

Viewer-Plane Experiments with Computed Holography with the MIT Holographic Video System

by

John David Sutter

B.S. Computer Engineering
Boston University
Boston, MA
1992

Submitted to the Program in Media Arts and Sciences,
School of Architecture and Planning
in partial fulfillment of the requirements for the degree of
MASTER OF SCIENCE IN MEDIA ARTS & SCIENCES

at the

Massachusetts Institute of Technology

September 1994

© Massachusetts Institute of Technology 1994.
All rights reserved.

Author _____
Program in Media Arts and Sciences
August 5, 1994

Certified by _____
Stephen A. Benton
Allen Professor of Media Arts and Sciences
Program in Media Arts and Sciences
Thesis Supervisor

Accepted by _____
Stephen A. Benton
Chairman
Departmental Committee on Graduate Students
Program in Media Arts and Sciences

**Viewer-Plane Experiments with
Computed Holography with
the MIT Holographic Video System**

by

John David Sutter

Submitted to the Program in Media Arts and Sciences,
School of Architecture and Planning
on August 5, 1994, in partial fulfillment of the
requirements for the degree of
MASTER OF SCIENCE IN MEDIA ARTS & SCIENCES

Abstract

Until this past year, all holographic video systems have been image-plane displays. This was not due to a lack of interest, but to a lack of width. The most recent version of the MARK II MIT holographic video system projects a hologram that is wide enough to accommodate both of the viewer's eyes and allow for a fair amount of movement. This thesis investigates the geometric aspects of the viewer-plane configuration and presents a novel method of computing the holographic fringe pattern. Viewzone data replication is investigated as a method of reducing computation time as well as reducing system throughput requirements. Finally, topics for future research are discussed. An original goal of this work was to take advantage of the viewer-plane configuration's less demanding bandwidth requirements, but the author found no method to do this.

Thesis Supervisor: Stephen A. Benton
Title: Allen Professor of Media Arts and Sciences
Program in Media Arts and Sciences

Research on the MIT holographic video system has been supported in part by US West Advance Technologies, Inc., the Advanced Projects Research Agency (ARPA) through the Rome Air Development Center (RADC) of the Air Force System Command (under contract No. F30602-89-C-0022) and through the Naval Ordnance Station, Indian Head, Maryland (under contract No. N00174-91-C0117), and by the Television of Tomorrow research consortium of the Media Laboratory, MIT.

I would like to thank all those who helped me in this work and made life here a little more fun: the entire Spatial Imaging group and especially Mark Lucente, Pierre St. Hilaire, Ravikanth Pappu, and Lynne.

**Viewer-Plane Experiments with
Computed Holography with
the MIT Holographic Video System**

by

John David Sutter

The following people have served as readers for this thesis:

Thesis Reader_____

V. Michael Bove, Jr.

Associate Professor of Media Technology

Program in Media Arts and Sciences

Thesis Reader_____

Neil Gershenfeld

Assistant Professor of Media Technology

Program in Media Arts and Sciences

Contents

1	Introduction	9
1.1	Scope	9
1.2	The MIT Holographic Video System	9
1.2.1	Optical Subsystem	10
1.2.2	Computational Subsystem	12
1.2.3	RF and Sync Subsystem	12
2	Viewer-Plane Configuration	13
2.1	Theory	13
2.1.1	Horizontal Viewzone	14
2.1.2	Vertical Viewzone	16
2.1.3	Image Size	16
2.1.4	Image Depth	18
2.1.5	Fringe Data Spectral Content	21
2.2	Implementation Based on Mark II	22
2.2.1	The Vertical Diffuser and Viewzone	22
2.2.2	Results	23
3	Viewer-Plane Configuration Computation	25
3.1	History	25
3.2	Computation	26
3.3	Results	29
3.3.1	The Problems	29
3.3.2	Evaluation	30
4	View Data Replication	31

4.1	Theory	31
4.2	Implementation	34
4.3	Results	34
4.4	Conclusion	35
5	Speculation	37
5.1	A Color Based upon the MARK II	37
5.2	Doubling the Viewing Angle	39
6	Summary	41
A	sp_segrep.c: Viewer-Plane Hologram Computation with Replication	43
	Bibliography	

List of Figures

1-1	<i>Simplified top view of the MARK II holographic video system.</i>	10
1-2	<i>Simplified side view of the MARK II holographic video system.</i>	11
2-1	<i>Basic hologram viewing configurations: (a) The viewer-plane configuration, (b) The image-plane configuration.</i>	14
2-2	<i>Horizontal viewzones: (a) The viewer-plane configuration, (b) The image-plane configuration.</i>	15
2-3	<i>Image widths: (a) Full aperture sliding window scheme. (b) Framing aperture, sliding window scheme. (c) Intersecting image area scheme (d) The image-plane configuration.</i>	17
2-4	<i>The disparity budget, θ_{budget}, is equal to $\theta_{near} - \theta_{far}$.</i>	19
2-5	<i>The image depth and area as determined by geometric constraints. (a) The viewzone is smaller than the image width. (b) The viewzone is larger than the image width.</i>	20
2-6	<i>Fringe data spectra for the image- and viewer-plane configurations.</i>	21
2-7	<i>The vertical diffuser and the vertical viewzone.</i>	23
2-8	<i>A test image (left) and a predistorted test image (right).</i>	24
3-1	<i>A single illuminated image point at the image plane: the basis for any viewer-plane hologram.</i>	26
3-2	<i>Geometry used to compute the fringe pattern.</i>	27
3-3	<i>A series of single frequency gratings is used to increase the size of the point.</i>	27
3-4	<i>A multiple point image requires shifting the basis fringe before it is added to the hologram data.</i>	28
3-5	<i>The hologram is split into pupil size viewzones and the hologram data is computed separately for each view from the basis fringe.</i>	28

3-6	<i>A video-captured image computed using the algorithm described in section 3.2. Note the loss of vertical lines due to vertical misalignment of the optical system.</i>	30
4-1	<i>Replication of viewzone fringe data. (a) Fully computed viewzone fringe data. (b) Half computed viewzone fringe data, other half replicated. (c) Segmented, half computed viewzone fringe data, other segments replicated. (Not to scale, D is approximately 500 mm and W is 3mm) . .</i>	32
4-2	<i>Test images showing affects of replication and segmentation factors (R,S).</i>	36
5-1	<i>Setup to use full system bandwidth for color display. Only one color and one direction are shown.</i>	38
5-2	<i>Using an AOM to align the green and blue beams with the red.</i>	39
5-3	<i>Using masking AOMs to double the viewing angle.</i>	40

List of Tables

4.1	Blur (in mm) Due to Replication of View Zone Fringe Data	33
4.2	Computation Times (m:ss) for Various Replication and Segmentation Factors	34

Chapter 1

Introduction

This thesis investigates a relatively unexplored area in the parameter-space of holographic video systems: the viewer-plane configuration. In the past, holographic video systems have used the image-plane configuration, where the image is at or near the hologram plane. Although there are many reasons why one might choose the image-plane configuration over the viewer-plane configuration, until recently there has been no choice. The MARK II implementation of the MIT holographic video system is the first system large enough to accommodate the viewer-plane configuration. This chapter presents the scope of this work and reviews the MARK II holographic video system.

1.1 Scope

This thesis presents the viewer-plane configuration for holographic video, a novel approach to hologram computation, a method to speed computation, and speculation on future areas of research based on the viewer-plane configuration.

1.2 The MIT Holographic Video System

Although holographic video displays were proposed shortly after the re-discovery of optical holography[5, 7, 3] it took nearly thirty years for a high quality holographic video image to be demonstrated. This system, developed by Benton, Kollin, and St. Hilaire later evolved into a 25 mm by 25 mm, 64 line, full-color display called the MARK I holographic video system[13, 12, 20, 21, 16]. St. Hilaire presents a brief history of holographic video in his doctoral thesis[22].

The MARK I was a remarkable development and was soon replicated by several other labs. While those labs were busy building their own MARK I-based systems, St. Hilaire was developing a scaled-up holographic video system that currently displays

images 150 mm wide by 75 mm high with a 30° viewing angle. This system, the MARK II, meets the primary requirement for implementing a viewer-plane system: both of the viewer’s eyes can lie within the projected hologram plane at the same time.

A brief description of the MARK II holographic video system follows with the optical, computational, and radio-frequency (RF) subsystems considered separately. A thorough treatment can be found in St. Hilaire’s doctoral thesis[22].

1.2.1 Optical Subsystem

The optical subsystem can be considered two separate systems. The horizontal system, shown figure 1-1, projects one line of a holographic image. The vertical system, shown in figure 1-2, multiplexes this line vertically but does not diffract light: the projected hologram displays horizontal parallax only (HPO). Each system is described below.

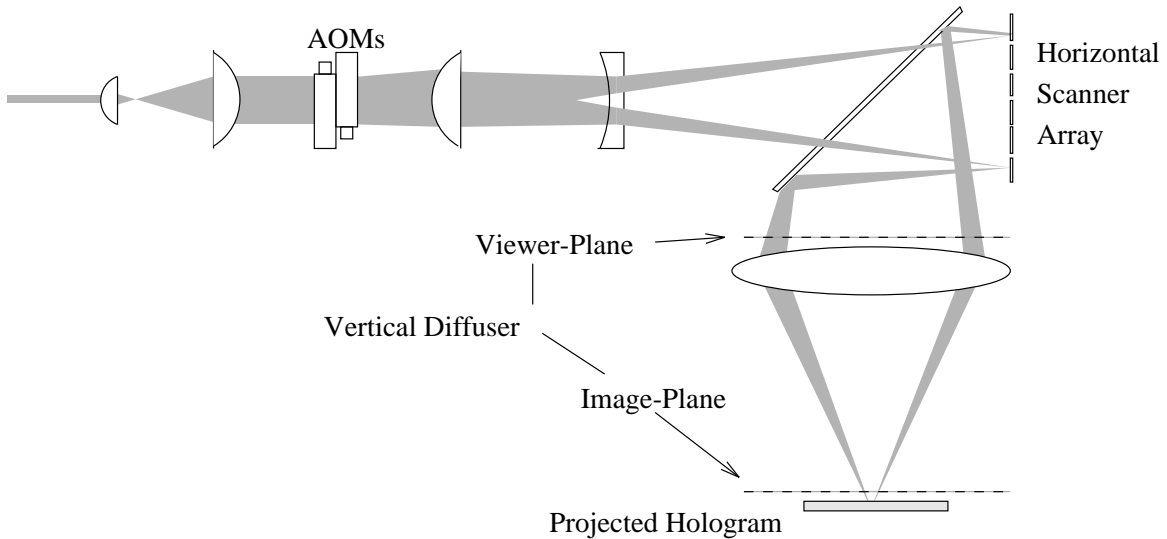


Figure 1-1: *Simplified top view of the MARK II holographic video system.*

The horizontal system is responsible for diffracting light and imaging it onto the hologram plane in a stable manner. The computational and RF subsystems feed the computed fringe pattern into two 18-channel acousto-optic modulators (AOMs). An AOM is a crystal of TeO_2 with an ultrasonic transducer at one edge. An RF signal in

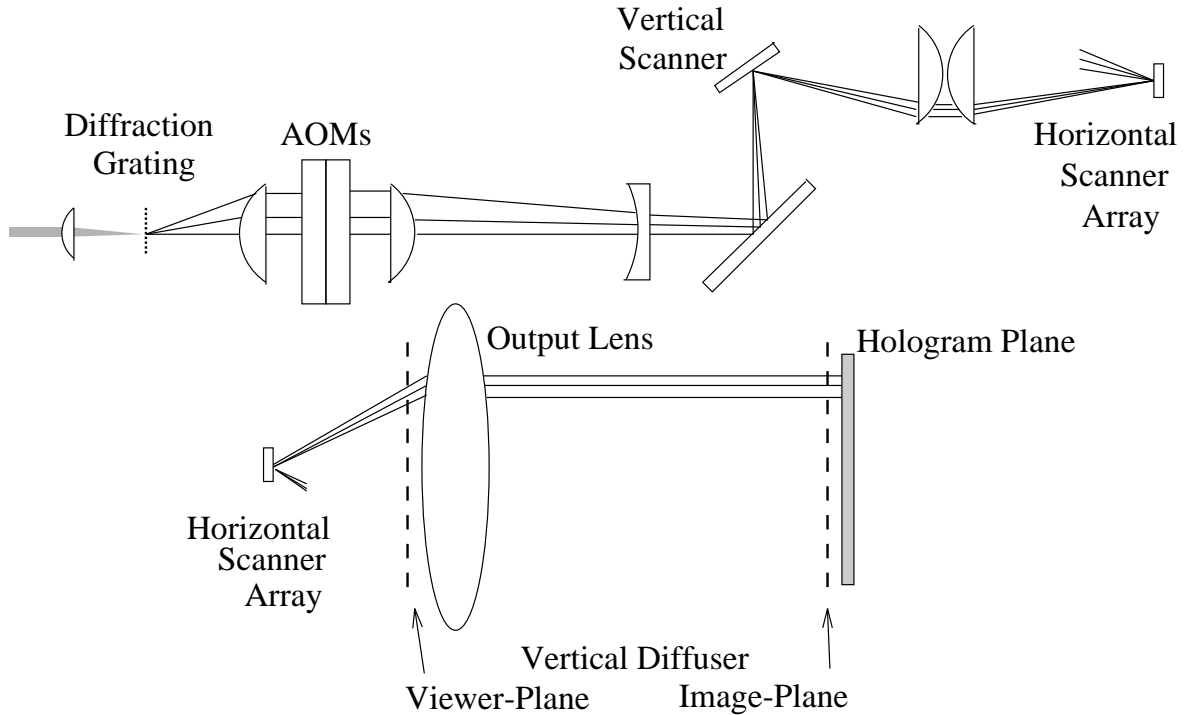


Figure 1-2: *Simplified side view of the MARK II holographic video system.*

the 50-100 MHz range produces shear waves that travel down the length of the crystal inducing changes in its index of refraction. These changes result in phase delays that diffract the collimated illumination beam.

