2010

Interactive natural user interfaces

Sean Patrick Janis

Interactive Natural User Interfaces

by

Sean Patrick Janis

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Science
in Computer Science

Supervised by

Assistant Professor Dr. Reynold Bailey
Department of Computer Science
B. Thomas Golisano College of Information Science
Rochester Institute of Technology
Rochester, New York
May 2010

Approved by:

Dr. Reynold Bailey, Assistant Professor
Thesis Advisor, Department of Computer Science

Dr. Hans-Peter Bischof, Associate Professor
Committee Member, Department of Computer Science

Dr. Joseph Geigel, Associate Professor
Committee Member, Department of Computer Science

Thesis Release Permission Form

Rochester Institute of Technology
B. Thomas Golisano College of Information Science

Title:

Interactive Natural User Interfaces

I, Sean Patrick Janis, hereby grant permission to the Wallace Memorial Library to reproduce my thesis in whole or part.

Sean Patrick Janis

Date

Dedication

This thesis is dedicated to my mother and father for their persistent love and support which continually enables me to follow my dreams.

"*If you can dream it, you can do it.*"
-Walt Disney

Acknowledgments

This Masters Thesis would have not been possible without the knowledge and support of my various advisors and colleagues.

First and foremost, I would like to thank my Masters Thesis advisors for guiding me through this incredible adventure. Thanks to Professor Bailey for allowing me to try something radically new and different; Professor Bischof for teaching me the art of giving a great presentation and challenging me to continually improve my work; Professor Geigel for making learning about Computer Animation, Computer Graphics and Virtual Reality the most fun, interesting experience I have ever studied.

Next, I would like to thank my professional colleagues who have provided countless support which lead to my project's success. Thanks to Jeff Hanzlik for teaching me the basics to soldering an infrared LED circuit; Kevin Peters, Mike Janis (my dad) and my Mike Janis, Jr. (my brother) for helping build a prototype fog screen water tank. Barry Nobles for helping develop concave mirror prototypes for my holographic display component; Joe Presicci for giving me advice about infrared tracking problems; Mike Doser for being a great mentor and inspiring me to explore various mad scientist projects; Adam Kent for teaching me about basic circuit design and components; Dave Garrison, Bob Post, Dave Garigen and Paul Voglewede for offering a lending ear for my various ideas.

Finally, in Winter 2007, I remember walking into RIT's Computer Science office looking to apply to graduate school. At the time, I was very excited to start the program and advance my skill set. On that day, I remember a very wise graduate advisor giving me the best academic advice I had received up to that point. "Most students that work full-time and attend graduate school part-time do very well taking one class per quarter. Take your time and enjoy the graduate program." Thank you again, Professor Bischof for the excellent advice.

Abstract

For many years, science fiction entertainment has showcased holographic technology and futuristic user interfaces that have stimulated the world's imagination. Movies such as *Star Wars* and *Minority Report* portray characters interacting with free-floating 3D displays and manipulating virtual objects as though they were tangible. While these futuristic concepts are intriguing, its difficult to locate a commercial, interactive holographic video solution in an everyday electronics store. As used in this work, it should be noted that the term holography refers to artificially created, free-floating objects whereas the traditional term refers to the recording and reconstruction of 3D image data from 2D mediums.

This research addresses the need for a feasible technological solution that allows users to work with projected, interactive and touch-sensitive 3D virtual environments. This research will aim to construct an interactive holographic user interface system by consolidating existing commodity hardware and interaction algorithms. In addition, this work studies the best design practices for human-centric factors related to 3D user interfaces.

The problem of 3D user interfaces has been well-researched. When portrayed in science fiction, futuristic user interfaces usually consist of a holographic display, interaction controls and feedback mechanisms. In reality, holographic displays are usually represented by volumetric or multi-parallax technology. In this work, a novel holographic display is presented which leverages a mini-projector to produce a free-floating image onto a fog-like surface. The holographic user interface system will consist of a display component: to project a free-floating image; a tracking component: to allow the user to interact with the 3D display via gestures; and a software component: which drives the complete hardware system.

After examining this research, readers will be well-informed on how to build an intuitive, eye-catching holographic user interface system for various application arenas.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Problem Introduction

1.1.1 Introduction

Every year, our living rooms become invaded with more and more cutting edge technology that facilitates our lives. Science fiction entertainment often portrays computing technology light years ahead of its actual capabilities. This research addresses the need for a feasible technological solution that allows users to work with projected, interactive and touch-sensitive 3D virtual environments. More specifically, this work explores why holographic user interfaces systems are not often readily available. We aim to research, design and build an interactive holographic user interface system that can be applicable to numerous graphics arenas. The end system will be evaluated by the degree to which it satisfies the below categories:

- *Hardware Availability* - The system shall make use of current state-of-the-art holographic displays, interaction tracking and touch feedback.
- *Believability* - The system shall take into account valuable human-computer interaction factors and be ergonomically friendly.
- *Practicality* - The system shall be applicable to one or more of the following areas including augmented reality, video games, advertisements, entertainment or futuristic user interfaces.

Holographic user interfaces provide a certain excitement to people. When first seeing a free-floating, 3D image, people are mystified by this futuristic technology. Not only can these futuristic interfaces astound people, but they also have very practical applications. There are numerous arenas where interactive holographic user interface systems can be applied such as augmented reality, video games, medical procedures and advertisements. Providing medical students with a 3D volumetric view of patient's brain can allow them to better understand details hidden in flat 2D images. Interactive holographic surfaces could allow video gamers to fight a projected 10-foot dragon in their living room without the need for a television. Bringing 360-degree product views to households could better educate online shoppers about merchandise features before final purchases. With futuristic user interfaces, the components to create an interactive holographic display exist and can be combined to develop an intuitive, eye-catching system.

This work's main objective is to educate the reader about interactive holographic user interfaces and also provide insight into how to build a feasible system from existing commodity hardware. We hope this work will excite readers about futuristic user interfaces and allow them to realize that this once advanced technology is more realizable than expected.

1.1.2 Background and Definitions

At a basic level, we need to create a solution that simulates a realistic environment. The realistic environment should exhibit native, intuitive sensory cues. As discussed by [17], physical movement is the strongest cue for distinguishing object depth. Hence, seeing an accurate projected image is critical to system believability. Moreover, intuitive hand gestures adapted from science fiction entertainment and control object placement should be considered. Most often, learning the user interface controls we observe in movies and television feel more natural because they are engrained in our semantic or context-based memory. Grasping or touching virtual system objects should provide similar reactionary, feedback traits as exhibited in real life. As detailed by [19], presenting users with a stimulator such as haptic feedback technology, depresses the user's focus from the holographic display. Basically, users become less consumed with believing the display's realism and

more consumed with the actual interface.

In this work, a holographic display is defined as a presentation surface that creates the illusion of free-floating 3D images. That is to say, given a graphics software package that renders perspective scenes, our holographic display will project a 3D scene representation in free space. Similar to a 2D display model, the holographic display surface will dynamically update as the graphics software data model changes. Previous authors such as [16] and [24], used enclosed volumetric, rotating mirrors for their holographic projections. Our holographic display will not contain any moving parts. These displays will be discussed in upcoming sections.

Interaction components are defined as the set of controls that allow users to interface with the holographic display. As the holographic display projects a free-floating image, users will use interaction components such as infrared tracking gloves and hand gestures to control objects within the the artificial 3D environment. Infrared tracking technology is often used for determining a user's location relative to the display. For example, Nintendo's Wii gaming system uses an infrared remote control and a reflective infrared light source to track a user's movement. Combining infrared tracking with custom gesture recognition, our holographic user interface can offer intuitive interactions such as pinching and hand sliding to manipulate system objects.

Feedback mechanisms are defined as visual, audio and touch sensation cues that are applied to the user as a result of interface interactions. Visual cues are the most basic resulting feedback mechanism and occur when the user augments an interface item. For example, if the user pushes a ball, the ball will move similar to real life. For an audio cue, the user may decide to bounce the ball with a simple hand gesture that causes an appropriate sound to play. Moreover, haptic technology is an important feedback mechanism because users need to believe the objects they touch are realistic. Haptic technology is defined as taking advantage of a user's sense of touch by applying forces, vibrations and motions upon the user. For a holographic user interface, developing a haptic component is the toughest system aspect as free-floating images are not tangible assets. In this work, we leave the haptic feedback component to our future work as it is outside the scope of our effort.

Finally, a software framework is defined as the application level source code which drives our system's hardware components. In our system, we will be creating a free-floating image illusion with which the user can interact. We will need to create an intuitive user interface which allows users to observe and evaluate the benefits of a holographic user interface; hence, we present Holovee, a simple social networking management system. Our social networking application will be based on Facebook's Application Programming Interface (API). Facebook is a popular online social networking website which connects millions of worldwide users. In our application, users will be able to view and manipulate their Facebook data with intuitive speech commands and hand gestures.

1.1.3 Description

With a basic description of system components, we can begin to visualize how our system will be constructed. Again, we are looking to build a holographic system that satisfies a growing need for more dynamic, visual user interfaces. As will be discussed, recent researchers such as [24] and [18] have made significant advances in display, interaction and feedback technology. This research will serve as a solid foundation for our work as each component will contribute to the larger system.

While the concept of interactive holographic user interfaces is very intriguing, even a highly futuristic system would be useless without any ergonomic considerations. There has been a plethora of work related to human-computer interactions with 3D user interfaces. Specifically, the authors in [6] have consolidated a rather extensive compilation of 3D user interface theory and practices. A distinguishing factor between our work and past work will be our emphasis and application of these best 3D user interface design practices into our system.

1.2 System Components

As discussed, our holographic user interface system will consist of a holographic display, user tracking, haptic feedback and software framework components. Figure 1.1 shows

how our system components interact to become operational. The following section briefly details the hardware used for each system component. Future sections will discuss each hardware component's technical specifications in more detail.

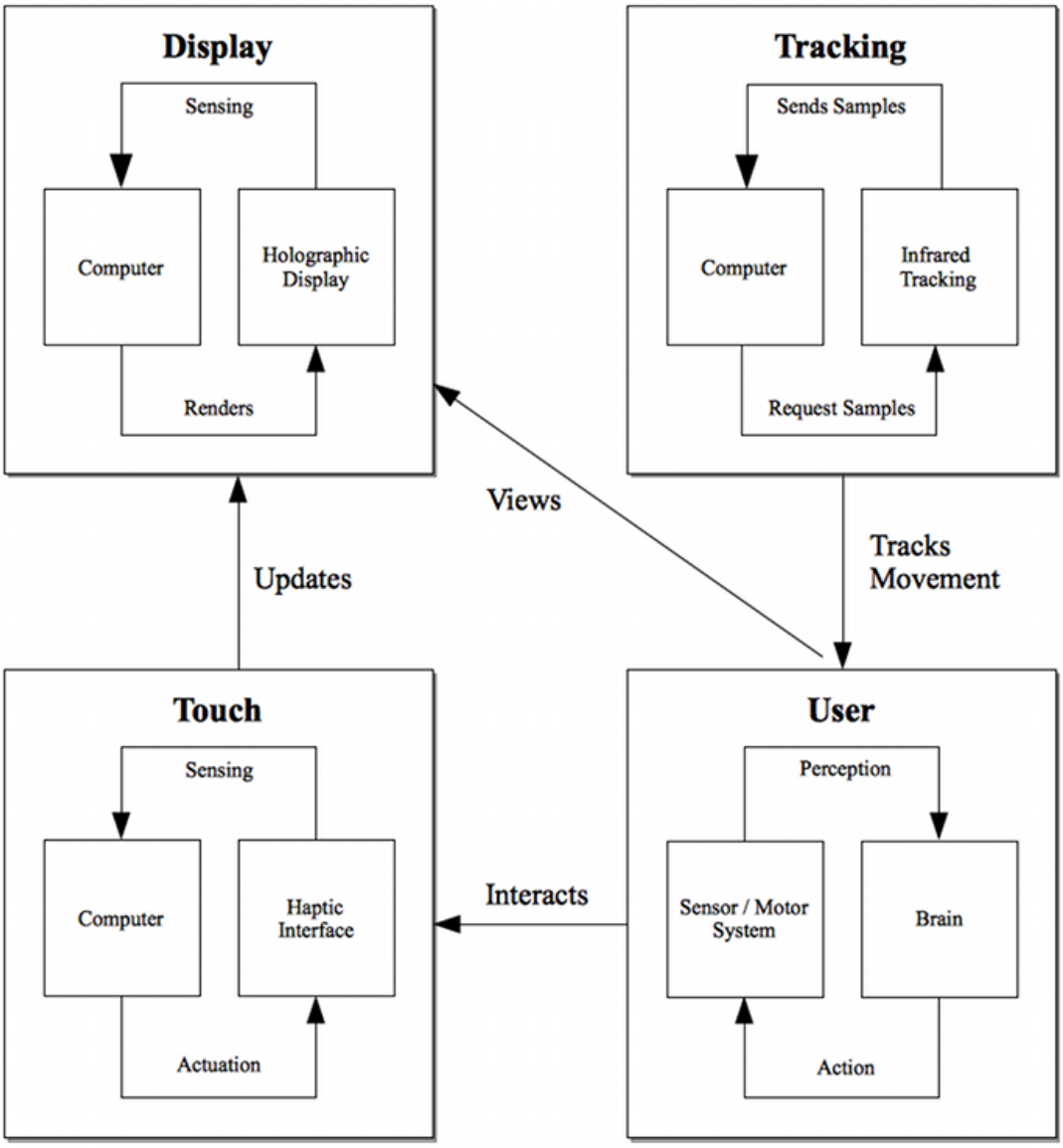

Figure 1.1: System Component Overview

1.2.1 Holographic Display

Our holographic display will use simple effects to create the illusion of free-floating images. These basic effects will be discussed in future sections, but are relatively easy to implement. The holographic display's hardware components include a 3M MPro120 micro projector, an ultrasonic water fogger and a commercial slim air fan to direct the produced fog. For our work, the fog-like mist will serve as a transparent display surface to project onto.

A projector is commonly used to project a digital signal from a personal computer or video source onto a reflective screen. Our system leverages a projector to project our computer's video output onto a custom display surface. Most often, static holographic display systems such as [48] and [53] use custom mirror arrangements to reflect and manipulate light in desired directions. In addition, the creators of [53] use a high intensity LCD screen to reflect into their lens system. We chose 3M's MPro120 micro projector because of its size and commercial availability. Instead of using an LCD screen and placing our system computer directly into our lens system, using an external projector allows easy access to the system computer and is flexible enough for larger scale setups. When operational, the slim air fan will vacuum the fog and direct it vertically to produce a steady, transparent fog screen.

Because our system uses a transparent water fog screen as its projection surface, room lighting and a water source are two important setup considerations. Our holographic display creates the illusion of a free-floating image without the need for complex lenses and mirrors. As exhibited in [53], concave mirror holographic display setups often require users to be at specific viewing locations. Because our system is not using a lens-mirror setup, we can be less concerned with problematic user viewing angle issues. Figure 1.2 shows a basic picture of our system's design.

1.2.2 User Tracking and Control

Infrared cameras are often used to track infrared light emitting diodes. Invisible to the human eye, infrared light receivers are common in remote controls and home entertainment systems. For our system, we require a tracking mechanism that allows users to seamlessly

Figure 1.2: Transparent Fog Screen

interact with our software interface. With this in mind, we use the Nintendo Wiimote as our infrared tracking mechanism because of its widespread availability, low cost and software development kit. The Nintendo Wiimote is a hardware device that uses an integrated infrared camera to track a stationary infrared emitting light sensor bar. The controller device pairs with Nintendo Wii video game consoles and allows players to interact with video games.

Our system inverts the standard use for a Nintendo Wiimote. Instead of a moving Wiimote tracking a stationary sensor bar, we keep the controller stationary and track moving infrared lights mounted to the user's hands. This method is simpler and less bulky than mounting a full-featured infrared camera onto our users' hands. To give granular control of our holographic interface, we present interactive, user wearable gloves equipped with infrared lights. The interactive gloves are inspired by Wiimote home brew project work

by university researcher Johnny Lee [29]. The infrared lights are mounted to the wearable gloves and are powered by 3-volt watch batteries. When powered on, the diodes emit an infrared light which is invisible to the human eye, yet recognizable by the Nintendo Wiimote. For better tracking performance, we cluster multiple infrared LEDs together. The infrared tracking gloves are pictured in Figure 1.3.

Figure 1.3: Tracking Gloves equipped with Infrared LED diodes

1.2.3 Software Framework

Our software framework is the driving force behind our system's operation. The software framework was written in the C# programming language and uses Windows Presentation Foundation (WPF) as its graphical subsystem to interact with our computer's rendering hardware. We use Microsoft's Visual Studio 2008 as our development and debugging environment. We chose these development tools because of our past experience with the

technologies and because they allow us to rapidly prototype high quality graphical Windows applications. Moreover, the WPF application programming interface allows developers to easily manipulate graphical objects and has an abundance of good coding samples. Having great programming resources allow developers to work both efficiently and reduce unnecessary code. Figure 1.4 shows a screenshot of our social networking application.

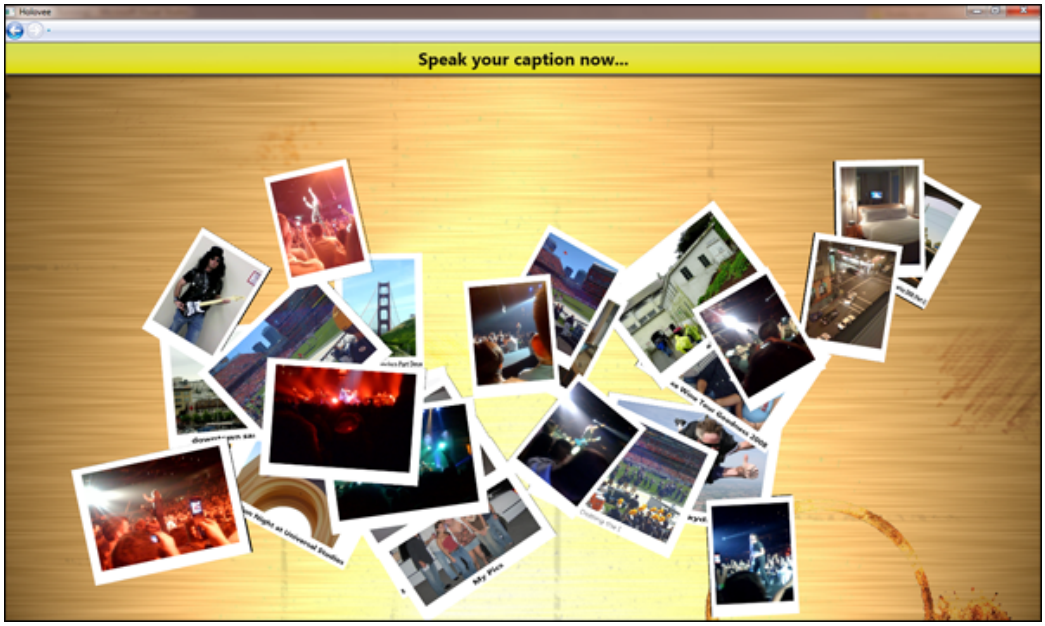

Figure 1.4: Holovee - Social Networking Management Application

As aforementioned, our system's main graphical component will be Holovee, the social networking management application. Holovee presents users with their uploaded Facebook photos, friend information and status data. We present the user with speech commands to easily manipulate their data. The software framework also encapsulates a tracking component which manages the computer's interactions with the Nintendo Wiimote. During runtime, the software framework will read the Wiimote's infrared tracking samples and translate them to user interface actions. For example, if the tracking component detects two points moving closer together, it may consider this a user selection or pinching indication. More detail will be provided in future sections about the available custom gestures and actions to control our social networking application.

Chapter 2

Related Work

2.1 Musion Eyeliner Holographic Projection System

In the search for futuristic display technology, Dimensional Studios Ltd., a London-based company, developed Musion Eyeliner, a commercially available 3D holographic video projection system [32]. At a basic level, the Musion Eyeliner system consists of an overhead projector, a reflective screen and custom transparent foil. When running, the overhead projector projects a high definition video feed onto the reflective display screen which is positioned on the ground floor. The display screen then reflects the image onto a 45-degree angled piece of transparent foil. From an audience's perspective, the reflected image on the transparent foil creates the illusion of artificial images being dynamically inserted into a real world scene.

The Musion Eyeliner system's artificial illusion is often known as Pepper's Ghost effect. In 1862, John Henry Pepper, a university lecturer, created an illusion which provided audiences with phantomly appearing and disappearing images [17]. Since the 19th century was not plentiful with electronic projectors and reflective screens, Pepper used an angled sheet of glass to reflect a hidden stage. Images were made visible by lighting the hidden stage, thereby reflecting the hidden stage object through the glass and producing a free-floating image illusion.

We experimented with the Pepper's Ghost phenomenon and felt it wasn't practical for our application. Because the Musion Eyeliner system and Pepper's Ghost effect are not volumetric, users will not be able to view our holographic display from multiple angles. In

addition, Musion Eyeliner's system works very well because audiences are usually positioned farther from the display unit. Hence, the effect is more believable and the user is not concerned with interacting with the display.

2.2 Dreamoc 3D Holographic Display

Similar to the Musion Eyeliner Holographic Projection System, RealFiction, a Copenhagen-based company, has developed Dreamoc, a 180-degree virtual projection system [48]. Although unpublished, the system appears to use a top-down version of Pepper's Ghost effect to project images onto a transparent surface which creates a free-floating illusion. Whereas the Musion Eyeliner Holographic Display system's reflected image is projected upward, the Dreamoc system's reflected image is projected downward; hence, creating a reverse volumetric pyramid sensation. When the Dreamoc system creates a free-floating image, users are able to place real objects into the system's projection space to simulate visually stunning, complementary effects. The Dreamoc holographic display's base dimensions are 45-inches wide by 24-inches high.

The Dreamoc's semi-enclosed surface design lends itself well to our application. By creating an enclosure which has a top cover and back surface, we can better integrate our micro projector and infrared tracking mechanism and obstruct them from the user's view. By doing so, the user will feel more engrained into the system's experience. However, because the Dreamoc system uses an enclosed surface, our projection space is now limited to the structure's volume. Therefore, we will experiment with both enclosed and non-enclosed holographic display designs to determine which proves most beneficial.

