Summer 1979

A Microcomputer Implementation of a Flight Simulator Visual Display System

Jerry Wayne Campbell
University of Central Florida

A MICROCOMPUTER IMPLEMENTATION OF A FLIGHT SIMULATOR VISUAL DISPLAY SYSTEM

BY

JERRY WAYNE CAMPBELL
B.S.E., Florida Technological University, 1972

RESEARCH REPORT

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Engineering
in the graduate studies program of the
College of Engineering
University of Central Florida, Orlando, Florida

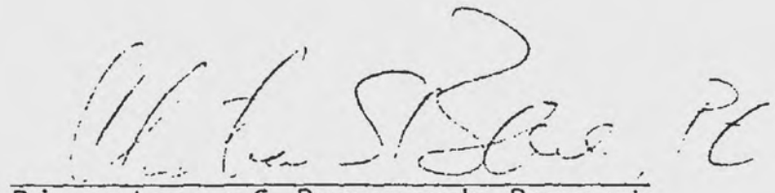
Summer Quarter
1979

ABSTRACT

The use of computer graphics has brought about a universal means of depicting phenomena or solving engineering problems. The combination of graphic and computer skills offers a solution to a number of technical needs. One major area that computer graphics can be used is in the field of flight simulation. The software packages necessary to project three-dimensional scenes for flight training are not very common. The purpose of this paper is to generate the programs necessary to allow the pilot to observe a three-dimensional scene from any location and angle.

The report discusses some of the basic aspects of computer graphics and presents specific computer software for projecting simulated flight landings at Herndon Airport in Orlando, Florida. The three-dimensional projections are calculated and displayed on the video monitor in the form of two-dimensional scenes. The Southwest Terminal System hardware was used as the interface to the video monitor. A description of the various operating parameters for simulated flight landings is presented. Included in the discussion are various pictorial representations of simulated landing approaches at Herndon Airport.

Suggestions for additional work that could be undertaken in this research area concludes the report.

Approved by: 

Director of Research Report

TABLE OF CONTENTS

List of Tables iv

List of Illustrations v

Introduction 1

Chapter

I. Graphic Principles of Flight Simulation 5

II. Simulator Control Parameters 24

III. System Operations 41

IV. Conclusions and Suggestions for future research 72

Appendix A

Flight Simulation BASIC Listing 77

Appendix B

Display Device Driver Routine 83

List of References 89

LIST OF TABLES

1. Three-Dimensional Data Points for Base Case (Hundreds of Feet) 44
2. Three-Dimensional Data Points for Various Landing Approaches to Herndon Airport 47

LIST OF ILLUSTRATIONS

1. Rotational Matrix Equation 8
2. Viewer's Direction and Field of View 10
3. Point Translation of a Data Base 12
4. A Line in Need of Clipping 14
5. Push Mathematic Transformation 16
6. The Difference Transformation Order Makes 19
7. Direction of Movement Conventions 21
8. General Layout of Herndon Airport and Associated Runways 25
9. The 3D to 2D Converter Program and Its Arrays . . .29
10. Flow Chart of Flight Simulator Program 31
11. Subroutine Structure for Flight Simulator 36
12. Transfer System View of a Module for Rotation, Translation, Clipping, and Perspective 42
13. Point Approaches to Herndon Airport 48
14. Aerial View of Herndon Airport at 5000 Feet 49
15. Aerial Approach to Runway 7 at 3000 Feet Altitude .51
16. Aerial Approach to Runway 7 at 1500 Feet Altitude .52
17. Aerial Approach to Runway 7 at 800 Feet Altitude . .53
18. Aerial Approach to Runway 7 at 200 Feet Altitude . .55
19. Aerial Approach to Runway 7 at 100 Feet Altitude . .56
20. Aerial Approach to Runway 25 at 800 Feet Altitude .57
21. Aerial Approach to Runway 25 at 400 Feet Altitude .59
22. Aerial Approach to Runway 25 at 200 Feet Altitude .60

23. Aerial Approach to Runway 31 at 600 Feet Altitude . .61
24. Aerial Approach to Runway 31 at 200 Feet Altitude . . 63
25. Aerial Approach to Runway 31 with a 45 Degree Bank 64
26. Aerial Approach to Runway 31 with a -45 Degree Bank 65
27. Aerial Approach to Runway 31 with a 180 Degree Bank 67
28. Aerial Approach to Runway 25 with a 90 Degree Bank .68
29. Aerial Approach to Runway 31 with a 90 Degree Bank . 70

INTRODUCTION

The increasing availability of digital computers has caused a basic change in the engineers and scientists ability to gain insight into their problems and analyses. The speed and accuracy of the digital computer has brought forth the tools necessary to solve complex mathematical models in a matter of seconds. The mathematical models which use to take hours and even days to solve for one set of data points can now be analyzed on a digital computer as a discrete process. This modeling and representation of a physical or conceptual situation through the use of digital computers has given rise to the field of "Simulation". As the engineers and scientists develop to keep pace with their tasks, they will find that simulation through the use of digital computers will be one of the most useful tools they will have at their disposal.

Once the mathematical model has been formulated, the simulation must be programmed to behave like the system under study and the digital computer thus becomes a working model of the system. The initial values of the system variables and parameter values become numbers in the digital computer program. Changing these values

provides a new situation and the computer provides the corresponding response. The relationship between the physical variables and the simulated system response gives an intuition into system analysis and problem solution.

Once the computer has calculated the responses that correspond to the changed values in the simulation, each new situation must be interpreted into a form suitable for human interpretation. A method is needed which will increase the speed of information transfer and allow dynamic interaction between the operator and the computer (Risher, 1976). Computer graphics is this combination of visual display of graphic art and computer method. It is simple, fast, and economical in converting large sums of engineering data into two-dimensional pictures.

Computer graphics are being used more and more in today's highly technical world. A major factor that has attributed to this increased use of computers and graphic displays has been in the declining costs of the system hardware and software. There is strong speculation that the price for the systems will continue to decline as more manufacturers enter the graphic display system market.

Computer graphics has become increasingly helpful in flight simulation. Today's needs for increased efficiency and reduced costs in the training of flight

crews--made more intense by the spectre of energy shortages--are being served more and more by simulation techniques (General Electric, 1977). Flight simulator training through the use of computer graphics is an area where three-dimensional graphics has many advantages over the real thing. Flight training costs are lowered, there is no interference from bad weather, and there is no risk of crashing. Spins, dives, near crashes and generally pushing the plane to and beyond its limits are all safe maneuvers and a pilot can learn what to expect in any of these situations (Elson, 1976).

In order to project flight simulation patterns, three-dimensional graphic programs must be developed. Three-dimensional graphic programs are very complex in nature. The programs require complicated mathematical theories and computations. The programs are difficult to write, debug, and test. Intricate software and hardware interactions are involved. Very few engineers or scientists have simultaneously the knowledge and background to effectively write, debug, and test three-dimensional graphics. Programs which allow a viewer to observe a three-dimensional scene from any location and angle are nearly non-existant. The complex transformations required to project three-dimensional graphic scenes on a two-dimensional screen requires each software program to be written for each application.

A problem of particular interest was to generate the necessary software to simulate flight landings at Herndon Airport in Orlando, Florida. The programs necessary to do point translation, point rotation, line clipping, and projection, were written or modified to be used on the Southwest Terminal graphic equipment. Although this is relatively slow process due to the use of BASIC programming, the projection of flight simulated approaches are very close to real world application. The application of microprocessor technology and graphic terminal display is the essence of this research.

CHAPTER 1

GRAPHIC PRINCIPLES OF FLIGHT SIMULATION

Since a display screen is a two-dimensional space, one cannot display three-dimensional flight perspectives without the use of some mathematical transformations. Three-dimensional coordinate transformation equations must be generated in order to convert flight perspectives, which are three-dimensional, into two-dimensional points. Computer Graphics can then be employed to generate from a base case true three-dimensional flight perspectives onto a two-dimensional space screen (Fetter, 1965).

The three coordinate transformations that must be considered are:

1. Rotation
2. Translation
3. Clipping

For each of these transformations, equations and associated matricies must be set up so that the proper perspectives can be generated.

These transformations can be classified under the generic name geometric transformation. Geometric transformations are bijective mappings from the coordinate

space (Giloi, 1978). The picture structure will not be affected and the base case relations between the points of a picture are preserved. Therefore, geometric transformations are performed on data points without changing the original data base.

A brief explanation of each coordinate transformation will be presented along with the mathematical equations that will be used to perform the transformations. If a person is interested in the mathematical derivations of the equations, then the book Principles of Interactive Computer Graphics, by William M. Newman, should be sought. This study was not intended to derive the equations, but simply to use the equations to solve the transformations necessary for flight simulation using computer graphics.

Rotation

A general rotation in the three-dimensional XYZ-space assumes that the world rotates around the viewer. This in essence is the viewers direction and assumes that the viewer's location is at 0, 0, 0 in three coordinate space. Because of the orthogonality, we obtain a general rotation simply by multiplying a 3 x 3 matrix by a three element vector (Sub Logic, 1976). This matrix only has to be generated once per each viewing direction since it applies to all points for that view.

The rotation or viewer's direction consists of four parameters:

1. Pitch
2. Bank
3. Heading
4. Angle of View

Pitch is a floating point variable that specifies the angle of inclination from which the viewer looks at the scene. Bank is a floating point variable representing the angle at which a viewer's head is tilted sideways when viewing the scene. Heading is the direction the viewer is facing while standing on the XZ plane.

Sines and Cosines of the pitch, bank and heading must be computed when generating the matrix. This is done in this study by a subroutine that takes the F argument and generates a cosine by expanding a cosine series. Sine is generated by initially shifting the angle 90 degrees and then expanding the cosine series. This type of calculation was necessary since the BASIC language used by the programmer did not have trigonometric functions.

The 3 x 3 matrix, which when multiplied by a three element vector, used to rotate the scene about the origin is shown in Figure 1. This figure shows the rotational matrix being multiplied by the original 3D space

$$
X'Y'Z' = [XYZ]
\begin{bmatrix} \cos H & \cos B \\ + & \\ \sin H & \sin P \\ & \sin B \end{bmatrix}
\begin{bmatrix} -\cos H & \sin B \\ + & \\ \sin H & \sin P \\ & \cos B \end{bmatrix}
\begin{bmatrix} \sin H & \cos P \end{bmatrix}
$$

$$
\begin{bmatrix} \cos P & \sin B \end{bmatrix}
\begin{bmatrix} \cos P & \cos B \end{bmatrix}
\begin{bmatrix} -\sin P \\ \cos H & \cos P \end{bmatrix}
$$

$$
\begin{bmatrix} -\sin H & \cos B \\ + & \\ \cos H & \sin P \\ & \sin B \end{bmatrix}
\begin{bmatrix} \sin H & \sin B \\ + & \\ \cos H & \sin P \\ & \cos B \end{bmatrix}
$$

Where $[X'Y'Z']$ = Rotated Point

$[X\ Y\ Z]$ = Original Point

P = Pitch

B = Bank

H = Heading

Figure 1. Rotational Matrix multiplied by original 3D space coordinate point yielding the rotated point

coordinate point yielding the rotated point (Sub Logic, 1976).

The fourth parameter used in the rotation of the original base case is called the viewer's field of view. This parameter is similar to the lens on a camera. This parameter limits the angle that the viewer will see when making the approach to the landing strip.

Field of view is a floating point value representing the tangent of the half field of view. A value of one represents a forty-five degree half field or ninety degree full view of field. A value of three would represent a narrow telephoto view. Figure 2 depicts the four parameters used in the rotation of a scene.

Anytime a change of viewer's direction or rotation takes place, a subroutine must take the change in pitch, bank, and/or heading, as well as field of view and create a new predetermined transformation matrix. This new transformation matrix must then be multiplied by the original data to create a new rotated scene.

Translation

The viewer's location in flight simulation is always considered to be at 0, 0, 0 in a coordinate space. When the position of the aircraft is at a location other than 0, 0, 0, the points in the data base must be translated while the viewer remains at 0, 0, 0. In other words,

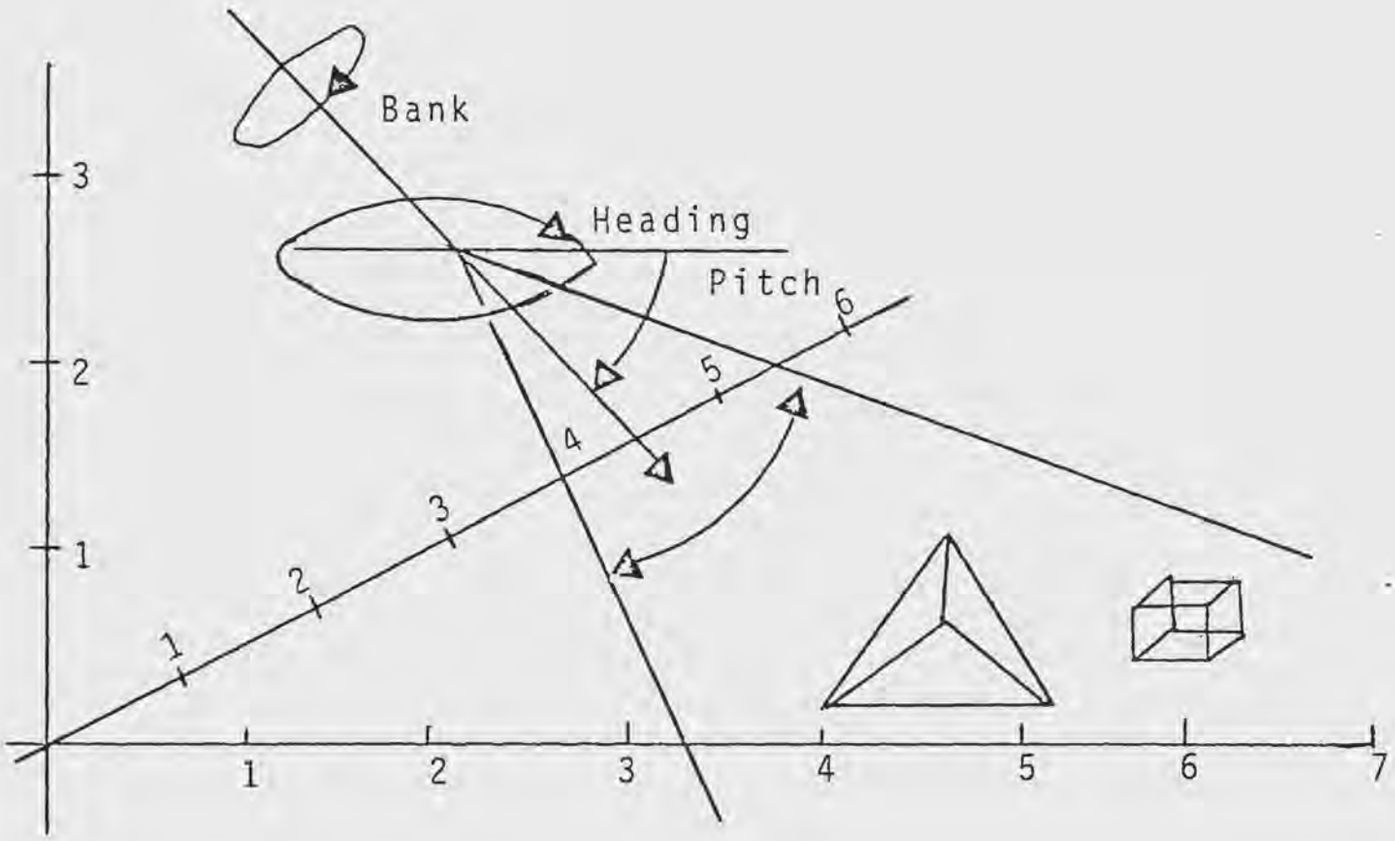


Figure 2. Viewer's direction and field of view

the world moves and the viewer remains fixed.

Translation of a point is performed by adding a positive or negative constant to each point of the three-dimensional data base. If these translation parameters are noted by X_{NP} (new position in X-direction), Y_{NP} (new position in Y-direction), and Z_{NP} (new position in Z-direction), the new translation position would be:

$X_{NTP} = X_{OP} + X_{NP}$ where X_{OP} = old position in X-direction.

$Y_{NTP} = Y_{OP} + Y_{NP}$ where Y_{OP} = old position in Y-direction.

$Z_{NTP} = Z_{OP} + Z_{NP}$ where Z_{OP} = old position in Z-direction.

Figure 3 illustrates translation of a data base.

Clipping

The simple application of translation and rotation to produce a perspective image has two undesirable effects.

1. Objects behind the viewpoint may appear on the screen.
2. Objects may exceed the prescribed limits of the viewpoint.

These effects are eliminated through the use of a clipping algorithm. The mathematics of clipping a line and pushing end points to screen boundaries are quite simple, but deciding which way to push them is what

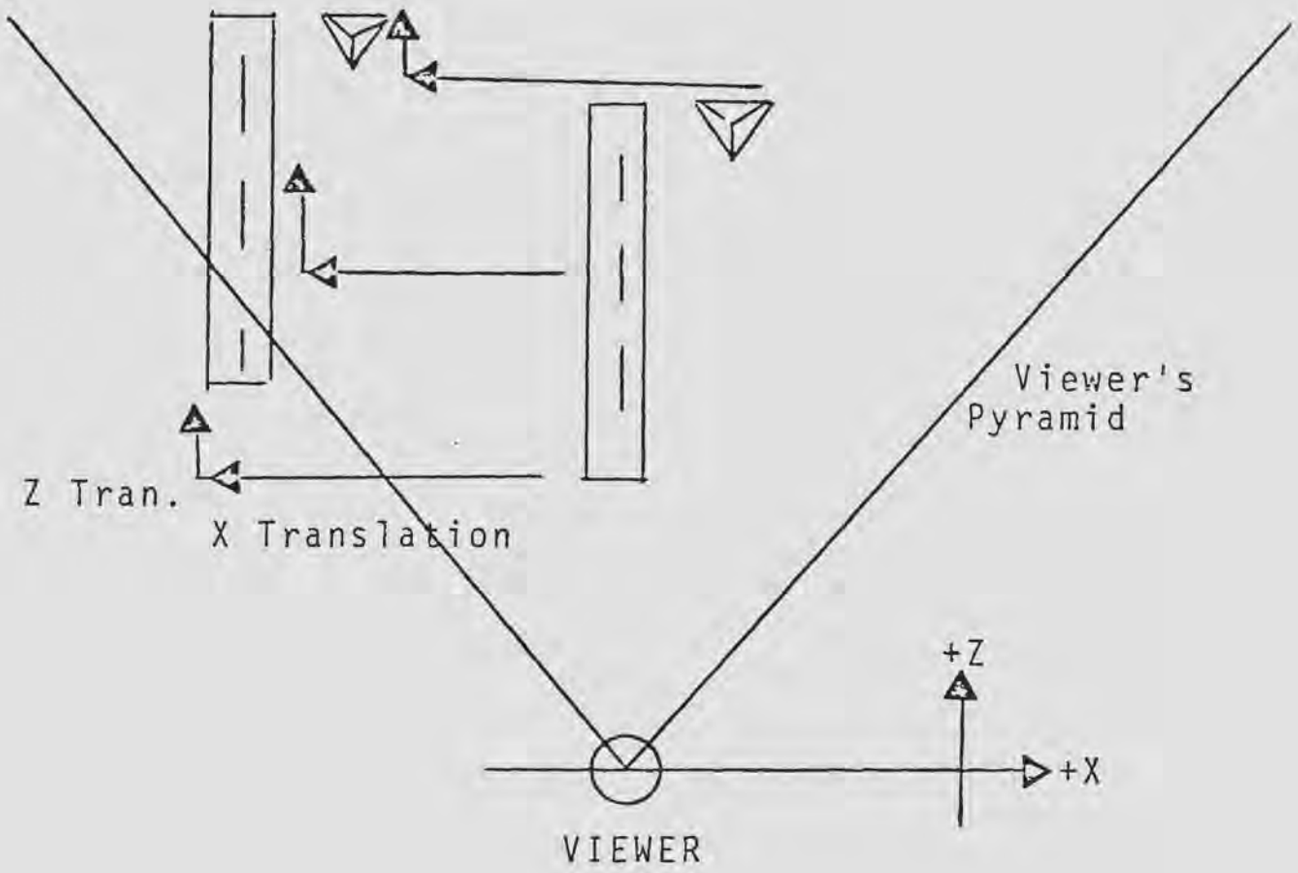


Figure 3. Point Translation of a Data Base

takes time (Sub Logic, 1976).

Each translated and rotated line of the object starting and ending points in the X, Y, and Z plane are assigned a four-bit code. This coded value will indicate which side of the viewing pyramid the data points lie. The viewing pyramid consists of four intersecting planes whose apex is at the observer's eye (Newman and Sproull, 1973). Figure 4 is an illustration of a line which requires some clipping in order to be within the boundary limits of the viewing screen.

The four-bit code is used to determine whether the line will be rejected or accepted as a line on the screen boundary. The four-bit code is:

first-bit: C0 = 1 = point to left of -X = Z plane.

second-bit: C1 = 1 = point to right of X = Z plane.

third-bit: C2 = 1 = point is above the Y = Z plane.

fourth-bit: C3 = 1 = point is below the -Y = Z plane.

