)
      WRITE(6,99)
   99 FORMAT(1H1)
54    WRITE(6,568)
  568 FORMAT(//10X,10H90 DEGREES)
      DO 52 I=1,NOBL
      NS=I*(I-1)/2+1
      NE=(I+1)*I/2
C
C      PRINT LEX1 FOR 90 DEGREES
C
   52 WRITE(6,700) (LEX1(J),J=NS,NE)
      IF(MERGE) RETURN
      WRITE(6,569)
  569 FORMAT(//10X,11H135 DEGREES)
      DO 53 I=1,NOBL
      NS=I*(I-1)/2+1
      NE=(I+1)*I/2
C
C      PRINT LEX3 FOR 135 DEGREES
C
   53 WRITE(6,700) (LEX3(J),J=NS,NE)
      RETURN
C
C      RITOWT FOR THE MERGE OPTION
C
C      CHECK TO SEE IF A PICTURE HAS BEEN PRINTED
C
22    IF(PICTUR) GO TO 23
C
C      INCREMENT PAGE COUNT
C
      IMT=IMT+1
C
C      IF PAGE IS FULL GO TO TOP OF NEXT PAGE AND WRITE TITLE
C
      IF(IMT.LE.1) WRITE(6,662)
      IF(IMT.GE.14) IMT=0
662   FORMAT(1H1,40X,'ERTS TEXTURE ANALYSIS'//
     1           1X,'ANGLE  ANGMOM  ENTROP   RATIO  SGMASQ  SGMAXY  IVDMO
     1M  DIFENT  DIFAVE  DIFVAR  SUMENT  SUMAVE  SUMVAR  CORINF  CORMUT
     1 CORMAX '/)
C
C      CHECK FOR PUNCH
C
23    IF(PNCH.NE.Y) GO TO 40
C
C      PUNCH THE MERGED TEXTURE FEATURES
```

```
C
      WRITE(43,663) M1,N1,NFT,ANGMOM(M),ENTROP(M),RATIO(M),SGMASQ(M),
     1 SGMAXY(M),AMEAN(M),VIDMOM(M),KOUNT
      KOUNT=KOUNT+1
      WRITE(43,664)    TMEAN,DIFENT(M),DIFAVE(M),DIFVAR(M),SUMENT(M),
     1 SUMAVE(M),SUMVAR(M),KOUNT
      KOUNT=KOUNT+1
      WRITE(43,665) CORINF(M),CORMUT(M),CORMAX(M),KOUNT
      KOUNT=KOUNT+1
663   FORMAT(I2,1X,I2,I2,1X,7F9.5,I9)
664   FORMAT(8X,7F9.5,I9)
665   FORMAT(8X,3F9.5,36X,I9)
40    CONTINUE
C      CHECK FOR TAPE OUTPUT
      IF(PNCH.NE.TAPE) GO TO 41
C
C       WRITE OUT ONTO FILE 'IF' THE MERGED TEXTURE FEATURES
C
      WRITE(IF) M1,N1,NFT,ANGMOM(M),ENTROP(M),RATIO(M),SGMASQ(M),
     1 SGMAXY(M),AMEAN(M),VIDMOM(M),TMEAN,DIFENT(M),DIFAVE(M),
     2 DIFVAR(M),SUMENT(M),SUMAVE(M),SUMVAR(M),CORINF(M),CORMUT(M),
     3 CORMAX(M)
C
C       IN ANY CASE,PRINT THE MERGED TEXTURE FEATURES
C
41    WRITE(6,60) M1,N1,NFT
      WRITE(6,666)          ANGMOM(M),ENTROP(M),RATIO(M),SGMASQ(M),SGMAXY(M)
     1,VIDMOM(M),DIFENT(M),DIFAVE(M),DIFVAR(M),SUMENT(M),SUMAVE(M),
     2SUMVAR(M),CORINF(M),CORMUT(M),CORMAX(M)
666   FORMAT(1X,'MERGE',15F8.4)
      WRITE(6,667)  TMEAN
667   FORMAT(1X,F9.5/)
C
C       NOW GO PRINT OUT THE MERGED TEX1 ARRAY AND RETURN
C
      GO TO 100
      END
```

IV.2-c Cross-Band Texture Analysis Program Listings

- SPECTR
  - GETIM / GETIT
    - ERTS (see IV.2-a)
  - DIFFER
  - COVAR
    - MNCVIN / MNCV
  - CORREL

```
CSPECTR              CROSS-BAND TEXTURE ANALYSIS
C                                                          JAN 1974
C     WRITTEN BY RJ BOSLEY
C
C     DESCRIPTION OF PROGRAM
C        THIS PROGRAM IS THE MAINLINE OF PROGRAMS WHICH OBTAIN A
C     NUMLIN X NUMPPL X NDIM SUBIMAGE FROM THE ERTS INPUT TAPE AND FOR
C     EACH SUBIMAGE CALCULATES THE COVARIANCE MATRIX AND THE
C     CORRELATION MATRIX.  THIS MATRIX IS WRITTEN TO FILE IFILE  FOR
C     FURTHER ANALYSIS.
C        THE ERTS TAPE IS PROCESSED IN HORIZONTAL ROWS OF SUBIMAGES.
C     SUBIMAGES MAY OVERLAP HORIZONTALLY AND VERTICALLY, AND THE
C     DISTANCE BETWEEN NEIGHBORING CELLS USED IN THE DIFFERENCE ARRAY
C     IS VARIABLE.
C
C     NOTE---
C     ERTS INPUT TAPE MUST BE ON FILE ES.
C     A RANDOM DISC FILE IS REQUIRED ON FILE 11--- $  FILE  11,A11R,30R
C
C     INTERNAL PARAMETERS
C              NDIM                   NUMBER OF GREY TONE N-TUPLE COMP.
C              NUMLIN                 NUMBER OF LINES IN A SUBIMAGE
C              NUMPPL                 NUMBER OF COLUMNS IN A SUBIMAGE
C              NROW                   ROW COORD FOR THE SUBIMAGE
C              NCOL                   COLUMN COORD FOR THE SUBIMAGE
C              IDIST                  DISTANCE BETWEEN NEIGHBORING RES.
C                                     CELLS FOR THE DIFFERENCE IMAGE
C              IRSTRT                 STARTING ROW FOR THIS RUN
C              IRSTOP                 STOPING ROW FOR THIS RUN
C                                      SET=0, THE STRIP WILL BE PROCESSED
C                                      TO ITS END OF FILE
C              LAPHOR                 HORIZONTAL OVERLAP OF SUBIMAGES
C              LAPVER                 VERTICAL OVERLAP OF SUBIMAGES
C              NHOR                   NUMBER OF OVERLAPING HORIZONTAL
C                                      SUBIMAGES IN A ROW
C              IPEND                  ENDING POINT FOR ONE LINE FROM ERTS
C              LASTIM                 FINAL ROW OF SUBIMAGES
C              COR,COV                CORRELATION,COVARIANCE MATRICES
C              TTL                    TITLE FOR THE MATRIX
C              TITLE                  TITLE FOR THE MATRIX
C              FMT                    FORMAT FOR PRINTING OUT MATRIX TERMS
C              OPT                    TRUE TO PRINT OUT COVARIANCE MATRIX
C              PNCH                   DETERMINES OUTPUT FILE FOR FEATURES
C                                      TRUE FOR FILE 43,PUNCHED CARDS
C                                      FALSE FOR FILE 01,TAPE OR DISC
C              IF                     OUTPUT FILE FOR FEATURES
C              LABEL                  ROW AND COL NAMES FOR MATRIX PRINTOUT
C              DET                    DETERMINANT OF THE CORRELATION MATRIX
C              ENTROP                 ENTROPY MEASURE
C              KT                     CARD COUNTER
C
C
```

```
C      INPUT ARGUMENTS.
C               IMAGE,X              ARRAY CONTAINING SUBIMAGE
C               ILINE                ARRAY FOR READING ERTS TAPE
C               IYDIM                ROW DIMENSION OF IMAGE
C               IXDIM                COL DIMENSION OF IMAGE
C               NDIN                 EQUAL TO NDIM,NO OF BANDS
C
C      ENTRY POINT.
C               CALL SPECTR(IMAGE,X,ILINF,IXDIM,IYDIM,NDIN)
C
C         EXAMPLE OF DRIVER.
C               DIMENSION IMAGE(16,17,4),X(16,17,4),ILINE(330))
C               EQUIVALENCE(IMAGE(1,1,1),X(1,1,1),ILINE(130))
C               IXDIM=17
C               IYDIM=16
C               NDIM=4
C               CALL SPECTRE(IMAGE,X,ILINE,IXDIM,IYDIM,NDIM)
C               STOP
C               END
C
C        THIS PROGRAM WILL SET UP THE TEXTUR RUN FOR 16 X 16 SUBIMAGES
C      OVER ALL FOUR BANDS, WITH IDIST=1.  ***NOTE***IXDIM MUST INCLUDE
C      NUMPPL PLUS IDIST, AND ARRAY ILINE MUST HAVE AT LEAST NUMPPL*NDIM
C      POINTS OUTSIDE OF ANY OTHER ARRAY.  THESE POINTS FORM ARRAY XLINE
C      WHICH IS USED IN COVAR TO SEND ONE LINE OF DATA TO MNCV.
C
C      SUBPROGRAMS REQUIRED.
C        DRIVER
C          SPECTR
C            GETIM
C            SETDIM
C            GETIT
C              ERTS (WITH EREWND)
C            DIFFER
C            COVAR
C              MNCVIN
C              MNCV
C            CORREL
C            SFA07F
C            HEMDET
C
C
       SUBROUTINE SPECTR(IMAGE,X,ILINE,IXDIN,IYDIN,NDIN)
C
       DIMENSION IMAGE(IYDIN,IXDIN,NDIN),ILINE(1),TITLE(14),AR(5),
      1     LABEL(8),COR(8,8),COV(8,8),X(IYDIN,IXDIN,NDIN)
       CHARACTER ROW*12,COL*12,TTL*6(14)
       EQUIVALENCE (ROW,TTL(13)),(COL,TTL(11))
       LOGICAL OPT,PNCH
       NAMELIST /PARAM/ NDIM,NUMLIN,NUMPPL,FMT,TITLE,OPT,IDIST,IRSTRT,
      1                 IRSTOP,LAPHOR,LAPVER,PNCH
       DATA TTL(1)/'COVARIANCE OVER SUBIMAGE  --- '/,TTL(6)/'CORRELATION
```

```
     1OVER SUBIMAGE --- '/                                              SPECTR0
      DATA LABEL(1)/'BAND 2 2X2  BAND 3 3X3   2X3  BAND 4 4X4   2X4  '/ SPECTR0
      DATA FORMAT/'E11.4'/,TITLE(1)/0.0/,FM/'F9.5'/                     SPECTR0
C                                                                       SPECTR0
C                                                                       SPECTR0
C     ****** SECTION I --- SET UP PARAMETERS FOR PROCESSING ******      SPECTR1
C                                                                       SPECTR1
      OPT=.FALSE.                                                       SPECTR1
      PNCH=.TRUE.                                                       SPECTR1
      IDIST=1                                                           SPECTR1
      IRSTRT=1                                                          SPECTR1
      IRSTOP=0                                                          SPECTR1
      LAPHOR=0                                                          SPECTR1
      LAPVER=0                                                          SPECTR1
      IXDIM=IXDIN                                                       SPECTR2
      IYDIM=IYDIN                                                       SPECTR2
      NDIM=NDIN                                                         SPECTR2
      MDIN=NDIN                                                         SPECTR2
      NUMLIN=IYDIN                                                      SPECTR2
      NUMPPL=NUMLIN                                                     SPECTR2
      FMT=FORMAT                                                        SPECTR2
      KT=0                                                              SPECTR2
C                                 READ IN PARAMETERS                    SPECTR2
      READ(5,PARAM)                                                     SPECTR2
C                                 WRITE OUT PARAMETERS                  SPECTR3
      WRITE(6,1)                                                        SPECTR3
    1 FORMAT(20X,'SPECTRAL-TEXTURAL ANALYSIS PROGRAM')                  SPECTR3
      IF(IRSTOP.EQ.0)  IRSTOP=999999                                    SPECTR3
      WRITE(6,2)NUMPPL,NUMLIN,NDIM,FMT,IDIST,LAPHOR,LAPVER,IRSTRT,IRSTOP SPECTR3
    2 FORMAT(1X,'NUMBER OF COLUMNS IN SUBIMAGE IS ',I3/                 SPECTR3
     1        ' NUMBER OF LINES IN SUBIMAGE IS ',I3/                    SPECTR3
     1        ' NUMBER OF BANDS IN SUBIMAGE IS ',I3/                    SPECTR3
     1        ' FORMAT USED TO OUTPUT MATRICES IS ',A5/                 SPECTR3
     1        ' DISTANCE BETWEEN CELLS FOR DIFFERENCE IMAGE IS ',I2/    SPECTR3
     1        ' HORIZONTAL SUBIMAGE OVERLAP IS',I3,' POINTS'/           SPECTR4
     1        ' VERTICAL   SUBIMAGE OVERLAP IS',I3,' POINTS'/           SPECTR4
     1        ' STARTING ROW  IN STRIP IS ',I6/                         SPECTR4
     1        ' FINAL ROW IN STRIP IS ',I6)                             SPECTR4
C                                 CHECK PARAMETERS                      SPECTR4
      IF(.NOT.OPT)  WRITE(6,11)                                         SPECTR4
   11 FORMAT(' ONLY THE CORRELATION MATRICES WILL BE LISTED')           SPECTR4
      IF(NUMPPL.LE.IXDIM)  GO TO 5                                      SPECTR4
      WRITE(6,4)                                                        SPECTR4
    4 FORMAT(' ERROR---NUMBER OF POINTS PER LINE EXCEEDS ',I4)          SPECTR4
      STOP                                                              SPECTR5
    5 IF(NDIM.LE.NDIN)  GO TO 7                                         SPECTR5
      WRITE(6,6) NDIN                                                   SPECTR5
    6 FORMAT(' ERROR---NUMBER OF BANDS,NDIM, MUST NOT EXCEED ',I3)      SPECTR5
      STOP                                                              SPECTR5
    7 K9=IXDIM-NUMPPL                                                   SPECTR5
      IF(IDIST.LE.K9)  GO TO 12                                         SPECTR5
      WRITE(6,32)  K9
```

