NASA CONTRACTOR
REPORT

NASA CR-149998

DIGITAL COMPUTER PROCESSING OF PEACH ORCHARD MULTISPECTRAL AERIAL PHOTOGRAPHY

By Robert J. Atkinson
Computer Sciences Corporation
8300 Whitesburg Drive, South
Huntsville, Alabama 35802

October 1976

Prepared for

NASA-George C. Marshall Space Flight Center
Marshall Space Flight Center, Alabama 35812

TECHNICAL REPORT STANDARD TITLE PAGE

1 REPORT NO	2 GOVERNMENT ACCESSION NO	3 RECIPIENT'S CATALOG NO.
NASA CR-149998		

4 TITLE AND SUBTITLE	5 REPORT DATE
Digital Computer Processing of Peach Orchard Multispectral Aerial Photography	October 1976
	6 PERFORMING ORGANIZATION CODE
7 AUTHOR(S)	8 PERFORMING ORGANIZATION REPORT #
Robert J Atkinson	
9 PERFORMING ORGANIZATION NAME AND ADDRESS	10 WORK UNIT NO.
Computer Science Corporation 8300 Whitesburg Drive Huntsville, Alabama 35802	11 CONTRACT OR GRANT NO. NAS8-21805
	13 TYPE OF REPORT & PERIOD COVERED
12 SPONSORING AGENCY NAME AND ADDRESS	Contractor Report
National Aeronautics and Space Administration Washington, D. C. 20546	14 SPONSORING AGENCY CODE

15 SUPPLEMENTARY NOTES

16 ABSTRACT

Several methods of analysis by digital computer which may be applied to digitized multispectral aerial photography are described, with particular application to peach orchard test sites. This effort was stimulated by the recent premature death of peach trees in the Southeastern United States The techniques discussed are· (i) correction of intensity variations by digital filtering, (ii) automatic detection and enumeration of trees in five size categories, (iii) determination of unhealthy foliage by infrared reflectances, and (iv) four band multispectral classification into healthy and declining categories.

17 KEY WORDS	18 DISTRIBUTION STATEMENT
digital filtering, object detection, multispectral classification	Unclassified-Unlimited Sanford Downs, COR Data Systems Laboratory, MSFC

19 SECURITY CLASSIF (of this report)	20 SECURITY CLASSIF (of this page)	21 NO OF PAGES	22 PRICE
Unclassified	Unclassified	103	NTIS

For sale by National Technical Information Service, Springfield, Virginia 22151

ACKNOWLEDGEMENTS

Gratitude is expressed to personnel of the Southeastern Fruit and Tree Nut Research Station of the United States Department of Agriculture, and of the Environmental Applications Office of NASA/ Marshall Space Flight Center, and to Dr A. D. Bond of Computer Sciences Corporation

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF ILLUSTRATIONS (continued)

LIST OF ILLUSTRATIONS (continued)

I. INTRODUCTION

The peach crop is a major contributor to the agricultural output of the Southeastern United States, particularly the state of Georgia. However, the intensive culture of peaches over a long period of time leads to the so-called "peach decline". Over the last few decades, the average productive life of peach trees in Georgia has declined from approximately twenty years to from five to eight years.

Peach decline has been investigated in Georgia since 1929 The premature death of peach trees in the Southeast has been attributed to such disorders as root rot, peachtree borer, nematodes, bacterial canker, fungi, cold damage, and virus diseases.

In 1972, between 100,000 and 200,000 trees died, and hence in that year aerial photography of several orchard test sites was obtained. Previously, the occurrence of tree decline due to the presence of three diseases, three insects, and one mite had been detected by the examination of color infrared aerial photographs obtained at elevations of 600-4500 feet.[1]

Color infrared photographs obtained in 1972 were analyzed using the General Electric Image 100 System[2] By displaying on a color television monitor, categories corresponding to selected density levels on the film are readily located. With this method, trees in various stages of decline and also different segments of the same tree fall into different categories.

Consequently, analysis by digital computer of multispectral aerial photography was undertaken. The purposes of this study were to determine the feasibilities of orchard inventories (enumeration and sizing of the crowns) and of determining the state of decline of trees based on an analysis of their multispectral reflectances.

This report presents the results of the analysis of four band multispectral photography by digital computer The major topics presented are (i) digital

correction of intensity variations, (ii) enumeration of trees by sizes, and (iii) classification into healthy and declining categories.

The aerial photography was undertaken by NASA/MSFC, and ground-truth data was furnished by the USDA Southeastern Fruit and Tree Nut Research Station, Byron, Georgia. The aerial imagery was obtained over the USDA test sites in Bibb, Crawford and Peach Counties in the state of Georgia The locations of these test sites and the USDA Research Station are shown in Figure 1.1. The photographs used were made by the MSFC I^2S four band camera system, which exposes four frames of film size 90x90 mm. through filters which transmit in the blue, green, red and infrared spectral bands. Digitization of the images and computer analysis were performed at Marshall Space Flight Center by Computer Sciences Corporation.

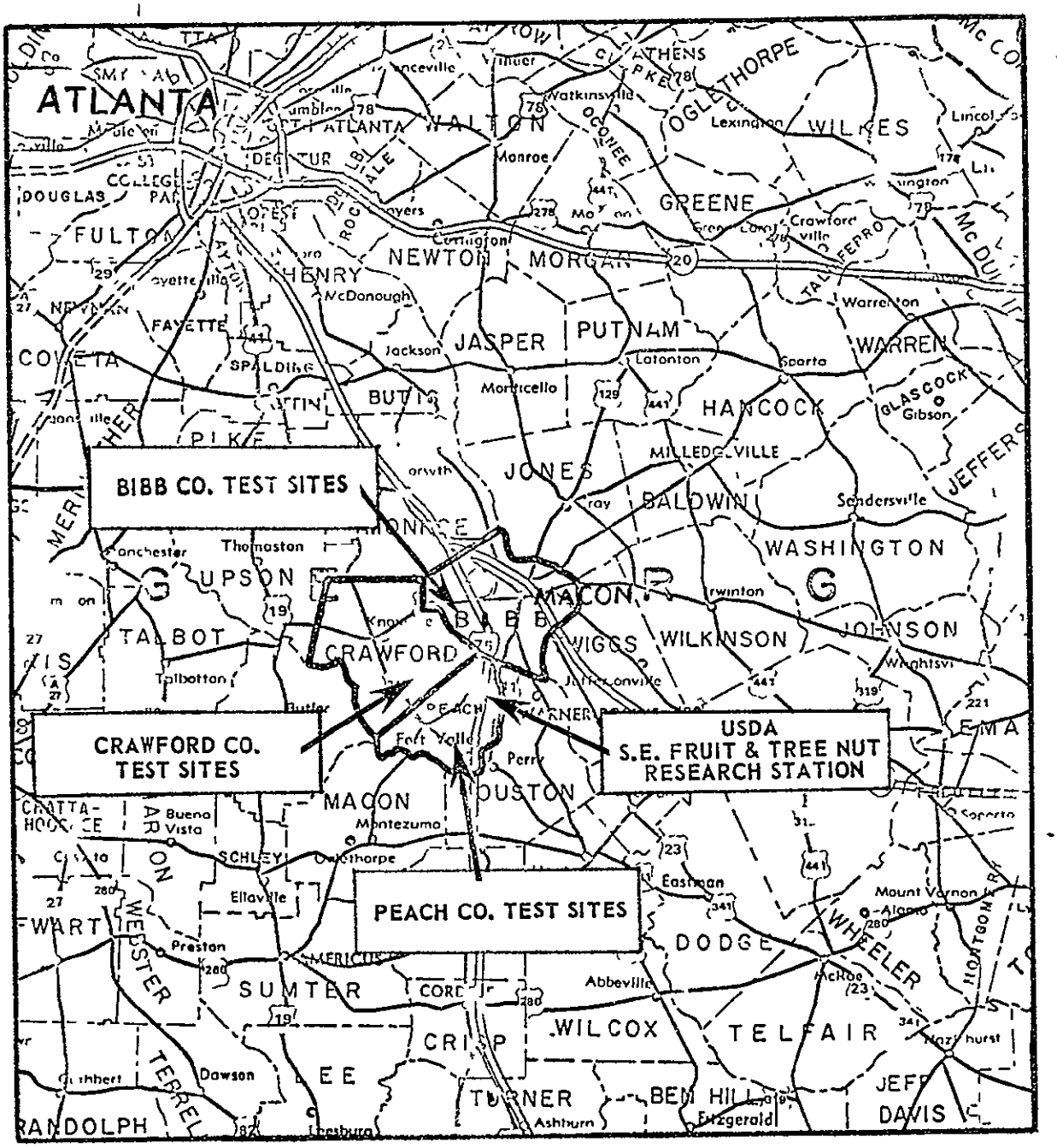


Fig 1.1 Locations of peach orchard test sites and USDA Research Station

II. DIGITAL FILTERING TECHNIQUES

2.1 Two Dimensional Linear Filtering

The first topic to be discussed is digital correction of intensity variations, by means of digital filters. This method is feasible as a procedure for removing variations which have spatial rates of change differing from those of the objects of interest in the scene. This is particularly true of an aerial photograph of a peach orchard, in which the objects of interest are of a consistent size.

The input/output relation of a linear digital filter is of the form

$$Z_i = \sum_{k=-L}^{L} g_k \, X_{i+k}$$

where $\{X\}$ is the input data sequence and $\{Z\}$ is the output data sequence. It may be seen that an output data value is a sum of the neighboring 2L input values, plus the input data point, each multiplied by a filter weight g_k The implementation of a filter consists of selecting the filter weights, followed by evaluating the input/output relation at each data point. Because of the large amount of data in a digitized image, it is essential that the implementations be fast in order to make digital processing practically feasible

A very fast implementation is obtained when the filter weights are given the value one, and the evaluation is done recursively In this case, the filter output is a moving sum over the input data, and in advancing by one data point, it is necessary to add only one new sample and subtract one sample which has left the filter length of (2L+1) samples The following diagram illustrates:

Z_{i-1}	i-L-1	i-L			i+L-1	
Z_i		i-L	i-L+1		i+L-1	i+L

$$Z_i = Z_{i-1} + X_{i+L} - X_{i-L-1}$$

Also, in the actual filtering of a digitized image, two dimensional filtering is performed This involves extending the summation over all neighboring points in a region of dimensions $(2L_x+1)$ $(2L_y+1)$ The output data values will occupy the same magnitude range as the input if the average of the $(2L_x+1)$ $(2L_y+1)$ terms is taken.

When the two-dimensional moving averages are applied, data variations which occur over a span of less than 2L+1 data points tend to be averaged out, while more slowly changing variations remain Hence, the operation has the effect of low pass filtering Denoting the moving average low pass filter operation by LP we can write

$$Z = LP\ (X)$$

Since the inverse of a low pass filter transfer function is that of a high pass filter, a high pass operation may be obtained as

$$Z = HP\ (X) = [1 - LP]\ (X) = (X) - LP\ (X).$$

This is equivalent to subtracting the moving averages (the slowly varying intensity variations) from each data point (A constant bias may be added to restore the data to its original range).

Band pass filters may be constructed from combinations of low and high pass. A filter which passes in the midrange is given by

$$BP = LP(L_1) - LP\ (L_2)$$

where $L_1 > L_2$. In this case, LP (L_1) removes the extreme high frequency variations, while subtracting the output of LP (L_2) removes the extreme low frequency variation. A filter which attenuates the midrange is given by

$$BP = LP\ (L_1)\ \ HP\ (L_2).$$

In this case, the high and low ranges are passed.

2.2 Digital Filtering of a Digitized Peach Orchard Aerial Photograph

Initial experiments were performed using a green band image obtained at 6000-feet altitude. A rectangular orchard consisting of approximately 37x84 rows of peach trees was digitized to 64 levels spanning a range of 0 to 2D in photographic density. The digitizing spot size was 50 microns, yielding for the digitized scene an array of 850x1800 pixels

Visual observation of the image indicated large variations in scene intensity, although maintaining distinct contrast between the tree crowns and background. The histogram of the scene density is given in Figure 2 1. The histogram verifies the wide range of densities present in the image, and lack of uniformly separable intensity ranges corresponding to tree crowns and background

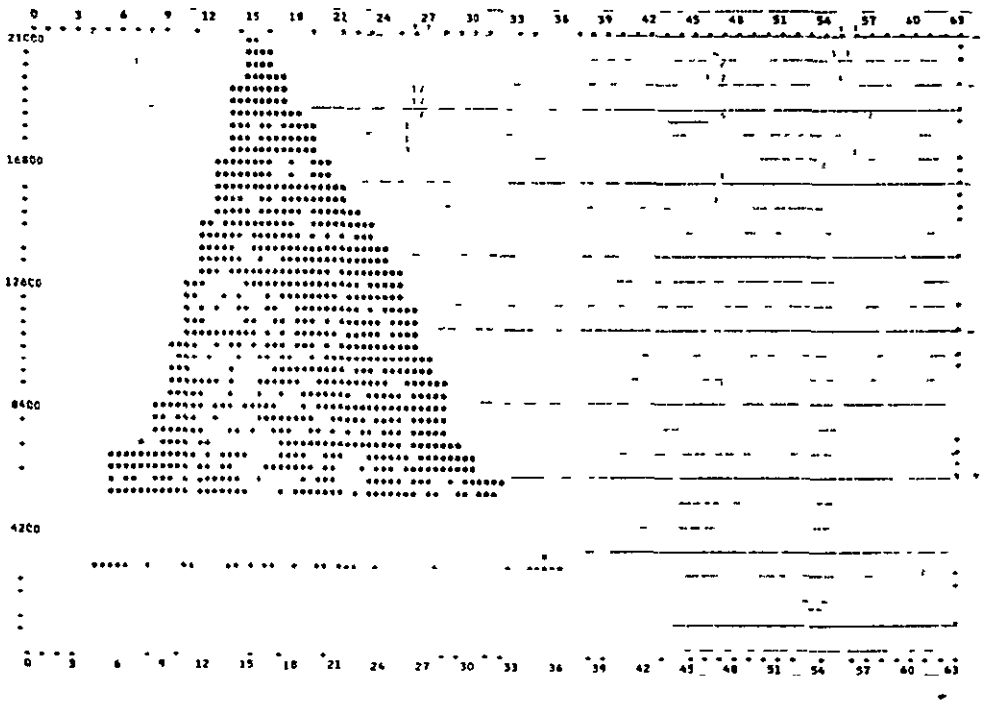


Figure 2.1. Histogram of densities present in the peach orchard image.

As a first step in the processing of the image, it was decided to attempt the removal of the background intensity variations. The possibility of performing this operation arises from the slower rates of spatial change of the background intensity variations compared to the variations due to the tree crowns This fact is illustrated in Figure 2 2, which is a trace of the density variations along a row of trees as the background density varies widely. The ten intensity peaks correspond to ten tree crowns which are dark in appearance on the photograph The overall intensity variation is evident in the plot, with a region of low intensity near pixel 100.

In this case the variations of intensity do not follow a positional dependence which would allow fitting to a function of position. Hence the moving averages (low pass output) must be computed for each point in the digitized image. If the size of the moving average region is chosen correctly, the low pass output will be a measure of the low frequency variations in image density. This output at each point in the image is subtracted from the original data value and a bias is added to restore the overall average density of the output image. As indicated previously, the result of subtracting a low pass filter output from the original data is characteristic of high pass filtering. Figure 2.3 is a trace along the row of pixels shown in Figure 2 2, after the image has been high pass filtered Twenty-five neighboring points were taken as the filter length in each direction, yielding a total area of 51x51, including the data point itself It can be seen that the high pass output retains the detailed structure of the original data while significantly reducing the slow drift exhibited in the data.

A further refinement was attempted using the high pass filtered data This data was low pass filtered, using a filter length small compared to the dimensions of the tree crowns Thus, in this case, the variations arising from the tree crowns are considered as slowly changing and are relatively unaffected, while smaller patches of vegetation and nonuniformity within tree crowns are reduced. The net result of these operations is a band pass filtered image, in

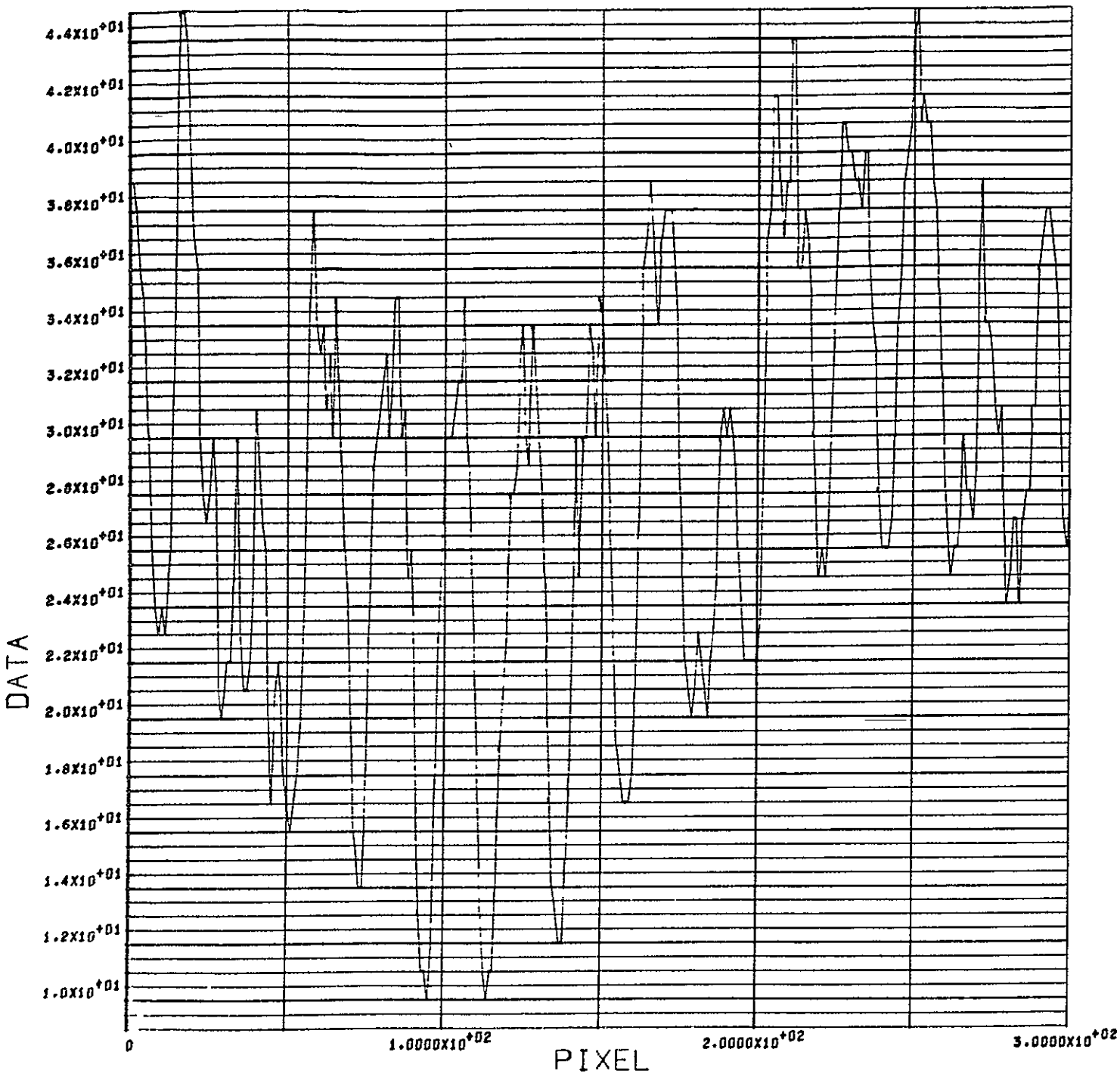


Fig.2.2. Density variations along a row of trees in the original data.

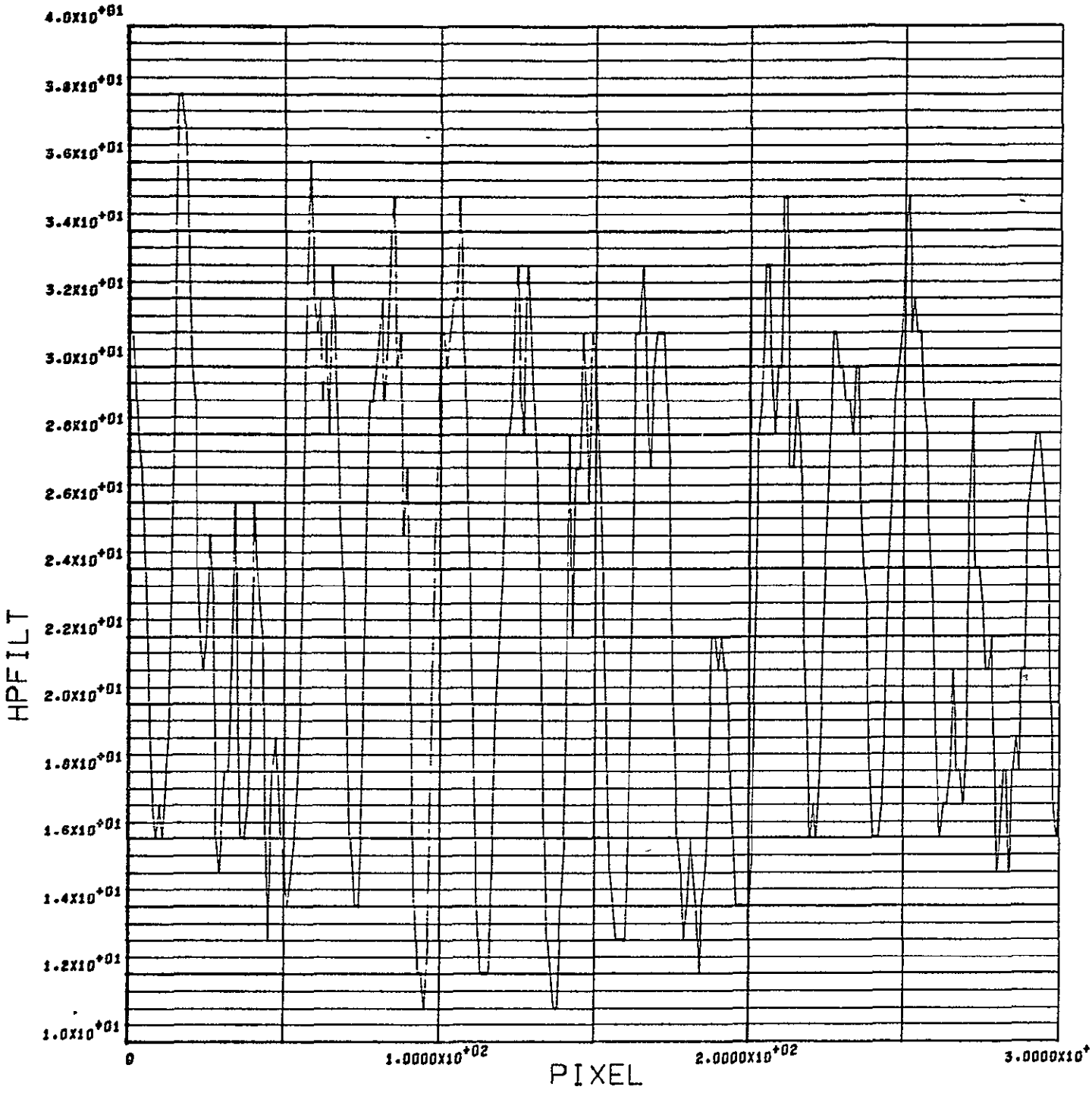


Fig. 2.3. Density variation along a row of trees after high pass filtering.

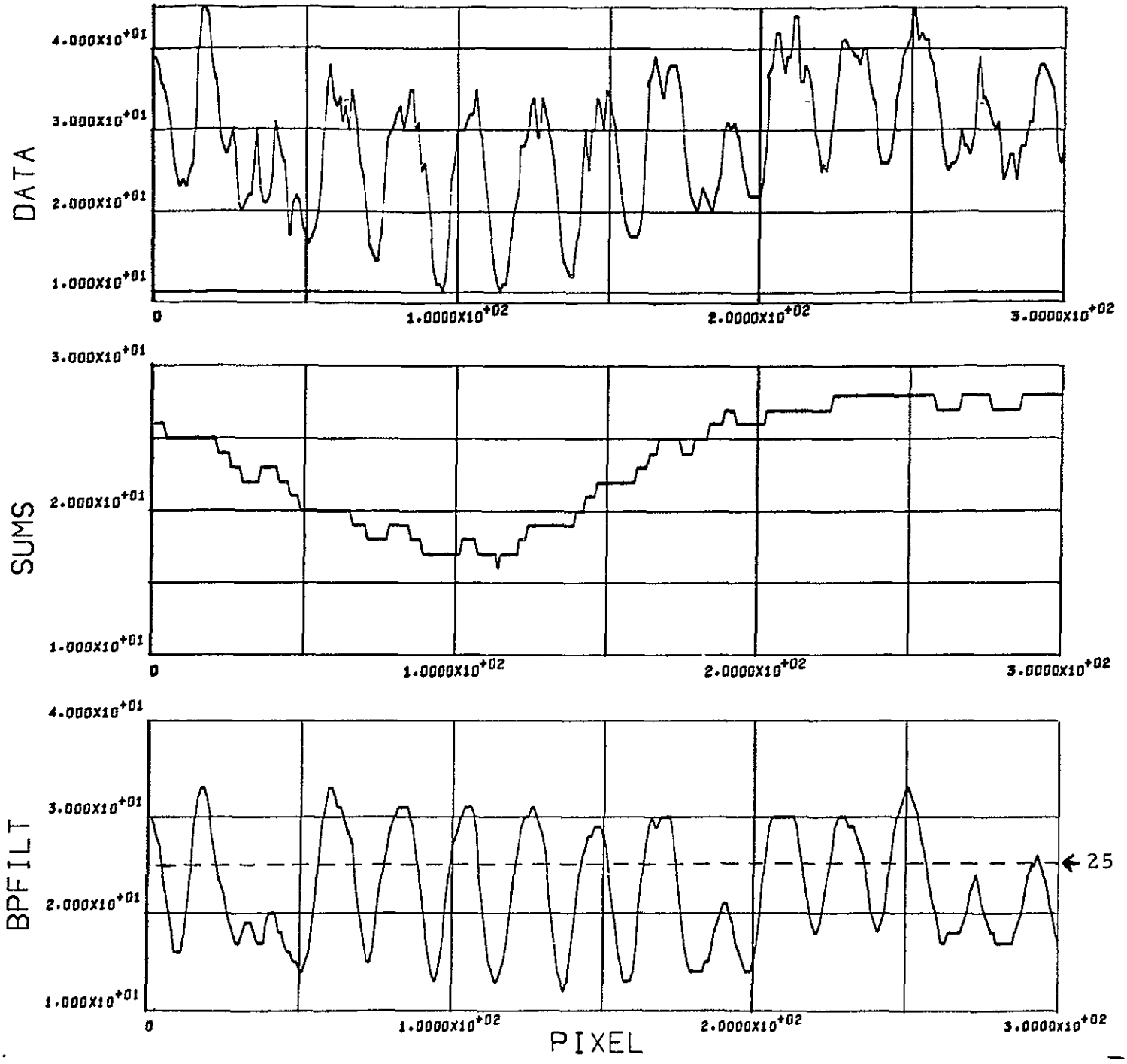


Fig 2.4. Original data, low pass and band pass outputs

which only those objects which are the size of tree crowns retain their original intensities. Figure 2.4 shows the original data sample and the low pass and band pass outputs.

