Calibration Methods for Head-Tracked 3D Displays

A Thesis Submitted to the
College of Graduate and Postdoctoral Studies
in Partial Fulfillment of the Requirements
for the degree of Master of Science
in the Department of Computer Science
University of Saskatchewan
Saskatoon

By
Andrew Wagemakers

Permission to Use

In presenting this thesis in partial fulfilment of the requirements for a Postgraduate degree from the University of Saskatchewan, I agree that the Libraries of this University may make it freely available for inspection. I further agree that permission for copying of this thesis in any manner, in whole or in part, for scholarly purposes may be granted by the professor or professors who supervised my thesis work or, in their absence, by the Head of the Department or the Dean of the College in which my thesis work was done. It is understood that any copying or publication or use of this thesis or parts thereof for financial gain shall not be allowed without my written permission. It is also understood that due recognition shall be given to me and to the University of Saskatchewan in any scholarly use which may be made of any material in my thesis.

Requests for permission to copy or to make other use of material in this thesis in whole or part should be addressed to:

Head of the Department of Computer Science
176 Thorvaldson Building
110 Science Place
University of Saskatchewan
Saskatoon, Saskatchewan
Canada
S7N 5C9

Abstract

Head-tracked 3D displays can provide a compelling 3D effect, but even small inaccuracies in the calibration of the participants viewpoint to the display can disrupt the 3D illusion. We propose a novel interactive procedure for a participant to easily and accurately calibrate a head-tracked display by visually aligning patterns across a multi-screen display. Head-tracker measurements are then calibrated to these known viewpoints. We conducted a user study to evaluate the effectiveness of different visual patterns and different display shapes. We found that the easiest to align shape was the spherical display and the best calibration pattern was the combination of circles and lines. We performed a quantitative camera-based calibration of a cubic display and found visual calibration outperformed manual tuning and generated viewpoint calibrations accurate to within a degree. Our work removes the usual, burdensome step of manual calibration when using head- tracked displays and paves the way for wider adoption of this inexpensive and effective 3D display technology.

Acknowledgements

I would like to say thank you to my supervisor, Dr. Ian Stavness, for his guidance and help throughout my graduate studies.

This thesis is dedicated to my parents: John and Shirley.

Co-Authorship Statement

Parts of this thesis have appeared in published work. Versions of Chapter 2, Chapter 5, and Chapter 6 have been published in Wagemakers, Fafard, and Stavness [56]. I formulated the viewpoint calibration algorithm with Mr. Fafard. Mr. Fafard designed and carried out the studies described in Sections 5.3.4 and 5.4. I designed the viewpoint calibration algorithm, constructed the pattern alignment study, and wrote the text for the manuscript under the guidance of Dr. Stavness.

Contents

List of Figures

List of Abbreviations

3D	Three Dimensional
2D	Two Dimensional
ANOVA	Analysis of Variance
AR	Augmented Reality
CAD	Computer Aided Design
CAVE	CAVE Automatic Virtual Environment
CPU	Central Processing Unit
DoF	Degrees of Freedom
EM	Electromagnetic
FTVR	Fish Tank Virtual Reality
HDMI	High-Definition Multimedia Interface
HMD	Head Mounted Display
IR	Infrared
LCD	Liquid Crystal Display
QR	Quick Response
RF	Radio Frequency
SPAAM	Single Point Axis Alignment Method
VR	Virtual Reality

Chapter 1

Introduction

Commercial 3D displays have eluded wide spread adoption. Glasses-based stereoscopic televisions have declined in popularity and the 3D feature is no longer offered on many new models. Virtual and augmented reality headsets, such as Oculus Rift and Microsoft HoloLens, have had a resurgence in recent years, but these systems require bulky headgear and either block out the physical world or lack the physicality of a real display surface. Head-tracked displays have a number of benefits over traditional 3D displays that give them promising potential for a 3D display situated within a physical work or living space, but their realization has been hampered by a lack of fast and accurate calibration methods to make the head-tracking illusion work effectively. This thesis will explore new calibration techniques for head-tracked displays that overcome past challenges, and will hopefully expand the use of head-tracked displays in work and play.

1.1 Sources of 3D Data

Recently, there have been a large increase in sources of 3D data. The entertainment industry has exploded with recent advances in technology. Movies entertain people by showing them what life is like for different individuals. These movies are created to instill different emotions into the audience's minds, whether they are positive or negative. Movie creators often add non-realistic concepts into the movies to show what it could be like if the world functioned differently. This allows them to be more creative with their ideas as they can modify physics,

show what life might be like in a distant galaxy, or even show the world from an animal's point of view. Since many of these ideas cannot be recorded in real life, they must be created digitally.

Similar to movies, video games also show what it would be like to be someone else, the main difference is the person playing the game has control over the character's actions. These games allow people to live out their dreams of being a professional athlete or experience the horror of surviving a zombie apocalypse. Since there is no way to determine what the person is going to do, the data must be generated in real time in order to respond to the person's inputs. Realistic character models are used to make video games and movies appear as realistic as possible. The motions of the characters are usually based on the motions of real people. Motion capture hardware is used to record how people do different actions. Markers are worn on an individual as they act out an action. The location of all of the markers are then recorded over time and the sequence of position data is used to control a character.

3D scanners and 3D cameras can capture the 3D position and shape of objects which allows for physical objects to be used in computer calculations. CAD software allows for the creation of virtual objects that could later be turned into physical objects. When designing new products, companies will generally create a 3D model of it before it is built. This allows for a faster design process and promotes modifications until a satisfactory result is obtained. With large construction or complex machinery projects, CAD software allows the design to be broken down into smaller parts. This allows for collaborative design and makes the process less complicated as the software can keep track of the assembly of parts. Once a product is finished, it can be run through a simulation to see how it would perform if it were to be placed in the real world. Vehicle designers can run aerodynamic simulations to see if their new design is more fuel efficient, and building designers can simulate an earthquake to see how well their building survives. These simulations can also be used to predict the outcome of future events. Weather forecasting works by applying physics equations to the current state of the weather to determine how it will change in the future. These simulations can also be used to predict how a medical patient will react to different procedures, allowing for the most appropriate procedure to be done. 3D data has become a major part of our lives

and being able to interpret and understand this information is necessary to maintain our current way of life.

1.2 Visual Cues

3D graphics represents the process of taking 3D data and presenting it onto a 2D screen. The 3D data that is being presented is generally information on the position of an object in 3D space. Positional information is commonly expressed in a Cartesian coordinate system with three different orthogonal axes. How the points get transformed to the screen is up to the person writing the program. Non-realistic techniques can be used to provide an artistic feel to the content, but the most common method is to try to emulate how people see objects in real life. This type of accurate rendering is desired in many different applications where the real world is being modeled. Being able to realistically express 3D data on a two dimensional screen is reliant on our brain picking up on different visual cues that we are used to seeing in day to day life.

1.2.1 Perspective

The most common visual cue, seen in almost all 3D content, is the perspective visual cue. This is the visual cue that makes objects farther away appear to be smaller than objects close to an individual. The reason why this cue is present is because our eyes function like perspective cameras. The viewing frustum of a perspective camera resembles the shape of a pyramid, with all light converging at a single point. Since all of the light converges to a single point, any information about absolute size of an object is lost and the apparent size is inversely proportional to the distance the object is from the camera. This effect can be seen in Figure 1.1a, the trees on the far right and left of the image appear much larger than the ones closer to the center. Since the trees appear to be of the same type we would expect them to be of comparable size, this indicates to us that the trees near the center of the image must be

further away than the ones on the edge of the image. We can also see the rails of the railway tracks appear to get closer together, this is another effect of perspective cameras. Parallel lines will appear to converge as they get further from the camera, they will also appear to vanish at a point in the image referred to as the vanishing point. Because of the way modern rendering pipeline is structured, the perspective cue can be easily implemented.

1.2.2 Stereopsis

Another visual cue that contributes to perceiving 3D is the stereo visual cue. This cue is the result of observing and object from two different viewpoints simultaneously, as is done with our eyes. Since each of our eyes can be represented as a perspective camera, there is a ray that intersects all objects that appear in the center of the image. When our eyes focus on an object, the rays will intersect at that point. Any object in front of where the eyes are focusing will not be in the center of the viewed images. The left eye will see the object more to the right, and the right eye will see it more to the left. The same effect occurs when an object is behind where the eyes are focused the only difference is that the left eye will see the object more to the left and the right eye will see it more to the right. This disparity in perceived image location informs us of how far away something is from where we are focusing, the larger the separation means a larger distance. The stereo cue can be seen in Figure 1.1b.

1.2.3 Motion Parallax

A third visual cue is motion parallax. Motion parallax is observed when the observer is moving relative to what they are viewing. Objects that are closer to the viewer will appear to move faster than those that are further away. In the situation where there are multiple stationary objects, they will all have the same velocity relative to the viewer. As the viewer walks close by an object, it will appear to be moving the fastest because it will move from one side of the viewers frustum to the other in the least amount of distance. Objects further

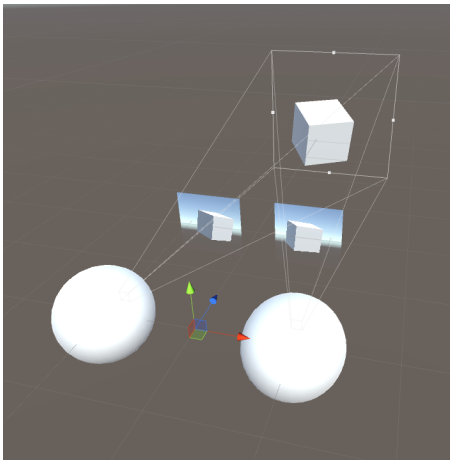
away will appear to have moved a smaller distance, even though the viewer moved the same distance relative to both objects. This effect can be seen in Figure 1.1c.

1.2.4 Occlusion and Lighting

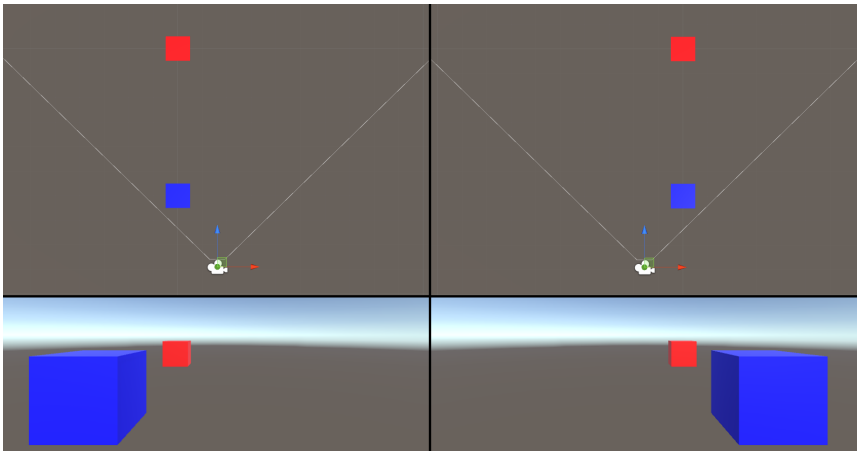
Occlusion can give a large amount of information about the relative depth of multiple objects. Occlusion occurs when an object is positioned in front of another object. Since the object in front will block the light coming from the back object, the viewer will no longer be able to see the back object. Occlusion does not tell us anything about absolute depth, only relative depth information is gained. We also gain a lot of depth information from how light interacts with an object. Lighting can tell us a lot about the surface of an object, especially if we know where the light is originating from. We are able to gain useful information about the shape of an object, but how the light reflects tells us about the surface texture of the object as well. Objects with smooth surfaces will have small bright spots that move as the viewer moves, where objects with rough surfaces will have a larger area that is less bright and is stationary. Lighting can also provide information about how objects are positioned relative to each other in a way similar to occlusion. Shadows are caused when an object blocks the light that is originating from a light source. When the shadow of one object is seen on another object, you can determine which object is closer to the light source.

Display and content designers can use these visual cues to their advantage in order to make displays that appear to contain 3D objects and scenes. Since people perceive depth from the combination of cues and 2D images, illusions of 3D can be generated by presenting 2D images generated with these cues in mind. This concept was initially presented during the Renaissance, where artists introduced perspective into paintings to give a notion of depth. Converging lines were accurately drawn to create an image that would match what our perspective eyes would see. These same concepts were utilized hundreds of years later with the introduction of the first 3D video game. Being able to populate a virtual environment with objects defined in 3D space, and presenting them on a screen as if a person was there viewing them, allowed for the development of realistic games. Since then, computer systems

(a) Perspective [52]



(b) Stereopsis



(c) Motion parallax

Figure 1.1: 3D visual cues

have been optimized to do calculations to determine where on the screen a 3D object should be rendered. Visual cues such as perspective, motion parallax, and occlusion are so routinely used that they are standard in all graphics based hardware.

1.3 3D Displays

As hardware advanced, people began to change the displays to help introduce a stronger feeling of 3D. Three noteworthy display designs have been created: glasses-based stereoscopic displays, head-mounted displays, and FTVR displays.

1.3.1 Stereoscopic Displays

Glasses-based stereoscopic displays utilize stereo cues to generate the 3D illusion, the challenge with this cue is that both eyes need to see different content. When looking at objects on a flat TV screen or computer monitor, both eyes are able to see the same content so the stereo cue is not present. Stereoscopic content can be created by displaying both what the left eye should see and what the right eye should see simultaneously, while having the user wear a pair of glasses that can filter out each eye's content. There are three main techniques that are used to filter each eye's content. The first method is by using anaglyph 3D glasses, shown in Figure 1.2c, which are made up of two filters of opposite colors; the most standard colors are red and cyan. To filter the contents of the opposing eye, content is rendered in the color of the filters. Red content will not be visible through the cyan filter and blue or green content will not be visible through the red filter. The downside to this method is that there will be a restricted color spectrum visible as certain colors are absorbed by the filters.

The second method, which is similar to the first, utilizes light polarization to filter the appropriate images. The glasses are made up of two polarizers with perpendicular polarizations, resulting in only light with the appropriate polarization to pass to each eye. The display

must then be able to produce two images of opposing polarizations, which standard monitors and televisions are unable to do. The two images are produced by projecting two different images through polarizers of perpendicular polarizations. The downside of this method is that it requires specialized projecting equipment, which is generally very expensive.

The previous two methods are referred to as passive stereo because the glasses do not need to actively change to do the filtering. The third method is referred to as active stereo since the glasses need to actively change to filter the appropriate images. Active stereo utilizes shutter glasses to block certain content from reaching each eye. Shutter glasses have the ability to block all light from passing through a lens by applying voltage across a liquid crystal material. The glasses can toggle which eye can see in order to match up with the displayed image. The displayed image must be able to refresh fast enough for both eyes to see a continuous stream of images; usually 120 frames per second is used. The display will switch between what the left and right eye should see, and a signal is sent to the glasses telling them to alternate eyes at the same rate. The downside of this method is that the system must be constantly communicating between the glasses and display.

1.3.2 Head-mounted Displays

Head-mounted displays use both stereo and motion parallax cues for an enhanced 3D illusion. Head-mounted displays function differently than traditional displays because the screens are physically attached to the viewer. A headset is worn over the eyes of the viewer that presents different images to each eye to produce the stereoscopic effect. A single screen is placed in front of each eye that display different content. By doing this, no specialized image filtering is required. The headset is also equipped with some form of tracking hardware which usually consists of a combination of optical trackers, accelerometers, magnetometers, and gyroscopes. By tracking the user, the location and orientation of their head can be determined, and a virtual head can be placed inside of the virtual environment. As the user moves and rotates their head, a pair of virtual cameras can be positioned to mimic the actions of the user. As a result, each eye will see exactly what they would if the viewer was in the virtual scene.

By making the perspective of the scene change as the viewer moves, the correct perspective can be seen, resulting in a stronger sense of motion parallax. Most head mounted displays can be classified into two different categories: virtual reality headsets, and augmented reality headsets.

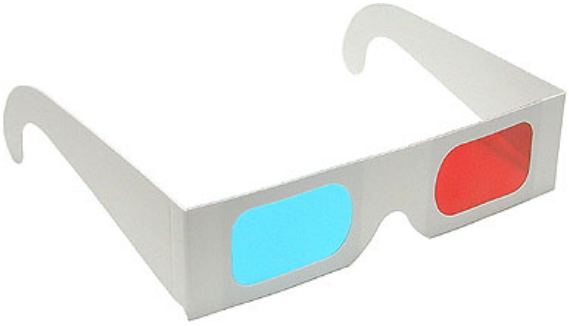
Virtual reality headsets, such as the Oculus Rift (Figure 1.2a) and HTC Vive, place screens in front of the eyes that completely block any external light from entering. These types of displays have recently become extremely popular, as many big video game titles have been released for them. These headsets work exceptionally well when the content is intended to take up all of the user's attention, such as certain video games or movies, but they may be less ideal for other circumstances. The user has no vision of their surrounding, which can make them feel detached from the objects and people around them. For this reason, virtual reality headsets can be good if used in a secluded environment, but they may be undesirable if the user is in a more public environment.

Augmented reality headsets help resolve this problem by allowing the user to see through the headset and overlaying virtual objects over-top of the physical objects. With augmented reality headsets, people could navigate a cluttered environment and even work collaboratively with others. This type of technology has great potential, but is still in early development. A lot of research is being done at Microsoft to develop the HoloLens (Figure 1.2b), which aims to provide a revolutionary augmented reality experience.

Smart phone applications, such as Pokemon Go (Figure 1.2d), provide an entertaining augmented reality experience. The camera on the phone is used to capture a video feed of the environment. Virtual characters are then overlayed on the video feed. By utilizing accelerometers in the phone, the characters can be made to appear stationary as the user moves their phone around. QR codes are commonly used to appropriately position and orient virtual objects as well. Software packages exist that allow for the development of augmented reality phone applications [3].

(a) Head-mounted VR [4]

(b) Head-mounted AR [38]



(c) Stereoscopic glasses [1]

(d) Hand-held AR [43]

Figure 1.2: Types of 3D displays

1.3.3 FTVR Displays

Fish Tank Virtual Reality (FTVR) displays can present a perspective corrected view of a virtual environment that appears to exist inside the display. FTVR displays utilize motion parallax cues to generate a 3D illusion. Head-tracking technology is used to determine where the display is being viewed from, which must be known to render the correct perspective. The motion parallax cue can be very strong if the display allows for a large variety of viewing angles. Multi-screen FTVR displays can offer a large variety of viewing angles by utilizing multiple screens each with different orientations. These displays can be made in many possible shapes and sizes; two common shapes are spherical [20] and cubic [50] shapes. Cubic displays are made up of a number of flat screens in the shape of a box. They generally have five screens, each facing perpendicular to its neighboring screen. These displays can be easily constructed because they do not require any specialized hardware; the display could be made up of computer monitors or tablets. A calibration step is needed to determine how each

of the screens are positioned and oriented relative to each other. Spherical displays are generally made up of a plastic sphere which is illuminated by one or multiple projectors. These displays are more challenging to construct as they require a special type of material to project onto. Since the projected areas can be overlapped, spherical displays lack the occlusion caused by the screen borders of the cubic display. Spherical displays require an extensive calibration process to determine where each projector pixel is on the surface of the sphere. If the hardware is available, any head-tracked display can use stereoscopic glasses to provide a stronger 3D effect.

These three display types have different hardware setups which make them feasible in different situations. The glasses-based stereoscopic displays have the ability to support a large number of viewers. Head-mounted displays are designed to work only for a single viewer as the display is worn on their head, and FTVR displays will only look proper from the tracked location. With active shutter glasses, FTVR displays can be made to work with multiple viewers. Glasses-based stereoscopic displays are commonly used today for 3D movies because they can be shown to a large number of people simultaneously. Head-mounted displays have the ability to provide an unparalleled level of immersion, which is the reason why they've been adopted into the gaming industry.

FTVR displays do not have the ability to compete with the other display types when it comes to mass entertainment or immersive game-play. However, head-tracked displays can provide a natural viewing experience for looking at contained 3D content. These displays would work great as a second monitor which could be used to visualize any 3D content. For example, in a CAD scenario, orthographic views could be presented on a computer monitor while the 3D perspective view could be presented on a FTVR display. 3D simulations would be enhanced by FTVR displays because you could observe the simulation from any viewing direction. By utilizing a third spatial axis, high dimensional data could be presented in a manner that is easier to understand. Finally, video games have a lot of potential for these types of displays. Head tracking provides a more immersive viewing experience and new possibilities for interaction.

Importance of Visual Cues

The incorporation of visual cues into a display provides more information about the location of virtual objects. Different cues have been found to provide more information than others. Ware et al. [57] found that a combination of head-tracking and stereo rendering provides the most information, but only having head-tracking is more informative than only having stereo. Head-tracking can provide valuable information, but only if it is accurately calibrated. Zhou et al. [63] performed an analysis on the effects of head-tracker error and projector alignment error for a spherical FTVR display. They found that head-tracking error caused significantly more visual error than projector alignment error. For these reasons, it is important to have an accurately calibrated head-tracker system when rendering on a FTVR display.

