

AD-A074 019

TECHNOLOGY SERVICE CORP SANTA MONICA CALIF
COMPUTER IMAGE GENERATION TEXTURE STUDY.(U)
AUG 79 T STENGER, W DUNGAN, R REYNOLDS

F/G 12/1

F33615-77-C-0063

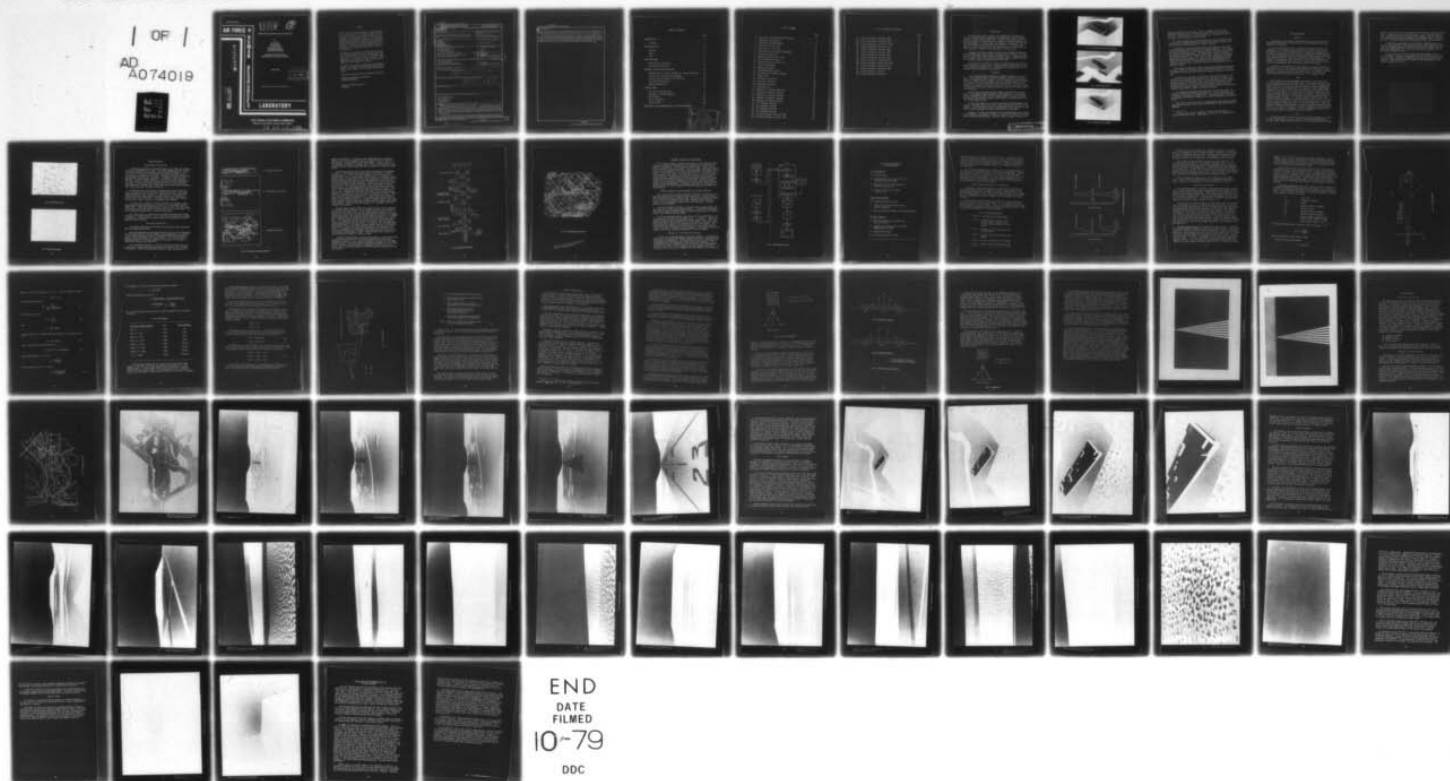
UNCLASSIFIED

AFHRL-TR-79-2

NL

| OF |

AD
A074019



END
DATE
FILMED
10-79
DDC

AIR FORCE



HUMAN RESOURCES

AD A 074019

DDC FILE COPY

LEVEL IV

2

COMPUTER IMAGE GENERATION TEXTURE STUDY

By

Tony Stenger

William Dungan

Richard Reynolds

Technology Service Corporation

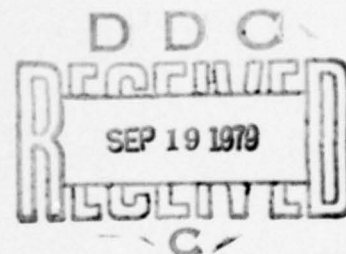
2811 Wilshire Boulevard

Santa Monica, California 90403

ADVANCED SYSTEMS DIVISION

Wright-Patterson Air Force Base, Ohio 45433

August 1979



Approved for public release; distribution unlimited.

LABORATORY

AIR FORCE SYSTEMS COMMAND

BROOKS AIR FORCE BASE, TEXAS 78235

79 09 18 140

NOTICE

When U.S. Government drawings, specifications, or other data are used for any purpose other than a definitely related Government procurement operation, the Government thereby incurs no responsibility nor any obligation whatsoever, and the fact that the Government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data is not to be regarded by implication or otherwise, as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

This final report was submitted by Technology Service Corporation, 2811 Wilshire Boulevard, Santa Monica, California 90403, under contract F33615-77-C-0063, project 6114, with Advanced Systems Division, Air Force Human Resources Laboratory (AFSC), Wright-Patterson Air Force Base, Ohio 45433. Capt Michael L. Ingalls (ASM) was the Contract Monitor for the Laboratory.

This report has been reviewed by the Information Office (OI) and is releasable to the National Technical Information Service (NTIS). At NTIS, it will be available to the general public, including foreign nations.

This technical report has been reviewed and is approved for publication.

GORDON A. ECKSTRAND, Technical Director
Advanced Systems Division

RONALD W. TERRY, Colonel, USAF
Commander

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

19 REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER AFHRL TR-79-2	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) COMPUTER IMAGE GENERATION TEXTURE STUDY	5. TYPE OF REPORT & PERIOD COVERED Final rept.	
6. AUTHOR(S) Tony Stenger William Dungan Richard Reynolds	6. PERFORMING ORG. REPORT NUMBER	
7. PERFORMING ORGANIZATION NAME AND ADDRESS Technology Service Corporation 2811 Wilshire Boulevard Santa Monica, California 90403	8. CONTRACT OR GRANT NUMBER(S) F33615-77-C0063	
9. CONTROLLING OFFICE NAME AND ADDRESS HQ Air Force Human Resources Laboratory (AFSC) Brooks Air Force Base, Texas 78235	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 62205F 61142107	
11. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) Advanced Systems Division Air Force Human Resources Laboratory Wright-Patterson Air Force Base, Ohio 45433	12. REPORT DATE August 1979	
13. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution unlimited.	13. NUMBER OF PAGES 72	
14. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)	15. SECURITY CLASS. (of this report) Unclassified	
15. SUPPLEMENTARY NOTES	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE	
16. KEY WORDS (Continue on reverse side if necessary and identify by block number) computer image generation motion and distance cues texture shading textured visual images	16. 6114	
17. ABSTRACT (Continue on reverse side if necessary and identify by block number) This developmental project resulted from the need to enhance visual flight simulators (based on computer image generation (CIG) techniques) with textured surfaces. Pilots viewing homogeneous surfaces experience an inadequate perception of motion above these surfaces as well as an ambiguity in the orientation of the surface. Therefore, it was felt that by using textured surfaces within the scene, both depth and motion cues would be made available. This report describes the approaches and techniques used to develop texture material and introduce it into a complex CIG scene. The approach is to create texture tile arrays of trees and grass from digitized photographs of natural texture. These arrays are then replicated over the surface in the fashion of tiles. Preprocessing techniques are described which	17. 21	

DD FORM 1 JAN 73 1473

EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

404 432

down
JOB

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

Item 20 (Continued)

eliminate the periodic effect due to the tile boundary and macropatterns within the tile itself. Level-of-detail texture tile arrays are created that match the resolution of the texture to the appropriate observer-scene viewing geometry.

The level-of-detail arrays and other descriptive parameters are merged into the existing CIG software to produce textured images. Scenes depicting a landing sequence, a divebomb, and a low-altitude flight were then generated. These images show the effect of the varying levels of detail, fields of view, viewing geometries, and scene contents on the textured scene. The texture tile technique and subsequent image generation show the feasibility of deriving distance and motion cues from the additional information contained in the textured images. 

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

TABLE OF CONTENTS

	Page
INTRODUCTION	5
Overview	5
DATA COLLECTION	8
Overview	8
Trees	8
Grass	8
SCENE DATA BASE	11
Description of the Scene	11
Data Base Construction	11
SOFTWARE STRUCTURE AND ALGORITHMS	16
Modification of Existing Software to Include Texturing	16
Texture Tile Prefiltering Techniques	16
Constructing the Texture Tile Data Array	19
The Texturing Algorithm	21
Image Postprocessing	29
TEXTURE SCENES	37
Overview of the Data Base	37
Approach to Landing Sequence	37
45° Divebomb	45
Terrain Avoidance	50
Special Views	66
CONCLUSIONS AND RECOMMENDATIONS FOR FURTHER RESEARCH	79

Accession For	MAIS GAM&I	☒	☐	☐
DDC TAB	Unannounced			
Justification				
By				
Distribution/				
Availability Codes				
Avail and/or special				
1st	A			

LIST OF FIGURES

	Page
1. Untextured Image With Constant Shading	6
2. Image With Tile Patterns	6
3. Image With Textured Shading	6
4. Primitive Tile for Trees	10
5. Primitive Tile for Grass	10
6. Three Stages of Data Base Development	12
7. Data Base Construction	14
8. Geographical Map of the Scene	15
9. Block Diagram of Routines	17
10. Averaging Process Texture Tiles	20
11. Footprint Algorithm	23
12. Sampling Algorithm	27
13. Pyramid Weighting Filter	31
14. Constructing an Impulse Response	32
15. Image Filter	33
16. Aliasing of Line Pairs	35
17. Line Pairs Filtered	36
18. Photograph Viewpoints	38
19. Scene Overview	39
20. Approach to Landing--Untextured	40
21. Approach to Landing--Textured	41
22. Approach to Landing--Textured	42
23. Approach to Landing--Textured	43
24. Approach to Landing--Textured	44
25. 45° Divebomb--Textured	46
26. 45° Divebomb--Textured	47
27. 45° Divebomb--Textured	48
28. 45° Divebomb--Textured	49
29. Terrain Avoidance--Textured, WFOV	51
30. Terrain Avoidance--Textured, WFOV	52

LIST OF FIGURES (Continued)

	Page
31. Terrain Avoidance--Textured, WFOV	53
32a. Terrain Avoidance--Textured, WFOV	54
32b. Terrain Avoidance--Untextured, WFOV	55
33. Terrain Avoidance--Textured, WFOV	56
34. Terrain Avoidance--Textured, WFOV	57
35. Terrain Avoidance--Textured, NFOV	58
36. Terrain Avoidance--Textured, NFOV	59
37. Terrain Avoidance--Textured, NFOV	60
38a. Terrain Avoidance--Textured, NFOV	61
38b. Terrain Avoidance--Untextured, NFOV	62
39. Terrain Avoidance--Textured, NFOV	63
40. Terrain Avoidance--Textured, NFOV	64
41a. Hilltop Surfaces--Textured	67
41b. Hilltop Surfaces--Untextured	68

INTRODUCTION

The objective of this project is to demonstrate and evaluate a texture tile algorithm's usefulness for generating textured representations of areas of grass and trees. Texture tiles, which consist of digitized arrays of texture data, are placed at periodic intervals over designated areas. If these areas are irregular, the texture tiles are made to conform to their boundary irregularities. Filtering, oversampling, and postfiltering are then used to diminish the problems of tile boundaries, geometric scene perspective, and aliasing.

To generate interesting and relevant textured imagery, sequences of photographs were used to depict an approach to an airfield, a dive-bomb at a 45° angle, and a terrain-avoidance flight. These photographs are included in this report along with others that illustrate the various technical aspects of employing texture tiles.

Texture tiles are shown to provide a viable means of realistically texturing areas of scenes and to afford additional information for deriving distance and motion cues. Postfiltering mitigates the critical problems of tile boundaries and aliasing.

Overview

There are three shading application methods involved in this project. The simplest method is constant shading, in which a constant shade is painted across a surface. The next simplest is smooth shading, in which a linearly varying shading value is applied across a surface. This involves a constant gradient across the surface. The third is texture tile shading, in which an array of tiles is fixed onto a surface in perspective and used to modulate the basic constant or smooth shade at each point.

Both constant and smooth shading methods are familiar and often used in the computer image generation (CIG) community. At Technology Service Corporation (TSC), a sensor image model is used to calculate the tone of the surface (constant shading case) or of each surface vertex (smooth shading case).

Texture tile shading is a natural extension of these shading techniques. A single square tile can be fixed at periodic intervals to the surface. The value at any point of the texture tile can be used to modulate the basic constant or smooth shading tone at that point.

Figures 1, 2, and 3 are presented as an introduction to the texture tile technique. The image in Figure 1 is typical of current CIG systems with constant surface shading. The same scene is shown in Figure 2 to

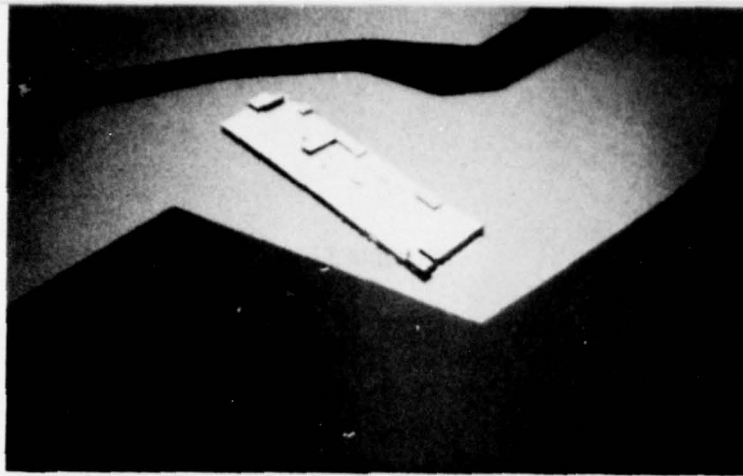


Figure 1. Untextured image with constant shading.

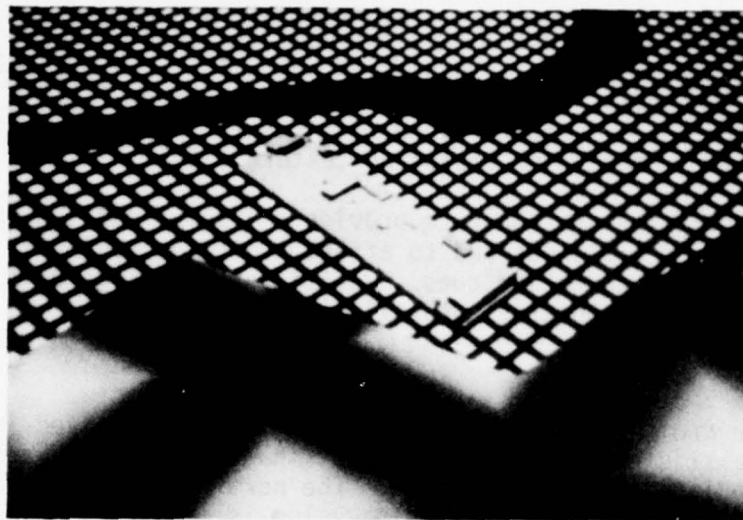


Figure 2. Image with tile patterns.

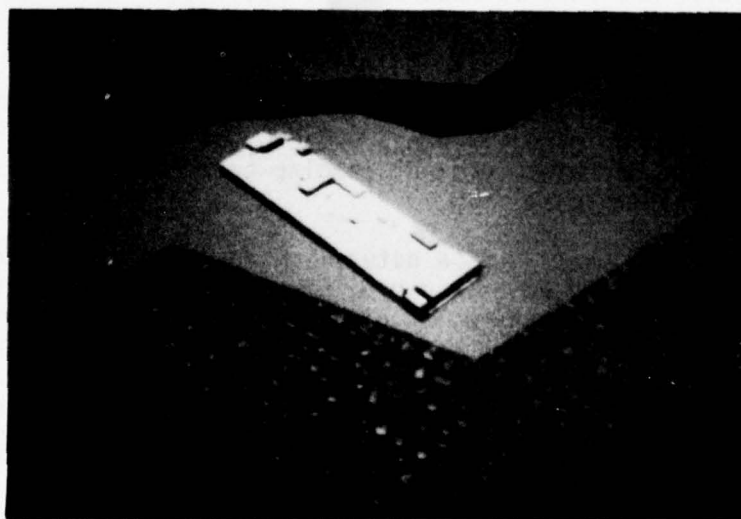


Figure 3. Image with textured shading.

depict the texture tile concept.* Here a synthetic tile has been generated and replicated in perspective over the scene in the fashion of tiles. Finally, Figure 3 shows the addition of realistic tiles to the grass and forested areas of the scene.

This report comprises sections on data collection, scene data base construction, software structure and algorithms, textured scenes, and conclusions and recommendations for further research.

The section on data collection relates the problem of acquiring aerial photographs to the properties of textured material. The intrinsic spatial frequencies of the textured material determine the altitude for obtaining the photographs. The photographic data are then digitized into an array, and a subset of the array is manually selected as the primitive texture tile. This array is then prefiltered to remove undesirable spatial frequencies. Other arrays, representing various levels of detail (LODs), are constructed in a cascading manner by averaging one array into successively smaller arrays which are used at different ranges.

The section on scene data base construction enumerates and explains the steps needed to construct a three-dimensional geometric data base. In particular, this section describes the component objects of the data base.

The section on software structure and algorithms describes 1) the existing TSC CIG system and the modification necessary to permit texturing, 2) the texturing algorithm, 3) the construction of the texture tile data array, 4) the texture tile prefiltering techniques, 5) the footprint equation, 6) the program initialization, and 7) the post-filtering of a textured scene. The prefiltering and postfiltering techniques are derived as a method to control aliasing.

The section on textured scenes describes the various effects of texture by presenting the textured material in logical sequences that, in general, have particular objectives. These photographic sequences are approach to landing, divebomb at 45°, terrain avoidance, and special views.

The section on conclusions and recommendations for further research specifies the results of this effort and identifies important texturing problems.

* Blinn, J. F., and M. E. Newell, "Texture and Reflection in Computer Generated Images," Comm. ACM, Vol. 19, No. 10, October 1976.

DATA COLLECTION

Overview

The purpose of the data collection activities was to provide the appropriate photographs of trees and grass for digitization and subsequent tile selection.

The initial attempts to obtain appropriate imagery proved to be inadequate. The original source of possible imagery for constructing the texture tiles was high-altitude aerial photography obtained from the Rome Air Development Center of the test range near Stockbridge, New York. The photographs of the Stockbridge test range, however, were not taken with the specific objective of obtaining textured material. Thus, although the photographs were suitable for the aerial reconnaissance purposes for which they were taken, they proved insufficient with respect to the homogeneity and resolution of the textured areas.

To obtain suitable material, new imagery at various altitudes was photographed by TSC. The representative imagery included grass, trees and clover. Due to difficulties (see section on grass, below), the new imagery was adequate for trees, but not for grass. The latter was finally obtained from model-railroad grass, i.e., synthetic grass.

Trees

TSC undertook its own photography of texture, first from a low-flying aircraft and then from a helicopter. Photographs of trees (see Figure 4) were taken by TSC at Chino, California, at an approximate altitude of 200 feet. At this altitude, the patch of trees is homogeneous in nature and is approximately the size required for the mid-altitude flights of this study. These trees were digitized into a 3000 by 3000 array of 512 gray levels. From this array, primitive texture tiles of 256 by 256 were subjectively selected by an operator. The criterion for selecting a tile was that it not produce a visible boundary when replicated throughout the scene. To see the effect of replication over a surface, selection software was generated that produced a 512 by 512 rendition of the selected tile. The 2 by 2 repetition of the tile matched both its upper and lower edges and its left and right edges. Thus, any boundary mismatch would be evident, as would any pattern repeated within the tile. Using this procedure, a tile was selected to be the primitive tile for the study if it produced a single, homogeneous textured area rather than a repetitive pattern of the tile itself.

Grass

As mentioned above, extreme difficulties were encountered in trying to secure photographs of grass. Several interesting phenomena were noticed. Golf courses had macropatterns that reflected the grass-cutting

pattern or were the result of different growth patterns (from lush green areas to patches of sun-dried brown grass). Also, there were special marks caused by golfers not replacing their divots. The macropatterns and "imperfections" were also present in photographs taken of parks and pastures. In short, these aerial photographs proved to be inadequate.

In addition, an attempt to utilize ordinary backyard grass also proved to be infeasible. The reason that grass, both backyard and open area, was deemed unacceptable was that criteria for acceptability was completely subjective; i.e., observers had differing mental models of what grass should look like.

The grass tile was finally created from photographs of model-railroad grass. This was an acceptable solution since the model-grass tile exhibited a completely homogeneous microstructure without macropatterns. The primitive grass tile used in the subsequent textured scenes is shown in Figure 5.

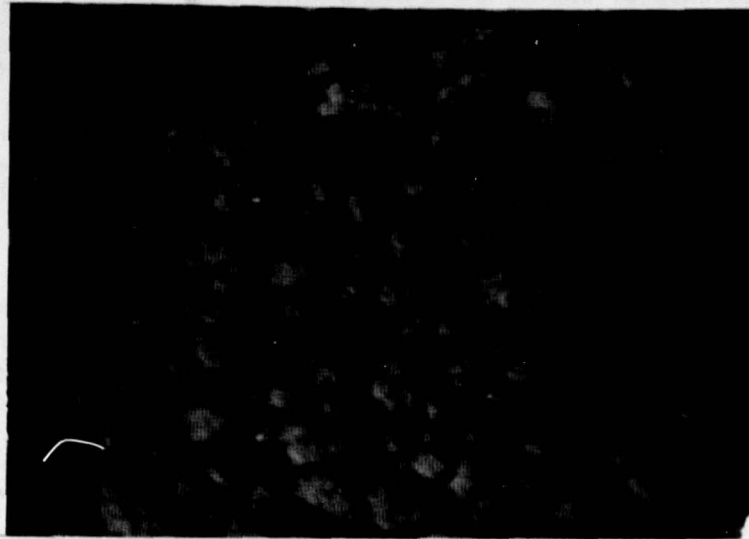


Figure 4. Primitive tile for trees.