The image of the light diffracted by the AOM is demagnified and projected onto the hologram plane by the three lenses to the right of the AOMs. The demagnification is necessary because the maximum spatial frequency within the AOM itself is only 89 cycles/mm giving a maximum deflection of 3° . The image is demagnified by approximately 10 giving a maximum spatial frequency of 874 cycles/mm for a deflection of 34° . The horizontal scanner array consists of six galvanometric scanners that scan the image of the fringes across the hologram plane. The movement is synchronized with the speed of the shear waves so that the fringes are stationary in the hologram plane[20]. Note that there are two AOMs: two are used to get maximum utility from the display without wasting time on horizontal retrace intervals. One AOM has fringes traveling in one direction when the scanners are rotating the proper direction. When the scanners change direction, the other AOM becomes active with fringes traveling in the opposite direction.

The vertical system is responsible for rastering the image of the AOMs vertically. An image of the AOMs is focused onto the vertical scanning mirror that is at the focal point of a telecentric system that relays the image onto the horizontal scanning array. The scanning array is at the focal point of the output lens: the vertically scanned lines are parallel after the output lens.

The important aspects of the optical system as they pertain to this thesis are that the projected hologram is 150 mm wide by 75 mm high and has 256 k samples horizontally giving a maximum spatial frequency of 874 cycles/mm and that the vertical diffuser can be moved to a different location without affecting the rest of the system.

1.2.2 Computational Subsystem

The computational subsystem consists of a custom, digital video testbed designed at the MIT Media Lab, Cheops, that was modified to meet the needs of holographic video[25, 1]:

110 MHz pixel clock
256 k pixels per line
8 scan lines per frame
18 video output channels

The system has its own local processor as well as a variety of special processors on daughter cards, but most processing is currently done on a remote host and the hologram fringe data is downloaded as required. This configuration remains unchanged for this work.

1.2.3 RF and Sync Subsystem

This section processes the video signals from the Cheops system. The video signals are upshifted, filtered, and amplified to place them within the operating range of the AOMs. Synchronization signals from Cheops are used to control the horizontal and vertical scanning systems. This subsystem dissipates an enormous amount of heat to keep the HVAC from freezing solid. This configuration remains unchanged as well.

Chapter 2

Viewer-Plane Configuration

This chapter presents the viewer-plane configuration for holographic video by comparing it with the image-plane configuration, comparing the bandwidth requirements, discusses the implementation of the viewer-plane configuration using the MARK II holographic video system, and then reports the system's performance.

2.1 Theory

As the name implies, the viewer-plane configuration for holographic video is the case where the plane of the projected hologram coincides with the plane of the viewer's eyes. This configuration is equivalent to the transmission master hologram shown in figure 2-1(a) where the viewer looks through the illuminated hologram and sees the object at a distance of 300-500 mm. In contrast, the usual configuration of the holographic video systems has been the image plane case, where the image is at the hologram plane and the viewer is 300-500 mm away as shown in figure 2-1(b). This setup is equivalent to a full-aperture transmission transfer hologram that projects the image of a master hologram onto the viewer's plane.

The first part of this analysis considers the geometry of the viewer-plane configuration and relates it to the image-plane case. Most of the holographic video system's operating parameters can be ignored for the moment. For the purposes of this analysis, a physical hologram exists with a width of W and a maximum spatial frequency of F_s at some distance D from the viewer. Fringe data at F_s and nearly 0 cycles/mm diffract light by $\pm\theta$:

$$\theta = \sin^{-1}\left(\frac{\lambda F_s}{2}\right)$$

where λ is the wavelength of the illumination source in millimeters. At some point the viewer needs to be considered: the interocular distance, the distance between the

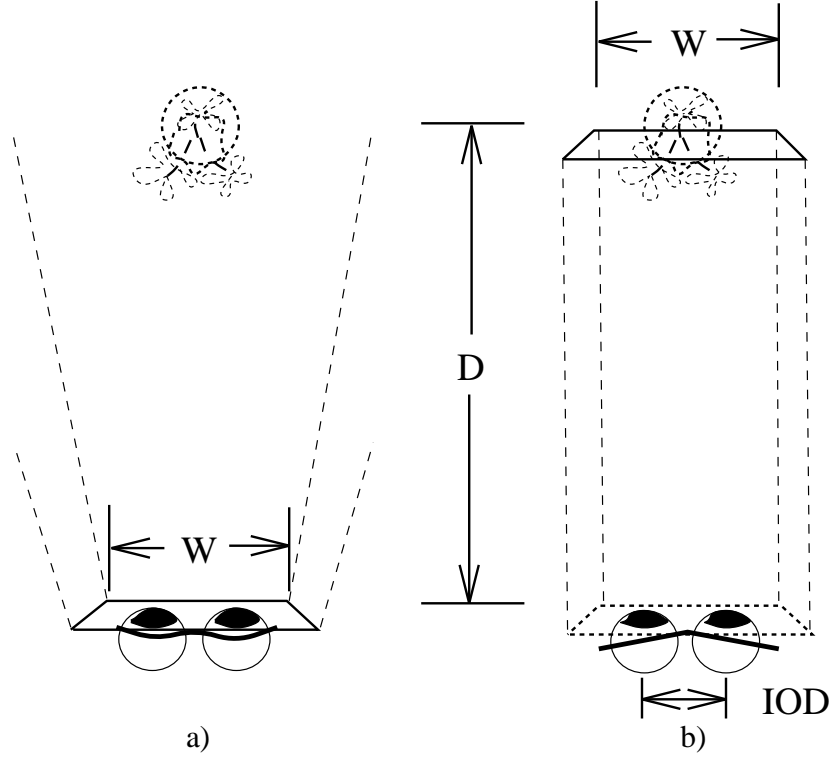


Figure 2-1: *Basic hologram viewing configurations: (a) The viewer-plane configuration, (b) The image-plane configuration.*

pupils, is referred to as IOD and the pupil diameter is D_p . Where quantitative examples are called for the following values from the MARK II holographic video are used:

W :	150 mm
F_s :	870 cy/mm
D :	500 mm
θ :	± 17 degrees
IOD :	65 mm
D_p :	3 mm

2.1.1 Horizontal Viewzone

The horizontal viewzone is one of the most important characteristics of any stereoscopic display. The extent of the viewzone dictates the how much perspective information can be presented to the viewer as well as the volume that the image can occupy.

Inspecting figure 2-2(a) it is clear that the horizontal viewzone for the viewer-plane configuration is equal to the the width of the projected hologram:

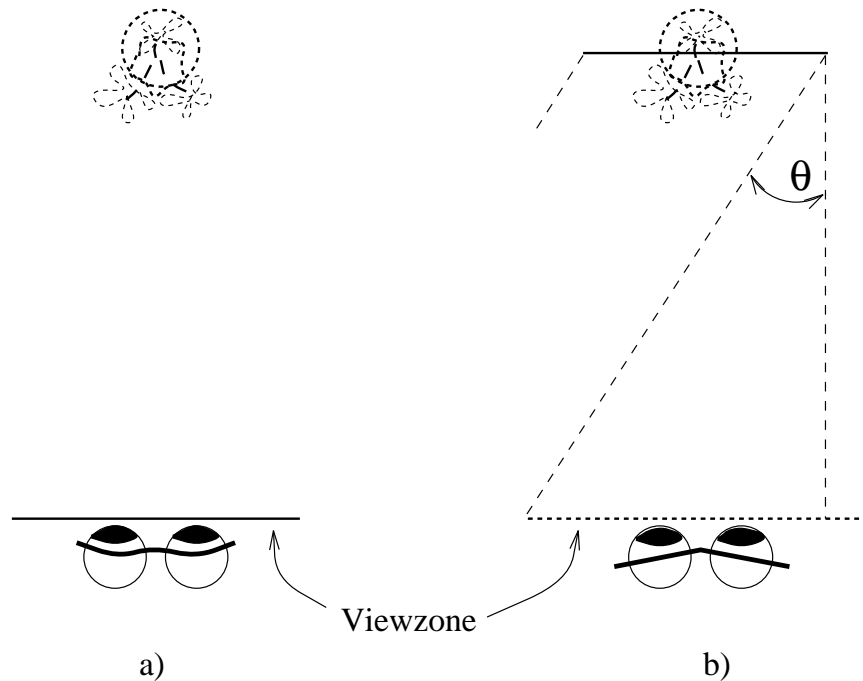


Figure 2-2: *Horizontal viewzones: (a) The viewer-plane configuration, (b) The image-plane configuration.*

$$VZ_{h_{vp}} = W$$

In contrast, figure 2-2(b) shows the viewzone of the image-plane configuration. At first glance it seems that the viewzone should be considerably larger, but this is not the case. As one of the viewer's eyes moves off the end of the viewzone the image becomes cut off and proper framing of the image is lost, yielding a visually confusing image. Increasing the width of the hologram without increasing F_s actually decreases the viewzone:

$$VZ_{h_{ip}} = 2D \tan \theta - W$$

Applying these formulas to the MARK II gives viewzone sizes that are similar. The viewer-plane viewzone is 150 mm and the image-plane viewzone is 156 mm. Not

too surprisingly, the image-plane viewzone is about what one would expect from a full-aperture transmission transfer hologram where a master hologram is projected into the viewer's plane.

2.1.2 Vertical Viewzone

The vertical viewzones for both viewer-plane and image-plane configurations are implementation-specific and are discussed in the implementation section below. There is no particular advantage to either configuration in this aspect of the system.

2.1.3 Image Size

Second in importance is the width of the viewed image. The image should be large enough to be present its component objects with a reasonable amount of parallax: bigger is better. Although the height, width, and depth are all important, the height of the image is dependent upon the holographic video system itself and not the viewing configuration. The width and depth are considered separately.

Image Width

Figures 2-3(a,b,c) show three possibilities for setting the size of the image in the viewer-plane configuration. The first two options feature sliding windows onto the image as the viewer moves through the viewzone. The first option gives impressive widths:

$$W_{obj_{window}} = 2D \tan \theta + IOD \quad (370mm)$$

$$W_{obj_{total}} = 2D \tan \theta + W \quad (455mm)$$

Unfortunately the left and right eyes each see images that do not completely overlap, preventing proper framing of the image. For this reason alone the first option is not reasonable.

