High-quality, real-time 3D video visualization in head mounted displays

by

Tyler Ray Bell

A thesis submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Major: Human Computer Interaction

Program of Study Committee:

Song Zhang, Major Professor

James H Oliver

Jonathan Kelly

Iowa State University

Ames, Iowa

2014

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGEMENTS

I would like to take this opportunity to express my greatest gratitude to everyone who has helped me throughout my time here at Iowa State University and throughout my thesis research. First and foremost I'd like to thank my advisor Dr. Song Zhang for allowing me the privilege to join his group to conduct not only innovative, but exciting, research. I'd also like to thank him for holding me to the highest of standards as it not only made me better as a researcher but also as a person. Much appreciation is also held for my committee members Dr. Jim Oliver and Dr. Jonathan Kelly. They not only inspired me to do my best work but also were the source of many great ideas for research. Further, this thesis would not have been possible without the help of former and current members of the research group: Nik Karpinsky, William Lohry, Beiwen Li, Laura Ekstrand, Chen Gong, and Yajun Wang. A thank you should also be given to any and all educators who have helped me in some way throughout my life. I do not believe enough recognition is given to those who quite literally help shape the future.

A thank you should go to all of my friends, my family, my grandparents, and especially my parents, Tim and Ericka, for an outstanding upbringing in which I was encouraged, supported, and most of all, loved wholly during any of my ventures. Also, a thank you to each of my beautiful sisters, Madison and Haley. Last, but certainly not least, I'd like to thank my wonderful girlfriend Melanie for keeping me focused, putting up with me during my writing, and for driving 45 minutes here and back just to bring me dinner and take a short walk on those long nights. To all of my family and friends, I love you.

ABSTRACT

The main goal of this thesis research was to develop the ability to visualize high-quality, three-dimensional (3D) data within a virtual reality head mounted display (HMD). High-quality, 3D data collection has become easier in past years due to the development of 3D scanning technologies such as structured light methods. Structured light scanning and modern 3D data compression techniques have improved to the point at which 3D data can be captured, processed, compressed, streamed across a network, decompressed, reconstructed, and visualized all in near real-time. Now the question becomes what can be done with this live 3D information?

A web application allows for real-time visualization of and interaction with this 3D video on the web. Streaming this data to the web allows for greater ease of access by a larger population. In the past, only two-dimensional (2D) video streaming has been available to the public via the web or installed desktop software.

Commonly, 2D video streaming technologies, such as Skype, FaceTime or Google Hangout, are used to connect people around the world for both business and recreational purposes. As the trend continues in which society conducts itself in online environments, improvements to these telecommunication and telecollaboration technologies must be made as current systems have reached their limitations. These improvements are to ensure that interactions are as natural and as user-friendly as possible.

One resolution to the limitations imposed by 2D video streaming is to stream 3D video via the aforementioned technologies to a user in a virtual reality HMD. With 3D data, improvements such as eye-gaze correction, obtaining a natural angle of viewing, and more can be accomplished. One common advantage of using 3D data in lieu of 2D data

is what can be done with it during redisplay. For example, when a viewer moves about their environment in a physical space while on Skype, the 2D image on their computer monitor does not change; however, via the use of an HMD, the user can naturally view and move about their partner in 3D space almost as if they were sitting directly across from them.

With these improvements, increased user perception and level of immersion in the digital world has been achieved. This allows users to perform at an increased level of efficiency in telecollaboration and telecommunication environments due to the increased ability to visualize and communicate more naturally with another human being. This thesis will present some preliminary results which support the notion that users better perceive their environments and also have a greater sense of interpersonal communication when immersed in a 3D video scenario as opposed to a 2D video scenario. This novel technology utilizes high-quality and real-time 3D scanning and 3D compression techniques which in turn allows the user to experience a realistic reconstruction within a virtual reality HMD.

CHAPTER 1. INTRODUCTION

This chapter serves as a general introduction to this thesis work. In this chapter, the motivations of this research will be given, related state-of-the-art research technology will be described, objectives which this thesis research hopes to accomplish will be presented, and the organization of the thesis body will be explained.

1.1 Motivation

Telepresence, a term coined by Minsky (1980), is the ability for a user to feel as though they are present in a location that differs from their own current physical location. This phenomena can take form through the transfer of one or more of one's senses (such as seeing or hearing) through some medium. That being said, a sense is captured, transmitted, and reconstructed in a different location.

Consider, for example, the case of the telephone: one user's voice is captured, transmitted, and reconstructed on the other end of the line making their voice present for the receiver; the voice on the other end has the same process done on it. The interaction of these two users gives them a sense of telepresence; that is, they are interacting in some sort of *virtual* environment that is different than either of their own physical ones. A more recent case in which users may feel telepresence is when they are communicating with one another via the usage of live 2D video (e.g., Skype and FaceTime).

Modern telecommunication systems primarily rely on the streaming of 2D video across the internet such that two or more individuals may visualize one another in real time.

Although enabling, this technology has its limitations such as a loss of eye contact (Karpinsky, 2013c) and a lack of depth information. A claim can then be made that these challenges affect a user's sense of immersion and the system's overall usability. For such reasons, research involving the usage of 3D video within telecommunication systems has started to increase. 3D video not only provides a higher fidelity of information but also opens many different avenues for the data to be visualized and interpreted; thus, a heightened sense of telepresence may be achievable. The motivation of this thesis research is to improve a user's sense of immersion and level of interaction in 3D video visualization environments. The following section will detail current state-of-the-art research in this area.

1.2 State of the Art

Recently there has been research done to improve telepresence for users, especially in the area of 3D telecommunications. The following provides a short description of several current state-of-the-art research projects dealing particularly with real-time 3D telecommunications that are closely related to this thesis research.

1.2.1 Blue-c

Blue-c is a system which was developed by Gross et al. (2003) that is a hybrid platform for capturing and displaying 3D video. The system featured three projection screens that can be opaque or transparent. When in the opaque mode, the screens catch a projector's light on which a 3D virtual reality scene is displayed. When in transparent mode, the user in the system's geometry is able to be captured by an array of cameras. As mentioned, the system consisted of three of these special projection screens which are comprised of glass panels and liquid crystal layers (Gross et al., 2003). Blue-c used 16 cameras, 11 of which were outside of the three displays, to acquire a 3D scan of the

object or user within the system. The 11 cameras on the outside of the glass projection screens could see inside of the system when it is in its transparent mode. The other 5, situated on the four corners of the system with one on the ceiling, could see inside at all times. Blue-c's computer clusters processed the frames from the cameras in parallel. The system also included three projectors; one for each of the screens. Blue-c required its user to wear a pair of shutter glasses as the projectors were synchronized to display images to the user's eyes one at a time. This is done at speeds fast enough to trick the user's brain into thinking it is viewing each image at the same time; thus allowing for a natural sense of sight and therefore depth perception. The system's 3D video pipeline reconstructed video inlays of 25k points at 5 fps or 15k points at 9 fps (Gross et al., 2003).

What this system is able to accomplish is a sense of realistic immersion as physical depths should translate into the virtual world appropriately due to their nature of capture. One advantage this system has is its ability to place the user within Blue-c into the virtual world thus successfully bridging a gap between the physical and digital worlds. One of its demonstrations of technology is mirror-like in nature and allows the participant within the system to view their own 3D geometry. Another demonstration takes the participant's 3D geometry and displays it in a virtual environment; thus, another example of the gap between the physical and virtual world being bridged. A challenge that Blue-c faced was the slow rate at which it was able to visualize geometry. Another is in the lower quality scans that are a result of stereo reconstruction.

1.2.2 One-to-Many

A system for achieving eye contact in a 3D video teleconferencing system was developed at the University of Southern California's Institute for Creative Technologies. In their paper, Jones et al. (2009) detailed their system which consisted of the ability to stream 3D video of a remote participant to an audience and stream 2D video of an

audience to the remote participant. The remote participant's 3D facial scan is taken at a rate of 30 fps via the usage of a structured light scanner. The scan's depth map image and its corresponding texture image are then streamed over a local area network (LAN) to be displayed. The depth map image is downsampled from a pixel resolution of 640 $\times$ 480 to 80 $\times$ 60 such that it is easier to render on their 3D display. The device which renders the captured 3D data is an autostereoscopic 3D display that was based on Jones et al. (2007). The display is comprised of rapidly rotating mirrors that reflects light to the audience in such a way that the 3D data appears to be holographic.

There is a camera that is capturing the audience, as well. The images captured are 2D in nature but serve multiple purposes. First, it gives the remote participant a view of their audience; at this point both parties can now view one another. Second, the camera's 2D frames are processed such that the facial positions of audience members are calculated; these positions are then used to render the correct vertical parallax of the virtual head to the audience members (Jones et al., 2009).

The advantages to a system such as this is that it allows for real-time streaming of 3D video to a different location at which a set of users can each see their own correct 3D view of the remote participant's geometry. The major advantage of this system is that in this 3D view a correct sense of eye contact is achieved.

The challenge that this system faces, as described in Jones et al. (2009), is that the scans produced by the structured light scanner must be downsampled. Also, the data is only being streamed across a LAN and thus a setup such as this may not be usable outside of the same building; compression of the 3D geometry may be one way to address this limitation. Some of these challenges are addressed by more recent research called Portal-s.

1.2.3 Portal-s

Another teleconferencing system entitled Portal-s, developed at Iowa State University by Karpinsky (2013c), allowed users to participate in real-time 3D video conferencing while maintaining a sense of eye contact. The Portal-s system consists of an algorithmic software pipeline and hardware architecture (Karpinsky, 2013c). The goal of Portal-s was to capture, transmit, and redisplay 3D information in real time over an internet connection with the redisplay taking place on another remote Portal-s node. Data would be captured by one Portal-s node using an infrared (IR) two-camera, one-projector structured light scanner; processed and compressed via the Holovideo compression technique (Karpinsky and Zhang, 2012b); and then transmitted to the web via HTML5's communication protocols. Another Portal-s client could then be listening to receive the compressed 3D scan at which point it would decompress the holoencoded image and redisplay the original 3D geometry. Realizing speeds of 30 3D fps is possible due to each step in the pipeline being performed in parallel while extensively utilizing a node's graphics processing unit (GPU). The result is a real-time 3D video conferencing system in which two users can stream 3D geometry of themselves to a Portal-s client and view 3D geometry of the other Portal-s node.

The advantages of such a technology is that it does not require any additional hardware to achieve its correct sense of eye contact for its users as they are 3D scanned through the same optical axis at which the display is viewed from. As mentioned, the original Portal-s system was built using an IR structured light scanner; the result of using such hardware technologies is a more comfortable experience for users as they avoid bright light from a visible light projector. However, the resulting scan quality is lower when using this system setup due to a lack of lighting intensity of the projector used, but this could be addressed by using a visible light Portal-s scanner (Karpinsky, 2013c). Further, one of the challenges the Portal-s system faces is its complex hardware architecture required to achieve a correct sense of eye contact. Portal-s will be discussed in

depth with each step in the pipeline explained further in Section 2.3.

1.3 Objectives

The main objectives of this research were to achieve 3D video display within an HMD and to gauge a user's sense of telepresence within such a system versus their experience with traditional 2D streaming systems. The level of research being performed in the usage of 3D video within telecommunication fields is exciting, and that is one of the reasons why the exploration for an immersive method to visualize the data is pertinent. Obtaining a high-quality 3D video is only the first step in the pipeline; the reconstruction and visualization is just as important as it offers others the ability to interpret for their own meaning what the data depicts. The first objective of this research was to visualize 3D video in an HMD in aims to improve a user's overall level of immersion and potential level of interaction within a telecommunication environment. The second objective of this thesis research was to conduct an exploratory user study to gauge whether or not a visualization of 3D video within an HMD would provide a better sense of telepresence for users versus their experience with traditional 2D video streaming technologies.

1.4 Thesis Organization

This thesis is organized as follows: Chapter 2 introduces the basic principles behind obtaining high-quality, real-time 3D video via the usage of a structured light scanner. It also provides an in depth discussion of the Portal-s system and the 3D data compression which makes real-time 3D streaming possible. Chapter 3 briefly introduces how we, as humans, perceive depth. HMD technologies and how a human's depth cues can be exploited to simulate a 3D world will also be discussed. Chapter 4 details research in visualizing high-quality 3D video within a particular HMD. Chapter 5 presents preliminary results and a discussion on an exploratory user study which was conducted to

gauge users' levels of immersion in a 3D video system utilizing an HMD. Lastly, Chapter 6 summarizes the thesis and discusses directions for future work.

CHAPTER 2. FUNDAMENTALS OF REAL-TIME 3D VIDEO TELEPRESENCE

Before 3D video can be visualized within an HMD, 3D data must first be captured. A modern advancement in 3D scanning, real-time structured light scanning, provides a high-quality of information at high quantities and fast rates (Zhang, 2010). To appropriately store and enable transmission of this data in real-time, a 3D data frame can be compressed to a 2D image via the Holovideo encoding technique (Karpinsky and Zhang, 2012b). Upon retrieval at a later date, potentially by a remote client, 3D geometry can recovered via Holovideo decoding. This chapter details the ideas behind structured light scanning as it relates to this thesis, gives an in-depth description of the Holovideo compression technique, and provides a review of the Portal-s pipeline as developed by Karpinsky (2013c). It should be explicitly noted that the material covered in this chapter is not original to this thesis research and is being covered within the context of providing a foundation for the following chapters.

2.1 Structured Light Scanning

A structured light scanning system is a type of 3D capturing system which commonly uses one projector and one camera to gather 3D information. The projector is used to project special coded images onto the object to be scanned; these images provide clues for easier correspondence establishment. These images are then captured, from a different angle, by the system's camera. The locations of the correspondence clues in the original

projected images, which are known by the system, differ from the location of the clues as found in the camera's captured images. This difference is due to the different angles of projection and capture as well as due to the geometry of the object distorting the light pattern (or structure) that's being projected. The structured light system can better be understood by its representation in Figure 2.1. If the structured light system is calibrated then the physical relationship between the camera and projector is known. Triangulation can then be used to calculate an object's 3D geometry (Zhang and Huang, 2006). Modern systems, known as real-time systems, can reach speeds of over 30 3D fps at a resolution of 640 × 480 (Karpinsky and Zhang, 2012a).

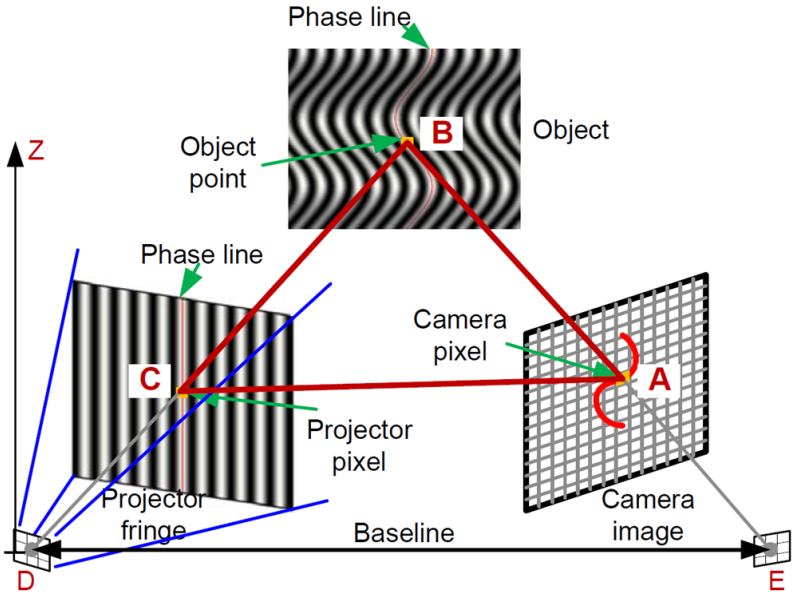


Figure 2.1 Structured light scanner system layout. (Modified from Zhang (2010). With permission.)