2.3 Provision 3D Media

Provision Interactive Technologies, Inc., an interactive display company, offers many varieties of 3D holographic displays which provide the illusion of free-floating images [53]. Targeted more towards in-store media and advertisements, Provision provides next-generation technology which has excited their customers. While very interesting, Provision's provides

few details as to their 3D displays inner workings. A further search for their display's technical specifications can lead to Provision's 3D 2009 patent award. Provision's 3D technology consists of a high intensity LCD screen, a concave mirror and an angled sheet of transparent material inside an enclosed display unit. When operational, the LCD screen reflects into the concave mirror and then back into the angled sheet of transparent material to produce a free-floating image 12-inches from the display.

Provision's 3D display technology uses a very simplistic design to produce an amazing 3D illusion effect. Moreover, Provision's 3D display technology has been used previously for interactive systems. In 2009, the authors in [19] used Provision's 3D display to create an interactive system integrated with an infrared tracking component and haptic feedback mechanism. Overall, Provision's enclosed display proved sufficient and provided the necessary free-floating simulation to the researchers' needs. Applicable to our system, Provision's display design appears semi-viable for creating an off-the-shelf holographic display while producing favorable results. However, obtaining a perfectly shaped, commodity concave mirror to produce a holographic effect was too difficult for our project application.

2.4 Heliodisplay

Invented by Chad Dyner in his apartment, Heliodisplay uses standard commercial projectors to display into a steady, compressed air stream that acts as a transparent projection screen [22]. Because it uses a rear projection design, Heliodisplay users often stand at off-center angles relative to the display to avoid being blinded by the projector. In addition, a main benefit of Heliodisplay is its portability and setup time. The system is capable of producing between 55-inch to 92-inch diagonal images and does not require any additional mirrors or angled sheets of transparent material.

While highly desirable, commercial versions of Heliodisplay are very expensive and range from $18,000 to $65,000. Most often, Heliodisplay display units require darker lit room settings to show higher contrast images. This dark lit constraint not only exists within Heliodisplay, but also, within other systems and is a limitation of most projector-based display approaches. Similarly, because our system uses a small projector to reduce system

hardware size, we sacrifice brightness as most pocket projects lack the intensity exhibited in standard-sized classroom setups.

For our work, we found the aforementioned angled transparent material illusions are not tailored towards interactive user interfaces. They often require specific viewpoints and a solid understanding of optics systems. Moreover, similar to Heliodisplay, we chose to engineer a low-cost mist generator to serve as our system's transparent projection surface. Using commodity components and a do-it-yourself attitude, we can achieve a similar transparent fog screen without the need for an expensive futuristic display.

2.5 FogScreen

Similar to the Heliodisplay, Finnish-based company FogScreen, Inc. has developed a transparent water-driven fog screen which can be used in various application arenas [15]. The patented FogScreen technology is usually ceiling-mounted and delivers a steady, vertical fog stream downward to create a transparent projection screen. A rear-projector light source is positioned behind the fog screen and projects an image onto the surface. To casual observers, FogScreens can be walked through and easily touched. Since the system creates a very granular water fog, it does not create a smoke-filled room as exhibited in traditional Halloween fogger machines. In general, water vapor evaporates very quickly and makes its FogScreen application very appealing. FogScreen, Inc. offers several projection variations which range from 3 to 8 foot display screen width.

Moreover, the authors in [45], [46] and [47] have used FogScreen technology to create highly exciting and interactive environments. The researchers in [47] found that designing suitable content for the FogScreen is a very important concept. Developers must engage their audience with useful content to allow them to see the full benefits of a futuristic display. In particular, they created virtual brick walls and fire screen advertisements for users to walk through. Doing so, users would be left with memorable experiences and become more engaged with the advertised products.

Our work is heavily based on FogScreen's transparent surface technology. Although FogScreen does not disclose their fog generation technology, we can hypothesize that it

incorporates an ultrasonic water fogger because the system is water-based. Ultrasonic water foggers use high energy vibrations to turn water into consistently-sized vapor particles [9]. We aim to produce a quickly evaporating water vapor which can be directed into a vertical air flow. We use a single, commercially available slim fan which directs the generated fog stream into mid-air. Our holographic display's size is limited to the slim air fan's output vent which is 20.5 inches wide. In addition, our display is easily portable because of its small size and light weight. Again, we are striving to create the *illusion* of free-floating images via this transparent fog screen. True holographic displays may someday work without the need for fog screens, but we use a reasonable alternative to create a similar futuristic effect.

2.6 Volumetric Finger Tracking

In 2004, the authors in [16] described their design and implementation of an interactive user interface for controlling a 3D volumetric display. The system consists of an enclosed holographic volumetric display, infrared finger tracking system and software to run an interactive application. The holographic display, developed by Actuality Systems, uses a swept volume to spin a 2D time-varying image about an axis to produce a perceived volumetric 3D image [16]. Since this display was volumetric, movement around the display allows users to see the projected image from multiple perspectives. An infrared camera system surrounding the display then tracks the user's fingers which are equipped with reflective infrared markers. As the user interacts with the display, objects transform in real-time relative to their interactions. Finally, the authors' software provided users with an intuitive, gestural-based interface to simulate a realistic environment.

To supplement their system, the authors created a geometric building application which allowed users to construct complex shapes from simple objects. While experimenting with interaction techniques, the authors found various best practices for building a multi-finger gestural interaction system. Via touch interactions and hand gestures, users can then manipulate system objects by various transformations. The authors intuitive interaction techniques such as pinching to translate an object or using two finger movements to scale an

object can be applicable to our system. Providing caption bubbles and object highlighting are also subtle yet effective techniques the authors used in their solution to make their user interface more understandable.

Unlike their system's holography display, our system will use a non-moving, novel approach to generate a free-floating image. Since our system will not feature moving parts, our users will be able to more granularly interact with the holographic image. Moreover, the authors' system uses external infrared tracking cameras, developed by Vicon, to track user hand gestures. While this tracking system proved sufficient and provided little latency, it was rather costly and must be aligned around the display surface every time the display is moved. In our work, we aim to provide a more integrated infrared tracking system that is built into the display surface. Overall, this system provides an excellent foundation for research into the best 3D holographic user interface techniques.

2.7 Wiimote Infrared Finger Tracking

In 2008, Johnny Lee, a Carnegie Mellon University computer scientist, produced a learning website dedicated to developing intuitive projects for the Nintendo Wii Remote [29]. The Nintendo Wiimote, is the main controller for the Nintendo Wii video gaming system. Operationally, the Wiimote contains a built-in infrared camera which is only sensitive to infrared light. While playing Wii video games, the Wiimote's infrared camera tracks an infrared light emitting sensor bar to compute the user's controller position and movement. In addition, the Wiimote encompasses a 3-axis accelerometer to track controller orientation and real-time rotations.

Focusing on the Wiimote's infrared tracking capability, Lee created a project which shows how we can track a user's fingers via a simple mechanism. For his demo, Lee aims a Wiimote and infrared light array at the user who wears reflective tape on their fingers. When operational, the infrared light array emits infrared light, bounces off the user's reflective tape-equipped fingers and is tracked by the Wiimote. From Lee's video documentation, the Wiimote tracks the user's fingers rather well and has minimal setup time.

For our system, we build a variation of Lee's Wiimote project that incorporates infrared

finger tracking to interact with our holographic display. Moreover, our system employs a user-worn glove, complete with infrared lights attached to each glove's index finger for consistent tracking by the Wiimote controller. For example, when a user wishes to select a holographic menu item or interact with a free floating object, they will make a finger movement which is tracked by the Wiimote, then is transformed against the selected scene element. Overall, our system's tracking is a superset of Lee's initial project and aims to provide the user with more control over interface objects.

2.8 Ultrasound Radiation Technology

The authors in [23] detail a new tactile device that uses airborne ultrasound transducers to produce vibration feedback. Presented at SIGGRAPPH 2008, the ultrasound device was considered a breakthrough in haptic feedback technology. Unlike past approaches which attached tactile devices to users' fingers, the authors' creation allows users to feel acoustic pressure without wearing additional equipment. The device works by aligning ultrasound transducers in a square arrangement and having them render a pressure pattern in free space. As users interact with the system, they break the transducers' pressure field which causes users to feel feedback on their fingers.

While highly attractive, the authors' airborne ultrasound feedback system is not commercially available and requires expensive transducer equipment [23]. Hence, building a complete transducer array for our system is not practical. However, since our system users will be wearing infrared tracking gloves, we could embed haptic vibration sensors near their fingers. These haptic finger sensors could activate when users near virtual system objects and produce vibration feedback for effective touch simulations. Again, haptic feedback technology is outside this project's scope and we will leave it to the future work section.

Chapter 3

Interactive 3D User Interfaces

3.1 Need for 3D User Interfaces

3.1.1 Technological Advancements

Before further moving into the system's design, it's useful to understand this problem's importance and have a solid foundation to plan our holographic user interface. Today, there are various traditional 2D user interface techniques related to productivity which satisfy a plethora of our everyday computing needs. With this in mind, some may question the practical need for an interactive holographic user interface. Similarly, we can explore the question of what defines a user interface and why they are useful. We develop user interfaces because people need a medium to interact naturally with computers. These user interface components translate our input feedback into binary instructions which the computer can recognize.

Early virtual reality systems such as [40] only offered users simple options to navigate menus and select actions. While this 2D paradigm allowed users to complete virtual reality tasks, it was not the most natural mechanism for interacting with a 3D environment. Rather than relying on traditional techniques, these early virtual reality systems could have been improved by focusing on real world interaction paradigms. Nowadays, we are seeing more virtual reality systems developed with composite interaction techniques such as grasping objects and intuitive hand gestures [51].

As computing hardware continues to improve, it is the software developer's responsibility to take advantage of its' new capabilities. We someday aim to effectively model a

computer interface comparable to natural real world interactions; hence, it's important to continue innovating and going beyond what is feasible. As IDEO general manager Tom Kelley discusses in [25], if Henry Ford would have given customers what they wanted, they would have said a faster horse. If we only translate traditional 2D interface techniques to 3D spaces, we will have missed the opportunity to take full advantage of the technology. We need to strive to throughly analyze a 3D application's end goals and only then will realize the appropriate user interface.

3.1.2 Comparing Traditional Users Interfaces

Working in an engineering environment, developers can sit at their computers for endless hours. Not because of an impromptu patience, but because their interactions are minimal and not overly exerting. Although Window, Icon, Mouse, Pointing Device (WIMP) interfaces seem outdated, they are successful because people can work fluidly without realizing they're using an input device. Their input device is an extension of their natural movements.

Most modern software applications use some WIMP interface variation to allow users to increase productivity and organize their data. While WIMP interfaces are effective, they limit user degrees of freedom (DOF). An interface's DOF is defined by the number of independent displacements such as rotations or translations a user is able to make with respect to the system. On standard computers, most users are only concerned with 2-DOF as that's all they need. Interactive 3D user interfaces often provide users with 6-DOF to rotate, translate and select objects in three-dimensional space.

Defining human-computer interaction tasks for a 3D user interface is a complex process. Surely, we can apply everything we have learned from 2D human-computer interaction studies to the 3D realm, but, greater thought needs to go into developing 3D environments. In a traditional 2D space, tasks such as selection and dragging are simplified as users navigate their mouse to precise screen coordinates. Conversely, 3D tasks go beyond traditional input hardware and control. Not so much to overburden the user, but to provide them with more tools to better navigate their virtual world. Thinking about a previous example, it's difficult to imagine a medical student interacting with a 3D volumetric brain image with

only a 2D mouse. The volumetric image can be better dissected with 3D controls which can help the user accomplish more complex tasks.

3.1.3 Practicality

Holographic user interfaces are only as good as the applications where they are applied. Even the most eye-catching holographic user interface could be deemed useless if it does not serve any practical purpose. To be practical, a holographic system must do more than simply allowing users to accomplish user interface tasks; it must also enrich the user's experience beyond what is imaginable. Before system architects decide whether to integrate a holographic user interface into their design, we present the following questions to consider:

- *User Experience* - Will the user's end experience be significantly enhanced by integrating a holographic user interface into the system?
- *Task Completion* - Will the holographic user interface allow users to complete tasks that cannot be completed by a traditional 2D user interface?
- *Viewing Constraints* - Does the holographic display require users to wear special glasses or be at a specific viewing angle? In addition, does the end system require certain lighting conditions which may restrict the use of a holographic display?
- *Tracking Granularity* - Will the user require granular controls of user interface components? For example, a building sculpting application may require finer controls than a building layout application.
- *Ergonomics* - Will the holographic system's input component be more ergonomically friendly and easy-to-use than a traditional keyboard and mouse?

When working with 2D displays, users are often limited to single n-inch-sized displays. In his 2006 time management lecture, computer scientist Randy Pausch characterized working on one monitor to that of working on airplane tray. Pausch stressed the need for double and triple display setups to maximize productivity. Most video games and animations have the necessary data to build a three-dimensional world displays, but instead

are projected onto 2D displays, thereby losing their true volumetric representation. Envision a futuristic holographic display that is able to project data into midair at an arbitrary size rather than be constrained to a fixed-sized monitor. Users would not have to sacrifice physical space for screen real estate. But rather, users could dynamically adjust their holographic display system to project larger or smaller images. This knowledge brings the topic of practicality to fruition.

3.2 Human-Centered Design

3.2.1 Basic Overview

As mentioned in the Introduction, our research not only aims to build a holographic user interface system, but to educate the reader about good user interface design practices. Even if our end system cannot apply all discussed design practices, it should serve as a fruitful survey for readers. Human-Centered Design (HCD) is a term coined by IDEO, a world-leading design and innovation firm. More recently, IDEO employees authored [20], which is a multi-applicable toolkit for creating new world solutions to challenging problems on a limited budget. The HCD toolkit focuses on human behavior and understanding how to maximize innovation.

At a basic level, each HCD phase involves viewing projects through a series of lenses as seen in Figure 3.1. Desirability, the first lens, strives to listen and learn to what is important to users. As stated by the IDEO authors, the first HCD phase is most important and aims to inspire developers about their users' needs. Feasibility, the second HCD lens, determines how much of the users' needs can be successfully obtained. Most often, users may dream beyond what is imaginable and envision their system to be light years ahead of current technology. Finally, viability, the third IDEO design lens, ensure system designers think about their financial constraints. These three design lens are far from serial and can continually be consulted as the design process develops.

With these lens in mind, we suggest allowing users to continue imagining above and beyond practical technical solutions. Most often, it is the wildest and exciting ideas that

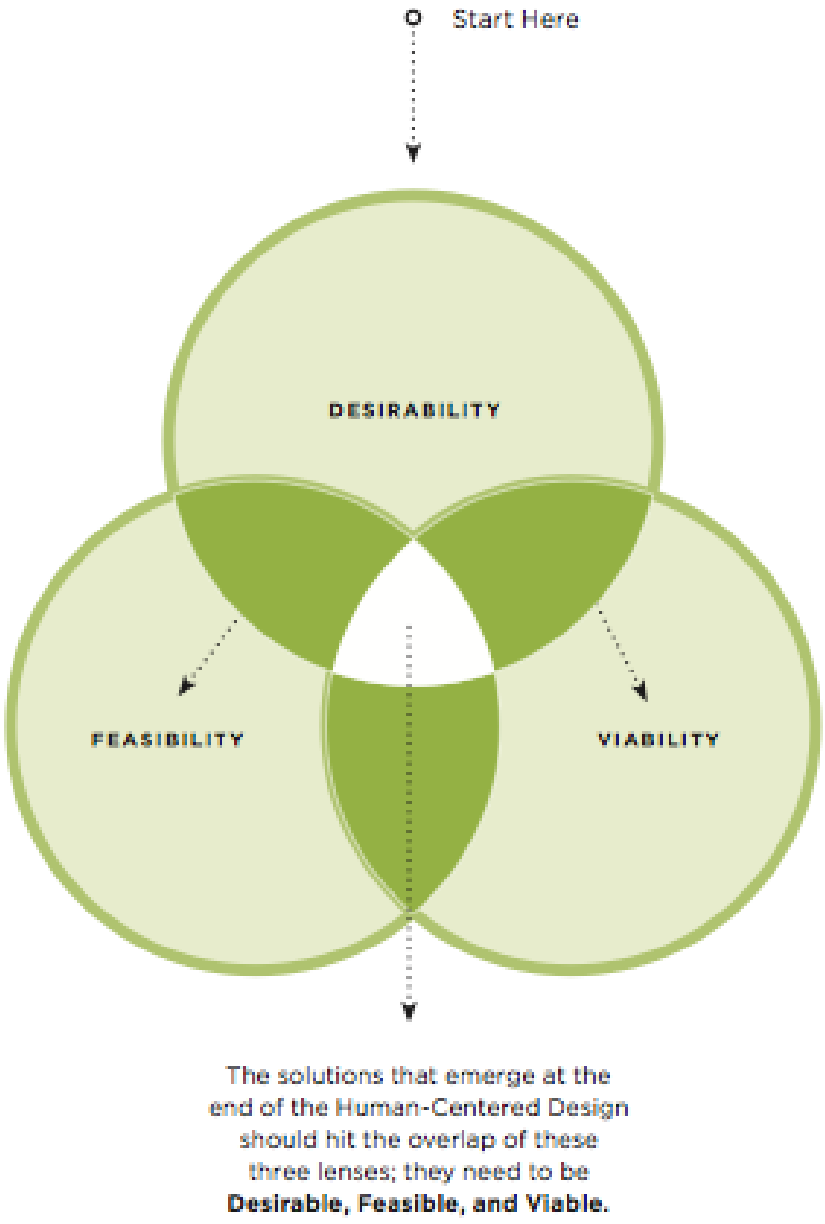

Figure 3.1: IDEO's Human-Center Designed Process viewed through a series lenses [20].

produce out-of-the-box thinking and new business models as proved in [10]. However, if the futuristic technology isn't available to fully satisfy user needs, we suggest developing a solution which comes close. Aligned with our work's paradigm, we understand *true*

holographic user interfaces from science fiction movies like *Avatar* and *Serenity* may be years away, but we can still present the *illusion* of free-floating images using projector and fog technology.

Like most development processes, the IDEO process is highly iterative. At each iteration, the system becomes more and more refined with higher quality prototypes and better overall functionality. In our experience, an iterative design process proves very successful for systems that have fuzzy design requirements and are better suited to be clarified via solid software cycles.

3.2.2 Application to our System

Because our interface is not being designed inline with users, we must make assumptions about which best design practices best fit our end system. While interesting, a lot of HCD material related to innovative development can be overlooked as we are not working in large team environment. We began to examine the problem space and determine which features were important. Moreover, thinking about the HCD toolkit, we spoke with whoever would listen and gathered various opinions about our holographic system. Then, we began experimenting with the plethora of ideas and determined workable methods. For example, it would have been premature to simply assume Johnny Lee's work in [29] functioned perfectly. When designing an infrared tracking system, we even experimented with infrared illuminators reflecting off user worn retro-reflective tape to see if it could be a viable solution.

As mentioned, we plan to use IDEO's iterative approach for delivering project components. First, we choose to architect the complete social networking management application. Knowing which user interface elements comprise our application will allow us to design around it. It's more viable to actually see the software onto which we will be creating a futuristic interface. Our second and concurrent iteration will focus on the infrared tracking component. Since this component adds functionality to the software framework and is independent of the holography display, it can be tested on a standard computer screen. Finally, our holography display can be brought into the system last as it is considered an

auxiliary display mechanism.

Different from IDEO's methods, our interviews with potential users were very informal. We spoke with as many people as possible and shared our ideas to gain feedback. Casual conversations about our holographic display surface or effective mechanisms for wiring circuits helped our development process. When designing a new system with many unknowns, the takeaway lesson is to try many things early and try many things often. These brainstorming sessions are not meant for others to assume devil's advocate roles, but in our situation, rather contribute tenfold to our system's success.

3.3 Crafting a Good User Interface

3.3.1 Choosing the Right Equipment

As aforementioned, the authors in [6] provide a comprehensive collaboration of 3D user interface design and speech components. The authors make a compelling argument as to the evolution of 3D user interfaces. In their opinion, it is the futuristic advances in 3D-related hardware that continually drives the software industry to develop better user interfaces. An interesting paradigm about current 3D user interface research is it's unstructured nature. Most often, 3D system developers creatively architect solutions that best suit specific interactive applications and are not constrained by rigid guidelines. Giving users increased user interface freedom allows them to tailor the interface to their liking and not vice versa.

For a starting point, system designers need to consider the four main components of a holographic user interface system. As seen in Figure 1.1, a standard system consists of display, tracking, touch and human components. When operational, these user interface components interact to produce a consolidated system which complement one another. Moreover, the authors in [6] provide a detailed pros and cons list for each hardware choice. From this list, we present a quick-start guide for our system design needs as seen in Table 3.1. This list should serve as a starting point for aspiring system developers who aim to create futuristic user interface hardware systems.

Component	Needs
Holographic Display	Small field-of-view (FOV), Inexpensive to build, Requires minimal physical space, No need for perspective viewing support, Relatively bright, Easily portable, No need for special glasses
Tracking	Lightweight, Inexpensive to build, Large field-of-view infrared LEDs to maximize tracking, Easily mountable to gloves
Haptic Feedback (Future Work)	Lightweight, Inexpensive to build, Easily integrates with tracking gloves, Simple API for issuing feedback

Table 3.1: Hardware Component Needs

3.3.2 Developing Appropriate Interactions

Defining appropriate user interactions rely heavily on how users need to operate the system. For example, try envisioning a physical task where a person picks up and throws a ball. Now, try envisioning how this physical task can be translated to a keyboard and a mouse. The user may hover over the virtual ball object and click the mouse to select the ball. After the ball is selected, the user may give a swift mouse movement to trigger a ball throw. In addition, the user may alter their ball's trajectory by pressing a keystroke combination. Limited to only their current controls, the user may quickly learn the ball grasping and throwing procedure. However, this procedure forces users to learn a complex chain of actions to accomplish a simple task; hence, it lacks intuition, a common human-centered design practice.

For our system, consider a keyboard-less, mouse-less environment where the user must be able to quickly learn interaction skills. Because we have access to user's hands for input, we can fully utilize their movement range. Thinking about the previous example,

instead of using a mouse to select the ball, we can allow users to pinch two fingers to grasp an object. Then, similar to a real-life movement, the user can flick their hand and throw the ball. While the flicking motion relates to the traditional 2D mouse method, it is more intuitive because the users' hands are now their direct input device.

With these examples in mind, let's think about general mechanisms for developing appropriate interactions for a holographic user interface. Rather than focusing on how the user should accomplish a task, we should focus on what the user wants to do. If successful, a holographic user will allow users to focus on their tasks rather than their performed actions. We will discuss the precise user interface interactions in the upcoming section about infrared tracking.