1.4 Problem

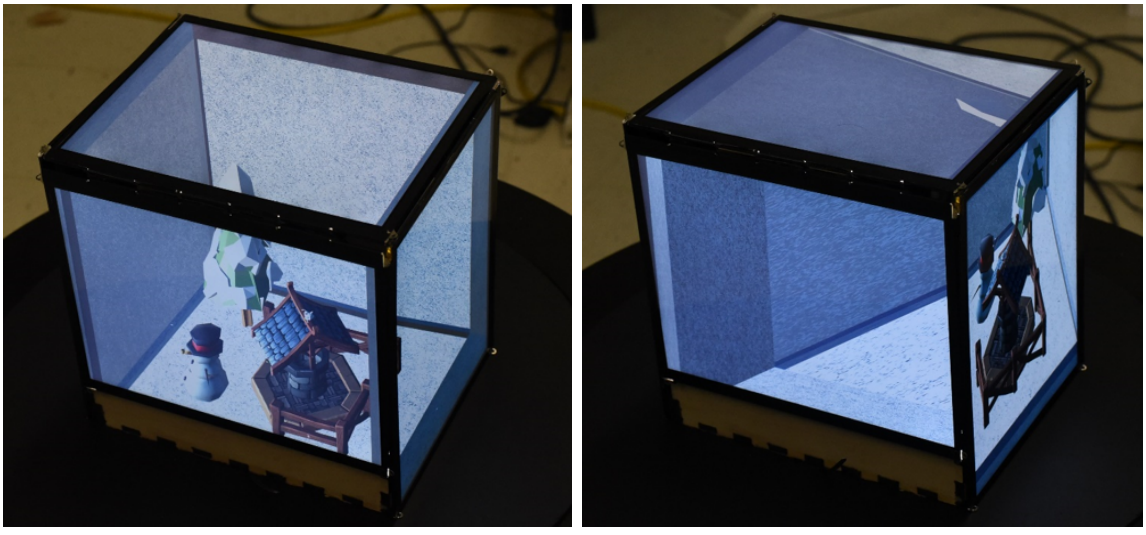
The principal drawback of FTVR displays is that the user's viewpoint must be accurately measured in real-time with respect to each screen of the display. Even small inaccuracies in the rendered viewpoint can create visual artifacts (kinked lines, oddly floating objects, and ghosting) that can immediately and severely disrupt the 3D effect of the display; these effects are shown in Figures 1.3 and 1.4. Multiple unknowns must be solved for in order to determine where the viewpoint is relative to each screen. Therefor, a calibration process must be done to be able to do the rendering. The calibration process can be broken down into two steps: screen calibration, and viewpoint calibration. Screen calibration is necessary to obtain accurate pixel mappings between each screen. This process is dependent on the shape of the display being rendered to. Viewpoint calibration is necessary to render a virtual scene from the correct perspective. In order to be able to render content inside of the display, both calibration steps must be accurately done. If the tracker is not rigidly attached to the display, calibration will need to be redone if either the display or tracker is moved. For this reason, a quick and easy calibration technique is needed, which is fine tuned for each user, to allow for accurate rendering.

(a) Screen alignment errors (ghosting)

(b) No errors

(c) Viewpoint position errors (warping)

Figure 1.3: FTVR calibration errors on a spherical display



(a) No errors

(b) Viewpoint position errors (warping)

Figure 1.4: FTVR calibration errors on a cubic display

1.5 Solution

In this thesis, we propose a novel user-friendly procedure to easily and accurately calibrate a multi-screen FTVR display. The procedure estimates both the location and orientation of the display, and the location of user's viewpoint relative to a head tracker. It does not require any physical markers in the work-space. It instead uses visual patterns rendered across the multi-screen display that are designed to appear aligned when viewed from a correct viewpoint, but to appear distorted otherwise. In this way, we can compel participants to place their viewpoint in known locations relative to the display. We can then measure the tracked position of the head at each location to calibrate the display. This method is similar to the SPAAM method for AR headset calibration [53].

A camera based approach was also developed to achieve accurate screen calibration on a cubic display. The process involves rendering chessboard patterns on the surface of each screen while taking pictures of the display from multiple viewing directions. Camera calibration techniques can then be used to calculate the orientation and location of each screen relative to each other.

1.6 Solution Steps

In order to understand and develop the viewpoint calibration method, several steps were completed during the research process.

1.6.1 Build Head-Tracked Displays

Two different display models were constructed to test our rendering and calibration algorithms. A cubic display was assembled out of five tablet screens and a wooden frame. A spherical display was made out of a plastic sphere with multiple projectors mounted under-

neath.

1.6.2 Develop Screen Calibration Algorithm for Planar Displays

In order to test the viewpoint calibration algorithm on a physical display, the display needed to be accurately calibrated. A camera based display calibration method was developed for displays made of planar screens, which was tested and used on a cubic display.

1.6.3 Test Spherical Display Projector Calibration Algorithm

In order to be able to render onto a spherical display, we needed to be able to calibrate the display. We used the algorithm presented by Zhou et al. [62] to calibrate each projector to the surface of the sphere.

1.6.4 Develop Unified Rendering Program for Both Display Types

A program was developed using the Unity game engine to be able to render to both planar displays and spherical displays. The program was designed to work with both display shapes such that content could be created and easily deployed into either display.

1.6.5 Develop Viewpoint Calibration Algorithm

The location of the viewpoint must be known relative to the display to render perspective corrected content. For this reason, an accurate viewpoint calibration method is needed to appropriately position the viewpoint relative to the display. The proposed method involves

rendering static patterns on the surface of the display that will only appear undistorted from a specific location. The calibrator will position their viewpoint in the location that results in the least amount of perceived distortion, and the tracked location is recorded. The two sets of points can then be used to calculate the location and orientation of the display relative to the tracking system, as well as the offset from the tracked marker to the user's viewpoint.

1.6.6 Determine Most Effective Alignment Conditions

In order to better understand the viewpoint calibration process, a virtual environment was created that simulates the alignment process used in the viewpoint calibration method. A computer mouse is used to revolve a camera around a 3D shape at a fixed distance. The shape has an easily recognizable pattern texture mapped onto it which is rendered to appear correct from a single location. A study was run that examined how people interacted with the virtual calibration environment. The speed and accuracy that each participant could align the camera with the pattern was recorded in order to determine the most effective pattern and display shape.

1.6.7 Compare Visual Viewpoint Calibration to Manual Viewpoint Calibration

In order to compare the effectiveness of the visual viewpoint calibration to alternative methods, an accuracy metric was developed. This metric involves rendering a pattern on the display that should look undistorted from any location, and taking pictures to evaluate the amount of distortion present. To validate the effectiveness of the viewpoint calibration algorithm, it was tested against a manual calibration technique. The comparison involved performing both calibrations, then the transformation accuracy metric was used to determine which method performed the best.

1.7 Evaluation

We conducted a user study to evaluate different candidate visual patterns (concentric circles, a grid, and a combination of circles and lines) on simulated spherical, cubic, and box-shaped displays. We found that participants were fastest and most accurate when aligning patterns on a spherical display, and that the combined circles and lines pattern was most accurately and quickly aligned across display shapes. We evaluated our procedure with a physical cubic display using a semi-automatic camera-based analysis. The analysis demonstrated that the method generates an accurate viewpoint calibration, regardless of head-tracker orientation, that accounts for a head-to-viewpoint offset and is invariant of depth errors.

1.8 Contributions

This work makes five main contributions. First, we constructed spherical and cubic FTVR displays. Second, we present a display calibration technique that is useful for calibrating displays composed of multiple planar screens. Third, we present a novel viewpoint calibration method for FTVR displays that estimates both the tracker-to-display transformation and the head-to-viewpoint offset. Fourth, we present a method for evaluating viewpoint calibration quality for cubic FTVR displays. Fifth, we evaluated the effect of rendered patterns and display shapes on the visual viewpoint calibration process.

1.9 Outline

Chapter two describes existing virtual reality rendering devices and how they are calibrated. Head mounted displays and see-through head mounted displays, as well as the calibration techniques for them, will be discussed. Displays that are not head mounted, but rather track the location of the user's head to present perspective corrected content, will then be examined.

The chapter will end with different methods of calibrating head tracked displays.

Chapter three will cover the process of rendering to multi-screen FTVR displays and mention what unknowns are required. It will then present the steps needed for rendering on a display made up of planar screens, as well as for a spherical display. The chapter ends with an explanation of the necessary modifications for stereoscopic rendering.

Chapter four will detail the displays that were constructed, as well as the screen calibration process for them. The mathematics behind camera calibration will be explained, as well as how it relates to calibrating a projector. The chapter will then present how a calibrated camera can be used to determine the location of a pixel on the constructed display.

Chapter five will go over the viewpoint calibration process. A pattern based approach for calibrating FTVR displays will be presented alongside two studies that were designed to validate the method. These studies were designed to test the depth invariant nature of the process and to compare the performance of it to a manual calibration.

Chapter six will detail the user study that was performed to better understand the visual alignment process. The study was designed to evaluate people's ability to perform the visual alignment process with different combinations of display shapes and rendered patterns. To do this, we had participants perform a simulated version of the alignment process to see how close to the ideal camera position they could get.

Chapter seven will conclude the thesis and go over potential future work ideas.

Chapter 2

Related Work

In this chapter, we cover the two main areas that serve as the background of our work. First, we look at the different types of displays which can render FTVR and AR content. Second, we cover different methods for calibrating these types of displays, which allow for the rendering of perspective correct content.

2.1 Head Mounted Displays

HMDs are worn on the head of the user with screens placed in front of the eyes. These displays have become very popular with the recent release of commercially available products (Oculus, Vive, Playstation). HMDs offer an immersive experience to the user by replicating what they would see if they were in a virtual environment. This is beneficial in that it provides a novel interaction experience, but it has also been found to facilitate improved searching performance [44]. The degree of immersion also detaches the user from their surroundings and has been known to cause feelings of nausea [37, 49]. Research has been done to better understand how to prevent these feelings [40]. See-through HMDs [19] allow the user to see their surroundings, removing the sensation of detachment.

2.2 Augmented Reality

AR displays can be any display that presents a view of the world which has been altered in some way. The two main devices which can present AR content are see-through HMDs (HoloLens) and mobile devices, such as cell phones. Research has been performed to evaluate the effectiveness of AR displays when working collaboratively [10]. AR mobile apps have become very popular for both entertainment and education. Games such as Pokemon Go allow people to capture creatures that appear in different geographical locations. These kind of games are beneficial because they encourage the player to be physically active. People will have to routinely walk around in order to capture different creatures and remain in control of certain territories. Mobile phone apps can also provide additional information about an individuals surroundings. This can be beneficial when visiting a new place [17, 34], or when working with complex machinery [35], or even in the education system [60].

Another method to present AR content is by directly projecting onto the user's surroundings. ShaderLamps [47] brought inanimate objects to life by projecting dynamic animations onto stationary objects. Microsoft Room Alive [28] turns the player into the controller as they directly interact with the projected content. A Microsoft Kinect is used to track the body of the player in order to present perspective corrected content and determine how they are interacting with the content. Microsoft Room Alive is publicly available and can be used on any room geometry; the system can scan objects in the room and properly project onto them. These two examples utilize fixed projectors, but a handheld projector can be used if QR codes are present [46, 58].

2.3 Head Tracked Displays

Head-tracked 3D displays that used stereoscopic and motion parallax cues were first introduced as a practical way to view 3D data on a desktop monitor [5]. This concept, termed Fish Tank Virtual Reality (FTVR), was proposed as an alternative to HMDs and immersive

VR rooms [15] of the time. The original user studies with FTVR displays demonstrated the surprising result that head-tracking provided stronger 3D cues than stereo alone [57]. An issue with early FTVR systems was the limited range of viewing angles. The user had to remain in front of the single monitor, thereby limiting their head motion, and consequently reducing the strength of motion parallax depth cues. Multi-screen FTVR displays have been developed that allow for a large variety of viewing angles, including 360 degree viewing around an enclosed volumetric display. Different configurations include: inward facing box [18], outward facing box [26, 50], tabletop [14, 24], cylinder [30], and sphere [11, 20]. Head-tracking has also been used to enhance video game experiences [32] and adapt stereoscopic display parameters [33].

2.4 AR Calibration

AR displays overlay virtual imagery onto real-world environments, either by projecting onto objects or using a see-through HMD. Calibration errors of the users viewpoint to the physical world in AR are highly noticeable and result in misalignment or ghosting of the virtual overlay. For this reason, static [7] and dynamic [8, 21, 27, 29] calibration schemes have been well studied in AR, including detailed quantitative error analysis [25, 39]. An advantage of HMDs is that the user's eye locations can be accurately measured while using the HMD device because of its close proximity to the user's eyes [45]. The semi-transparent screen of see-through HMDs also allows them to be visually aligned similar to a rifle scope, as proposed in the SPAAM method [53]. This method involves visually overlapping a virtual marker on the HMD screen with a physical marker at a known location in the head-tracker's reference frame. Measurements with different virtual markers allow for estimation of the tracker-to-screen planar homography, and the method can be extended to binocular viewing.

2.5 FTVR Calibration

Many multi-screen FTVR displays use multiple projectors to illuminate a seamless geometric display surface. However, these require careful screen calibration so that overlapping projector geometry can render without visual artifacts, such as ghosting or brightness disparity. Multi-projector calibration procedures for planar [59] and curved [48, 62] display surfaces have been reported that use a camera to automatically compute accurate transformations and blending between projection regions. Cubic displays have used small LCD screens for a compact design, which allows them to be hand-held, but has the downside of relatively thick seams between screens [50]. Seams themselves can enhance the 3D effect by providing occlusion cues, but thick seams can be obtrusive and potentially disruptive to viewing. Multiple display panels also require screen calibration, but the accuracy requirements are lower, as the screens do not overlap like projected screens. The few quantitative analyses of error sources in FTVR displays that have been reported suggest that screen calibration error contributes significantly less to overall visual error than head-tracker calibration error [15, 63]. However, little previous work has addressed or quantitatively measured head-tracker error for FTVR displays.

2.6 Calibrating Viewpoint Location

A motion tracking system is required for FTVR displays to measure the user's head position relative to the display. The tracking system must be calibrated to the display, i.e. the rigid transformation between the reference frames of the tracking system and display must be found. This is often done by tedious manual tuning, but standardized procedures have been proposed where the transformation is estimated from a corresponding set of tracker-measured and physically-measured 3D points. For example, Kindratenko et al. [31] used an EM tracking system to record the tracked position at several locations in the CAVE. The locations were precisely determined using a sensor mount consisting of a 1' x 1' x 0.1'

wooden board with housing for a 2', 4', 6', or 8' pipe. By moving the board between marks on the floor and changing the pipe length, the sensor could be placed at a number of known locations with a precision of 0.5 cm. Other systems have used an optical system for calibration [42, 55]. EM tracking systems are often calibrated in this way because EM distortion can skew position-tracking measurements across the workspace [12, 22].

Although a head-tracker can be accurately calibrated, the actual viewpoint of the user is often offset from the tracker's measured head position. If a head-marker is worn, there will be an offset; however, even markerless trackers, such as the Kinect, result in a reported head position that is not aligned with the user's viewpoint. Most FTVR displays assume a constant offset from head-marker to the users viewpoint [50]. Madritsch et al. [36] describe a model for locating the position of a user's eyes when viewing an FTVR display. Glasses with trackable markers are worn by the user and a constant offset is applied in the user's reference frame to locate the eyes.

Czernuszenko et al. [16] developed a method for accurate tracker calibration of a projector-based VR system which involves the user aligning physical and virtual markers at several different head positions. Physical markers are placed at known locations and virtual markers are rendered to appear at the same location. The viewer wears a pair of tracked glasses and uses a controller to align the virtual markers with the physical markers and record the offset vector at the current head position. This is repeated at any point with considerable misalignment, and an appropriate lookup table is built.

2.6.1 Point to Point Correspondence

A common component of calibration is finding the transformation between a tracking hardware's coordinate frame and a virtual coordinate frame. When solving for a transformation relating two coordinate frames, point to point correspondence is commonly used. If the 3D location of points are known in both frames, a matrix can be solved for that converts positions in one frame to the other. The unknown transformation can be solved for through

a least squares approximation. Since the two frames each have an orthonormal basis, the transformation should only consist of a: translation, rotation, and possibly scale component. This type of transformation is referred to as a rigid transform. Ideally, the transformation found through direct equation manipulation should be a rigid transform, but there will most likely be error in any measured points. The error in the measurements could cause the solved transformation to no longer be rigid, which is known to be incorrect. To ensure the solved transformation is rigid, additional constraints are established [6]. One downside to this method is that it assumes point to point correspondence.

An iterative approach can be used if a set of point pairs is not present. The iterative closest point algorithm [9] can be used to fit a set of 3D data to a model. The model could be a set of data (points, lines), or the equation of a line or surface. This method of finding a transformation is beneficial if the number of measured points does not match the number of points in the model, or if the data is being fit to a geometric shape.

Chapter 3

Display Rendering

Rendering is the process of converting 3D data into 2D data which can be drawn on a screen. Virtual object information is transformed through multiple different coordinate systems to determine the color of each of the screens' pixels. Virtual objects are created as a list of vertex positions, which are used to define the shape of the objects. The location of a vertex is represented by an x, y, and z coordinate expressed as a three element vector. When transforming a position, the data is usually expressed in homogenious coordinates as a four element vector to allow for a more computationally efficient process. The transformations in computer graphics are commonly made up of a translation, a rotation, and a scaling factor. Rotation and scaling could be accomplished by multiplying a three element vector by a three-by-three matrix, but translation would require an additional operation. A translation could be accomplished through matrix multiplication by using a four element vector and a four-by-four matrix. If we can express every part of a transformation as a four-by-four matrix, they can be multiplied together to form a single matrix for the transformation. For this reason, both rotation and scale are expressed as four-by-four matrices as well. If a position must go through multiple transformations, the matrices can be multiplied together to form a single matrix for the entire process.

Other information about the vertex can be specified alongside its position, such as: color, normal direction, texture coordinate, or any other useful information. A matrix is paired with the object which controls the location, orientation, and size of the object; this matrix is commonly referred to as the model matrix ($\mathbf{X}_{model}$). The model matrix transforms vertex information from model-space to world-space. Model-space is the coordinate frame where

the vertex information is originally defined. The origin of model-space is generally in the middle of the vertices with the y-axis pointing upwards. World-space is where the virtual scene is constructed, and where objects can be positioned, rotated, and scaled relative to any number of other objects. When multiple objects are present, it is important that each object is appropriately sized in relation to the other objects in the scene. The creator must establish consistent units that all object information is expressed in; these units are commonly metric units. A virtual camera is used to capture the objects and position them on the screen. In computer graphics, a camera is defined by two matrices: its view matrix ($\mathbf{X}_{view}$), and its projection matrix ($\mathbf{X}_{projection}$). The view matrix describes the location and orientation of the camera in world-space, and is used to transform points in world-space to camera-space. Camera-space is a coordinate frame that is aligned with the camera and has an origin located at the camera. As the camera moves and rotates, the view matrix will change; this will make it appear as though the objects are moving. The projection matrix converts points from camera-space to clip-space, which is the final coordinate frame. Clip-space is a coordinate frame that describes where the objects will appear to exist in the camera's viewing frustum. Different types of cameras have different shaped viewing frustums which results in different transformations to clip-space. The most commonly used camera model is the perspective camera, which functions the same as a human eye. The size of the frustum uniformly increases as it moves away from the camera. Objects farther away will be made smaller to match the expected perspective visual cue. Another camera model that is less commonly used is the orthographic camera. The size of the orthographic camera's frustum does not change as you move away from the camera. This means objects will appear the same size regardless of how far away they are. This type of camera is useful in design scenarios if precise measurements are needed, but since they do not function the same as a human eye they are used less often. Once the objects are in clip-space, filtering algorithms are used to determine which of the objects will be drawn on the screen.

The entire transformation process can be expressed in a single equation (3.1). The x, y, and z variables form a vector that represents where the vertex is in model-space, and the u and v variables represent where the vertex will appear on the screen. The left side of the equation

could have a scaling factor (w) applied to it. The scale factor will be the value of the third element of the vector.

$$w \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \mathbf{X}_{projection}\ \mathbf{X}_{view}\ \mathbf{X}_{model} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{3.1}$$

3.1 Multi-Screen FTVR Displays

Rendering for a FTVR display is very similar to rendering for a standard screen. The only difference is that the virtual camera must move to match how the user moves their head. By moving the virtual camera the same way the user moves their head, it can capture the scene from the correct perspective. Head-tracking hardware is used to measure the location of the user's head relative to the origin of the tracking system. The tracking system's coordinate frame will be referred to as tracker-space. The rendered objects are positioned inside of the display in a coordinate frame referred to as display-space. Display-space is a coordinate frame that is fixed relative to the display, and is often fixed relative to the tracking system as well. A transformation must be found that relates a measured head position to display-space in order to position the virtual camera properly. The transformation from tracker-space to display-space can be determined if the location and orientation of the display is known relative to the tracker.

The location of the user's head can be tracked in a variety of different ways, but there is almost always some sort of sensor or marker worn on the user's head. If the virtual camera is placed where the head is tracked to be, the scene will be rendered from the wrong location. In order to render from the correct location, the virtual camera must be placed at the viewpoint of the user. The viewpoint will be located at one of the user's eyes if viewed monocularly, and is assumed to be at some point between the eyes if viewed binocularly. The precise location

of the viewpoint in the binocular case may be hard to determine as different individuals have different eye dominance. This means that the location of the viewpoint relative to the head marker is likely to be different for each individual user. To position the virtual camera in the correct spot, two unknowns must be solved for: the transformation between tracker-space and display-space, and the location of the viewpoint relative to the head marker.

In order to render to the screen properly, the location of each pixel in display-space must also be known. The reason for this is because we want the screen to function like a window into a virtual environment. To see what color of light would be seen at a specific point on the window, we would need to see what object is behind it. The precise location of the pixel must be known to determine which object is behind it. If the display is made up of multiple screens, we can treat it as having multiple windows and the same process is repeated for each one. The process that is used to capture images of the scene is dependent on the shape of the display being rendered to. There are two types of displays that we considered: displays with planar screens, and spherical displays.

3.1.1 Planar multi-screen Display Rendering

Planar displays, such as a cubic display, can be rendered properly by creating a virtual camera for each screen. Each virtual camera is positioned at the viewpoint, facing in the opposite direction of the respective screens normal vector. Each virtual camera uses an off-axis projection matrix to make the frustum fill up the screen. The near plane of the virtual camera is generally set to be on the surface of the screen, but it can be closer to the viewpoint to render objects in front of the screen. This type of rendering will result in a distorted image being drawn to the screen that will appear undistorted at the viewpoint; this effect is shown in Figure 3.1. To be able to orient the virtual cameras properly, the location, orientation, and size of each screen must be known in display-space. A screen calibration procedure must be used to find these unknowns. The calibration process we used is described in chapter 4.

