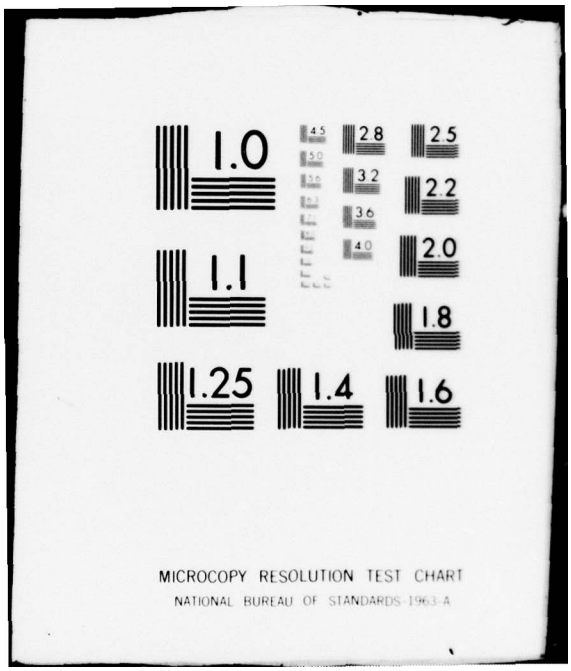
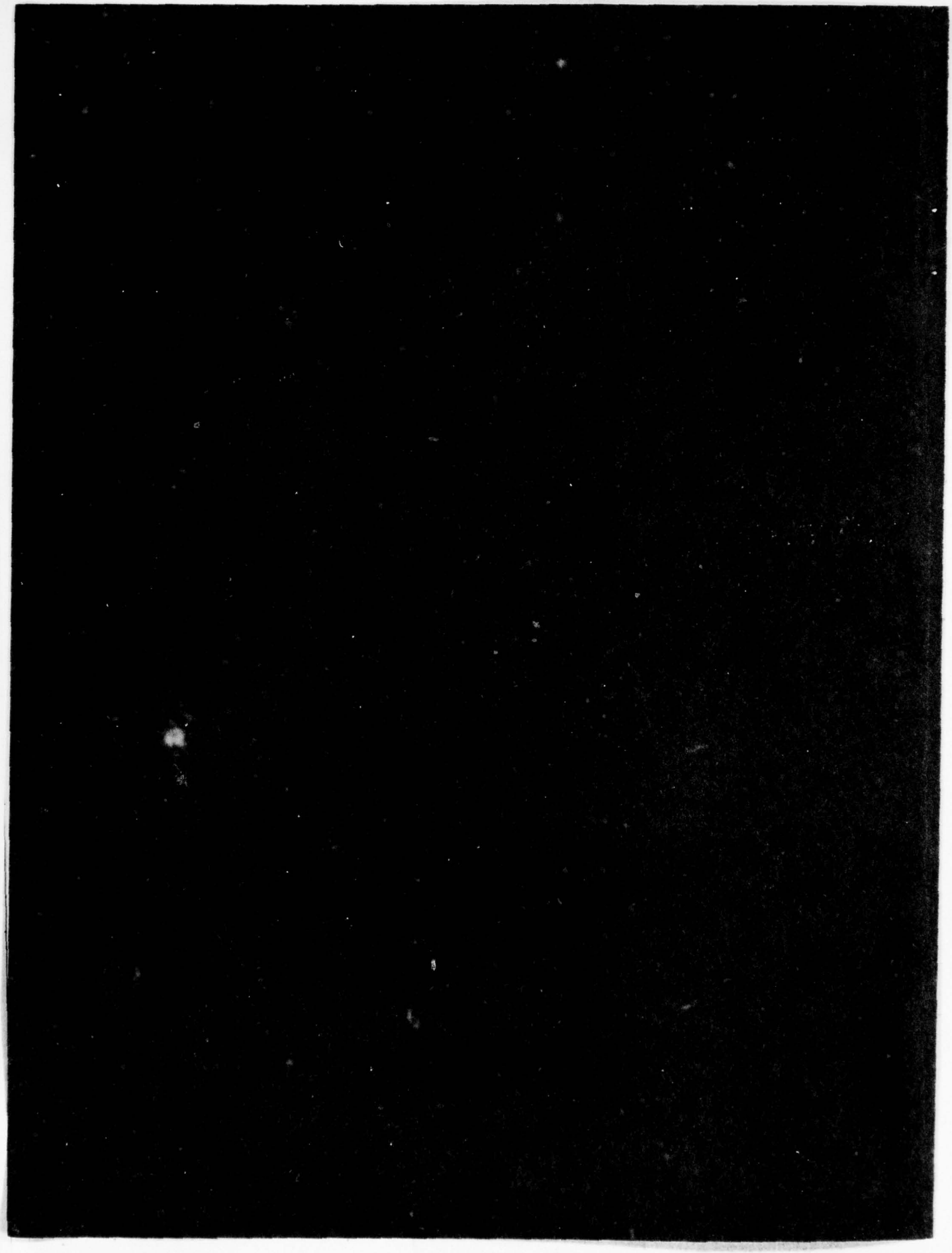
AD-A062 107 NAVAL SURFACE WEAPONS CENTER DAHLGREN LAB VA F/G 17/7
AUTOMATIC TARGET DESIGNATION.(U)
OCT 78 R D HILTON
UNCLASSIFIED NSWC/DL-TR-3906 NL

| OF |
AD A062107

END
DATE FILMED
3-79
DDC


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5
5.0
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A



UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: NSWC/DL-TR-3906
2. GOVT ACCESSION NO.
3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle): AUTOMATIC TARGET DESIGNATION
5. TYPE OF REPORT & PERIOD COVERED: Final Rept.
6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s): Richard D. Hilton
8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Naval Surface Weapons Center (K61), Dahlgren Laboratory, Dahlgren, Virginia 22448
10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: 62766N:ZF61312:ZF61312001
11. CONTROLLING OFFICE NAME AND ADDRESS: Chief of Naval Material, Department of the Navy, Washington, DC 20360
12. REPORT DATE: October 1978
13. NUMBER OF PAGES: 65
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)
15. SECURITY CLASS. (of this report): UNCLASSIFIED
15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report): 12 63 pp

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

16 ZF61312 17 ZF61312001

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Television Guidance
Automatic Target Designation
Terrain Models
Computerized Simulation

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

Television guidance systems have traditionally relied upon a human operator to designate a target before weapon release. This study uses computer simulations, which include a simulated terrain model, to evolve on-board hardware techniques for automatically designating targets after weapon release. Projectiles that use 128-line television guidance that automatically designates moving objects in a land target area are successfully simulated.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-LF-014-6601

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

391 598

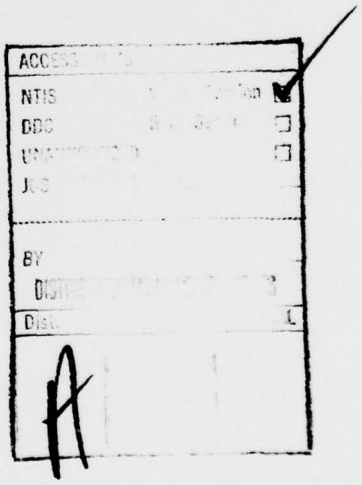
FOREWORD

Automatic Target Designation was a FY 1978 IED project designed to illustrate the applicability of television guidance technology to a "fire-and-forget" guided projectile.

This report was reviewed by C. T. Shelton, Head, Computer Engineering Branch and C. J. LaMonica, Head, Computer Facilities Division.

Released by:

R. A. NIEMANN, Head
Strategic Systems Department



CONTENTS

ILLUSTRATIONS

SI CONVERSION

Multiply	By	To Obtain
foot	0.3048	meter
mile	1.6094	kilometer
yard	0.9144	meter

AUTOMATIC TARGET DESIGNATION/MOVEMENT DETECTION

The goal of the automatic target designation effort is to overcome the limitation usually imposed by television guidance systems which require a human operator to place a curser on an object to be designated as a target. With the cursor in place, television guidance systems have been proven successful in guiding to a target without further assistance. A method is sought by which television camera output may be processed so as to designate a target without human intervention.

The process we seek, however, must take the place of a human brain that is making judgments based upon visual perception. The quantity of data available from a television sensor is very high (typically around 10^8 bits per second), and the processing is quite complex; thus, the search was approached in the following manner:

1. To attempt to process data so as to discriminate an object or objects that are moving with respect to a complex background of stationary features (land targets)

2. To carry out the search in the form of computer simulations (avoid the use of special facilities such as the Army's terrain model at Huntsville, Alabama)

3. To use simplified vehicle motions such as a simple trajectory with air drag but without vibrations or vehicle roll

4. To process data in such a way as to simulate configurations of hardware components that are either available now or are anticipated in the near future

5. To outline broadly any design features that may not be standard practice

Figure 1 illustrates the broad outline of the computer simulation. The blocks labeled *trajectory*, *optics*, *target*, and *background* are encompassed in a single program named VTRAJ3 which has the objective of providing simulated video signals from which target designations are extracted. With a number of simulated flights on disk file, other programs are used to process and edit the video data. The *data reduce* and *printout* functions are accomplished through the use of a simple transcribing program and the Stromberg Carlson 4060 Cathode Ray Tube Graphics system. All output is rendered on 35mm film and is used as a projection slide or is printed by the Photography Lab.

A process capable of designating moving targets has been found. The following sections describe the programs and processes in detail, and the final section evaluates the findings in quantitative terms.

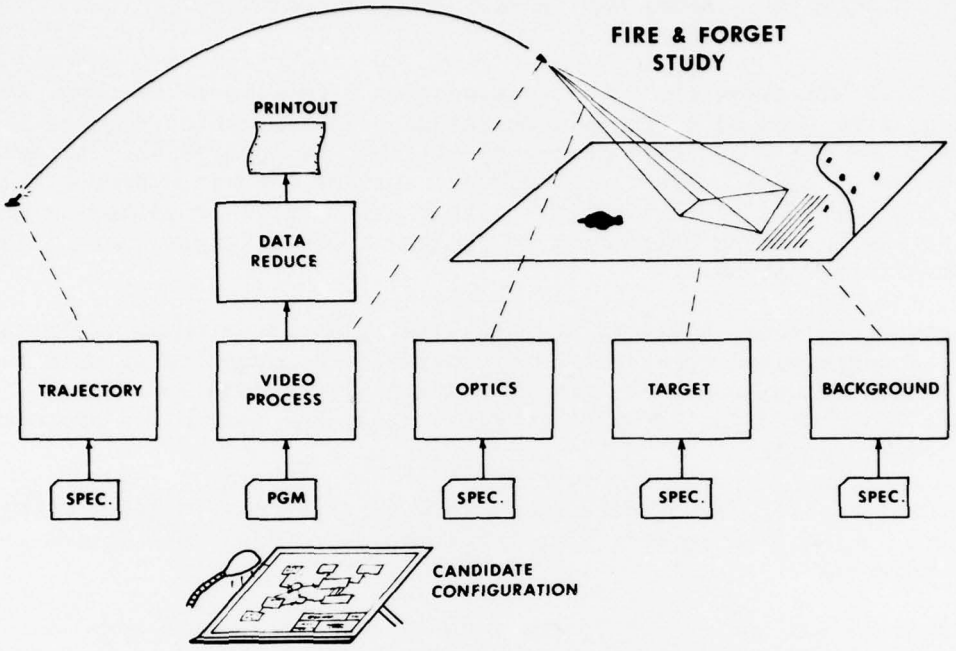


Figure 1. Outline of the Computer Study

VTRAJ3--A PROGRAM TO GENERATE A VIDEO SIGNAL

The Fortran program designated VTRAJ3 was written to generate files of data representing the video signal that would eminate from a 128-line TV camera mounted in a projectile that is descending along a ballistic path onto a terrain target area. The camera is mounted so as to look along the velocity vector; and, it is assumed that the vehicle is roll stabilized with the camera upright. There is a provision for superimposing a moving target upon the surface of the terrain.

TERRAIN MODEL

A terrain model was required to present the moving target against a realistic background. Since no physical terrain model was readily available, a terrain model simulation was built into the computer program. The model consists of a digitized mosaic representing a rectangular area 1920 by 4800 yd. Each square yard within this area is assigned a single octal-digit gray level with 0 representing black and 7 representing white. The mosaic of gray squares forms an isometric drawing of a terrain model with features portrayed as seen from above, looking down at a 45° angle.

In use, the terrain model is a table-look-up subroutine which is given integer (I, J) coordinates in yards and returns with the corresponding gray

level. To avoid excessive memory requirements, the table-look-up is completed in two steps.

The entire terrain model area is represented by a two-dimensional array called MAP (48, 120). Each element in MAP represents a square area, 40 yd on a side. The representation is a number in the range $(0 \text{ to } 77)_8$ which designates one of 64 standard subpictures. Each subpicture is a table in which each of the 1600 one-yard squares represented is a single octal digit. The octal digits are packed 20 to a 60-bit CDC 6700 word so that the entire subpicture set is contained in an array called NPIX (80, 64).

Figure 2 is a plan sketch of the terrain model. Each of the graph paper squares shown represents a subpicture. The sketch within each square is a rough rendering of one of the 64 standard subpictures. A corresponding set of data cards was then made with the input format containing 24 array elements per card. The entire array was punched on a deck of 240 cards. Figure 3 shows a section of the data preparation. The first 120 cards contain data for the left half of the terrain model, and the second 120 cards contain data for the right half. The card deck is given the permanent file name MAP and the local file name TAPE2. The file is read under control of the following Fortran statements:

```
READ (2,2)((IMAP(I,J),I=1,24),J=1,120)

READ (2,2)((IMAP(I,J),I=25,48),J=1,120)

2 FØRMAT (24Ø2)
```

The subpictures began as sketches drawn in groups of four on graph paper with each square representing 1 yd^2. The sketches were then digitized by sight onto keypunch forms and were made into a permanent file named SUBPIX. Each subpicture occupies 40 cards. Figure 4 is a typical sketch, containing roads and houses, of four subpictures together with a 16-sketch summary for the group. Figure 5 shows a section of the data preparation. File SUBPIX is given the local name TAPE1 and read under control of the following Fortran statements:

```
READ(1,1)((NPIX(I,J),I=1,80),J=1,64)

1 FØRMAT (2Ø20)
```

Figure 6 shows the entire array of subpictures. The pictures were produced, six to a frame, on the 4060 with a separate program designed to aid in the job of correcting errors in the data. The program, named DISPLAY, appears in Appendix A.