3.3.3 Presentation

When designing a futuristic user interface, presentation is the final component to consider. Presentation defines the key traits which define our user interface and make it easy to use. For example, button placement with respect to a text box is a part of the overall system presentation. First, we must consider first-time system users. With any software user interface, regardless of simplicity, a user must learn how to use it. At first, a new system can be daunting and overwhelming. Nowadays, it is common for most video games to provide walk throughs and basic task challenges to assimilate users into the interface's controls. Such online games as Zynga's *Farmville* and *Cafe World* immediately present tutorials to first-time users. These tutorials engage users with simple tasks that allow them to gain game knowledge without the burden of traditional ad hoc system learning. For our system, we present first-time users a simple help tutorial which teaches them how to use the infrared tracking gloves. By practicing user interface interactions such as pinching and moving objects, future interface interactions will be more fluid and easy to accomplish.

Second, because our users will be positioned several feet farther away than traditional computer users, it is necessary to present them with a low-clutter, large button graphical user interface. At Microsoft Tech-Ed 2008's developer conference, Mark Miller gave a presentation entitled *The Science of a Great User Experience.* In this presentation, Miller

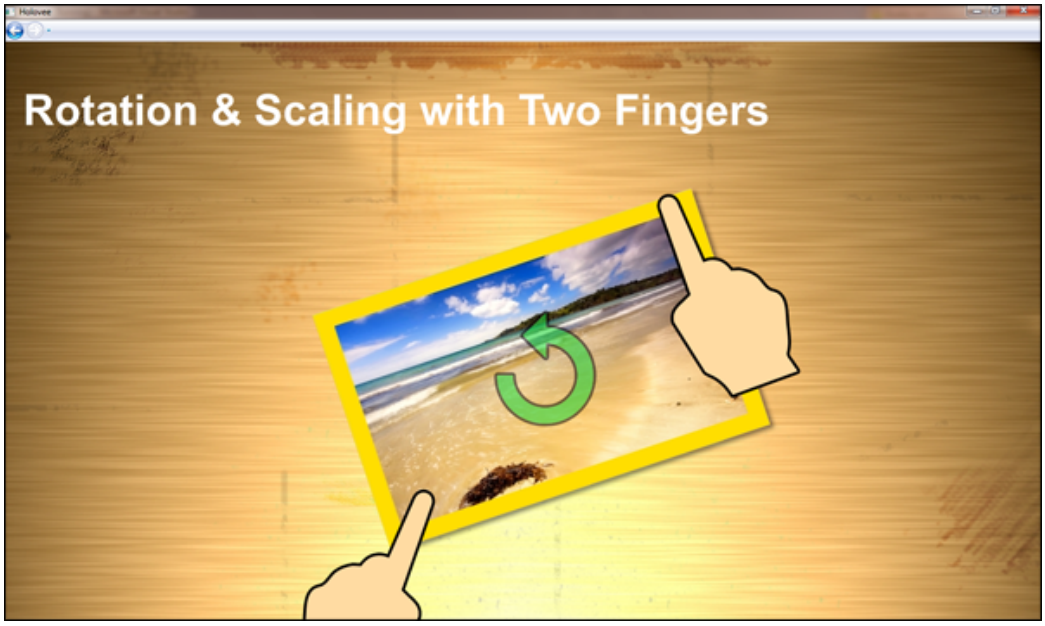

Figure 3.2: Holovee Help Tutorial for Hand Gesture Interactions

analyzed various Windows user interfaces and criticized Microsoft's mechanism of user interface design. Miller suggested such things as keystroke shortcuts and cluttered menus caused more user complexity than intended. Moreover, noise or excessive word verbiage leads to user confusion. To present the user with a clean user interface, we should emphasize relevant information with higher contrast colors and larger font size. In addition, whereas traditional user interfaces worry about minimizing mouse travel distance, our user interface should focus on minimizing excessive hand movement between user interface controls to prevent user fatigue. While these design suggestions seem simple, so many user interfaces do not take them into consideration.

Finally, we must present users with a fun, exciting experience. When interacting with a futuristic user interface, users should feel energized and entertained by its unique behavior. For example, Spore, developed by Electronic Arts, is a creature evolution game which created an unrivaled enjoyment for users. Surely, the game's concept of playing god alone was very intriguing. However, the game's real entertainment factor was its creature generator. According to [38], in under a year, over 100 million creatures were created by users and Spore's virtual population had exploded. This great success can be attributed to the creature

generator's amazing entertainment factor which allowed users to create endless body part combinations. With Spore in mind, we offer techniques to present users with a fun user interface:

- *Offer Full Control* - If developing a builder application, allow users full creative control to shape their creations. Doing so, users will feel empowered and be excited to unleash their imagination.
- *Reduce Thinking* - The less time a user has to think about remembering interface interactions, the more time they can spend being consumed in their tasks.
- *Animate User Interface Elements* - In [28], John Lasseter, a Disney animator, states that procedural concepts such as Squash, Stretch and Exaggeration should be applied to create a unique character and story personality which entertains the audience. If we apply those same animation principles, we can produce very exciting interface controls.
- *Inside Humor* - Insert subtle jokes or comical character movements into the user interface. The Sims, Electronic Art's top-selling PC game, became popular because of its quirky humor and taboo jokes.

3.4 Summary

In this section, we learned that human-computer interaction factors are deeply engrained into futuristic user interfaces.

- Understanding the needs for a futuristic, holographic user interface are very important. When developing a futuristic interface, system designers should consider various questions related to user experience, task completion and ergonomics for more successful results. Moreover, while traditional 2D user interfaces are efficient for everyday computing, they often do not translate well when trying to complex complex 3D tasks.

- Even if a proposed system's technology is highly futuristic, it does not mean that a solution is out of reach. We can think of imaginative solutions to create the *illusion* of a futuristic design.

- When planning our system, we can use several Human-Center Design approaches to distinguish between user interfaces ideas that are desirable, feasible and viable. Most often, iterative software approaches which present pieces of functionality every release cycle, are successful for systems with fuzzy requirements and early unknowns.

- Choosing the right system equipment and futuristic user interface interactions heavily rely on what the user needs to accomplish. Natural hand interactions such as grasping, rotating and translation are much easier to achieve with futuristic controls. In addition, these techniques are much more fluid than traditional interaction techniques.

- A futuristic user interface's overall presentation is the most important factor for engaging potential users. The interface should go beyond initial user expectations and create a fun, entertaining environment for them to see the true benefits of a futuristic system.

Chapter 4

Holovee: A Social Networking Application

4.1 System Design

4.1.1 Basic Overview

With a solid understanding of some best user interface design practices, we begin to visualize how they can be applied to our social networking software application. We will be presenting users with Holovee, a software application which allows them to manage and manipulate their Facebook social networking data. The software application will be tightly integrated with our infrared tracking and holographic display components. Before defining infrared tracking interactions and other components, our software framework should be well-designed and usable from a traditional 2D mouse-keyboard interface. We provide a screenshot of the social networking software application in Figure 4.1.

Knowing our user interface needs, we can discuss how user interface controls were positioned. When designing our user interface, we focus on providing users with less clutter and simple interactions. Upon startup, the software application presents users with two options: a button to manage their status updates and a button to manage their photo albums. In addition, we make use of high contrast colors for the currently selected user interface controls. High contrast colors will help users distinguish important user interface items. As the user selects items, the interface items will dynamically update.

Again, this work's end goal is to provide users with a fun, easy-to-use social networking application. We do not desire to design a user interface which dictates how a user should do something. Rather, we are focused on providing the user the tools to work freely and

Figure 4.1: Holovee Main Screen

creatively. From our human-computer interaction study, we need to define tasks which the user need accomplish with our interface:

- *Manage and Manipulate Photo Albums* - The user shall be able to rotate, scale and manipulate their uploaded Facebook photo albums. Users will be able to add captions and tag friends in specific photos.
- *Easily Update their Facebook Status* - The user shall be able to dictate their current status and publish their information to the Facebook website.
- *Search Friends* - The user shall be able to search their friends list and view profile updates.

These tasks will be a basis for our software application and serve as loose guidelines for our user's needs. Envisioning our rough software requirements, we can begin to formulate our system's design architecture.

4.1.2 Design Diagrams

From Figure 1.1, we showed a basic system component overview and each component's complementary interactions. For this section, we are only concerned with classes related to our user interface design and functionality. In Figure 4.2, we illustrate our software application's class architecture. We break down our user interface into specific software components which provide core functionality to our system. In upcoming sections, we will discuss additional class diagrams that relate to our infrared tracking and speech tracking components.

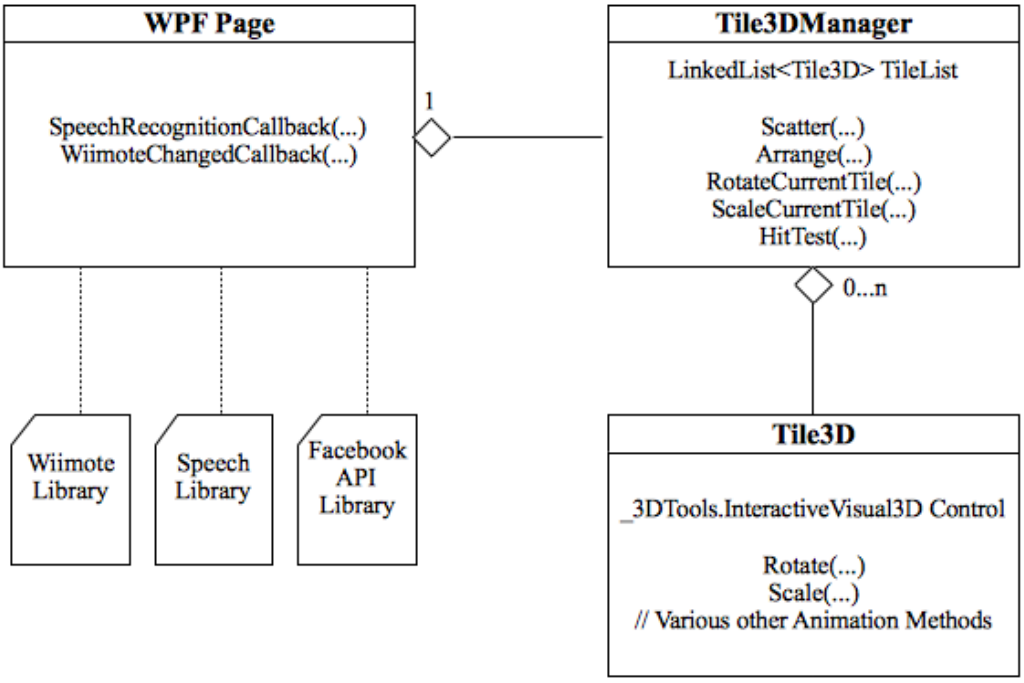

Figure 4.2: Basic Application Structure

Figure 4.2 shows the high level application classes which comprise our social networking software application. From Figure 4.2, we can derive new functionality for our User Photos and Friend Status pages as shown in Figures 4.3 and 4.4. For example, in 4.3, we can visualize how our base Tile3DManager can be tailored towards displaying our user's Facebook photos. We have built various class helper methods into the PhotoTile3DManager class implementation to handle such things as updating captions and showing user tags

in photos. In addition, we created a FacebookPhotoLoader class to asynchronously load photo objects into PhotoTile3D object classes. As seen in Figure 4.4, we create the same type of subclass behavior for our friend status page class. We tailor our friend status objects instances toward interacting with Tile3D instances. Table 4.1 provides a description of each application level class. These class descriptions are only a high level representation of our software framework's most important classes. They do not represent an exhaustive definition of every software application class structure.

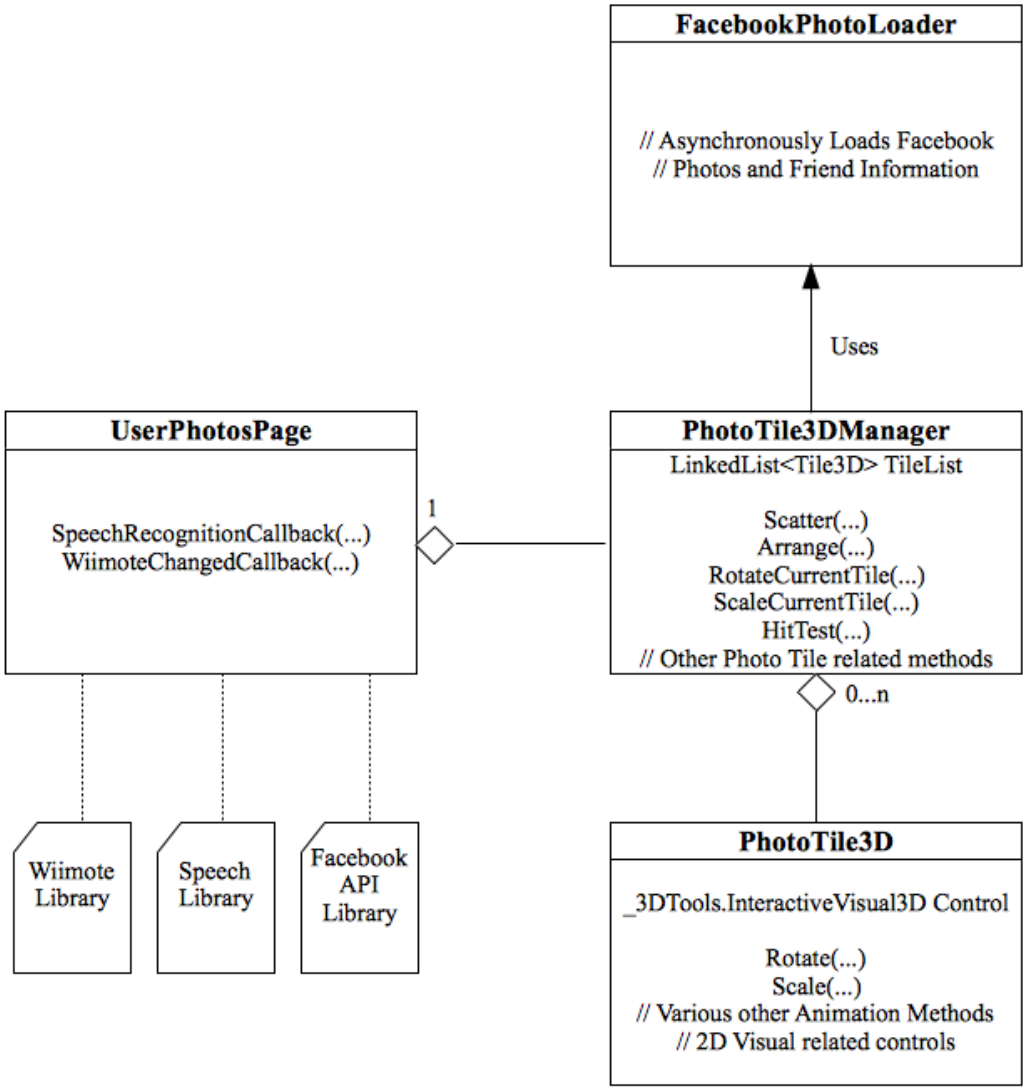

Figure 4.3: User Photos Page Structure

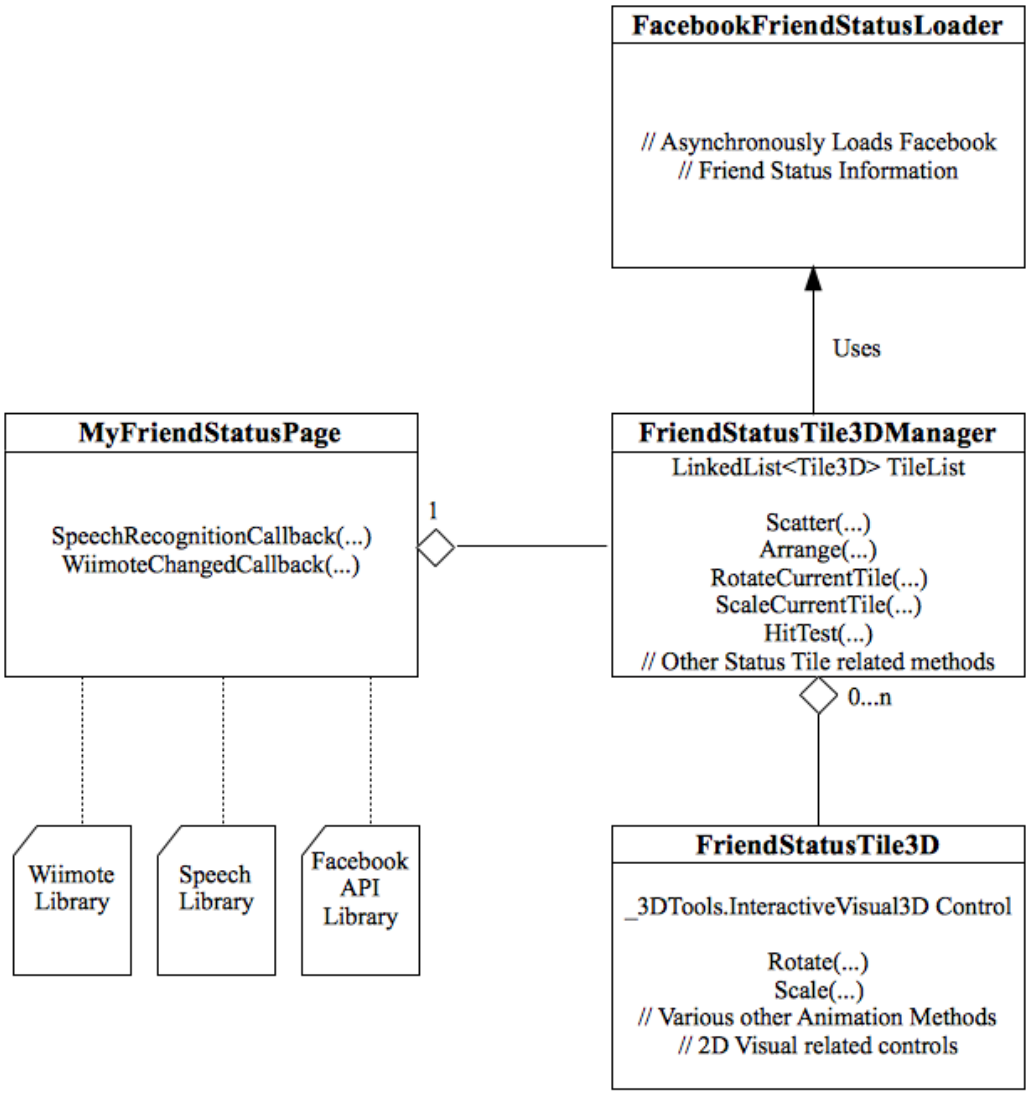

Figure 4.4: Friend Status Page Structure

Application Class	Purpose
FacebookManager	Responsible for managing all interactions with the Facebook Application Programming Interface (API). Given a Facebook developer key and user login, the FacebookManager can allow us to access all profile, photo and friend information about the currently logged-in user.
WiimoteManager	Encapsulates the necessary components for establishing a Bluetooth connection between the software application and the Wiimote hardware device. Also, the WiimoteManager allows various application components to subscribe to Wiimote infrared tracking updates.
SpeechManager	Sets up the Microsoft Speech Recognition engine which allows Holovee to recognize spoken commands and dictation. The Microsoft Speech Recognition allows developers to build grammars which the application can distinctly infer to be application commands.
Tile3D	Allows Windows Presentation Foundation (WPF) 2D Visuals to mapped to 3D cube objects. Tile3D encapsulates a 3D cube model representation and also handles all animation responses to user interactions. New 3D Holovee controls can derive from this class to get basic interactivity controls.
Tile3DManager	Controls all Tile3D objects which are added to its internal list. Includes basic functionality methods for hit testing, scattering and arranging Tile3D objects.

FacebookPhotoLoader	Threaded background photo loader which handles loading photos into the user interface. The FacebookPhotoLoader uses the FacebookManager to invoke the Facebook management API and asynchronously load photo content.
FacebookFriendStatusLoader	Threaded background status loader which handles loading the current user's friend status updates into the user interface. The FacebookFriendStatusLoader uses the FacebookManager to invoke the Facebook management API and asynchronously load status content.

Table 4.1: Software Class Diagram Definitions

4.2 Programming the Framework

4.2.1 Choosing a Development Environment

For our development environment, we were looking for a development environment and programming language which best fit our software application. As aforementioned, we chose Microsoft Visual Studio 2008, Windows Presentation Foundation (WPF) and C# because of our familiarity with the technologies. In particular, there are plentiful resources for quickly building WPF applications and creating dynamic graphical environments. Windows Presentation Foundation is a graphical system which is based on DirectX and allows developers to focus on building application components rather than programming for specific graphics hardware. We also have experience with OpenGL, another standard graphics

application programming interface (API) for interacting with a computer's graphics hardware. However, again, we are looking for a programming technology which has an abundance of resources. OpenGL has great tutorials and source code examples, but did not seem fitting for quickly prototyping our application.

Because of Visual Studio's great functionality, debugging applications is rather simple and facilitates tracking software bugs. Another great thing about C# application frameworks is that they can easily be reused in different applications. A standard DirectX application consists of several main methods which are called during execution time. During initialization, our application will present users with a main menu to manage their social networking data. Users will use both speech commands and their infrared tracking gloves to navigate the Holovee user interface.

As in any UI-based software application, there is a main message processing loop which handles updating and drawing the user interface. Unlike a traditional DirectX game loop, we do not have to define these update methods, but rather can subscribe to messages and be notified via events. For example, if the user issues an *Add Caption* speech command, this command will be applied to our user interface, allow a user to add a caption to their currently selected photo. Again, we do not have to continually check for this *Add Caption* state as C# simply notifies us when the event has occurred. In the future, we wish to abstract our input API into a generic layer which can later integrated with our infrared tracking component. Having a generic input abstraction layer will allow our Holovee interface to remain unchanged as we add additional future components. For example, if the user issues a wiggle command, the infrared component tracking layer may translate this to a "rapidly translate image position" event. The underlying code to translate the image remains unchanged as we are building on top of its generic interface.

Finally, C# provides numerous user interface components and visuals to rapidly prototype component functionality. Having an efficient mechanism to try different features is an essential tool for rapid software prototyping. We can easily layout 2D visuals onto 3D controls to make for interactive content. For example, Holovee's photo management component uses a 3D viewport containing several 3D Tile instances which have texture mapped

2D visuals. WPF takes the burden off developers to handle this mapping and allows for exciting user interface component combinations. With a solid understanding of our tool set, we can progress to how we will use the C# and WPF framework to create an appealing application.