If a data points code is all zeros then the data point lies on the boundary of the screen or within the viewing pyramid. Data points can have some ones in the code which may represent a line which intersects the screen but is off the screen. The lines starting and

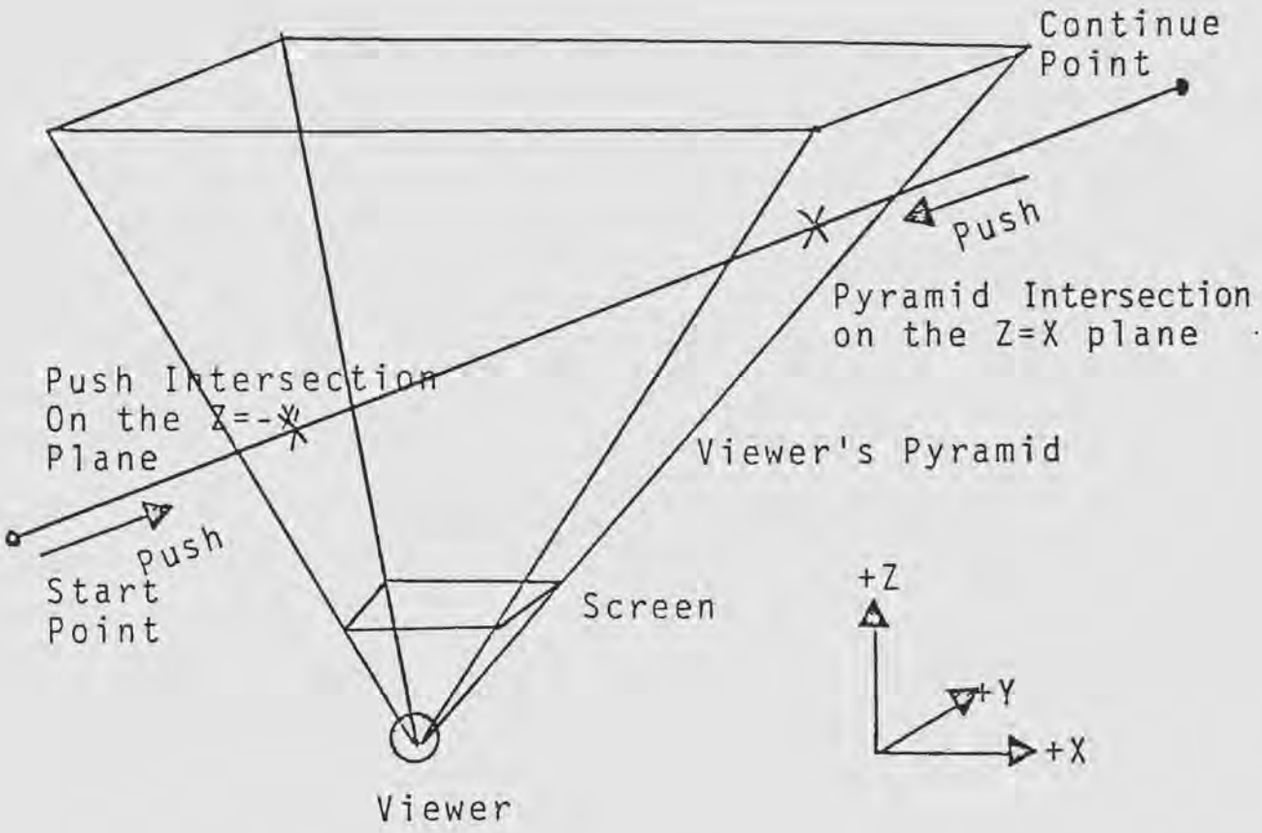


Figure 4. A Line in Need of Clipping

ending points are then coded and compared to check if the line is off the screen. If the start and end points are off the screen in the same direction, then the line is off the screen.

If the code is such that the lines starting and ending points are off the screen, but the points lie within the screen boundary then the push mathematics can be performed. If a lines starting point is to the left of the screen, its code will be 1000. If a lines ending point is to the right of the screen, its code will be 0100. If the two codes are "anded", then the code will be 0000 which means the line might be on the screen. The mathematics used to perform the right push are:

$$K = (Z(a) + X(a))/(X(a) - X(b) + Z(a))$$

$$X(a) = K^* (z(a) - 2(b)) - z(a)$$

$$Y(a) = K^* (y(b) - y(a)) + y(a)$$

$$Z(a) = -X(a)$$

The mathematics to perform a left push are:

$$K = (Z(a) - X(a))/(X(b) - X(a) - Z(b) + Z(a))$$

$$X(a) = K^* (Z(b) - Z(a)) + Z(a)$$

$$Y(a) = K^* (Y(b) - Y(a)) + Y(a)$$

$$Z(a) = X(a)$$

Once the push mathematics are completed, then the line is ready to be projected onto the screen. Figure 5 illustrates the use of push mathematics to obtain the line on the screen. Sometimes after one push, it

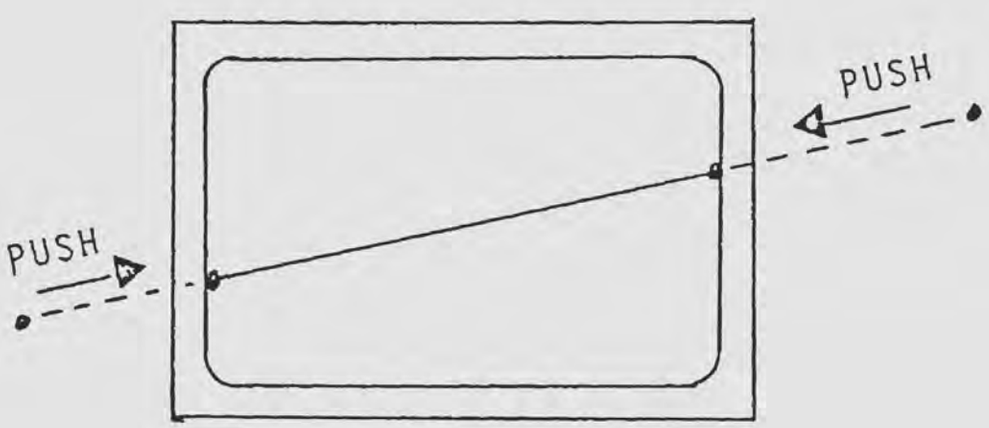


Figure 5. Push Mathematic Transformation

becomes apparent the line will not intersect the viewing pyramid after all and the line will be eliminated.

The operation of line clipping and coding in the program is an extensive process and thus slows down the output from the microprocessor to the video display terminal. That part of the program which performs the algorithm of coding and line clipping is written in BASIC language and thus adds additional computational time to the process. Even though the process is rather slow, it does not affect the accuracy of the transformed view of the field on the video output monitor.

Projection

Once the clipping process yields a line that is visible on the screen, then a perspective projection must be performed. Generating a true perspective image requires dividing by the depth of each point (Newman and Sproull, 1973). Dividing each starting and ending point of a line that falls within the viewing pyramid by the depth and then multiplying by the half width of the particular video output monitor screen that is being used will give the true perspective image. The equation for projection is:

$$X_S = \frac{X(I)}{Z(I)} * \frac{W}{2}$$

$$Y_S = \frac{Y(I)}{Z(I)} * \frac{W}{2}$$

The larger the value of Z will result in a smaller image projected on the video monitor. One major pitfall that must be avoided is trying to project images that lie at the base of the viewing pyramids (X=Y=Z=0). This will result in trying to divide by zero. A point at the base of the viewing pyramid is not definable because it implies a view of an infinitesimally small point from a distance of zero (at the viewer's eye) (Sub Logic, 1976). Care has been taken in the program to avoid division by zero. The program checks to see if Z is equal to 0. If Z does equal 0, then Z is set equal to .001 and the calculation for the projection of the image is completed.

Order of Transformation

A key factor in the projection of a point on the screen is the order in which the transformations are calculated. The position of a point on the display screen which corresponds to a point on some object will be different if different orders of translation and rotation are calculated. For example, if the viewer's location in space (translation) is considered before his viewing direction (rotation), a different projection would result if location had been considered after direction. Figure 6 shows these two orders of projection (Sub Logic, 1976).

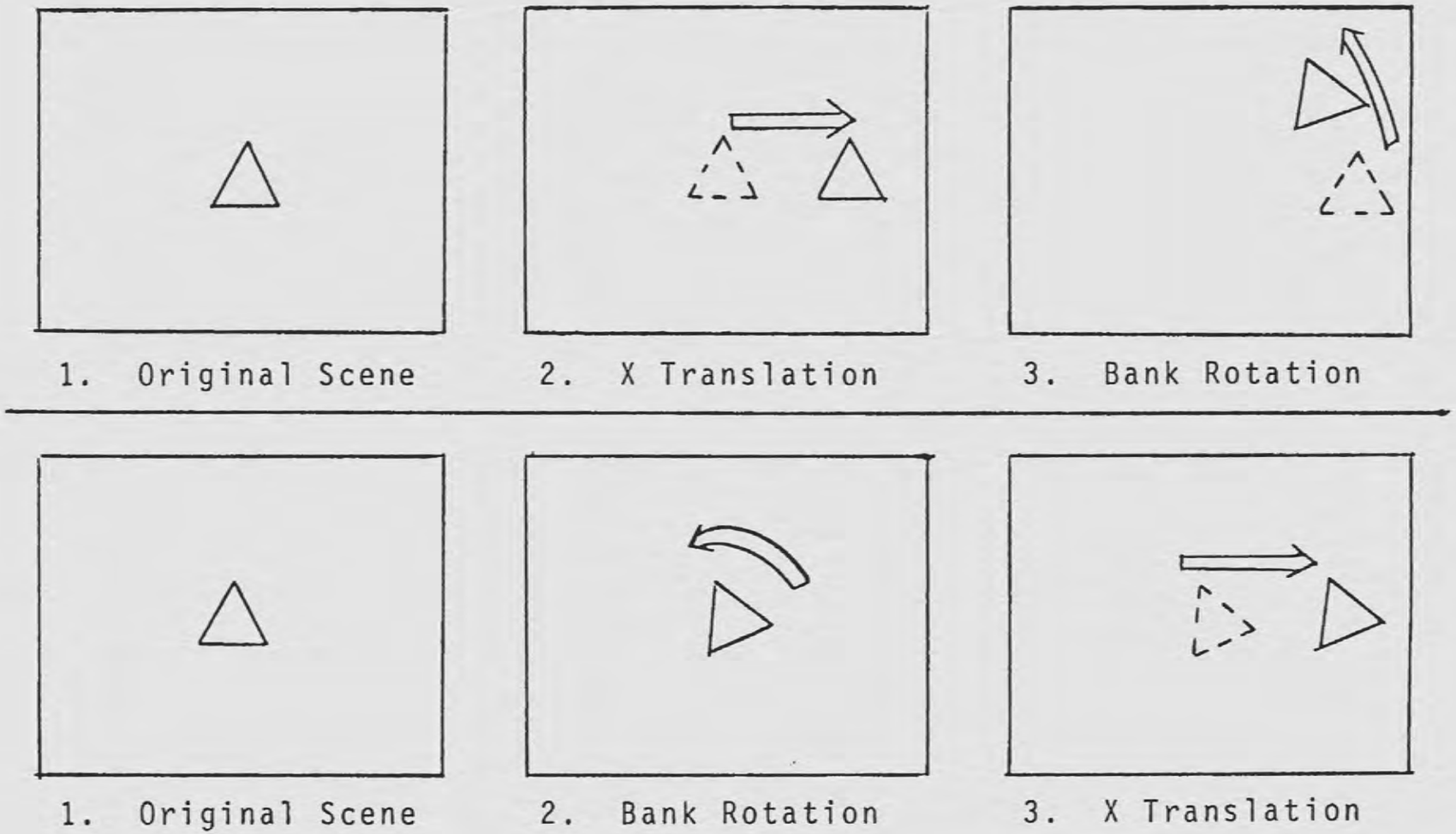


Figure 6. The Difference Transformation order makes

Throughout the program for flight simulation, the following is the order in which the transformations will be calculated:

1. X, Y, Z translation.
2. Heading (rotation about Y axis).
3. Pitch (angle of view to the X, Z plane).
4. Bank

One can generate many images of the same scene depending on the position from which the scene is observed. The sense of direction for each of the transforms will give the viewer a viewing direction and aperture that are dependent on one another. Figure 7 shows the sense of direction of each of the transforms. A positive change in X will cause an object to move to the right, if the viewer is at a 0 degree bank. If the viewer is in a 90 degree bank; however, the triangle will appear to move upward instead (Sub Logic, 1976).

The application of computer graphics to flight simulation is on the increase. A major advantage of numeric modelling through the use of computer graphics is its adaptability to new and changing requirements. Dramatic advances in Computer Image Generation (C.I.G.) and Digital Radar Landmass Simulation (D.R.L.M.S.) Technologies, for example, have spurred the use of visual simulation by the military and commercial communities alike to achieve

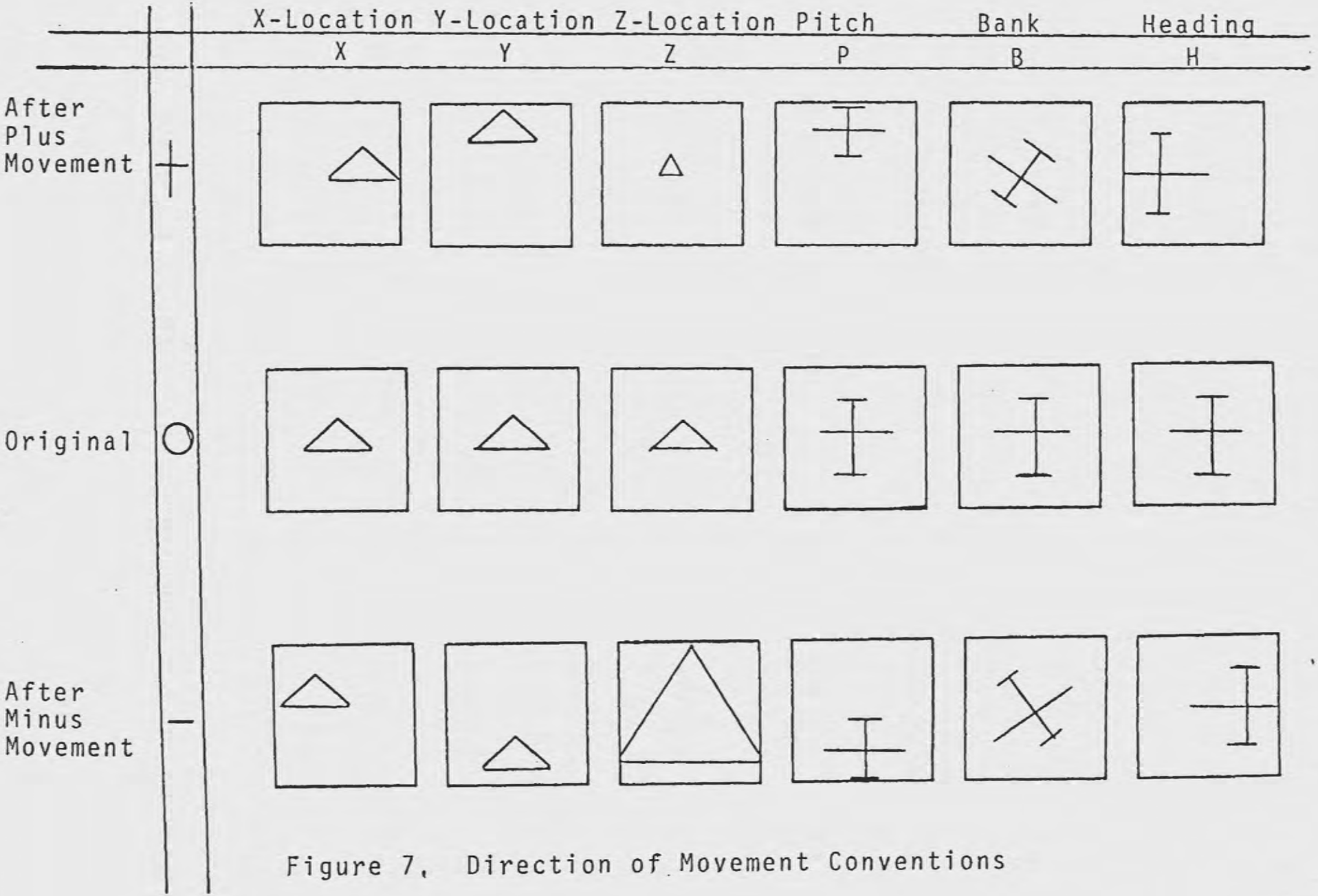


Figure 7. Direction of Movement Conventions

training effectiveness, efficiency, and cost objectives (General Electric, 1978).

An article appeared in the Arizona Republic concerning the flight simulator at Williams Air Force Base. Within a two-story block building, pilots are seated at the controls of a Cessna T37B cockpit. They bank, climb, dive, accelerate, take off and land at any base in the world (Law, 1975). According to Major Brian Winters, acting division chief, "We have built one super simulator." "It has all the capabilities that the future simulators are likely to have."

In April, 1976, Approach, the Naval Aviation Safety Review, discussed some of the Navy's experience with the General Electric computer generated image system. The device was the first of a new generation of simulators employing a computer-generated visual display which enabled the TA-4 2F-90 flight simulator to be expanded for use in the VFR flight regime. Additional advantages of the visual display included:

1. Presentation of any one of a number of visual scenes.
2. Accurate meatball simulation.
3. Bomb scoring.
4. Visual catapult launches.

The most significant aspect of simulator function is that the student actually controls the visual scene

through the cockpit controls rather than merely reacting to a preprogrammed visual stimulus presented to him (Evans, 1976).

The use of microprocessors for flight simulation is on the horizon. Hopefully, this study will show the validity of microprocessors for flight simulation and expand the interest for microprocessor applications in other areas of simulation of dynamic systems.

CHAPTER II

SIMULATOR CONTROL PARAMETERS

The intended purpose of any flight simulator is to accurately generate the perspective views of the data base for a given position. The data base selected for this study is that of Herndon Airport located in Orlando, Florida. Herndon was chosen for this study due to the complexity of its layout. As viewed in Figure 8, Herndon has two major runways which can be landed on or taken off depending on the approach. Runway 7 is used mostly for landing while 31 is most popularly used for takeoffs.

The program was designed such that any data base, which represents different airports, can be used for flight simulator training. The only problem that will be encountered when using other data bases is in the total number of data points. The maximum number of data points that can be used in this study is 255 points.

Hardware

The system used in this study was built by the Southwest Technical Products Corporation. The major components of the system are:

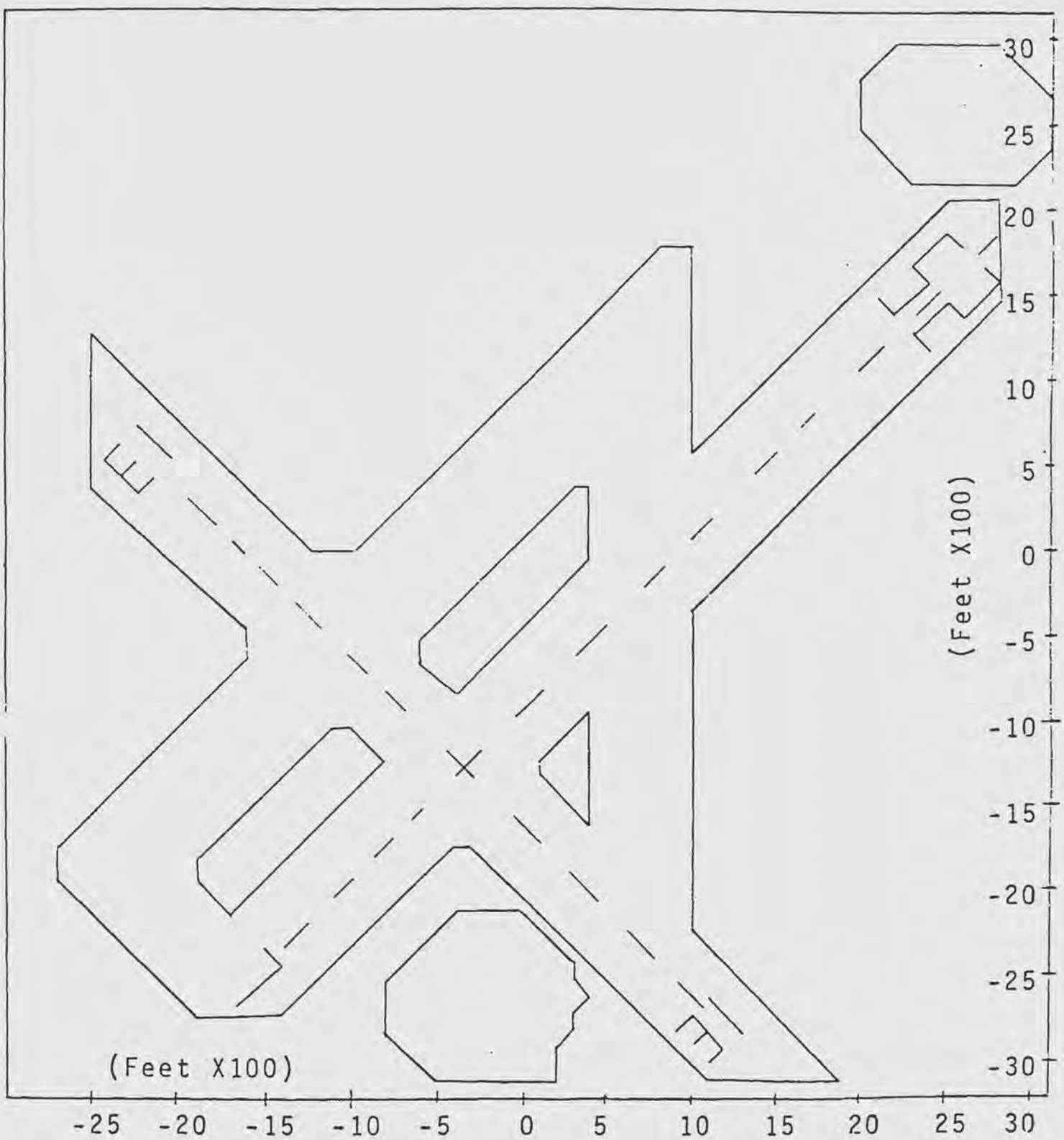


Figure 8.
General Layout of Herndon Airport and Associated Runways

1. 6800 micro computer.
2. Dual mini floppy disk.
3. Graphic Interface Terminal.
4. Video output monitor.
5. Dummy Terminal

The 6800 microcomputer is a 32 K-bytes machine which uses a hexadecimal number system. The unit has a basic interpreter which allows the user to write in the BASIC language.