2.1.1 Phase-shifting Algorithm

As mentioned, images are projected onto the object which is to be captured, and these images contain correspondence information such that triangulation requires. There are two specific types of images containing these points which can be projected: discrete and continuous (Salvi et al., 2004). For the sake of this thesis, continuous methods will

be discussed as they were the ones used to gather data. In general, however, discrete images can be used and projected through a defocused projector resulting in a continuous image (Bell et al., 2014). One way to encode an object to be captured is to project three phase-shifted fringe images onto it (Zhang, 2010). Fringe images are special structured light patterns and the three phase-shifted images used can be derived such that

$$I_1(x,y) = I'(x,y) + I''(x,y)\cos\left(\phi - \frac{2\pi}{3}\right), \tag{2.1}$$

$$I_2(x,y) = I'(x,y) + I''(x,y)\cos\left(\phi\right), \tag{2.2}$$

$$I_3(x,y) = I'(x,y) + I''(x,y)\cos\left(\phi + \frac{2\pi}{3}\right), \tag{2.3}$$

where $I'(x,y)$ represents the average intensity, $I''(x,y)$ the intensity modulation, and $\phi(x,y)$ the phase to be found.

2.1.2 Phase Wrapping

The three images above are projected unto the object to be captured and the result is a set of three images captured by the system's camera. To derive 3D geometry the next step is to simultaneously solve each equation via an arctan function given by

$$\phi(x,y) = \tan^{-1}\left[\frac{\sqrt{3}(I_1 - I_3)}{2I_2 - I_1 - I_3}\right]. \tag{2.4}$$

The result of the equation provides what is referred to as the wrapped phase map. Due to the nature of the arctan function, the wrapped phase has a range from 0 to 2π with 2π discontinuities. A texture image can also be obtained while processing the fringe captures by averaging them with each other. For example, to obtain a texture image with three phase shifted fringe images projected, the associated equation is simply

$$I'(x,y) = \frac{I_1 + I_2 + I_3}{3}. \tag{2.5}$$

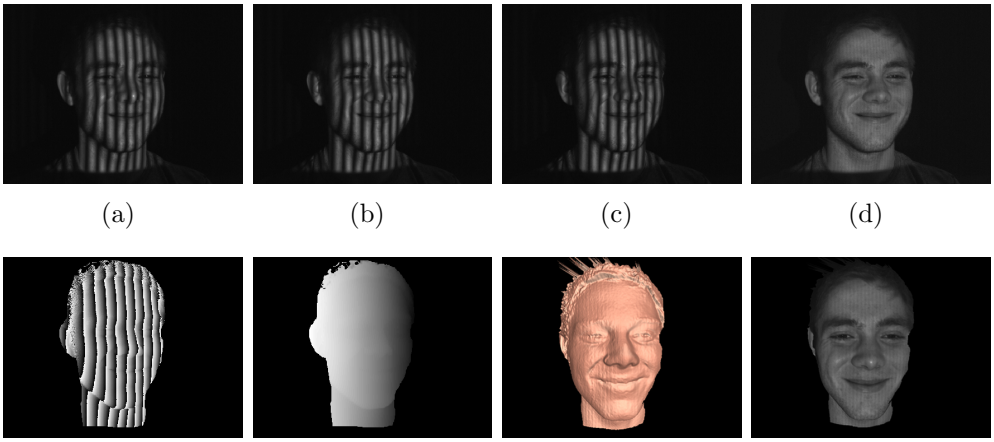
2.1.3 Phase Unwrapping

To address the aforementioned discontinuities in the phase map ϕ, phase unwrapping algorithms can be adopted (Ghiglia and Pritt, 1998). These algorithms detect the 2π discontinuous locations and add integer multiples of 2π at points along the phase map where these discontinuities exist. The result of this method is referred to as the unwrapped phase map; this unwrapped phase map which can be given mathematically by

$$\Phi(x, y) = \phi(x, y) + 2\pi \times K(x, y), \tag{2.6}$$

where Φ represents the unwrapped phase to be solved and $K(x, y)$ represents the integer to be used to multiply 2π with such that discontinuities along the wrapped phase map may be fixed. At this point in the process, 3D geometry may be reconstructed if the system has properly been calibrated (Zhang and Huang, 2006). If the calibration parameters are known then triangulation given the unwrapped phase map may occur and location within 3D space for each point can be derived. Figure 2.2 provides an example of each step in the capture process as different fringe images are projected, wrapped and unwrapped phase is obtained, and finally 3D geometry is constructed.

As claimed before, the ability to project fringe images, process the data, and reconstruct the object's 3D geometry can now happen at real-time speeds of 30 fps or above. This level of speed is great for applications which would require rapid acquisition, yet the size of the data is simply too large to store, let alone transmit, in real-time. For example, the system mentioned previously that can operate at 30 fps with a resolution of 640 $\times$ 480 generates 3D data at over 110 MB per second (Karpinsky, 2011). Recent work has been done by researchers at Iowa State University (Karpinsky, 2011; Karpinsky and Zhang, 2012b) to compress this 3D data into a viable 2D image format substantially reducing the data size.



(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 2.2 The capture pipeline resulting in reconstructed 3D geometry. Three phase shifted fringe images I_1, I_2, and I_3, (a), (b), and (c) respectively, are projected onto and captured on the object's surface. The averaging of (a), (b), and (c) result in the texture image (d) as given in Equation 2.5. The wrapped phase map as derived by Equation 2.4 is shown in (e) which is then used to obtain the unwrapped phase map, (f), through a phase unwrapping algorithm. 3D geometry can be reconstructed as shown in (g) via triangulation utilizing the system's calibration properties. 3D geometry from (g) with texture from (d) can be seen in (h).

2.2 Holovideo Compression

In order to efficiently store or transmit high quality data coming from a structured light scanner in real-time, 3D data should be compressed quickly and at high compression ratios. This section presents the concepts from 3D data compression techniques which were originally detailed in (Gu et al., 2006; Karpinsky, 2011; Karpinsky and Zhang, 2012b). These techniques allow for high-quality 3D data to be compressed into a single 2D image. This image can later then decompressed to reconstruct 3D geometry.

2.2.1 Holovideo Principle

To generate such an image, a virtual structured light scanner is built as it makes 3D shape reconstruction simple yet accurate. This virtual structured light scanner is comprised of a (virtual) projection device and a (virtual) camera device as real structured light scanners are. The virtual system, set up in the Open Graphics Library (OpenGL) for example, has its projector project required structured images onto the virtual object which are captured by the individual color channels of the virtual camera (Karpinsky and Zhang, 2012b).

Again, this virtual scanning system is theoretically similar to the structured light system (shown in Figure 2.1) and can be seen in Figure 2.3. The virtual system allows for more control as everything can be user defined. Example benefits of such control allows for the ability to control lighting conditions, the ability to precisely define the angle, θ, between the virtual projection and virtual capture device, and the ability to use orthogonal devices in lieu of perspective ones as this will simplify the reconstruction process (Karpinsky and Zhang, 2010). Another benefit to this virtual system is it alleviates the normal necessity to calibrate the projector and camera due again to the fact that everything is virtually controlled and thus can be exact. The goal of compression is to encode each point on the geometry to be saved such that its value can be stored in

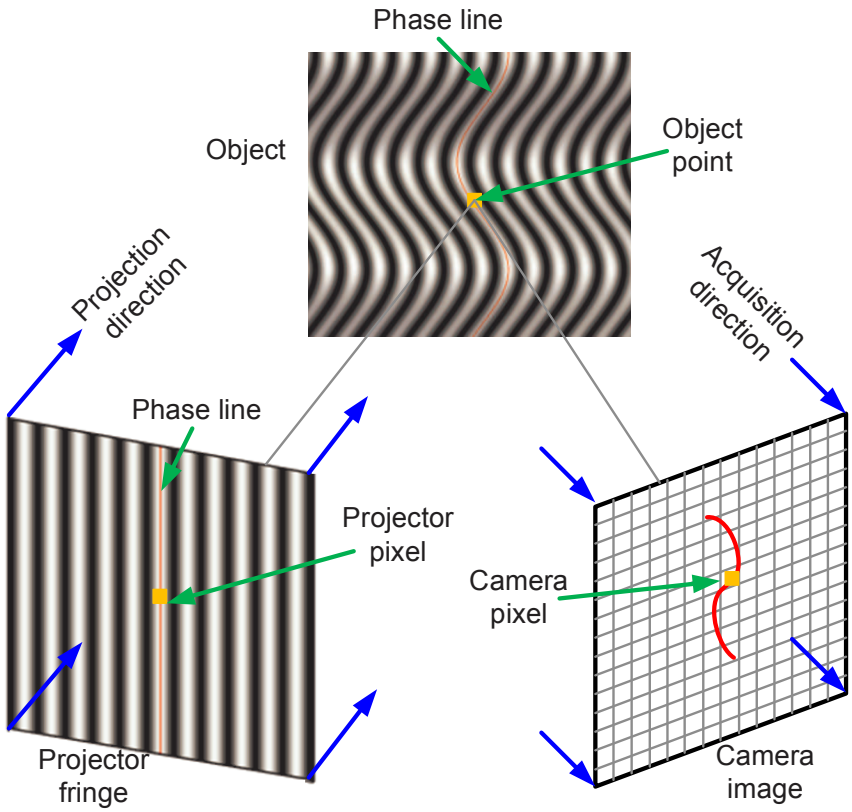


Figure 2.3 The virtual structured light scanning system. Notice that the projection directions and capture directions are orthogonal and not perspective based. (Modified from Karpinsky and Zhang (2010). With permission.)

a 2D image. To save data for each point to a 2D image, each of the red, green, and blue (RGB) color channels of the compressed image are used with each pixel being encoded via

$$I_r(x,y) = 0.5 + 0.5\sin\left(\frac{2\pi x}{P}\right), \tag{2.7}$$

$$I_g(x,y) = 0.5 + 0.5\cos\left(\frac{2\pi x}{P}\right), \tag{2.8}$$

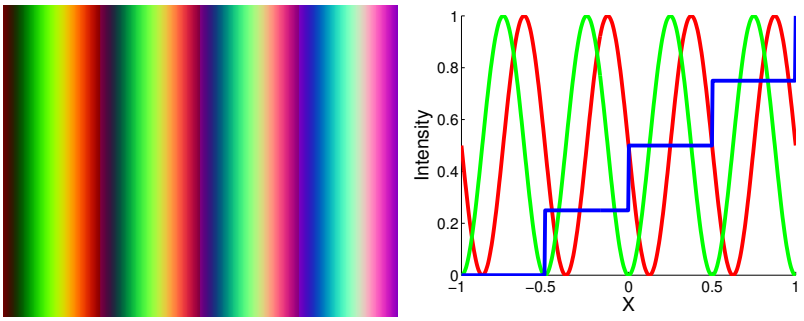
$$I_b(x,y) = S \times floor(\frac{x}{P}) + \frac{S}{2} + \frac{S-2}{2} \times \cos\left(2\pi \times \frac{Mod(x,P)}{P_1}\right), \tag{2.9}$$

where P represents the fringe pitch (number of pixels per fringe stripe), where $P_1 = \frac{P}{K+0.5}$ is the local fringe pitch and K an integer value, where S represents the stair height, and where $Mod(a,b)$ represents the modulus operator used to get a over b (Karpinsky and Zhang, 2012b). It should be mentioned that before 3D geometry is converted to 2D using Holovideo principles, 3D geometry is normalized to unit cube coordinates. The virtual camera, in this case the computer's screen, will then "capture" an image of the object as it was encoded by a virtual structured pattern. An example of this pattern can be seen in Figure 2.4. This encoding can be performed quickly at the parallel level on the GPU by simultaneously encoding many pixels of the resulting image. Each 2D pixel in the resulting encoded image, known as a Holovideo encoded image, contains information required to recover 3D geometry.

When the time comes to reconstruct 3D geometry from the holoencoded image, the image passes through the decoding pipeline as shown in Figure 2.5. Each point in the image contains its own phase information which can be extracted via Equation 2.10.

$$\Phi(x,y) = 2\pi \times floor\left(\frac{I_b - \frac{S}{2}}{S}\right) + \tan^{-1}\left(\frac{I_r - 0.5}{I_g - 0.5}\right). \tag{2.10}$$

where $\Phi(x,y)$ represents the unwrapped phase map as it is computed pixel by pixel and thus can be performed in parallel, I_r, I_g, and I_b correspond to the values of the current



(a) (b)

Figure 2.4 (a) Ideal Holovideo virtual structured light pattern to be projected by the virtual structured light scanner; (b) Single cross section of (a).

pixel's different color channels as encoded in Equations 2.7-2.9, and the other variables' or functions' descriptions remain unchanged from previous equations (Karpinsky and Zhang, 2012b).

As mentioned earlier, the virtual scanner's devices are orthogonal in nature and values obtained are rendered within a unit cube (Karpinsky, 2011). To obtain original geometry, normalized coordinates are obtained and then scaled, by some scaling factor S_c, back to their original dimensions. The coordinates are then repositioned relative to their original center by translating the geometry along 3D axis via C_x, C_y, and C_z.

From the unwrapped phase map, normalized point coordinates for this cube can be derived via

$$x^n = \frac{i}{W}, \tag{2.11}$$

$$y^n = \frac{j}{W}, \tag{2.12}$$

,

$$z^n = \frac{P\Phi - 2\pi i \cos(\theta)}{2\pi W \sin(\theta)}, \tag{2.13}$$

where P again represents the fringe pitch, (i, j) the index of the pixel currently being processed, θ the angle between the camera and projector optical axis, and W the number of pixels in the x direction if the fringe stripes are vertical (Karpinsky and Zhang, 2012b).

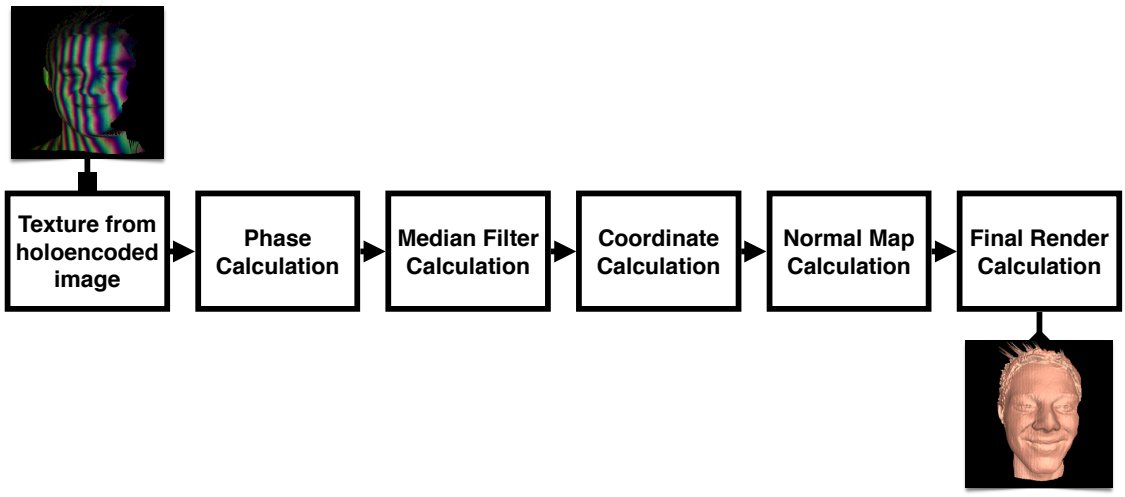


Figure 2.5 The Holovideo decoding pipeline. Starting with a compressed, holoencoded image, a GLSL texture is created. From there, a multi-pass approach is taken as the texture gets passed from shader to shader before the 3D geometry is reconstructed.

