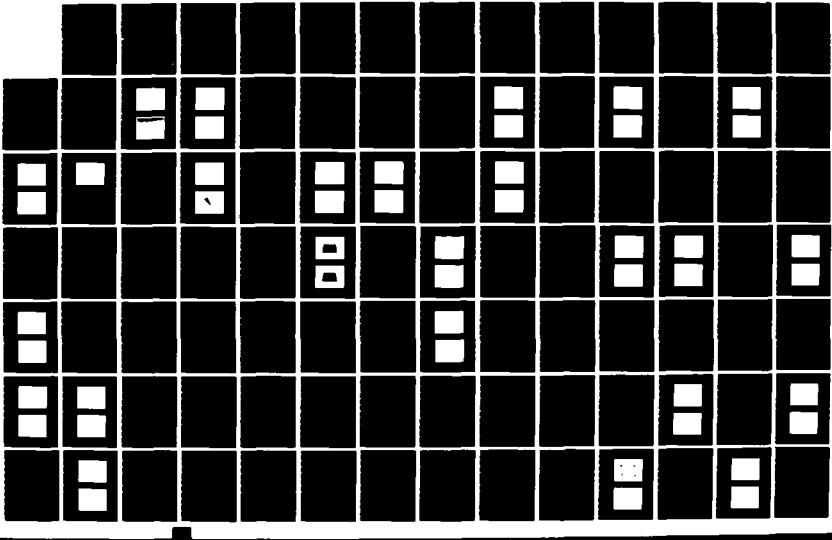
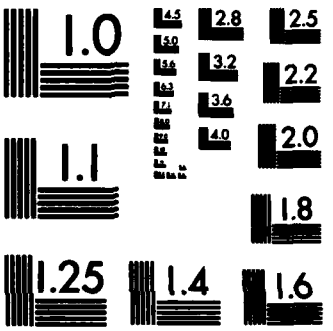
AD-A132 548 INTERACTIVE SURFACE VISUALIZATION USING RASTER GRAPHICS 1/2
(U) AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH
D L SCHWEITZER AUG 83 AFIT/CI/NR-83-31D

UNCLASSIFIED F/G 9/2 NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

INTERACTIVE SURFACE VISUALIZATION USING RASTER GRAPHICS

by

Dennis Lee Schweitzer

An abstract of a dissertation submitted to the faculty of
The University of Utah
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

Original contains color plates: All DTIC reproductions will be in black and white

Richard F. Riesenfeld Chairman, Supervisory Committee

Department Chairman of Computer Science

Department of Computer Science
The University of Utah

August 1983

UNCLASS

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER AFIT/CI/NR 83-31D	2. GOVT ACCESSION NO. AD-A132 548	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Interactive Surface Visualization Using Raster Graphics		5. TYPE OF REPORT & PERIOD COVERED THESIS/DISSERTATION 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Dennis Lee Schweitzer		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS AFIT STUDENT AT: The University of Utah		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS AFIT/NR WPAFB OH 45433		12. REPORT DATE Aug 1983 13. NUMBER OF PAGES 96
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) UNCLASS 15a. DECLASSIFICATION DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

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

18. SUPPLEMENTARY NOTES

APPROVED FOR PUBLIC RELEASE: IAW AFR 190-17

1 SEP 1983

Approved for public release: IAW AFR 190-17.

LYNN E. WOLAVER
Dean for Research and Professional Development
Air Force Institute of Technology (ATC)
Wright-Patterson AFB OH 45433

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

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

ATTACHED

"Original contains color plates: All DTIC reproductions will be in black and white"

DTIC COPY INSPECTED 2

Accession For	
NTIS GRA&I	☑
DTIC TAB	☐
Unannounced	☐
Justification	
By	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A	

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASS

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

ABSTRACT

Computer generated shaded images have evolved to a stage of extreme realism and can easily be confused for photographs of "real" objects. Such imitation of realism has been the major thrust of computer graphic research in recent years with efforts to model such physical phenomena as light reflection, surface texture, shadows, and transparency. Although this realism has been successful in imitating photographs, several applications exist in which the viewers are more interested in an exact knowledge of the three dimensional shape of the surface than simply a realistic shaded reproduction. The goal of this research is to investigate interactive techniques for displaying shaded images with enhanced three dimensionality. Two basic approaches are investigated: direct rendering and viewing of shape information, and imitation of visual perception stimuli. The first approach consists of directly rendering depth and normal information into the frame buffer and examining meaningful ways of viewing the information. The second approach uses the results of research in the field of visual perception to generate cues for enhanced three dimensional visualization. The stimulations which are investigated are texture gradients, gradients of illumination, binocular stereopsis, and motion. Each technique is investigated in an interactive environment to allow greater control over the shape analysis.

INTERACTIVE SURFACE VISUALIZATION

USING RASTER GRAPHICS

by

Dennis Lee Schweitzer

A dissertation submitted to the faculty of
The University of Utah
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

Department of Computer Science

The University of Utah

August 1983

To my wife, Suzi

CONTENTS

ACKNOWLEDGMENTS

There is a great deal of individual satisfaction associated with completing a major successful research effort, yet such work is rarely the effort of a single individual. I would like to express sincere thanks to those organizations and individuals whose support and assistance made this research both personally and professionally successful.

The major share of credit belongs to the Air Force which has provided both the opportunity and financial support for my entire academic career. Within the Air Force I would like to thank the many individuals who believed in my abilities enough to make such opportunities available to me.

I would like to thank my thesis committee for their support and technical advice throughout my work at Utah. A special note of thanks goes to my advisor, Dr. Richard Riesenfeld, who provided generous access to department resources. He also taught me a great deal about scientific professionalism as a friend.

Several graduate students also helped through their technical expertise. Beth Cobb and Spencer Thomas especially deserve recognition for helping to formulate ideas through many informal discussions. The entire Alpha_1 research group helped by providing excellent geometric models used extensively in this research.

Finally, I would like to thank my best friend and wife, Suzi. Her enthusiasm and encouragement were a continual boost to my morale and made our stay in Utah very enjoyable.

This work was supported in part by the National Science Foundation (MCS-8203692 and MCS-8121750) and the U.S. Army Research Office (DAAG29-81-K-0111 and DAAG29-82-K-0176) and the Office of Naval Research (N00014-82-K-0351). All opinions, findings, conclusions or recommendations expressed in this document are those of the author and do not necessarily reflect the views of the sponsoring agencies.

CHAPTER 1

INTRODUCTION

Interactive computer graphics is an increasingly valuable tool for many diverse applications including design, analysis, entertainment, and instruction. Advances in graphic hardware and software have led to interactive generation and manipulation of realistic images. The ability to produce high quality images of computer models provides an effective means of visual feedback quite unique in scientific research.

Although two dimensional images have many applications, the most impressive visual results have been achieved in the rendering of three dimensional models. Effective representation of three dimensional images on a two dimensional display device requires some means of imparting depth information to the viewer. The trend in computer graphics has been to provide such depth information by imitating the physical properties of a real scene in an attempt to make images as realistic as possible. Researchers outside of computer graphics have also been interested in the problem of three dimensional perception in human vision. A short discussion of some of the approaches taken by each research group will be presented. This will be followed by a description of a new approach which applies the principles of results in visual perception research to techniques currently used in computer

graphics.

1.1 Realism in Computer Graphics

The emphasis in computer graphic research has been the generation of realistic images with impressive results. Computer generated images can easily be mistaken for photographs of three dimensional physical objects. This success in realism has been achieved through many advances both in hardware and graphic algorithms.

Until recently, simple plotters and vector displays were the only affordable display medium available to most users of computer graphics. Today, because of technological advances, high quality raster displays are common and provide potential realism to many application areas. Current research is being directed toward the development of new modes of display providing even greater realism such as a three dimensional volume viewer [3]. Research is also being directed to the creation of "smart" systems which perform some of the rendering functions in hardware to speed picture generation [21].

In graphics algorithms, several advances have resulted in the capability to generate realistic images in a reasonable amount of time. Realism has been achieved using both simple methods such as perspective and hidden lines and surfaces, and complex techniques such as simulated surface texturing and shading. Speed has become a major performance factor with the increased complexity of scenes to be drawn, and much emphasis has been placed on taking advantage of image and display characteristics to provide faster processing [45].

1.2 Research in Visual Perception

The field of visual perception has been studied in detail for many years by artists, physiologists, psychologists, and others. Artists have attempted to capture three dimensional realism on two dimensional media by mimicking depth cues [49]. The history of art is rich with examples of the development of using such cues as shading and perspective to create great realism. Physiologists have studied the mechanics of the visual system. Psychologists have attempted to determine what factors affect visual perception through experimentation [26]. Although the artist's approach has been the direction taken in computer graphics to achieve realism, it is the psychologist's approach which is of primary interest in perception.

Part of the psychological research has dealt with defining what stimulations the human eye responds to, and how these stimulations are interpreted. Haber [26] has classified three types of stimulations with examples as shown in Figure 1. The static stimulation ideas of perspective, shading, and occlusion are well known to the graphics community and serve as the basis for many of the recent efforts toward realism. The concept of surface texture as a means for understanding shape is not a commonly used technique, but can be a powerful perception tool [26]. Dynamic stimulation deals with perception via the motion of an object. Relative motion, sequences of transformations, and changing perspectives are all strong perception clues [8]. Binocular stimulation is a result of the separation of the eyes and is simulated in computer graphics using various devices.

1. Static Monocular Stimulation

   - Surface textures, gradients
   - Linear perspective
   - Gradients of illumination
   - Occlusion

2. Dynamic Monocular Stimulation

   - Radial movement
   - Motion parallax
   - Motion perspective

3. Binocular Stimulation

Figure 1: Classification of Visual Perception Stimulations

1.3 Interactive Surface Visualization

The realism of smooth shaded images provides greater understanding of the three dimensional shapes of surfaces than traditional line drawings as shown in Figures 2 and 3. However, static smooth shaded pictures are not always sufficient to achieve shape understanding. Figure 4 illustrates a realistically shaded surface whose shape is not apparent from the given point of view. A view from a different point such as Figure 5 more clearly shows the surface shape. Although this is a contrived example, it illustrates the shortcomings of relying on realistic shading as the only clue to three dimensionality. When displaying a complex surface shape, several portions of the image may suffer from the same ambiguities.

The concept of using perceptual principles to enhance understanding of graphically displayed information has been applied to spatial arrangements and color selection [27, 47]. The goal of these

Figure 2: A Line Drawing of a Smooth Surface

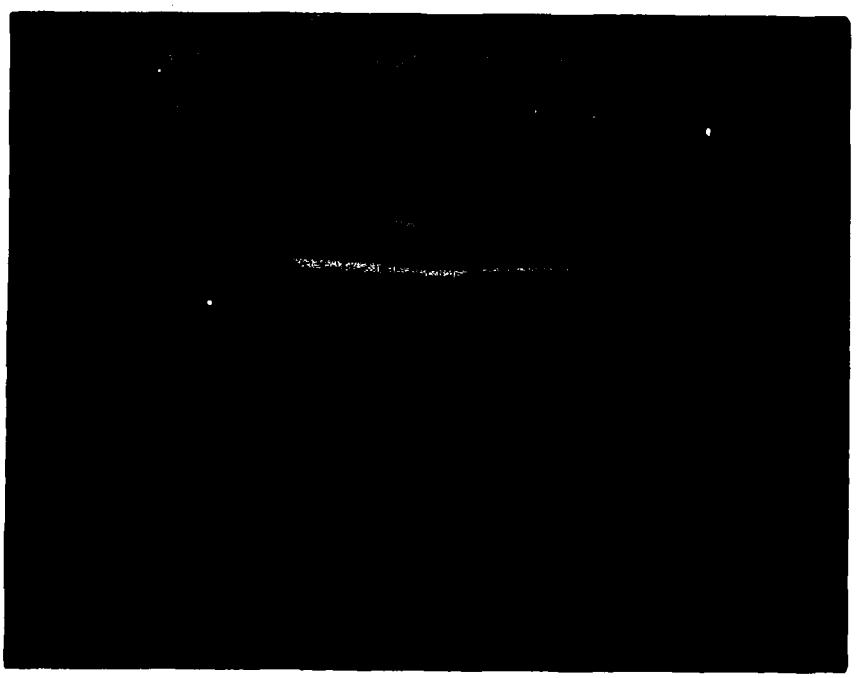


Figure 3: A Raster Rendering of a Smooth Surface

Figure 4: An Ambiguously Shaded Surface

Figure 5: An Oblique View of Ambiguously Shaded Surface

techniques has been to improve man-machine communications. Effective visualization of a three dimensional surface should take advantage of all stimulations available in the eye-brain perceptual link.

The goal of this research is to examine interactive methods of using visual perceptual stimulations to enhance the perception of a third dimension in computer generated shaded images. Although realistic images incorporate a certain number of these stimulations, additional information can be gained from interactive application of stimulations which trade extreme realism for enhanced three dimensionality. Interaction is necessary to allow user adjustment of visual parameters to maximize visual benefit.

The following chapters will describe interactive visualization techniques using raster graphic images. Chapter two will discuss direct visualization of three dimensional information and will not deal with a specific perceptual stimulation. The remaining chapters will deal with the specific stimulations of texture gradients, gradients of illumination, binocular stereopsis, and perception through motion. Each chapter will present the currently practiced computer graphics approach to simulate the stimulation. Chapters three through six will present the perceptual basis for the stimulation being discussed and present an interactive algorithm for using it. A brief appendix is included to describe the frame buffer system used in this research.

CHAPTER 2

DIRECT SHAPE VISUALIZATION

2.1 Introduction

One approach to enhanced three dimensional visualization is to directly view the three dimensional parameters. Thus, for an arbitrary image, the viewer would not only see the realistic shade and color associated with each point on the surface, but would optionally be able to discern shape information such as depth, normal, and curvature at the point as well. This additional knowledge could resolve ambiguous shading and give a clear understanding of the shape of the surface.

Concise presentation of such shape information in a form that facilitates point by point registration with the shaded image can be accomplished through some means of rendering the information into the frame buffer with the image. Such "functional" rendering is not new to computer graphics, and has been used for several different applications such as stress analysis [11], pressure and heat fringes [9], and music visualization [32]. The frame buffer's capability to display different color and intensity values makes it a useful tool for mapping functional information into a visual form.

Functional rendering is only valid if the information is displayed in a way that is readily understandable when visualized. For example, arbitrary color assignment to depth values would result in a colorful

collage but an image which is difficult to comprehend. To provide for improved understanding, some means of interaction is desirable. An interactive method will be presented which renders depth and normal information for an arbitrary surface and provides natural viewing for shape understanding.

2.2 Previous Work in Computer Graphics

Researchers in computer graphics have investigated rendering of different types of shape information. A display parameter which has been examined extensively is Gaussian curvature [20, 16]. Because this parameter is useful in geometry, but not directly viewable in a shaded image, its display can be of valuable assistance to surface designers. This information is primarily for higher order shape analysis, but can provide some direct shape information such as the location of hollows and protuberances [16]. However, to obtain the benefits of such pictures, it is necessary for the viewer to have an understanding of the meaning and significance of Gaussian curvature, and to have experience in viewing such information.

Research by Forrest dealt with the direct display of contour lines on a shaded image [20]. The advantage of using contour lines versus some other visual representation for depth is that it is a conventional drawing technique which is naturally understood by most viewers. The difficulty with the technique, according to Forrest, is the computational complexity associated with finding contour lines.

An attempt to directly view surface normal information led to the technique known as a "hedgehog" surface or overlaying lines normal to

the surface directly on an image [33]. One problem associated with this approach is the resulting confusion when an excessive number of normals are drawn. Another problem is the lack of information available from the length of the normal vector. Since several normals would project to the same vector, the length of the projected vector provides necessary information concerning the exact orientation of the normal. However, if too few vectors are drawn to provide for length comparison, than ambiguous normal information may be interpreted. Figures 6 and 7 illustrate some of these deficiencies.

Interactive viewing of functional information is typically accomplished through the use of the color map [42] [46]. Functional data is discretized into the range of integers representable in a pixel. Rather than representing color intensities as in a typical rendering, these values are used instead to look up intensities in a color map table. The advantage of this technique is the relative speed in which the color map can be modified since the number of values requiring change is only 2^n for an n-bit frame buffer as opposed to changing an entire 512 by 512 pixel image. This allows for immediate changes in the visual appearance of an image without re-rendering the scene. Applications which have used this technique are false coloring, gamma corrections, simple motion, and dynamic shading [42] [41].

2.3 Visualization of Depth and Surface Normals

Shape parameters which will be rendered and interactively viewed are depth and normal information for each pixel of an image. Although higher order information such as Gaussian curvature is advantageous for

Figure 8: An Image with Surface Normals

Figure 7: Interfering Surface Normals

certain applications, depth and normal data are of primary interest when the goal is to enhance three dimensional visualization. A convenient property of these parameters is their availability during normal rendering without additional computation. A z-buffer rendering algorithm using normal interpolation for shading computes a depth and normal for each pixel in the process of generating a shaded image. By simply writing these values into the frame buffer, no additional computation is necessary for the enhanced rendering.

Interactive viewing and visual representation will be discussed for presenting both depth and normal information. In each technique, a frame buffer image file is rendered with shade intensities stored in one color plane and either Z depths or normal values stored in the other two planes. This allows the viewer to interactively switch between a shaded representation of the image and various enhanced functional visual representations.

2.3.1 Depth

Depth knowledge is of immediate interest when trying to visualize the three dimensionality of a surface. However, simply digitizing and viewing depth values as intensity does not necessarily provide a great deal of depth information as illustrated by Figures 8 and 9. There is explicit information available in the figure, but because of the "unnaturalness" of the shading, it is only useful as a general depth indicator. For example, a viewer cannot accurately compare two points on the surface to determine if they are at the same depth. The information would be more useful if it could be displayed

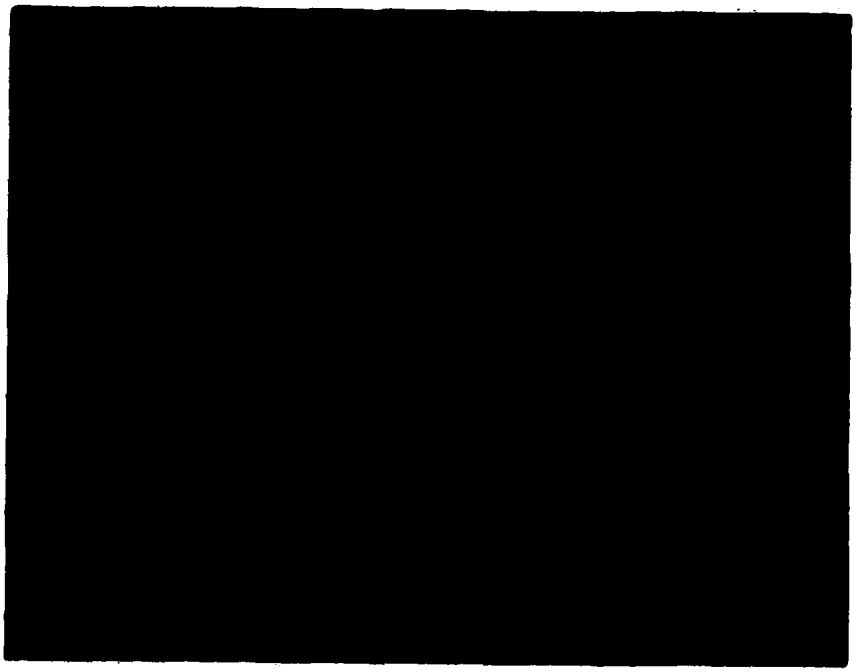


Figure 8: A Realistically Shaded Image

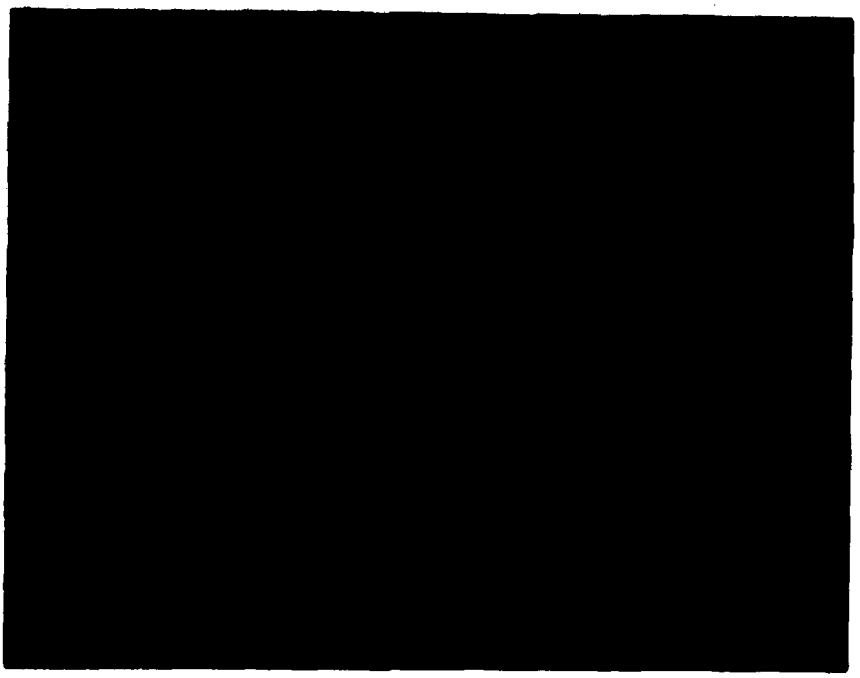


Figure 9: Depth Values Displayed as Intensities

simultaneously with the natural shading, either by color coding of depths or using multiple images.

As stated by Forrest [20], a more natural viewing mechanism for depth information is the use of contour lines because people are familiar with looking at contour lines on maps. Because individual depth values from the hidden surface computation are already present in the frame buffer, the depth image in Figure 9 actually represents "continuous tone" contours where different intensities represent different depths. To make these contours more familiar, the color map can be used to create distinct contour bands at discrete depth increments. For a given distance between contours, all color map table entries modulo that distance are zeroed creating black lines on the image as shown in Figure 10. These lines represent pixels of approximately equal Z depths, or contour lines. The use of the color map allows the user to interactively change the distance between contours as shown in Figure 11.

Further interaction is also possible with the use of the color map. Contours can be "stepped" through an image by simply shifting entries such that successive depths become contour lines. This can be done either in discrete steps or smoothly so that the contour lines "roll" across the figure. This added real time feature shows how depths are changing locally over the surface, and can give insights missed with contour lines a fixed distance apart.