Figure 5. Primitive tile for grass.

SCENE DATA BASE

Description of the Scene

To evaluate the effects of adding texture to an image, the textured areas must be embedded in a scene. The scene selected for this purpose is based on Area C of Wright-Patterson Air Force Base (WPAFB). Initially, only a relatively narrow strip surrounding the main runway of the base was considered for the model. Markings on the main runway were hand-encoded from available blueprints. Altogether, there were 160 markings. Several taxiways and most of the service roads adjacent to the strip were also digitized from the blueprints and were included in the model. Finally, four water reservoirs and part of the Mad River were digitized to complete the initial data base comprising 200 surfaces. The initial data base is shown in Figure 6.

To obtain a more varied terrain, simple hills (each with 20 surfaces) were added, one to each side of the original strip. This modified data base is shown in Figure 6. Adding hills to the scene left large areas without any features, thereby forcing us to construct a third version of the data base. The minor access roads were removed, and the major roads, freeways, and waterways were modeled.

As a next step, several artificial features were modeled. Seven tree-covered areas were added to the scene to fill the open areas between roads. These tree-covered areas not only fill out the scene, but, of course, are the vehicle for including tree texture in the scene. The entire remaining ground plane is textured with grass.

As a final step, six buildings with windows and doors were placed alongside the runway approach. A plan view of all the features is shown in Figure 6. In the total data base, there are 743 surfaces with 4725 edges and 4125 vertices.

Data Base Construction

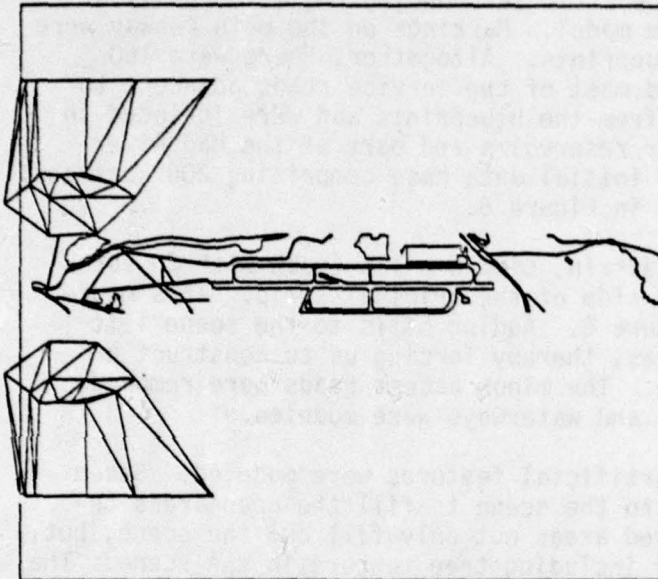
This section describes the process for constructing the scene model and its supporting data base.

The scene data base was derived with the aid of an aerial photograph of the area around and including WPAFB, the corresponding U.S. Geodetic Service (USGS) topographic map, and supporting engineering diagrams of certain areas and features of the base itself. It is not possible to generate a model on a digital computer that contains all the details contained in these source materials.

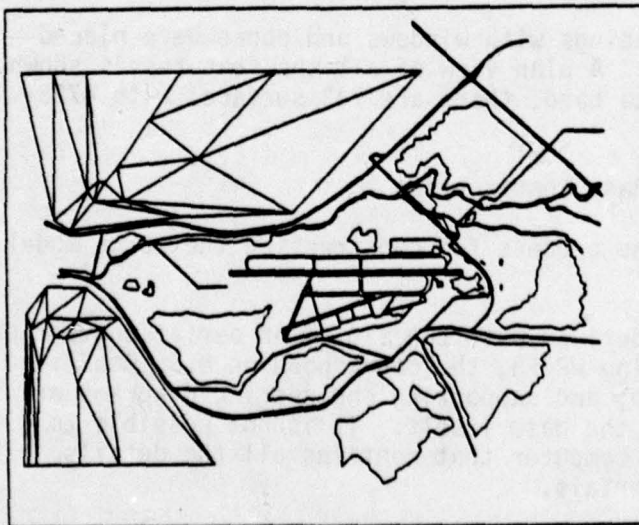
It is therefore necessary, in the first step of the data base construction, to examine the scene and select the features to be included in the model. Each of the selected features is then examined and its



a. Initial Data Base



b. Data Base with Terrain



c. Complete Data Base

Figure 6. Three stages of data base development.

model is constructed. In general, this model comprises a polygonal surface (or a set of polygonal surfaces) together with coordinates of the polygon's vertices and a "tonal descriptor." Finally, each of the individual surfaces is entered into a "source table" which is then processed by the display algorithm into an image on a cathode-ray tube (CRT).

The following discussion on data base construction is diagrammed in Figure 7. At present, the source table is constructed in three separate stages. First, available information must be encoded in a digital form. This is usually done with the help of a digitizer board. When the maps or blueprints fit the board, necessary coordinates can be directly sensed by the digitizer pen and transferred into a card file via analog-to-digital (A/D) converter. Otherwise, the printed information can be either redrawn in smaller scale or digitized in several smaller parts. For the textured model, roads and waterways were digitized from the supplied geographical map of the area, as shown in Figure 8. The runway and taxiways were digitized from the blueprint of the airbase. Coordinates of some of the features can also be entered manually, with higher accuracy (in terms of significant digits). These files can then be drawn on the plotter to ensure that the individual features have correct outlines (i.e., a vector-drawn image of the surfaces can be produced--Figure 6c). Before proceeding to the next stage, tonal, thermal, and any other specifications must be added to each digitized feature. These are entered by means of a "parameter card" that precedes each feature.

At this point, several files exist which may each contain features in different scales that may not be oriented correctly with regard to features in the other files. In the second stage, therefore, data in each of the files must be scaled and positioned with respect to a common coordinate system. This process is performed with the use of a "development language" compiler that converts the digitized files into an ASCII file. The conversion permits the user to verify and modify the data base. The scene data can also be entered directly into the compiler, thereby bypassing the previous steps. For example, the runway markings were constructed in this manner to obtain better accuracy. Additional parameters for later use by the display routines can be added at this point. For example, one additional parameter in this model was the "distance-to-the-eye test" parameter that determines the range at which the runway markings become visible. Once the data base is verified, the construction proceeds to the last stage.

A second software module, the data base compiler, converts the data from the existing format to one directly compatible with the image generation routines. First, several parameters required for the image generation (e.g., surface normals and bounding polyhedra) are calculated by the compiler. Then the compiler compresses the ASCII files into binary form so that the images can be more efficiently generated by the CIG routines.

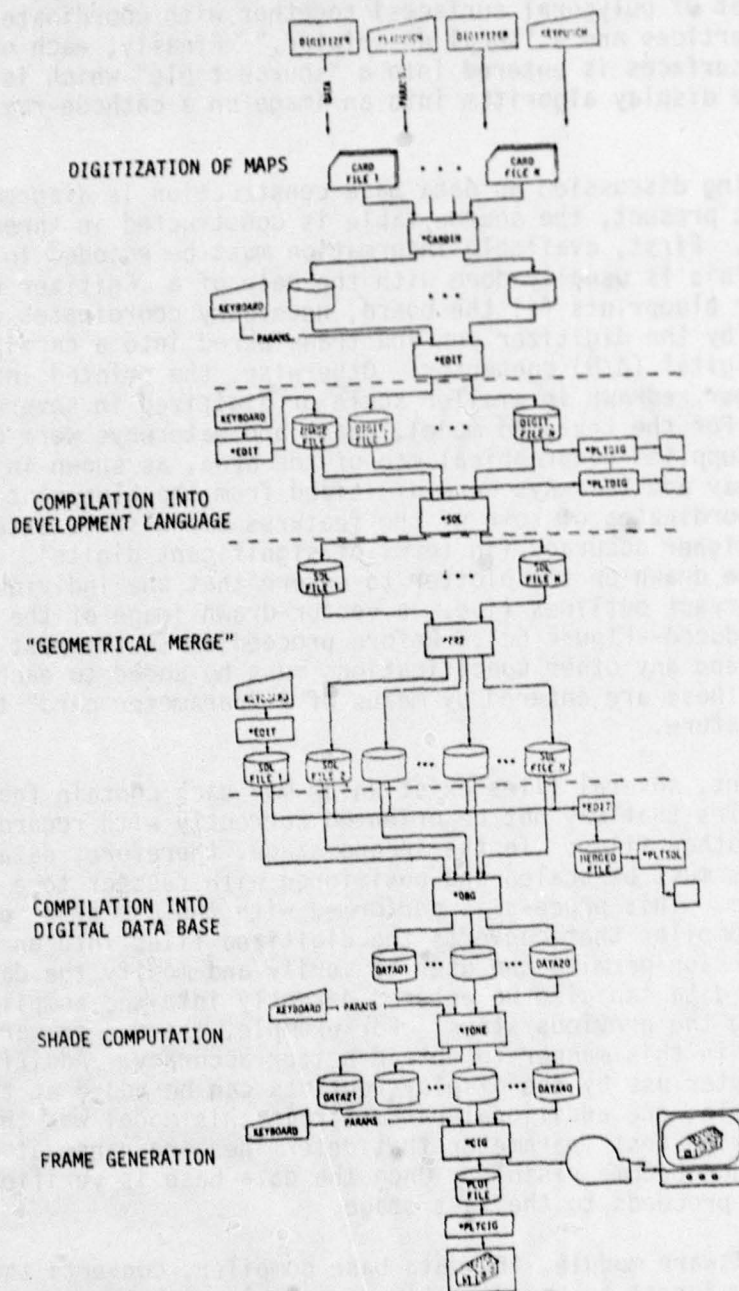


Figure 7. Data base construction.

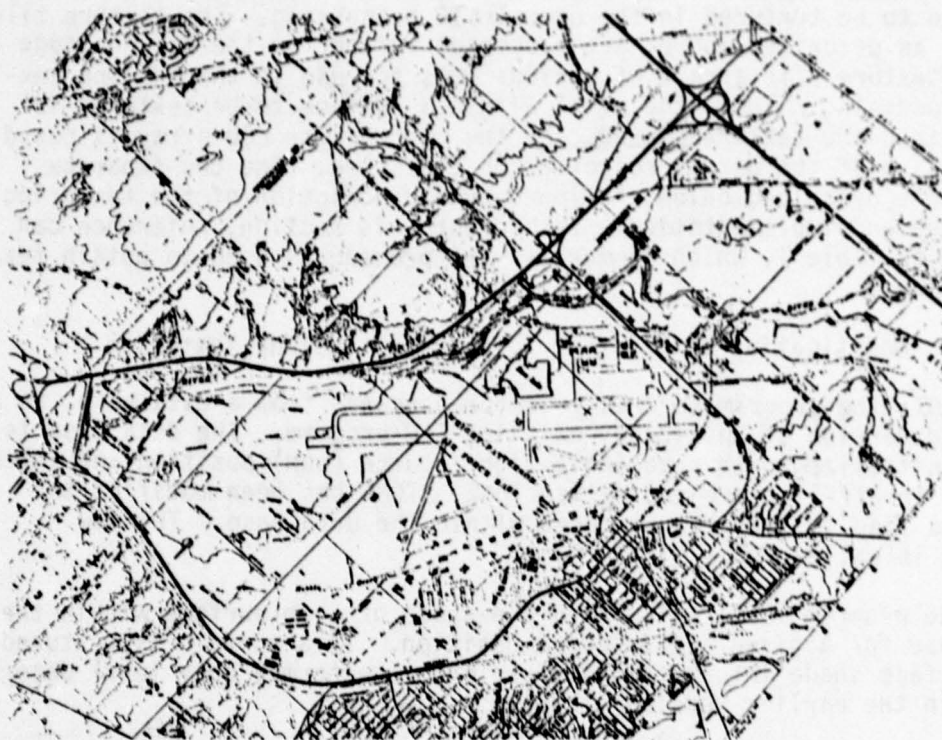


Figure 8. Geographical map of the scene.

THIS PAGE IS BEST QUALITY PRACTICABLE
FROM COPY FURNISHED TO DDC

SOFTWARE STRUCTURE AND ALGORITHMS

This section includes a general description of the texturing technique, an in-depth walk-through of the software, and a derivation of the basic algorithms. Figure 9 depicts the routines. There are two off-line texture processing stages. The first stage identifies surfaces that are to be textured in the normal CIG processing. The texture tile arrays, as described above, are processed further in the second stage to create texture tile arrays of various LODs for use in the on-line texture processing. There the pixel within a surface to be textured is mapped into the textured scene, and the appropriate LOD array is based on the size of the pixel projection on the scene. The two separate stages are described below, followed by a description of the texturing technique and implementation. Throughout this section, reference can be made to Table 1, which summarizes the procedures used to obtain textured images.

Modification of Existing Software to Include Texturing

CIGT, the program to create textured scenes from a data base, is a modified version of an already existing CIG program. The data base is first initialized with a variable light source (s.l.n) position and intensity in an off-line program called TONE. TONE has been modified to attach a flag to textured surfaces within the data base. The new version is called TONET.

The program TONET calculates the shade of each surface within the data base for a given light source position. If a surface is textured, the surface shade information differs from the normal gray level determined in the earlier version (TONE).

For every textured surface, the sign bit of the shade value is set as a flag, and the texture tile number (e.g., 1 = trees, 2 = grass) is packed within the word containing the surface shade value. This information provides a flag to indicate that the surface is textured, along with the average shade of the surface due to the sun position. The information also identifies the texture tile to use to cover the surface, all within an already existing framework. The texture flag, tile number, and surface shade are contained in a 16-bit integer.

Texture Tile Prefiltering Techniques

A tile may be averaged after it has been selected. The purpose of averaging is to improve the boundary matching statistics and help eliminate texture macropatterns. In the process, the tile is replicated at the boundaries so that smoothing occurs across the boundary. This averaging is done by convolving a unit step filter of a predetermined size over the tile. At each texture tile array element, the local average of the filter (a) is compared with the overall tile average (A),

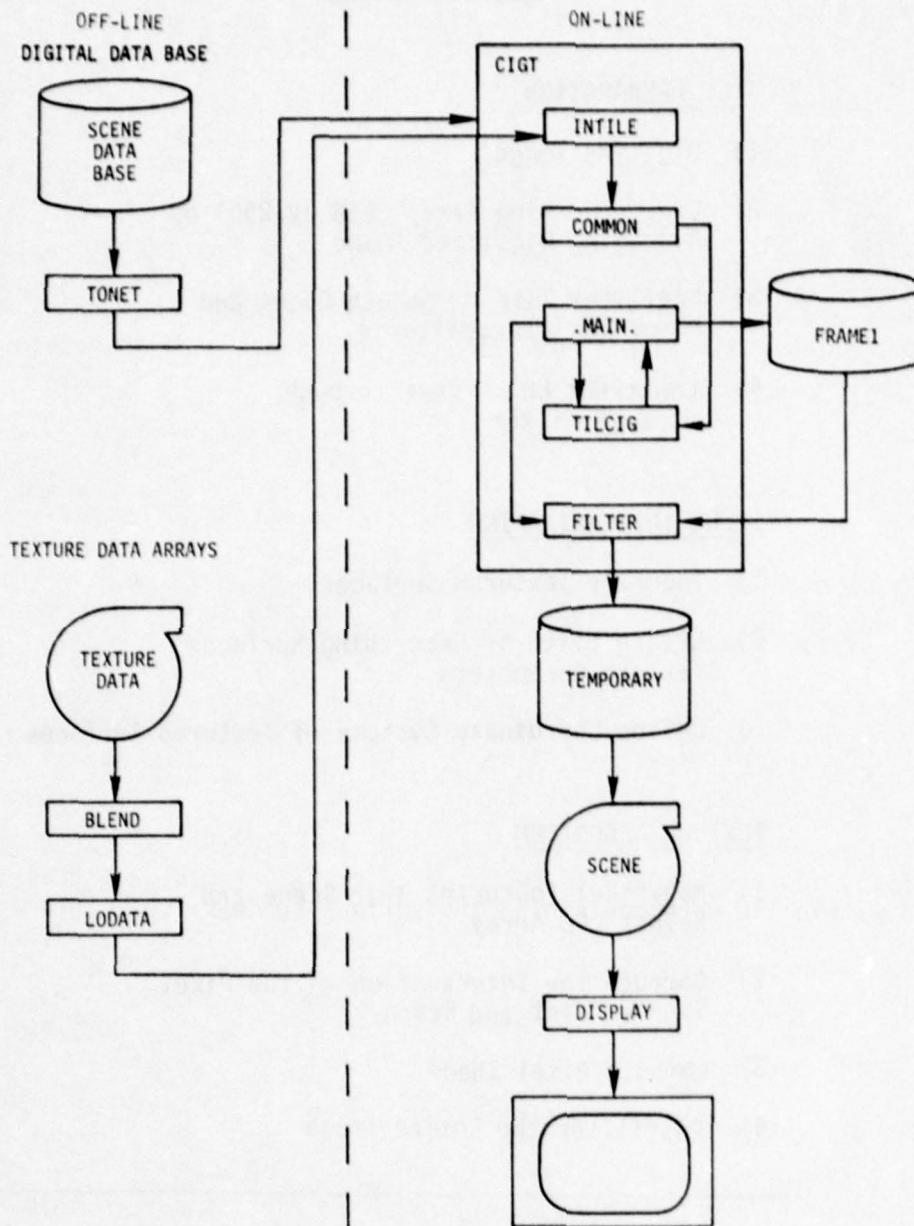


Figure 9. Block diagram of routines.

*Table 1. Summary of Procedures to
Create Textured Images*

TILE PREPARATION

- 1) Digitize Image
- 2) Create Working Array (256 by 256) by
Averaging Digitized Image
- 3) Prefilter Tile to Smooth Edges and
Eliminate Macropatterns
- 4) Construct LOD Arrays through
Pyramid Filter

SCENE INITIALIZATION

- 1) Identify Textured Surfaces
- 2) Create Files of Describing Surface/
Texture Parameters
- 3) Define Coordinate Systems of Textured Surfaces

TEXTURE ALGORITHM

- 1) Map Pixel Footprint into Scene and
Select LOD Array
 - 2) Compute the Intersection of the Pixel
Sample Point and Scene
 - 3) Compute Pixel Shade
 - 4) Postfilter the Entire Image
-

and the array value is adjusted by the ratio a/A . The results of the averaging process are shown in Figure 10. The first thing to note is that the general shape of the gray level distribution is preserved so that the tile has the same general appearance. However, the variance has been reduced in the smoothing process. The averaging program is called BLEND.

This filtered/smoothed tile is then used in the program LODATA to calculate each of the LODs needed in the texturing algorithm. A pyramid function is used to filter this basic tile to lower LOD arrays. Each LOD is a power of two smaller in size than the previous LOD. The coarsest LOD is a single value, i.e., the average value of the tile. The construction of the LOD tile arrays is further described below in the discussion on the footprint equation.

Constructing the Texture Tile Data Array

In addition to the scene data base, a data array of the LODs of the various texture types is needed to create a textured scene. The program LODATA constructs the texture tile data array off-line for use with the program CIGT. Once constructed, the array need not be altered but may be appended with future texture tiles as provided for in the software.

The data arrays used exclusively in CIGT for texturing surfaces consist of the six files listed in Table 2. File 1 identifies the texture tile used for a given surface. File 2 lists several surface parameters needed to calculate the textured shade value within TILCIG. Files 3 through 6 are generated off-line with LODATA.

Table 2. Texture Tile Data Array Files

File 1:	Tile number by surface number
File 2:	Surface normal, surface x-axis, surface y-axis, surface first vertex, surface normal dot first vertex
File 3:	Dimension of the finest LOD tile by tile number
File 4:	Texture tile data pointer by tile number
File 5:	Texture tile units by tile number
File 6:	Texture tile data array with LODs

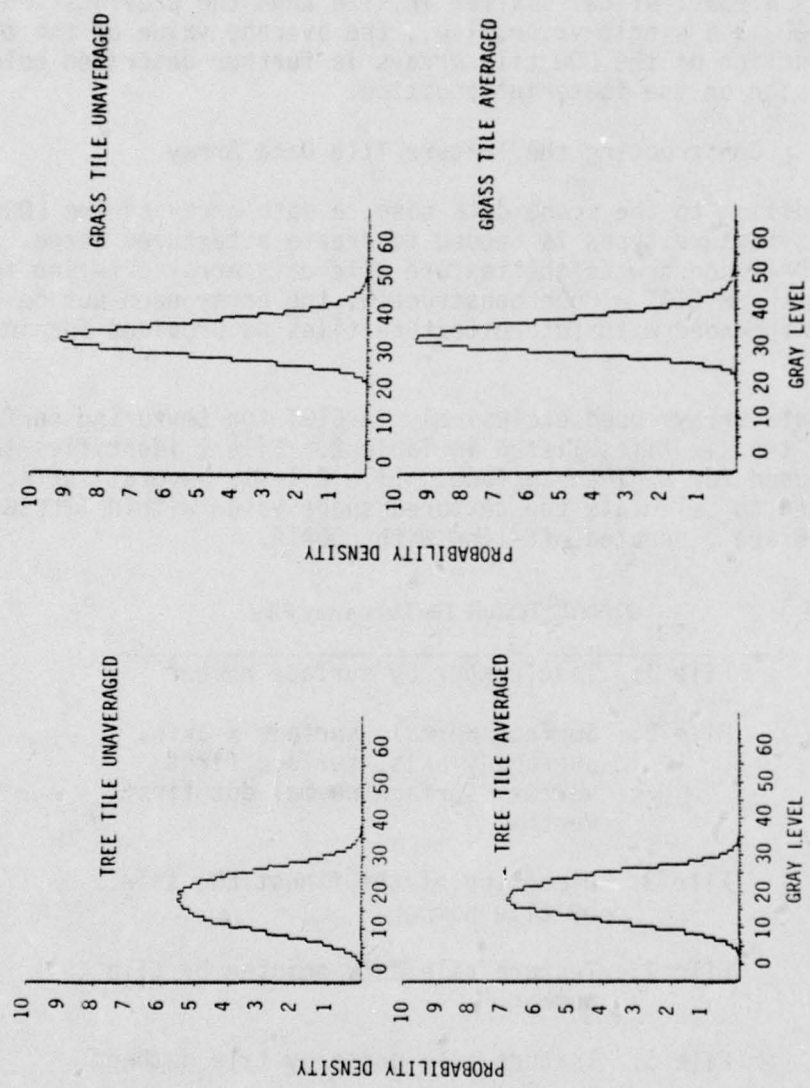


Figure 10. Averaging process texture tiles.