The second option presents a solution to this framing problem. The sliding windows are forced to have the proper framing. Alas, this scheme only works when the

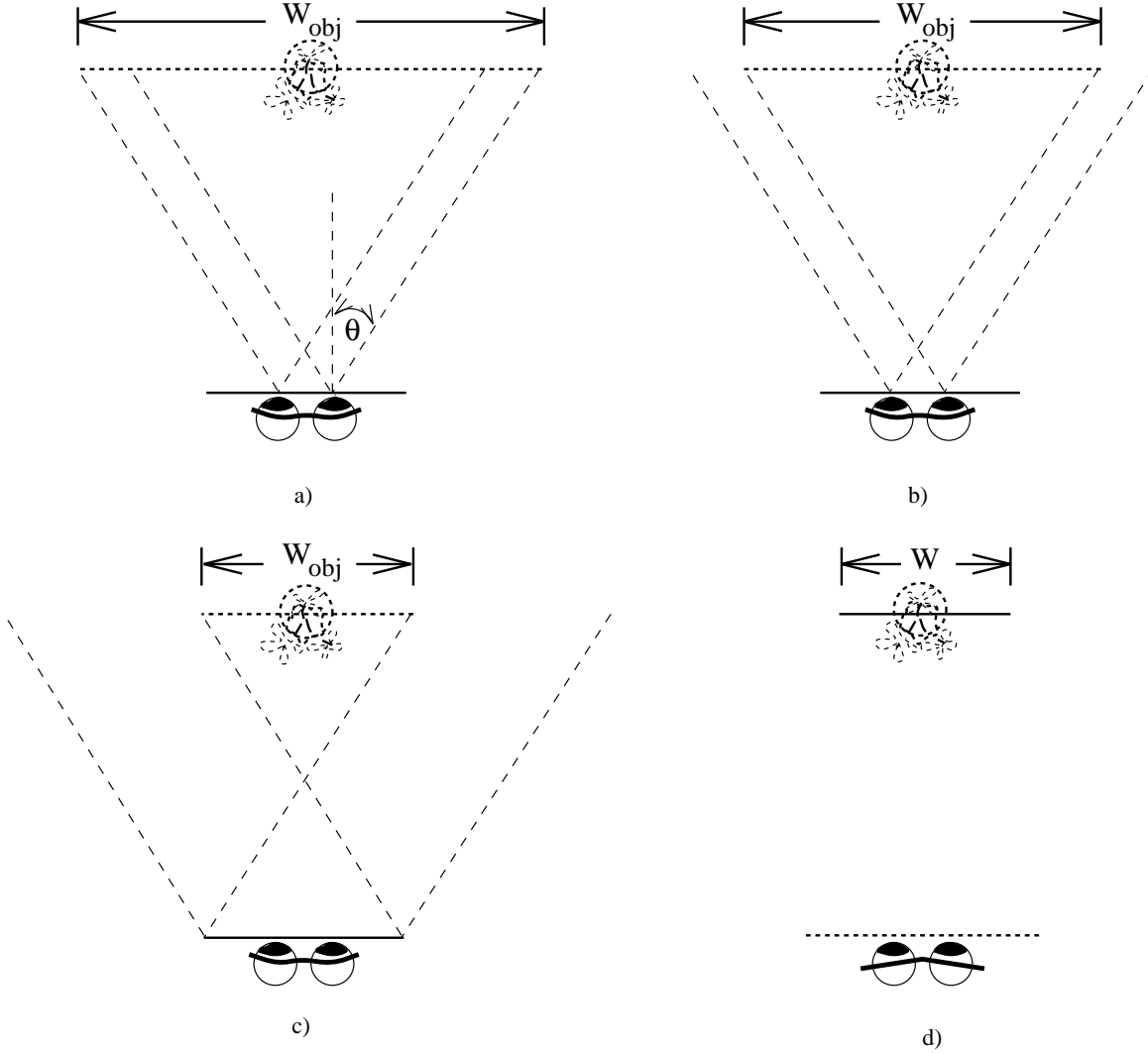


Figure 2-3: *Image widths: (a) Full aperture sliding window scheme. (b) Framing aperture, sliding window scheme. (c) Intersecting image area scheme (d) The image-plane configuration.*

viewzone is less than or equal to twice the IOD. For a W of 130 mm:

$$W_{obj_{window}} = 2D \tan \theta - IOD \quad (240mm)$$

$$W_{obj_{total}} = 2(D \tan \theta - IOD) + W \quad (305mm)$$

The final option is the solution used in the implementation described in a following section. Only the portion of the image plane visible from the entire viewzone is used. There is no sliding window in this scheme. The total width is given by:

$$W_{obj_{total}} = 2D \tan \theta - W \quad (155mm)$$

As in the case of the image-plane viewzone, increasing F_s results in a larger image width but increasing W decreases it, wasting more space in the image-plane.

The space to either side of the image area does not have to be wasted. If care is taken to leave space for proper framing around the active image area, this space can be used for desktop tools such as control panels or a clock. These images are not viewable from every viewzone so they should be two-dimensional existing only in the image plane. Making these peripheral images three-dimensional only confuses the viewer when one eye moves beyond the viewing area of these images. Note that this is a combination of the second and third options.

2.1.4 Image Depth

Although the maximum depth of the image is affected by the viewing geometry, the human visual system is the true limiting factor. HPO displays present visually confusing depth cues to the viewer. Astigmatism arising from different horizontal and vertical focal planes of the image, conflicting horizontal focal planes resulting from viewing amidst two viewzones, and disparity can easily result in a spatial display that makes no sense. These issues are covered in any good treatment of holographic stereogram. A good tool for determining the maximum depth of an image is the disparity budget.

The disparity budget, $\theta_{disparity}$, specifies the amount of convergence the eye can tolerate and still fuse perspective views into a spatial image.

The disparity budget limits the maximum depth of the image by specifying the maximum angular difference between the eye's convergence on the image plane and on the plane representing the maximum depth as shown in figure 2-4:

$$\theta_{disparity} = \theta_{near} - \theta_{far}$$

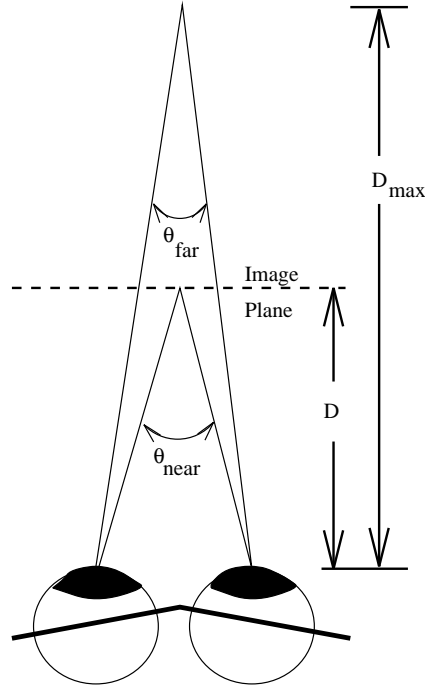


Figure 2-4: The disparity budget, θ_{budget} , is equal to $\theta_{near} - \theta_{far}$.

Although there have been many values for $\theta_{disparity}$ proposed a reasonable value 1.24°. The maximum depth can now be determined:

$$D_{max} = \frac{IOD}{2 \tan(\tan^{-1}(\frac{IOD}{2D}) - \frac{\theta_{disparity}}{2})}$$

or more as an approximation:

$$D_{max} = \frac{1}{\frac{1}{D} - \frac{\theta_{disparity}}{IOD}}$$

These formulas give a D_{max} of about 600 mm for the MARK II holographic video system. Note that the disparity budget is an issue because only stereograms are being considered. If the fringe data is computed from an image database instead of perspective views and the image points are not projected onto a single image plane, the disparity budget is no longer a constraint because there is no single plane that the eyes' convergence is tied to. The other problems mentioned earlier, astigmatism and straddling viewzones, limit the image depth in this situation.

Figure 2-5 shows that the maximum image depth depends greatly upon the relative

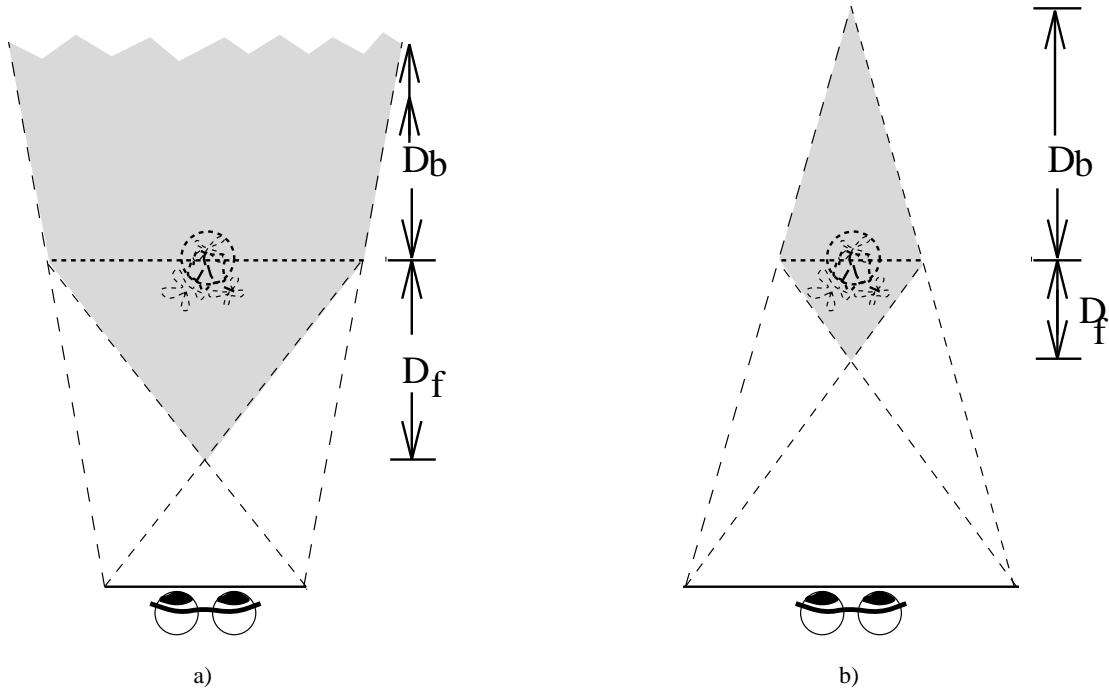


Figure 2-5: *The image depth and area as determined by geometric constraints. (a) The viewzone is smaller than the image width. (b) The viewzone is larger than the image width.*

size of the viewzone and the image width. If the viewzone is smaller than the image width, only the viewing geometry limits the depth of the image in front of the viewing plane, D_f :

$$D_{f_{vp}} = D - \frac{W}{2 \tan \theta} \quad (255 \text{mm})$$

$$D_{f_{ip}} = \frac{W}{2 \tan \theta} \quad (245 \text{mm})$$

If the viewzone is larger than the image width the depth of the image behind the image plane, D_b , is given by:

$$D_{f_{vp}} = \frac{D(D \tan \theta - \frac{W}{2})}{(W - D \tan \theta)}$$

$$D_{f_{ip}} = \frac{D(D \tan \theta - \frac{W}{2})}{(D \tan \theta - W)}$$

The maximum image depth using the MARK II is not, for all practical purposes, limited by the viewing configuration. Although the depth is technically limited in the

image-plane configuration at 13.1 meters this is beyond the optical resolution of the system.

2.1.5 Fringe Data Spectral Content

The simplicity of the viewer-plane configuration entices one to believe, and rightly so, that the bandwidth requirements are less than that of the image-plane case. If the spectra of small segments of fringe data computed for the image-plane and viewer-plane configuration are examined, they are expected to be quite different. In the image-plane configuration, each segment must be able to diffract light to illuminate each viewzone that it should be seen from. Figure 2-6(a) shows this as a sequence of equally spaced spatial frequency distributions: one for each viewzone. The width of these distributions determines the width of the viewzones. Lucente has investigated the distribution requirement quite thoroughly[17].

The viewer-plane configuration, on the other hand, simply requires a single spatial frequency for each illuminated point for each viewzone(Figure 2-6(b)). Although the spectra shift as the viewzone changes, the maximum number of spatial frequencies in any viewzone is always equal to the maximum number of illuminated points. The difference between the two configurations is that the image-plane case requires the complete bandwidth of the system to display a point and the viewer-plane case only uses a small portion of the available bandwidth in any given sample. Unfortunately, no method was found to utilize this unused bandwidth and remains a challenging problem for the next investigator.

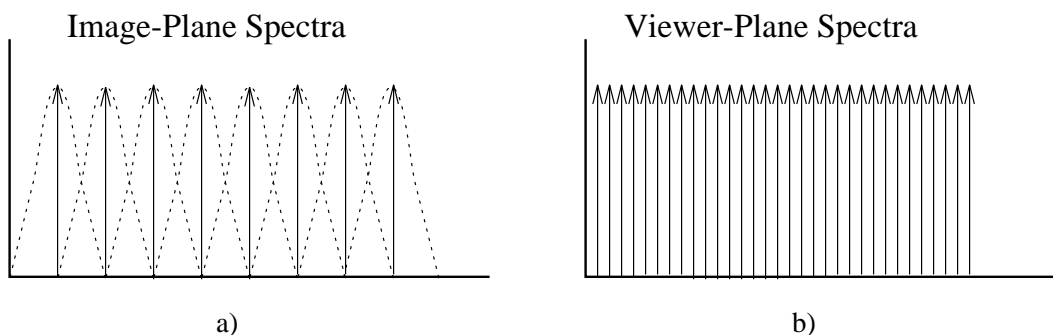


Figure 2-6: *Fringe data spectra for the image- and viewer-plane configurations.*

2.2 Implementation Based on Mark II

The MARK II holographic video system was studied to determine what needed to be done to implement a viewer-plane configuration with minimal modifications. It was found that the system only needed two changes to make it ready for these experiments. The first was to move the hologram plane to a reasonable viewing area. The hologram plane was projected 280 mm in front of the output lens and 200 mm from the edge of the table at a height of 250 mm. Extensive changes to the optical system would be necessary to move the hologram plane, but discussion with other researchers working on the system convinced the author that he would be ill advised to touch the optics. The solution was to leave the system as-is with the viewer somewhat uncomfortable. The other change was to relocate the vertical diffuser from the hologram plane to a more suitable some distance away from the viewer.