From these normalized points, original 3D coordinates can now be recovered via

$$x = x^n \times S_c + C_x, \tag{2.14}$$

$$y = y^n \times S_c + C_y, \tag{2.15}$$

$$z = z^n \times S_c + C_z, \tag{2.16}$$

where S_c is the scaling factor used to normalize the geometry and C_x, C_y, and C_z the geometry's original center in 3D space (Karpinsky and Zhang, 2012b). As with encoding, this decoding can be performed pixel by pixel and thus in parallel on the GPU. This allows Holovideo encoding and decoding to each be performed very efficiently by utilizing a parallel processing architecture.

2.2.2 Holovideo Implementation

The general idea is to have the virtual system project the generated virtual structured light pattern (as shown in Figure 2.4) onto the object in the scene, render the scene out to a texture, and finally retrieve, then save, the texture from the GPU. As this process can be done in pixel by pixel, it can be performed in parallel; the encoding takes place in the fragment shader which gets passed the x value from the respective vertex shader and the fringe pitch, P, from a uniform variable. Each fragment's output color is then rendered via the calculation of Equations 2.7, 2.8, and 2.9. After this has been done in parallel for every pixel in the image, the encoded image is rendered as a texture, fetched from the GPU by usage of direct memory access (DMA) and saved as an image (Karpinsky and Zhang, 2012b).

The resulting 2D image contains all information necessary to reconstruct the original 3D geometry. In the Portable Network Graphics (PNG) 2D image format there are four color channels available: R, G, B and alpha (A). As described so far, a holoencoded image contains data in the RGB. The Holovideo encoded image can also store texture information from the 3D frame's corresponding texture image, as derived in Equation 2.5,

in the output pixel's alpha channel if a 4-channel RGBA PNG image is being used. Overall, the result of Holovideo compression is a 2D image which represents high-quality 3D geometry and potentially its associated texture. This compression makes it more viable to work with such data now. The work of Karpinsky and Zhang (2012b), for example, claimed impressive compression ratios of 134 : 1 versus the OBJ file format for a 3D frame.

When it comes time to recover the 3D geometry from the compressed 2D image either at a later date or after the image has been transmitted to another machine, for example, decompression takes place. As each pixel contains its own phase information, the compressed image's pixels may be decoded to recover the phase map in parallel on a computer's GPU. This allows for rapid recovery of geometry. The decoding process again involves the usage of the OpenGL Shading Language (GLSL) and, as implemented, the decoding pipeline using a multi-pass approach with several different shaders being used (Karpinsky, 2011). Again, the high-level pipeline of this decoding process as developed by Karpinsky (2011) is shown in Figure 2.5.

The first shader that a Holovideo encoded image must be passed through, by being rendered to a texture, is the phase calculator. This fragment shader calculates phase value for each pixel as described in Equation 2.10. The output of this shader is a texture made up of colors, where each pixel's color represents its unwrapped phase, which is then used by other shaders later on in the decoding process (Karpinsky, 2011).

This phase map is then passed into the median filter shader which reads the phase map, does the filtering operation, and itself outputs another texture. The median filter shader is a fragment shader, and thus it will perform an operation for each pixel in the phase map texture. The operation that is performed in McGuire (2008)'s shader is a 3 $\times$ 3 median filter in which a pixel's value in the filtered phase texture being outputted is the median of its neighbors in the 3 $\times$ 3 kernel, relative to the currently being processed pixel's position, located within the phase map .

Taking in the texture after applying the median filter, the coordinate calculator shader's goal is to calculate the x, y, and z value for each pixel as described in Equations 2.11-2.13 where the texture's pixel coordinates are used for x and y in Equation 2.11 and 2.12 respectively and where P, θ, and W are passed into the fragment shader as constants for usage in the calculation of Equation 2.13. The resulting fragment color corresponds to the derived z value; the resulting texture represents the depth map and can then be used to calculate normals or for the final render (Karpinsky, 2011).

To calculate normals, a fragment shader will once again be used. Normal calculation averages adjacent surface normals to obtain a point normal for the pixel currently being calculated. To do this, a vector for each of the pixel's neighbors is placed into an array and the cross product is calculated for each consecutive index in the array; the last index has its value calculated in the crossed product with the first element. These calculations result in a surface normal for each of the pixel's neighbors which are then averaged and normalized. The resulting point normal is then sent along as the fragment color with the derived shader texture representing the calculated normal map (Karpinsky, 2011).

At this point in the pipeline a coordinate map and a normal map have been calculated. The last shader is the final render shader, and it renders a point in 3D space for each pixel in the original compressed image. Each point's z value is set based on the calculated coordinate map to correctly place the point and Phong shading is used along with the calculated normal map to correctly render each pixel. It should be noted that certain points are discarded from the final render if its corresponding pixel in the original image is zero in its RGB channels, representing the bad or background points. It should also be noted that in the case in which a 4-channel PNG image is used, the 2D texture associated with 3D geometry is used to color a pixel in the final rendering shader: instead of the normal map determining the fragment color, the fourth channel of the compressed is accessed and returned as the fragment color. The resulting 3D geometry in this case is more realistic in nature as it has a 2D texture mapped onto it. An example of an encoded

image as it progresses though the decoding pipeline such that its original 3D geometry is recovered can be seen in Figure 2.6. Figure 2.7 shows the original geometry and the reconstructed geometry rendered together to show that the reconstructed geometry is visibly identical to the original after the Holovideo encoding and decoding process.

In summary, Holovideo compression is a very enabling technology as it allows for the compression of high-quality 3D scans to be performed at real-time speeds via the usage of the GPU. Given these benefits the compressed data can then easily be stored for a later date or transmitted across a network; each having the benefit of full 3D geometry retrieval via decoding.

2.3 Portal-s

Portal-s was originally developed at Iowa State University by Karpinsky (2013c). It was a software and hardware solution that allowed high-quality 3D data to be captured, transmitted, and redisplayed in real-time. Despite being able to stream real-time, high-quality 3D data, the advantage of the Portal-s system from a telepresence and immersion standpoint was its ability to achieve a corrected sense of eye contact and gaze between two remote users in an unencumbered manner; that is, users did not have to wear any additional hardware (e.g., a tracking marker) for this to be accomplished. This sense of eye contact was present due to the nature of the Portal-s hardware architecture being that a Portal-s node was able to capture 3D information from the same optical axis at which it gets displayed (Karpinsky, 2013c). As previously mentioned, a Portal-s node had the ability to capture, transmit, and redisplay high-quality data in real-time such that it was able to be used as a 3D video conferencing solution. For this to be achievable each feature must be processed simultaneously and in real-time. A diagram containing an overview of the system's general setup and its hardware architecture can be seen in Figure 2.8.

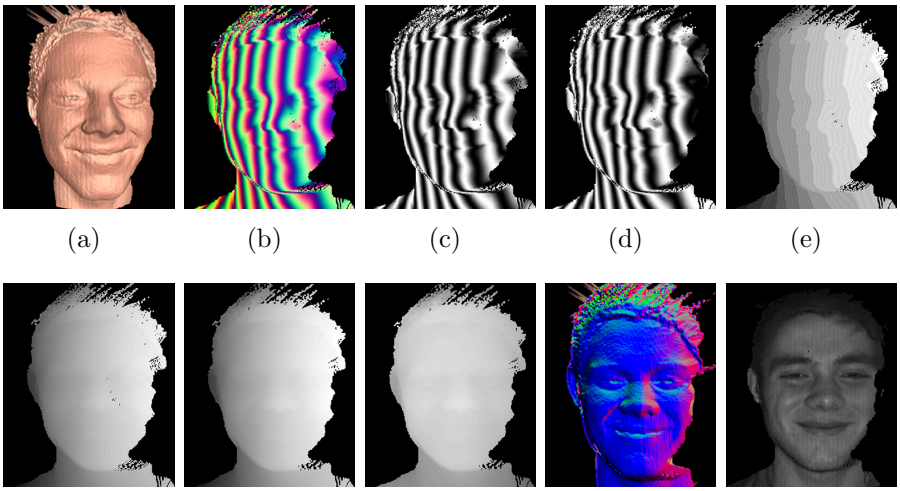


Figure 2.6 One example along the Holovideo decoding pipeline resulting in reconstructed 3D geometry with texture. (a) The original 3D reconstruction with texture mapping disabled; (b) holoencoded scan of the reconstruction; (c), (d), and (e) correspond to (b)'s RGB channels, respectively; (f) recovered unwrapped phase from (b); (g) filtered unwrapped phase map; (h) coordinate map; (i) normal map; (j) recovered 3D geometry with texture mapping enabled.

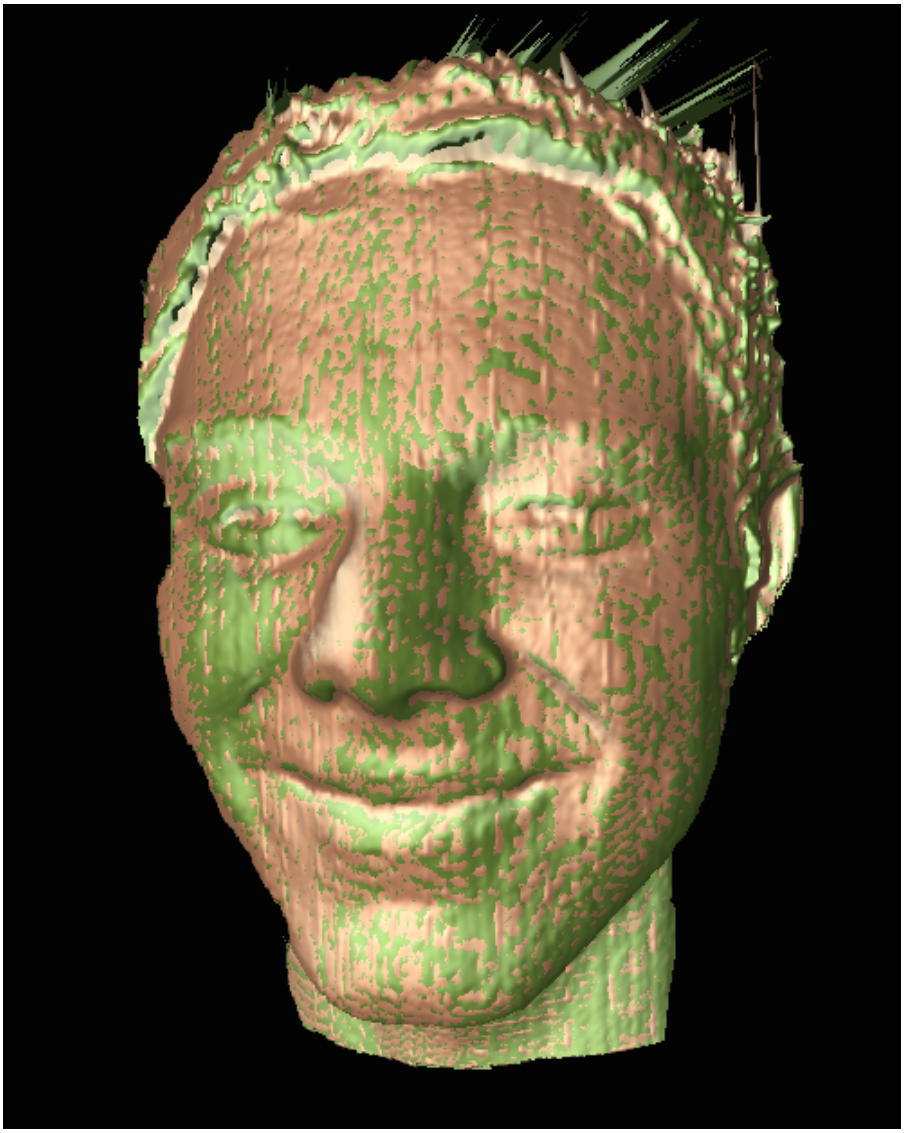


Figure 2.7 The original geometry (green) and the reconstructed geometry (gold) rendered together to show that there are no visible differences, or errors, in geometry even after the Holovideo encoding and decoding process.

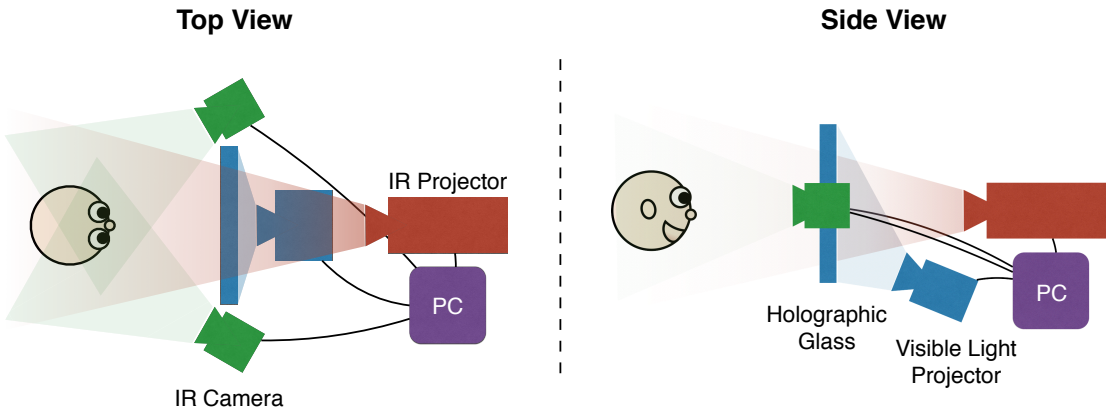


Figure 2.8 Portal-s node hardware architecture shown from two angles consisting of one IR projector, two IR cameras, one visible light projector, and a holographic display.

2.3.1 Real-time 3D Video Capture

To capture data, a user would sit in front of an IR two-camera, one-projector structured light scanner as portrayed in Figure 2.8's architecture diagram or in Figure 2.12's photograph. The IR projector projects its fringe images onto the user to be captured through a holographic glass. Due to the nature of the setup, the IR light was projected perpendicular to the display's orientation; at this angle the glass has a transmittance rate of 98% (Vislogix, 2013) thus allowing the fringe images to pass through the glass unto the person or object. Two IR cameras would each then capture the light from their unique angle. It should be noted that the cameras were situated parallel to the glass on either side, instead of behind the glass and next to the projector, as to avoid capturing any of the 2% of the light that did not pass through and may have reflected back off the glass. Using IR-based devices relieves the user from the harsh, bright lighting conditions that are present in scanners which use the visible light spectrum. The capture architecture of Portal-s essentially allowed for two structured light scanners (one camera and the one projector creating one structured light scanner pair; the second camera and the projector creating the second). Proper placement of these two systems within the one allows for more of the object, or person, to be captured. Each scanner pair will eventually generate its own 3D geometry of the object and each of these will be consolidated into one for the system of a whole thus providing a wider field of view (Karpinsky, 2013c).

To capture its object, Portal-s used the previously detailed three-step phase-shifting technique except two fringe pitches, pitch again meaning the number of pixels in each fringe stripe, were used in the generation of fringe patterns. This means that instead of using one set of three fringe images to encode an object (as described in Section 2.1.1), two sets of three fringe images are used: one for each pitch. This six fringe technique can be used to obtained unwrapped phase in parallel on the GPU (Karpinsky et al., 2014).