The data in Files 3 through 5 are indexed by texture tile number, which is contained in File 1 for each surface. Currently, 1 = trees and 2 = grass. Presently, there is space in these arrays to reference 500 possible surfaces with the two tiles. The number of surfaces or tiles permissible can easily be changed by increasing the array dimensions.

File 3 contains the texture tile array for the finest LOD. For trees and grass, the array is 128 elements along each side. File 4 contains the index value that points to the beginning of each separate texture tile array at the lower LODs. These other arrays are contained in the last file. A single texture tile with a finest LOD of 128 and coarser LODs of 64, 32, 16, 8, 4, 2, and 1 contains 21,845 array elements. With this texture tile data array stored on disc, the disc size is the only limitation to expanding texture tile techniques further.

File 5 contains the actual size of each texture tile in meters. This is used to scale the texture tiles to correspond with the scene units. The tile is 150 meters for trees and 15 meters for grass.

The Texturing Algorithm

The texturing version of the main scene-developing program is named CIGT. Several modifications were incorporated into the program CIG to implement the texturing algorithm. Program TONET sets a flag within the data base that allows a branch point to separate textured surfaces from the main path in program CIGT. The main path determines the visible surface for each display pixel. If the surface is textured, the texturing subroutine TILCIG is called. (Refer to Figure 9.)

TILCIG returns a surface shade at the point where the pixel line of sight intersects the scene. To calculate this shade value, TILCIG must have the average surface shade, the tile average shade value, the texture tile mapped onto the surface, the proper tile LOD due to the object's distance from the observer, and the tile array element of the point of intersection. The ratio of the average surface shade to the average tile shade is used to bias the tile array element value so that it relates to the simulated lighting conditions. A surface shaded to simulate curvature can also be textured without losing the curvature aspect. Each pixel value is similarly calculated to create a display of the scene. The mapping of the tile, the tile LOD, and the point of intersection are described next.

Program Initialization--The program CIGT reads in Files 3 through 6 that were developed by LODATA in the initialization process. These files contain the LOD arrays along with their dimensions, scales, and starting addresses. The data bases are also read in at this time, and the surfaces are sorted and translated into the observer coordinate system. The information needed in TILCIG is cataloged in Files 1 and 2 by an assigned surface number. The flat set by TONET on the surface shade indicates a textured surface. The tile number is now extracted from the word containing the surface shade value and placed in File 1, indexed by surface

number. File 2 is filled with the following information: surface normal, x-axis, y-axis, first vertex, and the inner product of the surface normal with the first vertex (a calculation that would otherwise be repeated every time the sample point is calculated--see the section on the texturing subroutine, below).

The x-axis and y-axis are determined so as to fix an invariant coordinate system to every textured surface. A unit vector from the surface first vertex to the second vertex is defined to be the surface x-axis. The cross-product of this x-axis with the surface normal defines the y-axis. With this fixed coordinate system, the texture tiles can be mapped onto the surface. Since the first and second vertex points are fixed within the data base, the texture tile is fixed to each surface in a consistent manner that is not dependent on observer position.

The Footprint Equation--The proper LOD tile array is indexed as a function of the size of the footprint on the scene; i.e., the size of the pixel as mapped into the scene. The footprint equation is derived below. Figure 11 shows the observer-display-scene geometry; the quantities shown are defined as follows:

Δa	Pixel size
d	Distance to display
$\Delta a/d$	DELTA
$\Delta \theta$	Sampling angular increment
ZN	Surface normal
$F1$	Minimum footprint diameter
$F2$	Maximum footprint diameter
L	Line of sight to intersection (XLINE, YLINE, ZLINE) in scene
θ	Angle between sample L and ZN

It can be seen from Figure 11 that for small $\Delta \theta/2$ and $\theta < \pi/2$,

$$\cos \theta \cong \frac{(F1/2)}{(F2/2)}$$

since the distance to the scene is large.

Define ℓ to be a unit vector

$$\ell = L/(L \cdot L)^{1/2}$$

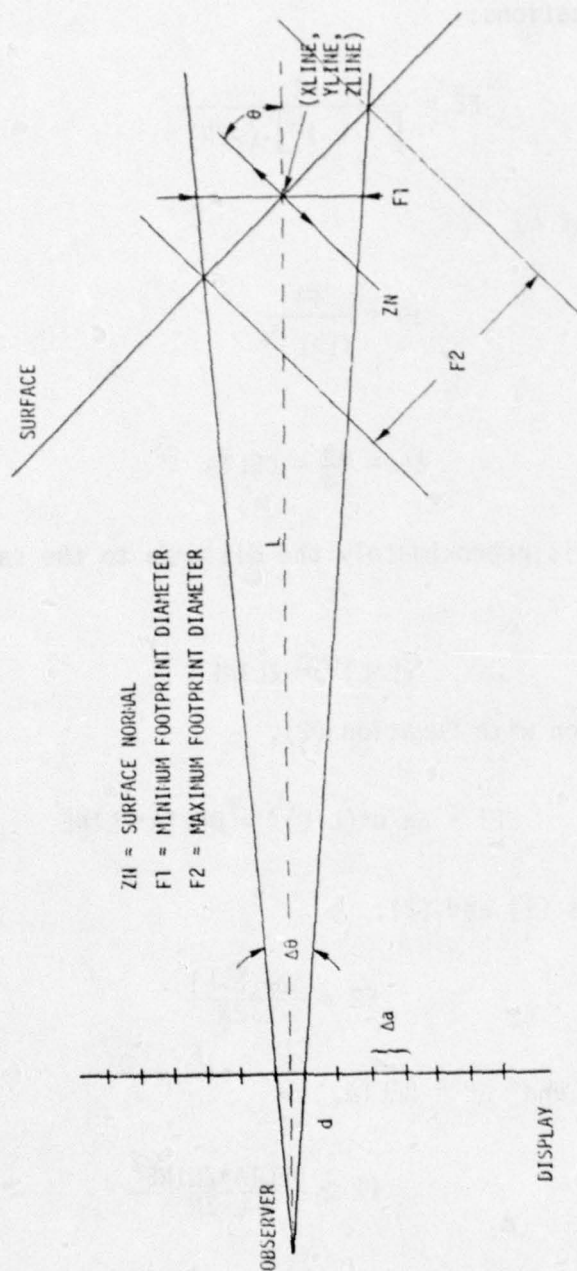


Figure 11. Footprint algorithm.

where $(L \cdot L)^{\frac{1}{2}}$ is the distance to L or $|L|$. For unit vectors ℓ and ZN ,

$$\ell \cdot (-ZN) = \cos \theta$$

From the above equations:

$$F2 = \frac{F1}{[L / (L \cdot L)^{\frac{1}{2}}] \cdot (-ZN)} \quad (1)$$

For small values of $\Delta\theta$,

$$\Delta\theta \approx \frac{F1}{(L \cdot L)^{\frac{1}{2}}} \quad (2)$$

and

$$\Delta\theta \approx \frac{\Delta a}{d} = \text{DELTA}$$

Assume that ZLINE is approximately the distance to the sample point, i.e.,

$$(L \cdot L)^{\frac{1}{2}} \approx \text{ZLINE} \quad (3)$$

Then, in conjunction with Equation (2),

$$F1 \approx \Delta a / d \cdot (L \cdot L)^{\frac{1}{2}} \approx \text{DELTA} \cdot \text{ZLINE} \quad (4)$$

Combining Equations (1) and (2),

$$F2 = \frac{\Delta\theta \cdot (L \cdot L)}{-L \cdot ZN}$$

Using Equation (3) and $\Delta\theta = \text{DELTA}$,

$$F2 \approx \frac{\text{DELTA} \cdot \text{ZLINE}^2}{-L \cdot ZN} \quad (5)$$

The average of F1 and F2 is the average footprint diameter

$$F = \frac{(F1 + F2)}{2}$$

Combining Equations (4) and (5),

$$F \approx \frac{(\text{DELTA} * \text{ZLINE}) - \text{DELTA} * \text{ZLINE}^2 / (\text{L} * \text{ZN})}{2}$$

$$\approx \frac{\text{DELTA} * \text{ZLINE}}{2} \left(1 - \frac{\text{ZLINE}}{\text{L} * \text{ZN}} \right)$$

This is the footprint equation used to index the proper LOD as outlined in Table 3.

Table 3. LOD Boundaries

<u>Footprint Average Diameter</u>	<u>LOD</u>	<u>Array Dimension</u>
$3/4 \leq f$	LOD1	(1)
$3/8 \leq f < 3/4$	LOD2	(2,2)
$3/16 \leq f < 3/8$	LOD3	(4,4)
$3/32 \leq f < 3/16$	LOD4	(8,8)
$3/64 \leq f < 3/32$	LOD5	(16,16)
$3/128 \leq f < 3/64$	LOD6	(32,32)
$3/256 \leq f < 3/128$	LOD7	(64,64)
$0 \leq f < 3/256$	LOD8	(128,128)

If the average diameter of the footprint is greater than three-quarters of the tile size, then the coarsest LOD is used (i.e., the average shade). If the average diameter of the footprint is greater than half of the previous limit, or three-eighths of a tile, the next LOD is used; then half of that, and so on until the finest LOD is reached.

Texturing Subroutine--TILCIG is called for each pixel. The parameters passed in the CALL statement are the display pixel angle in the x and y directions with respect to a line of sight through the center of the display, as described below. These angles, ALPHA and BETA, are expressed in terms of tangents. The surface number and surface shade, with respect to a light source, are also passed. Within COMMON, the sample interval and scene units in meters are stored as constants.

The first calculation made in TILCIG is to determine the intersection of the sample line of sight with the scene. This calculation uses the sampling algorithm described next (see Figure 12).

The observer z-axis intersects the display at its center. The x-axis is defined by a horizontal line through this center point in the plane of the display, and the y-axis is defined by a vertical line. Any pixel on the display (a,b) is a point on a line from the observer through the display at this pixel location to the scene. ALPHA is the slope of the line in the x,z plane, and BETA is the slope of the line in the y,z plane:

$$\text{ALPHA} = a/d$$

$$\text{BETA} = b/d$$

The observer position is the origin in this coordinate system. Thus, the respective two-dimensional lines follow the general form $Y = MX + B$ and, since the lines pass through the origin, $B = 0$.

$$\text{XLINE} = \text{ALPHA} * \text{ZLINE} \quad (6)$$

$$\text{YLINE} = \text{BETA} * \text{ZLINE} \quad (7)$$

At some ZLINE, the line intersects a surface within the scene. The point of intersection (XLINE, YLINE, ZLINE) and the surface first vertex, V1(J), define a vector that lies within the plane of the surface:

$$\text{XVECTR} = \text{XLINE} - \text{V1}(1)$$

$$\text{YVECTR} = \text{YLINE} - \text{V1}(2) \quad (8)$$

$$\text{ZVECTR} = \text{ZLINE} - \text{V1}(3)$$

The surface normal vector, ZN, is perpendicular to every vector in the surface plane, and the inner product of perpendicular vectors is zero. Using this fact the following equations can be set up:

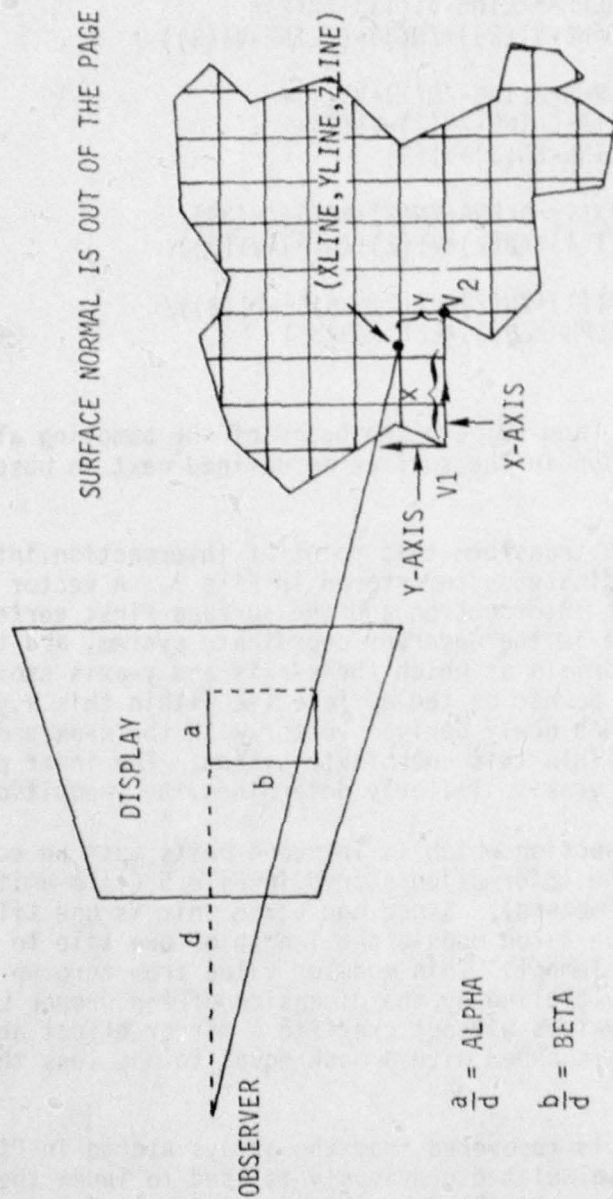


Figure 12. Sampling algorithm.

$$\begin{aligned}
0 &= ZN(1)*XVECTR+ZN(2)*YVECTR+ZN(3)*ZVECTR \\
0 &= ZN(1)*(XLINE-V1(1))+ZN(2)*(YLINE-V1(2))+ \\
&\quad ZN(3)*(ZLINE-V1(3)) \\
0 &= ZN(1)*(ALPHA*ZLINE-V1(1))+ZN(2)* \\
&\quad (BETA*ZLINE-V1(2))+ZN(3)*(ZLINE-V1(3)) \\
0 &= ZN(1)*ALPHA*ZLINE-ZN(1)*V1(1)+ \\
&\quad ZN(2)*BETA*ZLINE-ZN(2)*V1(2)+ \\
&\quad ZN(3)*ZLINE-ZN(3)*V1(3) \\
1 &= ZLINE*(ZN(1)*ALPHA+ZN(2)*BETA+ZN(3))/ \\
&\quad (ZN(1)*V1(1)+ZN(2)*V1(2)+ZN(3)*V1(3)) \\
ZLINE &= (ZN(1)*V1(1)+ZN(2)*V1(2)+ZN(3)*V1(3))/ \\
&\quad (ZN(1)*ALPHA+ZN(2)*BETA+ZN(3)) \quad (9)
\end{aligned}$$

Equations (6), (7), and (9) are the basis of the sampling algorithm. The point of intersection in the surface is defined next in observer coordinates.

The next step is to transform this point of intersection into the surface invariant coordinate system stored in File 2. A vector is derived from the point of intersection and the surface first vertex, Equation (8). Both are in the observer coordinate system, and the first vertex establishes an origin at which the x-axis and y-axis stored in File 2 are fixed. All points on the surface lie within this x,y plane. The inner product of this newly derived vector with the x-axis determines the x-position within this coordinate system. The inner product of the vector with the y-axis similarly determines the y-position.

The point of intersection which is in scene units must be scaled into tile units with the information stored in File 5 (tile units) and COMMON (scene units in meters). Since one scene unit is one tile unit, this x,y position may be taken modulo the length of one tile to locate the proper tile array element. This modulus value from zero up to but not including one is multiplied by the dimension of the proper LOD array. To eliminate negative values without creating a mirror effect about either axis, the value is ANDed with a mask equal to one less than the LOD.

The textured shade is recovered from the arrays stored in File 6. The footprint size as calculated previously is used to index the proper LOD array. The shade is multiplied by the ratio of the surface shade to the average shade of the tile to yield the final pixel shade.

Image Postprocessing

At this point in the process, the textured image is filtered to reduce aliasing. In fact, filtering was used to generate the LOD arrays in order to control aliasing. Since the control of aliasing is important in both conventional and textured CIG, some general properties of filtering as an aliasing control are discussed below, prior to the particular application of filtering in this program.

Aliasing Control--The predominant CIG use of aliasing control was in edge smoothing because, with constant or smooth shading, this is the only place in the image where high spatial frequency changes occur.

Two basic approaches to aliasing control were tried. One approach is to base the shade of a pixel on the area-averaged shade of each surface in a pixel. Both constant and variable weighting were used with sampling only within the pixel or partially outside. The pyramid function with a base of 2 by 2 pixels has been shown by Crow* to effectively control aliasing. This function is close to the Gaussian impulse response, which matches both the CRT and eye impulse responses. In this sense it is the optimal response.

The Nyquist sampling criterion is useful because it states that the maximum spatial frequency that can be represented properly (i.e., without aliasing) is $1/(2\Delta X)$ where ΔX is the interval between samples. Thus, any energy in spatial frequencies above $1/(2\Delta X)$ is aliased downward to a lower frequency.

Filtering can also be used to control aliasing. The effect of filtering the original frame by a Gaussian impulse response of unit area and variance, σ^2 , equal to $(2\Delta X/\pi)^2$ is to weight the spatial frequencies of the original image by a Gaussian function with variance, σ_1^2 , equal to $[1/(4\Delta X)]^2$. The point is that when the sample interval is at the pixel spacing, the "2- σ " point of the Fourier transform of the image is at the critical sampling frequency of 1/2 per pixel. For example, in the limiting case of white noise (i.e., the infinite high frequency spectral energy), less than 5 percent of the energy is misrepresented. Thus, the Gaussian function plays an important role in aliasing control. Also, since the pyramid function of base 2 by 2 is a close approximation to the Gaussian function of variance $(2\Delta X/\pi)^2$, it too is important in the control of aliasing.

There are other functions that have been convolved with an image to control aliasing. The sine function is fundamentally tied to sampling and spectral analysis, but it produces bell ringing parallel to edges in any practical application. It also does not match the eye or CRT impulse responses.

*Crow, F. C., "The Aliasing Problem in Computer Generated Shaded Images," Comm. ACM, Vol. 21, No. 11, November 1977.

This discussion of aliasing control has focused on the filtering of a continuous image by use of a pyramid or Gaussian impulse response. It has been shown that the variance can be set so that the bulk of the weighting falls under the Nyquist critical frequency. In the example, the variance was set so that, in the frequency domain, the critical frequency falls at the 2σ point, and 95+ percent of the energy is below the critical frequency.

The actual amount of energy aliased depends on the image. The pulse and ramp functions generated by constant and smooth shading have spectrums very heavily weighted toward low frequencies so that very little energy is aliased.

Textured shading has high frequencies that need control, and the above theory is general and powerful enough to supply a general solution to spatial aliasing. A pyramid impulse response can control aliasing.

Creation of the LOD Arrays--The pyramid weighting function was used in the creation of the texture tile arrays. Each tile sample in the LOD arrays is a pyramid weighted average of the surrounding values of the basic 256 by 256 tile that was created from the original photo-digitized array. The digitizer itself is an area sampler rather than a point sampler. In the photo-digitization process, a small square area about each sample point of the negative is illuminated from below, and the receiver centered above the unmasked area picks up the light intensity that passes through. The greatest contribution comes from the central points of the unmasked area. Thus, each array element of the basic texture tile is the result of a continuous weighting (by the aperture distribution function) of the area about it.

To increase the quasi-continuous nature of the texture tile technique, LOD arrays were created. Each texture tile array element is derived from several of the digitized values in the neighborhood of each array element. The area samples from the digitizer are weighted and summed by a pyramid function to arrive at each texture tile array element for the proper LOD. This result is a pyramid function of area samples (nearly continuous sampling).

The creation of the 128 by 128 LOD array from the basic 256 by 256 array is shown in Figure 13. Rather than just averaging in a 2 by 2 window in the N th array to generate a sample in the $(N-1)$ th array, a 4 by 4 window was used with the relative pyramid weighting as shown. Each succeeding LOD array is generated from the basic 256 x 256 array by filtering with a window of increasing size (by a factor of 2) with the corresponding pyramid function weights.

Oversampling and Postfiltering--As mentioned previously, Crow shows that a continuous pyramid filter with a base that is two pixels wide is an effective aliasing control. The ideal condition would be to sample the textured image continuously and filter the continuous image plane.