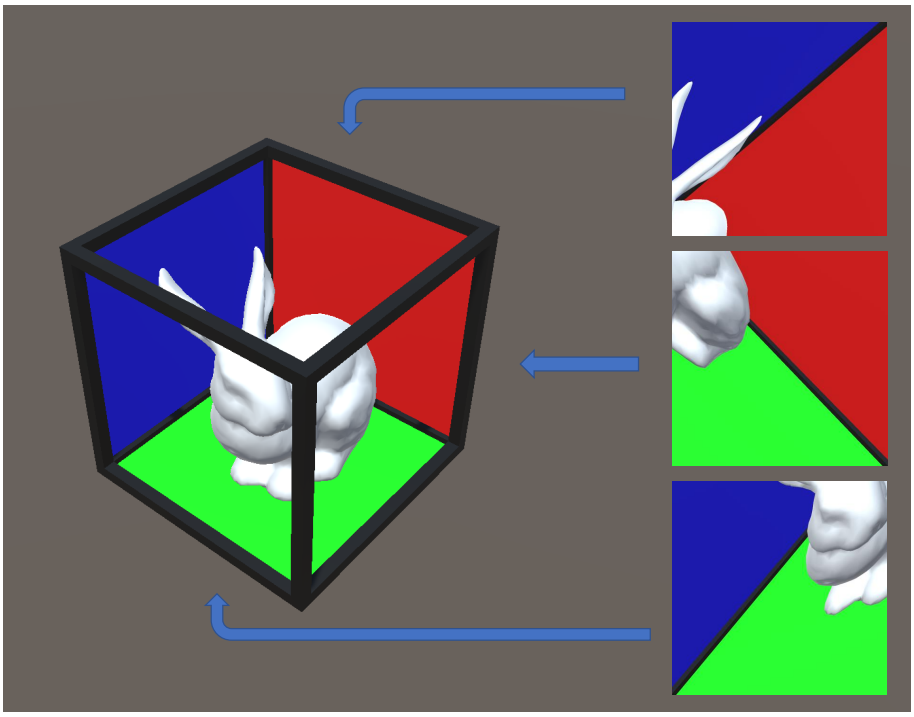


Figure 3.1: Cubic FTVR display rendering using off-axis perspective matrices. Images on the right represent the rendered content if viewed head on

3.1.2 Spherical Display Rendering

Spherical displays generally consist of projectors illuminating the inside of a spherical surface. A spherical display is more challenging to render to because you must determine where each pixel is located on the surface of the sphere. Rendering to the display requires a mapping for each pixel in each projector to their location on the sphere [62]. A shader is utilized that transforms each pixel location to determine where it is located in the frustum of a virtual camera, which is located at the viewpoint. Once this location is known, a render texture can be sampled in order to determine the color of each fragment.

3.2 Stereoscopic Rendering

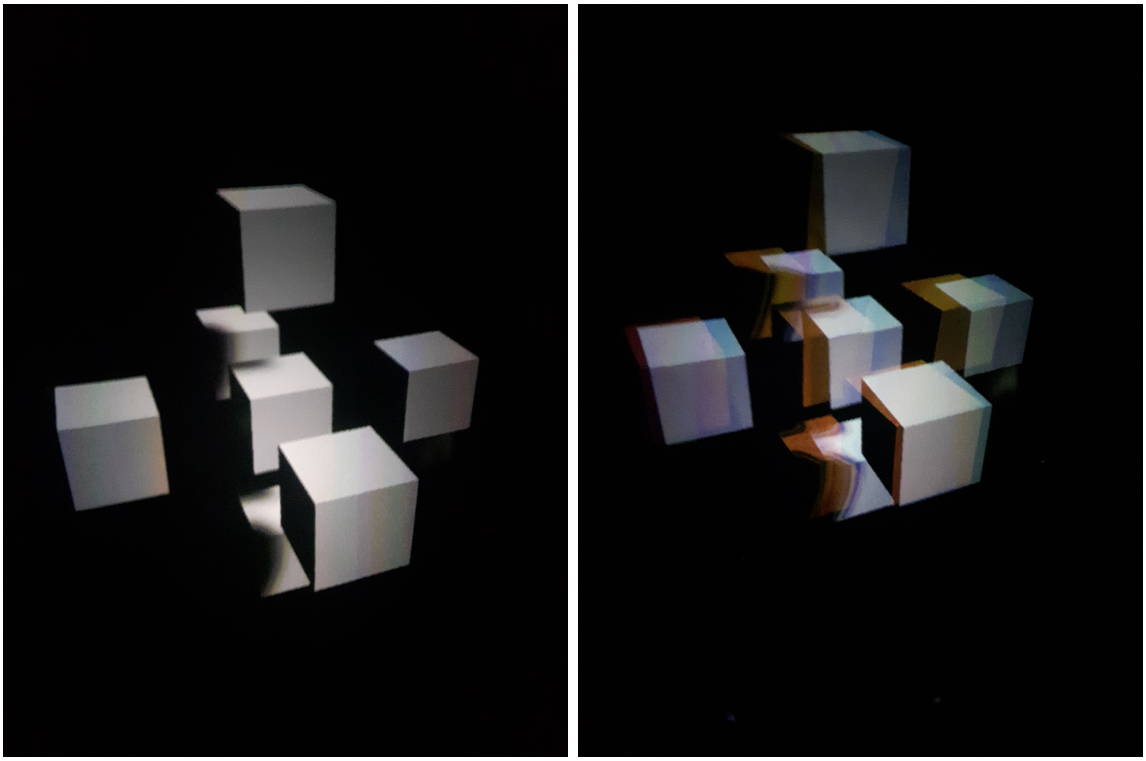
Standard rendering has the ability to capture a scene in the same way a camera can capture a picture. Capturing a picture preserves a majority of the visual cues that we rely on for depth

information, such as the relative size, shading, and occlusion. One cue that is not preserved is the stereo visual cue as the scene is only captured from a single viewpoint. The result is that the picture will look less 3D since both eyes are viewing the same perspective of the content. The same is true for a rendered scene. Without the stereo cue, a lot of the depth information is lost. The stereo cue can be introduced into the content by rendering the scene from two different viewpoints by using two different virtual cameras. Each of the two virtual cameras will capture content for each of the eyes. The virtual cameras are rigidly attached to each other, separated by the same distance between our eyes. Once the scene is captured from the two perspectives, the display must be able to present the correct perspective to each eye. Active shutter technology is commonly used for personal stereoscopic rendering as it is the cheapest option that does not negatively impact the image.

The active shutter system works by having the display alternate between the two perspectives on each frame. This system requires the display to be able to refresh twice as fast as the desired viewing frame rate. Generally, 120 Hz is used because it allows each eye to update at 60 Hz, resulting in minimal disruption. Glasses are worn by the user to filter away the incorrect perspective for each eye. If the glasses are not worn, the user would see the sum of the two images. The glasses have the ability to block the vision of an eye by having lenses made of a material that becomes opaque when voltage is applied across it. This way, the battery powered glasses can block the vision of the left eye when the right eye's content is shown, and similarly block the vision of the right eye when the left eye's content is shown. In order for the glasses to know which eye's content is being shown at a given time, the computer must communicate with the glasses, which is commonly done with IR or RF signals. A square wave signal is produced from the graphics card that changes state when the displayed content is changing from one eye to the other.

Modern displays apply different filters to the content to remove noise and enhance the quality of the image. This process takes time to do and will result in a delay between when the graphics card's signal was sent and when the display changes eyes. Different types and models of displays perform different amounts of filtering on the images which results in a variable amount of delay present. For this reason there is commonly a degree of ghosting

present when viewing stereoscopic content, as shown in Figure 3.2. Ghosting occurs when each eye can see a small amount of what the other can see, which will make it look like there are duplicates of an object. Ghosting can be reduced by adjusting the displays settings, but additional hardware is usually required to completely remove it. The ghosting can be removed by introducing a delay in the square wave signal that comes from the graphics card. By changing when the glasses change state, we can sync them up with the display. A micro-controller can be used to introduce the delay, and the signal from the graphics card can be used as a trigger for the controller. Once triggered, the controller will sleep for a preset amount of time before it propagates the input state to the glasses emitter. The amount of time that it must sleep varies per setup, and must be measured beforehand using an oscilloscope and a photo-diode.

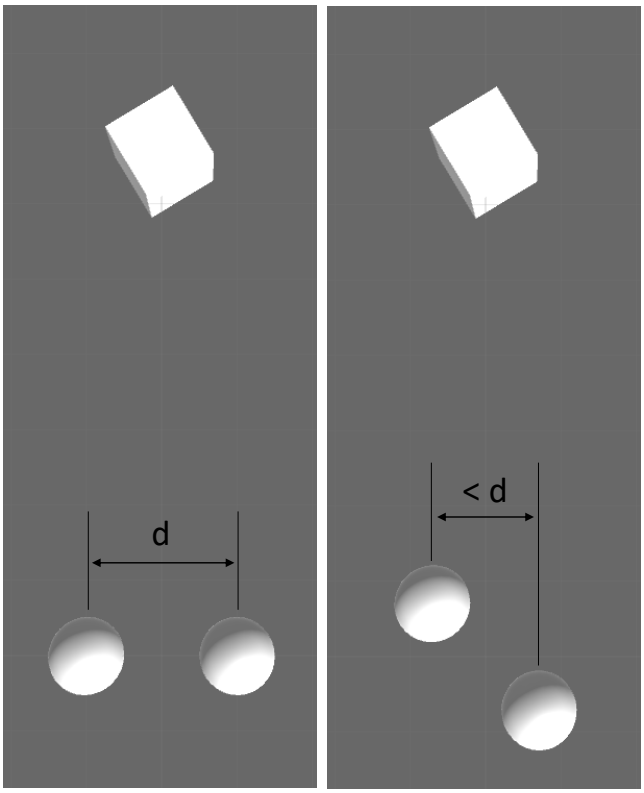


(a) No Ghosting **(b)** Ghosting

Figure 3.2: Ghosting artifact caused by poorly synced shutter glasses

The 3D effect of the multi-screen FTVR displays can be enhanced by incorporating stereoscopic rendering into it. The display must be constructed out of screens that refresh at 120 Hz, which could be high-end computer monitors or projectors. Many modern projectors have

the ability to refresh at 120 Hz, which makes it easy to incorporate stereoscopic rendering into a spherical head-tracked display. Capturing the correct perspective of the scene for each eye is more challenging because it depends on which direction the user is facing. As the user rotates their head, their eyes will become more aligned with the display, resulting in a smaller separation of rendered objects (Figure 3.3). To position the cameras properly, the head-tracker must be able to track the position and orientation of the user's head. If only the position can be found, the cameras could be placed by assuming the user is looking at the center of the display. This is a good assumption because the user will be looking around the center most of the time, but if they rotate their head, they may view small amounts of distortion. This assumption will be violated if the user tilts their head to either side. Proper stereoscopic rendering always separates the images along the head's horizontal axis. If the separation direction does not match the head's horizontal axis, the 3D illusion will break.



(a) No head rotation **(b)** Head rotation

Figure 3.3: Separation due to head rotation; as the head rotates, the eyes become more aligned

3.3 Multi-Viewer Rendering

Rendering on a head-tracked display will result in images being drawn that appear distorted everywhere except for at the user's viewpoint. If two people were to look at the same rendered content, it would only appear correct for one of them. For this reason, content rendered on a head-tracked display can only be properly viewed by a single person. Certain head-tracked displays can be made to work for more than one person if they are able to show different content to each individual. If the display is capable of doing stereoscopic rendering, two people can observe different content simultaneously. Rather than swapping between the left and right eye's content, the display could swap between the content for the two individuals. The glasses can then allow both eyes to see when the correct content is shown, and neither eye to see when the content intended for the other person is shown.

If the display is not able to do stereoscopic rendering, multiple people can observe perspective corrected content if the rendered areas do not overlap. If two people stand on opposite sides of the display, the content for each of them could be drawn simultaneously. Each person would only be able to see their own content because the display would occlude the other content. This method of multi-viewer rendering could be combined with the stereoscopic method to allow four people to view content. The four people would stand equally spaced around the display, each wearing shutter glasses. Each pair of glasses would be in sync with the glasses on the opposite side of the display. Even thought the rendered content would most likely overlap with the people beside an individual, the glasses would be able to block their content.

If the rendered areas overlap and stereoscopic rendering is not available, the content could be rendered to provide minimal distortion to each viewer. Nacenta et al. [41] studied perspective corrected rendering of a tabletop display for multiple users. They found that rendering for a specific person will likely cause problems for the others in the group, but that presenting a top down view and using an orthographic camera frustum can reduce the problem. This concept could be used for multi-screen head-tracked displays by averaging the position of each viewer and rendering from there. Although every viewer will likely observe distorted

content, the distortions will be minimal if the group of people are in close proximity.

3.4 Summary

Rendering on a multi-screen FTVR display follows a similar process to conventional rendering. The main difference is that the display must track the user. Planar displays function by utilizing multiple virtual cameras each with off-axis projection matrices that intersect their appropriate screens. Spherical displays capture the scene with a single virtual camera. The virtual camera's image is then sampled wherever each pixel intersects the sphere. Stereoscopic rendering can be introduced if the display hardware can support it. The scene is captured from the perspective of each eye, and specialized glasses are used to filter the appropriate content. The next chapter will go over the different types of multi-screen FTVR displays that were constructed, as well as the calibration process that was done to determine the location of each pixel in display-space.

Chapter 4

Multi-Screen FTVR Displays

This chapter describes the different multi-screen FTVR displays that were constructed, as well as how to calibrate the screens together.

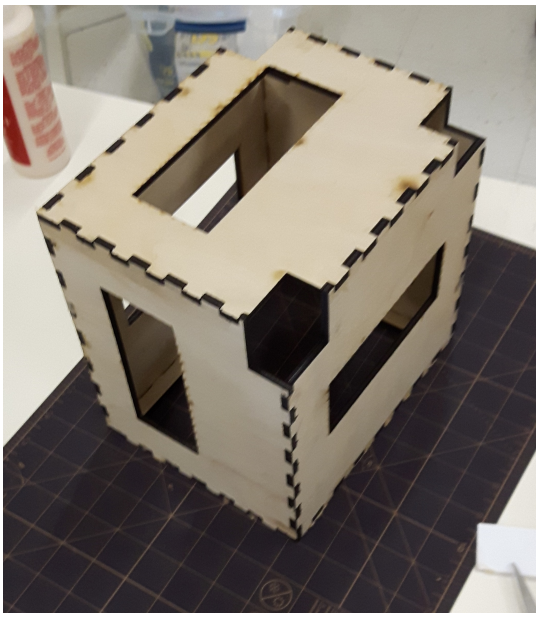
4.1 Display Assemblies

Two different shapes of displays (cubic and spherical) were constructed to do perspective corrected rendering.

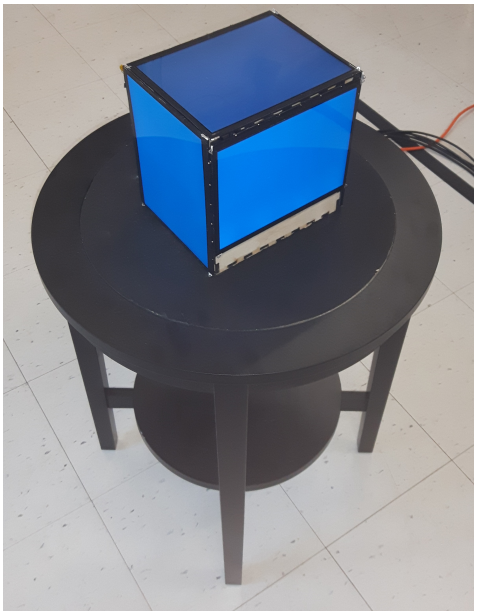
4.1.1 Cubic Display

The cubic display we built is made up of five screens in the shape of a box. The screens we used were 9.7" LP097QX1 LCD screens from LG, and the drivers were adafruit retina Ipad display-port drivers. We chose these screens for their high resolution (2048x1536) and the small screen bezel. Five 25 ft display port cables are used to connect the display drivers to the computer. The long cables were chosen as they allow the display to be positioned far away from the computer. The computer running the rendering application must have at least six display outputs: five for the display, and one for a desktop monitor. The computer we used has two graphics cards: a Quadro M4000 and a Quadro K2200, both from Nvidia.

A frame is required to support the screens. The frame could be any assembly that can keep



(a) Underlying cubic display frame



(b) Cubic display table

Figure 4.1: Cubic display assembly

the screens stationary. We chose to laser cut a box, shown in Figure 4.1a, out of plywood as it is cheap, fast, and can produce reliably orthogonal surfaces. The screens are mounted on the outside of the frame using Velcro tape, and the display drivers are mounted internally. The holes in the corners of the frame are to allow cables to transfer from the screen to the driver, and the holes in the center of each plane are to assist with heat dissipation. A custom table was made to support the display. The table was a standard coffee table with a square hole cut in the top to allow for the display and power cables to go through, and is shown in Figure 4.1b.

The dimensions of the constructed display are 7”x 8.5”x 8.5”. The display is not a perfect cube due to the screens having a larger width than height. Because of this, there are regions on the front and back that are not covered by screen.

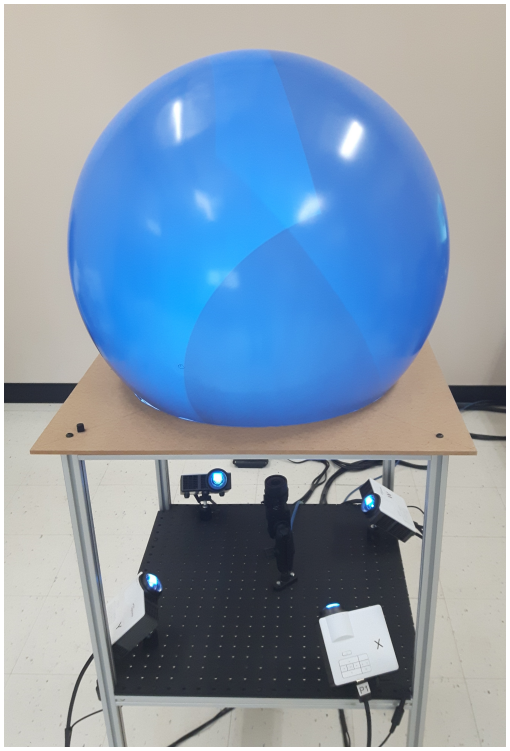


Figure 4.2: Spherical display assembly

4.1.2 Spherical Display

A sphere display was constructed out of a 60 cm diameter plastic sphere, provided by B-con Engineering [2], and four projectors. The plastic sphere has a 45cm hole in the bottom to allow for light to be projected on the inside of the sphere. We use four Optoma GT750ST projectors to project on the inside of the sphere. These short throw projectors are compact and have a low throw ratio, resulting in a large amount of surface coverage. Four 25 foot HDMI cables are used to connect the projectors to the computer. The computer we used has two graphics cards: a Quadro P4000 and a Quadro K2000, both from Nvidia. A camera is used to calibrate the display; we used a Grasshopper 3 GS3-U3-32S4M camera from Point Grey. The frame of the display is made of 1 inch T-slotted aluminum bars, which allow for easy height adjustment of the camera and projectors. The top of the frame is a 2'x2' piece of 1/4" particle board with a hole cut in the center of it.

4.2 Screen Calibration

In order to render to head-tracked displays, the 3D location of each pixel must be known relative to the display's coordinate frame (display-space). The calibration process that is done depends on the shape of the display. This chapter will go over two different shapes: cubic and spherical. These display shapes are the most common and the calibration process for both utilize a calibrated camera.

4.2.1 Camera Calibration

Cameras have the ability to capture a 2D image of a 3D environment at a point in time. The environment is made up of 3D objects, the location of which can be described by their Cartesian coordinate (x, y, and z). The location of objects in the image can be described by their image coordinates (u and v). There is a transformation that relates where an object is and where it will appear in the image, this is referred to as the camera matrix (C); the relationship is shown below:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \propto C \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{4.1}$$

The camera matrix can be expressed as the product of two matrices: an intrinsic matrix, and an extrinsic matrix. The intrinsic matrix describes how a point in the camera's reference frame will convert to its pixel location, and is given by:

$$A = \begin{bmatrix} \alpha & \gamma & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.2}$$

Where α and β represent the scale factors of the x and y directions respectively. u_0 and v_0 are the location of the principal axis, and γ represents the skewness of the image axes.

The extrinsic matrix of a camera describes its physical rotation and translation with respect to a specified world coordinate frame, and is given by:

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \tag{4.3}$$

In equation (4.3), each r represents an element of the rotation matrix describing the rotation between the camera's coordinate frame and the world coordinate frame, and each t represents an element of the translation vector separating the camera from the world coordinate frames. For the calibration process, the specified world origin is a point on the calibration plane, shown in Figure 4.3. The x and y axes exist in the plane, and the z axis points normally outwards. Every captured point will exist in this plane, and therefore will have a z component equal to zero. As a result, the third column of the extrinsic matrix has no effect on the resulting pixel location, and it can be removed from the matrix.

$$R = \begin{bmatrix} r_{11} & r_{12} & t_1 \\ r_{21} & r_{22} & t_2 \\ r_{31} & r_{32} & t_3 \end{bmatrix} \tag{4.4}$$

With the column removed, the product of the intrinsic and extrinsic matrix is now a 3x3 matrix, which is referred to as the camera's planar homography matrix (H). Equation (4.1) can then be written as follows:

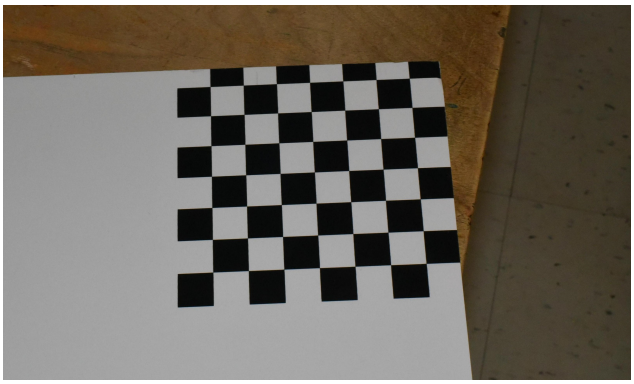


Figure 4.3: Calibration plane used in calibrating cameras; the corners of the squares are used as detected features

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \propto H \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{4.5}$$

Since the left-hand side of equation (4.5) is a vector that is proportional to the resulting vector on the right-hand side, the cross product of the two sides should equal the zero vector.