The dual mini-floppy disk has the ability to store programs on small magnetic disks. This puts into a compact form the programs and data bases for future use. The magnetic disk affords the ability to store many different data bases which are readily available to be called up by the operator. This gives the operator the ability to change the training environment from one area to another within seconds.

The graphic interface terminal is the buffer between the video output monitor and the 6800 microcomputer. The graphic interface takes the output data from the microcomputer and converts it to an electric field to generate a finely focused, high speed beam of electrons, and to deflect the beam to various parts of the screen surface so as to generate a visible trace (Newman-Sproull, 1973)

The heart of the computer graphic system is the

video output monitor. The monitor makes use of the Cathode Ray Tube (C.R.T.) to provide pictorial or visual information from the electrical signals generated by the graphic interface terminal.

The C.R.T. used in this study is a conventional television monitor. The vacuum tube type C.R.T. uses a phosphor with long persistence. Once a line has been displayed, it will remain unless erased by an electrical signal which erases the entire screen. This erasing technique restricts the display of dynamic movement of the picture components as it requires the complete redrawing of the display when any modifications to the picture are to be made (Rogers and Adams, 1976). The erasing process is also distracting in that a flash is generated on the face of the C.R.T. (McCormick, 1970). Since no refreshing of the tube is necessary, the picture will remain flicker free. Elimination of refreshing hardware along with the use of conventional storage techniques for the picture data allows the storage tube C.R.T. graphics terminal to be of relatively low cost when compared to other systems (Rogers and Adams, 1976).

The dummy terminal is the keyboard that is used to input data and receive responses from the micro computer. The keyboard is similar in nature to the keyboard of any typewriter. The dummy terminal is the communications

link between the operator and the processor.

Software

The basic idea of the software is to sequentially perform the operations in Figure 9. The program will in BASIC language perform the following sequential operations:

1. Get the screen width and field of view.
2. Get the viewer's position and direction of view.
3. Call the transformation matrix generator subroutine.
4. Feed an array of 3D input points to the 3D to 2D converter, one at a time.
5. Send the resulting screen start and end points to the display device to be displayed (Sub Logic, 1976).

The following is a listing of the eight control parameters that will be inputted by the operator. Also, included in the list is the program's feedback parameter:

1. X(3)--Integer point variable that represents the viewer's position on the X-axis.
2. Y(3)--Integer point variable that represents the viewer's altitude.
3. Z(3)--Integer point variable that represents the viewer's position on the Z axis.
4. P--floating point variable specifying the angle of inclination from which the viewer looks at the scene.

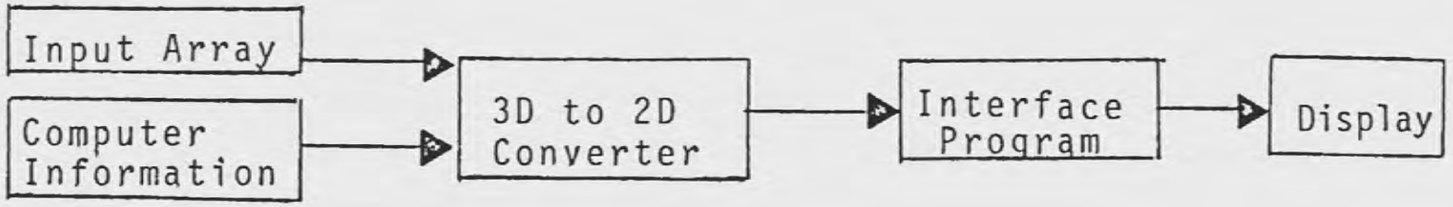


Figure 9, The 3D to 2D Converter Program and Its Arrays

5. B--floating point variable specifying the angle at which a viewer's head is tilted sideways when viewing the scene.

6. H--floating point variable specifying the direction the viewer is facing while standing on the XZ plane.

7. V--floating point variable representing the tangent of the half field of view.

8. W--Integer value representing half the screen width minus one. This value affects the scaling of the final output screen points.

P2-Feedback Parameter-This parameter is not submitted by the user. It is generated by the program. P2 is set to 1 if the 3D line just processed by the 3D to 2D converter is visible and on the screen. P2 is cleared to zero if the line is off the screen (Sub Logic, 1976).

Each of the eight parameters are inputted by the operator at the dummy terminal. After each point in the base case has been submitted to the 3D to 2D converter subroutine, the program will then ask for new parameters before the next output is calculated. Figure 10 is a simplified flow chart for the display simulator program.

Once the input array and control parameters have been integrated into the body of the program, the following subroutines as shown in Figure 11 are called. These subroutines are called upon to transform each starting and ending point of the base case and send the

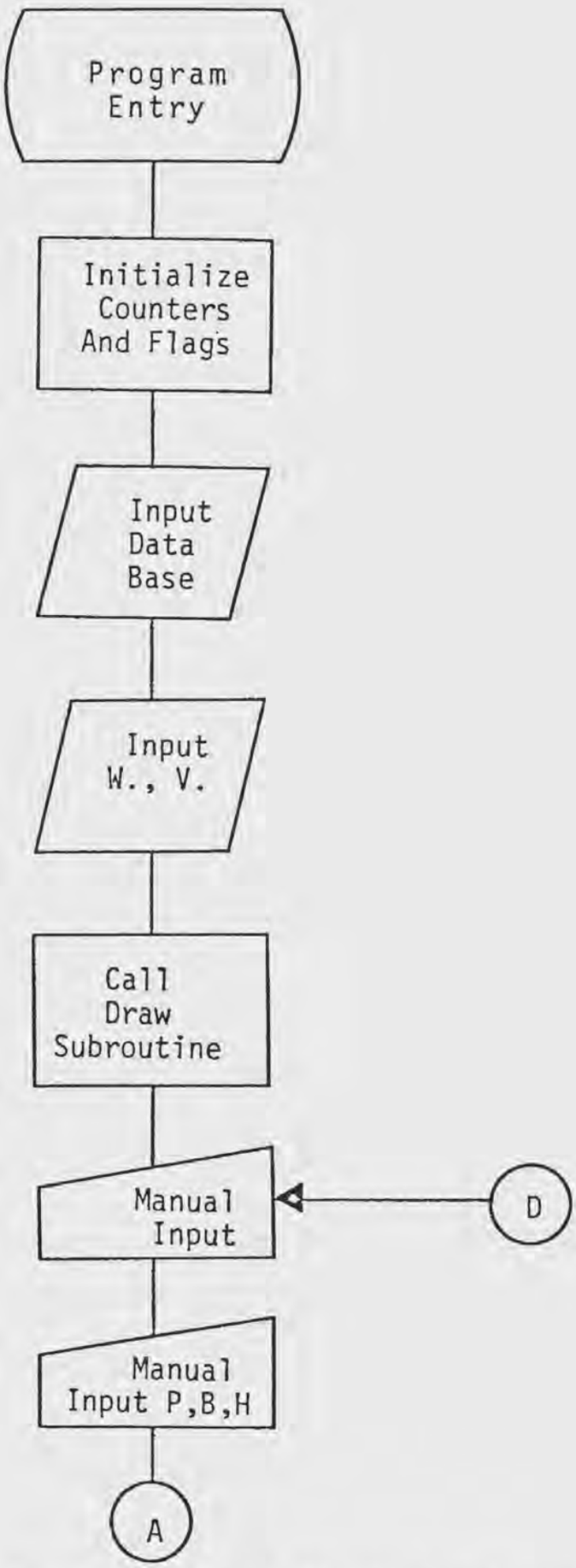


Figure 10. Flow Chart for Flight Simulation Program

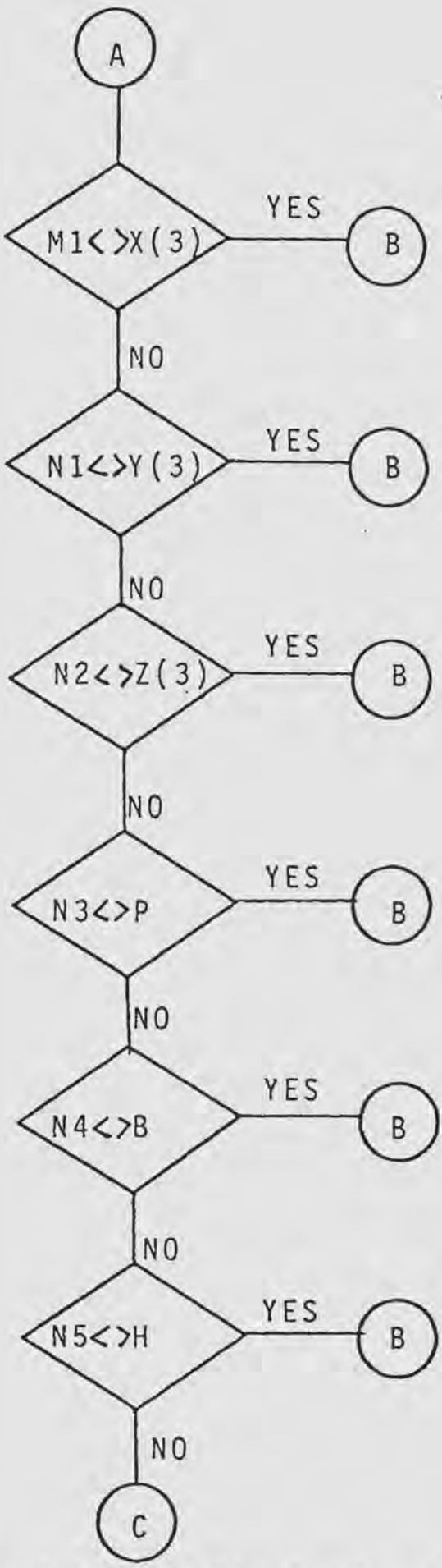


Figure 10. Flow Chart of Flight Simulation Program (continued)

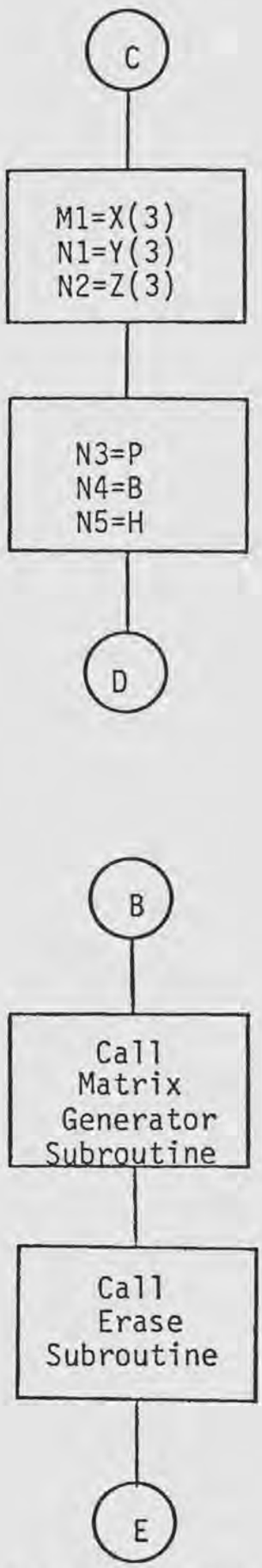


Figure 10. Flow Chart for Flight Simulation Program (continued)

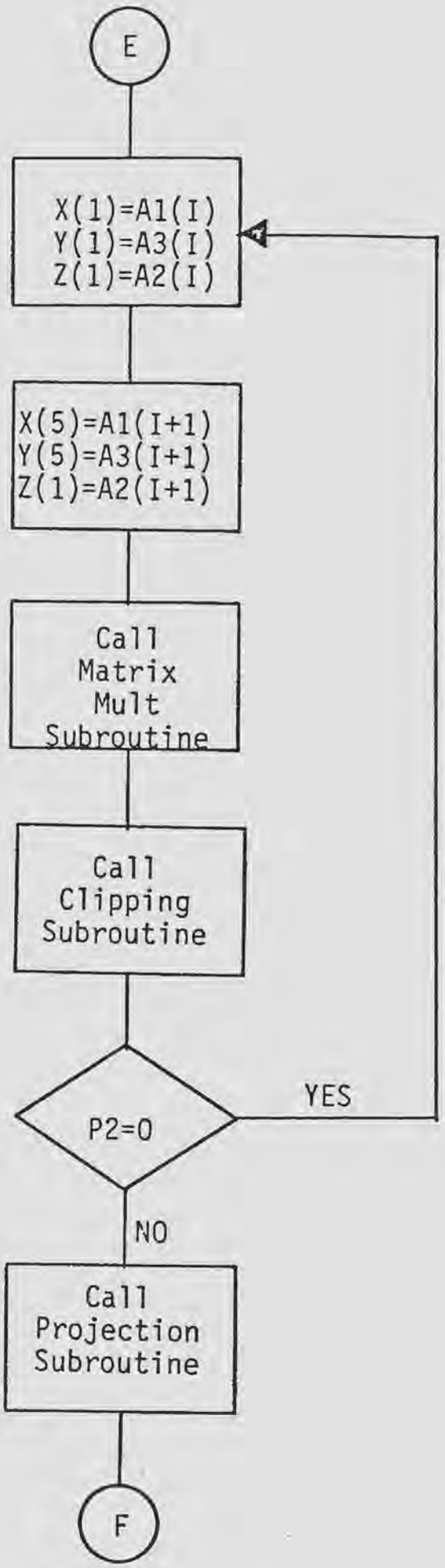


Figure 10. Flow Chart for Flight Simulation Program (continued)

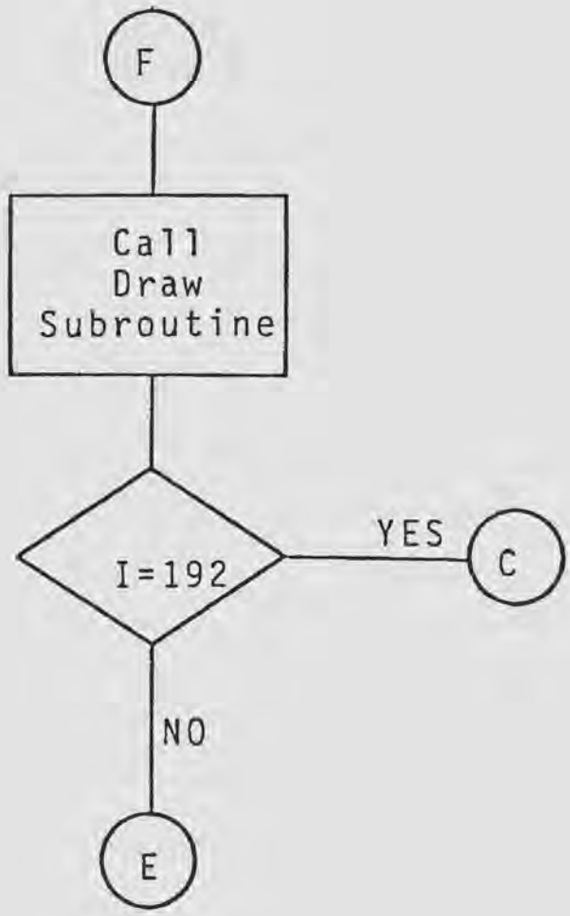


Figure 10. Flow Chart for Flight Simulation Program (continued)

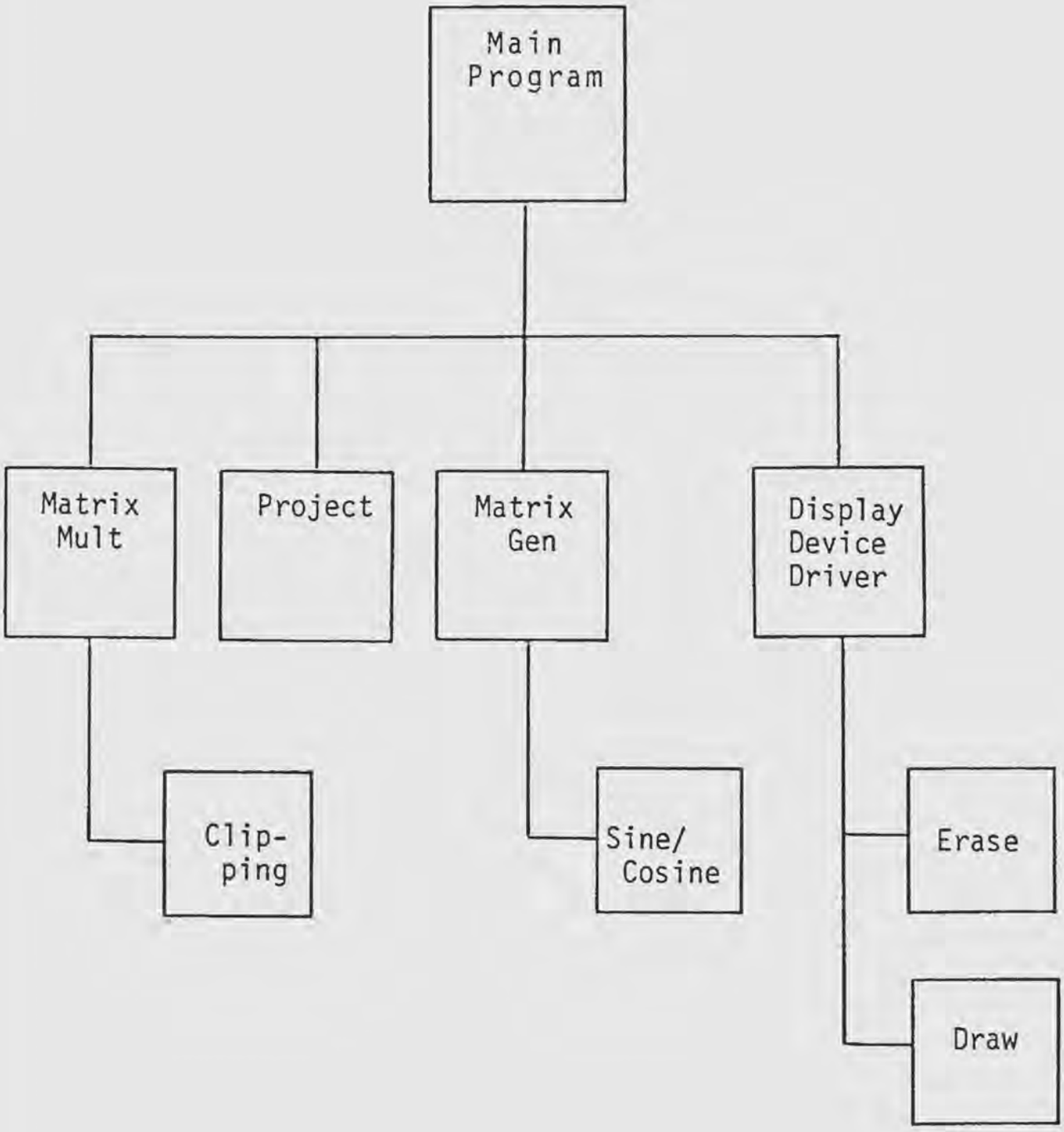


Figure 11. Subroutine Structure for Flight Simulator

resulting screen start and end points to the display device to be displayed.

The matrix generator subroutine generates the 3 x 3 transformation matrix required for rotation (Sub Logic, 1976). The input parameters P, B, H, and V are used to generate the rotational transformation that will be used in the 3D to 2D converter. Contained within the matrix generator subroutine is the sine/cosine subroutine. This subroutine generates the cosine and sine of a value in degrees. This subroutine is needed because BASIC does not have trigonometric functions. This matrix need only be calculated once for the given control parameters. All lines in the scene will use the same transformation matrix.

The program must then feed the array of 3D input points to the matrix multiplier subroutine. The matrix multiplier subroutine takes the viewer's location values and adds them to the start and end points of each line in the data base one line at a time. The points are then multiplied one at a time by the transformation matrix calculated in the matrix generator subroutine. Each rotated start and end point is then transferred to the clipping subroutine.