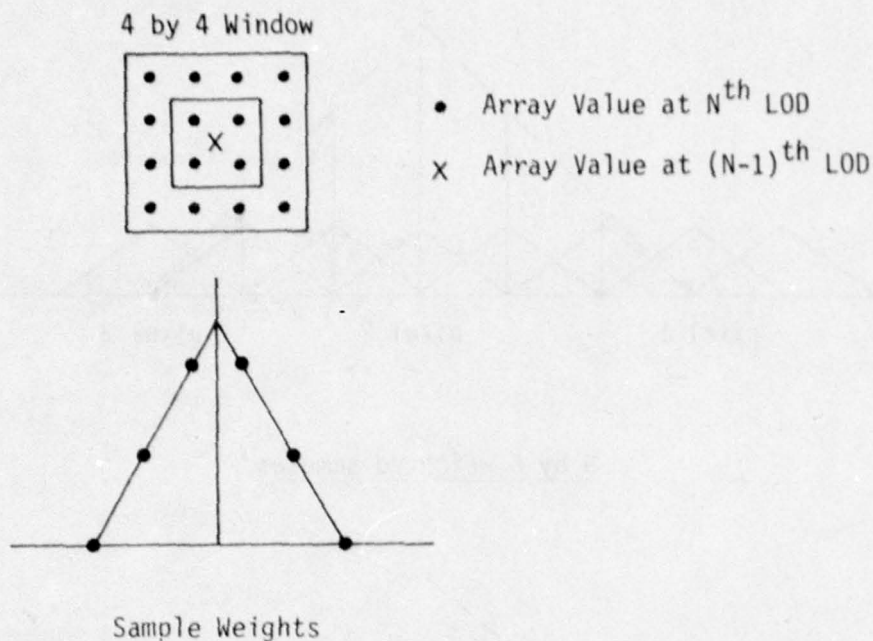
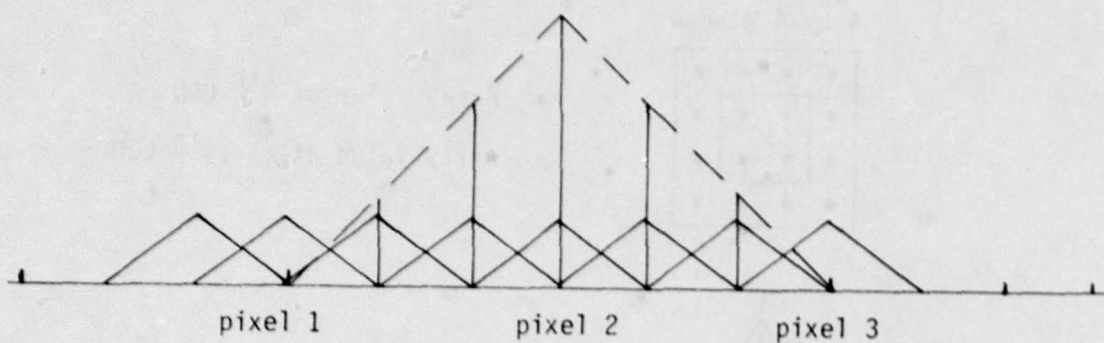


Figure 13. Pyramid weighting filter.

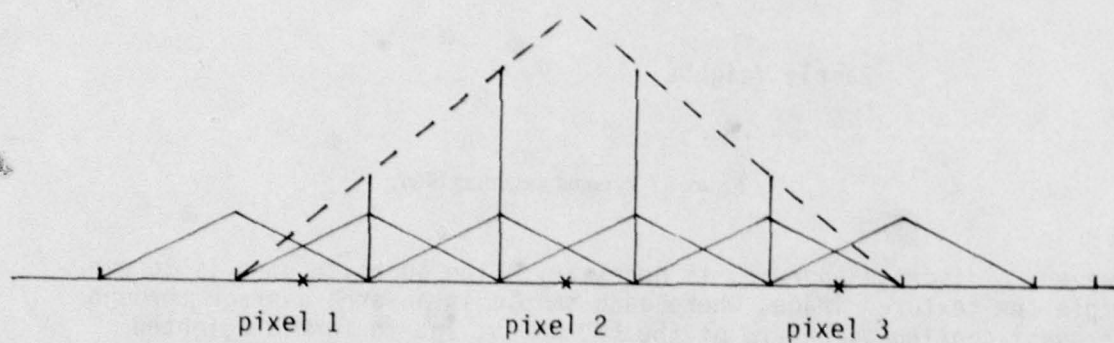
However, a discrete approach is necessary. The approach used is to over-sample the textured image, where each sample is an area average through the quasi-continuous nature of the LOD array, and to form a weighted average of these samples with a composite weighting that is a pyramid function.

The use of a small number of samples (such as 4 by 4 or 5 by 5) within the pyramid window still results in an acceptable approximation to a Gaussian function. The ensemble of samples from the proper LOD tile array are averaged together with a pyramid weighting as shown in Figure 14. The figure illustrates that when the tile samples exactly correspond to the screen samples (i.e., pixel projections) in center and extent, a pyramid function is constructed. Note that the center of the pyramid is leveled in the case of an even window size.

The use of a tile LOD introduces noise to the sampling process when the center of the tile sample does not exactly coincide with the screen samples or when the extent of the tile pyramid does not exactly match the projected pixel boundary. This noise does not present a difficulty because it varies as $1/N^2$, where N is the number of samples forming the pixel impulse response. Reflecting on this, if the tile samples exactly matched the screen samples, only one sample per pixel would be needed.



5 by 5 weighted samples



4 by 4 weighted samples

— individual tile samples
 ----- pixel weighting of samples

Figure 14. Constructing an impulse response.

Classical statistics show that for N independent samples of any distribution, the estimate of the mean value is unbiased and has variance proportional to $1/N^2$. In this case, the mismatch between screen and tile samples is independent unless the geometry is specified. Since a large number of surface/screen encounter geometries occur in a given image, the samples are independent. Thus, the variance of the error between absolute and estimated pixel shades is proportional to $1/N^2$. Typical cases involve 4 by 4 or 5 by 5 windows with samples both inside and outside the pixel so that N is either 16 or 25.

As explained above, the estimation error depends on the actual tile used. A tile that varies rapidly from sample to sample produces a higher noise in the estimation of pixel tone. Noise still varies as $1/N^2$, but the noise level starts higher. This case should not happen in practice because a tile that varies rapidly, sample to sample, is undersampled, and aliasing has probably occurred in tile construction. Practical tiles vary slowly from one sample to the next, so the mismatch does not produce much difference between actual and estimated tones. The sample-to-sample variation of the tree and grass tiles is slow due to the averaging process used in creating the original tiles.

The oversampling rate used in this program was 4 to 1. That is, each quarter-pixel was projected back into the scene, the appropriate LOD array for the subpixel was chosen via its footprint, and the proper array element corresponding to the intersection was used as the subpixel shade. These area samples were then averaged in a pyramid filter with a base that was two pixels wide, i.e., a 4 by 4 sample window. The window for the image filter is shown in Figure 15. The filter is identical to the pyramid filter used to construct the first LOD tile array (Figure 13).

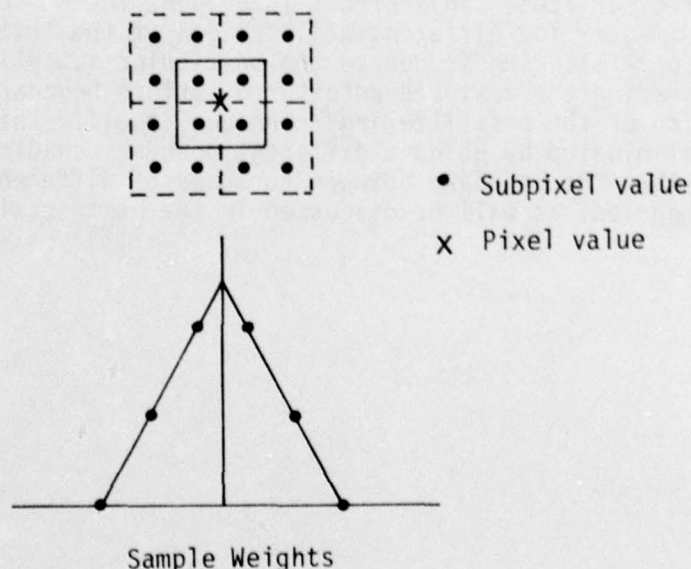


Figure 15. Image filter.

The aliasing associated with imaging closely spaced line pairs can be used to show the effect of the pyramid postfilter as Crow has done. Line pairs exhibit moire patterns which take on the appearance of ripples on a pond, as shown in Figure 16. The 16-point pyramid filter, with a base that is two pixels wide, eliminates most of the aliasing, as shown in Figure 17. The pyramid filter as implemented for this experiment does have a discrete number of point samples, and the parallel line problem cannot be completely eliminated in all cases. To reduce this high frequency aliasing, it is necessary to either increase the number of samples or make the sample points area samples, as has been done in the texture tile LOD arrays.

As stated earlier in this discussion of aliasing control, the technique used to control aliasing in conventional untextured images is the area average method. The shade value of the pixel is determined by the average of the shades of the individual surfaces within the pixel weighted according to their areas. For textured shading, the technique is to base the pixel shade on the pyramid weighted average of the individual samples of the appropriate LOD tile array.

The question then arises as to what composite technique should be used for the boundary problem; i.e., where a pixel projection contains both textured and untextured surfaces. Since this question was not addressed in depth by this study, the approach taken was to let the textured shading technique apply for composite surfaces. Thus, a pixel containing both textured and untextured surfaces was oversampled, taking the samples of textured surfaces from the area-averaged element of the appropriate LOD array, and using point samples as the samples of untextured surfaces. This method has the effect of eliminating some of the detail from untextured surfaces when they appear within a pixel that also has textured surfaces. This effect is evident in the next section, which presents imagery for different utilizations of the textured imagery. This effect of postfiltering is due to the particular suboptimal implementation for shading the textured-untextured surface boundary rather than a limitation of the postfiltering technique itself. The boundary effect can be eliminated by using a different boundary shading technique. Note, however, that the boundary between surfaces of different textures is adequately handled, as will be discussed in the next section.

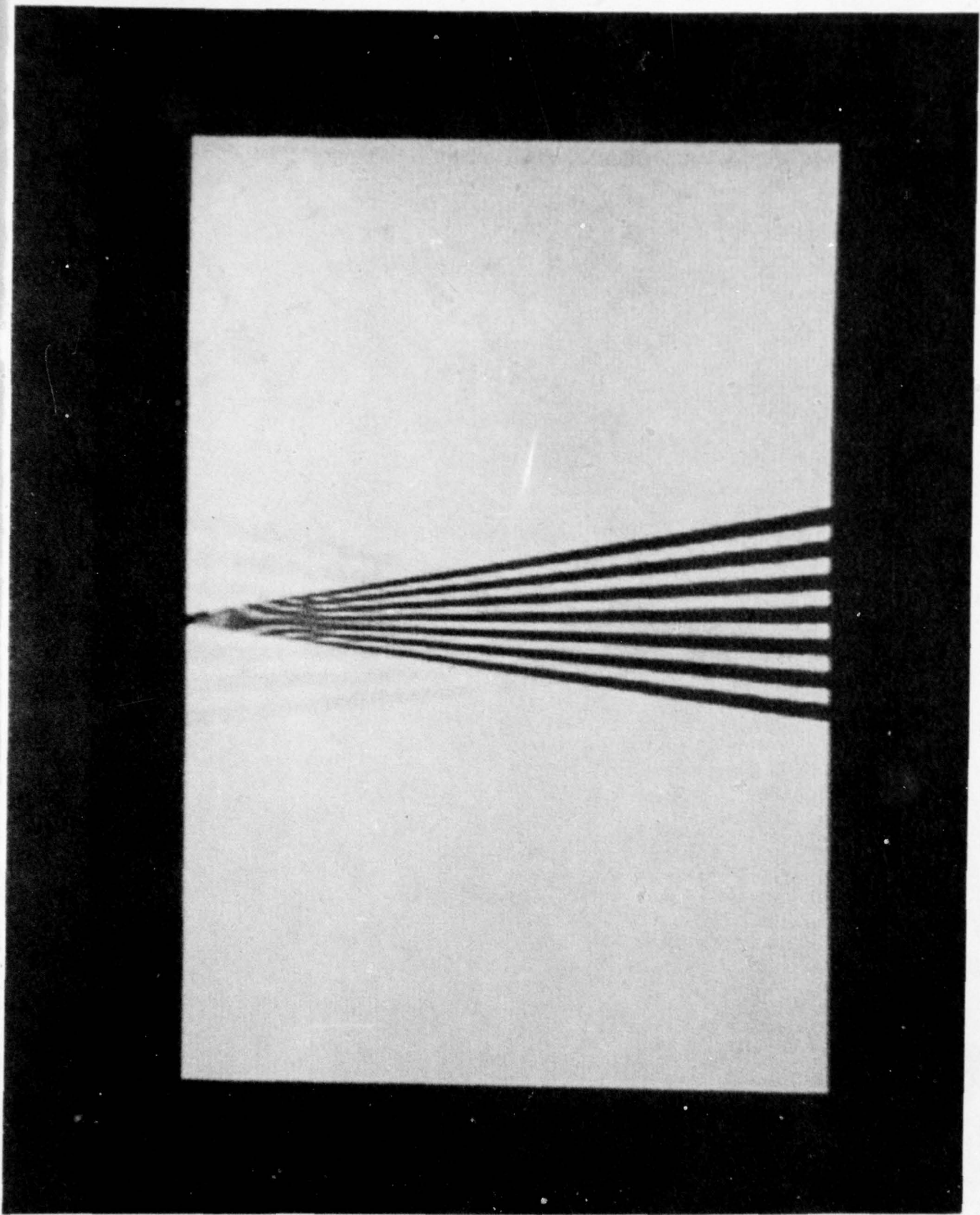


Figure 16. Aliasing of line pairs.

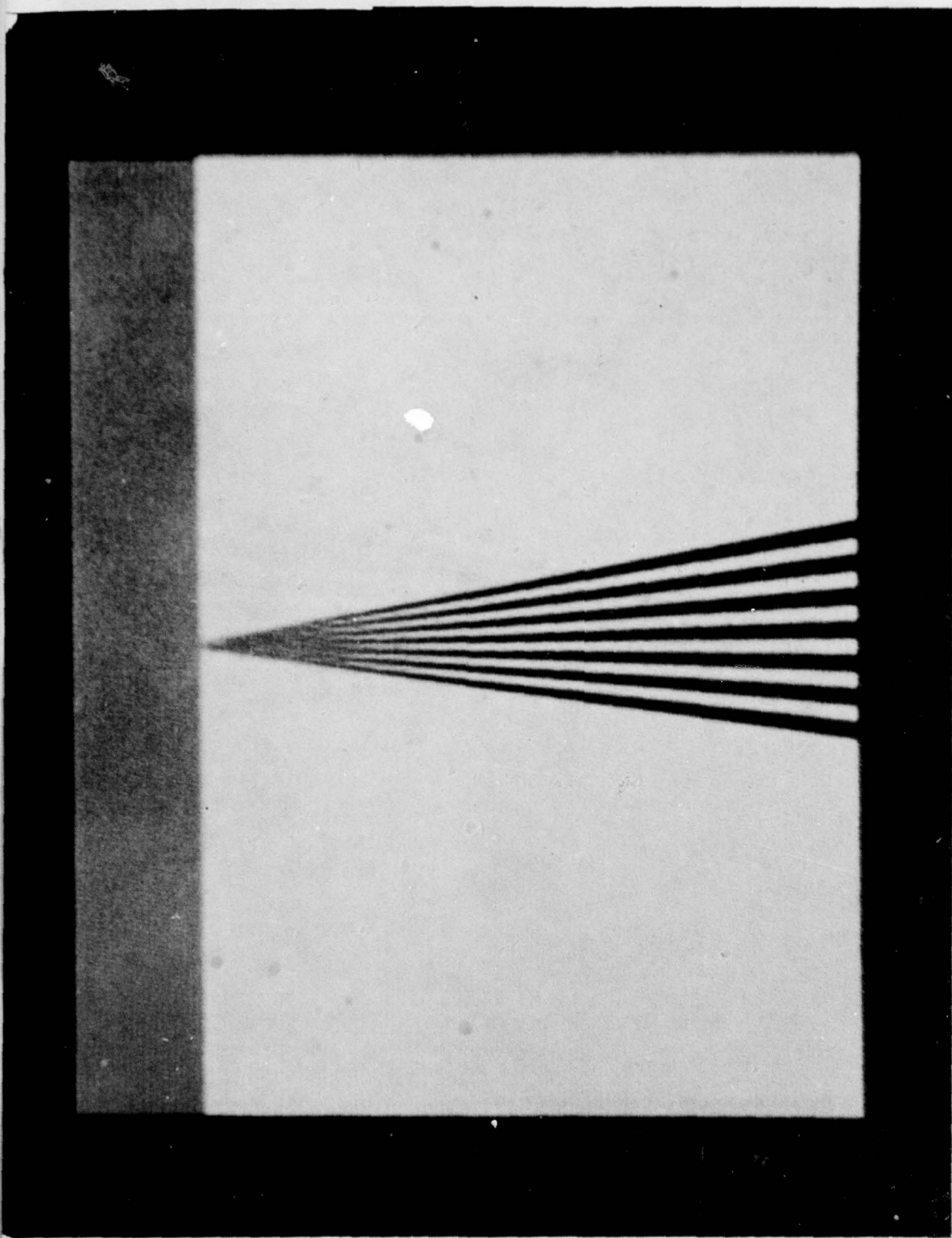


Figure 17. Line pairs filtered.

TEXTURE SCENES

Overview of the Data Base

The construction of the data base as well as the various data base entities was described earlier in the section that treated the scene data base. The method of generating textured scenes was discussed in the section on software structure and algorithms. This section is concerned with the utilization of the textured data base. The term utilization is meant to denote the general process of extracting information from the data base. This process is also characterized by the distinction of data and information. A single picture may be viewed as a single but complex data entity. However, given a sequence of these data entities, one can derive information that is based on the differences between pictures. It is also possible that if the differences are too great, no information can be inferred. Therefore, the most meaningful pictures to take are those in which a single parameter (e.g., field of view or position) is varied. In short, the process is really a sensitivity analysis, i.e., how various entities change as a function of a single parameter. To this end, coherent sequences of pictures are generated in order to extract information. Therefore, this section focuses on four sequences:

- Approach to landing
- Divebomb at 45°
- Terrain avoidance
- Special views

Figure 18 depicts the viewpoints of these sequences. Figure 19, which is a photograph depicting the equivalent data base scene, gives the spatial relationships between the various objects of the data base.

Approach to Landing Sequence

The purpose of the landing sequence is clearly to provide motion and distance cues for a pilot at various positions during a straight-in approach. In addition, the manner in which the scene changes and the reason for these changes are inferred from this sequence of pictures. The approach-to-landing sequence is shown in Figures 20 through 24.

Only the first of the textured pictures (Figure 21) in the landing sequence has any wooded texture. This textured area occurs in the foreground of the picture that is most distant from the touchdown zone. Because of the postfiltering that occurs after the textured frame has been generated, there is a general blurring or defocusing of the image. This can be seen by alternately scrutinizing the textured and untextured versions. For example, the horizontal taxiways to the left side of the runway are in the same pixels as the grass texture, and disappear in the textured rendition.

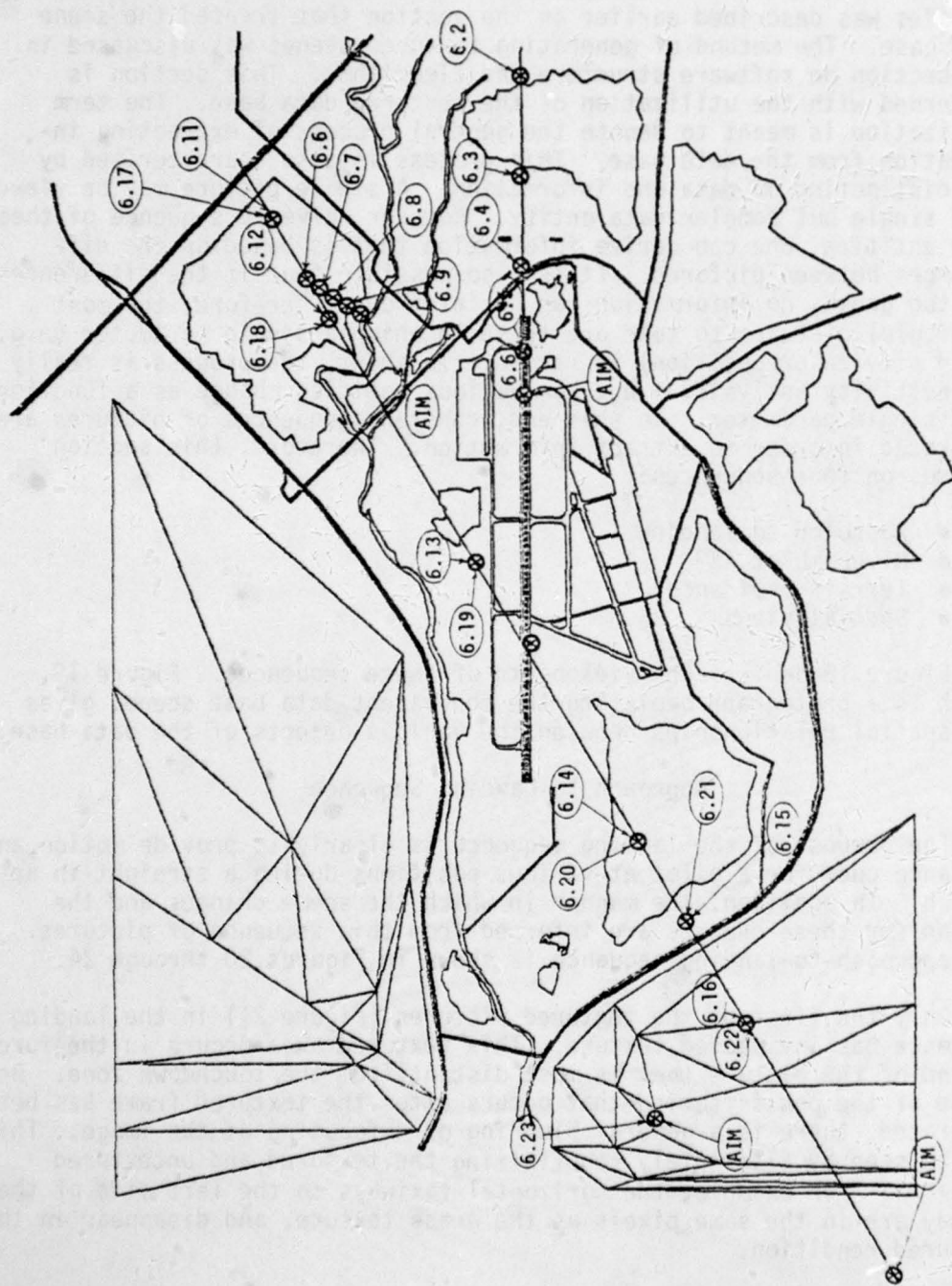


Figure 18. Photograph viewpoints.