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \times \begin{bmatrix} H_{11}x + H_{12}y + H_{13} \\ H_{21}x + H_{22}y + H_{23} \\ H_{31}x + H_{32}y + H_{33} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \tag{4.6}$$

The cross product provides us with three equations, two of which are linearly independent. To be able to solve for the nine unknowns, we must consider at least four points in the image. This way we can solve for the matrix, up to an unknown scale factor. The solution to the matrix can then be solved by singular value decomposition, and the unknowns can be reassembled to form the homography matrix. This method of solving for the homography was obtained from Hartley et al.[23]. The homography can then be broken down into the product of two matrices: the intrinsic and extrinsic matrix. This product is shown below:

$$H = AR \tag{4.7}$$

The first two columns in the extrinsic matrix (4.3) describe the direction of the x and y axes of the camera's coordinate frame respectively. Since the coordinate frame is orthogonal, the two axes must be perpendicular to each other, and the two axes must have the same magnitude. Knowing this, we can create two equations from each homography matrix; three homography matrices are required to solve for all parameters of the intrinsic matrix. Once the intrinsic parameters have been solved for, the extrinsic parameters can be solved for by multiplying the inverse of the intrinsic matrix by the homography matrix. The details of the calculation can be found in Zhang et al. [61].

Projector Calibration

A projector differs from a camera in that it produces an image to be shown in the physical world as opposed to capturing one. Since the projector has the opposite functionality of a camera, it can be thought of as an inverse camera. The same matrices can be used to describe a projector that are used to describe a camera.

A calibrated camera is generally needed in order to calibrate a projector. A pattern is projected on a rigid board which has a chessboard pattern printed on it, shown in Figure 4.4. The projected pattern could be another chessboard, or a grid of circles to help distinguish the two patterns. The orientation of the board is found by detecting the chessboard pattern that is printed on the board. Once this is found, the location of the projected features on the board can be calculated. These points are paired with the pixel location in the projected image, and the same calibration process that is done for a camera is done to find the intrinsic and extrinsic parameters of the projector.

4.2.2 Planar Displays

In order to render to a planar display, the location and orientation of each screen must be known relative to each other. The screen location and orientation could be estimated by

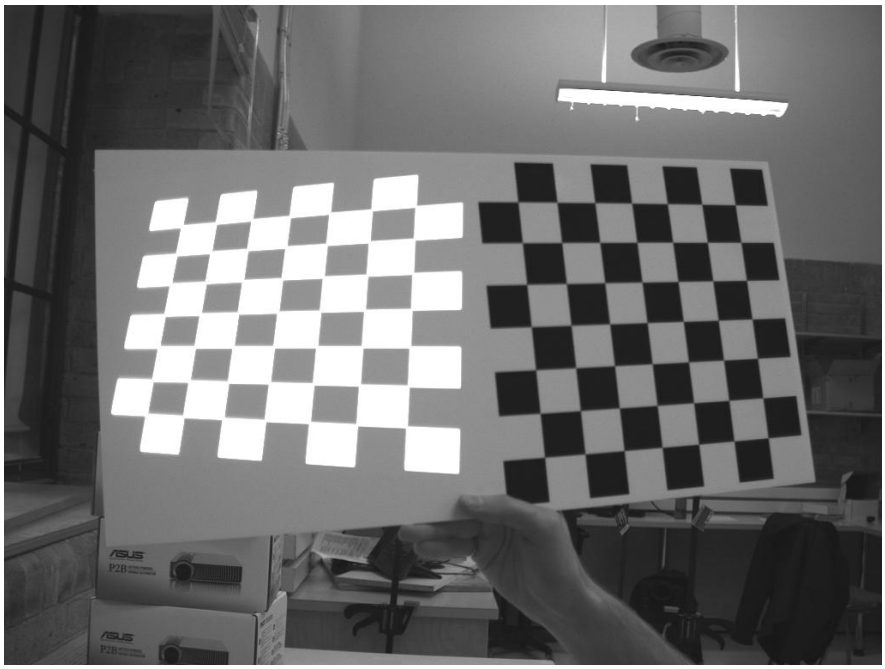


Figure 4.4: Projector calibration board; right pattern is printed on, left pattern is projected

manual measurements, but an accurate measurement can be hard to perform. A camera can be used to calibrate the screens in a way similar to calibrating a camera.

A chessboard pattern is rendered on each screen, and multiple pictures are taken of the display. A homography matrix is determined for each screen in each picture, and the camera's intrinsic matrix is calculated using every homography matrix. An extrinsic matrix is found relating each screen to the camera, and these matrices can be used to relate each screen to another screen. One screen is picked as the origin of the display, and the transformation from every other screen is determined relative to it. To maximize accuracy, the origin screen should be visible in all images as it minimizes the number of transformations to combine.

Cubic Display

A cubic display is made up of five planar screens in the relative shape of a cube. The display has a screen on each of the sides, excluding the bottom. The origin of a cubic display is chosen to be the top left corner of the top screen, with axes aligned with the top screen's

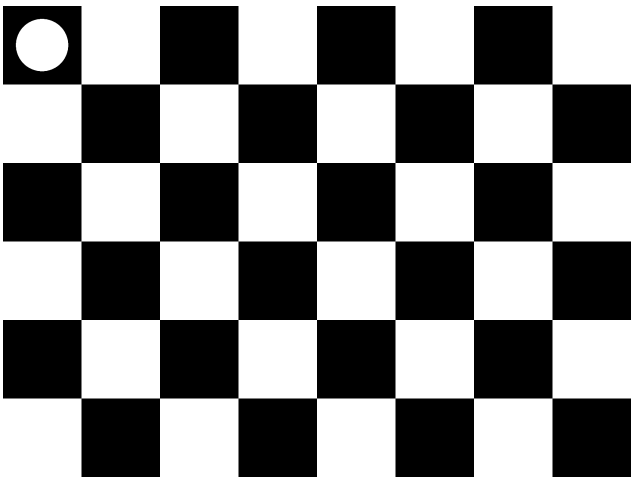
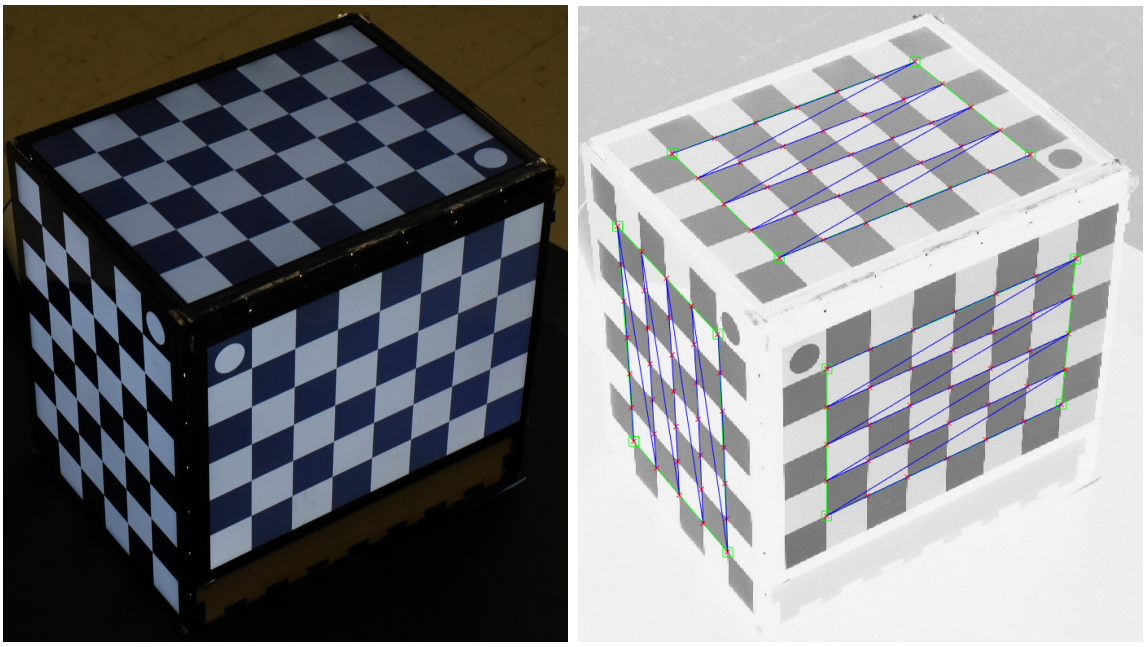


Figure 4.5: Chessboard pattern used for calibrating planar displays

coordinate frame. By doing this, we only need to find the relative position and orientation of every other screen with respect to the top screen. For a cubic display, we must find four transformations. A chessboard pattern is rendered on each screen for a convenient set of features to be detected. the pattern is shown in Figure 4.5. A circle is drawn in the top left corner to help locate the origin of the screen.

A program was created in MATLAB to solve for the unknown screen transformations. Pictures are taken of the cube from different viewing directions. In order to reduce the number of pictures taken, three screens are visible in each picture. A semi-automatic process is used to find the location of the corners on each screen. The user must click on the four most extreme corners of the pattern for each screen. The first corner is the one close to the top left corner of the screen, then top right, bottom right, and finally bottom left. The horizontal axis of the screen will be aligned with the vector separating the second point from the first point, and the vertical axis will be aligned with the vector separating the fourth point from the first point. To find the location of the other corners, a homography matrix is created that relates the corners of 1x1 square to the location of the extreme corners. The corners on the screen are equally spaced, and we can generate the same number of equally spaced points in the 1x1 square. each of these points are transformed using the homography matrix to determine where the corner should exist in the image. The precise location of the corner is found using corner finding algorithms obtained from the camera calibration toolbox for



(a) Captured Image of the display **(b)** User input on the processed image

Figure 4.6: Screen calibration process for cubic displays; red x's show detected corners, blue lines show order of corners, green lines show the bounding box

MATLAB [13]. To make the process easier for the user, lines are drawn on the image to show the location of the detected corners. If an incorrect corner location is found, the user can retry to detect the corners. The corner detection process is repeated for each screen in an image before moving to the next image. Figure 4.6 shows the corner detection process. The image that the user clicks on undergoes some image processing to make it easier for the user. The image is converted to gray-scale, sharpened, and negated to make it easier to see the corners and cursor. Even though the displayed image has been modified, all corner detection algorithms are done on the original image.

Once all corners are located, the camera that is used to capture the images is calibrated using MATLAB's estimateCameraParameters function. This function simultaneously calculates the cameras intrinsic matrix and determines the position and orientation of each screen relative to the camera. Transformation matrices can then be constructed for each screen in each image that will transform a point in the screen's coordinate frame to the camera's coordinate frame. The transformation matrices can be combined to find the transformation from each side screen to the top screen as follows:

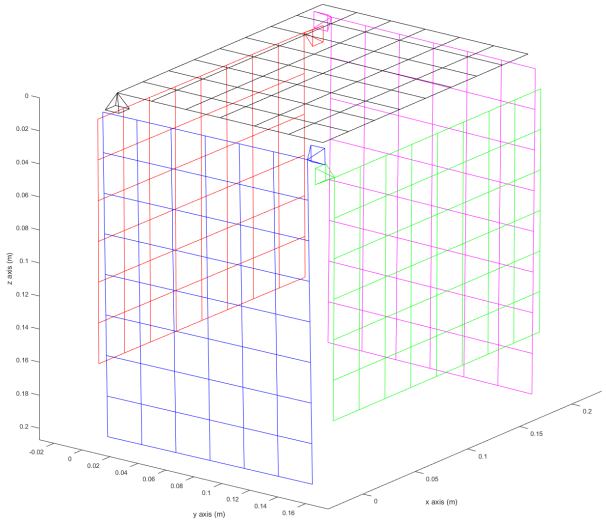


Figure 4.7: Screen calibration results; planes are plotted to indicate the calculated screen transforms

$$\mathbf{X} = \mathbf{X}_{Top}^{-1}\ \mathbf{X}_{Side} \tag{4.8}$$

Once each of the transformations are found, a k-nearest neighbors clustering algorithm is used to find similar matrices. With cubic displays, there are four side screens to find transformations for. The clustered matrices are averaged to determine the final result. The results are plotted to help visually verify the results, this is shown in Figure 4.7. The resulting transformations can be exported as a text file, which will be read in to the rendering application.

Evaluation

Evaluation of the results consisted of calculating the standard deviation of the clustered matrices. The matrices were separated into their rotation and translation components, and an average standard deviation was calculated for each. The standard deviation of the rotation

was calculated by finding the standard deviation for each element of the 3x3 rotation matrices, these values were then averaged together. The standard deviation of the translation was calculated by finding the average x, y, and z standard deviations, then averaging them together. The standard deviation of the rotation and translation were found to be 1.86×10^{-3} and 1.44×10^{-3}m respectively. These results show that the algorithm can find clustered transformations that are very similar, which indicates a reliable calibration.

To further evaluate the calibration, a pattern consisting of straight lines can be rendered on the display. As the lines transition from one screen to the other, the discontinuity can be visually analyzed. A good calibration would result in continuous lines, whereas a bad calibration would result in lines that appear disjoint. Our qualitative evaluation found that lines appeared continuous after the proposed screen calibration procedure.

4.2.3 Spherical Displays

We calibrated the sphere display using the same procedure outlined by Zhou [63]. A camera is first calibrated in the same manner described in Section 4.2.1 in order to determine the intrinsic parameters of the camera. This camera is then used to calibrate each of the projectors. Once the intrinsic parameters of the camera and all of the projectors are known, the display assembly can be constructed. The camera and projectors are mounted using ball head mounts. Our display setup can be seen in Figure 4.2. The projectors are oriented to obtain maximal surface coverage, while the camera is positioned directly below the sphere pointing straight up.

To determine the size and location of the sphere relative to the projectors, a stereo calibration procedure must be done. Each projector is calibrated with the camera as a stereo pair by projecting circles on to the surface of the sphere. The camera captures images of the projected circles, and a point pair is created for each circle. Triangulation is used to determine the 3D location of the point, and a spherical surface is fit to the points. This process is repeated for each projector to find the sphere equation that best fits all projected circles. At this

point, ray-casting is performed to determine the location where each projected pixel would intersect the sphere. The final step of the calibration is determining the alpha mask for each projector. Certain regions of the sphere will have projectors overlapping, this will result in areas that are brighter than others. An alpha mask will tell each pixel how much to dim their light in order to produce an image of uniform intensity. The alpha mask is calculated based on how many projectors are overlapped at a point, resulting in a lower intensity for each overlapping pixel.

4.3 Summary

This chapter presented the two different types of multi-screen FTVR displays that were constructed. The cubic display was built out of five tablet screens, where the spherical display was built out of four projectors and a large plastic sphere. A calibration process is required to determine the 3D location of each pixel of the display. For planar displays, a camera can be used to take pictures of the screens. Chessboard patterns are rendered on the screens and the pixel location of the corners can be used to solve for the orientation and location of each screen relative to each other. For spherical displays, a camera can be used to take pictures of the spherical surface as the projectors project circles on it. The location of the circles can be used to determine where on the sphere each pixel exists. The next chapter will introduce a new method of viewpoint calibration, which is necessary to properly position the viewpoint around the display.

Chapter 5

Viewpoint Calibration

In order to render perspective corrected content, a virtual camera must be placed in the proper location in the virtual scene. The virtual camera must be placed in the same location as the user's viewpoint, therefore the location of the user's viewpoint must be known. This location is determined by tracking the position of the user's head, generally by having the user wear a sensor or marker on their head. In order to determine the location of the viewpoint relative to the display, it must be transformed into a display centered reference frame. If the location of the viewpoint is not accurately known, the content will appear distorted; this effect is shown in Figure 5.1. Properly positioning the viewpoint relative to the display is difficult because we do not know the location or orientation of the display relative to the tracking system. The position and orientation of the display could be approximated by manual measurements (measuring tape), but an accurate approximation is unlikely.

5.1 Problem

A calibration procedure is required to determine how to properly position the user's viewpoint. Multiple coordinate systems must be considered in order to understand the viewpoint calibration process; these are shown in Figure 5.2. Display-space (**D**) is the display centered coordinate frame, and viewpoint-space (**V**) represents the coordinate frame of the user's viewpoint. Tracker-space (**T**) is the coordinate frame where measurements of head position are made. Head-space (**H**) is the coordinate frame centered on the head marker worn by

(a) Good viewpoint calibration **(b)** Bad viewpoint calibration

Figure 5.1: Fishbowl rendering with different quality of viewpoint calibration

the user. The 3D content can be rendered properly if we know the position of the user's viewpoint relative to the display ($\mathbf{d}_{VD}$). There is no way to directly measure this position; it must be calculated through a series of other transformations. The viewpoint location can be found by transforming the head to viewpoint offset ($\mathbf{d}_{VH}$) from head-space to display-space; this process is as follows:

$$\mathbf{d}_{VD} = \mathbf{X}_{TD}\ \mathbf{X}_{HT}\mathbf{d}_{VH} \tag{5.1}$$

$\mathbf{X}_{TD}$ is the transformation that converts a point defined relative to the tracking system to where it is relative to the display. $\mathbf{X}_{HT}$ describes the position and orientation of the head marker relative to the tracking system. $\mathbf{d}_{VH}$ is the vector separating the location of the viewpoint and the location that the head is tracked at. Since $\mathbf{X}_{HT}$ is given from the tracking system, this leaves $\mathbf{d}_{VH}$ and $\mathbf{X}_{TD}$ needing to be found.

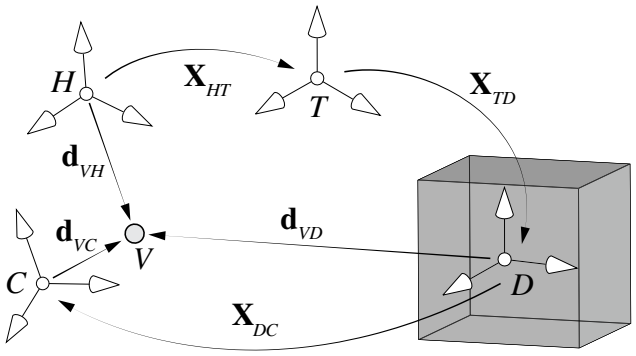


Figure 5.2: Relevant coordinate frames and transformations for viewpoint calibration

5.2 Existing Methods

Current methods of viewpoint calibration exist, but they generally enforce strict constraints on the system and the user. If the user is wearing a marker or sensor attached to a head mounted accessory (glasses, headphones), $\mathbf{d}_{VH}$ can be measured beforehand. This method is effective assuming all users will wear the accessory the same way, and they all have the same viewpoint location.

$\mathbf{X}_{TD}$ can be found using a point to point correspondence method. If the tracked marker can be placed in a specific location, defined in display-space, a singular value decomposition can be used to solve for the rigid transformation that best relates to the two sets of points [6]. This method is effective assuming the tracked object can be placed a specific location in display-space. If the geometry of the display surface is precisely known, the tracked object can be placed on the surface of the display where an indicator is rendered.

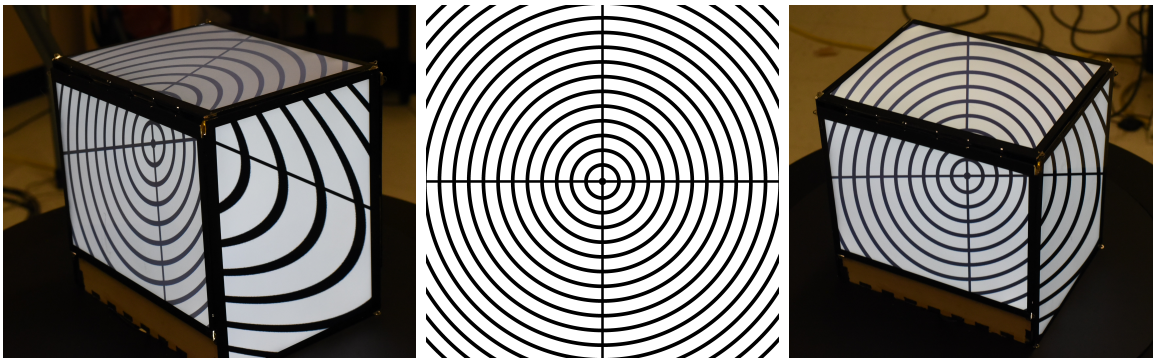
If the tracked object cannot be placed at a precise location in display space, or if $\mathbf{d}_{VH}$ is not consistent between users, then a manual tuning based approach can be used. An initial guess can be made at $\mathbf{X}_{TD}$ and $\mathbf{d}_{VH}$, and the transformations can be tuned in real time until they are correct. To determine if the transformation is correct, a scene is rendered inside of the display. If the scene appears distorted, the transformation is incorrect; once it appears undistorted when viewed from any location, the transformation is correct.

5.3 Pattern Based Approach

Viewpoint calibration requires the user to position their head at specific locations around the display. Visual feedback must be present to inform the user when they are at the correct position, and how to move when they are not. Physical markers can provide a very concrete indication of whether or not the user is in the correct position. The main downside with this is that it requires precise positioning of the physical markers. A predefined calibration rig could be used to ensure the physical markers are in the correct spot, but the user would need to keep the rig close by in case either the display or tracker is moved.

We propose to use rendered patterns instead of aligning physical and virtual markers. Perspective corrected patterns can be rendered without using head-tracker information; a location is picked in display-space, and a pattern is rendered for that location. As a result, the pattern will appear undistorted when viewed from that location. Calibration-space ($\mathbf{C}$) is the coordinate frame located where the rendered pattern will appear undistorted. Therefore, the pattern will look correct when the user's viewpoint is at the origin of calibration-space; in other words, when $\mathbf{d}_{VC}$ is zero. Multiple head-tracker measurements are needed to perform the calibration. Each measurement is paired with a unique calibration-space coordinate frame.

With any calibration process that involves physical measurements, there will be a degree of error in the measurement. The tracking system will not be able to perfectly measure the location of the head marker. Error is also introduced on the user side. The user should be able to position their viewpoint close to the intended viewing position, but they are unlikely to be exactly in the correct position. Since we have noisy measurements to relate to exact positions, an exact solution cannot be found. The transformation is chosen to be the one that minimizes the error between the two sets of points.