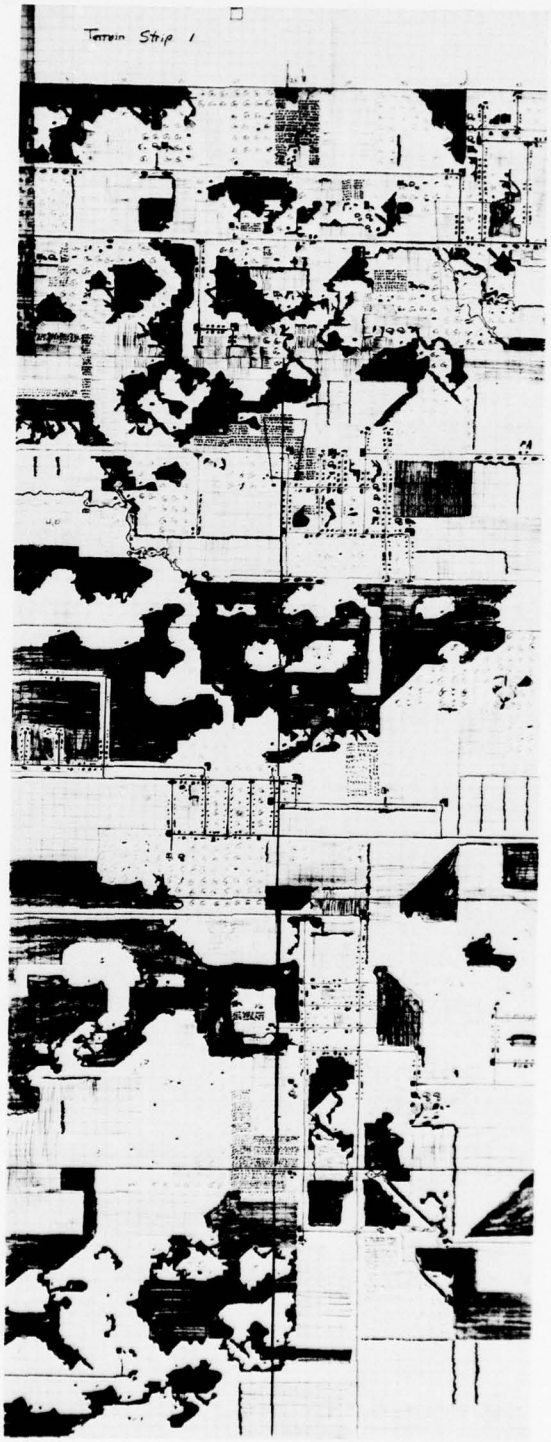


Figure 2. Plan Sketch of the Terrain Model

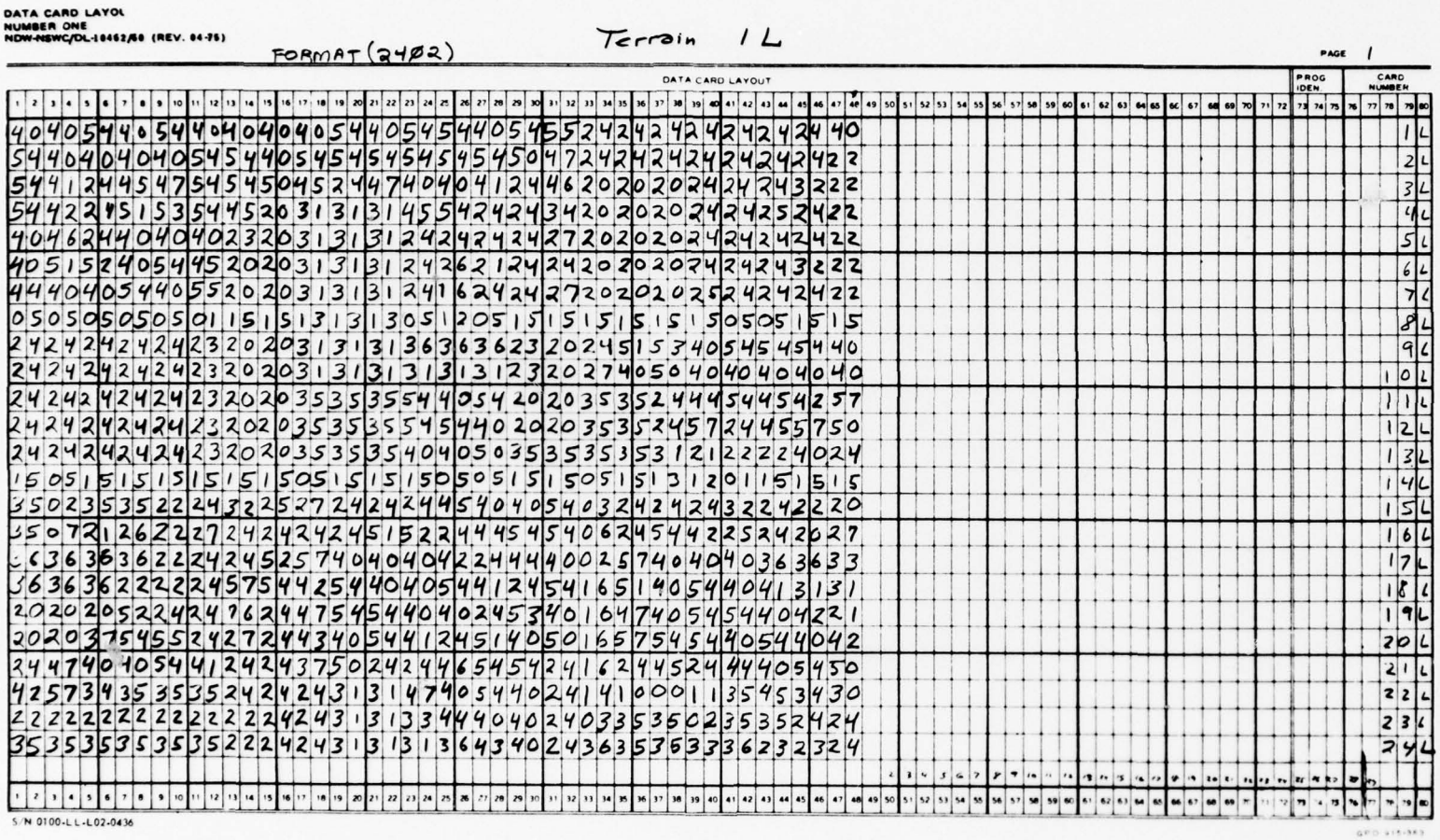


Figure 3. Data Card Representing a Portion of the Terrain Model MAP

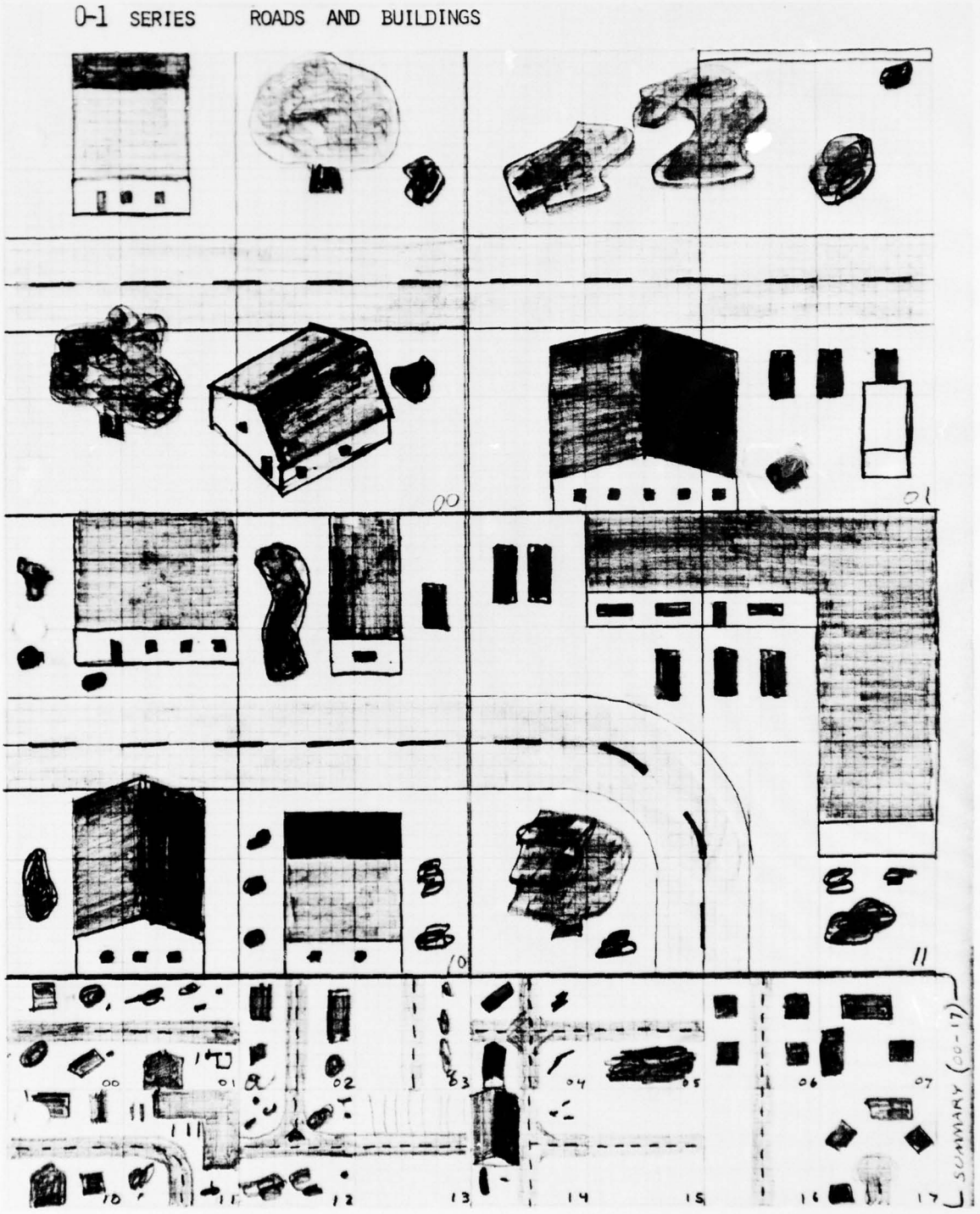


Figure 4. Typical Sketch Used to Formulate Subpictures

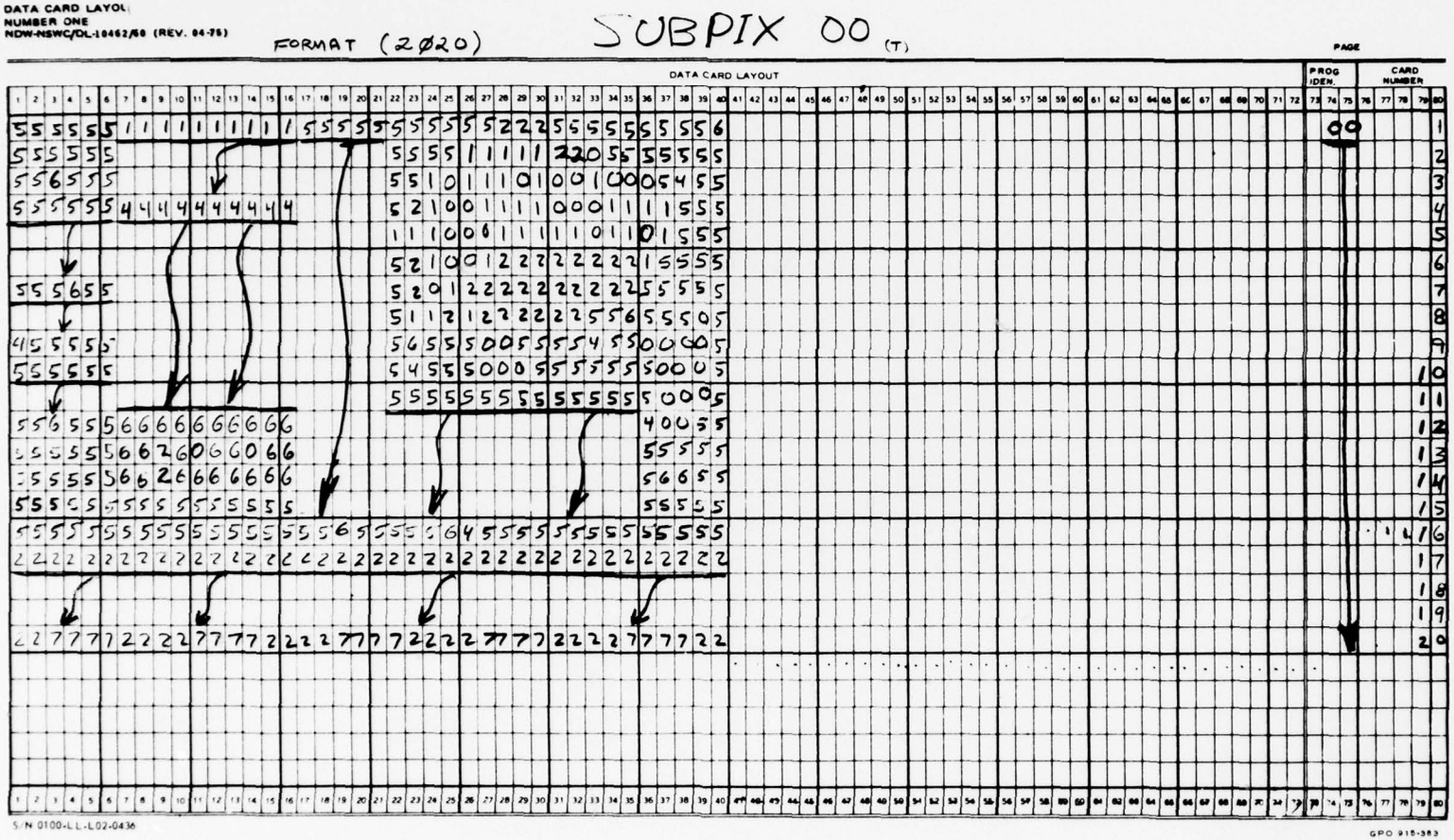


Figure 5. Data Card Representing the Upper Half of Subpicture 00

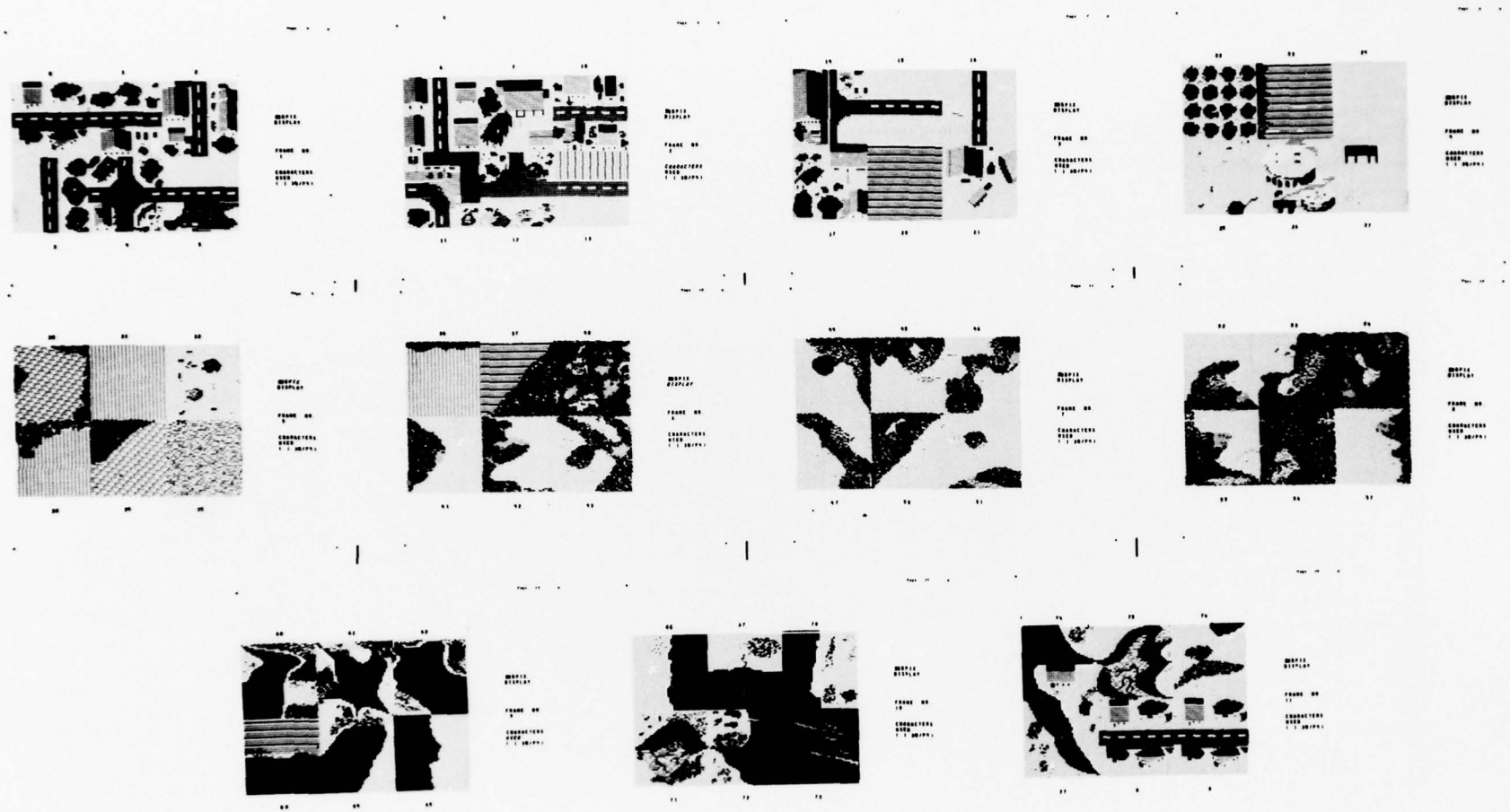


Figure 6. Output of Program DISPLAY Showing the 64 Subpictures

In use, the terrain model is accessed by way of a subroutine named TLU (IYDS, JYDS, JGRAY). It is entered with rectangular coordinates in integer yards measured from the upper left-hand corner of the model (across is IYDS and down is JYDS). In the first stage of the table-look-up, it is determined in which element of MAP the desired point lies, and the subpicture designation number is retrieved. In the second stage, the proper point within a subpicture is resolved, retrieved, and unpacked into a single digit for return as variable JGRAY. This subroutine appears in the appendix together with the rest of program VTRAJ3.

OPTICS SIMULATION

Another major block in the simulation is *optics*, wherein a subroutine named PHOTO (X, Y, Z, VX, VZ) provides the function of rendering a picture of the terrain model (possibly including a target) as seen from a point above the model specified by coordinates (X, Y, Z). The optic axis of the view is assumed to be along the vehicle velocity vector. The velocity vector is specified by VX and VZ which are normally negative quantities. The program does not allow for a Y-component of the velocity vector or for a roll angle for the camera away from the upright position. Figure 7 shows the various coordinates.

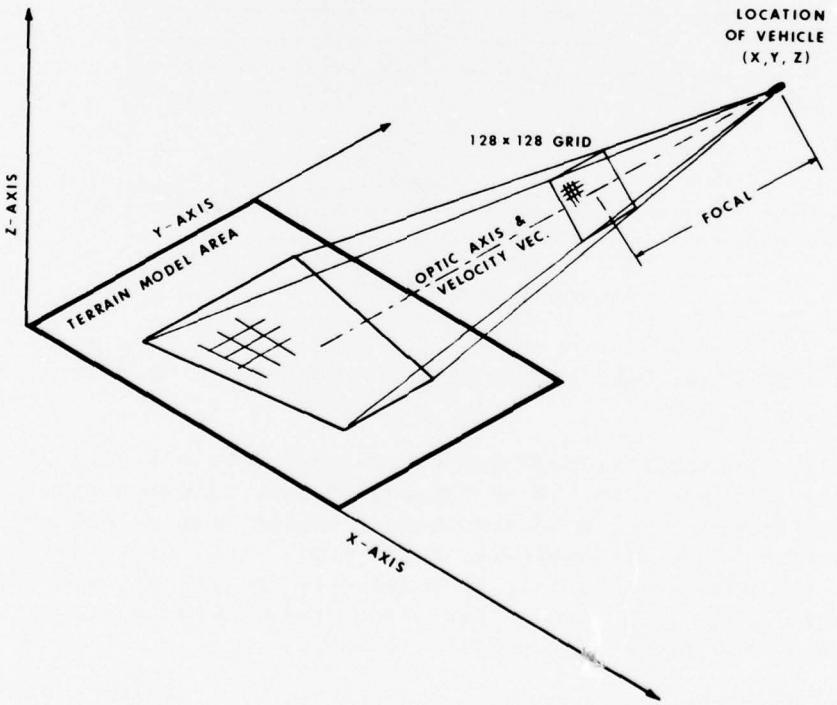


Figure 7. Coordinates Used for the Optics Simulation

The output of subroutine PHOTO is an unformatted write to tape. The picture format used by the simulation programs generally is a 128 by 128 matrix in which each element is a single octal digit representing gray levels in the same way as for the terrain model. In order to conserve memory, the octal digits are packed 20 to a word. The unformatted write is from a two-dimensional array named IPAK (7,128). Figure 8 illustrates how the array is interpreted as a picture. Also shown in this figure is an array of four variables, IHDR(4), which is not written out by PHOTO itself, but is expected on tape by the various other programs that use the tape as input. A value of IHDR(1)=0 is used to indicate an end-of-file condition. Also illustrated in Figure 8 is the format used when such data is transcribed to 35mm film.