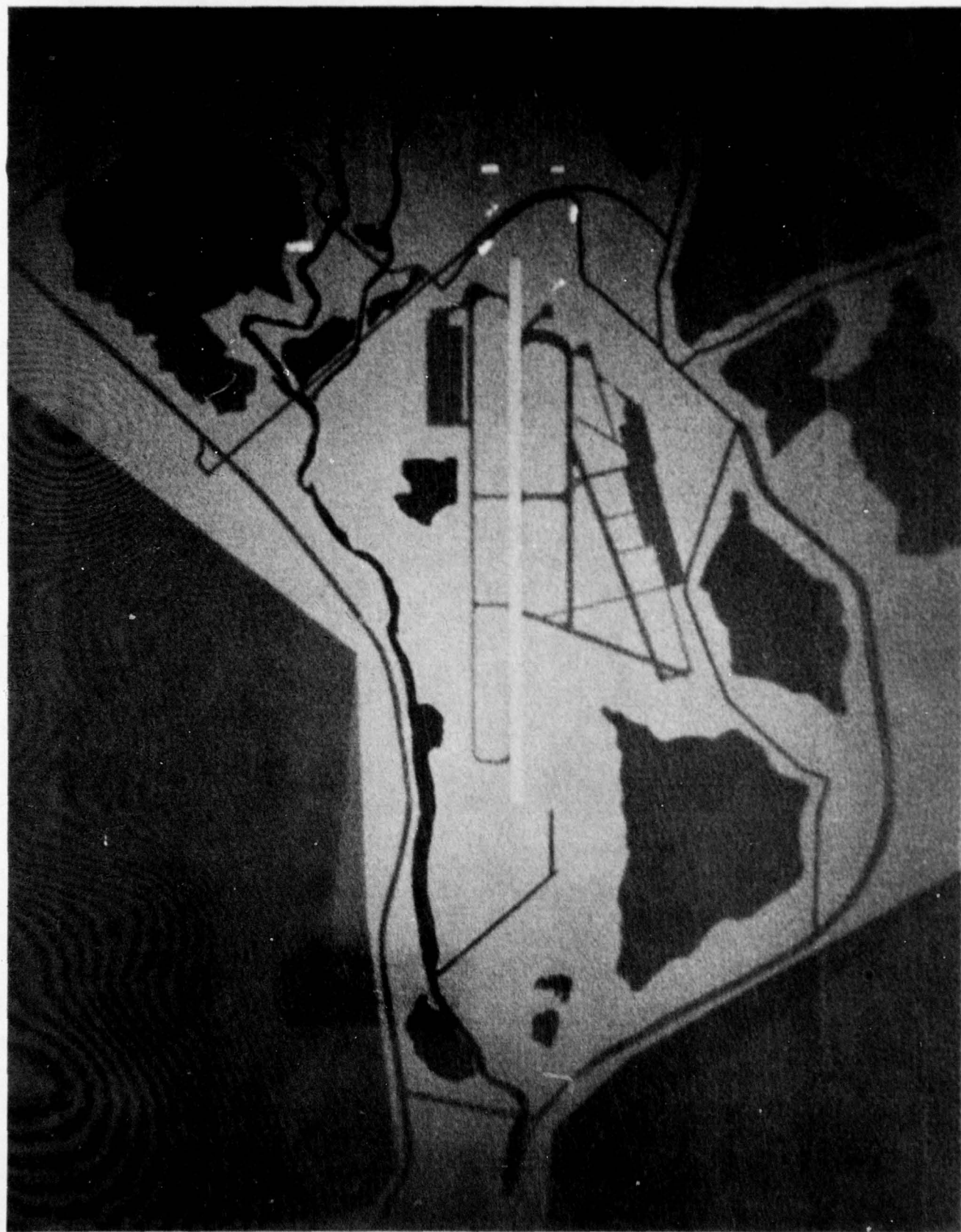


Figure 19. Scene overview.

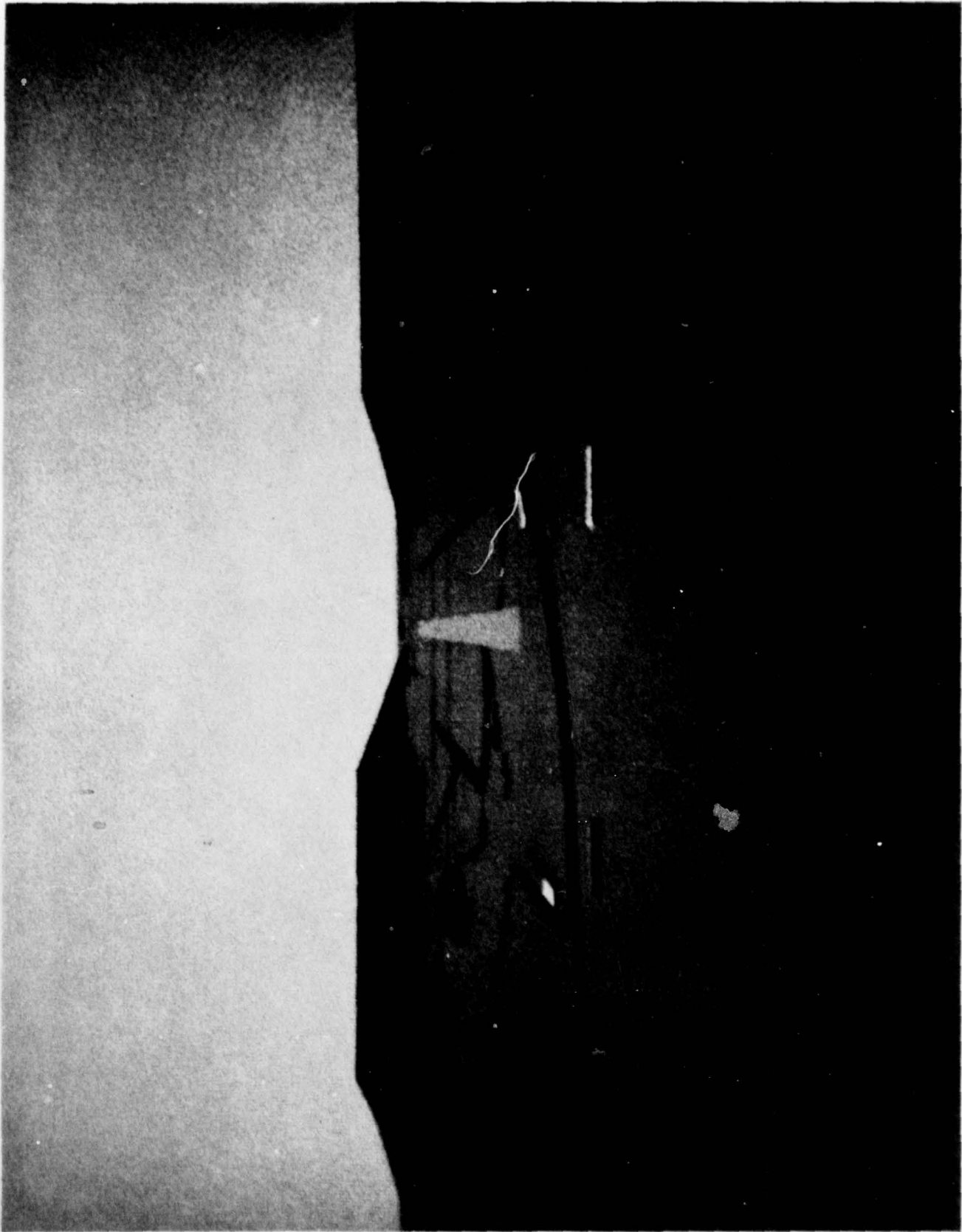


Figure 20. Approach to landing—untextured.



Figure 21. Approach to landing-textured.

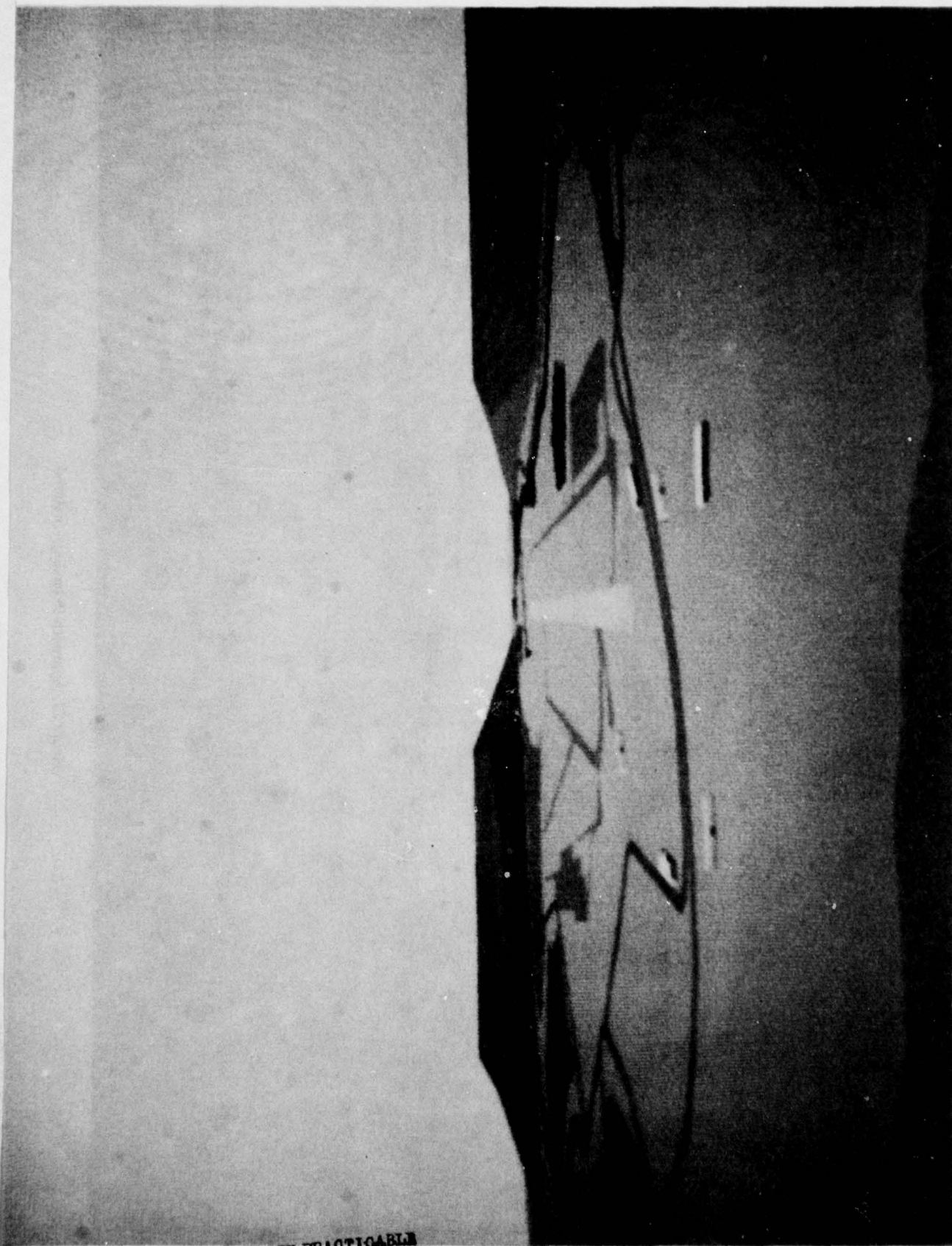


Figure 22. Approach to landing-textured.

THIS PAGE IS BEST QUALITY PRACTICABLE
FROM COPY FURNISHED TO DDC

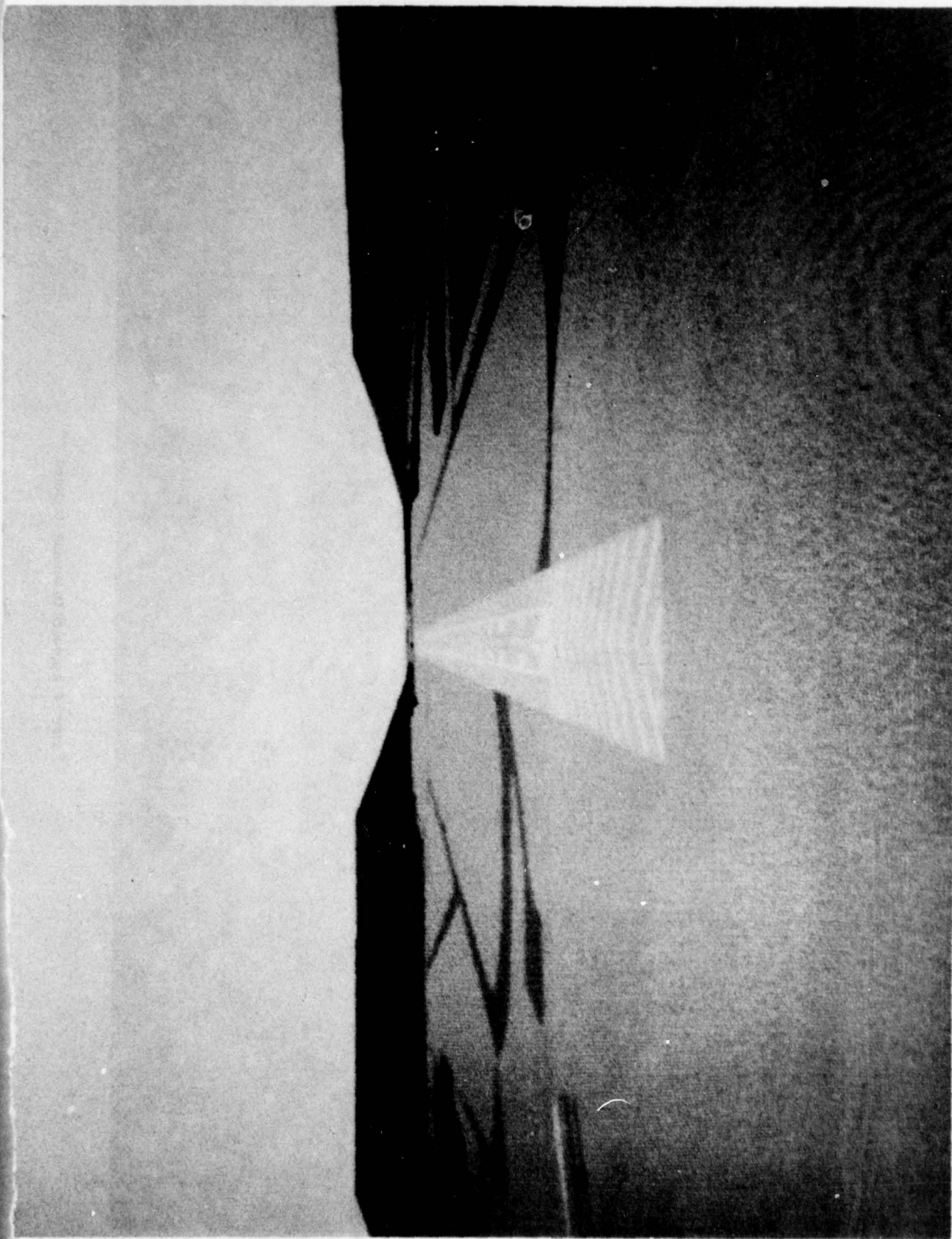


Figure 23. Approach to landing-textured.

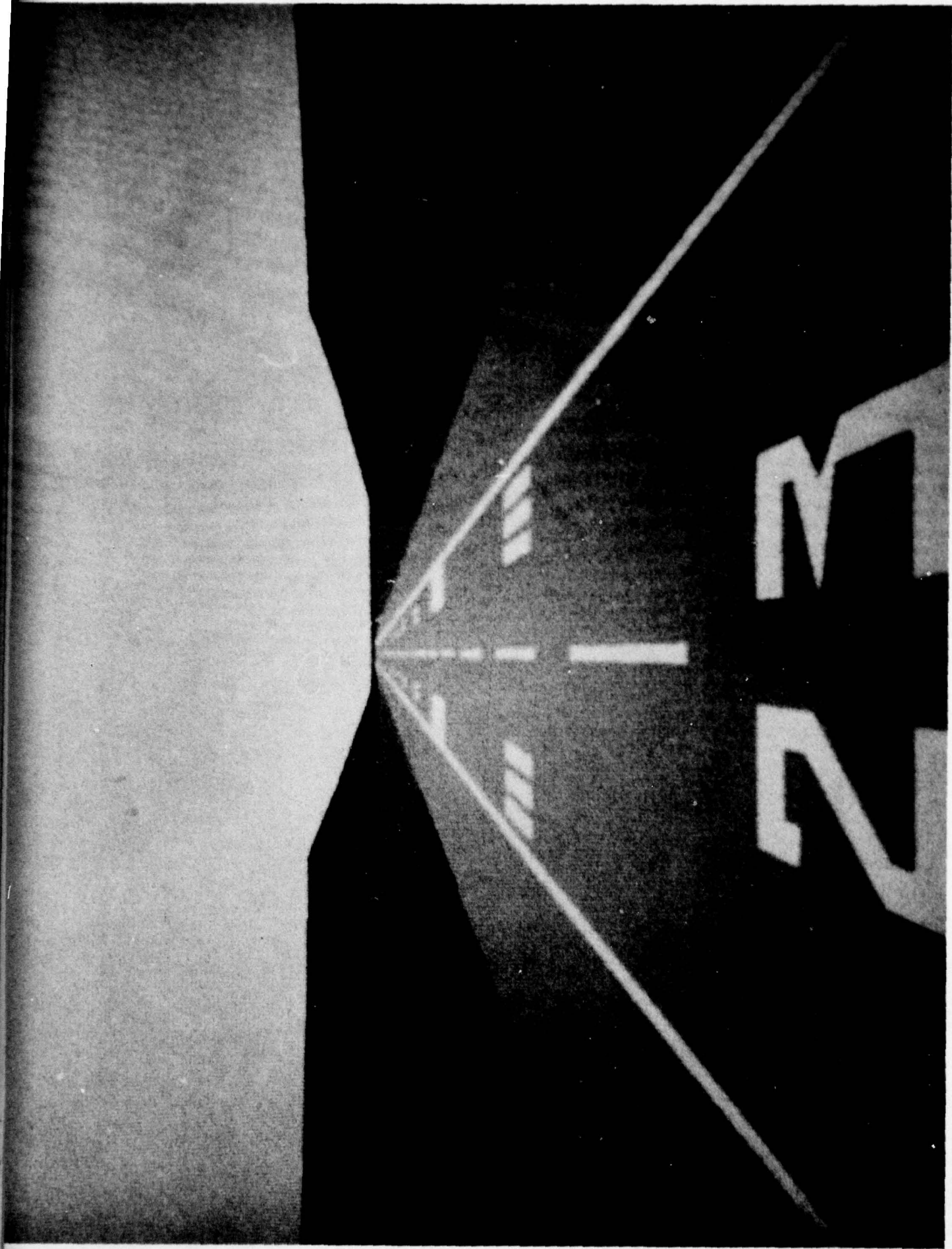


Figure 24. Approach to landing-textured.

As objects become closer to the viewing position, two phenomena are noticed: multiple LOD representations for the cultural objects within the scene and the texture tile LOD arrays. The data structure used to depict an object is a function of distance. There are two different data structures or LOD representations for each building in the TSC system. This can readily be seen by examining the runway markings of the five photographs in Figures 20 through 24. The runway markings are not discernible in Figures 20 and 21, whereas they are clearly visible in Figures 22, 23, and 24. This difference results because two different data structures are being used to represent the runway. The effects of postfiltering can be seen by noticing how all the vertical bars are diffused with one another except for the two sets of three vertical markers in Figure 24.

Additionally, there is LOD in texture. As previously mentioned, there are eight LODs for a texture tile. This phenomenon is manifested as a textured surface comes closer. At far range, the textured surface is a uniformly shaded surface; at closer ranges, more of the detailed texture becomes apparent. For surfaces textured with grass, the texture patterns are readily perceived within viewing distances of 50 feet.

45° Divebomb

The next photographs (Figures 25 through 28) depict a divebomb sequence at a 45° angle (the distance to the point of view differs by a factor of 2 between successive photographs). The corner of the building appearing in the photographs is the point of impact. By scrutinizing Figure 25, the periodicity of the texture tiles can be discerned; however, the boundaries between adjacent texture tiles are not discernible.

It is easy to compare the corresponding trees in the two images shown in Figures 25 and 26. Since relative linear dimensions in the two images differ by a factor of 2, their spatial frequencies also differ by a factor of 2. Figure 25 therefore appears to have sharper contrast than Figure 26. This is, of course, a result of higher spatial frequencies. Conversely, Figure 26 appears to be defocused when compared to Figure 25. This pattern of increasing defocusing is even more apparent when Figure 27 is compared to Figure 26. Finally, in Figure 28, a checkerboard effect is noticed. This effect results from a single entry of the texture tile array mapping into multiple pixels; thus, the constant intensity for contiguous groups of pixels. The use of only a single texture tile over large variations in range is a limitation on the texture tile technique, since the scene resolution requirements exceed the resolution available in the highest LOD. By adding more texture tile representations suitable for different range increments, the range of resolutions for depicting various textured material can be easily extended.

Another example of this range of resolution is provided by examining the textured grass surrounding the building. The textured surface



Figure 25. 45° Divebomb—textured.