```
32    FORMAT(' FRROR---IDIST MUST NOT EXCEED ',I4)                            SPECTR5
      STOP                                                                    SPECTR5
12    IF(LAPHOR.LT.NUMPPL)  GO TO 15                                          SPECTR5
13    WRITE(6,14)                                                             SPECTR6
      STOP                                                                    SPECTR6
14    FORMAT(' FATAL ERROR---OVERLAP EXCEEDS SIZE OF SUBIMAGE')               SPECTR6
15    IF(LAPVER.GE.NUMLIN)  GO TO 13                                          SPECTR6
      IF(NUMLIN.LE.IYDIM)  GO TO 17                                           SPECTR6
      WRITE(6,16) IYDIM                                                       SPECTR6
16    FORMAT(' NUMLIN EXCEEDS',I3,' LINES---EXECUTION TERMINATED')            SPECTR6
      STOP                                                                    SPECTR6
C                                                                             SPECTR6
C                                                                             SPECTR6
C     ****** SECTION II --- PROCESS THE SUBIMAGES ******                      SPECTR7
C                                                                             SPECTR7
C                                   INITIALIZE THE GETIM PROGRAM              SPECTR7
17    CALL GETIM(ILINE,IDIST,NDIM,IRSTRT,IRSTOP,NUMLIN,NUMPPL,LAPVER,         SPECTR7
     1    LAPHOR,NHOR,INCR,IPEND)                                             SPECTR7
C                                   INITIALIZE HEMP PACKAGE PGMS              SPECTR7
      CALL SETDIM(COV,MDIN,MDIN)                                              SPECTR7
C                                   SET THE OUTPUT FILE FOR TAPE OR PUNCH     SPECTR7
      KDIM=NDIM-1                                                             SPECTR7
      IF=01                                                                   SPECTR7
      IF(PNCH)  IF=43                                                         SPECTR8
C                                   FIND THE ROW COORD AND THE LAST ROW       SPECTR8
      NROW=IRSTRT/(NUMLIN-LAPVER)                                             SPECTR8
      LASTIM=(IRSTOP-IRSTRT+1)/(NUMLIN-LAPVER)+1                              SPECTR8
      KPE=NUMPPL+IDIST                                                        SPECTR8
      DO 100 M1=1,LASTIM                                                      SPECTR8
      NROW=NROW+1                                                             SPECTR8
C                                   SET UP FOR TITLE OF MATRICES              SPECTR8
C                                   PUT THE COORDINATE INTO HOLLERITH         SPECTR8
C                                   LITERAL FOR MATRIX PRINTING               SPECTR8
      ENCODE (ROW,8) NROW                                                     SPECTR9
8     FORMAT(I6,6X)                                                           SPECTR9
C                                   GET A ROW OF HORIZ OVERLAPPING            SPECTR9
C                                   SUBIMAGES FROM ERTS AND PUT THEM ON       SPECTR9
C                                   DISC FILES 11,12,13,14 FOR 4 BANDS        SPECTR9
      CALL GETIT                                                              SPECTR9
C                                   PULL OFF EACH SUBIMAGE, GOING ACROSS      SPECTR9
C                                   THE ENTIRE ROW                            SPECTR9
      KS=-INCR                                                                SPECTR9
      DO 90 NCOL=1,NHOR                                                       SPECTR9
C                                   SET START AND END PTS FOR SUBIMAGE        SPECTR0
      KS=KS+INCR                                                              SPECTR0
C                                   GO THRU NUMLIN LINES                      SPECTR0
      DO 50 LLN=1,NUMLIN                                                      SPECTR0
      JP=IPEND*(LLN-1)                                                        SPECTR0
C                                   READ A LINE OF THE SUBIMAGE               SPECTR0
      DO 45 KOL=1,KPE                                                         SPECTR0
      IP=JP+KS+KOL                                                            SPECTR0
C                                   READ IN THE RESOLUTION CELL               SPECTR0
```

```
      READ(11'IP)  (IMAGE(LLN,KOL,IB),IB=1,NDIM)                          SPECTR0
 45   CONTINUE                                                            SPECTR1
 50   CONTINUE                                                            SPECTR1
C                                GET THE DIFFERENCE IMAGE                 SPECTR1
      CALL DIFFER(IMAGE,X,IXDIM,IYDIM,NDIM,IDIST,NUMPPL,NUMLIN)           SPECTR1
C                                PUT THE COORDINATE INTO HOLLERITH        SPECTR1
C                                LITERAL FOR MATRIX PRINTING              SPECTR1
      ENCODE (COL,8) NCOL                                                 SPECTR1
C                                GET THE COVARIANCE MATRIX FOR THIS       SPECTR1
C                                SUBIMAGE                                 SPECTR1
      CALL COVAR(ILINE,NDIM,NUMPPL,X,IXDIM,IYDIM,NUMLIN,MDIN,COV)         SPECTR1
C                                WRITE OUT THE COVARIANCE MATRIX          SPECTR2
C                                WHOSE COORDINATES ARE (NCOL,NROW)        SPECTR2
      IF(OPT)  CALL SFA07F(COV,NDIM,NDIM,NDIM,2,1,2,FMT,TITLE,TTL(1),     SPECTR2
     1     TTL(11),TTL(13),LABEL,LABEL)                                   SPECTR2
C                                GET THE CORRELATION MATRIX               SPECTR2
      CALL CORREL(COV,NDIM,COR)                                           SPECTR2
C                                WRITE OUT THE CORREL MATRIX              SPECTR2
      CALL SFA07F(COR,NDIM,NDIM,NDIM,2,1,2,FM,TITLE,TTL(6),TTL(11),       SPECTR2
     1     TTL(13),LABEL,LABEL)                                           SPECTR2
C                                COPY COR TO COV SINCE HEMDET DESTROYS    SPECTR3
C                                ITS INPUT MATRIX                         SPECTR3
      DO 65 I5=1,NDIM                                                     SPECTR3
      DO 65 J5=1,NDIM                                                     SPECTR3
      COV(I5,J5)=COR(I5,J5)                                               SPECTR3
 65   CONTINUE                                                            SPECTR3
C                                GET THE DETERMINANT FOR THE              SPECTR3
C                                CORRELATION MATRIX                       SPECTR3
      CALL HEMDET(COV,NDIM,DET)                                           SPECTR3
      ENTROP=(-1.)*ALOG(DET)                                              SPECTR3
      WRITE(6,91)   ENTROP                                                SPECTR4
C                                SAVE THE CORRELATION COEFFICIENTS AS     SPECTR4
C                                VECTORS FOR PRINCIPLE COMPONENTS         SPECTR4
      KT=KT+1                                                             SPECTR4
      WRITE(IF,200) NROW,NCOL,ENTROP,((COR(J,K+1),K=J,KDIM),J=1,KDIM),KT  SPECTR4
 90   CONTINUE                                                            SPECTR4
 91   FORMAT(' ENTROPY MEASURE IS ',F15.5)                                SPECTR4
 100  CONTINUE                                                            SPECTR4
 200  FORMAT(2I3,1X,E13.6,6F10.6/8F10.6/8F10.6/6F10.6,15X,I5)             SPECTR4
      STOP                                                                SPECTR4
      END
7 MEMORY EXPANDED. USE $LIMITS OR CORE= OPTION FOR NEXT RUN
```

```
CGETIM                 GET THE IMAGE FROM ERTS
C
C     WRITTEN BY RJ BOSLEY                                       JAN 1974
C
C     DESCRIPTION OF PROGRAM
C        THIS PROGRAM IS INITIALIZED BY CALLING GETIM.  ALL FOLLOWING
C     CALLS MUST BE TO GETIT WHICH GETS THE SUBIMAGE FROM THE ERTS
C     DATA TAPE ON FILE CODE 'ES', AND OUTPUTS A ROW OF OVERLAPPING
C     SUBIMAGES  TO DISC FILE 11.
C
C     INTERNAL PARAMETERS.
C                LENGTH                LENGTH OF ONE ERTS LINE OF DATA
C                NOSK                  NUMBER OF LINES TO SKIP AFTER
C                                        REWINDING ERTS INPUT TAPE
C                LNSTRT                STARTING LINE IN ERTS DATA FILE
C                LNINCR                VERTICAL INCREMENT FOR THE NEXT ROW
C                F                     ARRAY CONTIANING CROSS BAND N-TUPLE
C                                        COMPONENTS
C
C
C     ENTRY POINT.
C        CALL GETIM(ILINE,IDIST,NDIM,IRSTRT,IRSTOP,NUMLIN,NUMPPL,
C                       LAPVER,LAPHOR,NHOR,INCR,IPEND)
C
C     INPUT ARGUMENTS.
C                ILINE                 ARRAY WHERE ERTS LINE IS READ INTO
C                IDIST                 DISTANCE BETWEEN NEIGHBORING CELLS
C                                        TO FORM DIFFERENCE ARRAY
C                NDIM                  NUMBER OF COMPONENTS OF EACH RES CELL
C                IRSTRT                STARTING LINE OF ERTS DATA FILE
C                IRSTOP                ENDING LINE OF ERTS DATA FILE
C                NUMLIN                NUMBER OF LINES IN SUBIMAGE
C                NUMPPL                NUMBER OF COLUMNS IN SUBIMAGE
C                LAPVER                NUMBER OF LINES SUBIMAGES OVERLAP
C                LAP HOR               NUMBER OF COLUMNS SUBIMAGES OVERLAP
C
C     OUTPUT ARGUMENTS.
C                NHOR                  NUMBER OF HORIZONTAL OVERLAPPING
C                                        SUBIMAGES PER ROW
C                INCR                  HORIZONTAL INCREMENT FOR FIRST COL OF
C                                        THE NEXT SUBIMAGE IN THE ROW
C                IPEND                 LAST POINT IN ROW
C
      SUBROUTINE GETIM(ILINE,IDIST,NDIM,IRSTRT,IRSTOP,NUMLIN,NUMPPL,
     1                 LAPVER,LAPHOR,NHOR,INCR,IPEND)
C
      DIMENSION ILINE(1),F(8)
C                                      INITIALIZE ERTS INPUT TAPE
      CALL EINIT(LENGTH)
      WRITE(6,4) LENGTH
    4 FORMAT(' LENGTH OF ONE ERTS LINE IS ',I6)
C                                      CHECK THE NO. OF WORDS PER RECORD
      IF(LENGTH.LE.3300)  GO TO 5
```

```
      WRITE(6,1)  LENGTH
1     FORMAT(' LENGTH OF ERTS LINE EXCEEDS 3300--LENGTH=',I6)
      STOP
C                                   REWIND THE INPUT TAPE
5     CALL EREWND
C                                   FIND THE STARTING LINE OF FIRST ROW
C                                   AND INCREMENT FOR SUCCESSIVE ROWS
      IRSTR=IRSTRT-1
      LNSTRT=IRSTR
      LNINCR=NUMLIN-LAPVER
      ASSIGN 951 TO JP
      ASSIGN 971 TO KP
C                                   IF THERE IS NO OVERLAP, DO NOT REWIND
      IF(LAPVER.NE.0)  GO TO 940
      ASSIGN 950 TO JP
      ASSIGN 970 TO KP
C                                   FIND ENDING HORIZONTAL POINT IN EACH
C                                   LINE OF THE ROW, AND THE NUMBER OF
C                                   OVERLAPPING SUBIMAGES OF NUMPPL POINTS
C                                   THAT WILL FIT ACROSS THE ROW
940   LEN= (LENGTH-32)/4
      NHOR=0
      IPEND=8
      INCR=NUMPPL-LAPHOR
C                                   INCREMENT THE ENDING POINT
1991  IPEND=IPEND+INCR
      NHOR=NHOR+1
      K=IPEND+NUMPPL+IDIST
C                                   TILL NO MORE SUBIMAGES WILL FIT
      IF(K.LE.LEN)  GO TO 1991
      IPEND=IPEND-INCR+NUMPPL+IDIST
C                                   SKIP THE FIRST 8 POINTS IN EACH LINE
      IS=32
C                                   FIND THE LAST POINT IN EACH ERTS LINE
      IE=IPEND*4+32-1
      WRITE(6,955) LEN,IPEND,NHOR
955   FORMAT(' TOTAL POINTS PER ROW IS ',I6 /' FINAL POINT IS ',I6/
     1  ' NUMBER OF OVERLAPPING HORIZONTAL SUBIMAGES IN THE ROW IS ',I5)
C                                   SET THE RANDOM DISC FILE FOR A FIXED
C                                   LENGTH OF NDIM WORDS PER RECORD
      CALL RANSIZ(11,NDIM,1)
C                                   SKIP TO THE LINE BEFORE IRSTRT
      IF(IRSTR.EQ.0)  RETURN
      CALL ESKIP(IRSTR)
      RETURN
C                                   ENTRY POINT TO GET A ROW OF SUBIMAGES
      ENTRY GETIT
C
      GO TO JP,(950,951)
C                                   CHECK FOR FIRST CALL-IF SO,DONT SKIP
951   IF(LNSTRT.EQ.IRSTR)  GO TO 950
      NOSK=LNSTRT
```

```
      K=NOSK+NUMLIN
C                                 CHECK IF END OF RUN
      IF(K.LE.IRSTOP)  GO TO 901
      WRITE(6,900) IRSTOP,NOSK
900   FORMAT(' PROCESSING TERMINATED--NEXT ROW WOULD EXTEND PAST LAST LI
     1NE ',I6,'   LAST LINE COMPLETED WAS ',I6)
      STOP
C                                 SKIP OVER NOSK LINES IN ERTS FILE
901   CALL ESKIP(NOSK)
950   CONTINUE
C                                 INCREMENT  FOR NEW STARTING LINE
      LNSTRT=LNSTRT+LNINCR
C                                 READ IN NUMLIN LINES FOR THE NEW ROW
      KT=0
      DO 903 L=1,NUMLIN
      CALL EREAD(ILINE,LN)
C                                 CHECK FOR EOF ON ERTS INPUT TAPE
      IF(LN.NE.0)  GO TO 902
      WRITE(6,905)
905   FORMAT(' EOF DETECTED ON ERTS INPUT TAPE--PROCESSING TERMINATED')
      STOP
C                                 GO THRU THE LINE CELL BY CELL
902   DO 960 IP=IS,IE,4
      KT=KT+1
C                                 GET ALL 8 CROSS-BAND COMPONENTS FOR
C                                 THIS RESOLUTION CELL
      F(1)=ILINE(IP+2)
      F(2)=(ILINE(IP+2))*ILINE(IP+2)
      F(3)=ILINE(IP+3)
      F(4)=(ILINE(IP+3))*ILINE(IP+3)
      F(5)=(ILINE(IP+2))*ILINE(IP+3)
      F(6)=ILINE(IP+4)
      F(7)=(ILINE(IP+4))*ILINE(IP+4)
      F(8)=(ILINE(IP+2))*ILINE(IP+4)
C                                 WRITE OUT THE N-TUPLE OF LENGTH NDIM
      WRITE(11'KT)   (F(I),I=1,NDIM)
960   CONTINUE
903   CONTINUE
      GO TO KP,(970,971)
C                                 IF THERE IS OVERLAPPING, REWIND TAPE
971   CALL EREWND
970   RETURN
      END
7 MEMORY EXPANDED. USE $LIMITS OR CORE= OPTION FOR NEXT RUN
```

```
CDIFFER                 GET THE DIFFERENCE ARRAY
C
C     WRITTEN BY RJ BOSLEY                                   JAN 1974
C
C     DESCRIPTION OF PROGRAM
C        THIS PROGRAM REPLACES THE ORIGINAL IMAGE IA WITH THE NEAREST
C     NEIGHBOR HORIZONTAL DIFFERENCE, (I1-J1,I2-J2,...,IN-JN) WHERE
C     I AND J ARE N-DIMENSIONAL HORIZONTALLY NEIGHBORING RESOLUTION
C     CELLS OF DISTANCE IDIST.  NOTE THAT THE ABSOLUTE VALUE IS USED
C     GIVING ONLY THE POSITIVE HALF OF THE DISTRIBUTION OF
C     DIFFERENCES I-J AND J-I.
C
C     ENTRY POINT.
C        CALL DIFFER(IA,X,IXDIM,IYDIM,NDIM,IDIST,NUMPPL,NUMLIN)
C     ARGUMENTS.
C              IA                       THE SUBIMAGE BEING PROCESSED
C              X                        DIFFERENCE ARRAY
C              IXDIM                    COL DIMENSION OF X
C              IYDIM                    ROW DIMENSION OF X
C              NDIM                     THE DIMENSION OF EACH RESOLUTION CELL
C              IDIST                    DISTANCE BETWEEN RESOLUTION CELLS
C                                       USED FOR THE DIFFERENCE ARRAY
C              NUMPPL                   NUMBER OF COLUMNS IN SUBIMAGE
C              NUMLIN                   NUMBER OF LINES IN SUBIMAGE
C
      SUBROUTINE DIFFER(IA,X,IXDIM,IYDIM,NDIM,IDIST,NUMPPL,NUMLIN)
C
      DIMENSION IA(IYDIM,IXDIM,1),X(IYDIM,IXDIM,1)
C                                       GO THRU EACH DIMENSION OR BAND
      DO 5 IBAND=1,NDIM
C                                       GO THRU EACH LINE
      DO 4 LINE=1,NUMLIN
C                                       GO THRU ALL BUT THE LAST COLUMN
      DO 3 KOL=1,NUMPPL
      KKOL=KOL+IDIST
C                                       REPLACE EACH RESOLUTION CELL COMPONENT
C                                       BY THE DIFFERENCE
      X(LINE,KOL,IBAND)=IABS(IA(LINE,KOL,IBAND)-IA(LINE,KKOL,IBAND))
    3 CONTINUE
    4 CONTINUE
    5 CONTINUE
      RETURN
      END
```