In addition to contours, another visual representation is interactive specification of Z regions. The regions are specified by defining two Z values (or Z planes) and color coding the parts of the

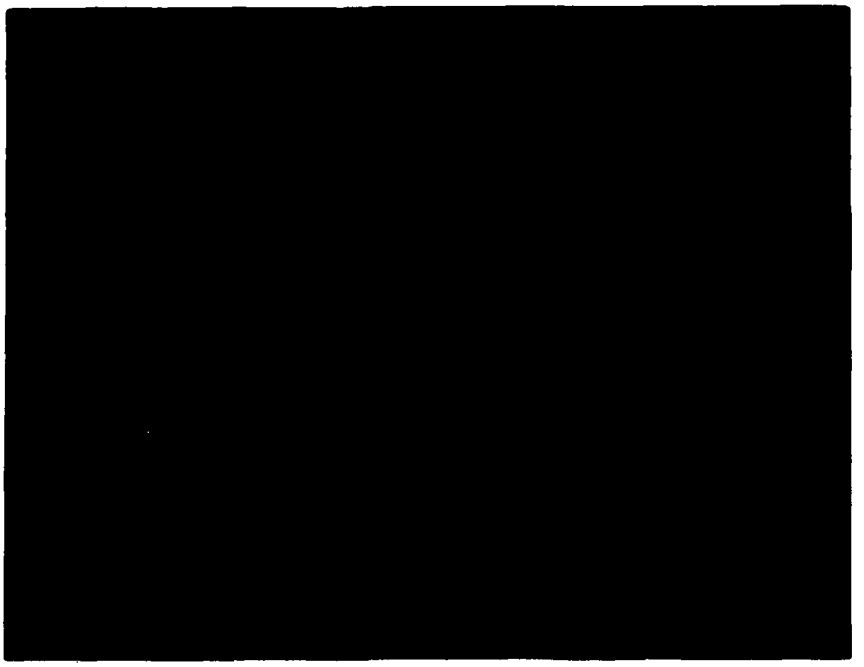


Figure 10: Color Map Contour Lines

Figure 11: Different Depth Distances Chosen Interactively

image to distinguish "in front of" and "behind" each plane. Figures 12 and 13 show images with one and two specified Z planes and the depths of the surface with respect to the planes. By using the color map to perform the color highlighting, the planes can be changed interactively to classify depth relations for different parts of the surface.

Another visual representation of image depth is a cross section of the surface along a specified line on the projected image. Figures 14 and 15 show a display of such information on a line drawing system used in conjunction with the frame buffer. The depths of each pixel along the chosen path are plotted as heights in the two dimensional cross section. This two dimensional depiction of the depth is useful when investigating relative depths along a straight path such as a scanline.

The visualizations described provide enhancements to understanding depths in an image. The same interactive techniques are applicable to the visualization of other values such as depths measured from a specified base. In this way true contour lines could be visualized on an object tilted for enhanced perspective. The fact that Z depths are available during the normal rendering procedure simplifies this specific value computation and provides increased understanding of the shape of the object whose projection formed the image.

2.3.2 Surface Normals

Depth information is necessary but not sufficient to understand shape. Information about the orientation of the surface, or the

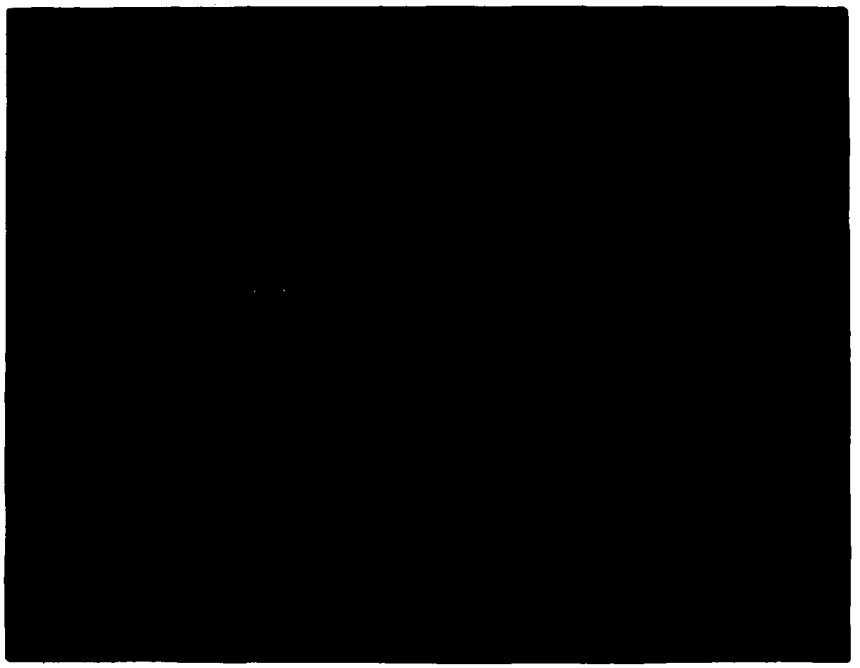


Figure 12: Depth Discrimination by Z Plane Specification

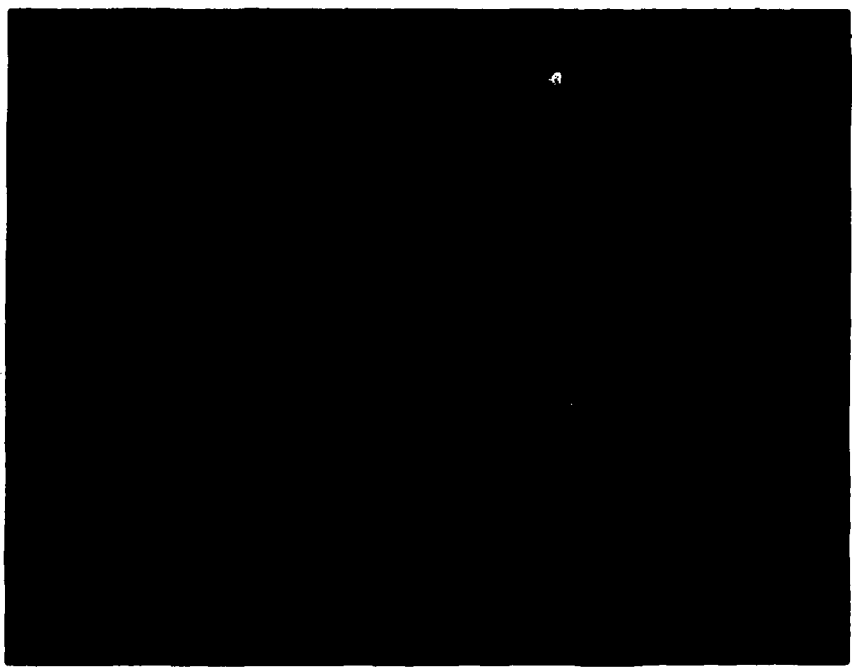


Figure 13: An Image with Multiple Z Planes

Figure 14: A Specified Cross Section Line

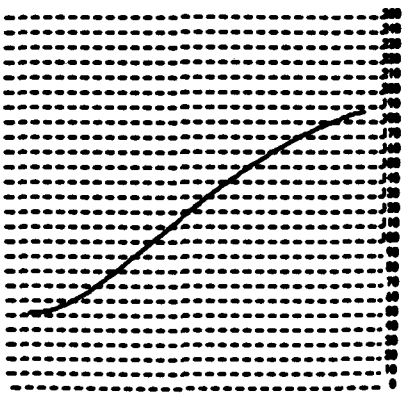


Figure 15: A Plotted Depth Cross Section

relative change in depth is also desirable. A direct measurement of this factor is the surface normal at every point on an image. As previously stated, such information is available when rendering an image with normal interpolation for shading. Direct visualization of such information differs from depth in that three values are present per pixel instead of a single depth value.

As with direct viewing of depth, direct viewing of surface normal information is hard to interpret. Figures 16 and 17 illustrate two possible direct renderings of surface normal information. Figure 16 shows the effect of assigning the X, Y, and Z components of the normal to red, green, and blue with intensity representing magnitude. Figure 17 represents an image containing spherical coordinates of surface normals in the red and green color planes and a shade intensity in the blue color plane. Although specific information is available in each image, the result is meaningless unless closely scrutinized by a knowledgeable observer. It is possible to produce contours similar to depth for each component, but they are hard to interpret and it is not clear that such information is useful.

One visual representation found to be useful is to display ranges of the individual normal components. The significance of the Z component of the normal has been investigated by researchers in computer graphics. Obviously, when it is opposite the viewer, the associated pixel is not visible. A more subtle property which has been used for image rendering is the fact that when the Z component equals zero, the point lies on a silhouette edge which separates visible portions of the object [40, 31]. A similar interpretation of the X and

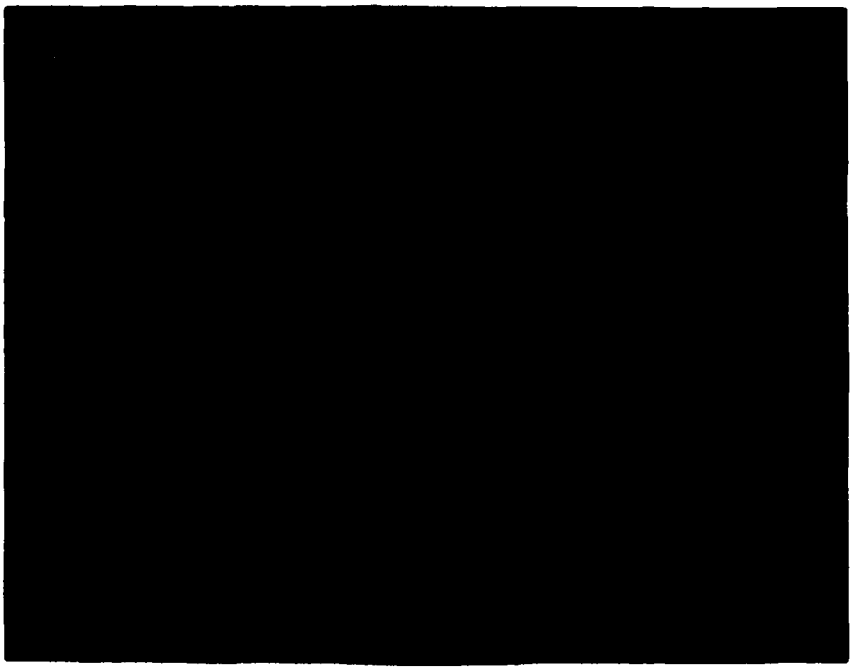


Figure 16: A Direct Rendering of X, Y, Z Surface Normal Components

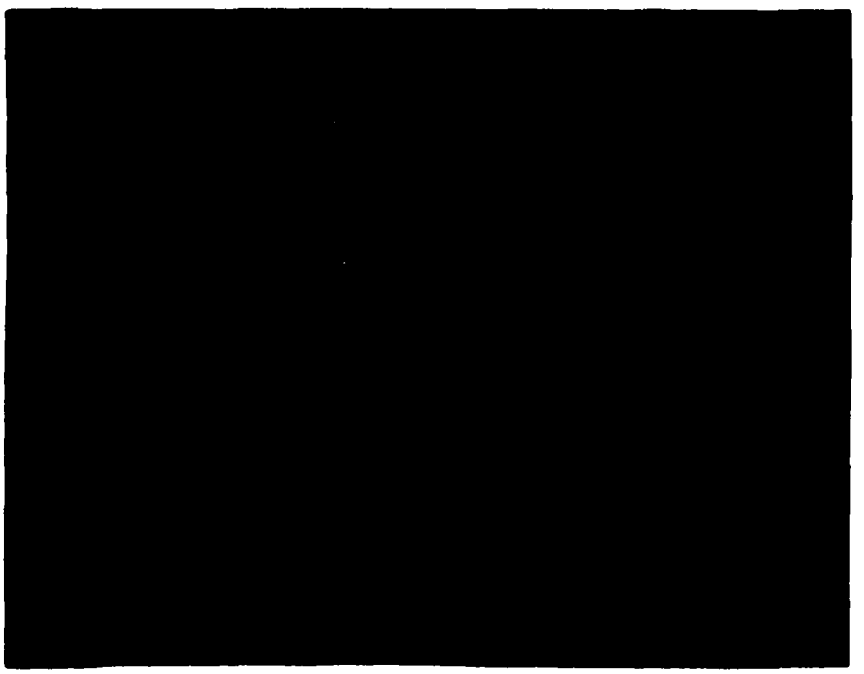


Figure 17: A Rendering of Surface Normal Spherical Coordinates

Y components can be understood. A line representing all normals with the X component equal to zero can be thought of as a "ridge" which represents a local minimum or maximum in the X direction. Similar to the Z = 0 line, this X ridge would be a silhouette edge if the surface were viewed along the X axis. On either side of this line the X component "falls off" the zero value. If such ridges are simultaneously displayed for both X and Y, the intersections represent the occurrence of peaks, dimples, or flat spots in the surface. Figures 18 and 19 are examples of such ridges. The cyan line represents normals with an X component within a small range of zero, and the purple line shows the normals with zero Y values.

Comparing the normal image to the corresponding contour image in Figure 11, the intersections of the normal ridges can be interpreted. The crossing at the upper right of the image represents the peak shown in the contour image. Similarly, the lower left crossing is a flat spot. If the image were viewed down the Y axis, the purple line would be a silhouette edge of the surface, and likewise for the cyan line if viewed along the X axis.

Real-time manipulation of a "normal" image is done by using the color map in a manner similar to displaying depth contours. The user can interactively specify a range for each normal component and immediately view the resultant image. Figure 20 shows the previous image with all portions of the surface containing negative X normal components highlighted. This area represents that portion of the surface that would be in shadow from a light source directly to the right of the image. Similarly, Figure 21 shows the range of all

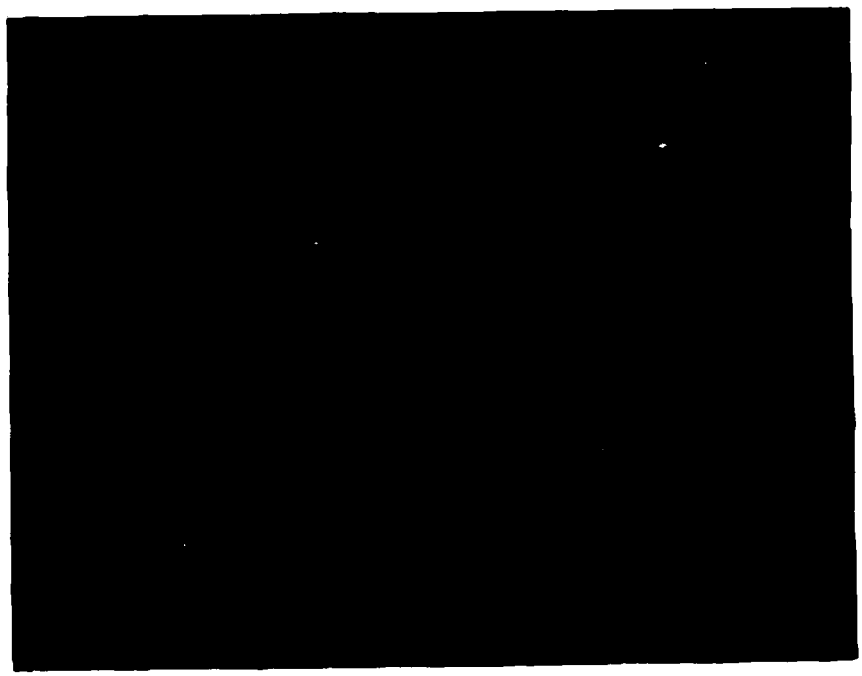


Figure 18: X Normal "Ridges" in an Image

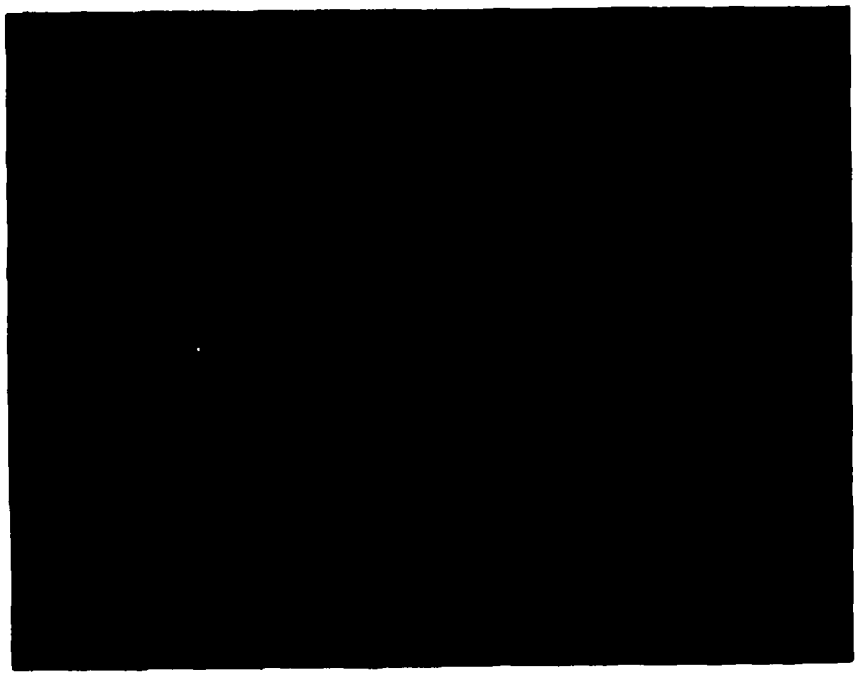


Figure 19: Y Normal "Ridges" in an Image

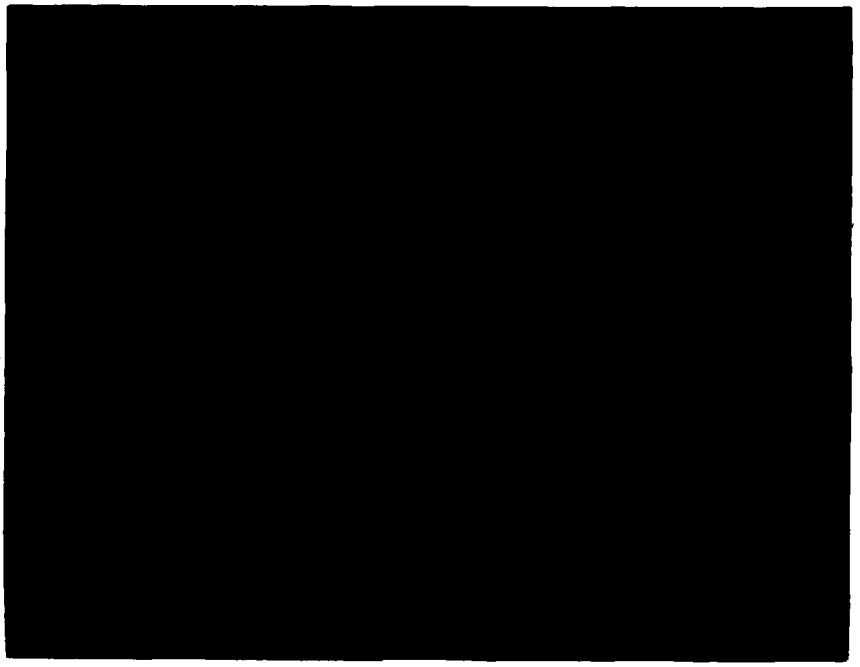


Figure 20: A Normal Range of Negative X Components

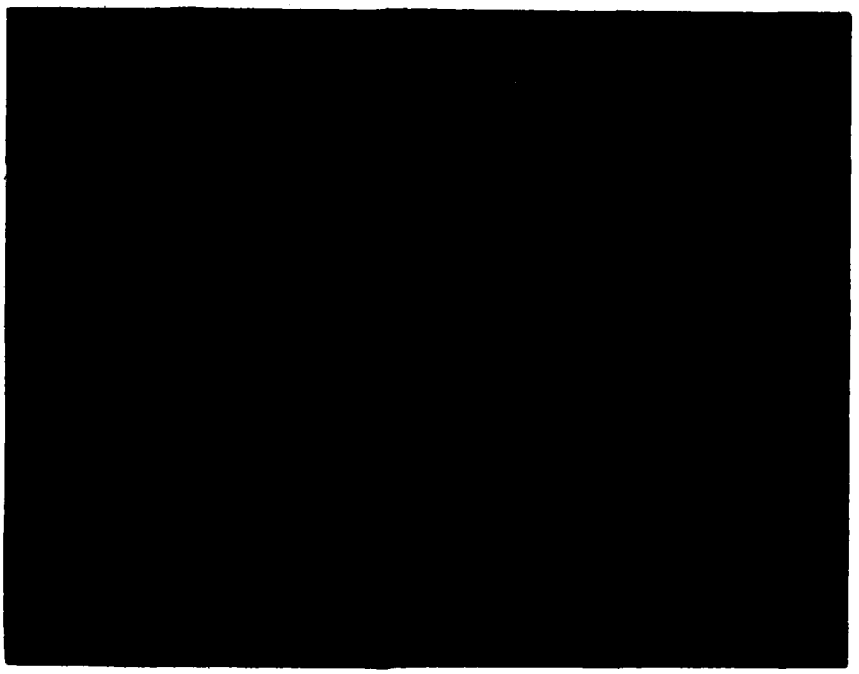


Figure 21: A Normal Range of Positive Y Components

positive Y normal components. The ability to change the ranges in real time allows the user to see how the normals change over a local area of the surface.

Another normal visualization technique is an interactive version of the "hedgehog" concept which was discussed earlier. Using the normal information rendered into the image, a single surface normal vector is drawn in the overlay plane at the point of the cursor. As the cursor is moved across the image, the normal information is read back from the frame buffer and the surface normal vector is redrawn accordingly. Figures 22 and 23 illustrate a single surface normal vector and a sequence of images with a moving vector. By using the overlay plane and reading back only a single point at a time, real-time feedback is achieved when moving the vector. This technique avoids the problems associated with a "hedgehog" surface as previously discussed. The vector can be moved to the area of interest without the confusion of several overlapping lines covering the picture. The ability to interactively move it helps to avoid misinterpretations of the length of the projected normal. Also, additional information can be gained about local changes in the surface orientation by viewing the effects of moving the vector around a small area of interest.