(a) Incorrect viewing location **(b)** Rendered pattern **(c)** Correct viewing location

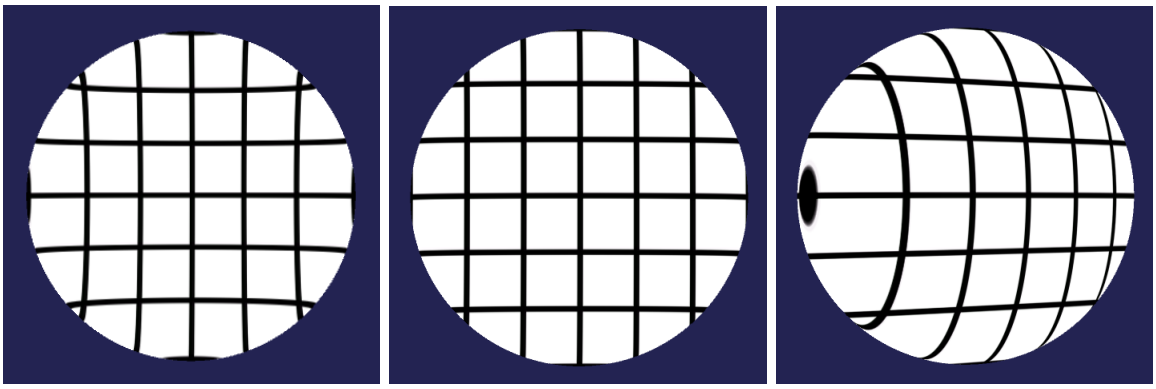
Figure 5.3: Rendered calibration pattern when viewed from the incorrect and correct location

5.3.1 Head to Viewpoint Correction

The transformation $\mathbf{X}_{TD}$ will transform the location of the tracked head marker from tracker-space to display-space. If the rendering is done with a camera placed where the head marker is, the content will look correct from that location. However, the content will appear distorted to the user because the display is not being viewed from the location of the head marker. The camera must be placed at the same location as the user's viewpoint, which is assumed to be somewhere between their eyes. The offset between the head marker and the viewpoint is required to position the camera correctly. The offset is assumed to be constant for each individual person, but it will most likely be different for each person. Each user is likely to have different eye dominance and will probably wear the head marker differently.

5.3.2 Point to Ray Correspondence

Finding a transformation that relates two sets of points has been studied in depth [6, 54], but a problem arises when doing the pattern based viewpoint calibration. When visually aligning the pattern, it is difficult for the user to determine if they are the correct distance away from the display. The reason for this is that there is less distortion present when viewed from the wrong distance, compared to being viewed from the wrong direction. This effect



(a) Wrong distance **(b)** No error **(c)** Wrong direction

Figure 5.4: Visual errors when performing the pattern based calibration on a spherical display

is shown in Figure 5.4. These images show what distortion would occur with a grid pattern rendered on a spherical display. The left image shows what would be seen if the display was viewed from too far away, and the right image shows what would be seen if viewed from the wrong direction. The center image shows how the pattern is intended to look, and is used to compare to the other two. When positioned the incorrect distance away from the display, the user would see either pincushion or barrel distortion in the image. Pincushion distortion would be seen if the user was too close to the display, where barrel distortion would be if they were to far away, as seen in Figure 5.4a. When viewing from the incorrect direction, the distortion becomes a lot more noticeable. The straight lines in the pattern appear to bend, and they give a much clearer indication of where the display should be viewed from. Even though there is more distortion in the image viewed from the wrong direction, both images were taken from the same distance away from the correct viewing position. Since it is harder for the user to tell if they are the correct distance away, the calibration algorithm must not rely on them being the correct distance away.

Rather than utilizing the standard point to point correspondence, an algorithm was developed that relates 3D positions to 3D rays. The user can then be any distance away from the display. The success of the calibration is only dependent on if they are viewing from the correct direction. An illustration of this is shown in Figure 5.5. The blue lines show the intended viewing directions and the green spheres show where a user could have placed their

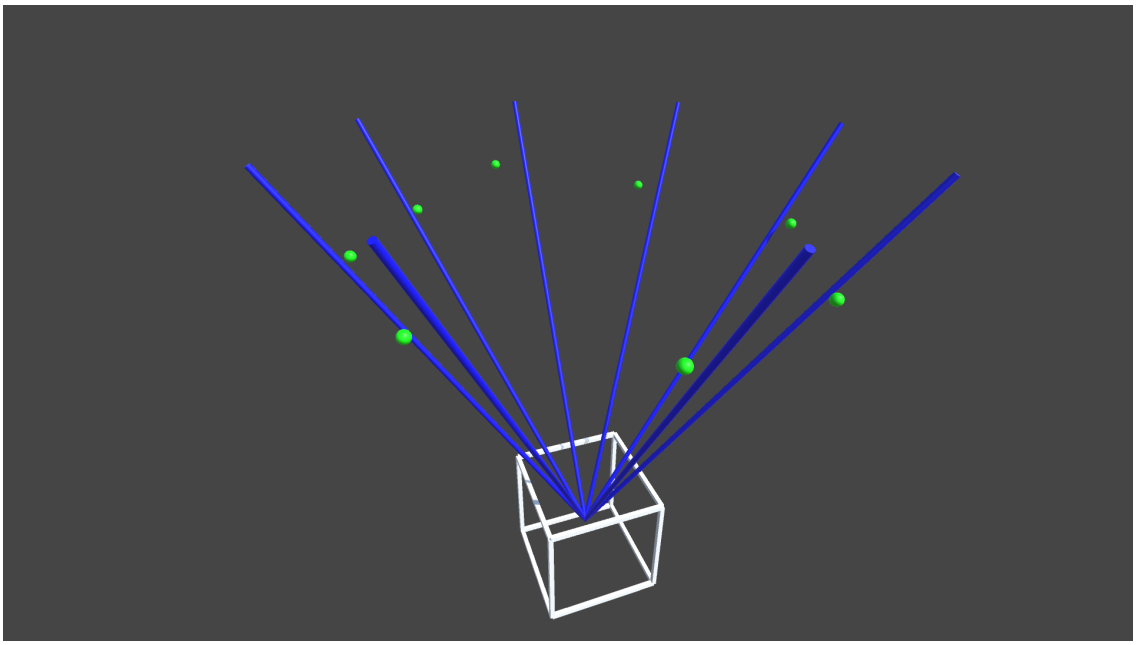


Figure 5.5: Viewpoint (green) alignment with intended viewing direction (blue) around cubic display (white)

head. The diagram shows eight viewing directions equally spaced around the display. In practice we found that having 24 measurements resulted in consistent calibration results. The rays were concentrated around the corners of the display because that is where three screens will be visible. Having more screens visible has been found to improve the user's ability to position their viewpoint.

5.3.3 Algorithm

An optimization process is used to find the transformation and offset that best transforms the measured head positions to the intended viewing rays. $\mathbf{X}_{TD}$ can be solved for if the location and orientation of the display can be found in tracker-space. A least-squared algorithm can be used to find the location that best describes the display given an orientation and $\mathbf{d}_{VH}$. The details of the calculations can be found in Appendix B. The orientation of the display and $\mathbf{d}_{VH}$ are found using an optimization process.

The cost function of the optimization is dependent on the tracker system being used. Certain

tracking systems only have the ability to record the location of the head, these systems are referred to as 3 DoF trackers. Others can measure the orientation of the head as well, these are referred to as 6 DoF trackers. The additional information that comes with a 6 DoF tracker can be used to improve the results of the optimization.

If a 6 DoF tracker is being used, $\mathbf{d}_{VH}$ can be applied in the head's coordinate frame since we know both the location and orientation of the head marker. Since we know that $\mathbf{d}_{VC}$ should be zero when the pattern appears correct, we can use the magnitude of $\mathbf{d}_{VC}$ as the cost function. The head-to-viewpoint offset is first transformed to tracker-space, then to display-space, and finally to calibration-space. The cost function for the 6 DoF tracker is shown in equation 5.2. If a 3 DoF tracker is used, $\mathbf{X}_{HT}$ cannot be found completely as the orientation of the head is unknown. Therefore, we cannot apply the offset in the head's coordinate frame, and it must be applied in a different coordinate frame. If we assume the user is always looking at the display, we can apply the offset in viewpoint-space. The offset would then describe the location of the head marker in the viewpoints coordinate frame ($\mathbf{d}_{HV}$). The position of the head marker ($\mathbf{d}_{HT}$) can be transformed to display-space, then to calibration-space. The location of the head marker in calibration-space should be the same as $\mathbf{d}_{HV}$, and the magnitude of their difference can be used as the cost function, which is shown in equation 5.3. The head marker is assumed to be rigidly attached to the user's head; there should be a constant rotation and displacement separating them, and the offset should be constant in either coordinate frames.

$$argmin_{\mathbf{R}_{TD},\mathbf{d}_{VH}} \sum_{i=1}^{n} ||\mathbf{X}_{DC}^{i}\ \mathbf{X}_{TD}\ \mathbf{X}_{HT}^{i}\ \mathbf{d}_{VH}|| \tag{5.2}$$

$$argmin_{\mathbf{R}_{TD},\mathbf{d}_{HV}} \sum_{i=1}^{n} ||\mathbf{d}_{HV} - \mathbf{X}_{DC}^{i}\ \mathbf{X}_{DT}\ \mathbf{d}_{HT}^{i}|| \tag{5.3}$$

For both equations, the i super script dictates a specific measurement out of a total of n measurements. Due to the lack of depth information in the rendered pattern, the solver must be able to account for the possibility that the user may be the incorrect distance away from

the display. Both equations 5.2 and 5.3 measure error with respect to the calibration-spaces, whose z-axes intersect the center of the display. Therefore, depth errors can be removed by only considering the x and y components when calculating the magnitude in both cost functions.

5.3.4 Verification with Synthetic Data

To verify the effectiveness of the calibration procedure, we performed a synthetic study. The study was designed to test how invariant the algorithm is to depth errors. This study was performed using synthetic data, and checked how well the algorithm would converge on the correct solution given noisy data.

A synthetic experiment was conducted to test how invariant the algorithm is to depth errors. A MATLAB program was written that would simulate the calibration procedure. The program generates two sets of 3D points and solves for the transformations that best relates them. The first set of points (calibration points) represents the locations where the rendered pattern would appear undistorted. The second set of points (recorded points) represent measurements obtained from the head tracker. The calibration points are randomly generated, as well as ground truth $\mathbf{X}_{TD}$ and $\mathbf{d}_{HV}$. The recorded points were found by adding $\mathbf{d}_{HV}$ to the calibration points, then transforming them with $\mathbf{X}_{TD}$; this is shown in equation 5.4. Gaussian noise is added to each point ($\mathbf{x}_{noise}$). The noise is applied in calibration-space because the z-axis intersects the display. This way, we can add more noise in the z direction to simulate depth inaccuracies. The transformation $\mathbf{X}_{DC}$ can be found for each point by creating a look-at matrix.

$$\mathbf{x}_{recorded} = \mathbf{X}_{TD}^{-1}\ \mathbf{x}_{calibration} + \mathbf{X}_{TD}^{-1}\ \mathbf{X}_{DC}^{-1}\ \mathbf{d}_{HV} + \mathbf{X}_{TD}^{-1}\ \mathbf{X}_{DC}^{-1}\ \mathbf{x}_{noise} \tag{5.4}$$

For validation, 1000 different combinations of $\mathbf{X}_{TD}$ and $\mathbf{d}_{HV}$ were randomly generated to create a large variety of test cases. For each test, 24 combinations of $\mathbf{x}_{noise}$ and $\mathbf{x}_{recorded}$

were generated. $\mathbf{x}_{noise}$ consisted of two components: Gaussian noise with a mean of 0 cm and a standard deviation of 1 cm in the x and y directions, and Gaussian noise with a mean of 0 cm and 11 levels of standard deviation (0 cm, 5 cm, 10 cm, 15 cm, 20 cm, 25 cm, 30 cm, 35 cm, 40 cm, 45 cm, 50 cm) in the z direction. All 1000 x 11 conditions were run through the 3 DoF version of the procedure to obtain estimates of $\mathbf{X}_{TD}$ and $\mathbf{d}_{HV}$. These estimates were used to transform the synthetic positions back into calibration-space to calculate a displacement vector. The mean magnitude of displacement vectors (excluding the z direction) were recorded as the error for each condition. A one-way ANOVA test reported no significant effect ($F_{10,10989} = 0.0007$, $p = 1.0000$) of standard deviation of depth error on displacement error (mean = 1.1 cm, standard deviation = 0.18 cm). The maximal pairwise Hedges g value was 0.0021. These results indicate that our proposed visual calibration technique is accurate and invariant to a broad range of depth errors.

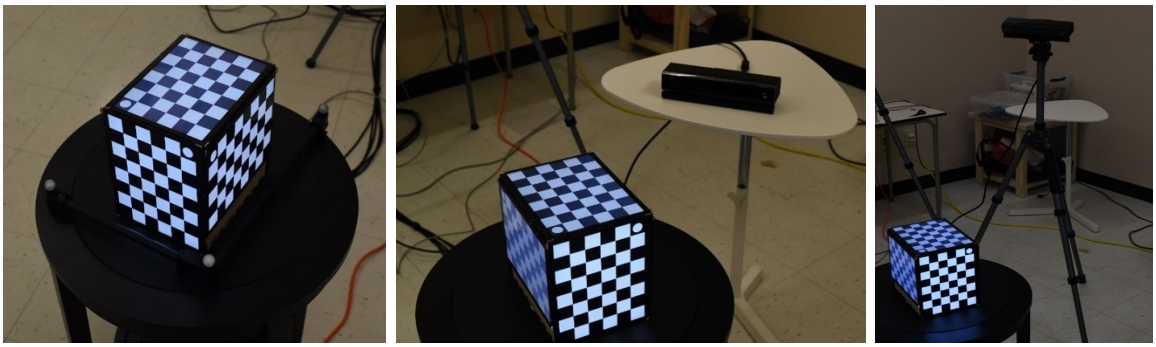
5.4 Camera-Based Calibration Experiment

A user study was designed to test how effective the calibration procedure is compared to a manual calibration. This study involved having users perform both the pattern based calibration procedure and a manual calibration using a physical camera on a cubic head-tracked display. The comparison was based on how accurate of a calibration could be obtained, as well as how long the calibration took to do. For all conditions, a marker was rigidly attached to a physical camera, as shown in Figure 5.6, and the goal was to calibrate to the camera's viewpoint. The user looked through the viewfinder of the camera for both calibrations. This setup is equivalent to monocular viewing, and allowed us to take pictures from the correct perspective afterwards, which is necessary for gauging calibration accuracy.

Figure 5.6: Tracked camera, used for calibration

5.4.1 Experimental Conditions

When performing a manual calibration, the difficulty of the task is largely dependent on how aligned the display and tracking system is. The easiest calibration occurs when all axes of the display and tracker are aligned. This case is the easiest since you only need to determine the translation between the display and tracker. This could be the case if you were using the OptiTrack optical tracker; a right angled metal bar is used to define the location and orientation of the tracking system, so it could be lined up with the display. A medium difficulty calibration occurs if only one axis of the display and tracker is aligned. This case is more challenging since you also need to determine how the display is rotated about the aligned axis, alongside its translation. This is the most likely case which could be achieved using most tracking systems; by placing the tracker on a level surface, the vertical axis of both the display and the tracker will be aligned. A hard calibration would be if none of the axes align. This is the hardest case since you need to determine the display's translation and full orientation relative to the tracker. This would most likely be the case if a Microsoft Kinect



(a) 3DoF **(b)** 4DoF **(c)** 6DoF

Figure 5.7: Calibration Difficulties: easy (3 degrees of freedom), medium (4 degrees of freedom), and hard (6 degrees of freedom)

was being used, mounted on a tripod, at a higher elevation. An example of the calibration difficulties are shown in Figure 5.7.

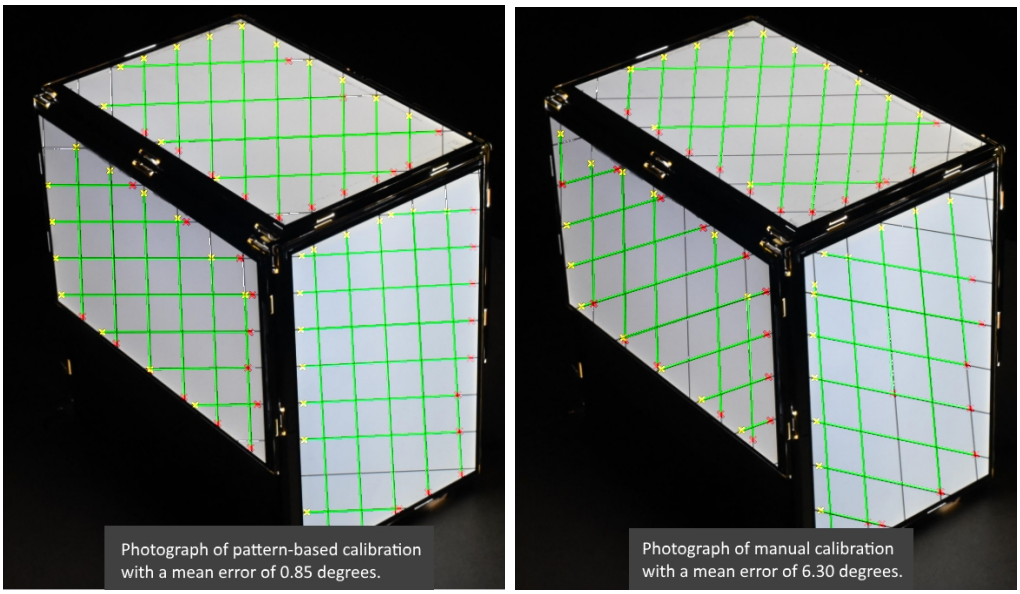
To test the calibration more thoroughly, we performed both calibration methods with all three alignments. The condition **AlignmentDoF** refers to how aligned the tracker and display spaces were at the start of the calibration trial. In 3DoF (easy), all the major axes are aligned. In 4DoF (medium), one of the major axes is aligned (up-axis in this case). In 6DoF (hard), none of the major axes are aligned. Since there is almost always some amount of offset between the user's viewpoint and head marker, we tested the effect of the offset on the result as well; **OffsetMagnitude** is the magnitude of $\mathbf{d}_{HV}$. The final condition, **CalibrationTechnique**, is the calibration method used. Pattern-based calibration is our proposed calibration technique of using a pattern to guide the viewer to a known position. Manual calibration is a real-time interactive process where the viewer inspects a perspective-corrected scene for visual alignment errors and attempts to reduce the noted errors by using a keyboard to modify the transformation $\mathbf{X}_{TD}$. Depending on the AlignmentDoF, the user can either modify only the translation (3DoF), the translation and one rotation (4DoF), or the translation and all three rotations (6DoF).

5.4.2 Procedure

To compare calibration techniques in a variety of orientations, two experts, experienced in manually tuning FTVR displays, performed Pattern-based and Manual calibrations through all levels of AlignmentDoF. To demonstrate the effect of viewpoint to head-tracked offset, a single expert performed Pattern-based and Manual calibrations with the highest level of alignment (3DoF) through all levels of OffsetMagnitude. The study was conducted on a 5-screen (7-inch screens, 1280x800 resolution) cubic FTVR display. The pattern-based calibration used the pattern shown in Figure 5.3b as the rendered pattern. The number of calibration points (positions at which the pattern will appear correct) was limited to 24 to balance time and accuracy. They were randomly generated in clusters of six around the top four corners of the display, and presented in a clockwise order to reduce backtracking of the user. These calibration points were used for all Pattern-based calibrations to acquire the corresponding point set data needed for our optimization algorithm. The manual calibration used a scene consisting of two cans of soda because they are an easily recognizable object. The Manual calibration requires an initial transformation to use as a starting point; the initial transformation we used consisted of a calibrated transformation which was offset by 50 cm in a random direction, and rotated by 10° about a random axis. The initial transformation represents what a person could achieve by examining the location and orientation of the display and tracking system without the aid of a rendered scene.

Once the calibrations were completed, the display was set to render a perspective-corrected grid pattern. The camera was placed on a tripod and photographs were taken of each successful calibration from eight different locations. The photographs were run through a semi-automatic image processing and analysis script that used a Hough transform to find lines in the images. Lines were paired with their corresponding lines in adjacent screens and the angle at which they met was recorded. A perfect calibration would result in parallel lines and an angle of 0°. Figure 5.8 illustrates two representative calibration test images.

We used an OptiTrack 6 DoF optical tracking system to perform head tracking, but only positional (3 DoF) data was recorded. The display software, which also recorded the data,



(a) Low error **(b)** High error

Figure 5.8: Calibration analysis images; green lines show detected lines, red and yellow x's show line ends

was written in the Unity game engine. The optimization algorithm and photograph analysis scripts were implemented in MATLAB. Time was recorded using a stopwatch. All photographs and calibrations were performed using a Nikon D750 with a Nikon 50mm f/1.4 lens.

Calibration time for the Pattern-based method was measured from the first recorded point to the last recorded point. Calibration time for the Manual method was combined from two sources: the time it took to setup the alignment condition, and the time between the start of the first modification and the time at which the participant voiced their completion.

5.4.3 Design and Hypotheses

The study used a 2x3 within-participants two-way RM-ANOVA with factors CalibrationTechnique (Pattern-based, Manual) and AlignmentDoF (3DoF, 4DoF, and 6DoF), as well as a 2x3 within-participant two-way RM-ANOVA with factors CalibrationTechnique (Pattern-based, Manual) and OffsetMagnitude (5 cm, 10 cm, and 15 cm). Dependent measures were

calibration time and error, and we examined the main effects of the factors. Hypotheses were:

H1. Pattern-based calibration will be at least as accurate as Manual calibration with zero OffsetMagnitude applied.

H2. Pattern-based calibration will be faster than Manual calibration.

H3. 3DoF will be the most accurate for Manual calibration.

H4. Pattern-based calibration will be more accurate than Manual calibration when there is a non-zero OffsetMagnitude applied.

H5. As OffsetMagnitude increases, Manual calibration will be less accurate.