The capture system's pipeline is shown in Figure 2.9. As it can be seen, the first step is to obtain wrapped phase maps. For each set of fringe images, a wrapped phase is obtained

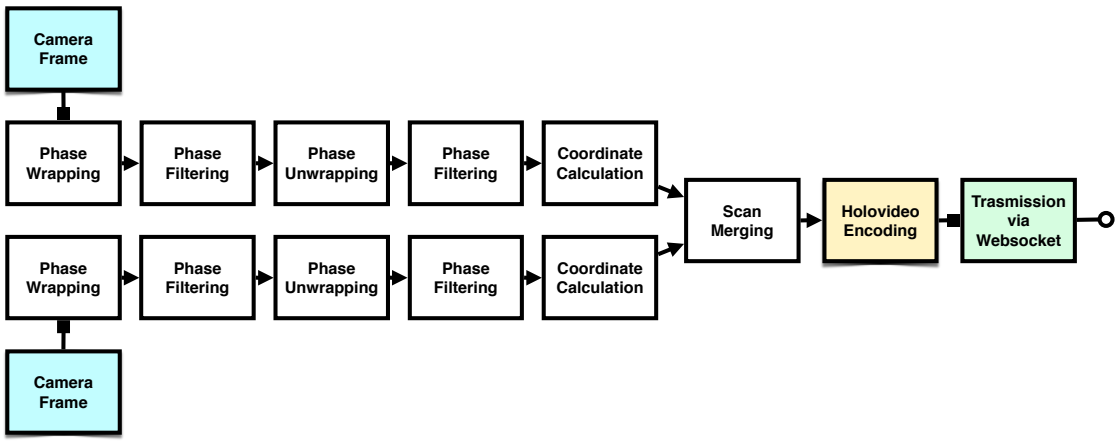


Figure 2.9 The Portal-s pipeline covering initial 3D capturing of an object from two structured light scanner pairs. Processing is performed on each pair before their respective coordinate maps are merged together. The unified coordinate map is compressed and video encoded before being sent to connected nodes via an HTML5 websocket.

utilizing Equation 2.4 where I', I'', and ϕ are defined as described in Equation 2.1-2.3. An *equivalent phase map* representing data from each of the phase maps can then be derived based on the differences between the individual phase maps (Karpinsky et al., 2014). In this six fringe case, the difference between phase maps ϕ_1 and ϕ_2 can be used to calculate the equivalent phase map ϕ_{12} via the equation

$$\phi_{12} = \phi_1 - \phi_2 \mod 2\pi. \tag{2.17}$$

The equivalent phase map ϕ_{12} can then be used to unwrap in parallel the wrapped phase maps ϕ_1 and ϕ_2 via Equation 2.6 where

$$K(x,y) = \text{round}\left[\frac{\phi_{12}(x,y) \times \frac{T_{12}}{T_1} - \phi_1(x,y)}{2\pi}\right]. \tag{2.18}$$

As each individual fringe pitch's phase map was used to derive the equivalent phase map ϕ_{12}, an *equivalent fringe pitch* is derived from the differences in the respective pitches. Therefore, in Equation 2.18, T_{12} represents the equivalent phase map's equivalent fringe period and it can be calculated via

$$T_{12} = \frac{T_1 T_2}{|T_1 - T_2|}, \tag{2.19}$$

where T_1 and T_2 represent ϕ_1's and ϕ_2's fringe pitch, respectively.

From here the pipeline resumes as it would in as described in 2.1.3. The resultant unwrapped phase map for each pitch can then be used, along with the system's calibration intrinsic and extrinsic properties, to derive its respective coordinate map. Lastly, coordinates from each structured light scanner pair are merged together to form one mesh; this is done by using the calibration's extrinsic matrix which provides a global coordinate system for the two scanners (Karpinsky, 2013c). The result of the Portal-s pipeline's meshing is a unified coordinate map which is now ready to be transmitted to another Portal-s node.

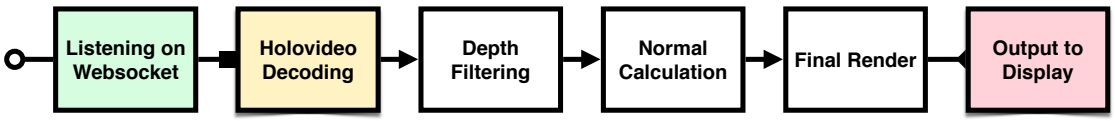


Figure 2.10 The Portal-s pipeline tasked with receiving a holoencoded image, decoding it, and rendering its reconstructed 3D geometry.

2.3.2 Real-time 3D Video Transmission

To transmit the data to another Portal-s node the amount of data must be manageable and small enough to feasibly work with. Portal-s utilizes Holovideo data compression developed by Karpinsky and Zhang (2013) which allows the system to stream data at a fast enough rate to achieve real-time transfer. As discussed in the previous section, the result of Holovideo compression is a 2D image. These images are then video encoded using motion PNG before being sent to another node (Karpinsky, 2013c). Portal-s uses HTML5 websockets to achieve a near-simultaneous stream to any other connected Portal-s client. HTML5 websockets were chosen as the communication protocol as they provide a low overhead and support full duplex communication (Karpinsky, 2013c). The pipeline shown in Figure 2.9 at this point has been traversed as capture, compression, and transmission are taking place. A diagram of two Portal-s nodes capturing and transmitting data can be seen in Figure 2.11.

2.3.3 Real-time 3D Video Redisplay

Once a holoencoded image is received by a Portal-s node client listening on a websocket, the next step is to decompress the 2D image to obtain the original high-quality 3D geometry. The rendered scene, as given by the Holovideo decoder's final rendering shader as described in Section 2.2, is then displayed. It should be noted that the decoding and visualization takes place in WebGL, an OpenGL implementation for modern web browsers. The decoding and final rendering pipeline can be viewed in Figure 2.10.

The scene is finally projected from a visible light projector onto the holographic

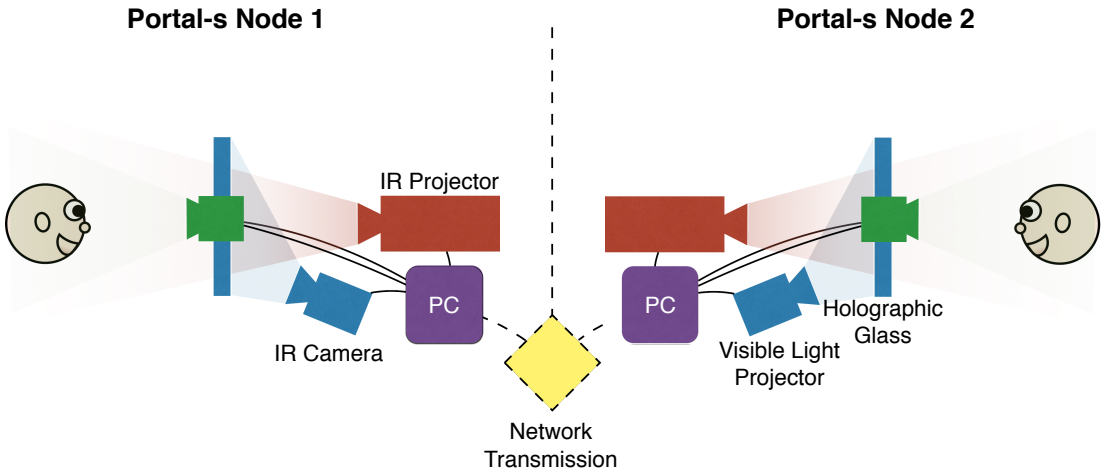


Figure 2.11 Two Portal-s systems architectures shown from the side angle. Each captures its user and sends its data over the network to the other node for redisplay.

display; a Vistique Hologlass by Vislogix was used in the construction of the original Portal-s system and can be seen in Figure 2.12. Due to the nature of the glass, light will pass through at a perpendicular angle whereas light projected at a decreasing angle will begin to be displayed on the glass. Instead of passing through the display, as the IR light did when projected perpendicularly during capture, this light is caught by the display as it is thrown at an angle of around 38 degrees perpendicular to the screen (Karpinsky, 2013c). The display projector is placed below the screen mainly for the purpose of achieving such an angle but it also does not interfere with the IR light at this placement.

2.3.4 Portal-s Discussions

Advantages to the end result of the process described is that it allows real-time 3D communication between two users over the internet. It also allows for the maintenance of eye contact and gaze if the hardware setup matches that detailed in Karpinsky (2013c). The work behind Portal-s is state of the art and solved many research goals, yet it is not without its own set of challenges. For example, the holographic glass used in the aforementioned system setup is not only quite expensive, but it also was one of the only ways such a system was able to achieve its sense of eye contact. Also, despite its transparent properties, such as the ability to catch light at certain angles and allow light to pass through at others, the end user is still forced to view 3D geometry on a 2D plane.

2.4 Summary

The preceding chapter detailed structured light scanning methods. It described how objects are captured, how their 3D geometries were derived, and how the geometries could be visualized. It also provided a solution to compress the original 3D geometry such that it can be streamed to, and therefore visualized on, another machine. An

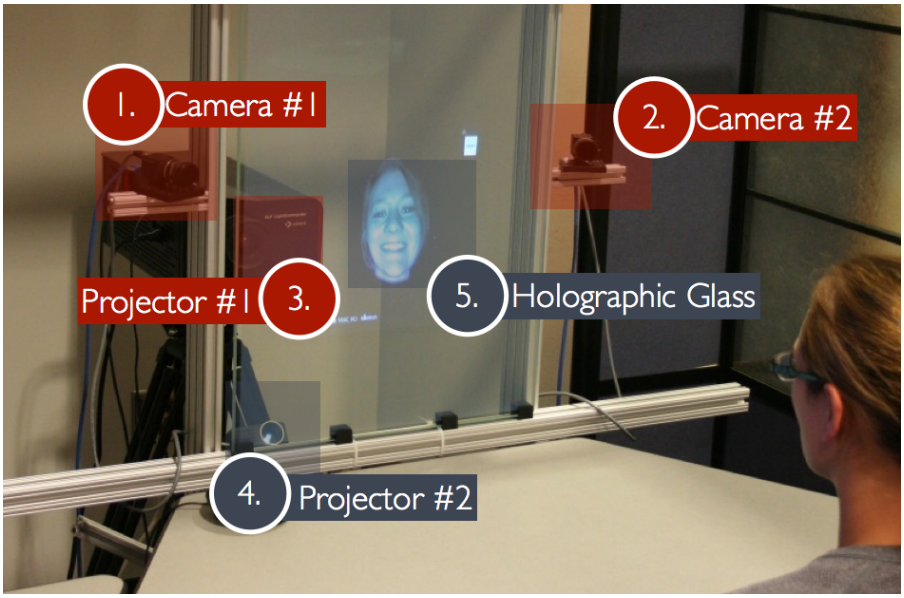


Figure 2.12 A Portal-s node hardware setup composed of two IR cameras (1)-(2), one IR projector (3), one visible light projector (4), and one holographic glass display (5). *Original photograph taken by Karpinsky (2013a).*

example of such a system doing this was presented in the discussion of Portal-s.

The foundation of this thesis research relies upon the principles covered in this chapter. The ultimate goal being realizing 3D video within a head mounted display, the techniques to obtain 3D video have now been covered. The next chapter will introduce HMD technologies.

CHAPTER 3. HEAD MOUNTED DISPLAYS (HMDs) AND HUMAN VISION

In this chapter, HMD technologies will be introduced along with certain concepts of human depth perception which are exploited within an HMD environment. It should be noted that the discussion of visual cues in Section 3.2 is based on *Chapter 4, Visual Sensory Systems* of Wickens et al. (2004).

3.1 Introduction to HMDs

An HMD is a piece of hardware which can be placed over a user's eyes to allow them the ability to feel as though they are in a virtual environment, among other things. HMDs consist of one or more displays, or screens, an array of sensors (such as an accelerometer or a gyroscope), and typically a pair of optical lens. In simple terms, the idea behind the HMD is to mimic a user's own eyes by presenting one image to each one of the user's eyes within the headset. To achieve this HMDs usually contain one or two monitors internally. For the two display case, one image is sent from the computer to each of the monitors. For the one display case, one image is sent to the monitor that contains a left-eye portion and a right-eye portion; examples of each case will be covered in this thesis. Regardless, the image to be presented could contain some sort of virtual environment which would give the user a sense of being immersed in virtual reality.

HMD headsets typically want to present their wearer with as much of the virtual environment as possible; this is done by providing a wide field of view (FOV). FOV is

an angle that represents how much of the environment can be seen from the perspective of the observer. Headsets that can achieve a wide FOV, and thus take advantage of a human's inherently wide FOV, can further stimulate its user's senses by presenting as much visual information of the virtual environment as possible. A example depicting horizontal FOV can be seen in Figure 3.1.

In this environment, distance cues can be exploited such that they imitate cues that we humans perceive in real environments; this is done to further any feeling of immersion for the user. Some of these cues will be described in Section 3.2. Finally, the different sensors on the device can be interpreted to track the wearer's head as it moves in physical space. This movement can be used to redraw the virtual view to match how we see different parts of the world around us when we humans move our head.

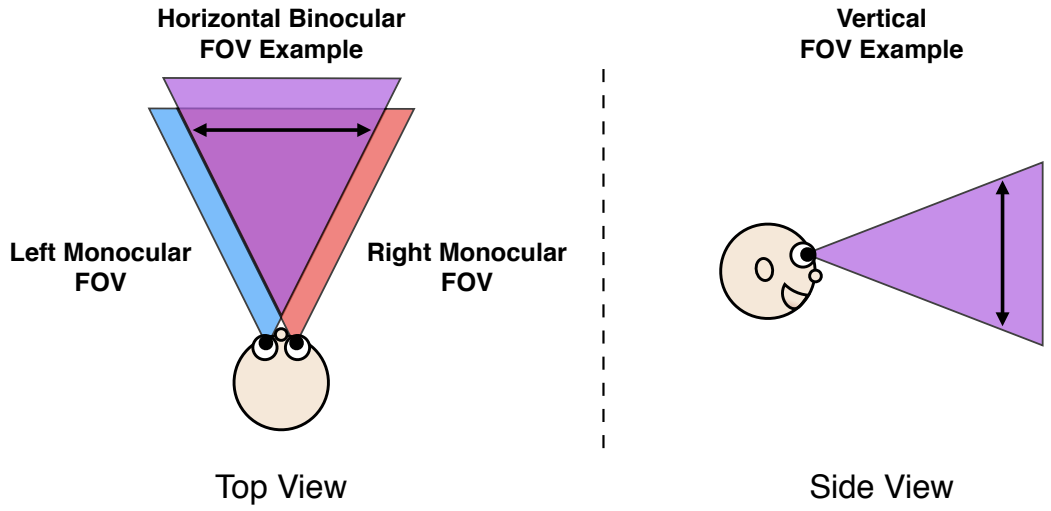


Figure 3.1 Example representations of horizontal and vertical field of views (FOV).

3.2 Visual Cues

As previously mentioned, the scenes displayed in the virtual worlds should aim to match those that we experience in the real world. A realistic portrayal of depth and relative distances enable a more realistic simulation. This can be done by exploiting how

we, as humans, naturally determine these: via the usage of distance, or depth, cues. By taking advantage of physical cues in a virtual environment we may be able to improve a user's level of immersion in such an environment. One such cue in particular that is inherently exploited by HMDs is stereopsis.

3.2.1 Stereopsis

Stereopsis is a binocular depth cue by nature which means that it involves data from both eyes. Due to the space between each of a human's eyes, each eye obtains a different view of the world around it; this is due to the two different angles at which an object is being seen. Another way to explain this concept is to consider that the objects in the environment provide differing 2D projections of themselves onto a human's retina. Fortunately, human brains are very well trained at composing these two images of the world together to gather depth information. This is done via the brain's ability to find *corresponding points* in two images. The brain finds a feature point in one of the eye's images, finds it in the other eye's image, and then via triangulation, can determine how far away the feature point is from the viewer.