```
CCOVAR                 FIND THE COVARIANCE MATRIX FOR THE SUBIMAGE
C                                                          JAN 1974
C        WRITTEN BY RJ BOSLEY
C
C        DESCRIPTION OF PROGRAM
C           THIS PROGRAM TAKES THE DIFFERENCE SUBIMAGE AND CALCULATES THE
C        COVARIANCE MATRIX FOR IT.
C
C        DESCRIPTION OF PARAMETERS
C                 PERCNT              PERCENTAGE OF TOTAL VECTORS X FROM
C                                     WHICH COVARIANCE IS CALCULATED
C                 SCR,XLINE,SAM       SCRATCH ARRAYS
C                 IER                 ERROR FLAG  FROM MNCVIN
C                 JER                 ERROR FLAG  FROM MNCV
C        ENTRY POINT.
C           CALL COVAR(XLINE,NDIM,NUMPPL,X,IXDIM,IYDIM,NUMLIN,NDIN,COV)
C        INPUT ARGUMENTS.
C                 XLINE               ARRAY USED TO SEND ONE LINE TO MNCV
C                 NDIM                DIMENSION OF VECTORS X, NO OF BANDS
C                 NUMPPL              NUMBER OF COLUMNS IN SUBIMAGE
C                 X                   FLOATING POINT DIFFERENCE ARRAY
C                 IXDIM,IYDIM         COL,ROW DIMENSIONS OF X
C                 NUMLIN              NUMBER OF LINES IN SUBIMAGE
C                 NDIN                DIMENSION OF COV ARRAY
C
C        OUTPUT ARGUMENTS.
C                 COV                 COVARIANCE MATRIX FOR THE SUBIMAGE
C
C
      SUBROUTINE COVAR(XLINE,NDIM,NUMPPL,X,IXDIM,IYDIM,NUMLIN,NDIN,COV)
C
      DIMENSION X(IYDIM,IXDIM,1),XLINE(NDIM,NUMPPL),XMEAN(8),SCR(8),
     1          NTRUTH(1),SAM(1),COV(NDIN,NDIN)
      PERCNT=100.0
C                                     INITIALIZE THE COVARIANCE PROGRAM
      CALL MNCVIN(NUMPPL,NDIM,NUMLIN,PERCNT,1,XLINE,NTRUTH,COV,XMEAN,
     1      SCR,SAM,IER,JER)
C                                     CHECK FOR AN ERROR
      IF(IER.EQ.0)   GO TO 1
      WRITE(6,2) IER
    2 FORMAT('   ERROR IN MNCVIN, IERROR IS ',I3)
      STOP
    1 CONTINUE
C                                     GO THRU EACH LINE OF SUBIMAGES
      DO 10 LINE=1,NUMLIN
```

```
C                                      GET ONE LINE OF DATA                COVAR046
      DO 6 I=1,NDIM                                                        COVAR047
      DO 5 KOL=1,NUMPPL                                                    COVAR048
5     XLINE(I,KOL)=X(LINE,KOL,I)                                           COVAR049
6     CONTINUE                                                             COVAR050
C                                      INCREMENT COVARIANCE CALCULATIONS   COVAR051
      CALL MNCV                                                            COVAR052
C                                      CHECK FOR GROUND TRUTH ERROR        COVAR053
      IF(JER.EQ.0)  GO TO 10                                               COVAR054
      WRITE(6,7) JER                                                       COVAR055
7     FORMAT('  ERROR IN MNCV, JERROR IS ',I2)                             COVAR056
      STOP                                                                 COVAR057
10    CONTINUE                                                             COVAR058
C                                      NOTE***ONLY THE POSITIVE DIFFERENCES COVAR059
C                                      WERE USED IN THE CALCULATIONS---THE  COVAR060
C                                      TRUE MEAN MUST BE ZERO.  SO WE MUST  COVAR061
C                                      ADD XMEAN**2 TO EACH ELEMENT        COVAR062
      DO 20 I=1,NDIM                                                       COVAR063
      DO 20 J=I,NDIM                                                       COVAR064
      COV(I,J)=COV(I,J)+XMEAN(I)*XMEAN(J)                                  COVAR065
20    COV(J,I)=COV(I,J)                                                    COVAR066
      RETURN                                                               COVAR067
      END                                                                  COVAR068
```

```
CMNCV        M N C V I N  -  M N C V
C
C-----------------------------------------------------------------------
C
C     IDENTIFICATION
C
C          PROGRAM NAME              MNCVIN
C          OTHER ENTRY POINT         MNCV
C          SYSTEM                    PDP-15
C          SOURCE LANGUAGE           FORTRAN IV
C          AUTHOR                    JAMES D YOUNG
C          DATE                      8/18/73
C
C     PURPOSE
C
C          TO CALCULATE THE MEAN VECTOR AND COVARIANCE MATRIX FOR EACH
C          CATEGORY OF A SET OF VECTORS.  THE CALCULATIONS WILL BE
C          BASED ON A SPECIFIED PERCENTAGE OF THE VECTORS RANDOMLY
C          CHOSEN WITHIN THE SET.
C
C     ENTRY POINT - MNCVIN(NVPCAL,NDIM,NCALL,PERCNT,NCAT,X,NTRUTH,COV,
C                          XMEAN,SCTMEN,SAMSZ,IERROR,JERROR)
C
C          THIS INITIALIZES THE ROUTINE.  AFTER MNCVIN HAS BEEN CALLED
C          CHECK IERROR TO SEE IF IT IS NONZERO WHICH INDICATES THAT AN
C          ERROR HAS OCCURRED.
C
C     ENTRY POINT - MNCV
C
C          THE CALL TO MNCV SHOULD BE PERFORMED NCALL TIMES, EACH
C          TIME WITH THE NEXT GROUP OF VECTORS IN X.  IF MORE THAN ONE
C          CATEGORY IS BEING CONSIDERED, THE GROUND TRUTH INTEGERS IN
C          NTRUTH ASSOCIATED WITH THE VECTORS SHOULD ALSO BE UPDATED
C          EACH TIME MNCV IS CALLED, AND JERROR SHOULD BE CHECKED AFTER
C          EACH CALL TO SEE  IF IT IS NONZERO WHICH INDICATES THAT AN
C          ILLEGAL GROUND TRUTH INTEGER HAS BEEN FOUND.  IF ONLY ONE
C          CATEGORY IS BEING CONSIDERED HOWEVER, THE VALUES IN NTRUTH
C          ARE NOT USED IN THE ROUTINE.
C
C          AFTER MNCV HAS BEEN CALLED NCALL TIMES THE MEAN VECTOR AND
C          COVARIANCE MATRIX FOR EACH CATEGORY IS COMPLETED.  IN
C          ADDITION, THE NUMBER OF VECTORS USED FOR THE CALCULATIONS FOR
C          EACH CATEGORY IS OUTPUT.  THESE NUMBERS SHOULD BE CHECKED
C          WHEN APPROPRIATE TO SEE IF ENOUGH VECTORS WERE USED FROM
C          EACH CATEGORY TO GIVE A RESONABLE ESTIMATE OF THE DESIRED
C          STATISTICS.
C
C     ARGUMENTS
C
C        INPUT -
C
C          NVPCAL  NUMBER OF VECTORS PER CALL
```

```
C
C           NDIM     DIMENSION OF DATA VECTORS
C
C           NCALL    NUMBER OF CALLS
C
C           PERCNT   PERCENTAGE OF TOTAL NUMBER OF VECTORS FROM WHICH
C                    MEAN AND COVARIANCE MATRICES WILL BE CALCULATED
C
C           NCAT     NUMBER OF CATEGORIES CONSIDERED
C                    =1 IF ONLY ONE SET OF STATISTICS WILL BE
C                       CALCULATED FOR ALL DATA
C                    =NUMBER OF CATEGORIES IN DATA IF ONE SET OF
C                       STATISTICS WILL BE CALCULATED FOR EACH CATEGORY
C
C           X        X(NDIM,NVPCAL) MATRIX CONTAINING INPUT DATA VECTORS
C                    IN ITS COLUMNS
C
C           NTRUTH   NTRUTH(NVPCAL) VECTOR CONTAINING THE GROUND TRUTH
C                    INTEGERS, 1 THROUGH NCAT, ASSOCIATED WITH THE DATA
C                    VECTORS OF X.  IF NCAT EQUALS 1 THIS VECTOR WILL NOT
C                    BE USED.
C
C        OUTPUT -
C
C           COV      COV(NDIM,NDIM,NCAT) MATRIX CONTAINING COVARIANCE
C                    MATRIX/MATRICES OF THE DATA
C
C           XMEAN    XMEAN(NDIM,NCAT) MATRIX CONTAINING MEAN
C                    VECTOR/VECTORS OF THE DATA
C
C           SCTMEN   SCTMEN(NDIM,NCAT) SCRATCH MATRIX CONTAINING AN
C                    ESTIMATE OF THE MEAN VECTOR/VECTORS
C
C           SAMSZ    SAMSZ(NCAT) VECTOR CONTAINING NUMBER OF VECTORS USED
C                    TO CALCULATE THE STATISTICS FOR EACH CATEGORY
C
C           IERROR   ERROR INDICATED IF RETURNED NONZERO
C                    =1 IF NVPCAL .LE. 0
C                    =2 IF NDIM   .LE. 0
C                    =3 IF NCAL   .LE. 0
C                    =4 IF PERCNT .GT. 100. OR PERCNT IS SO SMALL THAT
C                       FEWER THAN 2 VECTORS WILL BE USED TO CALCULATE
C                       ALL THE STATISTICS
C                    =5 IF NCAT   .LE. 0
C                 (IF MORE THAN ONE OF THESE ERRORS OCCURS,
C                  THE HIGHER VALUE WILL BE RETURNED)
C
C           JERROR   ERROR INDICATED IF RETURNED NONZERO
C                    =1 IF ILLEGAL GROUND TRUTH LABEL IS FOUND
C
C        INTERNAL PARAMETERS
C
```

```
C              ICALL    NUMBER OF TIMES MNCV HAS BEEN CALLED
C              VECTS    TOTAL NUMBER OF VECTORS WHICH WILL BE INPUT TO MNCV
C              VTBU     TOTAL NUMBER OF VECTORS TO BE USED IN CALCULATION
C                       OF STATISTICS
C              VLTBU    NUMBER OF VECTORS LEFT TO BE USED IN CALCULATION
C                       OF STATISTICS
C              VLEFT    NUMBER OF VECTORS LEFT TO BE CONSIDERED
C              INTRU    INTEGER DENOTING GROUND TRUTH CATEGORY
C              IP       ARGUMENT TO RCM
C              IQ       ARGUMENT TO RCM
C
C-----------------------------------------------------------------------
C
C                                   INITIALIZER ENTRY POINT
C
      SUBROUTINE MNCVIN(NVPCAL,NDIM,NCALL,PERCNT,NCAT,X,NTRUTH,COV,
     1                  XMEAN,SCTMEN,SAMSZ,IERROR,JERROR)
C
C
      DIMENSION X(NDIM, 1),NTRUTH(1),COV(NDIM,NDIM,NCAT),XMEAN(NCAT,1),
     1 SCTMEN(NDIM,1),SAMSZ(NCAT)
C
      ICALL=0
      VECTS=NVPCAL*NCALL
      IERROR=0
      NVTBU=VECTS*PERCNT/100.+0.49999
      VTBU=NVTBU
      VLTBU=VTBU
      VLEFT=VECTS
      INTRU=1
      IP=33333
      IQ=55555
      JERROR=0
C
C                                   CHECK LEGALITY OF SOME NUMBERS
C
      IF(NVPCAL.LE.0) IERROR=1
      IF(NDIM  .LE.0) IERROR=2
      IF(NCALL .LE.0) IERROR=3
      IF(PERCNT.GT.100..OR.VTBU.LT.2.) IERROR=4
      IF(NCAT  .LE.0) IERROR=5
C
C                                   ZERO OUT A FEW ARRAYS
C
      DO 14 K=1,NCAT
      SAMSZ(K)=0
      DO 14 J=1,NDIM
      XMEAN(J,K)=0
      SCTMEN(J,K)=0
      DO 14 I=1,NDIM
   14 COV(I,J,K)=0
C
```

```
C                                       SET TRANSFER TO REFLECT NUMBER
C                                       OF CATEGORIES
C
      ASSIGN  5 TO IGO
      IF(NCAT.GT.1) ASSIGN 15 TO IGO
      RETURN
C
C*************************************************************************
C
C                                       ENTRY POINT FOR ALL CALLS AFTER THE FIRST
C
      ENTRY MNCV
C
      ICALL=ICALL+1
      IF((ICALL/50)*50.NE.ICALL.AND.ICALL.NE.10)  GO TO 1
C
C                                       UPDATE ESTIMATE OF THE MEANS AND MODIFY
C                                       COVARIANCE CALCULATIONS TO REFLECT
C                                       THIS UPDATE
C
C
      DO 10 K=1,NCAT
      IF(SAMSZ(K).EQ.0.) GO TO 10
      DO 2 J=1,NDIM
      DO 2 I=1,J
    2 COV(I,J,K)=COV(I,J,K)-SAMSZ(K)*(XMEAN(I,K)/SAMSZ(K)-SCTMEN(I,K))*
     1  (XMEAN(J,K)/SAMSZ(K)-SCTMEN(J,K))
      DO 3 J=1,NDIM
      SCTMEN(J,K)=XMEAN(J,K)/SAMSZ(K)
    3 CONTINUE
   10 CONTINUE
C
C                                       CONTINUE TO CALCULATE MEAN AND
C                                       UPPER TRIANGLE OF COVARIANCE MATRICES
C
    1 DO 8 I=1,NVPCAL
C
C                                       DETERMINE WHETHER TO SKIP THIS VECTOR
C
      IF(RCM(IP,IQ).GT.VLTBU/VLEFT)  GO TO 11
C                                       INCLUDE THIS VECTOR IN CALCULATIONS
C
C
      VLTBU=VLTBU-1.
      GO TO IGO,(15,5)
C
C                                       WE ARE TO CONSIDER MORE THAN ONE
C                                       CATEGORY
C
   15 INTRU=NTRUTH(I)
C
C                                       CHECK LEGALITY OF GROUND TRUTH LABEL
C
```

```
      IF(INTRU.LE.0.OR.INTRU.GT.NCAT) GO TO 9
    5 DO 4 J=1,NDIM
C
C                                   SUM FOR MEAN
C
      XMEAN(J,INTRU)=XMEAN(J,INTRU)+X(J,I)
      DO 4 K=1,J
C
C                                   SUM FOR FOR COVARIANCE
C
      COV(K,J,INTRU)=COV(K,J,INTRU)+(X(J,I)-SCTMEN(J,INTRU))*(X(K,I)-
     1  SCTMEN(K,INTRU))
    4 CONTINUE
      SAMSZ(INTRU)=SAMSZ(INTRU)+1.
   11 CONTINUE
      VLEFT=VLEFT-1.
    8 CONTINUE
      IF(ICALL.LT.NCALL) RETURN
C
C                                   ALL VECTORS HAVE BEEN INPUT
C                                   FINISH CALCULATION OF STATISTICS
C
      DO 7 K=1,NCAT
      IF(SAMSZ(K).EQ.0.) GO TO 7
      DO 6 J=1,NDIM
      DO 6 I=1,J
      COV(I,J,K)=COV(I,J,K)/SAMSZ(K)-(XMEAN(I,K)/SAMSZ(K)-SCTMEN(I,K))*
     1  (XMEAN(J,K)/SAMSZ(K)-SCTMEN(J,K))
C
C                                   FILL IN LOWER TRIANGLE
C
    6 COV(J,I,K)=COV(I,J,K)
      DO 13 J=1,NDIM
      XMEAN(J,K)=XMEAN(J,K)/SAMSZ(K)
   13 CONTINUE
    7 CONTINUE
   12 CONTINUE
      RETURN
    9 JERROR=1
      GO TO 12
      END
```