2.2.1 The Vertical Diffuser and Viewzone

The proper location of the vertical diffuser in the holographic video system is crucial to obtaining quality images with correct parallax. The requirements are simple: the diffuser must be at the image plane and the diffuser must be able to diffuse the light enough to illuminate a vertical viewzone of reasonable height. Placing the image plane at 500 mm from the viewer's plane means placing the vertical diffuser directly behind the output lens (from the viewer's point of view). The diffuser was actually placed at 400 mm from the viewer and was imaged to 460 mm by the output lens. Figure 2-7 shows how the vertical viewzone is formed in the image-plane and viewer-plane configurations. The diffuser itself distributes light over an angle of $\pm\theta_{diff}$ so the vertical viewzone is:

$$VZ_v = 2D\tan\theta_{diff} - H$$

where H is the height of the projected hologram. The vertical viewzone for the MARK II in the image-plane case is 190 mm ($H = 75$ mm, $\theta_{diff} = 15^\circ$).

Placing the vertical diffuser behind the lens considerably complicates the determination of the viewzone. The lens images the diffused light so that a $\pm 15^\circ$ diffuser

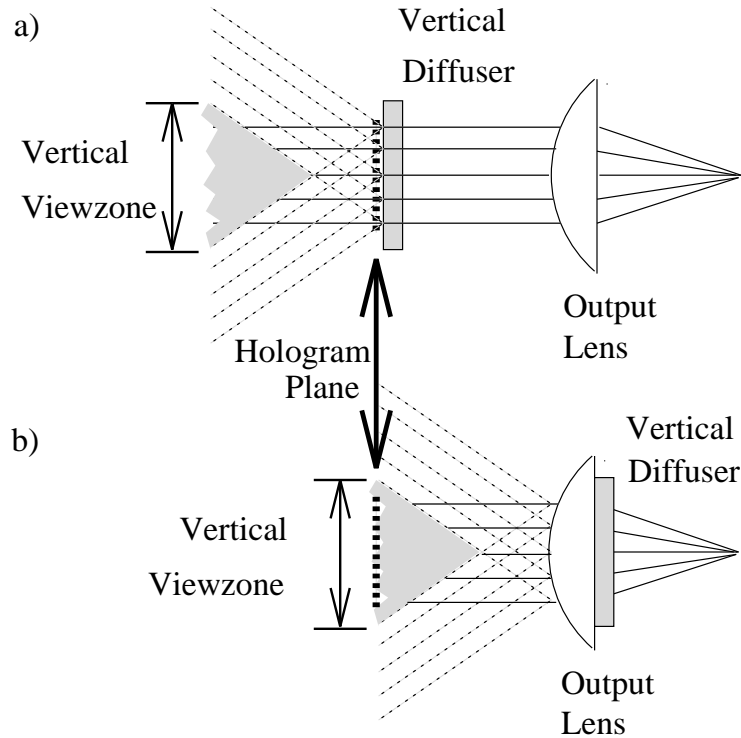


Figure 2-7: *The vertical diffuser and the vertical viewzone.*

spreads the light from $+16^\circ$ to -38° . The limiting region of diffusion is at the center of the diffuser where light is still diffused at $\pm 15^\circ$. The vertical viewzone with the diffuser behind the lens is:

$$VZ_v = 2D \tan \theta_{diff}$$

This works out to be 260 mm, and is more than adequate and helps alleviate the uncomfortable viewing arrangements.

2.2.2 Results

The first images displayed, though exciting to have any images at all, were disappointing. Only gross surface details could be seen and it was difficult to identify them. Investigations showed that the vertical optical system was out of alignment for any case other than the image-plane configuration. Minor adjustments were attempted but it became clear that the system would need to be rebuilt to fix the problem: this was not a practical option. The problem was that the image of the 18-channel AOM was warped such that a diagonal line was displayed as a sinusoid. The resolution was

to predistort the images to use as many of the vertical lines as possible. This worked but left areas of the image blank. Figure 2-8 shows a test image on the left and the predistorted image on the right.

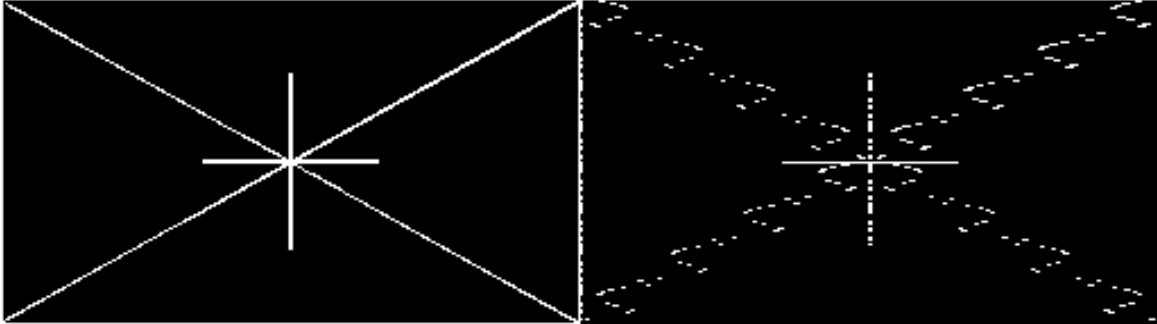


Figure 2-8: *A test image (left) and a predistorted test image (right).*

The other problem noticed was that alternating groups of 18 lines were at different depths. This was due to the AOMs being at different distances themselves. The only solution would be to compute separate a fringe table for each AOM and to use the appropriate table for computing each group of lines.

Despite these problems, the images produced using this configuration are bright and detailed. Resolving the alignment problem would produce a more pleasing image, but that will be left to the next researcher.

The next chapter discusses how the fringe data for the viewer-plane holograms are computed.

Chapter 3

Viewer-Plane Configuration Computation

The MARK II holographic video system is a versatile tool for studying the use of diffracted light to display spatial images. The projected hologram consists of an arbitrary fringe pattern with a maximum spatial content, F_s , of 874 lp/mm. This chapter presents a novel approach to computing the fringe data for the view-plane configuration.

3.1 History

Computational generated holography has been around nearly as long as the author and has been published extensively[23, 4]. Most of the work has focused on producing fringe patterns that are true to nature or at least produce equivalent wavefronts. Although such accuracy may be required for optical signal processing systems, the demands of display holography are not as stringent. For example, the fringe pattern of a physical hologram can be determined from:

$$I_{Total} = |\mathbf{E}_O|^2 + |\mathbf{E}_R|^2 + 2\Re\{\mathbf{E}_O \mathbf{E}_R^*\},$$

where \mathbf{E}_O and \mathbf{E}_R are the complex electric fields of the object and the reference beams respectively and I_{Total} is the intensity of the light exposing the photographic plate. The first term is due to self-interference by the object and results in noise upon image playback. The second term results in a uniform DC offset to the fringe data that can be used to properly bias the exposure in physical holography although in electroholography it not needed. The third term provides all the information required to reconstruct the original image wavefront. Actually computing a hologram based on this term still takes a considerable amount of time even considering the power of today's supercomputers.

Early publications on the MIT holographic video system discuss eliminating ver-

tical parallax and reducing vertical resolution to increase the speed of computing holographic fringe patterns[12, 13, 24]. Although these efforts made the computational task possible, it was far from being fast. Lucente’s bipolar intensity method of computing holograms using precomputed elemental fringes provided a significant speed-up allowing frame rates of 3 frames per second using the MARK I holographic video system[16]. The elemental fringes were computed for discretized values of x and z and stored in a two-dimensional array. Computing a hologram consisted of accumulating the elemental fringe patterns for all the image points and then normalizing the data for the display system.

3.2 Computation

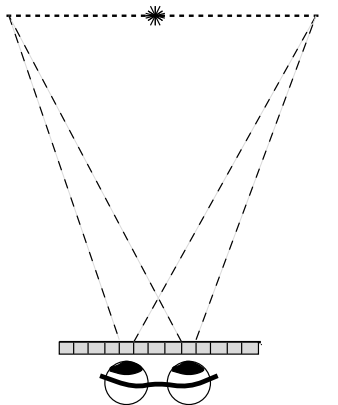


Figure 3-1: *A single illuminated image point at the image plane: the basis for any viewer-plane hologram.*

The method used to compute holograms for this thesis starts with a single illuminated point at the image plane as shown in figure 3-1. Instead of computing an array of elemental fringe patterns, only one fringe pattern is computed. Figure 3-2 shows the parameters used to do the computation: θ_{ill} is the angle of the collimated illumination beam and θ is the maximum that it can be diffracted relative to the normal. This basis fringe is a chirped sinusoid with a maximum spatial frequency of F_s and a minimum near 0 cy/mm. The sharpness of the point using this fringe pattern is limited only by the resolution of the optical system. The point is actually smaller than convenient.

The size of the point can be controlled by approximating the chirped sinusoidal

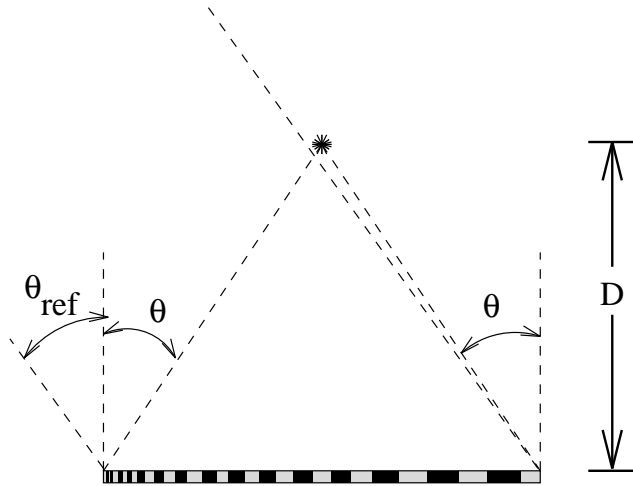


Figure 3-2: *Geometry used to compute the fringe pattern.*

pattern with a series of single frequency gratings as shown in figure 3-3. The width of the grating segment, W_g , is set equal to the desired point size, W_s . The basis fringe is computed one grating at a time each with an increasing spatial frequency until the maximum spatial frequency is reached.

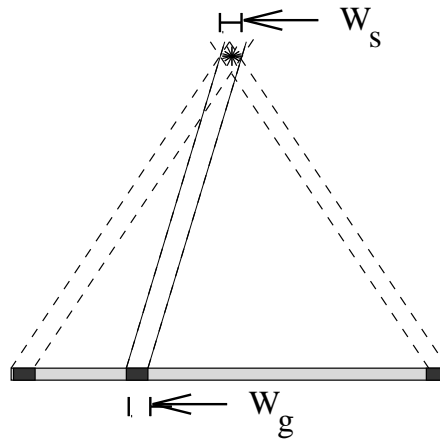


Figure 3-3: *A series of single frequency gratings is used to increase the size of the point.*

Now that the basis fringe has been determined, the hologram can be computed. In this example, the computation is trivial because the image is a single point. An index into the fringe data is calculated that indicates where the light is diffracted at an angle normal to the viewer's plane. The fringe pattern for the image is simply the basis fringe data centered around this index. Note that it is not a problem that

only part of the basis fringe can fit into the frame buffer. The other parts of the basis fringe are used if the image point is offset from the center. Computing a more complex image is a simple matter of summing the basis fringe centered upon each point (figure 3-4). A array of indexes for each possible point location is compiled to speed the computation process.

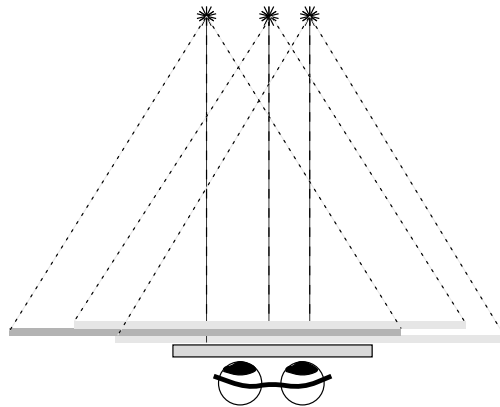


Figure 3-4: A multiple point image requires shifting the basis fringe before it is added to the hologram data.

The algorithm just described will only produce a flat image. What can be done to give it depth? There are two options at this point: compute a number of basis fringes for different depths or split the hologram plane into pupil sized viewzones and compute a holographic stereogram. The latter method was chosen for simplicity.