Take for example the finger test in which you hold your finger a short distance in front of your face. As you focus on your finger with your left eye, keeping your right eye shut, you can see your finger in one position. Next, view your finger with your right eye, keeping your left eye shut. You should see your finger appear to jump to the left. To determine how far away your finger is from your eyes, your brain essentially locates your finger in the image from each eye and performs a triangulation. Figure 3.2 provides an example of how each eye image perceives a different angle of the object in focus.

The concept of stereopsis may be exploited when rendering views for the HMD's left and right eye. If the disparity between a human's eye images is simulated correctly, stereopsis within the HMD can be performed almost naturally for the user. That is the difference between the virtual left eye image and virtual right eye image should be

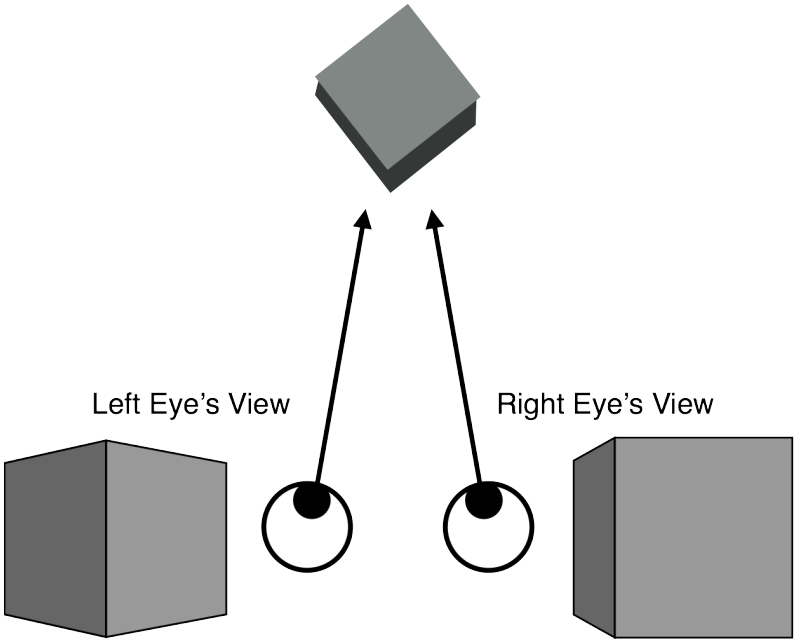


Figure 3.2 Due to binocular disparity, each eye sees a different angle of the object in focus. The brain then finds the same feature point in each image. The point's depth can be calculated via triangulation from the feature point's location in each image. *This regenerated figure was derived from NMU.edu (2013).*

almost identical to the difference between the physical left eye image and physical right eye image.

3.2.2 Convergence

Convergence is another binocular depth cue used by humans to perceive distance in their environments. In this cue, the angle of rotation inward between the two eyes offers a hint to the brain as to how far away the object in focus may be from the observer. For example, if the eyes have to each turn inward, via the eyes' extraocular muscles, to achieve focus then this gives a clue to the brain; the more inward the eyes must turn, the closer the object in focus most likely is. For this reason, convergence is only beneficial at shorter distances as when objects exceed a certain range, a human's line of sight will become parallel to bring the point into focus. At this parallel state the angle of convergence between the two eyes becomes ineffective. This angle as it relates to one's line of sight is demonstrated in Figure 3.3.

Take the finger test again as an example: as your finger gets closer to your eyes you may feel them moving inward; if you get too close you may become "cross-eyed" as your eyes each try to bring your finger into focus. As you move your finger away from your face you may feel your eyes start to face more forward as the lines of sight are now becoming parallel. This angle created between your two eyes as they bring your finger into focus is one hint to the brain at how far away that finger might be from the observer.

3.2.3 Monocular Depth Cues

Distance between an observer and an object or between two or more objects can be estimated with data from only one eye, as well; cues in which only data from one eye is necessary are called monocular depth, or distance, cues. The following list offers a brief description of a subset of different monocular cues:

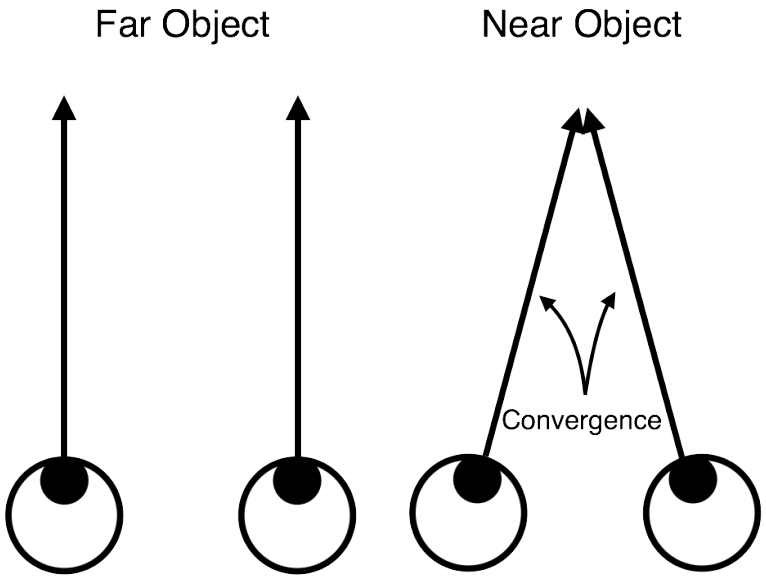


Figure 3.3 Convergence gives the brain a clue as to how far an object in focus is from the observer based on the angle created between two eyes. The angle closes inward as the object gets closer and becomes parallel as the object moves away. *This regenerated figure was derived from Davydov (2010).*

- *Familiar Size.* Using past knowledge about an object's size, an object's current perceived size can be used to judge its distance from the observer. For example, many humans know the approximate size of a soccer ball. Given the current size of the ball and the information perviously known about the ball's size, the brain can estimate the approximate depth of the ball in an environment. If the ball appears to be very large then a valid assumption may be that the ball is very close to the observer. Conversely, if the ball appears very small then it may be at a decent distance from the observer.

- *Relative Size.* If the physical size of an object is unknown yet there are two or more of these objects, size information of the individual objects in relation to each other may be used to determine which of the objects is closer or further away from the observer.

- *Color and Shadowing.* The way natural light falls upon an object and any possible shadow that the object may cast may be used to determine characteristics about the object's size and shape.

- *Motion Parallax.* This phenomenon is evident when the observer is traveling or moving in relation to the objects being observed. Objects in the foreground appear to be moving by quickly while objects in the distance appear to be moving at a slower rate, if at all. The speed at which objects appear to be moving as the observer travels lends its hand to its depth information in relation to the user: closer objects will appear to move at a faster rate.

- *Occlusion.* Occlusion takes place when one object appears to be in front of the other; that is that they overlap. Although occlusion does not provide any physical depth information, it at least allows the observer an accurate means to judge which object may be closer to them.

3.3 Summary

In this chapter, HMD technology was discussed. As HMDs aim to mimic a user's sense of vision, such that virtual worlds can be simulated, certain aspects of human vision were also discussed. Via the exploitation of depth cues, the creators of the virtual worlds can *trick* observers into seeing their environment naturally and in 3D. The next chapter will discuss the main research behind this thesis: how 3D video visualization was achieved in a virtual reality HMD by primarily exploiting the stereopsis cue.

CHAPTER 4. 3D VIDEO IN A HEAD MOUNTED DISPLAY

The previous chapters have described how high-quality 3D video can be obtained, compressed, transmitted, and redisplayed; provided an introduction into how humans perceive depth and the visual world around us; and have detailed the general concepts behind visualizing virtual environments within HMDs. This chapter presents the major contribution of this thesis work which has been focused on the application of these technologies into realizing high-quality 3D video within an HMD, specifically the Oculus Rift, and further developing a web-based 3D visualization environment. From a high level, the first step in achieving 3D video within an HMD is to reconstruct the original 3D geometry from compressed data. Next, split-screen stereo views must be rendered, one for each of the user's eyes within the HMD. Afterward, virtual lens distortion correction and rescaling is applied to adjust for a distortion on the light that passes through the Rift's lens from its screen. Finally, the Rift's location and movement about the real world is interpreted to update the rendering of the virtual views as the user moves their head in physical 3D space. A high-level diagram of this pipeline can be seen in Figure 4.1.

A web-based application originally developed by Karpinsky for Portal-s (Karpinsky, 2013c) allowed for the visualization of 3D video data within a modern web browser. For this thesis work, this system was further developed such as to increase a user's sense of immersion and potential level of interaction. This thesis will refer to the improved system as *Nimbus*.

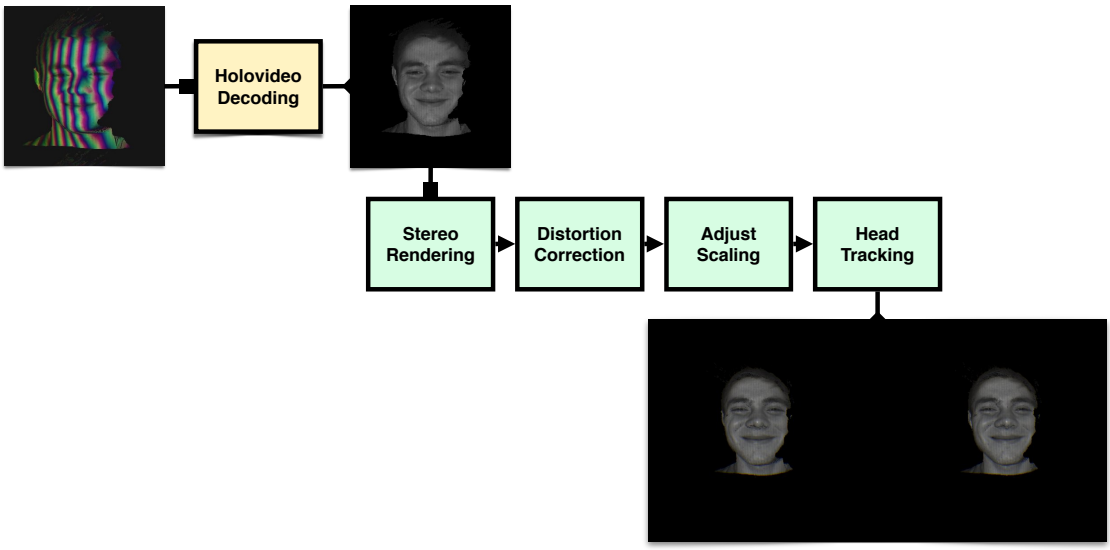


Figure 4.1 A high-level overview of the process required to achieve 3D video in an HMD. Holovideo decoding is first done on compressed geometry to realize the original 3D object. Before redisplay, however, stereo views will be computed, distortion will be accounted for, the model's scale will be adjusted, and finally head tracking will be implemented. The result is responsive high-quality 3D video suited for viewing in an Oculus Rift.

4.1 Motivations

As detailed in Section 2.3, the Portal-s platform developed by Karpinsky (2013c) faces some challenges. One of the challenges lies inherently in the holographic glass which was used for a display. It is quite a pivotal piece in the hardware architecture as it allows for the maintenance of eye contact by letting the structured light pass through it during capture while presenting 3D geometry from another node during redisplay. Albeit a novel idea, the issue lies in a lack of potential immersion while users are still forced to view 3D geometry on a flat 2D plane. Therefore, a specific motivation of this research is to remove the requirement for the holographic glass in a Portal-s node and thus allow the user to be immersed in 3D environment within an HMD; this can essentially be done by exploiting the stereopsis depth cue during redisplay of 3D geometry. By eliminating the need for specialized glass in the system, the need for a second display projector to display reconstructed geometry is also eliminated. This being the case, by visualizing 3D video in an HMD, two pieces of system hardware can be replaced: the holographic glass and the display projector.

Another piece of hardware that can be replaced is one of the cameras used in the Portal-s two-camera, one-projector system. Commonly, structured light scanners are one-camera, one-projector in nature, so another goal of this thesis research was to make the capture technology required for such a visualization system more portable to 3D scanning technologies. This was done such that any structured light or 3D scanner's geometry could potentially be visualized in the HMD.

4.2 Oculus Rift

The HMD which was used for visualization purposes was the Oculus Rift Development Kit v1 which was developed by Oculus VR, Inc. and can be seen in Figure 4.2. The Oculus Rift includes a 7-inch (diagonal) monitor with a resolution of 1280×800 (720p).

As previously mentioned in the discussion of HMDs, a headset can either have one or two displays. As the Rift has one display, a view is rendered for each of the user's eyes, each filling half of the display, and thus the resulting screen resolution is 640 × 800 for each eye. The Rift's monitor is refreshed via a DVI-D connection to the host PC. Each human eye views its portion of the screen through one of several lens provided with the Oculus Rift Development Kit v1 (Antonov et al., 2013); this allows users in a correctly configured software/hardware environment to experience phenomena such as depth and scale. The distance between the user's eyes and lens can be adjusted physically on the headset to allow for users wearing glasses and for a greater sense of comfort, in general. The headset weighs less than one pound and thus will typically not over-encumber its wearer. The headset also includes several sensors such that it can track the wearer's head in 360-degree physical space. This is done via its three-axis gyroscope, which senses angular velocity, and its three-axis accelerometer, which senses acceleration. The Oculus Rift provides users with a FOV of over 100 degrees (Yao et al., 2014).

Figure 4.2 The Oculus Rift Development Kit v1 HMD hardware.

4.3 3D Model Reconstruction

This research utilized Holovideo encoded images to recover 3D video to be displayed in the visualization system and within the HMD. As each encoded image represents a single 3D frame, a mosaic of Holovideo encoded images was used as one way to realize 3D video. An example of this concept can be seen in Figure 4.4. During the decoding and visualization process, the correct index of the individual holoencoded image to be

rendered out of the mosaic is be calculated based on the visualization's desired frame rate and a timer on the central processing unit (CPU). The single holoencoded frame at the specified index in the mosaic is then essentially cropped out and decoded as normal on the GPU as described in Section 2.2.2.

The result of such decoding is coordinate map which can be used to derive original 3D geometry as shown in Figure 4.3 when rendered in WebGL within Nimbus. At this point in the pipeline, 3D textured frames, and video via the mosaic, can be recovered; following the pipeline presented in Figure 4.1, the next step is to create a unique view of the geometry for each of the user's eyes to be displayed in the HMD.

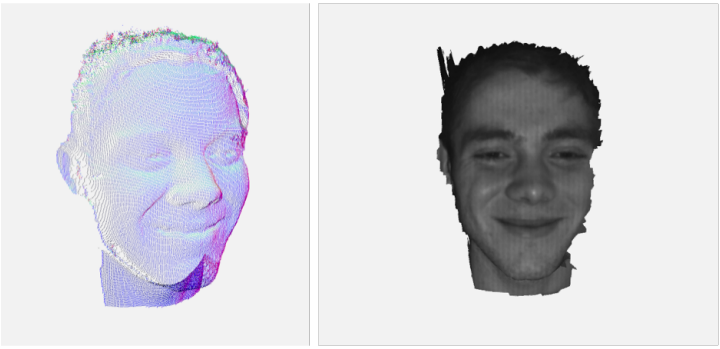


Figure 4.3 Left: 3D geometry reconstruction and visualization. Right: 3D visualization with texture information. Each 3D object was decoded from a 3D compressed image saved in the PNG format.

4.4 Split Screen Stereo

After the 3D model has been reconstructed, one view for each of the user's eyes must be rendered. As the Oculus Rift contains one monitor, instead of two separate monitors, the split views are created virtually. Essentially, the scene is rendered twice: once for each eye. A difference between each rendering is applied such that they vary slightly. The equations which will be described to derive these views were originally detailed in the Oculus Rift's Developer Software Development Kit (SDK) documentation (Antonov et al., 2013).