Examination of the original data scan shows that a separation of tree crowns and background is not possible for any value of threshold. However in the filtered output image all of the tree crowns can be separated from their surrounding background by a density threshold of approximately 25 (on the 0-63 scale of scanner output). This is further demonstrated by the histogram of the filtered data (Figure 2.5), which has become bi-modal, with a minimum occuring at the value 25.

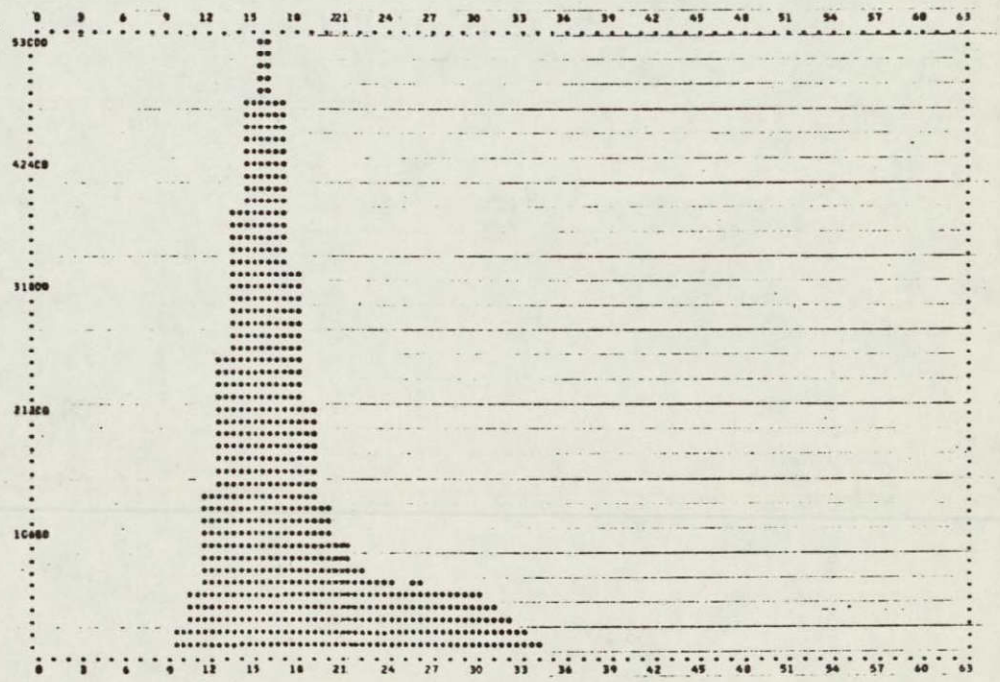


Figure 2.5. Histogram of densities in the peach orchard image after digital filtering.

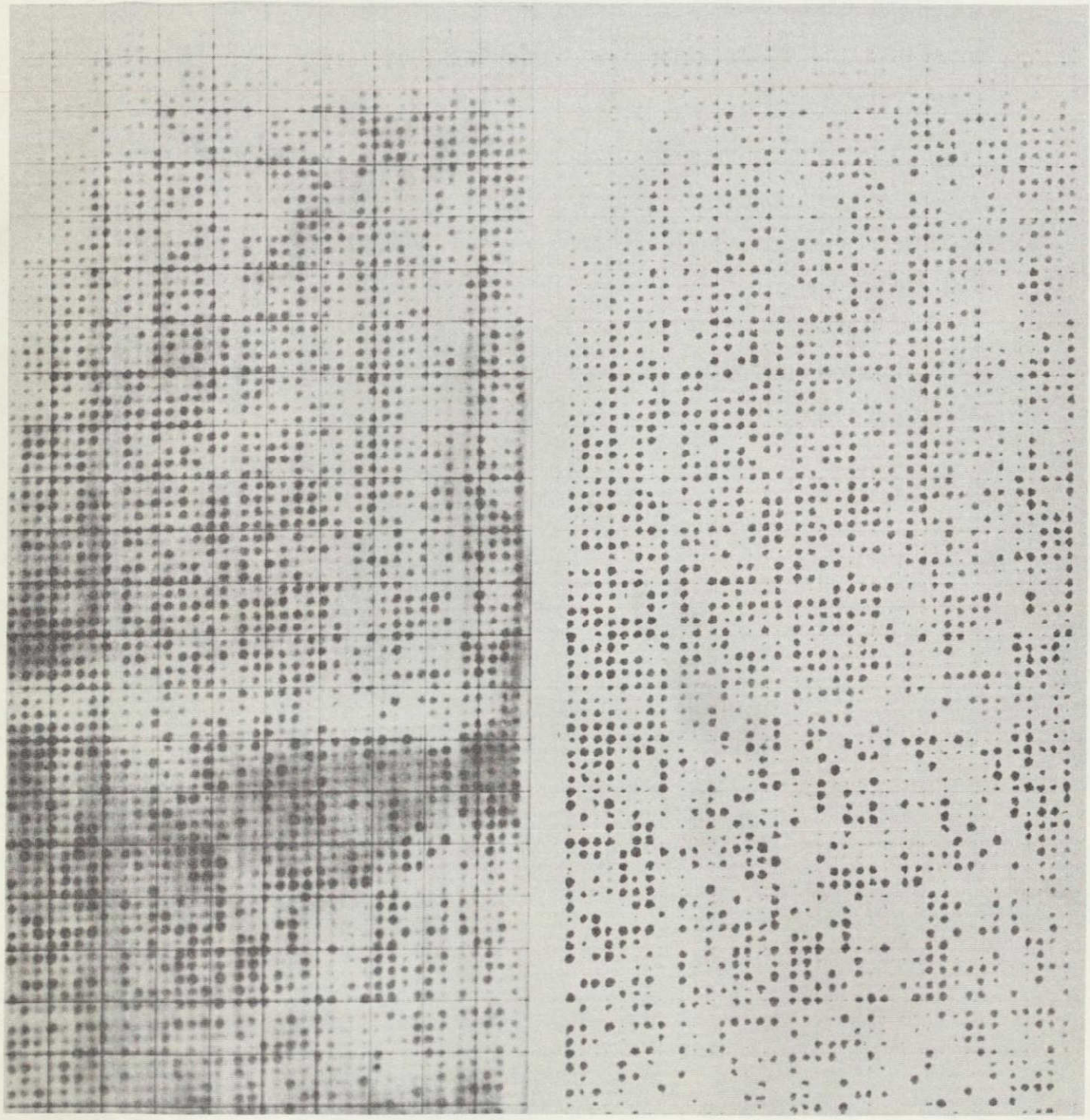


Fig. 2.6. Original image and result of background removal following digital filtering of a peach orchard test site.

Using a threshold of 25, a digital image can be generated in which the tree crowns remain while the background is set to zero density. However, as the densities of the tree crowns have been altered somewhat by the filtering operation, a more accurate result is obtained if the filtered image is used as a mask to be compared with the original image. In this manner, the background in the original image can be set to zero density. A resulting image obtained in this manner and the original image are shown in Figure 2.6. The histogram of the tree crowns only is shown in Figure 2.7. The symmetry of the distribution indicates the exclusion of the background pixels.

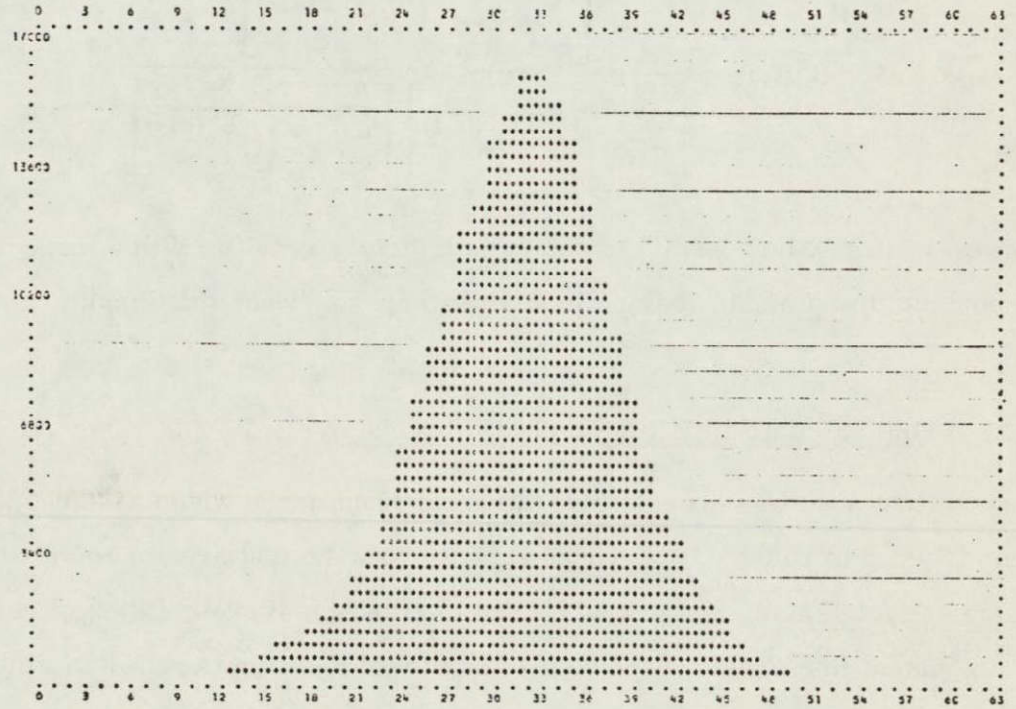


Fig. 2.7. Histogram of tree crown densities [following background removal]

III. DETECTION TECHNIQUES

3.1 Introduction

The problem of determining the positions of specified objects in a digitized image array may be approached by comparing, in some way, a local region in the image with a mathematical template which represents the searched-for objects. A match is considered to have occurred when the similarity between the image and the template reaches a certain level, in terms of a mathematical criterion. The traditional, but time consuming, method is a search for peaks in the cross-correlation function, at image coordinates [i, j] the sum of the product of each template term with its corresponding image term. For a template T of size mxm, with an image array I, the normalized cross-correlation is given by:

$$R^2(i,j) = \frac{\left[\sum_{k=1}^{m}\sum_{l=1}^{m} T(k,l)\, I(k,l)\right]^2}{\left[\sum_{k=1}^{m}\sum_{l=1}^{m} T^2(k,l)\right]\left[\sum_{k=1}^{m}\sum_{l=1}^{m} I^2(k,l)\right]}$$

However, it may be possible to obtain satisfactory results, with a large decrease in computation time, by using simple differences between the template and the image.[3]

3.2 Template Matching with Simple Differences

A test area was chosen from the green band image which had been high pass filtered to remove background variations. The test area of 100 pixels square was relatively small, but was so chosen to allow rapid presentation of results on the output line printer. A line printer plot of the area is shown in Figure 3.1.

The template chosen consisted of a circular region inscribed in a square of eleven pixels. The template elements representing the disk and the border were assigned values approximating the density levels of the tree crowns and background.

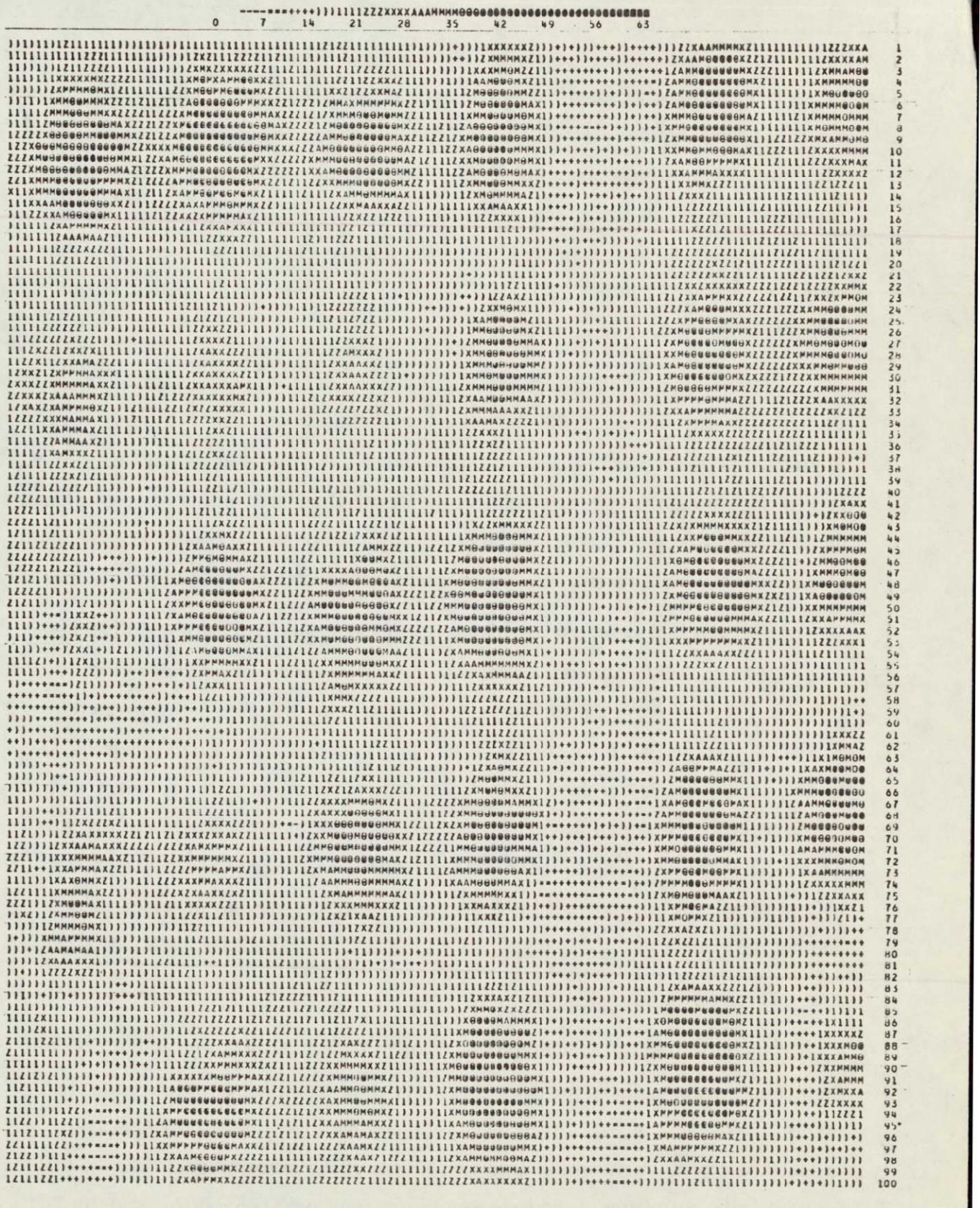


Figure 3.1. Line printer plot of the digitized peach orchard test site.

The template was centered over each pixel in the image, and the differences between each template element and the corresponding image pixels were summed. This sum, obtained at each point in the image, is termed the cumulative error. A printer plot of this error for the test region is given in Figure 3.2. As expected, the errors decrease steadily as the template becomes superimposed on each tree crown. The pixel position for which minimum error occurs will be considered to be the match position.

Plots of the errors obtained along two rows of the image are given in Figure 3.3. The upper plot is the error along line twenty-one of the test site, in an area between rows of trees. Consequently, the matching errors are large and fairly constant. The lower plot is the error along a line of tree crowns, and demonstrates the wide variation in error and in particular the rapid change in the error near the match positions (the minima in the plotted curve).

It is apparent from the preceding figures that an error threshold can be determined such that image points whose corresponding errors exceed the threshold are far from the locations of the tree crowns. This is illustrated by Figure 3.4 which is a printer plot of the errors at only those pixel locations having errors below the threshold. Comparison with Figure 3.1 demonstrates the correlation of these low error locations with the tree crowns in the image. This type of thresholding to indicate the locations of the tree crowns is preferable to simple density thresholding because it includes a dependence on shape and not only density.

3.3 Error Evaluation Sequences

The definition of the cumulative error at each pixel location is the sum of the absolute values of the difference between corresponding template and image pixels. However, in the present case, the computations can be formulated to reduce the computation time. This is a result of the circular symmetry of the template, the definition of the template in terms of only two values, and the proportion of background area to tree crown area in the imagery. Several methods of computation were examined in a series of experiments.

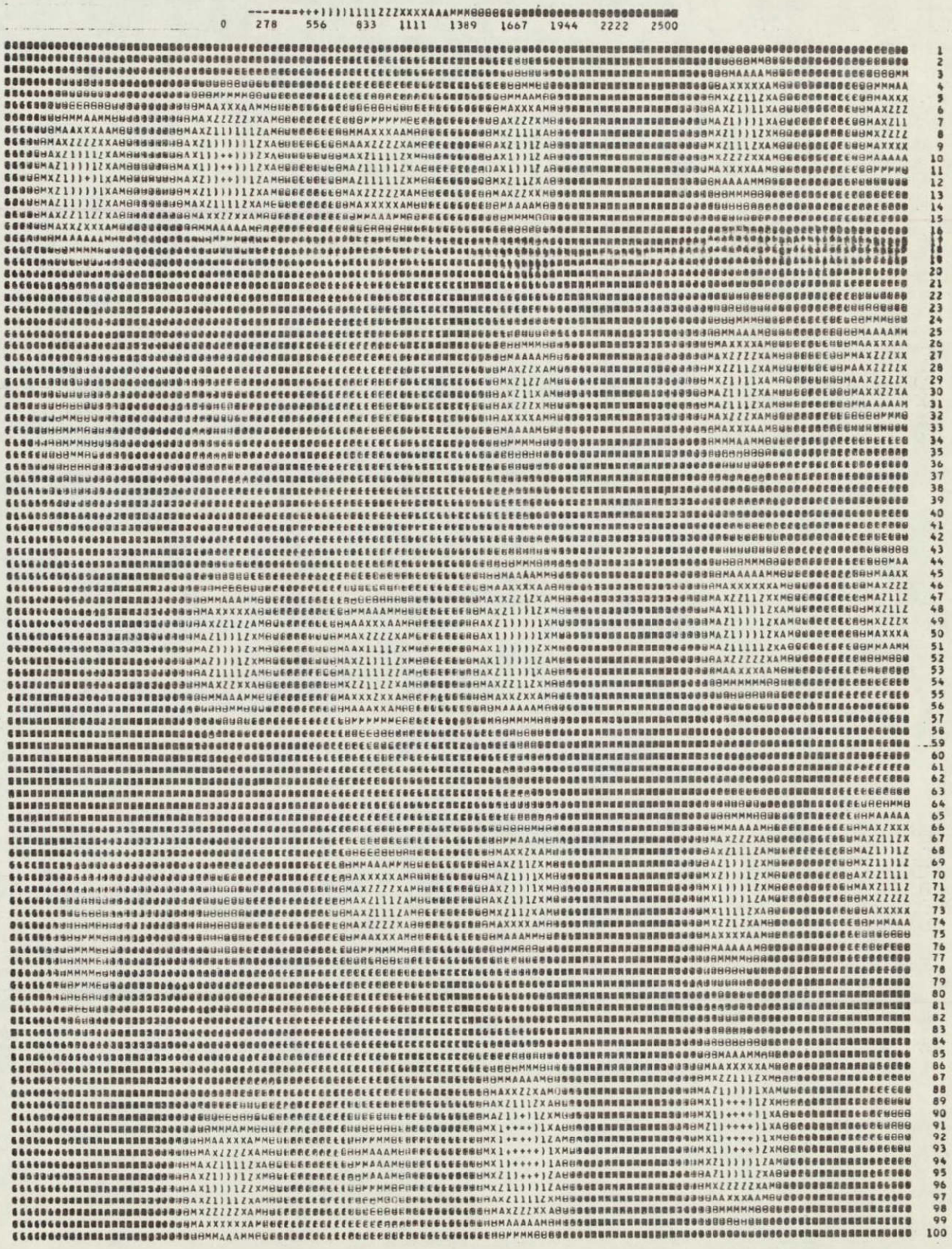


Figure 3.2 Cumulative errors obtained at each pixel in the test scene.

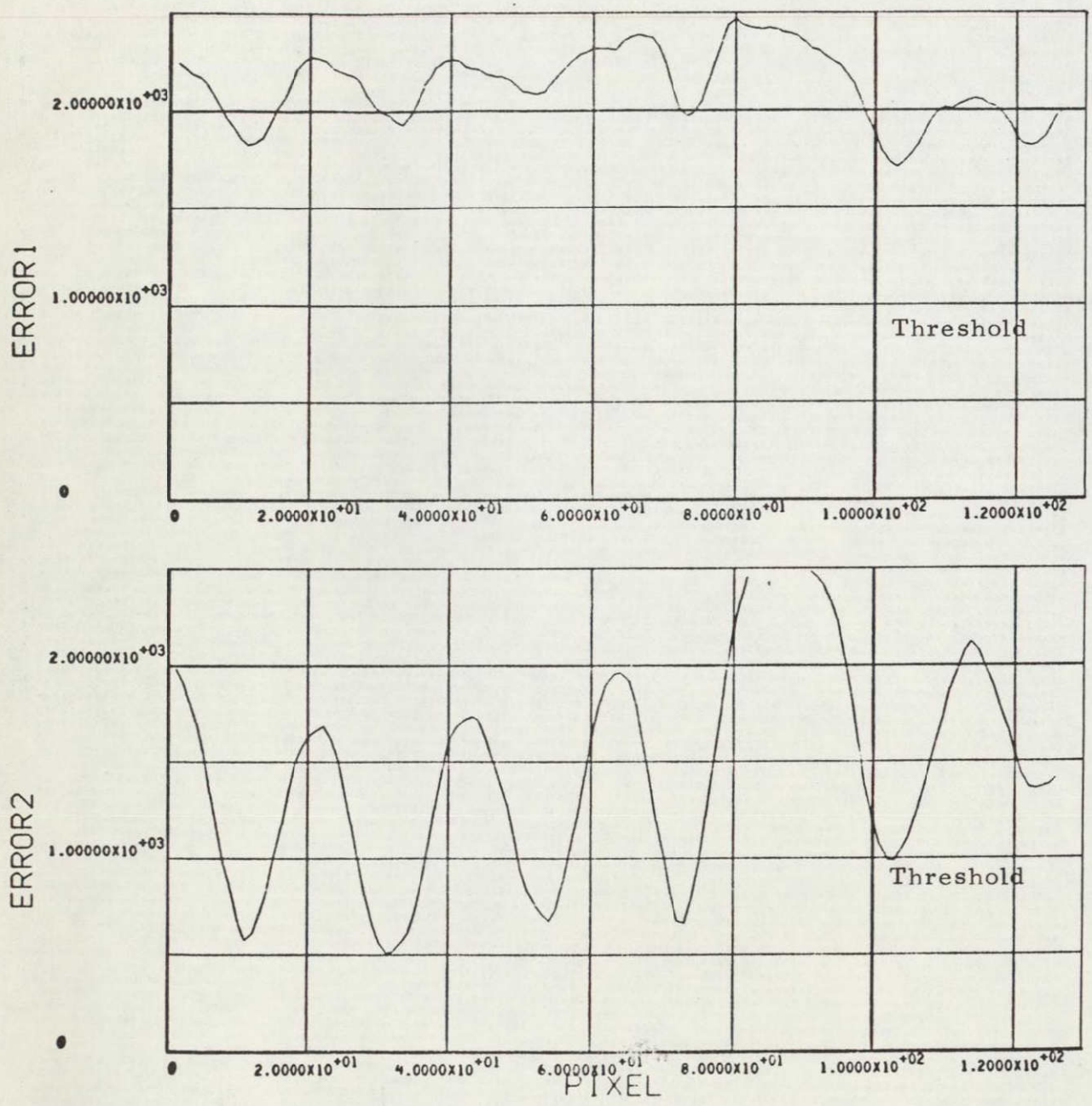


Figure 3.3. Cumulative errors along line 21 and line 11 of the test scene.

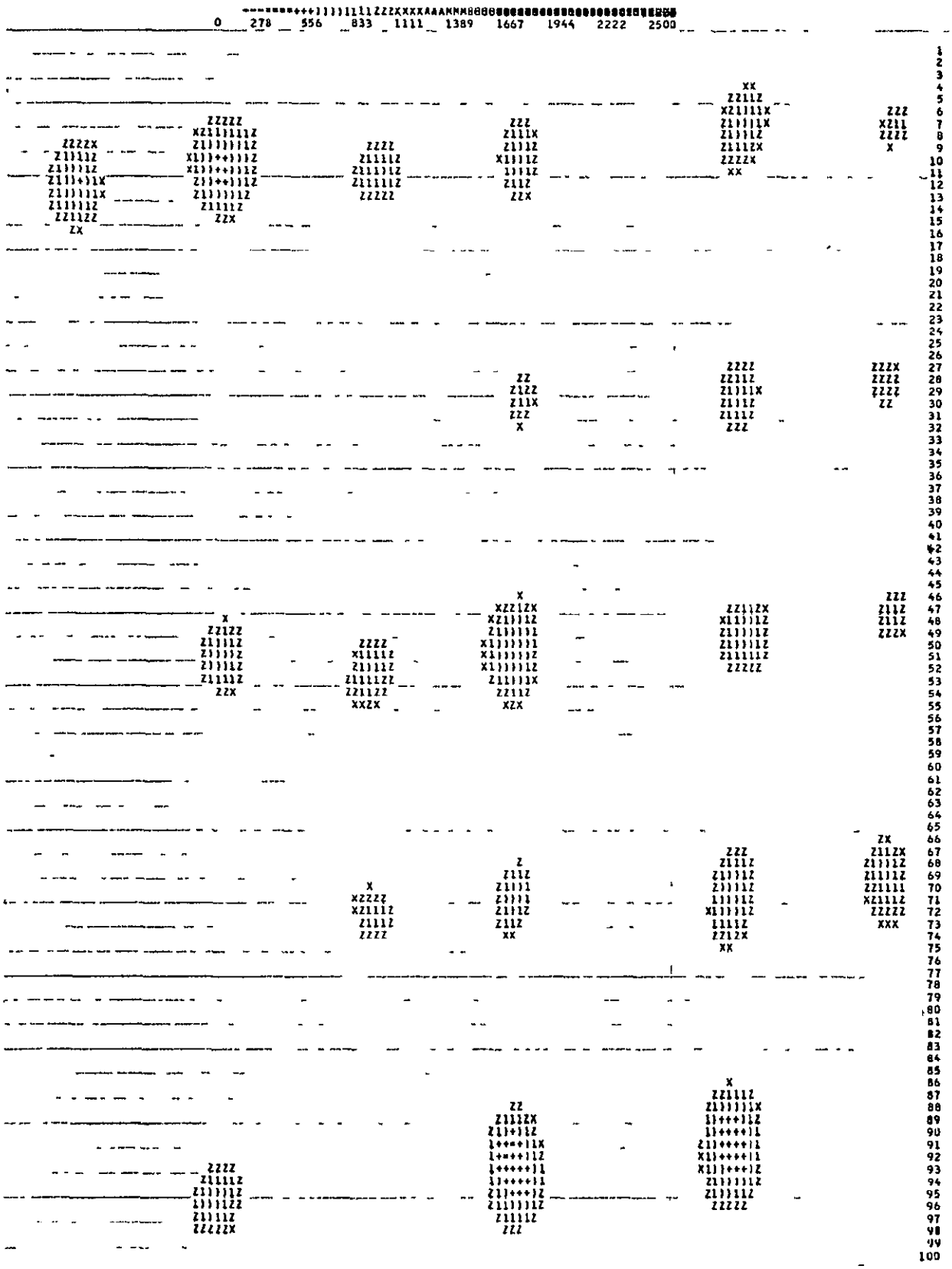


Figure 3.4. Cumulative Error after Thresholding

(i) Natural Sequence

This method is a straightforward summation of the absolute values of the differences for all the terms of the template taken in a raster scan order. The computation time required on the test site was 91 seconds. (All times given are computation times only, and do not include time required for input/output operations. Machine used was a 7094/1.)

