4

ETL-0567

Automated Extraction of Airport Runway Patterns From Radar Imagery

Richard A. Hevenor
Pi-Fuay Chen

June 1990

Approved for public release; distribution is unlimited.

U.S. Army Corps of Engineers
Engineer Topographic Laboratories
Fort Belvoir, Virginia 22060-5546

Destroy this report when ııo longer needed.
Do not return it to the originator.

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

The citation in this report of trade names of commercially available products does not constitute official endorsement or approval of the use of such products.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No 0704-0188

1. AGENCY USE ONLY *(Leave blank)*

2. REPORT DATE
June 1990

3. REPORT TYPE AND DATES COVERED
Research Note: June 85 - March 89

4. TITLE AND SUBTITLE
Automated Extraction of Airport Runway Patterns from Radar Imagery

5. FUNDING NUMBERS
✓
4A161102B52C, B, 017

6. AUTHOR(S)
Richard A. Hevenor
Pi-Fuay Chen

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
U.S. Army Engineer Topographic Laboratories
Fort Belvoir, Virginia 22060-5546

8. PERFORMING ORGANIZATION REPORT NUMBER
ETL-0567

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

A method is presented to extract linear terrain features from Synthetic Aperture Radar imagery. An input radar image is smoothed with an edge-preserving smoothing operation. Edge detection is performed using a Sobel operator, and both the magnitude and directional images are computed. The edges are then strengthened using several iterations of a relaxation operation in which both the magnitude image and the directional image are updated with each iteration. The output of the relaxation operation is a binary edge image, which is then thinned. A connected components routine is then run in which two passes through the image are used to provide a unique label for each connected component. The connected components related only to the runway pattern are then extracted by computing certain properties of each component. A border-following algorithm is then used to follow only the outermost borders and give each of the pixels on an outermost border a maximum brightness value. A tracking algorithm is used to change the binary image array into a set of Freeman chain codes, which serve as the input to a line-forming routine that uses a standard polygon approximation algorithm. Experimental results on a real synthetic aperture radar image are presented.

14. SUBJECT TERMS
Radar Image Feature Extraction, Computer Vision, Edge Enhancement, Connected Component

15. NUMBER OF PAGES
41

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT
UNCLASSIFIED

18. SECURITY CLASSIFICATION OF THIS PAGE
UNCLASSIFIED

19. SECURITY CLASSIFICATION OF ABSTRACT
UNCLASSIFIED

20. LIMITATION OF ABSTRACT
UL

CONTENTS

ILLUSTRATIONS

PREFACE

This work was done under DA Project, 4A161102B52C, Task B, Work Unit 017, "Automated Feature Extraction from All-Source Imagery." The objective of this work unit is to develop concepts, methods, and techniques for automated feature extraction from all-source imagery. The work associated with this work unit is accomplished primarily through in-house efforts. Terrain features are selected and prioritized to keep the work unit manageable. The automated extraction of airport runway patterns from radar imagery was considered a high priority and this led to the research described in this report.

The work was performed during the period June 1985 to March 1989 under the supervision of Dr. Frederick W. Rohde, Division Chief, Imagery Research Division, and Dr. Richard B. Gomez, Director, Research Institute.

Colonel David F. Maune, EN, was Commander and Director, and Mr. Walter E. Boge was Technical Director of the U.S. Army Engineer Topographic Laboratories during the report preparation.

AUTOMATED EXTRACTION OF AIRPORT RUNWAY PATTERNS FROM RADAR IMAGERY

INTRODUCTION

The problem of extracting airport runway patterns from optical photography has been the subject of study for several years.[1,2] In both referenced works, the emphasis was on low-level vision computations, and little effort was made to isolate the connected components of the airfield. Also, very little work has been done in extracting airport runway patterns from synthetic aperture radar imagery. Since airports represent a potential military target, they should be extracted from radar imagery as quickly as possible.

The purpose of this report is to present a systematic procedure consisting of a number of image processing algorithms that allow one to go from an original radar image containing an airfield to a binary image consisting only of components that are related to the airfield. All connected components in the image that relate to other terrain features will be eliminated. No attempt was made to make the procedure robust or general. The main objective was to see if the components of an airfield could be isolated for the sample image used. This image was a 512 by 512 pixel image that contained an airfield located near Elizabeth City, North Carolina (see figure 1). The radar system used to obtain this image was the UPD-4 system which is an X-band radar with HH-polarization. The radar image was digitized with 8-bits. The image processing algorithms were written in the LISP programming language and executed on a Symbolics 3670 LISP machine. The same algorithms were later recoded in the C programming language and implemented on a SUN 3/180 microcomputer system for ease of transfer to a development laboratory. The rest of this report will discuss the procedure used to extract the airfield from the radar image. Each of the algorithms used in this procedure will be discussed. Finally, the results obtained after each algorithm is applied will be presented.

METHODOLOGY

The procedure for extracting the airfield from the image shown in figure 1 is given in figure 2. In this procedure, eleven processing routines are applied that begin with edge-preserving smoothing and end with polygon approximation. Some of the processing routines are elementary, such as edge detection; however, some of them are very complicated, such as relaxation for edge reinforcement and border following. The purpose of edge-preserving smoothing is to eliminate noise and at the same time preserve edges so that they are not blurred. The edge detection algorithm will enhance edges and compute the direction of the edge at each pixel. The direction of each edge is needed as an input to the next algorithm, which is relaxation for edge reinforcement. The purpose of this routine is to enhance edges using the contextual information of the surrounding neighborhood. The output of the relaxation calculations is a binary edge image in which many of the strong edges are too thick. A thinning operation is then required in order to thin most edges down to the thickness of 1 pixel

[1] R. Nevatia and K. R. Babu, "Linear Feature Extraction and Description," *Computer Graphics and Image Processing*, vol. 13 pp. 257-269, 198C.

[2] Y. T. Zhou, V. Venkateswar, and R. Chellappa, "Edge Detection and Linear Feature Extraction Using A 2-D Random Field Model," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol.11, No.1, pp. 84-95, January 1989.

in width. The connected components algorithm provides a unique label (number) for each pixel in a given component of 1-pixels. Use is made of the definition of 8-connectivity for 1-pixels and 4-connectivity for 0-pixels. These definitions were also used for the thinning operation. Eleven region property calculations were performed to isolate the connected components that belong only to the airfield. A border-following algorithm is then used to determine and uniquely label all of the 1-pixels that exist between a given connected component of 1-pixels and a connected component of 0-pixels. Two border-following algorithms were implemented for this work. The first algorithm finds and labels all border pixels for both outer borders and hole borders. The second algorithm finds just the outermost borders. The difference between outer borders and hole borders will be explained in detail later. After border following has been completed, lists are generated, each of which consists of an 8-direction chain together with the coordinates of the initial point in the chain. These lists are referred to as Freeman chain codes, and are used as input data to a polygon approximation routine. The purpose of polygon approximation is to approximate irregular curves with a consecutive series of line segments. The particular polygon approximation routine used will be discussed later. The following sections of this report discuss in more detail each of the algorithms mentioned above. The final section will present the results of applying the algorithms to the radar image in figure 1.

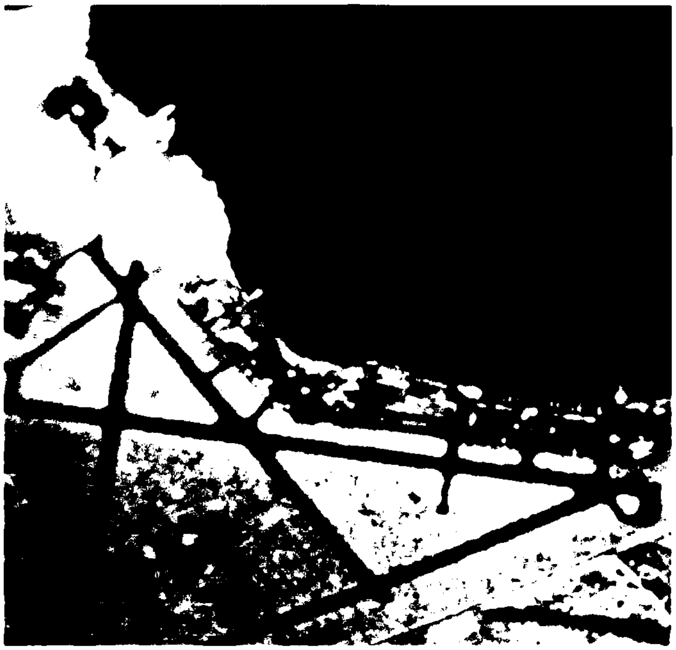

Figure 1. Original Radar Image of an Airfield.