2.4 Results

Interactive functional rendering of shape information appears to assist in understanding the depth and orientation of a surface. Because the visual representations are unnatural to view, however, they are primarily useful in analyzing specific portions of an image, rather

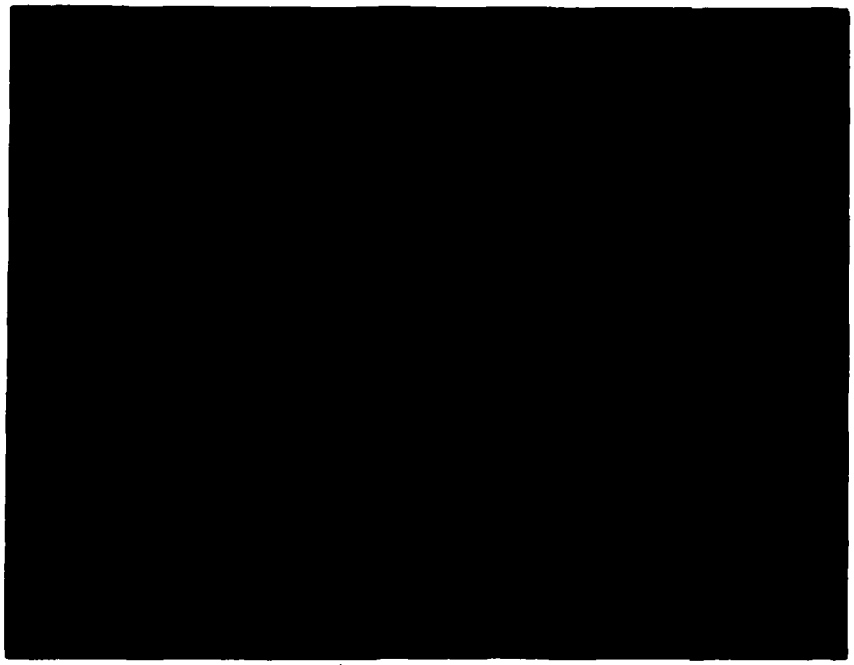


Figure 22: An Image with a Surface Normal Vector

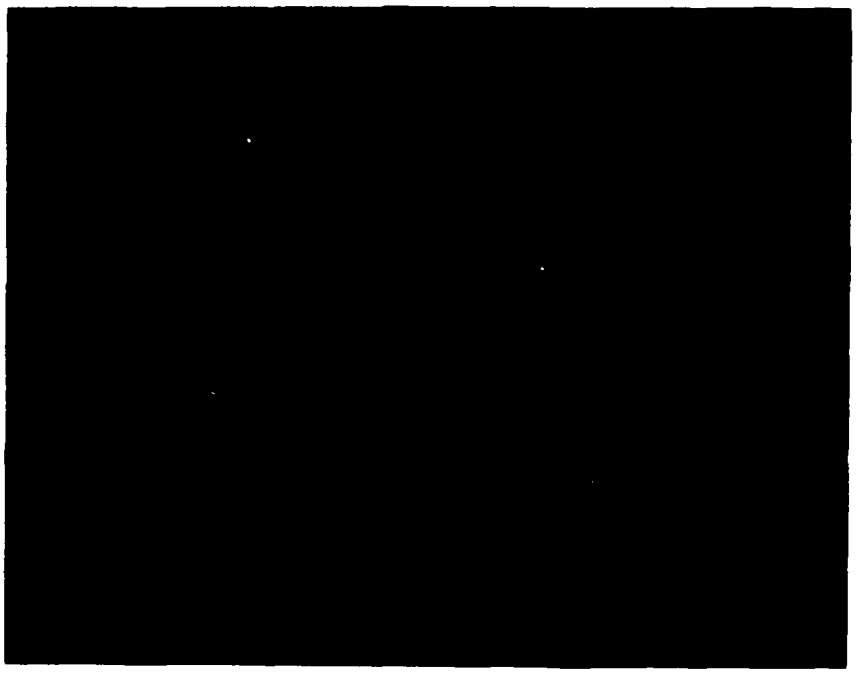


Figure 23: A Sequence with a Moving Normal Vector

than getting an overall feel for the shape.

It was found to be extremely beneficial to retain the natural shading information in one color plane of the image, so that one could switch back to this view to clarify the functional representation. In the normal range technique, this was accomplished by simplifying the lighting model to that of cosine shading with the light source at the eye. In other words, the Z component of the normal was actually the shade for this simple lighting model, and could thus be used to perform both functions.

Real time interaction was very effective in all techniques, enhancing the shape information given by the static contour lines and normal ridges. By visualizing the change in depth and normals as one modifies the parameters, much additional information can be gained about local surface characteristics.

CHAPTER 3

TEXTURE GRADIENTS

3.1 Introduction

Almost all physical surfaces contain some microstructure or texture visible to the human eye. This texture gives us information such as the type of material which composes the object and the relative smoothness or coarseness of the surface. Computer generated images which portray such texture provide the same types of information and appear remarkably realistic, but the algorithms for generating such images are generally very time consuming.

Perceptual psychologists have analyzed a different type of information provided by texture: that of space perception [22, 8, 26]. The microstructure of a surface provides a somewhat regular pattern for visualization. The changes in this pattern, or texture gradient, give strong cues to the orientation and depth of the surface. Extensive studies and experiments have shown that there are three characteristics of texture which provide this perceptual information: size, shape, and density. Changes in these components because of perspective projection provide knowledge about surface depth and changes in the orientation of the surface.

To avoid confusion in terminology, "texture" will not refer to the intuitive meaning of changes in the normal direction due to the surface

microstructure. Rather, it is the pattern on the surface which is of interest. This pattern may be caused by different colors as in a tiled floor, or different intensities because of changing normals such as the skin of an orange. As will be discussed, it is the changes in this pattern which provide one basis for 3-D shape perception.

A texturing technique will be described which approximates the texture changes caused by distance from the viewer and orientation of a surface without attempting to exactly render a realistic texture. This approach supplies several of the same visualization cues as provided by exact texture rendering algorithms without the complexity of generating a realistic texture. Thus, "artificial" texturing is a means for providing an inexpensive aid to visualizing the shape of a shaded surface.

3.2 Visual Perception of Texture

Surface texture provides a fairly regular pattern over a large surface. For example, the shape of individual blades of grass in a field may be random, but taken as a whole, the blades create a constant textural pattern. There are also regular textures such as bricks in a wall or a checkerboard. As a surface is slanted away from the eye, the texture pattern gradually changes due to perspective projection. This gradual change is called the texture gradient [26]. Perception researchers isolate three separate gradients which influence perception: texture size, shape, and density. Several experiments have shown that each of these gradients strongly influences the perception of surface depth and slant [8]. Figure 24 is a simple example of a

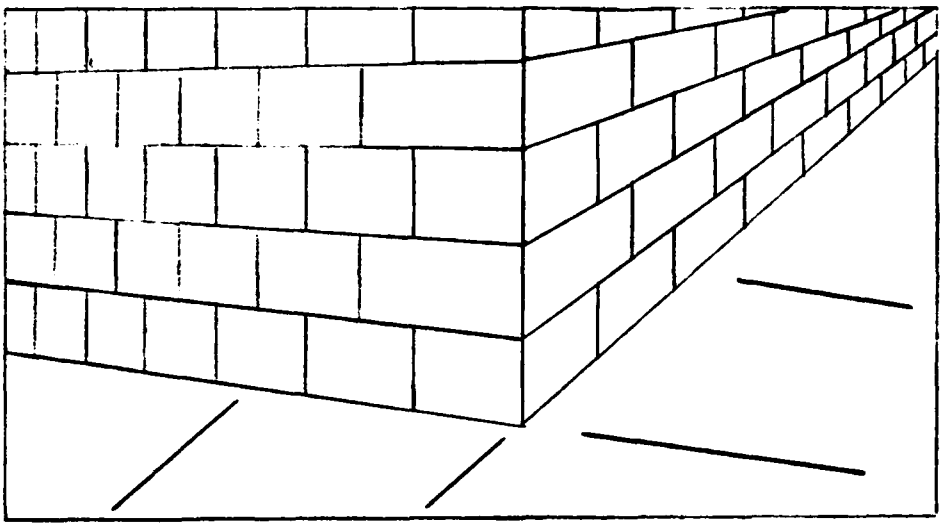


Figure 24: A Simple Texture

textured surface, and illustrates the effects of texture size, shape, and density to perception.

3.3 Existing Algorithms

In an ever increasing quest for realism, computer graphic researchers have developed different methods to model surface texture. Catmull suggested one of the earliest techniques for mapping a defined texture onto subdivided patches [10]. His rendering algorithm subdivided parametric patches into smaller subpatches until the boundaries of the subpatch were contained within a single pixel. Once at the pixel level, the parametric coordinates of the pixel corners specified an area in a texture definition array defined in parametric space. The color of the pixel was determined by averaging the values in this texture definition area.

Blinn and Newell took Catmull's technique a step further by

introducing a more sophisticated filtering method [4]. The quadrilateral formed in texture definition space by the (u,v) corners of the pixel was used as the base for a square pyramid to weight the texture values. Their paper also discussed different techniques for creating texture patterns such as (u,v) functions, hand drawings, and scanned photographs.

Blinn also suggested a more novel approach to creating actual bumps in the surface via normal perturbation [6]. In this method, a perturbation function was mapped onto the surface to modify the normal vectors giving the illusion of wrinkles. This technique produced strikingly realistic images, but at about twice the time cost of standard texture mapping.

Because of the amount of detail involved, textured images are highly susceptible to aliasing. More recent approaches have concentrated on the filtering aspect of the texture mapping [18] [34].

In each approach to creating texture the same two general functions must be performed: a mapping is required from texture space to image space, and filtering is necessary to create acceptable images. Both of these are computationally expensive. Additionally, the mapping procedure is extremely data/texture dependent, and is not always easy to define. For example, if one had a collection of polygons representing a surface and wanted to simulate bricks, it is not obvious how to map the coordinates of each polygon to the brick texture definition. Although both of these functions are necessary to create realistic texturing, a simpler approach can be used to obtain the perceptual benefits of texture: artificial texturing.

3.4 Artificial Texturing

Artificial texturing is a means of applying a pattern to a surface in order to obtain the perceptual benefits of texture gradients. The goal of this approach is not to produce realistic textured images, but rather to aid the visualization of rendered surfaces. No knowledge of the type of object space data is required, but it is assumed that depth and normal information is available at each pixel as is required for z-buffer hidden surface rendering with normal interpolation for shading. Each of the perceptual characteristics of texture mentioned earlier will be presented, together with an examination of the information required to generate each.

3.4.1 Texture Size

Psychological experiments have shown that the texture pattern most effective in aiding perception is one of equally sized elements positioned on the surface at a regular density [8]. An example would be a set of equally spaced grid lines. As mentioned earlier, the problem with projecting such lines to an arbitrarily warped surface is that some mapping knowledge is required to know how such lines continue from surface element to surface element. However, a type of pattern which avoids this problem is one composed of individual texture elements, or "texels," such as a pattern of disjoint squares on a piece of cloth. The same texture gradients apply to individual texels: their changing size, shape, and density.

Along with the advantage of not requiring mapping knowledge, the use of individual texels also lends itself to a simplifying assumption:

that each texel is only a function of the center of the individual element. Although this assumption does not imitate realism, it allows for a significant decrease in relative computation time. This simplification will be used in the calculation of all three texture gradients.

The size of each individual texel is strictly a function of the depth of the element from the projecting plane. The idea of perspective projections is well understood in the field of computer graphics and has been addressed by many authors [19]. Using the same simplifying assumption as previously stated, the size of the texel can be determined as a function of the depth of the center of the element. Thus, if square texels are being generated, the width of an element for a given pixel is defined as:

Projected Width = Width / (Z Depth / Eye Distance + 1)

The eye distance is from the eye to the projection plane and is a scale factor for the perspective transformation. When producing artificial textures, element size is determined from the z value at the pixels representing texel centers.

3.4.2 Texture Shape

When equally shaped texels are used, the shape gradient is a strong clue as to the surface orientation [35]. The shape will be a function of the surface orientation over the texel as well as the orientation of the texel itself. For example, square elements will not necessarily be aligned with all edges parallel, and would appear

unnatural if so forced. Completely symmetric texels, i.e. polka dots, are used in this algorithm to avoid this problem.

The shape as a function of surface orientation can be simplified using the same assumption as before, that it is strictly based on the orientation of the element's center. This will not result in exact realism since individual texels will not follow the surface curvature over the area of the element. However, as a visualization aid, this assumption allows for a simple texture generating method which produces perceptually helpful gradients. The shape of individual polka dots on the surface can be determined by projecting a circle perpendicular to the normal at the circle's center onto the display screen. Figure 25 shows a graphic description of this projection. The projection of the circle will be an ellipse whose eccentricity depends on the direction of the normal.

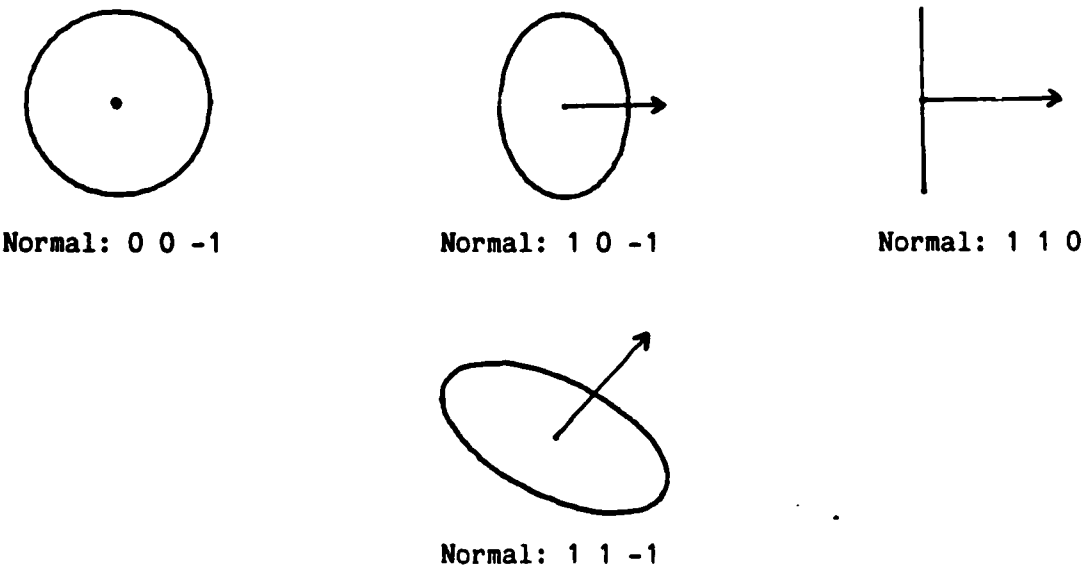


Figure 25: Circle Projections for Different Normals

3.4.3 Texture Density

Another texture gradient providing perceptual information is the density gradient. When the texture pattern has regular density, the changes in the distance between elements, or density gradient, on the textured surface give clues to the depth and orientation of points on the surface. Researchers have distinguished two types of density gradients, compression and convergence [8]. Compression is a decrease in distance between texels due to the perspective transformation of elements at a greater depth from the viewer. Convergence is a result of elements being projected closer together when the surface is at a slant to the viewer. Thus, compression is a function of depth while convergence is based on normal information. As with the other two gradients, the approximation of using only the element's center simplifies the texture density calculations at any given point on the surface.

Density information is required to determine where to put each texel. If only a density change is desired in either the horizontal or vertical direction, a simple solution is possible. The normal information available at each pixel during rendering can be used to estimate the surface distance between pixel centers. For a given density, the rendering process can determine where the next element goes by accumulating the distances across a scan line. Similarly, images could be scanned in a vertical fashion to create a vertical density change. Although either of these density techniques may provide perceptive information, the images they produce appear awkward and off balance.

To produce density information in both directions requires more detailed computation than simple horizontal or vertical distances. One solution is to approximate areas covered by each pixel's boundaries rather than distances between pixel centers. For a given normal at the center of each pixel, an approximation can be derived to the area bounded by the projection of the pixel boundaries onto the surface. This approximation can be modified by a perspective factor based on the depth of the surface at the pixel center. For a specified texture density, the placement of texels can be determined by summing surrounding pixel areas until enough area is found to place one element. Because the area calculation encompasses both depth and normal information, both compression and convergence density results. Figures 26 and 27 illustrate the difference in images using the variable density calculation.

3.5 Implementation

The implemented version of artificial texturing uses a modified z-buffer image file as input. Rather than the standard red, green, and blue components at each pixel, the input z-buffer contains normal and intensity information. This data is readily available to the rendering program and can be encoded into the same size fields for compatibility. For this implementation, data was obtained from a rendering program which contained a modified shading routine to return spherical normal coordinates in the red and green values, and an intensity in the blue. It requires no additional computation time to generate such input files beyond normal rendering.

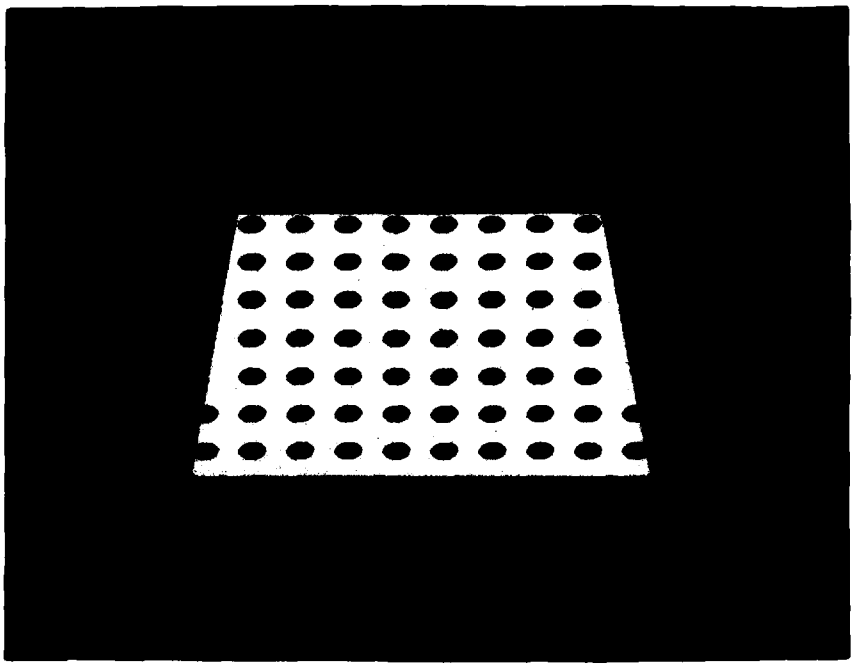


Figure 28: A Surface with Static Density Texture

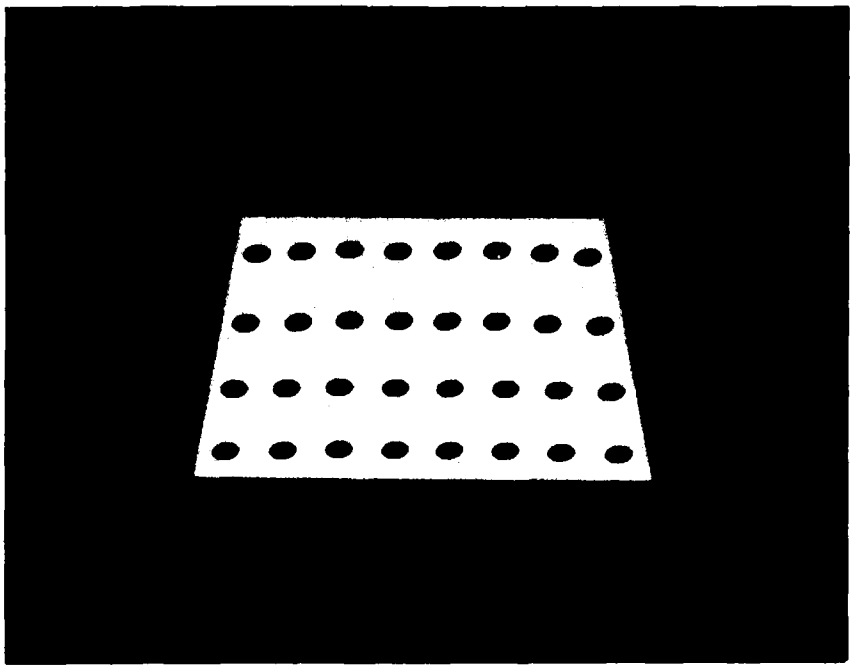


Figure 27: A Surface with Variable Density Texture

3.5.1 Texel Projection

The shape of each texel is determined by projecting pixels surrounding the texel center onto the plane defined by the center normal. The distance from the projected point to the texel center is compared to the radius to decide which pixels lie within the projected texel. The equation for this distance is:

Normal: N_x, N_y, N_z

Point: P_x, P_y (relative to texel center)

Projected Point $P_z = (P_x N_x + P_y N_y) / N_z$

Distance to Texel Center $= (P_x^2 + P_y^2 + P_z^2)^{1/2}$

The texel radius is specified by the user and modified by the z depth of the texel center to reflect changes in the element's size due to perspective as described earlier.