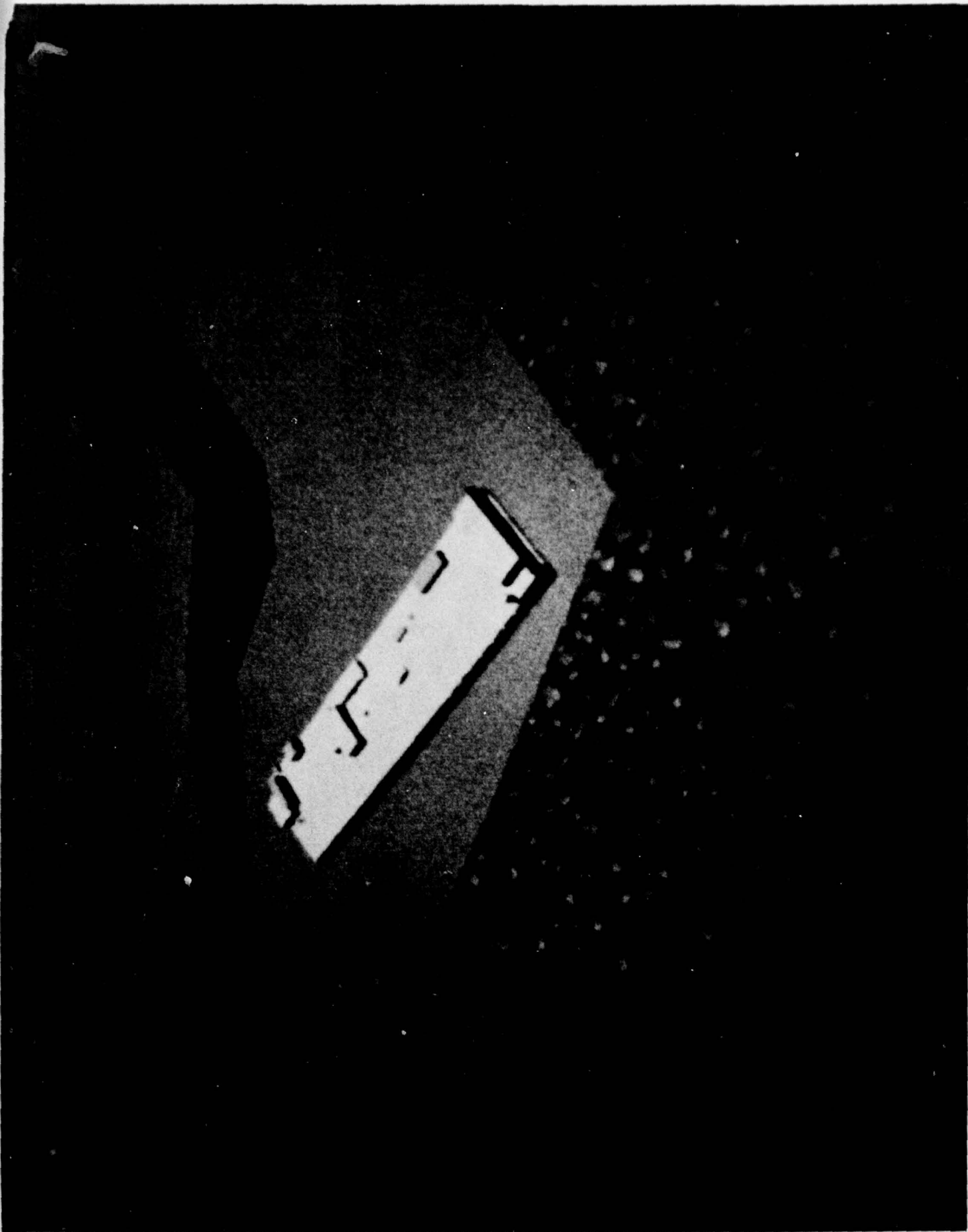


Figure 26. 45° Divebomb-textured.



Figure 27. 45° Divebomb-textured.

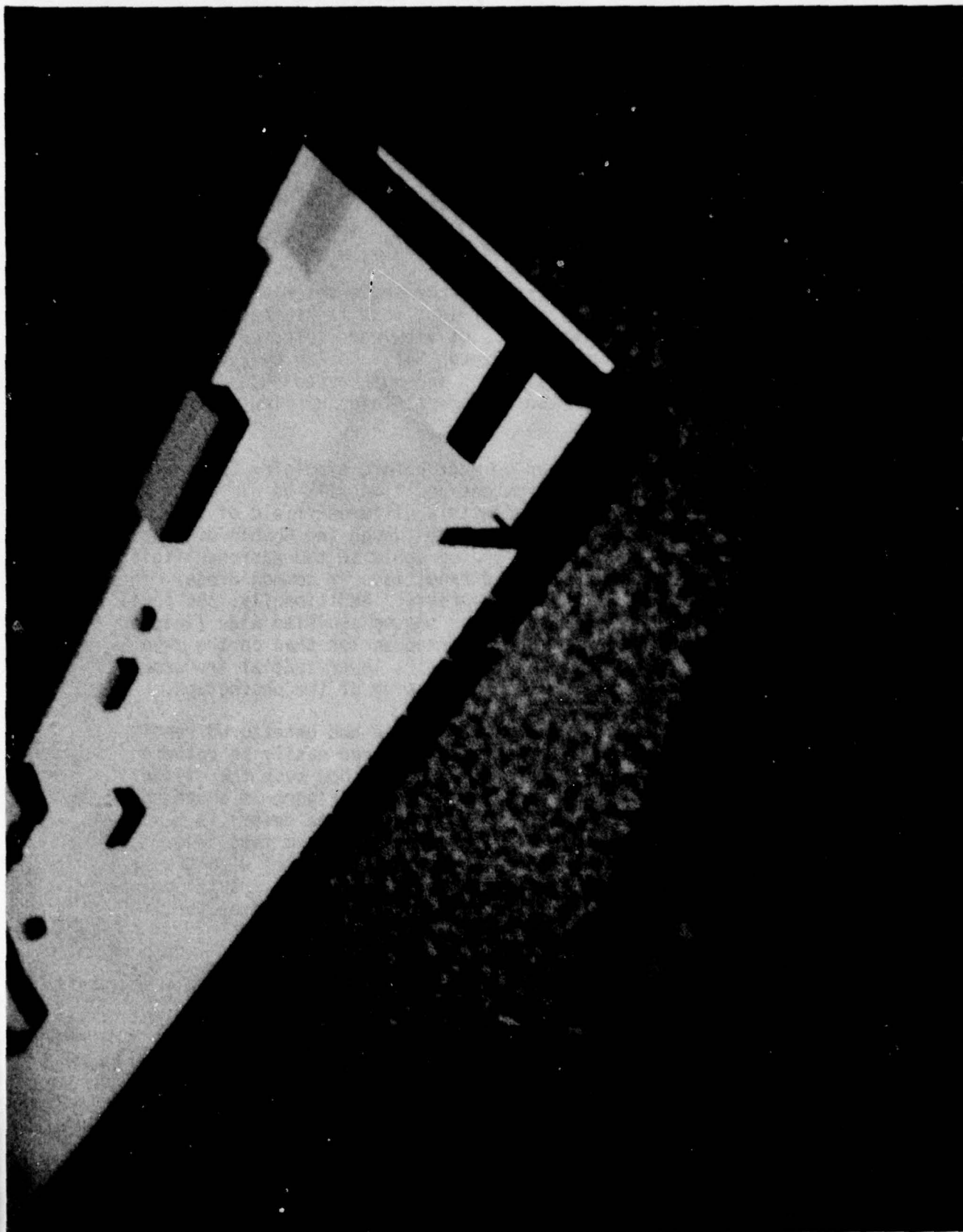


Figure 28. 45° Divebomb—textured.

becomes uniform at far range (a distance of somewhere between 500 and 1000 feet). The other end of the range is determined by the appearance of the checkerboard effect. However, referring back to the section on texture tiles for grass, the highest LOD was digitized at a resolution of 14 inches, which greatly exceeds the size of a blade of grass.

Terrain Avoidance

The purpose of the two terrain avoidance sequences was to simulate the view of a sensor in a low-altitude flight that maintained a 200-foot altitude above the terrain. Two fields of view were selected for the photographs: 60° by 45° (Figure 29 through 34), and 22° by 15° (Figures 35 through 40). Therefore, both textured and untextured photographs as well as the photographs for the two different fields of view will be discussed.

Several general statements can be made before examining the pertinent photographs. A photograph taken from one position will be different from a photograph taken from that position with a different field of view. This difference is due to the fact that even though the number of pixels is the same in both photographs, the photograph with the narrower field of view will, in effect, have magnified renditions of common areas. This will cause the spatial frequencies to decrease. Additionally, the post-filtering process will have more effect on the unmagnified wide field of view (WFOV) image that has higher spatial frequencies than on the magnified narrow field of view (NFOV) image with its lower spatial frequency content. This occurs in the perceived defocusing of the photograph.

The next paragraph will examine the textured and untextured renditions shown in Figures 29 and 35. Magnification can easily be detected between the 22° and 60° fields of view by noticing how much the distant hill covers in the photographs. The fine detail in Figure 35 above the second body of water is removed in Figure 29. The fine detail is magnified over several pixels in the NFOV image, but is merged into fewer pixels in the WFOV image. Even though the hill is textured with trees, its component planes appear uniformly shaded. This effect results because the lowest LOD is utilized in tiling the surfaces, and at the distant ranges the intensity is essentially constant.

The effect that the field of view has on the textured image can be seen by examining the textured area around the triangular point of the forest below the main body of water in the center of Figures 29 and 35. In the WFOV image, a textured pixel has a large footprint when mapped into the scene. The LOD texture array used to shade this pixel has low resolution. Conversely, the same pixel has a smaller footprint in the NFOV image, and is consequently represented through a higher resolution LOD array.

This paragraph is concerned with the textured photographs shown in Figures 30 and 36. Both photographs are from the same viewpoint and have the same aim point. However, the fields of view are 60° by 45° and



Figure 29. Terrain avoidance—textured, WFOV.



Figure 30. Terrain avoidance—textured, WFOV.

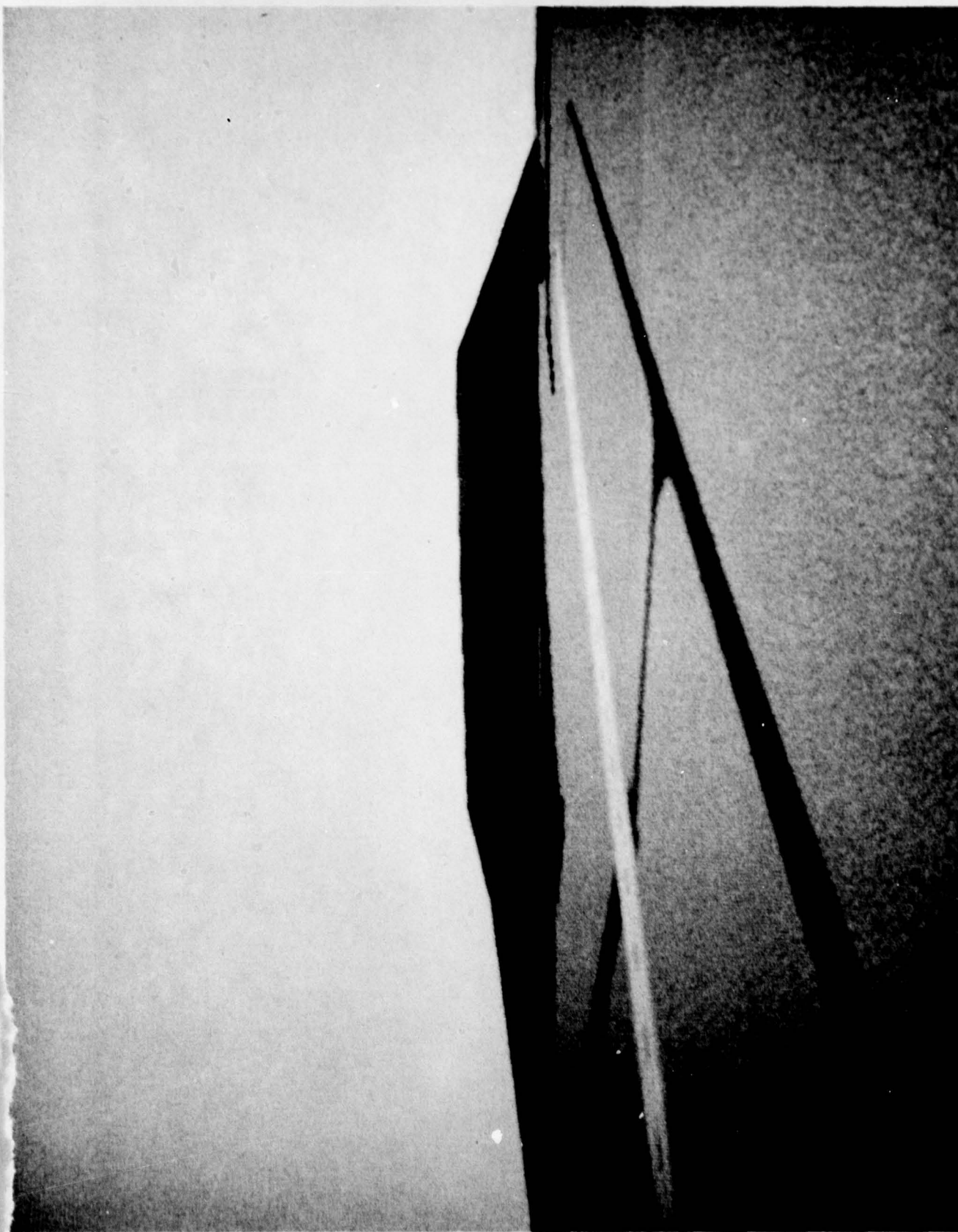


Figure 31. Terrain avoidance—textured, WFOV.

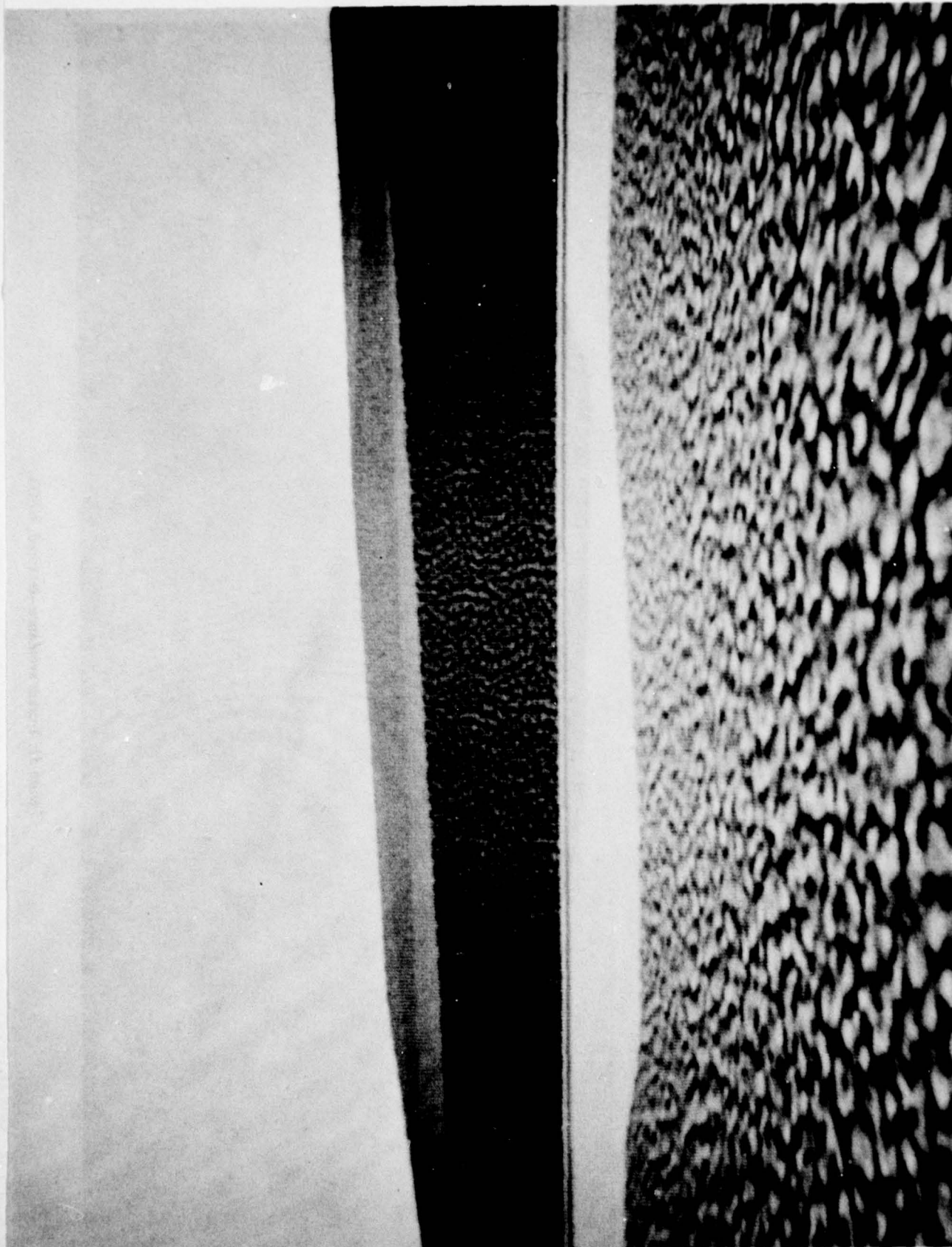


Figure 32a. Terrain avoidance-textured, WFOV.

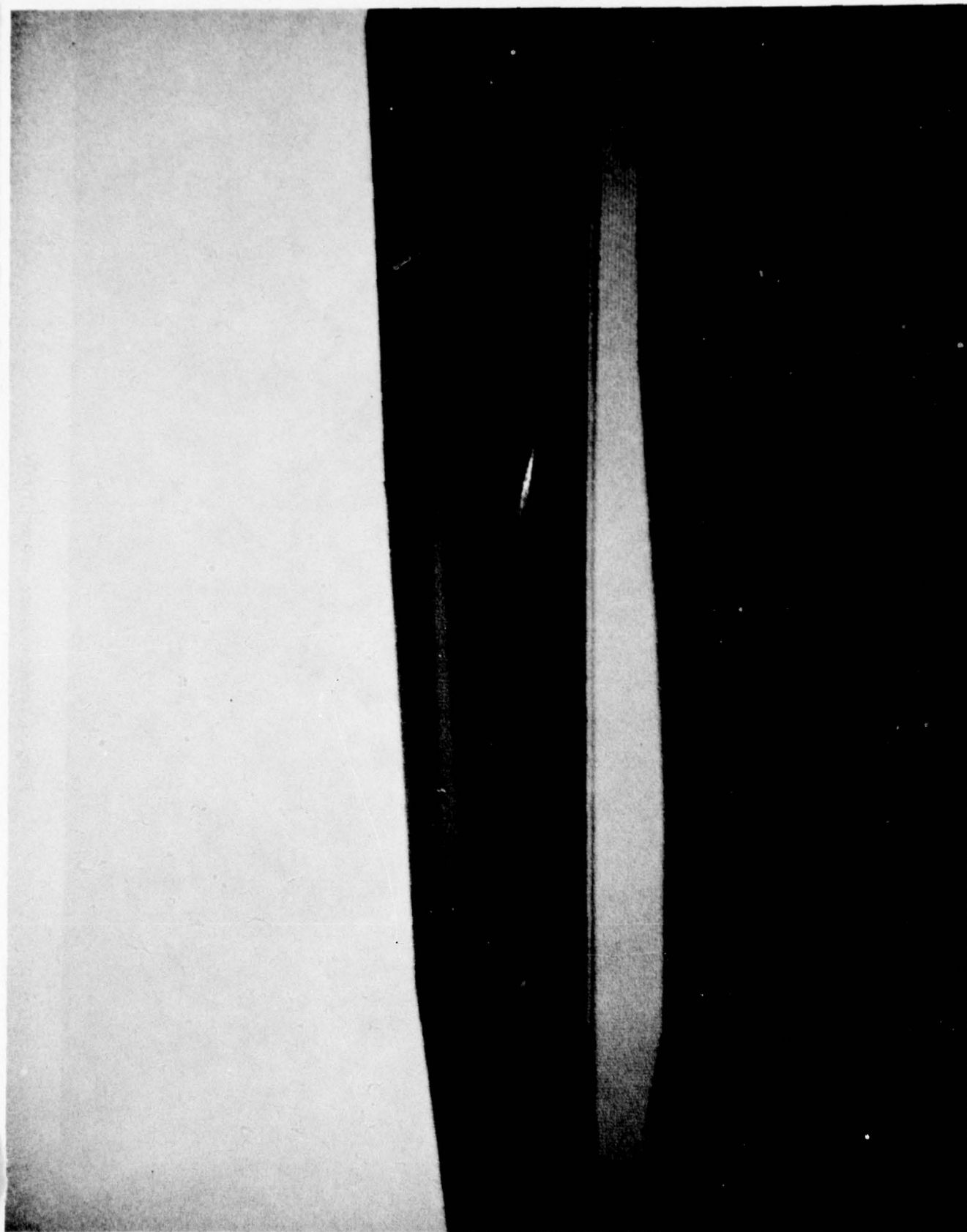


Figure 32b. Terrain avoidance—untextured, WFOV.