PROCEDURE

EDGE-PRESERVING SMOOTHING → EDGE DETECTION (MAGNITUDE AND DIRECTION) → RELAXATION FOR EDGE REINFORCEMENT → THINNING → CONNECTED COMPONENTS → REGION PROPERTY CALCULATIONS → EXTRACT CONNECTED COMPONENTS OF THE AIRFIELD → BORDER FOLLOWING → ELIMINATE PIXELS NOT ON THE OUTERMOST BORDERS → GENERATE FREEMAN CHAIN CODES → POLYGON APPROXIMATION

Figure 2. Procedure Used for Extracting the Airfield.

EDGE-PRESERVING SMOOTHING

The purpose of an edge-preserving smoothing algorithm is to eliminate noise and to preserve edges from degradation. The variation of the gray tone in a neighborhood around each pixel is used to determine the direction that is most homogeneous. Smoothing is then performed in this direction. A significant question that arises with all smoothing operations is how large should the averaging neighborhood be? In order to reduce noise fluctuations significantly, one should use a large averaging neighborhood. However, if the averaging neighborhood becomes too large, then small regions and details of important boundaries will be destroyed. A good compromise is to use a 5- by 5-pixel neighborhood for the averaging process.

The particular approach to edge-preserving smoothing used in this research consisted of analyzing the gray tone variations within each 5- by 5-pixel area in the image. For each 5- by 5-pixel area, nine geometric figures are formed using the center pixel. Four of the geometric figures are pentagons. Four of the geometric figures are hexagons. One of the geometric figures is a square. The three different kinds of geometric figures are illustrated in figure 3. Each of the four pentagon figures is formed by using the center pixel and one of the outermost edges of the 5- by 5-pixel area. Each of the hexagon figures is formed by using the center pixel and one of the outermost corners of the 5- by 5-pixel area. The 3- by 3-pixel square is formed using the center pixel and its first nearest neighbors. The pixels associated with each of the nine geometric figures are used to compute the mean and variance of the gray tones for each figure. The pentagon and hexagon figures each have 7 pixels associated with them. The square has 9 pixels associated with it. A list of nine means and nine variances is generated from all of the computations involving the nine geometric figures. The gray tone value of the center pixel is replaced by the particular mean gray value that is associated with the smallest variance. The theory behind this edge-preserving smoothing technique was developed by Nagao and Matsuyama.[3] The algorithm can also be used in an iterative manner, that is, the output of one smoothing operation can be used as the input to another.

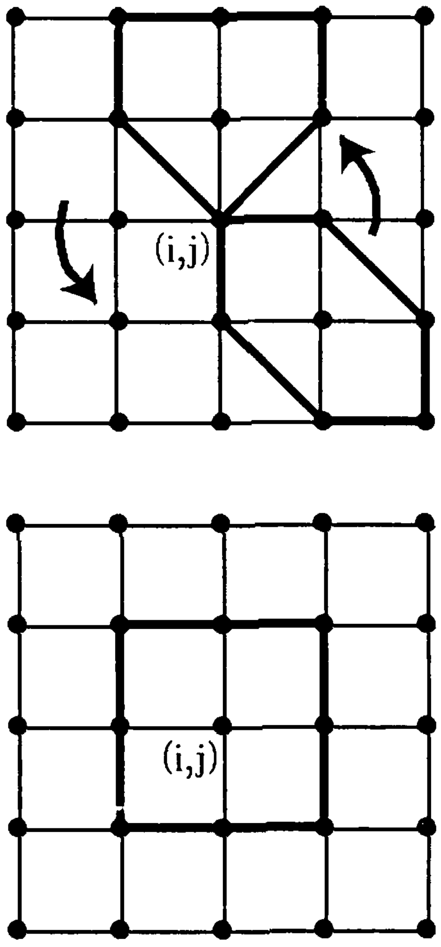

Figure 3. Geometric Figures Used for Edge-Preserving Smoothing.

EDGE DETECTION (MAGNITUDE AND DIRECTION)

After edge-preserving smoothing has been performed, an edge detection operator is used to enhance edges and to compute the direction of each edge. The edge detection operator used was the Sobel operator. This operator consists of two 3 by 3 masks as shown in figure 4. The masks are applied to each pixel to calculate a magnitude image and a directional image. The magnitude image is the edge enhanced image. The directional image contains the direction of the edge at each pixel. The direction of an edge is defined as the angle between the edge and the *x* axis. The *x* axis extends along the top row of the image with the origin at the pixel in the upper left-hand corner. The *y* axis extends downward along the first column of the image. The magnitude image is computed by taking the square root of the sum of the squares of the result of applying the two masks at each pixel. The directional image is calculated by taking the inverse tangent of the ratio of the results of applying the two masks. Because the direction of an edge has a 180 degree ambiguity, a convention must be established to eliminate this ambiguity and establish a fixed direction for each edge. The convention used in this research was that the edge direction was taken in such a way that the darker side is always on the left when facing in the direction of the edge.

EDGE DETECTION (MAGNITUDE AND DIRECTION)

USE THE SOBEL OPERATOR

$$S_x = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \qquad S_y = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$$

$$G_x = f(i + 1, j -1) + 2f(i + 1, j) + f(i + 1, j + 1) - f(i - 1, j - 1) -2f(i - 1, j) - f(i - 1, j + 1)$$

$$G_y = f(i - 1, j -1) + 2f(i,j - 1) + f(i + 1, j - 1) - f(i - 1, j + 1) -2f(i, j + 1) - f(i + 1, j + 1)$$

IF f(i,j) IS THE IMAGE ARRAY mag(i,j) IS THE MAGNITUDE OF THE SOBEL OPERATION AND a(i,j) IS THE EDGE DIRECTION AT THE POINT (i,j), THEN:

$$\mathrm{mag}(i,j) = [G_x^2 + G_y^2]^{1/2}$$

$$a(i,j) = \mathrm{atan}(G_y/G_x)$$

EDGE DIRECTION IS TAKEN IN THE SENSE THAT THE DARKER SIDE IS ON THE LEFT WHEN FACING IN THE DIRECTION OF THE EDGE.

Figure 4. Edge Detection Using the Sobel Operator.

RELAXATION FOR EDGE REINFORCEMENT

The result of applying the Sobel edge operator yields an image in which some edges are defined very well, some edges are poorly defined, and some edges have holes in them. In addition, some large responses are obtained where there are no edges. These errors occur because of noise in the original image and also because the Sobel edge detector is not perfect.

The purpose of the relaxation calculations is to enhance edges by increasing the gray tone value of the pixels that are really on edges, and to decrease the gray tone value of the pixels that are not on edges. Initially, the magnitude and the direction of the edge at each pixel are obtained from the edge detection operation. The magnitude at each pixel location is divided by the maximum of the magnitudes over the entire image in order to define the probability of an edge at each pixel. The location of each pixel will be designated by the quantity (i,j), where i represents the row dimension and j represents the column dimension. The relaxation process for edge reinforcement consists of defining a new edge probability and a new edge angle at each (i,j) in terms of the old ones at (i,j) and its neighbors. The neighbors used in this research were the first and second nearest neighbors. This definition of neighbors is not a restriction on the basic technique. As explained by Schachter,[4] the number of neighbors used in the relaxation calculations is arbitrary. However, if more neighbors are used, the result will be longer computation times.