A simple filtering mechanism is incorporated during texel projection. For each surrounding pixel, the corners of the pixel are projected and compared rather than the pixel centers. Thus, each surrounding pixel will have an associated count of 0 - 4 corners inside of the projected texel. This count can be used to weight the color of the texel with the surface to provide a simple filter. The weighted color is then modified by the intensity at each pixel. This is important to ensure that dots appearing in darker shaded portions of a surface are similarly darkened. Figures 28 and 29 show the different shapes and sizes of projected dots caused from a variety of normals at

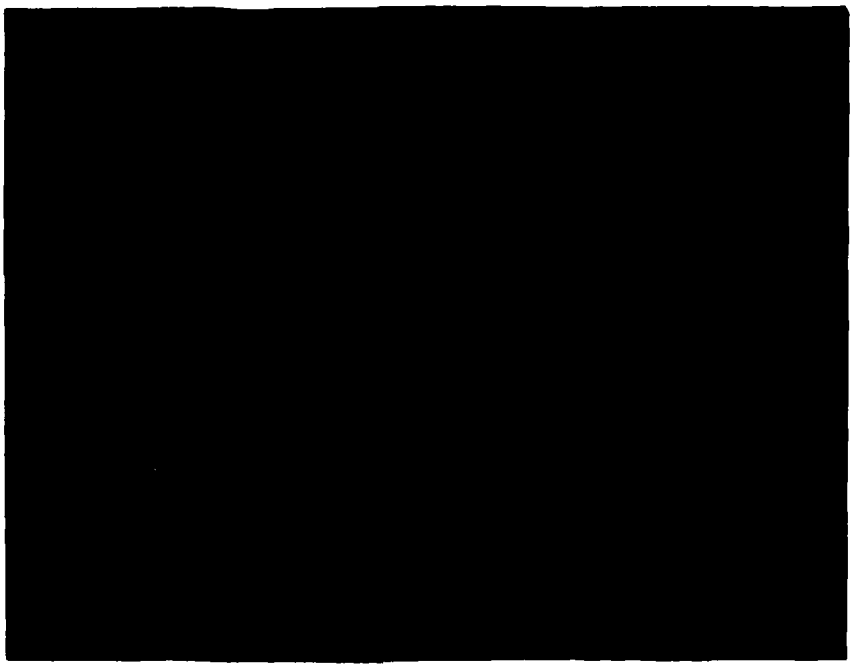


Figure 28: A Textured Torus

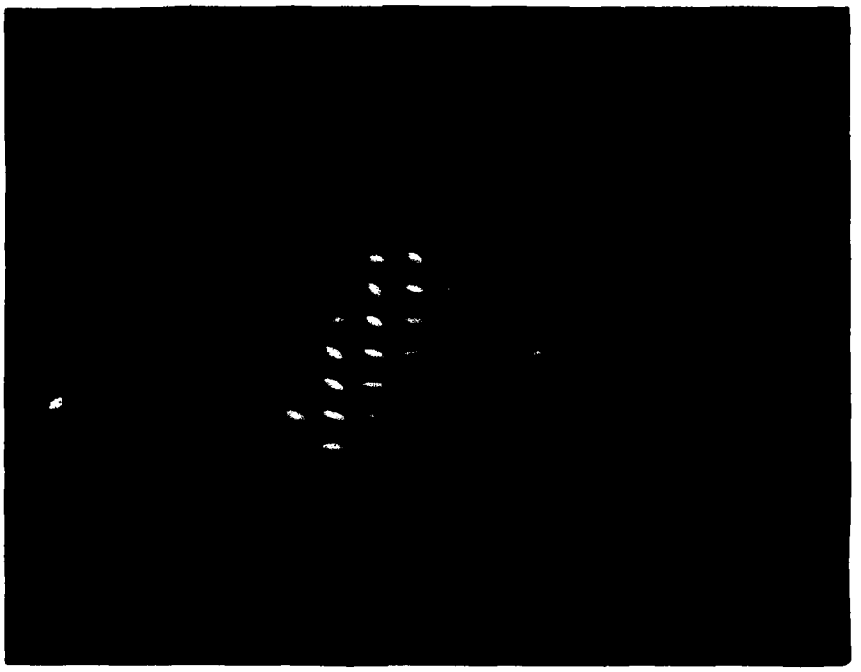


Figure 29: A Textured Airfoil

different depths.

3.5.2 Texel Density

The algorithm as described could be implemented using a simple density function for texel positions such as placing them at equal x, y screen intervals. Although this provides size and shape information, it tends to appear awkward as in Figure 26, because the density is not changing as our intuition tells us it should. To provide for the density gradient the input file is preprocessed to determine positions for texel centers. As described earlier, this involves computing areas under each pixel in the image. To approximate these values the area cosine principle is used to determine the area on the surface bounded by the projected pixel. This approximation assumes that the surface bounded by the projected pixel is a plane defined by the normal at the pixel. The equation for this area is:

$$\text{Surface Area} = \text{Pixel Area} / \cos(\text{pa})$$

where

pa = Angle between pixel plane and surface plane

thus,

$$\cos(\text{pa}) = \text{Surface Normal} \bullet \text{Screen Normal} = N_z \quad (\text{z component of surface normal})$$

$$\text{Surface Area} = 1 / N_z$$

To account for perspective this area is further modified by the z depth of the sampled point similar to the texel radius.

Once individual pixel areas are found, it is necessary to group them into cohesive sections of equal area. This implementation uses a simple approach to this grouping by building an "area binary tree" from the individual areas. At the top level is the bounding box for the

entire surface. This image is divided into two subareas by a horizontal or vertical line with approximately equal areas on either side of the line. Each of these subareas is further subdivided into two equal segments, and so forth down to some specified level. At each level n, 2^n rectangular sections are defined of approximately equal areas. The direction of division for each area is determined by the longest side of the bounding box in an attempt to keep the subareas relatively square. The area center of each rectangle can be determined by weighting the pixel locations within the rectangle by their individual areas. This area center is stored as a texel location for the density defined by this level. Figure 30 shows different levels of bounding boxes and texel centers for an arbitrary warped surface. Using these texel centers, Figure 31 shows the resulting textured image.

The z-buffer file is preprocessed to build an "area binary tree" of several levels (usually around 6). The resulting tree consists solely of texel locations for each level and becomes an input to the artificial texture rendering program. The user specifies a desired density level, and the appropriate level of the tree is loaded which specifies texel locations to be projected. Figures 32 and 33 show the difference between static and variable density. The cylinder in Figure 32 has a static density (texel centers equally spaced in x and y directions in screen space). The lower cylinder was textured using the area binary tree, and does not demonstrate the "gaps" along the edges apparent in the upper cylinder.

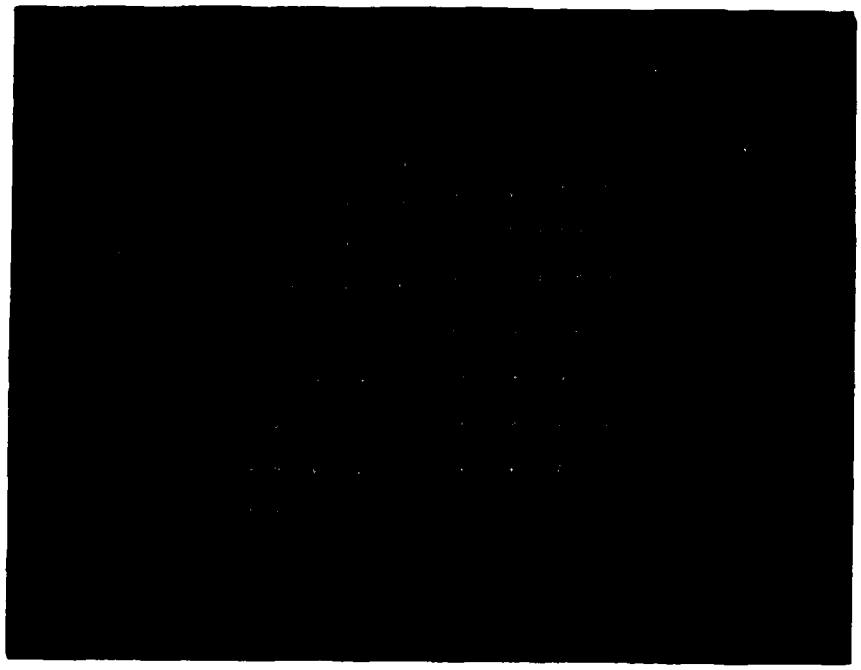


Figure 30: An Equal Area Decomposition of a Warped Surface

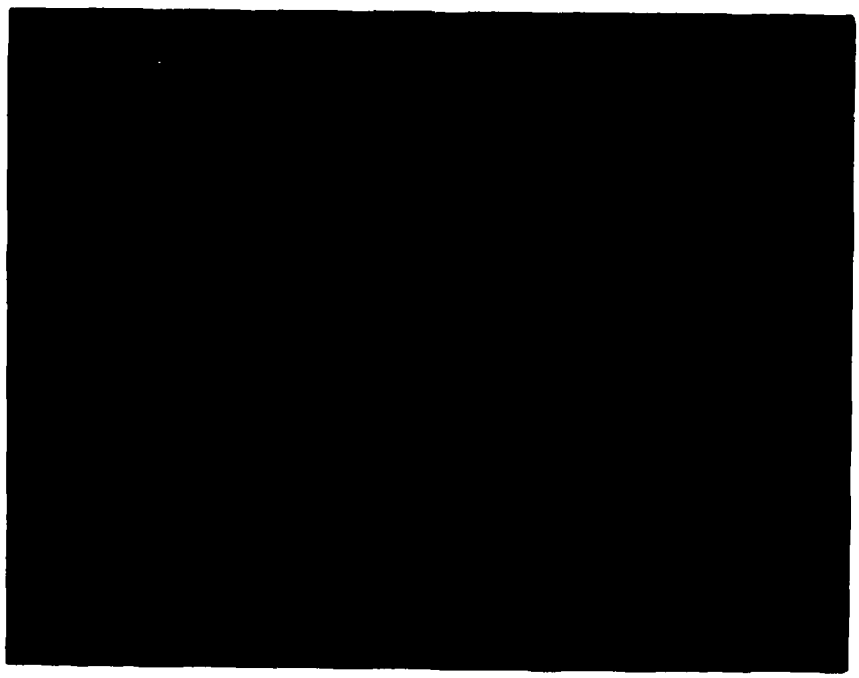


Figure 31: A Warped Surface with Texture

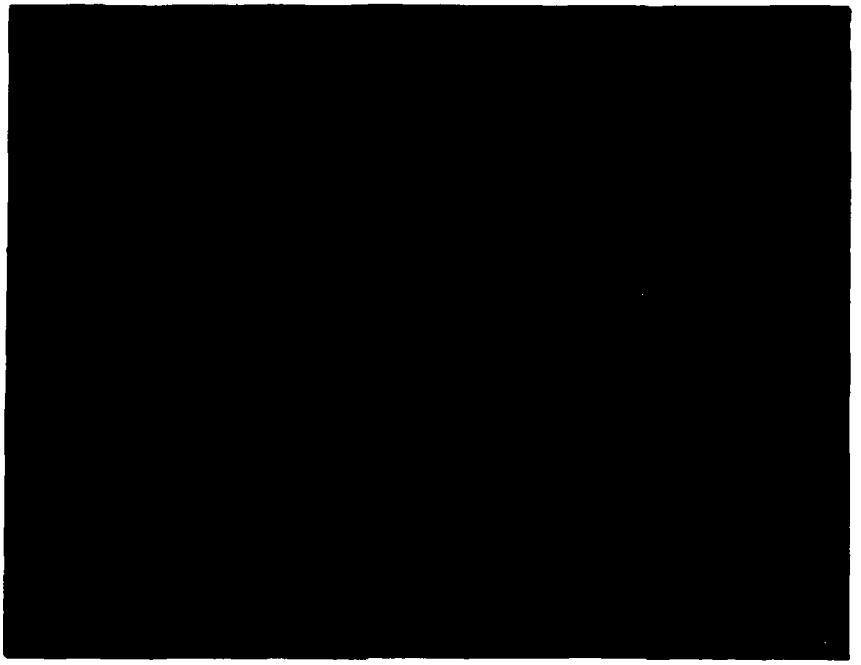


Figure 32: A Cylinder with Static Texture Density

Figure 33: A Cylinder with Variable Texture Density

3.6 Results

Artificial texturing provides simple visualization enhancement at an inexpensive computation cost. Figures 34 and 35 are examples of the information available using artificial texturing. The lower left portion of the shaded image in Figure 34 contains very little shading information to help interpret the shape of the surface in that area. It appears from the shading to be relatively flat. The artificially textured image, Figure 35, gives a visual impression of the curvature of the surface.

The information provided can be compared to the "hedgehog" technique (displaying normal vectors attached to the surface) [33]. In fact, projected polka dots can be viewed as sort of "flattened" surface normals. However, they provide a much more natural means of visualizing shape information because textures are interpreted in our everyday visual experiences. Also, additional information, such as depth, is more readily interpreted from texture gradients. One effective enhancement to artificial texturing is to combine the display of surface normals and "polka-dots." Since projected dots can represent one of two possible normals, the combined technique disambiguates the texels. Figures 36 and 37 show the effects of displayed surface normals and a combined approach on a warped surface.

Because of the number of assumptions and simplifications made, textures are sometimes generated which do not appear realistic. For example, on a surface with a sharp corner, if a texel is centered close to the edge of the corner, it will not "bend" around the corner as would seem natural. Another example is that because of the simple

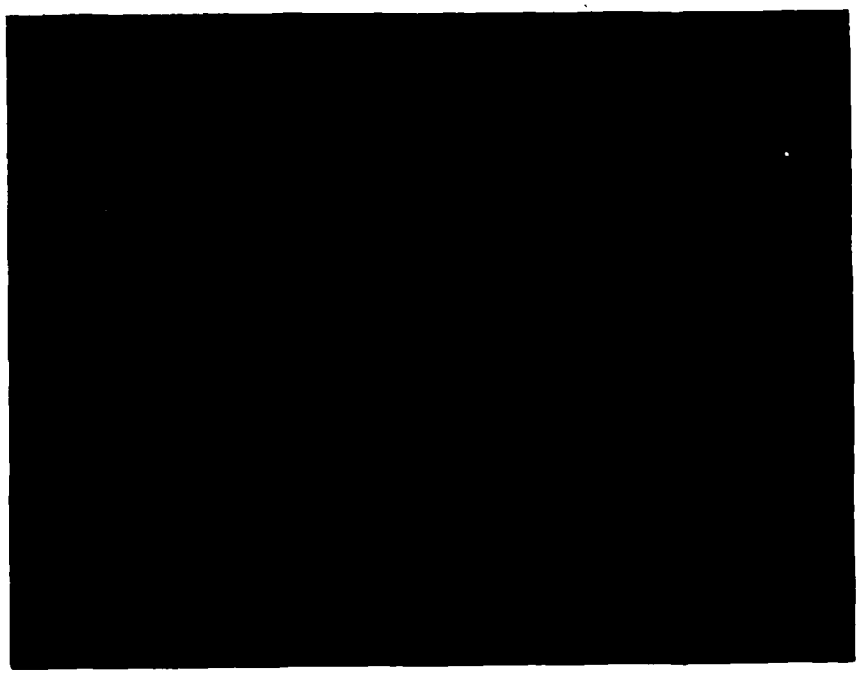


Figure 34: A Realistically Shaded Surface

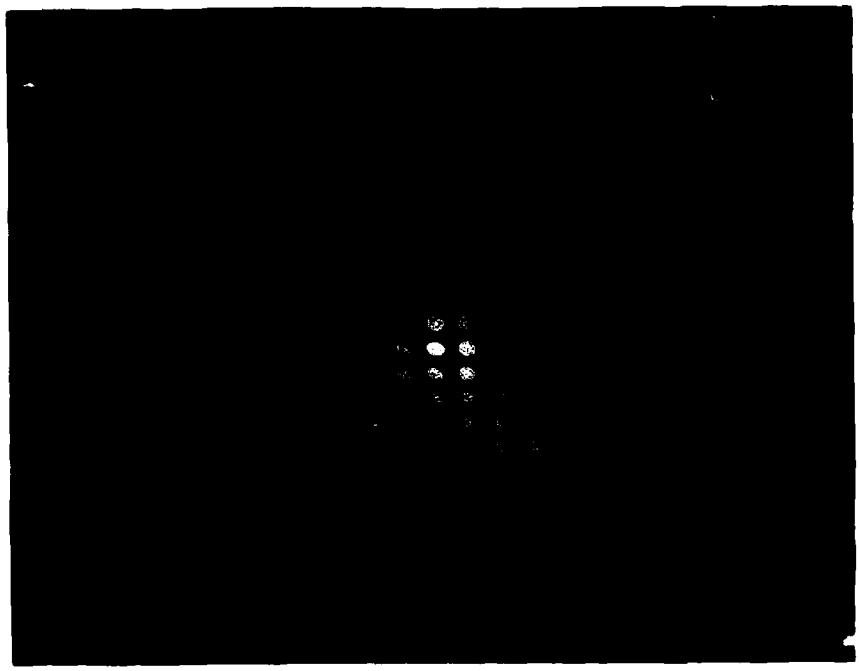


Figure 35: A Surface with Artificial Texture

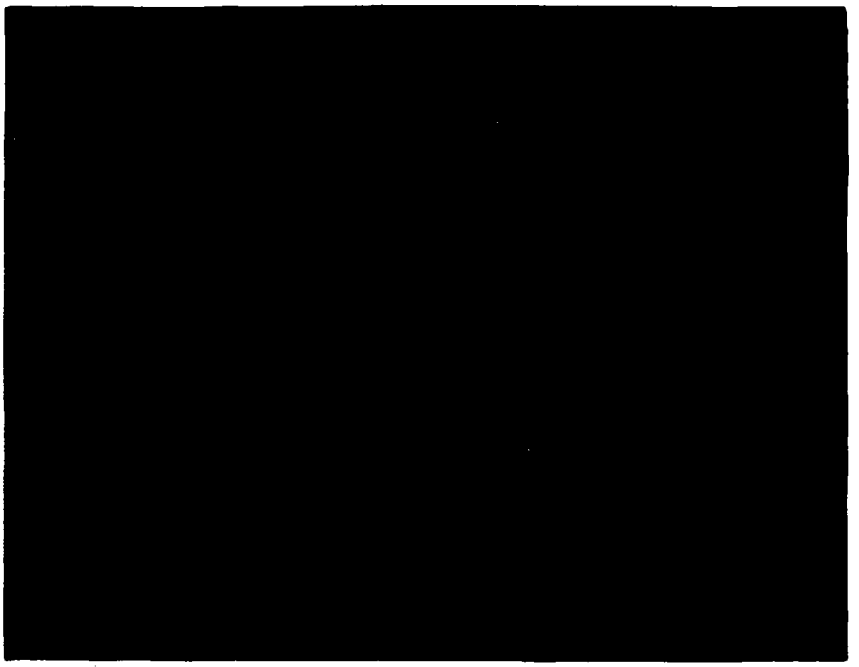


Figure 38: A Surface with Surface Normals

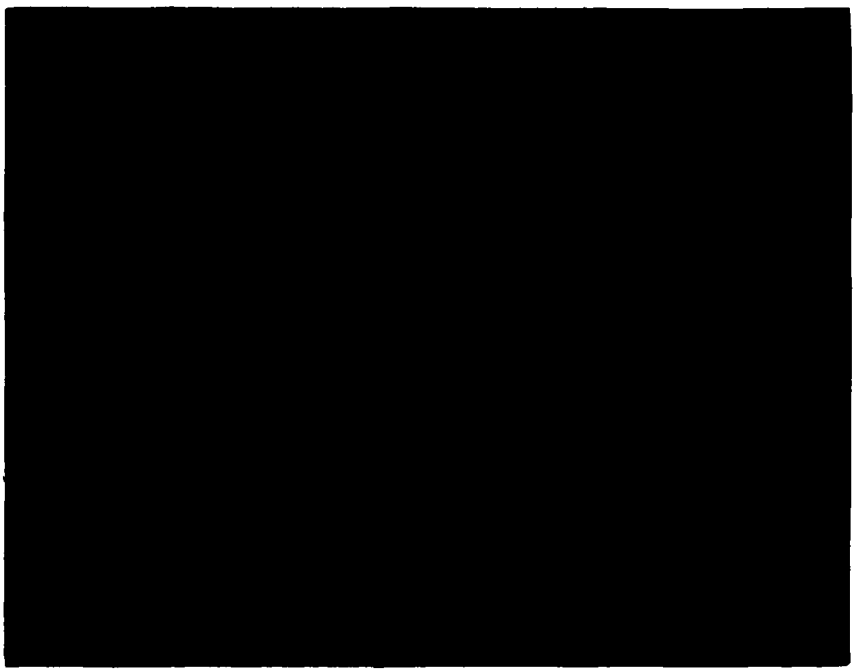


Figure 37: A Combination of Polka-Dots and Surface Normals

means of generating regions of equal surface area, densities at times will not look aligned exactly as they should. However the goal of this technique is not to simulate extreme realism, but to simulate the texture gradients which provide strong perceptive information. To this goal, texturing provides a fast, inexpensive approach to enhance surface visualization.

CHAPTER 4

GRADIENTS OF ILLUMINATION

4.1 Introduction

Shading and shadows add a strong degree of realism to images, and have been studied by artists and researchers in computer graphics. The subtle changes in illumination over the surface of an object provide a visual gradient which gives information about relative shape in much the same manner as texture gradients. Sharp discontinuities in intensities caused by shadows can distinguish the relative positions of objects in a scene in relation to the light source.

Early artists recognized shading as the primary cue for revealing the relief of an object [49]. Leonardo da Vinci distinguished two separate cues associated with shadows: attached shadows and cast shadows. Attached shadows are shadows which lie on an object caused by a surface oriented away from the light source. Cast shadows are projections of silhouette edges such as the shadow of a cube projected onto a background. By combining these two effects, paintings are created which convey a sense of depth.

Researchers in computer graphics have similarly studied the effects of shading to create the illusion of realism in raster images. Most of the research has been directed toward creating an accurate mathematical representation of the physical phenomena associated with

light reflection. Applying such a representation to every point in a picture replicates the gradients of illumination visible in natural scenes. Shadows have been approximated by several different algorithms for further realism, usually at a high computation cost.

An interactive algorithm for shading and shadows will be presented which avoids the computational complexity of typical graphic algorithms. The goal of this technique is to provide information about surface shape by imitating the perceptual aspects of shading. The background for this algorithm is provided by looking at shading in perceptual research and previous algorithms in computer graphics.

4.2 Perceptual Study of Shading

Although shading is a primary source of realism in art, its effect on visual perception has not been extensively explored [29]. Conclusions about the effects of shading have been limited to a few general observations which will be presented, but a comprehensive study of the perceptual aspects is lacking in the literature. However, there is general agreement within the research that a necessary condition for accurate perception of shape from shading is knowledge about the position of the light source in a scene.

One finding of perceptual research is that brighter objects appear closer than dimmer ones. Although shading from a light source is independent of distance to the viewer, large distances will result in distant objects appearing less sharp and less saturated. This dimming is known as aerial perspective and has been used by artists as a distance cue [49].

Lighting geometry dictates that shading is primarily a function of the position of the viewer and the relationship of the surface to the light source. Areas of equivalent shade in a scene are interpreted as having identical relationships to the light source [26]. Thus, indentations and protuberances are perceived based on relative light and dark shades in a scene. When the position of the light source is not made explicit in an image, the viewer cannot distinguish between indentations and protuberances [8].

Cast shadows present information about the relative position of objects with respect to the light source. Once again, confusion can occur if the position of the light source is not known. Additional shape information is available from cast shadows via the shape of the shadow. Because the shadow is a projection of a silhouette edge with respect to the light source, the shape of the shadow represents the shape of the silhouette. Thus, the viewer receives an additional view of the object from the point of view of the light source.

4.3 Realistic Rendering of Shade

Da Vinci's distinction between attached and cast shadows is analogous to the two distinct shading computations in computer graphics. Attached shadows are parts of a surface oriented away from the light source such that they receive only ambient light. Lighting models and smooth shading algorithms are used to determine shade intensity of a point based on the surface orientation, position of the viewer and light source, and surface reflective characteristics. A separate algorithm computes whether or not the point lies in the

projected (or cast) shadow from another part of the image. The results of these algorithms are combined to calculate the overall intensity of each point on the surface.