The clipping subroutine determines if the line just translated and rotated is on the screen or off the screen

and accordingly displays, clips, and displays, or eliminates it. This is the part of the program that slows down the process due to the push mathematics that must be performed.

The clipping subroutine first generates a code for the start and end points based on where they are in relation to the viewing pyramid (Sub Logic, 1976). From these codes, the decision to clip, project or eliminate is made. If a line is found to be within the screen boundary, the program shifts the line to that part of the subroutine that performs the appropriate "push" mathematics. It is within this subroutine that the projection code, P2, is set to one if line is on the screen and cleared to zero if it is not.

As each starting and ending points for a line is translated, rotated, and had the necessary clipping performed, it must be outputted to the display device driver subroutine. The display device driver subroutine was written in machine language and contains several subroutines within the program. The two subroutines used by the writer was the erase and draw subroutines.

The erase subroutine is ran once each time a new transformation matrix has been calculated. The subroutine will in essence clear the video monitor display of any previous objects. A listing of the program can be found in Appendix B.

The draw subroutine accepts the data from the 3D to 2D converter and outputs to the video display monitor electrical impulses that correspond to lines on the C.R.T. Each time the 3D to 2D converter calculates a new translated, rotated, and clipped line, the draw subroutine is called to output to the video monitor. A listing of the program can be found in Appendix B.

Once all the lines from the base data has been translated, rotated, clipped, or eliminated, and been outputted to the video monitor, the program will return to ask for a new position and direction. After the operator has entered these parameters, the program will check to see if the input variables had changed. If none of the variables were changed, the program will not erase the screen, but instead ask again for new viewer parameters. Once any one of the six parameters are changed, the program will then calculate a new transformation matrix, erase the screen, and calculate new translated, rotated, clip or eliminate lines. The program then outputs the lines to the video monitor. A complete listing of the program can be found in Appendix A.

The program will run continuously until no further simulations are required. The program can be shut down and the magnetic disk removed from the mini floppy disk reader by hitting the reset button on the microcomputer

front panel.

This program was written with the versatility to include many different data bases, larger or smaller screens, wider or narrower fields of view, and different positions and directions. The major portion of the program is written in BASIC, which makes the calculations of perspectives slow, but easy to change and update for future extensions.

CHAPTER III

SYSTEM OPERATION

The most crucial part of constructing a three-dimensional graphic program is testing the system to see if results are what is expected. The entire process of rotation, translation, clipping, perspectives, and visual display must be integrated into a software module or modules that have coordinate values of a point in the three-dimensional space.

This module can be depicted as a transfer system as shown in Figure 12 (Giloi, 1978). The input consists of points X, Y, Z, in the three-dimensional space. Along with the inputs are eight transformation parameters.

- The three angles of rotation P, B, and H.
- The three translation parameters X(3), Y(3), and Z(3).
- The two scaling factors W, V.

The output variables that one is expecting to display are X_p and Y_p.

The simulated landings at Herndon Airport must include the input parameters, transformation parameters, and the output variables to be displayed on the video

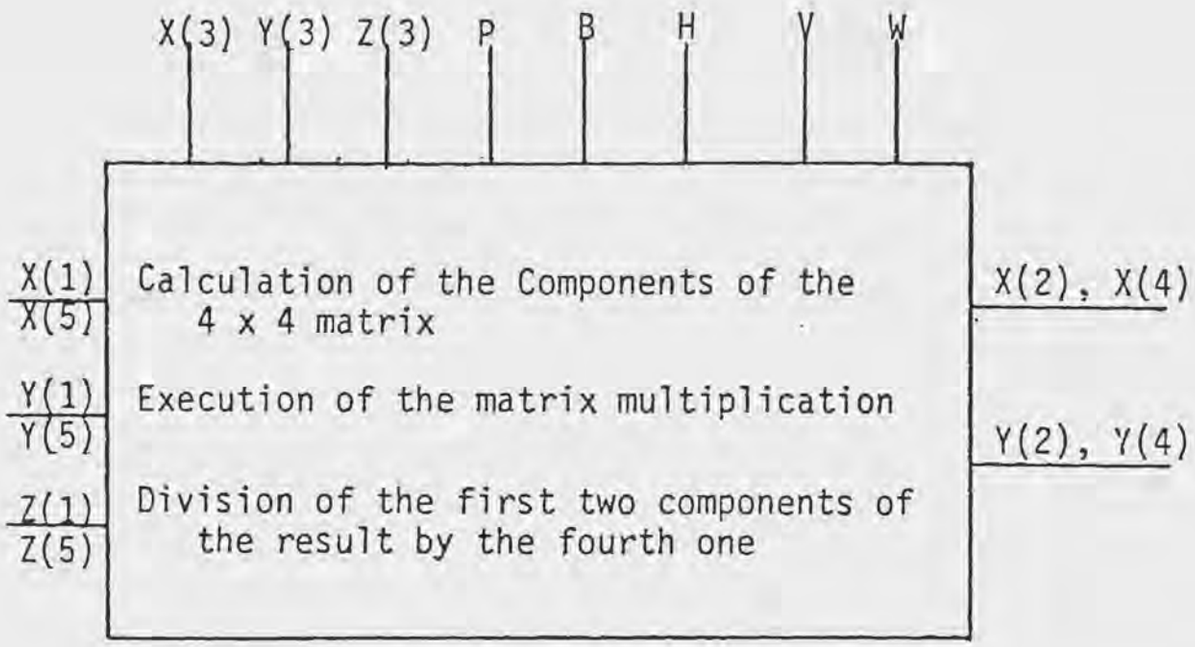


Figure 12. Transfer System view of a module for rotation, translation, clipping, and perspective.

monitor. Table 1 is a listing of the input data points for the layout of Herndon Airport. The 3D to 2D converter subroutine requires a screen width parameter to enable the generation of user device compatible coordinates (Sub Logic, 1976). Regardless of the screen width, the screen origin (Screen Coordinate X=0, Y=0) is assumed to be at the screen center. For this purpose, base case data will have points which are negative in value. A negative X only means the viewer's position is to the left of the center of the screen on the X axis. A negative Z value means that the viewer's position is out of the screen since positive Z is into the screen.

In order to simulate flight landings on the runways at Herndon Airport, the three angles of rotation and the three translation parameters must be inputted to the program. Figure 31 depicts various points on and off the runway that are used to simulate flight landings. Table 2 is a listing of the various points on the runway layout that have been inputted to the computer and the translated and rotated picture displayed on the video monitor.

In order to simulate an approach landing on the runways at Herndon, a general view of the field should be generated. Figure 14 is a view of Herndon Airport when the values of the angles of rotation and translation parameters are:

TABLE 1

Three-Dimensional Data Points For Base Case
(Hundreds of Feet)

X	Y	Z	X	Y	Z
-5	0	-31	22	0	30
2	0	-31	22	0	30
2	0	-31	20	0	28
2	0	-29	20	0	28
2	0	-29	20	0	25
3	0	-28	20	0	25
3	0	-28	23	0	22
3	0	-27	19	0	-31
3	0	-27	10	0	-22
4	0	-26	10	0	-22
4	0	-26	10	0	-3
3	0	-25	10	0	-3
3	0	-25	28	0	15
3	0	-24	28	0	15
3	0	-24	28	0	21
0	0	-21	28	0	21
0	0	-21	25	0	21
-4	0	-21	25	0	21
-4	0	-21	10	0	7
-8	0	-25	10	0	7
-8	0	-25	10	0	18
-8	0	-28	10	0	18
-8	0	-28	8	0	18
-5	0	-31	8	0	18
23	0	22	-10	0	0
29	0	22	-10	0	0
29	0	22	-12	0	0
31	0	24	-12	0	0
31	0	24	-25	0	13
31	0	27	-25	0	13
31	0	27	-25	0	4
28	0	30	-25	0	4
28	0	30	-25	0	-4
-16	0	-4	-10	0	-10
-16	0	-6	-11	0	-10
-16	0	-6	-11	0	-10
-27	0	-17	-19	0	-18
-27	0	-17	-19	0	-18
-27	0	-19	-19	0	-19

Table 1. Three-Dimensional Data Points For Base Case (continued)

X	Y	Z	X	Y	Z
-27	0	-19	-19	0	-19
-19	0	-27	-17	0	-21
-19	0	-27	-17	0	-21
-14	0	-27	-8	0	-12
-14	0	-27	4	0	-16
-4	0	-17	4	0	-9
-4	0	-17	4	0	-9
-3	0	-17	1	0	-12
-3	0	-17	1	0	-12
11	0	-31	1	0	-13
11	0	-31	1	0	-13
19	0	-31	4	0	-16
4	0	0	-15	0	-23
4	0	4	-14	0	-24
4	0	4	-14	0	-24
3	0	4	-17	0	-27
3	0	4	25	0	13
-6	0	-5	24	0	24
-6	0	-5	24	0	14
-6	0	-6	25	0	15
-6	0	-6	25	0	15
-4	0	-8	26	0	14
-4	0	-8	26	0	14
4	0	0	27	0	15
-8	0	-12	27	0	15
-10	0	-10	26	0	16
22	0	16	-22	0	4
23	0	15	-21	0	5
23	0	15	-23	0	5
24	0	16	-22	0	6
24	0	16	9	0	-28
23	0	17	10	0	-27
23	0	17	10	0	-27
24	0	18	12	0	-29
24	0	18	12	0	-29
25	0	17	11	0	-30
-13	0	-22	11	0	-28
-11	0	-20	10	0	-29
-9	0	-18	11	0	-26
-7	0	-16	13	0	-28
-4	0	-13	-19	0	3
-3	0	-12	-17	0	1
0	0	-9	-15	0	-1
2	0	-7	-13	0	-3
5	0	-4	-11	0	-5

Table 1. Three-Dimensional Data Points for Base Case (continued)

X	Y	Z	X	Y	Z
7	0	-2	-9	0	-7
11	0	2	-4	0	-12
13	0	4	-3	0	-13
17	0	8	-2	0	-14
19	0	10	0	0	-16
21	0	12	2	0	-18
23	0	14	4	0	-20
25	0	16	8	0	-24
27	0	18	10	0	-26
-23	0	7	13	0	-29
-24	0	6	15	0	-31
-24	0	6			
-22	0	4			

TABLE 2

Three-Dimensional Data Points for Various Landing Approaches to Herndon Airport

	(Feet)			(Degrees)		
No.	X	X	Z	P	B	H
1	-6000	600	-6500	-45	0	40
2	-3500	400	-3700	-45	0	40
3	-2000	200	-2800	-35	0	40
4	-1500	100	-2400	-30	0	40
5	-1000	100	-1900	-30	0	225
6	1400	200	-3000	-15	0	140
7	1600	200	-3600	-35	0	145
8	2500	500	-3900	-35	0	145
9	-5000	800	2600	-40	0	-45
10	-3500	300	1500	-40	0	-45
11	-2600	200	900	-35	0	-45
12	-2300	200	700	-10	0	-45
13	-1000	300	-500	-35	0	135
14	4200	800	3000	-35	0	225
15	3500	600	2300	-35	0	225
16	3000	400	2000	-40	0	225
17	2700	200	1800	-30	0	225

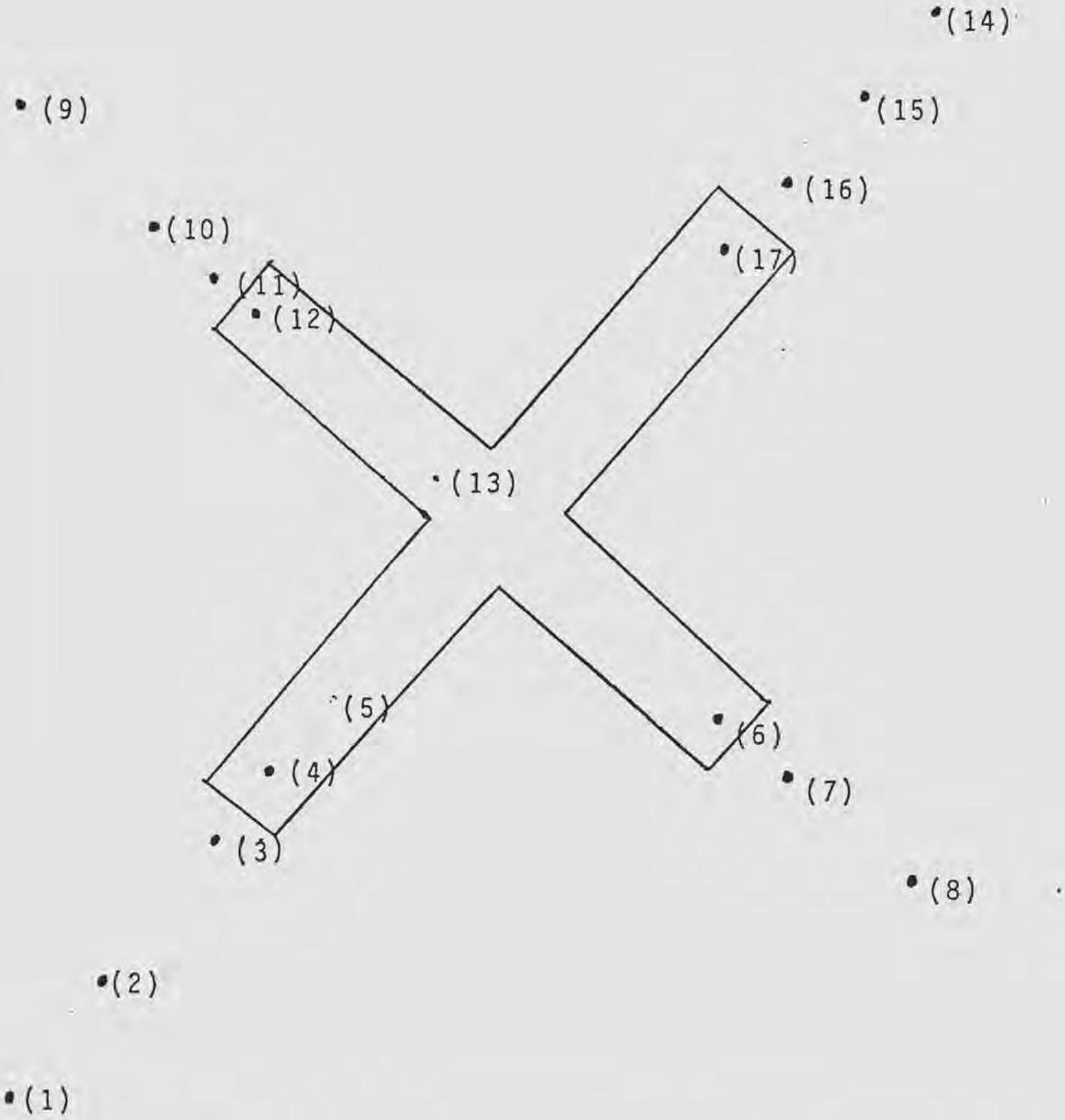


Figure 13. Point Approaches to Herndon Airport

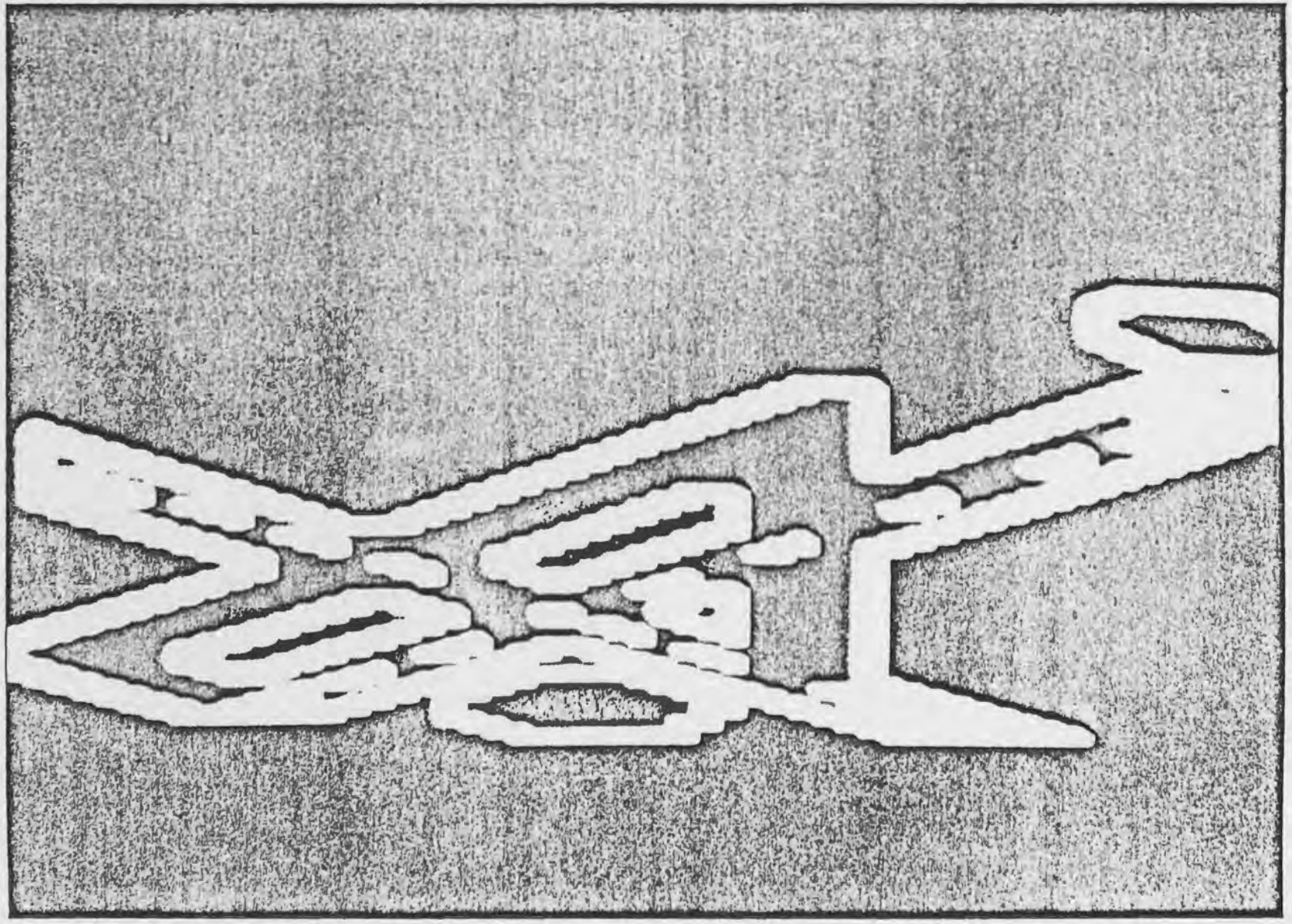


Figure 14. Aerial View of Herndon Airport at 5000 Feet

X(3) = 0 (origin of X-axis)

Y(3) = 5000 feet (altitude)

Z(3) = 0 (origin at Z-axis)

P = -90°

B = 0°

H = 0°

Approach landing to Runway 7 will be depicted in a series of figures. Figure 15 is a view of Herndon Airport when the values of the angles of rotation and translation parameters are:

X(3) = -10,000 feet

Y(3) = 3,000 feet (altitude)

Z(3) = 10,000 feet

P = -15 degrees

B = 0 degrees

H = 50 degrees

Figure 16 is a view of the airport from:

X(3) = -5,000 feet

Y(3) = 1,500 feet

Z(3) = -5,000 feet

P = -20 degrees

B = 0 degrees

H = 50 degrees

Figure 17 depicts an even closer view of Herndon's Runway 7. The parameters are:

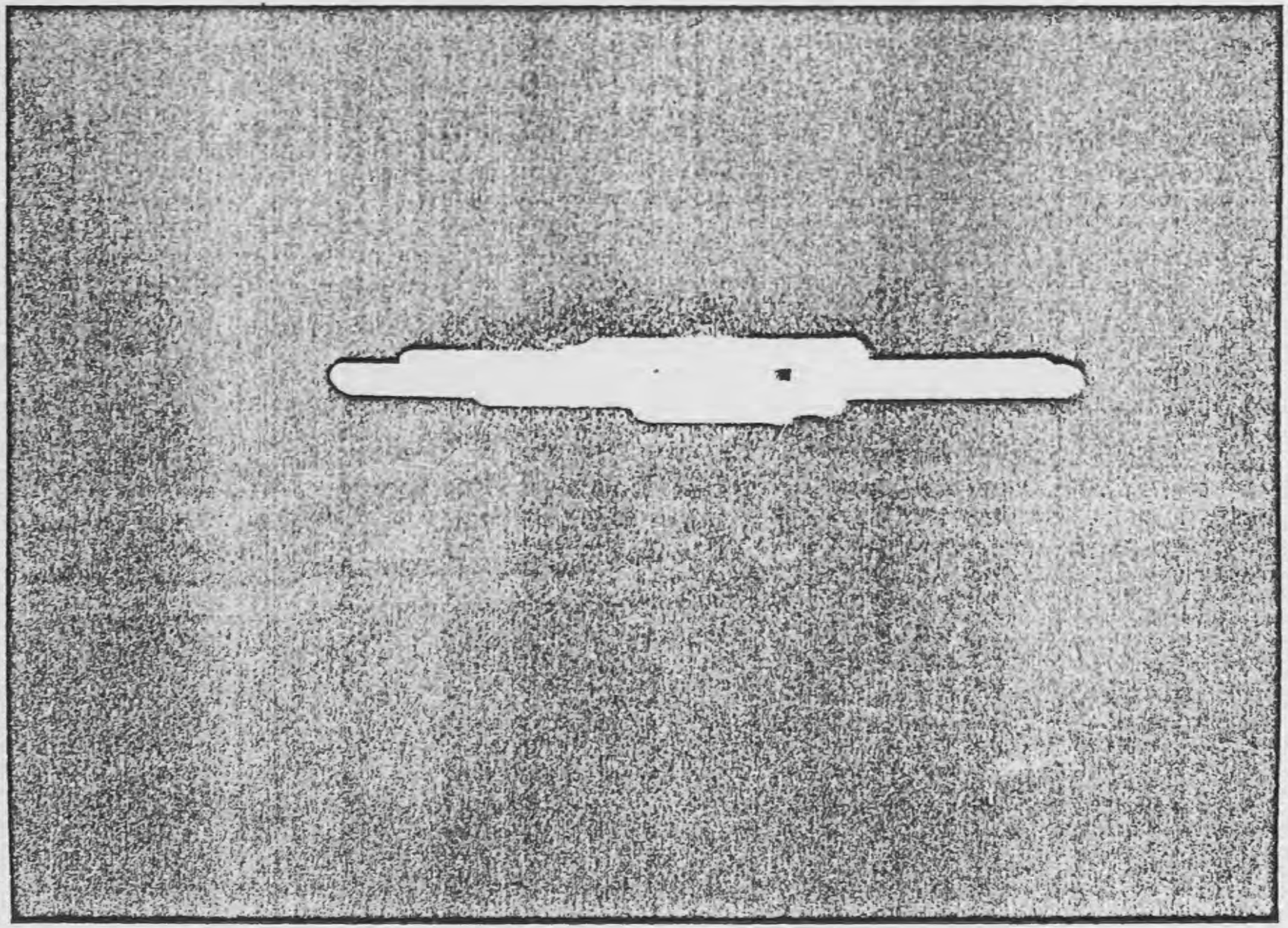


Figure 15. Aerial Approach to Runway 7 at 3000 Feet Altitude

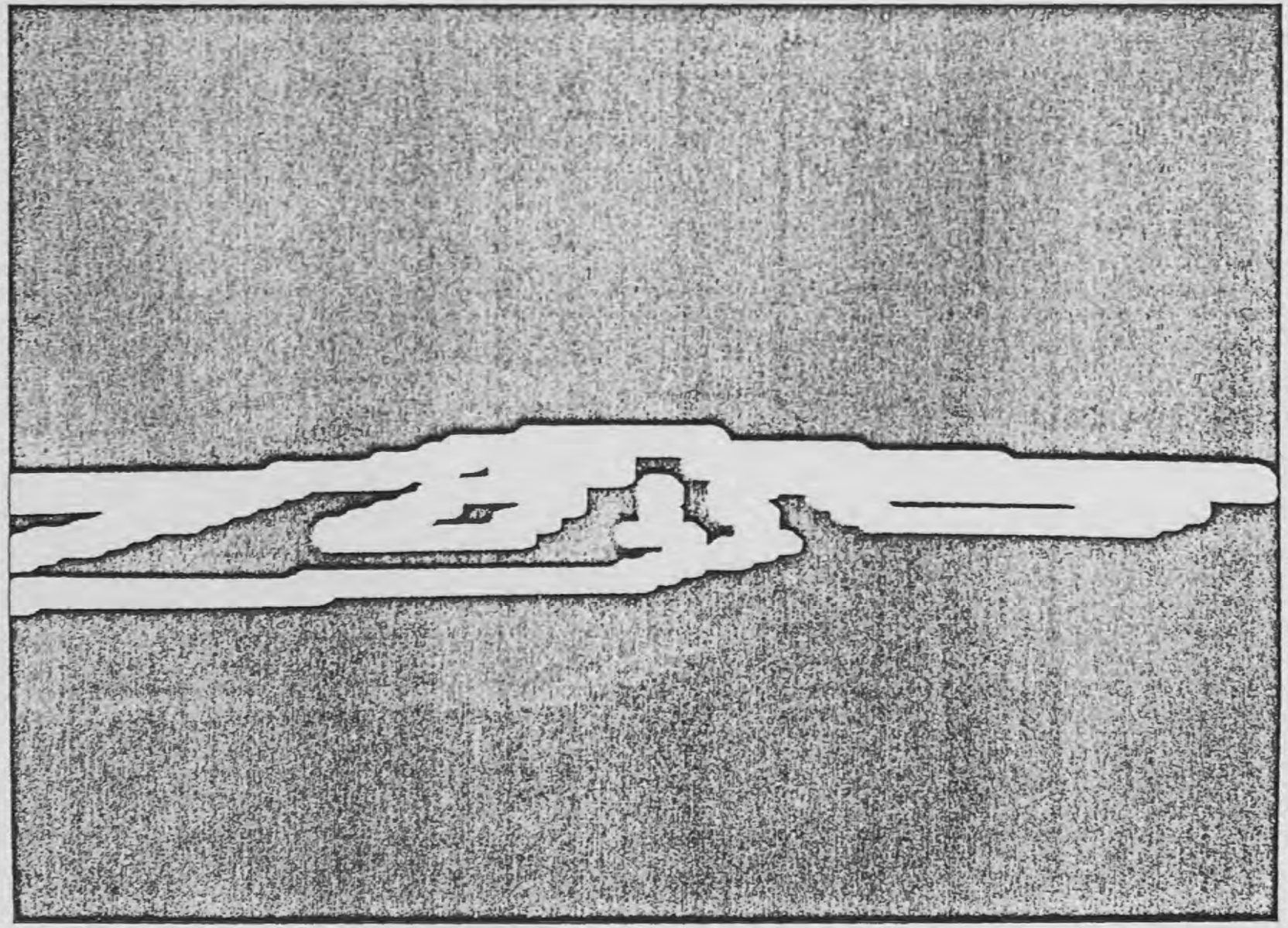


Figure 16. Aerial Approach to Runway 7 at 1500 Feet Altitude

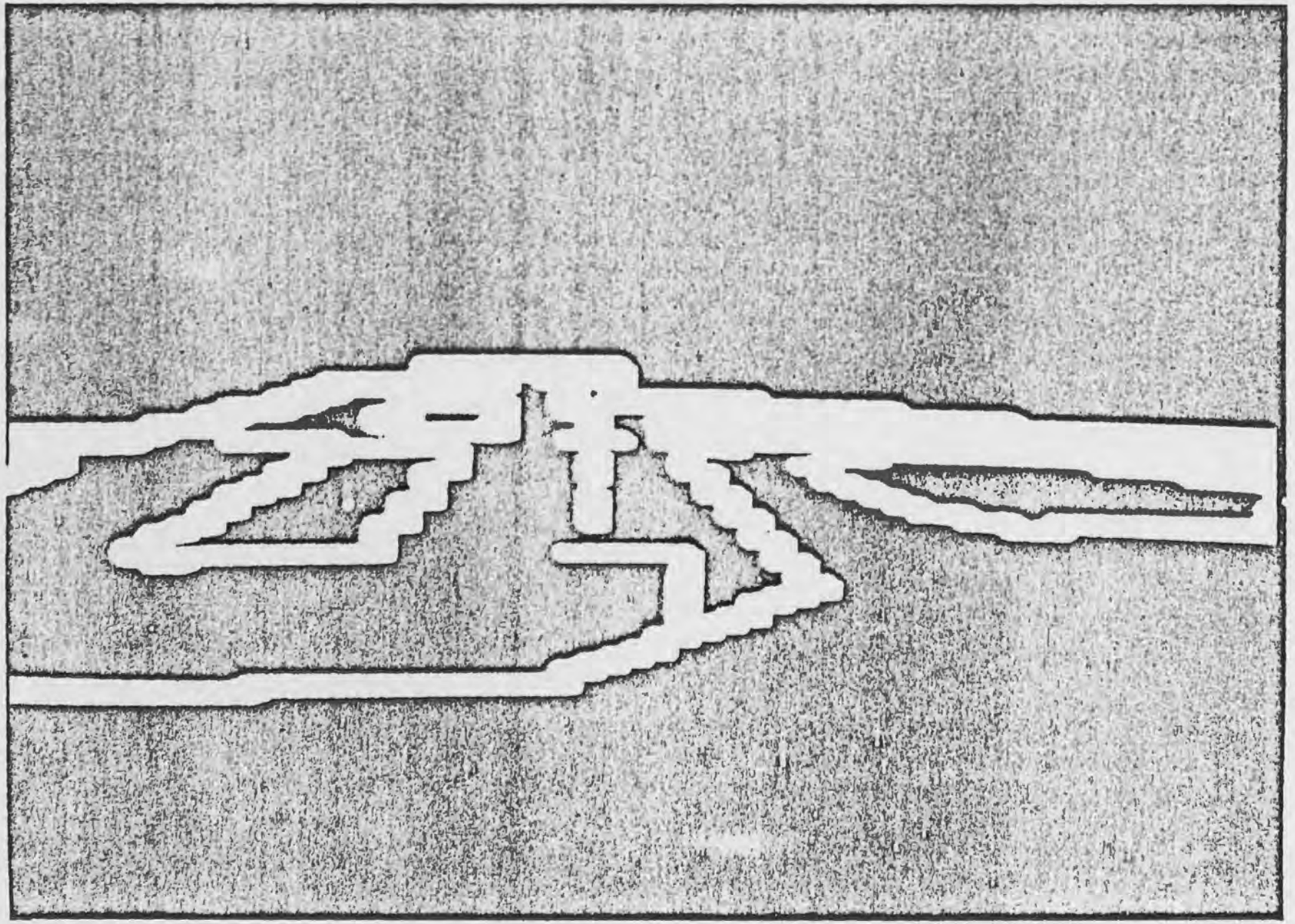


Figure 17. Aerial Approach to Runway 7 at 800 Feet Altitude

X(3) = -2,500 feet

Y(3) = 800 feet

Z(3) = -3,500 feet

P = -25 degrees

B = 0 degrees

H = 50 degrees

Examination of Figure 18 shows that the plane is about to touch down on Runway 7. The parameters are:

X(3) = -1,700 feet

Y(3) = 200 feet

Z(3) = 2,700 feet

P = -25 degrees

B = 0 degrees

H = 50 degrees

The final approach and touchdown on Runway 7 is depicted in Figure 19. The parameters are:

X(3) = -1,500 feet

Y(3) = 100 feet

Z(3) = -2,400 feet

P = -15 degrees

B = 0 degrees

H = 50 degrees

Approaches to Runway 25 are depicted by Figures 20, 21, and 22. Figure 20's input parameters are:

X(3) = 3,500 feet

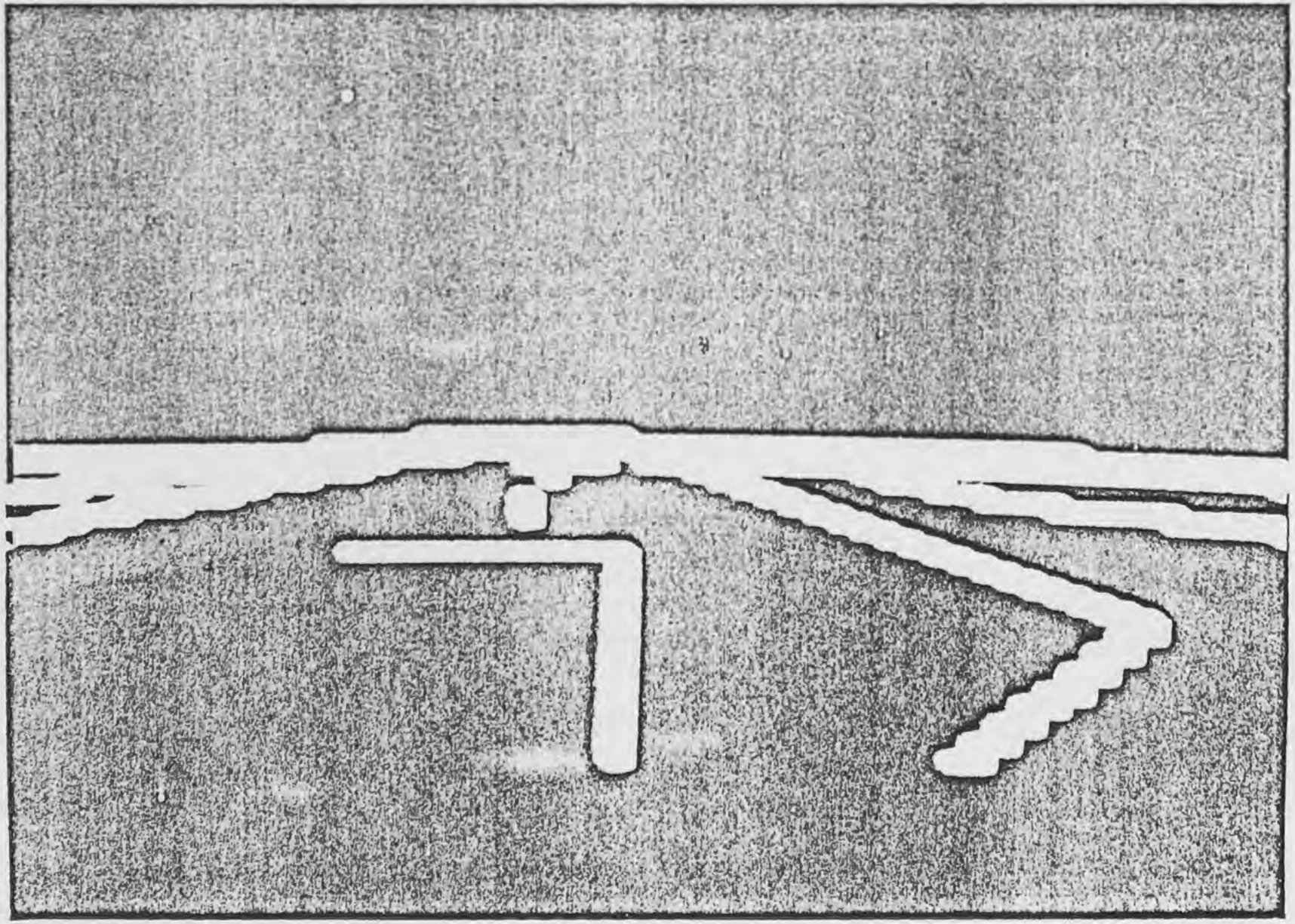


Figure 18. Aerial Approach to Runway 7 at 200 Feet Altitude

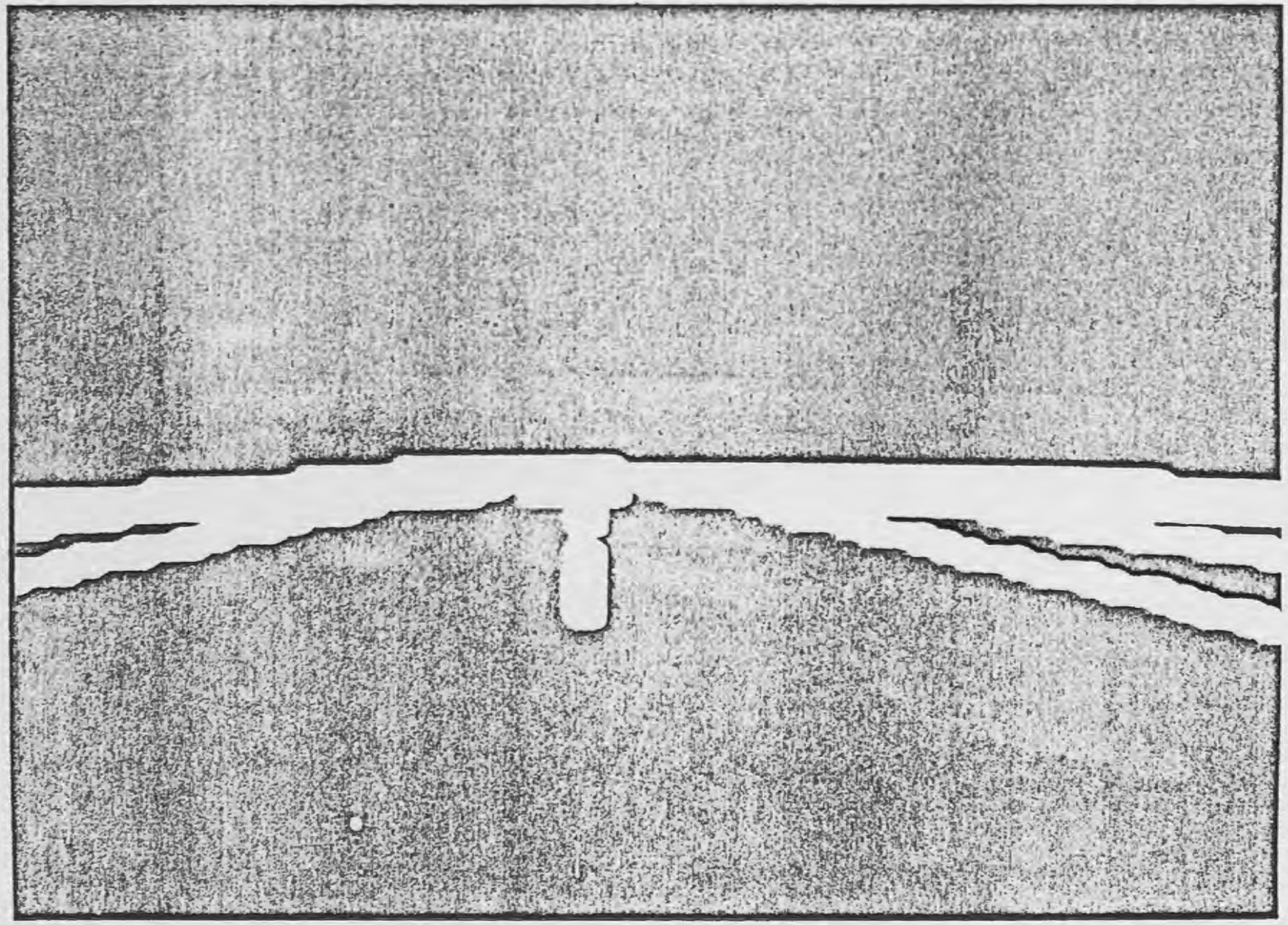


Figure 19. Aerial Approach to Runway 7 at 100 Feet Altitude

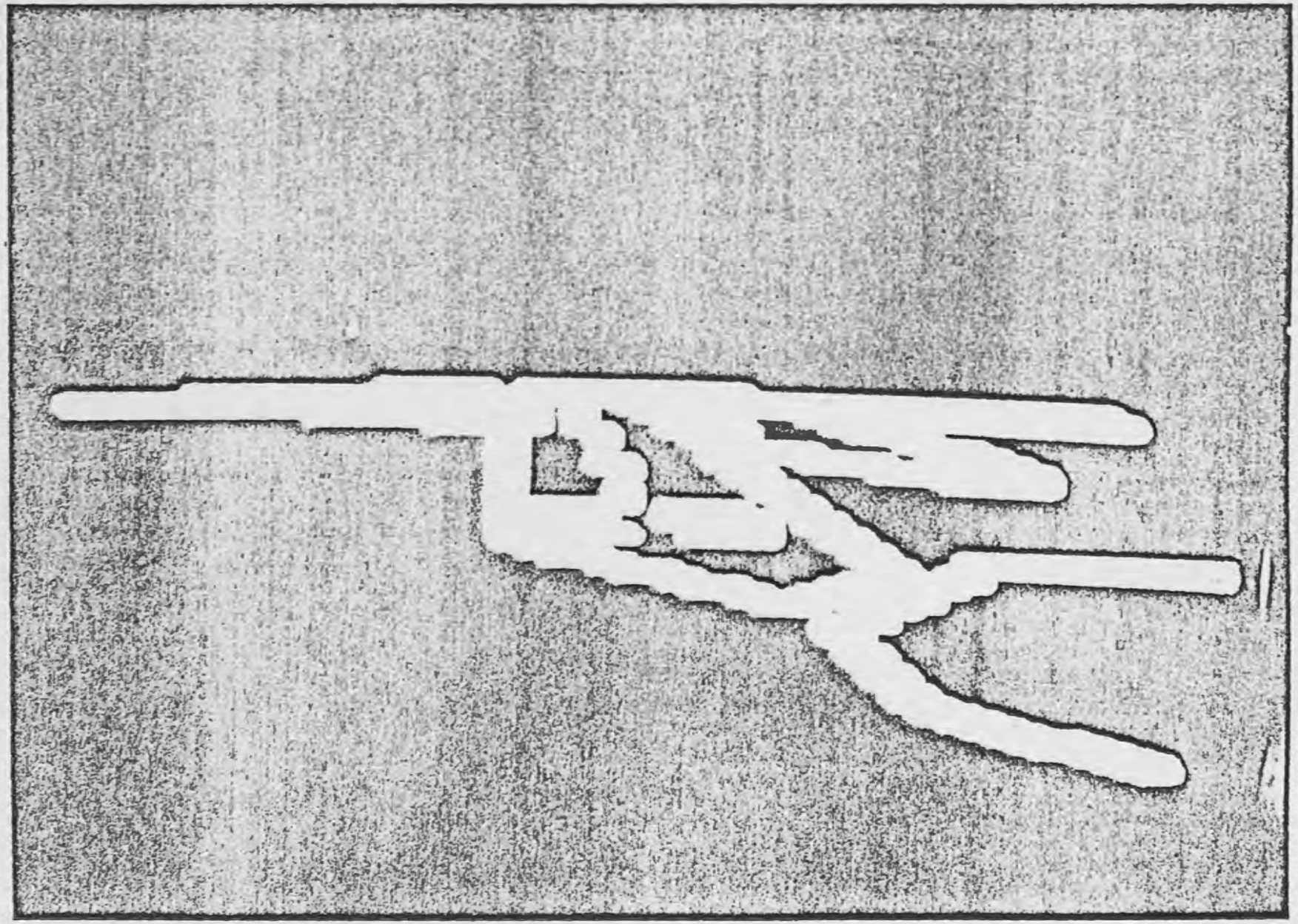


Figure 20. Aerial Approach to Runway 25 at 800 Feet Altitude

Y(3) = 800 feet

Z(3) = 2,300 feet

P = -40 degrees

B = 0 degrees

H = 225 degrees

Figure 21's parameters are:

X(3) = 3,000 feet

Y(3) = 400 feet

Z(3) = 2,000 feet

P = -30 degrees

B = 0 degrees

H = 225 degrees

Figure 22's parameters are:

X(3) = 2,700 feet

Y(3) = 200 feet

Z(3) = 1,800 feet

P = -15 degrees

B = 0 degrees

H = 225 degrees

Approaches to Runway 31 are depicted by Figures 23 and 24. Figure 23's parameters are:

X(3) = 1,800 feet

Y(3) = 600 feet

Z(3) = -3,400 feet

P = -35 degrees

B = 0 degrees

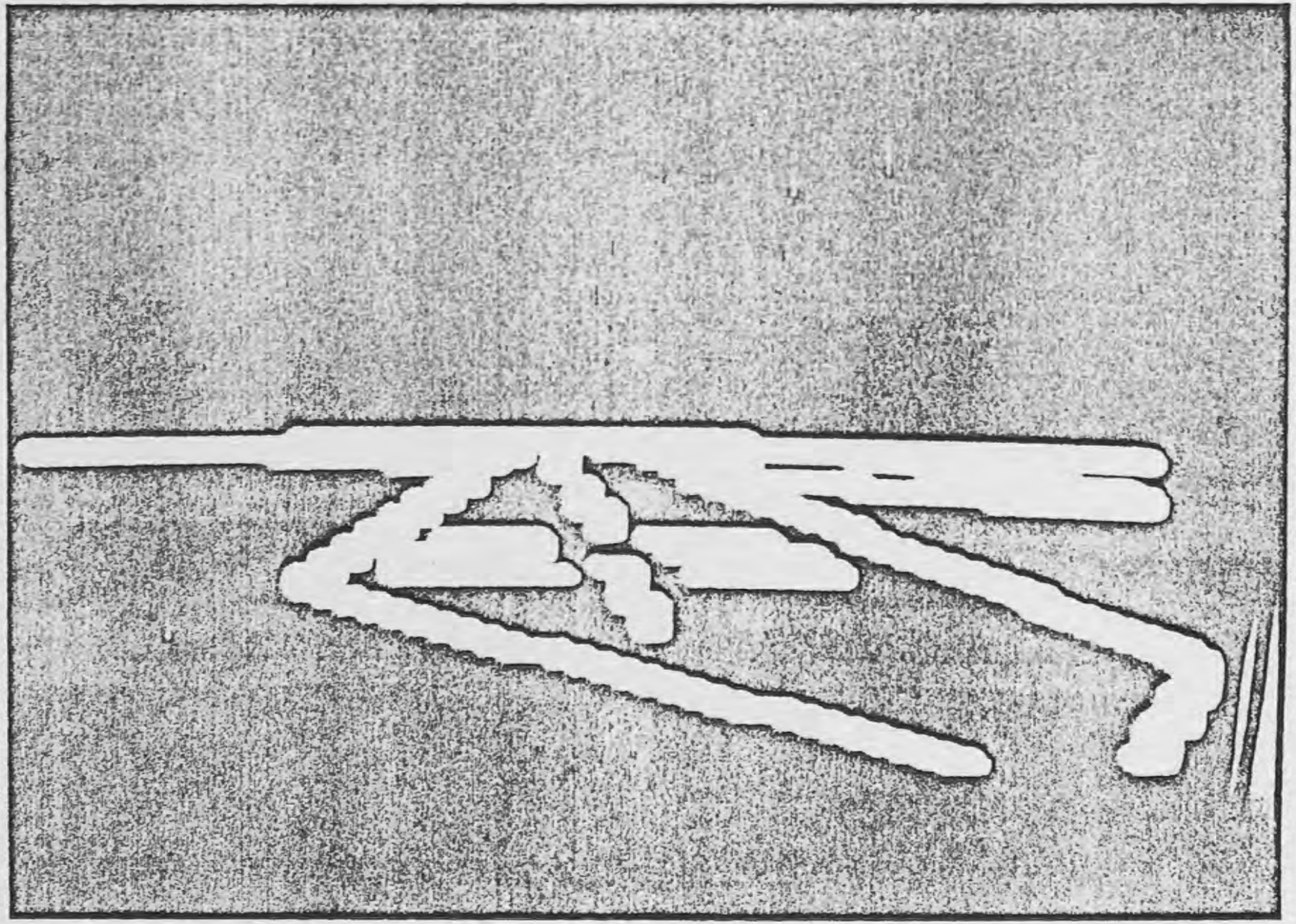


Figure 21. Aerial Approach to Runway 25 at 400 Feet Altitude

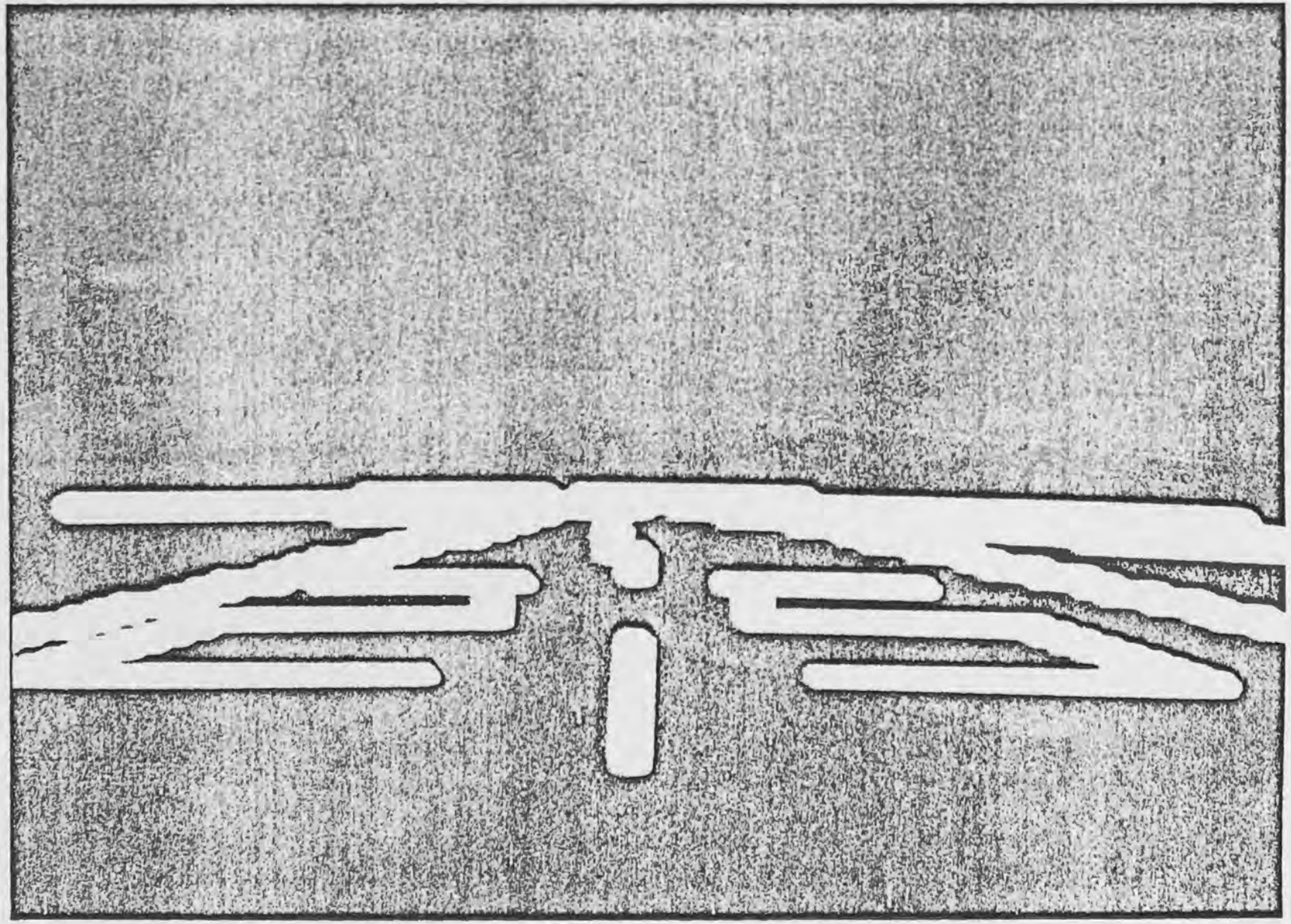


Figure 22. Aerial Approach to Runway 25 at 200 Feet Altitude

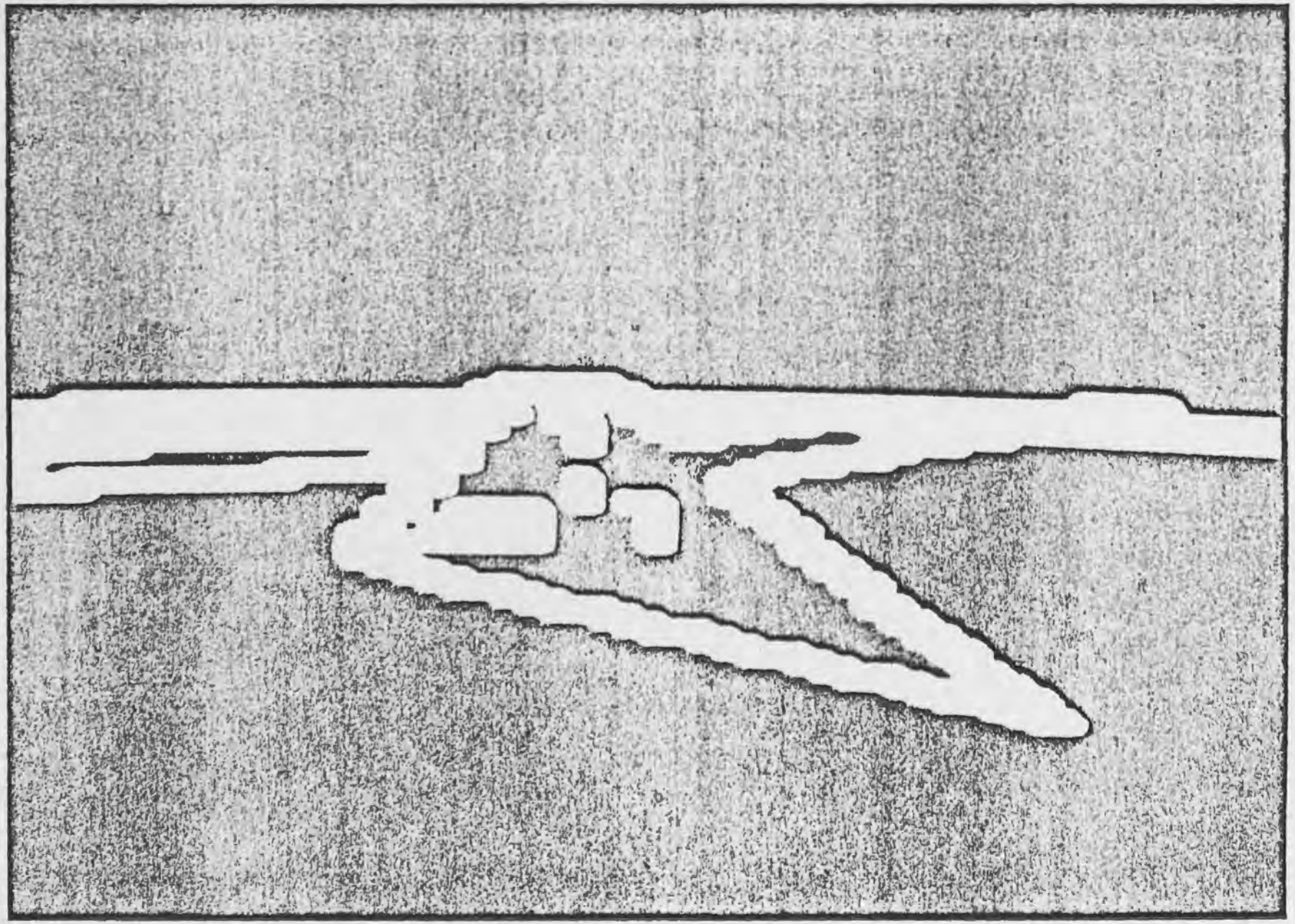


Figure 23. Aerial Approach to Runway 31 at 600 Feet Altitude

H = 140 degrees

Figure 24's parameters are:

X(3) = 1,400 feet

Y(3) = 200 feet

Z(3) = -3,000 feet

P = -15 degrees

B = 0 degrees

H = 140 degrees

In order to verify the accuracy of the program, the following figures 25 and 26 were projected by making the bank a value other than 0. Figure 25 was calculated and projected with the following parameters:

X(3) = 1,400 feet

Y(3) = 200 feet

Z(3) = -3,000 feet

P = -15 degrees

B = 45 degrees

H = 140 degrees

Figure 26's parameters are:

X(3) = 1,400 feet

Y(3) = 200 feet

Z(3) = -3,000 feet

P = -15 degrees

B = -45 degrees

H = 140 degrees

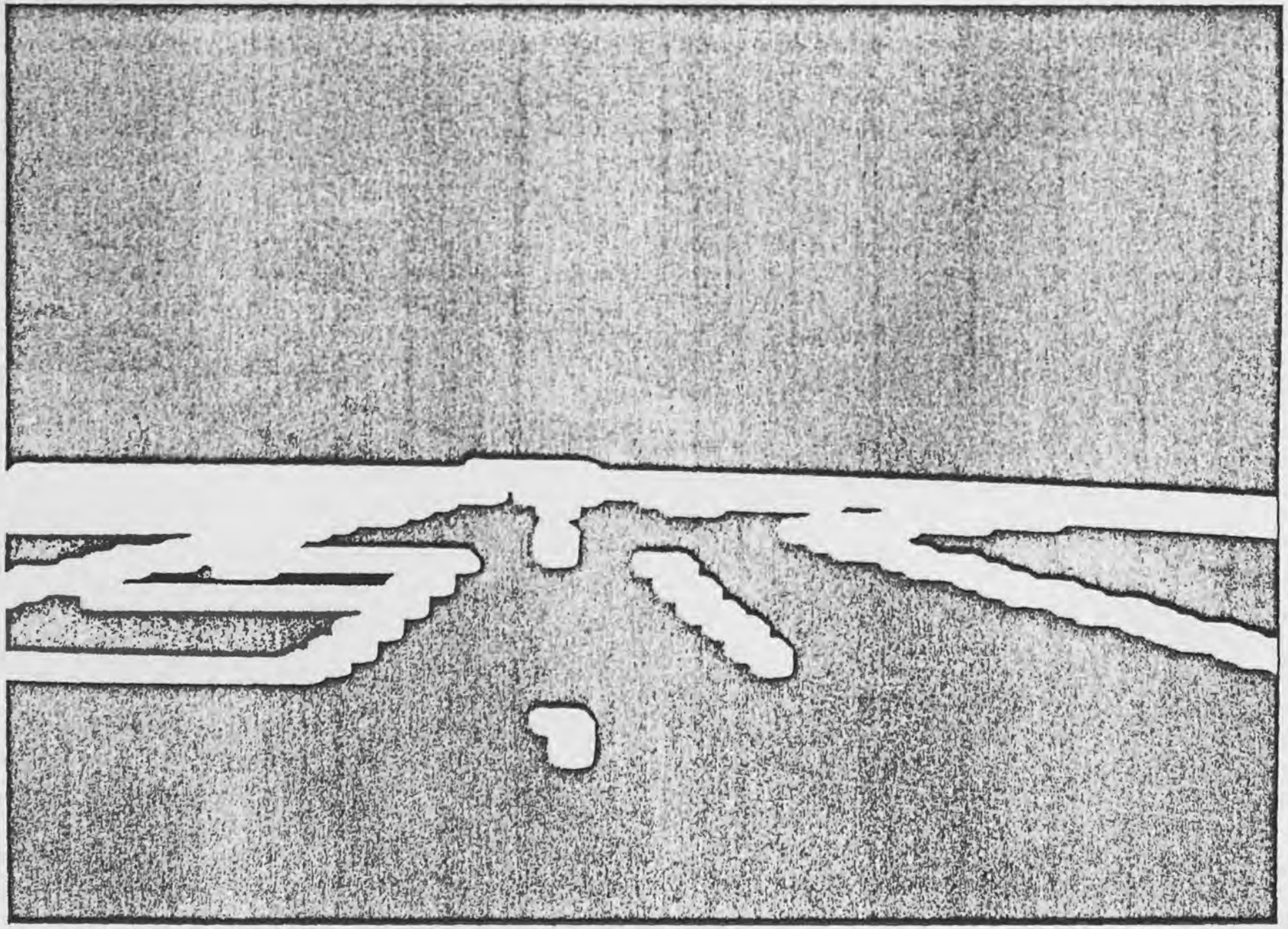


Figure 24. Aerial Approach to Runway 31 at 200 Feet Altitude

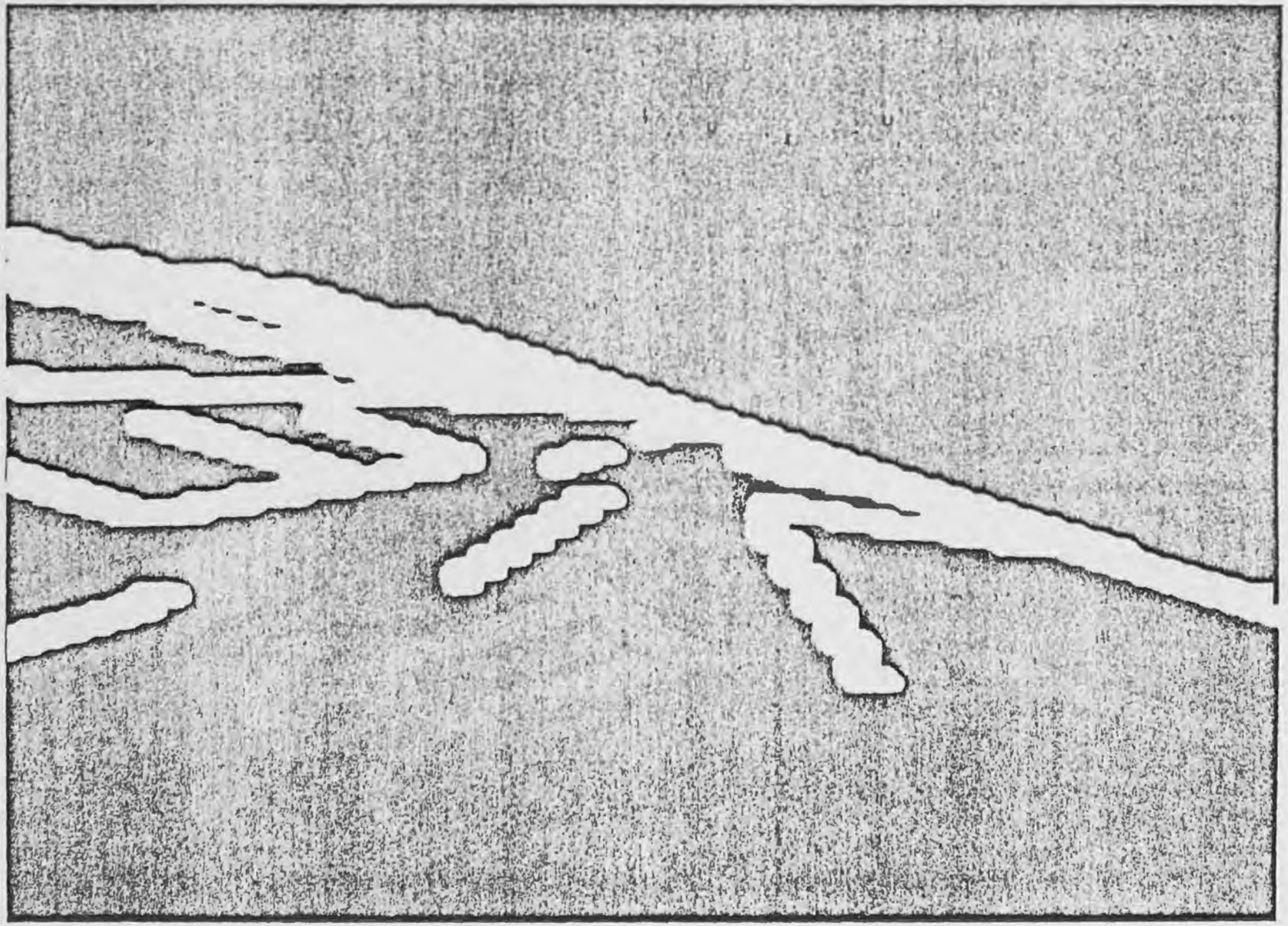


Figure 25. Aerial Approach to Runway 31 with a 45 Degree Bank

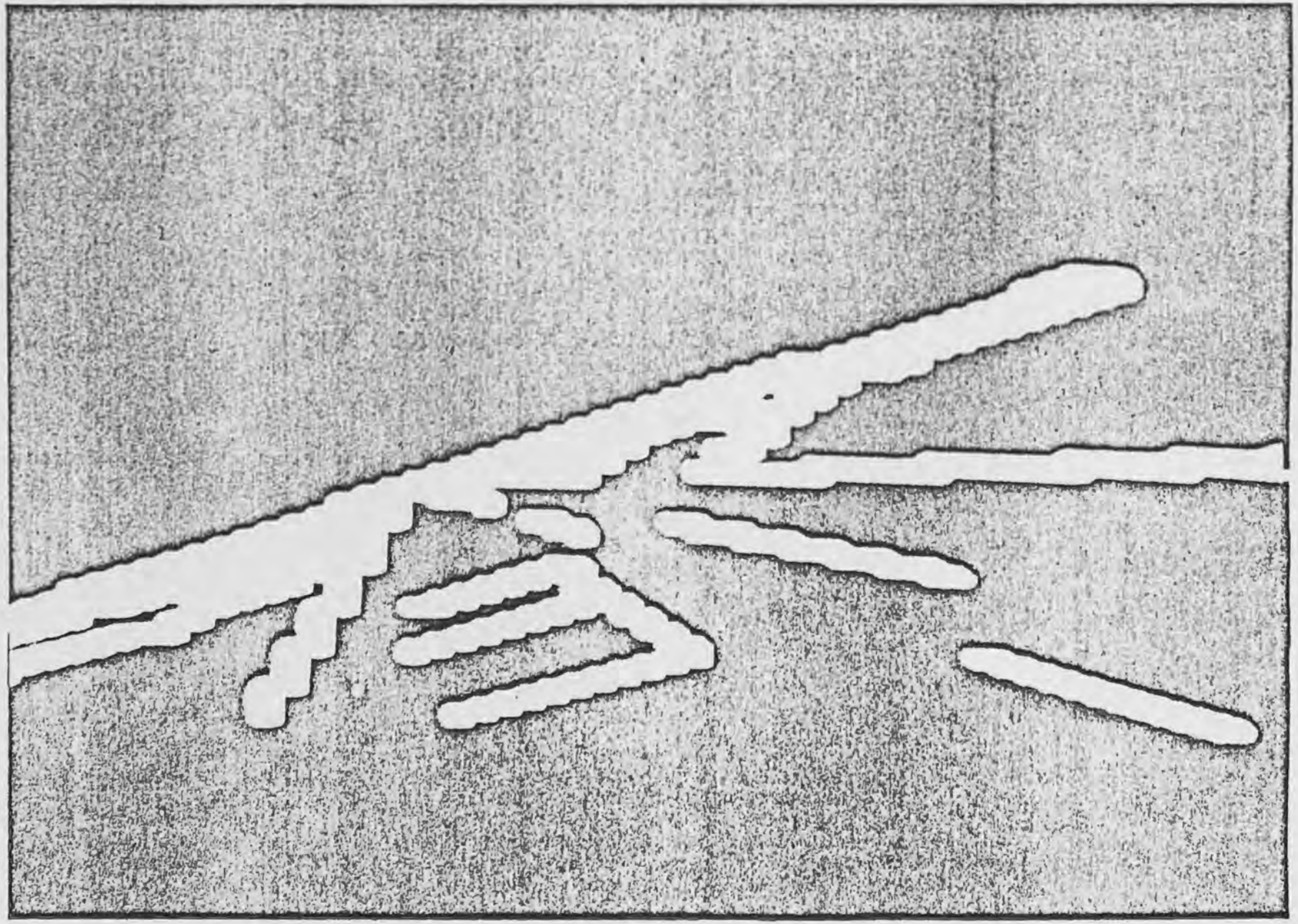


Figure 26. Aerial Approach to Runway 31 with a -45 Degree Bank

As can be viewed from the two figures, a positive rotation of 45 degrees for bank caused the picture to swing up. The input of -45 degrees caused the picture to roll the same amount in the opposite direction as the 45 degrees.

A very interesting projection can be seen in Figure 27. In this picture the bank was made equal to 180 degrees or in essence turn the plane upside down and approach the landing strip. The parameters for Figure 27 are:

X(3) = 1,400 feet

Y(3) = 200 feet

Z(3) = -3,000 feet

P = -15 degrees

B = 180 degrees

H = 140 degrees

The final two figures depict 90 degree bank approaches to Runways 25 to 31. Once again the field is rolled to its side as one would expect when approaching the runway with the plane rolled on its side. Figure 28's input parameters are:

X(3) = 2,700 feet

Y(3) = 200 feet

Z(3) = 1,800 feet

P = -15 degrees

B = 90 degrees

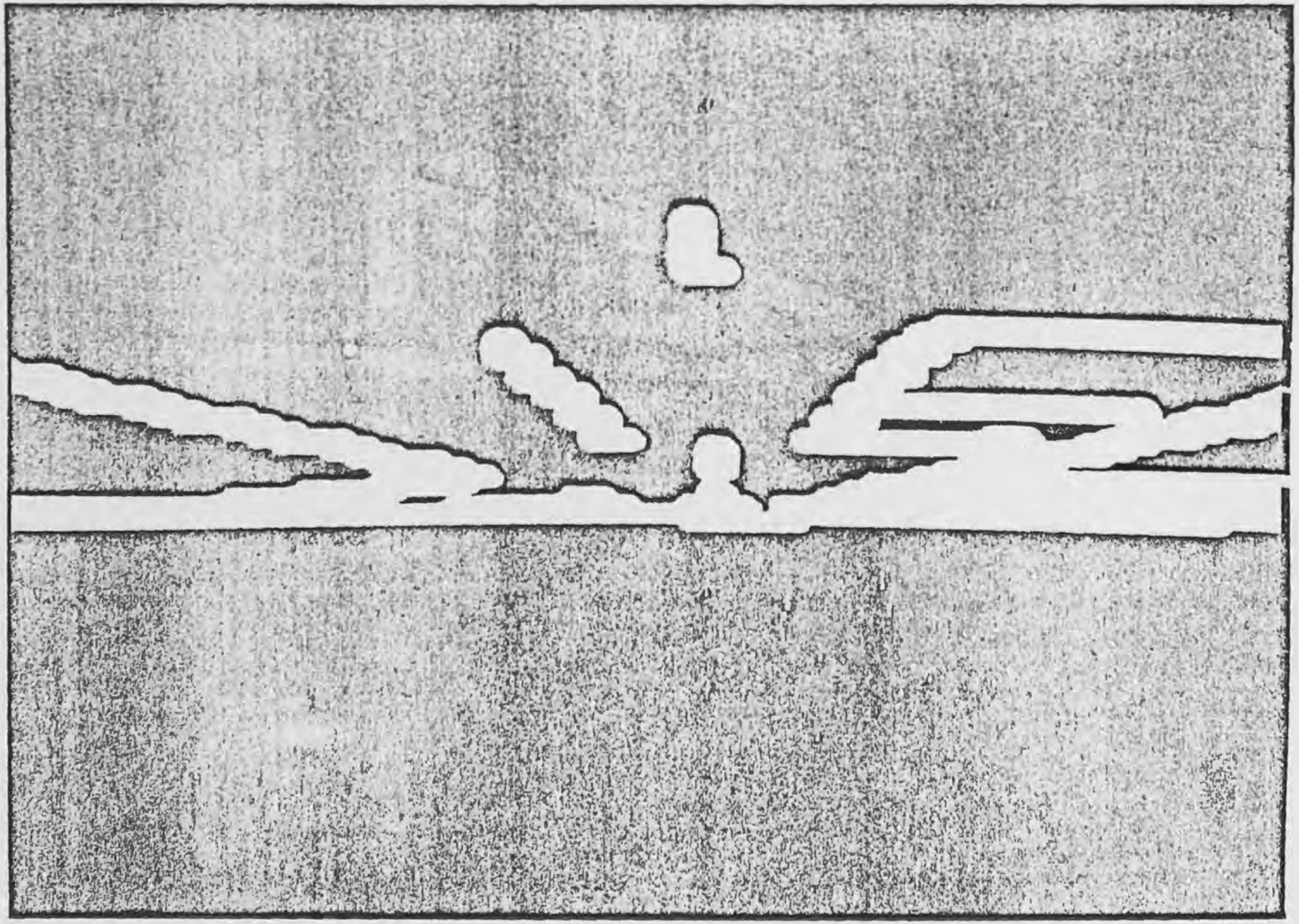


Figure 27. Aerial Approach to Runway 31 with a 180 Degree Bank

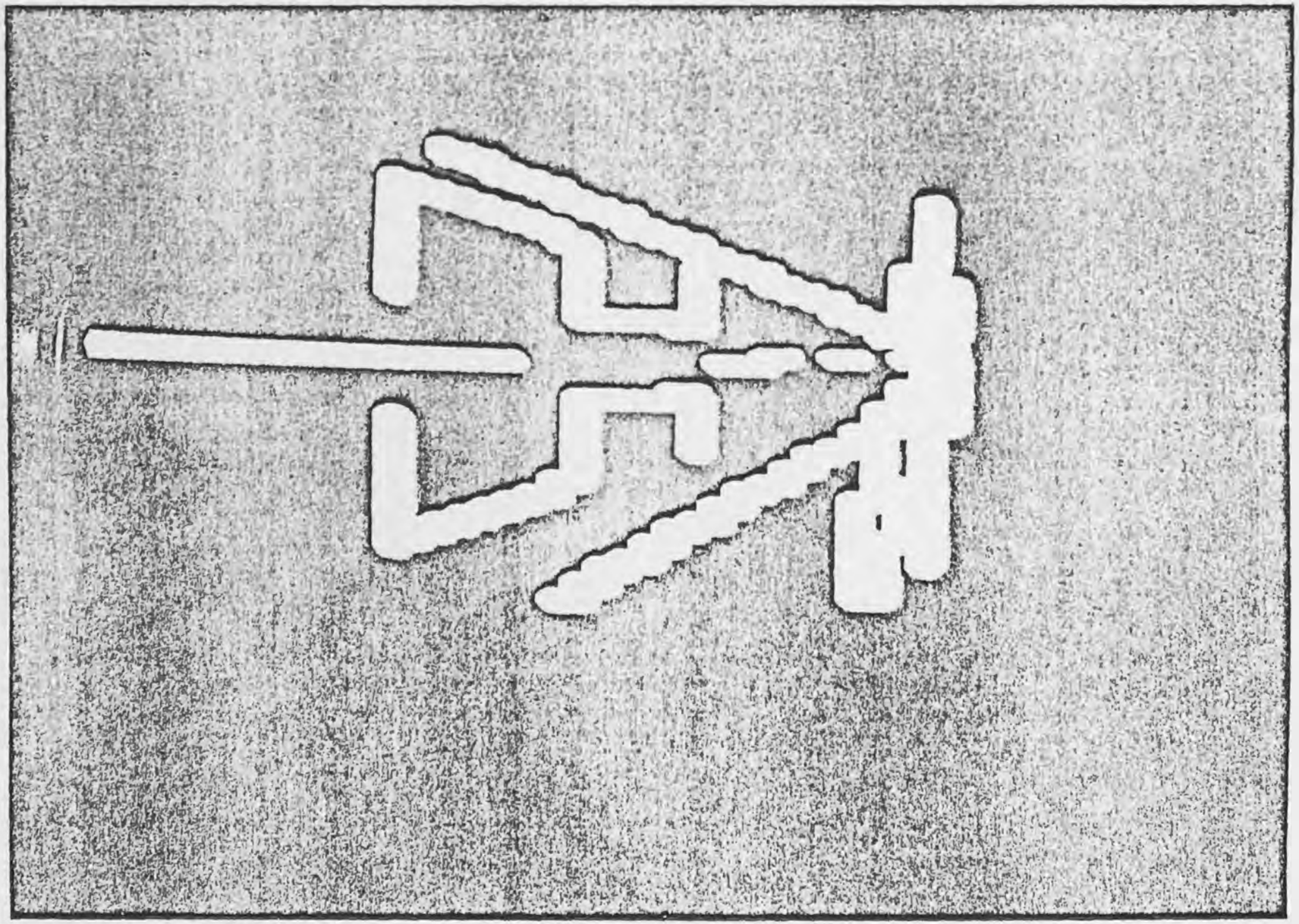


Figure 28. Aerial Approach to Runway 25 with a 90 Degree Bank

H = 225 degrees

Figure 29's input parameters are:

X(3) = 1,400 feet

Y(3) = 200 feet

Z(3) = -3,000 feet

P = -15 degrees

B = 90 degrees

H = 140 degrees

Because display devices have limitations, display systems are a compromise between the interests of men and capabilities of the equipment (Miller, 1968). Two of the major limitations of the video monitor used in this program are resolution and round-off.

Resolution is a measure of discrimination within a fine detail. Resolution depends partly upon the visual activity of the eye and partly upon the resolution of the display itself (Miller, 1968). The resolution of the display will depend upon the size of the screen, the strength of the phosphor and the graininess of the web upon which the display is spread out. Thus, some of the blurriness which shows up on the pictures has to do with the size of the screen which causes resolution.

The second limitation of the screen deals with round off. The graphics display terminal can only accept integer values as inputs to the video monitor. Thus,

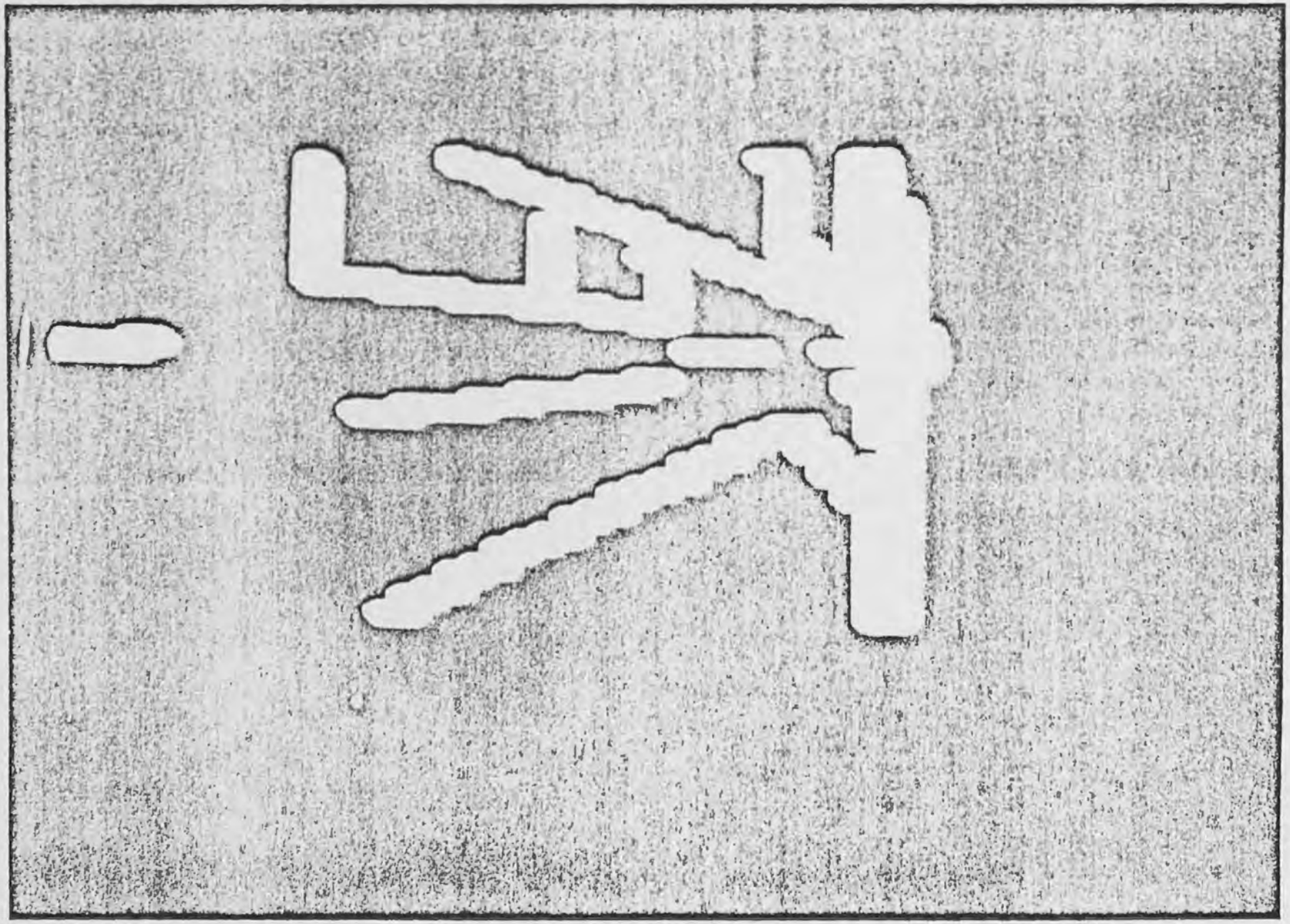


Figure 29. Aerial Approach to Runway 31 with a 90 Degree Bank

some of the lines do not line up, such as the center lines of the runways because of the roundoff that must take place. This is an inherent limitation of the equipment that is currently being used.

The graphical display is, without a doubt, one of the most fascinating devices that computer technology has produced. To see one in action for the first time can be an unforgettable experience. Loading in the aforementioned software and simulate flight landings will leave one with a great awareness of the computer's potentialities.

CHAPTER IV

CONCLUSIONS AND RECOMMENDATIONS FOR FUTURE STUDY

The evolution of computer graphics with its associated graphical displays has evolved as one of the most fascinating areas of computer technology. The essence of this research was to explore the feasibility of a microprocessor to accurately demonstrate the use of computer graphics for flight simulation. Many large scale flight simulators are on the market that utilize computers to create flying conditions. The major setback in the use of these simulators is that only the military and large airlines can afford these systems. There is a large market for flight simulators if certain parameters can be obtained. This research demonstrates how these parameters are satisfied.

There are three main parameters that must be satisfied if flight simulators are to be used by small flight training schools. The three parameters are:

1. Low Cost: The system must be designed to be implemented on a small computer, equipped with a relatively small highspeed

memory, a backing store, a linedrawing display and input devices including a keyboard.

2. Versatility: The use of a local dedicated computer rather than a remote time-shared system enables the system to provide a good response. The use of a paged associated data structure enables large data files to be constructed very easily without the need for excessive fast memory.
3. Ease of Use: The system is designed to be operated and programmed by means of a single, high-level language (Newman and Sproull, 1973).

The software and hardware package put together by the writer has taken into account the three basic parameters of low cost, versatility, and ease of use. Each parameter was optimized to ensure a computer graphics system that could eventually be at the disposal of many small flight training schools.

The system used in this study was built by the Southwest Terminal Systems. The system contains a small computer that is equipped with a relatively high-speed memory, a back store called a dual floppy disk unit, a line drawing

display, and a keyboard. The total package would cost less than $10,000 based on 1979 estimates. This is truly a relatively low price for a 32K memory computer. As more and more companies find it economically feasible to enter the microprocessor business, it is conceivable that the initial cost of a graphics system will decline as it did in the calculator industry.

The versatility of the computer graphics system used in this study is evident in that it is a local computer dedicated to one task and not time-shared. The addition of the floppy disk unit enables the user to store an infinite number of data bases. The various data bases can then be retrieved within a matter of seconds off the magnetic disks that the data was stored. The ease and versatility that data bases can be created and changed is demonstrated by this study. Very little storage area is required for the magnetic disks that contains the data bases is just another excellent feature of this type of computer graphic system.

Since ease of use is a major design parameter, the choice of software language must be of prime importance. Even though the mathematical formulas used in this study were complex in nature, the language used to represent the formulas was simple in structure. The main body of this study was written in BASIC. The system was designed to be

operated and programmed by means of BASIC which is a single, high-level language.

One major area for future research is to increase the speed at which the new projections are calculated and projected on the video monitor. In order for the system to be used for actual flight simulation, the program must be able to generate twenty new projections and output to the video monitor these projections within one second. This can conceivably be done by converting the program from BASIC to machine language.

The second area where additional research can be done from this study is to make the position and direction parameters inputs from the cockpit instead of the dummy terminal. These inputs can be generated by the use of transducers. The output voltages from these transducers can be fed into an analog to digital converter. The outputs from the converter would then be the inputs to the microprocessor to be used for generating new projections.

It is clear that great advances have been made in building low cost, highly versatile, and user oriented, transformation processors. Hopefully, the next few years will also see the development of high-quality, high-performance, and inexpensive alternatives to large scale flight simulators that are currently on the market.