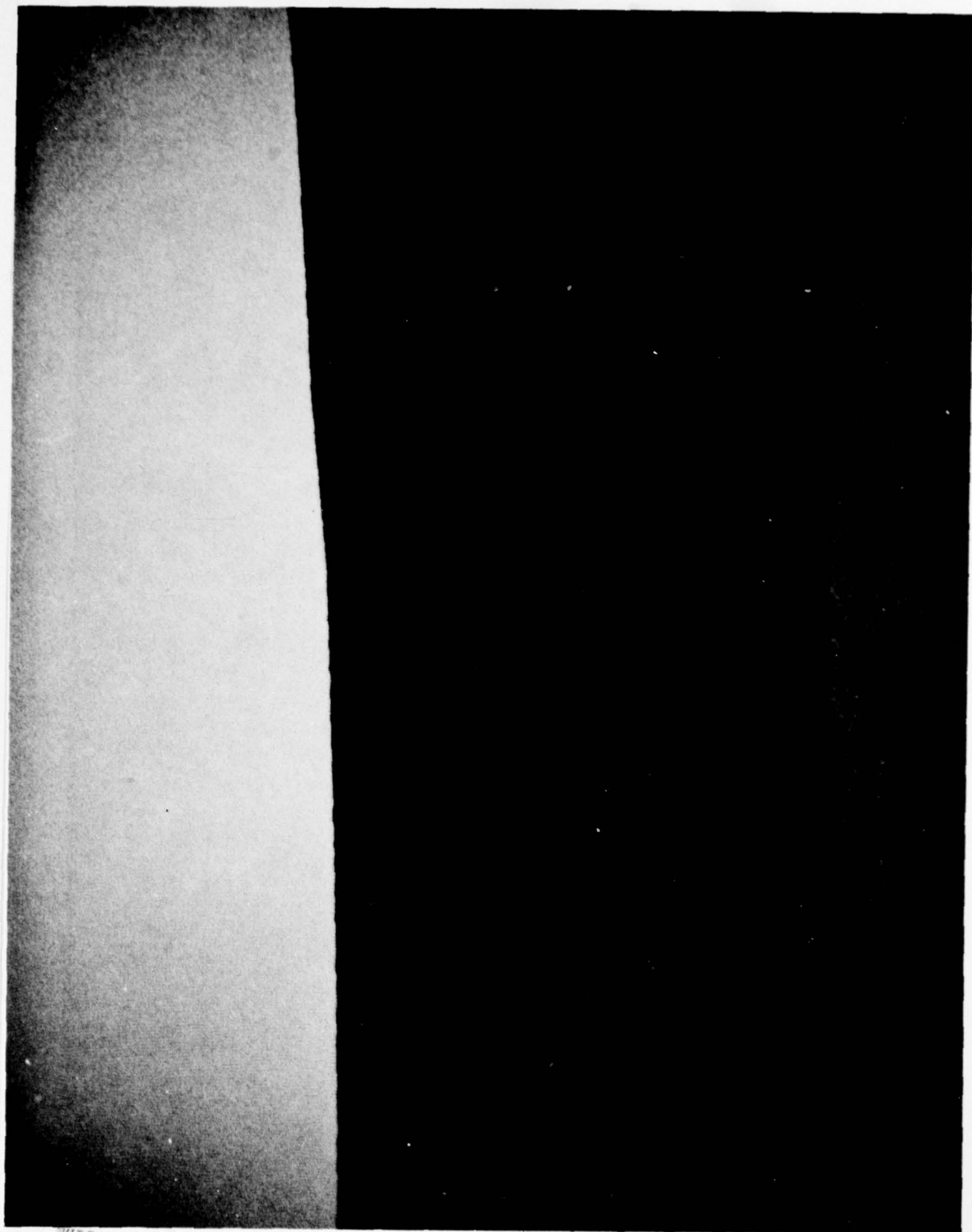


Figure 33. Terrain avoidance—textured, WFOV.

THIS PAGE IS BEST QUALITY PRACTICABLE
FROM COPY SUBMITTED TO DDC



Figure 34. Terrain avoidance—textured, WFOV.

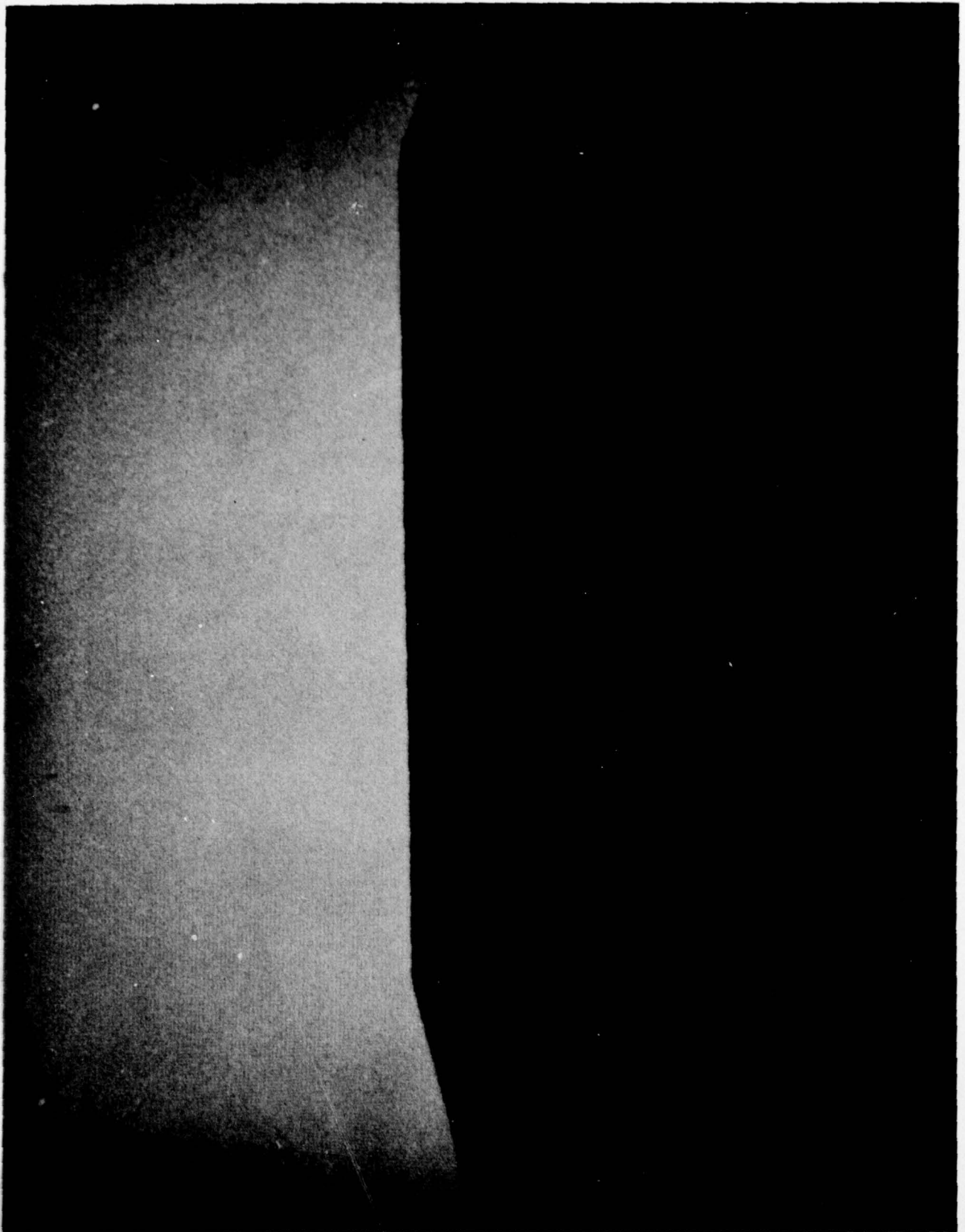


Figure 35. Terrain avoidance—textured, NFOV.

THIS PAGE IS BEST QUALITY PRACTICABLE
FROM COPY FURNISHED TO DDC

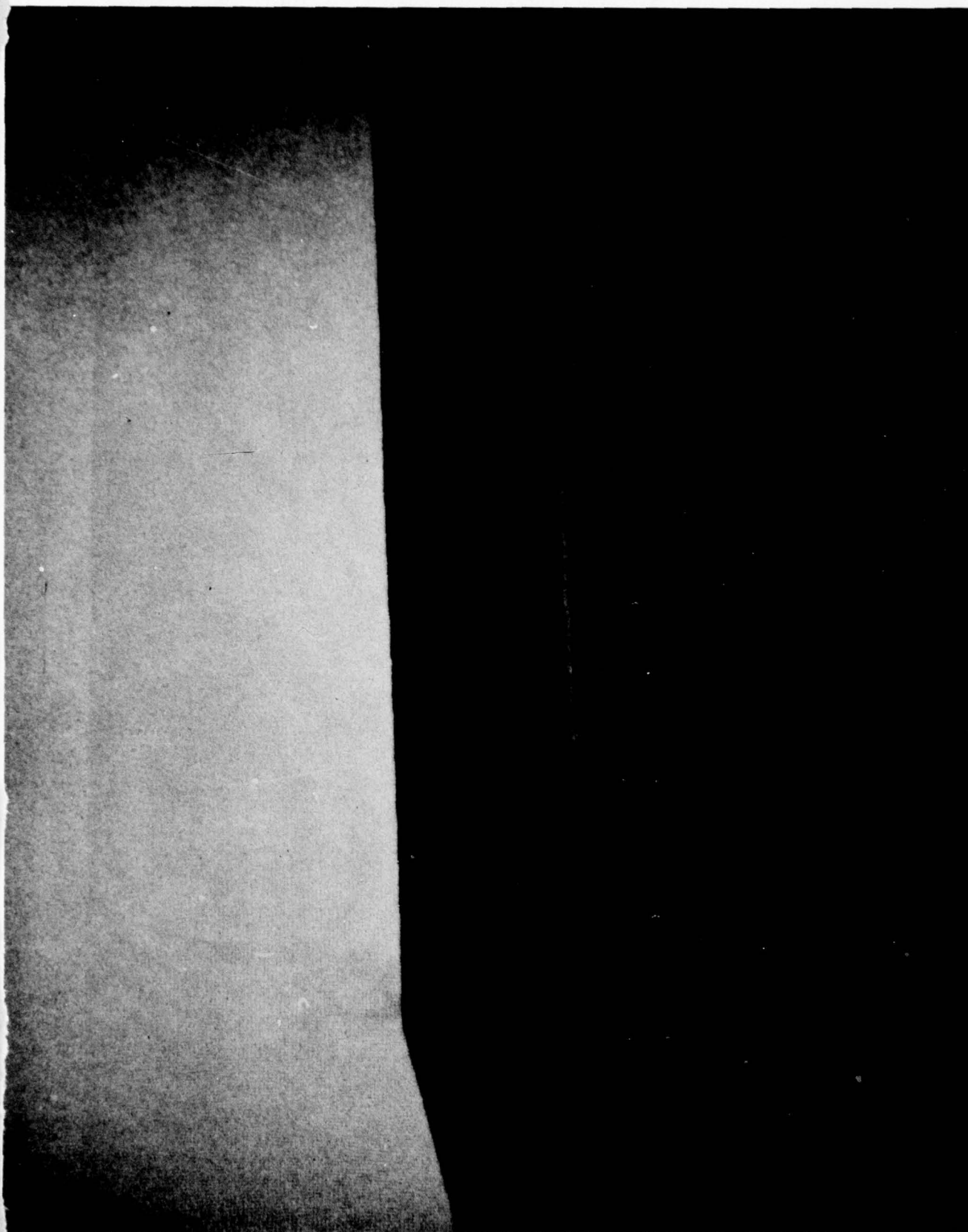


Figure 36. Terrain avoidance-textured, NFOV.

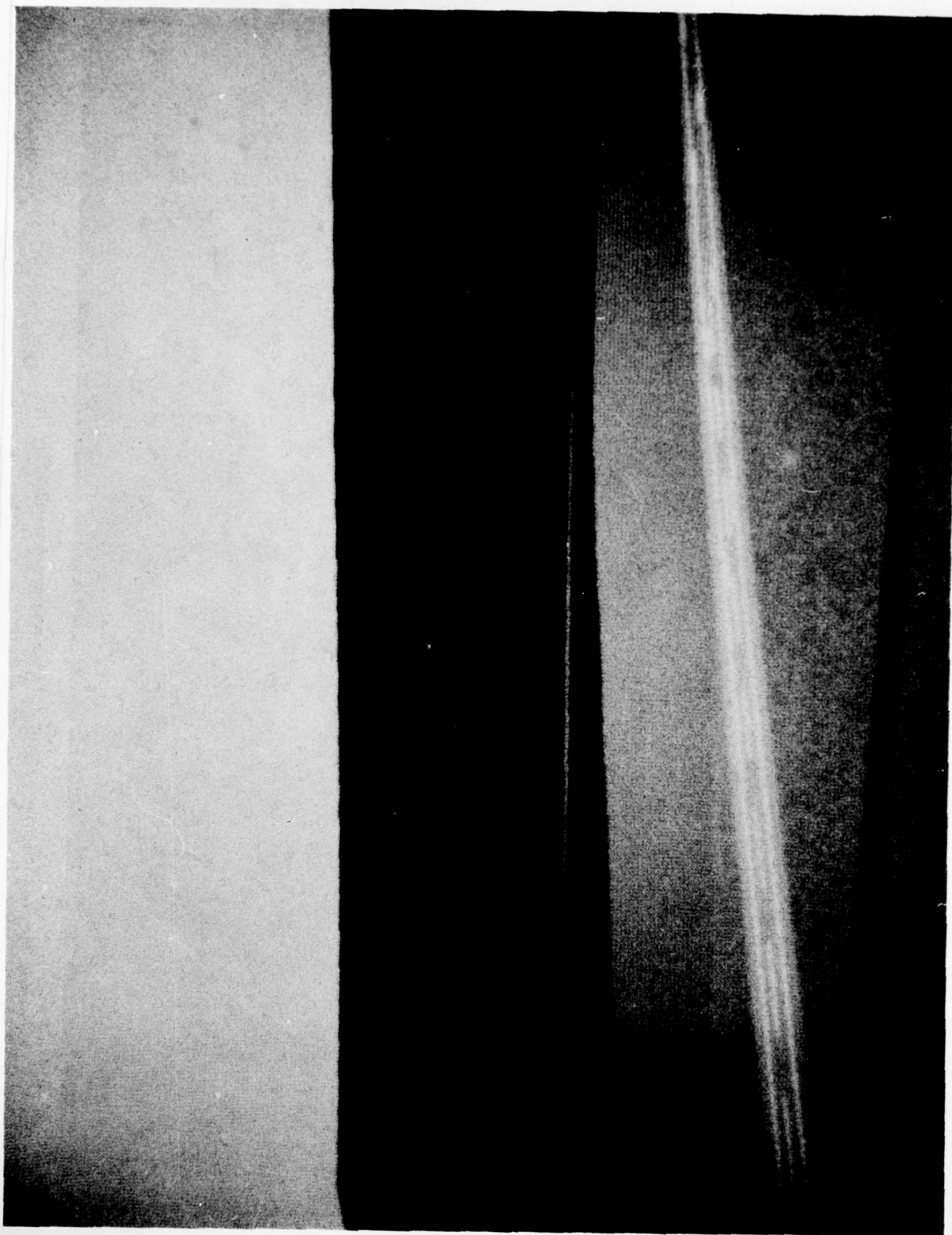


Figure 37. Terrain avoidance-textured, NFOV.

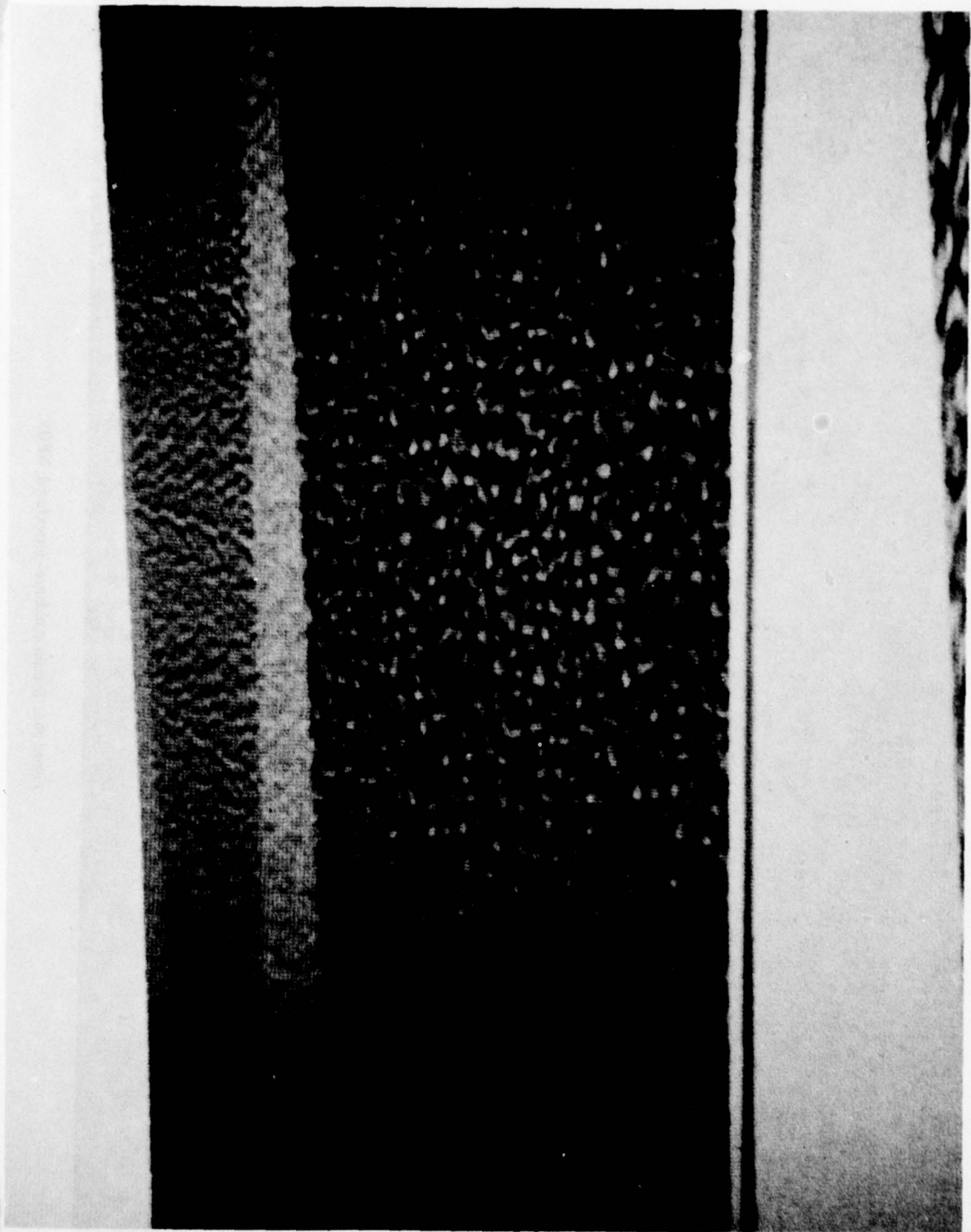


Figure 38a. Terrain avoidance-textured, NFOV.

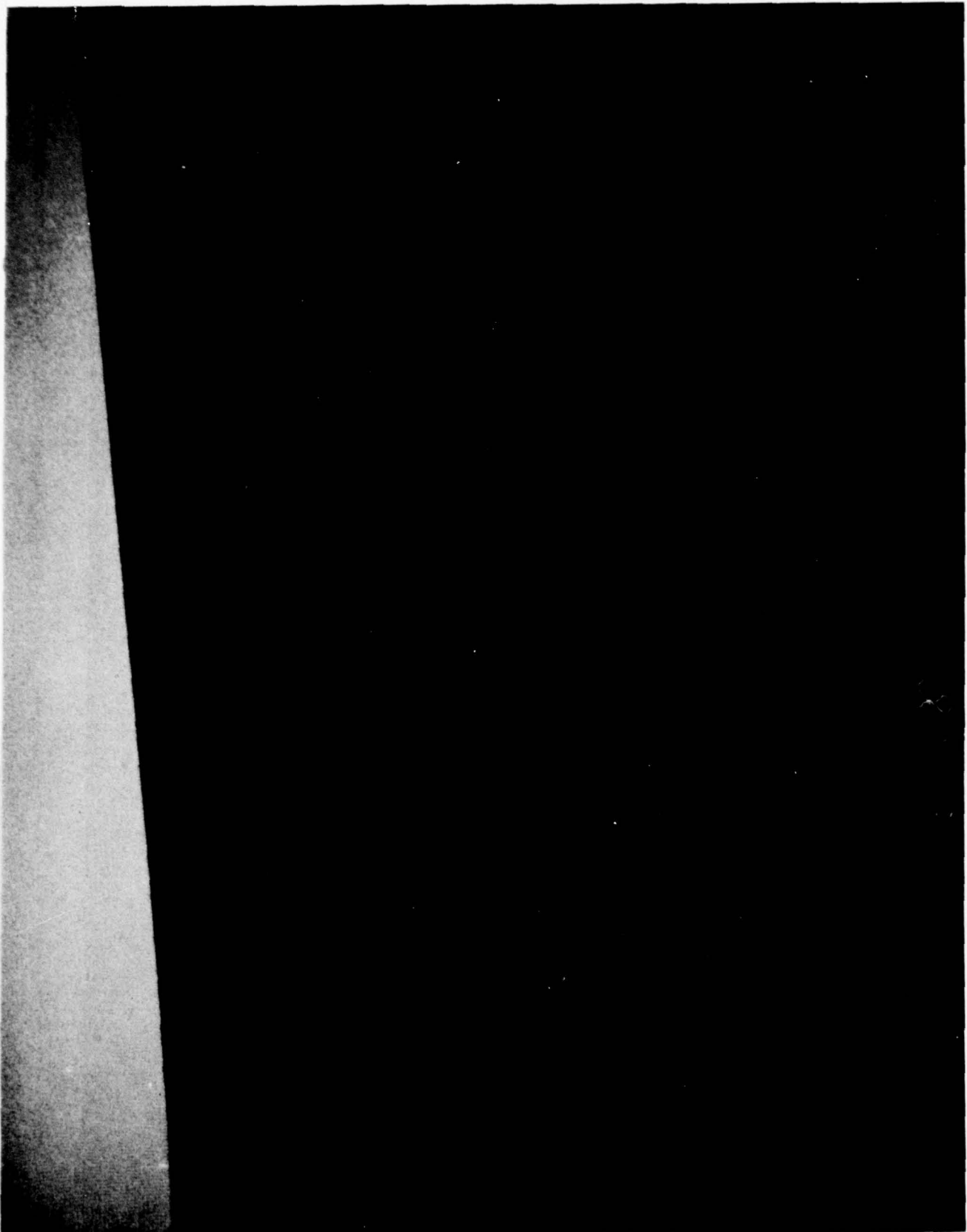


Figure 38b. Terrain avoidance—untextured, NFOV.