4.2.2 Working with the Facebook API

When developing on top of any application programming interface (API), it helps to become very familiar with the development interface. At its core, the Facebook API uses a Representational State Transfer (REST) communication interface for requesting and receiving data [14]. REST interfaces are commonplace in internet communication and use HTTP protocol to interface with a backend REST server. Moreover, the Facebook API was originally built to work with a Javascript interface. Since the Facebook API does not natively support working with C#, we used a C#-based Facebook API library developed by [21]. With this C# library, we did not have to be concerned with translating C# to Javascript commands or parsing XML-like data response results. The C#-based library encapsulates all REST-based method calls via wrapper classes for handling Facebook API data. These wrapper class objects allow developers to focus on applying that content to their application.

Remember, we are not defining the available interface methods and or their data response formats. Therefore, at any time, an application could experience issues with a slight change in data parameters. With the Facebook API, we experienced this occurrence multiple times where our code remained unchanged, but our application broke in several places because of an unannounced API change. However, the developers in [21] were very responsive in updating their C# library to match Facebook API changes. Moreover, before planning a API-based user interface, it's very useful to consider a few areas:

- *Available Interface Methods* - Which interface methods does the API expose and which ones do we want to take advantage of for our application?
- *API Invocation Format* - How do developers invoke an exposed API method and what

kind of parameter formats do each method support?

- *API Data Response Format* - How is the data returned from the REST-based server and how is the data formatted?
- *API Library Support* - Do developers regularly fix bugs and issues in the API? How responsive are developers to programming bugs and issues?

When accessing the Facebook REST servers to read content, there may be a slight delay for contacting the server and then receiving the data response. This network lag could be attributed to local computer network issues, Facebook backend traffic control or many other reasons. With this in mind, for each method in the C#-based Facebook API library, there are also complementary asynchronous methods for requesting and receiving data. However, for our implementation, we use the library's synchronous methods and wrap them in our own asynchronous thread-based method calls. We take this approach because of our code's layout and loading method. For example, when we load a user's photo album set, we create a new loader thread which invokes a Facebook API call. Since we want to create C# bitmap images from this data, we wait for the data to be downloaded and then asynchronously load each bitmap into the user interface. Moreover, since the API call already exists in its own background thread, it does not block the main user interface and appends data when its ready to be loaded. Listing 4.1 shows an example C# invocation method and Listing 4.2 shows an example XML-like data response.

```
List<string> AlbumCoverList = new List<string>();
AlbumCoverList.Add(album.cover_pid);

// Invoke Facebook API method to get user photos
var albumcover = FacebookManager.Instance.SessionApi.Photos.Get("", album.aid,
    AlbumCoverList);

foreach (Facebook.Schema.photo photo in albumcover)
{
  // Load Photo Data into User Interface
}
```

Listing 4.1: Facebook C# API Invocation Code

```
<?xml version="1.0" encoding="UTF-8"?>
<photos_getAlbums_response xmlns="http://api.facebook.com/1.0/" xmlns:xsi="http:
    //www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://api.
    facebook.com/1.0/ http://api.facebook.com/1.0/facebook.xsd" list="true">
<album>
<aid>34595963571485</aid>
<cover_pid>34595991612812</cover_pid>
<owner>8055</owner>
<name>Films you will never see</name>
<created>1132553109</created>
<modified>1132553363</modified>
<description>Some random image</description>
<location>Rochester, NY</location>
<link>http://www.facebook.com/album.php?aid=225&id=85</link>
<size>30</size>
<visible>friends</visible>
<modified_major>1241834423</modified_major>
</album>
</photos_getAlbums_response>
```

Listing 4.2: Facebook API Response

4.2.3 Working with the Wiimote Tracking Library

As aforementioned, Holovee uses a Wiimote game controller paired with infrared LEDs to create a hand gesture based user interface. We use a C#-based Wiimote library developed by Brian Peek in [41] to interact with the Wiimote game controller. Peek's Wiimote library allows application developers to easily connect to a Wiimote game controller via a Bluetooth interface. After a successful connection is established, applications can receive event notifications upon infrared light discovery. Once tracking, the Wiimote library gives developers access to the infrared light's (x, y) coordinates. These infrared light coordinates can then be normalized and mapped to our Holovee interface which draws basic screen cursors. From our testing the Wiimote game controller can track up to four points simultaneously at a range of about 20-30 feet [26]. During our testing, we found that clustering multiple infrared LEDs together tracked much better than single LEDs. Even though our application does not make use of them, the Wiimote library gives easy access to the Wiimote's accelerometer and rumble controls. More information about our infrared LED component will be discussed in the upcoming Input Devices chapter.

Moreover, because we are within close proximity of our display surface, we found that there is not any need for Wiimote tracking calibration. Unlike previous projects such as the Wiimote whiteboard project [29] which require precise control of user interface elements, our research showed that infrared tracking responses were very accurate within our five to ten foot display distance range. Past approaches presented users with four corner calibration dots which then allowed the software application to transform every infrared point therein to the appropriate coordinate system. Again, we did not see immensely better or worse performance results by calibrating the Wiimote hardware device.

4.2.4 2D Visuals on 3D Controls

As aforementioned, Holovee takes advantage of WPF's built-in 3D capability and map traditional 2D visual controls to 3D objects. Unsupported by WPF's standard functionality, the WPF development team created a simple 3D control library which allows 2D visuals such as stack panels, list boxes and more items to be set as the visual content for a 3D

object [52]. While our object is simply a 3D cube, these 2D visuals can be mapped to any model mesh. The 2D on 3D control library even handles all user interaction event notifications. For example, when clicking a 2D visual mapped on a 3D object, the control library maps the 2D mouse click to a spot in 3D space. As seen in Figure 4.5, we map a 2D image object control onto a 3D cube.

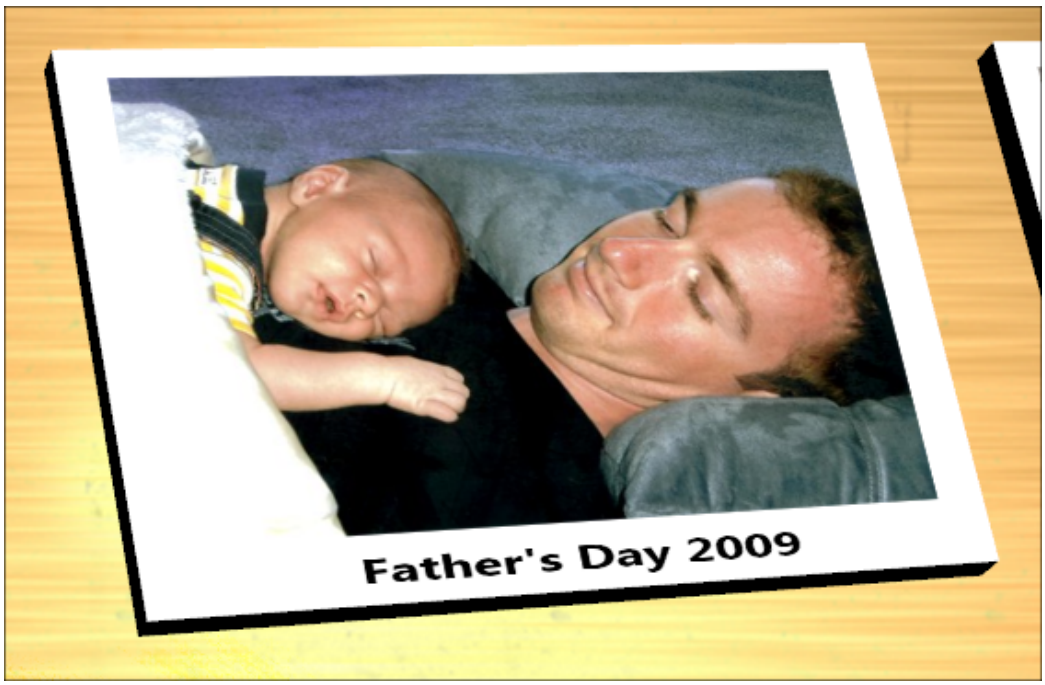

Figure 4.5: 2D Image Visual mapped onto a 3D Object

Thinking back to our application-level defined tasks, we want users to be able to manage their Facebook photos and status updates. For photo controls, we use a simple image control with space for an image caption. For status updates, we use a simple stack panel with encapsulated elements for display user comments and status updates. Mapping these 2D controls onto 3D objects allows for greater future system flexibility and broadens our skill set. If we were to expand our system to mapping 2D visuals onto more complex 3D objects, we now have a the framework to do so. In addition, with 3D WPF objects, we can now perform eye-catching transformations which are not obtainable with traditional 2D objects.

4.3 Application Usage

This section is directed towards our system users. It serves as a rudimentary, concise manual for interacting with our social networking management application.

4.3.1 First-Time Use

A major system goal is for our software framework to be easily learnable. When using Holovee for the first time, users should feel comfortable knowing they can easily acquire the necessary skills to manage their social networking data. Before creating a simple help tutorial for users, we must first understand the art of a good tutorial. Tutorials are highly metaphoric. With tutorials, we need to relate our system's functionality to something which users can relate. We aim to teach users the simple hand gestures and speech commands which can be used to control Holovee. To this extent, we created a simple Adobe Flash tutorial which guides users through the process of interacting with our user interface. Simple hand icons with text annotations are presented to simulate how 3D tiles can be manipulated. Moreover, we continually provide speech command cheat sheets to make users aware of their available speech commands.

After viewing our quick startup tutorial, users are presented with Holovee's main menu. Holovee has two main menu options for navigating their friends' status updates and managing their photo albums respectively. From the friend's status updates screen, users will be presented with several 3D tiles which represent recent status updates by their friends. Each of these tiles will contain several components as detailed in Figure 4.6. For example, the status tile contains a picture of the user's friend along with their name and current status. In addition, the status tile contains a rotating block of the status' current comments and likes. As the current user adds new comments to their friends' statuses, these updates will be displayed in real-time on the 3D tile.

Similarly, users can select the photo management menu option for interacting with their Facebook photos. Upon first entering this view, users are presented with every photo album in their repository. Each photo album cover is tagged with the album's name. To open a

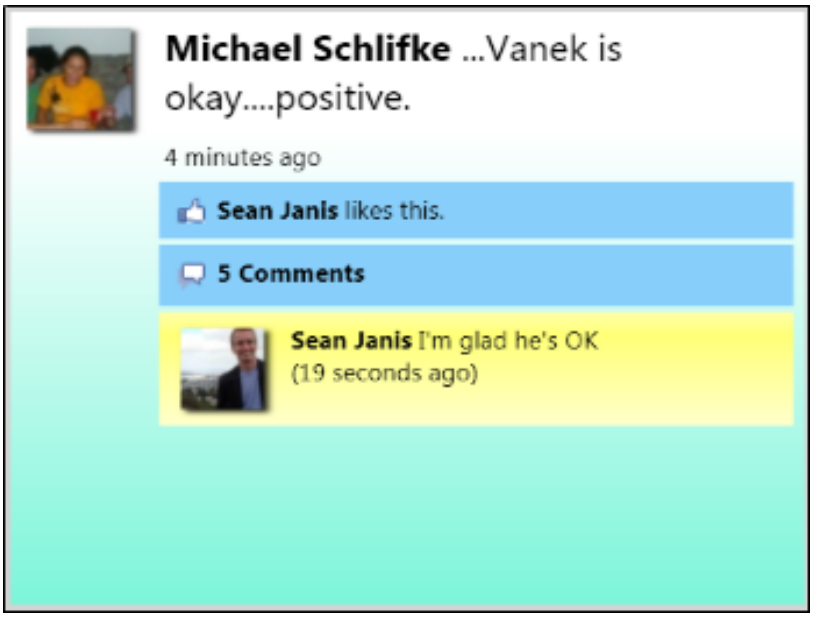

Figure 4.6: Facebook Friend Status Tile

photo album, users can simply hover over their selected album with their infrared tracking cursor and say *open*. A complete listing of all available speech commands is provided in the next section. Moreover, each of these 3D tiles can be manipulated via translation, scaling and or rotation with our infrared tracking controls.

4.3.2 Tasks and Navigation via Speech Commands

As will be discussed further in the upcoming Input Devices chapter, we take advantage of Microsoft's Speech Recognition C# library to control certain Holovee user interface interactions. Table 4.2 shows a listing of the available commands and their typical usage scenarios. For example, as seen in Figure 4.6, we can append comments by speaking the command, *Comment*, followed by a specific phrase. Comments will be displayed in a transitioning panel within the 3D friend status tile. Similarly, if the user dictates the *Like* command, the friend status tile will reflect the current user's interest in their friend's status. Figures 4.8 and 4.9 show the Holovee user interface recognizing user speech commands and directing users with directions.

Figure 4.7: Facebook Photo Tile

Speech Command	**Availability**	**Description**
Main Menu	Any Screen	Navigates the user back to the Main Menu screen.
Status Updates	Main Menu	Navigates the user to the Status Updates screen
Manage Photos	Main Menu	Navigates the user to the Manage Photos screen
Arrange	Status Updates and Manage Photos	Arranges the currently displayed 3D Tiles into a grid-like fashion for easy manipulation.

Scatter	Status Updates and Manage Photos	Randomly scatters the currently displayed 3D Tiles into various orientations and positions.
Find <friend name>	Status Updates and Manage Photos	Filters the currently displayed 3D Tiles and only displays tiles tagged with the specified friend name.
Caption	Manage Photos	Allows users to add captions to the currently selected photo. When the user speaks the phrase, *Caption*, the program will alert the user to *Speak your caption now...* At this point, the user can dictate any phrase to be attached to the photo.
Tag <friend name>	Manage Photos	Allows users to tag friends in the currently selected photo. For example, users can dictate, *Tag Sean Janis*, to tag him in the current photo.
Photo Albums	Manage Photos	Returns the user to the main photo album screen.
Like	Status Updates	Allows users to say they *like* the currently selected friend status update.

Comment	Status Updates	Allows users to add comment to the currently selected friend status update. When the user speaks the phrase, *Comment*, the program will alert the user to *Speak your comment now...* At this point, the user can dictate any phrase to be attached to the status update.

Table 4.2: Available Speech Commands

Figure 4.8: Adding Captions to Facebook Photo Tiles

Figure 4.9: Adding Tags to Facebook Photo Tiles

4.3.3 Manipulating 3D Controls

As aforementioned, our infrared tracking controls will allow users to manipulate Holovee's 3D tiles via translation, rotation and scaling transformations. When a user wears the infrared tracking gloves, infrared light is tracked by the stationary Wiimote game controller. As new infrared tracking points are discovered, Holovee will overlay a colored square over the corresponding screen point. When the same infrared tracking points are lost, Holovee will remove the colored square from the display. Because we are only tracking the user's index fingers, we will at most display two colored squares within the software application. Similarly, tracking only two index fingers limits our ability to simulate complex hand gestures such as pinching and or hand sliding.

As seen in Figure 4.10, when a user hovers one finger over a 3D tile, the desired tile is highlighted in yellow. This action indicates that the current photo is selected and is ready to be transformed. Hence, the user can use their other infrared tracked finger to stretch and rotate the image. Moreover, since we only have two tracked user hand points, only one 3D

tile can be rotated or scaled at any time. However, considering 3D tile translations do not require two fingers, we can simultaneously drag and drop two pictures at the same time.

Figure 4.10: Simple 3D Tile Selection and Translation

As seen in Figure 4.11, when a specific 3D tile is highlighted, users can use two fingers to stretch and contract the image's size. By moving their fingers farther apart, the current image is expanded to the distance between the two tracked infrared points. Moreover, the art of scaling an image is cumulative. For example, every time Holovee loses and then reacquires infrared tracking lights, reacquiring the infrared lights does not reset the image to its current size. Stretching one's fingers after a reacquire scenario will only enlarge the image even more. Similarly, when a user moves their fingers closer together, the current image is contracted to the distance between the two tracked infrared points.

Figure 4.11: 3D Tile Scaling

As seen in Figure 4.12, our scaling movements can be combined with rotation transformations. The first infrared point that is tracked is considered our rotation axis of orientation. For example, if the user's left index finger is tracked first, then the user's right index finger can be moved to rotate the 3D tile around their left finger. Similar to the scaling transformation, losing and then reacquiring the infrared tracking points leads to a cumulative rotation operation. This is the same Wiimote tracking rotation logic method as used in [29]. Again, because our 3D tile class encapsulates all logic for translation, scaling and rotation transformations, we can easily derive from this base functionality to allow us to create very unique 2D visual designs on 3D objects.

4.4 Other Software Approaches

4.4.1 XNA Game Framework

Before settling on a C#-based, WPF-driven development environment, we experimented with the XNA game framework and DirectX technologies. The XNA game framework is a C#-based software development framework which allows developers to easily prototype Windows and Xbox360 games [61]. XNA aims to reduce redundant code by providing

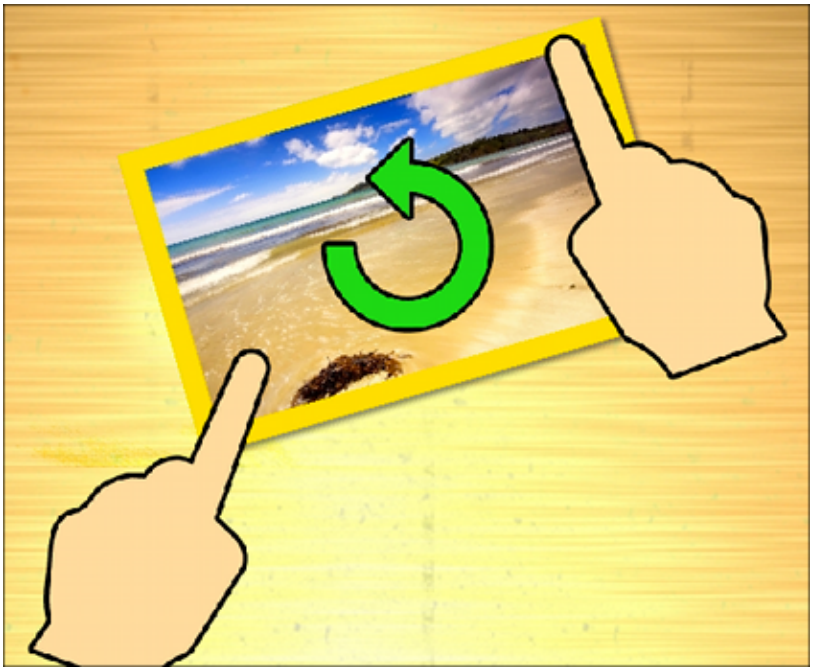

Figure 4.12: 3D Tile Rotation

developers with boilerplate scaffolding classes. Similar to WPF-based applications, there are plentiful resources for quickly building XNA-based applications and creating dynamic graphical environments. While we have experience with these technologies, they are often tailored towards programs which feature pure 3D environments rather than user interface elements. Although WPF-based applications allow for 3D object manipulation as made evident by Holovee.

Another great thing about XNA-based application frameworks is that they can easily be reused in different applications. A standard XNA-based application consists of several main methods which are called during execution time:

- *Initialization* - Contains all calls to set up standard DirectX objects which will allow our application to make use of a computer's graphics hardware.
- *Update Scene* - Updates environment objects with respect to the current application's system clock. In addition, this method usually polls user input data and applies it to scene objects.

- *Draw Scene* - Draws updated environment objects onto the screen. This method usually involves flushing graphical data from a background rendering buffer to the main screen buffer.

In our early XNA prototypes, the Update Scene method would accept user input which could be applied to our user interface. Similar to our WPF implementation, the Update Scene method would be an entry point for Wiimote tracking data. For example, if the user issues a rotate command, this rotate command will be applied to our 3D tile, causing it to change orientation when being drawn. With the XNA framework, we would have more granular control over our render screen, but would have write more code to accomplish simple tasks.

Finally, XNA and DirectX provide High Level Shading Language (HLSL), a programmable shading language, which allows developers to dynamically shade and manipulate vertices during runtime. Instead of continually changing C++ code to try new effects, HLSL allows developers to write script-like programs which can be easily loaded into applications without the need for recompiling code. Having an efficient mechanism to try different features is an essential tool for rapid software prototyping. Again, while both frameworks provide great features, the traditional WPF-based application's abundance of user interface elements seemed proficient for our application.

4.5 Summary

In this section, we gave a detailed overview of the Holovee software framework which drives our hardware system. Again, we strive to produce a meaningful user interface which can show users the possibilities of a futuristic user interface. There are many components and features we would like to add to our system, but feel are outside the scope of this project.

- Before defining a futuristic user interface, it is highly beneficial to detail potential tasks which the user need accomplish with our interface. These tasks are important

for understanding the system's purpose and also building a set of software requirements.

- Various management classes are responsible for Holovee's core functionality. These core classes are tailored towards asynchronous data loading methods and 3D user interface controls to make for an interesting, intuitive software application.

- Using a great programming environment and existing class libraries allows for rapid prototyping and expedited development time. Instead of reinventing the wheel, it's very helpful to use preexisting code libraries such as the C# Facebook API and Wiimote Tracking libraries and then focus on other application components.

- A futuristic user interface's controls should be intuitive and exciting to engage user interest. We walked through Holovee, a social networking management application, which shows the possible exciting application arenas that can be developed for a futuristic user interface.

Chapter 5

User Tracking and Input Devices

5.1 Input Devices

As discussed in our user interface design practices section, input devices are our users' natural extension for interacting with computing hardware. Well-designed input devices allow users to complete simple tasks. Great-designed input devices allow users to seamlessly feel connected to their device and accomplish complex interactions. Moreover, input devices are one of the most important system components as a poor input device can diminish a user's interface experience. Again, we are less concerned with traditional 2D input hardware as it is not used in our end system.

5.2 Tracking Mechanisms

5.2.1 Infrared LED Tracking

We begin our futuristic input device discussion with infrared tracking mechanisms, a well-known means to track users in a computing environment. The basic components behind an infrared tracking system are an infrared camera and infrared light emitting diodes (LEDs). Infrared cameras are used to detect infrared light. Figure 5.1 shows the range of wave lengths in the electromagnetic spectrum. Infrared light falls within the 1 millimeter to 750 nanometer wavelength, which is invisible to the human eye [59]. Comparable to higher frequency wavelengths such as x-rays and gamma rays, infrared radiation is favorable for tracking applications because it is relatively harmless to humans.