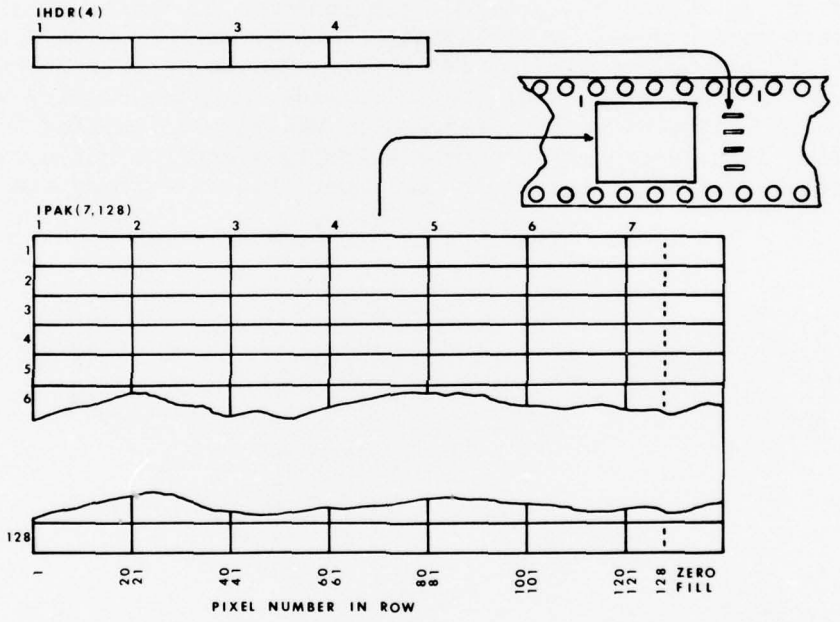


Figure 8. 128^2 Pixel Data Format With the Corresponding 35mm Film Format

Internally, subroutine PHOTO geometrically projects a grid (shown in Figure 7) having an array of 128 by 128 square apertures onto the surface of the terrain model. This grid simulates a charge transfer device (CTD) television sensor. The field-of-view angle may be controlled by the parameter FOCAL, which is the distance from the grid to the lens. FOCAL is in units of length equal to a side of the grid. This parameter is not read in but is set in the program by a DATA statement.

The four corners of each projected aperture, or a pixel in the simulated CTD sensor, are located as integer coordinate points on the terrain model. These points will generally describe a trapezoid (Figure 9). A subroutine

named TRAP then uses subroutine TLU to obtain all of the terrain model elements that comprise the trapezoid and forms an average of their values. This average becomes the gray level value of the corresponding pixel in the array IPAK. Since the sides of the trapezoid are generally not parallel to the y-axis of the coordinate system, a variable slope "staircase" algorithm is used to obtain an approximation.

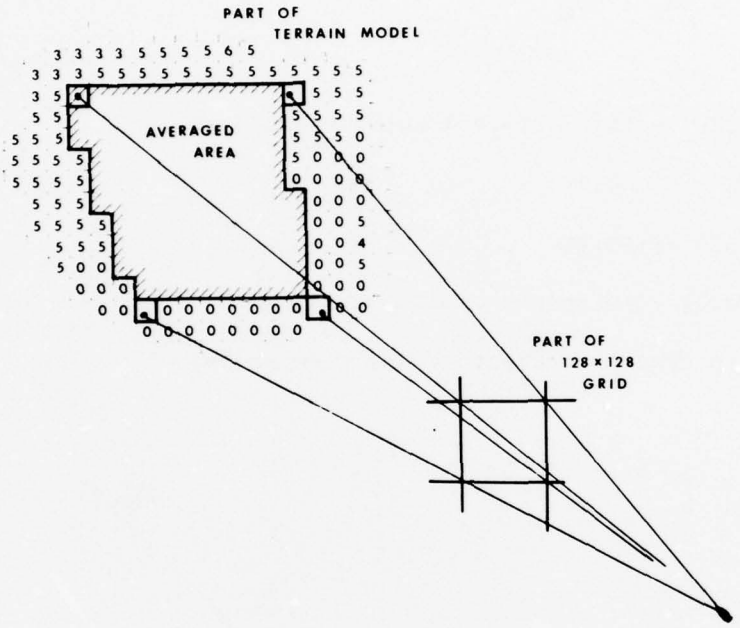


Figure 9. Trapezoidal Projection of a Pixel on the Terrain Model

TRAJECTORY SIMULATION

A simple subroutine named FLIGHT (X, Z, VX, VZ, DELT) is used to update the coordinates (X, Z) of the projectile at a time interval DELT. Since the trajectory is assumed to have no Y-velocity, this parameter does not appear. X, Z, and velocities VX and VZ are updated. One hundred iterations are performed for each entry of the subroutine. The equations used are

$$dt = \Delta t/100$$

$$\Delta V_x = - C_d \, Vx_i \, dt$$

$$X_{i+1} = X_i + \left(Vx_i + \frac{\Delta Vx}{2}\right) dt$$

$$Vx_{i+1} = Vx_i + \Delta V_x$$

$$\Delta V_z = -(32.16 + C_d\, Vz_i)\, dt$$

$$Z_{i+1} = Z_i + \left(Vz_i + \frac{\Delta V_z}{2}\right) dt$$

$$Vz_{i+1} = Vz_i + \Delta V_z$$

where

Δt = DELT = time interval between pictures

Vx = X - velocity

Vz = Z - velocity

C_d = drag coefficient (set to 0.04)

Figure 10 shows a typical trajectory and view angle.

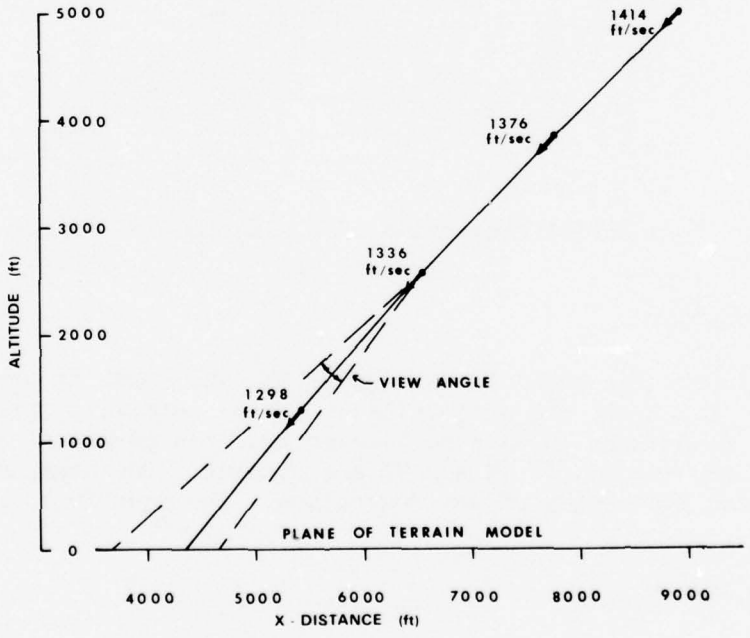


Figure 10. Typical Trajectory and View Angle

MOVING TARGET

A simple provision in the basic terrain model subroutine TLU is used to superimpose a black square moving object onto the scene. The size and location of the target is determined by the three variables (IT, JT, and NSIZE) which are in COMMON storage. NSIZE is the size of the square target (given in yd^2). The coordinates IT and JT are updated by a subroutine CRAWLR which simply updates the position of the square according to straight-line velocities and the same time interval used by subroutine FLIGHT. The actual superimposition occurs in subroutine TLU.

A full Fortran listing of program VTRAJ3 and a set of operating instructions are found in Appendix A. Figure 11 shows a typical video data file. There are 39 pictures taken at 0.12-sec intervals along the trajectory of Figure 10. A column of header information is beside each picture. These variables are (starting at the top) Frame Number, X-coordinate (ft), Y-coordinate (ft), and Z-coordinate or altitude (ft). A moving target is shown in the pictures. It is a black square, 15 ft on a side, moving up and to the right at a speed of 29 mi/hr. It appears in the upper left-hand corner of frame 33 and can be traced from there.

The output file of VTRAJ3 is transcribed into Stromberg-Carlson 4060 (and then to photographic format) by a program named MOVIE. The listing and instructions for this program also appear in Appendix A.

TARGET DISCRIMINATION

Once data files containing sequences of pictures (such as those of Figure 11) are established, the next task is to find a process that will discriminate moving objects from surrounding stationary clutter. If the observation of a target area could be made from a stationary position, moving target discrimination would be very easy. It would be necessary only to

1. Take a picture
2. Store it
3. Wait a suitable time interval
4. Take another picture
5. Compare the new picture and the stored picture, pixel by pixel.

Those objects in motion would show up prominently in the comparison process.

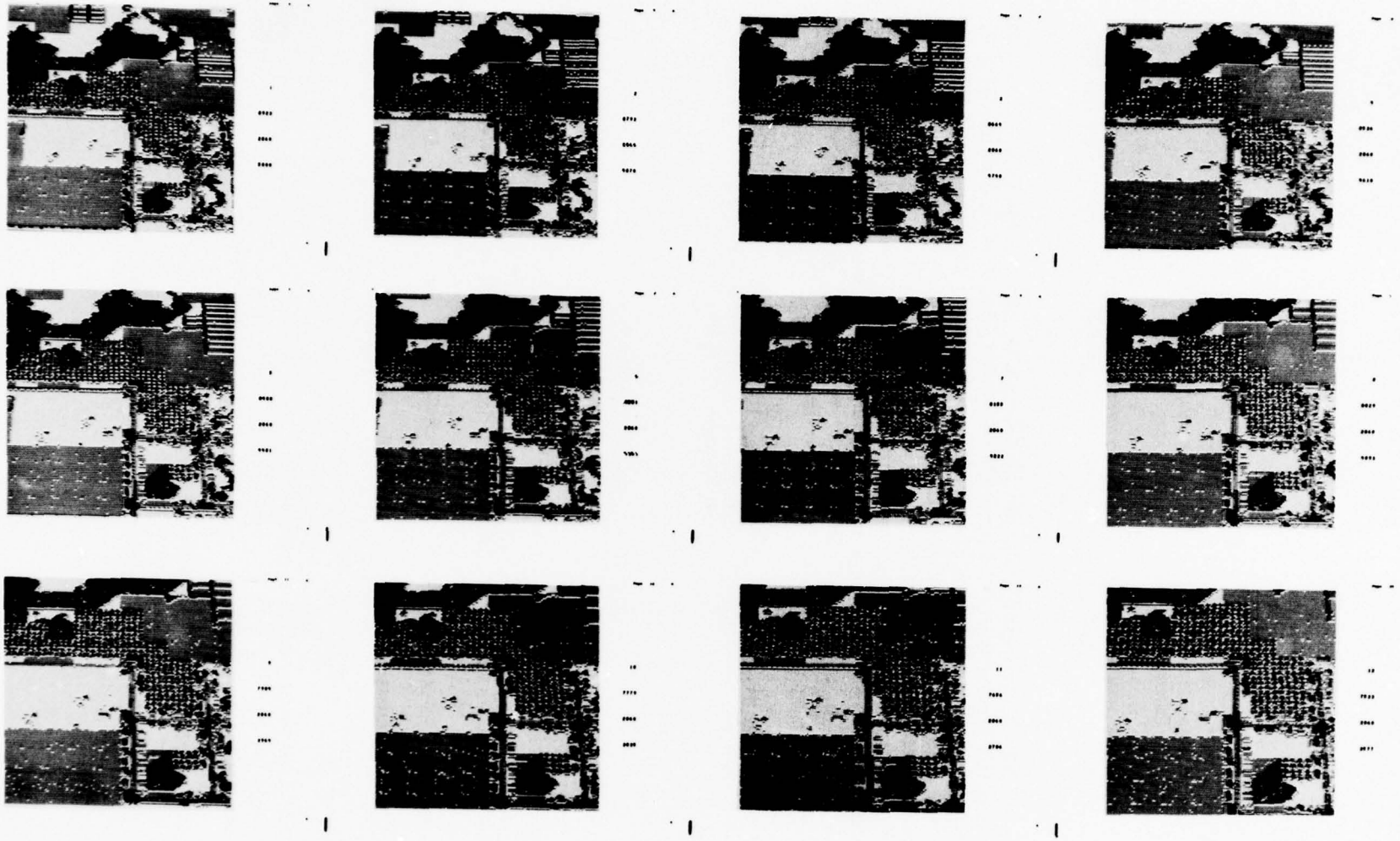


Figure 11. Computer-Generated Pictures of an Area Containing a Moving Target

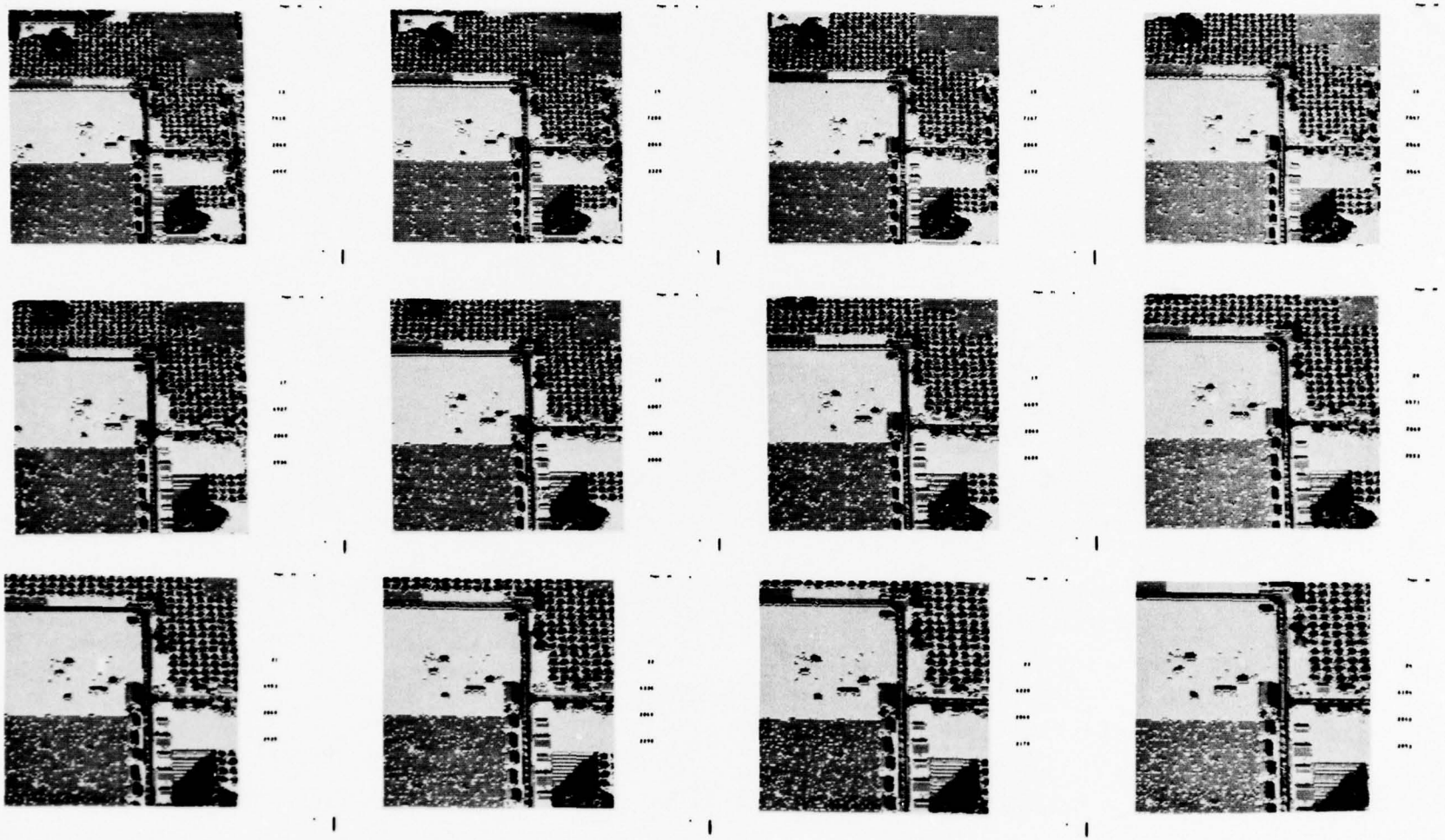


Figure 11. Computer-Generated Pictures of an Area Containing a Moving Target (Continued)

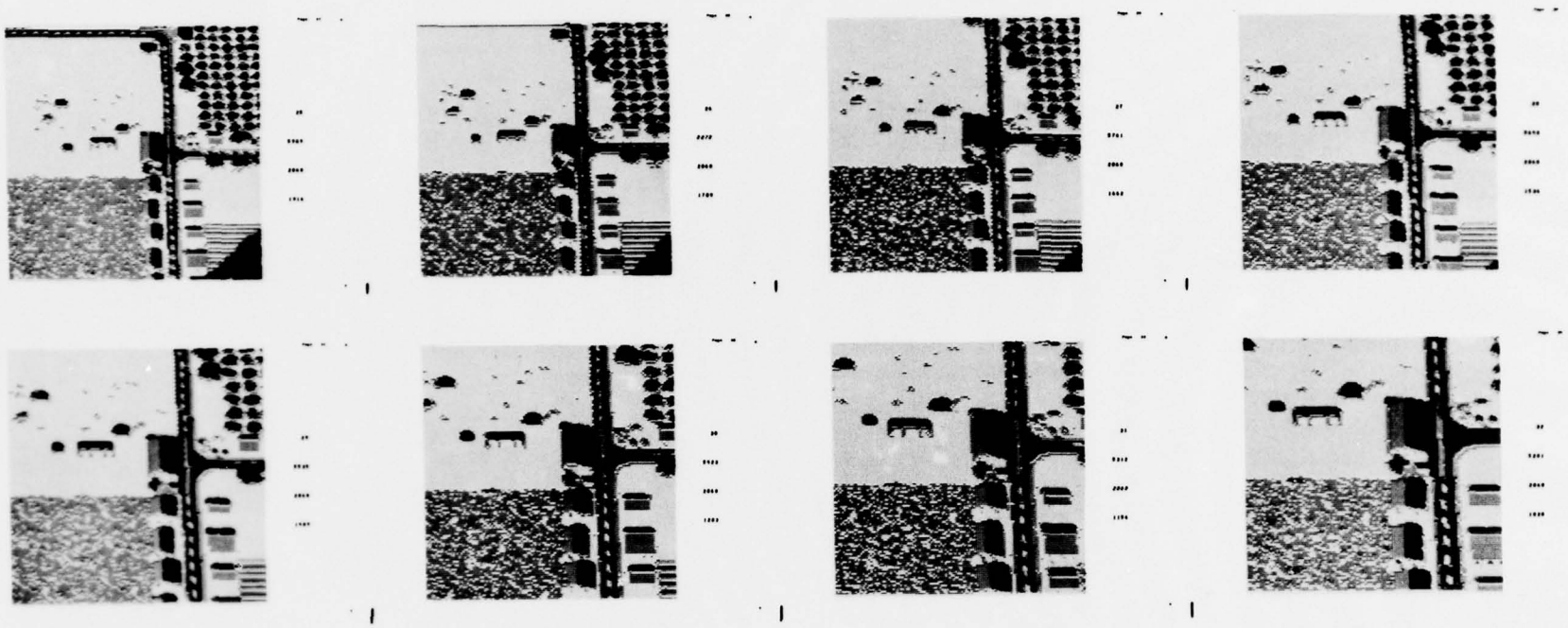


Figure 11. Computer-Generated Pictures of an Area Containing a Moving Target (Continued)