The calculation of the new edge probability and the new edge angle uses the compatibility coefficients Ree, Ren, Rne, and Rnn. These coefficients are used to determine the relationship between a pixel (i,j) and one of its neighbors at (u,v). For example, if an edge exists at (i,j) and an edge exists at (u,v), one must have a quantitative relationship which explains how an edge at (u,v) will affect the edge at (i,j). In this example, the relation is given by the compatibility coefficient Ree. If an edge exists at (i,j) and no edge exists at (u,v), the relation is given by the compatibility coefficient Ren. Ree is known as the edge/edge coefficient and depends upon the edge angles at the two points, on the angle between a line joining the point (i,j) with (u,v), and the x axis, and on the distance between (i,j) and (u,v). Collinear edges will reinforce one another, while edges at different angles will weaken one another. Similarly, edge points are weakened by points containing no edges that are collinear with them, and the no-edge points are strengthened by edges alongside them. The definitions of the compatibility coefficients, the method of computing the edge probability, and the method of computing the new edge angle are the same as those given by Schachter.[5] The probability of an edge P(i,j) at the pixel location (i,j) is given by:

$$P(i,j) = \frac{mag(i,j)}{max} \tag{1}$$

where max is equal to the maximum value of mag(i,j) taken over the entire image. The probability of no edge at the location (i,j) is 1 - P(i,j). The expressions used for each of the compatibility coefficients will be discussed next.

[4] B. J. Schachter, A. Lev, S. W. Zucker, and A. Rosenfeld, "An Application of Relaxation Methods to Edge Reinforcement," *IEEE Transactions on Systems Man, and Cybernetics*, vol. SMC-7, No. 11, pp. 813-816, November 1977.

[5] Ibid.

The Compatibility Coefficients. The direction of the edge at the point (i,j) is designated by a(i,j). The direction of the edge at the point (u,v), which is a neighbor of (i,j), will be designated by b(u,v). The direction of the line joining (i,j) to (u,v) will be designated g. The chessboard distance (d) from (i,j) to (u,v) is given by:

$$d = \text{maximum}\ (|i-u|, |j-v|) \tag{2}$$

The edge-edge compatibility coefficient Ree is given by:

$$Ree = \cos[a(i,j) - g]\cos[b(u,v) - g]\ /(2)^{d} \tag{3}$$

It can be seen from equation (3) that, when a(i,j) = g = b(u,v), a maximum value for Ree is obtained. This occurs when the edge direction at pixel (i,j) has the same value as the edge direction at pixel (u,v), and both edge directions are equal to the direction of the line joining (i,j) to (u,v). The compatibility coefficient given by equation (3) is not unique and other possibilities could be developed. However, for this research only equation (3) was used for Ree.

The edge probability at point (i,j) will be weakened by a no-edge point, which is collinear with it. The following expression was used for the edge-no-edge compatibility coefficient Ren:

$$Ren = \text{minimum}[0, -\cos(2a(i,j) - 2g)\ /(2)^{d}] \tag{4}$$

In equation (4), minimum means take the minimum of the two quantities enclosed in the brackets.

The no-edge probability at pixel (i,j) will be influenced by the probability of an edge at pixel (u,v), according to the following expression for Rne:

$$Rne = [1 - \cos(2b(u,v) - 2g)]/(2)^{d+1} \tag{5}$$

The no-edge probability at pixel (i,j) will be strengthened by the no-edge probability at pixel (u,v) according to the following equation for Rnn:

$$Rnn = 1/(2)^{d} \tag{6}$$

The Reinforcement Process. The result of considering the total effect of all the first and second nearest neighbors on the edge probability and the no-edge probability at pixel (i,j) is given by Q(i,j) and Qn(i,j) respectively:

$$Q(i,j) = \sum_{\substack{u=i-2 \\ (u,v)}}^{i+2} \sum_{\substack{v=j-2 \\ \neq (i,j)}}^{j+2} \{ C_1 P(u,v)Ree[(i,j),(u,v)] + C_2[1-P(u,v)]Ren[(i,j),(u,v)] \} \qquad (7)$$

$$Qn(i,j) = \sum_{\substack{u=i-2 \\ (u,v)}}^{i+2} \sum_{\substack{v=j-2 \\ \neq (i,j)}}^{j+2} \{ C_3 P(u,v)Rne[(i,j),(u,v)] + C_4 [1-P(u,v)]Rnn[(i,j),(u,v)] \} \qquad (8)$$

The quantities C_1, C_2, C_3, and C_4 are each chosen to have a value between 0 and 1. Also, the sum of the four quantities is chosen to be always equal to 1. We will now define the quantities Qh and Qhb as:

$$Qh = \frac{Q}{|Q| + |Qn|} \qquad (9)$$

$$Qhb = \frac{Qn}{|Q| + |Qn|} \qquad (10)$$

We will also define the quantities Pp and Ppb as:

$$Pp = P(i,j)\ [1 + Qh] \qquad (11)$$

$$Ppb = [1-P(i,j)]\,[1 + Qhb] \qquad (12)$$

The new probability of an edge at the location (i,j) will be designated as Pn(i,j) and is:

$$Pn(i,j) = \frac{Pp}{Pp + Ppb} \qquad (13)$$

Equation (13) provides an expression for the new edge probabilities in terms of the old probability P(i,j) and in terms of the edge probabilities of the first and second nearest neighbors of (i,j). The direction of the edge at location (i,j), given by a(i,j), is also updated to generate a new angle, an(i,j), as follows:

$$Dx(i,j) = W\{P(i,j)\cos[a(i,j)]\} + \sum_{\substack{u=i-2 \\ (u,v) \neq (i,j)}}^{i+2} \sum_{v=j-2}^{j+2} P(u,v)Ree[(i,j),(u,v)]\cos[b(u,v)] \tag{14}$$

$$Dy(i,j) = W\{P(i,j)\sin[a(i,j)]\} + \sum_{\substack{u=i-2 \\ (u,v) \neq (i,j)}}^{i+2} \sum_{v=j-2}^{j+2} P(u,v)Ree[(i,j),(u,v)]\sin[b(u,v)] \tag{15}$$

$$an\ (i,j) = \operatorname{atan}[Dy(i,j)/Dx(i,j)] \tag{16}$$

In the above equations, the quantity W is a constant that is determined, along with C_1, C_2, C_3, and C_4, by an experimental iterative process. Usually, W assumes values between 2 and 10. When W is large, the new edge direction, an(i,j), is almost the same as a(i,j). On each iteration through the image, a new edge probability and a new edge direction are computed.

THINNING

After several iterations of relaxation, the image is essentially binary. However, some of the strong edges are now too thick, and a thinning routine must be used in order to obtain linelike patterns. We will assume that the image consists of only two gray values, represented by 0 and 1. A frame around the image consisting of the first row, the first column, the last row, and the last column will be assumed to contain only 0-pixels. A pixel and its 8 neighbors are *illustrated in figure 5. If the center pixel has the value of 1, then a 1 in any of the 8* neighboring positions will be considered connected to the center pixel. This is known as 8-connectivity. We will assume 8-connectivity exists for 1-pixels and 4-connectivity exists for 0-pixels. The four pixels involved in 4-connectivity are x_1, x_3, x_5, and x_7. The objective of thinning is to determine if a 1 at pixel position x_0, can be changed to a 0. This must be done without destroying the connectivity of any given component of 1-pixels. The particular thinning routine that was employed here makes use of a parameter called the 8-connectivity number. *This parameter is given the symbol* $N(x_0)$ *and is defined by the following expression:*

$$N(x_0) = \sum_{k \in s_1} (\bar{x}_k - \bar{x}_k\ \bar{x}_{k+1}\ \bar{x}_{k+2}) \tag{17}$$

In the above equation, the set S_1 consists of the following odd integers {1, 3, 5, 7}, where $\bar{x}_k$ means 1 - x_k, and the value of a subscript larger than 8 is equal to k - 8. Equation (17) was developed by Yokoi, Toriwaki, and Fukumura[6] to analyze the topological properties of a binary image. The following steps are used to perform the thinning operation:

1. Start a scan of the entire image beginning with the upper left-hand pixel.

2. If a 0 is encountered, continue the scan of the image.

3. If a 1 is encountered, compute the 8-connectivity number.

4. If the 8-connectivity number is equal to 1 and the number of 1-pixels in the 8-neighborhood of x_0 is greater than 1 and $x_3 = 0$, then mark the pixel x_0 to be changed from a 1 to a 0.