(ii) Natural Sequence + Total Error Threshold

Since the only areas of interest are those with errors below the threshold, the accumulation of error may be terminated whenever the threshold is reached. This procedure reduced the computation time to 78 seconds.

(iii) Error Sort Method - A

The following two methods are based on sorting of the error terms such that the template evaluation sequence corresponds to the errors obtained, from the highest to the lowest In this case, the template is shifted ±1 pixel from a match position with itself along rows and columns, the errors in the four cases evaluated, and sorted so that the evaluation sequence determines errors in order of their magnitude [4] The resulting sequence is given in Figure 3 5, and the computation time required was 80 seconds. The method is not advantageous for this particular problem, probably due to the small percentage of evaluations which are performed near match positions

(iv) Error Sort Method - B

This is a more generalized error sort method. The test scene was used as a "training" area, to determine the average error for each term in the template as the template was placed at each pixel in the scene. The average errors at each point in the template are given in Figure 3.6. The smaller errors in the corners are at the template positions corresponding

29	35	1	2	37	45	88	4	5	105	107
113	7	3	30	31	32	33	34	6	9	36
11	8	38	39	40	41	42	43	44	10	13
12	46	47	48	49	50	51	52	53	54	14
55	56	57	58	59	60	61	62	63	64	65
66	67	68	69	70	71	72	73	74	75	76
77	78	79	80	81	82	83	84	85	86	87
17	89	90	91	92	93	94	95	96	97	20
15	16	98	99	100	101	102	103	104	18	19
106	21	22	108	109	110	111	112	23	24	114
115	116	25	26	117	118	119	27	28	120	121

Figure 3 5 Evaluation Sequence for Error Sort Method-A (Method No iii)

to the background of the scene, and are smaller since the majority of the scene is background. The evaluation sequence is given in Figure 3 7, and the computation time required was 71 seconds. This method has the disadvantage of being training data dependent, and hence similar, but circularly symmetric sequences were tested

(v) Disk - Background Sequence

As a better approximation to the empirical sequence of method No. (iv), a sequence was evaluated which spirals inward over the region of the disk, followed by an outward spiral over the region of the background The computation time was 72 seconds.

7.16	7.23	7.30	18.15	18.11	18.09	18.08	18.08	7.43	7.40	7.36
7.20	7.28	18.16	18.11	18.07	18.05	18.04	18.04	18.06	7.44	7.39
7.26	18.16	18.10	18.06	18.02	18.00	17.99	17.99	18.01	18.04	7.44
18.15	18.10	18.04	17.99	17.95	17.93	17.92	17.93	17.95	17.98	18.03
18.09	18.03	17.97	17.93	17.89	17.87	17.86	17.87	17.89	17.92	17.98
18.03	17.97	17.91	17.87	17.83	17.81	17.80	17.81	17.83	17.87	17.93
17.97	17.91	17.86	17.81	17.78	17.76	17.75	17.76	17.78	17.82	17.88
17.92	17.87	17.81	17.77	17.73	17.72	17.71	17.72	17.75	17.79	17.84
7.60	17.84	17.78	17.74	17.71	17.69	17.68	17.70	17.72	17.76	7.72
7.61	7.68	17.77	17.73	17.70	17.68	17.68	17.69	17.72	7.78	7.72
7.60	7.67	7.73	17.74	17.71	17.69	17.68	17.70	7.82	7.77	7.71

Figure 3.6. Average template errors throughout the test scene

121	119	116	4	5	9	12	11	112	113	115
120	117	2	6	13	16	19	17	14	111	114
118	1	7	15	24	26	29	27	25	18	110
3	8	20	28	36	37	43	38	36	30	22
10	21	32	39	46	50	54	52	47	42	31
23	33	44	53	59	61	65	62	58	49	40
34	45	55	64	69	74	75	73	68	60	48
41	51	63	71	79	84	85	82	76	66	56
100	57	67	77	86	91	95	90	81	72	103
117	105	70	80	89	96	97	92	93	99	102
108	106	101	78	87	93	94	88	98	100	104

Figure 3.7. Evaluation sequence for Error Sort Method-B (Method No. iv]

(vi) Increasing Radius Sequence

The sequence in order of increasing radius evaluates the error for all of the template terms representing tree crowns before proceeding to the terms representing background. Since the majority of the scene is not tree crowns, errors on the average accumulate rapidly and this sequence yields a rapid computation, requiring 62 seconds.

(vii) Elimination of Absolute Value Computation

In general, the simple difference between a template term and an image pixel must be the absolute value of the difference to allow for image data values above and below the template values However, in the present case of a double-valued template, the template values may be chosen to insure that all difference terms have the same sign. For example, the tree crowns have high density in the transparencies, and the corresponding template region is set at the maximum image value. This is infact preferable in the present case, since the optimal match is obtained for the densest tree crown and lightest background. Using this procedure, combined with the increasing radius sequence and error thresholding , the computation time drops to 46 seconds.

(vii) Recursive Evaluation

The recursive evaluation sequence takes account of the fact that as the template is centered on successive pixels of the image, the majority of the error terms retain their previous values. Hence it is only necessary to consider the new terms appearing at the leading edges of the template regions (both the leading terms of the whole template and each disk-background interface), and the terms being dropped at the trailing edges. This method is extremely rapid, requiring 18 seconds, and it is a property of the method that the complete cumulative error is obtained

3.4 Error Threshold Sequences

The result shown in Figure 3.4, obtained by thresholding the cumulative error, indicates the locations and approximate sizes of the tree crowns This result is preferable to simple thresholding, in which all pixels above the threshold would be retained, irregardless of the shapes of the regions retained A further increase in computation speed is obtained when a constant threshold is replaced by a monotonically increasing threshold sequence In this case the threshold may be exceeded when a smaller number of error terms has been accumulated.

The threshold sequence curve should approximate, in its average slope, the curve of cumulative error versus number of terms. An example is given in Figure 3 8. The error at test points A and B accumulates so rapidly that they are discarded as candidates for match after very little computation. Point C, which is at or near a match, is not discarded although the error for all terms is accumulated.

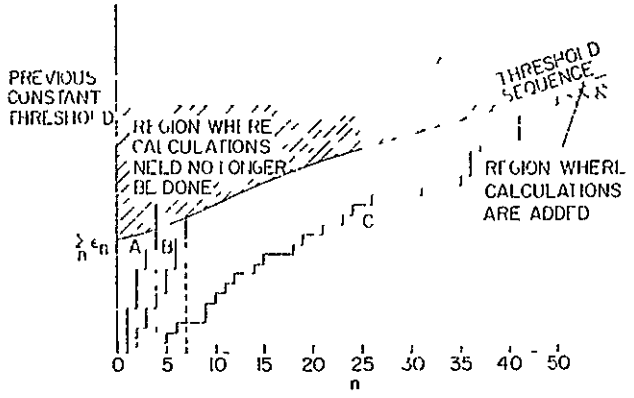


Figure 3 8. Reduction of Calculations with use of a Monotonically Increasing Threshold Sequence.

At the point of match, the cumulative error is the sum of the noise due to the fact that the areas being compared are different. If this image noise has a mean λ and zero deviation, the ratio of cumulative error to mean of the noise, E/λ, is unity at each point in the sum, and the cumulative error is linear with the number of terms. However, an increase in the variance of image noise is reflected in an increased rate of accumulation of error, and the threshold curve should take on higher values. Barnea and Silverman[3] have calculated threshold sequences for various deviations. A set of such curves is given in Figure 3.9. Barnea and Silverman also present calculated curves based on the probability of the cumulative error exceeding the threshold being held constant at each term in the sum. A set of these curves is given in Figure 3.10. Further analysis of this type of computation is given by Ramapriyan.[5]

Error sequences were plotted for the peach test site, using an evaluation sequence which starts at the center of the circular template and spirals outward The cumulative errors at each term in the sum over a template of size 11x11 pixels are plotted as the solid lines in Figure 3.11. The upper plot is for a non-match position, while the lower is obtained for the template centered on the upper left tree of the test site The area of the template circular region was nominally 100 pixels, but an actual area of 97 pixels was obtained, due to the discreteness of pixels. Thus, after 97 terms of the error sum, there is an abrupt change in the template value, and this accounts for the slope change in the error sum plots near the 100-th term.

From the cumulative error plots, threshold sequences can be obtained such that the error sums at non-match points exceed the threshold terms, while remaining below the threshold near match points. For this test, linear threshold sequences were chosen, since the error sequences are nearly linear for the initial terms. The threshold sequences shown as dotted lines increase by 8 and 5 for each term in the sum. It may be seen from the plots that, for non-match positions, the initial term of the error sum will exceed the threshold. The linear threshold curve passing through the origin does not make allowance for variance in the image noise The smoothness of the actual cumulative error plot justifies

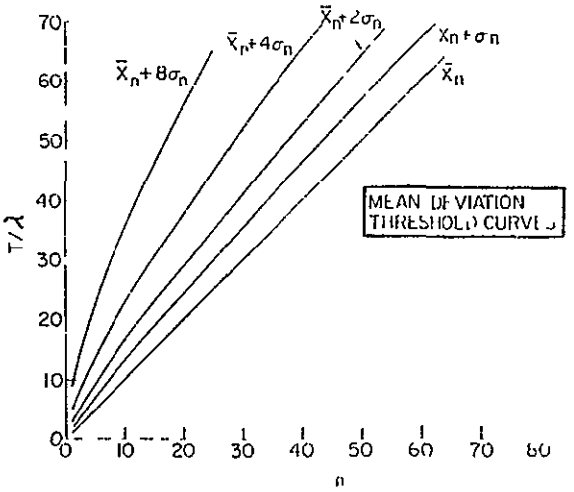


Figure 3.9. Calculated Threshold Sequences Based on Deviation of Image Noise

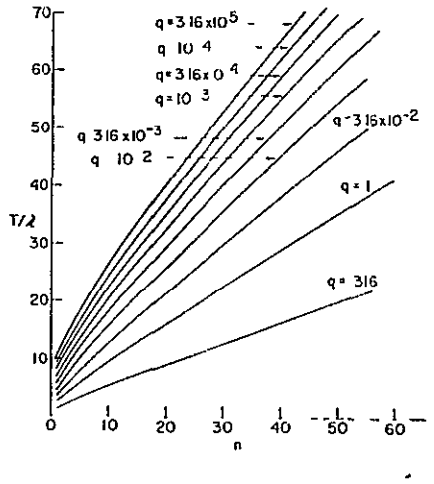


Figure 3.10. Calculated Threshold Sequences Based on Probability q of Exceeding the Threshold at the n-th Term

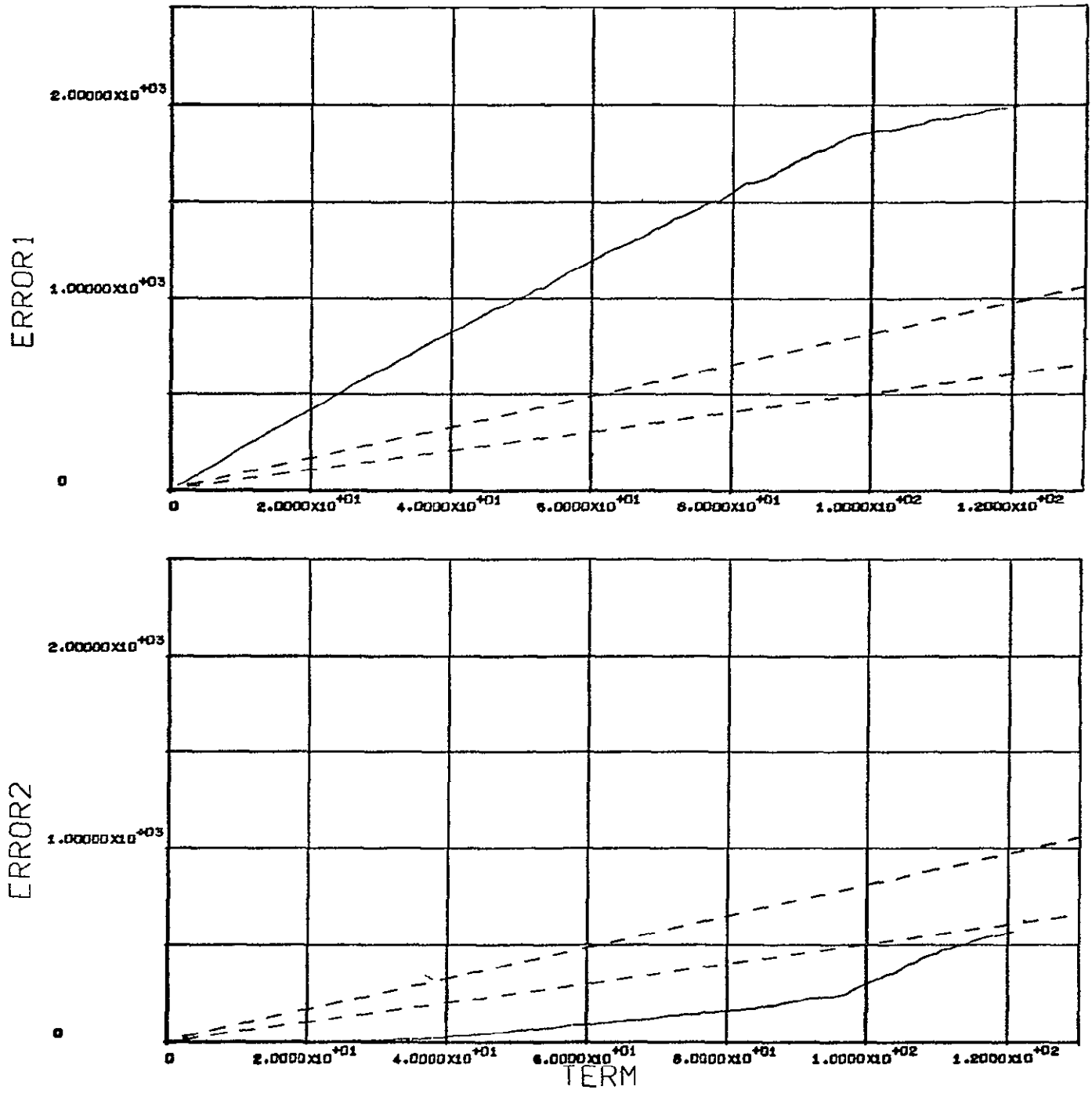


Figure 3.11. Cumulative error at a non-match and a match point. The dotted lines are threshold sequences with slopes of 8 and 5.

this assumption. Similarly, it may be seen that, near match points, the error is accumulated for all terms of the template, without exceeding the threshold. Hence, an appropriate output function is the number of terms in the error sum at which the threshold is exceeded, or the total number of terms when the threshold is not exceeded. Printer plots of this output are given in Figures 3.12 and 3.13 for threshold sequence slopes of 8 and 5, respectively. It is apparent that the sums are terminated quickly at non-match points, effecting a large reduction in computation time. The actual computation times were 11 seconds and 7 7 seconds, respectively The regions in which the error accumulates for a large number of terms approximate the sizes of the trees in Figures 3.12 and 3.13, as do the regions where the error is below a constant threshold in Figure 3.4. However, the result in Figure 3.13 was obtained with a six-fold increase in computation speed.

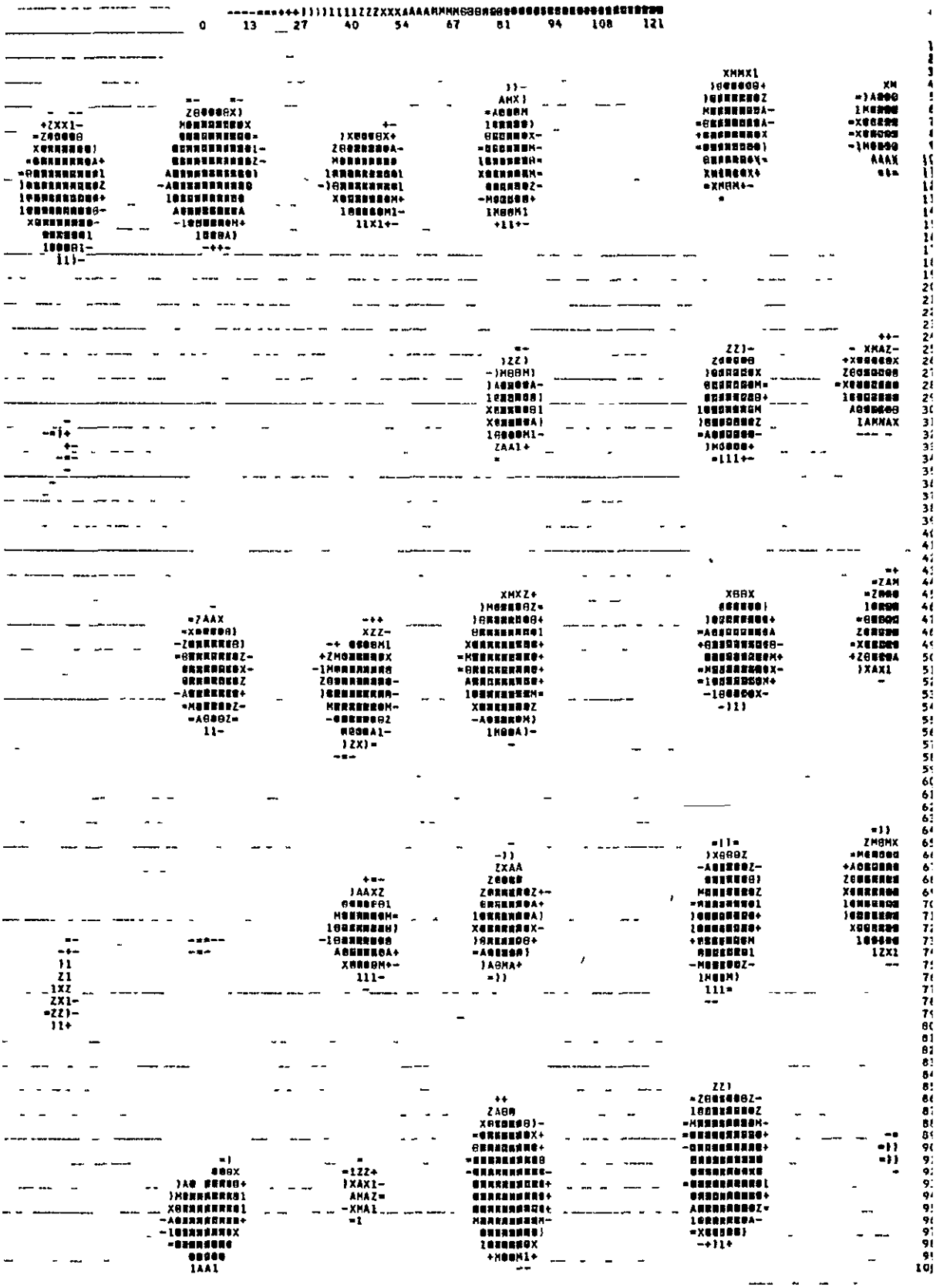


Figure 3.12 Plot of Error Sequence Terms for Threshold Sequence Slope of 8 Computation time 11 seconds.

Figure 3.13 Plot of Error Sequence Terms for Threshold Sequence Slope of 5. Computation Time 7.7 Seconds

3.5 Enumeration, Sizing and Coordinate Determination

3.5.1 Technique

Previous sections have demonstrated that simple differencing of a circular template and a digitized image of a peach orchard produces an error function with pronounced minima at the locations of tree crowns. The coordinates of a minimum are readily obtained and may be taken as the coordinates of the tree center. However, it is necessary to repeat the process of finding a new minimum value in the vicinity of each tree crown Hence an approximate position of match is found initially by considering the gradient between the leading and trailing edges of the circular template positions in the data.

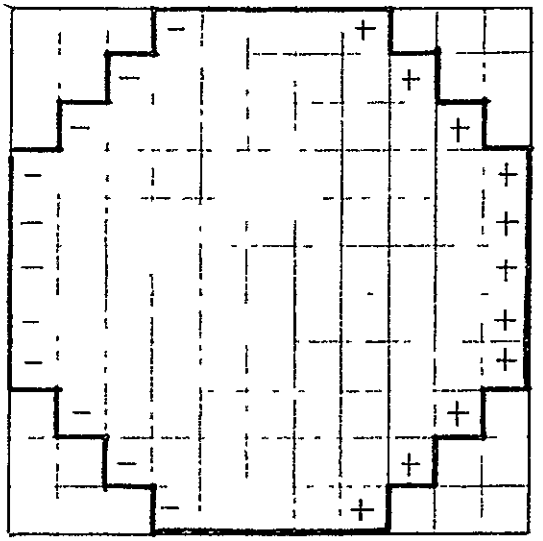


Figure 3.14 Positive and Negative Gradient terms for a Circular Template of Nominal Area 100 Pixels.

The template gradient is defined as the positive sum of all terms on the leading edge of the circular region, plus the negative sum of all terms on the trailing edge The positive and negative terms for the horizontal gradient of a template with a circular area of 97 pixels are given in Figure 3.14 The horizontal gradients along lines 13 and 23 of the test site are plotted in Figure 3.15. The upper plot is along a line centered on a row of trees, while the lower is between rows.

As a consequence of the facts that (i) the leading edge of the template defines the image points contributing to the positive terms in the gradient, and (ii) the tree crowns have digitized values of greater magnitude than the background, the gradient values are positive as the template approaches the position of a tree crown in a raster type scan. The value of the gradient falls rapidly through zero as the template passes over the center of a tree crown. By detecting a similar passage through zero in the orthogonal direction of scan, it is possible to rapidly obtain approximate coordinates of match

An alternative description of this procedure is obtained by regarding it as a recursive calculation of the sum of the image values corresponding to the circular area of the template disk. The gradient terms at each pixel location are the terms to be added to the preceding sum to obtain the magnitude of the sum at the current pixel location. Thus the gradients are the differential of the sums over circular regions centered at each pixel location, and as such pass through zero at the extremum of the sum.

This method of locating the approximate positions of match was chosen over others (such as total error threshold or monotonic threshold sequence] for the following reasons:

(i) A recursive-type computation is approximately as rapid as a threshold type,

(ii) passage through zero is easily detected,

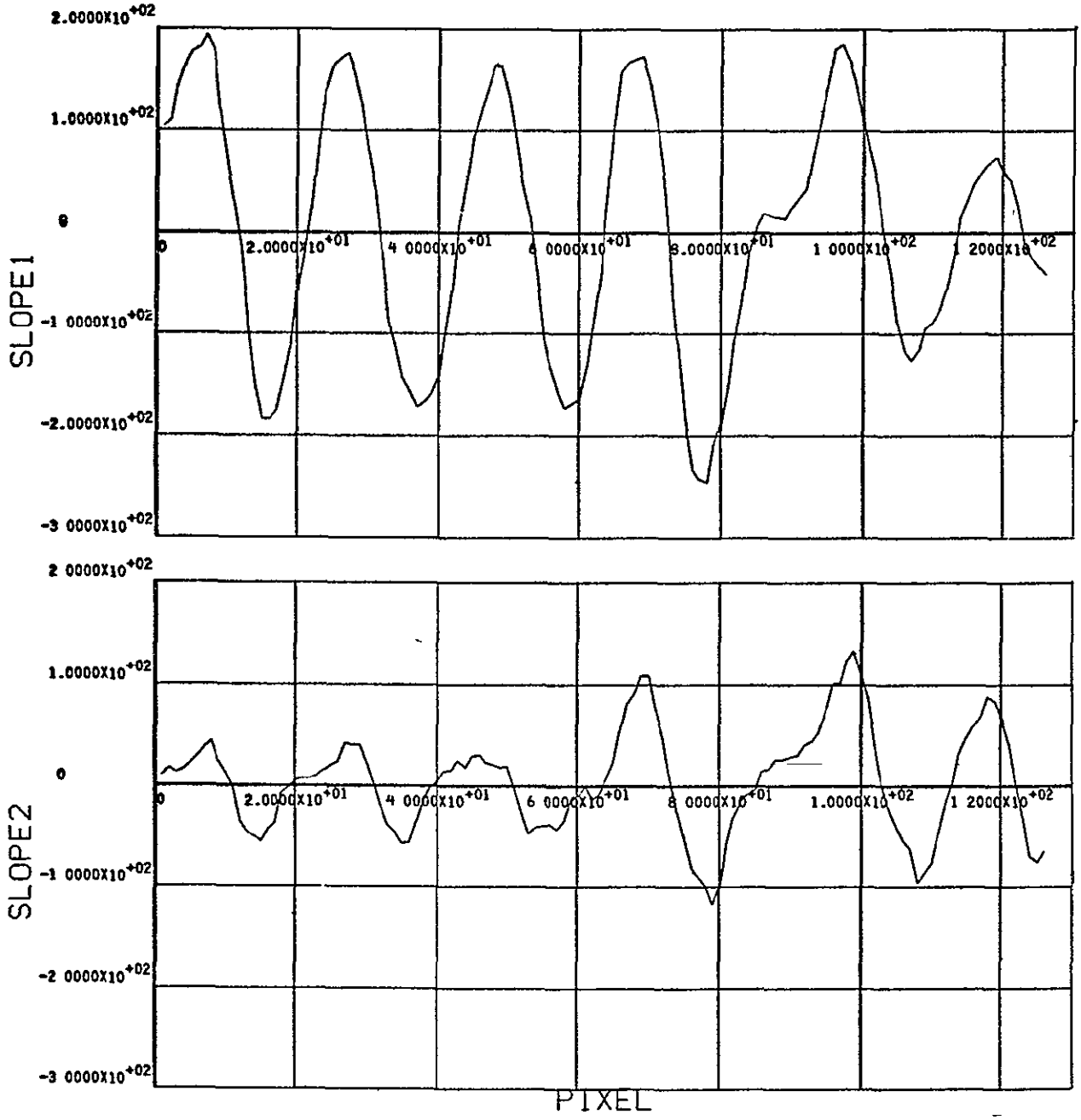


Figure 3.15 Horizontal Gradients along lines 13 and 23 of the Test Site.