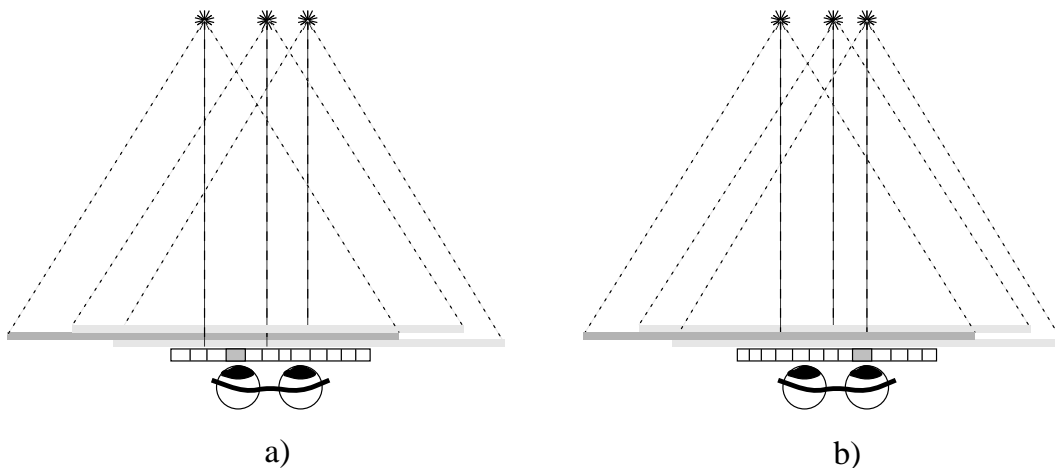


Figure 3-5: The hologram is split into pupil size viewzones and the hologram data is computed separately for each view from the basis fringe.

The hologram data is computed separately for each viewzone in the hologram plane using its perspective view and the basis fringe as shown in figure 3-5. Each view needs its own array of indexes to image points, so a two dimensional array of indexes is compiled. Note that if the image exists completely within the image plane then the resulting hologram data should be exactly the same as if viewzones were not used.

3.3 Results

Although there were several problems with the images displayed using this implementation of the viewer-plane configuration, the results demonstrated the viability of this configuration.

3.3.1 The Problems

None of the problems encountered in this work indicated any intrinsic problem with the viewer-plane configuration. The most annoying problem was that nearly 40 percent of the display lines were blank due to the predistortion required by the misalignment of the MARK II's vertical system. Resolving this problem requires completely rebuilding the system.

Other problems were that the image was slightly shorter than in the image-plane configuration and that the image distorted slightly when viewed from the extremes of the viewzone. Both problems are the result of placing the vertical diffuser behind the output lens. Moving the hologram plane further from the output lens and placing the diffuser directly in front of the lens will resolve these problems.

An example of the image is shown in figure 3-6. The image was captured using a miniature video camera and the *SGI Sirius* video system. Unfortunately the camera is totally automatic and provided poor contrast images that, when averaged over several shots, yielded a poor image.

3.3.2 Evaluation

Overall, the viewer-plane and image-plane configurations are equivalent in terms of what they can offer when implemented on today's holographic video systems. There are, however, several advantages to the viewer-plane configuration. The first is that each viewzone is independent of any other viewzone and can be manipulated separately. If a view changes with the image-plane case, the entire hologram needs to be recomputed. Second is that if a large enough θ can be achieved, a very immersive display is possible. A last advantage is that because the viewer is located at the hologram plane, the system requires less physical space .

As for disadvantages, the primary one is that the viewer-plane configuration will be a single viewer system for the foreseeable future until significantly wider displays are available requiring significant improvements in framebuffer speed, AOM technology, and scanner speed.

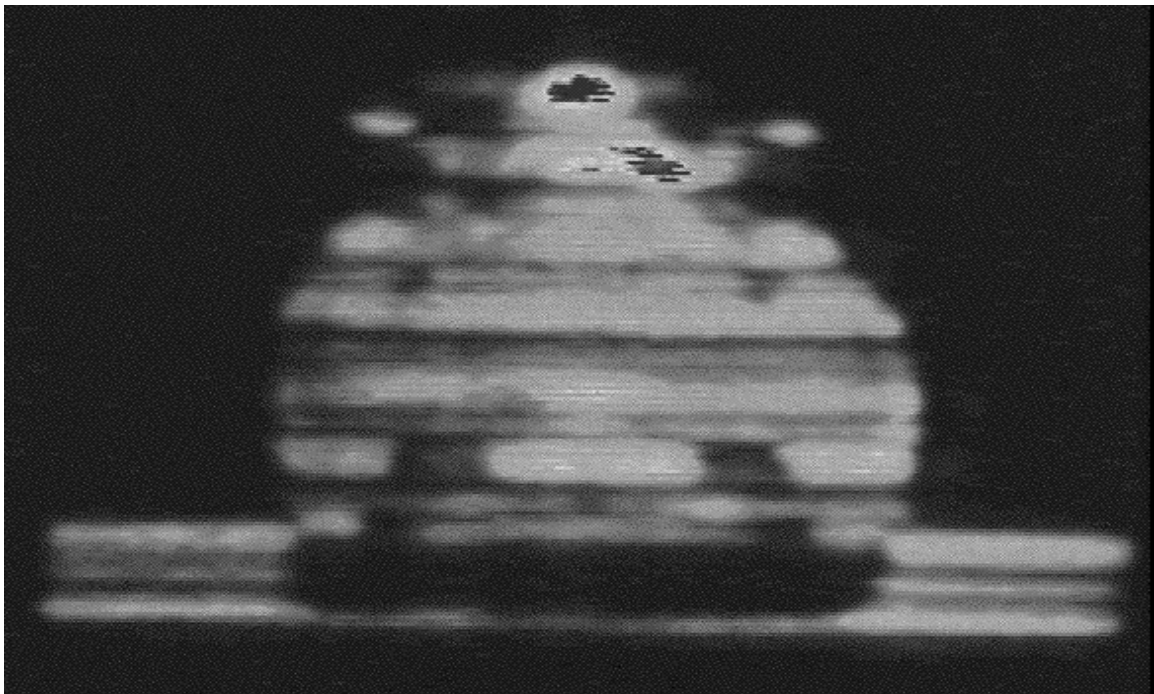


Figure 3-6: A video-captured image computed using the algorithm described in section 3.2. Note the loss of vertical lines due to vertical misalignment of the optical system.

Chapter 4

View Data Replication

Lucente has researched fringe data replication to a great extent with emphasis upon the image-plane configuration and a preliminary discussion of the viewer-plane case[17]. This chapter discusses view data replication and its impact upon the viewed image in the viewer-plane configuration. It is shown that the computation time of the fringe data can be reduced significantly without affecting the perceived image quality.

4.1 Theory

As in previous chapters, the view plane is split laterally into a number of viewing zones each with the appropriate fringe pattern to produce the correct perspective view. Each illuminated point in the image contributes its own fringe pattern to the viewzone fringe data such that any sample of the fringe data produces the entire perspective view through a restricted aperture. If the sampling is done carefully the sampled fringe data can be replicated across the view zone with little blurring.

Early experiments implemented view data replication by computing the fringe data for a portion of the viewzone and repeating it to fill the viewzone. Although simple, this method introduces considerable blur because each copy of the computed fringe data produces an image that is horizontally offset an amount equal to its own offset relative to the original fringe data. Consider the case shown in figure 4-1(a) where all the fringe data is computed. The dot can be seen throughout all portions of the viewzone and its sharpness is limited by the accuracy of the fringe data and the resolution of the optical system. If only half of the viewzone fringe data is computed and then copied into the remaining portion of the viewzone (figure 4-1(b)), the viewer will see two discrete dots. This is clearly undesirable. Increasing the replication factor (the number of times fringe data is replicated) only replaces the two dots with a continuous blur. The width of the blur is given by:

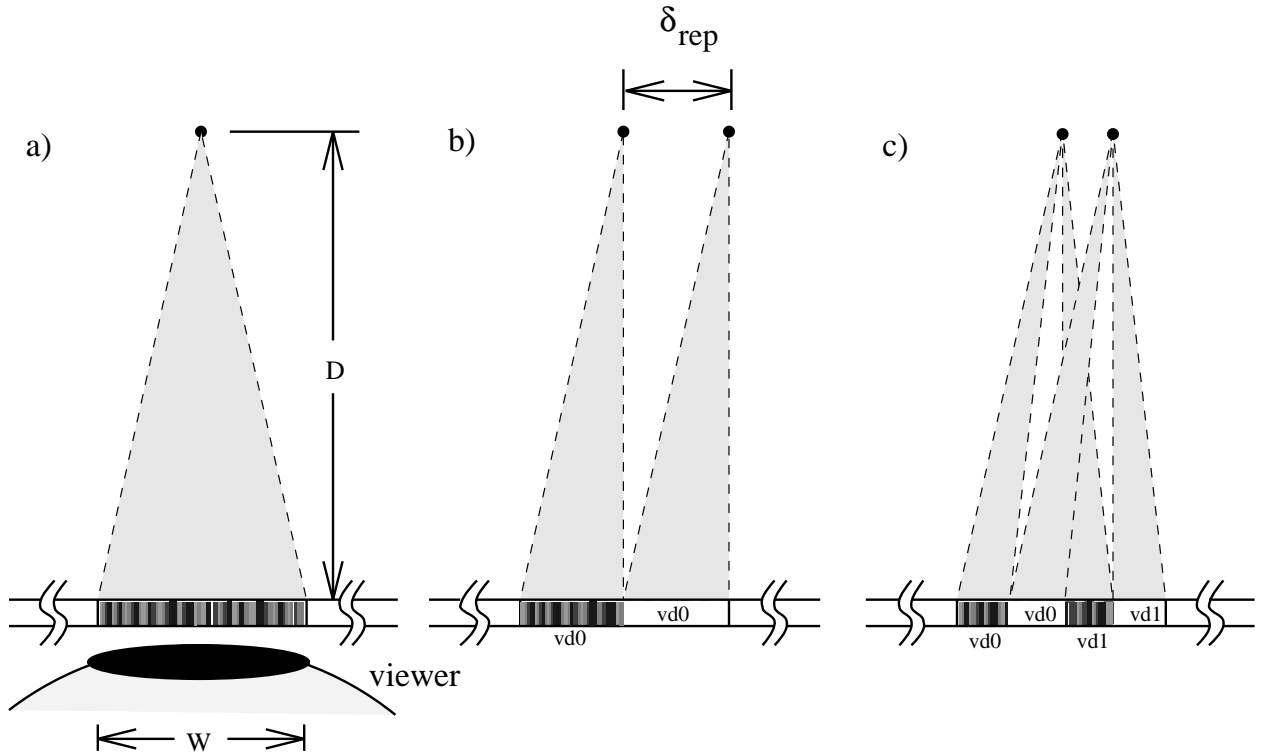


Figure 4-1: *Replication of viewzone fringe data. (a) Fully computed viewzone fringe data. (b) Half computed viewzone fringe data, other half replicated. (c) Segmented, half computed viewzone fringe data, other segments replicated. (Not to scale, D is approximately 500 mm and W is 3mm)*

$$\delta_{rep} = W * \frac{R - 1}{R},$$

where R is the replication factor and W is the width of the viewzone. Given a viewzone of 3 mm the blur increases from 1.5 mm to 2.7 mm as the replication factor increases from 2 to 10 which is many times larger than the 0.5 mm point size. This is clearly unacceptable.

The blur can be significantly reduced by sampling the computed fringe data in several segments across the viewzone instead of just one and then replicating these segments to fill the remaining space. This is the same as breaking the viewzone into a number of subview zones (the segmentation factor) and performing view data replication in each subview zone independently as shown in figure 4-1(c). The width of the blur now becomes:

Table 4.1: Blur (in mm) Due to Replication of View Zone Fringe Data

Replication Factor	Segmentation Factor					
	1	2	4	6	8	10
1	0.0					
2	1.5	0.75	0.375	0.25	0.188	0.15
4	2.25	1.125	0.563	0.375	0.281	0.225
6	2.5	1.25	0.625	0.417	0.313	0.25
8	2.625	1.313	0.656	0.438	0.328	0.263
10	2.7	1.35	0.675	0.45	0.338	0.27

$$\delta_{rep} = W * \frac{R - 1}{R * S},$$

where R is the replication factor, S is the segmentation factor and W is the width of the viewzone. Table 4.1 shows how the blur varies with the replication and segmentation factors. This data indicates that if the viewzone fringe data is replicated, it should be segmented as much as possible to reduce the blur. Although increasing the segmentation does not increase computation time it does introduce other problems.

The first problem encountered as the segmentation factor increases is diffraction due to the restricted aperture of the segment. This diffraction produces a dim picket-fence effect when the image is viewed away from the view plane. The picket-fence itself is in the view plane and it is not noticeable when the image is viewed at the view plane.