```
CCORREL               GET THE CORRELATION MATRIX FOR THE SUBIMAGE
C                                                              JAN 1974
C     WRITTEN BY RJ BOSLEY
C
C     DESCRIPTION OF PROGRAM
C        THIS PROGRAM CALCULATES THE CORRELATION MATRIX GIVEN THE
C     COVARIANCE MATRIX AND ITS ORDER.
C
C     ENTRY POINT.
C               CALL CORREL(COV,NDIM,COR)
C     ARGUMENTS.
C                COV                 COVARIANCE MATRIX ARRAY
C                NDIM                ORDER OF MATRICES
C                COR                 CORRELATION MATRIX ARRAY
C
      SUBROUTINE CORREL(COV,NDIM,COR)
C
      DIMENSION COV(NDIM,NDIM),COR(NDIM,NDIM)
C                                    GO THRU EACH ROW OR LINE
      DO 10 LINE=1,NDIM
      COVL=ABS(COV(LINE,LINE))
C                                    GO THRU UPPER DIAGONAL
      DO 9 KOL=LINE,NDIM
      COVC=ABS(COV(KOL,KOL))
      COR(LINE,KOL)=COV(LINE,KOL)/SQRT(COVC*COVL)
    9 COR(KOL,LINE)=COR(LINE,KOL)
   10 CONTINUE
      RETURN
      END
```

IV.2-d Piecewise Linear Classification Program Listings

RCLASS
  XIN
  LINEAR
    WEIGHT

```
CRCLASS              PATTERN VECTOR CLASSIFICATION
C
C      WRITTEN BY SAM SHANMUGAM                                   SEPT 1972
C      DOCUMENTED BY RJ BOSLEY                                    DEC  1973
C
C
C      DESCRIPTION OF PROGRAM.
C         THIS SUBROUTINE CLASSIFIES A GIVEN SET OF PATTERN VECTORS INTO
C      ONE OF MANY POSSIBLE CATEGORIES USING THE INFORMATION CONTAINED
C      IN A SET OF TRAINING PATTERNS WHOSE CATEGORIES ARE KNOWN.
C
C      ENTRY POINT.              CALL RCLASS(WORK,ISIZE)
C
C
C      INPUT ARGUMENTS.
C                        WORK - SCRATCH ARRAY USED BY RCLASS
C                        ISIZE - SIZE OF THE SCRATCH ARRAY WORK . THE ARRAY
C                                WORK MUST BE DIMENSIONED IN THE CALLING
C                                PROGRAM,WITH A DIMENSION GREATER THAN OR
C                                EQUAL TO ISIZE. THE VALUE OF ISIZE IS GIVEN
C                                BY
C                                     ISIZE=NDIM*(NTRAIN+10)+1000
C                                          +(NC*(NC+1)/2)*ND+NPAIR*ND
C                                WHERE ND=NDIM-1
C
C
C      INPUT PARAMETER CARDS.
C         CARD 1. SHOULD CONTAIN THE PROGRAM OPTION PARAMETERS NOPT1,
C                 NOPT2,NOPT3,NOPT4, IN FORMAT(4I1).
C
C                 IF(NOPT1.NE.0) THE TRAINING PATTERNS ARE PRINTED OUT
C                 IF(NOPT2.NE.0) THE TEST PATTERNS ARE PRINTED OUT
C                 IF(NOPT3.NE.0) THE CLASSIFICATION OF EACH TRAINING
C                                PATTERN IS LISTED. OTHERWISE,ONLY THE
C                                CONTINGENCY TABLE IS PRINTED OUT
C                 IF(NOPT4.NE.0) THE CLASSIFICATION OF EACH TEST
C                                PATTERN IS LISTED.  OTHERWISE ONLY
C                                THE CONTINGENCY TABLE IS PRINTED OUT.
C
C         CARD 2. SHOULD CONTAIN PARAMETERS NTOT,NPART,NDIM,NC,NPAIR IN
C                 FORMAT (5I5).
C                 NTOT  -- TOTAL NUMBER OF PATTERN VECTORS IN THE DATA
C                          SET
C                 NPART -- NPART OUT OF EVERY 10 PATTERN VECTORS IN THE
C                          DATA SET WILL BE USED FOR TRAINING.  THE
C                          REMAINING PATTERNS WILL BE USED FOR TESTING
C                          THE CLASSIFIER
C                 NDIM  -- NUMBER OF MEASUREMENTS PER VECTOR  PLUS 2
C                 NC    -- NUMBER OF GROUND TRUTH CATEGORIES
C                 NPAIR -- 2*(MAXIMUM NUMBER OF TRAINING PATTERNS IN
C
```

```
C                        ANY ONE CATEGORY)
C
C
C     INPUT DATA SET.
C        INPUT DATA VECTORS ARE READ IN BINARY FROM FILE CODE 01
C     ACCORDING TO THE FOLLOWING READ---
C                 READ(01) IGT,M1,N1,NFT,(FEAT(I),I=1,NMEAS)
C     WHERE   IGT         IS THE GROUND TRUTH CATEGORY
C             M1,N1,NFT ARE NOT USED (MAY BE USED AS TO TAGS)
C             FEAT(I)     IS THE FEATURE VECTOR I-TH COMPONENT
C             NMEAS       IS THE NUMBER OF COMPONENTS OR MEASUREMENTS IN
C                         EACH FEATURE VECTOR=NDIM-2.
C     THE DATA FILE SHOULD HAVE A TOTAL OF NTOT LOGICAL RECORDS IN
C     BINARY FORM.  EACH LOGICAL RECORD MUST BE OF LENGTH (NDIM+2)
C     WORDS.  WORD 1 IS THE GROUND TRUTH CATEGORY AND 5 THRU NDIM+2 ARE
C     THE MEASUREMENT VALUES.  THE PATTERN VECTORS MUST BE SORTED BY
C     CATEGORY.  THE DATA SET IS SORTED INTO TRAINING AND TEST SETS
C     ACCORDING TO THE USER SPECIFIED RATIO IN NPART. TRAINING PATTERNS
C     ARE COPIED TO CORE AND TEST PATTERNS ARE COPIED TO DISC ON FILE
C     CODE 02.
C
C     LIMITATIONS.
C             1. MAXIMUM NUMBER OF CATEGORIES IS 15.
C             2. MAXIMUM VALUE OF NDIM IS 100.
C             3. SCRATCH DISC FILE MUST BE ON 02 AND INPUT DATA ON 01.
C             4. PATTERN VECTORS MUST BE SORTED BY CATEGORY.
C
C     THEORY.
C                 USING A REGRESSION TYPE ALGORITHM, THE PROGRAM
C                 OBTAINS A SET OF HYPERPLANES  FOR SEPARATING THE
C                 TRAINING PATTERNS . THE SEPARATION IS DONE FOR
C                 DIFFERENT CATEGORY PAIRS. A TOTAL OF NC*(NC-1)/2
C                 HYPER PLANES ARE DETERMINED. TEST PATTERNS ARE
C                 CLASSIFIED ON THE BASIS OF A MAJORITY VOTE ON THESE
C                 HYPERPLANES.
C
C     BIBLIOGRAPHY.
C        FOR A COMPLETE DISCUSSION OF DETAILS, SEE ''INTRODUCTION TO
C     STATISTICAL PATTERN RECOGNITION'' BY FUKUNAGA, ACADEMIC PRESS,
C     NEW YORK,1972.
C
C     SUBPROGRAMS REQUIRED.
C        DRIVER
C            RCLASS
C                XIN
C                LINEAR
C                    WEIGHT
C                        MINV  -- A FORTRAN CALLABLE ROUTINE FROM IBM SSP
C
C     EXAMPLE.
C        IN ORDER TO PROCESS A DATA SET REQUIRING A SCRATCH ARRAY OF
```

```
C     SIZE 15000, THE DRIVER  SET UP IS AS FOLLOWS ---
C
C             CDRIVE          SAMPLE DRIVER FOR CLASSIFICATION PGM
C                    DIMENSION WORK(15000)
C                    ISIZE = 15000
C                    CALL RCLASS(WORK,ISIZE)
C                    STOP
C                    END
C
C
      SUBROUTINE RCLASS(WORK,ISIZE)
C
C
      DIMENSION WORK(ISIZE),DD(50)
      COMMON NTRAIN,NTEST,NDIM,ND,NC,NPAIR,NH,NOPT1,NOPT2,NOPT3,NOPT4,
     1 NTOT,NPART
C                                  READ IN PARAMETER CARDS AND DATA SET
C
      CALL XIN(WORK(1),DD(1))
C                                  LOCATE THE BEGINNING ADDRESS OF THE
C                                  VARIOUS ARRAYS FOR THE LINEAR SUBPGM
C
      I1=1
      I2=I1+NTRAIN*NDIM
      I3=I2+NDIM
      I4=I3+NH*ND
      I5=I4+NPAIR*ND+1500
C                                  CHECK THAT THE DIMENSION OF THE WORK
C                                  ARRAY IS ADEQUATE
      IF(I5.LE.ISIZE)  GO TO 53
      WRITE(6,82)
   82 FORMAT(1H1,5X,'PROBLEM WILL NOT FIT THE CORE ALLOTED FOR THIS PROG
     1AM.')
      RETURN
   53 CONTINUE
C                                  NOW GO DO THE CLASSIFICATION
      CALL LINEAR(WORK(I1),WORK(I2),WORK(I3),WORK(I4),WORK(I5))
       RETURN
      END
```

```
CXIN                  READ IN THE DATA
C
C      WRITTEN BY SAM SHANMUGAM                              SEPT 1972
C      DOCUMENTED BY RJ BOSLEY                               DEC  1973
C
C
C      DESCRIPTION OF PROGRAM.
C         THIS SUBROUTINE READS IN THE TWO PARAMETER CARDS AND THEN
C      READS IN THE DATA SET, COPYING THE  TEST PATTERNS TO DISC FILE 02.
C
C      ENTRY POINT.
C              CALL XIN(WORK,U)
C
C      INPUT ARGUMENTS.
C              WORK                ARRAY DATA IS READ INTO
C              U                   SCRATCH ARRAY
C
C      INTERNAL PARAMETERS.
C                NTRAIN            NO. OF TRAINING PATTERN VECTORS
C                NTEST             NO OF TEST PATTERN VECTORS
C                NDIM              2+NO. OF MEAS PER FEATURE VECTOR
C                ND                NDIM-1
C                NC                NO. OF GROUND TRUTH CATEGORIES
C                NPAIR             TWICE MAX NO. OF TRAINING PATTERNS
C                NH                NO. OF HYPERPLANES
C                NOPT1-4           PROGRAM OPTIONS. SEE RCLASS
C                NSKIP             NO. OF VECTORS OUT OF 10 TO SKIP
C                                     TRAINING
C                NTIMES            NO. OF BLOCKS OF 10 PATTERN VECTORS
C                                     IN INPUT DATA FILE
C                ILEFT             NO. OF VECTORS LEFT AFTER READING ALL
C                                     BLOCKS OF 10 VECTORS
C
C
C
       SUBROUTINE XIN(WORK,U)
C
       DIMENSION WORK(1),U(1)
       COMMON NTRAIN,NTEST,NDIM,ND,NC,NPAIR,NH,NOPT1,NOPT2,NOPT3,NOPT4,
      1  NTOT,NPART
C                                  READ IN PARAMETER CARD 1.
       READ(5,100) NOPT1,NOPT2,NOPT3,NOPT4
C                                  READ IN PARAMETER CARD 2.
       READ(5,101) NTOT,NPART,NDIM,NC,NPAIR
  100  FORMAT(4I1)
  101  FORMAT(5I5)
C                                  READ THRU THE INPUT FILE TWO TIMES --
C                                  ONCE TO GET THE TRAINING PATTERNS AND
C                                  ONCE TO GET THE TEST PATTERNS
       REWIND 01
C                                  INITIALIZE  INTERNAL PARAMETERS
       IBEGIN=1
```

```
      NTRAIN=0
      NDD=NDIM+2
      NTIMES=NTOT/10
C                                  OUT OF EACH OF THE NTIMES BLOCKS OF
C                                  TEN INPUT PATTERNS PICK THE FIRST
C                                  NPART PATTERNS AS TRAINING PATTERNS
      NSKIP=10-NPART
C
C                                  WRITE HEADING
      IF(NOPT1.NE.0) WRITE(6,201)
  201 FORMAT(1H1,10X,'TRAINING PATTERNS')
C                                  GO THRU EACH BLOCK OF TEN
C                                     PATTERN VECTORS
      DO 20 NN=1,NTIMES
      IF(NSKIP.EQ.0) GO TO 16
C                                  SKIP OVER THE TEST VECTORS
      DO 15 JJ=1,NSKIP
C                                  READ PAST THIS VECTOR
   15 READ(1) (U(K),K=1,NDD)
C
   16 CONTINUE
C                                  READ NPART TRAINING PATTERN VECTORS
      DO 10 JJ=1,NPART
C                                  COUNT THE NO. OF TRAINING VECTORS
      NTRAIN=NTRAIN+1
C                                  INDEX THIS VECTOR INTO WORK
      IBEGIN=(NTRAIN-1)*NDIM+1
      IEND=NTRAIN*NDIM
      IB1=IBEGIN+1
      IB2=IB1+1
C                                  SET THE FIRST COMPONENT TO 1.0
      WORK(IB1)=1.0
C                                  READ THE TRAINING PATTERN VECTORS
      READ(1) WORK(IBEGIN),NM1,NM2,NM3,(WORK(K),K=IB2,IEND)
C                                  WRITE OUT THE TRAINING PATTERNS
      IF(NOPT1.NE.0) WRITE(6,102) NTRAIN,WORK(IBEGIN),NM1,NM2,(WORK(K),
     1K=IB1,IEND)
  102 FORMAT(1X,I3,3X,6(F10.4))
   10 CONTINUE
   20 CONTINUE
C                                  READ IN THE LEFTOVER PATTERNS
      ILEFT=NTOT-10*NTIMES
      IF(ILEFT.EQ.0) GO TO 26
      DO 25 JJ=1,ILEFT
C                                  INCREMENT TRAINING VECTOR COUNT
      NTRAIN=NTRAIN+1
C                                  INDEX THE VECTOR INTO WORK
      IBEGIN=(NTRAIN-1)*NDIM+1
      IEND=NTRAIN*NDIM
      IB1=IBEGIN+1
```