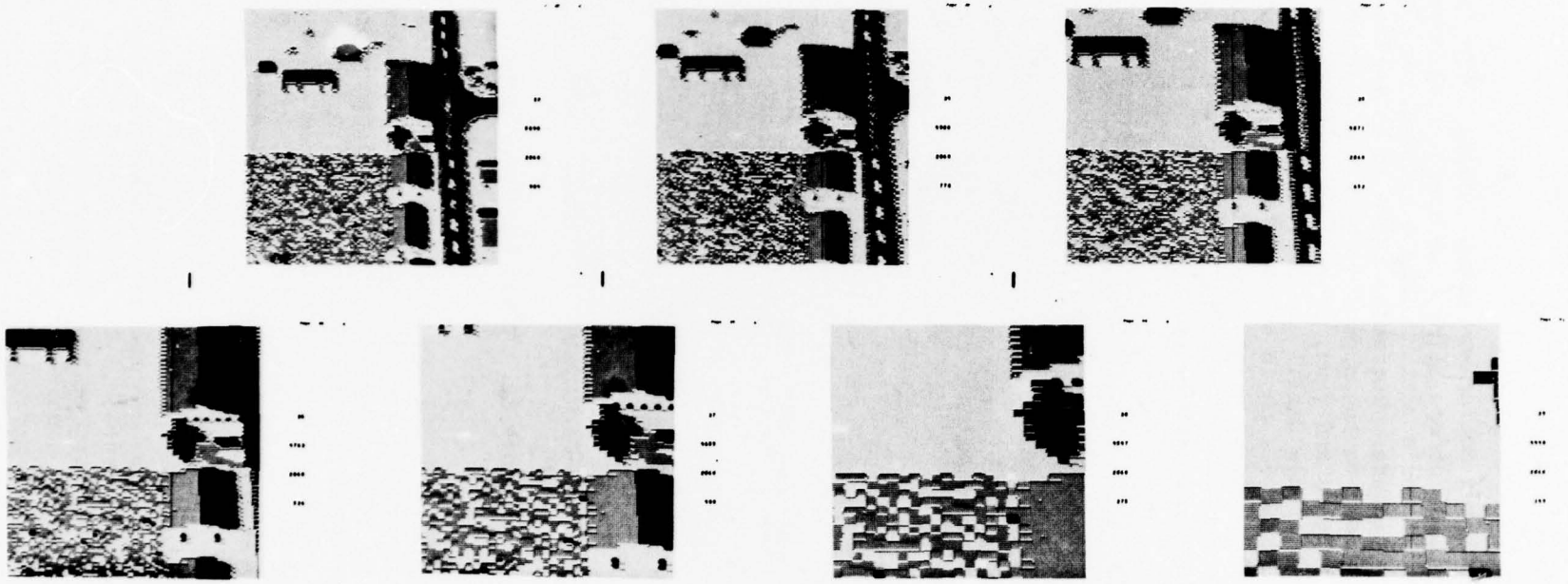


Figure 11. Computer-Generated Pictures of an Area Containing a Moving Target (Continued)

However, hovering motionless while looking for a target is not practical, and a process capable of operating from a moving observation platform must be found. The observer's motion along a ballistic trajectory produces several effects upon the picture. Foremost, there is an enlargement process as the observer comes closer to the ground. There are also shifting effects as the optic axis pitches and yaws, as well as quantization noise brought on by the appearance of new detail at lower altitudes. It was initially assumed that a correction process could be found whereby the above five steps could be augmented with an image transformation step that applies corrections to the stored image and compensates for the effects of observer motion. Figure 12 shows the video signal path to be simulated.

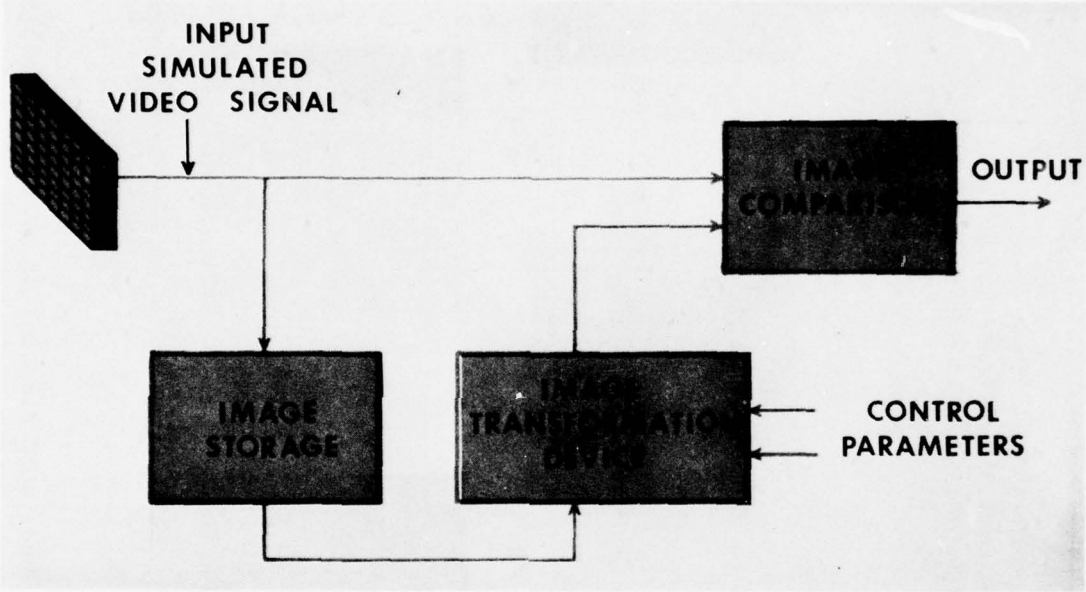


Figure 12. Video Processor Simulation

An appreciation for the nature of the required image transformation required can be gained from Figure 7. The picture taken is that of a trapezoidal "footprint" on the ground. As the trajectory progresses, motion along the optic axis will cause the footprint to shrink to a similar but smaller trapezoid. The effect upon the picture is an enlargement and an apparent motion away from the optic axis of all objects in the field of view. Some objects will disappear from view and some fine details will be resolved for the first time.

There are other components to the observer's motion. While rotation about the optic axis has been ruled out, the optic axis can still pitch over to a steeper angle of descent. The main effect from a pitching motion is an upward vertical displacement of all the objects in the picture. This shift is accompanied by second-order distortions that result from the fact

that pitch causes the new trapezoidal footprint to be somewhat dissimilar from the original. This dissimilarity means that the enlargement of the scale of the picture is not uniform.

For simulation purposes, the second-order effects are ignored. An algorithm for them has not been devised, and such a step may prove unnecessary. The *image transformation device* (Figure 12) is capable of enlarging a picture and shifting it upward. Two considerations that remain are

1. How are standard electronic components to be arranged to carry out these transformations?

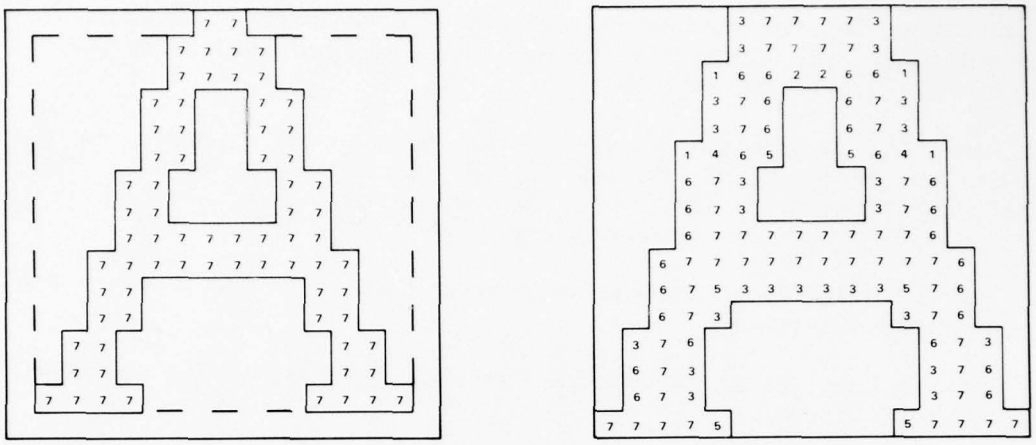
2. How are the control parameters (how much enlargement and how much shift) to be generated?

First, the electronics and their computer simulation will be treated. While a microprocessor is assumed to be a part of the electronic mix, it cannot be reasonably assumed that a microprocessor (or even a fairly large computer) capable of handling the quantity of data generated by a television sensor will appear in the near future. The microprocessor in the system simulated in this study will be a "supervisor" that performs numerical computations in support of other electronic devices that actually process video data.

The system simulated assumes that video information is generated, stored, and processed in long analog shift registers. Given such a storage medium for video data, image shifting, horizontal, and vertical is easily accomplished by supplying the storage shift register with extra shift pulses after the entire image has been read in. An image magnification process will prove to be more trouble.

Figure 13 illustrates the processing needed when an image to be magnified is stored in a long shift register. The upper portion of the figure contains a 16 x 16 matrix (folded 256-bit shift register) containing the digitized image of the letter "A". Superimposed on this matrix is another, slightly smaller, 16 x 16 matrix representing the area that is to be expanded into the full-size picture format. The lower portion of Figure 13 shows the result of the magnification process. Each pixel in the output matrix is computed from the weighted contents of up to four pixels in the original matrix.

The simulation uses a subroutine with the Fortran name of MAGFY (IPAK, JPAK, ISHIFT, JSHIFT, MAGGIE). The purpose of this subroutine is to simulate the *image transformation device* (Figure 12). Its action is to fetch a picture from IPAK (7,128), shift it in the I (horizontal) direction by ISHIFT pixels, shift it in the J (vertical) direction by JSHIFT, increase its scale by an amount controlled by MAGGIE, and place the result in JPAK (7,128).



15^2 Segment 16^2 Segment

Figure 13. Sample Magnification Process of a 15^2 Segment Expanded into a 16^2 Segment

Figure 14 illustrates the magnification process on a pixel level. It shows two superimposed grids representing the pixel array of an input picture (the larger and darker grid) and the pixel array of an output or magnified grid. The magnification occurs as the contents of the smaller squares are reinterpreted in a framework that has the same dimensions as the input grid. The process by which the contents of a given output pixel is formed follows:

I,J = indices specifying the pixel being filled in the output picture

i,j = indices specifying the pixels being accessed in the input picture

f(I,J) = contents of pixel (I,J)

g(i,j) = contents of pixel (i,j)

M = magnification to be applied

Then to compute f(I,J) (illustrated by the shaded square in Figure 14), the machinery simulated or the computer algorithm must find:

$$i = \text{integer part of} \left(\frac{I}{M}\right)$$

$$j = \text{integer part of} \left(\frac{J}{M}\right)$$

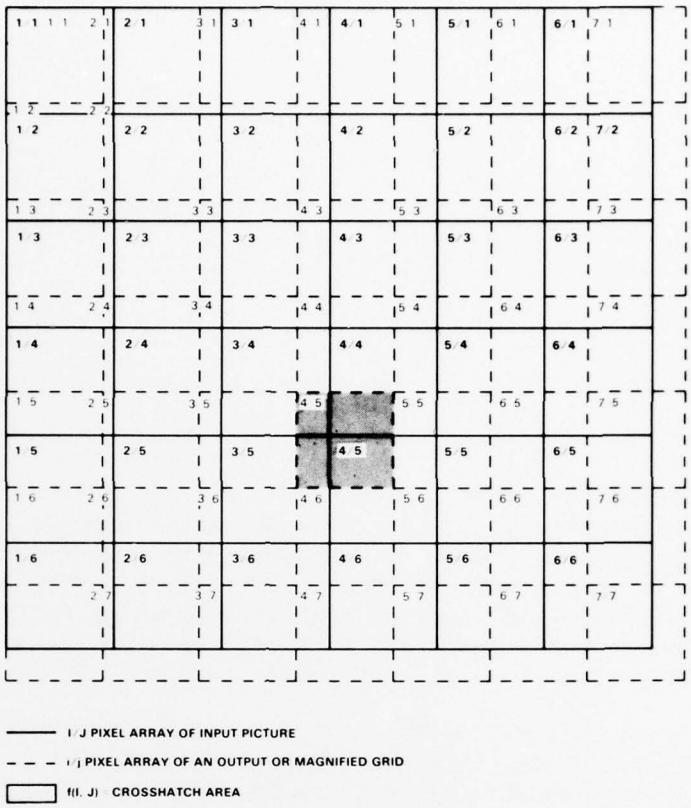


Figure 14. Superimposed Pixels From an Input Picture and a Magnified Output Picture

$$T = M \cdot \left[\text{fractional part of } \left(\frac{I}{M}\right)\right]$$

$$U = 1 - T$$

$$V = M \cdot \left[\text{fractional part of } \left(\frac{J}{M}\right)\right]$$

$$W = 1 - V$$

$$f(I,J) = U \cdot W \cdot g(i,j) + T \cdot W \cdot g(i,j+1) + U \cdot V \cdot g(i+1,j) + T \cdot V \cdot g(i+1,j+1)$$

In the shaded area illustrated in Figure 14:

(I,J) = (4,5)

M = 1.11111...

(i,j) = (3,4)

T = 0.66666...

U = 0.33333...

V = 0.55555...

W = 0.44444...

Figure 15 outlines a possible configuration of analog shift (CCD) registers and high-speed sequencing logic to effect the above process. Figure 16 shows a sequence of pictures that have been processed by subroutine MAGFY. The picture at top center was used as input. The sequence of pictures below show the results of applying various control parameters. The column of parameters beside each of the transformed pictures represent sequence number, horizontal shift, vertical shift, and magnification (128/MAGGIE).

It can be seen that as the shift quantities are held constant, the increasing values of magnification (decreasing 128/MAGGIE) cause the scale of the picture to expand and the features to enlarge. This sequence bears some outward resemblance to earlier sequences wherein each new picture was rederived from the terrain model.

With simulations in hand for the various black boxes of Figure 12, the next step undertaken was to develop a process for supplying the proper values of magnification and shift parameters to the image transformation device. These values, which must properly compensate for vehicle motion, should not depend upon specific internal information of the trajectory being followed. Such data could only be supplied by a data link or by expensive internal sensors.

The process evolved in the simulation effort is one that finds proper values for the parameters by experimentation. Two quantities, vertical shift and magnification, are optimized for the best vehicle motion compensation attainable. The quantity used to score the various experimental results is the correlation between the stored, transformed picture and a picture taken after a time interval (0.12 sec).

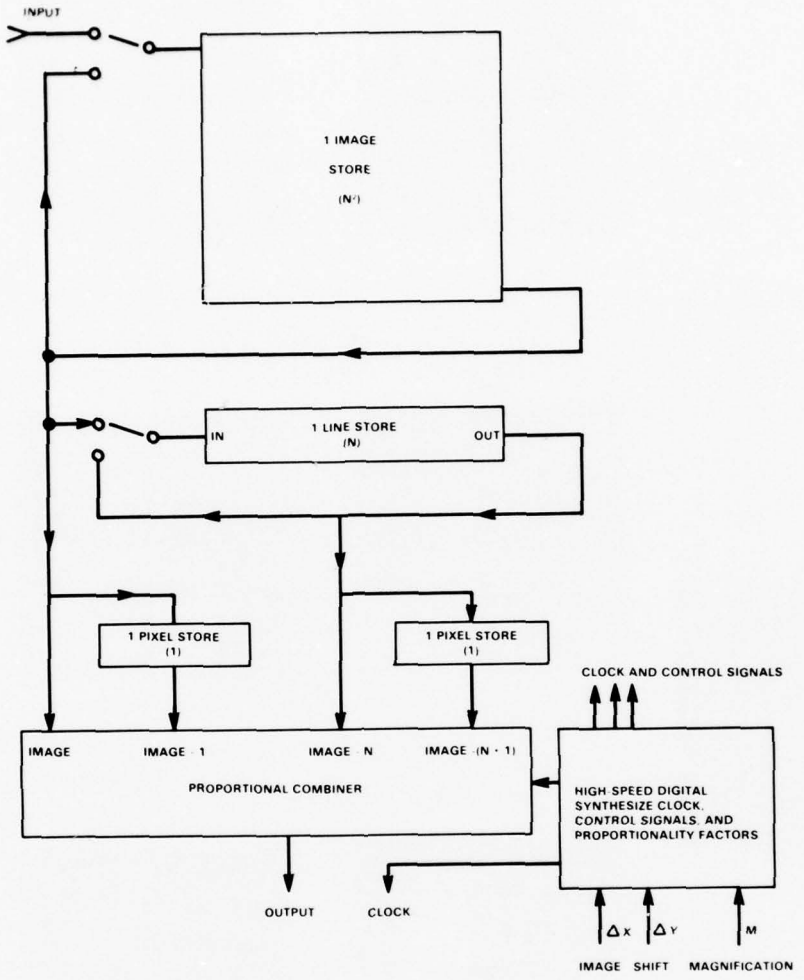


Figure 15. Hardware Array to Effect Image Transformation

Correlation ϕ is an index by which two pictures are compared pixel by pixel. Usually, correlation is normalized so that $\phi = 1.0$ for two perfectly matched pictures; $\phi = 0$ is an improbable result; and $\phi = 0.5$ is the expected value for two unrelated pictures. If the correlation of two similar pictures as a function of displacement is determined, typically one finds that the correlation function has the shape of a cusp, with the peak occurring at the displacement that provides the best overall match. A similar function is found with respect to scale or magnification. The exact shape of the curve is, of course, dependent upon the contents of the picture. Some kinds of pictures (e.g., ones containing a marked spatial periodicity) can greatly modify the correlation function from the typical curve. It is assumed in this simulation effort that such pictures are too atypical to be taken into account.