Figure 4.4 A mosaic of holoencoded images. The system determines which index to be displayed, crops out the desired image, decodes its coordinates, and finally displays its geometry.

An easy way to conceive what rendering split-screen stereo is doing is to imagine two virtual cameras being placed such that they are viewing the virtual 3D object; the goal is to place the two cameras such that they resemble the placement of the user's eyes. To do this as accurately as possible and to render the most realistic scene, the device's physical dimensions must first be used. The physical dimensions of the device that the Oculus SDK (Antonov et al., 2013) provides, which are required to derive the split views, are: *HScreenSize*, *VScreenSize*, *VScreenCenter*, *EyeToScreenDistance*, *LensSeparationDistance*, *HResolution*, and *VResolution*. Another property that is provided, that's not one of the Rift's physical measurements, is *InterpupillaryDistance* which represents the physical distance between the device wearer's two eyes. These properties are described in Table 4.1 which is derived from (Antonov et al., 2013).

The first step to render split screen stereo is to calculate the aspect ratio and FOV for each eye's viewport. Aspect ratio eludes to the relationship between a view's, or image's, width and height as shown in Figure 4.5 and FOV is defined as in Section 3.1 with an example shown in Figure 3.1.

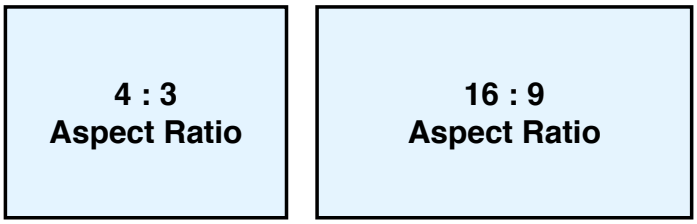


Figure 4.5 Examples of two different aspect ratios. In each image the ratio between its width and height is given with the left having a 4:3 and right having a 16:9 aspect ratio, respectively.

The aspect ratio, a, can be derived from the following equation using the Rift's physical properties,

$$a = \frac{\text{HResolution}}{2 \times \text{VResolution}}, \tag{4.1}$$

and the angle of the FOV can also be derived via the physical properties, while ignoring

Oculus Rift Property	**Description**
HScreenSize, VScreenSize	Physical dimensions of the entire HMD screen in meters. Half HScreenSize is used for each eye.
VScreenCenter	Physical offset from the top of the screen to the eye center, in meters. Currently half VScreenSize.
EyeToScreenDistance	Distance from the eye to the screen, in meters. This combines distance from the eye to the lens and from the lens to the screen.
LensSeparationDistance	Physical distance between the lens centers, in meters.
InterpupillaryDistance	Distance between the eye centers as configured by the user.
HResolution, VResolution	Resolution of the entire HMD screen in pixels. Half the HResolution is used for each eye.

Table 4.1 The Oculus Rift's physical device properties which will be used to calculate split-screen stereo views to be displayed within the headset.

any notion of lens distortion for the time being, with

$$F = 2 \times \tan^{-1}\left(\frac{\text{VScreenSize}}{2 \times \text{EyeToScreenSize}}\right). \tag{4.2}$$

Based on these values, a projection matrix, M, is calculated as,

$$M = \begin{bmatrix} \frac{1}{a \times \tan(\phi_{fov}/2)} & 0 & 0 & 0 \\ 0 & \frac{1}{\tan(\phi_{fov}/2)} & 0 & 0 \\ 0 & 0 & \frac{z_{far}}{z_{near} - z_{far}} & \frac{z_{far} \times z_{near}}{z_{near} - z_{far}} \\ 0 & 0 & -1 & 0 \end{bmatrix}. \tag{4.3}$$

The z_{near} and z_{far} values represent the clipping plane's depth coordinates and values of 0.01 and 1000 were used, respectively, in the implementation. Currently, the center of the projection matrix M lies in the middle of each screen yet the final rendering requires P to lie at the center of each lens of the device; thus, the matrix is adjusted based on *LensSeparationDistance* (Antonov et al., 2013). To do this, the transformation matrix H is applied to M to obtain M' where $M' = HM$ where

$$H = \begin{bmatrix} 1 & 0 & 0 & \pm h \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{4.4}$$

The value of h as used in Equation 4.4 is defined as,

$$h = \frac{4 \times h_{meters}}{\text{HScreenSize}} \tag{4.5}$$

where,

$$h_{meters} = \frac{\text{HScreenSize}}{4} - \frac{\text{LensSeparationDistance}}{2}; \tag{4.6}$$

the term h is positive in the right-eye's view and negative in the left-eye's view (Antonov et al., 2013).

The final step is then to transform the "non-stereo game view transform" (Antonov et al., 2013), V, as it would originally be at the center of two eyes. To match the location of each eye, V must be transformed based on the user's configured *InterpupillaryDistance* via,

$$V' = \begin{bmatrix} 1 & 0 & 0 & \pm\frac{\text{InterpupillaryDistance}}{2} \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} V. \tag{4.7}$$

The result from applying split-screen stereo is one view for each of the user's eyes. As seen in Figure 4.6, there are two images of the 3D object side by side. The angle at which the 3D geometry is being visualized differs between the two and thus when visualized inside of an Oculus Rift, the two images appear to be one 3D image via convergence of the two separate images. The differing angles on the virtual geometry have been calculated to match the differing angles between the user's two eyes. Although data can now be perceived in 3D in the Rift, due to the device's properties the geometry may appear unnatural.

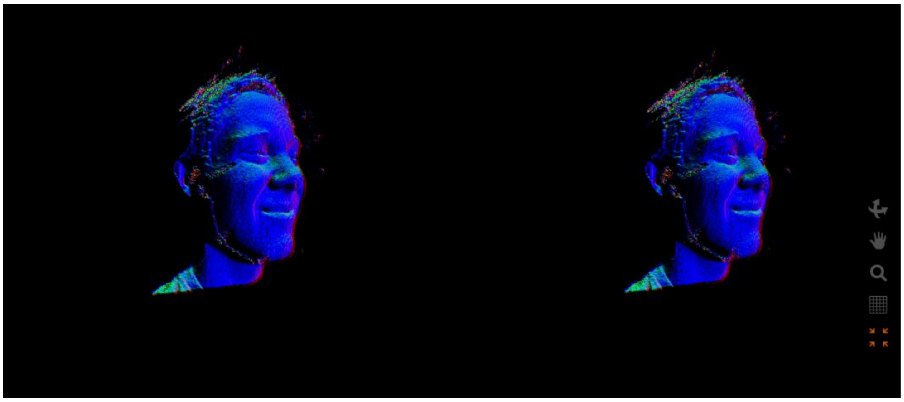


Figure 4.6 Visualization of a 3D frame after split-screen stereo has been applied; there is one view for each of the user's eyes. When visualized inside of an Oculus Rift, two images appear to be one 3D image.

4.5 Distortion Correction

Now that stereo-rendering has taken place, there is one view for each eye and they differ such as to simulate binocular disparity. Next, lens distortion must be accounted for as one property of the Rift's lenses is that they magnify the image passing through them such that the user's FOV is increased. While increasing the user's FOV is theoretically a good feature, the downside is that this magnification induces an optical "pincushion" distortion. To keep the advantages of the magnification yet eliminate the distortion, a "barrel" distortion can be applied virtually to the rendered views. By taking this step, the virtual barrel distortion can cancel out the physical pincushion distortion; remaining is a magnified sense of the geometry without distortion. An example visualization of the two types of distortion can be seen in Figure 4.7.

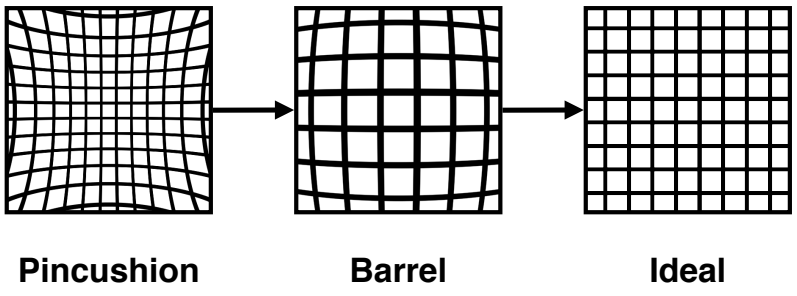


Figure 4.7 Two types of distortion are involved with when rendering geometry for the Oculus Rift. The Rift's lens induces a physical pincushion distortion when magnifying light; this can be cancelled out by simulating a barrel distortion on the rendered image. The goal of this is to have the two distortions cancel one another out such that ideal straight lines remain straight lines.

To simulate this distortion, each pixel is either expanded away from or contracted towards the lens center via the distortion model,

$$(r, \sigma) \mapsto (f(r)r, \sigma), \tag{4.8}$$

where r is the distance from the center, where σ is the pixel's angle in respect to the

center, and where the function applied is

$$f(r) = k_0 + k_1 r^2 + k_2 r^4 + k_3 r^6 \tag{4.9}$$

where k_0, k_1, k_2, and k_3 are positive and obtained from the Oculus SDK. It should be noted that the angle, θ, remains unchanged. The lens center for which the distortion model can be applied is calculated using the aforementioned physical properties of the Oculus Rift HMD,

$$\text{LensCenter} = 1 - 2 \times \frac{\text{LensSeparationDistance}}{\text{HScreenSize}}. \tag{4.10}$$

After the barrel distortion has been simulated, the two differing distortions will cancel each other out; however, due to the nature of barrel distortion there will be some scaling issues as barrel distortion brings in those pixels from the outside edges toward the center leaving nothing in their place. To get around this, a scaling factor s can be derived to select an appropriate radius for the distortion function (Equation 4.9) and can be described as,

$$s = f(\pm 1 - \text{LensCenter}). \tag{4.11}$$

This scaling function allows the image to be scaled back out to the edges of the image removing any potentially wasted space while maintaining the barrel distortion. A prominent example of the split screen stereo rendered with barrel distortion can be seen in Figure 4.8.

Finally, the field of view, which was originally calculated via Equation 4.2 when distortion was ignored, needs to be adjusted due to the offset, distortion corrections, and scale which have been introduced. To account for this, however, all that must be done is to introduce the scaling function, s, into the calculation of the FOV such that,

$$F' = 2 \times \tan^{-1}\left(\frac{s \times \text{VScreenSize}}{2 \times \text{EyeToScreenSize}}\right). \tag{4.12}$$

When visualized at this point in the pipeline, 3D data now appears to be naturally geometric. What remains unnatural is that different angles of the geometry will not

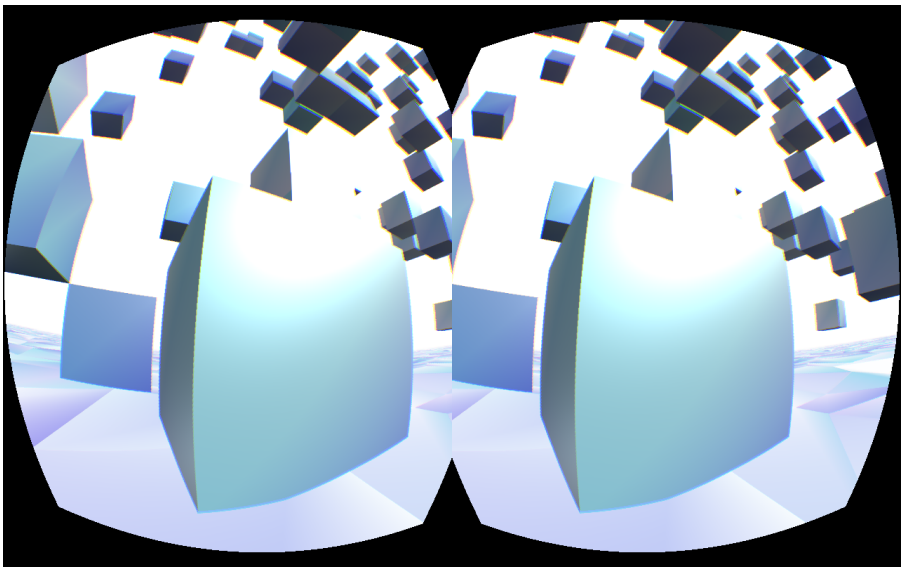


Figure 4.8 Split-stereo rendering of a virtual environment with barrel distortion.

display when moving one's head while wearing the Rift. To achieve this, head tracking can be implemented such that the two virtual cameras on the scene move according to the user's head in physical space.

4.6 Head Tracking

The Oculus Rift's head-tracker delivers up to 1000 Hz sampling rate which is essential to immersion as well as limiting user sickness. Using the device's gyroscope and accelerometer, the Oculus Rift SDK provides the user's motion around x (pitch), y (yaw), and z (roll). The change in these parameters are monitored, interpreted and the views are updated accordingly. The important concept here is to move the virtual cameras which are responsible for generating the split-stereo view in a natural manner based on the user's physical movements. If it is not natural, such as if it is too fast (perhaps trying to incorporate movement prediction) or if the camera movement has some sort of lag, the user will be discomforted due to the inconsistency between natural vision and simulated vision. Not only does this discomfort hinder a user's sense of immersion but it can also induce motion sickness. Once head tracking is appropriately implemented,

however, visualization of 3D geometry feels very natural in that one can move their head around and view different the object from different angles.

4.7 Results

The preceding concepts have been implemented within WebGL which is a JavaScript API, based on OpenGL, for implementing 2D and/or 3D graphics within modern web browsers (such as Google Chrome or Mozilla Firefox). The open-source 3D JavaScript library *three.js* was also used and essentially acts as a layer on top of the browser supported WebGL library. Another open-source library, *vr.js*, was used to expose the Oculus Rift's hardware to the web browser such that head-tracking sensors and correct HMD scene rendering could be accomplished via usage of WebGL's shaders.

Data used for this research were either single compressed Holovideo encoded images or a mosaic of such images (as discussed in Section 4.3). Each of the images were 512 $\times$ 512 pixels in nature and were either in the 3-channel or 4-channel PNG image format depending on if texture information had been encoded during compression.

The result is a virtual reality system that runs on almost any modern computer hardware with a graphics card, an internet connection, and a web browser allowing any user with an Oculus Rift HMD device to connect to the web and visualize 3D video almost instantly. An example of what they might visualize on their 2D monitors can be seen in Figure 4.6 and Figure 4.9 but once visualized in the Rift, the stereo-rendered, barrel-distorted light would pass through each of the lens and into the user's eyes allowing the human brain to take over such that one object can be seen and immersive 3D can be visualized.

Head tracking has also been implemented into the system such that as the user moves their head around in physical 3D space they are also moving the virtual cameras around the 3D model. Take again the example in Figure 4.6 or 4.9; as the user moves their head

around they would see different angles of the 3D face. If the user moved their head to their right then they would see the 3D model rotate to its right such that more of the left side of its face would be displayed.

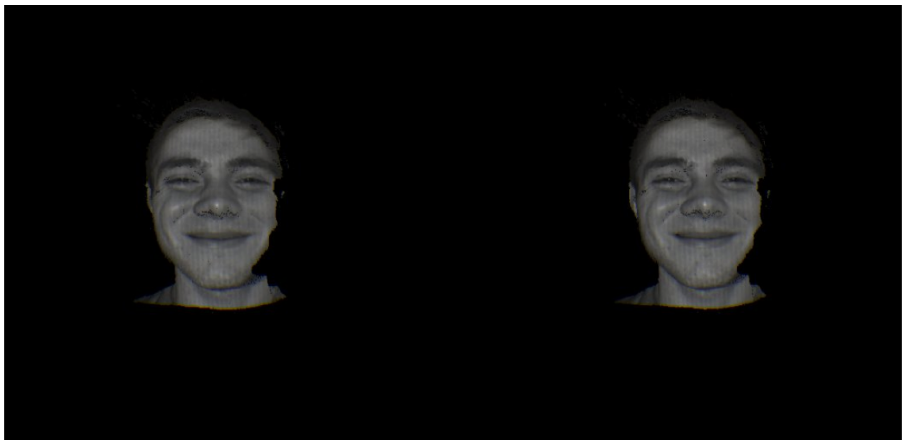


Figure 4.9 Final output of the pipeline. Data was rendered in split-screen stereo with lens distortion correction applied. Texture information has been toggled on.