(iii) the effect of bias in the image density levels on the choice of threshold values is eliminated,

(iv) differentials involving a small number of terms at the edges of the template disk yielded inaccurate match positions

After the approximate match position is found by means of gradient sequences, a block of area five pixels by five pixels centered on the approximate position is analyzed in detail. In this region, the total cumulative error between the template and the image is calculated using a recursive method, and the position of minimum error is determined If this error does not exceed a predetermined threshold, this position is accepted as the coordinates of the center of a tree crown In addition, a tree count is readily obtained.

3.5.2 Test Site Tree Detection

The peach orchard test site was analyzed in this manner. Figure 3.16 is an intensity plot of the cumulative errors which were computed over the five pixel by five pixel areas centered on the approximate match locations. The intense positions within regions of small error are the final tree coordinate positions, as determined by the minimum total cumulative error. The total error is above the threshold for those five by five squares which are intense at all positions, as in the leftmost three in the second row At these positions, the gradient procedure was sufficiently sensitive to detect the presence of an object, but the size or contrast did not match the template very closely, and hence the cumulative error was large.

The test site was analyzed using total error thresholds of 1000 and 1500. The computer printouts of tree coordinates and enumerations are given in Figures 3.17 and 3 18

In the first case, the average total error for the 21 matches was 634, or 5 2 per difference term between the template and the image In the second case higher error positions were accepted as matches, and the corresponding figures are 768 or 6.3. Line printer plots of the test sites with the outlines of the

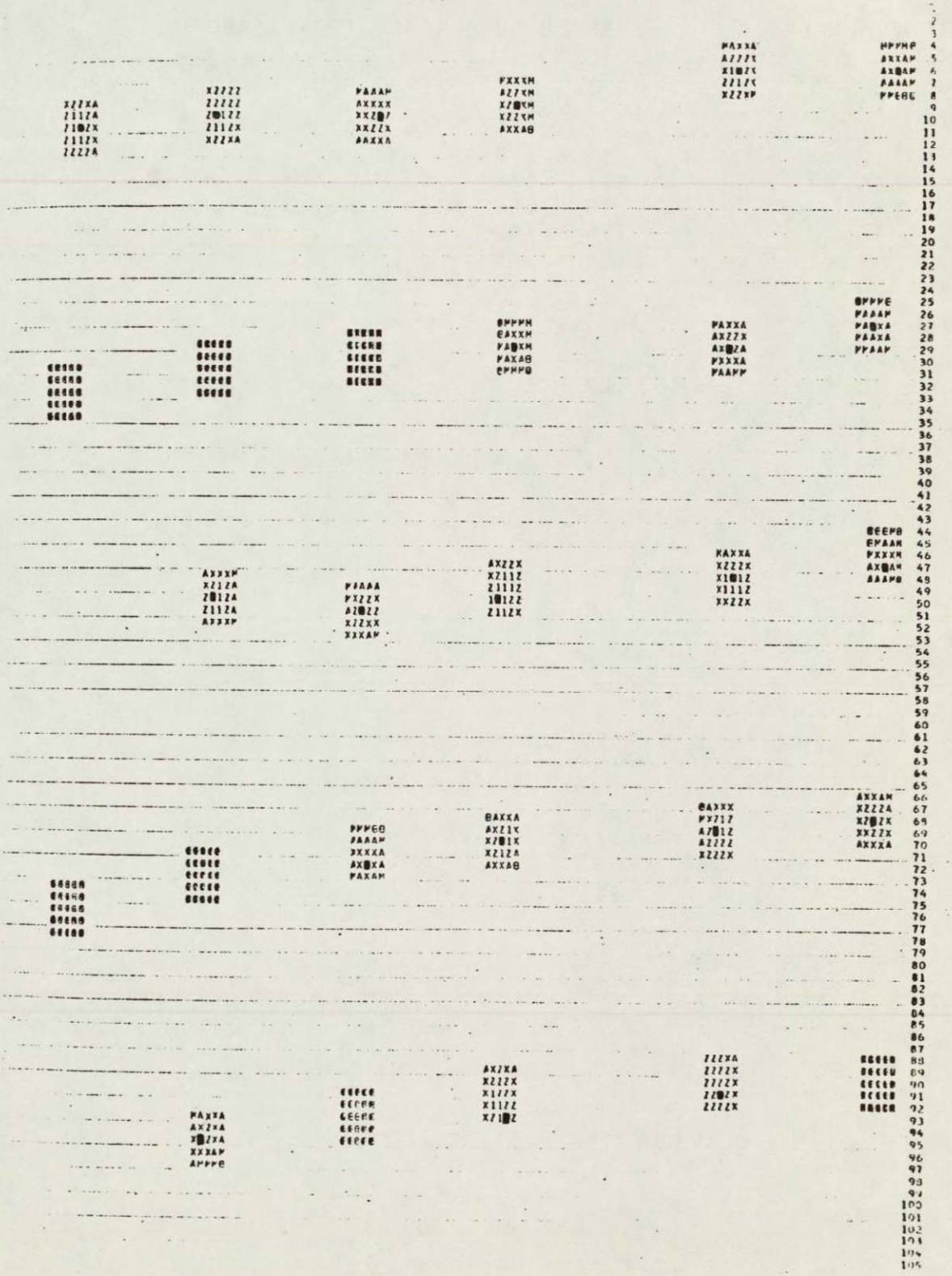


Figure 3.16. Cumulative errors for regions surrounding the approximate match positions.

TREE COORDINATES
****

LINE NUMBER	SAMPLE NUMBER AND TOTAL ERROR	
7	104 - 618	126 - 814
10	73 - 639	
11	31 - 515	54 - 673
12	12 - 547	
28	123 - 837	
30	73 - 758	103 - 661
48	125 - 786	
49	104 - 575	
50	72 - 557	
51	31 - 571	
52	52 - 686	
69	124 - 664	
70	102 - 602	
71	72 - 601	
73	53 - 737	
92	71 - 403	103 - 462
96	30 - 603	

TREE COUNT
****

NUMBER	AREA
******	****
21	97

ELAPSED TIME = 13.42 SECONDS

Figure 3.17 – Tree Coordinates and Enumeration using a Threshold of 1000.

TREE COORDINATES
****

LINE NUMBER	SAMPLE NUMBER AND TOTAL ERROR	
7	104 - 618	126 - 814
10	73 - 639	
11	31 - 515	54 - 673
12	12 - 547	
28	123 - 837	
30	73 - 758	103 - 661
33	10 -1318	
48	125 - 786	
49	104 - 575	
50	72 - 557	
51	31 - 571	
52	52 - 686	
69	124 - 664	
70	102 - 602	
71	72 - 601	
73	53 - 737	
74	30 -1470	
78	11 -1284	
91	125 -1442	
92	71 - 403	103 - 462
94	52 -1145	
96	30 - 603	

TREE COUNT
****

NUMBER	AREA
******	****
26	97

ELAPSED TIME = 12.30 SECONDS

Figure 3.18 **Tree Coordinates and Enumeration using a Threshold of 1500.**

detected trees superimposed are given in Figures 3.19 and 3.20. It may be noticed that the higher error threshold results in the inclusion of additional trees which have weak foliage and exhibit less contrast with the background. The apparently paradoxical reduction in computation time as additional trees are detected is due to the fact that an area occupied by a previously detected tree is not searched in succeeding scans, but rather bypassed.

3.5.3 Programming Considerations

This section presents a description of the algorithm in greater detail, with particular emphasis on methods of computer programming the algorithm.

In order to rapidly evaluate expressions involving the image pixels corresponding to a template of size nxn pixels, it is necessary to hold n records of the image data in core. If the template is to move to adjacent lines of the digitized image, a corresponding number of additional records should be held in core. In the present case the template is allowed to move over five scan lines of the image, and hence two additional records of data before and after the n records corresponding to the template size are stored.

Now, the simplest method of referring to the various template terms is by numbering them. If the elements of the template are numbered along its columns (consistent with the storage in memory of a two-dimensional array defined in the FORTRAN language), the element (i, j) is located at location n(j-1)+i. The corresponding element of the image data is obtained simply by adding a bias term due to the position of the template along the scan line. Thus, one sequence of location numbers is determined for each evaluation required, such as the gradient terms, cumulative error scans, recursive evaluation sequences.

Three types of evaluation sequences are used. These are:

(i) differential sequences used for recursive evaluation of five sizes of disk and border along rows and columns,

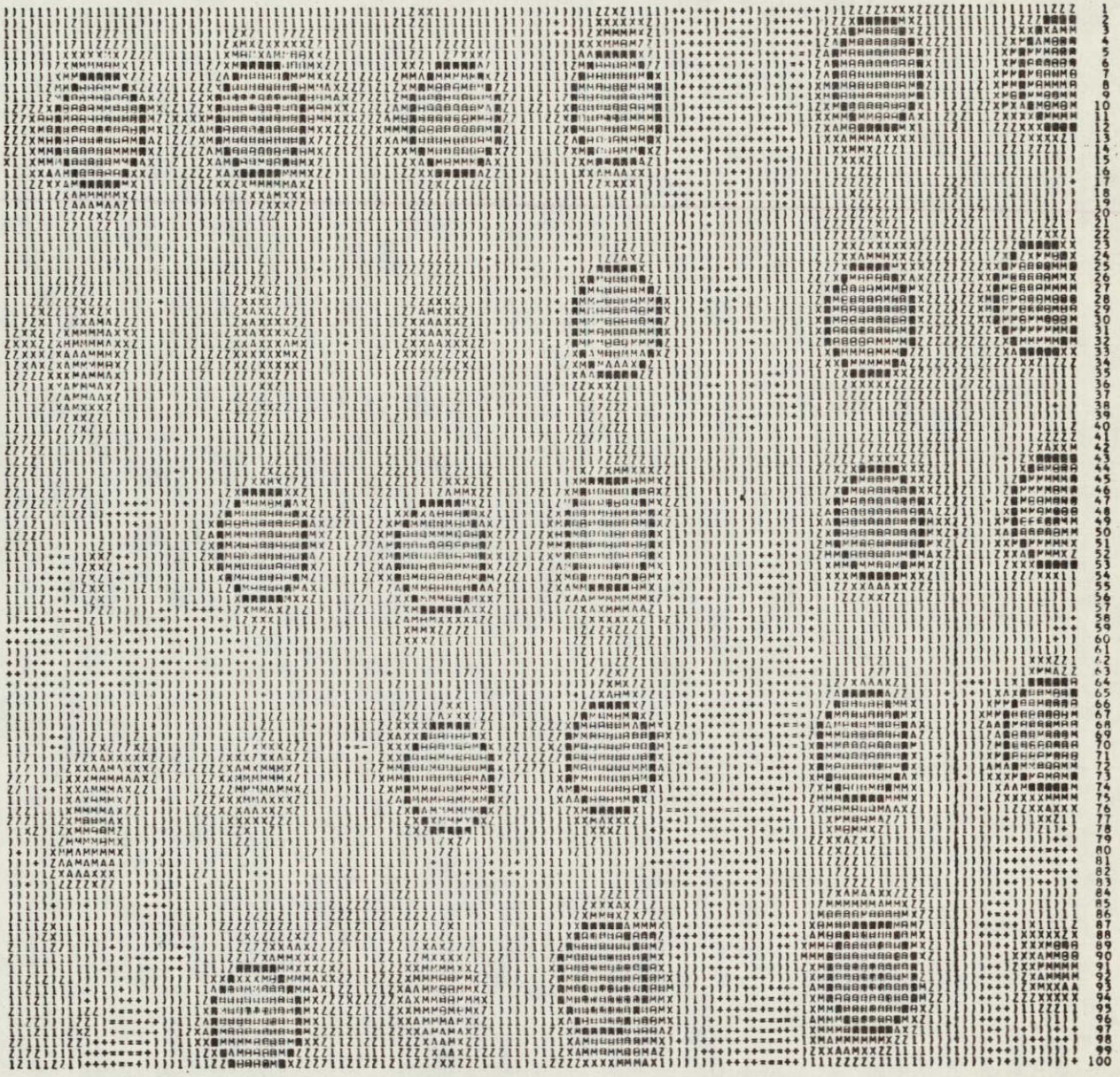


Figure 3.19 Detected Trees using a Threshold of 1000.

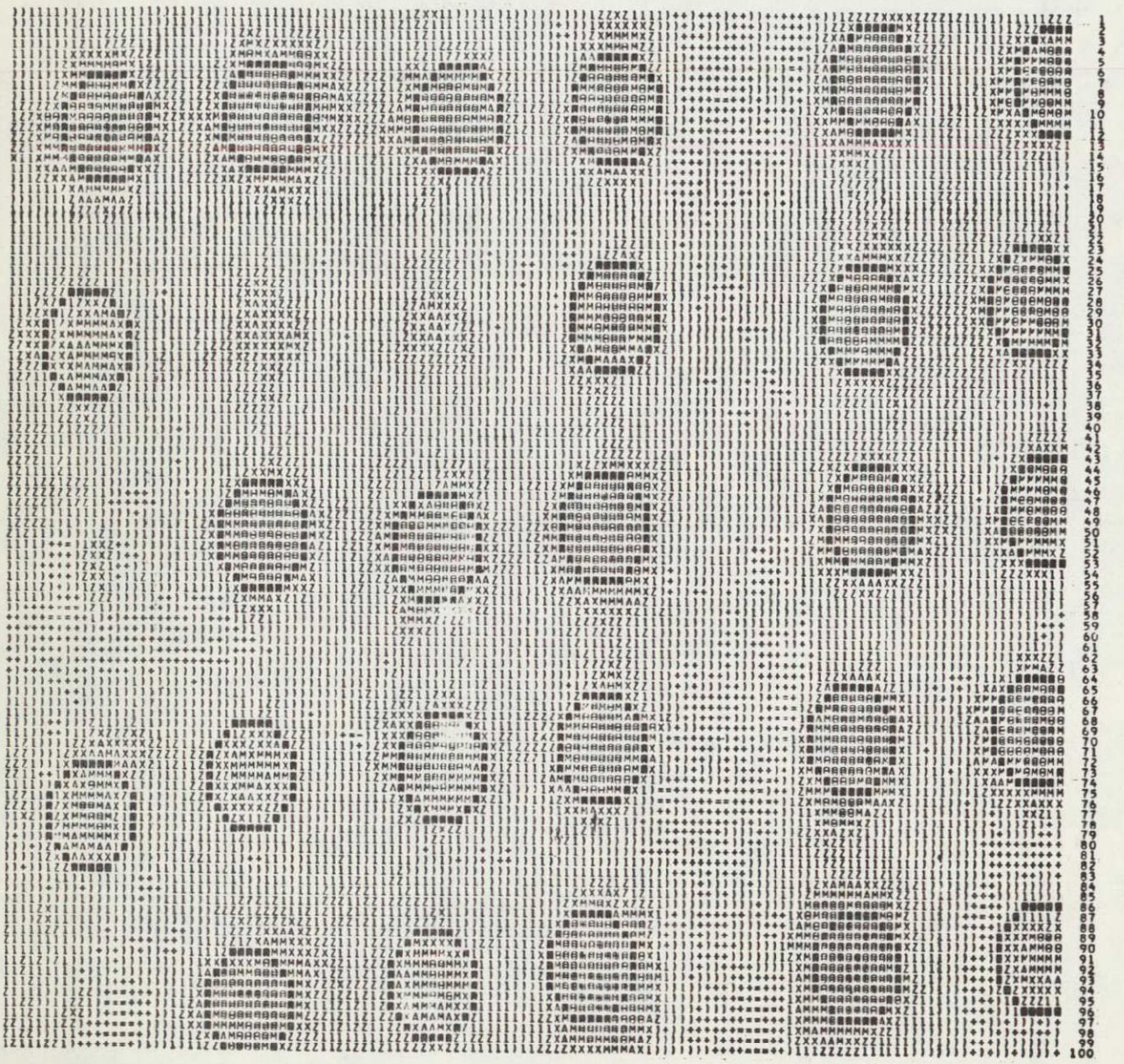


Figure 3.20 Detected Trees using a Threshold of 1500.

(ii) a sequence over the template in order of increasing radii, used for the initial evaluation of the five disk size cumulative sums,

(iii) a sequence used to flag the area surrounding a match position to suppress re-searching that area.

The initial search for approximate match positions, using the differential of the disk sum, is done using the smallest and the largest disk areas, to ensure detection of all objects of the appropriate size. (If a large mismatch of sizes occurs, the differential sequence does not pass rapidly through zero). A detailed analysis is made if the gradient sequences along both rows and columns passing through a given pixel consist of two positive terms followed by two negative terms, indicating passage through zero. When this occurs, the match coordinates are determined as the best position found in a five pixel by five pixel area surrounding the approximate match position.

The cumulative error which is computed is effectively the difference between the sum of the image values corresponding to the disk area and those of a border area. This is a measurement of the contrast between the circular region and its background. In order to compute this measure over five disk areas without summing over the same area more than once, partial sums are computed over annular rings, whose area is the increment in the disk areas. Then sums of terms over disk regions and border regions of the five specified areas are obtained by summing a small number of the partial sums.

The additional difference sums for the remainder of the five pixel square area are determined recursively, using the sequences of array elements for the new and deleted terms as the templates are moved along rows and columns of the digitized image. If at the position of maximum disk-border contrast a specified threshold is exceeded, the coordinates of the match position and the template number (1-5) are output on the printer and sequential mass storage.

When advancing to the next record of data, it is necessary to rotate the records of data held in a two-dimensional array in core by one row, and then

read the next record into the last row of the core array. (An alternate method, in which the evaluation sequences rather than the imagery data records are rotated, was tested. The execution time of this program was greater by one-third). Similarly, since the match can occur in any of five lines of data, the match parameters must be placed in arrays of five rows, which must be rotated for each record read.

In order to prevent multiple detection of the same match position, and to accelerate the detection process, a specified area surrounding a match position is not searched again. This is accomplished by marking the data around the match in some manner. This region may be rapidly distinguished from the data by the use of negative numbers. If the area to be skipped is large enough that no part of the template will fall on this area, the magnitudes of the elements may be changed from the data values in order to convey useful information, such as the number of pixels to advance along the scan line. This method was used in the 100x100 pixel test area, where the overall computation time was seen to decrease as a greater number of trees were detected (and hence more area was bypassed in the search). In general, positions of the template near its edges may fall on the marked area, and so the data values are simply negated, so that their magnitudes remain available. With this method, only the midpoint of the template is tested for falling on the marked area, and hence this test is very rapid. However, this method limits the dimensions of the bypassed area to the size of the template. If a larger bypassed area is desired, the corners of the template must be tested for falling on the marked area.

An additional programming consideration which accelerates the execution is the use of a look-ahead test, which cancels the search for passage of the differential through zero if there are not sufficient pixels remaining before collision with a previously detected match.

3.5.4 Large Area Test Site

The test site known as Bateman's orchard in Bibb County, Georgia was photographed by the MSFC I^2S camera at an altitude of 3000 feet. The blue band image was scanned and digitized to 64 levels in the density range 0 to 2, yielding a digitized image size of 600 lines X 850 samples (510,000 pixels). The orchard contained 28 X 38 rows of trees, and hence 1064 tree positions. The majority of the trees in the orchard have been examined from the ground, and it was found that 50.5 percent of the original trees remained alive. Extrapolation to the total area of 1064 grid positions indicates 537 live trees present.

The tree detection algorithm was applied to this digitized image, searching for five sizes of trees whose areas ranged from 100 pixels to 300 pixels in increments of 50. The corresponding diameters ranged from 11 pixels to 19 pixels in increments of 2. The threshold used was 7.0. (A threshold value of 7.0 means that, on the 64 level scale of densities, the average density over the disk region must exceed that over a ring-shaped border region of comparable area by 7.0).

In this test site, the tree crowns overlap in many cases, and an important parameter is the allowable separation between match positions, which strongly affects the execution time due to the bypassing of a specified area surrounding previously detected matches. A set of experiments was performed in which the bypassed area varied from 9 pixels on a side to 25 pixels. The results are presented graphically in Figure 3.21. The computation times do not include I/O operations. The machine used was an IBM 360/65. It may be noticed that bypassing larger areas decreases the execution time more rapidly than the number of detected trees decreases.

Sample printouts of the coordinates and parameters of each match position, and of the summary of match sizes and enumeration, are given in Figure 3.22. The five sizes of detected trees, for two values of the area bypassed around previous matches, are indicated on the blue band photograph in Figure 3.23.

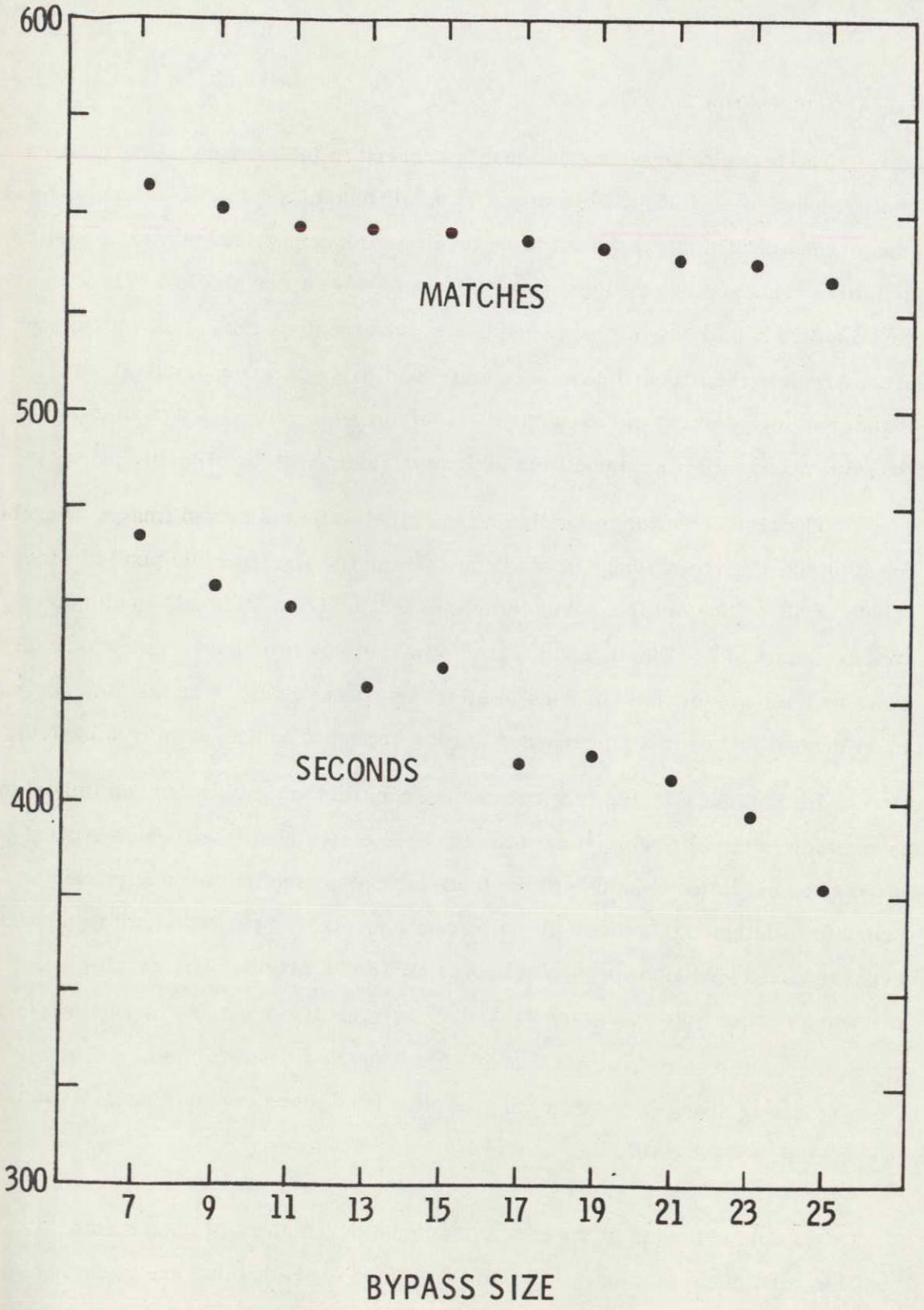


Figure 3.21 Computation time and tree count as a function of bypassed area.