```
      IB2=IB1+1
      WORK(IB1)=1.0
C                                READ A TRAINING PATTERN
      READ(1) WORK(IBEGIN),NM1,NM2,NM3,(WORK(K),K=IB2,IEND)
C                                WRITE OUT TRAINING PATTERNS
      IF(NOPT1.NE.0) WRITE(6,102) NTRAIN,WORK(IBEGIN),NM1,NM2,(WORK(K),
     1K=IB1,IEND)
C
   25 CONTINUE
   26 CONTINUE
      REWIND 01
      REWIND 02
C                                COPY TEST PATTERNS TO DISC FILE 02
C
C                                WRITE HEADING
      IF(NOPT2.NE.0) WRITE(6,103)
  103 FORMAT(1H1,10X,'TEST PATTERNS')
      NTEST=0
C                                SKIP IF THERE ARE NO TEST PATTERNS
      IF(NPART.EQ.0) GO TO 99
      DO 45 NN=1,NTIMES
      DO 43 JJ=1,NSKIP
C                                COUNT THE NO. OF TEST VECTORS
      NTEST=NTEST+1
C                                READ INPUT DATA FILE THE SECOND TIME
      READ(1) (U(KK),KK=1,NDD)
C                                SET COMPONENT FOUR TO 1.0
      U(4)=1.0
C                                WRITE OUT THE TEST PATTERNS
      IF(NOPT2.NE.0) WRITE(6,102) NTEST,U(1),U(2),U(3),(U(KK),KK=4,NDD)
C                                WRITE TEST PATTERN TO DISC FILE 02
   43 WRITE(2) (U(KK),KK=1,NDD)
C                                SKIP OVER THE TRAINING VECTORS
      DO 44 JJ=1,NPART
   44 READ(1) (U(KK),KK=1,NDD)
   45 CONTINUE
      END FILE 02
      REWIND 02
   99 CONTINUE
      REWIND 01
C                                WRITE OUT A PROGRESS NOTE
      WRITE(6,702)
  702 FORMAT(5X,'DATA HAS BEEN COPIED ON TO DISK 02')
C                                DETERMINE THE NO. OF HYPERPLANES
      NH=NC*(NC+1)/2
      ND=NDIM-1
      RETURN
      END
```

```
CLINEAR            LINEAR DISCRIMINANT FUNCTION
C
C       WRITTEN BY SAM SHANMUGAM                               SEPT 1972
C       DOCUMENTED BY RJ BOSLEY                                DEC  1973
C
C       DESCRIPTION OF PROGRAM.
C          THIS SUBROUTINE CALLS THE REGRESSION ROUTINE WEIGHT TO GET THE
C       DISCRIMINATION FUNCTIONS, COMBINES THESE TO OBTAIN A PIECEWISE
C       LINEAR DISCRIMINANT FUNCTION FOR EACH CATEGORY, AND CLASSIFIES
C       THE TRAINING AND TEST PATTERNS.  THIS ROUTINE ALSO OUTPUTS THE
C       RESULTS.
C
C       THEORY.
C          THE ALGORITHM FOR DETERMINING THE WEIGHT VECTOR FOR SEPARATING
C       THE I AND J-TH CATEGORIES IS   W=((U*U')**-1)*(U*T), WHERE T IS
C       A VECTOR OF LENGTH ND WITH COMPONENTS OF VALUES EQUAL TO 1.  FOR
C       DETAILS SEE ''INTRODUCTION TO STATISTICAL PATTERN RECOGNITION''
C       BY FUKUNAGA, ACADEMIC PRESS,1972.
C          THE MATRICES (DIMENSIONED AS ONE DIMENSIONAL IN PGM) HAVE THE
C       FOLLOWING STRUCTURES--
C          XTRAIN  --- XTRAIN IS AN NDIM X NTRAIN MATRIX.  EACH COLUMN
C                      VECTOR IN XTRAIN REPRESENTS ONE TRAINING PATTERN.
C                      THE FIRST ENTRY IN EACH COLUMN IS THE CATEGORY NAME
C                      THE SECOND ENTRY IS THE CONSTANT 1.  THE ENTRIES 3
C                      TO NDIM ARE THE VALUES OF THE COMPONENTS OF THE
C                      PATTERN VECTOR X(J).  XTRAIN HAS NTRAIN COLUMNS AND
C                      ND+1 = NDIM ROWS.
C
C          XTEST --- XTEST HAS THE SAME CONFIGURATION AS XTRAIN.
C
C          U     --- DATA MATRIX U IS USED FOR CALCULATING THE BOUNDARY
C                    BETWEEN THE CATEGORIES I AND J.  U HAS ND ROWS AND
C                    NTR(I)+NTR(J) COLUMNS, AS FOLLOWS---
C        ***PATTERNS FROM CATEGORY I*********PATTERNS FROM CATEGORY J -***
C           1         1     .........        -1        -1        ....   -1
C           .         .     ..........        .         .        ....    .
C           .         .     ..........        .         .        ....    .
C           X         X     .........        -X        -X        ....   -X
C           .         .     ..........        .         .        ....    .
C           .         .     ..........        .         .        ....    .
C        ****** U HAS ND ROWS AND NTR(I)+NTR(J) COLUMNS ********************
C
C
C       ENTRY POINT.
C               CALL LINEAR(XTRAIN,XTEST,W,U,DUMMY)
C
C       INPUT ARGUMENTS.
C               XTRAIN                  MATRIX CONTAINING TRAINING PATTERNS
C               XTEST                   MATRIX CONTAINING TEST PATTERNS
C               W                       WEIGHT VECTORS
```

```
C              U                     MATRIX USED FOR CALCULATING THE
C                                       BOUNDARY BETWEEN THE CATEGORIES I
C                                       AND J.
C              DUMMY                 SCRATCH ARRAY
C
C      INTERNAL PARAMETERS.
C              NC                    NUMBER OF CATEGORIES
C              NAME                  ARRAY CONTAINING CATEGORY NAMES
C              NTR(I)                NUMBER OF TRAINING PATTERNS IN CAT I
C              NIJ                   TOTAL NO OF TRAINING VECTORS IN
C                                      CATEGORIES I AND J
C              NTRAIN,NTEST          NUMBER OF TRAINING,TEST PATTERNS
C              NCOLU,NROWU           COL,ROW INDEX FOR MATRIX U
C              NO                    NDIM-1, NO. OF COMP IN WEIGHT VECTOR
C              W                     ARRAY CONTAINING WEIGHT VECTORS
C              WT                    ARRAY CONTAINING ONE WEIGHT VECTOR
C              INDEX                 CLASSIFICATION LOOP INDEX, =1 FOR
C                                       TRAINING SET, =2 FOR TEST SET
C                                       BEING CLASSIFIED
C              NPT                   NO. OF PATTERNS TO CLASSIFY
C              ICLASS                CAT FOR WHICH VOTING IS MAXIMUM
C              KOUNT                 VOTE TALLY
C              KMAX                  MAXIMUM VOTE, MAX KOUNT
C              IERROR                CONTINGENCY TABLE
C
C
      SUBROUTINE LINEAR(XTRAIN,XTEST,W,U,DUMMY)
C
C
      DIMENSION XTRAIN(1),XTEST(1),W(1),U(1),DUMMY(1)
      DIMENSION IERROR(15,15),NTR(15),NAME(15)
      DIMENSION L(100),M(100)
      DIMENSION WT(100)
      COMMON NTRAIN,NTEST,NDIM,NO,NC,NPAIR,NH,NOPT1,NOPT2,NOPT3,NOPT4,
     1  NTOT,NPART
      REAL NAME
C                                    WRITE OUT INPUT PARAMETERS
      WRITE(6,1401)
 1401  FORMAT(1H1,10X,'INPUT DATA SUPPLIED BY YOU')
      WRITE(6,1402) NTOT,NPART,NDIM
 1402 FORMAT(//10X,'TOTAL NUMBER OF PATTERNS=',I4/10X,I3,'OUT OF 10 PATT
     1ERNS WERE USED IN TRAINING'/10X,'NDIM=',I3)
C                                    INITIALIZE NO. OF TRAINING ARRAYS
      DO 1 J=1,NC
 1    NTR(J)=0
C                                    SET UP NAME OF FIRST CATEGORY
      J=1
      NAME(1)=XTRAIN(1)
C                                    GO THRU ALL THE TRAINING PATTERNS
      DO 3 I=1,NTRAIN
```

```
      I1=(I-1)*NDIM+1
C                                       SKIP IF WE HAVE THIS CATEGORY NAME
      IF(NAME(J).EQ.XTRAIN(I1)) GO TO 2
C                                       OR, SET UP A NEW CATEGORY
      J=J+1
C                                       GET THE NEW NAME
      NAME(J)=XTRAIN(I1)
C                                       COUNT THE NUMBER IN THIS CATEGORY
    2 NTR(J)=NTR(J)+1
    3 CONTINUE
C                                       WRITE OUT NO. OF TRAINING AND TEST
C                                       PATTERNS AND THE NO. OF CATEGORIES
C                                       FOUND
      WRITE(6,1403) NTRAIN,NTEST,J
 1403 FORMAT(10X,'NUMBER OF TRAINING PATTERNS USED=',I4/10X,'NUMBER OF
     1TEST PATTERNS USED=',I4/10X,'NUMBER OF GROUND TRUTH LABELS FOUND='
     1,I3)
C                                       WRITE OUT THE NO. OF CATS SPECIFIED
C                                       BY THE USER
      WRITE(6,1404) NC
 1404 FORMAT(/10X,'NUMBER OF GROUND TRUTH LABELS SPECIFIED =',I3)
      WRITE(6,1411)
 1411 FORMAT(1H1,5X,'SUMMARY OF WEIGHT VECTORS'///)
C
C                                       FIND THE HYPERPLANES WHICH SEPARATE
C                                       CATEGORIES I AND J, WHERE I GOES FROM
C                                       1 TO NC AND J GOES FROM I+1 TO NC.
C
      IWCOL=0
      DO 10 I=1,NC
      IPLUS1=I+1
      DO 10 J=IPLUS1,NC
C                                       CHECK FOR FINAL LOOP
      IF(J.GT.NC) GO TO 868
C                                       INCREMENT COLUMN INDEX
      IWCOL=IWCOL+1
      ISUM=0
C                                       SET UP MATRIX U FOR CAT I AND J
      DO 9 K=1,I
    9 ISUM=ISUM+NTR(K)
C                                       FIND THE COL IN XTRAIN WHERE THE
C                                       PATTERNS THAT BELONG TO I AND J BEGIN
      IWBEG=ISUM-NTR(I)
      ISUM=0
      DO 8 K=1,J
    8 ISUM=ISUM+NTR(K)
      JWBEG=ISUM-NTR(J)
C
C                                       GET ENTRIES FROM XTRAIN AND PUT INTO
C                                       THE MATRIX U
```

```
C
      NI=NTR(I)
      NJ=NTR(J)
      NIJ=NTR(I)+NTR(J)
C                                    GO THRU EACH TRAINING PATTERN IN CAT I
      DO 7 NCOLU=1,NI
C                                    INDEX COL FOR XTRAIN
      NCOLXT=NCOLU+IWBEG
C                                    GO THRU EACH COMP IN VECTOR
      DO 6 NROWU=1,ND
C                                    SET UP INDEXS
      NROWX=NROWU+1
      NRCU=(NCOLU-1)*ND+NROWU
      NRCX=(NCOLXT-1)*NDIM+NROWX
C                                    TRANSFER PATTERN TO U FROM XTRAIN
    6 U(NRCU)=XTRAIN(NRCX)
    7 CONTINUE
C                                    DO THE SAME FOR CATEGORY J
C                                    GO THRU EACH PATTERN IN CAT J
      DO 19 NN=1,NJ
C                                    SET UP COL INDEX
      NCOLU=NI+NN
      NCOLXT=JWBEG+NN
      DO 18 NROWU=1,ND
C                                    SET UP ROW INDEX
      NROWX=NROWU+1
      NRCU=(NCOLU-1)*ND+NROWU
      NRCX=(NCOLXT-1)*NDIM+NROWX
C                                    TRANSFER THE VECTOR IN XTRAIN TO U
C                                    AND CHANGE THE SIGN
   18 U(NRCU)=(-1.0)*XTRAIN(NRCX)
   19 CONTINUE
C                                    CALL THE REGRESSION ROUTINE
      CALL WEIGHT(U,DUMMY,WT,ND,NIJ)
C                                    GET WEIGHT VECTOR FOR CATS I AND J
      DO 40 IROWW=1,ND
C                                    INDEX THE VECTOR INTO W ARRAY
      IRCW=(IWCOL-1)*ND+IROWW
   40 W(IRCW)=WT(IROWW)
C                                    WRITE OUT THE WEIGHT VECTOR
      WRITE(6,100) I,J
      IRCW1=(IWCOL-1)*ND+1
      IRCW2=IWCOL*ND
      WRITE(6,101) (W(IRCW),IRCW=IRCW1,IRCW2)
  100 FORMAT(2X,44HTHE WEIGHT VECTOR FOR SEPARATING CATEGORIES ,I3,3HAND
     ,I3)
  101   FORMAT((1X,10F12.5))
C                                    PUNCH OUT THE WEIGHT VECTOR
      WRITE(43,100) I,J
      WRITE(43,1761) (W(IRCW),IRCW=IRCW1,IRCW2)
```

```
 1761 FORMAT((4X,6F12.5))
 868  CONTINUE
 10   CONTINUE
C
C                                  BEGIN CLASSIFICATION
C
C                                  GO THRU FIRST FOR TRAINING PATTERNS
C                                  THEN FOR TEST PATTERNS
C
      DO 67 INDEX=1,2
C                                  WRITE HEADING
      IF((INDEX.EQ.1).AND.(NOPT3.NE.0))WRITE(6,1498)
      IF((INDEX.EQ.2).AND.(NOPT4.NE.0)) WRITE(6,1499)
 1498 FORMAT(1H1,5X,'CLASSIFICATION OF TRAINING PATTERNS')
 1499 FORMAT(1H1,10X,'CLASSIFICATION OF TEST PATTERNS')
C                                  INITIALIZE CONTINGENCY TABLE FOR ALL
C                                  CATEGORIES
      DO 78 IMN=1,15
      DO 78 JMN=1,15
 78   IERROR(IMN,JMN)=0
C                                  WRITE HEADING
      IF((INDEX.EQ.1).AND.(NOPT3.NE.0)) WRITE(6,1500)
      IF((INDEX.EQ.2).AND.(NOPT4.NE.0)) WRITE(6,1500)
 1500 FORMAT(////2X,'PATTERN NO.',5X,'TRUE CAT',6X,'ASSIGNED CAT',9X,'VO
     1TE')
C                                  SET NO. OF PATTERNS TO CLASSIFY
      IF(INDEX.EQ.1) NPT=NTRAIN
      IF(INDEX.EQ.2) NPT=NTEST
C                                  RETURN IF NO TEST SET
      IF((INDEX.EQ.2).AND.(NTEST.EQ.0)) RETURN
C                                  GO THRU EACH PATTERN TO BE CLASSIFIED
      DO 66 III=1,NPT
C                                  GO THRU EACH COMPONENT AND PULL OUT
C                                  THE TRAINING VECTORS FROM CORE
      DO 55 NDUM=1,ND
      IXROW=NDUM+1
      IXIII=(III-1)*NDIM+IXROW
      IF(INDEX.EQ.1) U(NDUM)=XTRAIN(IXIII)
 55   CONTINUE
C                                  OTHERWISE,READ TEST VECTORS FROM 02
      IF(INDEX.EQ.2) READ(2) DUMMY1,DUMMY2,DUMMY3,(U(NDUM),NDUM=1,ND)
      KMAX=-20
C                                  GO THRU EACH CAT FOR VOTING
      DO 65 IIC=1,NC
      KOUNT=0
C
C                                  LOCATE THE HYPERPLANE THAT SEPARATES
C                                  CATS IIC AND JJC.  THE SIGN OF THE
C                                  WEIGHT VECTORS IS CHANGED DEPENDING
C                                  UPON WHETHER OR NOT JJC.GT.IIC
C
```