The research done by the writer to develop an inexpensive flight view display system has dealt with a

technology of computer graphics that is still at an early stage of development. As research proceeds in this area, it is conceivable that the end result will be a low cost, highly versatile, and totally user-oriented flight simulator.

APPENDIX A

```
0003 POKE (103, 112)
0004 POKE (104, 00)
0005 LET A=USER (1)
0006 LET A=USER (3)
0007 LET A=USER (2)
0009 DIM A1(192),A2(192),A4(192),A5(192)
0010 DIM A7(192),A8(192),C(8)
0011 M1=0
0012 N1=0
0013 N2=0
0014 N3=0
0015 N4=0
0016 N5=0
0020 FOR I=1 TO 192
0030 READ A1(I)
0040 NEXT I
0050 FOR I=1 TO 192
0060 READ A2(I)
0070 NEXT I
0080 DATA -5,2,2,2,2,3,3,3,3,4,4,3,3,3,3,0,0,-4,-4,-8,-8,
     -8,-8
0081 DATA -5,23,29,29,31,31,31,31,28,28,22,22,20,20,20,20,
     23
0082 DATA 19,10,10,10,10,28,28,28,28,25,25,10,10,10,10,8,
     8,-10
0083 DATA -10,-12,-12,-25,-25,-25,-25,-16,-16,-16,-16,-27,
     -27
0084 DATA -27,-27,-19,-19,-14,-14,-4,-4,-3,-3,11,11,19,4,
     4,4,3
0085 DATA 3,-6,-6,-6,-6,-4,-4,4,-8,-10,-10,-11,-11,-19,-19,
     -19
0086 DATA -19,-17,-17,-8,4,4,4,1,1,1,1,4,-15,-14,-14,-17,
     25,24
0087 DATA 24,25,25,26,26,27,27,26,22,23,23,24,24,23,23,24,
     24,25
0088 DATA -13,-11,-9,-7,-4,-3,0,2,5,7,11,13,17,19,21,23,
     25,27
0090 DATA -23,-24,-24,-22,-22,-21,-23,-22,9,10,10,12,12,
     11,11
0091 DATA 10,11,13,-19,-17,-15,-13,-11,-9,-4,-3,-2,0,2,4,
     8,10
0092 DATA 13,15
0100 DATA -31,-31,-31,-29,-29,-28,-28,-27,-27,-26,-26,-25,
     -25
0101 DATA -24,-24,-21,-21,-21,-21,-25,-25,-28,-29,-31,22,
     22,22
0102 DATA 24,24,27,27,30,30,30,30,28,28,25,25,22,-31,-22,
     22
0103 DATA -3,-3,15,15,21,21,21,21,7,7,18,18,18,18,0,0,0,0,
     13
```

```
0104 DATA 13,4,4,-4,-4,-6,-6,-17,-17,-19,-19,-27,-27,-27,
     -27,-17
0105 DATA -17,-17,-17,-31,-31,-31,0,4,4,4,4,-5,-5,-6,-6,
     -8,-8
0106 DATA 0,-12,-10,-10,-10,-10,-18,-18,-19,-19,-21,-21,
     -12,-16
0107 DATA -9,-9,-12,-12,-13,-13,-16,-23,-24,-24,-27,13,14,
     14
0108 DATA 15,15,14,14,15,15,16,16,15,15,16,16,17,17,18,18,
     17
0109 DATA -22,-20,-18,-16,-13,-12,-9,-7,-4,-2,2,4,8,10,12,
     14
0110 DATA 16,18,7,6,6,4,4,5,5,6,-28,-27,-27,-29,-29,-30,
     -28
0111 DATA -29,-26,-28,3,1,-1,-3,-5,-7,-12,-13,-14,-16,-18
0112 DATA -20,-24,-26,-29,-31
0200 FOR I=1 TO 191 STEP 2
0210 POKE (24304,A1(I)+32)
0211 POKE (24305,A1(I+I)+32)
0212 POKE (24306,A2(I)+32)
0213 POKE (24307,A2(I+I)+32)
0214 LET A=USER (5)
0220 NEXT I
0260 W=31
0270 V=1
0271 INPUT X(3), Y(3), Z(3)
0272 X(3)=-(X(3)/100)
0273 Y(3)=-(Y(3)/100)
0274 Z(3)=-(Z(3)/100)
0275 INPUT P,B,H,
0276 P=-P
0277 B=-B
0278 IF H>90 GOTO500
0279 IF H<0 GOTO540
0282 IF M1<>X(3) GO TO 300
0283 IF N1<>Y(3) GO TO 300
0284 IF N2<>Z(3) GO TO 300
0285 IF N3<>P GO TO 300
0286 IF N4<>B GO TO 300
0287 IF N5<>H GO TO 300
0288 LET M1=X(3)
0289 LET N1=Y(3)
0290 LET N2=Z(3)
0291 LET N3=P
0292 LET N4=B
0293 LET N5=H
0294 GOTO 271
0300 GOSUB 8200
0301 LET A=USER (3)
0310 FOR I=1 TO 191 STEP 2
```

```
0320 X(1)=A1(I)
0330 Y(1)=0
0340 Z(1)=A2(I)
0350 X(5)=A1(I+I)
0360 Y(5)=0
0370 Z(5)=A2(I+I)
0380 GOTO 8500
0390 A7(I)=INT(X(2))
0400 A8(I)=INT(Y(2))
0410 A7(I+I)=INT(X(4))
0420 A8(I+I)=INT(Y(4))
0430 POKE (24304,A7(I)+32)
0431 POKE (24305,A7(I+I)+32)
0432 POKE (24306,A8(I)+32)
0433 POKE (24307,A8(I+I)+32)
0434 LET A=USER(5)
0440 NEXT I
0450 GOTO 288
0500 IF H<180 GOTO 520
0510 GOTO 282
0520 H=H- 180
0530 GOTO 282
0540 IF H>-90 GOTO 560
0550 GOTO 282
0560 H=180+H
0570 GOTO 282
8200 F=P
8203 GOSUB 8300
8204 R1=N
8206 F=P
8209 GOSUB 8310
8212 R2=N
8215 F=B
8218 GOSUB 8300
8221 R3=N
8222 F=B
8224 GOSUB 8310
8227 R4=N
8230 F=H
8233 GOSUB 8300
8236 R5=N
8237 F=H
8239 GOSUB 8310
8245 T1=N*R4+R5*R1*R3
8248 T2=N*R3+R5*R1*R4
8251 T3=R5*R2*V
8253 T4=R2*R3
8256 T5=R2*R4
8259 T6=-R1*V
8262 T7=-R5*R4+N*R1*R3
```

```
8265 T8=R5*R3+N*R1*R4
8269 T9=R2*N*V
8272 RETURN
8300 F=F-90
8305 IF F<=-180 THEN F=F+360
8310 IF F<0 THEN F=-F
8315 S=F
8320 IF S>=90 THEN F=180-F
8330 N=F*.01745329
8340 A=N*N
8350 M=A*A
8360 N=1-A/2+M/24-A*M/720+M*M/40320
8370 IF S>= 90 THEN N=-N
8380 RETURN
8500 FOR A=1 TO 5 STEP 4
8510 G=X(A)+X(3)
8520 S=Y(A)+Y(3)
8530 K=Z(A)+Z(3)
8540 X(A)=G*T1+S*T4+K*T7
8550 Y(A)=G*T2+S*T5+K*T8
8560 Z(A)=G*T3+S*T6+K*T9
8570 NEXT A
8600 FOR A=1 TO 5 STEP 4
8610 C(A)=0
8612 C(A+1)=0
8614 C(A+2)=0
8616 C(A+3)=0
8618 IF X(A)<-Z(A) THEN C(A)=1
8620 IF X(A)>Z(A) THEN C(A+1)=1
8622 IF Y(A)<-Z(A) THEN C(A+2)=1
8624 IF Y(A)>Z(A) THEN C(A+3)=1
8626 NEXT A
8632 FOR A=1 TO 4 STEP 1
8634 IF C(A)=0 THEN GO TO 8638
8636 IF C(A)=C(A+4) THEN GO TO 8668
8638 NEXT A
8644 FOR A=1 TO 4 STEP 1
8646 IF C(A)=1 THEN GO TO 8676
8648 NEXT A
8654 FOR A=5 TO 8 STEP 1
8656 IF C(A)=1 GOTO 8686
8658 NEXT A
8662 P2=1
8664 GOTO 8800
8668 P2=0
8670 GOTO 440
8676 A=1
8678 S=5
8680 GOTO 8694
8686 A=5
```

```
8688 S=1
8694 IF C(A)=1 THEN GO TO 8728
8696 IF C(A+1)=1 THEN GO TO 8714
8698 IF C(A=2)=1 THEN GO TO 8742
8700 IF C(A+3)=1 THEN GO TO 8756
8706 GOTO 8662
8714 K=(Z(A)-X(A))/(X(S)-X(A)-Z(S)+Z(A))
8716 X(A)=K*(Z(S)-Z(A))+Z(A)
8718 Y(A)=K*(Y(S)-Y(A))+Y(A)
8720 Z(A)=X(A)
8722 GOTO 8600
8728 K=(Z(A)+X(A))/(X(A)-X(S)-Z(S)+Z(A))
8730 X(A)=K*(Z(A)-Z(S))-Z(A)
8732 Y(A)=K*(Y(S)-Y(A))+Y(A)
8734 Z(A)=-X(A)
8736 GOTO 8600
8742 K=(Z(A)+Y(A))/(Y(A)-Y(S)-Z(S)+Z(A))
8744 X(A)=K*(X(S)-X(A))+X(A)
8746 Y(A)=K*(Z(A)-Z(S))-Z(A)
8748 Z(A)=-Y(A)
8750 GOTO 8600
8756 K=(Z(A)-Y(A))/(Y(S)-Y(A)-Z(S)+Z(A))
8758 X(A)=K*(X(S)-X(A))+X(A)
8760 Y(A)=K*(Z(S)-Z(A))+Z(A)
8762 Z(A)=Y(A)
8764 GOTO 8600
8800 IF Z(1)=- THEN Z(1)=.001
8845 IF Z(5)=0 THEN Z(5)=.001
8855 X(2)=X(1)/Z(1)*W
8860 Y(2)=Y(1)/Z(1)*W
8865 X(4)=X(5)/Z(5)*W
8870 Y(4)=Y(5)/Z(5)*W
8871 IF X(2)=Y(2) GOTO 8876
8875 GOTO 390
8876 IF Y(2)=X(4) GOTO 8878
8877 RETURN
8878 IF X(4)=Y(4) GOTO 440
8879 RETURN
```

→

APPENDIX B

```
00010                                   NAM     SCREEN
00020                                   OPT     O
00030                                   OPT     S
00040 7000                              ORG     $7000
00050 7000 CE 005D TYPE                 LDX     #$005D
00060 7003 EE 00                        LDX     0,X
00070 7005 A6 00                        LDA A   0,X
00080 7007 84 0F                        AND A   #$0F
00090 7009 81 01                        CMP A   #01
00100 700B 27 15                        BEQ     FIRST
00110 700D 81 02                        CMP A   #02
00120 700F 27 60                        BEQ     JINIT
00130 7011 81 03                        CMP A   #03
00140 7013 27 10                        BEQ     ERASE
00150 7015 81 04                        CMP A   #04
00160 7017 27 47                        BEQ     PIXEL
00170 7019 81 05                        CMP A   #05
00180 701B 27 52                        BEQ     SHOW 2
00181 701D 81 06                        CMP A   #06
00182 701F 27 5C                        BEQ     START
00190 7021 39                           RTS
00200             *PIA INITIALIZATION SECTION
00210 7022 4F   FIRST                   CLR A
00220 7023 B7 8009                      STA A   $8009
00230 7026 86 FF                        LDA A   #$FF
00240 7028 B7 8008                      STA A   $8008
00250 702B 86 3F                        LDA A   #$3F
00260 702D B7 8009                      STA A   $8009
00270 7030 39                           RTS
00280             *ERASE SCREEN SECTION
00290 7031 86 3F ERASE                  LDA A   #63
00300 7033 B7 5EF8                      STA A   HPOS
00310 7036 B6 5EF8 HSET                 LDA A   HPOS
00320 7039 8D 14                        BSR     SENDA
00330 703B 86 80 VBAR                   LDA A   #128
00340 703D 8D 10 CLEAR                  BSR     SENDA
00350 703F 4C                           INC A
00360 7040 81 E0 MOD                    CMP A   #224
00370 7042 26 F9                        BNE     CLEAR
00380 7044 7A 5EF8                      DEC     HPOS
00390 7047 2C ED                        BGE     HSET
00400 7049 86 C0                        LDA A   #192
00410 704B B7 7041                      STA A   MOD+1
00420 704E 39                           RTS
00430             SCREEN DRIVE SUBROUTINES
00440 704F 16                           SENDA   TAB
00450 7050 F7 8008 SENDB                STA B   $8008
00460 7053 C6 37                        LDA B   #$37
00470 7055 F7 8009                      STA B   $8009
00480 7058 E6 00                        LDA B   0,X
```

```
00490 705A C6 3F                      LDA B    #$3F
00500 705C F7 8009                    STA B    $8009
00510 705F 39                         RTS
00520           *PIXEL OUTPUT
00530 7060 F6 5EF0 PIXEL              LDA B    X1
00540 7063 CB 40                      ADD B    #64
00550 7065 8D E9                      RSR      SENDR
00560 7067 C6 E0                      LDA B    #224
00570 7069 F0 5EF2                    SUB B    Y1
00580 706C 8D F2                      BSR      SENDR
00590 706E 39                         RTS
00600           *
00610 706F 20 56 SHOW 2               BRA      SHOW
00620           *DATA INPUT PGM FOR JOYSTICK BOX
00630           *INITIALIZE PIA INTERFACE TO JOYSTICK
00640 7071 86 3D JINIT                LDA A    #$3D
00650 7073 CE 8008                    LDX      #$8008
00660 7076 C6 00                      LDA B    #$00
00670 7078 E7 02                      STA B    2,X
00680 707A A7 03                      STA A    3,X
00700           *SAVE MACHINE REG'S ON INPUT
00710 707D B7 5EFB START              STA A    ASAVE
00720 7080 F7 5EFC
00730 7083 FF 5EFD                    STX      ISAVE
00740           *GET HORIZONTAL STICK POSITION
00741 7086 CE 8008                    LDX      #$8008
00750 7089 A6 03 LOOP1                LDA      3,X
00760 708B 84 80                      AND A    #$80
00770 708D 27 FA                      BEQ      LOOP1
00780 708F A6 02                      LDA A    2,X
00790 7091 16                         TAB
00800 7092 84 80                      AND A    #$80
00810 7094 26 F3                      BNE      LOOP1
00820 7096 C4 3F                      AND B    #$3F
00821 7098 C0 20                      SUB B    #$20
00822 709A 2C 03                      BGE      RTN1
00823 709C 50                         NEG B
00824 709D CB 40                      ADD B    #64
00830 709F F7 5EF9 RTN1               STA B    JHPOS
00840 70A2 A6 03 LOOP2                LDA A    3,X
00850 70A4 84 80                      AND A    #$80
00860 70A6 27 FA                      BEQ      LOOP2
00870 70A8 A6 02                      LDA A    2,X
00880 70AA 16                         TAB
00890 70AB 84 80                      AND A    #$80
00900 70AD 26 02                      BNE      GOBACK
00910 70AF 20 D8                      BRA      LOOP1
00920 70B1 C4 3F GO BACK              AND B    #$3F
00921 70B3 C0 20                      SUB B    #$20
00922 70B5 2C 03                      BGE      RTN2
```

```
00923 70B7 50                          NEG B
00924 70B8 CB 40                       ADD B    #64
00930 70BA F7 5EFA RTN2                STA B    JVPOS
00940 70BD B6 5EFB                     LDA A    ASAVE
00950 70C0 F6 5EFC                     LDA B    BSAVE
00960 70C3 FE 5EFD                     LDX      ISAVE
00970 70C6 39                          RTS
00980            *ROUTINE TO DRAW LINE FROM (X1,Y1)
                  TO (X2,Y2)
00990 70C7 4F     SHOW                 CLR A
01000 70C8 C6 01                       LDA B    #1
01010 70CA F7 5EF4                     STA B    M
01020 70CD F7 5EF5                     STA B    N
01030 70D0 F6 5EF1                     LDA B    X2
01040 70D3 F0 5EF0                     SUB B    X1
01050 70D6 F7 5EF6                     STA B    D
01060 70D9 2C 06                       BGE      BPI
01070 70DB 70 5EF4                     NEG      M
01080 70DE 70 5EF6                     NEG      D
01090 70E1 26 02 BPI                   BNE      BP2
01100 70E3 86 FF                       LDA A    #$FF
01110 70E5 F6 5FF3 RP2                 LDA B    Y2
01120 70EB F0 5EF2                     SUB B    Y1
01130 70EB F7 5EF7                     STA B    E
01140 70EE 2C 06                       BGE      B8963
01150 70F0 70 5EF5                     NEG      N
01160 70F3 70 5EF7                     NEG      E
01170 70F6 BD 7060 B8963               JSR      PIXEL
01180 70F9 F6 5EF0                     LDA B    X1
01190 70FC F1 5EF1                     CMP B    X2
01200 70FF 27 1F                       BEQ      B8990
01210 7101 4D   B8969                  TST A
01220 7102 2D 0E                       LBT      B8981
02130 7104 F6 5EF0                     LDA B    X1
01240 7107 FB 5EF4                     ADD B    M
01250 710A F7 5EF0                     STA B    X1
01260 710D B0 5EF7                     SUB A    F
01270 7110 20 E4                       BRA      B8963
01280 7112 F6 5EF2 B8981               LDA B    Y1
01290 7115 FB 5EF5                     ADD B    N
01300 7118 F7 5EF2                     STA B    Y1
01310 711B BB 5EF6                     ADD A    D
01320 711E 20 D6                       BRA      B8963
01330 7120 F6 5EF2 B8990               LDA B    Y1
01340 7123 F1 5EF3                     CMP B    Y2
01350 7126 26 D9                       BNE      B8969
01360 7128 39                          RTS
01380 5EF0 X1                          EQU      $5EF0
01390 5EF1 X2                          EQU      $5EF1
```

```
01400 5EF2 Y1       EQU     $5EF2
01410 5EF3 Y2       EQU     $5EF3
01420 5EF4 M        EQU     $5EF4
01430 5EF5 N        EQU     $5EF5
01440 5EF6 D        EQU     $5EF6
01450 5EF7 E        EQU     $5EF7
01460 5EF8 HPOS     EQU     $5EF8
01470 5EF9 JHPOS    EQU     $5EF9
01480 5EFA JVPOS    EQU     $5EFA
01490 5EFB ASAVE    EQU     $5EFB
01500 5EFC BSAVE    EQU     $5EFC
01510 5EFD ISAVE    EQU     $5EFD
                    END

TYPE 7000
FIRST 7022
ERASE 7031
HSET 7036
VBAR 703B
CLEAR 703D
MOD 7040
SENDA 704F
SENDB 7050
PIXEL 7060
SHOW2 706F
JINIT 7071
START 707D
LOOP1 7089
RTN1 709F
LOOP2 70A2
GOBACK 70B1
RTN2 70BA
SHOW 70C7
BPI 70E1
BP2 70E5
B8963 70F6
B8969 7101
B8981 7112
B8990 7120
XI 5EF0
X2 5EF1
Y1 5EF2
Y2 5EF3
M  5EF4
N  5EF5
D  5EF6
E  5EF7
HPOS 5EF8
JHPOS 5EF9
JVPOS 5EFA
ASAVE 5EFB
```

```
BSAVE 5EFC
ISAVE 5EFD
```

LIST OF REFERENCES

Bauer, C. S. Jr. 1978. "Display device driver routines." Interoffice memorandum, University of Central Florida, Orlando, Florida.

Elson, B. M. October 4, 1976. "Simulators slash pilot transition time." Aviation Week. 105: 59-61.

Evans, C. L. April 12, 1976. "A (computer generated) picture is worth 1000 words." Approach. 108: 10-12.

Fetter, W. A. 1965. Computer graphics in communications. New York: McGraw-Hill.

General Electric, 1977. "Computer image generation; visual simulation systems for today's training needs." 4th edition. Daytona Beach, Florida: General Electric.

Giloi, W. K. 1978. Interactive computer graphics, Englewood Cliffs, New Jersey: Prentice-Hall.

Law, G. M. August 10, 1975. "Simulator at Williams base trains pilots in all aspects of flying." The Arizona Republic Newspaper. B-3.

McCormick, E. J. 1970. Human factors engineering. New York: McGraw-Hill.

Miller, W. E. 1968. "Visual information display systems." Survey presented to National Aeronautics and Space Administration, Washington, D. C.

Newman, W. M. and Sproull, R. F. 1973. Principles of interactive computer graphics. New York: McGraw-Hill.

Risher, T. A. 1976. "A computer graphics analysis of a freeway merge control system." Master's thesis, Florida Technological University.

Rogers, D. F. and Adams, J. A. 1976. Mathematical elements for computer graphics. New York: McGraw-Hill.

Sub Logic, Inc., 1976. "BASIC microcomputer graphics user's manual." Tampa, Florida: Sub Logic, Inc.