COORDINATES OF MATCH POSITIONS

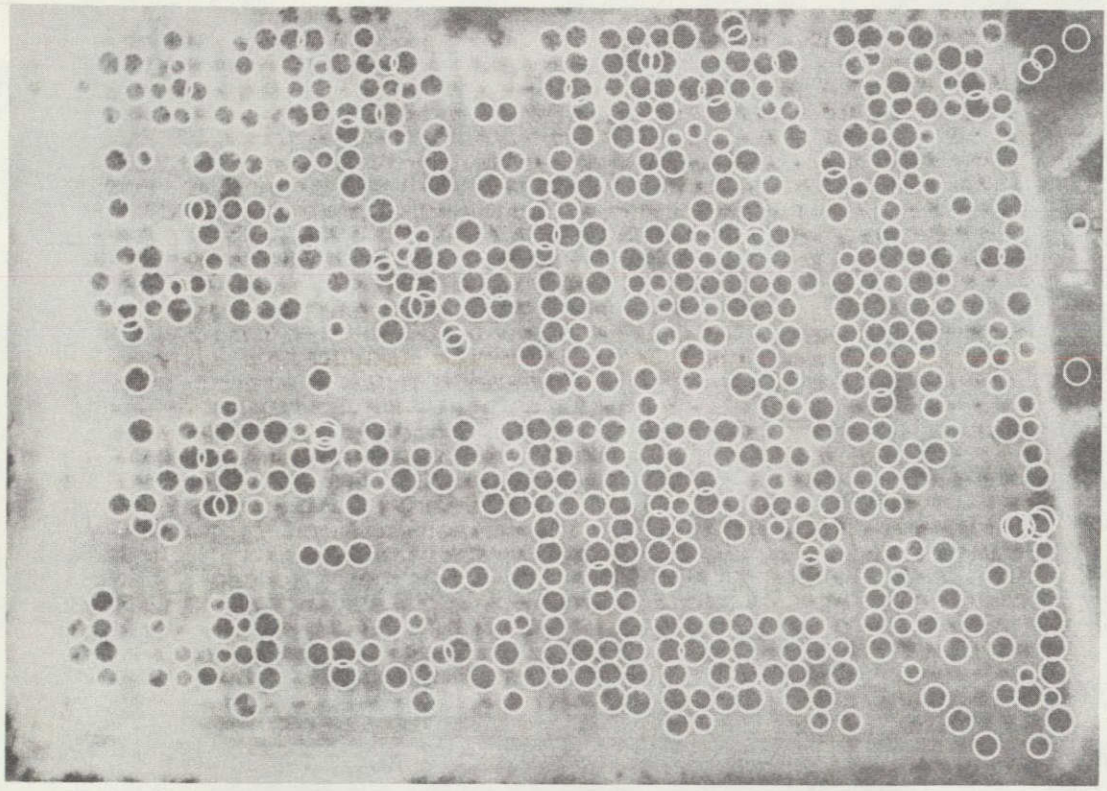
LINE NUMBER	SAMPLE NUMBER, RELATIVE AREA, CONTRAST																			
15	562	4	9.01																	
20	763	3	10.01																	
22	649	4	13.42																	
23	524	5	11.61	708	5	13.29	487	3	12.03											
24	441	4	14.35	669	5	11.56	506	3	11.15	689	2	9.31	828	4	7.50					
26	277	3	11.65	467	4	14.14														
27	201	4	12.93	421	4	11.07	221	4	10.59											
28	237	5	10.75																	
29	128	3	14.07																	
38	181	2	9.47																	
39	670	2	9.93	731	5	13.79	803	3	7.20	749	3	12.06								
41	566	5	11.99																	
42	489	5	9.68	604	5	15.29														
43	509	5	11.71	710	3	12.14	587	4	12.50											
44	469	4	12.75	545	3	13.44	441	4	11.93											
45	295	2	9.78	309	5	12.34	655	2	10.34											
46	259	4	12.21	278	3	11.30														
47	79	5	11.65	222	4	13.87	97	4	11.04	184	3	14.05								
48	244	1	7.37																	
49	132	1	8.28																	
58	691	5	14.05	731	1	9.90														
59	749	5	13.51																	
62	329	4	12.32	491	4	11.48	587	2	11.86	470	3	10.99	568	3	14.57	509	4	10.18	671	2	9.43
63	426	5	10.72	530	5	14.99														
64	298	2	9.52	443	5	11.33	547	5	8.25	281	3	10.84	244	3	10.85					
65	134	3	9.18	59	1	8.19	224	2	12.46											
66	22	2	8.43	311	2	9.35														
67	117	2	9.37	147	3	12.91														
68	99	3	9.89																	
74	771	3	13.08																	
75	694	3	8.97	752	5	10.21														
77	590	1	7.47	675	3	10.73	713	4	13.34	733	2	12.23								
79	552	5	13.05	493	2	10.68	607	3	7.17											
80	447	4	11.13	510	3	9.68	475	3	12.63											
82	243	3	11.26	369	3	14.83	387	3	13.91											
84	297	5	7.60	282	4	10.34	264	4	10.51											
85	168	3	12.69	226	4	12.40														
86	119	3	10.33	135	2	8.89	80	3	8.75											
87	188	3	10.38																	
93	772	2	8.24																	
95	533	1	10.07																	
97	204	5	9.47																	
98	658	5	14.97	696	5	11.85														
99	446	4	13.18	494	4	13.58	611	5	15.57	713	1	8.16	676	2	7.89					
100	478	5	9.53	332	5	7.59														
102	302	2	9.78	517	1	8.07														
103	553	2	9.60																	
114	775	4	13.98																	
116	678	3	11.22																	
117	699	3	11.37																	
118	574	5	15.25																	
119	266	3	12.46	516	5	12.14	106	1	8.64	246	2	10.11	429	4	9.82	497	3	8.82	641	5	12.54
120	391	4	9.33	554	3	10.60	172	4	11.86	333	4	11.04								
121	84	4	13.73	152	4	13.01														
122	228	3	9.76	191	3	11.49														

COUNT OF MATCH POSITIONS

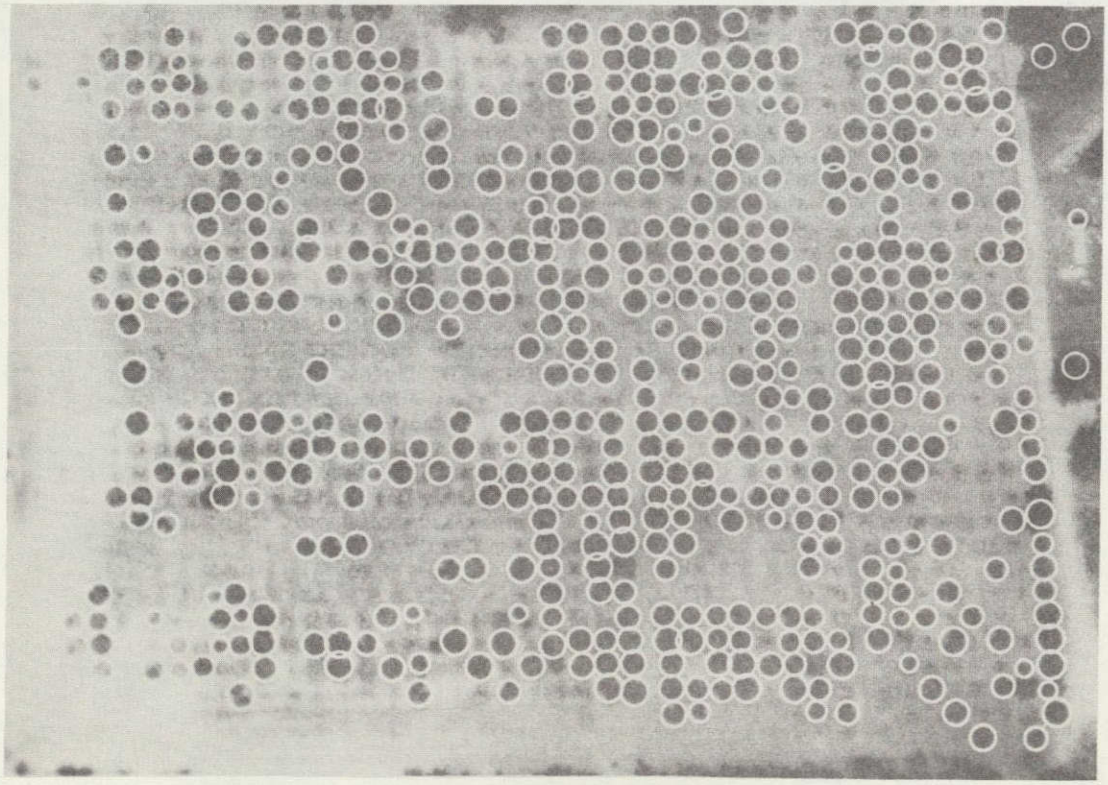
NUMBER	NOMINAL AREA & DIAMETER		ACTUAL PIXEL AREA & DIAMETER	
37	100	11.3	97	11
73	150	13.8	145	13
169	200	16.0	193	15
150	250	17.8	241	17
115	300	19.5	293	19
544				

ELAPSED TIME -- MINUTES - 6 SECONDS - 49.59

Figure 3.22 Sample printout of detection algorithm results.



BYPASS SIZE = 7



BYPASS SIZE = 25

Figure 3.23. Detected trees for two values of the bypassed area

3.5.5 Performance Summary

The performance of the algorithm will be discussed with regard to computation speed and detection accuracy.

As stated previously, the computation speed increases with the number of detected trees, due to the bypassing of areas surrounding matches. The present scene is not advantageous in this regard, since one-half of the trees are missing. Also, the detection rate will depend on the resolution at which the imagery is digitized. It is apparent that, for good detection and sizing, the tree diameters should be on the order of several pixels, ranging from eleven to nineteen in the present case. With an IBM 360/65, the digitized image was analyzed at an average rate of approximately 1350 pixels per second. (377 seconds were required with bypass size 25.) Expressed in a different fashion, the rate of analysis is 3.8 tree cells per second, where a tree cell is a square region of side the average distance between rows of trees, which is 18.8 pixels.

Using the lower image of Figure 3.23, it was determined manually that, within the 28 by 38 row area of the orchard, fifteen trees were not detected. Subtracting the nineteen trees detected outside of the orchard from the total of 534 detected trees, we obtain a detection accuracy of 515/530 or 97 percent.

IV. ANALYSIS OF THE INFRARED BAND PHOTOGRAPH

It has been known for some time that healthy vegetation is strongly reflective in the near infrared spectral band, while vegetation under stress from any source is markedly less reflective. This trait has been widely exploited in the determination of disease and insect infected areas of deciduous forests and citrus orchards. Encouraging results in peach orchard photography have been obtained by Georgia Institute of Technology in cooperation with the USDA Southeastern Fruit and Tree Nut Research Station.[2] In this work, color infrared transparencies were viewed through color separation filters by a television camera. The image was then analyzed by computer, and it was found that orchard sections containing unhealthy trees were discriminated by their infrared reflectances.

Bateman's orchard in Bibb County, Georgia, was the site of a 9-year experiment in various treatment methods. An aerial photograph, with the treatment areas marked, is given in Figure 4-1. Herbicides were used to control weeds in strips lettered "H". Weeds were controlled by disking in strips lettered "D". Blocks of live trees in the disked strips indicate where Fumazone was injected for nematode control. The almost perfect stands within the "H" strips indicate where both herbicides and Fumazone were applied.

The infrared band image taken by the MSFC I^2S camera was scanned and digitized to 64 levels in the density range 0 to 2. In order to establish the existence of variations in infrared reflectance as a function of tree vigor, the digitized image was thresholded at several closely spaced density levels. The digitized image was then rewritten on a film writer with the regions having densities less than the thresholds being blacked out. The results for four thresholds are given in Figure 4-2. The healthiest trees, having the highest reflectance, will be blacked out at the lowest thresholds. It can be seen that only the healthiest trees have densities of 25 or less. Setting the threshold at 28 includes nearly every tree within the perfect stands treated by herbicides and Fumazone, while excluding

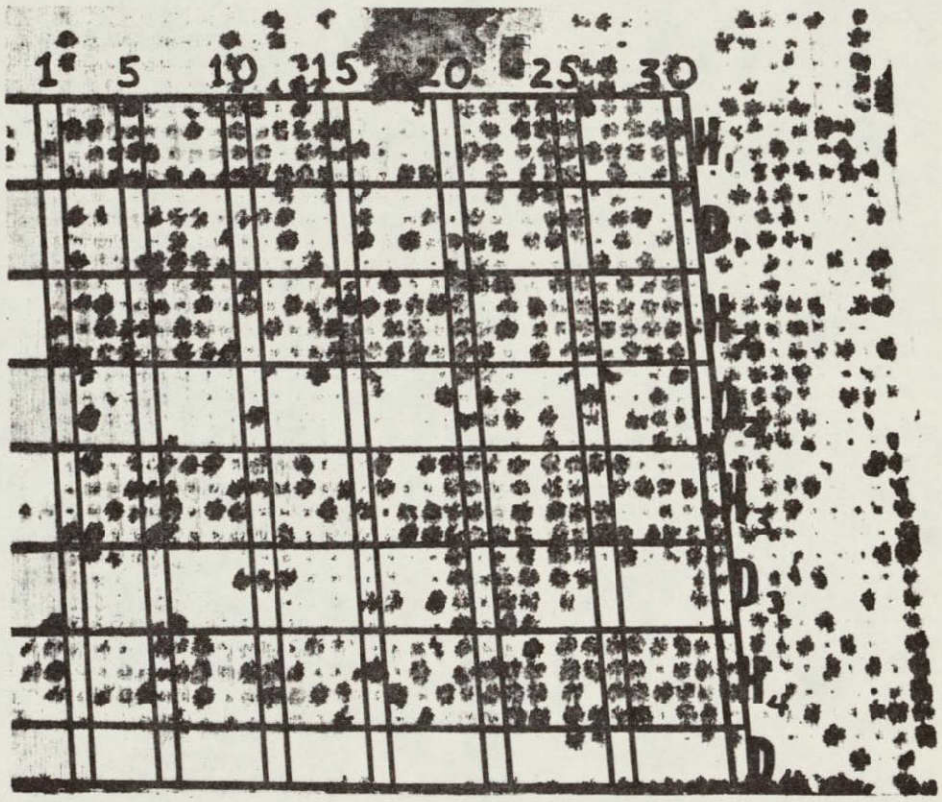


Figure 4-1. Bateman's Orchard Showing Treated Areas

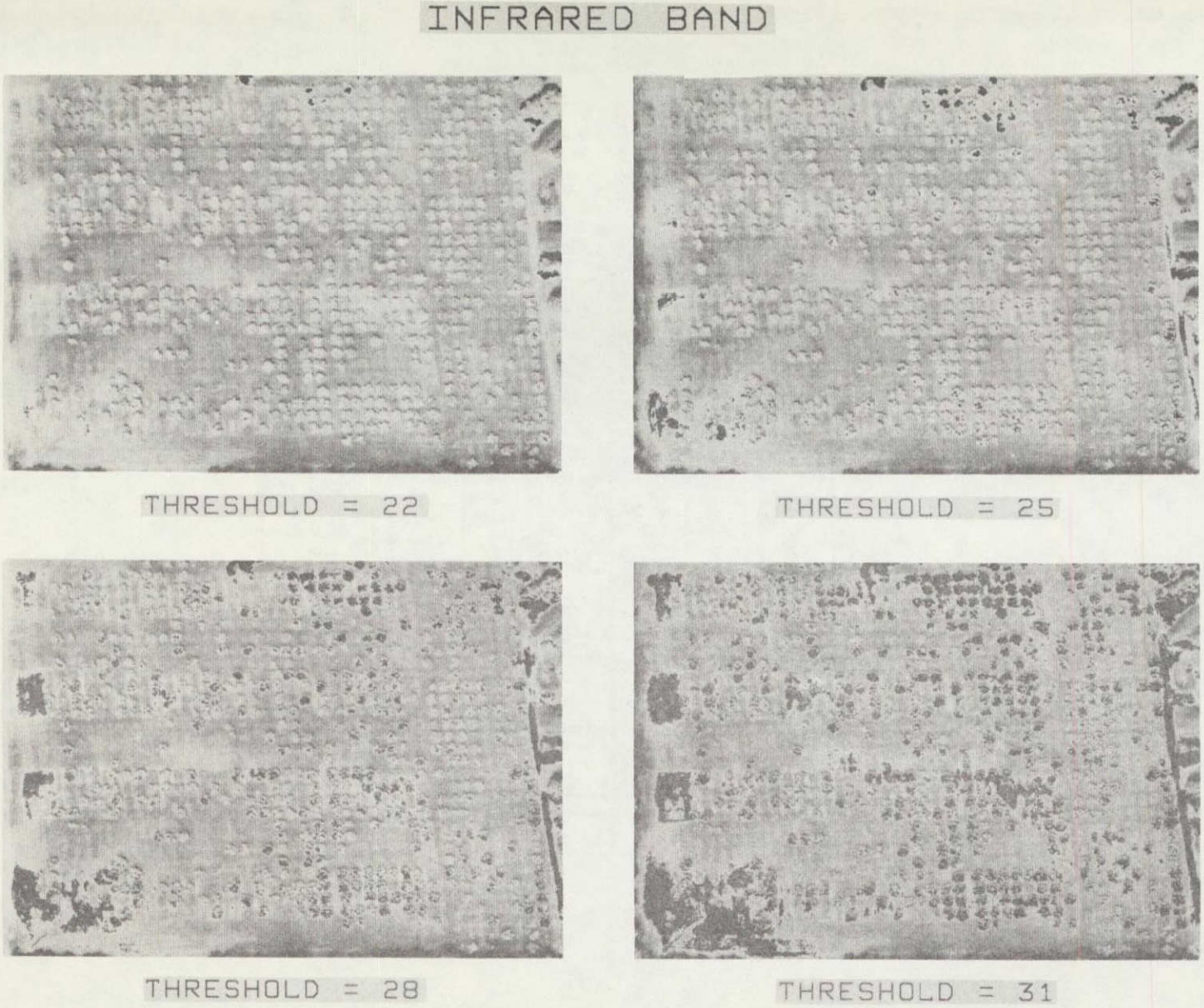


Figure 4-2. Infrared Photograph of Bateman's Orchard Showing Four Density Thresholds

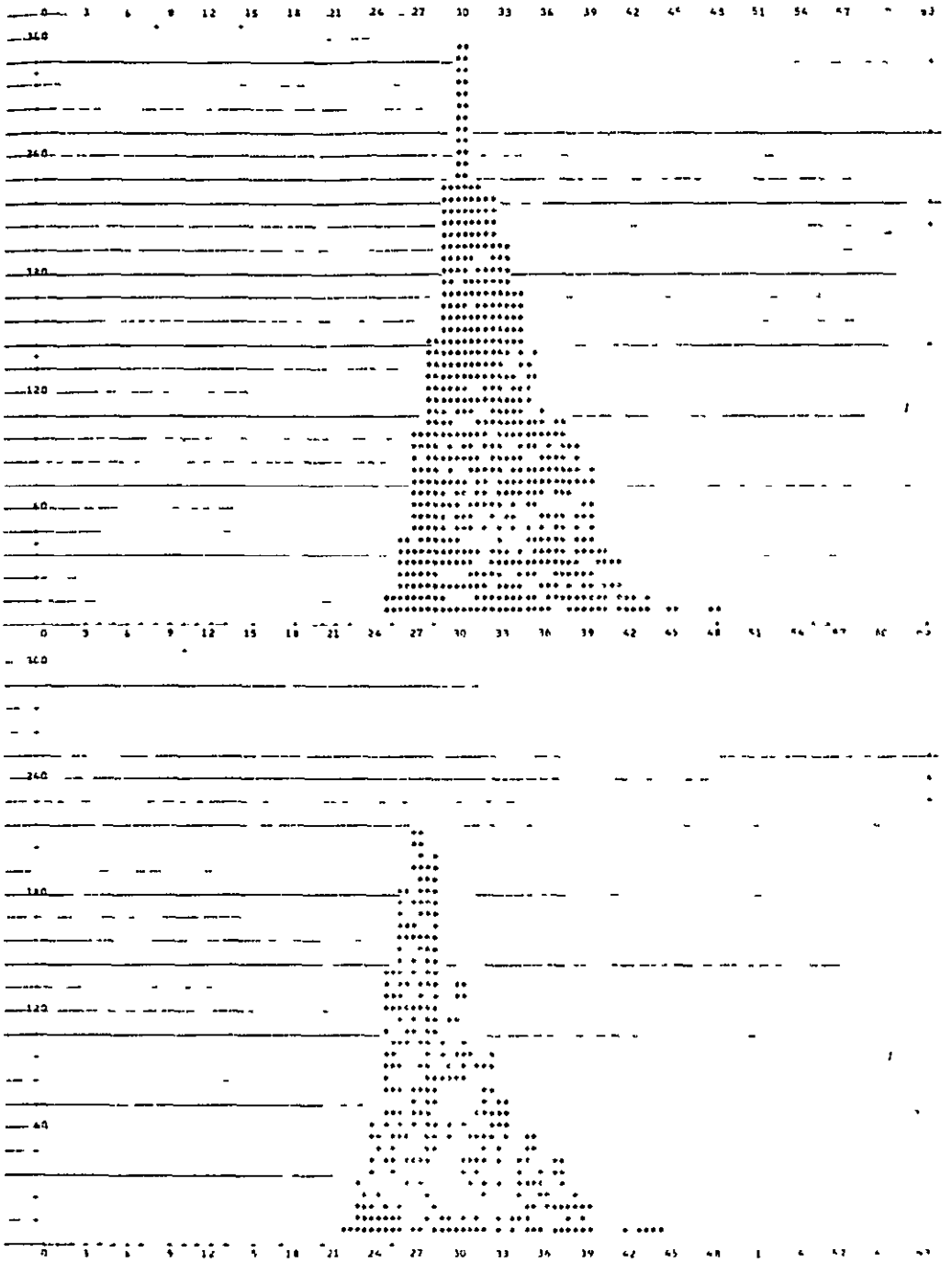


Figure 4-3. Histograms of Declining and Healthy Trees

V. ANALYSIS OF MULTIBAND AERIAL PHOTOGRAPHY

5.1 INTRODUCTION

The photography used in this study was obtained with an International Imaging Systems (I^2S) Mark I four-band multispectral camera, using Kodak type 2424 black and white infrared film and type 2420 duplicating film. The four spectral bands were defined by filters, whose peak transmissions occur at wavelengths 0.430, 0.530, 0.635, and 0.800 microns. The transmittance curves are shown in Figure 5-1.

5.2 IMAGE REGISTRATION

The four photographic transparencies were digitized separately and, hence, require digital registration of the imagery. This was accomplished by choosing five template areas in one photograph and searching for match positions in the other three images by means of a sequential similarity detection algorithm.[3] An error threshold sequence was calculated based on a 10 percent probability of exceeding the threshold at the n-th term of the error summation (see Figure 3-10). The templates from the first image were then used to search for matches in each of the three remaining images and the match positions were taken as those with minimum cumulative error.[4] The four digitized files of data were then truncated by the appropriate numbers of pixels to produce registered files. These are then merged so that the four density levels from corresponding locations of the imagery form a four-dimensional feature vector. These steps are illustrated in Figure 5-2.

5.3 TRAINING SAMPLE SELECTION

The data values for all four spectral bands were then selected from the template regions corresponding to 50 declining and 30 healthy trees. The total numbers of samples obtained were 9906 for the declining and 7718 for the healthy. This is a prohibitive number of samples when using a nonparametric method of

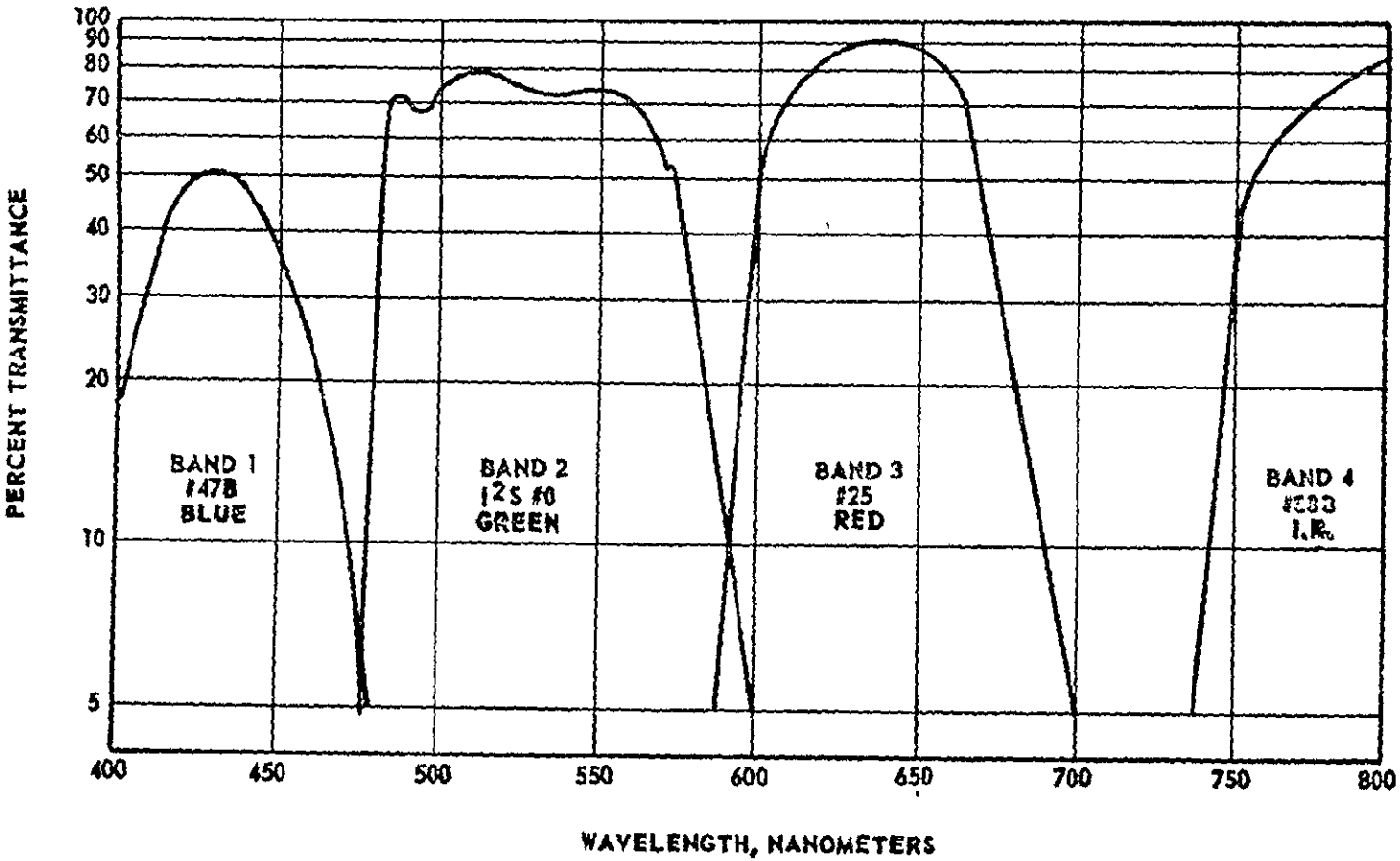


Figure 5-1. Filter Transmittances of the Multispectral Camera

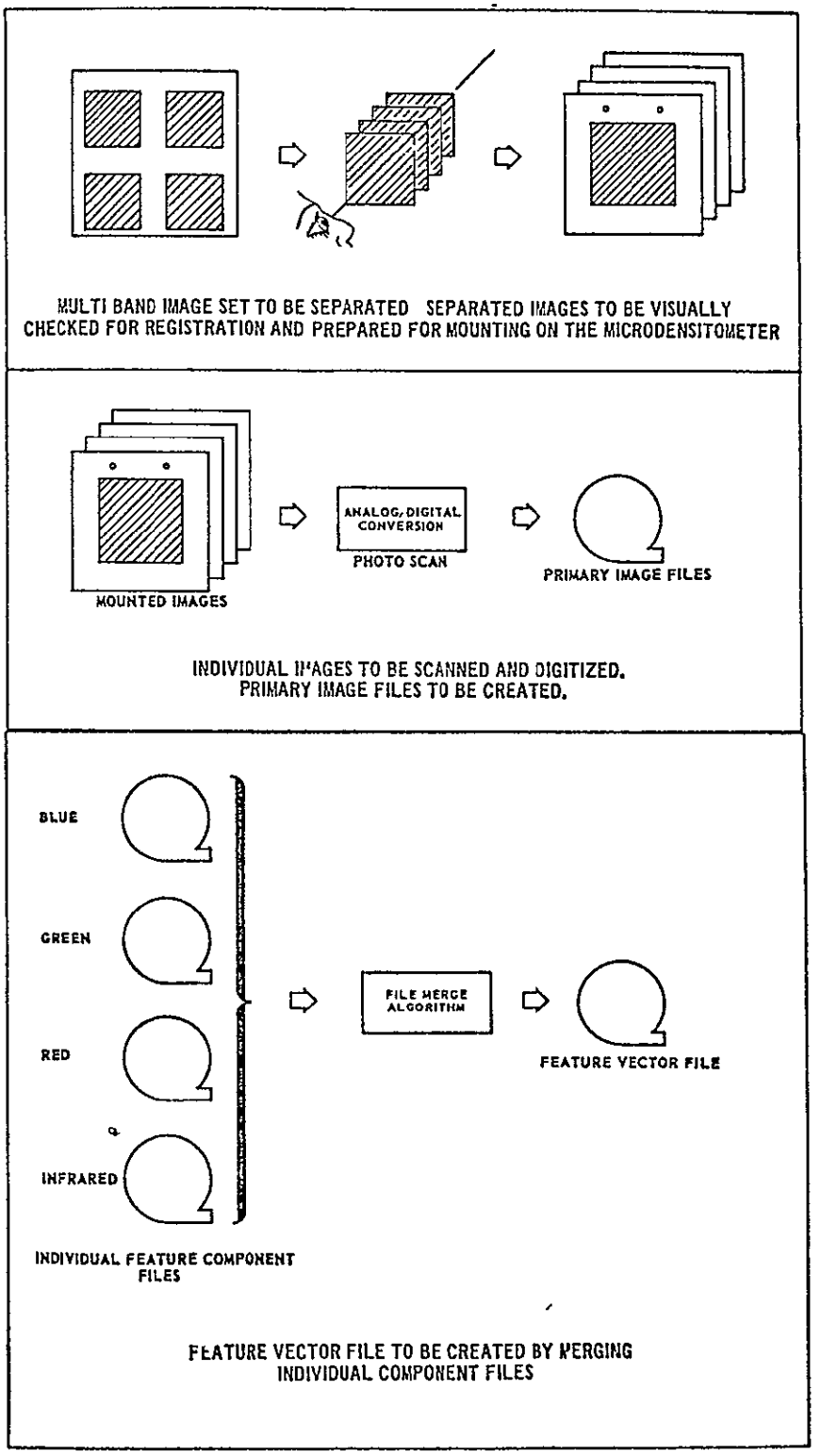


Figure 5-2. Summary of Preprocessing Requirements

determining a set of classification functions. The number of training samples was reduced by two methods.