5. Complete the scan of the entire image.

6. Change all the pixels from 1's to 0's that have been marked for change.

7. Go to step 1 and repeat steps 1 through 6 except that in step 4 change the criterion $x_3 = 0$ to $x_7 = 0$.

8. Go to step 1 and repeat steps 1 through 6 except that in step 4 change the criterion $x_3 = 0$ to $x_5 = 0$.

9. Go to step 1 and repeat steps 1 through 6 except that in step 4 change the criterion $x_3 = 0$ to $x_1 = 0$.

10. Repeat steps 1 through 9 until all possible deletable 1-pixels are changed from 1 to 0.

X_4	X_3	X_2
X_5	X_0	X_1
X_6	X_7	X_8

Figure 5. Notation Used for the Gray Values in an 8-Neighborhood Surrounding Pixel X_0.

[6] S. Yokoi, J. Toriwaki, and T. Fukumura, "An Analysis of Topological Properties of Digitized Binary Pictures Using Local Features," *Computer Graphics and Image Processing*, No. 4, pp. 63-73, 1975.

CONNECTED COMPONENTS

The purpose of the connected components routine is to provide a unique label for each component of 1-pixels in the binary image that has been thinned. Each label is a number that is assigned to every pixel in a given connected component. This labeling operation can be performed by scanning the entire binary image with a 3 by 3 array and considering the following pattern:

```
C B E
D A
```

If we scan along the image from left to right and from top to bottom, and if pixel A is the pixel presently being considered and it has a value of 1, then a label must be assigned to A. The pixels D, C, B, and E have already been labeled and can be used to label A. If the pixels at D, C, B, and E are all 0, then A is given a new label. If pixels C, B, and E are all 0 and D = 1, then A is given the label of D. Each possible construction of 0's and 1's for the pixels D, C, B, and E must be considered when providing a label for A. If two or more pixels in the set D, C, B, and E are equal to 1 and they all have the same label, then A is also given the same label. The real difficulty comes when two or more of the pixels D, C, B, and E have different labels. This can occur when two or more separate components, which were originally assigned different labels are found to be connected at or near pixel A. For these cases the pixel A is given the label of any one of the pixels D, C, B, or E, which has a value equal to 1. An equivalence list consisting of pairs of equivalent labels is formed. After the binary image has been completely scanned, the equivalence list is restructured in such a way as to contain a number of lists. Each of these lists contains all of the equivalent labels associated with a particular connected component. A new label is then given to each of the new lists and is assigned to each of the appropriate pixels.

REGION PROPERTY CALCULATIONS

The purpose of region property calculations is to determine if the computation of certain quantities can be used to isolate the particular connected components that belong only to the airfield runway pattern. Computations are performed on each connected component. Eleven region properties were used in this research and are:

1. Area.
2. Centroid.
3. Orientation of the axis of least inertia.
4. Maximum moment of inertia.
5. Minimum moment of inertia.
6. Elongation.
7. Measure of region spread.
8. Scatter Matrix.
9. Eigenvalues of the scatter matrix.

10. Perimeter.
11. Compactness.

The definitions of each of the above properties will be discussed. Let the quantity p_{ij} take on the value 1 when (i,j) is located in a connected component of interest, and take the value 0 otherwise. The value of p_{ij} is not equal to the label of the connected component derived previously. The label of a connected component is used only to isolate that component for this analysis. The area of a connected component that has a particular label will be given the symbol A and can be determined from the following equation:

$$A = \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} p_{ij} \tag{18}$$

where m is the number of rows and n is the number of columns in the binary image. The centroid or center of area will be considered located at the coordinates (i_0,j_0), which can be expressed in terms of the previously found area as:

$$i_0 = \frac{1}{A}\sum_{i=0}^{m-1} \sum_{j=0}^{n-1} i p_{ij} \tag{19}$$

$$j_0 = \frac{1}{A}\sum_{i=0}^{m-1} \sum_{j=0}^{n-1} j p_{ij} \tag{20}$$

The orientation of the axis of least inertia is defined as the angle between a vertical line pointing downward and the axis of least inertia for the connected component being considered. The symbol θ will be used for this orientation and can be expressed by the following equation:

$$\begin{aligned} \theta &= \frac{1}{2} \tan^{-1} [2b/ (a - c)] \\ &\text{where:} \\ a &= \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} i^2 p_{ij} - i_0^2 A \\ b &= \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} ij p_{ij} - i_0 j_0 A \\ c &= \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} j^2 p_{ij} - j_0^2 A \end{aligned} \tag{21}$$

The moment of inertia of a region or connected component about a line is just the sum of the squares of the distances from each pixel in that region to the line. Using this basic definition, one can find expressions for the maximum and minimum moments of inertia. These two quantities will be given the symbols I_{max} and I_{min} and equations for them are:

$$I_{max} = \frac{a+c}{2} + \frac{\sqrt{4b^2 + (a-c)^2}}{2} \tag{22}$$

$$I_{min} = \frac{a+c}{2} - \frac{\sqrt{4b^2 + (a-c)^2}}{2} \tag{23}$$

A measure of elongation E can be obtained in terms of the maximum and minimum moments of inertia as follows:

$$E = \frac{I_{max} - I_{min}}{I_{max} + I_{min}} \tag{24}$$

Equation (24) provides a measure of how elongated a region is. The more elongated a region is in one particular direction, the greater the value of E. A measure of region spread, s, can also be obtained by using the maximum and minimum moments of inertia and the area:

$$s = \frac{I_{max} + I_{min}}{A^2} \tag{25}$$

The scatter matrix (S) is a symmetric 2 by 2 matrix and is given as:

$$S = \begin{bmatrix} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} i^2 p_{ij} & \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} ij p_{ij} \\ \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} ij p_{ij} & \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} j^2 p_{ij} \end{bmatrix} \tag{26}$$

The two eigenvalues of S are both real and can easily be computed from (26). The solutions for the two eiyenvalues λ_1 and λ_2 are:

$$\lambda_1 = \frac{d+g}{2} + \frac{\sqrt{(d+g)^2 + 4(f^2-dg)}}{2} \tag{27}$$

$$\lambda_2 = \frac{d+g}{2} - \frac{\sqrt{(d+g)^2 + 4(f^2-dg)}}{2} \tag{28}$$

where d, f, and g are defined as:

$$d = \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} i^2 p_{ij}$$

$$f = \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} ij p_{ij}$$

$$g = \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} j^2 p_{ij}$$

The eigenvalues in (27) and (28) are measurements of the length and width, respectively, of the region under consideration. The perimeter of a region is defined as the number of places that a 0 is adjacent to a 1. Adjacent is taken to mean here that a 1 in the position of x_0 in figure 5 must have a 0 in the position of x_1, x_3, x_5, or x_7 in order to be counted as part of the perimeter. The symbol to be used for the perimeter will be P. The last region property to be considered is compactness, which will be given the symbol C and can be defined in terms of the perimeter and the area as follows:

$$C = \frac{P^2}{A} \tag{29}$$

BORDER FOLLOWING

Border following determines and uniquely labels all 1-pixels that exist between a given connected component of 1-pixels and a connected component of 0-pixels. A border point is defined as a 1-pixel that has at least one 0-pixel in its 4-neighborhood. The 4-neighborhood of a pixel consists of the pixels x_1, x_3, x_5, and x_7 as shown in figure 5. A border point can also be described as a 1-pixel located on the boundary between a connected component of 1-pixels and a connected component of 0-pixels. The frame of a binary image consists of the first row, the last row, the first column, and the last column. It will be assumed that all the pixels on the frame have a gray value of 0. Given a connected component S_1 of 1-pixels and a connected component S_2 of 0-pixels, and if there exists at least one pixel belonging to the component S_2 on every possible 4-connected path from a pixel in S_1 to a pixel on the frame, then the component S_2 is said to surround S_1. If S_2 surrounds S_1 and there exists at least one border point between them, then S_2 is said to surround S_1 directly. Figure 6 shows the definition of surroundings and borders among connected components.