Figure 16. Typical Picture (Top) Operated Upon by Various Values of Magnification

The simulated process for automatically designating targets resides in a program with the Fortran name XMARK. This program accepts target data files from program VTRAJ3 and experiments with displacements and magnifications so as to find the best correlation between a stored, transformed picture and a picture taken at a later time. It then subtracts the best correlated pair of pictures it can find, and searches the difference picture for targets. If it finds a target, it superimposes a pair of intersecting black lines over the target in the input picture and writes it out as an output file. The program also provides a printed record of its experimental progress during the flight. Figure 17 illustrates the overall data flow of a simulated trajectory and target designation.

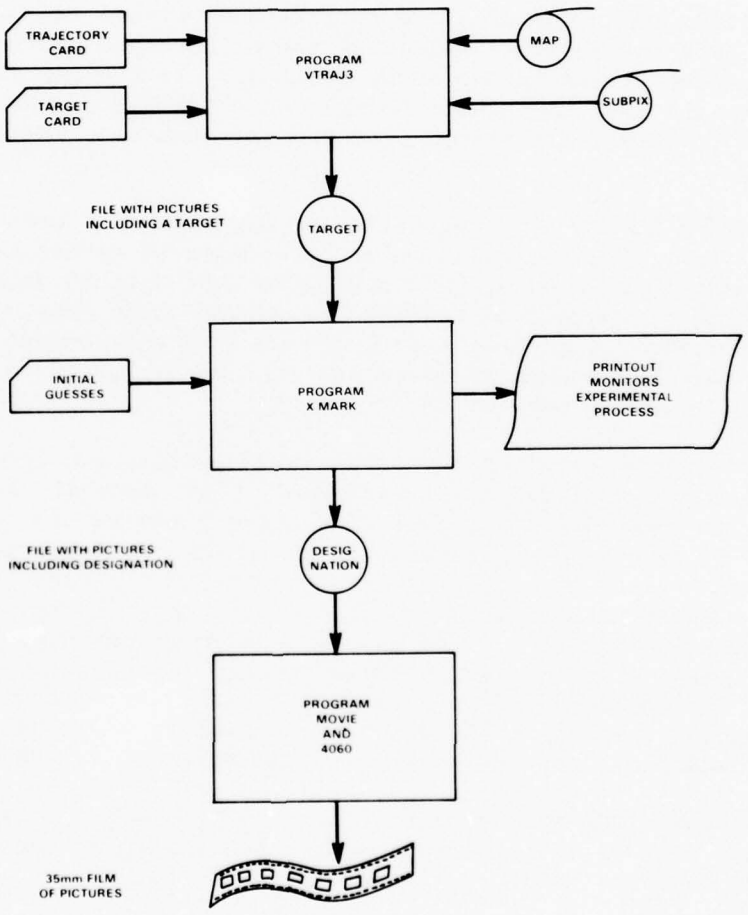


Figure 17. Data Flow of Simulation

The actual process within XMARK is a rather crude one which represents a beginning for an evolutionary process. Actually, scheduling the processing implied in real hardware might prove difficult. It is also expected that due to further refinements, such scheduling will also prove unnecessary. Figure 18 is a flow chart of XMARK. Initially the program reads in one "old" picture from the target file and one card containing a beginning value for the two critical parameters JJO = vertical shift and MAGO = magnification. The program then enters into a loop wherein it reads in a "new" picture from the target and enters the sequence of computations in Figure 18. There are calls to several subroutines whose functions are outlined below.

Subroutine MAGFY, which has already been described, simulates the image transformation device. Its calling sequence contains three parameters: ISHIFT, or horizontal shift in integer pixels; JSHIFT, or vertical shift in integer pixels; and MAGGIE, or magnification factor. In program XMARK, ISHIFT is always zero since yaw in the projectile has been disallowed.

Subroutine CORR2 computes the correlation between two 128 by 128 pictures. It adds the absolute values of the differences between the contents of corresponding pixels in two input pictures. It returns with the raw summation as SUM, and also computes a normalized correlation COEF which reaches 1.0 for two identical pictures. Only the raw summation SUM is used in program XMARK.

Subroutine CURFIT is a curve-fitting algorithm for use with trial values of correlation. Since correlation as a function of either magnification or shift tends to be a cusp, this subroutine attempts to determine the best value of those parameters by finding the peak of that cusp. Figure 19 shows a typical correlation cusp as a function of some independent variable. In practice, that independent variable can only be evaluated for integer values given the quantized nature of the pictures.

The subroutine receives three values of correlation from three evenly-spaced values of the independent variable i. It computes the coefficients for a second-order polynomial that fits those three points, and then computes the second derivative of that function at the center value of i. If the second derivative is positive (as it would be for the three values designated by X on Figure 19), it computes the intercept of the polynomial and the line where correlation = 1.0 and returns an incremental value for i equal to Δ_1 (as shown). If the second derivative is negative (as it would be for the values designated by "0" on Figure 19), then the subroutine finds the nearest integer value of i for which the parabolic curve has a null first derivative and again returns an incremental value of Δ_2 (as shown).

Subroutine MINUS simply subtracts two 128 by 128 pictures to produce a difference picture. The differences of the values of the pixel pairs are rescaled to have the range (0,7).

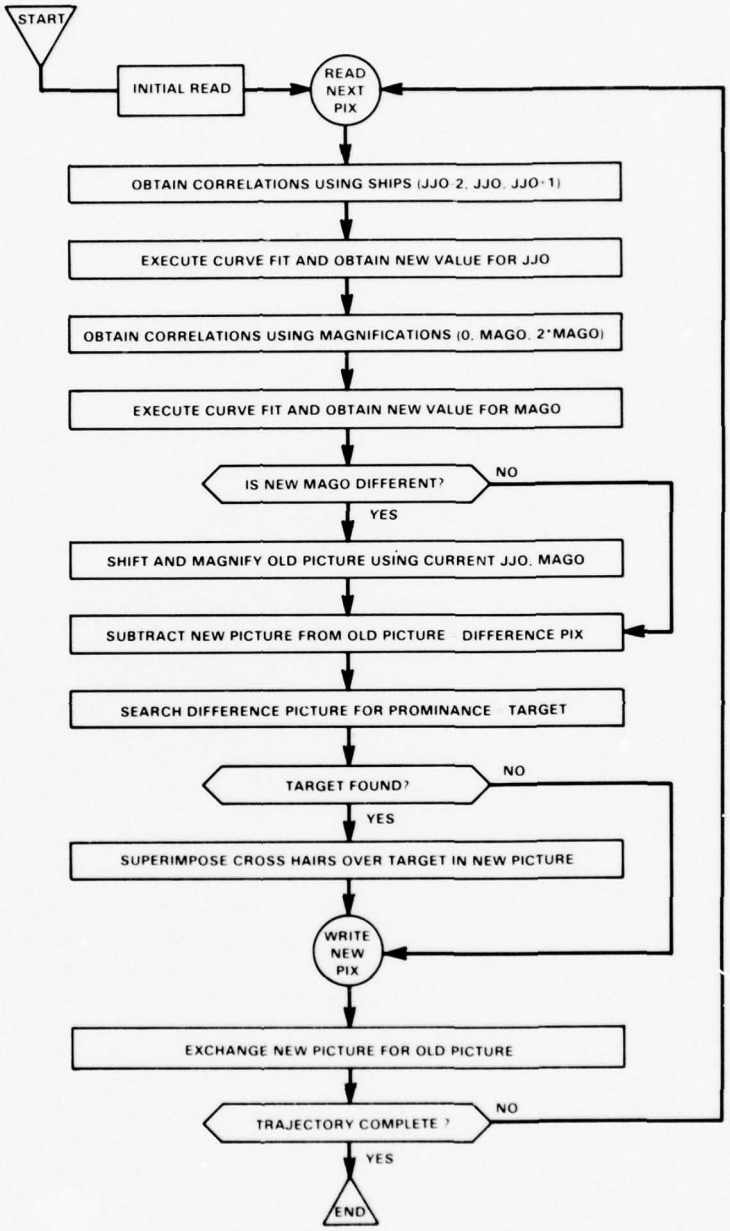


Figure 18. Flow Chart of XMARK

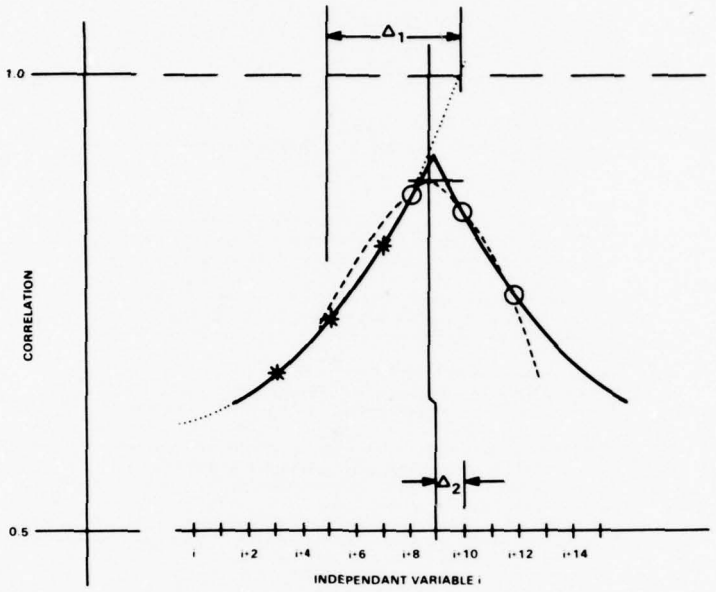


Figure 19. Correlation Function Curve Fit

Subroutine SIFT searches through a difference picture looking for a 3 x 3 matrix of values that lie outside the range (3,4). When it finds one, it generates two intersecting black lines over the spot and superimposes them over the picture that is the minuend of the subtraction process in MINUS. Subroutine SIFT also reports its find on the printer.

The main simulation program XMARK uses these subroutines (plus two other housekeeping subroutines) in a sequence (Figure 18). The program first reads in one picture from the target file and reads a card containing starting values for vertical shift (JJO) and magnifications (MAGO). It then enters a loop during which it reads in a subsequent picture, updates JJO and MAGO for the best correlation between the modified, previous picture and the current picture, subtracts the two, and searches the difference for targets.

The first step in optimizing the values of JJO and MAGO is to transform and correlate the previous picture with the present one three times, using JJO-2, JJO, and JJO+2. These three correlations are processed by subroutine CURFIT in order to obtain a corrected value of JJO. Using this corrected JJO, the program then transforms and correlates three more times using values of magnification corresponding to no magnification, the previous magnification, and twice the previous magnification. These results are also processed by CURFIT to obtain a possible correction to MAGO. If the correction obtained is nonzero, another transformation is done. The final optimized, transformed, previous picture is subtracted from the current picture and the result searched for targets by subroutine SIFT. During this procedure,

a printout occurs showing the values of JJO, MAGO used, the type of optimization undertaken (S, M, or R for shift, magnification, or recomputation), and the three values of correlation (displayed as the raw summation) obtained. If a target is found, a print line identifying the coordinates appears.

Figure 20 shows a sample picture output from program XMARK operating on the target file displayed in Figure 11. In the 25th frame, representing an altitude of 1789 ft, the simulation program correctly identified and designated the object in the field of view that was in motion.

CONCLUSIONS

Several runs of the simulation were made. The trajectory was changed only in landing area; the size of the target was maintained at 15 ft^2 and its speed was maintained at 30 mph. The target direction on the ground was varied, however. Figure 21 shows the designations made. Several false designations occurred. These were always prominent objects found near the edge of the picture. Assuming that designations near the edge of the picture are to be ignored, the targets were properly designated at the following altitudes:

Run	Simulated Altitude (ft)	Extrapolated for 512 Lines-Altitude (ft)
1	1789	7156
2	652	2608
3	1030	4120
4	904	3616
5	1283	5132
Average	1132	4528

The pixel array size simulated was, of course, only 128 x 128--a small number for CCD sensors. If the array size were increased to a state-of-the-art figure like 512 x 512, the altitudes at which designations could be expected would rise proportionally (by similar triangles) to values four times those listed. Similarly, the expected altitude for designations would be directly proportional to target size and target speed. Generally, targets in the upper portion of the picture are harder to detect than those in the lower portion, because perspective makes them appear smaller.