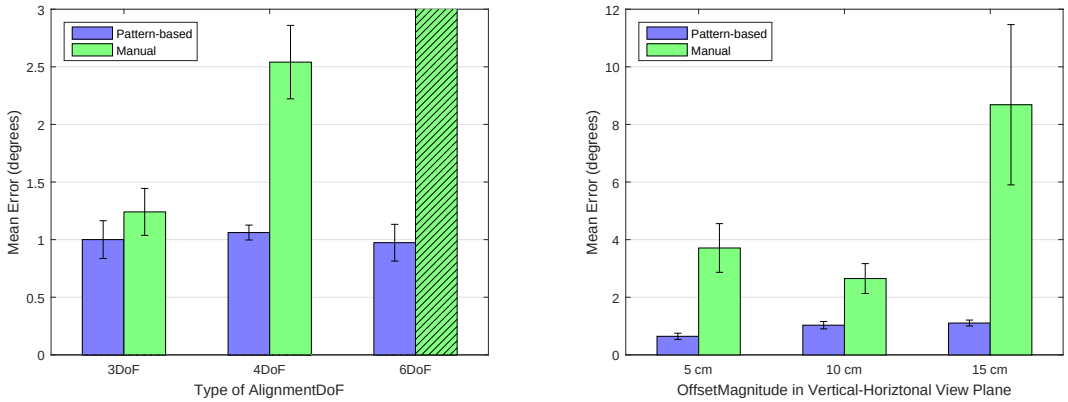
5.4.4 Results

Errors

Mean calibration errors for varying AlignmentDoF and OffsetMagnitude are shown in Figures 5.9a and 5.9b. A two-way RM-ANOVA showed a main effect of CalibrationTechnique and AlignmentDoF on error ($F_{4,35}=11.12$, $p<0.01$). A Tukey-HSD multiple comparison post-hoc test reported a significant ($p<0.01$) difference between Manual 3DoF and Manual 4DoF. We therefore accept H3.

The post-hoc test also reported that the only significant difference ($p<0.01$) observed of CalibrationTechniques over AlignmentDoF is between Pattern-based 4DoF and Manual 4DoF. We therefore accept H1.

A two-way RM-ANOVA showed a main effect of CalibrationTechnique and OffsetMagnitude on error ($F_{5,42}=6.29$, $p<0.01$). A Tukey-HSD multiple comparison post-hoc test reported a significant difference ($p<0.01$) between Manual 15 cm and Pattern-based 15 cm, Pattern-based 10 cm, and Pattern-based 5cm. We therefore accept H4.



(a) Mean error vs. DoF **(b)** Mean error vs. $\mathbf{d}_{HV}$ magnitude

Figure 5.9: Calibration results for comparing Pattern-based and Manual calibration techniques; error bars represent standard error

It also showed a significant ($p<0.01$) effect of OffsetMagnitude on Manual with Manual 15 cm error being much larger than both Manual 10 cm and Manual 5cm. We therefore accept H5.

Times

A two-way RM-ANOVA did not show significant results for completion time. We therefore fail to accept H2. Across all successful trials, we recorded a mean completion time of 2 minutes and 24 seconds and a standard deviation of 57 seconds.

5.4.5 Discussion

The results show that, for every condition, the Pattern-based calibration method performs either comparable to, or better than, doing a Manual calibration. It showed consistent and accurate results in all the tested scenarios, while not significantly impacting the amount of time a calibration takes. The quick completion times for both calibration techniques can be partly attributed to the expert's familiarity with the techniques, and the ease of re-defining

the head-tracker space using OptiTrack. Both participants spent a considerable amount of time (>10 minutes) attempting a Manual calibration in the 6DoF alignment case, but they were unsuccessful in achieving a workable transformation. The axes were so misaligned that a camera-based analysis of the error was not possible. Our inability to measure the error in this case is represented in Figure 5.9a as a hatched bar extending beyond the chart.

We tested a range of AlignmentDoF conditions to demonstrate the robustness of our Pattern-based calibration, even though a 6DoF case is unlikely and avoidable if the head-tracker or display can be moved easily. The 4DoF case is the most likely to be seen in practice, as it is quite easy to align one of the axes. For example, one could align both the head tracker and display to the floor, thus aligning at least one of their axes.

A cubic display was used for testing the calibration procedure because it is a challenging calibration problem. The shape of the display allows it to be viewed from almost any location, which makes calibration errors more noticeable. Our cubic display has seams between each screen, which breaks up the pattern, making calibration more challenging. The calibration algorithm has also been tested on spherical and corner displays, and the pattern-based calibration worked equally well.

We used a camera, instead of the participants eyes, to allow for a quantitative analysis of calibration errors. The camera also removes the binocular disparity that would be present with participant viewing [51], which is problematic for our current cubic display that does not have screens that can do stereoscopic rendering. When calibrating with both eyes, the user would choose the best visual alignment, which should naturally account for eye dominance. However, since the pattern would be viewed from two viewpoints, it would not appear perfectly undistorted as it does for a monocular viewpoint and we would expect this to introduce a degree of subjective error. Another limitation of our study is that calibrations were performed by experts with a high degree of familiarity with the technique.

Our results show that OffsetMagnitude has a large effect on the accuracy of a calibration if it is left unaccounted for. We solved this problem without requiring the tracking system to measure the orientation of the participant's head so that our technique would have fewer

requirements, making it would be accessible to a larger variety of tracking systems, including those that only provide 3D position data and not head/gaze direction.

Unlike previously reported line-of-sight methods [16] or see-through HMD approaches [53], our approach does not require any additional physical apparatus attached to the display or head-mounted screens that the user looks through. This makes our method much simpler to implement, but at the cost of having poor depth accuracy at each measured viewpoint. We solve this with our depth-invariant calibration formulation: each individual viewpoint measurement ignores tracker errors along the ray of the viewpoint, and we use multiple measurements surrounding the display to reconstruct a full rigid transformation between the display and tracker. For this reason, Pattern-based calibration would not work for single-screen FTVR displays with limited viewing angles, but our interest is primarily in multi-screen FTVR displays that provide a stronger 3D effect and have better potential for use in work and entertainment applications.

5.5 Summary

Viewpoint calibration is necessary to be able to correctly position the user's viewpoint around the display. By utilizing rendered patterns, the user can place their viewpoint at specific locations around the display and the system can solve for the unknown transformations. The presented algorithm is able to produce accurate results even if the user is the incorrect distance from the display. The pattern-based calibration method was found to generate more accurate calibrations than a manual calibration could.

The pattern-based calibration procedure relies on the user's ability to position their viewpoint properly. A pattern is used to assist the user in positioning their viewpoint, but an exact positioning is unlikely to occur. There are multiple factors that impact how well the user can do the calibration; the two main factors are the pattern being rendered, and the shape of the display. The next chapter will describe a user study designed to examine how these factors affect individual's viewpoint alignment ability.

Chapter 6

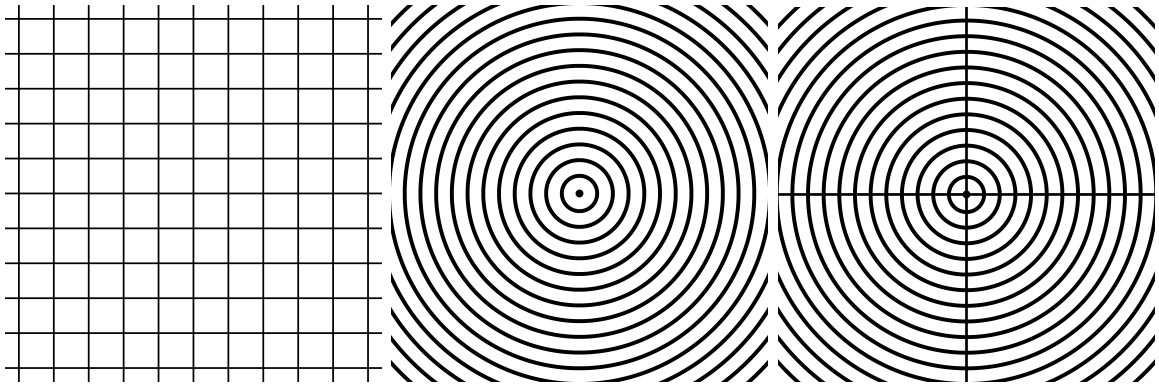
Pattern Alignment Study

The previous chapter detailed a pattern based approach to performing the viewpoint calibration. The success of the algorithm is largely dependent on the user's ability to position their viewpoint properly. If the viewpoint is positioned perfectly, the algorithm could find a near perfect $\mathbf{X}_{TD}$ and $\mathbf{d}_{VH}$. If the user is unable to position their viewpoint properly, errors will be introduced into the transformations. A user study was created to better understand people's interaction with the visual alignment task. The study was setup to simulate the calibration process in order to examine how different rendered patterns and display shapes impact the performance of the calibration.

Pattern

The calibration utilizes an image rendered on the display, but there is no restriction about the image being used. Different types of patterns in an image can provide different information to help with the alignment process. We wanted to study the effect that different pattern information has on people's alignment ability. To simplify the comparison, we considered two different types of visual cues which commonly occur in images: straight lines and circles. Three different pattern designs were generated which contained these cues: the patterns are Circles, Grid, and Combo, and are shown in Figure 6.1. The Circles pattern consists of concentric circles converging at the center of the pattern; this pattern has a bulls eye effect to it. The Grid pattern consists of a series of equally spaced horizontal and vertical lines, which will appear straight if viewed from the correct position. The lines will also appear

parallel to lines of the same direction, and perpendicular to lines in the other direction. The Combo pattern utilizes visual cues present in both the Circles and the Grid pattern by overlaying the Circles pattern with a horizontal and a vertical line converging in the center of the image.

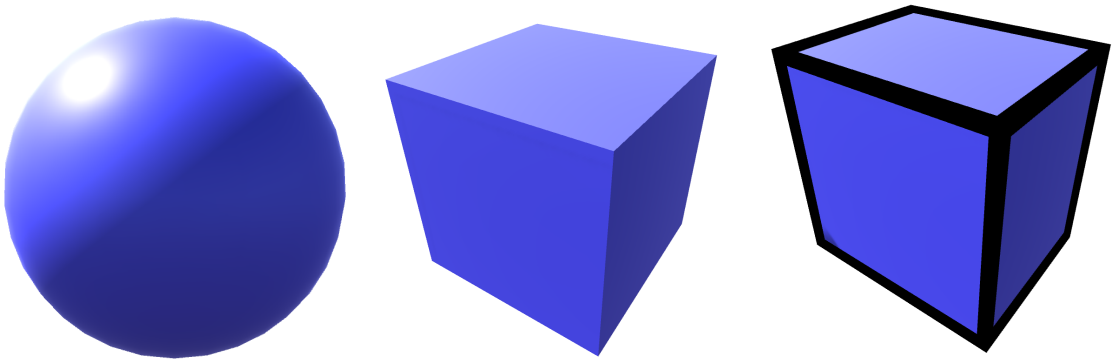


(a) Grid patterns **(b)** Circles pattern **(c)** Combo pattern

Figure 6.1: Calibration patterns used in the pattern alignment study

Shape

Since the calibration process can be done on any display shape, we also wanted to test how the display shape affects the calibration process. Different display shapes will distort images in different ways which will affect what the user will see when calibrating. The viewed distortion acts as a guide to the user, indicating how the viewpoint should be moved to reach the intended viewing position. Different distortions may be deciphered differently which could affect the calibration process. The most common FTVR display shapes are cubic and spherical displays, and we generated three different shapes to represent the displays. The three shapes include a Sphere, a Cube, and a Box shape; these are shown in Figure 6.2. The Sphere and the Cube represent seamless displays which could be achieved with back projected concave shapes. The Box shape represents a display comprised of multiple screens in the shape of a box. Because the Box display is made up of screens with noticeable borders around them, the displayed content will be disrupted and a continuous pattern will not be visible.



(a) Sphere **(b)** Cube **(c)** Box

Figure 6.2: Display shapes used in the pattern alignment study

6.1 Study Description

The study had participants perform a simulated calibration process on a desktop computer. An application was developed in the Unity game engine that simulated the visual alignment process. The goal of the study was to determine which rendered pattern and which display shape results in the best calibration. The quality of the alignment was based on how fast and accurately the viewpoint could be positioned.

6.1.1 Apparatus

A Unity application was developed for the visual alignment task which recorded accuracy and completion time. Participants performed the study on a Windows 10 machine with an Intel Xeon E5-2643 CPU, a standard Dell mouse, and a 24 inch BenQ monitor running at 60 Hz. Mouse input was used to pan the camera around an object at a fixed radius. Patterns were texture mapped onto the object such that they appeared undistorted from a set camera location (goal location). The texture mapping was accomplished by placing a virtual camera at the goal location, and oriented towards the center of the object. Each fragment color was determined by sampling the 2D pattern at the same location as it appeared on the virtual camera's screen. The shader that was used is shown in Appendix C.

6.1.2 Participants and Procedure

Eighteen participants were recruited from a local university (14 males and 4 females). In each trial, the participant used the mouse to move the camera around the shape until the pattern on the shape no longer appeared distorted. Once the participant was satisfied with the alignment, they clicked the left mouse button. Between each trial, the camera would reset to its default location and the goal location would change. Alignment accuracy was measured as the angle in degrees between the recorded virtual camera location and the goal location.

Before starting the study trials, participants completed nine training trials (one per pattern and shape combination) to familiarize themselves with the controls and to better understand the possible distortions. Participants completed trials using every combination of the three shapes and three patterns. The conditions were rotated between participants to account for possible sequencing effects. For each combination, 24 unique goal locations, equally divided into two blocks, were used. The order of the goal locations was randomly generated for each combination. Participants observed each of the patterns on a single shape before changing shapes. After each pattern, participants filled out a NASA-TLX questionnaire. They also answered summary questions comparing the different patterns. Participants were informed that both their accuracy and completion time would be recorded for each trial, but that they should focus more on accuracy.

6.1.3 Task

The process involved having the participants use the mouse to orbit a virtual camera around the shape. Once the pattern that is mapped onto the shape appears undistorted, the participants were to the left mouse button.

6.1.4 Dependent Measures

The study used a 3x3 within-participant two-way RM-ANOVA with factors **Shape** (Sphere, Cube, Box) and **Pattern** (Circles, Grid, Combo). Dependent measures were completion time and alignment error, and we were examining the main effects of the factors. We expected Cube to result in a higher accuracy than Sphere because small distortions would be more noticeable due to the sharp edges around the shape. We also expected that the Combo would be most accurate as it has both visual cues, but that Circle would be faster as it has less visual information to process. The hypotheses were:

H1. Mean error will be lowest for Cube.

H2. Mean error will be lowest for Combo.

H3. Mean completion time will be lowest for Sphere.

H4. Mean completion time will be lowest for Circles.

6.2 Results

6.2.1 Accuracy

The mean alignment error across the conditions is shown in Figure 6.4a. A two-way RM-ANOVA test over all trials with Shape and Pattern showed that they have a main effect on accuracy of the task ($F_{2,3879} = 177.81$, $p < 0.01$; $F_{2,3879} = 6.15$, $p < 0.01$). The Tukey-HSD multiple comparison post-hoc test showed that participants were significantly more accurate with Sphere than both Cube ($p < 0.01$) and Box ($p < 0.01$). It also showed that Cube was significantly more accurate than Box ($p < 0.001$). We therefore reject H1. The test also found that participants were significantly more accurate when using Combo than Circles ($p < 0.01$), but Combo and Circles were not significantly different than Grid ($p = 0.1357$, $p =$

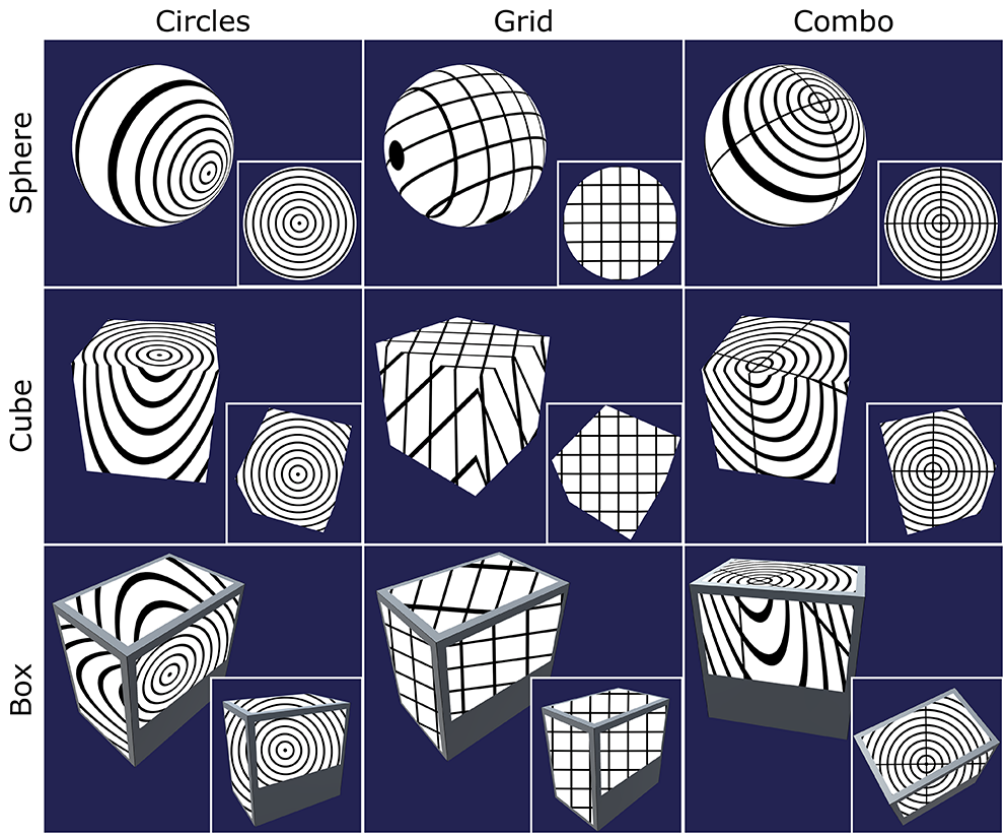


Figure 6.3: Alignment conditions for the pattern alignment study. This shows what would be seen if unaligned (larger) and aligned (smaller) for each combination of pattern and shape

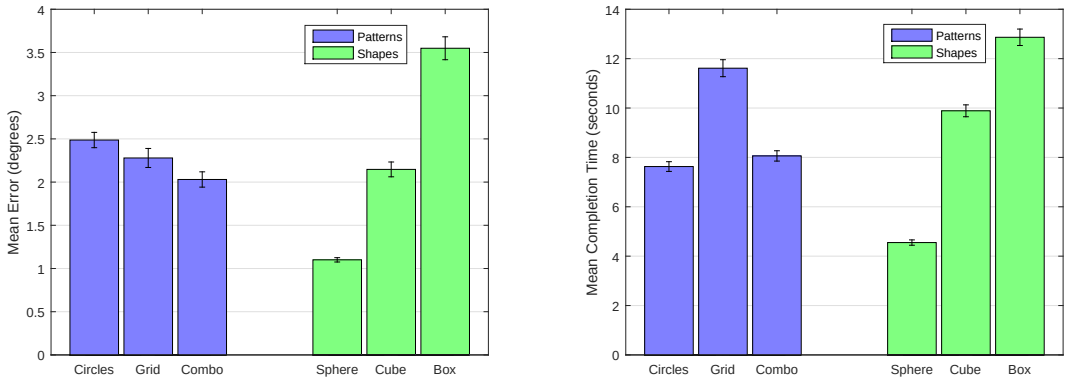
0.2485). We therefore reject H2.

6.2.2 Time

Mean alignment error across conditions is shown in Figure 6.4b. A two-way RM-ANOVA over Shape and Pattern showed that there was a main effect in task completion time ($F_{2,3879} = 311.53$, $p < 0.01$; $F_{2,3879} = 83.9$, $p < 0.01$). The Tukey-HSD multiple comparison post-hoc test showed that Sphere is significantly faster than both Cube ($p < 0.01$) and Box ($p < 0.01$). It also showed that Cube was significantly faster than Box ($p < 0.01$). We therefore accept H3.

The post-hoc analysis also found that Combo and Circles are significantly faster than Grid

($p < 0.01$, $p < 0.01$), but not significantly different from each other ($p = 0.409$). We therefore reject H4.



(a) Mean Alignment Error **(b)** Mean Alignment Time

Figure 6.4: Pattern alignment study results; error bars represent standard error

6.2.3 Questionnaire

NASA-TLX Questionnaire responses were similar across Pattern conditions, but varied more considerably for the Shape conditions, shown in Figure 6.5. A one-way RM-ANOVA showed significant differences for mental demand ($F_{2,150} = 30.19$, $p < 0.01$), performance ($F_{2,148} = 6.74$, $p < 0.01$), difficulty ($F_{2,146} = 29.6$, $p < 0.01$), and insecurity ($F_{2,156} = 8.48$, $p < 0.01$). A one-way RM-ANOVA failed to show significant differences for physical demand ($F_{2,152} = 4.46$, $p = 0.0131$) and pace ($F_{2,150} = 1.25$, $p = 0.289$). A Tukey-HSD multiple comparison post-hoc test was run on the mental demand, performance, difficulty, and insecurity responses. Box was more mentally demanding than both Cube ($p < 0.01$) and Sphere ($p < 0.01$), as well as Cube was more mentally demanding than Sphere ($p < 0.01$). Participants felt more successful with Sphere than Box ($p < 0.01$), but there was no considerable difference between Sphere and Cube ($p = 0.1586$) nor Cube and Box ($p = 0.01493$). Box scored higher in difficulty than both Sphere ($p < 0.01$) and Cube ($p < 0.01$). Cube scored higher than Sphere ($p < 0.0001$). Participants felt more insecure with Box than Sphere ($p < 0.001$), but there was no considerable difference between Box and Cube ($p = 0.3750$) nor

between Cube and Sphere ($p = 0.0182$).