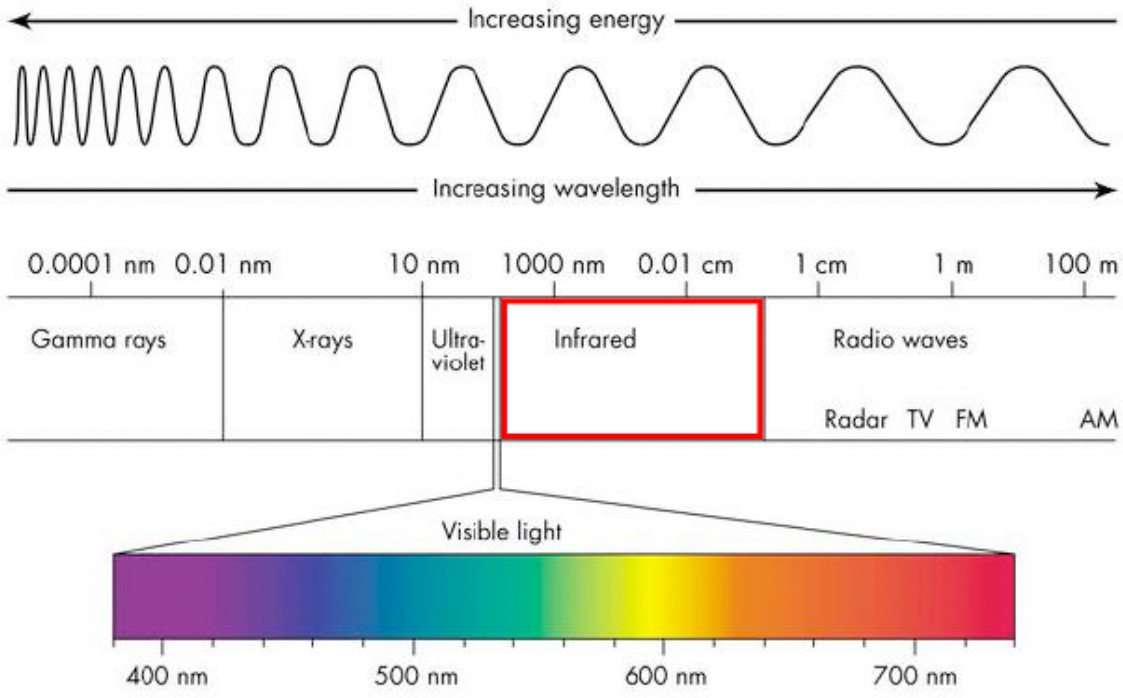

Figure 5.1: The Electromagnetic Spectrum - Infrared Light is invisble to the human eye [13]

Infrared tracking mechanisms can be broken into two categories, inside-out and outside-in tracking [4]. An inside-out system involves users wearing sensors and being tracked by an external source. Similar inside-out tracking approaches can be realized in actors wearing motion capture suits and having their movements translated to virtual characters. In our infrared tracking system, the user would wear attached infrared LED emitters to their fingers or bodies and then be tracked by an infrared camera. An inside-out approach is favorable because infrared LEDs are easily trackable, low power and lightweight. Inside-out infrared tracking systems are much more successful when the user-worn infrared LEDs have a wide field-of-view (FOV). Having a wider FOV allows a system's infrared camera to recognize infrared cues even at odd angles, making for smoother user interactions.

Conversely, an outside-in approach involves no user-worn equipment and tracks user movement using stationary sensors. Most often, outside-in approaches requiring greater tracking algorithms to interpolate the user's precise position [4]. With an infrared tracking system, an outside-in approach might encompass a stationary infrared camera and several

clusters of infrared LEDs. Then, the user would wear reflective tape to redirect the emitted infrared LED light back towards the infrared camera. For reflective tape, retro-reflective tape is often used because it can reflect light even at odd angles, having a similar effect as the inside-out system's wide FOV infrared LEDs. While more attractive, the outside-in mechanism proved less relied in our testing results. We will discuss the precise testing procedure in the upcoming section. Figure 5.2 shows a general diagram to the Wii video game console's tracking approach.

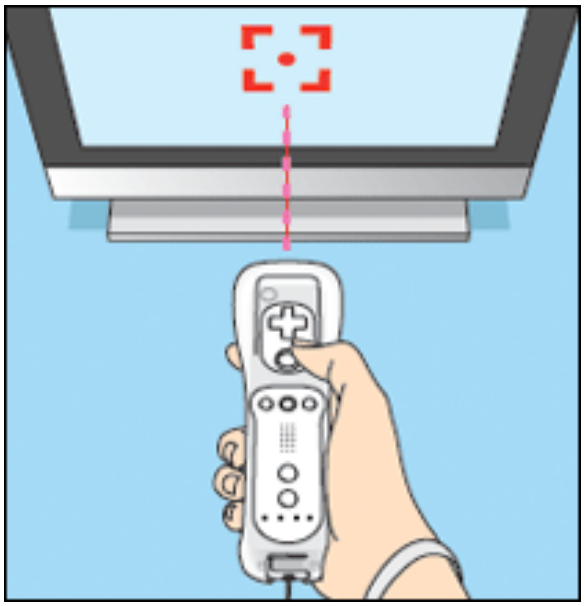

Figure 5.2: Wii Video Game Tracking System [39]

Depending on the user's distance from the infrared camera and the infrared camera's tracking range, we can cluster together multiple infrared LEDs for better performance results [26]. As aforementioned, the Nintendo Wii gaming system features several infrared tracking controls to track a stationary infrared sensor bar to determine a user's position. The infrared sensor bar clusters together several infrared LEDs for bettering tracking performance. By equipping the user with infrared tracking gloves, we give them ability to make user interface selections and interact with Holovee. For our system, we are only concerned with tracking a user's hand in 3D space for the purpose of interacting with our holographic user interface. We are not concerned with tracking other things such as hand

orientation via an accelerometer. Moreover, users will need to manipulate their Facebook social networking data and navigate through system menus, all of which can be accomplished with infrared LED gloves and speech commands.

5.2.2 Markerless Tracking

As the name implies, markerless tracking focuses on outside-in approaches which do not require the user to wear tracking sensors. Rather, the user simply steps into the scene and interacts with the user interface without any need for additional equipment. While not used in our end system, it is important to understand how markerless tracking technology works and how it can be applied to future systems. The recent inspiration for markerless tracking technology can be linked to Microsoft's Project Natal hardware.

Project Natal is a hardware accessory for Microsoft's popular Xbox 360 gaming console that allows users to interact with video games by intuitive body gestures without the need for a controller [37]. The Project Natal hardware features embedded infrared and visible cameras which can detect user movement under various lighting conditions and room obstacles. In various technology demos, The recent advances in markerless technology via Microsoft's Project Natal illustrate how input devices are becoming more engrained into our body. In this case, the user's actual limbs and extremities are the direct input device without the need for additional hardware.

Similar to Microsoft's Project Natal, the authors in [7] and [55] explore various markerless options for capturing motion capture actors and their accompanying garments. As seen in Figure 5.3, a common approach to markerless motion tracking is to build a simple hierarchical representation of the tracked model. For instance, if our algorithm is tracking a user's full body, then we would build a rigged skeleton template mesh to facilitate tracking. While the systems are great for capturing canned actor performances, they were reported to have slight tracking errors which could be corrected during the post-processing procedure. In addition, both motion capture systems used an 8-camera setup which tracking performers in a 360-degree fashion. Applying these concepts to our user interface, we would not need to track our user's full body and would not require a 360-degree view of the user.

Rather, we could use a frontal tracking camera to construct a skeleton template mesh of the user's hands. By doing so, this would allow us to develop an algorithm that analyzes user hand gestures and translates them to user interface actions. In the future, this could allow our users to not have to wear special gloves to interact with a futuristic interface and be able have a greater granularity over their selections.

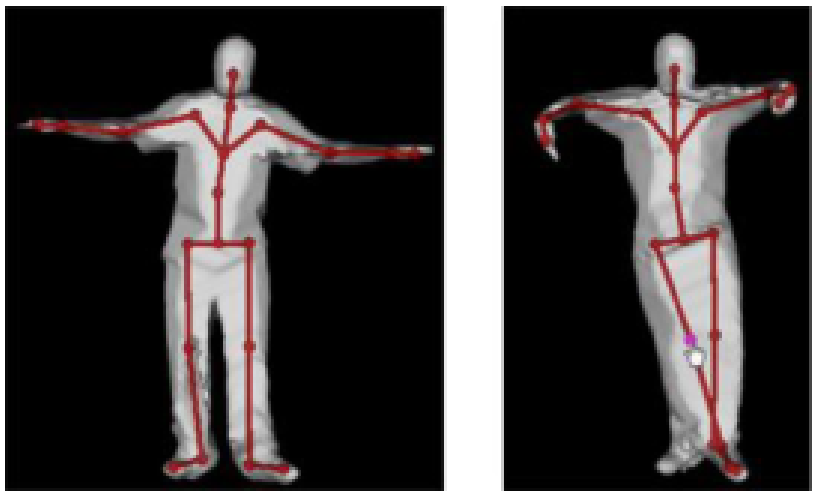

Figure 5.3: Markerless Figure Representation [55]

As stated by the authors in [6], bridging the gap between seamless input hardware and user interactions is an important and challenging problem. With markerless tracking solutions, there is also greater concern for multiple viewers interfering with the primary user's interface interactions. Moreover, unlike the infrared marker approach, the markerless approach forces the system software to determine which user to track.

5.2.3 Voice Commands and Speech Tracking

Speech input devices are an exciting new way to control user interfaces and track a user's position. Surely, microphones have existed as computer input hardware for many years. However, in the last decade, we are seeing a steady resurgence in virtual reality and software applications using speech input as a complement to futuristic user interfaces [43]. In addition, the Microsoft Windows 7 and Vista operating systems have made great advances in speech recognition technology [36]. Microsoft's speech technology easily allows

software developers to embed speech recognition functionality into their programs. By encapsulating low level speech input hardware controls into high level accessibility classes, developers need only be concerned with what to do with certain speech commands, rather than how to process them.

Voice commands have been embedded into such military video games as *SOCOM Navy Seals II* and *Tom Clancy's End War* to control user interfaces and in-game characters. Like many speech-based user interfaces, these games forced users to explicitly trigger speech command mode. By forcing user initiated speech modes, software applications need not be concerned with distinguishing between casual user conversations and actual input commands [6]. Figure 5.4 shows a typical speech recognition processing engine block diagram. Although we don't use Microsoft's Text-To-Speech (TTS) feature, it is still a very powerful feature of the speech framework. Acoustic input data is fed into the speech engine and is broken into a lexical grammar. This lexical grammar can then be translated by a processing layer and be accessed by easy-to-use API for command input. Moreover, with our system, Microsoft's Speech Recognition engine does most of the heavy lifting for distinguishing between casual conversations and user interface commands. To help the speech recognition, Microsoft's SDK allows us to define lexical grammars such that it will have a greater probability for recognizing the command. For example, in our system, it was very difficult for the speech engine to understand the phrase *Tag <friend name>*, where *friend name* is the precise person the user may be looking to tag in their photos. Hence, we use Microsoft Speech library objects to create semantic grammars with a key, *Tag*, and a value, a specific friend name derived from the Facebook API.

The authors in [43] present a virtual reality solution which uses both discrete and continuous voice recognition. Discrete voice commands are defined as specific commands for which the user wishes to accomplish. Continuous voice recognition allows for more language command processing via natural sentences and less reliance on remembering individual commands. Another voice-driven user interface could use speech as a user tracking and identification mechanism. In addition to tracking users with an infrared camera, Microsoft's upcoming Project Natal advertises tracking users via speech and determining their

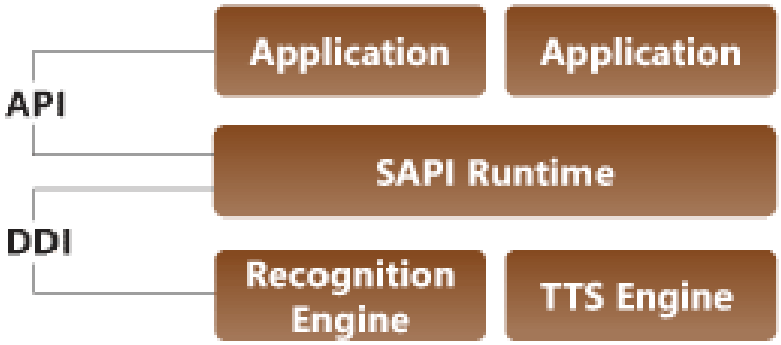

Figure 5.4: Speech Recognition Processing Engine [36]

attributes [37]. However, Project Natal's exact algorithms to determine a user's position and identity are unpublished.

Now, envision Holovee, our social networking management application. We have two options. We can either eliminate our infrared tracking component completely and just use a speech input component. Or conversely, we can mesh the two input approaches into a hybrid approach which has great potential. We chose the latter hybrid input approach. With our system, our goal was to minimize command learning time. For instance, users can navigate through menus by speaking the labels on specified menu items. Moreover, as we discussed, we present users with occasional reminds for which speech commands they have available. Hence, even if our users cannot remember how to navigate Holovee, we provide helpful hints to facilitate the process.

5.3 Gesture Recognition Formulas

5.3.1 Basic Algorithms

As discussed in the last chapter, we based our infrared tracking transformation on the work described in [29]. In this work, Johnny Lee, the main researcher, used simple yet powerful formulas to compute the correct distance and rotation angle between two infrared light points. After computing these transformation values, much trial and error went into adjusting them to better fit our application. For example, when we first ported Johnny Lee's rotation angle computation code, our 3D tile rotated erratically because did not properly

scale the rotation value. Similarly, every time the Wiimote would lose our tracking points, reacquiring them would lead to a complete, undesired image resize. Again, we only compute the distance and rotation calculations when we are successfully tracking two infrared points.

In Listings 5.1 through 5.4, we provide the mathematical formula for computing the distance between two infrared points. During each tracking cycle, we store the last known infrared cursor positions. We then compute a distance calculation between these two points and also the current two known infrared cursor positions. If the distance between the current tracked points is less than the last known distance, then Listing 5.3's scale quotient will be larger than the previous value. Hence, the currently selected 3D tile would be enlarged as the scale value is greater. Similarly, if the scale quotient is smaller, then the user interface's 3D tile would be shrunk. The smoothing constant value was determined by trial and error to observe the best possible scaling effect for 3D tiles.

$$SmoothingValue = 3.5 \tag{5.1}$$

$$LastScale = CurrentScale \tag{5.2}$$

$$CurentScale = \frac{Distance(CurrCursorPos_1, CurrCursorPos_2)}{Distance(LastCursorPos_1, LastCursorPos_2)} \tag{5.3}$$

$$\Delta Scale = (CurrentScale - LastScale) \times SmoothingValue \tag{5.4}$$

In Listings 5.5 through 5.10, we provide the mathematical formula for computing the rotation angle between two infrared points. The *FindAngle* function defines an encapsulated arctangent function. First, we compute the arctangent between the last cursor position x-value and the last cursor position y-value. Then, we perform arctangent computation for the current cursor position's values. By computing these arctangent values, we can determine the angle formed between the slope of these two tracked point sets. We perform a simple radians to degrees computation to ensure proper rotation transformations. This rotation value can then be applied to our 3D tile for a cumulative tilt effect.

$$LastRotation = CurrentRotation \tag{5.5}$$

$$LastAngle = FindAngle(\Delta LastCursorPos_x, \Delta LastCursorPos_y) \tag{5.6}$$

$$CurrAngle = FindAngle(\Delta CurrCursorPos_x, \Delta CurrCursorPos_y) \tag{5.7}$$

$$CurrentRotation = CurrAngle - LastAngle \tag{5.8}$$

$$CurrentRotation = CurrentRotation \times \frac{180}{\pi} \tag{5.9}$$

$$\Delta Rotation = (CurrentRotation - LastRotation) \tag{5.10}$$

The more difficult code logic involves maintaining the correct tracked points ordering. For example, thinking back to last chapter, we discussed how our 3D tile's fixed rotation origin depended on which infrared point was tracked first. Hence, in our Wiimote data manager, we maintain a running internal state of which infrared point was tracked first and use this information to properly compute the above formulas.

5.4 Tracking Design

5.4.1 Infrared LED Circuit Design

When designing a basic electronics circuit, it's useful to examine our application's needs. In our application, we wish to maximize the Nintendo Wiimote's tracking capability and track four infrared LEDs. Two infrared LEDs will be mounted on the user's left hand and other two on their right hand. As the user moves their hand, these infrared LEDs will be tracked by our stationary Nintendo Wiimote. Moreover, when designing our circuit, we found it useful to consider the following items:

- *Power Source* - Our system's power source is the basic driver for our circuit. We wanted both a lightweight, sufficient power supply that would be favorable for our users.

- *Infrared LED Specifications* - Since our Nintendo Wiimote will need to track our infrared LEDs, we need to consider, infrared LED field of view (FOV) for consistent tracking and power specifications for appropriate resistor placement.
- *Proper Resistor Placement* - Infrared LEDs will burn out if proper resistors are not places into the end circuit. Proper circuit resistors ensure our infrared LEDs are not overpowered past their specifications.

Table 5.1 details our infrared LED circuit parts list. Figure 5.5 shows an enhance picture of our circuit design. As mentioned, the overall infrared LED circuit is rather simple, but requires proper knowledge of limiting current via resistors to ensure the infrared LEDs are not damaged. Instead of manually computing the required resistance per infrared LED, we found it easier to input our various parts parameters into an LED resistor calculator in [2].

Part	Specifications	Cost
High-Ouput Infrared LED	Forward Voltage (V): 1.28V, Rated at: 100mA	$1.99 each
Circuit Resistor	10 ohm	$1.99 each
Watch Battery	3V	$1.99 each
Watch Battery Holder	N/A	$1.99 each
Wire	N/A	N/A

Table 5.1: Infrared LED Circuit Parts List

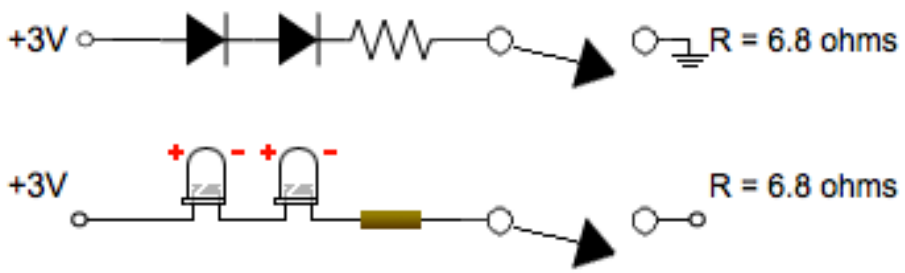

Figure 5.5: Infrared LED Circuit Design: A minimum 6.8 ohms resistor is required [2].

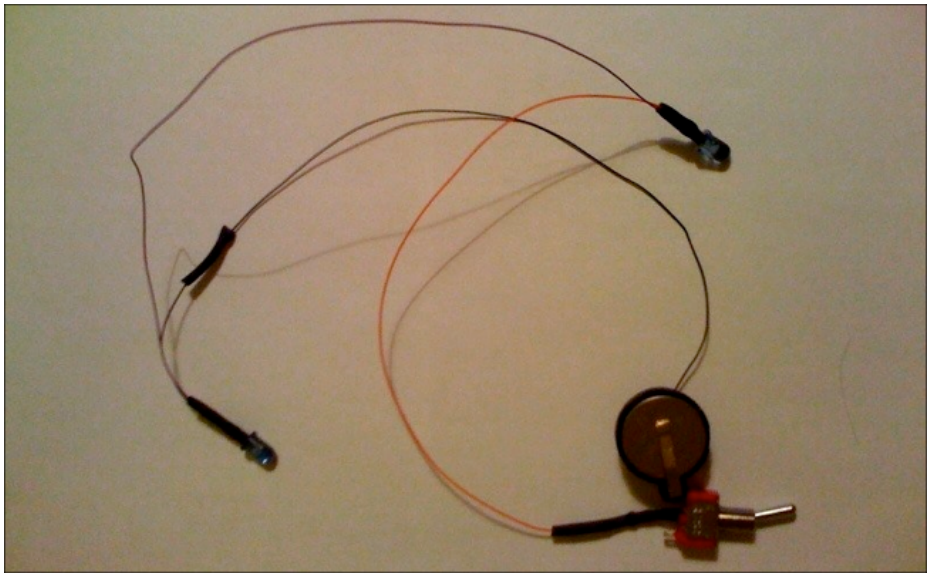

Figure 5.6: The completed Infrared LED Circuit: Assembled with two Infrared LEDs, a 10 ohm resistor, a watch battery and a simple switch.

5.4.2 User Glove Design

For the actual wearable tracking gloves, we first obtained traditional baseball batting gloves from a sporting goods store. We use electrical tape to mount the infrared circuit in Figure 5.6 to the batting gloves. During our initial design phase, we planned to implement a haptic feedback device in our system. Hence, we removed the three end fingers from our baseball batting glove such that only the index and thumb are covered. We were aiming to give the user as much haptic touch feedback as possible. Since our current system does not have a haptic component, these removed finger slots did not have the desired effect.

Figure 5.7: Infrared LEDs mounted on User Gloves

5.4.3 Wiimote Hardware and Interface

As aforementioned in previous sections, we use Nintendo's Wiimote as our user hand tracking interface. The Wiimote features a fully embedded infrared camera, accelerometer and force feedback generator all controllable via a Bluetooth interface. Bluetooth is a wireless communication technology that uses short radio waves to exchange data over smaller networks [58]. For our Holovee application, we are only concerned with using the Wiimote's infrared camera because it will be tracking moving infrared LEDs on our users' hands. Moreover, since the Wiimote will remain stationary, the accelerometer and force feedback generators are not applicable. The authors of [26] give a very detailed overview of the Wii's low level hardware commands API. However, since we are working with higher level application code, we use Brian Peek's premade C# Wiimote library to interface with the device [29].

To use the Bluetooth-enabled Wiimote in our system, we must use a Bluetooth-enabled computer to wirelessly interface to the device. To satisfy this requirement, we our test

computer's built-in Bluetooth adapter. Since we are working within a close proximity, we need not be concerned with Bluetooth distance issues. If larger setups, Class 1 Bluetooth devices are more viable as they have a maximum range of 100 meters and are the highest classified range [58]. In addition, the Wiimote interface library handles all lower level Bluetooth stack management calls. Moreover, our application code need only be concerned with reading infrared LED input tracks and applying that data to user interface interactions. The Wiimote can track up to four possible infrared LED points.