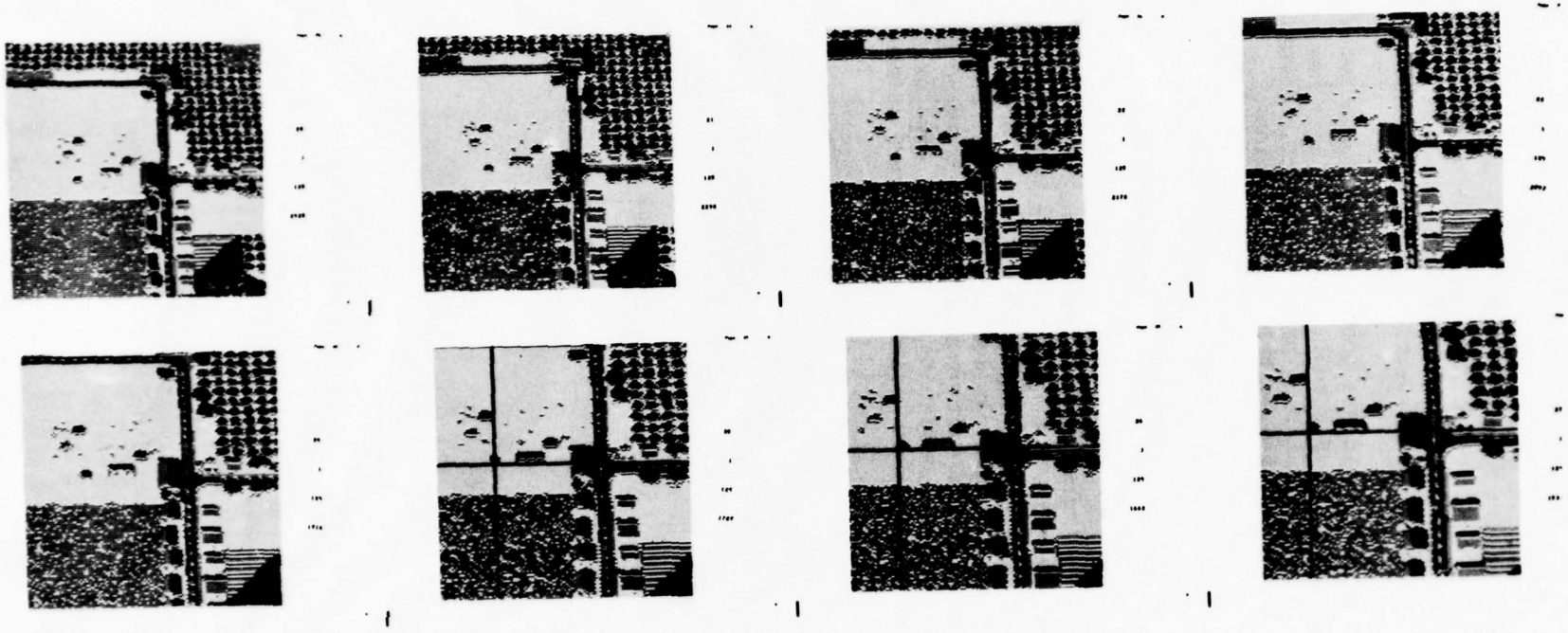


Figure 20. Output of Program XMARK Showing a Target Designation

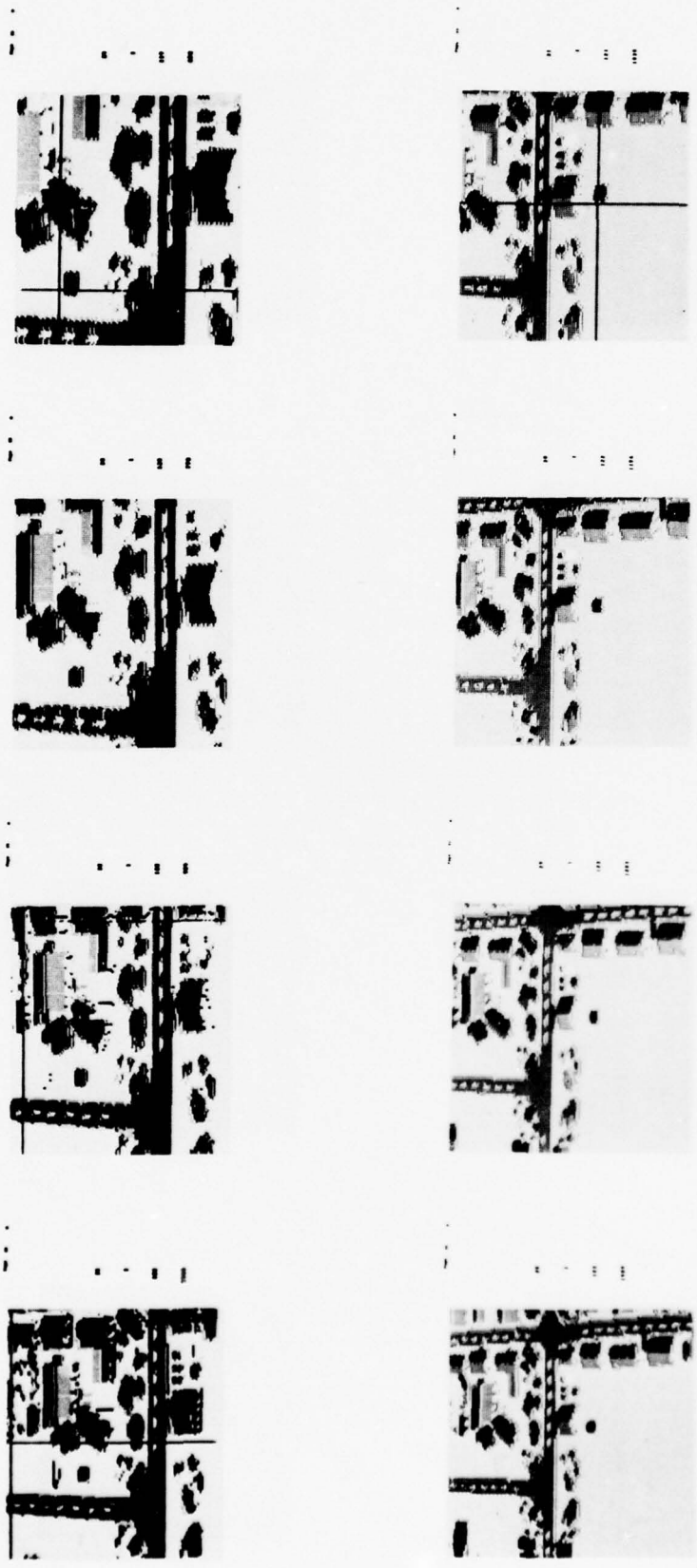


Figure 21. Four Designations of Moving Targets

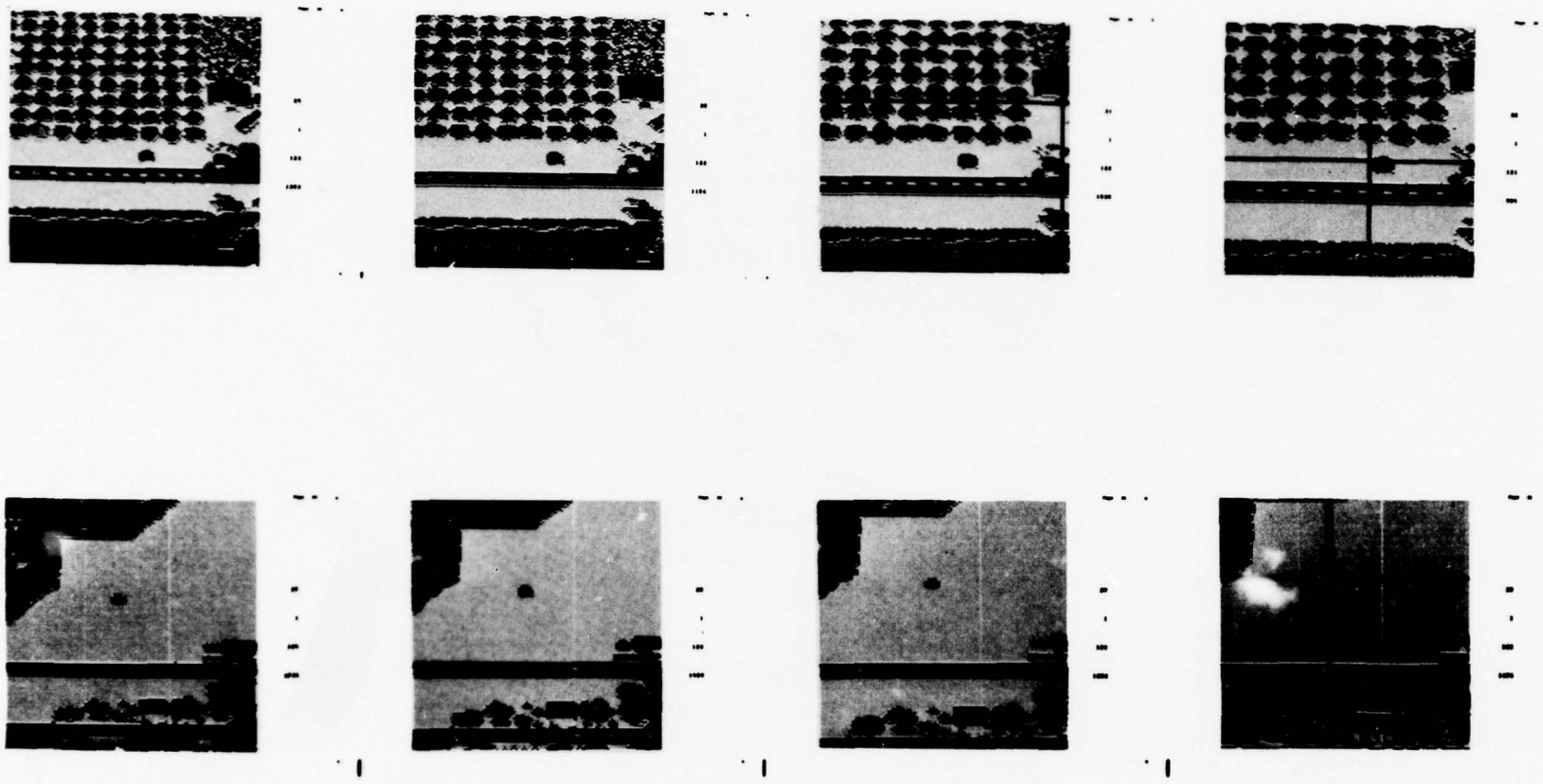


Figure 21. Four Designations of Moving Targets (Continued)

In the course of these simulations, an alternate hardware configuration has been conceived. The main advantage of this alternate is the elimination of the hardware for image transformation. When it was assumed that the image transformation device would be used, the parameters that were continuously optimized during the flight were shift and magnification, with the time between picture "takes" (ΔT) held constant. If magnification were to be kept at some fixed value, presumably one could optimize shift and ΔT instead. With magnification a constant, it would be possible to obtain that function by optical means. Figure 22 shows a hardware reconfiguration that takes advantage of these possibilities. The *image transformation* and the *separate image storage* blocks are eliminated. The *separate image storage* is eliminated because the second image sensor which is added to accept the optically magnified image can also serve as a storage and shifting device.

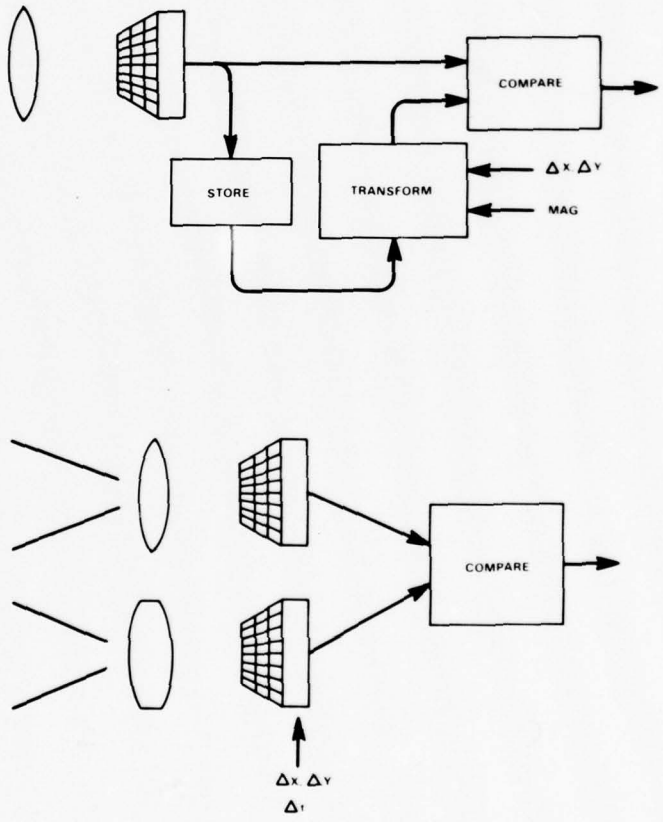


Figure 22. Alternate Hardware Configuration to Eliminate Image Transformation Device

This second configuration has not been simulated because of program deadlines; however, a future simulation is anticipated.

APPENDIX A

PROGRAM LISTINGS

PROGRAM VTRAJ3

Program VTRAJ3 creates files picture data given terrain model data, trajectory data, and target data.

Scope and Data Cards for VTRAJ3

ATTACH, TAPE1, SUBPIX, ID..=....

ATTACH, TAPE2, MAP, ID =....

REQUEST, TAPE4, *PF.

ATTACH, VTRAJ3, ID=....

VTRAJ3.

CATALOG, TAPE4, XXX, ID=...

....

....

$(^{7}8_{9})$

TRAJECTORY CARD

TARGET CARD

(6789)

Where TRAJECTORY CARD contains X, Y, Z, VX, VZ, ΔT, N in Format

(6F10.0, I4)

X = Initial x coordinate of projectile in feet

Y = Initial Y coordinate of projectile in feet

Z = Initial Z coordinate of projectile in feet

VX = Initial X-velocity in feet/sec (negative number)

VZ = Initial Z-velocity in feet/sec (negative number)

ΔT = Time between pictures in seconds

N = number of pictures to be taken

and where TARGET CARD contains TS, TY, TVX, TVY, SIZE in Format

(5F10.0)

TX = Initial target X-position in feet

TY = Initial target Y-position in feet

RVX = Target X-velocity in feet/sec

TVY = Target Y-velocity in feet/sec

SIZE = Target size in yards square

Subroutine Crawler--Updates the position of the target on the terrain map during flight

Subroutine Photo--Performs iteration at 128 x 128 pixels; projects each pixel on terrain model and finds corners of each resulting trapezoid

Subroutine Flight--Updates the position of the projectile during the flight

Subroutine TLU--Does table-look-up on terrain model given coordinates in yards

Subroutine Trap--Accepts location of four corners of a projected pixel on the terrain model and computes average gray level for pixel

Subroutine Pakrat--Accepts an array containing one raster line in 128 locations and packs the information into seven locations within a 7 x 128 array

PROGRAM MOVIE

Program MOVIE accepts files of picture information and renders them on 35mm film using the Stromberg-Carlson 4060.

Scope cards for MOVIE

```
ATTACH,MOVIE,ID=....

ATTACH,SYSLIB.

LIBRARY,SYSLIB.

ATTACH,TAPE4,XXX,ID=....

REQUEST, TAPE3*PF.
```

MOVIE.

DISPOSE, TAPE3, PR=CCB.

$(^{6}7_{89})$

PROGRAM XMARK

Program XMARK simulates target designation hardware operating on simulated video data from program VTRAJ3.

Scope and Data cards for XMARK

ATTACH,XMARK,ID=....

ATTACH,TAPE1,XXX,ID=....(from VTRAJ3)

REQUEST,TAPE4,*PF.

XMARK.

(TAPE4 now contains a copy of the picture file on TAPE1 with the possible superimposition of cross-hairs on designated targets; it may be viewed by means of program MOVIE if desired.)

....

$(^{7}8_{9})$

DATA

(6789)

Where Data Card contains n, JJO, MAGO in * format

N = number of frames to be processed

JJO = initial guess of vertical shift (1)

MAGO = initial guess of magnification (126)

Subroutine MINUS--Subtracts two picture arrays pixel by pixel and places rescaled difference into a third array

Subroutine MAGFY--Receives a picture in one array, shifts it by specified numbers of pixels in I,J directions, applies the magnification specified, and places the result in a second array

Subroutine CURFIT--Receives three values of correlation (as raw scores) that are assumed to result from three values of some parameter

spaced apart by some delta; subroutine computes correction increment to be applied to parameter

Subroutine SIFT--Searches through a difference picture array for a 3 x 3 area containing gray levels outside the range (3,4); search is left to right and down; the first one found is designated by a cross placed over the picture in a second array

Subroutine FETCH--Obtains a single unpacked pixel value from a picture array

Subroutine CORR2--Computes the correlation between two pictures; result is presented as a raw score as well as a normalized value

PROGRAM DISPLAY

Program DISPLAY renders the file of subpictures onto film via the Stromberg-Carlson 4060.

```
PROGRAM VTRAJ3     74/7-   OPT=1

 1                  PROGRAM VTRAJ3 (INPUT,OUTPUT,TAPE1,TAPE2,TAPE4)
                    COMMON NPIX(80,64),IMAP(48,120),IT,JT,NSIZE
                    DIMENSION IHDR(4)
                    ICOUNT=0
 5                  READ (1,1) ((NPIX(I,J),I=1,80),J=1,64)
                1   FORMAT (2020)
                    READ (2,2) ((IMAP(I,J),I=1,24),J=1,120)
                    READ (2,2) ((IMAP(I,J),I=25,48),J=1,120)
                2   FORMAT (2402)
10                  READ 3,X,Y,Z,VX,VZ,DELT,N
                3   FORMAT (6F10.0,I4)
                C   ...
                C   ENTER TARGET POSITION  X,Y FT./VELO(VX,VY)/SIZE
                C   ...
15                  READ 3,TJ,TI,VJ,VI,SIZE
                    NSIZE=SIZE-1.
                    IT=TI/3.
                    JT=TJ/3.
                4   ICOUNT=ICOUNT+1
20                  IHDR(1)=ICOUNT
                    IHDR(2)=X
                    IHDR(3)=Y
                    IHDR(4)=Z
                    WRITE (4) IHDR
25                  PRINT 99,N,X,Y,Z,VX,VZ
                99  FORMAT (I7,3X,5F12.2)
                    CALL PHOTO (X,Y,Z,VX,VZ)
                    N=N-1
                    IF(N.LE.0)GO TO 5
30                  CALL FLIGHT (X,Z,VX,VZ,DELT)
                    IF (Z.LT.120.0) GO TO 5
                    CALL CRAWLR (TI,TJ,VI,VJ,DELT)
                    GO TO 4
                5   IHDR(1)=0
35                  WRITE (4) IHDR
                    STOP
                    END
```

```
SUBROUTINE CRAWLR     74/74   OPT=1

 1                  SUBROUTINE CRAWLR (TI,TJ,VI,VJ,DELT)
                    COMMON  NPIX(80,64),IMPA(48,120),IT,JT,NSIZE
                C   ...
                C   UPDATE TARGET POSITION
 5              C   ...
                    TI=TI+VI*DELT
                    TJ=TJ+VJ*DELT
                    IT=TI/3.
                    JT=TJ/3.
10                  RETURN
                    END
```

```
  SUBROUTINE PHOTO      74/74   OPT=1

 1                SUBROUTINE PHOTO (X,Y,Z,VX,VZ)
                  DIMENSION IO(129),IN(129),IPR(128)
                 1,IPAK(7,128)
                  DATA  FOCAL/5.0/,HAFPI/1.5707963/
 5          C     ...
            C     COMPUTE BORESIGHT ANGLE THETA
            C     ...
                  IF(VZ.NE.0.0) THETA=ATAN( VX/VZ)
                  IF(VZ.EQ.0.0) THETA=HAFPI
10          C     ...
            C     COMPUTE INITIAL ETAO,PHIO BASED ON DEL=.5
            C     ...
                  ETAO=ATAN(.5/FOCAL)
                  PHIO=THETA+ETAO
15          C     ...
            C     COMPUTE INITIAL X-INTERCEPT , J1 POINTER
            C     ...
                  XIO=X-Z*TAN(PHIO)
                  J1=XIO/3.
20                IF(XIO.LT.0.0) J1=-1
                  IF(PHIO.GE.HAFPI) J1=-2
            C     ...
            C     COMPUTE INITIAL ARRAY OF Y-INTERCEPTS IO(I)
            C     ...
25                DO 200 I=1,129
                  RI=I
                  EPS=-.5+(RI/128.)
                  YIO=EPS*Z*COS(ETAO)/(FOCAL*COS(PHIO))+Y
            200   IO(I)=YIO/3.
30          C     ...
            C     DOUBLE DO, COMPUTE 128 COLS PER 128 LINES
            C     ...
                  DO 300  J=1,128
                  RJ=J
35                DEL=.5-RJ/128.
                  ETA=ATAN(DEL/FOCAL)
                  PHI=THETA+ETA
                  XI=X-Z*TAN(PHI)
                  J2=XI/3.
40                IF(XI.LT.0.0) J2=-1
                  IF(PHI.GE.HAFPI) J2=-2
                  DO 400 I=1,129
                  RI=I
                  EPS=-.5+(RI/128.)
45                YIN=EPS*Z*COS(ETA)/(FOCAL*COS(PHI))+Y
            400   IN(I)=YIN/3.
                  DO 500 I=1,128
                  I11=IO(I)
                  I12=IO(I+1)
50                I21=IN(I)
                  I22=IN(I+1)
                  CALL TRAP (J1,J2,I11,I12,I21,I22,IG)
            500   IPR(I)=IG
            C     ...
55          C     SAVE SCAN LINE WITHIN IPAK
            C     ...
                  CALL PAKRAT (J,IPR,IPAK)
```

```
  SUBROUTINE PHOTO      74/74   OPT=1


                C       ...
                C       J2 BECOMES J1, IN(I) BECOMES IO(I)
 60             C       ...
                        J1=J2
                        DO 600 I=1,129
                600     IO(I)=IN(I)
                C       ...
 65             C       LOOP ON SCAN LINES
                C       ...
                300     CONTINUE
                C       ...
                C       WRITE OUT PICTURE ON TAPE
 70             C       ...
                        WRITE (4) IPAK
                        RETURN
                        END
```

```
  SUBROUTINE FLIGHT     74/74   OPT=1


  1                     SUBROUTINE FLIGHT (X,Z,VX,VZ,DELT)
                        DATA DT/.01/, DRAG/.04/
                        T=0.
                  1     DELVX=-VX*DRAG*DT
  5                     X=X+(VX+DELVX/2.)*DT
                        VX=VX+DELVX
                        DELVZ=-(32.16+VZ*DRAG)*DT
                        Z=Z+(VZ+DELVZ/2.)*DT
                        VZ=VZ+DELVZ
 10                     T=T+DT
                        IF(T.LT.DELT)GO TO 1
                        RETURN
                        END
```

```
    SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
    3  FLIGHT

VARIABLES     SN  TYPE          RELOCATION
    0  DELT       REAL              F.P.        35  DELVX      REAL
   36  DELVZ      REAL                          31  DRAG       REAL
   30  DT         REAL                          34  T          REAL
    0  VX         REAL              F.P.         0  VZ         REAL
    0  X          REAL              F.P.         0  Z          REAL
```

```
      SUBROUTINE TLU          74/74   OPT=1


     1                SUBROUTINE TLU (IYDS,JYDS,JGRAY)
                      COMMON NPIX(80,64),IMAP(48,120),IT,JT,NSIZE
                C     ...
                C     TEST IF ON TABULATED REGION
     5          C     ...
                      IF(IYDS.LT.0.OR.IYDS.GT.1919)GO TO 100
                      IF(JYDS.LT.0.OR.JYDS.GT.4799)GO TO 100
                C     ...
                C     TEST IF ON TARGET
    10          C     ...
                      IF(IYDS.GE.IT.AND.IYDS.LE.(IT+NSIZE).AND.
                     1   JYDS.GE.JT.AND.JYDS.LE.(JT+NSIZE))GO TO 101
                C     ...
                C     FIND I,J,K LOCAL COORDINATES
    15          C     ...
                      IM=IYDS/40
                      JM=JYDS/40
                      K=IMAP(IM+1,JM+1) +1
                      I=IYDS-IM*40+1
    20                J=JYDS-JM*40+1
                C     ...
                C     LOOK UP ON SUBPIX
                C     ...
                      LR=I/21
    25                NSHIFT=(I-(LR*20))*3
                      JROW=((J-1)*2)+1+LR
                      JPIX=NPIX(JROW,K)
                      JGRAY=SHIFT(JPIX,NSHIFT).AND.7B
                      RETURN
    30          C     ...
                C     TREAT OFF-MAP SITUATION
                C     ...
                100   JGRAY=4
                      IF(JYDS.LT.2) JGRAY=7
    35                RETURN
                101   JGRAY=0
                      RETURN
                      END
```