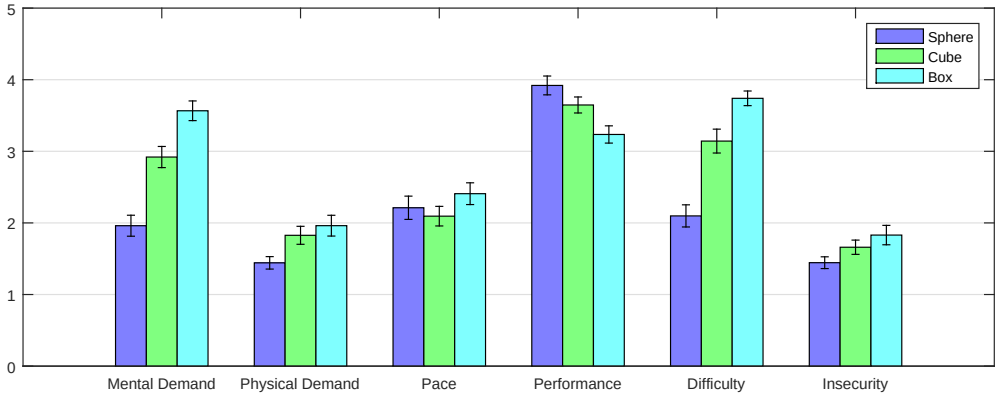


Figure 6.5: NASA-TLX questionnaire results for display shape; error bars represent standard error

6.2.4 Sides Visible

For the Cube and Box shapes, the number of sides visible (one, two, or three) from every goal location was also recorded for each trial. Mean alignment error for trials with different numbers of sides visible is shown in Figure 6.6 . A one-way RM-ANOVA found that having three sides visible resulted in significantly better accuracy than having two sides visible ($p < 0.01$), and having two sides visible resulted in significantly better accuracy than having one side visible ($p < 0.01$).

6.3 Discussion

The study found a considerable difference in both accuracy and completion time when different shapes are used. Sphere achieved the lowest mean error as well as the lowest mean completion time. This is a surprising and strong result as it breaks the typical speed vs. accuracy trade-off found in most pointing or alignment tasks. A possible reason for this is that the pattern distortion is much more gradual on a spherical display, which could make it

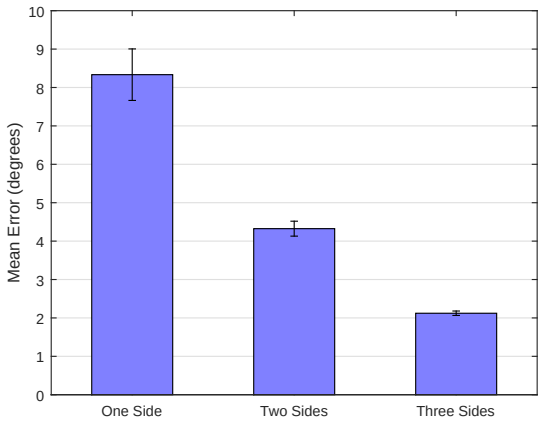


Figure 6.6: Mean alignment error vs. number of sides visible for cubic display shapes; error bars represent standard error

easier to determine where the camera should be moved to. Performance with the Cube shape was worse than we expected. Although the cube edges provided strong visual cues (kinked lines at the edge between screens), the distortion effect may have been difficult to interpret by participants, and required more visual processing effort and time. Box had the worst performance which indicates that the seams between screens made pattern alignment significantly more difficult. This is an important design consideration as cubic displays constructed with LCD screens have seams, while rear-projection cubic displays can be made seamless.

The results show that there is a difference in both accuracy and completion time when different patterns were used. Combo resulted in the highest accuracy, but was only considerably more accurate than Circles. This could indicate that straight lines provide additional visual cues that aid in correcting distortion. However, given that Grid was observed to be considerably slower than the other two patterns, lines by themselves may not give enough visual cues to quickly align the pattern. Circles and Combo naturally provide a better sense of direction than Grid because the bullseye pattern has a defined center. Thus, the participants could have had a more accurate initial guess at how to correct the distortion using Circles and Combo, making them faster.

The results shown in Figure 6.4a indicate how much alignment error should be expected when calibrating, but they also convey how much error is allowed in the system before it becomes

noticeable. The Sphere was able to achieve the lowest mean alignment error which indicates that minor calibration errors could be more noticeable on the Sphere than the cubic display shapes. The Box had the highest mean alignment error which is most likely attributed to the visual disruption of the pattern. While this may be undesirable, it also indicates that calibration errors may be less noticeable due to the disruption. The same can be said about the type of content being rendered. The results indicate that a combination of straight lines and circles will appear more distorted than just straight lines or circles when calibration errors are present. These findings should be taken into consideration when designing content for multi-screen FTVR displays.

The questionnaire results show a general trend in all questions except for pace. On average, people felt the best about the Sphere shape in terms of performance and how demanding the task was, and people felt the worst about the Box shape. These results are expected as they match what is shown in Figure 6.4. The pace results are most likely attributed to the participants being instructed to focus more on accuracy than completion time.

When doing an alignment on display made up of flat screens, the distortion is most noticeable when comparing the images on multiple screens together. As straight lines transition from one screen to another, you will see a bend in the line if viewed from the incorrect location. Since this distortion helps to find the correct viewing position, having more screens visible should help with the alignment. The results of the study have shown that having more sides visible does reduce alignment error when using a cubic shape display. For this reason, calibration positions should be chosen around the corners of the display, since that is where the maximal number of screens are visible. When using a spherical display, the distortion is not concentrated between screen transitions, therefore calibration positions can be chosen freely.

6.4 Summary

A user study was conducted to compare the effectiveness of different patterns and different display shapes, while doing a visual alignment task. The study had participants use a computer mouse to move a camera around a virtual object, which had a pattern texture mapped onto it. The goal was to position the camera where the pattern no longer appeared distorted. The results showed that the type of display used has a larger impact on alignment quality than the pattern that is used. The Sphere display shape performed the best and the Box display performed the worst. These results indicate that the visual disruption of screen borders, and concentrated distortion of cubic displays, negatively impact alignment ability. For patterns, the Combo pattern outperformed the other two. This tells us that a combination of straight lines and circles provides more useful information than only straight lines or only circles.

Chapter 7

Conclusion

This chapter provides a general discussion of the contributions and limitations of the work described in this thesis. This chapter also proposes ideas for future work, as well as a final conclusion.

7.1 Contributions

The contributions presented in this thesis are as follows:

1. We constructed novel spherical and cubic FTVR displays;
2. We presented a semi-automatic method for the screen calibration of cubic FTVR displays;
3. We presented a novel visual viewpoint calibration technique for multi-screen FTVR displays;
4. We presented a method for evaluating viewpoint calibration quality for a cubic FTVR display; and
5. We performed an evaluation of patterns and display shapes for visual viewpoint calibration.

1. FTVR Display Construction

Section 4.1 describes the FTVR displays that were constructed. The cubic display was built out of five LCD screens mounted on a wooden frame and offers higher resolution and smaller bezels than its predecessor: the pCubee [50]. The stereo capable sphere display consists of four projectors illuminating the inside of a large plastic sphere. Both of these displays provide a large variety of viewing angles by allowing the user to walk all the way around them.

2. Cubic Display Screen Calibration

The screen calibration method discussed in Section 4.2.2 is able to find the position and orientation of each screen relative to every other screen. This provides us with the location of each screen pixel relative to the displays coordinate frame, which is necessary for rendering. This is done by using a camera to take pictures of the screens. A chessboard pattern is rendered on the screens, and the location of the pattern's corners are used to simultaneously calibrate the camera and display. An interactive program was made to find the location of the corners and solve for the location and orientation of the screens.

3. Visual Viewpoint Calibration Technique

The viewpoint calibration method described in Chapter 5 is able to determine the transformation relating the displays coordinate frame to the tracking systems coordinate frame. It can also find any offset that may exist between the location of the user's viewpoint and the head marker worn by the user. The calibration process involves rendering patterns on the display that only appear undistorted from a single location, and having the user position their viewpoint at that location. The calibration process is done without the need for any additional equipment as patterns are used to guide the user to the correct location. This method of calibration was found to be more accurate than a manual calibration, and can

easily be redone when needed. This method does not require the user to be the correct distance away from the display, and is robust enough to work on any shape of display.

4. Viewpoint Calibration Evaluation

The evaluation method explained in Section 5.3.4 has the ability to determine the quality of a viewpoint calibration. This method works by rendering a grid pattern onto the display, and taking pictures of the display from a variety of different locations. The quality of the calibration is determined based on how much the lines change direction when transitioning from one screen to another. This algorithm can provide a quantitative representation of the visual errors that would be seen when viewing the display monocularly.

5. Pattern and Display Shape Evaluation

The study explained in Chapter 6 was designed to better understand people's interaction with the viewpoint calibration process. The quality of the calibration is dependent on how accurately the user can position their viewpoint; different pattern information and distortion will affect how easy the task is. The study that was conducted analyzed how effectively people could position their viewpoint with different rendered patterns and display shapes. The study found that a combination of straight lines and circles is the best pattern to use, and that spherical displays are easier to calibrate than cubic displays.

7.2 Limitations

There are limitations to the calibration and evaluation techniques presented in this thesis.

To make the screen calibration method discussed in Section 4.2.2 more convenient, one of

the screens can be chosen to be the origin of the coordinate frame, which would ideally be visible in each picture that is taken. If the geometry of the display is too complex, this may not be possible. In this situation, more transformations will need to be combined together, resulting in a larger error in the calibration result. The calibration process also assumes the screens will be rigidly attached to each other, ideally with an underlying frame. The frame of the display can be moved after calibration without effecting the result, but if the screens move relative to each other, calibration will need to be redone.

While the viewpoint calibration method described in Chapter 5 can be done on any shape of display, it performs best on displays with a large variety of viewing angles (sphere and cubic displays). For displays that do not support a large amount of viewing angles (corner displays), the algorithm may obtain less accurate results. The optimization algorithm solves for the transformations that best relate two sets of points. With less variation in the points, the algorithm will have a harder time distinguishing the effects from the different transformation contributions. This could result in both the tracker-to-display transformation and the head-to-viewpoint offset being incorrect. Even though the individual components of the transformation may be wrong, their combination will be fairly correct, and the rendered content will still appear relatively good in the region where calibration points were recorded.

In its current state, the evaluation method explained in Section 5.3.4 only works when used on displays made up of discrete planar screens. If the surface of the display is not planar, any calibration errors will cause straight lines to bend gradually as opposed to having the bend concentrated at the screen transition. The algorithm that is used would need to detect the curvature in the line rather than the angle between two lines. This method also requires an accurate display calibration to be effective; errors in the display calibration may result in a lower computed quality.

7.3 Future Work

The work presented in this thesis has revealed a number of interesting directions for future work to be done with FTVR displays and their calibration. The following are three suggestions of possible future work.

Calibration Modification for Different Viewers

The viewpoint calibration algorithm is able to find the head-to-viewpoint offset for the individual performing the calibration. This offset is dependent on the calibrator's eye dominance and how they wear the head marker used for tracking. A different individual would have a different head-to-viewpoint offset, and the rendered content would look distorted if not calculated. The entire calibration could be redone, but this would be unnecessary as the tracker-to-display transformation is still valid. A secondary calibration method, that only modifies the head-to-viewpoint offset, would allow multiple different viewers to be able to view the display properly without each having to do the full calibration.

Stereoscopic Calibration Process

For stereoscopic rendering, the location of both eyes must be known. There will be a head-to-viewpoint offset for each eye that must be calculated. Currently, this can be done by performing the calibration with one eye closed to find one offset; the other offset can then be found by adding the interpupillary distance. By utilizing stereoscopic rendering, the calibration could be done for both eyes simultaneously. This process would be preferred as it does not require the interpupillary distance of each user to be known.

Collaborative Viewing Study

Multi-screen FTVR displays have great potential for collaborative work when viewing 3D data. By utilizing stereoscopic rendering, two viewers can observe perspective corrected content simultaneously. This type of collaborative viewing could allow for a more natural interface for discussions about 3D data. A comparison of collaborative tasks being performed in multi-screen FTVR displays and standard computer monitors should be done to better understand the potential benefits.

7.4 Concluding Remarks

This thesis addresses the challenges of calibrating head-tracked FTVR displays, and details new, easy to use procedures for quickly obtaining accurate display calibrations. The work also introduces a novel calibration technique for performing the viewpoint calibration step which utilizes rendered patterns. This method of calibration does not require any additional hardware, and has been found to achieve accurate calibrations. The method is also not dependent on the shape of the display, and can work with any type of head tracking system. We believe that head-tracked FTVR displays can provide an interactive way to visualize 3D data like no other display, and that the contribution in this work will help with their adoption into industry and consumer markets.

References

[1] Anaglyph red/cyan 3d glasses with retail display box - 50 3d glasses. https://www.rainbowsymphonystore.com/products/3d-anaglyph-glasses-retail-box.

[2] B-con engineering. http://www.bconeng.com/.

[3] The world's most widely used tracking library for augmented reality. https://www.artoolkit.org/.

[4] Oculus rift. https://en.wikipedia.org/wiki/Oculus_Rift, Oct 2017.

[5] K. W. Arthur, K. S. Booth, and C. Ware. Evaluating 3d task performance for fish tank virtual worlds. *ACM Transactions on Information Systems (TOIS)*, 11(3):239–265, 1993.

[6] K. S. Arun, T. S. Huang, and S. D. Blostein. Least-squares fitting of two 3-d point sets. *IEEE Transactions on pattern analysis and machine intelligence*, (5):698–700, 1987.

[7] R. Azuma and G. Bishop. Improving static and dynamic registration in an optical see-through hmd. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, pages 197–204. ACM, 1994.

[8] M. Bajura and U. Neumann. Dynamic registration correction in video-based augmented reality systems. *IEEE Computer Graphics and Applications*, 15(5):52–60, 1995.

[9] P. J. Besl, N. D. McKay, et al. A method for registration of 3-d shapes. *IEEE Transactions on pattern analysis and machine intelligence*, 14(2):239–256, 1992.

[10] M. Billinghurst and H. Kato. Collaborative augmented reality. *Communications of the ACM*, 45(7):64–70, 2002.

[11] J. Bolton, K. Kim, and R. Vertegaal. Snowglobe: A spherical fish-tank vr display. In *CHI'11 Extended Abstracts on Human Factors in Computing Systems*, pages 1159–1164. ACM, 2011.

[12] C. W. Borst. Tracker calibration using tetrahedral mesh and tricubic spline models of warp. In *Virtual Reality, 2004. Proceedings. IEEE*, pages 19–26. IEEE, 2004.

[13] J.-Y. Bouguet. http://www.vision.caltech.edu/bouguetj/calib_doc/, Oct 2015.

[14] L. Cosmo, A. Albarelli, F. Bergamasco, and A. Torsello. Design and evaluation of a

viewer-dependent stereoscopic display. In *ICPR*, pages 2861–2866, 2014.

[15] C. Cruz-Neira, D. J. Sandin, and T. A. DeFanti. Surround-screen projection-based virtual reality: the design and implementation of the cave. In *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, pages 135–142. ACM, 1993.

[16] M. Czernuszenko, D. Sandin, and T. DeFanti. Line of sight method for tracker calibration in projection-based vr systems. In *Proceedings of 2nd international immersive projection technology workshop*, pages 11–12. Citeseer, 1998.

[17] A. Damala. *Interaction Design and Evaluation of Mobile Guides for the Museum Visit: A Case Study in Multimedia and Mobile Augmented Reality*. PhD thesis, Conservatoire national des arts et metiers-CNAM, 2009.

[18] J. Djajadiningrat, G. Smets, and C. Overbeeke. Cubby: a multiscreen movement parallax display for direct manual manipulation. *Displays*, 17(3):191–197, 1997.

[19] S. Feiner, B. MacIntyre, and D. Seligmann. Annotating the real world with knowledge-based graphics on a see-through head-mounted display. In *proceedings of Graphics Interface*, volume 92, pages 78–85, 1992.

[20] F. Ferreira, M. Cabral, O. Belloc, G. Miller, C. Kurashima, R. de Deus Lopes, I. Stavness, J. Anacleto, M. Zuffo, and S. Fels. Spheree: a 3d perspective-corrected interactive spherical scalable display. In *ACM SIGGRAPH 2014 Posters*, page 86. ACM, 2014.

[21] A. Fuhrmann, D. Schmalstieg, and W. Purgathofer. Fast calibration for augmented reality. In *Proceedings of the ACM symposium on Virtual reality software and technology*, pages 166–167. ACM, 1999.

[22] J. G. Hagedorn, S. G. Satterfield, J. T. Kelso, W. Austin, J. E. Terrill, and A. P. Peskin. Correction of location and orientation errors in electromagnetic motion tracking. *Presence: Teleoperators and Virtual Environments*, 16(4):352–366, 2007.

[23] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[24] O. Hilliges, D. Kim, S. Izadi, M. Weiss, and A. Wilson. Holodesk: direct 3d interactions with a situated see-through display. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 2421–2430. ACM, 2012.

[25] R. L. Holloway. Registration error analysis for augmented reality. *Presence: Teleoperators and Virtual Environments*, 6(4):413–432, 1997.

[26] M. Inami. Media 3: the virtual hologram. In *ACM SIGGRAPH 97 Visual Proceedings: The art and interdisciplinary programs of SIGGRAPH'97*, page 107. ACM, 1997.

[27] A. L. Janin, D. W. Mizell, and T. P. Caudell. Calibration of head-mounted displays

for augmented reality applications. In *Virtual Reality Annual International Symposium, 1993., 1993 IEEE*, pages 246–255. IEEE, 1993.

[28] B. Jones, R. Sodhi, M. Murdock, R. Mehra, H. Benko, A. Wilson, E. Ofek, B. MacIntyre, N. Raghuvanshi, and L. Shapira. Roomalive: magical experiences enabled by scalable, adaptive projector-camera units. In *Proceedings of the 27th annual ACM symposium on User interface software and technology*, pages 637–644. ACM, 2014.

[29] H. Kato and M. Billinghurst. Marker tracking and hmd calibration for a video-based augmented reality conferencing system. In *Augmented Reality, 1999.(IWAR'99) Proceedings. 2nd IEEE and ACM International Workshop on*, pages 85–94. IEEE, 1999.

[30] K. Kim, J. Bolton, A. Girouard, J. Cooperstock, and R. Vertegaal. Telehuman: effects of 3d perspective on gaze and pose estimation with a life-size cylindrical telepresence pod. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 2531–2540. ACM, 2012.

[31] V. V. Kindratenko. A survey of electromagnetic position tracker calibration techniques. *Virtual Reality*, 5(3):169–182, 2000.

[32] A. Kulshreshth and J. J. LaViola Jr. Evaluating performance benefits of head tracking in modern video games. In *Proceedings of the 1st symposium on Spatial user interaction*, pages 53–60. ACM, 2013.

[33] A. Kulshreshth and J. J. LaViola Jr. Dynamic stereoscopic 3d parameter adjustment for enhanced depth discrimination. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, pages 177–187. ACM, 2016.

[34] A. H. Lashkari, B. Parhizkar, and M. A. Mohamedali. Augmented reality tourist catalogue using mobile technology. In *Computer Research and Development, 2010 Second International Conference on*, pages 121–125. IEEE, 2010.

[35] K. Lee. Augmented reality in education and training. *TechTrends*, 56(2):13–21, 2012.

[36] F. Madritsch, F. Leberl, and M. Gervautz. Camera based beacon tracking: accuracy and applications. In *VRST*, volume 96, pages 101–108, 1996.

[37] O. Merhi, E. Faugloire, M. Flanagan, and T. A. Stoffregen. Motion sickness, console video games, and head-mounted displays. *Human Factors*, 49(5):920–934, 2007.

[38] Microsoft. Detail of light relfecting on lens of hololens. https://www.microsoft.com/en-ca/hololens.

[39] K. Moser, Y. Itoh, K. Oshima, J. E. Swan, G. Klinker, and C. Sandor. Subjective evaluation of a semi-automatic optical see-through head-mounted display calibration technique. *IEEE transactions on visualization and computer graphics*, 21(4):491–500, 2015.

[40] J. D. Moss and E. R. Muth. Characteristics of head-mounted displays and their effects on simulator sickness. *Human factors*, 53(3):308–319, 2011.

[41] M. A. Nacenta, M. Hancock, C. Gutwin, and S. Carpendale. The effects of changing projection geometry on perception of 3d objects on and around tabletops. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 23(2):11, 2016.

[42] K. Nakada, M. Nakamoto, Y. Sato, K. Konishi, M. Hashizume, and S. Tamura. A rapid method for magnetic tracker calibration using a magneto-optic hybrid tracker. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 285–293. Springer, 2003.

[43] D. Patrick. Pokmon go: Gym & raid update overview — everything we know (so far!). http://gameranx.com/features/id/110911/article/pokemon-go-gym-raid-update-overview-everything-we-know-so-far/, Jun 2017.

[44] R. Pausch, M. A. Shackelford, and D. Proffitt. A user study comparing head-mounted and stationary displays. In *Virtual Reality, 1993. Proceedings., IEEE 1993 Symposium on Research Frontiers in*, pages 41–45. IEEE, 1993.

[45] A. Plopski, Y. Itoh, C. Nitschke, K. Kiyokawa, G. Klinker, and H. Takemura. Corneal-imaging calibration for optical see-through head-mounted displays. *IEEE transactions on visualization and computer graphics*, 21(4):481–490, 2015.

[46] R. Raskar, J. Van Baar, P. Beardsley, T. Willwacher, S. Rao, and C. Forlines. ilamps: geometrically aware and self-configuring projectors. In *ACM SIGGRAPH 2006 Courses*, page 7. ACM, 2006.

[47] R. Raskar, G. Welch, K.-L. Low, and D. Bandyopadhyay. Shader lamps: Animating real objects with image-based illumination. In *Rendering Techniques 2001*, pages 89–102. Springer, 2001.

[48] B. Sajadi and A. Majumder. Automatic registration of multi-projector domes using a single uncalibrated camera. In *Computer Graphics Forum*, volume 30, pages 1161–1170. Wiley Online Library, 2011.

[49] S. Sharples, S. Cobb, A. Moody, and J. R. Wilson. Virtual reality induced symptoms and effects (vrise): Comparison of head mounted display (hmd), desktop and projection display systems. *Displays*, 29(2):58–69, 2008.