The simplest method consists of sampling the data values at equal intervals such that a tractable number (e.g., 500) of samples is obtained. A second method consists of obtaining the four-dimensional histogram of the data in which each cell of the histogram represents a unique set of the four data values forming a feature vector.[6] The result is 8655 cells for the declining tree data and 6856 cells for the healthy. A reduction in the number of cells to 993 (healthy) and 737 (declining) was then accomplished by eliminating all cells having a frequency of occurrence of one. This procedure is physically acceptable as these cells are at the extreme edges of the distributions and probably represent mixtures with soil data values, since the irregular shapes of the tree crowns permit such a mixture within the circular template regions.

Succeeding calculations of the classification functions were coded to use the four data components and the frequencies of each cell and thus avoid the repetition of calculations for feature vectors which are identical. These two methods will be referred to as the sampling method and the cell method in the following sections.

5.4 SUPERVISED CLASSIFICATIONS

The two supervised classification methods which were applied are maximum likelihood and sequential linear.

The maximum likelihood algorithm is based on the assumption that the samples within a given class are distributed according to a normal (Gaussian) multivariate density function.

$$P = \frac{1}{\sqrt{(2\pi)^n D}} \; e^{\frac{-(X-m)^T K^{-1} (X-M)}{2}}$$

where M is the mean vector, K is the covariance matrix, and D is the determinant of K.

Such a distribution is completely specified in terms of a mean vector and a covariance matrix for each class, which are determined from the training samples. In the classification phase, the probability of an unknown feature vector belonging to each class is calculated, and assignment is made to the class for which the probability is the greatest.

Linear discriminant functions are also determined for each class, using the training samples. If the linear discriminant function corresponding to a certain class is evaluated by substituting an unknown feature vector, assignment is made to that class when the result is positive. If the assignment is not made, the data sample belongs to one of the remaining classes, and successive discriminant functions are evaluated. This effectively means that the multiclass problem is treated as a series of two-class problems, in both the training phase (determination of linear discriminant functions) and the classification phase (assignment of data samples to classes). Visualizing the data from different classes occupying different regions in the spectral measurement space, one can recognize that the outer clusters can be separated from the inner clusters and that this process can be executed sequentially. Thus it is necessary to order the classes of data such that each can be separated from the remaining classes by a linear function (a hyperplane in the spectral measurement space). This is accomplished by computing for each spectral band the totals of the separations between all pairs of points in different classes (interclass) and within each class (intraclass). The optimum separation is obtained when the interclass distance is maximized and the intraclass distance is minimized. The coefficients of the discriminant functions may then be determined by maximizing a criterion such as the distance of the training samples from the discriminant hyperplane.[7] The processing flow is shown in Figure 5-3. Applications of these algorithms have been discussed in greater detail in other publications.[8,9]

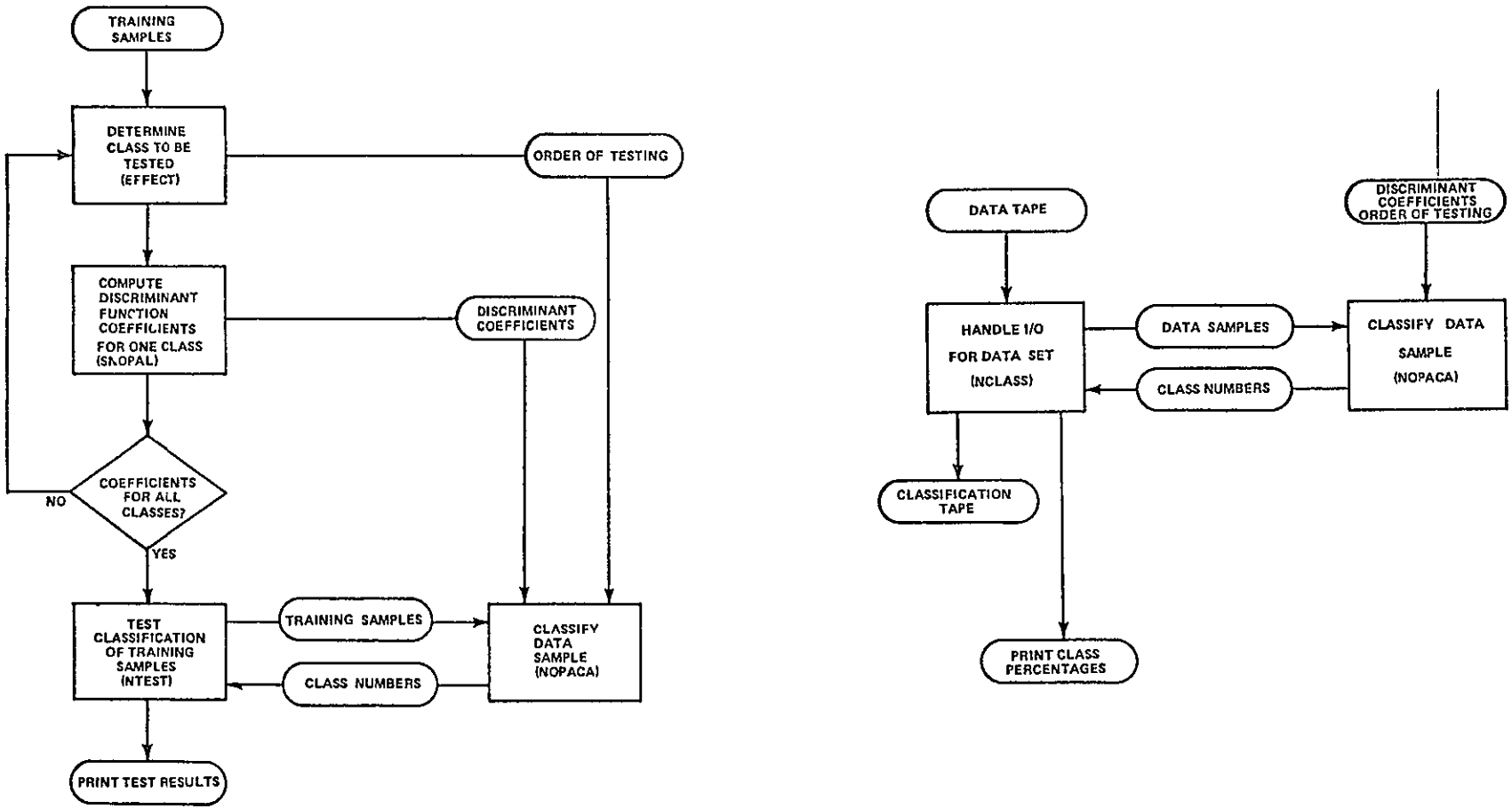


Figure 5-3. Sequential Linear Classifier System

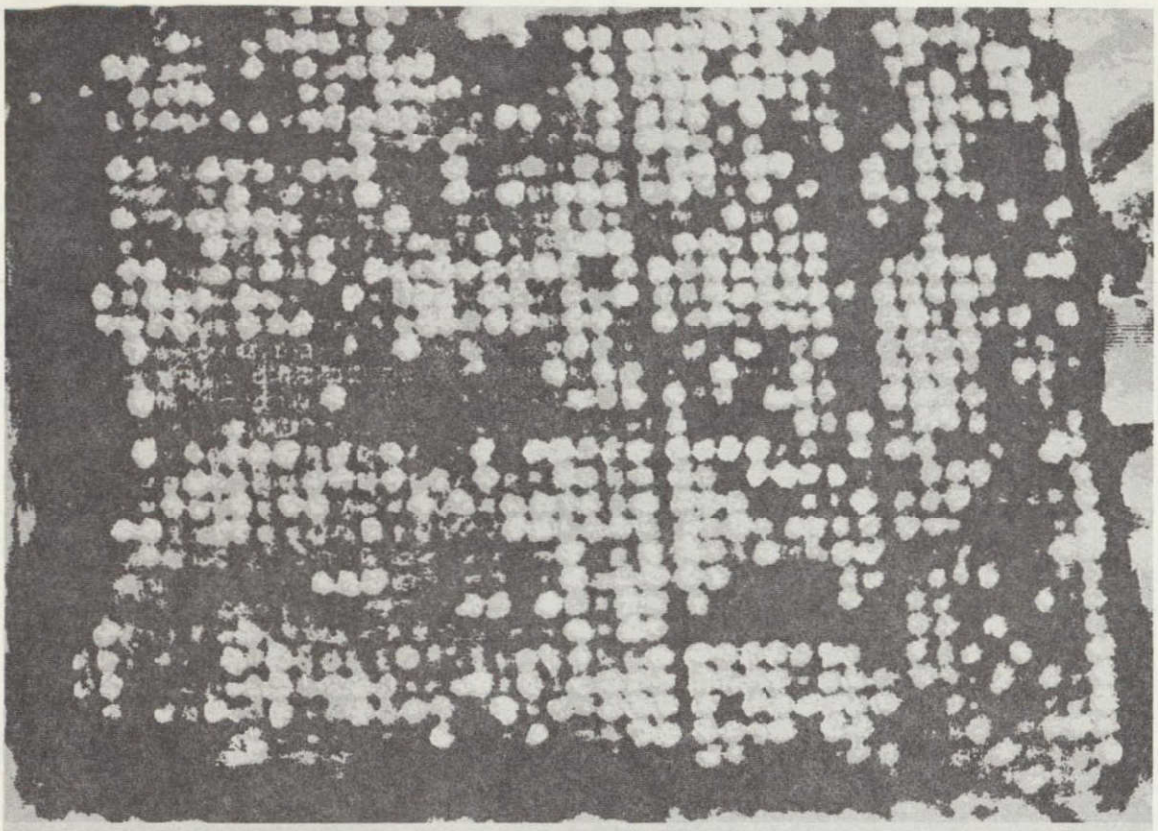
Training samples derived by the sampling and the cells methods were used in maximum likelihood and sequential linear classifiers. The digitized scene, consisting of 510,000 pixels, was classified on a pixel-by-pixel basis into three classes--healthy trees, declining trees, and soil. These results were then further analyzed to obtain the classification of each tree. The tree classes were taken as the majority class within the area of each tree as determined by the match coordinates and the five sizes of templates.

The two classification maps are shown in Figure 5-4, obtained using the cells method of training samples selection and data from the four spectral bands. The classifications of the previously detected trees are shown in Figure 5-5. An example of the output of the tree classification algorithm is given in Figure 5-6. The classification results are given in Table 5-1. When using the infrared band photographs only, the two classification algorithms give similar results. The density levels assigned to the three classes are as follows:

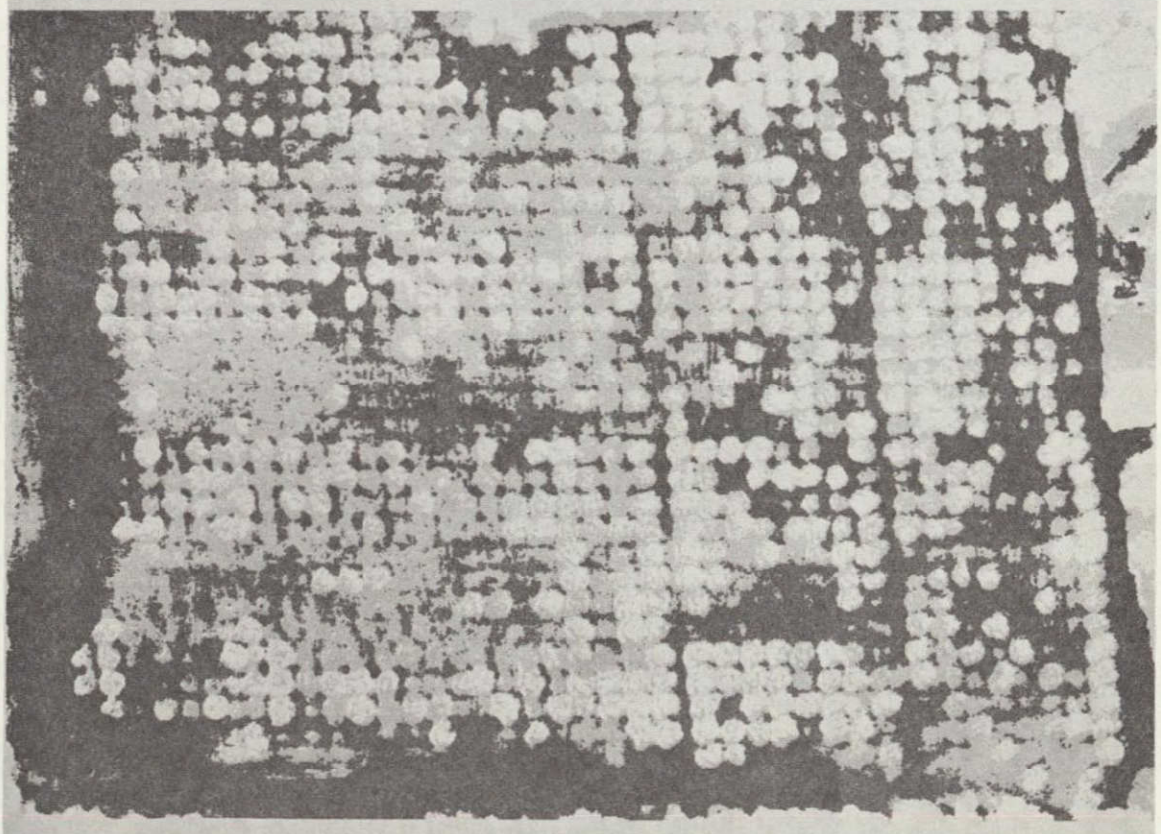
LINEAR:	0-30 healthy	31-38 declining	39-63 soil
GAUSSIAN:	0-30 healthy	31-37 declining	38-63 soil

5.5 PERFORMANCE SUMMARY

The goal of the multiband classification procedure is to determine those trees which are remaining healthy and those which are in decline. One problem in this procedure is the variation within a single tree crown. However, if the classification is sufficiently accurate, this may indicate damage in some limbs of the tree. An attempt was made to overcome this problem by classifying trees as healthy or declining according to the majority of the pixels. It may be seen from Table 5-1 that this procedure results in a classification of the 80 ground survey trees with an accuracy of approximately 70 percent. Additional ground survey information included the statistics of 989 trees, but not the status of individual trees. Of 499 live trees, the ground survey indicated 46.69 percent as healthy and 53.31 percent as declining. The computer classifications also indicate a majority of trees in the declining category.

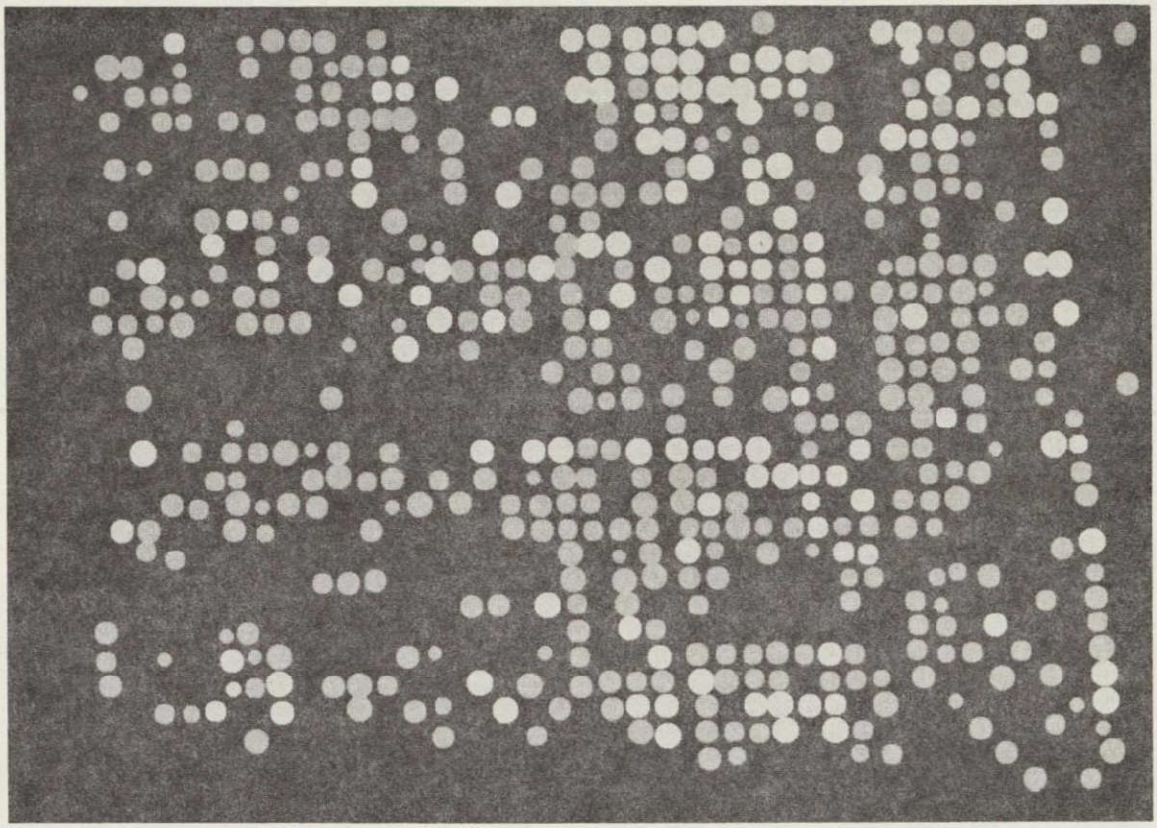


SEQUENTIAL LINEAR CLASSIFICATION

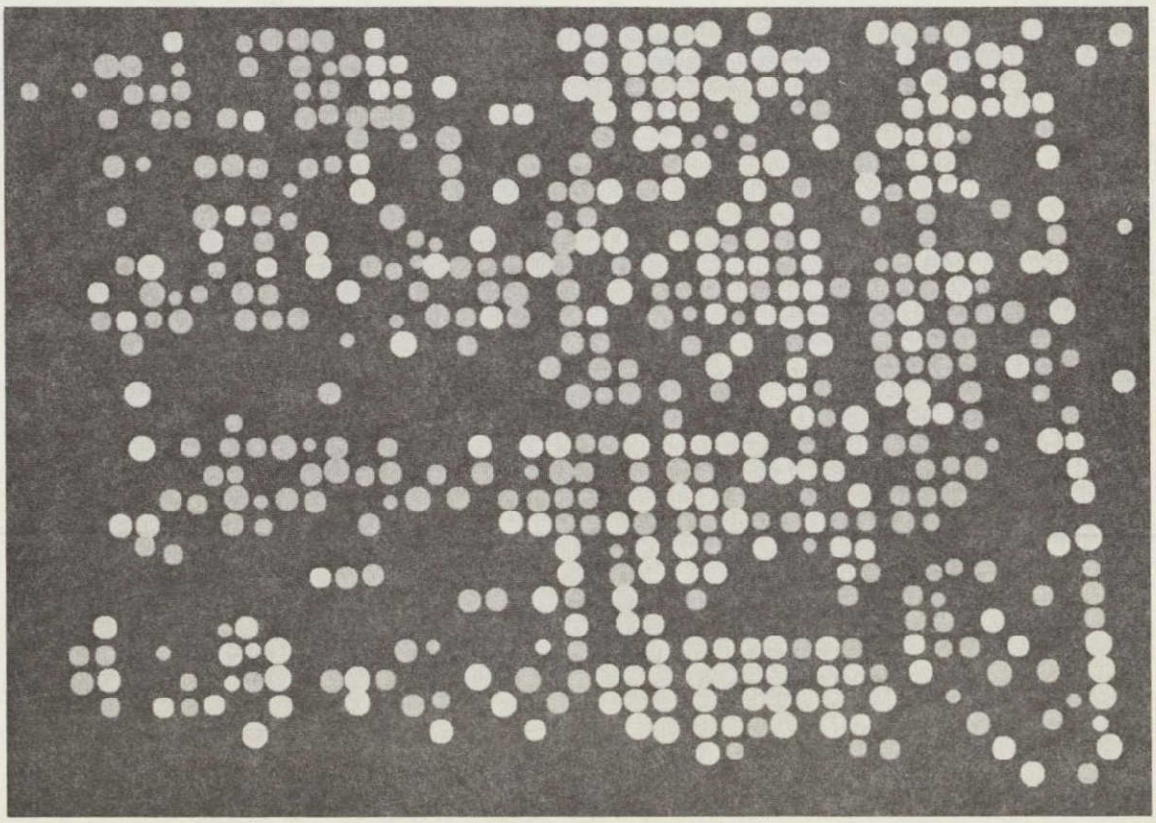


MAXIMUM LIKELIHOOD CLASSIFICATION

Figure 5-4. Classification maps obtained by two methods.



SEQUENTIAL LINEAR CLASSIFICATION



MAXIMUM LIKELIHOOD CLASSIFICATION

Figure 5-5. Classification of the detected trees.

CLASSIFICATION SUMMARY

CLASS	SIZE NUMBER	SAMPLES	PERCENT
DECLINE	1	31	5.81
DECLINE	2	55	10.30
DECLINE	3	100	18.73
DECLINE	4	70	13.11
DECLINE	5	27	5.06
		283	53.00
HEALTHY	1	6	1.12
HEALTHY	2	13	2.43
HEALTHY	3	68	12.73
HEALTHY	4	81	15.17
HEALTHY	5	83	15.54
		251	47.00
SOIL	1	0	0.0
SOIL	2	0	0.0
SOIL	3	0	0.0
SOIL	4	0	0.0
SOIL	5	0	0.0
		0	0.0
TOTAL		534	

Figure 5-6. Output of the tree classification algorithm.

Table 5-1. Classification Results

Classification Method	Training Data Accuracy By Pixels	Training Data Accuracy By Trees	Orchard Classifications (Percentages)						Classification Time (Seconds)
			By Pixels			By Trees			
			Healthy	Declining	Soil	Healthy	Declining	Soil	
Sampling Method									
Maximum Likelihood	73.7	76.25	18.62	31.14	50.23	37.39	62.43	0.18	564
Linear	70.9	67.5	14.88	33.29	51.83	47.01	52.27	0.73	126
Cells Method									
Maximum Likelihood	79.1	77.5	16.48	43.04	40.48	47.00	53.00	0.0	524
Linear	76.8	67.5	11.24	26.73	62.02	27.15	71.35	1.50	71
Infrared Maximum Likelihood	62.3	68.75	17.09	32.40	50.51	41.01	42.13	16.85	183
Infrared Linear	63.2	80.0	17.09	39.17	43.73	39.70	51.12	9.18	47
Average	71.0	72.9	15.90	34.30	49.80	39.88	55.38	4.74	
Ground Survey						46.69	53.31		

The infrared band classifications were performed since it is known that decline would be indicated primarily in the infrared spectral range. However, the accuracy is diminished in this case. This is probably an indication of the fact that the spectral band being used is in the very near infrared. It is anticipated that better results could be obtained with better infrared response in the detector used.

REFERENCES

1. J.A. Payne, W.G. Hart, M.R. Davis, L.S. Jones, D.J. Weaver and B.D. Horton, "Detection of Peach and Pecan Pests and Diseases with Color Infrared Aerial Photograph," Proc. Third Biennial Workshop on Color Aerial Photography in the Plant Sciences, 1971.

2. G. William Spann, "A New Method for Studying Peach Tree Decline," USDA Seminar on Infrared Photography of Peach Short Life Sites, 1973 (unpublished).

3. D. S. Barnea and H. F. Silverman, "A Class of Algorithms for Fast Digital Image Registration," IEEE Trans. Computers, Vol. C-21, February, 1972.

4. The FORTRAN programs required were designed and coded by H. K. Ramapriyan.

5. H. K. Ramapriyan, "A Multilevel Approach to Sequential Detection of Pictorial Features," IEEE Trans. Computers, Vol. C-25, January 1976.

6. The FORTRAN programs required were designed and coded by B. V. Dasarathy.

7. Ho, Y. C. and Kashyap, R. L., "A Class of Iterative Procedures for Linear Inequalities," J. Siam on Control, Vol. 4, 1966.

8. Bond, A. D. and Atkinson, R. J., "An Integrated Feature Selection and Supervised Learning Scheme for Fast Computer Classification of Multi-Spectral Data," Proc. Earth Resources Observation and Information Analysis System Conference, U. of Tennessee Space Institute, March 1972.

9. Bond, A. D., Ramapriyan, H. K., Atkinson, R. J., Hodges, B. C., and Thomas, D. T., "Image Analysis Techniques Associated with Automatic Data Base Generation," AIAA Computer Network Systems Conference, April, 1973.