Higher segmentation factors introduce a more serious problem. As the segmentation factor increases the size of the segment decreases: the number of fringe data points decreases. If the number of data points is too small the segment does not have sufficient throughput to represent the image and a very noisy image is displayed. Lucente has shown that the sampling theory applies here so the minimum number of data points for a 256 point wide image is 512[17].

Table 4.2: Computation Times (m:ss) for Various Replication and Segmentation Factors

Replication Factor	Segmentation Factor					
	1	2	4	6	8	10
1	3:21					
2	1:42 (1:41)	1:41	1:41	1:41	1:42	1:41
4	0:51 (0:50)	0:52	0:51	0:52	0:51	0:51
6	0:34 (0:33)	0:35	0:34	0:34	0:34	0:34
8	0:26 (0:25)	0:26	0:26	0:26	0:26	0:26
10	0:21 (0:20)	0:21	0:21	0:21	0:21	0:20

4.2 Implementation

The holographic fringe computation method described in an earlier chapter was modified to implement the replication and segmentation scheme discussed above. The viewzone fringe data was replicated by copying it into the appropriate empty segments. Although this method of replication wastes a significant portion of the system's throughput it is the most straightforward to implement and does not require modifying the holographic video system itself. The next chapter discusses other ways in that this wasted throughput may be put to use. The source for the program, `sp_segrep`, is in appendix A.

4.3 Results

It is no surprise that viewzone fringe data replication reduces computation time. The table 4.2 shows computation times for a range of replication and segmentation factors adjusted for overhead (29 seconds). The times closely match the expected computation times (in parentheses) showing that a replication factor of R results in a time proportional to $1/R$. These data show also that the segmentation factor does not measurably increase the fringe computation time.

More important than computation time is the effect of replication of viewzone fringe data upon image quality. Figure 4-2 shows the impact of a range of replication and segmentation factors upon the test image. The images in the first column are

all fully computed indicating the cleanest image possible given the distortions from the current optical system. The replication factor increases left to right and the segmentation factor increases top to bottom. As expected, the blur increases with increasing replication factors but is again reduced by an increasing the segmentation factor. At some point the blur due to undersampling becomes significant (e.g. the $(10,10)$ case). Based on these images the maximum replication and segmentation factors for the viewer-plane case are 8 and 6 respectively. These values vary with personal aesthetics to some extent and also with image complexity. A good overall set of factors is $(4,4)$.

4.4 Conclusion

View zone fringe data can be replicated six to eight times and still yield a reasonably sharp image if the computed fringe data is segmented across the viewzone before replication. Failure to do so yields an image with considerable blur, but excessive segmentation yields even more blur due to undersampling. This chapter is foundation for the next that discusses what can be done with the liberated throughput using the MIT holographic video system.

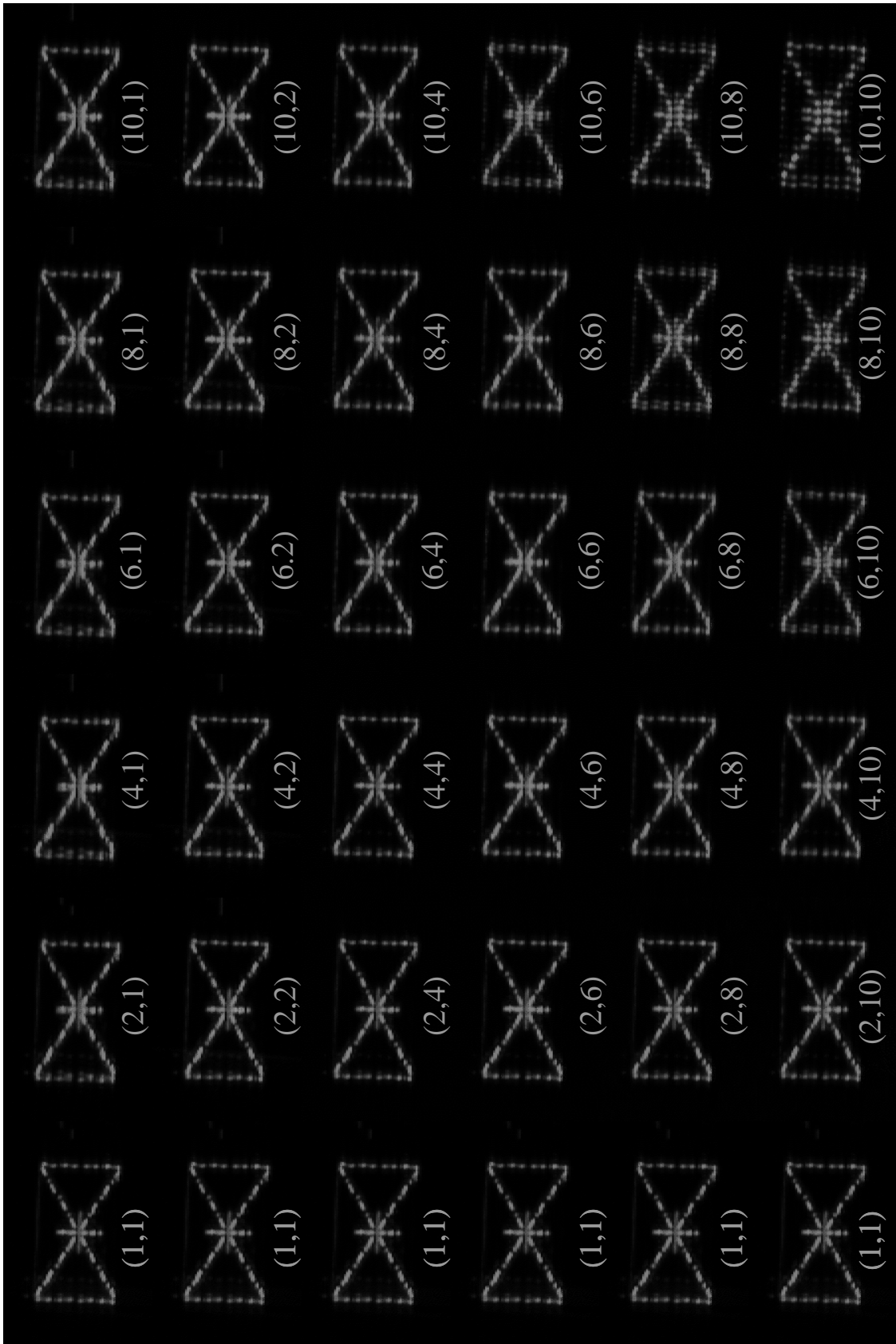


Figure 4-2: Test images showing affects of replication and segmentation factors (R,S) .

Chapter 5

Speculation

This chapter presents several ideas for future research. All of these projects require optical replication of fringe data, first proposed by Lucente[15]. Optical replication uses a simple optical system to project multiple images of the AOM onto the hologram plane. If two extra images are projected, only one third of the fringe data is required. No scheme for optical replication is presented in this thesis.

5.1 A Color Based upon the MARK II

It should be possible to transform the MARK II holographic video system into a full-color display using optical fringe data replication without the loss of vertical resolution as was done with the MARK I or adding additional 18-channel AOMs for each color[22]. The problems that need to be solved are:

- 1) Illuminating the fringe data with the correct laser
- 2) Arranging the fringe data
- 3) Realigning the diffracted light after the AOMs

Early thoughts on this system involved using electro-optic switches to rapidly illuminate the color fringe data in the AOM in sequence(red,green,blue). This just does not work. Switching times waste a considerable amount fringe data and, worse yet, is that the color fringe data segments are too small to completely fill the aperture of the AOM. This latter point is important because, unless even more fringe data is wasted to provide buffer zones around the color fringe data, fringe data for more than one color will be within the aperture simultaneously resulting in optical noise if it is illuminated by the incorrect laser. A solution is to restrict the illuminated aperture of the AOM to the size of the segment plus switching time. This works but results in a significantly dimmer image.

A novel solution is shown in figure 5-1. A single channel AOM is used to mask and deflect light directly onto the appropriate color fringe data in the 18-channel AOM.

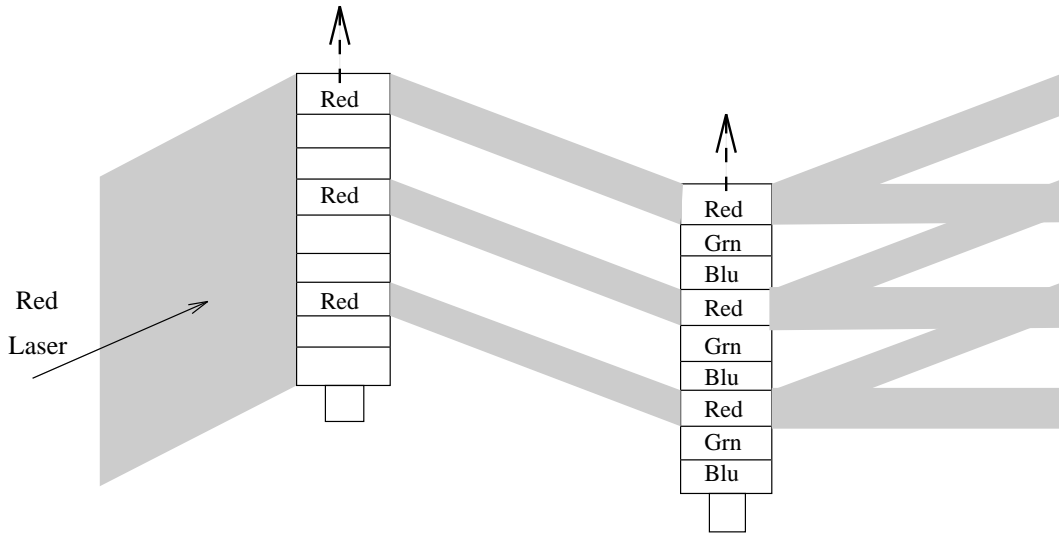


Figure 5-1: *Setup to use full system bandwidth for color display. Only one color and one direction are shown.*

A separate masking AOM is used for each color in each direction. Although proper alignment is critical, some buffer space can be added between the color fringe data to reduce the impact of slight misalignments without greatly affecting image brightness. This also solves the second problem of how the fringe data is to be arranged: a sequence of red, green, and blue fringe data segments with no intervening spaces.

The third problem is considerably more difficult. The MARK I uses two prisms to align the green and blue diffracted beams with the red. That is not possible with this configuration. One solution is to use an 18-channel masking AOM to diffract light from the green and blue channels to align with the red. Figure 5-2(a) shows how red, green, and blue beams of the same spatial frequency are diffracted by an AOM and 5-2(b) shows how the AOM can be used to align them. No other solution has come to mind.

Although the implementation of this system would be very difficult, it is certainly worth investigating. The use of masking AOMs to direct light to specific segments of fringe data interests the author greatly and independent work may continue along this line. Experiments to test the viability of this technique should not be difficult to perform.

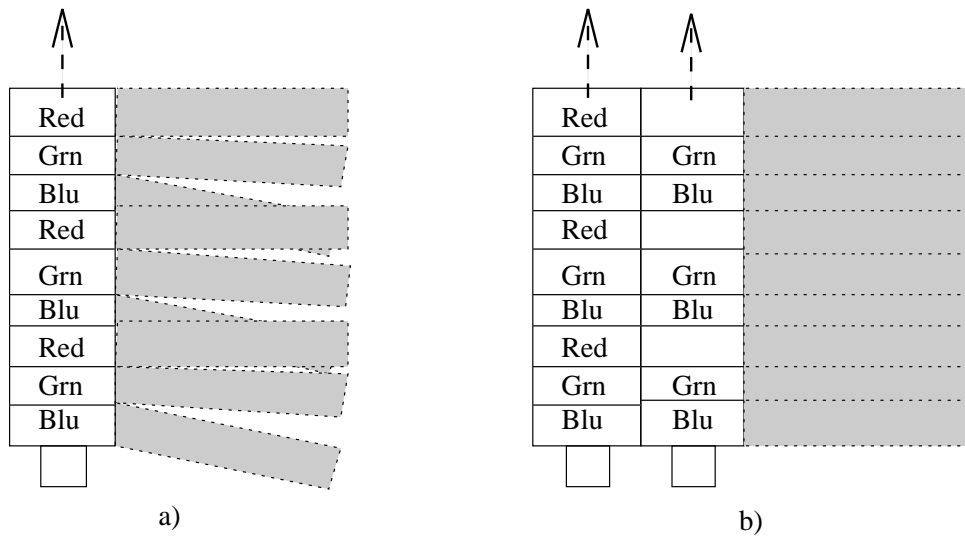


Figure 5-2: *Using an AOM to align the green and blue beams with the red.*

5.2 Doubling the Viewing Angle

The most interesting use of masking AOMs is presented here. Two masking AOMs can be used to double the viewing angle of the MIT holographic video system as shown in figure 5-3. Alternating segments of fringe data are illuminated by left and right AOMs. The AOMs in the current systems can only diffract light in a range of 0° to 3° . This configuration will cover than range of -3° to 3° . Unfortunately, doubling the viewing angle in this way strains the rest of the system: the horizontal scanners and output lens are not large enough. Replacing the output lens with cylindrical or perhaps Fresnel lenses has been proposed by others in the past. Perhaps it is time to reconsider these options.