4.3.1 Shading Algorithms

Different lighting models have been proposed based on the physics of light reflecting off a surface [33] [19]. The simplest model is to consider only the surface orientation and the position of the light source. Lambert's Law states that the amount of diffuse reflection is proportional to the cosine of the angle between the light vector and the surface normal. Such "cosine" shading is easy to calculate as a dot product. More sophisticated models include the specular portion of reflected light and specify surface reflective properties based on surface microstructure to create images appearing metallic or ceramic [36] [5]. A more complex model attempts to imitate the reflectance and absorption properties of specified material types [12]. Special purpose lighting models have been developed to simulate such effects as clouds [7]. These models have resulted in extremely realistic images.

Application of lighting models to a scene requires normal information at every point to be shaded. The mathematical representation of the object determines whether this information can be calculated directly or requires some form of interpolation. Both shade interpolation and normal interpolation will produce smooth shaded renderings [33]. Researchers have investigated methods for speeding up these time-consuming calculations [17]. Special purpose shading algorithms have been proposed for limited model types such as objects

composed exclusively of spheres [37].

Sophisticated lighting models and smooth shading are typically used to create high quality static images. Real-time shading generally requires special purpose hardware such as that used for flight simulation [39]. A notable exception is a technique which renders normal information into the frame buffer and uses the color map to look up intensities [42]. This method allows the user to interactively move the light source and view the resulting shading changes. One requirement of this technique is that the color map lookup table must be large enough to contain a sufficient number of normals for smooth shading transitions in the image. A typical eight bit color map results in serious shading discontinuities which detract from the perceptual information gained from shading as demonstrated in Figures 38 and 39.

4.3.2 Shadow Algorithms

The generation of shadows has been recognized as a valuable addition to the realistic appearance of an image. In addition, Williams points out the spatial relation benefits and three dimensional clarification which can be gained from the inherent second point of view of the light source [51]. Two distinct approaches to shadow computation will be discussed: those based on scanline rendering algorithms and those based on the ray tracing approach.

Crow discusses different classes of shadow generation schemes based on algorithms which scanline render polygonal data [13]. Each of the techniques requires either a distinct two pass approach, or some

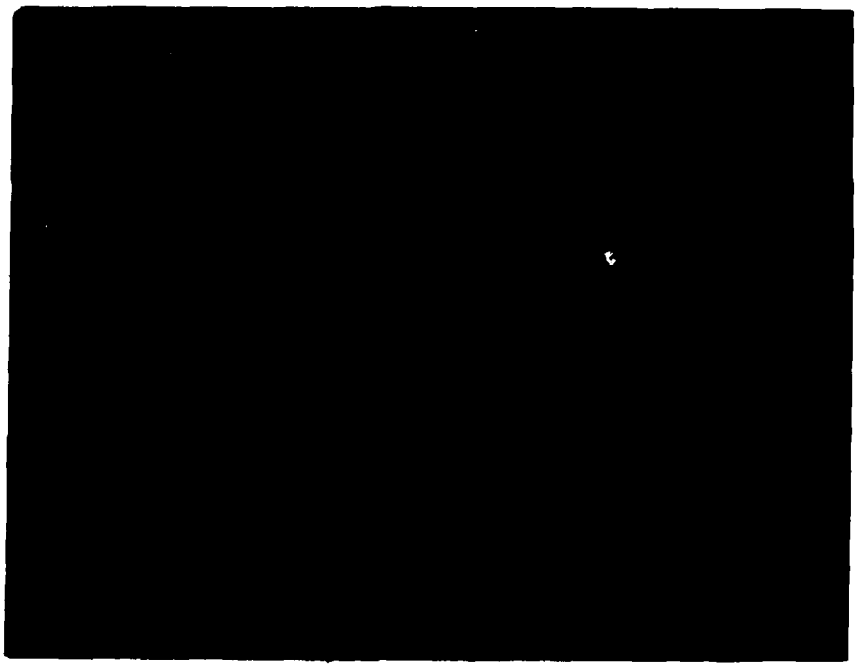


Figure 38: An Image with Color Map Normal Shading

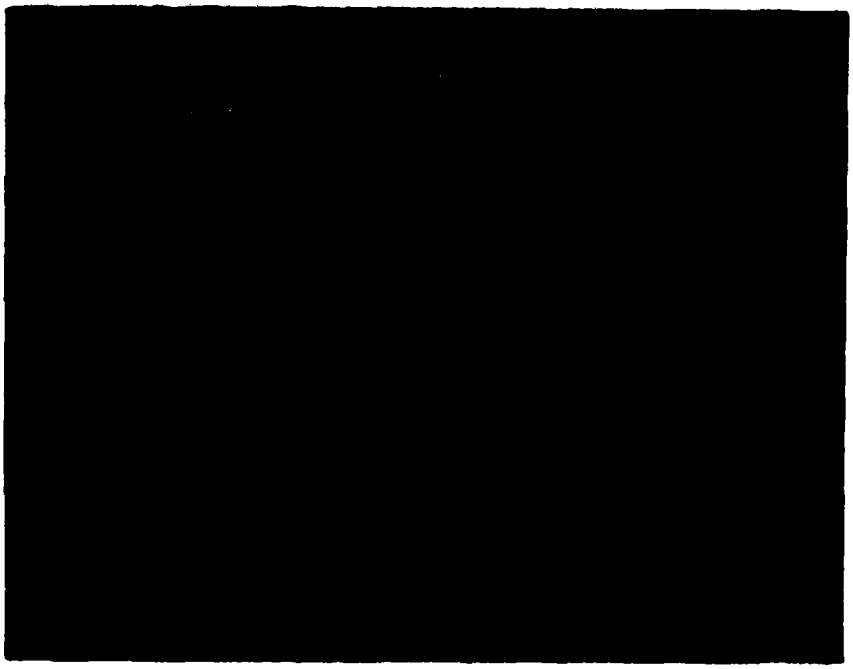


Figure 39: An Image Rendered with a Different Light Position

preprocessing of the input data to determine possible shadowing polygons. The basic outline of these algorithms is to scan the scene from the point of view of the light source, and distinguish areas in shadow either through the addition of shadow boundaries or shadow polygons. The conventional rendering algorithm then uses this information to determine where the visible surface is shadowed, and modifies the shade accordingly. Atherton generalizes this approach through the use of a general purpose clipping algorithm to produce "illuminated polygons" [2]. Williams presents a simplified approach by creating a z-buffer of the scene from the point of view of the light source [51]. Although these algorithms produce high quality results, they are too slow for an interactive implementation.

Ray tracing algorithms produce the most realistic images by simulating such effects as shadows, transparency, reflection, and refraction. To determine the shade of each display point, a ray from the eye to the display point is traced through the scene as it intersects objects and is reflected or refracted. Whitted provides a summary of the contributing shading factors and equations [50]. In order to determine shadows in a scene, it is necessary to decide for each visible point whether any object lies between the point and the light source. A ray is generated from the point to the light source and intersected with all remaining objects to ascertain this information. The impressive visual effect of ray tracing algorithms is currently offset by high computation cost which eliminates any interactive applications.

The perceptual advantages gained from shading and shadows assume

that the viewer is aware of the position of the light source. Because different light positions will produce different sources of information, a shading and shadowing program for enhanced perception ideally should allow the user to specify interactively the light source position and view the resulting image. One such approach for interactive shading has previously been discussed. An interactive shadowing algorithm will be presented which generates "visible surface shadows" to enhance the perceptual understanding of a surface shape.

4.4 Visible Surface Shadows

Generalized shadow algorithms require some form of global scene knowledge since any point may be shadowed by any other object in the scene depending upon the light source position. In order to shadow a scene interactively based on an arbitrary light vector, all objects would have to be considered as possible shadow sources. This requirement makes interactive computation impractical for scenes of even modest complexity. To make such interaction practical on a standard frame buffer configuration the described algorithm uses some limitations and simplifications concerning the light source position and the types of shadows generated.

4.4.1 Constraints

To reduce the number of possible shadowing objects for any point, restrictions are placed on the possible positions of the light vector. If the light source is assumed to be at infinity, the direction of the light vector will be the same for every point. If the light source is constrained to lie on a specified plane, objects which shadow a given

point must intersect that plane. Figure 40 illustrates such a configuration. Only objects intersecting the plane can shadow the point for any position of the light source in the plane.

Another simplification is made concerning the types of cast shadows generated. Only visible surface points will be used to determine cast shadows. This allows a standard z-buffer input file to be used with no special processing required, and simplifies the calculation. The disadvantage is that the shadows generated will not always appear realistic because hidden surfaces will not produce shadows which would normally be visible. The next section will discuss this limitation in more detail.

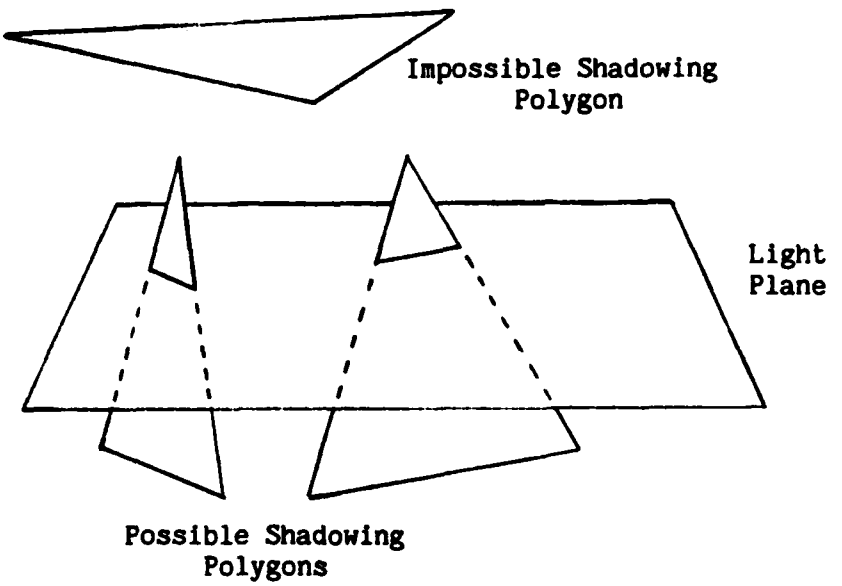


Figure 40: A Light Source Limited to a Plane

4.4.2 Implementation

Interactive shadowing does not affect the hidden surface calculation, only the resultant intensities. Thus an image is rendered only once into a z-buffer which is then used to determine shadowing information.

Visible surface shadows can be easily generated if the plane of the light source is restricted to lie in the Y-Z plane. For every scanline in the image, each point can only be shadowed by other points in the same scanline regardless of the angle of light vector within the plane. Figure 41 shows an example of points along a scanline and their associated shadow projection. At each pixel, the determination of whether the point is in a shadow is strictly a function of the depth of the pixel and the current depth of the projected shadow. In Figure 41, pixels 2, 3, and 6 are all in shadow.

To calculate quickly which pixels are in shadow, the z-buffer is scanned in the X direction of the light vector. When the light vector is exactly in the Z direction, notice that no shadows are generated. As the z-buffer is scanned, a current shadow height is kept and updated for each pixel position. If the depth of the pixel is greater than the current shadow height, the pixel lies in shadow, otherwise the pixel depth defines the new shadow height. The angle of the light vector determines the update value of the shadow height for each pixel increment, and remains the same throughout the scene.

As previously mentioned, the technique described will not always create realistic shadows. Figure 42 illustrates a situation in which a hidden surface casts a shadow which would be seen in a realistic

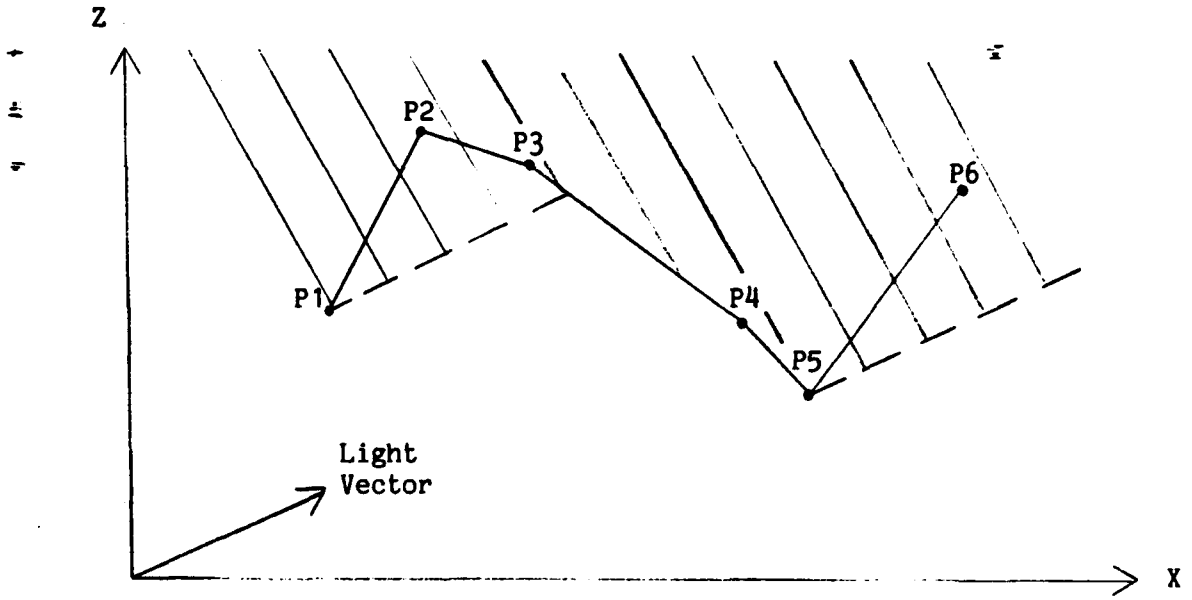


Figure 41: Shadow Projections of Scanline Points

shadowed image, but is not generated by this simplified technique. Another way of describing this limitation is that each visible surface point is rendered as if it is the tip of solid material stretching from the point backward to infinity.

4.5 Results

Figures 43 and 44 show an image with shadows generated from two different light source positions using the described algorithm. The direction of the light source is indicated in the Z-X plane. The Phong lighting model was used to calculate shading intensities.

An interesting perceptual feature associated with being able to interactively turn shadows off and on is the different perceptions the viewer has concerning the position of the light source. Figures 45 and

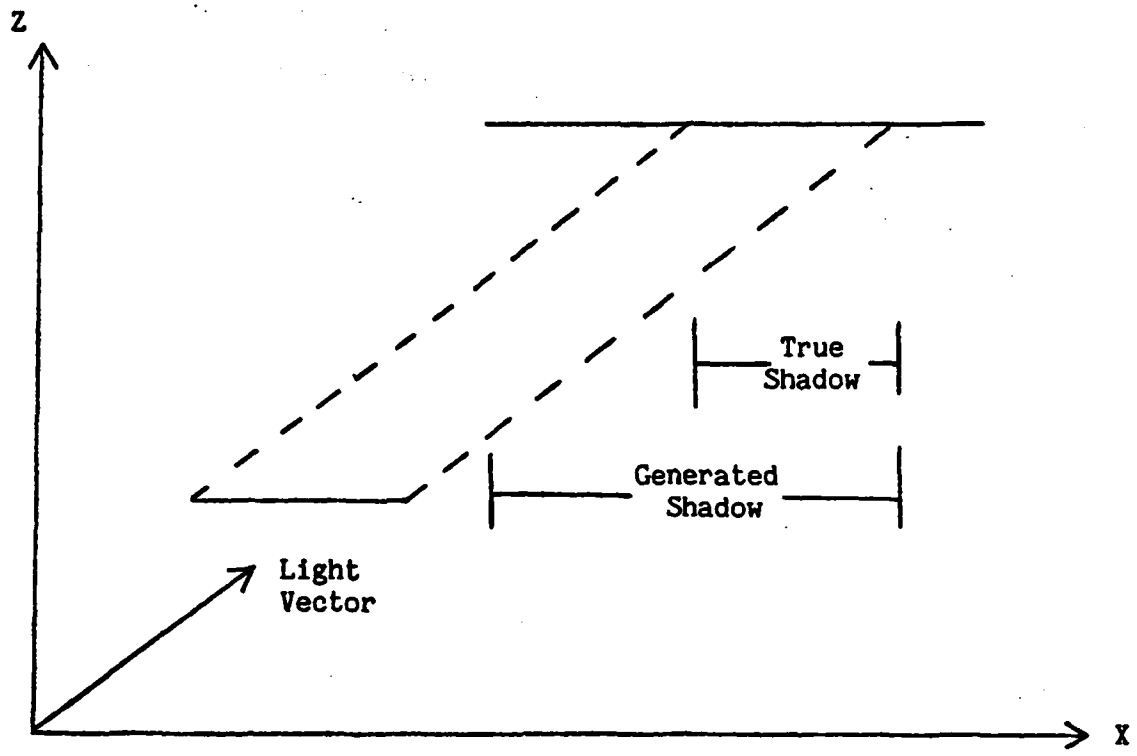


Figure 42: Generated Surface Shadow

46 show two shaded images with and without shadows. Although both shading calculations are based on light source position and surface orientation, the light source position in the image without shadows appears to be different than in the image with shadows when in fact they are the same. This indicates the importance of shadows in understanding the lighting geometry which is necessary to accurately interpret shading.

The described algorithm requires no special processing of the image, and interactively generates shadowed surfaces based on a user defined lighting direction.

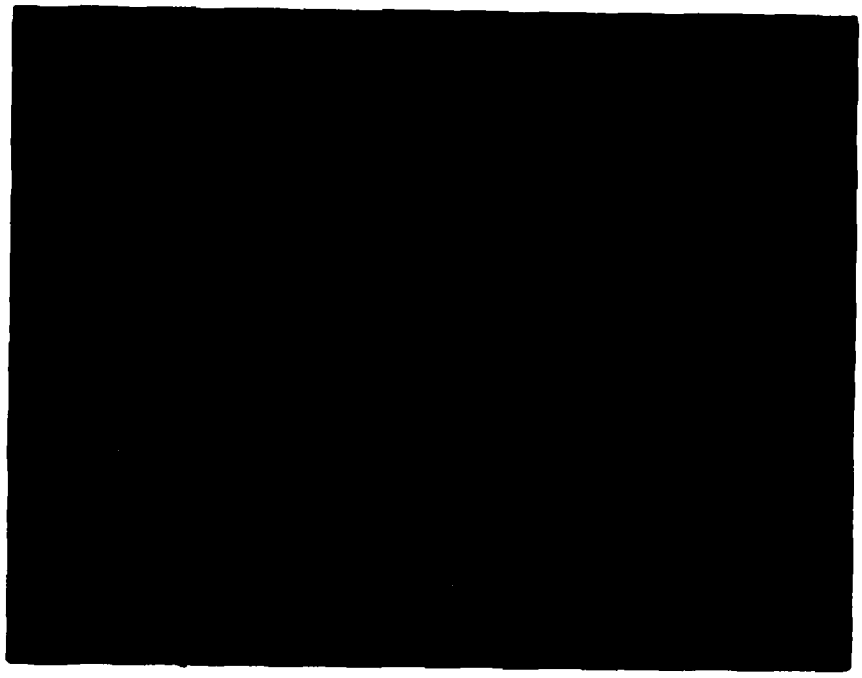


Figure 43: Shadows with Light Vector from Right

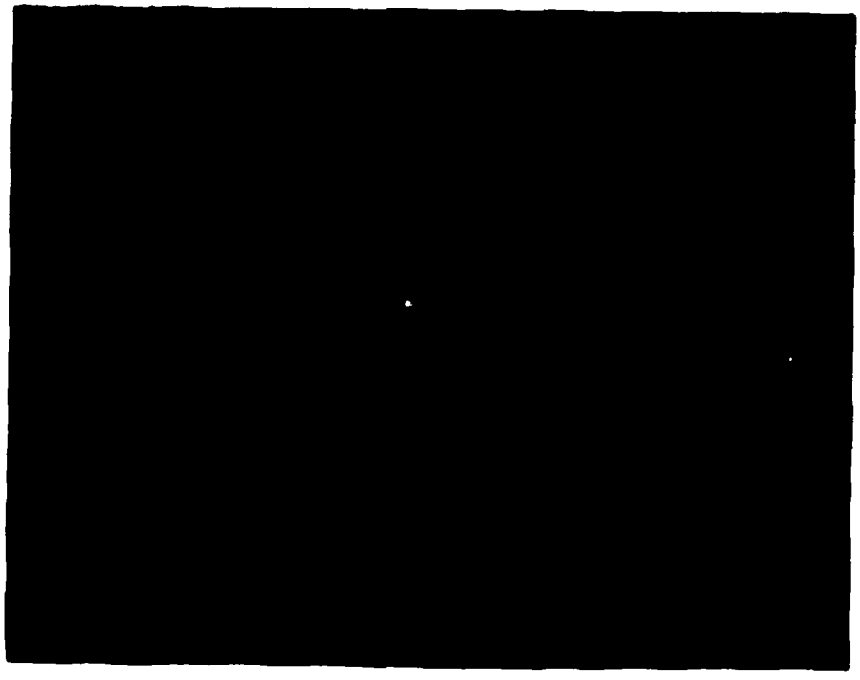


Figure 44: Shadows with Light Vector from Left

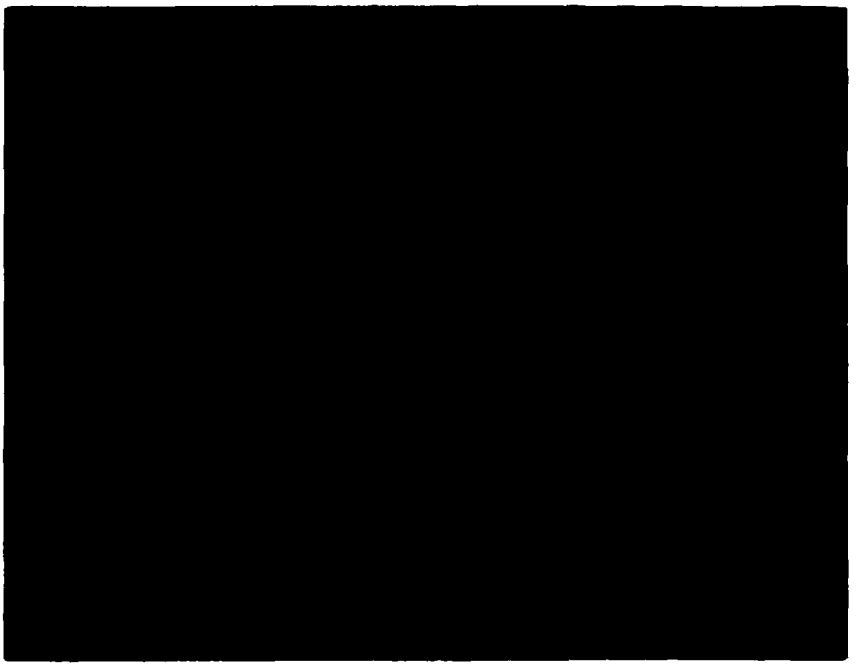


Figure 45: A Shaded Image without Shadows

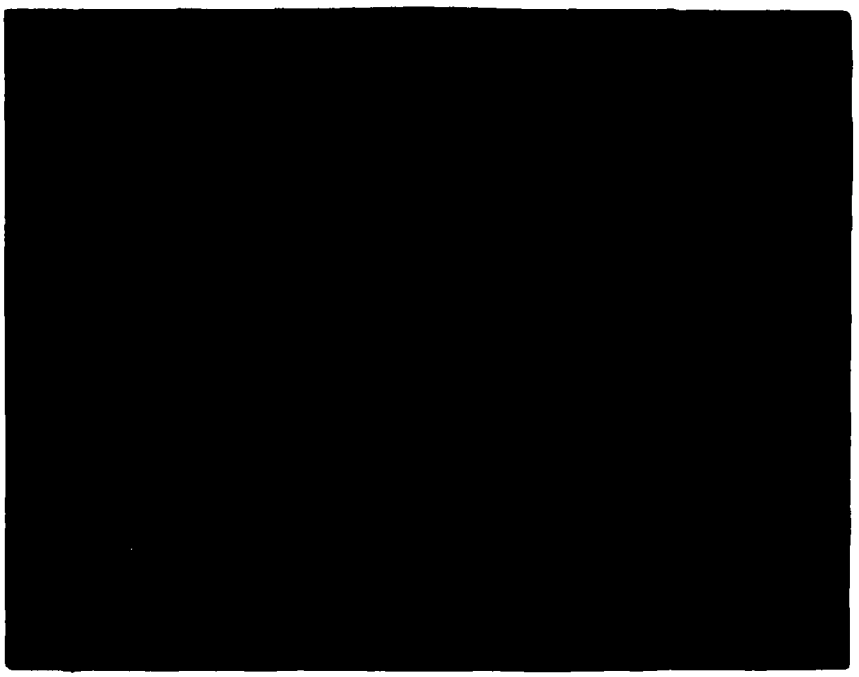


Figure 46: A Shaded Image with Shadows

CHAPTER 5

BINOCULAR STEREOPSIS

5.1 Introduction

In the visual process we automatically adjust several physiological factors which affect our perception. Focusing of the eyes causes muscular changes which can be felt and interpreted as depth indicators. Researchers have studied perception from such muscular effects as accommodation (the change in lens curvature) and convergence (bringing the eyes together to focus on a closer object). Another physical phenomenon affecting perception is a result of viewing the world through two eyes, or binocular vision.