5.4.4 Other Available Hardware Options

A missing feature of our infrared tracking gloves is the ability to click or pinch. Surely, our speech recognition component is a sufficient substitute for opening photo albums and or issuing commands. However, a much more intuitive mechanism would be to mount a simple push button onto the user's tracking gloves. In our search for a simple push button, we researched a company called Phidgets [42]. Phidgets manufactures a plethora of plug-and-play USB components which have stable high level programming interfaces for easy manipulation. The only downside to Phidgets is that they are only offered as wired USB interfaces. Our infrared gloves are meant to be both comfortable and able to be moved freely. By adding a wired component to our device, we are limited the user to be within wired distance of a computer to utilize a simple push button mechanism. A more viable solution would be to find a wireless simple push button solution that could mount to our user's gloves. Moreover, another solution may be to mount additional infrared LEDs on the user's glove thumbs. Since our Wiimote can simultaneously track up to four points, we could develop an algorithm which detects when a user's index and thumb fingers perform a pinching motion.

5.5 Summary

In this section, we gave an overview of futuristic interaction technologies and detailed our holographic user interface's input tracking mechanism.

- Input devices are an essential component to any user interface. With the evolution of futuristic user interfaces, input devices which give users more degrees of freedom (DOF) are favored because they are more tailored for 3D tasks.

- Infrared cameras are a popular, cheap mechanism for tracking user input via user-worn infrared LEDs. Markerless tracking approaches are more complex to build, but are favored because they require users to wear virtually no tracking sensors.

- Inside-out and outside-in are the two main methods exist for tracking user input. An inside-out involves moving users wearing sensors and being tracked by stationary equipment. An outside-in approach involves the user not wearing any equipment and tracking sensors being mounted around the user's surrounding environment. Inside-out tracking systems require less computing power and are generally easy to build.

- Voice control and speech recognition are exciting, futuristic mechanisms for working with user interfaces. The field has great potential and can be realized by using existing speech parsing APIs.

- Using simple logic circuits and pre-made device interface libraries proved very effective for our system. It was much easier to use existing code bases to read Wiimote tracking data than implementing lower level code from scratch.

Chapter 6

3D Displays

6.1 Basic Optics

6.1.1 Eye Perception

When creating virtual images, we strive to trick the human mind into believing our computer-generated creations are realistic. Before doing so, it is useful to have a high level understanding of lens optics. With this optics knowledge, we can feed into a viewer's sense of space and dimension using our holographic display. Thinking about standard photograph and digital cameras, our camera's lenses convert 3D data into 2D data. Hence, during this conversion process, data is lost. Assuming we are not using a holographic image recorder, simple photographs lose 3D data which often cannot be preserved [17]. While 2D images are sufficient for most imaging applications, our perception is more engaged with a 3D scene representation. As mentioned in previous sections, we need to provide users with 3D images so that their mind's can better process scene representations.

From a 2D scene representation, our brain's may be able to process simple features such as background distance or object occlusion, but complicated features such as motion may be more difficult to perceive. Figure 6.1 shows a basic representation of a human's optics system. When we view an object, our eye's lens projects the object image onto our eye's sensor. Each part of our eye's lens captures a different perspective of the viewing scene [17]. Moreover, these multi-perspective lens images align at the user's eye sensors. However, when considering 3D holographic displays, we need to consider preserving or simulating those multi-prespective viewpoints.

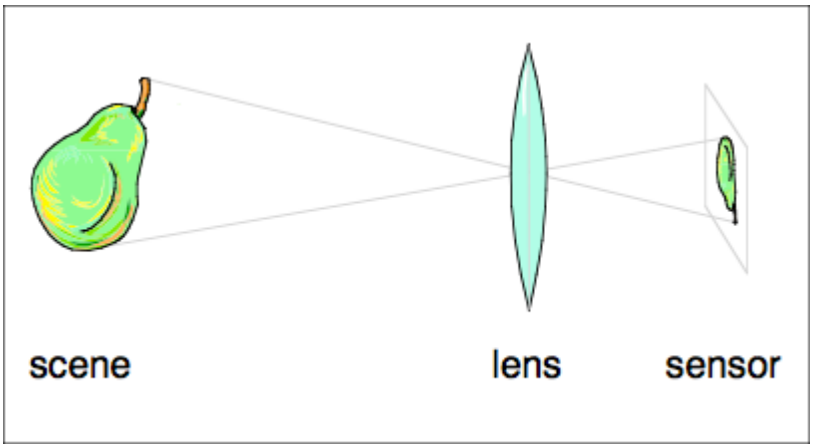

Figure 6.1: Human Optics System [17]

6.1.2 Light Travel

As light hits our eye, we are able to determine how virtual images are formed. Basically, light travels around a scene object, then diverges towards our eye which then uses visual cues to perceive depth. When working with holographic displays, lighting arrangements are an important consideration for proper viewing conditions. As seen in projector-driven holographic displays such as [32], [48], [22] and [53], room conditions must be dimly lit to see sharp, crisp images. Thinking about standard projectors, this lighting observation is often correct. When sitting in a lighted room, a projected image on even a reflective screen is often difficult to view. This viewing artifact can be attributed to the projector's brightness, measured in lumens. A lumen is the amount of measured light perceived by the human eye [60]. The brighter the projector's output, the better results we can observe.

Most commercial overhead projectors output between 2000 and 2500 lumens. Unfortunately, our 3M MPro120 micro-projector only outputs 12 lumens in its highest brightness mode. Again, we chose our micro-projector because of its size and easy portability. We are more concerned with setting up a smaller package rather than optimal lighting conditions. Regardless of output lumens, we understand room lighting will be still be a factor. We will accommodate these viewing constraints and offset them by requiring dimmer rooms for our system. No matter our brightness constraints, we must ensure our eye is on the same line as the projecting source. Or more simply, there is a defined safe field-of-view range where

users can view virtual scenes when moving side-to-side [17].

6.1.3 Visual Depth Cues

When viewing virtual images, our mind uses various depth cues to understand a scene's composition. Visual depth cues allow our brain to process image data and perceive 3D shapes. As discussed by the authors in [17], there are three major depth cues which our eye considers when analyzing an image. First, accommodation, or eye focus, allows viewers to bring image rays into focus. While accommodation is good for depth perception, it is a weak cue because it is only successful for short range scenes. For example, our eyes would have a difficult time using accommodation to determine depth with object's positioned at far distances. Second, stereopsis, uses both eyes to create a depth cue which is powerful at an arm's length. Using both eyes, we can converge our eyes on scene objects to determine where they reside. Finally, motion parallax involves observing object movement to perceive depth information. In a physical scene, we observe motion parallax when we move around a scene or an object moves around us. Because viewers have the control to moving around, motion parallax is the strongest visual depth cue. In addition, as stated in [64], our depth perception is only good out to 200 yards. Hence, even an object is moving, if it is farther than 200 yards away, it will be difficult to distinguish its depth from neighboring objects.

While simple to implement, the process of correctly constructing stereoscopic 3D images is arduous. With stereoscopic images, we want our viewers to feel they can touch images, but must proceed with caution to prevent undesired eye fatigue and motion sickness. In addition, the author in [64] discusses various mechanisms for successfully creating a stereoscopic scene. To accommodate these needs, we want to tastefully integrate the 3D effect into a virtual environment. Hence, we never want to have a user's eyes converge or diverge at extremes.

First, we should analyze how the stereoscopic image will be presented to the user and their space existence. In our system, the user will normally be positioned within arm's reach or approximately 3 feet away from the holographic display. In our virtual avatar creator application, the character and user interface buttons will appear to have depth because

of the stereoscopic effect. Moreover, we do not want to stress the user's eyes and cause fatigue. Second, the author in [64] suggests focusing the stereoscopic camera on single objects. For example, if the user is bombarded with multiple 3D objects jumping out of the screen, they will feel disoriented and look in too many directions. Finally, any user 3D stereoscopic effects should control their motion speed. With accommodation being a short range visual depth cue and our holographic display being close to users, we should slow down any sudden motion movements. If a holographic display's elements move too fast, users may perceive the same image twice rather than a smooth interpolation of object movement.

6.2 3D Display Types

With an understanding of how our eye operates and perceives virtual images, we can now examine how creative 3D displays can be applied to produce optical illusions. In the related work section, we gave a high level overview of several 3D display technologies which can be used for futuristic user interfaces. While very appealing, it is useful to understand the science behind these displays to better realize their full potential. Most holographic display systems can be classified into either a volumetric or parallax category.

6.2.1 Volumetric Displays

Volumetric solutions are enclosed displays where each voxel location emits light rays to produce a reconstructed image [17]. Figure 6.2 shows an advanced spinning mirror representation of a volumetric display used by USC researchers in [24]. To better visualize a volumetric display, envision an enclosed cube structure where an image is projected within the unit. The 3D image exists within the context of the bounding display's volume and the 3D image can only be as large as the bounding volume which contains it. Volumetric holographic displays such as [48] and [53] use commercial projectors and special optics to create holographic illusions. These optical effects are simple to understand and rely on laws of reflection and refraction.

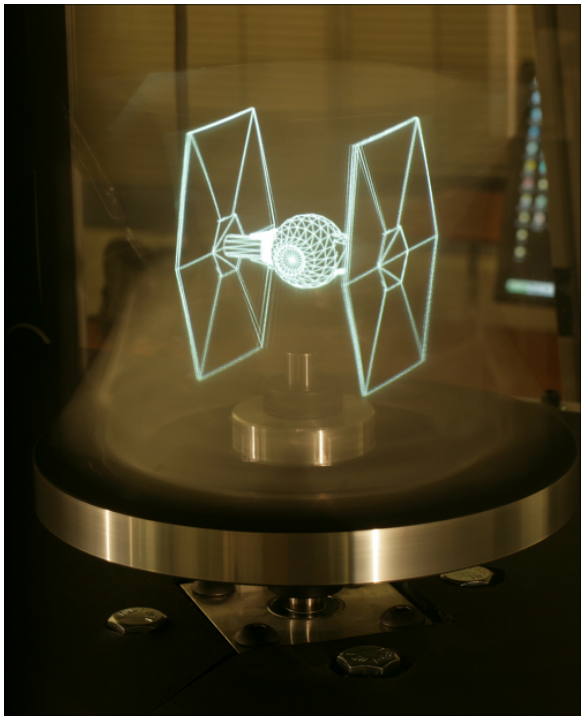

Figure 6.2: 3D Volumetric Display [24]

In the scientific realm, we use Snell's law to describe how light refracts or deflects from specified surfaces depending on the surface's index of refraction and the light ray's angle of entry [63]. Similarly, the law of reflection states that a light ray entering a perfectly mirrored surface will leave the surface with exactly the same angle. Refraction and reflection are important concepts for creating optical illusions for holographic imagery. For example, as discussed in the related work section, the Pepper's Ghost effect uses traditional optics to create the illusion of disappearing and reappearing phantom images. The effect comes to fruition by reflecting a lighted image source at an angled sheet of glass. As seen Figure 6.4, as light enters the angled sheet of glass, the resulting image is refracted due to the glass' index of refraction and tilt angle. The audience is effectively seeing a reflection from the image source's light. However, it should be noted that Pepper's Ghost effect only is visible from front-facing viewing angles and does accommodate for side image profiles. To expand Pepper's Ghost to multiple viewing angles, several projectors and sheets of angled

glass are added to create an volumetric illusion as seen in [48]. For our holographic display, we experimented with variations of Pepper's Ghost effect, but felt it to be insufficient for hosting our user interface. Mainly, the effect required that users be positioned at exact view angles and would have had unfavorable results for interacting with our user interface. Moreover, Pepper's Ghost driven applications are better targeted towards large audiences and stages because of its simple nature.

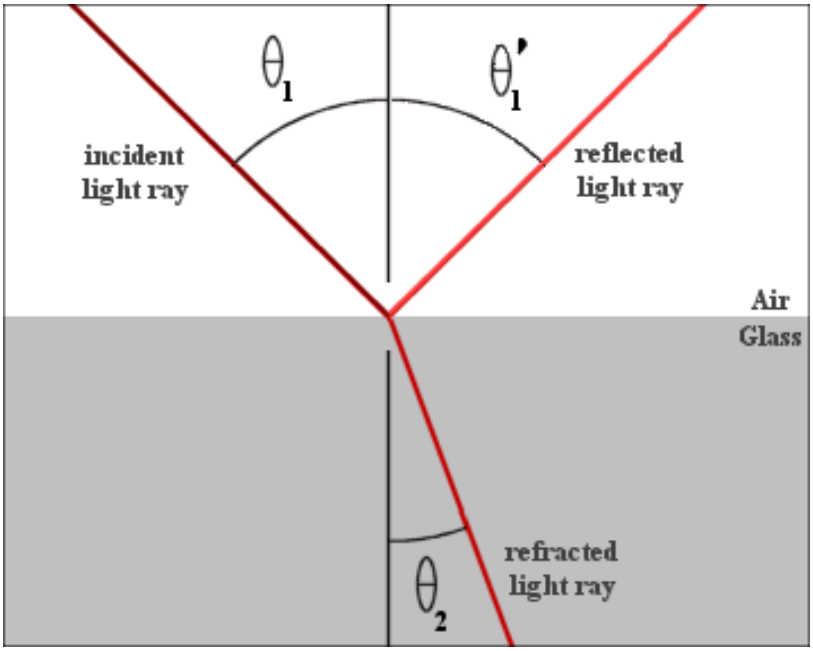

Figure 6.3: Refraction Properties via Snell's Law [56]

Another approach for achieving a volumetric display is to project images onto a rotating, time-swept spinning mirror to create the illusion true volumetric images. The researchers in [24] created a cutting-edge 360-degree volumetric display using a high speed projector and a rapidly rotating mirror. The volumetric display does not require any special viewing glasses and can be viewed from 360 degrees. The researchers state that the system can be made of low-cost components and provide high quality viewing experiences. While highly attractive, the main drawback for our application is the rotating mirror. Our users will need to directly interact with the holographic surface and their user interface interactions cannot be constrained by moving parts.

As discussed, most volumetric displays use some form of projection medium which

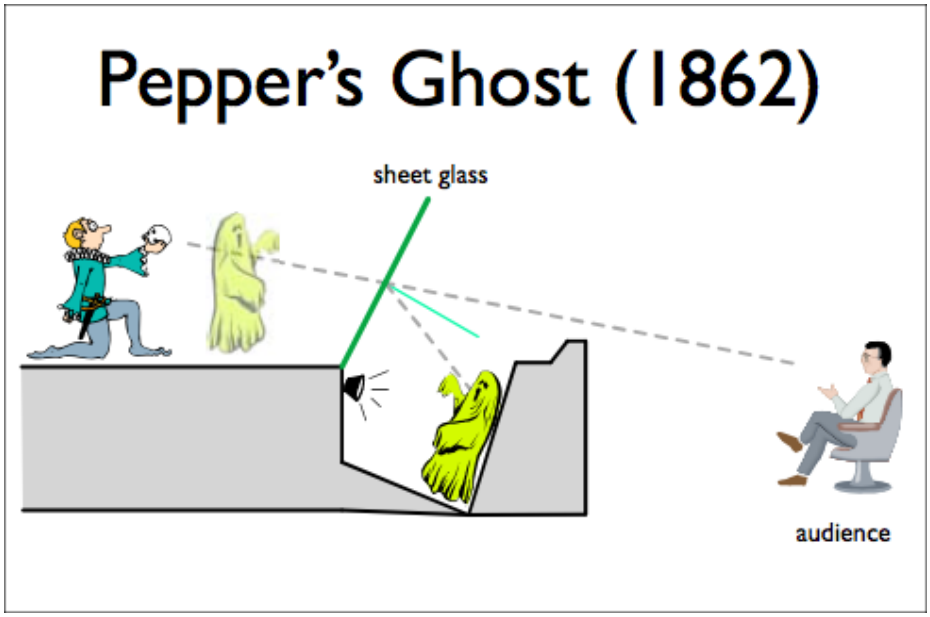

Figure 6.4: Pepper's Ghost effect using Traditional Optics effects [50]

cause the display's voxels to emit light [17]. In modern holographic displays, volumetric approaches are the most prevalent as their technology is widely available. The biggest challenge is finding a display medium for system projectors to project onto. The authors in [17] consider air and fog display mediums to be bad choices for display mediums because of their unpredictability. However, the air and fog display technology presented in [22] and [44] have shown great results because the transparent medium is regulated in a controlled vertical fashion. Each time the systems were run, we can predict the projected image will be displayed in a consistent manner. For our holographic display, we strive for the same consistency.

6.2.2 Parallax Displays

Parallax displays are represented as surfaces where each surface element emits light rays in multiple directions [17]. True parallax surface displays are considered the holy grail of holographic displays. Envision a holographic display unit which has fewer bounding volume constraints and projects outward to produce an arbitrary-sized 3D image. There is not any need for a large enclosed holographic display unit and or managing how that

display will fit into the surrounding space. Discrete pixel emitters usually reside on display surface and project outward to form a free-floating 3D image. Figure 6.5 shows a simple parallax display.

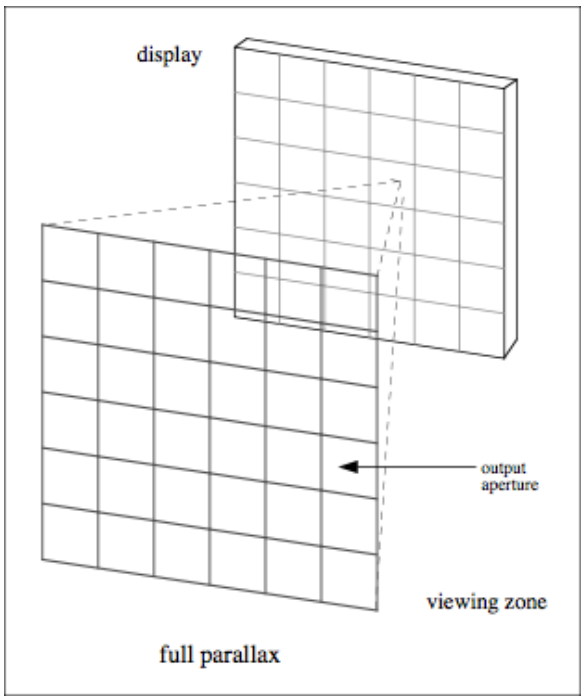

Figure 6.5: Futuristic Parallax Discrete Pixel Display [17]

Parallax displays can be represented using horizontal and or vertical light emitters. When our parallax display emits light in vertical columns patterns, our eye perceives horizontal parallax, or the ability to move left-to-right and see light from each surface element. Similarly, if our parallax display emits light in horizontal column patterns, our eye perceives vertical parallax, or the ability to move up and down and see light from each surface element. As discussed in [17], full parallax more difficult to accomplish because it requires n-squared as many image pixel samples as horizontal parallax alone. This complexity comparison can be attributed to the need for more viewing data to achieve vertical parallax when the user moves their head up and down. Basically, with only horizontal parallax, the holographic display would only appear favorable from fixed distances as vertical 3D image details are excluded from the display.

Again, we are less concerned with parallax displays because of their increased complexity and because our system does not use this technology. However, it is very useful to understand where the future of holographic displays can evolve once the appropriate technology is available.

6.3 3D Stereoscopy

In the past few years, movies featuring 3D stereoscopic content have been experiencing a strong resurgence in the entertainment industry. This strong resurgence can be attributed to a growing need for more cutting edge, high quality entertainment techniques. While 3D stereoscopic technology are exciting, the technology has existed for well over 150 years [57]. As discussed in the previous visual depth cues section, we gauge the world through both eyes and are able perceive depth. Moreover, if we present users with two slightly differently positioned images the viewer's brain can reconstruct the scene into a 3D image [66].

6.3.1 Passive Red-Blue Anaglyphs

In an ideal holographic display system, we want to the user to be as unrestricted as possible. Displays that require special glasses are considered more restricted because they require an additional supplement to view the 3D effect. Hence, an autostereoscopic display which doesn't require special viewing glasses is more desired, but is usually equated with a higher cost. In the future, for our application's display, we could users with a simple stereoscopic effect that can be realized by wearing traditional 3D red-blue viewing glasses. Combined with our system's free-floating image display, this 3D effect could give users increased depth awareness in the user interface.

Most often, a simple 3D stereoscopic effect can be created by an anaglyph. Most movie goers may be familiar with an anaglyph as it is the traditional red, blue filtering effect that leads to a perceived 3D depth perception. As seen in Figure 6.6, when viewed without special 3D glasses, the anaglyph appears to be a mix of a red color layer, a blue color layer

and a main centered image. An anaglyph's end goal is to provide its viewers with an image for each eye. Normally, the left eye's image is presented as the red layer and the right eye's image is presented as the blue layer [66].

Figure 6.6: Traditional 3D Anaglyph

The user is given viewing glasses with red and blue filters which offset their opposite color layers. These 3D viewing glasses will allow the human brain to merge the red and blue layers into a perceived 3D image. Because each eye perceives a slightly different image, the human brain thinks the image has depth cues [57]. Most often, the distance between a 3D anaglyph's red and blue layers is about 10 centimeters or the average distance between a human's eyes. It is very simple to recreate the red-blue viewing glasses approach because as it has limited setup time.

6.3.2 Passive Polarization

Another passive, more complex 3D stereoscopy approach uses polarized glasses to produce a similar 3D depth effect. As discussed in [66], the authors state a better stereoscopic effect can be created by using polarized viewing glasses. Polarization is the process of restricting the light that reaches a viewer's eyes [62]. Polarization can often be used to

create a 3D stereoscopic effect by projecting two images through different polarizing filters [62]. Figure 6.7 shows how a typical polarization filter can restrict light. Similar to the aforementioned red-blue anaglyph method, viewers can wear special glasses encompassing polarization filters to view the constructed scene and perceive a high quality, true color stereoscopic effect.