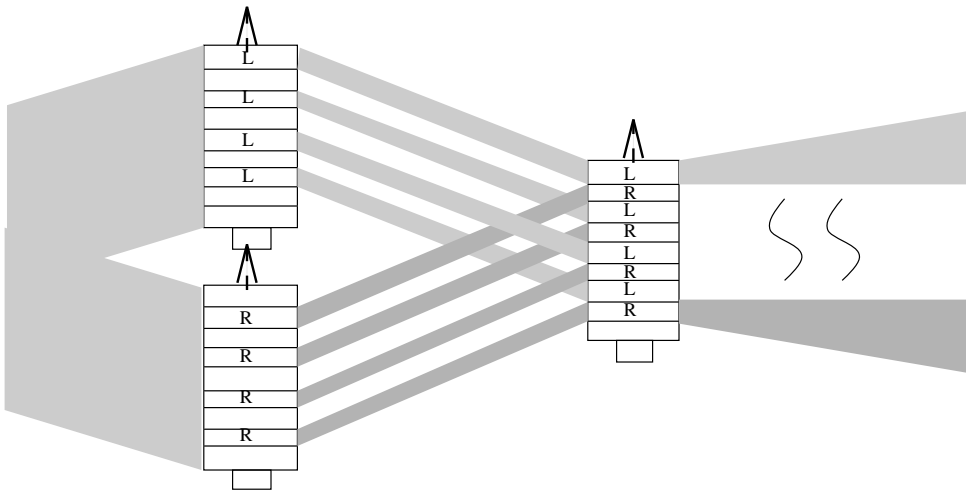


Figure 5-3: *Using masking AOMs to double the viewing angle.*

Chapter 6

Summary

The goal of this work was to study the viewer-plane configuration for a holographic video and find ways to exploit it. The viewer-plane configuration was investigated and compared with the image-plane case. It was found that, given the current holographic video systems, both configurations are limited by the same constraints upon image height and depth but that the width of the image and the viewzone vary inversely with each other.

The MIT holographic video systems are horizontal parallax only (HPO) displays, and limitations inherent to them, such as disparity and astigmatism, prevent the image depth from approaching the geometrical limits. The depth of an image located at 500 mm is limited to about 100 mm. This is a limitation of the human visual system and can not be changed. The height of the image is limited by the number of horizontal lines in the display: increasing the number of lines allows the height of the image to be increased.

The width of the image and viewzone vary with the width of the projected hologram and the maximum angle by which light can be diffracted, θ . In the image-plane configuration, the image width is simply limited to the width of the hologram, but the viewzone, on the other hand, is limited by both the width of the hologram and by θ such that increasing the hologram width without increasing θ actually decreases the size of the viewzone. The viewer-plane configuration is opposite this, with the viewzone width fixed to the size of the projected hologram and the image width depending upon both the hologram width and θ . Given the same system parameters, the image-plane case gives a larger viewzone, but the viewer-plane case allows a larger image.

The viewer-plane configuration was implemented on the MARK II MIT holographic video system and a novel method of computing fringe patterns was developed. The change of configuration required only the careful relocation of the vertical dif-

fuser. Other changes to the system would have made it optimal but were not possible due to other priorities. The holographic fringe patterns were computed by adding the fringe pattern of a single point at the right depth to the total pattern for each point in the image with the pattern offset by the point's offset from center. For simplicity, the fringe pattern was approximated by a series of sinusoidal gratings of increasing spatial frequency. The resulting images were bright and sharp but indicated problems with the vertical alignment of the system.

It was originally believed that the system's sparse use of the available bandwidth could be used to improve the display by increasing the viewing zone. Unfortunately, no way was found to defeat the geometrical and optical constraints of the system. The replication of fringe data was believed to reduce the bandwidth requirements of the system, but it only reduced the throughput requirements instead.

Replication of fringe data, with respect to the viewer-plane configuration, was investigated and its limits were demonstrated. The fringe data was replicated electronically and optical replication remaining an area for future work. A color holographic video system based upon replication and an idea for doubling θ were also presented. These systems would be quite difficult to implement but are worth investigating further.

Future systems will undoubtedly feature a larger θ yielding image-plane displays with large viewzones for presentation to groups of people and also immersive, viewer-plane workstations. The same system can actually be used for both simply by changing the location of the vertical diffuser. Such systems will, of course, require significant technological improvements in computational and scanning technology.

Appendix A

sp_segrep.c: Viewer-Plane Hologram

Computation with Replication

This program was created to generate a viewer-plane holographic stereogram with user specified replication and segmentation factors. Input to the program is a file of all the perspective views. By default the plane of the images is set by the IMG_DEPTH define and the number of views is set by the HOLO_NUM_VIEWS define. The fringe pattern is stored in the specified file. This file can then be loaded into the Cheops framebuffer for display. The replication and segmentation factors are specified using command line options. A typical command line is:

```
% sp_segrep -s 4 -r 4 epv.views.fixed epv.4x4.fringe
```

```
#include <stdio.h>
#include <sys/file.h>
#include <math.h>

typedef unsigned char byte;

#define LAMBDA (632.8E-6)

#define HOLO_OUT_SCALE 8
#define HOLO_OUT_OFFSET 0

#define HOLO_SF_MIN 10.0
#define HOLO_SF_MAX 1190.0

#define HOLO_IMG_THRESH 0
#define HOLO_VIEWS_NUM 32
#define HOLO_VIEWS_NUM_MAX 100

#define MIN_STEPS 30

#define HOLO_TH_REF -16.4

#define HOLO_WIDTH 130.0

#define IMG_DEPTH 254.0 /* diffuser as close to the lens in front as
                        possible */
#define IMG_DEPTH 460.0 /* diffuser behind lens, almost as far away as
                        possible */

#define IMG_WIDTH 100.0
#define HOLO_WIDTH 134.0

#define LIN_LEN 256*1024
```

```

#define HOLO_IMG_WIDTH_MAX 1024
#define HOLO_IMG_WIDTH 256
#define HOLO_IMG_HEIGHT 144

#define dfprintf if(debug) fprintf

#define MAX_PORTION 20
#define MAX_SEGMENT 20

/*
  this experiment places the points at a specified z-depth.
  pretty boring eh? well, this is a rather simple minded
  approach where we start at one end, go through a complete
  cycle and then determine what the next spatial frequency
  is, and keep going until we're out of range..

  this experiment also does view data replication with
  a variable number of segments
  the following values need to be supplied:
  replication factor: j (1/j of the full data is computed)
  segmentation factor: k (computed data is split into k segments)

*/

#include <stdlib.h>

main(argc,argv)
int argc;
char *argv[]; {

  int x, grey, i, val, val2, j, k, l, m, n, num, den, done, fft, perc, cheops, lin_len;
  int xprime, basE;
  double *mem, *fill, *base, *sp_mem, *mptr;
  int sp_len;
  int *vals, len, linlen, llen;
  int hfp, hbp;
  int i1, i2, i3, i4, i5, i6, i7;
  int special, nonum;
  int valx, offset, rlen;
  char buf[128];
  char *line, *outmem, *images;
  char *fout;
  FILE *fopen(), *fp;
  double sf, step, phi;
  int min_ix, max_ix, cnt;
  int fd;
  int begin, end, sp_offset;
  int offsets[HOLO_VIEWS_NUM_MAX][HOLO_IMG_WIDTH_MAX];
  int segoffsets[MAX_PORTION][MAX_SEGMENT];
  int view_offset;
  int portion, segment;
  int seglength;
  double min, max, sum;
  char *pname;
  int flag;
  extern int optind, opterr;
  extern char *optarg;
  int debug, squeeze, blank, total_views, first_view, num_views;
  int view_width, view_height;
  double holo_width, image_width, image_depth, intensity, grating_width;

  pname = argv[0];
  portion = 1;

```

```

segment = 1;
debug = 0;
image_depth = IMG_DEPTH;
squeeze = 0;
blank = 0;
grating_width = -1;
total_views = HOLO_VIEWS_NUM;
first_view = 0;
num_views = -1;;
holo_width = HOLO_WIDTH;
image_width = IMG_WIDTH;
view_width = HOLO_IMG_WIDTH;
view_height = HOLO_IMG_HEIGHT;
intensity = HOLO_OUT_SCALE;

/* accept flags for
   replication,segmentation,debug,depth,squeeze,blank,Views,firtst,num,Width*/
while ((flag = getopt (argc, argv, "r:sDd:S'b'G'g'v'f'i'n'W'w'h")) != EOF)
  switch(flag) {
    case 'r':  portion = atoi(optarg); /* Replication Factor */
              break;
    case 's':  segment = atoi(optarg); /* Segmentation Factor */
              break;
    case 'D':  debug = 1; /* debug mode */
              break;
    case 'd':  image_depth = atof(optarg); /* depth in mm of viewplane */
              break;
    case 'S':  squeeze = 1; /* Squeeze mode */
              break;
    case 'b':  blank = 1; /* blank mode (don't replicated */
              break;
    case 'G':  sscanf(optarg,"%dx%d",&view_width,&view_height);
              break; /* geometry of images */
    case 'g':  grating_width =atof(optarg);/* width of gratings in mm */
              break;
    case 'v':  total_views = atoi(optarg); /* number of persepective views */
              break;
    case 'f':  first_view = atoi(optarg); /* first view to generate (defaults 0*/
              break;
    case 'i':  intensity = atof(optarg); /* scale factor for output (def 8) */
              break;
    case 'n':  num_views = atoi(optarg); /* number of viewszones to fill */
              break;
    case 'W':  holo_width = atof(optarg); /* width of the hologram */
              break;
    case 'w':  image_width = atof(optarg); /* width of the image */
              break;
    case 'h':  fprintf(stderr,"usage: %s [-flags] image_views output file\n",pname);
              fprintf(stderr," b : don't fill in replicated data\n");
              fprintf(stderr," D : debug on\n");
              fprintf(stderr," d depth : image depth (460mm def)\n");
              fprintf(stderr," f view : first view to fill (0 def)\n");
              fprintf(stderr," G WxH : geometry of images (256x144 def)\n");
              fprintf(stderr," g length : grating width in mm\n");
              fprintf(stderr," h : print this message\n");
              fprintf(stderr," i scale : scale factor for intensity(3 def)\n");
              fprintf(stderr," n views : number of views to fill (32 def)\n");
              fprintf(stderr," R ref : reference angle (-16.4 def)\n");
              fprintf(stderr," r num : replicaton factor (1 def)\n");
              fprintf(stderr," S : squeeze output file\n");
              fprintf(stderr," s num : segmentation factor (1 def)\n");
              fprintf(stderr," v views : total number of views (32 def)\n");
              fprintf(stderr," W width : width of hologram (130 mm def)\n");
              fprintf(stderr," w width : width of image (130 mm def)\n");

```

```

                exit(0);
                break;
            }

if((argc-optind)<2) {
    fprintf(stderr,"usage: %s [-flags] image_views output file\n",pname);
    exit(-1);
}

if(num_views===-1)
    num_views = total_views;

/* this table is of precomputed offsets into viewzone fringe data */
/* only segment 0 is computed, the rest is replicated */
seglength = LIN_LEN/(total_views*portion*segment);
for(i=0;i<segment;i++)
    for(j=0;j<portion;j++) {
        segoffsets[i][j] = i*(portion*seglength) + j*seglength;
        fprintf(stderr,"segoffsets[%d][%d] %d\n",i,j,segoffsets[i][j]);
    }

if((fp = fopen(argv[optind],"r"))==NULL) {;
    fprintf(stderr,"unable to open %s for reading\n",argv[optind]);
    exit(-1);
}

if(*argv[2]=='-')
    fd = 1;
else
    if((fd = open(argv[optind+1],O_WRONLY|O_CREAT,0755))<0) {
        fprintf(stderr,"unable to open %s for writing\n",argv[optind+1]);
        perror("open");
        exit(-1);
    }

if(portion<1 || portion > MAX_PORTION) {
    fprintf(stderr,"portion of computed fringe must be >=1 <=%d\n",MAX_PORTION);
    exit(-1);
}
if(segment<1 || segment > MAX_SEGMENT) {
    fprintf(stderr,"segment must be >=1 <=%d\n",MAX_SEGMENT);
    exit(-1);
}

if((outmem = (char *) malloc(LIN_LEN*sizeof(int)))==NULL) {
    fprintf(stderr,"Couldn't malloc space for outmem\n");
    exit(-1);
}
if((images = (char *) malloc(view_width*view_height*num_views))==NULL) {
    fprintf(stderr,"Couldn't malloc space for images\n");
    exit(-1);
}
if((mem = (double *) malloc(LIN_LEN*sizeof(double)))==NULL) {
    fprintf(stderr,"Couldn't malloc space for mem\n");
    exit(-1);
}

/* seek ahead into images file if we don't need all the images */
if(total_views!=num_views)
    fseek(fp,(first_view*view_width*view_height),0);

/* read the perspective views into 'images' */
for(i=0;i<num_views;i++)
    fread((images+i*view_width*view_height),view_width,view_height,fp);