SYMBOLIC REFERENCE MAP (R=1)

```
ENTRY POINTS
    3  TLU

VARIABLES     SN  TYPE              RELOCATION
   61  I         INTEGER                          56  IM        INTEGER
12000  IMAP      INTEGER    ARRAY    / /         25200  IT        INTEGER
    0  IYDS      INTEGER               F.P.       62  J         INTEGER
    0  JGRAY     INTEGER               F.P.       57  JM        INTEGER
   66  JPIX      INTEGER                          65  JROW      INTEGER
25201  JT        INTEGER              / /          0  JYDS      INTEGER
   60  K         INTEGER                          63  LR        INTEGER
    0  NPIX      INTEGER    ARRAY    / /         64  NSHIFT    INTEGER
```

SUBROUTINE TRAP 74/74 OPT=1

```
      SUBROUTINE TRAP (J1,J2,I11,I12,I21,I22,IG)
      JD=J2-J1+1
      JDP=JD+1
      IF(I21.GE.I11) ISTAG1=JDP/(I21-I11+1)
      IF(I21.LT.I11) ISTAG1=JDP/(I21-I11-1)
      IF(I22.GE.I12) ISTAG2=JDP/(I22-I12+1)
      IF(I22.LT.I12) ISTAG2=JDP/(I22-I12-1)
      IG=0
      N=0
      DO 1 JDX=1,JD
      JYDS=J1+JDX-1
      ILOLIM=I11+(JDX+1)/ISTAG1
      IHILIM=I12+(JDX+1)/ISTAG2
      DO 1 IYDS =ILOLIM,IHILIM
      CALL TLU (IYDS,JYDS,LEVLG)
      N=N+1
    1 IG=IG+LEVLG
      IG=((N/2)+IG)/N
      RETURN
      END
```

SUBROUTINE PAKRAT 74/74 OPT=1

```
      SUBROUTINE PAKRAT (J,IPR,IPAK)
      DIMENSION IPR(128),IPAK(7,128)
C     ...
C     CLEAR OUT J-TH ROW IPAK
C     ...
      DO 1 K=1,7
    1 IPAK(K,J)=0
C     ...
C     PACK IPR(128) INTO IPAK (K,J),K=1,7
C     ...
      DO 2 N=1,128
      MN=(N-1)/20
      MNI=MN+1
      ISHIFT=20-(N-20*MN)
      KSHIFT=3*ISHIFT
    2 IPAK(MNI,J)=IPAK(MNI,J).OR.SHIFT(IPR(N),KSHIFT)
      RETURN
      END
```

```
      PROGRAM MOVIE      74/74   OPT=1

 1                PROGRAM MOVIE (INPUT,OUTPUT,TAPE3,TAPE4,TAPE51=OUTPUT)
                  DIMENSION  IHDR(4),IPAK(7,128),LTAB(8),IPR(128),Z(200)
                  DATA LTAB/1R4,1RP,1RC,1R3,1RQ,1R2,1R1,1R /
             C    ...
 5           C    INITIALIZE 4060
             C    ...
                  CALL MODESG(Z,3)
                  CALL CRTID(Z,5HNEZRH)
                  CALL PAGEG(Z,0,0,1)
10           C    SET SCALE TO RASTER SIZE
             C    ...
                  CALL SETSMG(Z,19,4095.)
                  CALL SETSMG(Z,20,3071.)
             C    ...
15           C    READ HEADER - POSSIBLE END OF RUN
             C    ...
             10   READ(4) IHDR
                  IF(IHDR(1).NE.0)GO TO 1
                  CALL PAGEG(Z,0,0,1)
20                CALL CRTID(Z,5H     )
                  CALL EXITG(Z)
                  STOP
             C    ...
             C    WRITE HEADER INFORMATION LARGE CH.-UPPER CASE
25           C    ...
             1    CALL SETSMG (Z,45,1.5)
                  CALL SETSMG (Z,52,0.0)
                  X=3400.
                  Y=2000.
30                CALL FMTSG(Z,1,5,J,IHDR(1),IPL)
                  CALL LEGNDG(Z,X,Y,5,IPL)
                  Y=Y-333.
                  CALL FMTSG(Z,1,5,0,IHDR(2),IPL)
                  CALL LEGNDG(Z,X,Y,5,IPL)
35                Y=Y-333.
                  CALL FMTSG(Z,1,5,0,IHDR(3),IPL)
                  CALL LEGNDG(Z,X,Y,5,IPL)
                  Y=Y-333.
                  CALL FMTSG(Z,1,5,0,IHDR(4),IPL)
40                CALL LEGNDG(Z,X,Y,5,IPL)
             C    ...
             C    READ IN COMPACTED VIDEO ARRAY, INITIALIZE PICTURE
             C    ...
                  READ (4) IPAK
45                CALL SETSMG(Z,45,.75)
                  CALL SETSMG(Z,52,2.0)
                  X=64.
                  Y=3008.
             C    ...
50           C    LOOP OVER 128 ROWS
             C    ...
                  DO 111 J=1,128
             C    ...
             C    DECOMPOSE J-TH LINE INTO PRINT ARRAY
55           C    ...
                  DO 1111 L=1,13
             1111 IPR (L)=0
```

```
      PROGRAM MOVIE      74/74   OPT=1

                    DO 11 N=1,128
                    MN=(N-1)/20
   60               MNI=MN+1
                    NSHIFT=(N-20*MN)*3
                    MYWORD=(SHIFT(IPAK(MNI,J),NSHIFT).AND.7B)+1
                    ICHAR=LTAB(MYWORD)
                    NDX=(N-1)/10
   65               NDXI=NDX+1
                    KROT=6*(10-(N-10*NDX))
               11   IPR(NDXI)=IPR(NDXI).OR.SHIFT(ICHAR,KROT)
               C    ...
               C    WRITE ON CRT
   70          C    ...
                    CALL LEGNDG(Z,X,Y,128,IPR)
               111  Y=Y-23.
                    CALL PAGEG(Z,0,0,1)
                    GO TO 10
   75               END
```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
10223 MOVIE

VARIABLES	SN	TYPE	RELOCATION				
10561	ICHAR	INTEGER		10565	IHDR	INTEGER	ARRAY
10571	IPAK	INTEGER	ARRAY	10551	IPL	INTEGER	
12401	IPR	INTEGER	ARRAY	10552	J	INTEGER	
10564	KROT	INTEGER		10553	L	INTEGER	
12371	LTAB	INTEGER	ARRAY	10555	MN	INTEGER	
10556	MNI	INTEGER		10560	MYWORD	INTEGER	
10554	N	INTEGER		10562	NDX	INTEGER	
10563	NDXI	INTEGER		10557	NSHIFT	INTEGER	
10547	X	REAL		10550	Y	REAL	
12601	Z	REAL	ARRAY				

FILE NAMES					
MODE					
0	INPUT	2043	OUTPUT	4106	TAPE3
2043	TAPE51				

EXTERNALS	TYPE	ARGS		
CRTID		2	EXITG	1
FMTSG		6	LEGNDG	5
MODESG		2	PAGEG	4
SETSMG		3		

INLINE FUNCTIONS	TYPE	ARGS	
SHIFT	NO TYPE	2	INTRIN

STATEMENT LABELS

10250	1	10236	10
0	111	0	1111

```
PROGRAM XMARK      73/74   OPT=1

 1                 PROGRAM XMARK(INPUT,OUTPUT,TAPE1,TAPE4)
                C  ...
                C  PROGRAM TO OPTOMIZE CORRELATION AND WRITE OUT
                C                DIFFERENCE PICTURE
 5              C  ...
                   DIMENSION IHDR(4),NPIXO(7,128),NPIXT(7,128),
                  1NPIXN(7,128),NPIXD(7,128)
                C  ...
                C  READ IN CARD PARAMETERS AND PRINT
10              C  ...
                   READ * ,N,JJO,MAGO
                   PRINT 9,N,JJO,MAGO
                9  FORMAT (4H1 N=,I4,4X,4HJJO=,I4,4X,5HMAGO=,I4)
                   PRINT 10
15                 FORMAT(55H    I    JJO     MAGO     X     SUM-N       SUM-O       SUM-P)
                   PRINT 11
                11 FORMAT(554    -    ---     ----     -     -----       -----       -----)
                12 FORMAT(I5,I6,I8,4X,A1,I9,I11,I11)
                C  ...
20              C  READ IN OLD PICTURE
                C  ...
                   READ (1) IHDR
                   IF (IHDR(1).EQ.0) STOP
                   READ (1) NPIXO
25              C  ...
                C  INITIATE OUTER LOOP COMPUTATIONS
                C  ...
                   DO 7 I=1,N
                C  ...
30              C  READ IN NEW PICTURE
                C  ...
                   READ (1) IHDR
                   IF (IHDR(1).EQ.0) STOP
                   READ (1) NPIXN
35              C  ...
                C  DO SET OF THREE SHIFTS AND CORRELATIONS
                C  ...
                   JJ=JJO-2
                   CALL MAGFY (NPIXO,NPIXT,0,JJ,MAGO)
40                 CALL CORR2 (NPIXN,NPIXT,COEF,SUMN)
                   ISUMN=SUMN
                   JJ=JJO+2
                   CALL MAGFY (NPIXO,NPIXT,0,JJ,MAGO)
                   CALL CORR2 (NPIXN,NPIXT,COEF,SUMP)
45                 ISUMP=SUMP
                   JJ=JJO
                   CALL MAGFY (NPIXO,NPIXT,0,JJ,MAGO)
                   CALL CORR2 (NPIXN,NPIXT,COEF,SUMO)
                   ISUMO=SUMO
50              C  ...
                C  PRINT INTERMEDIATE RESULTS
                C  ...
                   X=1HS
                   PRINT 12,I,JJO,MAGO,X,ISUMN,ISUMO,ISUMP
55              C  ...
                C  DO PARABOLIC CURVE FIT
                C  ...
```

```
PROGRAM XMARK      73/74   OPT=1

                      CALL CURFIT (SUMN,SUMO,SUMP,2.0,INCRE)
                      JJO=JJO+INCRE
 60             C     ...
                C     DO SET OF THREE MAGNIFICATIONS AND CORRELATIONS
                C     ...
                      MAG=128
                      CALL MAGFY (NPIXO,NPIXT,0,JJO,MAG)
 65                   CALL CORR2 (NPIXN,NPIXT,COEF,SUMN)
                      ISUMN=SUMN
                      IZOOM=128-MAGO
                      MAG=128-2*IZOOM
                      CALL MAGFY (NPIXO,NPIXT,0,JJO,MAG)
 70                   CALL CORR2 (NPIXN,NPIXT,COEF,SUMP)
                      ISUMP=SUMP
                      MAG=MAGO
                      CALL MAGFY (NPIXO,NPIXT,0,JJO,MAG)
                      CALL CORR2 (NPIXN,NPIXT,COEF,SUMO)
 75                   ISUMO=SUMO
                C     ...
                C     PRINT INTERMEDIATE RESULTS
                C     ...
                      X=1HM
 80                   PRINT 12,I,JJO,MAGO,X,ISUMN,ISUMO,ISUMP
                C     ...
                C     DO PARABOLIC CURVE FIT
                C     ...
                      DELTA=128-MAGO
 85                   CALL CURFIT (SUMN,SUMO,SUMP,DELTA,INCRE)
                      MAGO=MAGO-INCRE
                      IF(MAGO.GT.127) MAGO=127
                C     ...
                C     SKIP RECOMP IF MAGNIFICATION NOT CHANGED
 90             C     ...
                      IF (INCRE.EQ.0) GO TO 4
                C     ...
                C     RECOMPUTE TRANSFORMED PICTURE
                C     ...
 95                   CALL MAGFY (NPIXO,NPIXT,0,JJO,MAGO)
                C     ...
                C     PRINT TRACE
                C     ...
                      X=1HR
100                   PRINT 12,I,JJO,MAGO,X
                C     ...
                C     SUBTRACT ARRAYS BY PIXEL NPIXT-NPIXO=NPIXD
                C     ...
                4     CALL MINUS (NPIXT,NPIXN,NPIXD)
105             C     ...
                C     SEARCH OUT TARGET IN SUBTRACTED ARRAY
                C     ...
                      CALL SIFT (NPIXN,NPIXD)
                C     ...
110             C     WRITE NEW PICTURES OUT TO FILE
                C     ...
                      IHDR(1) = I
                      IHDR(2) = JJO
                      IHDR(3) = MAGO
```

PROGRAM XMARK

```
      PROGRAM XMARK      73/74   OPT=1

115                 WRITE (4) IHDR
                    WRITE (4) NPIXN
                C   ...
                C   TRANSFER NPIXN TO NPIXO
                C   ...
120                 DO 17 IDX=1,7
                    DO 17 JDX=1,128
                17  NPIXO(IDX,JDX) = NPIXN (IDX,JDX)
                C   ...
                C   END OF OUTER LOOP COMPUTATION
125             C   ...
                7   CONTINUE
                C   ...
                C   WRITE NULL HEADER
                C   ...
130                 IHDR (1) = 0
                    WRITE(4) IHDR
                    REWIND 4
                    STOP
                    END
```

```
   SUBROUTINE MINUS      73/74   OPT=1

 1                  SUBROUTINE MINUS (NPIXT,NPIXO,NPIXD)
                    DIMENSION NPIXT(7,128),NPIXO(7,128),NPIXD(7,128)
                C   ...
                C   CLEAR OUT NPIXD ARRAY
 5              C   ...
                    DO 2 I=1,7
                    DO 2 J=1,128
                2   NPIXD(I,J)=0
                C   ...
10              C   SUBTRACT TWO PICTURES AND RESCALE RESULT
                C   ...
                    DO 1 I=1,7
                    DO 1 J=1,128
                    DO 1 K=4,61,3
15                  NPIXLT=SHIFT(NPIXT(I,J),K-1).AND.78
                    NPIXLO=SHIFT(NPIXO(I,J),K-1).AND.78
                    NPIXLD=(7+NPIXLT-NPIXLO)/2
                1   NPIXD(I,J)=NPIXD(I,J).OR.SHIFT(NPIXLD,61-K)
                    RETURN
20                  END
```

PROGRAM XMARK

SUBROUTINE MAGFY 73/74 OPT=1

```
      SUBROUTINE MAGFY(IPAK,JPAK,ISHIFT,JSHIFT,MAGGIE)
      DIMENSION IPAK(7,128),JPAK(7,128),IASSY(128)
C     ...
C     SUBROUTINE TO SHIFT PICTURE IN IPAK BY (ISHIFT,JSHIFT)
C     PIXELS, ENLARGE RESULT BY A MAGNIFICATION FACTOR OF
C     (128/MAG), AND PLACE RESULT IN JPAK.
C     ...
C     PREPROCESS MAG
C     ...
      MAG = 128 - 2 * (128 - MAGGIE)
      MAGO=(MAG*100000)/128
      NORMF=12800000/MAG
      ISHPRE=(128-MAG)/2
C     ...
C     LOOP 128 LINES ON INDEX J
C     ...
      DO 1111 J=1,128
C     ...
C     CLEAR OUT ASSEMBLY ARRAY
C     ...
      DO 1 I=1,128
    1 IASSY(I)=0
C     ...
C     COMPUTE J-DEPENDANT QUANTITIES
C     ...
      JN=J*MAGO
      JNIP=JN/100000
      JNFP=JN-(JNIP*100000)
      JNFPNM=(JNFP*NORMF)/100000
      JNFPSB=100000-JNFPNM
C     ...
C     INNER LOOP 128 PIXELS ON INDEX I
C     ...
      DO 111 I=1,128
C     ...
C     COMPUTE WEIGHTING FACTORS P,Q,R,S
C     ...
      IN=I*MAGO
      INIP=IN/100000
      INFP=IN-(INIP*100000)
      INFPNM=(INFP*NORMF)/100000
      INFPSB=100000-INFPNM
      NP=(INFPSB*JNFPSB)/100000
      NQ=(INFPNM*JNFPSB)/100000
      NR=(INFPSB*JNFPNM)/100000
      NS=(INFPNM*JNFPNM)/100000
C     ...
C     COMPUTE FETCH INDICES
C     ...
      ID1=INIP+ISHPRE+ISHIFT
      ID2=ID1+1
      JD1=JNIP+ISHPRE+JSHIFT
      JD2=JD1+1
C     ...
C     CALL SUBROUTINE TO FETCH PIXELS FROM IPAK
C     ...
      CALL FETCH (IPAK,ID1,JD1,KP)
```