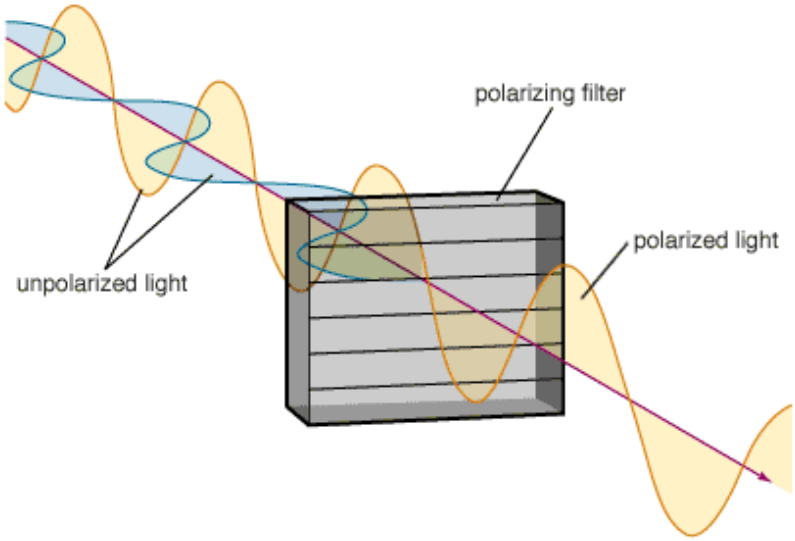

Figure 6.7: Polarization Display filtering Unpolarized Light [8]

To accommodate avid display enthusiasts, the authors in [34] created a do-it-yourself guide for creating a 3D stereoscopic enclosed theater. The system, dubbed Vizard, features two LCD screens, a square two-way mirror, a computer with two graphics cards and a stereoscopic movie player. The entire system is packaged into an enclosure which holds the hardware. Remember, when viewed with special glasses, the enclosure is trying to reproduce a stereoscopic effect. One LCD screen faces the user and the other faces downward to create a 90-degree angle with the first screen. Next, the silvered transparent mirror is positioned at a 45-degree angle between the two displays. The suggested stereoscopic movie playing software creates two slightly different images from a supplied video. When operational, the stereoscopic movie is reflected onto the display mirror to produce a 3D optical effect as seen in modern day movie theaters.

While the described polarization methods produces excellent results, it requires more

expensive equipment and greater setup time. Because a passive polarization setup requires dual projected images, system designers must use two projectors or display screens. Hence, the burden of setting projectors the correct distance apart relies on the system designer. Conversely, the red-blue anaglyph method can be achieved using simple shading effects using the Microsoft XNA gaming framework. The red-blue 3D stereoscopic effect is not concerned with preserving true color in images. In addition, because a stereoscopic display requires two separate images, our system's display would need at least two graphics output ports. Most mobile computers, such as the one that will drive our system's display, only contain one graphics card.

6.3.3 Active Shutter

A final option for viewing 3D stereoscopic effects is to use active shutter viewing glasses. Active shutter glasses provide users with two rapidly shuttering eye viewports. The viewing glasses are synchronized with the polarized display and give viewers a depth illusion [66]. While active shutter glasses are highly attractive, they are usually very expensive and require high refresh rates to prevent flickered images. In addition, for every viewer of the polarized display, each would require a pair of synchronized active shutter glasses to view the output. It is much easier to provide users with less expensive red-blue anaglyph or cheap polarization filter glasses to view a scene.

Another active stereoscopic display approach features a head mounted display. Vuzix, a Rochester-based head mounted display company, has developed the VR920 virtual reality headset [35]. Different from bulky virtual reality headsets, the VR920 headset more closely resembles traditional stereoscopic viewing glasses. Different from active shutter glasses, the VR920 headset embeds two eye-sized displays directly into the hardware unit. The VR920 headset connects to a computer graphic card and outputs the display signal to the two eye displays. When using Vuzix's programming API, developers can create stereoscopic programs for the head mounted display. During runtime, the software application rapidly alternates the left and right images on the VR920's headset which allows the user to perceive a 3D scene. First-hand experience results with VR920 head mounted display were

favorable, but they seemed to produce eye fatigue and dizziness after prolonged wearing periods.

6.4 System Display

With knowledge of the various forms of holographic displays, we can detail our hybrid approach which will display Holovee, our social networking management application.

6.4.1 Ultrasonic Water Fogger

As aforementioned, we have created a transparent, water-driven display similar to [22] and [15]. Today's commercial projectors work by shooting light rays onto a reflective display surface. The projector uses an internal lens system which converts a digital video signal into light ray. With today's commercial projectors, we cannot project images into mid-air without a display medium. This is the reason light can reach a reflective screen and not be interrupted by air. By itself, air is transparent and is less dense than the light that is passing through it [54]. This characteristic allows light to not be refracted and not cause the projected image to be distorted. Conversely, if we were to project our image into water, Snell's law dictates the image would be refracted because water is more dense than light. Thinking about water's continually erratic movement, it is not a stable projection surface. Instead, if we were to control water and produce a steady stream, we could produce a more stable projection surface. For our display, water streams are not a good display surface because we need be concerned with users touching a wet surface and the subsequent mess it would leave behind.

With that in mind, we researched ultrasonic water foggers which produced little mess, quickly evaporates and can be controlled to produce a stable, transparent display surface. Ultrasonic water foggers are commonly used in reptile tanks and home gardens to produce a very granular water vapor. They run on regular tap water and will keep producing water fog until the water runs out. Each ultrasonic fogger jet uses high energy ultrasonic waves to turn water into a fine dry-feeling fog [9]. As seen Figure 6.8, our ultrasonic water fogger

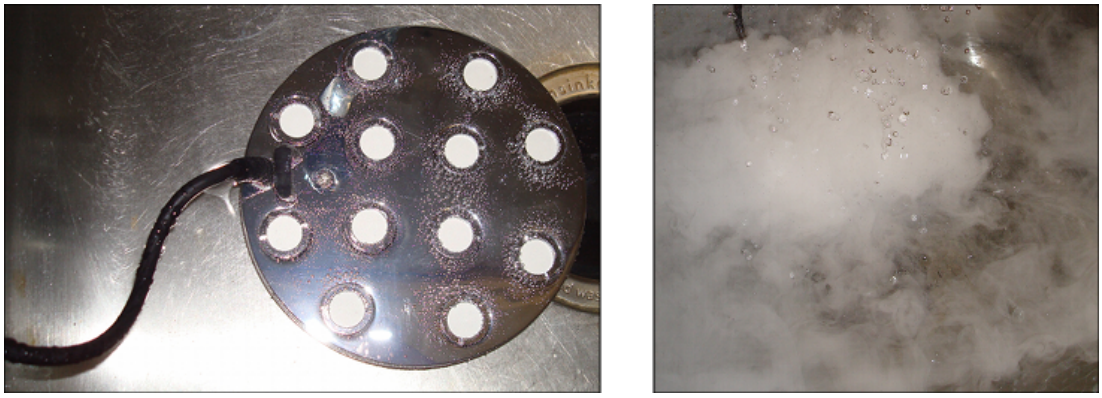

Figure 6.8: 12-Jet Ultrasonic Water Fogger

is submerged in water and produces fog. On the ultrasonic fogger, there is a sensor which determines if the device is submerged enough to produce fog. For example, if we place the ultrasonic fogger in too shallow water, it will not function. These larger scale ultrasonic water foggers are available via [11]. We also considered Halloween fog machines, but found they were not ideal because they require fog juice, a fogger heat-up delay and leave our surrounding environment with a smoke-filled mess that doesn't quickly evaporate.

6.4.2 Slim Air Fan Flow

Before building the custom water tank which houses our ultrasonic water fogger, we experimented with directing the fog flow in different directions. Again, for our application, we want to direct our fog stream upward to create a vertical projection surface. One air stream solution to direct the fog upward is to let the fog drift upward and then sandwich it between two outer air streams. By doing so, we are steadily controlling the fog stream by ensuring it does not deviate from inside its air stream boundaries. A simpler air stream solution is to use a commercial slim air fan. Figure 6.9 show the commercial slim air fan which we used for controlling ultrasonic water fogger. We lay this fan horizontally to get the vertical desired air flow effect. The back of the Honeywell fan intakes air and outputs it in a narrow stream fashion. We leverage this fan's flow property to intake our ultrasonic fogger output and have it feed through. During our testing, we found that feeding fog through our fan produced a very consistent fog stream which could be projected upon. Figure 6.10

shows our initial testing results with our slim air fan and ultrasonic water fogger. Our 3M MPro120 micro projector is positioned behind the fog screen display and is displaying a default image. At this stage, we did not have a custom water tank and were using our sink as the water source. In addition, the airflow between the fan's intake and the ultrasonic fogger was not sealed. Hence, we can see how the fog gravitates towards the fan's center and is not equally distributed. In the next section, we discuss our custom water tank which produces better flow and fog distribution results.

Figure 6.9: Honeywell Tower Air Fan

6.4.3 Custom Water Tank

Our 12-jet ultrasonic fogger can work through 6500 milliliters of water per hour [11]. For our display to be viable, we must provide a sufficient water source and flow control which can fuel our fogger for demonstrations. First, let's begin with our water tank. As we discussed, our ultrasonic water fogger must be completely submerged in water. Moreover, our ultrasonic water fogger is 6 inches in diameter. Hence, we build our custom water tank such that the ultrasonic fogger can safely fit. By creating an enclosed tank to hold our fogger and water, excess fog cannot escape into the room. This important enclosure feature

Figure 6.10: Initial Projection with Ultrasonic Fogger and Honeywell Tower Air Fan

ensures the most possible fog is suctioned into the slim air fan's intake.

The second and most important custom water tank component is its vent. We want our produced water fog to be tightly sealed and evenly sealed into the fan's intake. Our Honeywell slim air fan's intake is 20.5 inches by 3 inches. To accommodate this, we created 20.5 inches by 3 inches opening at the top of our custom water tank. During operation, the slim air fan sits on top of the water tank and is sealed to the opening. This fan positioning will create a secure vacuum for the water fog to flow through. Moreover, our transparent fog screen display has a final width close to 20.5 inches, matching that of the Honeywell slim air fan's output vent.

Finally, as seen in Figure 6.11, we must created a 7 inch by 7 inch square opening in the top of the custom water tank. Doing so, we can easily submerge the ultrasonic water fogger with 6 inch diameter into the tank. Also, in our experimentation, we found that having this opening allowed for better air flow and an overall better transparent fog screen. Closing off the 7 inch by 7 inch square opening led to narrower display screens and unevenly distributed

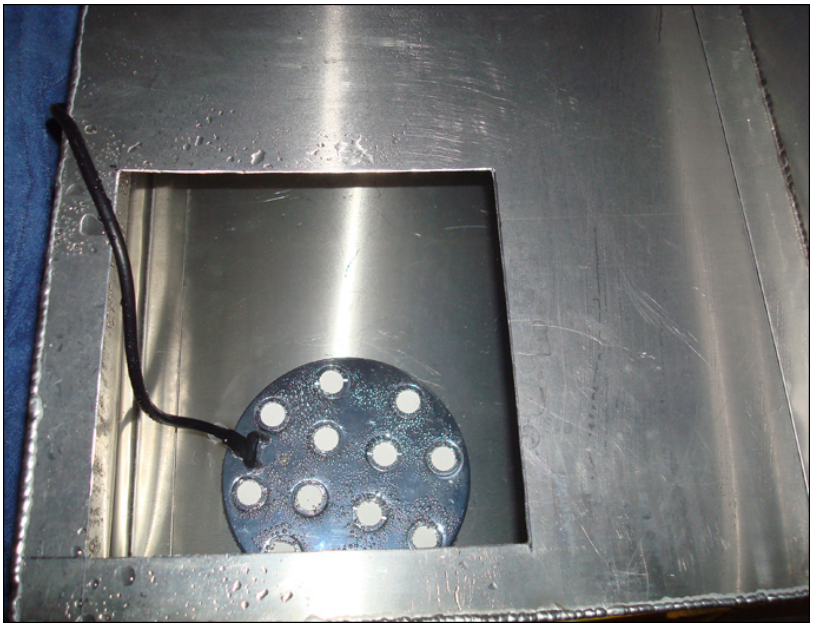

Figure 6.11: Custom Water Tank Air Intake Opening

fog. Moreover, we found positioning the ultrasonic water fogger towards the tank's rear and directly underneath the square opening produced the best results. We hypothesize that the slim air fan needs sufficient air flow to produce a favorable display. Basically, having the ultrasonic water underneath the vent allows air to flow into the tank and be distributed to the opposite end. When positioning the fogger at the end without the opening, we found that the transparent display was diminished as the fog was unbalanced. Figure 6.14 shows a high-level drawing of our custom water tank's design.

6.4.4 Projector and Mirror Positioning

For Holovee's operation, our 3M MPro120 micro projector and Nintendo Wiimote are positioned behind the transparent fog screen display. We attach the projector and Wiimote to separate, adjustable mounting brackets. The optimal position for the Wiimote device is centered between the ends of the slim air fan and its height adjusted to halfway between the fog screen's base and it's height. As made evident in Figure 6.10, we can see that directing our projector directly into the transparent fog screen display produces a reversed image. From a user standpoint, this is not optimal because we want users to see a correct image

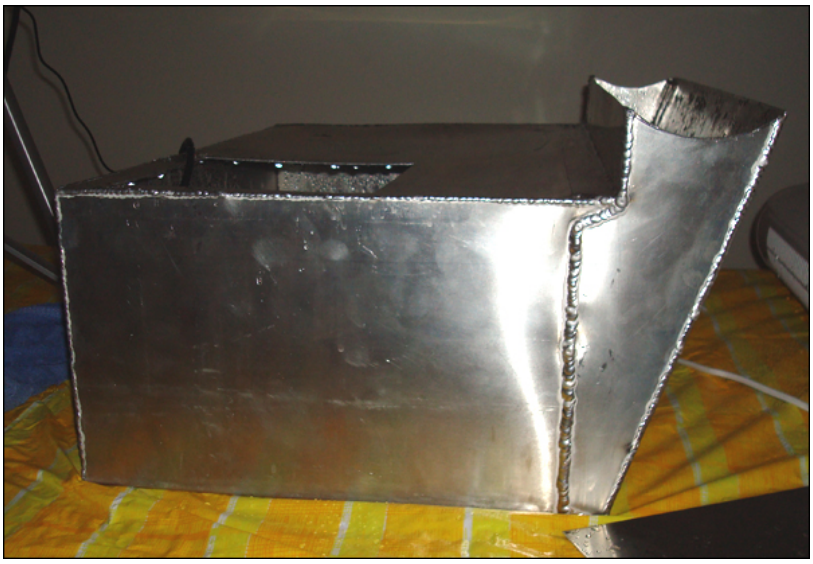

Figure 6.12: Custom Water Tank Side Profile

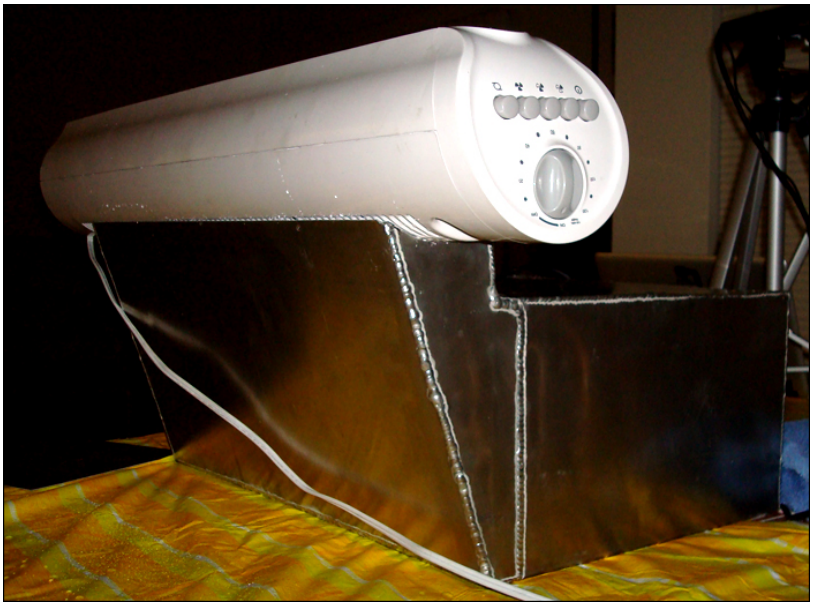

Figure 6.13: Custom Water Tank with Fan

orientation. Because our system hardware does not let us easily reverse Holovee's output image, we use a simple acrylic mirror to flip the image before it reaches the display. Hence, we aim the projector at a small section of acrylic mirror and have the mirror reflect back into the transparent fog screen display.

As seen in Figure 6.15, the micro projector and mirror setup involves sufficient setup

time to achieve favorable results. Hence, as shown in Figure 6.17, we use semi-transmissive, acrylic rear-projection sheet material to correctly position our micro projector. Since our acrylic rear-projection sheet material is about the same dimensions as our desired fog screen display, we can use it to visualize a brighter, crisper image. Then, once we determine the correct micro projector and mirror angle relevant to our acrylic material, we can swap in our custom water tank for the final display setup. Again, because we are projecting onto a fluctuating surface, the results will not be as optimal as the acrylic material. Nonetheless, we can still very clearly visualize specific software application elements with our transparent fog screen display.

6.4.5 Holovee's Final Holographic Display

After successfully positioning our micro projector and mirror, we can add our Wiimote facing the user behind the acrylic mirror. Figure 6.16 shows an example setup scenario with each labeled hardware item. Figures 6.18 and 6.19 show our system's final results while projecting Holovee onto our transparent fog screen display.

6.5 Summary

In this section, we learned that custom 3D displays can create optical illusions which allow the human eye to perceive depth and believe virtual objects are realistic.

- Understanding how the human optics system processes virtual images is very important to creating an effective holographic display. System designers should consider lighting, viewpoint needs and visual depth cues when creating their custom display.
- Volumetric and parallax displays are the two major forms of 3D displays. Volumetric displays are usually enclosed units and more prevalent in modern culture because of their setup simplicity and hardware availability. Parallax displays are usually represented as surfaces and are more desired, but require great complexity and more costly parts.

- 3D stereoscopic effects can be applied to standard displays to increase a user's depth awareness towards virtual options. Traditional methods such as red-blue anaglyphs are very successful at creating stereoscopic effects, but sacrifice true color. More modern stereoscopic methods which use polarization filters or active shutter glasses are preferred because of their high quality viewing results.
- Using off-the-shelf hardware, we were able to create a very appealing transparent display which creates the illusion of free-floating images. A true futuristic holographic display may not require a transparent medium to project onto. But rather, future devices may truly project into mid-air using some new laser light component.

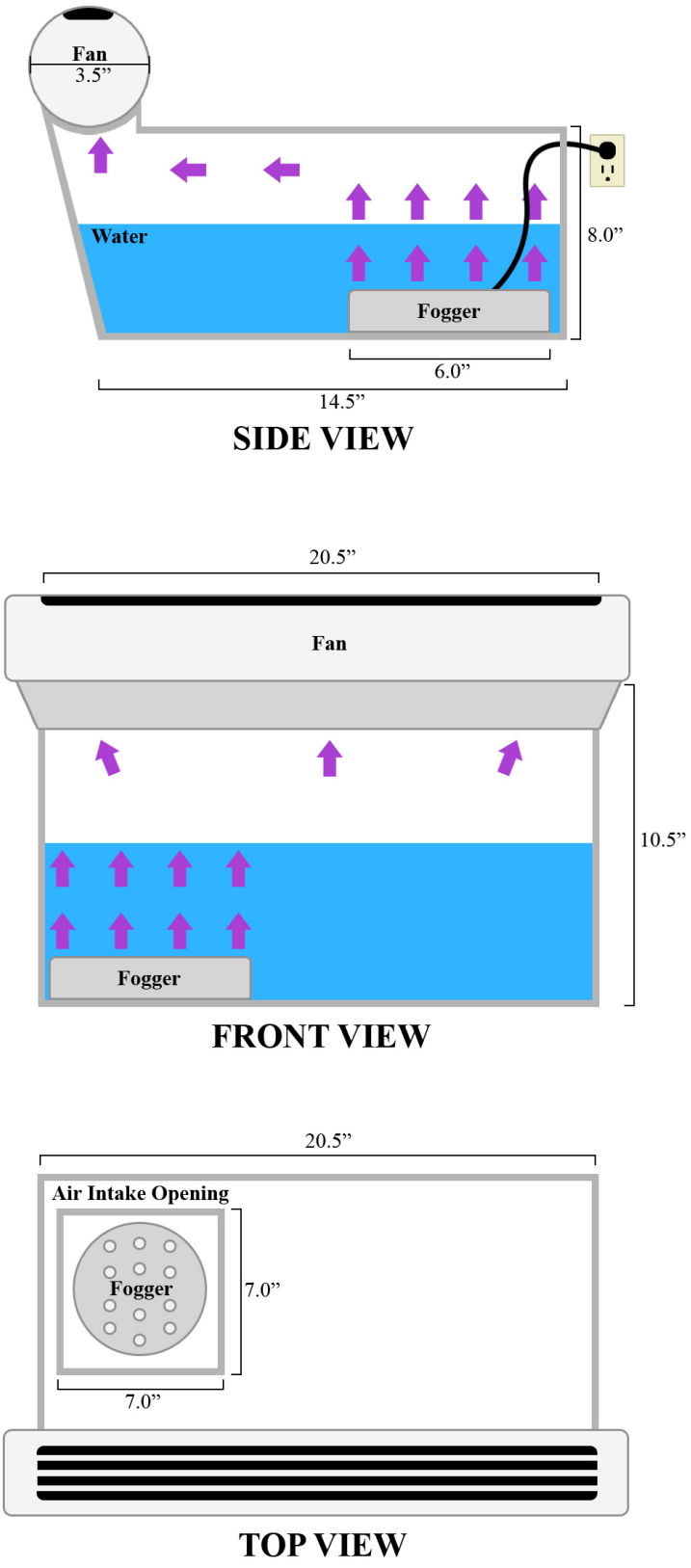

Figure 6.14: Custom Water Tank Design

Figure 6.15: The 3M MPro120 Micro Projector aimed at the Mirror positioned behind our Fog Screen Display

Figure 6.16: Complete System Setup

Figure 6.17: Using an Acrylic Rear Projection Sheet to Position our Projector

Figure 6.18: Removing the Acrylic Rear Project Sheet and Replacing with our Fog Screen

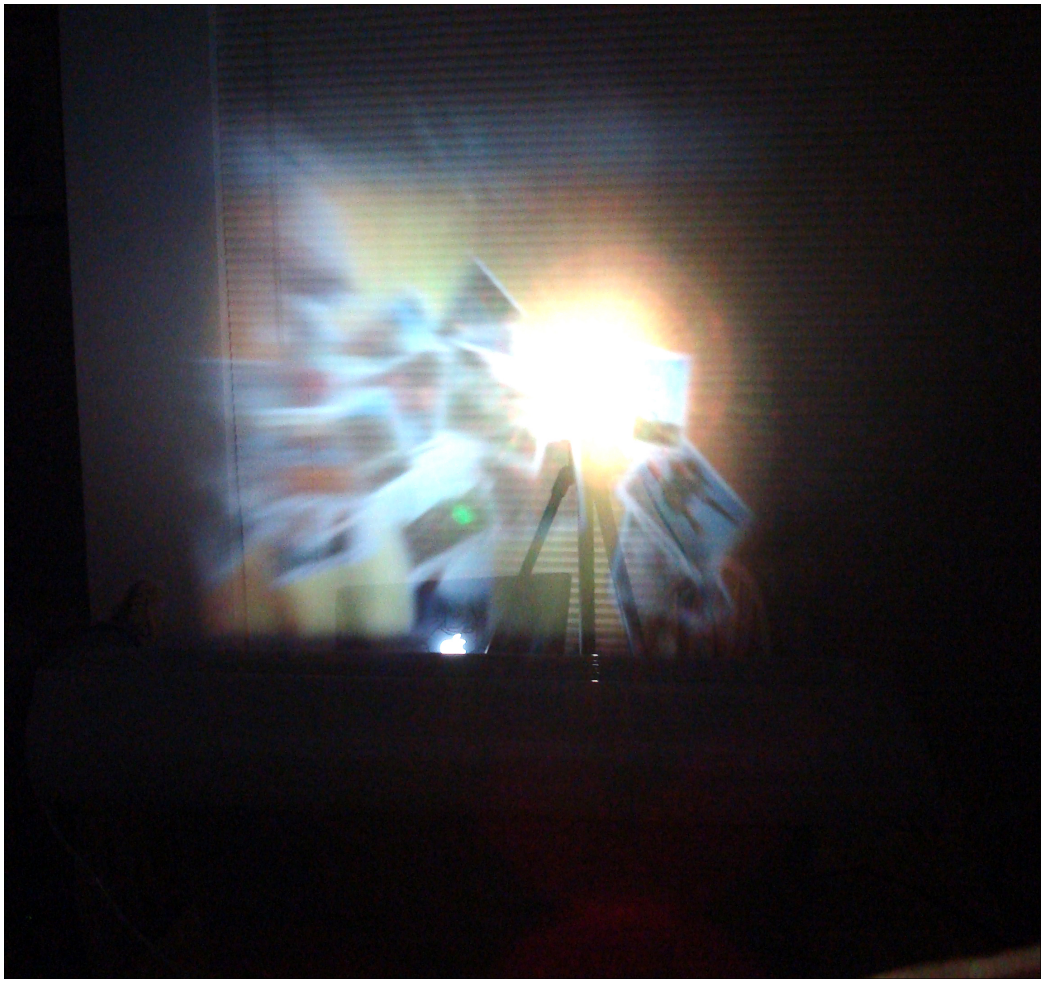

Figure 6.19: Straight Shot of Holovee projected onto our Fog Screen

Chapter 7

Results

7.1 Results

In this section, we provide our system results and compare them against our original system goals. Overall, we were very satisfied with our system results and felt the end product to be very appealing.