The importance of binocular vision was recognized as early as 1500 by Leonardo da Vinci working out depth cues for realistic paintings [49]. He concluded that one could not represent such a phenomenon in a painting since binocular vision of a close small object permitted viewing the entire scene behind the object as shown in Figure 47. Charles Wheatstone performed a more detailed analysis of binocular vision in the 1800s which led to a popular parlor device called the stereoscope which viewed two images simultaneously to simulate depth. More recent examples of the use of binocular stereopsis are 3-D movies requiring filtered glasses and specially processed postcards which present different images to each eye when viewed.

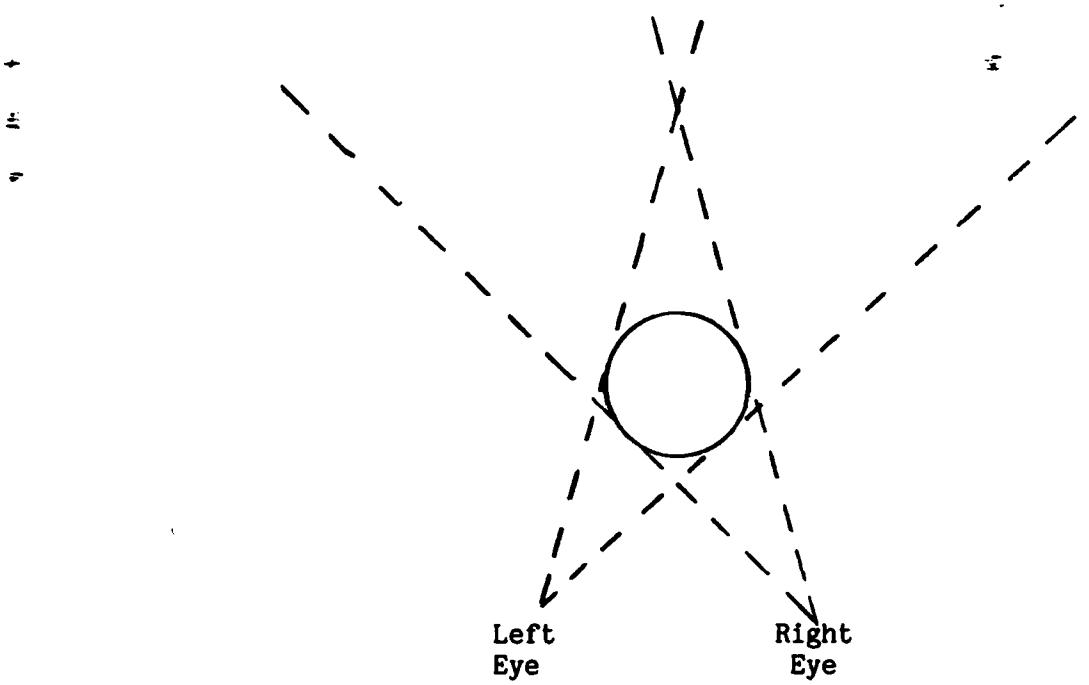


Figure 47: Da Vinci's Demonstration of Partial Occlusion

The remainder of this chapter is concerned with the perceptual aspects of binocular vision, and the manner in which it has been used in computer graphics to date. A simple algorithm is presented which generates stereo images from a single z-buffer which provide enhanced depth understanding.

5.2 Perceptual Analysis of Binocular Stereopsis

Several theories have been developed over the years about the relationship between binocular vision and depth perception [25]. Current research recognizes three primary factors which affect depth judgment based on experimental evidence: binocular disparity, interposition, and retinal correspondence.

Disparity is a horizontal shift in an image based on the perspective view from each eye. A geometric interpretation is shown in Figures 48 and 49. Given a point of fixation, a circle can be drawn connecting the point and the center of rotation of each eye as shown in Figure 48. All points on this circle, known as the Vieth-Muller circle, will project onto the same corresponding position of each eye. This phenomenon is a result of the geometric property that the angle formed by three points on a circle remains constant as the vertex of the angle is moved around the circle. Points not on the circle will project to different positions at the back of the retina based on the depth of the point and the point of fixation as shown in Figure 49. This difference is perceived as a horizontal shift between the right and left eye image and is called disparity. Note that points inside the circle have the opposite disparity of points outside the circle. By convention, inside disparity is called "crossed" and outside is "uncrossed" [26]. This horizontal shift between images seen with the left and right eye aids the viewer in interpreting the depth of points in the image.

Interposition refers to the same phenomenon pointed out by da Vinci. Parts of an image seen with one eye is occluded from the other viewpoint. The information from binocular interposition is similar to the standard occlusion depth cue in determining relative positions of surfaces.

Retinal correspondence is the registration of separate images within each eye, and is closely related to disparity. In other words, in order for the visual system to distinguish when disparity occurs, it

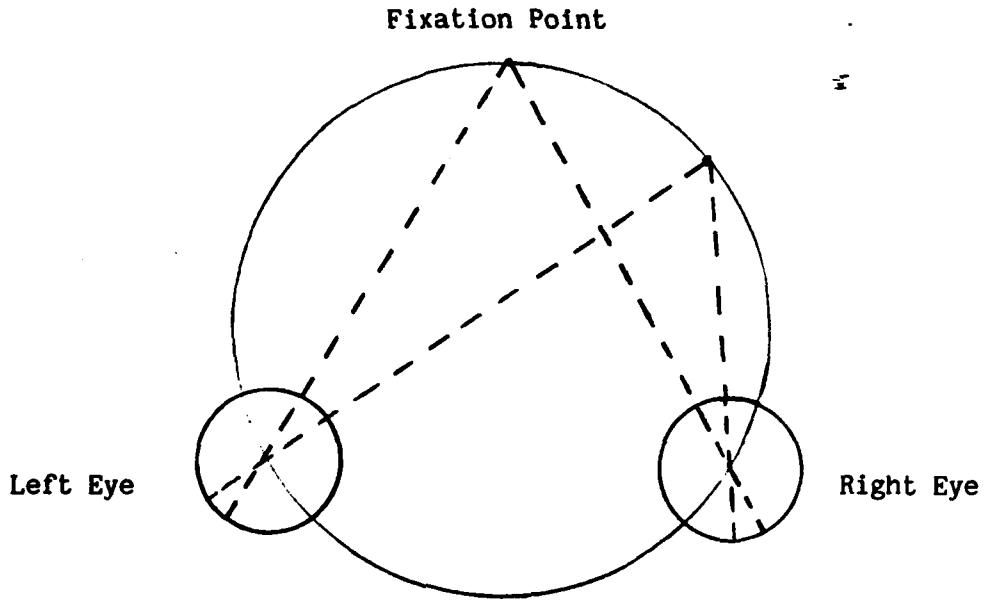


Figure 48: The Vieth-Muller Circle of Points with Equal Disparity

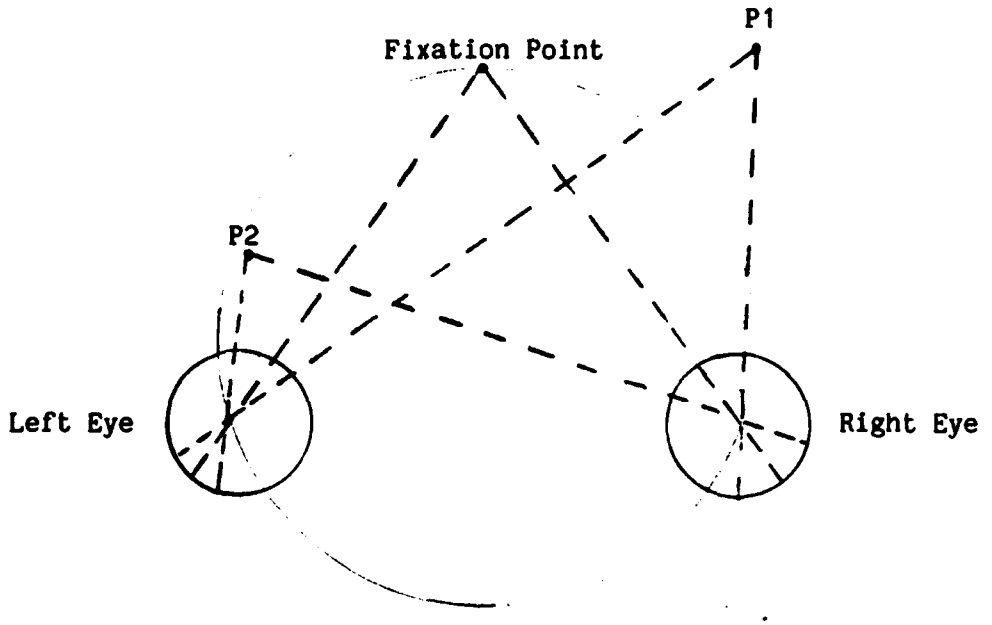


Figure 49: Points with Crossed and Uncrossed Disparity

is necessary for shifted points in the images to be perceived as the same point in the scene. When this occurs, the image appears as a single image to the viewer and the two separate images have been "fused" into one. Such fusion would not be possible looking at a homogeneous surface. Older theories of binocular vision suggested that it was the contours of an image which provided this correspondence [25]. More recent investigations show that it is not necessary to have contours, but that random texture patterns with disparity are sufficient to produce the correspondence necessary for binocular perception [30].

5.3 Binocular Images

Stereoscopic images have been generated with computer graphics to satisfy several application requirements such as molecular and atmospheric studies [15] [38]. Both line drawings and smooth shaded images have been used for binocular images with positive perceptive results.

Most of the binocular research in computer graphics has been in the area of developing viewing devices for the images once they have been generated. The stereoscopic effect is based on the fact that each eye is viewing a different version of the same scene. One way to achieve this result is to render the separate images in different colors and use filtered glasses which allow only the appropriate image to reach the corresponding eye. Problems with this approach are that each image is limited to different intensities of a single color, and ghosting occurs when the filters are not exactly correlated to the

display. More sophisticated equipment which allow multiple colors and avoid ghosting have been developed. This equipment includes electro-optic glasses which flip between images, and hoods which present half a display to each eye [15].

The general technique of image generation has been to simply render two separate views of the same scene; one from the position of each eye. When viewed stereoscopically, such images contain the disparity and interposition necessary to provide binocular depth information. If the scene is complex, retinal correspondence is provided by the details of the scene. Two problems arise from this approach: lack of flexibility to change viewing parameters and insufficient retinal correspondence in scenes lacking in detail. Since disparity depends on perspective, it may be desirable to interactively change the distance between the viewer's eyes to highlight details of a scene, or account for the different screen widths of different monitors. The traditional approach would require two complete scene renderings to change such parameters. As an example of the second problem, consider the surface shown earlier in Figures 4. When viewed stereoscopically, the only retinal correspondence occurs along the edges of the surface which in this example are flat. The smooth shading in the interior of the surface does not provide any correspondence between the two images, and when viewed in stereo the surface appears flat. Although this is a contrived example, it demonstrates a binocular limitation when surface depths change independent of recognizable contours.

A simple binocular rendering algorithm will be presented which

provides stereoscopic depth perception while avoiding the problems associated with normal stereo image generation. For simplicity, the viewing technique used is filtered glasses, but the algorithm is valid for any stereo viewing technology.

5.4 Stereo Pairs from a Single Image

The primary goal of this algorithm is to provide interactive surface shape analysis through binocular stereopsis. The user can select viewing parameters and quickly generate a stereo pair for binocular viewing. To decrease the image generation time, a simplification is made to only generate the perceptive features of disparity and retinal correspondence. Although interposition can provide some additional information about relative surface depths, it is not necessary for depth analysis through binocular fusion [30].

5.4.1 Simulating Disparity

Disparity only occurs in a horizontal direction. This means that for a given computer generated image, the only difference between the right and left stereo pair occurs along a scan line. The amount of horizontal shift is based on the viewing parameters and the depth of the point. Figure 50 illustrates the perspective change of a given point for each eye. The equations for the disparity are:

From similar triangles:

$$\frac{X_p - E_x}{X_r - E_x} = \frac{Z_p + E_z}{E_z}$$

Solving for X_r:

$$X_r = \frac{(X_p - E_x)}{(Z_p / E_z + 1)} + E_x$$

and similarly for X_l.

One means of utilizing these equations to rapidly generate stereo

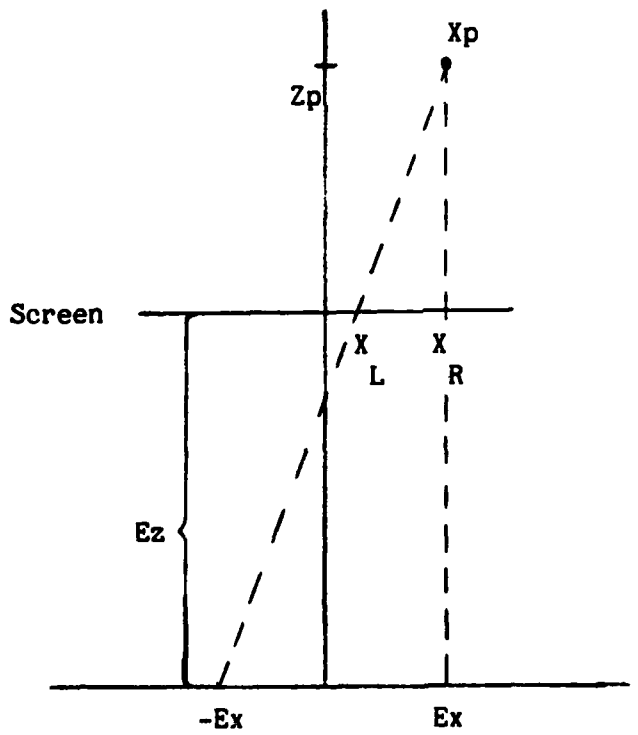


Figure 50: The Geometry of Binocular Perspective

views is to begin with a rendered z-buffer image. At each pixel the depth of the point is kept along with the appropriate shading intensity. To generate the stereo pair, each pixel is shifted according to the calculated disparity based on depth. Right and left images are rendered into the appropriate color plane according to the color filters being used for binocular viewing. Figures 51 and 52 are examples of stereo images drawn with this technique. Vertical perspective is accomplished by originally rendering the image with perspective, and modifying the disparity equations to account for the "normal" horizontal perspective. The modified equations are:

$$X_r = X_p + E_x * \frac{1}{1 + E_z / Z_p}$$

$$X_l = X_p - E_x * \frac{1}{1 + E_z / Z_p}$$

Because the input image consists of point sampled data of the scene, care must be taken to avoid "stretch marks" which occur when successive pixels are shifted by different disparities. One solution would be to use input spans rather than points, shift the endpoints of each span, and resample the span in the X direction. A simpler approximation is to detect when "stretch marks" occur, and average intensities to fill in between pixels. The images shown in this section were generated using this approximation.

Figure 51: A Stereo Pair of a Plane with Surrounding Box

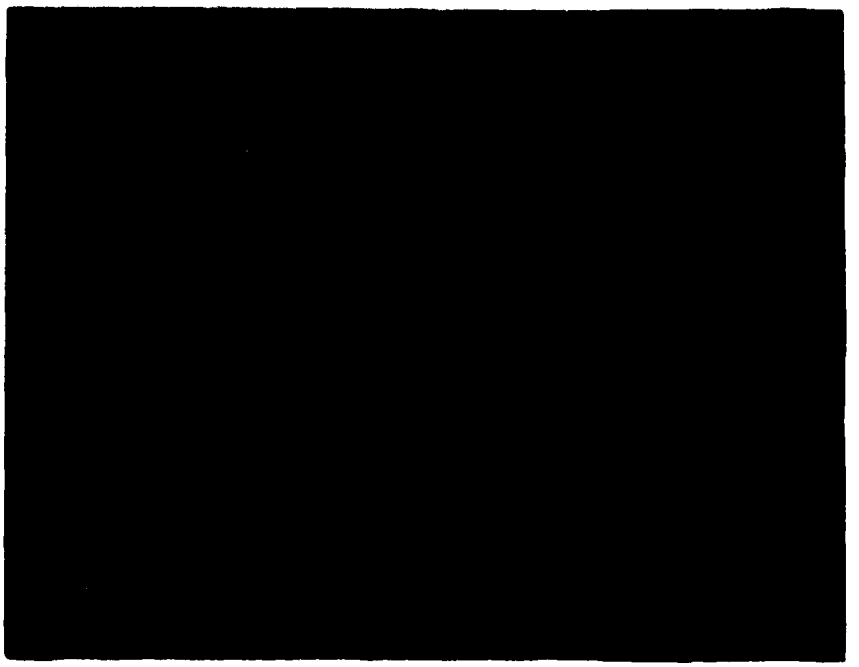


Figure 52: A Stereo Pair from a Single Image of a Helicopter

5.4.2 Providing Retinal Correspondence

Retinal correspondence is necessary to register the different images within each eye so that fusion can occur. Figure 53 is a stereo pair of a previously discussed image which does not contain such correspondence. The disparity seen along the edge of the surface is constant providing no binocular depth. When viewed stereoscopically, the image appears as a flat plane floating in space. To provide the necessary registration, some detail is required in the middle of the surface.

Using the described algorithm, a simple and fast method is available to provide surface detail. As the input scene is processed, intensity values are increased prior to shifting for each pixel at some user specified X, Y increment. Thus, a pattern of dots appears on the output stereo pair which correspond to the same points on the input scene. Figure 54 shows the computed detail for the same image previously lacking correspondence. The density of the dots is chosen with an Y increment of one creating detail lines. When overlayed onto the shaded stereo image and viewed stereoscopically, the correspondence lines on the image provide the depth lacking in the previous image.