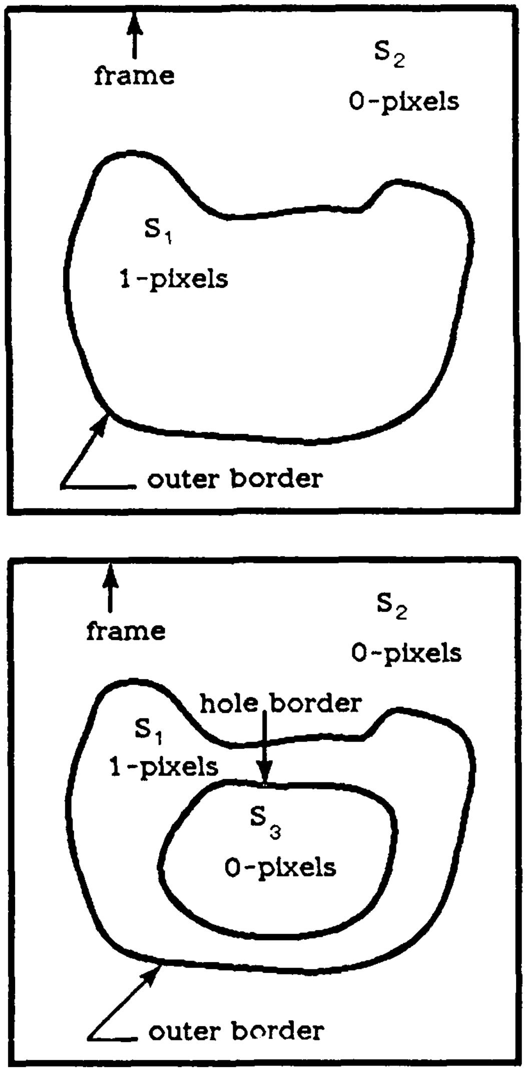

Figure 6. Surroundings and Borders Among Connected Components.

There are two types of borders that can exist in binary images of interest, outer borders and hole borders. An outer border is defined as the set of 1-pixels located between a connected component of 1-pixels and a connected component of 0-pixels which surrounds it directly. A hole border is defined as the set of 1-pixels located between a connected component of 0-pixels and a connected component of 1-pixels that surrounds it directly. For both outer borders and hole borders, the border itself is defined by a set of 1-pixels. The 0-pixels are never used to define a border, except in the case of the frame of the image that is considered a special hole border. For any given connected component of 1-pixels there is one, and only one, outer border and it is unique. For any given hole located within a connected component of 1-pixels the hole border is also unique. If a connected component S of 0-pixels contains the frame of the image, then S is called the background of the image; otherwise S is a hole. The parent border of an outer border located between a connected component S_1 of 1-pixels and a connected component S_2 of 0-pixels is defined as either: (1) the hole border between S_2 and the connected component of 1-pixels that surrounds S_2 directly, if S_2 is a hole; or (2) the frame of the image if S_2 is the background.

The parent border of a hole border between a connected component of 0-pixels S_3 and a connected component of 1-pixels S_1 that surrounds S_3 directly is defined as the outer border between S_1 and the connected component of 0-pixels that surrounds it directly. There are two algorithms that were implemented for border following. The theory behind both of these algorithms was developed by Suzuki and Abe.[7] The basic ideas involved in each of these algorithms will be discussed here. A detailed presentation of the border-following method is given in the appendix. An algorithm that finds all of the borders is initiated by scanning the binary image row by row, beginning in the upper left-hand corner and interrupting the scan only when a pixel at the coordinates (i,j) is found that satisfies the condition for the starting point of an outer border or a hole border. These conditions are given in figures 7 and 8:

	j - 1	j
i	0	1

Figure 7. Starting Condition For An Outer Border.

	j	j + 1
i	> = 1	0

Figure 8. Starting Condition For A Hole Border.

If a pixel at location (i,j) satisfies both of the above conditions, then (i,j) must be regarded as the starting point of an outer border. A unique number is assigned to each newly found border. This number will be called the sequential number of the border and will be designated NBD. The parent border of a newly found border is also determined as follows. While scanning the image, the number of the border (NBD) encountered just prior to finding the newly found border is given the designation LNBD. This previous border is either the parent of the newly found border or a border that shares a common parent with the newly found border. If border

[7] S. Suzuki and K. Abe, 'Topological Structural Analysis of Digitized Binary Images By Border Following,' *Computer Vision, Graphics and Image Processing*, No. 30, pp. 32-46, 1985

C is the newly found border and border B is the most previously found border, then the parent border of C is determined by using figure 9. As an example, if C is an outer border and B is an outer border, then the parent border of C is equal to the parent border of B. If C is a hole border and B is an outer border, then the parent border of C is equal to B.

Type of border for B Type of border for C	Outer border	Hole border
Outer border	The parent border of B	The border B
Hole border	The border B	The parent border of B

Figure 9. Determining the Parent Border of the Newly Found Border C.

The parent border of a newly found border can be used to determine various topological properties of a binary image. However, these topological properties of a binary image were not utilized in this research. The newly found border is followed by using the following marking scheme. The symbols (p,q) will be used to designate the location of an arbitrary point on the border being followed. If this border that is being followed is between a connected component of 0-pixels, which includes the pixel at (p,q + 1), and a connected component of 1-pixels, which includes the pixel at (p,q), then the value of the pixel at location (p,q) is changed to −NBD. Otherwise, the value of the pixel at location (p,q) is set equal to NBD unless the pixel at (p,q) is on a border that has already been followed. These conditions prevent the pixel at (p,q) from being the starting point of an already followed hole border or outer border, respectively.

After following and marking the entire border, the scan is resumed. When the scan reaches the pixel in the lower right-hand corner of the image, the process is completed. This border-following algorithm can be modified so that only the outermost borders of a binary image are followed. An outermost border is an outer border between a connected component of 1-pixels and the background. Border following is initiated only at the pixels that satisfy the condition for the starting point of an outer border and the parameter LNBD is less than or equal to 0. The marking policy is the same as that given by the previous algorithm except that the values 2 and −2 are substituted for the values NBD and −NBD, respectively.

Generate Freeman Chain Codes. Once the border pixels have been found, the binary image must be converted into a format that can be used to form line segments. The Freeman chain code is used to represent a boundary with a sequence of numbers, each of which indicates the change in direction from one border pixel to the next. The 8-direction chain is shown in figure 10. Each number corresponds to a particular change in direction from one pixel to the next. For example, the number 0 represents a change in direction of 0 degrees from one pixel to the next. The angular change is measured from the horizontal to a line formed by joining the current pixel in the chain to the next pixel in the chain. The number 4 represents a change of 180 degrees from one pixel to the next, etc. One entire chain code is represented by a list whose first two elements are the (i,j) coordinates of the initial point in the chain, followed by a sequence of numbers each of which comes from the set {0, 1, 2, 3, 4, 5, 6, 7}. Each of these numbers represents the change in direction from one pixel to the next. The entire binary image will consist of many such chains. To form the chain codes, a tracking routine was developed that examines the nearest neighbors of a pixel in the chain to find the next pixel in the chain. An index array, which is as big as the original image and is initially set to contain all zeroes, is used to indicate if a 1 has already been included in a chain or not. The input binary image is scanned by starting with the pixel in the upper left-hand corner of the image. When a 1 is found at the location (i,j) and the contents of the index array at (i,j) is 0, a new chain is started. As each pixel in the chain is found, the appropriate number from the set {0, 1, 2, 3, 4, 5, 6, 7} is placed into the chain list and a 1 is placed into the corresponding location of the index array. When the scan reaches the pixel in the lower right-hand corner, the algorithm terminates.