```
      DO 64 JJC=1,NC
      IF(JJC.EQ.IIC) GO TO 63
      IF(JJC.GT.IIC) GO TO 50
      IF(JJC.LT.IIC) GO TO 41
   50 CONTINUE
C                                 SET PARAMETERS
      SIGN=1.0
      ISUM=0
      IS=IIC-1
      IF(IS.EQ.0) GO TO 53
      DO 51 KS=1,IS
      ITERM=NC-KS
   51 ISUM=ISUM+ITERM
   53 CONTINUE
      NCOLW=ISUM+JJC-IIC
C                                 THEN GO TO VOTE CALCULATIONS
      GO TO 45
   41 CONTINUE
C                                 SET PARAMETERS FOR SIGN CHANGE
      ISUM=0
      SIGN=-1.0
      IS=JJC-1
      IF(IS.EQ.0) GO TO 43
      DO 42 KS=1,IS
      ITERM=NC-KS
   42 ISUM=ISUM+ITERM
   43 CONTINUE
      NCOLW=ISUM+IIC-JJC
   45 CONTINUE
C
C                                 COMPUTE XTRAIN'*W(NCOLW)
      SUM=0.0
      DO 52 IWROW=1,ND
      IXROW=IWROW+1
      IWRCW=(NCOLW-1)*ND+IWROW
      SUM=SUM+(U(IWROW))*(W(IWRCW))*(SIGN)
   52 CONTINUE
      IVOTE=0
C                                 VOTE POSITIVE IF SUM .GT. ZERO
      IF(SUM.GT.0.0) IVOTE=1
C                                 ELSE VOTE NEGATIVE
      IF(SUM.LT.0.0) IVOTE=-1
   63 CONTINUE
C                                 IF SAME CATEGORY, DONT VOTE
      IF(JJC.EQ.IIC) IVOTE=0
C                                 TALLY THE COUNT
   64 KOUNT=KOUNT+IVOTE
C                                 DETERMINE THE CAT FOR WHICH VOTE IS
C                                 MAXIMUM
      IF(KOUNT.GT.KMAX) ICLASS=IIC
```

```
C                                    RESET MAX COUNT                    LINEAR0
      IF(KOUNT.GT.KMAX) KMAX=KOUNT                                      LINEAR0
   65 CONTINUE                                                          LINEAR0
C                                                                       LINEAR0
C                                    GET THE RESULTS AND OUTPUT THEM    LINEAR0
C                                                                       LINEAR0
      III1=(III-1)*NDIM+1                                               LINEAR0
      IF(INDEX.EQ.1) TCAT=XTRAIN(III1)                                  LINEAR0
      IF(INDEX.EQ.2) TCAT=DUMMY1                                        LINEAR0
C                                    WRITE RESULTS FOR THIS PATTERN VECTOR LINEAR1
      IF((INDEX.EQ.1).AND.(NOPT3.NE.0).AND.(KMAX.EQ.0)) WRITE(6,1501)IIILINEAR1
     1,TCAT                                                             LINEAR1
      IF((INDEX.EQ.2).AND.(NOPT4.NE.0).AND.(KMAX.EQ.0)) WRITE(6,1502)IIILINEAR1
     1,TCAT                                                             LINEAR1
      IF((INDEX.EQ.1).AND.(NOPT3.NE.0).AND.(KMAX.GT.0)) WRITE(6,1502)IIILINEAR1
     1,TCAT,NAME(ICLASS),KMAX                                           LINEAR1
      IF((INDEX.EQ.2).AND.(NOPT4.NE.0).AND.(KMAX.GT.0)) WRITE(6,1502)IIILINEAR1
     1,TCAT,NAME(ICLASS),KMAX                                           LINEAR1
 1501 FORMAT(6X,I3,11X,A5,10X,14HNOT CLASSIFIED)                        LINEAR1
 1502 FORMAT(6X,I3,11X,A5,10X,A5,12X,I3)                                LINEAR2
      DO 71 ITCAT=1,NC                                                  LINEAR2
      IF(TCAT.EQ.NAME(ITCAT))JTCAT=ITCAT                                LINEAR2
   71 CONTINUE                                                          LINEAR2
      ITCAT=JTCAT                                                       LINEAR2
C                                    UPDATE THE CONTINGENCY TABLE       LINEAR2
   72 IF(KMAX.GT.0) IERROR(ITCAT,ICLASS)=IERROR(ITCAT,ICLASS)+1         LINEAR2
   66 CONTINUE                                                          LINEAR2
C                                    WRITE OUT THE HEADING              LINEAR2
      IF(INDEX.EQ.1) WRITE(6,721)                                       LINEAR2
      IF(INDEX.EQ.2) WRITE(6,722)                                       LINEAR3
  721 FORMAT(1H1,15X,'CONTINGENCY TABLE FOR TRAINING PATTERNS')         LINEAR3
  722 FORMAT(1H1,15X,'CONTINGENCY TABLE FOR TEST PATTERNS')             LINEAR3
      WRITE(6,723)                                                      LINEAR3
  723 FORMAT(4X,'TRUE'/2X,'CATEGORY')                                   LINEAR3
C                                    WRITE OUT THE CONTINGENCY TABLE    LINEAR3
      DO 74 ITCAT=1,NC                                                  LINEAR3
   74 WRITE(6,724)  NAME(ITCAT),(IERROR(ITCAT,ICLS),ICLS=1,NC)          LINEAR3
  724 FORMAT(//3X,A6,15(4X,I3))                                         LINEAR3
C                                    LIST THE CATEGORIES                LINEAR3
      WRITE(6,725) (NAME(ICLA),ICLA=1,NC)                               LINEAR4
  725 FORMAT(//11X,15(1X,A6))                                           LINEAR4
      WRITE(6,726)                                                      LINEAR4
  726 FORMAT(/15X,'ASSISNED CATEGORY')                                  LINEAR4
   67 CONTINUE                                                          LINEAR4
      RETURN                                                            LINEAR4
      END                                                               LINEAR4
```

```
CWEIGHT              GET THE WEIGHT VECTOR
C
C       WRITTEN BY SAM SHANMUGAM                             SEPT 1972
C       DOCUMENTED BY RJ BOSLEY                              DEC  1973
C
C       DESCRIPTION OF PROGRAM.
C          GIVEN A MATRIX U CONTAINING PATTERNS OF CATEGORY I AND J,
C       THIS ROUTINE FINDS A HYPERPLANE TO SEPARATE THESE PATTERNS.  THE
C       WEIGHT VECTOR DEFINING THE HYPERPLANE IS GIVEN BY
C                     WT=((UU')**-1)*(U*T)
C       WHERE    WT          IS THE WEIGHT VECTOR
C                T           IS A COLUMN VECTOR WHOSE COMPONENTS ARE +1'S
C                ND          IS THE LENGTH OF THE PATTERN VECTORS, AND
C                NIJ         IS THE NUMBER OF PATTERNS FROM CAT I + NUMBER
C                            OF PATTERNS FROM CATEGORY J
C       THE DATA MATRIX U, USED FOR CALCULATING THE BOUNDARY BETWEEN THE
C       CATEGORIES I AND J, IS FORMED AS FOLLOWS ---
C
C       ***PATTERNS FROM CATEGORY I*********PATTERNS FROM CATEGORY J -***
C           1          1       .........          -1         -1      ....  -1
C           .          .       ..........          .          .      ....   .
C           .          .       ..........          .          .      ....   .
C           X          X       .........          -X         -X      ....  -X
C           .          .       ..........          .          .      ....   .
C           .          .       ..........          .          .      ....   .
C       ****** U HAS ND ROWS AND NIJ COLUMNS ******************************
C
C
C       ENTRY POINT.
C                 CALL WEIGHT(U,DUMMY,WT,ND,NIJ)
C
C       INPUT ARGUMENTS.
C                U                     MATRIX CONTAINING PATTERNS OF
C                                      CATEGORIES I AND J
C                DUMMY                 DUMMY ARRAY USED FOR MINV
C                ND                    LENGTH OF PATTERN VECTORS IN U
C                NIJ                   NUMBER OF PATTERNS FROM CAT I AND J
C
C       OUTPUT ARGUMENT.
C                WT                    WEIGHT VECTOR DEFINING THE HYPERPLANE
C                                      SEPARATING PATTERNS OF CAT I AND J
C
C       SUBPROGRAMS REQUIRED.
C                MINV                  MATRIX INVERSION PGM FROM IBM SSP
C
C       INTERNAL PARAMETERS.
C                L,M                   SCRATCH ARRAYS USED BY MINV
C                D                     DETERMINANT OF DUMMY,FROM MINV
C                IROWW                 ROW, OR COMPONENT,INDEX FOR WT
C                IDJD                  INDEX FOR A VECTOR IN DUMMY
C                IROWWK                INDEX FOR A VECTOR IN DUMMY AFTER IT
```