[50] I. Stavness, B. Lam, and S. Fels. pcubee: a perspective-corrected handheld cubic display. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1381–1390. ACM, 2010.

[51] R. J. Teather and W. Stuerzlinger. Pointing at 3d target projections with one-eyed and stereo cursors. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 159–168. ACM, 2013.

[52] A. Thomas. Perpective drawing - lessons - tes teach. https://www.tes.com/lessons/ygS22Q5pve4ESQ/perpective-drawing.

[53] M. Tuceryan and N. Navab. Single point active alignment method (spaam) for optical see-through hmd calibration for ar. In *Augmented Reality, 2000.(ISAR 2000). Proceedings. IEEE and ACM International Symposium on*, pages 149–158. IEEE, 2000.

[54] S. Umeyama. Least-squares estimation of transformation parameters between two point patterns. *IEEE Transactions on pattern analysis and machine intelligence*, 13(4):376–380, 1991.

[55] A. Vorozcovs, W. Stürzlinger, A. Hogue, and R. S. Allison. The hedgehog: a novel optical tracking method for spatially immersive displays. *Presence*, 15(1):108–121, 2006.

[56] A. J. Wagemakers, D. B. Fafard, and I. Stavness. Interactive visual calibration of volumetric head-tracked 3d displays. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, pages 3943–3953. ACM, 2017.

[57] C. Ware, K. Arthur, and K. S. Booth. Fish tank virtual reality. In *Proceedings of the INTERACT'93 and CHI'93 conference on Human factors in computing systems*, pages 37–42. ACM, 1993.

[58] K. D. Willis, T. Shiratori, and M. Mahler. Hideout: mobile projector interaction with tangible objects and surfaces. In *Proceedings of the 7th International Conference on Tangible, Embedded and Embodied Interaction*, pages 331–338. ACM, 2013.

[59] B. Wöldecke, D. Marinos, and C. Geiger. Flexible registration of multiple displays. In *Proceedings of The International Symposium on Pervasive Displays*, page 86. ACM, 2014.

[60] H.-K. Wu, S. W.-Y. Lee, H.-Y. Chang, and J.-C. Liang. Current status, opportunities and challenges of augmented reality in education. *Computers & Education*, 62:41–49, 2013.

[61] Z. Zhang. A flexible new technique for camera calibration. *IEEE Transactions on pattern analysis and machine intelligence*, 22(11):1330–1334, 2000.

[62] Q. Zhou, G. Miller, K. Wu, D. Correa, and S. Fels. Automatic calibration of a multiple-projector spherical fish tank vr display. In *Applications of Computer Vision (WACV), 2017 IEEE Winter Conference on*, pages 1072–1081. IEEE, 2017.

[63] Q. Zhou, G. Miller, K. Wu, I. Stavness, and S. Fels. Analysis and practical minimization of registration error in a spherical fish tank virtual reality system. In *Asian Conference on Computer Vision*, pages 519–534. Springer, 2016.

Appendix A

Consent Form

DEPARTMENT OF COMPUTER SCIENCE
UNIVERSITY OF SASKATCHEWAN
INFORMED CONSENT FORM

Research Project: **Visual Pattern Alignment**

Investigators: Dr. Carl Gutwin, Department of Computer Science (966-8646)
Dr. Ian Stavness, Department of Computer Science (966-7995)
Andrew Wagemakers, Department of Computer Science
Ashley Coveney, Department of Computer Science

This consent form, a copy of which has been given to you, is only part of the process of informed consent. It should give you the basic idea of what the research is about and what your participation will involve. If you would like more detail about something mentioned here, or information not included here, please ask. Please take the time to read this form carefully and to understand any accompanying information.
This study is concerned with detecting **the speed and accuracy of participants' while they align 3D visual patterns on a computer screen with a computer mouse.**

The goal of the research is to **examine the conditions that produce the fastest and most accurate visual alignment**.

The session will require **45** minutes, during which you will be asked **in a variety of conditions to move a computer mouse to align a pattern on an assortment of virtual 3D objects** in the Human-Computer Interaction Lab at the University of Saskatchewan.

At the end of the session, you will be given more information about the purpose and goals of the study, and there will be time for you to ask questions about the research. As a way of thanking you for your participation and to help compensate you for your time and any travel costs you may have incurred, you will receive a **$10** honorarium at the end of the session.

The data collected from this study will be used in articles for publication in journals and conference proceedings.

As one way of thanking you for your time, we will be pleased to make available to you a summary of the results of this study once they have been compiled (usually within two months). This summary will outline the research and discuss our findings and recommendations. This summary will be available on the HCI lab's website: **http://www.hci.usask.ca/**

All personal and identifying data will be kept confidential. Confidentiality will be preserved by using pseudonyms in any presentation of textual data in journals or at conferences. The informed consent form and all research data will be kept in a secure location under confidentiality in accordance with University policy for 5 years post publication. Do you have any questions about this aspect of the study?

You are free to withdraw from the study at any time without penalty and without losing any advertised benefits. Withdrawal from the study will not affect your academic status or your access to services at the university. If you withdraw, your data will be deleted from the study and destroyed. Your right to withdraw data from the study will apply until results have been disseminated, data has been pooled, etc. After this, it is possible that some form of research dissemination will have already occurred and it may not be possible to withdraw your data.

Your continued participation should be as informed as your initial consent, so you should feel free to ask for clarification or new information throughout your participation. If you have further questions concerning matters related to this research, please contact:

- Dr. Carl Gutwin, Professor, Dept. of Computer Science, (306) 966-8646, gutwin@cs.usask.ca

Your signature on this form indicates that you have understood to your satisfaction the information regarding participation in the research project and agree to participate as a participant. In no way does this waive your legal rights nor release the investigators, sponsors, or involved institutions from their legal and professional responsibilities. If you have further questions about this study or your rights as a participant, please contact:

- Dr. Carl Gutwin, Professor, Dept. of Computer Science, (306) 966-8646, gutwin@cs.usask.ca
- Research Ethics Office, University of Saskatchewan, (306) 966-2975 or toll free at 888-966-2975.

Participant's signature:_____

Date:_____

Investigator's signature:_____

Date:_____

A copy of this consent form has been given to you to keep for your records and reference. This research has the ethical approval of the Research Ethics Office at the University of Saskatchewan.

Appendix B

Viewpoint Calibration Calculations

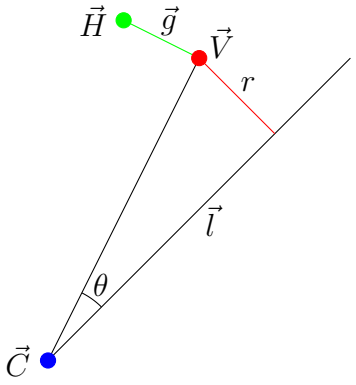


Figure B.1: Least squared method for finding the center of the display

$$\vec{V} = \vec{H} + \vec{g} \tag{B.1}$$

$$r = |(\vec{V} - \vec{C}) \times \vec{l}| \tag{B.2}$$

$$r = |(\vec{H} + \vec{g} - \vec{C}) \times \vec{l}| \tag{B.3}$$

$$r = |\vec{H} + \vec{g} - \vec{C}||\vec{l}|Sin(\theta) \tag{B.4}$$

$$r^2 = |\vec{H} + \vec{g} - \vec{C}|^2|\vec{l}|^2Sin^2(\theta) \tag{B.5}$$

$$r^2 = |\vec{H} + \vec{g} - \vec{C}|^2|\vec{l}|^2 - |\vec{H} + \vec{g} - \vec{C}|^2|\vec{l}|^2Cos^2(\theta) \tag{B.6}$$

$$r^2 = |\vec{H} + \vec{g} - \vec{C}|^2 - (\vec{H} \cdot \vec{l} + \vec{g} \cdot \vec{l} - \vec{C} \cdot \vec{l})^2 \tag{B.7}$$

$$\frac{\partial r^2}{\partial \vec{C}} = -2\frac{|\vec{H}+\vec{g}-\vec{C}|(\vec{H}+\vec{g}-\vec{C})}{|\vec{H}+\vec{g}-\vec{C}|} + 2(\vec{H}\cdot\vec{l}+\vec{g}\cdot\vec{l}-\vec{C}\cdot\vec{l})\vec{l} \tag{B.8}$$

$$\sum_{i=1}^{n}\frac{\partial r^2}{\partial \vec{C}} = 0 \tag{B.9}$$

$$0 = \sum_{i=1}^{n}(\vec{C}-\vec{H_i}-\vec{g}+\vec{l_i}(\vec{H_i}\cdot\vec{l_i}+\vec{g}\cdot\vec{l_i}-\vec{C}\cdot\vec{l_i})) \tag{B.10}$$

$$\mathbf{M} = \begin{bmatrix} \sum_{i=1}^{n}((l_i.x)^2-1) & \sum_{i=1}^{n}(l_i.x\cdot l_i.y) & \sum_{i=1}^{n}(l_i.x\cdot l_i.z) \\ \sum_{i=1}^{n}(l_i.y\cdot l_i.x) & \sum_{i=1}^{n}((l_i.y)^2-1) & \sum_{i=1}^{n}(l_i.y\cdot l_i.z) \\ \sum_{i=1}^{n}(l_i.z\cdot l_i.x) & \sum_{i=1}^{n}(l_i.z\cdot l_i.y) & \sum_{i=1}^{n}((l_i.z)^2-1) \end{bmatrix} \tag{B.11}$$

$$\vec{b} = \begin{bmatrix} \sum_{i=1}^{n}(-H_i.x-g.x+l_i.x(\vec{H_i}\cdot\vec{l_i}+\vec{g}\cdot\vec{l_i})) \\ \sum_{i=1}^{n}(-H_i.y-g.y+l_i.y(\vec{H_i}\cdot\vec{l_i}+\vec{g}\cdot\vec{l_i})) \\ \sum_{i=1}^{n}(-H_i.z-g.z+l_i.z(\vec{H_i}\cdot\vec{l_i}+\vec{g}\cdot\vec{l_i})) \end{bmatrix} \tag{B.12}$$

$$\mathbf{M}\vec{C} = \vec{b} \tag{B.13}$$

Appendix C

Study Shader

```
Shader "Custom/Spheree_shader" {
    Properties{
        _MainTex("Diffuse Texture", 2D) = "White" {}
    }
        SubShader{
            Pass{
                Tags{"LightMode" = "ForwardBase"}
                CGPROGRAM
                #include "UnityCG.cginc"
                #pragma vertex vert
                #pragma fragment frag
                uniform sampler2D _MainTex;
                uniform float4x4 view_VP;

                struct vertexInput {
                    float4 vertex : POSITION;
                };
                struct vertexOutput {
                    float4 pos : SV_POSITION;
                    float4 tex : TEXCOORD0;
                };

                vertexOutput vert(vertexInput v) {
                    vertexOutput o;
                    // Interpolate vertex position to fragment shader
                    o.tex = v.vertex;
                    o.pos = mul(UNITY_MATRIX_MVP, v.vertex);
                    return o;
                }

                float4 frag(vertexOutput i) : COLOR{
                    // Sample the texture based on where the fragment appears
                    // in the calibration camera's frustum
                    float4 temp = mul(view_VP, mul(unity_ObjectToWorld,i.tex));
                    temp = temp / temp.w;
                    float4 tex = tex2D(_MainTex, temp.xy + float2(0.5,0.5));
                    return float4(tex.xyz, 1.0);
                }
                ENDCG
            }
        FallBack "Diffuse"
    }
}
```

Appendix D

Questionnaire

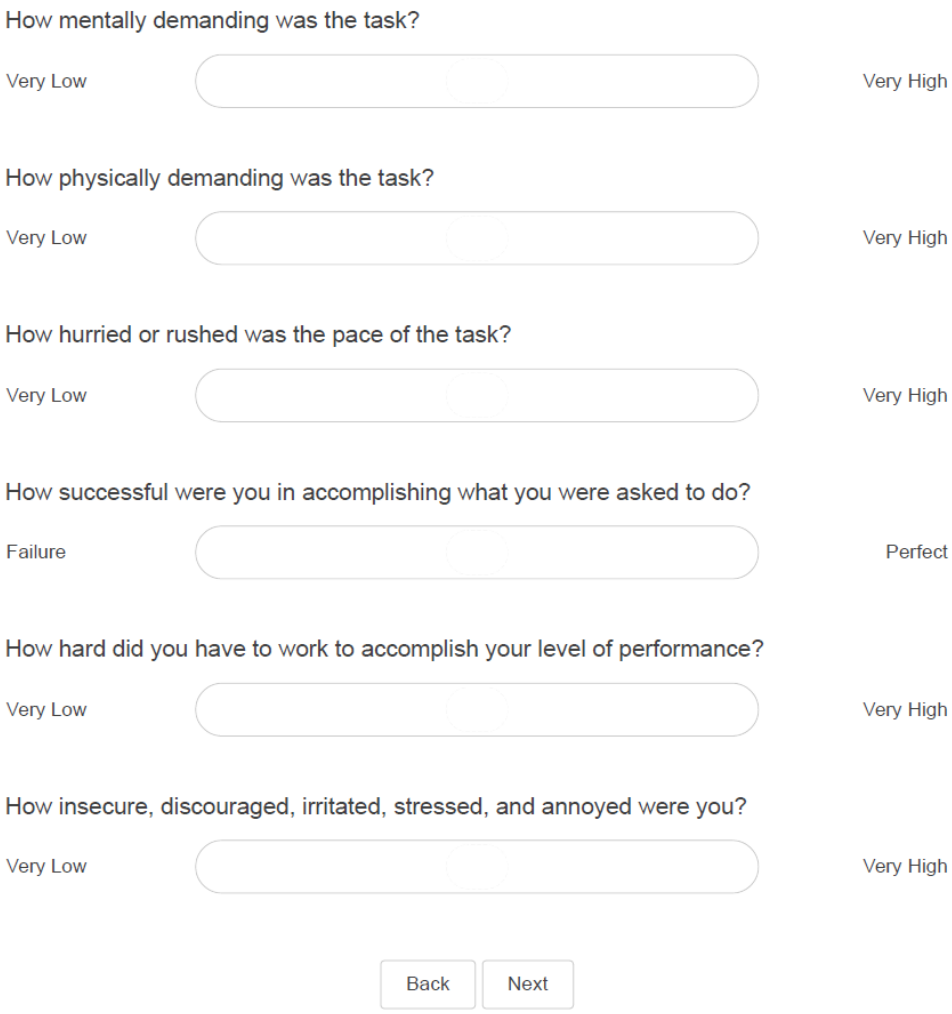


Figure D.1: End of Trial Questions

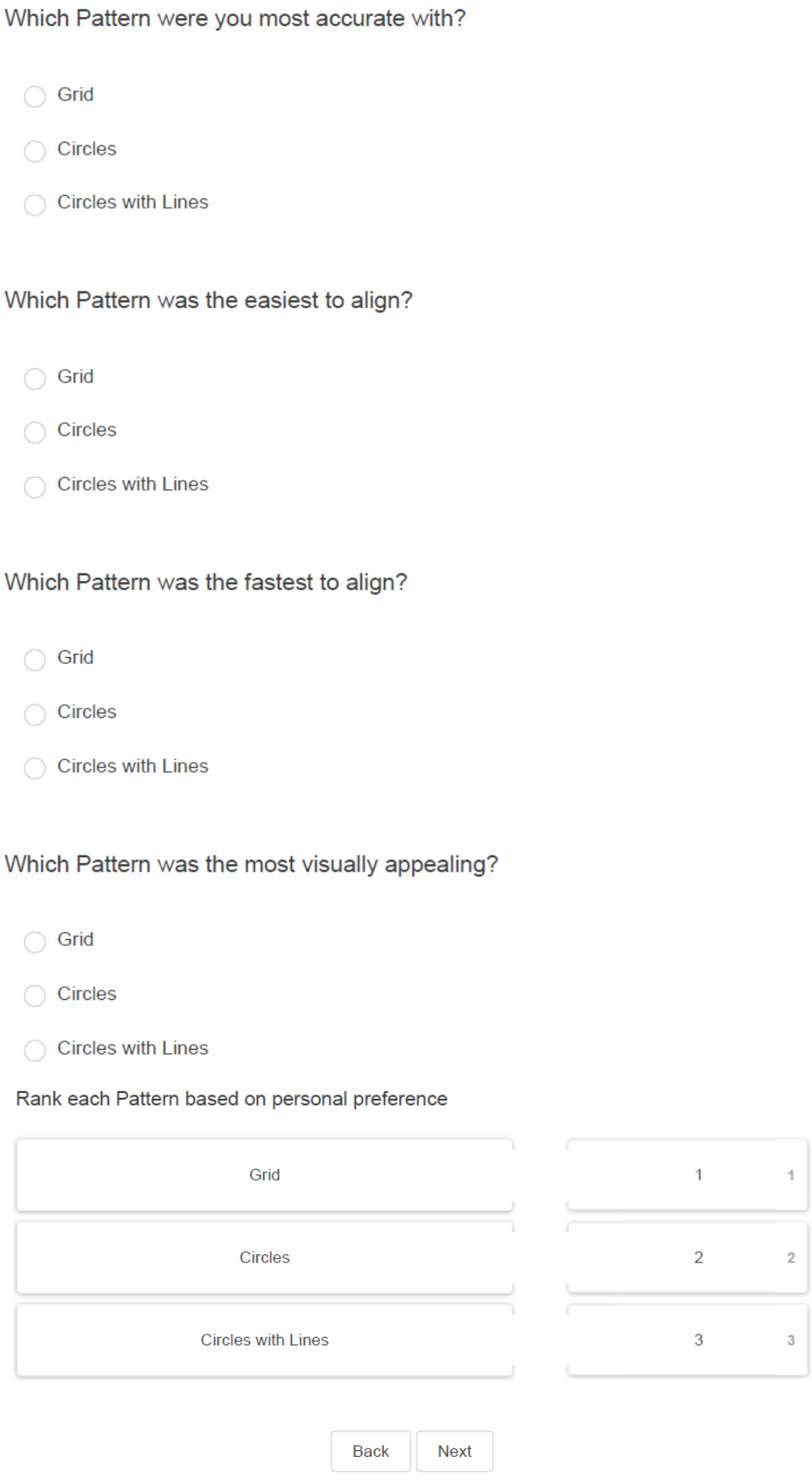


Figure D.2: Pattern Questions

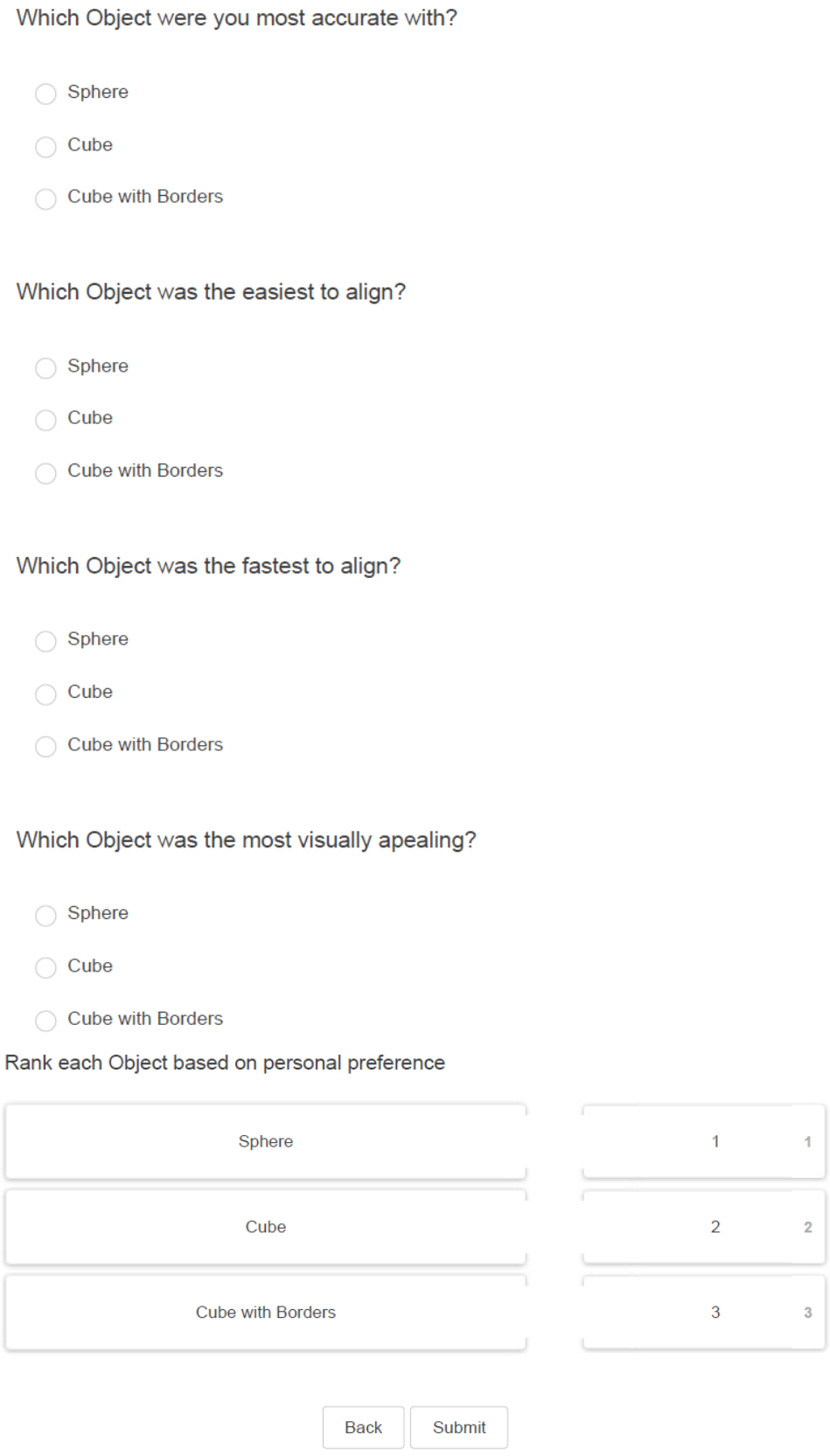


Figure D.3: Shape Questions