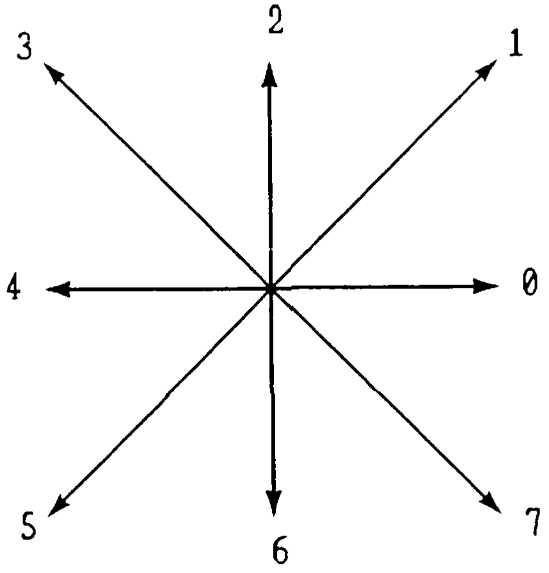

Figure 10. Freeman Chain Code Directions.

POLYGON APPROXIMATION

An arbitrary, two-dimensional, open or closed digital curve is represented by a Freeman chain code and consists of an ordered set C of N pixels. The purpose of polygon approximation is to replace each digital curve with a consecutive series of line segments, each of which are within a certain error of the original curve. There are many polygon approximation techniques, and the theory behind the one used in this research was developed by Ramer.[8] The pixels in the set C can be considered as the vertices p_i of a polygon if the curve is closed. If the curve is not closed, the set C can be considered as the vertices of a consecutive series of line segments. In either case, the task of the polygon approximation algorithm is to provide

[8] U. Ramer, 'An Iterative Procedure For The Polygonal Approximation Of Plane Curves,' *Computer Graphics and Image Processing,* No. 1, pp. 244-256, 1972.

a reduced set C' with a smaller number of edges N' and whose vertices p'_k coincide with those of the original set C. The basic idea behind Ramer's method is illustrated in figure 11. The solid curve between points A and B represents a digital curve to be approximated by a consecutive series of line segments. The first approximation is the dashed line between points A and B. If this line approximates the given curve within an acceptable error, then the algorithm terminates. If, however, it is decided that the error is unacceptable, then the curve is divided into two sections, AC and CB. The point C is the point on the curve AB, which has the maximum perpendicular distance from the dashed line between A and B. The curve is then approximated with the two line segments, AC and CB. If these two line segments approximate the curve good enough, then the algorithm terminates. If not, the curve between points A and C is broken at D and the curve between C and B is broken at E. This process continues until all points on the curve are within an acceptable error of the approximating line segments. The final result for the curve in figure 11 is the series of line segments AD, DC, CE, EF, and FB. This algorithm is applied to each chain in the binary image.

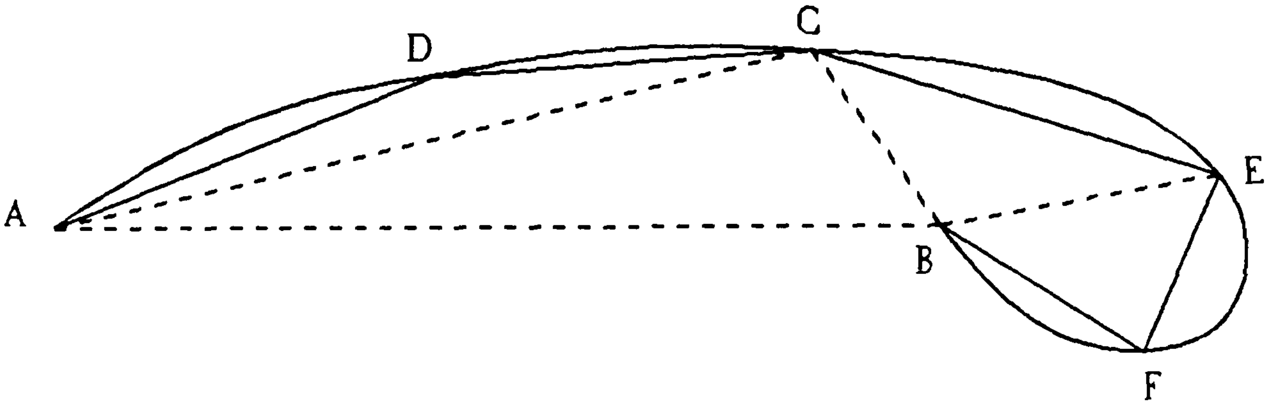

Figure 11. Polygon Approximation.

RESULTS

In this section, the results of applying the above algorithms to the radar image shown in figure 1 will be presented. Figure 12 shows the original radar image in pseudocolor. Each gray value is associated with a unique color for this presentation. The colors were chosen in such a way that black is associated with a gray value of 0 and white is associated with a gray value of 255. The image in figure 13 shows the result obtained after two applications of edge-preserving smoothing. Although there appears to be little difference between this image and the original, it was found that eliminating this step produced much poorer results further on. The image in figure 14 is the result of applying the Sobel edge operator to the smoothed image. This image is the magnitude of the Sobel operation and another array is used to hold the edge direction information. It can be seen that although most edges have been enhanced, the results are not uniform along any one edge. Also, there are many responses where there are no real edges. The edge direction information is computed and placed in a separate array, but cannot be displayed as an image. The image in figure 15 is the result obtained after nine iterations of relaxation for edge reinforcement. These nine iterations took 3½ to 4 hours of computer time on the Symbolics 3670. Computations were performed for several different values of the coefficients C_1, C_2, C_3, C_4 and W. It was found that making the value C_1 larger resulted in thicker edges, while if the value of C_4 became too small, gaps or holes would appear at various places. The final values that were utilized to obtain the image in figure 15 are given as follows:

$$C_1 = 0.76 \quad C_2 = 0.23 \quad C_3 = 0.005 \quad C_4 = 0.005 \quad W = 8.0$$

The image obtained from the relaxation calculations is almost binary and can be used as an input to a thinning operation. The image in figure 16 is the result of applying the thinning operation, which was discussed earlier, to the edge-reinforced image. Most edges are now only one pixel wide. The appearance of the image in figure 16 is such that the edges seem to be more than one pixel in width. This is due to the jitter that existed in the color monitor at the time the picture was taken. The image in figure 17 is the result of applying the connected components algorithm to the image of figure 16. There are 114 separate components in this image and each one has been assigned a slightly different color. The largest component in this image is the orange component in the upper left corner. This component consists of the top portion of the runway pattern along with a small parking lot just north of the airfield. The image in figure 18 shows that the connected components belonging to the airfield have been extracted and the other components thrown away. This was accomplished by using the region properties of area, the orientation of the axis of least inertia, and the measure of region spread. It was discovered later that the orientation of the axis of least inertia was not required to extract the connected components belonging only to the airfield. This figure shows 11 connected components that make up the airfield. The image in figure 19 presents the results after the outermost border algorithm was applied to the image in figure 18. The white pixels are all pixels that are located on outermost borders. The orange and yellow pixels that are still in the image are pixels that are not located on an outermost border. Figure 20 shows the result of following the outermost borders with the nonborder pixels removed. Figure 21 shows the application of polygon approximation to the outermost borders obtained previously. The scale of this final image has been increased in order to show clearly the linear segments of the polygon approximation. A maximum error of 2 pixels was used to arrive at this image. The runway pattern of the airfield has clearly been extracted; however, it remains to be seen as to how robust the approach is.

Figure 12. Original Radar Image In Pseudocolor.