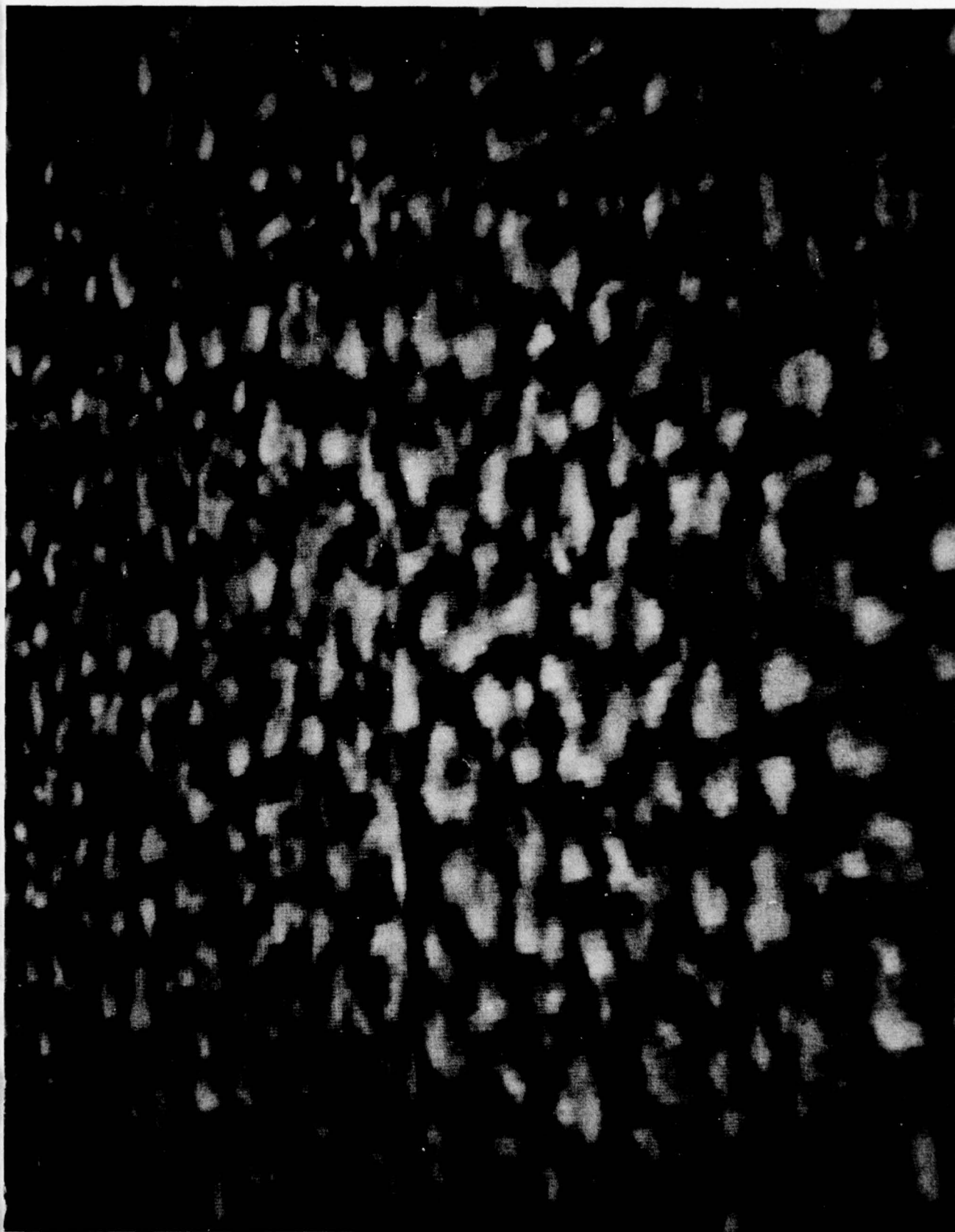


Figure 39. Terrain avoidance—textured, NFOV.

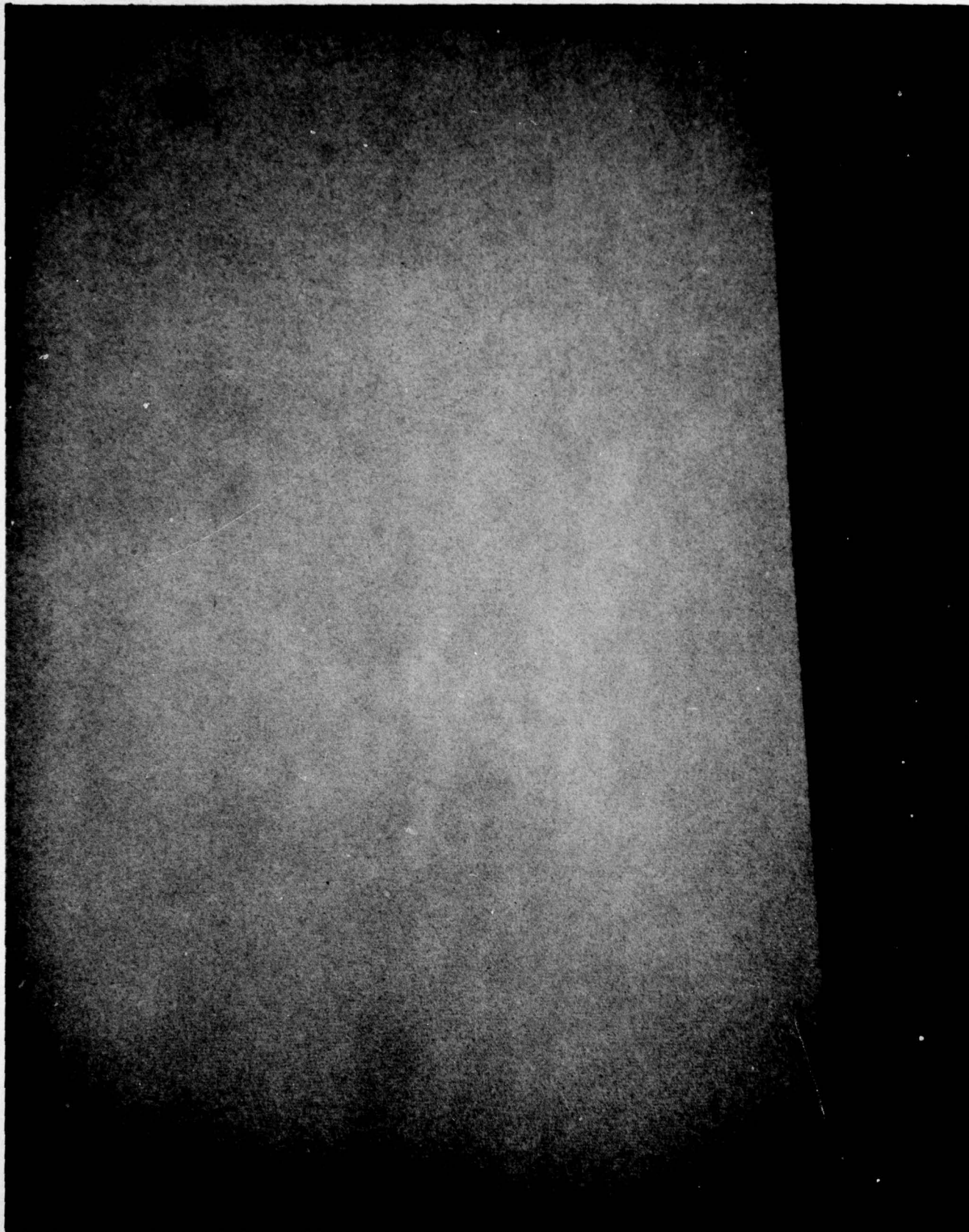


Figure 40. Terrain avoidance--textured, NFOV.

THIS PAGE IS BEST QUALITY PRACTICABLE
FROM COPY FURNISHED TO DDC

22° by 15°, respectively. Because each photograph has the same number of pixels, Figure 36 is a magnified subpicture of Figure 30. Therefore, Figure 30 contains elements (e.g., the forest area in the foreground) that are out of the field of view of Figure 36. The small peninsular indentation in the center of Figure 36 is essentially removed as a noticeable feature in Figure 30. Also, there is a similar difference in the thin whitish line (just above the small lake) that represents the airport runway. The common textured area of these photographs is of grass at far range, and it appears at the lowest LOD for both fields of view. Thus, these two photographs cannot be used to compare the field-of-view effects on texture.

This paragraph is concerned with the photographs shown in Figures 31 and 37. Notice the thin horizontal line of the ground plane at the base of the hill in Figure 37. This line is almost unrecognizable in Figure 31, which maps the thin line into fewer pixels. Also, notice in Figure 37 the residual effects of aliasing exhibited by the runway strips. In contrast, the postfiltering (actually an averaging technique) used in Figure 31 causes the runway to appear whitish because it distributes the higher intensities over a larger number of pixels.

The different planes that compose the hill in Figure 37 are at different distances from the viewpoint. These distances are just right so that the closest plane is textured with a more refined LOD than are the farther planes. While the hill is entirely within the lowest LOD footprint in Figure 31, other field-of-view effects are evident there. Due to the wider field of view, close-in grass at the lower depression angles is visible in the foreground, and is represented by a higher resolution LOD array than in the foreground grass of Figure 37, which is at a farther range.

One interesting thing to note in Figures 32 and 38 is that only the 22° field-of-view photograph yields a textured hilltop. Also, the WFOV permits the forested area in the foreground to be represented with a higher resolution tile array than does the NFOV. Again, this is due to the fact that the foreground of the NFOV image is at a higher depression angle and hence a successively longer range.

That the textured images contain additional information that can be used to supply motion and distance cues is easily inferred by comparing the textured images of Figures 32a and 38a with the comparatively bland, untextured images of Figures 32b and 38b. This difference is the most graphic effect of this series of photographs.

In Figures 33 and 39, the hill is very close and the altitude of the sensor will have to change in order to reflect the 200-foot terrain-following constraint. In the 60° field-of-view photograph, the skyline of the hill and the boundary of the hill and the grassy plain help to establish an orientation. On the other hand, in the 22° field-of-view photograph, there are no references to establish the orientation. The

part of the hill closest to the viewpoint commences to exhibit the checker-board effect, giving some indication as to the relative distance.

In Figures 34 and 40, the pitch-up maneuver has occurred and the sky now becomes a major portion of both photographs. In the 60° field of view, the boundary between two hillside planes can be distinctly discerned.

Special Views

This section is concerned with the problem of surface boundaries. A vertical shot of a textured and untextured hilltop is used to demonstrate the problem's solution.

The intent of Figures 41a and 41b is to demonstrate the texturing algorithm with respect to the interface between plane surfaces. Note that the interface problem between adjacent texture tiles was overcome by post-filtering. In the untextured photograph, only two planes are readily distinguishable. The important point here is that the texture tiles have been matched over the boundary, and the visible boundary is due to the relative sun shading which is used to adjust the shade of the tile.

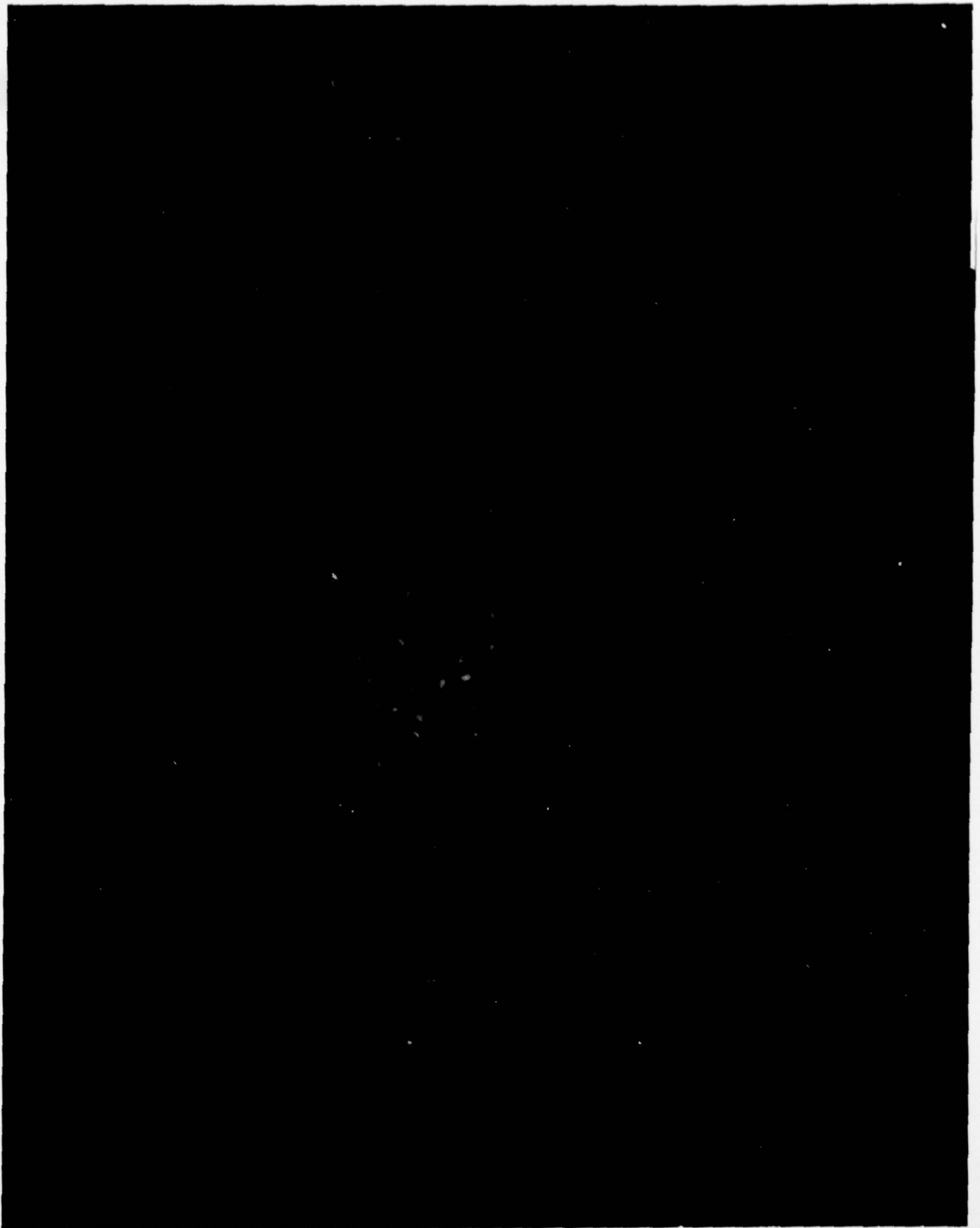


Figure 4/a. Hilltop surfaces-textured.

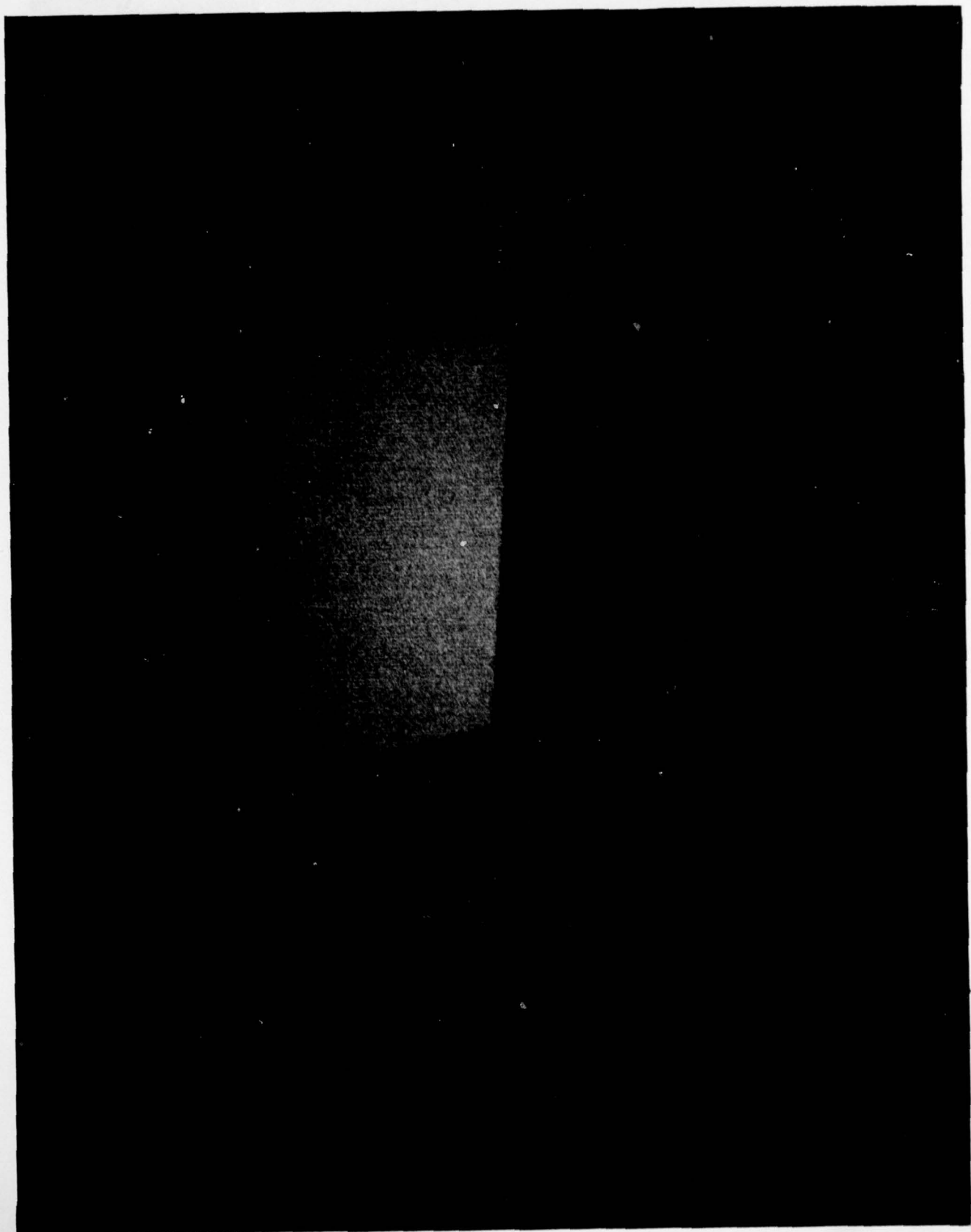


Figure 41b. Hilltop surfaces—untextured.

THIS PAGE IS BEST QUALITY PRACTICABLE
FROM COPY FURNISHED TO DDC

CONCLUSIONS AND RECOMMENDATIONS FOR FURTHER RESEARCH

The most important result of this program is that texture tiles have proved to be a viable means for generating textured scenes. The first step taken to add texture to a scene was to construct and process the tile itself. With the use of the primitive tile in a scene, the repetitive nature of the tile concept was readily apparent. First, because the tile was not matched across the boundary, the replication of the discontinuity of the tile perimeter stood out. Secondly, the macrostructure or non-homogeneity of the tile was also repeated. During this program, techniques were developed to preprocess a tile in such a way that an entire plane can be textured in perspective with only a subtle indication of repetition.

The aliasing problem associated with the high frequency nature of the texture tile was controlled via two methods. First, quasi-continuous LOD tile arrays were created to generate the correct tonal value. Secondly, the entire textured image was filtered. Both techniques reduced the aliasing evident in the unprocessed textured imagery.

Finally, the texture tiles were added to a complex scene of cultural content, and the combined scene was evaluated under different applications. All scenes showed the viability of the texture concept.

A number of problems were revealed during this program. The first was the difficulty (in fact, the shortcoming) of portraying a given texture with a single representation at the multiple observer-scene ranges of interest. For example, the tree tile used in this program was suitable at mid-range, and the range transitions within this region were adequately handled by the LOD arrays. However, even though LODs were used, they were obtained from the same basic tile and could not be used at far or near range. At far range the lowest LOD array was used, but this is a single value and results in constant shading. Conversely, at near range the highest LOD array was used, but the resolution is still insufficient once the pixel size within the scene approaches that of an array element. The present tile represents a "stand of trees" at high aspect; it must be augmented with one tile that conveys the concept of a "forest" at long range, and another tile that portrays individual trees at close range. Note that the tile used at close range should have the added quality necessary to perceive three dimensions. With the use of three basic tiles and their LOD renditions, the transition of texture will be continuous so that the view is transferred from the forest to a stand of trees within it, and then gradually drawn to the individual trees within the grouping. The quasi-continuous representations of texture need further development.

Another topic for further study is the importance of realism. Although the original tiles were realistic and looked like trees and grass, the realism was lost in the process of creating a continuous tile boundary and eliminating macropatterns within the tile. However, a definite

texture pattern remained, and this pattern was sufficient to develop the texture tile concept and show the viability of this technique in creating textured scenes. A tradeoff study is needed to examine the use of a realistic type of texture versus a generic type of texture, in terms of training effectiveness, creation, and implementation.

The texture used in this report was derived from real imagery. However, the subsequent processing showed the benefits of creating and using synthetic texture. In fact, synthetically derived texture can be used to support the two previously suggested developments in addition to being useful in itself. For example, while a homogeneous microstructure is desirable within a tile (to avoid the pattern effect), macropatterns over a large number of tiles can be used to enhance the quality of an image. The ground plane was completely textured with a grass tile of uniform appearance, with subsequent invisible boundaries. However, homogeneity over so large an area results in a bland appearance. This problem could be eliminated and the overall image improved by 1) superimposing a different macropattern over large areas, or 2) shading to simulate either the effect of different growing rates, grass colors, etc., or the natural rolling terrain.

A prime method of creating synthetic texture tiles is the functional or statistical approach. When the tree and grass tiles were developed in this program, their statistics were matched across the boundary. The statistical approach can be enlarged to develop a generating function of texture with varying spatial and temporal properties.

As an additional task of the texture study program, a dynamic sequence of the divebomb was created that adequately demonstrated the viability of the texture tile approach for generating textured scenes. However, a problem arose in that the texture evidenced scintillation in the dynamic sequence. This problem should be resolved in another study that would identify the mechanism of the temporal aliasing, find a solution for it, and investigate whether temporal aliasing exists in other forms for the texture tile approach and/or other texturing concepts.