4.8 3D Visualization System: Nimbus

Another primary success point for this thesis research is the level of work that has been done to extend Nimbus, which was introduced at the beginning of this chapter, to include different types of visualization devices; allow for a number of different ways for users to manipulate and interact with 3D video; and to include a larger feature-set of capabilities, in general. The underlying goal of Nimbus is to provide users with an easy to understand and use method for interacting with, potentially live, 3D video; even if it is their first time working with such content. Being a web-application, Nimbus runs on modern web browsers thus it does not require any specialized hardware or software other than a graphics card. Given this, the intent is to be able to disperse such technologies to many different types of users around the world. Figure 4.10 shows one screenshot of the current Nimbus system and eludes to some of its controls which will be described next.

Currently within Nimbus users can visualize 3D video on their normal output displays

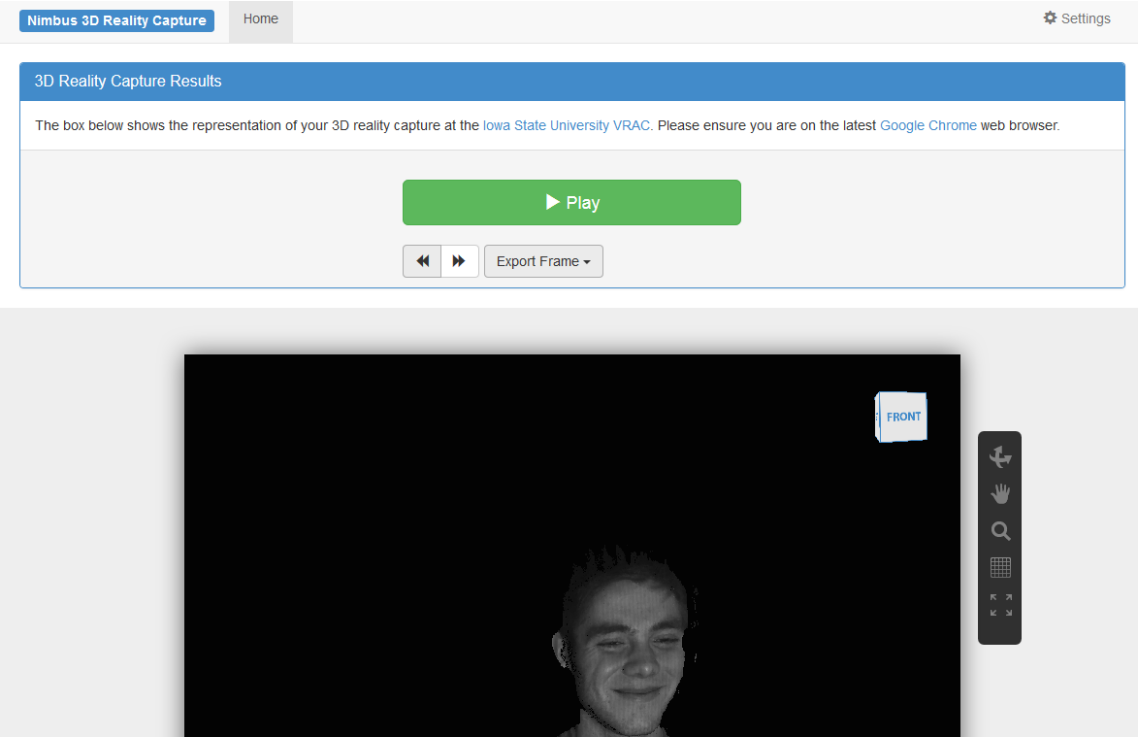


Figure 4.10 Nimbus 3D Reality system that currently runs on Google Chrome or Mozilla Firefox web browsers.

or via their Oculus Rift due to the process described above. The goal is that by allowing users to view 3D geometry in an HMD, Nimbus provides a more immersive experience to the user. Due to the higher level of spatial data that a 3D scanning system provides, intuitive ways to interact with the data becomes an important feature of a system.

Users can choose to use a normal mouse and keyboard combination when manipulating 3D models or they may choose to use a Leap Motion Controller (Motion, 2014a). The Leap Motion Controller is a 3D sensor which captures the positions and orientation of a user's hands and fingers; these readings can then be interpreted and used in different types of applications. During this thesis research support for the Leap Motion Controller was implemented into Nimbus such that it provided users a way to interact with the data in a more natural and intuitive manner by using their hands. Users can manipulate a view on a 3D video by moving their hands toward the screen to zoom in, away from the screen to zoom out, left to rotate the 3D model left, and right to rotate it to the

right; this was implemented by defining a relationship between the finger positions, as emitted by the Leap Motion JavaScript API (Motion, 2014b), and the virtual camera which dictates the orientation of the 3D model to be displayed. Figure 4.12 shows the author interacting with 3D data via the Leap Motion Controller.

Another overarching goal, from a human computer interaction perspective, is to provide intelligent methods for users to use and be immersed in the data. Other features were implemented into Nimbus to further a user's ability to manipulate their experience with 3D video such the ability to:

- *Toggle texture information on and off.* As previously detailed, holoencoded images may contain texture information for its associated 3D geometry. This control allows the user to visualize their data with this texture mapping turned on or off thus giving them control over the fidelity of their output on the web-based system. This feature is controlled via the Settings control panel shown in Figure 4.11 and dictates if the texture information should be a point's output color (as calculated in the fragment shader) or if the normal map should be used.

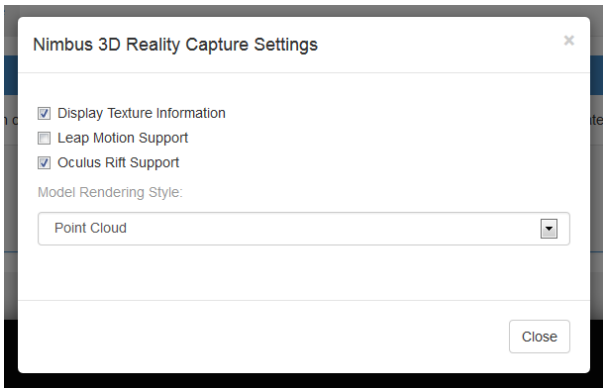


Figure 4.11 The settings panel in the visualization system allowing users to toggle texture, toggle support for the Leap Motion Controller, toggle support for the Oculus Rift HMD, and change the 3D model's rendering style.

- *Control 3D video feedback.* When viewing 3D video in a non-live use case, users can pause playback and use controls to move back or forward a frame providing them

Figure 4.12 The author interacts with 3D video in the visualization system using the Leap Motion Controller. His hand movements control the virtual camera's view of the 3D geometry. *Original photograph taken by Karpinsky (2013b).*

the ability to visualize a specific frame of interest for an extended amount of time. Also when paused users have the ability to output the currently paused frame to an OBJ or STL file such that they can work with it in another 3D application of their choice. The ability to change frames and export data becomes available after the user has paused the video; it hides once the video resumes. These controls are shown in Figure 4.13. As described previously, a timer runs continuously and is kept in check with the system's desired frame rate to dictate which index of a mosaic containing Holovideo encoded images to display. Pausing is implemented by essentially halting the timer such that this index does not advance and no other frame is displayed. Exportation of 3D geometry to OBJ and STL file formats was implemented by conforming to each file format's respective file template in regards to how each point should be described.

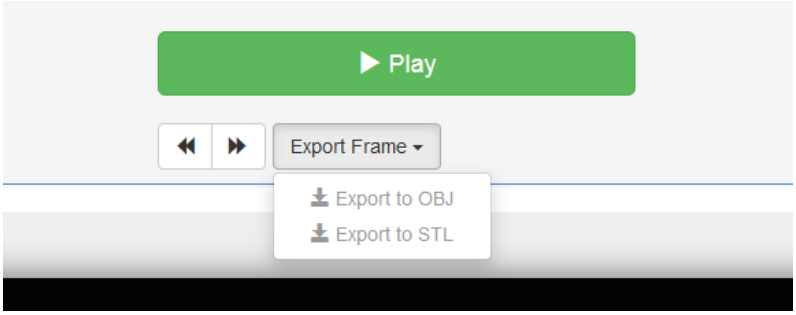


Figure 4.13 The playback controls in the visualization system allowing users to play and pause 3D video. When the player is paused, users can move back frames, go forward frames, and export a frame as an OBJ or STL 3D file.

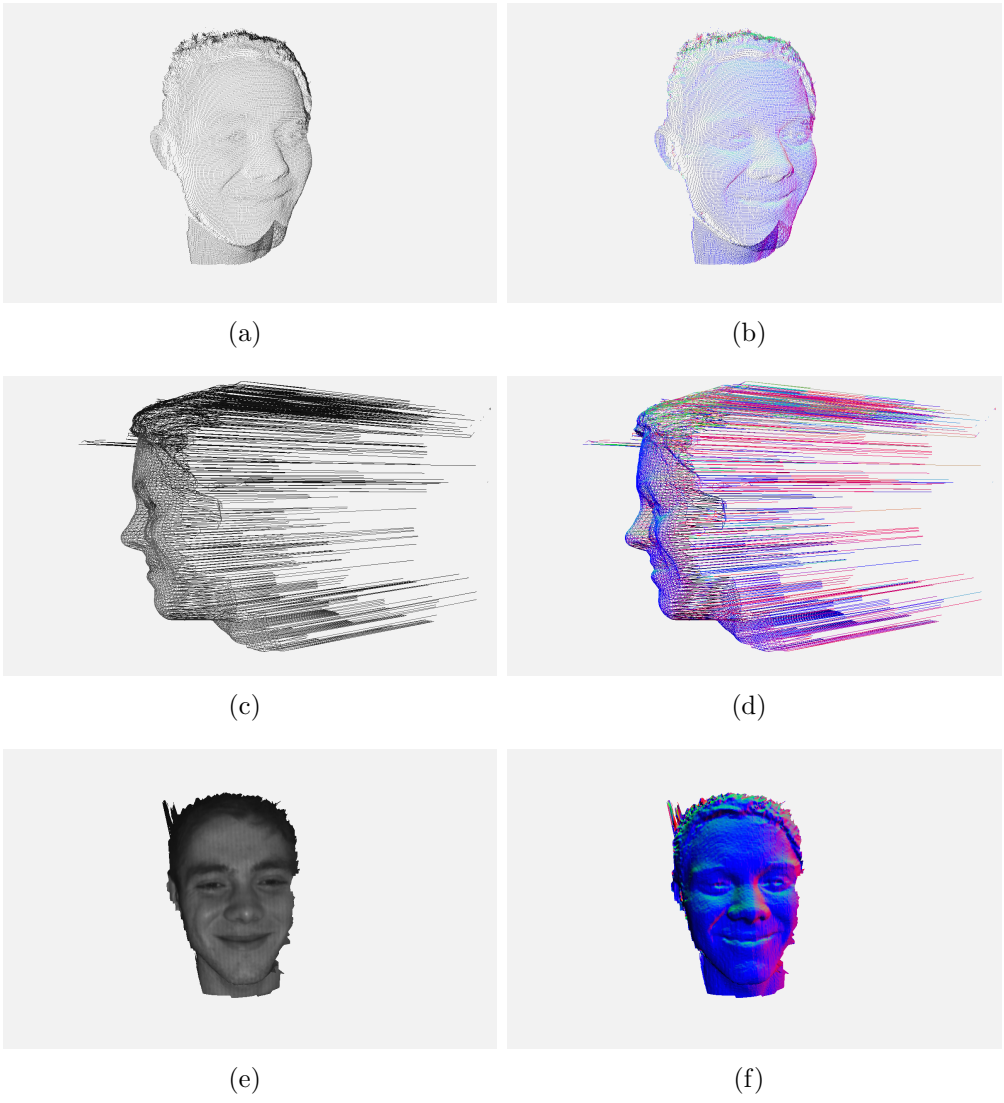
- *Alter 3D rendering style.* At the time of writing users can select to have their model rendered in a point-cloud, wireframe, or shaded-in mesh manner. The point cloud displays each 3D point when rendering whereas the shaded-in mesh displays what is perceived as more of a solid object by filling in the mesh created between points. The wireframe rendering style simply displays the raw mesh, or the connections created between the 3D points. This feature is controlled via the Settings control panel shown in Figure 4.11 and it dictates the type of *three.js* object should be

used to render the model. The different types of styles as rendered within WebGL on a web browser can be seen in screenshots of Nimbus in Figure 4.14. Note that some artifacts can be seen on the shaded, and especially the wireframe, rendering styles. This may be due to a few points being ignored and not being discarded by the fragment shader thus allowing triangles to be drawn back to them. The exact reason for these artifacts are unclear at the time of writing and a solution is being worked toward.

4.9 Discussions

Many modern technologies that deal in visualizing 3D geometry end up displaying it on a 2D plane: be it on a monitor, a specialized piece of glass, etc. The advantage of visualizing 3D data/video in an HMD comes in the form of immersion. There is an inherent level of immersion when wearing the headset as it occludes it's wearer's vision allowing them to focus on what is within the headset. By rendering the virtual scene in such a way that resembles natural vision, virtual 3D worlds appear to have a sense of realism. While immersive and natural, 3D video in an HMD is not without its downside.

One common application of this technology that's been extensively portrayed in this thesis is that of visualizing another individual. Although this visualization has been achieved, consider HMDs being used in a two-way communication system such as Portal-s. If each user had on an HMD and was telecommunicating via a Portal-s system then each would see the other's HMD when it was captured. A system such as the one which has been developed for this thesis research has the same fundamental disadvantage as the system described previously in Subsection 1.2.2 in that it really only works well for one of the two users participating in a telecommunication session. Despite this, there are several intriguing options for future work which are presented in Chapter 6.



(a) (b)

(c) (d)

(e) (f)

Figure 4.14 Different types of 3D geometry rendering styles incorporated within the Nimbus web-application. (a) Point cloud with texture; (b) Point cloud without texture; (c) Wireframe with texture; (d) Wireframe without texture; (e) Shaded mesh with texture; (f) Shaded mesh without texture.

4.10 Summary

This chapter presented the pipeline which enabled the creation of a system to visualize 3D video in the Oculus Rift HMD. If successful in each step of the pipeline—reconstructing the 3D model; rendering split-screen stereo such that there is an independent view for each eye; applying distortion correction; and incorporating natural head-tracking—the result is an immersive virtual environment in which 3D video can be consumed. The chapter also detailed how the visualization system itself had been expanded upon to allow for an increased level of user interaction. Chapter 5 will begin to look into how immersive an environment such as this really is based on preliminary user feedback.

CHAPTER 5. PRELIMINARY USER STUDY

In order to determine if a system that could visualize 3D video in an HMD is beneficial or not over existing 2D telecommunications, a preliminary user study was performed. The main objective of this evaluation was to gain very initial feedback on the system such that future features and directions could better be illuminated. This chapter presents the preliminary user study along with its results and discussions.

5.1 Experimental Setup

During the Graduate And Professional Student Research Conference at Iowa State University in April of 2014, demonstrations of the 3D HMD video system were given to attendees. Users were excited by the combination of high-quality 3D video and an HMD system; some even claimed how immersed they felt while visualizing 3D geometry in the system. In order to get initial data such that empirical evaluation of this sensation of immersion could begin, a preliminary user study was organized.