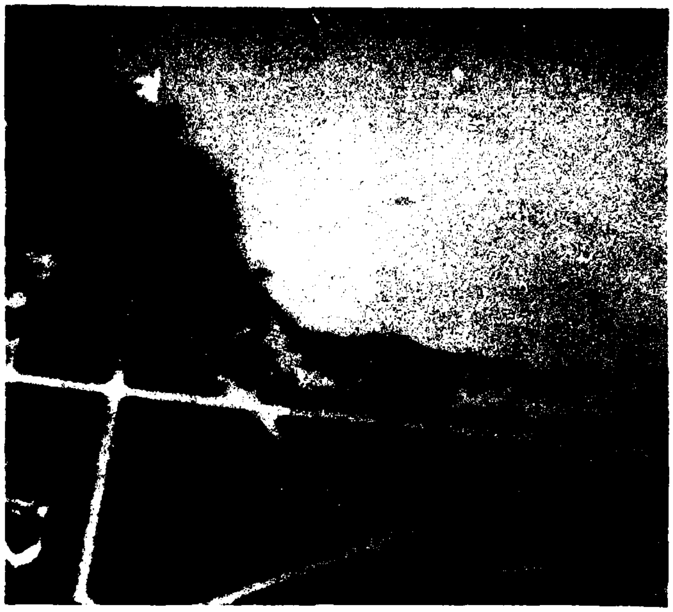

Figure 13. Edge-Preserving Smoothing.

Figure 14. Edge Detection.

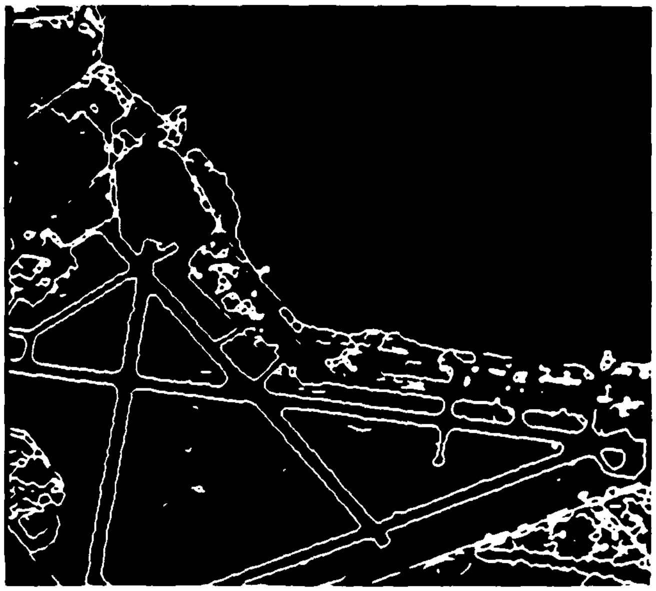

Figure 15. Relaxation for Edge Reinforcement.

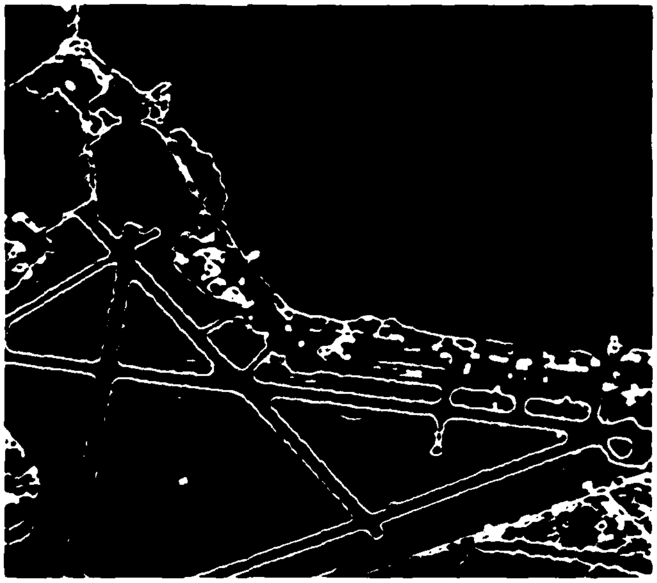

Figure 16. Thinning.

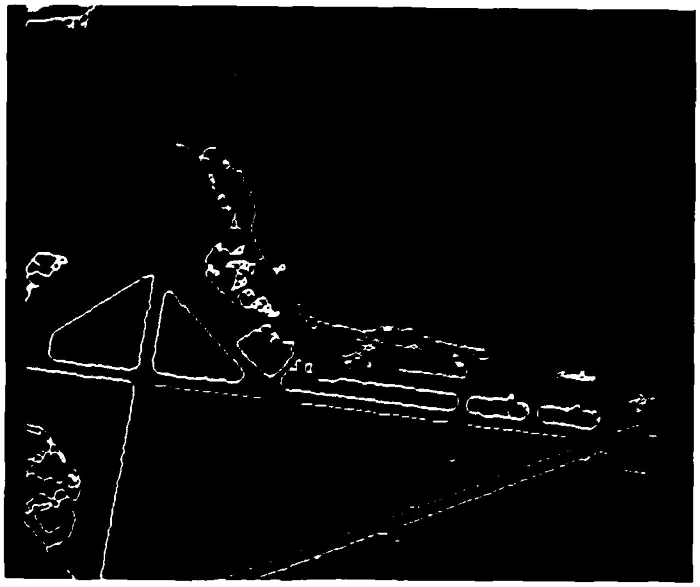

Figure 17. Connected Components.

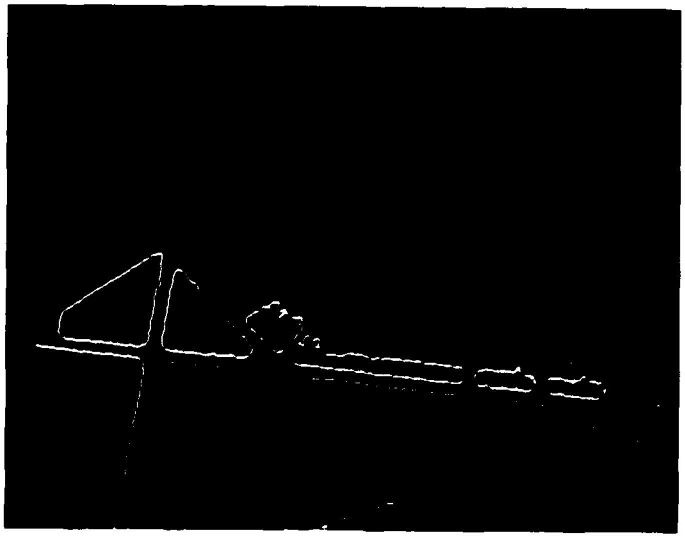

Figure 18. Airfield Connected Components Extracted.

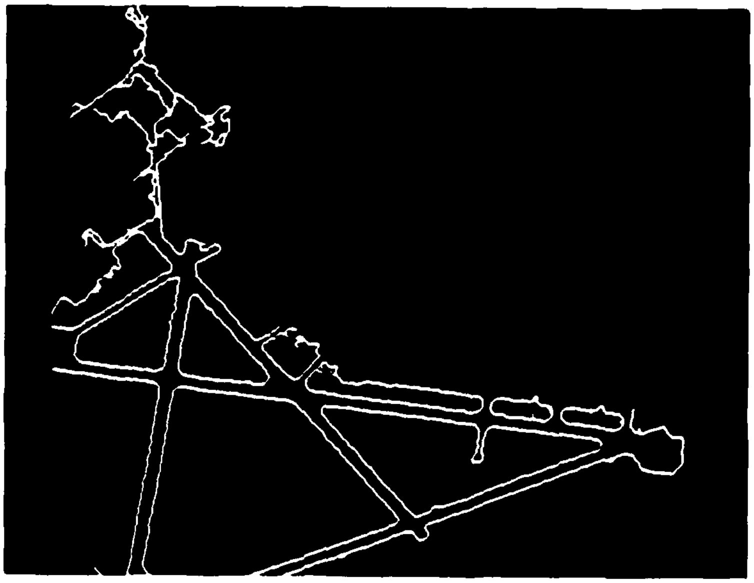

Figure 19. Outermost Borders.