PROGRAM XMARK

```
  SUBROUTINE MAGFY      73/74   OPT=1


                      CALL FETCH (IPAK,ID2,JD1,KQ)
                      CALL FETCH (IPAK,ID1,JD2,KP)
60                    CALL FETCH (IPAK,ID2,JD2,KS)
                C     ...
                C     COMPUTE PIXEL VALUE
                C     ...
                      L=(KP*NP+KQ*NQ+KR*NR+KS*NS+50000)/100000
65                    IF(L.GE.7)L=7
                C     ...
                C     PLACE IN ASSEMBLY ARRAY
                C     ...
                111   IASSY(I)=L
70              C     ...
                C     PACK ASSEMBLY ARRAY INTO JPAK
                C     ...
                      CALL PAKRAT(J,IASSY,JPAK)
                1111  CONTINUE
75                    RETURN
                      END
```

```
  SUBROUTINE PAKRAT     73/74   OPT=1

 1                    SUBROUTINE PAKRAT (J,IPR,IPAK)
                      DIMENSION IPR(128),IPAK(7,128)
                C     ...
                C     CLEAR OUT J-TH ROW IPAK
 5              C     ...
                      DO 1 K=1,7
                1     IPAK(K,J)=0
                C     ...
                C     PACK IPR(128) INTO IPAK(K,J),K=1,7
10              C     ...
                      DO 2 N=1,128
                      MN=(N-1)/20
                      MNI=MN+1
                      ISHIFT=20-(N-20*MN)
15                    KSHIFT=3*ISHIFT
                2     IPAK(MNI,J)=IPAK(MNI,J).OR.SHIFT(IPR(N),KSHIFT)
                      RETURN
                      END
```

PROGRAM XMARK

```
      SUBROUTINE CURFIT      73/74    OPT=1


     1                      SUBROUTINE CURFIT (FM,FO,FP,DEL,INCRE)
                       C     ...
                       C     DETERMINE NEW TRIAL VALUE.
                       C     ...
     5                       A=(FP+FM-2.0*FO)/(2.0*DEL*DEL)
                             B=(FP-FM)/(2.0*DEL)
                             IF(A)1,2,3
                       C     ...
                       C     POSITIVE PARABOLA ( USE INFLECTION POINT.
    10                 C     ...
                       3     ZIP=-B/(2.0*A)
                             IF(ZIP) 31,31,32
                      31     INCRE=ZIP-0.5
                             RETURN
    15                32     INCRE=ZIP+0.5
                             RETURN
                       C     ...
                       C     NEGATIVE PARABOLA , TEST SLOPE
                       C     ...
    20                 1     IF (B.LT.0.) GO TO 11
                       C     ...
                       C     POSITIVE SLOPE , USE (+) FORMULA.
                       C     ...
                             INCRE=(-B+SQRT(B*B-4.0*A*FO))/(2.0*A)+.5
    25                       RETURN
                       C     ...
                       C     NEGATIVE SLOPE ( USE (-) FORMULA)
                       C     ...
                      11     INCRE=(-B-SQRT(B*B-4.0*A*FO))/(2.0*A)+.5
    30                       RETURN
                       C     ...
                       C     STRIGHT LINE INTERCEPT
                       C     ...
                       2     INCRE=-B/FO
    35                       RETURN
                             END
```

SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
3 CURFIT

VARIABLES	SN	TYPE	RELOCATION				
50	A	REAL		61	B	REAL
0	DEL	REAL	F.P.	0	FM	REAL
0	FO	REAL	F.P.	0	FP	REAL
0	INCRE	INTEGER	F.P.	62	ZIP	REAL

EXTERNALS	TYPE	ARGS
SQRT	REAL	1 LIBRARY

SUBROUTINE SIFT 73/74 OPT=1

```
      SUBROUTINE SIFT (NPIXN,NPIXD)
C     ...
C     SEARCH FOR TARGET AS A 3 X 3 ARRAY OF NOT (3.OR.4)
C     ...
      DIMENSION NPIXN(7,128),NPIXD(7,128)
C     ...
C     LOOP OVER ENTIRE PICTURE, EXTRACT I,J-TH PIXEL,PUSH
C     ...
      DO 1 J=1,128
C     ...
C     ZERO OUT STACK
C     ...
      K1=0
      K2=0
      K3=0
      DO 1 I=1,128
      K1=K2
      K2=K3
      K3=0
      CALL FETCH (NPIXD,I,J,KP)
      IF (KP.LT.3.OR.KP.GT.4)K3=1
C     ...
C     TEST STACK
C     ...
      IF (K1+K2+K3.EQ.3) GO TO 7
    1 CONTINUE
      RETURN
    7 J2=J+1
      J3=J+2
      I1=I-2
      I2=I-1
      CALL FETCH (NPIXD,I1,J2,KP)
      IF(KP.EQ.3.OR.KP.EQ.4)GO TO 1
      CALL FETCH (NPIXD,I2,J2,KP)
      IF(KP.EQ.3.OR.KP.EQ.4)GO TO 1
      CALL FETCH (NPIXD,I ,J2,KP)
      IF(KP.EQ.3.OR.KP.EQ.4)GO TO 1
      CALL FETCH (NPIXD,I1,J3,KP)
      IF(KP.EQ.3.OR.KP.EQ.4)GO TO 1
      CALL FETCH (NPIXD,I2,J3,KP)
      IF(KP.EQ.3.OR.KP.EQ.4)GO TO 1
      CALL FETCH (NPIXD,I ,J3,KP)
      IF(KP.EQ.3.OR.KP.EQ.4)GO TO 1
C     ...
C     TARGET FOUND - INSERT CROSS
C     ...
      ICROSS=I-1
      JCROSS=J+1
      DO 88 K=1,7
   88 NPIXN(K,JCROSS)=0
      MASK=7777777777777777708
      ICOL=(ICROSS-1)/20
      ICOL1=ICOL+1
      NRES=ICROSS-20*ICOL
      NSHIFT=(20-NRES)*3
      MASK=SHIFT(MASK,NSHIFT)
      DO 99 K=1,128
```

PROGRAM XMARK

```
  SUBROUTINE SIFT        73/74   OPT=1


            99      NPIXN(ICOL1,K)=NPIXN(ICOL1,K).AND.MASK
                    C     ...
60                  C     PRINT RESULT
                    C     ...
                          PRINT 17,ICROSS,JCROSS
                    17    FORMAT (23H0******TARGET AT(I,J)=(I3,1H,I3,7H)******/)
                          RETURN
65                        END
```

```
  SUBROUTINE FETCH       73/74   OPT=1

 1                        SUBROUTINE FETCH (IPAK,I,J,K)
                          DIMENSION IPAK (7,128)
                    C     ...
                    C     PLACE I,J-TH PIXEL FROM IPAK INTO K
 5                  C     ...
                          IF (I.LT.1.OR.I.GT.128)GO TO 7
                          IF (J.LT.1.OR.J.GT.128)GO TO 7
                          MN=(I-1)/20
                          MNI=MN+1
10                        NSHIFT=(I-20*MN)*3
                          K=SHIFT(IPAK(MNI,J),NSHIFT).AND.7B
                          RETURN
                    7     K=3
                          RETURN
15                        END
```

```
  SUBROUTINE CORR2       73/74   OPT=1

 1                        SUBROUTINE CORR2(IPIX1,IPIX2,COEF,SUM)
                          DIMENSION IPIX1(7,128),IPIX2(7,128)
                          SUM=0.
                          DO 50 I=1,7
 5                        DO 50 J=1,125
                          DO 50 K=4,61,3
                          NIPIX1=SHIFT(IPIX1(I,J),K-1).AND.7B
                          NIPIX2=SHIFT(IPIX2(I,J),K-1).AND.7B
                          DIFF=NIPIX1-NIPIX2
10                  50    SUM=SUM+ABS(DIFF)
                          COEF=SUM/114665.
                          COEF=1.0-COEF
                          RETURN
                          END
```

PROGRAM DISPLAY

```
PROGRAM DISPLAY     74/74    OPT=1

 1                  PROGRAM DISPLAY(INPUT,OUTPUT,TAPE1,TAPE3,TAPE51=OUTPUT)
                    DIMENSION SHADES(8),PLINE(120),NPIX(80,64),TITLES(7),CHR(12)
                    COMMON/BLK1/Z(200)
                    DATA SHADES/1R4,1RP,1RC,1R3,1RC,1R2,1R1,1R /    ,
 5                 1      TITLES/10HSUBPIX    ,10HDISPLAY    ,
                   1             10H          ,10HFRAME  NO.,
                   1             10HCHARACTERS,10HUSED       ,
                   1             10H( 103N/P4)/
                    CALL MODESG(Z,3)
10                  CALL PAGEG(Z,0,0,1)
                    CALL OPTID(Z,12HN90 GH  A115)
                    CALL PAGEG(Z,0,0,2)
                    CALL SETSMG(Z,19,4095.)
                    CALL SETSMG(Z,20,3071.)
15            C     *****
              C     INPUT SUBPIX DATA INTO NPIX ARRAY
              C     *****
                    READ(1,62)((NPIX(I,J),I=1,80),J=1,64)
                 62 FORMAT(2020)
20            C     *****
              C     CARD INPUT PARAMETERS
              C         1) PRC -  END OF FILE PARAMETER( HOLERITH CHARACTER )
              C                     ... IF PRC = "S" THEN PROGRAM EXITS.
              C                     ... IF PRC IS ANY OTHER CHARACTER THE PROGRAM
25            C                         CONTINUES.
              C         2) K1  -  SUBPIX COORDINATE( OCTAL NUMBER )
              C         3) K2  -  SUBPIX COORDINATE( OCTAL NUMBER )
              C         4) K3  -  SUBPIX COORDINATE( OCTAL NUMBER )
              C         5) NUM -  FRAME NUMBER( INTEGER )
30            C         6) ILY -  FRAME ORIENTATION PARAMETER( INTEGER )
              C                     ... IF ILY = 0 THEN DATA IS FOR TOP HALF
              C                     ... IF ILY = 1 THEN DATA IS FOR BOTTOM HALF
              C         7) ITYPE - SINGLE OR QUADRUPLE PRINT PARAMETER( INTEGER )
              C                     ... IF ITYPE = 0 THEN SINGLE PRINT
35            C                     ... IF ITYPE = 1 THEN QUADRUPLE PRINT
              C     *****
                 19 READ 20, PRC,K1,K2,K3,NUM,ILY,ITYPE
                 20 FORMAT(A1,3O2,I2,2I1)
                    IF(PRC.EQ.1HS)  GO TO 99
40                  CALL CONVERT(K1,KK1)
                    CALL CONVERT(K2,KK2)
                    CALL CONVERT(K3,KK3)
              C     *****
              C     CHECK TO SEE IF PROCESSING TOP OR BOTTOM SUBPIXS
45            C     *****
                    IF(ILY) 30,25
                 25 CALL HEADER(TITLES,NUM)
                    XX=514.
                    YY=2556.
50                  CALL PIXDIS(XX,YY,KK1,KK2,KK3)
                    X=64.
                    Y=2456.
                    IF(ITYPE) 27,29
                 27 XBAR=75.
55                  YBAR=2445.
                 29 K1=K1+1
                    K2=K2+1
```

```
      PROGRAM DISPLAY    74/74   OPT=1


                        K3=K3+1
                        CALL SETSMG(Z,45,,75)
   60                   CALL SETSMG(Z,52,2.0)
                        DO 1  J=1,40
                        DO 2  I=1,40
                        CALL TLU(I,J,K1,IDX,NPIX)
                      2 PLINE(I)=SHADES(IDX+1)
   65             C     *****
                        DO 3  I=41,80
                        IX=I-40
                        CALL TLU(IX,J,K2,IDX,NPIX)
                      3 PLINE(I)=SHADES(IDX+1)
   70             C     *****
                        DO 4  I=81,120
                        IX=I-80
                        CALL TLU(IX,J,K3,IDX,NPIX)
                      4 PLINE(I)=SHADES(IDX+1)
   75             C     *****
                        CALL NSTUFF(PLINE,CHR)
                        CALL LEGNDG(Z,X,Y,120,CHR)
                        IF(ITYPE) 6,7
                      6 CALL LEGNDG(Z,X,YBAR,120,CHR)
   80                   CALL LEGNDG(Z,XBAR,Y,120,CHR)
                        CALL LEGNDG(Z,XBAR,YBAR,120,CHR)
                        YBAR=YBAR-23.
                      7 Y=Y-23.
                      1 CONTINUE
   85                   IF(ILY) 40,45
                     40 XX=514.
                        YY=466.
                        CALL PIXDIS(XX,YY,KK1,KK2,KK3)
                        CALL PAGEG(Z,0,0,1)
   90                45 GO TO 19
                  C     *****
                     30 X=64.
                        Y=1536.
                        IF(ITYPE) 35,36
   95                   35 XBAR=75.
                        YBAR=1525.
                     36 GO TO 29
                  C     *****
                  C     CLEAN UP AND EXIT
  100             C     *****
                     99 CALL CRTID(Z,16HN90 END OF  RUN )
                        CALL PAGEG(Z,0,0,2)
                        CALL EXITG(Z)
                        STOP
  105                   END


   SYMBOLIC REFERENCE MAP (R=1)
```

PROGRAM DISPLAY

SUBROUTINE TLU 74/74 OPT=1

```
1                SUBROUTINE TLU(I,J,K,IDX,NPIX)
                 DIMENSION NPIX(80,64)
                 LR=I/21
                 NSHIFT=(I-(LR*20))*3
5                JROW=((J-1)*2)+1+LR
                 JPIX=NPIX(JROW,K)
                 IDX=SHIFT(JPIX,NSHIFT).AND.7B
                 RETURN
                 END
```

SUBROUTINE HEADER 74/74 OPT=1

```
1                SUBROUTINE HEADER(HDR,NUMBER)
                 DIMENSION HDR(7)
                 COMMON/BLK1/Z(200)
                 CALL SETSMG(Z,45,1.5)
5                CALL SETSMG(Z,52,0.0)
                 X=3400.
                 Y=2000.
                 DO 1111  NMMM=1,3
                 CALL LEGNDG(Z,X,Y,10,HDR(NMMM))
10           1111 Y=Y-67.
                 Y=1591.
                 CALL LEGNDG(Z,X,Y,10,HDR(4))
                 Y=Y-67.
                 CALL NUMBRG(Z,X,Y,2,NUMBER)
15               Y=1328.
                 DO 2222  NGDH=5,7
                 CALL LEGNDG(Z,X,Y,10,HDR(NGDH))
             2222 Y=Y-67.
                 RETURN
20               END
```

SUBROUTINE PIXDIS 74/74 OPT=1

```
1                SUBROUTINE PIXDIS(X,Y,KA,KB,KC)
                 COMMON/BLK1/Z(200)
                 CALL SETSMG(Z,45,1.5)
                 CALL SETSMG(Z,52,0.0)
5                CALL NUMBRG(Z,X,Y,2,KA)
                 X=X+920.
                 CALL NUMBRG(Z,X,Y,2,KB)
                 X=X+920.
                 CALL NUMBRG(Z,X,Y,2,KC)
10               RETURN
                 END
```

PROGRAM DISPLAY

```
      SUBROUTINE NSTUFF     74/74   OPT=1

    1                 SUBROUTINE NSTUFF(PRLIN,CHARS)
                      DIMENSION PRLIN(120),CHARS(12)
                      COMMON/BLK1/Z(200)
                      M=0
    5                 DO 5  JK=1,12
                      DO 5  JL=1,10
                      M=M+1
                      CALL PUTCZZ(Z,PRLIN(M),JL,CHARS(JK))
                    5 CONTINUE
   10                 RETURN
                      END



      SUBROUTINE CONVERT    74/74   OPT=1

    1                 SUBROUTINE CONVERT(I,II)
                  C   *****
                  C   SUBROUTINE CONVERT TAKES A TWO DIGIT OCTAL NUMBER AND CONVERTS
                  C   THIS NUMBER(I) INTO A DECIMAL NUMBER(II) THAT HAS A DIFFERENT
    5             C   ABSOLUTE VALUE BUT HAS THE SAME DIGITS SO WHEN THE OUTPUT IS
                  C   PRINTED THE NUMBER APPEARS IN OCTAL FORMAT.
                  C   *****
                      JONE=I.AND.7B
                      JTEN=(SHIFT(I,-3).AND.7B)*10
   10                 II=JTEN+JONE
                      RETURN
                      END



      SYMBOLIC REFERENCE MAP (R=1)

ENTRY POINTS
    3  CONVERT

VARIABLES     SN  TYPE            RELOCATION
    0  I          INTEGER               F.P.          0  II         INTEGER
   13  JONE       INTEGER                            14  JTEN       INTEGER

INLINE FUNCTIONS   TYPE   ARGS
      SHIFT      NO TYPE    2  INTRIN

STATISTICS
  PROGRAM LENGTH                158       13
```

DISTRIBUTION

Library of Congress
Washington, DC 20540
Attn: Gift and Exchange Division (4)

Defense Documentation Center
Cameron Station
Alexandria, VA 22314 (12)

Chief of Naval Material
Department of the Navy
Washington, DC 20360

Local:

C051 (Mr. Gillicuddy)
E1A (McKnight)
E21 (Mebane)
E41
E41 (Hall)
F14 (Ely)
F14 (Poljak)
F56 (Puglielli)
G14 (Cox)
G50 (Culbertson)
K04 (Schindel)
K60 (LaMonica)
K61 (Shelton)
K61 (Hilton) (35)
K74 (Thombs)
N12 (Ramsbotham)
N43 (Pater)
R43 (DeSavage)
U02 (Fanjoy)
X210 (2)
X2101 (GIDEP)