The user study took place in April 2014 at Iowa State University in the 3D Machine Vision Research Laboratory. In the study, participants experienced 3D video in an Oculus Rift. The visualization system had been implemented as described in Chapter 4 with users viewing the 3D video within the Oculus Rift HMD. In the demonstration system, users visualized a short 3D facial video clip in the HMD as the face made different facial expressions; an example of what this might look like can be seen in Figure 4.9. Participants had the option of viewing different angles of the face while moving their

head around (as head tracking was implemented); they could manipulate the camera's view on the geometry with a mouse and a keyboard; they could change the display type of the 3D geometry (e.g., point cloud, filled-in mesh); and they could toggle grayscale texture mapping on or off.

The user study consisted of 17 university students, 12 females and 5 males, between the ages of 17 and 24. After their experience, participants were asked a series of close- and open-ended questions (given in 5.1.1 and 5.1.2) on an online form. The essential goal of the questions was to elicit some disparity in usability and immersion between traditional 2D video platform (YouTube, Skype, FaceTime, etc.) and a 3D video system which utilizes an HMD for visualization purposes.

5.1.1 Close-ended Questions

The following questions, unless otherwise noted, were either answered on a close-ended scale from 1-7 (1 representing no immersion, 7 representing total immersion) or with a simple yes/no response. These questions in particular were asked to gain numeric data such that empirical evaluations could be performed.

1. How immersed did you feel the last time you participated in a 2D video chat / conference call?

2. How immersed did you feel in the 3D system in the Head Mounted Display (HMD)?

3. How well do you think this system simulated natural vision?

4. Did you experience any sort of motion illness or visual conflicts while immersed? (yes or no)

5. Would you use a 3D system such as this over existing 2D solutions for telecollaboration and/or telecommunication? (yes or no)

5.1.2 Open-ended Questions

The following questions were asked to gather insight into what the user was thinking while participating in the study. The goal of such opinion-based questions was to bring to the forefront what seemed inconsistent or "off" for the user.

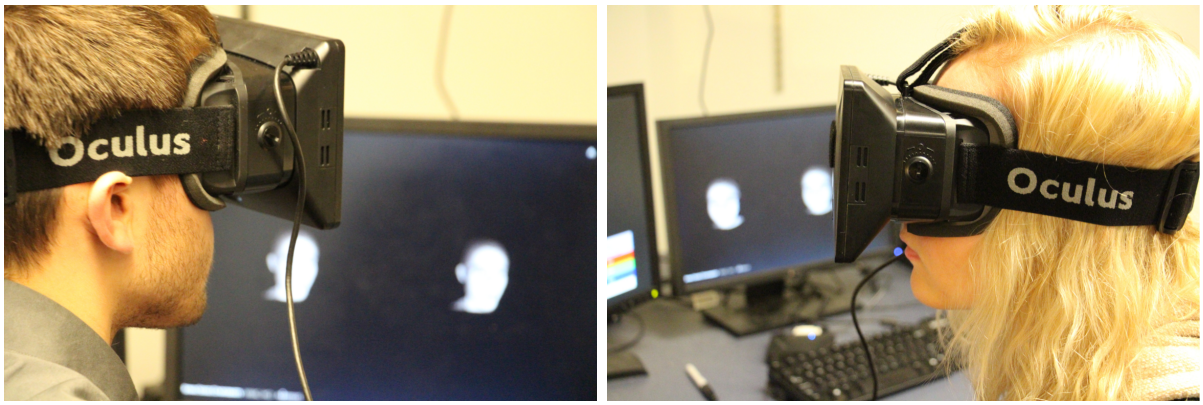
1. Please compare and contrast the differences, advantages, and disadvantages of your experience in today's 2D systems (Skype, FaceTime, etc.) vs. the 3D system with the HMD.
2. What did you enjoy about the 3D system with the HMD?
3. What did you not enjoy about the 3D system with the HMD?
4. Is there anything you'd like to see or thought was missing in the 3D system with the HMD?

5.2 Preliminary Results

The average rating for level of immersion in existing 2D systems, as experienced by the users, was 3.29/7 compared to 5.82/7 for the 3D HMD system used in the experiment based on the close-ended questions. Participants answered on average that the 3D HDM video system simulated natural vision with a rating of 5.65/7. Two of the 17 individuals answered "Yes" when asked if they experienced any sort of motion illness or visual conflicts while immersed with the demo in the Oculus Rift. 16 of the 17 individuals answered "Yes" when asked if they would use a 3D HMD system such as the one they used in the experiment over existing 2D solutions for telecommunication, such as Skype or FaceTime. The answers given for the close-ended questions to calculate the averages can be seen in Table .1.

When answering the open-ended questions, one user said that the 3D HMD video system allowed for "more effective [visual] communication with increased definition of

facial expressions." Another user said, "I could get a real emotional response from what I experienced in the simulator." Overall, the open-ended responses had a positive tone to them and elicited what the users enjoyed as well as some improvements they thought could be made.



(a) (b)

Figure 5.1 Users participating in the preliminary user study within the Oculus Rift HMD; a short 3D video was shown of the author's face.

5.3 Discussions

First, it should be noted that there was no 2D system setup for the users to experience before they used the 3D HMD video system. Before participating in the study, it was made sure that each user had recently used some 2D video/streaming platform (e.g., Skype) such that their responses could be more reliable. Therefore, when answering the questions about their experiences with 3D versus 2D platforms, past experience was their basis for comparison. A future user study would ensure a controlled group of users viewing a video in 2D while the experimental group would view it in the 3D HMD system.

That being said and as preliminary in nature as it was, this short user study proved to be very insightful. The close-ended questions were able to provided numerical data which pointed toward an overall increase in immersion. The 3D HMD video system offered a 71.73% improvement in level of immersion based on the gathered data. Also,

it can be argued that a rating of 5.65/7 in simulating natural vision is quite good as there is no real sense of natural vision in 2D platforms (at least the ones the users had exposure to).

The open-ended questions brought to light some very good ideas and elicited discussion from the users. Functionality that users cited they would like to see or they thought was missing in the experimental system was the ability to view the 3D video in color and with sound. It is a hypothesis that a user's level of immersion would increase substantially if improvements were made to the 3D scanning technology (such that color could be visualized) and to the 3D redisplay system (such that sound would be available alongside 3D geometry when viewing a video). Many of the users cited that the ability to see a person with a sense of natural 3D, and thus be able to see different angles as they moved their head, significantly improved their level of immersion and sense of connectedness to the author's model. Overall, with 16/17 individuals answering they would use a 3D video system within an HMD over existing 2D solutions, it appears that this area is worth investing more work into.

5.4 Summary

In this chapter, a preliminary user study and the system to be studied were explained along with the study's initial results. The study's aim was to gauge very initial user feedback and the responses indicate that a system such as the one in which participants viewed 3D video in the Oculus Rift is indeed immersive. All but one of the seventeen users reported that they'd use a 3D immersion system such as the one developed over current 2D systems and thus more exploration into this area should be performed.

CHAPTER 6. SUMMARY AND FUTURE WORK

This thesis has discussed state of the art 3D display technologies; introduced how 3D data can be acquired, compressed, transmitted, and redisplayed; summarized HMD technologies and how humans perceive depth in physical and virtual worlds; and presented its work on visualizing high-quality 3D video within an Oculus Rift HMD. This chapter summarizes these discussions and presents directions for future research to be conducted.

6.1 Research Achievements

As previously mentioned, this thesis research successfully merged high-quality 3D video technologies and the Oculus Rift HMD visualization platform in search of a more natural way to visualize and consume 3D video. The web-based application Nimbus was updated and improved upon such that users could have a wide array of options to choose from when it came to visualizing and interacting with 3D video. Also, the simple nature of Nimbus being built on web technologies helps its potential for dissemination to society as a whole. A preliminary user study was conducted to gather early findings into how users received 3D video visualization in an HMD. Although initial, the results were positive in that 16 / 17 users claimed that they would use a 3D video system in an HMD versus traditional 2D video methods they were familiar with, such as Skype or FaceTime. These introductory findings are very exciting in the context of this thesis work hence there are many future directions in which additional research can be performed.

6.2 Future Work

The possibilities for future work are many, especially as 3D, virtual reality, and HMD technologies become more commonplace. A few of the opportunities are listed below.

- *More encompassing user study.* As mentioned, the user study performed in this work was preliminary in nature. As the results of it were positive, a more controlled and encompassing user study could be performed as to really find out what was and what wasn't quite working out well for the users. Also, the study performed in this work was comprised only of university students. It could be interesting to gauge levels of immersion within this 3D video system in an HMD for different user demographics.

- *Live streaming on mobile devices.* It should be clear by the realization of 3D video in an HMD that 3D technologies are becoming more flexible and expandable due to many contributions in the field. One potential application of such technologies would be the ability to send, stream, and visualize live 3D data to and from mobile phones. This achievement would require developments along the entire capture, compression, transmission, and redisplay pipeline; yet, the potential adoption rate of such technologies in today's society is very high.

- *Improved immersion.* Immersion for users could be improved if certain key elements were added to the capture and/or display system. These improvements can include, but are not limited to: including color texture images, playback associated audio along with the 3D video, an implementation of a 3D GUI specifically for users within HMDs on Nimbus, and increased compatibility with interaction devices such that users within HMDs do not have to interact with a mouse or keyboard and may be able to use their voice, body, or hands, for example.

- *3D Collaboration.* This direction involves extending Nimbus such that multiple users with HMDs could interact within the same shared 3D space. The 3D object(s) representation would be the same for all of the users in the shared environment and would reflect any changes or manipulations made by the other users. A "toolbox" of sorts could be implemented such that users may cut/alter the geometry, measure it virutally, and more.

6.3 Summary

In final summary, high-quality 3D video has been visualized on the Oculus Rift HMD within an extendable web-based system which allows for greater interaction and improved sense of immersion for the user. Preliminary findings elude to the notion that visualizing 3D data in an HMD is more immersive than existing 2D telecollaboration solutions. This notion is promising for the future of the consumption of 3D, virtual reality, and HMD technologies. This being the case, several future research opportunities in these areas have presented.

APPENDIX.

Sex	**Age**	**Q. 1**	**Q. 2**	**Q. 3**	**Q. 4**	**Q. 5**
Male	17	1	4	6	No	Yes
Female	19	3	5	6	Yes	Yes
Female	24	3	6	5	No	Yes
Male	22	4	6	5	No	No
Male	22	4	6	4	No	Yes
Female	20	3	5	6	No	Yes
Female	19	5	6	6	No	Yes
Male	19	3	6	6	No	Yes
Female	19	4	7	4	No	Yes
Female	19	2	6	5	No	Yes
Female	20	4	6	5	No	Yes
Female	19	3	6	6	No	Yes
Female	19	2	6	7	Yes	Yes
Female	18	4	6	6	No	Yes
Female	21	3	6	5	No	Yes
Male	21	4	6	7	No	Yes
Female	21	4	6	7	No	Yes

Table .1 Data from the close-ended questions asked during the preliminary user study. Numerical responses represent level of immersion with 1 representing no immersion and 7 total immersion.

BIBLIOGRAPHY

Antonov, Mitchell, Reisse, Cooper, LaValle, and Katsev (2013). *SDK Overview*. Oculus VR, Inc., sdk version 0.2.5 edition.

Bell, T., Karpinsky, N., and Zhang, S. (2014). High-resolution, real-time 3d sensing with structured light techniques. In Bhowmik, A. K., editor, *Interactive Displays*. Wiley.

Davydov, A. (2010). Eye physiology, accommodation and convergence. http://www.forbestvision.com/accommodation-and-convergence/.

Ghiglia, D. and Pritt, M. (1998). *Two-Dimensional Phase Unwrapping: Theory, Algorithms, and Software*. Wiley-Interscience.

Gross, M., Würmlin, S., Naef, M., Lamboray, E., Spagno, C., Kunz, A., Koller-Meier, E., Svoboda, T., Van Gool, L., Lang, S., Strehlke, K., Moere, A. V., and Staadt, O. (2003). Blue-c: A spatially immersive display and 3d video portal for telepresence. *ACM Transactions on Graphics*, 22(3):819–827.

Gu, X., Zhang, S., Huang, P., Zhang, L., Yau, S.-T., and Martin, R. (2006). Holoimages. In *Proceedings of the 2006 ACM symposium on Solid and physical modeling*, pages 129–138. ACM.

Jones, A., Lang, M., Fyffe, G., Yu, X., Busch, J., McDowall, I., Bolas, M., and Debevec, P. (2009). Achieving eye contact in a one-to-many 3d video teleconferencing system. *ACM Transactions on Graphics*, 28(3):64:1–64:8.

Jones, A., McDowall, I., Yamada, H., Bolas, M., and Debevec, P. (2007). Rendering for an interactive 360 deg; light field display. *ACM Transactions on Graphics*, 26(3).

Karpinsky, N. (2011). 3d geometry compression with holoimage. Master's thesis, Iowa State University.

Karpinsky, N. (2013a). Photograph of portal-s hardware in use.

Karpinsky, N. (2013b). Photograph of user manipulating perspective on 3d data via leap motion controller.

Karpinsky, N. (2013c). *Portal-s: High-resolution real-time 3D video telepresence*. PhD thesis, Iowa State University.

Karpinsky, N., Hoke, M., Chen, V., and Zhang, S. (2014). High-resolution, real-time three-dimensional shape measurement on graphics processing unit. *Optical Engineering*, 53(2):024105.

Karpinsky, N. and Zhang, S. (2010). Composite phase-shifting algorithm for three-dimensional shape compression. *Optical Engineering*, 49(6):063604–063604–6.

Karpinsky, N. and Zhang, S. (2012a). High-resolution, real-time 3d imaging with fringe analysis. *Journal of Real-Time Image Processing*, 7(1):55–66.

Karpinsky, N. and Zhang, S. (2012b). Holovideo: Real-time 3d range video encoding and decoding on {GPU}. *Optics and Lasers in Engineering*, 50(2):280 – 286.

Karpinsky, N. and Zhang, S. (2013). 3d range geometry video compression with the h. 264 codec. *Optics and Lasers in Engineering*, 51(5):620–625.

McGuire, M. (2008). A fast, small-radius gpu median filter. In *ShaderX6*.

Minsky, M. (1980). Telepresence. *Omni*, 2(9):45–52.

Motion, L. (2014a). Leap motion controller. https://www.leapmotion.com/product.

Motion, L. (2014b). Leap motion developer portal. https://developer.leapmotion.com.

NMU.edu (2013). Retinal disparity. http://art.nmu.edu/groups/cognates/wiki/1617e/Retinal_Disparity.html.

Salvi, J., Pages, J., and Batlle, J. (2004). Pattern codification strategies in structured light systems. *Pattern Recognition*, 37(4):827–849.

Vislogix (2013). Vistique hologlass. http://vislogix.com/solutions/projection/transparent/vistique-hologlass/.

Wickens, C., Lee, J., Liu, Y., and Gordon-Becker, S. (2004). *Introduction to Human Factors Engineering, 2/E*. Pearson, Upper Saddle River, New Jersey.

Yao, Heath, Davies, Forsyth, Mitchell, and Hoberman (2014). *Oculus VR Best Practices Guide*. Oculus VR, Inc., v0.008 edition.

Zhang, S. (2010). Recent progresses on real-time 3d shape measurement using digital fringe projection techniques. *Optics and Lasers in Engineering*, 48(2):149 – 158. Fringe Projection Techniques.

Zhang, S. and Huang, P. S. (2006). Novel method for structured light system calibration. *Optical Engineering*, 45:083601.