APPENDIX A

A FORTRAN listing of the complete detection package follows. The core required on the IBM 360/65, when using standard FORTRAN input and output tapes, is 148 K bytes [K= 1024] .

```
      DIMENSION S(29,850), TMPLT(25,25)
      LOGICAL TMPLT
      DATA NR,IAREA1,MBYP,THRESH,NRECS,NSAMP /25,50,7,7.0,600,850/
C
C  NR      - LENGTH OF A TEMPLATE SIDE
C  IAREA1  - TEMPLATE AREA INCREMENT
C  MBYP    - LENGTH OF A BYPASSED AREA SIDE
C  THRESH  - THRESHOLD ON AVERAGE DISK AND BORDER DENSITY DIFFERENCE
C  NRECS   - NUMBER OF RECORDS TO BE READ
C  NSAMP   - NUMBER OF SAMPLES PER RECORD
C
      CALL DSEGNC (S, TMPLT, NR, IAREA1, MBYP)
C
      REWIND 10
      REWIND 11
      CALL DETECT (S, S, NR, NR+4, IAREA1, THRESH, NRECS, NSAMP)
      END FILE 11
      STOP
      END
```

```
C
      SUBROUTINE DSEQNC (ST, TMPLT, NR, IAREA1, MBYP)
C
C  DEFINE EVALUATION SEQUENCES
C
      INTEGER ST(1), HSN(25,10), VSN(25,10), HSD(25,10), VSD(25,10)
      DIMENSION NS(10), ISEQ(625), KAREA(11), ASIZE(5,2), MSEQ(500)
      LOGICAL TMPLT(NR,NR)
      COMMON /SEQNCE/ NS,HSN,VSN,HSD,VSD,ISEQ,KAREA,ASIZE,MS,MSEQ
C
C  FIND ELEMENTS PRESENTLY DISK, PREVIOUSLY BACKGROUND, AND VICE VERSA
C  ASSUME A SEARCH AREA OF SIZE (NR+4) LINES X NR SAMPLES
C  COMPUTE VERTICAL MOTION ELEMENTS BY TRANSFORMATION I = J, J = NR-I+1
C
      N = 0
      DO 43 N1=1,2
      DO 43 N2=1,5
      N = N + 1
      N3 = 0
      N4 = 0
C
C  OBTAIN TEMPLATE WITH DISK AREA 'IAREA'
C
      IAREA = N1 * (N2+1) * IAREA1
      CALL TEMPLT (TMPLT, IAREA, NR)
C
      DO 32 I=1,NR
      DO 32 J=0,NR
      IF (J.EQ.0) GO TO 44
      IF (J.EQ.NR) GO TO 33
      IF (.NOT.TMPLT(I,J).AND.TMPLT(I,J+1)) GO TO 35
      IF (.NOT.TMPLT(I,J+1).AND.TMPLT(I,J)) GO TO 36
      GO TO 32
   44 IF (TMPLT(I,J+1)) GO TO 35
      GO TO 32
   33 IF (TMPLT(I,J)) GO TO 36
      GO TO 32
C
C  COMPUTE NEW DISK ELEMENTS
C
   36 N3 = N3 + 1
      HSN(N3,N) = (NR+4)*(J-1) + I + 2
      VSN(N3,N) = (NR+4)*(NR-I) + J + 2
      GO TO 32
C
C  COMPUTE DELETED DISK ELEMENTS
C
   35 N4 = N4 + 1
      HSD(N4,N) = (NR+4)*(J-1) + I + 2
      VSD(N4,N) = (NR+4)*(NR-I) + J + 2
   32 CONTINUE
   43 NS(N) = N3
C
C  DEFINE ISEQ IN ORDER OF RADII BEGINNING AT THE DISK CENTER
C
      J1 = 0
      DO 22 J=1,NR
```

```
            J1 = J1 + 1
            ST(J1) = II**2 + JJ**2
         22 ISEQ(J1) = (NR+4)*(JW-1) + IW + 2
            CALL SORTSL (ST, ISEQ, 1, NR*NR)
      C
      C  COMPUTE SEQUENCE USED FOR DEFINING REGION SURROUNDING A MATCH
      C
            MID = (NR+1)/2
            MRAD = MBYP / 2
            IREC1 = MID - 2
            IREC2 = MID + MRAD + 2
            ISAMP1 = MID - MRAD
            ISAMP2 = MID + MRAD + 1
            MS = 0
            DO 10 IW=IREC1,IREC2
            DO 10 JW=ISAMP1,ISAMP2
            MS = MS + 1
         10 MSEQ(MS) = (NR+4)*(JW-1) + IW + 2
      C
      C  COMPUTE DISK AND BORDER SEQUENCES AND AREAS
      C
            DO 60 I=1,11
            KAREA(I) = (I+1)*IAREA1
         60 CALL TEMPLT (TMPLT, KAREA(I), NR)
            DO 5 NSIZE=1,5
            ASIZE(NSIZE,1) = KAREA(NSIZE)
          5 ASIZE(NSIZE,2) = KAREA(2*NSIZE+1) - KAREA(NSIZE)
            RETURN
            END
```

```
C
      SUBROUTINE TEMPLT (TMPLT, IAREA, NR)
      LOGICAL TMPLT(NR,NR)
C
      R2 = IAREA / 3.14159265
      IAREA = 0
      DO 1100 I=1,NR
      AI = I - (NR+1)/2
      DO 1100 J=1,NR
      AJ = J - (NR+1)/2
      TMPLT(I,J) = .FALSE.
      IF (AI**2+AJ**2.GT.R2) GO TO 1100
      IAREA = IAREA + 1
      TMPLT(I,J) = .TRUE.
 1100 CONTINUE
      RETURN
      END
```

```
C
      SUBROUTINE DETECT (S, ST, NR, NR1, IAREA1, THRESH, NRECS, NSAMP)
C
C  DETECT CIRCULAR REGIONS BY CONTRAST COMPARISON
C
      INTEGER S(NR1,NSAMP), ST(1), SAMP(5,20), SIZE(5,20), DELH, DELV,
     .HSN(25,10), HSD(25,10), VSN(25,10), VSD(25,10)
      DIMENSION NSET(4), NTEST(4), NSEQ(5), NMAT(5), DIFF(5,20), NSIZE
     .(5), NS(10), ISEQ(625), KAREA(11), ASIZE(5,2), MSEQ(500)
      DATA NTEST /2*1, 2*-1/, NSET /0, 0, 2, 1/
      COMMON /SEQNCE/ NS,HSN,VSN,HSD,VSD,ISEQ,KAREA,ASIZE,MS,MSEQ
C
  100 FORMAT (1H1,20X,'COORDINATES OF MATCH POSITIONS'/21X,30('*')//5X,
     .'LINE NUMBER',10X,'SAMPLE NUMBER, RELATIVE AREA, CONTRAST'//)
  101 FORMAT (I10,3X,8(I6,I3,F6.2))
  102 FORMAT (13X,8(I6,I3,F6.2))
  103 FORMAT (1H1,30X,'COUNT OF MATCH POSITIONS'/31X,24('*')//21X,'NUMBE
     .R   NOMINAL AREA + DIAMETER   ACTUAL PIXEL AREA + DIAMETER'/21X,
     .6('*'),3X,12('*'),3X,8('*'),3X,17('*'),3X,8('*'))
  104 FORMAT (/I26,I12,F13.1,2I15)
C
      PRINT 100
      MIDROW = NR/2 + 1
      MID = (NR+4)*(MIDROW-1) + MIDROW + 2
      NREC2 = NRECS - NR1 + MIDROW - 1
      NSAMP2 = NSAMP - MIDROW
C
C  INITIALIZE SIZE COUNTER, MATCHES PER RECORD COUNTER, AND ARRAY S
C
      DO 2 I=1,5
      NSIZE(I) = 0
    2 NMAT(I) = 0
      DO 5 JW=1,NR1
    5 READ (10) (S(JW,IEL), IEL=1,NSAMP)
C
      DO 1 NREC=MIDROW,NREC2
C
      IEL = MIDROW
      JS = 0
      NSEQ(1) = 0
      NSEQ(5) = 0
C
 5330 CONTINUE
      IEL = IEL + 1
      IF (IEL.GT.NSAMP2) GO TO 1111
      JS = JS + NR1
C
      IF (ST(JS+MID).GE.0) GO TO 80
      NSEQ(1) = 0
      NSEQ(5) = 0
      GO TO 5330
C
C  CANCEL SEARCH IF INSUFFICIENT PIXELS AHEAD TO SATISFY DIFFERENCE SEQ.
C
   80 NPIX = 3 - MAXO (NSEQ(1), NSEQ(5))
      JS1 = JS + NPIX*NR1
      IF (ST(JS1+MID).GE.0) GO TO 89
```

```
C
C  SEARCH FOR MAXIMUM IN SUM OVER CIRCULAR REGION BY SEARCHING FOR
C  DIFFERENCE SEQUENCE WITH SIGNS + + - -
C
   88 DO 40 N=1,5,4
      N3 = NS(N)
      DELH = 0
      DO 45 JSEQ=1,N3
   45 DELH = DELH + IABS(ST(JS+HSN(JSEQ,N))) - IABS(ST(JS+HSD(JSEQ,N)))
      NSEQ(N) = NSEQ(N) + 1
      IF (DELH*NTEST(NSEQ(N)).LT.0) GO TO 41
      IF (NSEQ(N).EQ.4) GO TO 1250
      GO TO 40
   41 NSEQ(N) = NSET(NSEQ(N))
   40 CONTINUE
      GO TO 5330
C
C  COMPUTE DIFFERENCE SEQUENCE ALONG COLUMNS
C
 1250 NSEQ(1) = 0
      NSEQ(5) = 0
      DO 2100 N=1,5,4
      JS1 = JS - 2*NR1 - 1
      N3 = NS(N)
      DO 2010 K=1,4
      DELV = 0
      DO 35 JSEQ=1,N3
   35 DELV = DELV + IABS(ST(JS1+VSN(JSEQ,N)))-IABS(ST(JS1+VSD(JSEQ,N)))
      IF (DELV*NTEST(K).LT.0) GO TO 2100
 2010 JS1 = JS1 + 1
      CALL MATCH (ST, NR1, NMAT, SAMP, SIZE, DIFF, THRESH, IEL, JS)
      GO TO 5330
 2100 CONTINUE
      GO TO 5330
 1111 CONTINUE
C
C  ROTATE ARRAY S BY ONE ROW
C
      DO 6 JW=2,NR1
      DO 6 IEL=1,NSAMP
    6 S(JW-1,IEL) = S(JW,IEL)
C
C  OUTPUT LISTS OF MATCH COORDINATES.  READ NEXT DATA RECORD.
C
      IF (NMAT(1).EQ.0) GO TO 2000
      NM = NMAT(1)
      NMATCH = NMATCH + NM
      DO 2121 I=1,NM
 2121 NSIZE(SIZE(1,I)) = NSIZE(SIZE(1,I)) + 1
      IF (NM.GT.8) GO TO 800
      PRINT 101, NREC, (SAMP(1,I), SIZE(1,I), DIFF(1,I), I=1,NM)
      GO TO 85
  800 PRINT 101, NREC, (SAMP(1,I), SIZE(1,I), DIFF(1,I), I=1,8)
      PRINT 102, (SAMP(1,I), SIZE(1,I), DIFF(1,I), I=9,NM)
   85 WRITE (11) NREC, NM, (SAMP(1,I), SIZE(1,I), I=1,NM)
 2000 CONTINUE
      READ (10) (S(NR1,IEL), IEL=1,NSAMP)
C
```

```
      IF (NMAT(I).EQ.0) GO TO 120
      NM = NMAT(I)
      DO 110 IEL=1,NM
      SAMP(I-1,IEL) = SAMP(I,IEL)
      SIZE(I-1,IEL) = SIZE(I,IEL)
  110 DIFF(I-1,IEL) = DIFF(I,IEL)
  120 NMAT(I-1) = NMAT(I)
      NMAT(5) = 0
    1 CONTINUE
C
      PRINT 103
      NMATCH = 0
      DO 10 N=1,5
      NAREA = (N+1)*IAREA1
      D = 2.0 * SQRT(NAREA/3.141592o5)
      ISIZE = ASIZE(N,1)
      ID = 2*INT(D/2.0+1.0) - 1
      PRINT 104, NSIZE(N), NAREA, D, ISIZE, ID
   10 NMATCH = NMATCH + NSIZE(N)
      PRINT 104, NMATCH
      RETURN
      END
```

```
C
      SUBROUTINE MATCH (ST,NR1,NMAT,SAMP,SIZE,DIFF,THRESH,IEL,JS)
C
C  COMPUTE CONTRAST FOR 5 SIZES OF DISK OVER A 5 X 5 PIXEL AREA
C
      INTEGER ST(1), SAMP(5,1), SIZE(5,1), SUM(11), DISK, BORD, DSAVE,
     .BSAVE, DDISK, HSN(25,10), VSN(25,10), HSD(25,10), VSD(25,10)
      DIMENSION NMAT(1), DIFF(5,1), NS(10), ISEQ(625), KAREA(11), ASIZE
     .(5,2), MSEQ(500)
      COMMON /SEQNCE/ NS,HSN,VSN,HSD,VSD,ISEQ,KAREA,ASIZE,MS,MSEQ
C
C  COMPUTE SUMS OF DATA VALUES OVER ANNULAR RINGS
C
      JS1 = JS - 4*NR1 - 2
      KEL = KAREA(11)
      K = 1
      SUM(1) = 0
      DO 305 JSEQ=1,KEL
      IF (JSEQ.NE.KAREA(K)+1) GO TO 305
      K = K + 1
      SUM(K) = 0
  305 SUM(K) = SUM(K) + IABS(ST(JS1+ISEQ(JSEQ)))
C
C  LOOP OVER 5 SIZES OF DISK AND 5 X 5 PIXEL AREA
C
      AO = 0.0
      DO 18 NSIZE=1,5
      JS = JS - 5*NR1
      ND = NS(NSIZE)
      NB = NS(NSIZE+5)
C
      DO 18 KHOR=1,5
      JS = JS + NR1
      DO 18 KVER=1,5
      JS1 = JS + KVER - 3
      IF (KVER.NE.1) GO TO 200
      IF (KHOR.NE.1) GO TO 201
C
C  COMPUTE DISK AND BORDER SUMS AT FIRST POSITION AND SAVE FOR RECURSIVE
C  CALCULATIONS
C
      DISK = 0
      BORD = 0
      DO 21 N=1,NSIZE
      DISK = DISK + SUM(N)
   21 BORD = BORD + SUM(NSIZE+N)
      BORD = BORD + SUM(2*NSIZE+1)
      DSAVE = DISK
      BSAVE = BORD
      GO TO 9210
C
C  COMPUTE RECURSIVE SUM ALONG ROWS
C
  201 DDISK = 0
      DO 34 JSEQ=1,ND
   34 DDISK = DDISK + IABS(ST(JS1+HSN(JSEQ,NSIZE))) - IABS(ST(JS1+HSD
     .(JSEQ,NSIZE)))
```

```
   35 BORD = BORD + IABS(ST(JS1+HSN(JSEQ,NSIZE+5))) - IABS(ST(JS1+HSD
     .(JSEQ,NSIZE+5)))
C
C  SUBTRACT SUM OF TERMS ADDED TO PREVIOUS DISK SUM
C
      BORD = BORD - DDISK
      DSAVE = DISK
      BSAVE = BORD
      GO TO 9210
C
C  COMPUTE RECURSIVE SUM ALONG COLUMNS
C
  200 DDISK = 0
      DO 32 JSEQ=1,ND
   32 DDISK = DDISK + IABS(ST(JS1+VSN(JSEQ,NSIZE))) - IABS(ST(JS1+VSD
     .(JSEQ,NSIZE)))
      DISK = DISK + DDISK
      DO 33 JSEQ=1,NB
   33 BORD = BORD + IABS(ST(JS1+VSN(JSEQ,NSIZE+5))) - IABS(ST(JS1+VSD
     .(JSEQ,NSIZE+5)))
      BORD = BORD - DDISK
C
C  FIND MAXIMUM OF DIFFERENCE BETWEEN AVERAGE DISK TERM AND AVERAGE
C  BORDER TERM
C
 9210 CONTINUE
      DIF = DISK/ASIZE(NSIZE,1) - BURD/ASIZE(NSIZE,2)
      IF (DIF.LT.A0) GO TO 18
      A0 = DIF
      K1 = KVER
      K2 = KHOR
      K3 = JS1
      K4 = NSIZE
   18 CONTINUE
C
C  STORE MATCH PARAMETERS IF CONTRAST IS GREATER THAN THRESHOLD
C
      IF (A0.LT.THRESH) RETURN
      NMAT(K1) = NMAT(K1) + 1
      NM = NMAT(K1)
      SAMP(K1,NM) = IEL - 5 + K2
      SIZE(K1,NM) = K4
      DIFF(K1,NM) = A0
C
C  IF MATCH FOUND, SET CENTRAL REGION TO NEGATIVE DATA VALUES FOR
C  SKIPPING ALGORITHM
C
      DO 10 JSEQ=1,MS
      JJ = K3 + MSEW(JSEQ)
   10 ST(JJ) = ISIGN(ST(JJ),-1)
      RETURN
      END
```

```
C
      SUBROUTINE SORTSL (A1, A2, II, JJ)
C
C  SORT ARRAY A1 AND ARRANGE ARRAY A2 CORRESPONDINGLY
C  DO FROM ELEMENTS II TO JJ
C
      DIMENSION A1(1), A2(1), IU(16), IL(16)
      INTEGER A1, A2, T1, T2, TT1, TT2
      LOGICAL*1 SL
      SL = .TRUE.
      GO TO 1
C
      ENTRY SORTLS (A1, A2, II, JJ)
C
      SL = .FALSE.
    1 ND = JJ - II + 1
      M=1
      I=II
      J=JJ
    5 IF(I.GE.J) GOTO 70
   10 K=I
      IJ=(J+I)/2
      T1 = A1(IJ)
      T2 = A2(IJ)
      IF (A1(I).LE.T1) GO TO 20
      A1(IJ) = A1(I)
      A2(IJ) = A2(I)
      A1(I) = T1
      A2(I) = T2
      T1 = A1(IJ)
      T2 = A2(IJ)
   20 L=J
      IF (A1(J).GE.T1) GO TO 40
      A1(IJ) = A1(J)
      A2(IJ) = A2(J)
      A1(J) = T1
      A2(J) = T2
      T1 = A1(IJ)
      T2 = A2(IJ)
      IF (A1(I).LE.T1) GO TO 40
      A1(IJ) = A1(I)
      A2(IJ) = A2(I)
      A1(I) = T1
      A2(I) = T2
      T1 = A1(IJ)
      T2 = A2(IJ)
      GOTO 40
   30 A1(L) = A1(K)
      A2(L) = A2(K)
      A1(K) = TT1
      A2(K) = TT2
   40 L=L-1
      IF (A1(L).GT.T1) GO TO 40
      TT1 = A1(L)
      TT2 = A2(L)
   50 K=K+1
      IF (A1(K).LT.T1) GO TO 50
```

```
      IU(M)=L
      I=K
      M=M+1
      GOTO 80
   60 IL(M) = K
      IU(M)=J
      J=L
      M=M+1
      GOTO 80
   70 M=M-1
      IF (M.NE.0) GO TO 75
      IF (SL) RETURN
      ND2 = ND/2
      I1 = II
      I2 = JJ
      DO 110 I=1,ND2
      T1 = A1(I1)
      T2 = A2(I1)
      A1(I1) = A1(I2)
      A2(I1) = A2(I2)
      A1(I2) = T1
      A2(I2) = T2
      I1 = I1 + 1
  110 I2 = I2 - 1
      RETURN
   75 I=IL(M)
      J=IU(M)
   80 IF(J-I.GE.II) GOTO 10
      IF(I.EQ.II) GOTO 5
      I=I-1
   90 I=I+1
      IF(I.EQ.J) GOTO 70
      T1 = A1(I+1)
      T2 = A2(I+1)
      IF (A1(II).LE.T1) GO TO 90
      K=I
  100 A1(K+1) = A1(K)
      A2(K+1) = A2(K)
      K=K-1
      IF (T1.LT.A1(K)) GO TO 100
      A1(K+1) = T1
      A2(K+1) = T2
      GOTO 90
```

APPENDIX B

Listings of the following data handling subroutines are given:

SETREE	-	selects training data from the feature vector file
CELLS	-	determines the frequencies and feature vectors in the four-dimensional histogram analysis
CLTREE	-	classifies trees according to the majority of pixels
RETREE	-	reconstructs the tree templates at the match points

```
C
      SUBROUTINE SETREE (X, S, NR, NRECS, NSAMP, NN, MM)
C
C  SELECT GROUND TRUTH TREES
C
      DIMENSION X(NN,1), S(NN,NSAMP), NTOT(2), NT(2)
      INTEGER SAMP, SIZE, S11
      LOGICAL TMPLT(25,25,5)
  100 FORMAT (10I5)
  101 FORMAT (I20,' OCCUPIED CELLS',I10,' TOTAL')
  102 FORMAT (1X,20I6)
  106 FORMAT (5I6)
C
      DO 20 N=1,5
      R2 = 50.0 * (N+1) / 3.14159265
      IAREA = 0
      DO 10 I=1,NR
      AI = I - (NR+1)/2
      DO 10 J=1,NR
      AJ = J - (NR+1)/2
      TMPLT(I,J,N) = .FALSE.
      IF (AI**2+AJ**2.GT.R2) GO TO 10
      IAREA = IAREA + 1
      TMPLT(I,J,N) = .TRUE.
   10 CONTINUE
   20 PRINT 102, IAREA
C
      DO 150 NL=1,NRECS
      READ (10) S
  150 WRITE (90'NL) S
C
      DO 260 NC=1,2
      NTOT(NC) = 0
      NT(NC) = 0
      KS = 0
      JC(1) = 0
C
C  EXTRACT DATA FOR A TREE
C
  250 READ (5,100,END=220) LINE, SAMP, SIZE
      NT(NC) = NT(NC) + 1
C
      L11 = LINE - NR/2
      N1 = 0
      DO 40 NL1=1,NR
      READ (90'L11) S
      S11 = SAMP - NR/2
      DO 44 NS1=1,NR
      IF (.NOT.TMPLT(NL1,NS1,SIZE)) GO TO 44
      N1 = N1 + 1
      DO 60 NF1=1,4
   60 X(NF1,N1) = S(NF1,S11)
   44 S11 = S11 + 1
   40 L11 = L11 + 1
      NTOT(NC) = NTOT(NC) + N1
C
      WRITE (6,106) NT(NC), NC, LINE, SAMP, SIZE
C
C  DETERMINE CELL NUMBERS
C
```

```
      KSO = KS
      CALL CELLS (X, N1, MC, JC, KS, NN)
      KSO = KS - KSO
      PRINT 101, KSO, KS
      GO TO 250
  220 CONTINUE
      CALL SORTLS (MC, JC, 1, KS)
      WRITE (8) KS, (MC(I), JC(I), I=1,KS)
  260 CONTINUE
      PRINT 102, NTOT
      RETURN
      END
```

```
C
      SUBROUTINE CELLS (X, NS, MC, JC, KS, NN)
C
      DIMENSION X(NN,1), MC(1), JC(1)
      LOGICAL*1 BYTE(4)
      EQUIVALENCE (NCO, BYTE(1))
C
      DO 37 NS1=1,NS
      DO 140 NF=1,NN
  140 BYTE(NF) = X(NF,NS1)
C
C  CHECK FOR CELL OCCUPANCY
C
      DO 36 K=1,KS
      IF (NCO.EQ.JC(K)) GO TO 37
   36 CONTINUE
C
C  START NEW CELL
C
      KS = KS + 1
      JC(KS) = NCO
      MC(KS) = 0
      K = KS
C
   37 MC(K) = MC(K) + 1
      RETURN
      END
```

```
C
      SUBROUTINE CLTREE (S, ST, NR, IAREA1, MODE, NRECS, NSAMP)
C
C  CLASSIFY DETECTED TREES ACCORDING TO THE MAJORITY OF PIXELS
C      IF MODE=1, READ COORDINATES FROM TAPE UNIT 8
C      IF MODE=2, READ COORDINATES FROM CARDS
C
      DIMENSION ISEQ(5,300), NCLASS(3), MCLASS(50), PCT(50), KS(5,3)
      INTEGER ST(1), S(NR,1), PIXL(5), SAMP(50), SIZE(50), CLASS(3)
      DOUBLE PRECISION CLASS1(3)
      DATA CLASS /'DC', 'OK', 'SD'/, CLASS1 /' DECLINE', ' HEALTHY',
     .'    SOIL'/
  100 FORMAT ('1',20X,'TREE CLASSIFICATION'/21X,19('*')//5X,'LINE NUMBER
     .',5X,'SAMPLE, SIZE, CLASS, PERCENT'//)
  101 FORMAT (3I5)
  102 FORMAT ('1',30X,'CLASSIFICATION SUMMARY'/31X,22('*')//20X,'CLASS',
     .8X,'SIZE NUMBER',8X,'SAMPLES',10X,'PERCENT'/20X,5('*'),8X,11('*'),
     .8X,7('*'),10X,7('*'))
  103 FORMAT (/A26,I13,I17,F18.2)
  104 FORMAT (/41X,'TOTAL',I10/41X,15('*'))
  105 FORMAT (I12,7X,6(I6,I3,A6,F4.1))
  106 FORMAT (19X,6(I6,I3,A6,F4.1))
  107 FORMAT (52X,5('-'),11X,7('-')/I56,F18.2/52X,5('-'),11X,7('-')//)
C
C  DETERMINE FIVE TEMPLATE SEQUENCES
C
      DO 310 NT=1,5
      R2 = (NT+1)*IAREA1 / 3.14159265
      NSEQ = 0
      DO 20 I=1,NR
      II = I - (NR+1)/2
      DO 20 J=1,NR
      JJ = J - (NR+1)/2
      IF (II**2+JJ**2.GT.R2) GO TO 20
      NSEQ = NSEQ + 1
      ISEQ(NT,NSEQ) = NR*(J-1) + I
   20 CONTINUE
  310 PIXL(NT) = NSEQ
C
C  INITIALIZE ARRAYS, READ INITIAL COORDINATES
C
      PRINT 100
      DO 10 L=1,5
      DO 10 N=1,3
   10 KS(L,N) = 0
      DO 22 JW=1,NR
   22 READ (10) (S(JW,IEL), IEL=1,NSAMP)
      NT = 1
      IF (MODE.EQ.1) READ (8) NREC, NT, (SAMP(I), SIZE(I), I=1,NT)
      IF (MODE.EQ.2) READ (5,101) NREC, SAMP(1), SIZE(1)
      MIDROW = NR/2 + 1
      NREC2 = NRECS - NR + MIDROW - 1
C
      DO 130 NREC=MIDROW,NREC2
      IF (NREC.NE.NREC) GO TO 95
   50 DO 75 MT=1,NT
C
C  COUNT NUMBER OF PIXELS ASSIGNED TO EACH CLASS
C
      DO 74 N=1,3
```

```
   74 NCLASS(N) = 0
      IAREA = PIXL(SIZE(MT))
      JS = NR*(SAMP(MT)-1-NR/2)
      DO 76 NSEQ=1,IAREA
      JJ = JS + ISEQ(SIZE(MT),NSEQ)
      IC = ST(JJ)
   76 NCLASS(IC) = NCLASS(IC) + 1
C
C  DETERMINE CLASS HAVING MAJORITY OF PIXELS
C
      MAX = 0
      DO 70 N=1,3
      IF (NCLASS(N).LT.MAX) GO TO 70
      MAX = NCLASS(N)
      NC = N
   70 CONTINUE
C
      KS(SIZE(MT),NC) = KS(SIZE(MT),NC) + 1
      MCLASS(MT) = NC
      PCT(MT) = MAX * 100.0 / IAREA
   75 CONTINUE
C
      IF (MODE.EQ.1) WRITE (11) MREC, NT, (SAMP(I), SIZE(I), MCLASS(I),
     .I=1,NT)
      IF (NT.GT.6) GO TO 80
      PRINT 105, NREC, (SAMP(I),SIZE(I),CLASS(MCLASS(I)),PCT(I), I=1,NT)
      GO TO 85
   80 PRINT 105, NREC, (SAMP(I),SIZE(I),CLASS(MCLASS(I)),PCT(I), I=1,6)
      PRINT 106, (SAMP(I),SIZE(I),CLASS(MCLASS(I)),PCT(I), I=7,NT)
C
   85 NT = 1
      IF (MODE.EQ.1) READ (8,END=30) MREC, NT, (SAMP(I),SIZE(I), I=1,NT)
      IF (MODE.EQ.2) READ (5,101,END=30) MREC, SAMP(1), SIZE(1)
      IF (MREC.EQ.NREC) GO TO 50
C
   95 DO 6 JW=2,NR
      DO 6 IEL=1,NSAMP
    6 S(JW-1,IEL) = S(JW,IEL)
      READ (10) (S(NR,IEL), IEL=1,NSAMP)
  130 CONTINUE
C
   30 NTOT = 0
      DO 210 NC=1,3
      NCLASS(NC) = 0
      DO 200 N=1,5
  200 NCLASS(NC) = NCLASS(NC) + KS(N,NC)
  210 NTOT = NTOT + NCLASS(NC)
C
      PRINT 102
      DO 1100 NC=1,3
      DO 1110 N=1,5
      PCT(NC) = 100.0 * KS(N,NC) / NTOT
 1110 PRINT 103, CLASS1(NC), N, KS(N,NC), PCT(NC)
      PCT(NC) = 100.0 * NCLASS(NC) / NTOT
 1100 PRINT 107, NCLASS(NC), PCT(NC)
      PRINT 104, NTOT
      RETURN
      END
```

```
C
      SUBROUTINE RETREE (S, ST, NR, IAREA1, NRECS, NSAMP)
C
C  RECONSTRUCT THE TREE TEMPLATE AT THE MATCH POINTS
C
      INTEGER S(NR,1), ST(1), ISEQ(5,300), PIXL(5), SAMP(50), SIZE(50),
     .CLASS(50)
      LOGICAL MATCHS
C
C  DETERMINE FIVE TEMPLATE SEQUENCES
C
      DO 2 NT=1,5
      R2 = (NT+1)*IAREA1 / 3.14159265
      NSEQ = 0
      DO 1 I=1,NR
      AI = I - (NR+1)/2
      DO 1 J=1,NR
      AJ = J - (NR+1)/2
      IF (AI**2+AJ**2.GT.R2) GO TO 1
      NSEQ = NSEQ + 1
      ISEQ(NT,NSEQ) = NR*(J-1) + I
    1 CONTINUE
    2 PIXL(NT) = NSEQ
C
C  READ INITIAL COORDINATES, SET BACKGROUND TO CLASS '3'
C
      READ (8) MREC, NT, (SAMP(I), SIZE(I), CLASS(I), I=1,NT)
      MATCHS = .TRUE.
      MIDROW = NR/2 + 1
      NREC2 = NRECS - NR + MIDROW - 1
      DO 7 J=1,NR
      DO 7 IEL=1,NSAMP
    7 S(J,IEL) = 3
C
C  WRITE OUTPUT FILE OF RECONSTRUCTED TREES
C
      DO 10 NREC=MIDROW,NREC2
      IF (NREC.NE.MREC) GO TO 95
      DO 180 MT=1,NT
      IAREA = PIXL(SIZE(MT))
      JS = NR*(SAMP(MT)-1-NR/2)
      DO 75 NSEQ=1,IAREA
      JJ = JS + ISEQ(SIZE(MT),NSEQ)
   75 ST(JJ) = CLASS(MT)
  180 CONTINUE
C
      IF (MATCHS) READ (8,END=30) MREC, NT, (SAMP(I), SIZE(I), CLASS(I),
     .I=1,NT)
      GO TO 95
   30 MATCHS = .FALSE.
C
   95 DO 6 JW=2,NR
      DO 6 IEL=1,NSAMP
    6 S(JW-1,IEL) = S(JW,IEL)
      DO 8 IEL=1,NSAMP
    8 S(NR,IEL) = 3
   10 WRITE (11) (S(1,IEL), IEL=1,NSAMP)
C
      DO 200 JW=1,NR
  200 WRITE (11) (S(JW,IEL), IEL=1,NSAMP)
      RETURN
      END
```