```

```

/* convert grating_width from millimeters to pixels */
if(grating_width != -1)
    grating_width = (grating_width * LIN_LEN) / holo_width;
fprintf(stderr,"grating_width: %g\n",grating_width);

/*
    calculate a line of fringe data based on a point in the
    middle of the image plane.  determine the min and max
    angles and then indexes
*/

/* THIS TABLE CAN BE PRECOMPUTED AND STORED IN A TABLE!! */

/* these values are computed based on the HOLO_SF_MAX and HOLO_SF_MIN
    defines set above.  these index values represent the maximum extent
    of a diffracted beam at a distance of image_depth */
/* min_ix and max_ix are based on LIN_LEN.  the length may actually be
    be larger than LIN_LEN */
min_ix = LIN_LEN/2 - tan(asin(HOLO_SF_MAX*LAMBDA+sin(HOLO_TH_REF*M_PI/180.0)))
    *image_depth * LIN_LEN/holo_width;
max_ix = LIN_LEN/2 - tan(asin(HOLO_SF_MIN*LAMBDA+sin(HOLO_TH_REF*M_PI/180.0)))
    *image_depth * LIN_LEN/holo_width;

/* allocate space for the fringe table */
sp_len = max_ix - min_ix;
if((sp_mem = (double *) malloc(sp_len*sizeof(double)))==NULL) {
    fprintf(stderr,"Couldn't malloc space for sp_mem\n");
    exit(-1);
}
dfprintf(stderr,"min_ix: %d max_ix: %d sp_len: %d\n",min_ix,max_ix,sp_len);

/* the fringe table actually consists of a sequence of cosinusoidal
    gratings of increasing spatial frequency.  this is an approximation
    of the real fringe pattern that would be generated by a point in
    space interfering with a plane wave.  it works quite well.

    the method of computation is to start with an initial spatial frequency
    and fill the array with some number of cycles.  at this point the
    spatial frequency for the new location is checked and the process
    repeated.
*/
/* set up the initial period, use a cntr, do one cycle and then recompute
    period */
sf = (sin(atan(( LIN_LEN/2 - min_ix )/(image_depth * LIN_LEN/holo_width)))-
    sin(HOLO_TH_REF*M_PI/180.0))/LAMBDA;

if(grating_width===-1) {
    cnt = LIN_LEN/(holo_width * sf);
    if(cnt<MIN_STEPS)
        cnt=MIN_STEPS;
}
else
    cnt = (int)grating_width;

/* don't combine these two, precision is lost */
step = 2 * M_PI / (LIN_LEN/(holo_width * sf));
dfprintf(stderr,"sf: %g cnt: %d step: %f\n",sf,cnt,step);

phi = 0.0;
for(i=0;i<=sp_len;i++) {
    sp_mem[i] = cos(phi);
    if(--cnt==0) {
        sf = (sin(atan(( LIN_LEN/2 - (i+min_ix) )/(image_depth * LIN_LEN/holo_width)))-

```

```

        sin(HOLO_TH_REF*M_PI/180.0))/LAMBDA;
if(grating_width==-1) {
    cnt = LIN_LEN/(holo_width * sf);
    if(cnt<MIN_STEPS)
        cnt=MIN_STEPS;
    }
else
    cnt = (int)grating_width;
dfprintf(stderr,"sf: %g cnt: %d\n",sf,cnt);

/* don't combine these two, precision is lost */
step = 2 * M_PI / (LIN_LEN/(holo_width * sf));
}
phi+=step;
}

/* build an array of pointers into the fringe data.  each point for
each view has a different offset value, but they ALL come from this
fringe data. */
for(i=0;i<total_views;i++) {
    double view_pos;
    double point_pos;
    double view_SF;

    /* remember, 0 is center.. */
    view_pos = (i - (total_views -1)/2) * holo_width / total_views;
    for(j=0;j<view_width;j++) {
        point_pos = (j - (view_width -1)/2) * image_width / view_width;
        view_SF = (atan((point_pos-view_pos)/image_depth) - sin(HOLO_TH_REF*M_PI/180.0))
            /LAMBDA;
        offsets[i][j] = LIN_LEN/2 - tan(asin(view_SF*LAMBDA+sin(HOLO_TH_REF*M_PI/180.0)))
            *image_depth * LIN_LEN/holo_width - min_ix - LIN_LEN/(2*total_views);
        dfprintf(stderr,"view_pos: %g point_pos: %g view_SF %g\n",view_pos,point_pos,view_SF);
        dfprintf(stderr,"    offsets[%d][%d]: %d\n",i,j,offsets[i][j]);
    }
}

/* compute one hololine at a time */
for(i=0;i<view_height;i++) {
    int empty_line;
    int lin_begin;
    int lin_end;

    empty_line = 1;
    fprintf(stderr,"doing line %d\n",i);
    /* clear the old line data */
    bzero((char*)mem,LIN_LEN*sizeof(double));

    /* select the right line of the image (index off this for other views */
    line = images + i * view_width;

    /* start filling in the fringe data in view 0 and copy below.. */
    lin_begin = first_view * LIN_LEN/total_views;

    /* make sure that the end doesn't run on more than the subview.. */
    lin_end = (LIN_LEN / (total_views));

    /* do each view zone of this line */
    for( j = first_view; j < (num_views+first_view); j++ ) {
        dfprintf(stderr,"doing view %d\n",j);

```

```

/* track the number of  on pixels in this view */
cnt = 0;

for( k = 0; k < view_width; k++ ) {

    if(line[k]<=HOLO_IMG_THRESH) continue;
    cnt++;
    empty_line = 0;
    for( l = 0; l < segment; l++ ) {

        base = &(sp_mem[(offsets[j][k] +segooffsets[l][0])]);
        begin = lin_begin + segooffsets[l][0];
        if((offsets[j][k] +segooffsets[l][0]+seglength)>sp_len)
            end = begin + (sp_len - (offsets[j][k] + segooffsets[l][0]));
        else
            end = begin + seglength;
        /* check to make sure scale is having some effect */
        for( m = begin; m < end; m++ )
            mem[m] += base[m-begin] * line[k] / 255;
    }
}

/* if there was actually any image here do the following find
   the minimum and maximum values */
if(cnt!=0) {
    min = 1000000000;
    max = -1000000000;
    for( l = 0; l < segment; l++ ) {
        int begin,end;
        begin = lin_begin +segooffsets[l][0];
        end = begin + seglength;

        for( m = begin; m < end; m++ ) {
            if(mem[m]>max) max = mem[m];
            if(mem[m]<min) min = mem[m];
        }
    }
    sum = max - min;

    fprintf(stderr," min %d max %d\n",min,max);

    for( l = 0; l < segment; l++ ) {
        int begin,end;

        begin = lin_begin +segooffsets[l][0];
        end = begin + seglength;

        /* quantitize image data */
        for( m = begin; m < end; m++ )
            mem[m] = intensity * (mem[m] - min) / sum + HOLO_OUT_OFFSET;
    }

    /* replicate subviews */
    if(blank==0) {
        fprintf(stderr,"replicate subviews ");
        for( l = 0; l < segment; l++ ) {
            for( m = 1; m < portion; m++ ) {
                int begin,end;
                int begin2,end2;
                begin = lin_begin +segooffsets[l][0];
                begin2 = lin_begin +segooffsets[l][m];
                fprintf(stderr," segment %d portion %d begin %x begin2 %x seglenth %x\n",
                    l,m,begin,begin2,seglength);
                for( n = 0; n < seglength; n++ )
                    mem[begin2++] = mem[begin++];
            }
        }
    }
}

```

```

        }
    }
}

lin_begin += (LIN_LEN / total_views);
lin_end += (LIN_LEN / total_views);
/* move down one whole image */
line += view_width * view_height;
}

offset = i%3;
/* clear out outmem if we're starting a new group of 3 lines */
if(offset == 0 )
    bzero(outmem,LIN_LEN*sizeof(int));

/* scale pixel and place in appropriate offset.  don't bother
   if count is zero though.. */
if(!empty_line) {
    /* swap line around opposite swaths */
    if((i/18)&0x01)
        for( j = 0; j < (LIN_LEN); j++ ) {
            outmem[j*4+offset] = (char)((mem[LIN_LEN - j - 1])) & 0xff;
        }
    else
        for( j = 0; j < (LIN_LEN); j++ ) {
            outmem[j*4+offset] = (char)((mem[j])) & 0xff;
        }
}
if(offset==2)
    write(fd,(char*)outmem,sizeof(int)*LIN_LEN);
}
close(fd);
}

```

Bibliography

- [1] D.L. Arias. *Design of an 18 channel framebuffer for holographic video*. Sb thesis, Massachusetts Institute of Technology, EECS Dept., Feb 1992.
- [2] H. H. Arsenault. *Optical Processing and Computing*. Academic Press, INC., 1989.
- [3] Enloe Burkhardt. Television transmission of holograms with reduced resolution requirements on the camera tube. *Bell System Tech. Journal*, XXX:1529–1535, May-June.
- [4] William J. Dallas. Computer generated holograms. In B.R. Frieden, editor, *The Computer in Optical Research, Methods and Applications*, chapter 6, pages 291–366. Springer-Verlag, 1980.
- [5] K. Hildebrand K. Haines E. Leith, J. Utpatnieks. Requirements for a wavefront reconstruction television facsimile system. *J. SMPTE*, 74:893–896, 1965.
- [6] Editor. Real-time 3d moving-image display. *Holography News*, 7(7):59,60, September 1993.
- [7] Rubinstein Enloe, Murphy. Hologram transmission via television. *Bell System Tech. Journal*, I:335–339, February.
- [8] Joseph W. Goodman. *Introduction to Fourier Optics*. McGraw-Hill Publishing Company, 1968.
- [9] Milton Gottlieb. *Electro-Optic and Acousto-Optic Scanning and Deflection*. Marcel Dekker, INC., 1983.
- [10] P. Hariharan. *Optical Holography*. Cambridge University Press, 1984.

- [11] Eugene Hecht. *Optics*. Addison-Wesley Publishing Company, 1987.
- [12] Mary Lou Jepsen. *Holographic Video: Design and Implementation of a Display System*. Master's thesis, Massachusetts Institute of Technology, June 1989.
- [13] Joel S. Kollin. *Design and Information Considerations for Holographic Television*. Master's thesis, Massachusetts Institute of Technology, June 1988.
- [14] Adrian Korpel. *Acousto-Optics*. Marcel Dekker, INC., 1988.
- [15] Mark Lucente. Private Communication, October 1993.
- [16] Mark Lucente. Interactive computation of holograms using a look-up table. *Journal of Electronic Imaging*, 2(1):28–34, January 1993.
- [17] Mark Lucente. *Diffraction-Specific Fringe Computation for Electro-Holography*. PhD thesis, Massachusetts Institute of Technology, September 1994.
- [18] L.M. Myers. The scophony system and analysis of its possibilities. *TV and Shortwave World*, pages 201–294, April 1936.
- [19] Johnson Richard. Scophony spatial light modulator. *Optical Engineering*, 24(1):93–100, January/February 1985.
- [20] Pierre St. Hilaire. *Real Time Holographic Display: Improvements Using Higher Bandwidth Electronics and a Novel Optical Configuration*. Master's thesis, Massachusetts Institute of Technology, June 1990.
- [21] Pierre St. Hilaire. Color images with the mit holographic video display, 1993.
- [22] Pierre St. Hilaire. *Scalable Optical Architectures for Electronic Holography*. PhD thesis, Massachusetts Institute of Technology, September 1994.
- [23] G. Tricoles. Computer generated holograms: an historical review. *Applied Optics*, 26(20):4351–4360, October 1987.
- [24] J.S. Underkoffler. *Toward Accurate computation of optically reconstructed holograms*. Master's thesis, Massachusetts Institute of Technology, June 1991.

- [25] Jr. V. M. Bove and J. A. Watlington. Cheops: A data-flow processor for real-time video processing. *MIT Media Laboratory Technical Memo*, 1993.