```
C                                  HAS BEEN INVERTED BY MINV
C
C
      SUBROUTINE WEIGHT(U,DUMMY,WT,ND,NIJ)
C
      DIMENSION U(1),DUMMY(1),WT(100)
      DIMENSION L(100),M(100)
C                                  SET UP MATRIX DUMMY=U*U'
C                                  GO THRU EACH COMPONENT
      DO 15 ID=1,ND
      DO 15 JD=1,ND
      SUM=0.0
C                                  GO THRU EACH VECTOR
      DO 14 K=1,NIJ
      IDK=(K-1)*ND+ID
      JDK=(K-1)*ND+JD
C                                  FIND U*U'
   14 SUM=SUM+U(IDK)*U(JDK)
      IDJD=(JD-1)*ND+ID
C                                  STORE IT IN DUMMY FOR INVERSION
      DUMMY(IDJD)=SUM
   15 CONTINUE
C                                  FIND THE INVERSE OF U*U'
      CALL MINV(DUMMY,ND,D,L,M)
      DO 30 K=1,ND
      SUM=0.0
      DO 25 KK=1,NIJ
C                                  COMPUTE U*T
      KKK=(KK-1)*ND+K
   25 SUM=SUM+U(KKK)
C                                  STORE IT IN U(K,1),K=1,ND
   30 U(K)=SUM
C                                  COMPUTE THE WEIGHT VECTOR
C                                   W(IROWW,NCOL),IROWW=1,ND
      DO 40 IROWW=1,ND
      SUM=0.0
      DO 35 K=1,ND
      IROWWK=(K-1)*ND+IROWW
   35 SUM=SUM+DUMMY(IROWWK)*U(K)
C                                  STORE IT IN WT ARRAY AND RETURN
   40 WT(IROWW)=SUM
      RETURN
      END
```

APPENDIX V

Pattern Recognition Pergamon Press 1973. Vol. 5, pp. 391–403. Printed in Great Britain

Glossary and Index to Remotely Sensed Image Pattern Recognition Concepts*

ROBERT M. HARALICK†

Remote Sensing Laboratory, University of Kansas, Lawrence, Kansas 66045, U.S.A.

(*Received* 29 *March* 1973 *and in revised form* 21 *June* 1973)

Abstract—The purpose of the glossary is to state in the simplest possible way the general meaning or word usage for many of the terms in image pattern recognition. There is no intent to provide definitive statements for terms such as "resolution" but rather only statements about the general nature of what resolution is. There is no intent to provide mathematical formulas involving integrals or derivatives in any of the statements. Those who need the mathematics can get it from technical papers or texts.

The glossary is designed to be read by those generally unfamiliar with the area and provide for them an overall perspective. The organization approaches that of programmed learning material and can be smoothly (I hope) read from beginning to end. Those needing to look up a specific term can do so via the index.

There is some overlap of terms in this glossary with those glossaries or definitions in radiometry and aerial photography. There is no intent that the way the terms are described here replace the way they are described in those glossaries and definitions. The overlap is provided here so that the reader can get a perspective of a cluster of terms frequently used in our field. The perspective is intended to start from what the image concept is through the recording of an image by some sensor, the possible conversion of image format and the simple analog or more complex digital processing which must be done on the imagery. In short, the perspective is one of image pattern recognition.

1. An *Image* is a spatial representation of an object, scene, or another image. It can be real or virtual as in optics. In pattern recognition, image usually means a recorded image such as a photograph, map, or picture. It may be abstractly thought of as a continuous function I of two variables defined on some bounded region of a plane. When the image is a photograph, the range of the function I is the set of grey shades usually considered to be normalized to the interval $[0, 1]$. The grey shade located at spatial coordinate (x, y) is denoted by $I(x, y)$ and is usually proportional to the radiant energy in the electromagnetic band to which the photographic sensor is sensitive. When the image is a map, the range of the function I is a set of symbols or colors, and the symbol or color located at spatial coordinate (x, y) is denoted by $I(x, y)$. A recorded image may be in photographic, video signal, or digital format.

2. The *grey shade* or *grey tone* is a number or value assigned to a position (x, y) on an image. The number is proportional to the integrated output, reflectance, or transmittance of a small area, usually called a resolution cell or pixel, centered on the position (x, y). The grey shade can be measured as or expressed in any one of the following ways:

(1) transmittance
(2) reflectance
(3) a coordinate of the ICI color coordinate system

* The glossary was prepared as the report from the definitions and standards subcommittee, Automatic Image Pattern Recognition Committee of Electronic Industries Association.

† I would like to acknowledge the helpful suggestions, comments and corrections which many of my colleagues made. Particular thanks are due to R. Asendorf, B. Bulluck, L. Kirvida, P. Pryor, A. Rosenfeld, B. Scheps, R. Swonger, S. Viglione and R. Zaputowycz.

(4) a coordinate of the tristimulus value color coordinate system
(5) brightness
(6) radiance
(7) luminance
(8) density
(9) voltage
(10) current.

3. A *photograph* is a "hard copy" pictorial record of an image formed by a sensor. The photograph is usually recorded on some type of phososensitive emulsion. It can be either reflective, as is a paper print, or transmissive, as is a transparency. It is usually two-dimensional and its reflectance or transmittance, (either monochromatic or polychromatic) varies as a function of position. If it is a multi-colored image (polychromatic), it can be either natural color where the colors are similar to the original, or false color where the colors of the photograph are radically different from the original. The sensor used to form the image may be any type such as an optical camera with or without spectral filtration, infrared optical–mechanical scanners, TV systems, radars, or sonic sensors, etc. The type of sensor recording the image and spectral region the sensor is sensitive to, should always be indicated when referring to a photograph.

4. A *map* is a representation, of physical and/or cultural features (natural, artificial or both) of a region (such as the sky) or a surface such as that of the earth or a planet. It indicates by a combination of symbols and colors those regions having designated category identifications. Very often ground truth and/or decision rule category assignments are displayed by maps. A photograph with limited symbolism and annotation is often called a *photo-map*.

5. The *radiant intensity* of a point object is a measure of the radiant power per steradian radiated or reflected by an object. In general, radiant intensity is a function of the nature of the object, the viewing angle, spectral wavelength and band-width.

6. The *reflectance or reflection coefficient* is the ratio of the energy per unit time per unit area (radiant power density) reflected by the object to the energy per unit time per unit area incident on the object. In general, reflectance is a function of the incident angle of the energy, viewing angle of the sensor, spectral wavelength and bandwidth, and the nature of the object.

7. The *transmittance* or *transmittance coefficient* is the ratio of the energy per unit time per unit area (radiant power density) transmitted through the object to the energy per unit time per unit area incident on the object. In general, transmittance is a function of the incident angle of the energy, viewing angle of the sensor, spectral wavelength and band-width, and the nature of the object.

8. The *density* of an (x, y) position on a photograph is a measure of the light absorbing capability of the silver or dye deposited on that position. It is defined by the logarithm of the position's reciprocal transmittance. The density measured should be specified as to whether it is specular or diffuse.

9. *Densitometry* is the field devoted to the measurement of optical image densities on film or print grey shades usually caused by the absorption or reflection of light by developed photographic emulsion.

10. A *densitometer* is a device used to measure the average image density of a small area of specified size on a photographic transparency or print. The measurement may be a

meter reading or an electronic signal. When the small area is smaller than a few hundred microns square, the instrument is called a *micro-densitometer*.

11. The *contrast* for a point object against its background can be measured by: (1) its *contrast ratio*, which is the ratio between the higher of object transmittance or background transmittance to the lower of object transmittance or background transmittance; (2) its *contrast difference*, which is the difference between the higher density of object or background to the lower density of object or background; (3) its *contrast modulation*, which is the difference between the darker of object or background grey shade and the lighter of object or background grey shade divided by the sum of object grey shade and background grey shade.

12. *Resolution* is a generic term which describes how well a system, process, component or material, or image can reproduce an isolated object or separate closely spaced objects or lines. The *limiting resolution*, *resolution limit* or *spatial resolution* is described in terms of the smallest dimension of the target or object that can just be discriminated or observed. Resolution may be a function of object contrast, spatial position as well as element shape (single point, number of points in a cluster, continuum, or line etc.).

13. The *resolving power* of an imaging system, process, component or material is a measure of its ability to image closely spaced objects. The most common practice in measuring resolving power is to image a resolving power target composed of lines and spaces of equal width. Resolving power is usually measured at the image plane in line pairs per millimeter, i.e. the greatest number of lines and spaces per millimeter that can just be recognized. This threshold is usually determined by using a series of targets of decreasing size and basing the measurement on the smallest one in which all lines can be counted. In measuring resolving power the nature of the target (number of lines and their aspect ratio), its contrast and the criteria for determining the limiting resolving power *must be* specified.

14. *Acutance* is a measure of the sharpness of edges in a photograph or image. It is defined for any edge by the average squared rate of change of the density across the edge divided by the total density difference from one side of the edge to the other side of the edge.

15. The *spread function* of an image system, process, component, or material describes the resulting spatial distribution of grey shade when the input to the system is some well defined object much smaller than the width of the spread function. If the input to the system is a line, the spread function is called the *line spread function*. If the input to the system is a point, the spread function is called the *point spread function*.

16. The *Modulation Transfer Function* of an imaging system or component measures the spatial frequency modulation response of the system or component. As an imaging system or component processes or records an image, the contrast modulation of the processed or recorded image is different from the input image. In fact, there is always a spatial frequency beyond which the contrast modulation of the processed or recorded (output) image is smaller (worse) than the contrast modulation of the input image. The modulation transfer function can be thought of as a curve indicating, for each spatial frequency, the ratio of the contrast modulation of the output image to the contrast modulation of the input image. It is formally defined as the magnitude of the Fourier transform of the line spread function of the imaging system or component.

17. A *resolution cell* is the smallest most elementary areal constituent of grey shades considered by an investigator in an image. A resolution cell is referenced by its spatial coordinates. The resolution cell or formations of resolution cells can sometimes constitute the basic unit for pattern recognition of image format data.

18. A *digital image*, or *digitized image*, or *digital picture function* of an image is an image in digital format and is obtained by partitioning the area of the image into a finite two-dimensional array of small uniformly shaped mutually exclusive regions, called resolution cells, and assigning a "representative" grey shade to each such spatial region. A digital image may be abstractly thought of as a function whose domain is the finite two-dimensional set of resolution cells and whose range is the set of grey shades.

19. A *picture element* or *pixel* or *pel* is a pair whose first member is a resolution cell and whose second member is the grey shade assigned by the digital image to that resolution cell. Sometimes picture element, pixel, or pel refer only to the grey shade or grey shade n-tuple in a resolution cell.

20. A *multi-image* is a set of images, each taken of the same subject at different times, or from different positions, or with different sensors, or at different electromagnetic frequencies, or with different polarizations. Although there is a high degree of information redundancy between images in a multi-image set, each image usually has information not available in any one of or combinations of the other images in the set.

21. A *multi-digital image* is a multi-image in digital form. It can be, for example, a set of digital images obtained from the images in a multi-image. A multi-digital image is often called a multi-image for short when it is understood from context that digital images are involved.

22. A *flying spot scanner* is a device used to rapidly convert image data from photographic format to electronic video signal format. Normally, the scanner directs an electron beam across the face of a cathode ray tube (CRT) in a TV-like raster. The photographic transparency is placed in front of the CRT (either directly or through some optics) and the light coming from the CRT is passed through it. The modulated light beam is detected by a photomultiplier or other photo detector and amplified to a usable video signal level.

23. A *scanning densitometer* is a device used to convert image data from transparency photographic format to electronic video signal format. Usually, the photographic transparency is placed on a glass cylinder which rotates and slowly translates. A fine beam of light is focused on the transparency, passed through it, and is detected by a photo-multiplier where it is amplified to a usable video signal. The scanning densitometer is a much slower conversion device than the flying spot scanner. However, this disadvantage is compensated by its fine resolution capability of a few microns.

24. The *vidicon* is an imaging vacuum tube having a photosensitive surface and is a means of converting image data from instantaneous radiance format to electronic video signal format. The scene being viewed is imaged on the photosensitive surface which can be scanned by an electron beam generating a signal whose amplitude corresponds to the radiant intensity focused on the surface at each point. This signal is called a *video signal* and may be amplified to any desired level.

25. A *video image* is an image in electronic signal format capable of being displayed on a cathode ray tube screen. The video signal is generated from devices like a vidicon or flying spot scanner which converts an image from photographic form to video signal form by scanning it line by line. The video signal itself is a sequence of signals, the ith signal representing the ith line of the scanned image.

26. *Registering* is the translation–rotation alignment process by which two images of like geometries and of the same set of objects are positioned coincident with respect to one another so that corresponding elements of the same ground area appear in the same place on the registered images. In this manner, the corresponding grey shades of the two images

at any (x, y) coordinate or resolution cell will represent the sensor output for the same object over the full image frame being registered.

27. *Congruencing* is the process by which two images of a multi-image set are transformed so that the size and shape of any object on one image is the same as the size and shape of that object on the other image. In other words, when two images are congruenced, their geometries are the same and they coincide exactly.

28. *Rectifying* is a process by which the geometry of an image area is made planimetric. For example, if the image is taken of an equally spaced rectangular grid pattern, then the rectified image will be an image of an equally spaced rectangular grid pattern. Rectification does not remove relief distortion.

29. *Change detection* is the process by which two images may be compared, resolution cell by resolution cell, and an output generated whenever corresponding resolution cells have different enough grey shades or grey shade n-tuples.

30. An *optical color combiner* is an instrument which produces "false" or "true" color images by linearly combining a few black and white transparencies of the same scene. The transparencies are usually obtained from multi-spectral, multi-band, or time-sequential photography. The transparencies are placed in projectors which are all focused and registered on the same screen and which have various color filters placed in front of their lenses. The viewing brightness of the projector's lamp in each projector can be changed independently thereby changing chromaticity balance. An optical color combiner is sometimes called an *additive color display.*

31. An *electronic color combiner* is an instrument which produces a "false" color image by linearly combining video signals of images of the same scenes. The images are usually obtained from multi-spectral, multi-band, or time-sequential photography. If the original image format is photographic, then the image format is changed from photographic to video signal format by synchronized vidicons or flying spot scanners. The resulting video signals are linearly combined through a matrix multiplier circuit, and the three linearly combined signals then drive the color gun of a color TV tube. An electronic color combiner usually has greater versatility for congruencing or registering than an optical color combiner.

32. *Level slicing* or *density slicing* or *thresholding* is an operation performed by an instrument (usually electronic) called a level slicer to change one or more a grey scale images to one binary image.

33. The *level slicer, density slicer* or *thresholder* is an instrument (usually electronic) which takes a single or multi-image as an input and produces a binary image for an output. A binary "one" is produced on the output image whenever the grey shades on each of the input images lie within the independently set minimum and maximum thresholds. A set of N input images would, therefore, require a setting N minimum and N maximum levels.

34. A *figure F*, or a *subimage F* in a continuous or digital image I is any function F whose domain is some subset A of the set of spatial coordinates or resolution cells, whose range is the set G of grey shades, and which is defined by $F(x, y)=I(x, y)$ for any (x, y) belonging to A.

35. A figure F is *connected* if there is a path between any two spatial coordinates or resolution cells contained in the domain of F. More precisely, F is connected if for each pair of spatial coordinates (x, y) and (u, v) belonging to the domain of F, there exists some sequence $\langle(a_1, b_1), (a_2, b_2), \ldots, (a_m, b_m)\rangle$ of spatial coordinates belonging to the domain

of F such that $(x, y)=(a_1, b_1), (u, v)=(a_m, b_m)$, and (a_i, b_i) and (a_{i+1}, b_{i+1}) are sufficiently close neighboring coordinates, $i = 1, 2, \ldots, m-1$.

36. A figure F is *convex* if the domain of F contains the line segment which joins any pair of spatial coordinates in the domain of F.

37. A *discrete tonal feature* on a continuous or digital image is a connected set of spatial coordinates or resolution cells all of which have the same or almost the same grey shade.

38. *Texture* is concerned with the spatial distribution of the grey shades and discrete tonal features. When a small area of the image has little variation of discrete tonal features, the dominant property of that area is grey shade. When a small area has wide variation of discrete tonal features, the dominant property of that area is texture. There are three things crucial in this distinction: (1) the size of the small areas, (2) the relative sizes of the discrete tonal features, and (3) the number of distinguishable discrete tonal features.

39. *Quantizing* is the process by which each grey shade in an image of photographic, video, or digital format is assigned a new value from a given finite set of grey shade values. There are three often used methods of quantizing:

(1) in *equal interval quantizing* or *linear quantizing*, the range of grey shades from maximum grey shade to minimum grey shade is divided into contiguous intervals each of equal length, and each grey shade is assigned to the quantized class which corresponds to the interval within which it lies;

(2) in *equal probability quantizing*, the range of grey shades is divided into contiguous intervals such that after the grey shades are assigned to their quantized class there is an equal frequency of occurrence for each quantized grey shade in the quantized digital image or photograph; equal probability quantizing is sometimes called *central stretching*;

(3) in *minimum variance quantizing*, the range of grey shades is divided into contiguous intervals such that the weighted sum of the variance of the quantized intervals is minimized. The weights are usually chosen to be the grey shade interval probabilities which are computed as the proportional area on the photograph or digital image which have grey shades in the given interval.

40. A *quantizer* is an instrument which does quantizing. The quantizer has three functional parts. The first part allows the determining and/or setting of the quantizing intervals, the second part is a level slicer which indicates when a signal is in any quantizing interval, and the third part takes the binary output from the level slicers and either codes it to some binary code or converts it to some analog signal representing quantizing interval centers or means.

41. The simplest and most practical *unit* to observe and measure in the pattern recognition of image data is often the basic picture element (the grey shade or the grey shade n-tuple in its particular resolution cell). This is what makes pattern recognition so hard sometimes for the objects requiring analysis or identification are not simple picture elements but are often complex spatial formations of picture elements such as houses, roads, forest, etc.

42. A *measurement n-tuple* or *measurement pattern* or *pattern* or *measurement vector* is the ordered n-tuple of measurements obtained of a unit under observation. Each component of the n-tuple is a measurement of a particular quality, property, feature, or characteristic of the unit. In image pattern recognition, the units are usually picture elements or simple formations of picture elements and the measurement n-tuples are the corresponding grey shades, grey shade n-tuples, or formations of grey shade n-tuples.

43. The *range set* R_i for the ith sensor which produces the ith image in the multi-image set, is the set of all measurements which can be produced by the ith sensor. Simply, it is the set of all grey shades which could possibly exist on the ith image.