APPENDIX C

The following subroutines were used in performing the supervised classifications:

SUBLPC	-	Compute Gaussian statistics of training samples
PTEST	-	Test maximum likelihood classification of training samples
PCLASS	-	Handle I/O for classification of the data
MALICA	-	Maximum likelihood classifier
EFFECT	-	Determine class to be tested for linear classifier using training samples
SNOPAL	-	Compute discriminant function coefficients for one class
NTEST	-	Test linear classification of training samples
NCLASS	-	Handle I/O for classification of the data
NOPACA	-	Nonparametric (linear) classifier

```
C
      SUBROUTINE SUBLPC (LC, MC, NS, CLASS, S, EM, EK, B, NN, MM)
C
C  SUPERVISED BATCH LEARNING OF PARAMETERS
C
      DIMENSION MC(1), NS(MM,3), EM(NN,MM), EK(NN,NN,MM), B(MM)
      DOUBLE PRECISION CLASS(MM), S(NN,NN), DET
      LOGICAL*1 LC(NN,1)
 4550 FORMAT ('1'/20X,'ESTIMATED GAUSSIAN PARAMETERS'/20X,29('*')//5X,
     .'MEAN VECTORS',10X,'COVARIANCE MATRICES')
 4554 FORMAT (F15.2,(5X,16F7.2))
 4557 FORMAT (//20X,'DETERMINANT =',1PE10.3)
 4577 FORMAT (//I20,A10,I6,' SAMPLES')
C
C  COMPUTE MEAN VECTORS AND COVARIANCE MATRICES
C
      WRITE (6,4550)
      DO 1110 I3=1,MM
      NC11 = NS(I3,2)
      NC12 = NS(I3,3)
      DO 9000 I1=1,NN
      EM(I1,I3) = 0.0
      DO 4000 NS1=NC11,NC12
 4000 EM(I1,I3) = EM(I1,I3) + MC(NS1)*LC(I1,NS1)
 9000 EM(I1,I3) = EM(I1,I3) / NS(I3,1)
C
      DO 9200 I1=1,NN
      DO 9200 I2=1,I1
      EK(I1,I2,I3) = 0.0
      DO 9100 NS1=NC11,NC12
 9100 EK(I1,I2,I3) = EK(I1,I2,I3) + MC(NS1) * (LC(I1,NS1)-EM(I1,I3))
     . * (LC(I2,NS1)-EM(I2,I3))
      EK(I1,I2,I3) = EK(I1,I2,I3) / (NS(I3,1)-1)
      S(I1,I2) = EK(I1,I2,I3)
      S(I2,I1) = S(I1,I2)
 9200 CONTINUE
C
      WRITE (6,4577) I3, CLASS(I3), NS(I3,1)
      DO 5008 I1=1,NN
 5008 PRINT 4554, EM(I1,I3), (EK(I1,I2,I3), I2=1,I1)
C
C  INVERT COVARIANCE MATRICES, COMPUTE GAUSSIAN FUNCTION CONSTANT TERMS
C
      CALL GASINV (S, NN, DET)
      DO 1000 I1=1,NN
      DO 1000 I2=1,NN
 1000 EK(I1,I2,I3) = S(I1,I2)
      B(I3) = -0.5 * (NN*ALOG(2.0*3.14159265) + ALOG(SNGL(DET)))
 1110 WRITE (6,4557) DET
      RETURN
      END
```

```
C
      SUBROUTINE PTEST (LC, MC, NS, CLASS, EM, EK, E, NN, MM)
C
C  CLASSIFIES KNOWN DATA SAMPLES - PARAMETRIC CLASSIFICATION
C
      DIMENSION MC(1), NS(MM,3), X(20), EM(1), EK(1), E(1), KS(20)
      DOUBLE PRECISION CLASS(MM)
      LOGICAL*1 LC(NN,1)
  104 FORMAT (//30X,16HAVERAGE ACCURACY =,F6.1,8H PERCENT/30X,32(1H*))
 2008 FORMAT ('1'/30X,28('*')/30X,'* RESULTS OF CLASSIFICATION *'/30X,'*
     .     TRAINING SAMPLES      *'/30X,29('*')///14X,'NUMBER OF  NUMBER
     .     PERCENT     NUMBER OF SAMPLES CLASSIFIED AS'/6X,'CLASS    SAM
     .PLES   CORRECT    CORRECT',10A9/(43X,10A9))
 2009 FORMAT (/I4,A9,I7,I10,F12.1,10I9,(/42X,10I9))
C
      PRINT 2008, CLASS
      TE = 0.0
      DO 1301 NC = 1,MM
      DO 1221 NW=1,MM
 1221 KS(NW) = 0
      NSC = NS(NC,1)
      NC11 = NS(NC,2)
      NC12 = NS(NC,3)
C
      DO 1400 NS1=NC11,NC12
      DO 1500 NF=1,NN
 1500 X(NF) = LC(NF,NS1)
      CALL MALICA (X, ICLASS, KS, EM, EK, E, 1, NN, MM)
 1400 KS(ICLASS) = KS(ICLASS) + MC(NS1) - 1
C
      EFF = 100.0 * KS(NC) / NSC
      PRINT 2009, NC, CLASS(NC), NSC, KS(NC), EFF, (KS(NC1), NC1=1,MM)
 1301 TE = TE + EFF
      AVE = TE / FLOAT(MM)
      PRINT 104, AVE
      RETURN
      END
```

```
C
      SUBROUTINE PCLASS (S,MCLASS,CLASS,EM,EK,B,NRECS,NSAMP,NN,MM)
C
C  CLASSIFIES UNKNOWN DATA SAMPLES - PARAMETRIC CLASSIFICATION
C
      DIMENSION S(NN,NSAMP), MCLASS(NSAMP), EM(1), EK(1), B(1), KS(20)
      DOUBLE PRECISION CLASS(MM)
 2010 FORMAT ('1'/30X,29('*')/30X,'* RESULTS OF CLASSIFICATION *'/30X,
     .'*',7X,'DATA SAMPLES',9X,'*'/30X,29('*')///20X,'CLASS',15X,'SAMPLE
     .S',15X,'PERCENT'/20X,5('*'),2(15X,7('*')))
 2011 FORMAT (/I18,A9,I20,F21.2)
 2012 FORMAT (/14X,13HTOTAL SAMPLES,I20/14X,13(1H*))
C
      DO 1415 NC=1,MM
 1415 KS(NC) = 0
C
      DO 12 NREC=1,NRECS
      CALL MALICA (S, MCLASS, KS, EM, EK, B, NSAMP, NN, MM)
   12 WRITE (11) MCLASS
C
      NTOT = NSAMP * NRECS
      PRINT 2010
      DO 1301 NC = 1,MM
      PCT = 100.0 * KS(NC) / NTOT
 1301 PRINT 2011, NC, CLASS(NC), KS(NC), PCT
      PRINT 2012, NTOT
      RETURN
      END
```

```
C
      SUBROUTINE MALICA (X1, KMAX, KS, EM, EK, B, NSS, NN, MM)
C
C          MAXIMUM LIKELIHOOD CLASSIFICATION
C
      DIMENSION X1(NN,NSS), KMAX(NSS), EM(NN,MM), EK(NN,NN,MM), B(MM),
     .DX(20), KS(1)
      DATA GMAXX /ZFFFFFFFF/
C
      DO 2000 NS1=1,NSS
      GMAX = GMAXX
      DO 1900 I=1,MM
      G = B(I)
      DO 6300 II=1,NN
      DX(II) = X1(II,NS1) - EM(II,I)
      SUM = 0.0
      DO 6200 JJ=1,II
      A = DX(JJ) * EK(JJ,II,I)
 6200 SUM = SUM - A
 6300 G = G + (SUM + 0.5*A) * DX(II)
      IF (G.LT.GMAX) GO TO 1900
      KMAX(NS1) = I
      GMAX = G
 1900 CONTINUE
 2000 KS(KMAX(NS1)) = KS(KMAX(NS1)) + 1
      RETURN
      END
```

```
C
      SUBROUTINE EFFECT (LC, MC, NS, CLASS, MOC, DE, NW1, NN, MM)
C
C  EFFECTIVE FIGURE OF MERIT FEATURE SELECTION CRITERION
C
      DIMENSION MC(1), NS(MM,3), DE(MM,MM,NN), MOC(MM), CFM(20), FC(20)
      LOGICAL*1 LC(NN,1)
      DOUBLE PRECISION CLASS(MM)
  100 FORMAT ('1'/20X,'EFFECTIVE FIGURES OF MERIT'/20X,26('*')//)
  143 FORMAT (I5,A9,5X,16E7.4/(19X,16E7.4))
  150 FORMAT (/20X,'COMBINED FIGURES OF MERIT'/20X,25('*')/)
  151 FORMAT (I25,A9,F10.5,5X,1315)
C
C COMPUTES INTER-CLASS AND INTRA-CLASS DISTANCES
C
      PRINT 100
      IF (NW1.NE.1) GO TO 2000
      DO 1005 NF1=1,NN
      DO 2 I1=1,MM
      NC11 = NS(I1,2)
      NC12 = NS(I1,3)
      DO 2 I2=1,I1
      NC21 = NS(I2,2)
      NC22 = NS(I2,3)
      DE(I1,I2,NF1) = 0.0
      DO 3 LK1 = NC11,NC12
      IF (I1.EQ.I2) NC22 = LK1 - 1
      DO 3 LK2 = NC21,NC22
    3 DE(I1,I2,NF1) = DE(I1,I2,NF1) + IABS(LC(NF1,LK1)-LC(NF1,LK2))
     .*MC(LK1)*MC(LK2)
    2 DE(I2,I1,NF1) = DE(I1,I2,NF1)
 1005 CONTINUE
      DO 1109 NC=1,MM
 1109 MOC(NC) = NC
C
C COMPUTES THE NORMALIZED FIGURE OF MERIT OF ALL REMAINING PATTERN
C CLASSES ALONG EACH OF THE FEATURE DIRECTIONS
C
 2000 DO 1000 J3=NW1,MM
      CFM(J3) = 1.0
      I3 = MOC(J3)
      AI3 = NS(I3,1)
      DO 3000 I=1,NN
      FCMIN = 1.0 E 50
      NST = 0
      SUM1 = 0.0
      SUM2 = 0.0
C
C  COMPUTE SUM1 - TOTAL OF INTERCLASS DISTANCES FROM CLASS I3 TO ALL
C  REMAINING CLASSES
C
      DO 6 J4=NW1,MM
      IF (J4.EQ.J3) GO TO 6
      I4 = MOC(J4)
      AI4 = NS(I4,1)
      NST = NST + NS(I4,1)
      SUM1 = SUM1 + DE(I3,I4,I)
C
C  COMPUTE SUM2 - TOTAL OF DISTANCES AMONG ALL REMAINING CLASSES,
C  EQUIVALENT TO INTRACLASS DISTANCE OF ALL REMAINING CLASSES CONSIDERED
```

```
C  AS ONE CLASS
C
      DO 7 J5=NW1,J4
      IF (J5.EQ.J3) GO TO 7
      I5 = MOC(J5)
      SUM2 = SUM2 + DE(I4,I5,I)
    7 CONTINUE
C
C  COMPUTE MINIMUM FIGURE OF MERIT FOR INDIVIDUAL CLASSES I3 AND I4
C
      S1 = DE(I3,I4,I) / (AI3 * AI4)
      S2 = DE(I3,I3,I) / (AI3*(AI3-1.0)) + DE(I4,I4,I) / (AI4*(AI4-1.0))
      F = S1 / S2
      IF (F.LT.FCMIN) FCMIN = F
    6 CONTINUE
C
      ANST = NST
      SUM1 = SUM1 / (AI3*ANST)
      SUM2 = DE(I3,I3,I) / (AI3*(AI3-1.0)) + SUM2 / (ANST*(ANST-1.0))
      FC(I) = FCMIN * SUM1 / SUM2
      FC(I) = EXP(-1.0/FC(I))
C
C  COMPUTE CFM, COMBINED FIGURE OF MERIT, AND ORDER BY CFM TO DETERMINE
C  THE MOST SEPARABLE CLASS
C
 3000 CFM(J3) = CFM(J3) * FC(I)
      CFM(J3) = CFM(J3) ** (1.0/NN)
      WRITE (6,143) I3, CLASS(I3), (FC(NF), NF=1,NN)
 1000 CONTINUE
      CALL SORTLS (CFM, MOC, NW1, MM)
C
      WRITE (6,150)
      DO 1251 NC1=NW1,MM
      NC = MOC(NC1)
 1251 WRITE (6,151) NC, CLASS(NC), CFM(NC1)
      CALL SORTSL (MOC, MOC, NW1+1, MM)
      RETURN
      END
```

```
C
      SUBROUTINE SNOPAL (LC,MC,NS,CLASS,W,MOC,S,Y,B,NW1,NN,MM,NN1,MM1)
C
C  SUPERVISED NON-PARAMETRIC LEARNING
C
      DIMENSION MC(1), NS(MM,3), W(MM1,NN1), MOC(MM), Y(1), B(1)
      LOGICAL*1 LC(NN,1)
      DOUBLE PRECISION CLASS(MM), S(NN1,NN1), DET
      LOGICAL TEST
      DATA NI /100/
   99 FORMAT (/1P6E15.4)
  100 FORMAT (I8,I11,I8,4X,1P7E14.3/(31X,1P7E14.3))
  101 FORMAT (/I13,A10,10X,1P7E14.3/(33X,1P7E14.3))
  102 FORMAT (//22X,22('*')/22X,'* CLASS',I3,A10,' *'/22X,22('*')//' ITE
     .RATION NO.',5X,'ERRORS',10X,'LINEAR DISCRIMINANT COEFFICIENTS'/
     .A22,'  OTHER'/)
  216 FORMAT ('1'/10X,'ORDERED CLASSES',20X,'ELEMENTS OF THE DISCRIMINAN
     .T VECTOR'/10X,15('*'),20X,35('*')/)
  220 FORMAT (/7X,'TOTAL ERRORS',I5)
C
C  INITIALIZE W, Y, AND B ARRAYS
C
      NW = MOC(NW1)
      NSW1 = NS(NW,2)
      NSW2 = NS(NW,3)
      NW2 = NW1 + 1
      DELTA = NN/400.0
      DO 1112 NFA=1,NN1
 1112 W(NW1,NFA) = 0.0
      J = 0
      NST2 = 0
      DO 1110 NC1=NW1,MM
      NC = MOC(NC1)
      NST2 = NST2 + NS(NC,1)
      NS11 = NS(NC,2)
      NS12 = NS(NC,3)
      DO 1110 NS1=NS11,NS12
      J = J + 1
      Y(J) =-1.0
 1110 B(J) = 1.0
C
C  COMPUTE INVERSE OF A(TRANSPOSE) A WHERE 'A' IS AUGMENTED MATRIX OF SAMPLES
C
      DO 130 I=1,NN
      DO 130 J=I,NN1
      S(I,J) = 0.0
      DO 131 NC1=NW1,MM
      NC = MOC(NC1)
      NS11 = NS(NC,2)
      NS12 = NS(NC,3)
      DO 131 NS1=NS11,NS12
      K1 = LC(I,NS1)
      IF (J.NE.NN1) K1 = K1*LC(J,NS1)
  131 S(I,J) = S(I,J) + MC(NS1)*K1
  130 S(J,I) = S(I,J)
      S(NN1,NN1) = NST2
      CALL GASINV (S, NN1, DET)
C
C  DO NI ITERATIONS OF THE HO-KASHYAP ALGORITHM, UNLESS ALL COEFFICIENTS
C  CHANGE BY LESS THAN DELTA = NN/4 PERCENT
```

```
C
      PRINT 102, NW, CLASS(NW), CLASS(NW)
      DO 1040 I,DEX=1,NI
      TEST = .TRUE.
C
      DO 1101 I=1,NN1
      W0 = W(NW1,I)
      J = 0
      DO 2101 NS1=NSW1,NSW2
      J = J + 1
      A2 = S(I,NN1)
      DO 140 K=1,NN
  140 A2 = A2 + S(I,K)*LC(K,NS1)
 2101 W(NW1,I) = W(NW1,I) + MC(NS1)*A2*ABS(Y(J))
      DO 2000 NC1=NW2,MM
      NC = MOC(NC1)
      NS11 = NS(NC,2)
      NS12 = NS(NC,3)
      DO 2000 NS1=NS11,NS12
      J = J + 1
      A2 = S(I,NN1)
      DO 141 K=1,NN
  141 A2 = A2 + S(I,K)*LC(K,NS1)
 2000 W(NW1,I) = W(NW1,I) - MC(NS1)*A2*ABS(Y(J))
      IF (ABS(W(NW1,I)-W0).GT.ABS(DELTA*W0)) TEST = .FALSE.
 1101 CONTINUE
C
C  COMPUTE NEW DISCRIMINANT VALUES AND CLASSIFICATION ERRORS
C
      NERR1 = 0
      I = 0
      DO 1004 NS1=NSW1,NSW2
      I = I + 1
      IF (Y(I).GT.0.0) B(I) = B(I) + 2.0*Y(I)
      Y(I) = W(NW1,NN1)
      DO 1141 NF2=1,NN
 1141 Y(I) = Y(I) + W(NW1,NF2)*LC(NF2,NS1)
      IF (Y(I).LE.0.0) NERR1 = NERR1 + MC(NS1)
 1004 Y(I) = Y(I) - B(I)
C
      NERR2 = 0
      DO 1005 NC1=NW2,MM
      NC = MOC(NC1)
      NS11 = NS(NC,2)
      NS12 = NS(NC,3)
      DO 1005 NS1=NS11,NS12
      I = I + 1
      IF (Y(I).GT.0.0) B(I) = B(I) + 2.0*Y(I)
      Y(I) = W(NW1,NN1)
      DO 145 NF2=1,NN
  145 Y(I) = Y(I) + W(NW1,NF2)*LC(NF2,NS1)
      IF (Y(I).GT.0.0) NERR2 = NERR2 + MC(NS1)
 1005 Y(I) = -Y(I) - B(I)
C
      PRINT 100, INDEX, NERR1, NERR2, (W(NW1,NFA), NFA=1,NN1)
      IF (TEST) GO TO 1010
 1040 CONTINUE
 1010 NERR = NERR1 + NERR2
      PRINT 220, NERR
C
      IF (NW1.NE.MM1) RETURN
```

```
      WRITE (6,216)
      DO 1220 NW=1,MM1
      NC = MOC(NW)
 1220 PRINT 101, NC, CLASS(NC), (W(NW,NFA), NFA=1,NN1)
      PRINT 101, MOC(MM), CLASS(MOC(MM))
      RETURN
      END
```

```
C
      SUBROUTINE NTEST (LC, MC, NS, CLASS, W, MDC, NN, MM)
C
C  CLASSIFIES KNOWN DATA SAMPLES - NONPARAMETRIC CLASSIFICATION
C
      DIMENSION MC(1), NS(MM,3), X(20), W(1), MDC(1), KS(20)
      LOGICAL*1 LC(NN,1)
      DOUBLE PRECISION CLASS(MM)
  104 FORMAT (//30X,18HAVERAGE ACCURACY =,F6.1,8H PERCENT/30X,32(1H*))
 2008 FORMAT ('1'/30X,29('*')/30X,'* RESULTS OF CLASSIFICATION *'/30X,'*
     .     TRAINING SAMPLES       *'/30X,29('*')///14X,'NUMBER OF  NUMBER
     .     PERCENT     NUMBER OF SAMPLES CLASSIFIED AS'/6X,'CLASS    SAM
     .PLES   CORRECT    CORRECT',10A9/(43X,10A9))
 2009 FORMAT (/I4,A9,I7,I11,F10.1,10I9/(41X,10I9))
C
      NN1 = NN + 1
      MM1 = MM - 1
      PRINT 2008, CLASS
      TE = 0.0
      DO 1301 NC=1,MM
      DO 1109 I=1,MM
 1109 KS(I) = 0
      NSC = NS(NC,1)
      NC11 = NS(NC,2)
      NC12 = NS(NC,3)
C
      DO 1400 NS1=NC11,NC12
      DO 1500 NF=1,NN
 1500 X(NF) = LC(NF,NS1)
      CALL NOPACA (X, ICLASS, W, MDC, 1, NN, NN1, MM1)
 1400 KS(ICLASS) = KS(ICLASS) + MC(NS1)
C
      EFF = 100.0 * KS(NC) / NSC
      PRINT 2009, NC, CLASS(NC), NSC, KS(NC), EFF, (KS(NC1), NC1=1,MM)
 1301 TE = TE + EFF
      AVE = TE / FLOAT(MM)
      PRINT 104, AVE
      RETURN
      END
```

```
C
      SUBROUTINE NCLASS (S, MCLASS, CLASS, W, MOC, NRECS, NSAMP, NN, MM)
C
C NONPARAMETRIC CLASSIFICATION
C CLASSIFIES DATA SAMPLES AND STORES CLASSIFICATION RESULTS ON TAPE
C
      DIMENSION S(NN,NSAMP), MCLASS(NSAMP), W(1), MOC(1), KS(20)
      DOUBLE PRECISION CLASS(MM)
 2010 FORMAT ('1'/30X,29('*')/30X,'* RESULTS OF CLASSIFICATION *'/30X,
     .'*',7X,'DATA SAMPLES',8X,'*'/30X,29('*')///20X,'CLASS',15X,'SAMPLE
     .S',15X,'PERCENT'/20X,5('*'),2(15X,7('*')))
 2011 FORMAT (/I17,A9,I20,F22.2)
 2012 FORMAT (/13X,15HTOTAL SAMPLES,I20/13X,13(1H*))
C

      NN1 = NN + 1
      MM1 = MM - 1
      DO 1009 NC=1,MM
 1009 KS(NC) = 0
C
      DO 12 NREC=1,NRECS
      READ (10) S

      CALL NOPACA (S, MCLASS, KS, W, MOC, NSAMP, NN, NN1, MM1)

   12 WRITE (11) MCLASS
C
      NTOT = NSAMP * NRECS
      PRINT 2010
      DO 1221 NC=1,MM
      PCT = 100.0 * KS(NC) / NTOT
 1221 PRINT 2011, NC, CLASS(NC), KS(NC), PCT
      PRINT 2012, NTOT

      RETURN
      END
```

```
C
      SUBROUTINE NOPACA (X, NN, KS, W, MOC, NSS, NN1, MM1)
C
C NON-PARAMETRIC CLASSIFICATION OF A STRING OF NSS FEATURE VECTORS
C USING PRE-LEARNED LINEAR DISCRIMINANT FUNCTIONS
C
      DIMENSION X(NN,NSS), NW(NSS), W(NN1,MM1), MOC(1), KS(1)
C
      DO 20 NS1=1,NSS
      DO 1 NW1=1,MM1
      G = W(NN1,NW1)
      DO 2 NF1=1,NN
    2 G = G + W(NF1,NW1)*X(NF1,NS1)
      IF (G.GT.0.0) GO TO 3
    1 CONTINUE
      NW(NS1) = MOC(MM1+1)
      GO TO 20
    3 NW(NS1) = MOC(NW1)
   20 KS(NW(NS1)) = KS(NW(NS1)) + 1
      RETURN
      END
```


☆U.S. GOVERNMENT PRINTING OFFICE 1976—740-049/4 REGION NO. 4