5.4.3 Depth Filtering

The quantization associated with shifting pixels by integer jumps creates depth discontinuities when viewed in stereo. This became especially apparent when using detail to provide retinal correspondence. Depth "ridges" would appear along equal depth contours. To reduce this quantization, a simple filtering technique

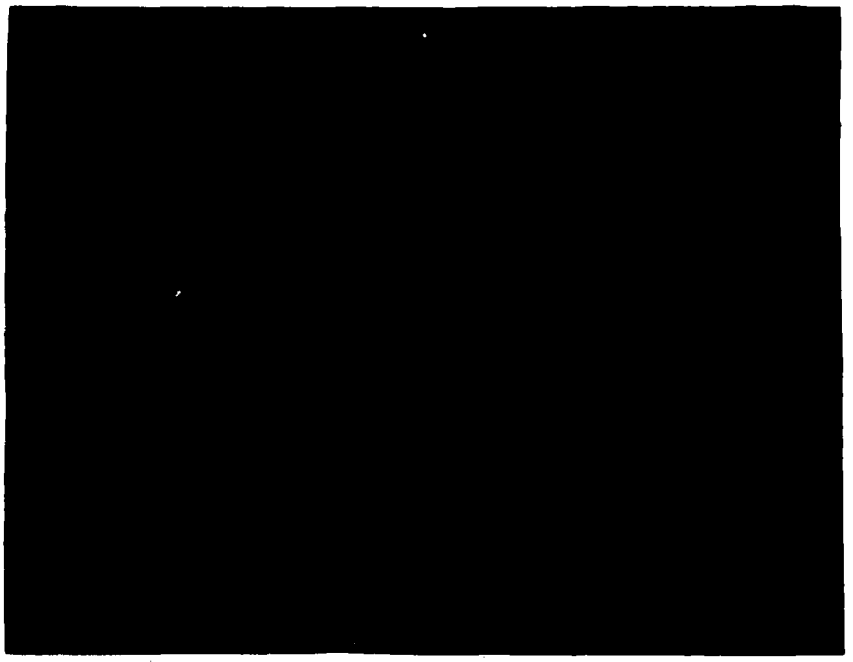


Figure 53: A Stereo Pair with No Retinal Correspondence

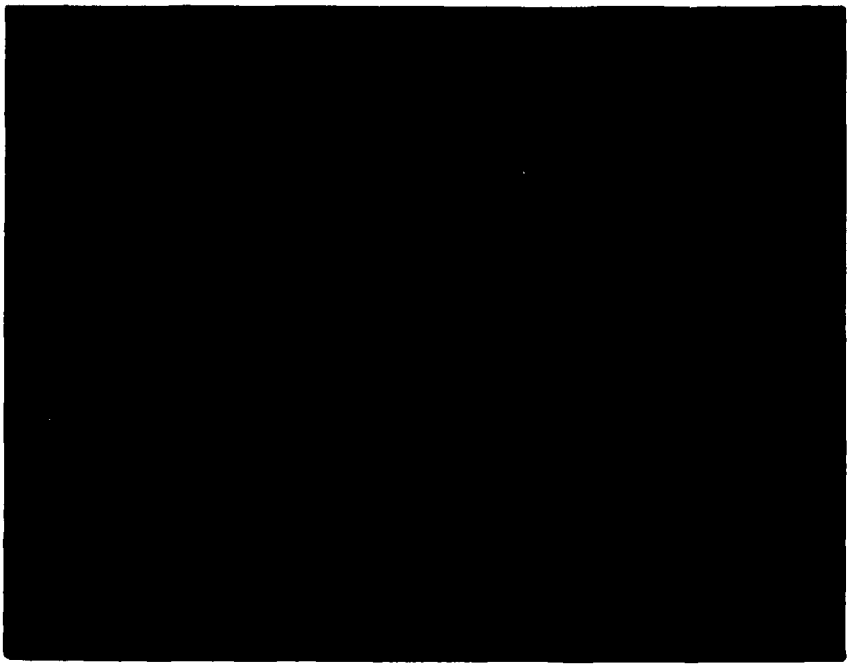
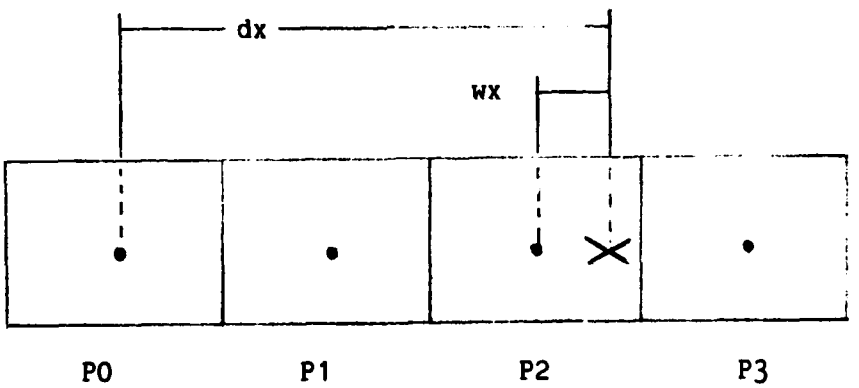


Figure 54: A Stereo Pair with Correspondence Detail of Lines

was implemented. Rather than shift detail pixels an integral distance, each detail affects two adjacent pixels by a weighted intensity amount as shown in Figure 55. Figures 56 and 57 show a closeup of stereo lines before and after filtering for depth quantization. When viewed stereoscopically, the filtered image shows no discontinuities in depth.

5.5 Results

Stereo pair generation from a single image z-buffer provides a fast means of binocular image rendering. The effects of disparity and retinal correspondence are simulated to create a strong visual depth effect when viewed stereoscopically. To avoid depth quantization artifacts, a simple filtering mechanism is applied to registration



dx = Calculated Disparity
wx = dx - (int) dx
In = Intensity at Pixel n
I2 = (1 - wx) * I0
I3 = wx * I0

Figure 55: Filtering Depth Quantization

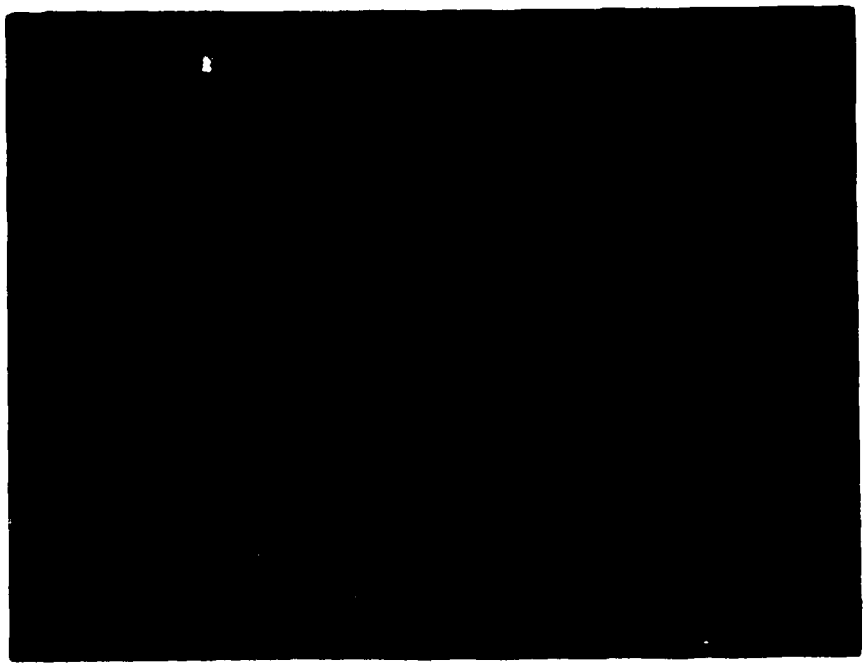


Figure 56: Stereo Lines Before Filtering

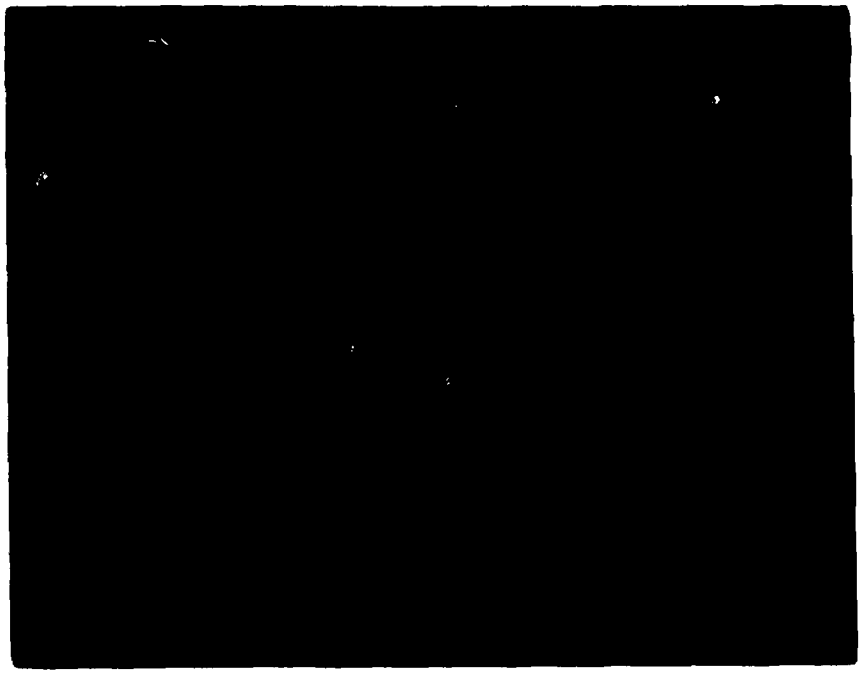


Figure 57: Stereo Lines After Filtering

points on the surface of interest. Images generated with this technique appear continuous in depth.

Interposition information has been sacrificed in this technique as a trade-off for speed. This simplification does not affect the perception of depth, but is noticeable when viewing silhouette edges of an object. For example, a sphere viewed binocularly would present a different part of the surface to one eye than the other, and give the effect of seeing around the object. The described algorithm would not have the same effect as only the surface visible from a single view is used to create both stereo images.

Although viewing parameters can be interactively modified, one must be aware of limitations inherent in using a point sampled input. Perspective changes can only be effected in the horizontal direction; vertical perspective is assumed to have been accounted for previously. Thus factors such as distance between eyes and screen width can correctly be changed, but other factors such as distance of the viewer and screen height will cause a distortion in the vertical perspective of the final image.

These limitations preclude the generation of totally accurate stereo images. However, as a tool for visual perception, this algorithm quickly provides depth information in a natural viewing fashion.

CHAPTER 6

SHAPE PERCEPTION THROUGH MOTION

6.1 Introduction

The perceptive stimulations presented thus far in this research can be classified as static stimulations. That is, they deal with enhancing a single image to create another static image which hopefully provides an increased perception of three dimensions. The interaction just serves as a tool for exploring the shape of the static image. Another class of perceptual cues which has been studied deals with the perception of depth through motion [22, 8, 26]. As an observer moves with respect to an object, or vice-versa, several factors such as occlusion, relative speeds, and changing sizes provide information about the shape of the object being viewed. Such dynamic stimulations are an inherent part of our everyday visual experiences.

Interactive movement in computer graphics has generally been limited to line drawing displays or special purpose displays for shaded images. Simulated movement on raster displays has traditionally been performed via frame by frame image generation and playback. This method is time consuming and lacks the advantage of interactive control of the movement to assist perception. The success of special purpose hardware is offset by the enormous associated cost.

This section will describe a simple algorithm for simulating

limited movement of a surface on a raster display with a color map. The goal is not to provide high quality animation, but rather to provide dynamic cues to aid the visualization of the surface.

6.2 Perceptual Analysis of Movement

To understand the perceptual effects of an object moving in space, researchers have concentrated on the motions of rotation and translation along the three cartesian axes [8]. As an object moves in these directions its changing shape and velocities provide considerable shape information.

One of the primary sources of shape information gained from movement is the ability to see the object from different viewpoints. These viewpoints allow hidden surfaces to be seen in shaded images. The occlusion of parts of the surface by other parts provides information about relative positioning.

Information provided by translation can be divided into radial movement and motion parallax [26]. Radial movement refers to translation along the Z axis. The size of the image appears larger or smaller depending on depth according to the rules of perspective projection. The resulting stimulus is perceived as an object of constant size moving in depth, but no information is gained about the relative depths within the image. Translations in the X and Y directions provide information known as motion parallax. As an object moves, the velocity will appear greater for the closer portions of the image. Once again, it is the effect of perspective which is being observed. Figure 58 illustrates the effect of motion parallax.

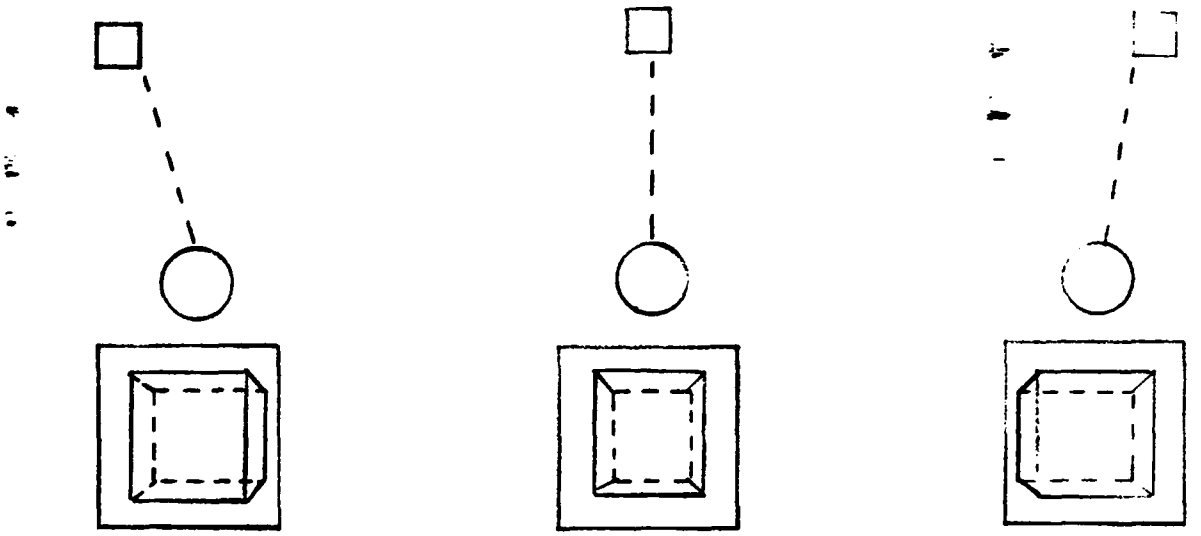


Figure 58: The Effect of Motion Parallax

Rotation about the Z axis (perpendicular to screen) provides no additional information about the relative depths on the surface beyond the information provided by a standard perspective projection [8]. Rotations about the X and Y axes exhibit the same effects in different directions. A point rotating about the Y axis changes velocity in the horizontal direction depending on the position of the point along the path of rotation. As shown in Figure 59, the apparent velocity of the projected point will be greatest when travelling perpendicular to the viewer. A vertical change in position will occur due to perspective and the relative depth of the point along the rotated path. The radius of the path depends on the distance from the axis of rotation, and thus provides a good measure of relative positions of points on a surface.

Braunstein has summarized the effects of these six motions using

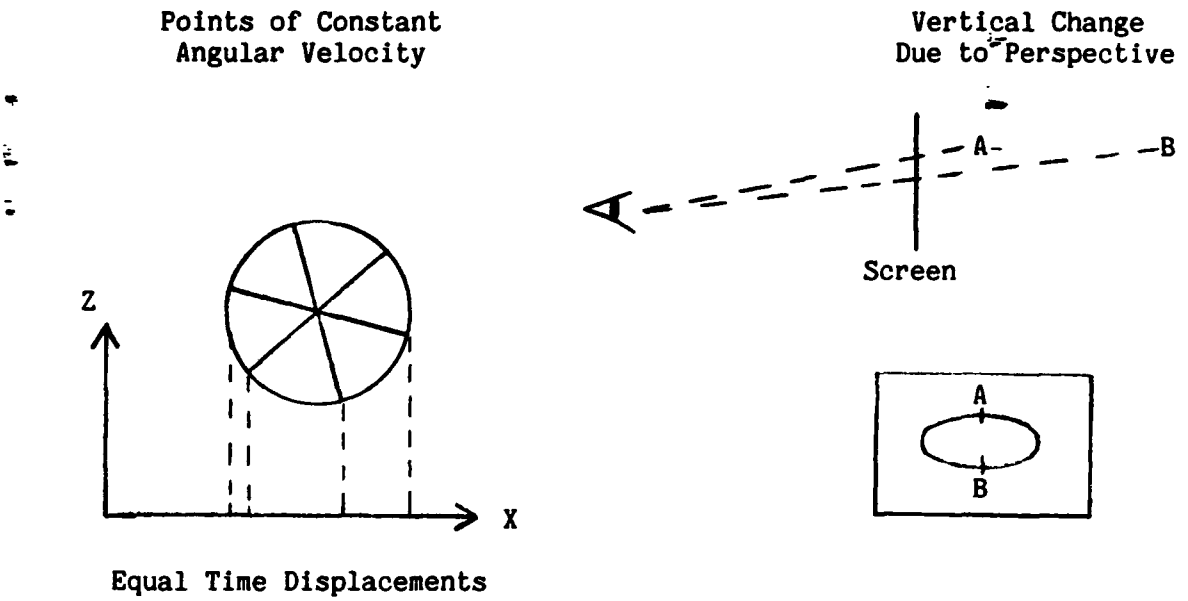


Figure 59: The Effect of Rotation

both perspective and parallel projections as shown in Table 1 [8]. He separates the available information into whether the direction of movement is apparent and whether dimensionality or relative depth information is obtained.

6.3 Use of Motion in Computer Graphics

High quality real-time animation has been an aspiration of computer graphics researchers for many years. Because of the computational intensity of high quality rendering, the only real success has been through the use of special purpose hardware, and then only for specific applications. Research efforts toward general purpose animation can generally be broken into the areas of hardware advances, fast frame-by-frame generation and playback, and algorithmic

Table 1: Information Available in Rigid Transformations

		Projection	Direction	Dimensionality
Rotations	X	Polar	yes	yes
		Parallel	no	yes
	Y	Polar	yes	yes
		Parallel	no	yes
	Z	Polar	yes	no
		Parallel	yes	no
Translations	X	Polar	yes	yes
		Parallel	yes	no
	Y	Polar	yes	yes
		Parallel	yes	no
	Z	Polar	yes	yes
		Parallel	no	no

approaches to simple motion.

The most successful efforts in real-time animation have been in the area of flight simulation [39]. The use of special-purpose hardware provided the means to produce impressive results with the simulation of the space shuttle [48], and more recently, with instruments like the Evans and Sutherland CT-5 image generator. Such systems perform much of the rendering procedure in hardware for real-time image generation. They are constrained, however, to scenes that do not require certain computationally prohibitive visual effects associated with high quality realistic images. For example, no attempts have been made to replicate such realistic effects as refraction or reflection. Similar efforts have been directed toward more general animation of limited polygonal scenes [1].

High quality animation requires individual frame rendering which must then be played back. Research in this area has been directed

toward rapid individual frame generation and quick play back. Speeding up the rendering process is performed by attempting to take advantage of certain coherences which exist within each frame as well as between frames [44, 21, 28]. Another technique to decrease rendering time is selective screen updating, leaving an unchanged background in the frame buffer and only modifying that portion of the image which requires updating between frames [43]. Research at Ohio State has taken advantage of run length encoding to facilitate fast playback of generated animation sequences [14].

A third area of research has been directed toward taking advantage of existing hardware to produce simple but effective animation. Color map animation is an example of using raster hardware to produce continuous motion [41]. The concept of storing separate images in separate bit planes of a frame buffer is also a means of storing and viewing separate frames in real-time [19]. Both of these techniques are severely limited in terms of animation quality, but afford a simple and inexpensive approach to motion simulation.

6.4 Multiple Frame Generation and Viewing

A technique for interactive surface motion is presented for the purpose of increasing three dimensional understanding. By taking advantage of experimental results in perception and providing only limited degrees of motion, real-time interaction is achieved with no requirement for special purpose hardware. The goal of this approach is to provide the visual benefits of interactively moving a shaded surface without the time costs associated with a high quality frame by frame

rendering. Two aspects of this technique will be discussed: the viewing algorithm and the multiple frame generation.

6.4.1 Multiple Frame Viewing

Perceptual research has shown that rotations and translations about the X and Y axes are sufficient to provide cues about the dimensionality associated with a surface. As discussed earlier, transformations about the Z axis are not necessarily helpful, and thus will not be examined further. Both rotation and translation will be simulated by generating multiple frames of the image transformed through a small increment of the total desired motion. Motion is created by interactively displaying separate frames of the generated sequence.

Interactive playback capability is achieved using a variation of the separate bit plane concept presented in the previous section. Because smooth shading requires a full intensity range, only three separate planes are used; the red, green, and blue values for each pixel contain intensities for separate shaded images. The range of motion which can be simulated smoothly depends on the number of frames available in the sequence. In order to allow rapid access to several frames for controllable movement, a diminished resolution is utilized to keep either 4 or 16 images in each color plane simultaneously. Thus either 12 or 48 frames of a surface reside in the frame buffer memory at once, as shown in Figures 60 and 61. The number of frames used to simulate motion is a tradeoff between image resolution and the range of smooth motion which is represented.

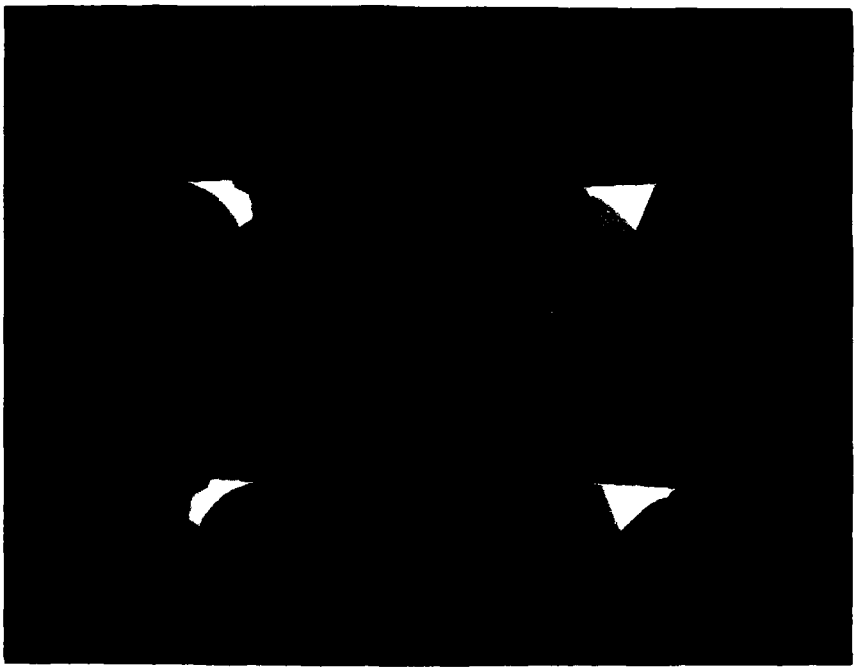


Figure 60: 12 Frames of Motion

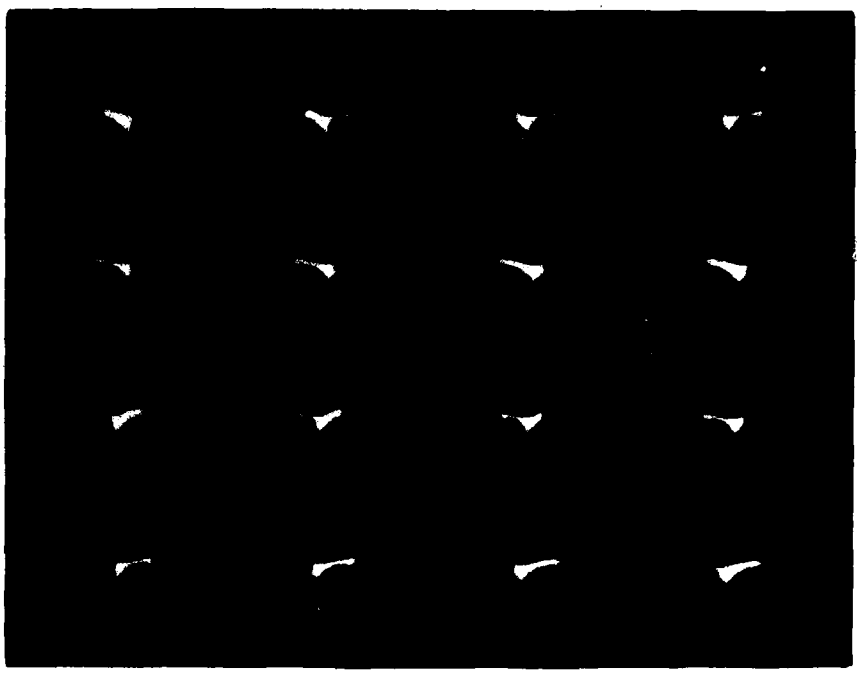


Figure 61: 48 Frames of Motion

Interactive movement is performed using the hardware features of zoom, pan, and color plane selection to display individual frames. The zoom feature allows each frame to fill the entire screen with a coarser image. Which frame to display is determined from an input device such as a knob or tablet. Because individual frames can be rapidly displayed, smooth interactive movement results. Figures 62 and 63 show the resolution of individual frames for both 48 and 12 frame sequences. Color map lookup tables are used to provide color during rotation and translation, although the entire surface is necessarily limited to shades of the same color.