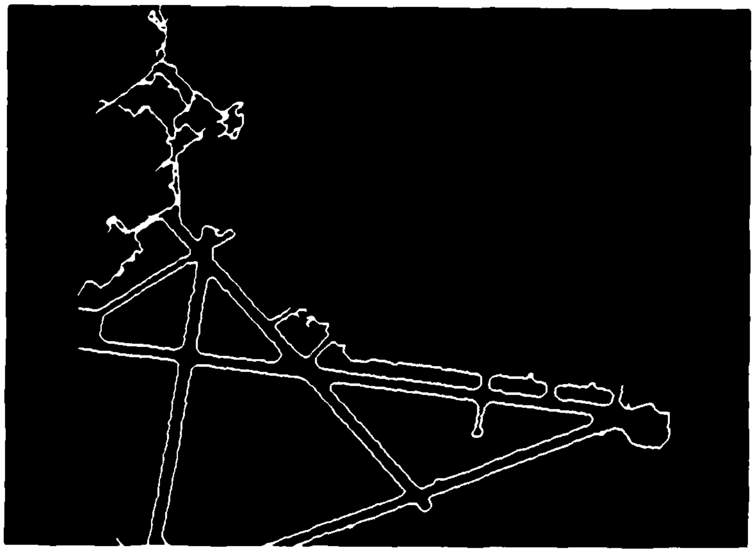

Figure 20. Outermost Borders with Nonborder Pixels Removed.

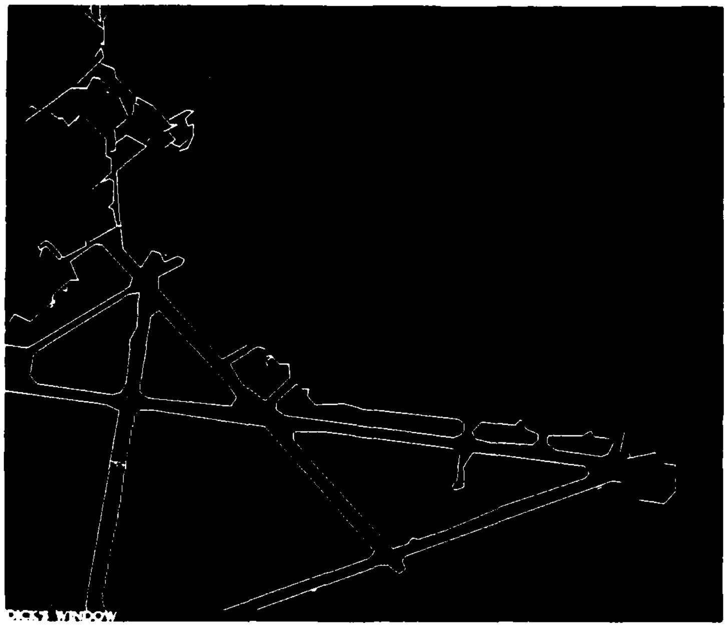

Figure 21. Polygon Approximation.

REFERENCES

Nagao M. and T. Matsuyama. *A Structural Analysis of Complex Aerial Photographs*, New York: Plenum Press, 1980.

Nevatia R. and K. R. Babu. "Linear Feature Extraction and Description," *Computer Graphics and Image Processing*, vol.13, pp. 257-269, 1980.

Ramer, U. "An Iterative Procedure For The Polygonal Approximation Of Plane Curves," *Computer Graphics and Image Processing,* No.1, pp. 244-256, 1972.

Schachter, B. J., A. Lev, S. W. Zucker, and A. Rosenfeld. "An Application of Relaxation Methods to Edge Reinforcement," *IEEE Transactions on Systems Man, and Cybernetics,* vol. SMC-7, No. 11, pp. 813-816, November 1977.

Suzuki, S. and K. Abe. "Topological Structural Analysis of Digitized Binary Images By Border Following," *Computer Vision, Graphics and Image Processing*, No. 30, pp. 32-46, 1985.

Yokoi, S., J. Toriwaki, and T. Fukumura. "An Analysis of Topological Properties of Digitized Binary Pictures Using Local Features," *Computer Graphics and Image Processing*, No. 4, pp. 63-73, 1975.

Zhou, Y. T., V. Venkateswar, and R. Chellappa. "Edge Detection and Linear Feature Extraction Using A 2-D Random Field Model," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol.11, No.1, pp. 84-95, January 1989.

APPENDIX. DESCRIPTION OF THE BORDER-FOLLOWING METHOD

Let an input binary image be designated by the symbol f(i,j). The value of f(i,j) will be either 1 or 0 at the location (i,j). A unique number, designated NBD, is assigned to each newly found border. Initially, NBD is set equal to 1 to correspond with the frame of the image. The frame of the image forms a special hole border and receives the sequential number (NBD) equal to 1. The input binary image is scanned in a raster fashion beginning with the pixel in the upper left-hand corner. The number of the border (NBD) encountered just prior to finding a new border is designated by the quantity LNBD. Each time the scan starts a new row in the image, the value of LNBD is reset to 1. The following four steps are carried out for each pixel that has a value of f(i,j), unequal to 0:

Step 1. Select one of the following:

a. If f(i,j) = 1 and f(i,j-l) = 0, then make the decision that the pixel (i,j) is the starting point of an outer border, the quantity NBD is incremented by 1 and set $i_2 = i$ and $j_2 = j - 1$. The following convention is used to designate the border type:

outer border → type = 1

hole border → type = 0

b. Else, if f(i,j) > 1 or f(i,j) = 1 and f(i,j+1) = 0, then make the decision that the pixel (i,j) is the starting point of a hole border. The quantity NBD is incremented by 1. Also, set $i_2 = i$ and $j_2 = j + 1$ and set LNBD = f(i,j) when f(i,j) > 1.

c. Otherwise, go to step 4.

Step 2. Depending on the types of the newly found border and the border with the sequential number LNBD (the last border met on the current row), determine the parent border of the newly found border by using the method illustrated in figure 9.

Step 3. Starting with the point (i,j), follow the newly detected border. This is accomplished by using steps 3.1 to 3.5:

step 3.1 Starting with the pixel at the location (i_2,j_2), move around the pixel (i,j) in a clockwise direction and look for the first pixel that has a value unequal to 0. Let the coordinates of the first nonzero pixel found be (i_1,j_1). If no nonzero pixel is found, then set f(i,j) = —NBD and go to step 4.

step 3.2 Set $i_2 = i_1$, $j_2 = j_1$, $i_3 = i$, and $j_3 = j$.

step 3.3 Starting with the pixel located next to the pixel at (i_2,j_2) in a counterclockwise direction, examine each pixel in the neighborhood of the pixel (i_3,j_3) to find a nonzero pixel and let the coordinates of the first one be (i_4,j_4).

step 3.4 The value of the pixel at location (i_3,j_3) is changed according to the following rule:

(a) If the pixel at $(i_3,j_3 + 1)$ is a 0 pixel examined in substep 3.3 above, then set the pixel at location (i_3,j_3) equal to —NBD.

(b) If the pixel at location $(i_3,j_3 + 1)$ is not a 0 pixel examined in substep 3.3 and the value of $f(i_3,j_3) = 1$, then change $f(i_3,j_3)$ to be equal to NBD.

(c) Otherwise, do not change the value of the pixel at the location (i_3,j_3).

step 3.5 If $i_4 = i$, $j_4 = j$, $i_3 = i_1$ and $j_3 = j_1$, which means coming back to the original starting point, then go to step 4; otherwise, set $i_2 = i_3$, $j_2 = j_3$, $i_3 = i_4$, $j_3 = j_4$ and go back to step 3.3.

Step 4. If f(i,j) is not equal to 1, then put $|f(i,j)| \rightarrow$ LNBD and resume the raster scan at the pixel located at $(i,j + 1)$. The algorithm terminates when the scan reaches the pixel in the lower right-hand corner of the image.