44. The *Cartesian product* of two sets A and B, denoted by $A \times B$, is the set of all ordered pairs where the first component of the pair is some element from the first set and the second component of the pair is some element from the second set. The Cartesian product of N sets can be inductively defined in the usual fashion.

45. *Measurement space* is a set large enough to include in it the set of all possible measurement n-tuples which could be obtained by observing physical attributes of some set of units. When the units are single resolution cells or picture elements, measurement space M is the Cartesian product of the range sets of the sensors; $M = R_1 \times R_2 \times \cdots \times R_n$.

46. Each unit is assumed to be of one and only one given type. The set of types is called the set of *pattern classes* or *categories C*, each type being a particular category. The categories are chosen specifically by the investigator as being the ones of interest to him.

47. A *feature* or *feature pattern* or *feature n-tuple* or *pattern feature* is a n-tuple or vector with (a small number of) components which are functions of the initial measurement pattern variables or some subsequence of the measurement n-tuples. Feature n-tuples or vectors are designed to contain a high amount of information relative to the discrimination between units of the types of categories in the given category set. Sometimes the features are predetermined and other times they are determined at the time the pattern discrimination problem is being solved. In image pattern recognition, features often contain information relative to grey shade, texture, shape or context.

48. *Feature space* is the set of all possible feature n-tuples.

49. *Feature selection* is the process by which the features to be used in the pattern recognition problem are determined. Sometimes feature selection is called *property selection.*

50. *Feature extraction* is the process in which an initial measurement pattern or some subsequence of measurement patterns is transformed to a new pattern feature. Sometimes feature extraction is called *property extraction.*

51. The word *pattern* can be used in three distinct senses:

(1) as measurement pattern;

(2) as feature pattern; and

(3) as the dependency pattern or patterns of relationships among the components of any measurement n-tuple or feature n-tuple derived from units of a particular category and which are unique to those n-tuples, that is, they are dependencies which do not occur in any other category.

52. A *signature* is the observable or characteristic measurement or feature pattern derived from units of a particular category. A category is said to have a signature only if the characteristic pattern is highly representative of the n-tuples obtained from units of that category. Sometimes a signature is called a *prototype pattern.*

53. A *data sequence* $S_d = \langle d_1, d_2, \ldots, d_J \rangle$ is a sequence of patterns derived from the measurement patterns or features of some sequence of observed units. d_1 is the pattern associated with the first unit; d_2 is the pattern associated with the second unit; and d_J is the pattern associated with the Jth unit.

54. A *decision rule f* usually assigns one and only one category to each observed unit on the basis of the sequence of measurement patterns in the data sequence S_d or in the corresponding sequence of feature patterns.

55. A *simple decision rule* is a decision rule which assigns a unit to a category solely on the basis of the measurements or features associated with the unit. Hence, the units are treated independently and the decision rule f may be thought of as a function which assigns one and only one category to each pattern in measurement space or to each feature in feature space.

56. A *compound decision rule* is a decision rule which assigns a unit to a category on the basis of some non-trivial subsequence of measurement patterns in the data sequence or in the corresponding sequence of feature patterns. A compound decision rule is not a simple decision rule.

57. Provision can be made for the decision rule to *reserve judgement* or to *defer assignment* if the pattern is too close to the category boundary in measurement or feature space. With this provision, a deferred assignment is an assignment to the category of "reserved judgement."

58. A *category identification sequence* or *ground truth* $S_c = \langle c_1, c_2, \ldots, c_J \rangle$ is a sequence of category identifications obtained from some sequence of observed units. c_1 is the category identification of the first unit; c_2 is the category identification of the second unit; and c_J is the category identification of the Jth unit.

59. A *training sequence* is a set of two sequences: (1) the data sequence and (2) a corresponding category identification sequence (sometimes called ground truth). The training sequence is used to estimate the category conditional probability distributions from which the decision rule is constructed.

60. The *conditional probability* of a measurement or feature n-tuple d given category c is denoted by $P_c(d)$, or by $P(d/c)$, and is defined as the relative frequency or proportion of times the n-tuple d is derived from a unit whose true category identification is c.

61. A *distribution-free* or *non-parametric decision rule* is one which makes no assumptions about the functional form of the conditional probability distribution of the patterns given the categories.

62. A simple *maximum likelihood decision rule* is one which treats the units independently and assigns a unit u having pattern measurement or features d to that category c whose units are most probable to have given rise to pattern or feature vector d, that is, such that the conditional probability of d given c, $P_c(d)$, is highest.

63. A simple *Bayes decision rule* is one which treats the units independently and assigns a unit u having pattern measurements or features d to the category c whose conditional probability, $P_d(c)$, given measurement d, is highest.

64. Let $\langle u_1, u_2, \ldots, u_J \rangle$ be a sequence of units with corresponding data sequence $\langle d_1, d_2, \ldots, d_J \rangle$ and known category identification sequence $\langle c_1, c_2, \ldots, c_J \rangle$. A simple *nearest neighbor decision rule* is one which treats the units independently and assigns a unit u of unknown identification and with pattern measurements or features d to category c_j where d_j is that pattern closest to d by some given metric or distance function.

65. A *discriminant function* $f_i(d)$ is a scalar function, whose domain is usually measurement space and whose range is usually the real numbers. When $f_i(d) \geq f_k(d)$, $k = 1, 2, \ldots, K$, then the decision rule assigns the ith category to the unit giving rise to pattern d.

66. A *linear discriminant function* f is a discriminant function of the form

$$f(d) = b + \sum_{j=1}^{n} a_j \delta_j \text{ where } d = (\delta_1, \delta_2, \ldots, \delta_n).$$

67. A *decision boundary* between the ith and kth categories is a subset H of patterns in measurement space M defined by

$$H = \{d \in M | f_i(d) = f_k(d)\},$$

where f_i and f_k are the discriminant functions for the ith and kth categories.

68. A *hyperplane decision boundary* is the special name given to decision boundaries arising from the use of linear discriminant functions.

69. A *linear decision rule* is a simple decision rule which usually treats the units independently and makes the category assignments using linear discriminant functions. The decision boundaries obtained from linear decision rules are hyperplanes.

70. The *pattern discrimination* problem is concerned with how to construct the decision rule which assigns a unit to a particular category on the basis of the measurement pattern(s) in the data sequence or on the basis of the feature pattern(s) in the data sequence.

71. *Pattern identification* is the process in which a decision rule is applied. If $S_u = \langle u_1, u_2, \ldots, u_J \rangle$ is the sequence of units to be observed and identified, and if $S_d = \langle d_1, d_2, \ldots, d_J \rangle$ is the corresponding data sequence of patterns, then the pattern identification process produces a category identification sequence $S_c = \langle c_1, c_2, \ldots, c_J \rangle$ where c_i is the category in C to which the decision rule assigns unit u_i on the basis of the J patterns in S_d. In general, each category in S_c can be assigned by the decision rule as a function of all the patterns in S_d. Sometimes pattern identification is called "*pattern classification*" or "*classification*".

72. A *cluster* is a homogeneous group of units which are very "like" one another. "Likeness" between units is usually determined by the association, similarity, or distance between the measurement patterns associated with the units.

73. A *cluster assignment function* is a function which assigns each observed unit to a cluster on the basis of the measurement pattern(s) in the data sequence or on the basis of their corresponding features. Sometimes the units are treated independently; in this case the clustering assignment function can be considered as a transformation from measurement space to the set of clusters.

74. The *pattern classification* problem is concerned with constructing the cluster assignment function which groups similar units. Pattern classification is synonymous with *numerical taxonomy* or *clustering*.

75. The *cluster identification* process is the process in which the cluster assignment function is applied to the sequence of observed units thereby yielding a cluster identification sequence.

76. A *misidentification*, or *misdetection*, or *type I error* occurs for category c_i if a unit whose true category identification is c_i is assigned by the decision rule to category c_k, $k \neq i$. A misidentification error is often called an *error of omission*.

77. A *false identification*, or *false alarm*, or *type II error* occurs for category c_i if a unit whose true category identification is c_k, $k \neq i$, is assigned by the decision rule to category c_i. A false identification error is often called an *error of commission*.

78. A *prediction sequence*, or *test sequence*, or a *generalization sequence* is a set of two sequences: (1) a data sequence (whose corresponding true category identification sequence may be considered to be unknown to the decision rule) and (2) a corresponding category identification sequence determined by the decision rule assignment. By comparing the category identification sequence determined by the decision rule assignment with the

category identification sequence determined by the ground truth, the misidentification rate and the false identification rate for each category may be estimated.

79. A *confusion matrix* or *contingency table* is an array of probabilities whose rows and columns are both similarly designated by category label and which indicates the probability of correct identification for each category as well as the probability of type I and type II errors. The (ith, kth) element P_{ik} is the probability that a unit has true category identification c_i; and is assigned by the decision rule to category c_k.

80. A unit is said to be *detected* if the decision rule is able to assign it as belonging only to some given subset A of categories from the set C of categories. To detect a unit does not imply that the decision rule is able to identify the unit as specifically belonging to one particular category.

81. A unit is said to be *recognized, identified, classified, categorized* or *sorted* if the decision rule is able to assign it to some category from the set of given categories. In military applications, there is a definite distinction between recognize and identify. Here, for a unit to be recognized, the decision rule must be able to assign it to a type of category, the type having included within it many subcategories. For a unit to be identified, the decision rule must be able to assign it not only to a type of category but also to the subcategory of the category type. For example, a small area ground patch may be recognized as containing trees, which may be specifically identified as apple trees.

82. A unit is said to be *located* if specific coordinates can be given for the units physical location.

83. A unit is said to be *acquired* if it can be located and recognized.

84. A *target* is one type of category used in the pattern recognition of image data. It usually occupies some relatively small area on the image and has a unique or characteristic set of attributes. It has a high *a priori* interest to the investigator.

85. *Target discrimination* is the process by which decision rules for targets (small area extensive categories) are constructed.

86. *Target identification* or *target recognition* is the process by which targets contained within image data are identified by means of a decision rule.

87. An *image transformation* is a function or operator which takes an image for its input and produces an image for its output. The domain of the transform operator is often called the spatial domain. The range of the transform operator is often called the transformed domain. Some transformations have spatial and transform domains of entirely different character. For these transforms, the image in the spatial domain may appear entirely different from and have a different interpretation from the image in the transformed domain. Specific examples of these kinds of transformations are the Fourier, Hadamard, and Karhunen–Loéve transformations. Other transformations have spatial and transform domain of similar character. For these transformations, the image in the transformed domain may appear similar to the image of the spatial domain. These types of transformations are often called spatial filters.

88. A *spatial filter* is an image transformation, usually a one-one operator used to lessen noise or enhance certain characteristics of the image. For any particular (x, y) coordinate on the transformed image, the spatial filter assigns a grey shade on the basis of the grey shades of a particular spatial pattern near the coordinates (x, y).

89. A *linear spatial filter* is a spatial filter for which the grey shade assignment at coordinates (x, y) in the transformed image is made by some weighted average (linear combination) of grey shades located in a particular spatial pattern around coordinates (x, y) of the domain image. The linear spatial filter is often used to change the spatial

frequency characteristics of the image. For example, a linear spatial filter which emphasizes high spatial frequencies will tend to sharpen the edges in an image. A linear spatial filter which emphasizes the low spatial frequencies will tend to blur the image and reduce salt and pepper noise.

90. *Template matching* is an operation which can be used to find out how well two photographs or images match one another. The degree of matching is often determined by cross-correlating the two images or by evaluating the sum of the squared corresponding grey shade differences. Template matching can also be used to best match a measurement pattern with a prototype pattern.

91. *Matched filtering* is a template matching operation done by using the magnitude of the cross-correlation function to measure the degree of matching.

92. In pattern recognition problems such as target discrimination, for which the category of interest is some specific formation of resolution cells with characteristic shape or tone–texture composition, the problem of pattern segmentation may occur. *Pattern segmentation* is the problem of determining which regions or areas in the image constitute the patterns of interest, i.e. which resolution cells should be included and which excluded from the pattern measurements.

93. *Screening* is the operation of separating the uninteresting photographs or images from those photographs containing areas of potential interest.

94. *Preprocessing* is an operation applied before pattern identification is performed. Preprocessing produces, for the categories of interest, pattern features which tend to be invariant under changes such as translation, rotation, scale, illumination levels, and noise. In essence, preprocessing converts the measurements patterns to a form which allows a simplification in the decision rule. Preprocessing can bring into registration, bring into congruence, remove noise, enhance images, segment target patterns, detect, center, and normalize targets of interest.

95. *Image compression* is an operation which preserves all or most of the information in the image and which reduces the amount of memory needed to store an image or the time needed to transmit an image.

96. *Image restoration* is a process by which a degraded image is restored to its original condition. Image restoration is possible only to the extent that the degradation transform is mathematically invertible.

97. *Image enhancement* is any one of a group of operations which improve the detectability of the targets or categories. These operations include, but are not limited to, contrast improvement, edge enhancement, spatial filtering, noise suppression, image smoothing, and image sharpening.

98. *Image processing* encompasses all the various operations which can be applied to photographic or image data. These include, but are not limited to, image compression, image restoration, image enhancement, preprocessing, quantization, spatial filtering, and other image pattern recognition techniques.

99. *Interactive Image Processing* refers to the use of an operator or analyst at a console with a means of assessing, preprocessing, feature extracting, classifying, identifying and displaying the original imagery or the processed imagery for his subjective evaluation and further interactions.

100. *Pattern recognition* is concerned with, but not limited to, problems of: (1) pattern discrimination, (2) pattern classification, (3) feature selection, (4) pattern identification, (5) cluster identification, (6) feature extraction, (7) preprocessing, (8) filtering, (9) enhancement, (10) pattern segmentation, or (11) screening.

INDEX TO GLOSSARY OF REMOTELY SENSED IMAGE PATTERN RECOGNITION CONCEPTS

REFERENCES

Afarani, M. K., Pattern Recognition Using Discrete Probability Distributions, Ph.D. Dissertation, University of Kansas, October, 1972.

Anderson, J. R., E. E. Hardy, and J. T. Roach, "A Land-Use Classification System for Use with Remote-Sensor Data," Geological Survey Circular 671, U. S. Geological Survey, Washington, D. C., 1973.

Cutrona, L. J., E. N. Leith, C. J. Palermo, and L. J. Porcello, "Optical Data Processing and Filtering Systems," IRE Transactions on Information Theory, vol. IT-6, no. 3, pp. 318-400, June, 1960.

Darling, E. M., and R. D. Joseph, "Pattern Recognition from Satellite Altitudes," IEEE Transactions on Systems, Man, and Cybernetics, vol. SSC-4, pp. 38-47, March, 1968.

Egbert, D., J. McCauley, and J. McNaughton, "Ground Pattern Analysis in the Great Plains," Semi-Annual ERTS A Investigation Report, Remote Sensing Laboratory, The University of Kansas Center for Research, Inc., Lawrence, Kansas, August, 1973.

Fu, K. S., and J. M. Mendel, "Adaptive Learning and Pattern Recognition Systems," Academic Press, New York, 1972.

Fukunaga, K., "Introduction to Statistical Pattern Recognition," Academic Press, New York, 1972.

Goodman, J. W., "Introduction to Fourier Optics, McGraw-Hill Book Co., New York, 1968.

Gramenopoulos, Nicholas, "Terrain Type Recognition Using ERTS-1 MSS Images," Symposium on Significant Results Obtained From the Earth Resources Technology Satellite, NASA SP-327, pp. 1229-1241, March, 1973.

Haralick, R. M., et al, "Preliminary Report on Land Use Classification Using Texture Information in ERTS-A MSS Imagery," Report No. 2262-1, NASA Goddard Space Flight Center Contract NAS 5-21822, The University of Kansas Center for Research, Inc., Lawrence, Kansas January, 1973.

Haralick, R. M., et al, "Texture-Tone Study with Application to Digitized Imagery," U. S. Army Engineer Topographic Laboratories Contract DAAK02-70-C-0388, CRES Technical Reports 182-2, and 182-3, The University of Kansas Center for Research, Inc., Lawrence, Kansas 1971, 1972.

Haralick, R. M., "A Texture-Context Feature Extraction Algorithm for Remotely Sensed Imagery," Proceedings 1971 IEEE Decision and Control Conference, Gainesville, Florida, pp. 650-657, December, 15-17, 1971.

Haralick, R. M., and D. E. Anderson, "Texture-Tone Study with Application to Digitized Imagery," CRES Technical Report 182-2, The University of Kansas Center for Research, Inc., Lawrence, Kansas, November, 1971.

Haralick, R. M., K. Shanmugam, and I. Dinstein, "Texture Features for Image Classification," IEEE Transactions on Systems, Man, and Cybernetics, vol. SMC-3, pp. 610-621, November, 1973.

Hardy, E. E., and J. R. Anderson, "A Land Use Classification System for Use with Remote-Sensor Data," Machine Processing of Remotely Sensed Data, (Conference Proceedings), Lafayette, Indiana, October, 1973.

Horning, R. J., and J. A. Smith, "Application of Fourier Analysis to Multispectral/ Spatial Recognition," Management and Utilization of Remote Sensing Data ASP Symposium, Sioux Falls, South Dakota, October, 1973.

Kaizer, H., "A Quantification of Textures on Aerial Photographs," Boston University Research Laboratories, Technical Note 121, 1965, AD 69484.

Kirvida, L., and G. Johnson, "Automatic Interpretation of ERTS Data for Forest Management," Symposium on Significant Results Obtained from the Earth Resources Technology Satellite, NASA SP-327, pp. 1076-1082, March, 1973.

Kullbach, S., Information Theory and Statistics, Wiley, 1957.

Lendaris, G. G., and G. L. Stanley, "Diffraction-Pattern Sampling for Automatic Pattern Recognition," Proceedings of the IEEE, vol. 58, no. 2, pp. 198-216, February, 1970.

Lewis, A. J., "Geomorphic Evaluation of Radar Imagery of Southeastern Panama and Northwestern Columbia," CRES Technical Report 133-18, The University of Kansas Center for Research, Inc., Lawrence, Kansas, February, 1971.

MacDonald, H. C., "Geologic Evaluation of Radar Imagery from Darien Province, Panama," CRES Technical Report 133-6, The University of Kansas Center for Research, Inc., Lawrence, Kansas, July, 1969.

Matherson, G., "Elements Pour Une Theorie des Milieux Poreaux, Masson, Paris, 1967.

Miesel, W., "Computer Oriented Approaches to Pattern Recognition," Academic Press, New York, 1972.

O'Neill, E., "Spatial Filtering in Optics," IRE Transactions on Information Theory, vol. IT-2, no. 2 , pp. 56-65, June, 1956.

Preston, K., "Coherent Optical Computers, McGraw-Hill Book Co., New York, 1972.

Read, J. S., and S. N. Jayaramamurthy, "Automatic Generation of Texture Feature Detectors," IEEE Transactions on Computers, vol. C-21, no. 7, pp. 803-812, July, 1972.

Rosenfeld, A., and M. Thurston, "Edge and Curve Detection for Visual Scene Analysis," IEEE Transactions on Computers, vol. C-20, no. 5, pp. 562-569, May, 1971.

Rosenfeld, A., and E. Troy, "Visual Textural Analysis," University of Maryland Computer Science Center, TR-70-116, June, 1970.

Serra, J., and G. Verchery, "Mathematical Morphology Applied to Fibre Composite Materials," Film Science and Technology, vol. 6, pp. 141-158, 1973.

Shulman, A. R., Optical Data Processing, John Wiley & Sons, Inc., New York, 1970.

Sutton, R., and E. Hall, "Texture Measures for Automatic Classification of Pulmonary Disease," IEEE Transactions on Computers, vol. C-21, no. 7, pp. 667-676, July, 1972.

Swanlund, G., "Honeywell's Automatic Tree Species Classifier," Honeywell Systems and Research Division, Report 9D-G-24, December 31, 1969.

Yaglom, A. M., Theory of Stationary Random Functions, Prentice-Hall, Inc., New Jersey, 1962.