6.4.2 Multiple Frame Generation

Real-time motion through multiple frame viewing requires the sequence of frames to be individually rendered prior to viewing. If the images are rendered with high quality shading and a sophisticated lighting model, such image generation can be quite time-consuming. Since the separate images are all of the same surface, some simplifications can be made to reduce image generation time for both translation and rotation sequences. It is assumed for these simplifications that the image consists of polygons to be rendered.

Perceptual benefits arising from image translation must include perspective changes from image to image as discussed in the perception section. Without this requirement, the image would only be rendered once with the individual pixel values translated in the appropriate direction to create different frames. However, if the sequence only allows translation along one axis, then the positional changes due to

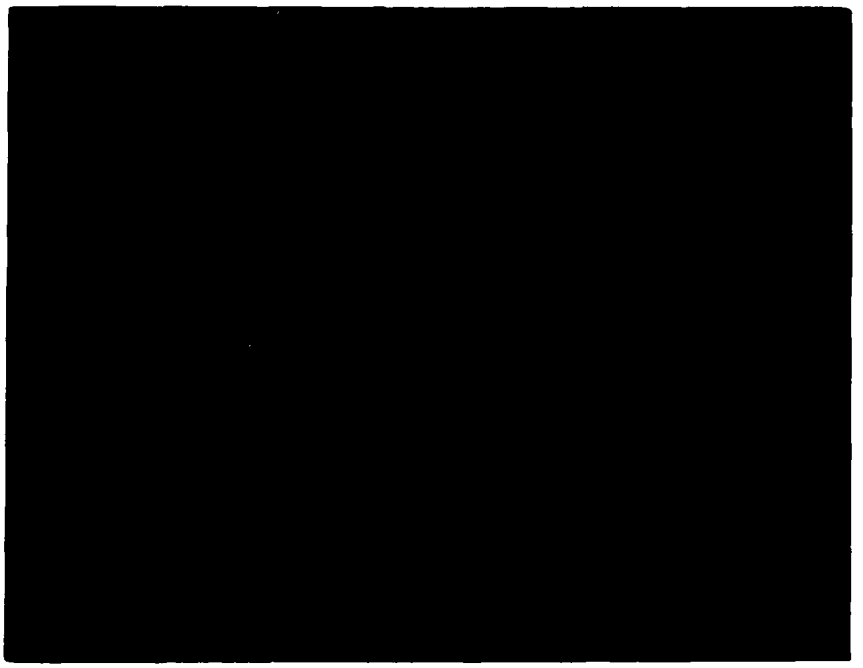


Figure 82: A Single Frame from 48 Sequence

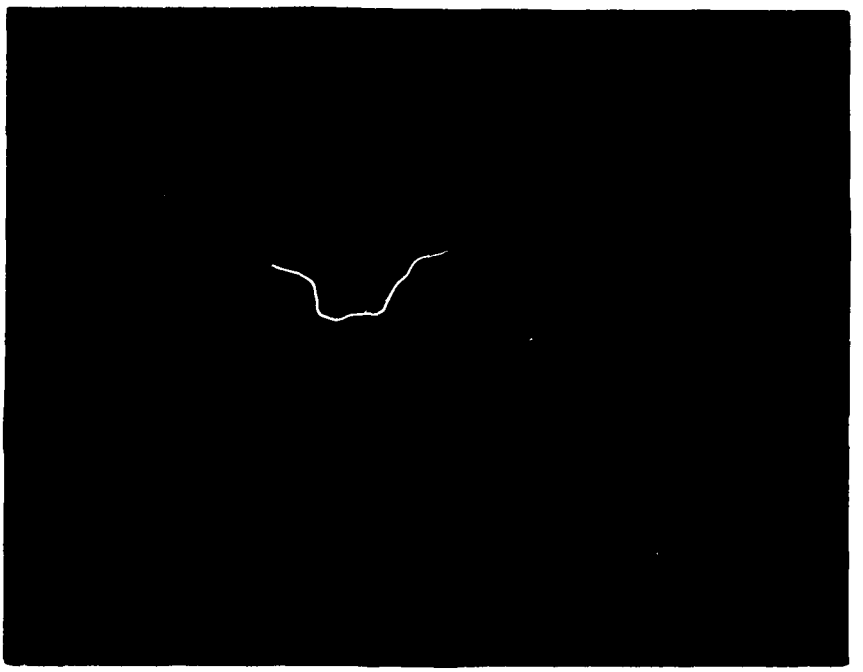


Figure 83: A Single Frame from 12 Sequence

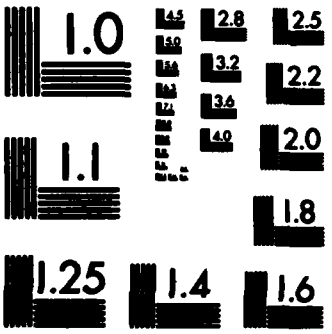
changing perspective have components only in the direction of the translation. This knowledge can be combined with the scanline conversion process to simplify the rendering of several images by scan converting the image into an intermediate form. Such a form is then used to create all final frames without requiring a total scan conversion for each frame.

Assume that images are to be generated for translation along the X axis. A scanline algorithm would first perform the perspective transformation of the polygons, and then intersect each scanline with the transformed polygons to create spans consisting of starting and ending X and Z values, and necessary shading information. As other researchers have suggested, these span buffers can be saved as an intermediate form of the image for further processing [50]. To complete the rendering, the spans for each scan line are point sampled at the X values associated with pixel centers, and Z values are used to determine visibility. By saving the span buffer information from one image, the process of rendering the modified spans for each frame is simple and fast. A similar approach can be taken to changes in the Y direction by creating spans which lie in planes parallel to the Y-Z axes, and point sampling the spans at appropriate Y values.

Individual frames of translation sequences can be created with span buffers by performing a perspective modification of the spans based on Z depth. The modification is applied to the coordinate along which translation is occurring. For each frame, the end coordinates of the span are first translated according to which frame is being generated, and then put through a perspective change. After all spans

AD-A132 548 INTERACTIVE SURFACE VISUALIZATION USING RASTER GRAPHICS 2/2
(U) AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH
D L SCHWEITZER AUG 83 AFIT/CI/NR-83-31D

UNCLASSIFIED F/G 9/2 NL



for a given scan line are modified, the scan conversion is completed to create a finished scan line for the frame.

Rotation sequences can also be generated from span buffers if no perspective is used. As described in the section on movement perception, the dimensionality of a surface can be perceived with non-perspective rotation through velocity changes and occlusion. This simplification allows for a set of rotation frames to be created in a manner similar to the translation sequence. Span buffers are created for each scan line, rotated based on the frame being generated, and point sampled for final rendering. The desired axis of rotation determines in which direction the spans are created and processed.

6.5 Results

Multiple frame viewing provides smooth motion of a transforming image through 48 frames. The lower resolution of the individual images tends to highlight the aliasing problem, and could be corrected to a degree through edge filtering. However, the focus of this technique is on shape perception through movement, and edge filtering would be an expensive addition which would not necessarily enhance the perceptual aspects. Similarly, for simplicity, the shading of each span was not changed as the span was transformed. This is equivalent to the viewer moving around the surface versus the surface moving with respect to the light source. Once again, movement perception is not hampered by this simplification.

Span buffers greatly reduce the frame generation time. The consequence of no perspective in the rotation sequence does

occasionally cause confusion in the direction of rotation as expected from the discussion on motion perception, but does not seriously affect the perception of shape.

CHAPTER 7

CONCLUSION

Advances in image generation techniques have resulted in high quality shaded renderings which are used in a variety of applications. High quality in computer graphics has traditionally meant replicating realism to the greatest extent possible and has served as the impetus for most graphics research. Although a great deal of success has been attained toward generating such "photographic quality" images, it has been at the expense of either expensive hardware or excessive computation costs.

Researchers in visual perception have addressed the problem of how the human visual system interprets the three dimensionality of a scene. The techniques which have been presented interactively employ visual perception stimulations to enhance three dimensional visualization without attempting to imitate total realism. Two approaches to enhanced shape visualization have been presented: direct rendering of three dimensional information, and incorporation of the visual perception stimulations of texture gradients, gradients of illumination, binocular stereopsis, and motion perception. Interaction is an inherent part of each technique to enhance perceptual understanding.

Direct visualization of depth and normal information involve

rendering the desired information directly into the frame buffer memory. Interactive viewing algorithms attempt to provide a natural means of coherently visualizing the three dimensional information using mechanisms such as contour lines, hedgehogs, cross sections, and depth planes. Interaction is implemented through the use of the color map and separate image planes.

The perceptual stimulation techniques presented imitate visual cues to enhance three dimensional interpretation at a reasonable computation cost. Artificial texturing uses a simplified projection of "polka-dots" onto a surface to replicate the effects of texture size, shape, and density gradients. Visible surface shadows are generated interactively in a limited lighting environment. Binocular stereo images are generated from a single input image and optionally provide interior surface detail to enhance retinal correspondence. Limited real time motion is provided through multiple image viewing at a reduced resolution.

The ability to interactively modify visualization parameters is an effective tool toward understanding surface shape in three dimensions. Methods of interaction in computer graphics will change with the advent of new hardware display devices, but the application of visual perception techniques should remain to provide maximum visualization results.

APPENDIX

FRAME BUFFER HARDWARE

The purpose of this appendix is to provide a brief description of the frame buffer hardware which was used in this research. Several of the described techniques make use of hardware features for fast user interaction. Although these features are common to many frame buffers, a brief overview of how these features operate will be presented. For a detailed hardware description of the system which was used, the reader is referred to the frame buffer user manuals [23] [24].

The frame buffer used in this research was a Grinnell GMR 270. The resolution of the display was 512 by 512 by 27 bits which is broken into eight bits of red, green, and blue values, and one bit each of red, green, and blue overlays. The overlay planes are one-bit memory planes which can be used to turn a color (or colors) on at full intensity at any point at which they have a value of one. When the value of the overlay is zero, the color value in the appropriate color channel is visible. Turning the overlay bits on and off does not affect the other color channels.

The Grinnell system allows memory readback of the frame buffer. There is also a hardware zoom and pan feature which provides for selecting the center of a viewed area expanded 1, 2, 4, or 8 times normal size. A hardware cursor and digitizing tablet allow user input.

The Grinnell frame buffer provides three color maps each consisting of 256 8-bit locations. The logical structure of the color map is shown in Figure 64. The 8-bit color value of each frame buffer location is used to look up an 8-bit value in the color map which is then sent to the appropriate digital-to-analog converter (DAC). The advantage of using a color map is the relative speed by which a color map can be updated affecting the entire picture, i.e., during one refresh cycle. In addition, the color channel used to look up the value in each map is selectable. Thus, if an intensity is stored in the blue value of each frame buffer location, the blue value can be used to address all three maps giving the image full color capability. The output of each map can be directed to any DAC for individual channels to be viewed.

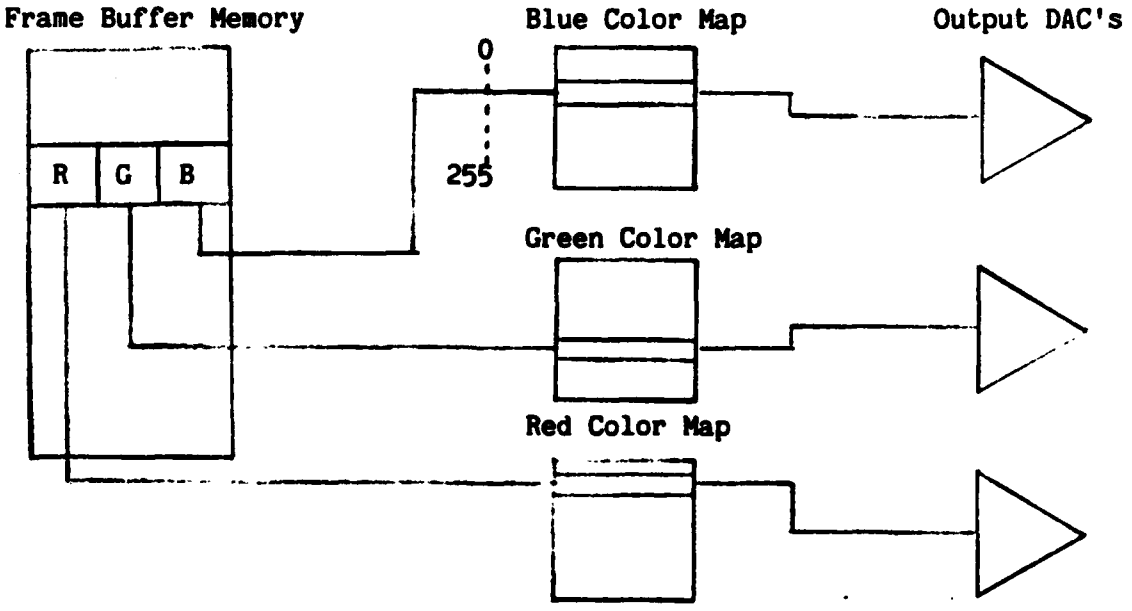


Figure 64: Color Map

REFERENCES

1. Ackland, B. and Weste, N., "Real Time Animation Playback on a Frame Store Display System," Computer Graphics, Vol. 14, No. 3, July 1980, pp. 182-188.

2. Atherton, P., Weiler, K. and Greenberg, D., "Polygon Shadow Generation," Computer Graphics, Vol. 12, No. 3, August 1978, pp. 275-281.

3. Baxter, Brent, "3-D Viewer for Interpretation of Multiple Scan Sections," Proceedings of the National Computer Conference, AFIPS, 1980, pp. 825-829.

4. Blinn, James F. and Newell, Martin E., "Texture and Reflection in Computer Generated Images," Communications of the ACM, Vol. 19, No. 10, October 1976, pp. 542-547.

5. Blinn, J. F., "Models of Light Reflection for Computer Synthesized Pictures," Computer Graphics, Vol. 11, No. 2, Summer 1977, pp. 192-198.

6. Blinn, J. F., "Simulation of Wrinkled Surfaces," Computer Graphics, Vol. 12, No. 3, August 1978, pp. 286-292.

7. Blinn, J. F., "Light Reflection Functions for Simulation of Clouds and Dusty Surfaces," Computer Graphics, Vol. 16, No. 3, August 1982, pp. 21-29.

8. Braunstein, M. L., Depth Perception through Motion, Academic Press, New York, 1976.

9. Brown, Bruce E., "Computer Graphics for Large Scale Two- and Three-Dimensional Analysis of Complex Geometries," Computer Graphics, Vol. 13, No. 2, August 1979, pp. 33-40.

10. Catmull, E., "A Subdivision Algorithm for Computer Display of Curved Surfaces," Tech. report UTEC-CSc-74-133, University of Utah, December 1974.

11. Christiansen, H., "Applications of Continuous Tone Computer Generated Images in Structural Mechanics," Structural Mechanics Computer Programs - Surveys, Assessments and Availability, Pilkey, Saczalski, and Schaeffer, eds., University Press of Virginia, 1974.

12. Cook, R. L. and Torrance, K. E., "A Reflectance Model for Computer Graphics," Computer Graphics, Vol. 15, No. 3, August 1981, pp. 307-316.

13. Crow, F. C., "Shadow Algorithms for Computer Graphics," Computer Graphics, Vol. 11, No. 2, Summer 1977, pp. 242-248.

14. Csuri, C., Hackathorn, R., Parent, R., Carlson, W., and Howard, M., "Towards an Interactive High Visual Complexity Animation System," Computer Graphics, Vol. 13, No. 2, August 1979, pp. 289-299.

15. desJardins, M. and Hasler, A. F., "Stereographic Displays of Atmospheric Model Data," Computer Graphics, Vol. 14, No. 3, July 1980, pp. 134-139.

16. Dill, John C., "An Application of Color Graphics to the Display of Surface Curvature," Computer Graphics, Vol. 15, No. 3, August 1981, pp. 153-162.

17. Duff, T., "Smoothly Shaded Renderings of Polyhedral Objects on Raster Displays," Computer Graphics, Vol. 13, No. 2, August 1979, pp. 270-275.

18. Feibush, E. A., Levoy, M., and Cook, R. L., "Synthetic Texturing using Digital Filters," Computer Graphics, Vol. 14, No. 3, July 1980, pp. 294-301.

19. Foley, J. D. and Van Dam, A., Fundamentals of Interactive Computer Graphics, Addison-Wesley, Reading, Mass., 1982.

20. Forrest, A. R., "On the Rendering of Surfaces," Computer Graphics, Vol. 13, No. 2, August 1979, pp. 253-259.

21. Fuchs, Henry, "On Visible Surface Generation by a Priori Tree Structures," Computer Graphics, Vol. 14, No. 3, July 1980, pp. 124-133.

22. Gibson, J. J. , The Perception of the Visual World, The Riverside Press, Cambridge, Mass, 1950.

23. GMR-27 User's Manual GSC-100599A, Grinnell Systems Corporation, Santa Clara, California, 1978.

24. GMR 270 Series User's Manual GSC-102370B, Grinnell Systems Corporation, Santa Clara, California, 1981.

25. Gulick, W. L. and Lawson, R. B., Human Stereopsis, Oxford University Press, New York, 1976.

26. Haber, Ralph N. and Henderson, Maurice, The Psychology of Visual

Perception, 2nd ed., Holt, Rhinehart and Winston, New York, 1980.

27. Haber, R. N. and Wilkenson, L., "Perceptual Components of Computer Displays," IEEE Computer Graphics and Applications, Vol. 2, No. 3, May 1982, pp. 23-35.

28. Hubschman, H. and Zucker, S. W., "Frame-To-Frame Coherence and the Hidden Surface Computation: Constraints For a Convex World," Computer Graphics, Vol. 15, No. 3, August 1981, pp. 45-54.

29. Ittelson, W. H., Visual Space Perception, Springer Publishing Company, Inc, New York, 1960.

30. Julesz, B., "Binocular Depth Perception without Familiarity Cues," in Contemporary Theory and Research in Visual Perception, Holt, Rinehart and Winston, Inc., New York, 1968.

31. Lane, J. M., Carpenter, L. C., and Whitted, T., "Scan Line Methods for Displaying Parametrically Defined Surfaces," Communications of the ACM, Vol. 23, No. 1, January 1980, pp. 23-34.

32. Mitroo, J. B., "Movies From Music: Visualizing Musical Compositions," Computer Graphics, Vol. 13, No. 2, August 1979, pp. 218-225.

33. Newman, William F. and Sproull, Robert F., Principles of Interactive Computer Programming, 2nd ed., McGraw-Hill, New York, 1979.

34. Norton, A., Rockwood, A. P., and Skolmoski, P. T., "Clamping: A Method of Antialiasing Textured Surfaces by Bandwidth Limiting in Object Space," Computer Graphics, Vol. 16, No. 3, July 1982, pp. 1-8.

35. Phillips, R. J., "Stationary Visual Texture and the Estimation of Slant Angle," Quarterly Journal of Experimental Psychology, No. 22, 1970, pp. 389-397.

36. Phong, B. T., "Illumination for Computer Generated Images," Tech. report UTECH-CSC-73-129, University of Utah, July 1973.

37. Porter, T. K., "Spherical Shading," Computer Graphics, Vol. 12, No. 3, August 1978, pp. 282-285.

38. Roese, John A. and McCleary, Lawrence E., "Stereoscopic Computer Graphics for Simulation and Modeling," Computer Graphics, Vol. 13, No. 2, August 1979, pp. 41-47.

39. Schachter, "Computer Image Generation for Flight Simulation," IEEE Computer Graphics and Applications, Vol. 1, No. 4, October

1981, pp. 29-68.

40. Schweitzer, D. and Cobb, E. S., "Scanline Rendering of Parametric Surfaces," Computer Graphics, Vol. 16, No. 3, July 1982, pp. 265-271.

41. Shoup, Richard G., "Color Table Animation," Computer Graphics, Vol. 13, No. 2, August 1979, pp. 8-13.

42. Sloan, Kenneth R. and Brown, Christopher M., "Color Map Techniques," Computer Graphics and Image Processing, Vol. 101979, pp. 297-317.

43. Stern, G., "SoftCel - An Application of Raster Scan Graphics to Conventional Cel Animation," Computer Graphics, Vol. 13, No. 2, August 1979, pp. 284-288.

44. Sutherland, I. E., "A Head-mounted Three Dimensional Display," FJCC 1968, Thompson Books, Washington, D.C., 1968, pp. 757-764.

45. Sutherland, Ivan E., Sproull, Robert F., and Schumacker, Robert A., "A Characterization of Ten Hidden-Surface Algorithms," Computing Surveys, Vol. 6, No. 1, March 1974, pp. 1-55.

46. Tanimoto, S. L., "Color-mapping Techniques for Computer-aided Design and Verification of VLSI Systems," Computers and Graphics, Vol. 5, 1980, pp. 103-113.

47. Truckenbrod, J. R., "Effective Use of Color in Computer Graphics," Computer Graphics, Vol. 15, No. 3, August 1981, pp. 83-90.

48. Weinberg, Richard, "Computer Graphics in Support of Space Shuttle Simulation," Computer Graphics, Vol. 12, No. 3, August 1978, pp. 82-86.

49. Weintraub, Daniel J. and Walker, Edward L., Perception, Brooks/Cole, Belmont, Calif., 1968.

50. Whitted, Turner and Weimer, David, "A Software Test-Bed for the Development of 3-D Raster Graphics," Computer Graphics, Vol. 15, No. 3, August 1981, pp. 271-278.

51. Williams, L., "Casting Curved Shadows on Curved Surfaces," Computer Graphics, Vol. 12, No. 3, August 1978, pp. 270-274.


FILM