7.1.1 Hardware Availability

With our system, our initial goal was to make use of current state-of-the-art holographic displays, interaction tracking and touch feedback devices. As discussed in our Introduction chapter, we aimed to accomplish this goal by using commodity, off-the-shelf hardware components which could easily be constructed into a usable system. Based on our research and limited budget, we understand such things as brighter projectors, haptic feedback components and larger ultrasonic water foggers were less obtainable. Similarly, complete non-commodity products such as the Heliodisplay use proprietary components to create a transparent display screen and are offered at costs between $39,000 and $66,000 [22]. Granted, the Heliodisplay unit is offered in much larger sizes and more commercially polished; however, we were able to construct our complete, very appealing system for under $600 and with some very hard work. We feel our work's practical nature to be reproduced by potential readers is a great attribute and we are excited to see new project derivations. We understand our system could have been improved with additional ultrasonic water foggers for denser display screens and or brighter projectors for crisper images. However, our

underlying goal was to simply demonstrate that a futuristic system could be built from commodity hardware.

7.1.2 Believability

Our system's second goal was to consider valuable human-computer interaction factors and be ergonomically friendly. In our Interactive 3D User Interfaces chapter, we discussed several well-known best practices for developing high quality, intuitive user interfaces. Within our software application, we applied some, but not all of these principles to Holovee, our social networking management application. For example, we created a simple help tutorial to teach users our system's hand gesture interactions and speech commands. However, a more complete solution would have been a completely integrated help tutorial which enables users to complete basic tasks such as rotating and scaling images before using the application. This type of ad hoc tutorial would have allowed users to learn our system in-place and has been made effective by various modern video games. Moreover, making a believable display which simulates free-floating images was an exciting challenge. We experimented with concave mirrors, acrylic reflection effects and even reflective Mylar coating. While none of these approaches reached our final solution, it was great to learn from experience than simply reading that an approach doesn't work.

7.1.3 Practicality

Finally, we wanted our system to be fun to use and be applicable to a real world computing arena. As aforementioned, we felt Holovee, our social networking management application built on the Facebook platform to be a very effective mechanism for demonstrating the various benefits of a futuristic user interface. Our initial software framework was based on a virtual avatar creator similar as seen in video games such as *The Sims* and *Spore*. However, after early prototyping, we decided a virtual avatar creator application would not be practical for our system. Moreover, we felt that a virtual avatar creator would not allow users to see the full advantage of using infrared tracking gloves and or speech commands.

With that in mind, we understand an important part of developing great software is creating applications that are personal. Personal software can be defined as applications that make users feel connected to the user interface via sentimental artifacts. In Holovee, we present users with very personal management tasks such as managing their photo albums, commenting on their friends' status updates and manipulating social networking data. The great thing about our application is that it is automatically tailored to the currently logged in user and loads their specific Facebook data. Overall, we took advantage of a very popular computing concept, social networking, and successfully applied it to our futuristic user interface.

7.2 Future Work

In this section, we give a very concise overview of proposed work to add future feedback functionality to our holographic display system.

7.2.1 Haptic Feedback

A missing system component which we greatly desired was haptic feedback hardware. Our initial goal was recreate the airborne ultrasonic feedback system discussed in the Related Work section and [23]. Very quickly, we realized that the specific ultrasonic hardware feedback component was not easy to obtain and was at best, a Japanese research prototype. The concept of feeling touch sensations without the need for vibration sensors was highly appealing. More so, we even designed our tracking gloves to be missing three fingers such that the exposed fingers could *feel* this ultrasonic feedback. After realizing this prototype hardware was unavailable, we began searching for commercially available tactile feedback hardware. In particular, we came across Immersion Corporation, a California-based company specializing in various forms of haptic components. In the past, Immersion Corporation has worked with video game, medical equipment and even automotive manufacturers to integrate feedback components into their products. From their website, Immersion Corporation offers an engineering tool kit which allows their components to be

easily programmed. We would hope to mount a haptic feedback component to our infrared tracking gloves and allow users to feel feedback for common user interface interactions. Finally, it would be more optimal embed a simple one-button interface to our gloves for quicker user interface item selection.

7.2.2 3D Stereoscopic Image Viewing

We also discussed the ability to create a 3D stereoscopic effect for greater user depth perception. Although we successfully achieved a display that produces free-floating images, a true multi-viewpoint 3D image would be more optimal. As we mentioned, 3D stereoscopic images can be produced by using slightly different image viewpoints and allowing the user's eyes to merge those images into a perceived 3D scene. Much work has been done in creating 3D stereoscopic scenes on physical displays as researched in [64]. However, little work has been researched for creating stereoscopic scenes on transparent fog screens. Since most stereoscopic displays rely on properly placed image viewpoints, the real challenge would be maintaining a completely stable vertical fog stream to ensure users could view the effect. If the effect is too erratic, users may experience eye fatigue or not be able to realize the effect at all.

7.2.3 Discrete Pixel 3D Holographic Display

In the last section about 3D displays, we hinted that the ultimate nirvana in holographic devices would be a discrete pixel projection surface as discussed in [17]. In our future work, we would like to research the technology required to create such a device. Such hardware might embed futuristic lasers or even distributed light sources that can reconstruct a 3D image. Overall, we would expect this research area to be very costly and also require a greater knowledge of electrical hardware.

7.3 Conclusion and Lessons Learned

In summation, we had an exciting time navigating through this incredible adventure researching interactive holographic user interfaces. Surely, there were several lessons learned from this incredible research. First and foremost, we learned to experiment early and prototype often. Several times, we found ourselves optimistic that a particular researched solution could be easily reproduced. For example, in our early holographic display development stages, we were very adamant about pursuing a concave mirror based display similar to [53]. However, we found very quickly that this holographic display had to be viewed from precise angles, making it less practical for our end system. Similarly, it proved very beneficial to simultaneously write our research paper and also develop our working prototypes. For one, prototyping while researching allowed us to easily test theories and write about our results to prevent unnecessary time wasters.

Before choosing a Masters Thesis topic, we were heavily inspired by science fiction movies and television. The endless futuristic device possibilities we have observed has been our work's motivating factor. From *Star Wars'* original R2D2 holographic Princess Leia projection to *Avatar's* portrayal of scientists interacting with holographic displays on the fictional planet of Pandora, we hope our work inspires potential do-it-yourselfers looking to experimenting with exciting new technologies to push new limits and go beyond believed expectations. Our primary goal was for potential readers to realize that holographic user interfaces are more viable than expected. Although the specified technology may be lagging beyond what is desired, good system architects can create the *illusion* of something exciting. Moreover, we strived to provide readers with a supporting framework for developing a futuristic user interface with components that are available now.

From our Dedication page, when in doubt, we suggest system architects follow Walt Disney's famous quote, "If you can dream it, you can do it."

Bibliography

[1] Jérémie Allard, Clément Menier, Bruno Raffin, Edmond Boyer, and François Faure. Grimage: markerless 3d interactions. In *SIGGRAPH '07: ACM SIGGRAPH 2007 emerging technologies*, page 9, New York, NY, USA, 2007. ACM.

[2] Rob Arnold. LED series parallel array wizard. <http://led.linear1.org/led.wiz>, 2010. [Online; accessed 3-April-2010].

[3] Tibor Balogh, Péter Tamás Kovács, and Zoltán Megyesi. Holovizio 3d display system. In *ImmersCom '07: Proceedings of the First International Conference on Immersive Telecommunications*, pages 1–5, ICST, Brussels, Belgium, Belgium, 2007. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

[4] Woodrow Barfield and Thomas Caudell. *Fundamentals of Wearable Computers and Augumented Reality*. L. Erlbaum Associates Inc., Hillsdale, NJ, USA, 2000.

[5] Oliver Bimber and Ramesh Raskar. Modern approaches to augmented reality. In *SIGGRAPH '07: ACM SIGGRAPH 2007 courses*, page 1, New York, NY, USA, 2007. ACM.

[6] Doug A. Bowman, Ernst Kruijff, Joseph J. LaViola, and Ivan Poupyrev. *3D User Interfaces: Theory and Practice*. Addison Wesley Longman Publishing Co., Inc., Redwood City, CA, USA, 2004.

[7] Derek Bradley, Tiberiu Popa, Alla Sheffer, Wolfgang Heidrich, and Tamy Boubekeur. Markerless garment capture. In *SIGGRAPH '08: ACM SIGGRAPH 2008 papers*, pages 1–9, New York, NY, USA, 2008. ACM.

[8] Encyclopedia Britannica. Encyclopedia britannica. <http://media-2.web.britannica.com/eb-media/90/96590-004-DBAC1219.gif>, 2010. [Online; accessed 20-March-2010].

[9] Buzzle.com. Ultrasonic Fogger: How Does It Work. <http://www.buzzle.com/articles/ultrasonic-fogger-how-does-it-work.html>, 2010. [Online; accessed 9-April-2010].

[10] Clayton M. Christensen. *The Innovator's Dilemma: When New Technologies Cause Great Firms to Fail*. Harvard Business School Press, Boston, 1997.

[11] Mainland Mart Corporation. Ultrasonic Water Fogger-The Mist Maker. <http://www.mainlandmart.com/foggers.html>, 2010. [Online; accessed 7-April-2010].

[12] Philip L. Davidson and Jefferson Y. Han. Extending 2d object arrangement with pressure-sensitive layering cues. In *UIST '08: Proceedings of the 21st annual ACM symposium on User interface software and technology*, pages 87–90, New York, NY, USA, 2008. ACM.

[13] Antonine Education. Welcome to the Antonine Education Website. <http://www.antonine-education.co.uk/physics_gcse/Unit_1/Topic_5/em_spectrum.jpg>, 2010. [Online; accessed 13-March-2010].

[14] Facebook. API - Facebook Developer Wiki. <http://wiki.developers.facebook.com/index.php/API>, 2010. [Online; accessed 11-April-2010].

[15] Inc. FogScreen. Fogscreen - Welcome to www.FogScreen.com! <http://www.fogscreen.com/en/home/>, 2010. [Online; accessed 7-April-2010].

[16] Tovi Grossman, Daniel Wigdor, and Ravin Balakrishnan. Multi-finger gestural interaction with 3d volumetric displays. In *UIST '04: Proceedings of the 17th annual ACM symposium on User interface software and technology*, pages 61–70, New York, NY, USA, 2004. ACM.

[17] Michael Halle, Joshua Napoli, and Wendy Plesniak. Three-dimensional displays and computer graphics. In *SIGGRAPH '05: ACM SIGGRAPH 2005 Courses*, page 2, New York, NY, USA, 2005. ACM.

[18] Takayuki Hoshi, Takayuki Iwamoto, and Hiroyuki Shinoda. Non-contact tactile sensation synthesized by ultrasound transducers. In *WHC '09: Proceedings of the World Haptics 2009 - Third Joint EuroHaptics conference and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, pages 256–260, Washington, DC, USA, 2009. IEEE Computer Society.

[19] Takayuki Hoshi, Masafumi Takahashi, Kei Nakatsuma, and Hiroyuki Shinoda. Touchable holography. In *SIGGRAPH '09: ACM SIGGRAPH 2009 Emerging Technologies*, pages 1–1, New York, NY, USA, 2009. ACM.

[20] IDEO. Human Centered Design: An Introduction (2nd Edition). <http://www.ideo.com/work/item/human-centered-design-toolkit/>, 2010. [Online; accessed 18-February-2010].

[21] Clarity Consulting Inc. Facebook Developer Toolkit. <http://facebooktoolkit.codeplex.com/>, 2010. [Online; accessed 9-April-2010].

[22] IO2Technology. IO2Technology: Heliodisplay / Interactive Free-Space Display. <http://www.io2technology.com>, 2010. [Online; accessed 3-February-2010].

[23] Takayuki Iwamoto, Mari Tatezono, Takayuki Hoshi, and Hiroyuki Shinoda. Airborne ultrasound tactile display. In *SIGGRAPH '08: ACM SIGGRAPH 2008 new tech demos*, pages 1–1, New York, NY, USA, 2008. ACM.

[24] Andrew Jones, Ian McDowall, Hideshi Yamada, Mark Bolas, and Paul Debevec. Rendering for an interactive 360-degree light field display. In *SIGGRAPH '07: ACM SIGGRAPH 2007 papers*, page 40, New York, NY, USA, 2007. ACM.

[25] Thomas Kelley. *The Ten Faces of Innovation: IDEO's Strategies for Defeating the Devil's Advocate and Driving Creativity Throughout Your Organization*. Broadway Business, USA, 2005.

[26] Carl Kenner. Wiimote — Wiibrew. <http://wiibrew.org/wiki/Wiimote>, 2010. [Online; accessed 12-February-2010].

[27] Laurens R. Krol, Dzmitry Aliakseyeu, and Sriram Subramanian. Haptic feedback in remote pointing. In *CHI EA '09: Proceedings of the 27th international conference extended abstracts on Human factors in computing systems*, pages 3763–3768, New York, NY, USA, 2009. ACM.

[28] John Lasseter. Principles of traditional animation applied to 3d computer animation. In *SIGGRAPH '87: Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, pages 35–44, New York, NY, USA, 1987. ACM.

[29] Johnny Chung Lee. Johnny Chung Lee - Projects - Wii. <http://johnnylee.net/projects/wii/>, 2008. [Online; accessed 28-January-2010].

[30] Ming Lin and Kenneth Salisbury. Haptic rendering–beyond visual computing. *IEEE Comput. Graph. Appl.*, 24(2):22–23, 2004.

[31] Frank Loewenich and Frederic Maire. Hands-free mouse-pointer manipulation using motion-tracking and speech recognition. In *OZCHI '07: Proceedings of the 19th Australasian conference on Computer-Human Interaction*, pages 295–302, New York, NY, USA, 2007. ACM.

[32] Dimensional Studios Ltd. Musion Eyeliner - 3D Holographic Projection Technology. <http://www.dimensionalstudios.com/3d_holographic_projection.html>, 2008. [Online; accessed 29-January-2010].

[33] Frank Luna. *Introduction to 3D Game Programming with Direct X 9.0c: A Shader Approach (Wordware Game and Graphics Library)*. Wordware Publishing Inc., Plano, TX, USA, 2006.

[34] Wired Magazine. Build a 3-D Theater - Wired How-To Wiki. <http://howto.wired.com/wiki/Build_a_3-D_Theater>, 2008. [Online; accessed 3-March-2010].

[35] Wired Magazine. Vuzix iWear VR920 - The New Virtual Reality for Gamers. <http://www.vuzix.com/iwear/products_vr920.html>, 2010. [Online; accessed 5-March-2010].

[36] Microsoft. Microsoft Speech Technologies. <http://www.microsoft.com/speech/>, 2010. [Online; accessed 14-March-2010].

[37] Microsoft. Xbox.com — Project Natal. <http://www.xbox.com/en-US/live/projectnatal/>, 2010. [Online; accessed 12-March-2010].

[38] CNET News. Spore's crazy creature population: 100 million. <http://news.cnet.com/8301-10797_3-10232596-235.html>, 2009. [Online; accessed 26-February-2010].

[39] Nintendo. Nintendo. <http://www.nintendo.com/consumer/systems/wii/en_na/images/system/wiiRemoteOpPoint.gif>, 2010. [Online; accessed 11-April-2010].

[40] Randy Pausch. Virtual reality on five dollars a day. In *CHI '91: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 265–270, New York, NY, USA, 1991. ACM.

[41] Brian Peek. Managed library for nintendo's wiimote. <http://wiimotelib.codeplex.com/Wikipage>, 2010. [Online; accessed 1-April-2010].

[42] Inc. Phidgets. Phidgets, Inc. - Unique and Easy to Use USB Interfaces. <http://www.phidgets.com/>, 2010. [Online; accessed 13-April-2010].

[43] George N. Phillips Jr. Modular approach of multimodal integration in a virtual environment. In *ICMI '02: Proceedings of the 4th IEEE International Conference on Multimodal Interfaces*, page 331, Washington, DC, USA, 2002. IEEE Computer Society.

[44] Ismo Rakkolainen. Mid-air displays enabling novel user interfaces. In *SAME '08: Proceeding of the 1st ACM international workshop on Semantic ambient media experiences*, pages 25–30, New York, NY, USA, 2008. ACM.

[45] Ismo Rakkolainen, Stephen DiVerdi, Alex Olwal, Nicola Candussi, Tobias Hüllerer, Markku Laitinen, Mika Piirto, and Karri Palovuori. The interactive fogscreen. In *SIGGRAPH '05: ACM SIGGRAPH 2005 Emerging technologies*, page 8, New York, NY, USA, 2005. ACM.

[46] Ismo Rakkolainen and Karri Palovuori. Interactive digital fogscreen. In *NordiCHI '04: Proceedings of the third Nordic conference on Human-computer interaction*, pages 459–460, New York, NY, USA, 2004. ACM.

[47] Ismo K. Rakkolainen and Artur K. Lugmayr. Immaterial display for interactive advertisements. In *ACE '07: Proceedings of the international conference on Advances in computer entertainment technology*, pages 95–98, New York, NY, USA, 2007. ACM.

[48] RealFiction. RealFiction (Dreamoc 3D Holographic Display). <http://www.realfiction.com>, 2010. [Online; accessed 1-February-2010].

[49] Miao Song, Serguei A. Mokhov, Alison R. Loader, and Maureen J. Simmonds. A stereoscopic opengl-based interactive plug-in framework for maya and beyond. In *VRCAI '09: Proceedings of the 8th International Conference on Virtual Reality Continuum and its Applications in Industry*, pages 363–368, New York, NY, USA, 2009. ACM.

[50] Jim Steinmeyer. Hiding the elephant: How magicians invented the impossible and learned to disappear. In *Hiding the Elephant: How Magicians Invented the Impossible and Learned to Disappear*, page N/A, New York, NY, USA, 2003. Da Capo Press; 3rd Printing edition.

[51] Brandon T. Taylor and V. Michael Bove, Jr. Graspables: grasp-recognition as a user interface. In *CHI '09: Proceedings of the 27th international conference on Human factors in computing systems*, pages 917–926, New York, NY, USA, 2009. ACM.

[52] WPF Development Team. WPF3D Team Blog : Interacting with 2D on 3D in WPF. <http://blogs.msdn.com/wpf3d/archive/2006/12/12/interacting-with-2d-on-3d-in-wpf.aspx>, 2010. [Online; accessed 9-April-2010].

[53] Provision Interactive Technologies. 3d Display Types — Provision.tv. <http://www.provision.tv/3d_display_types>, 2010. [Online; accessed 29-January-2010].

[54] The Engineering Toolbox. Air Properties. <http://www.engineeringtoolbox.com/air-properties-d_156.html>, 2010. [Online; accessed 18-March-2010].

[55] Daniel Vlasic, Ilya Baran, Wojciech Matusik, and Jovan Popović. Articulated mesh animation from multi-view silhouettes. In *SIGGRAPH '08: ACM SIGGRAPH 2008 papers*, pages 1–9, New York, NY, USA, 2008. ACM.

[56] Wikimedia. Wikimedia. <http://upload.wikimedia.org/wikipedia/commons/6/62/Example_snells_law.gif>, 2010. [Online; accessed 18-March-2010].

[57] Wikipedia. Anaglyph (Image). <http://en.wikipedia.org/wiki/Anaglyph_image>, 2010. [Online; accessed 1-February-2010].

[58] Wikipedia. Bluetooth. <http://en.wikipedia.org/wiki/Bluetooth>, 2010. [Online; accessed 10-March-2010].

[59] Wikipedia. Electromagnetic spectrum. <http://en.wikipedia.org/wiki/Electromagnetic_spectrum>, 2010. [Online; accessed 10-March-2010].

[60] Wikipedia. Lumen (Unit). <http://en.wikipedia.org/wiki/Lumen_unit>, 2010. [Online; accessed 15-February-2010].

[61] Wikipedia. Microsoft XNA - Wikipedia, the free encyclopedia. <http://en.wikipedia.org/wiki/Microsoft_XNA>, 2010. [Online; accessed 13-April-2010].

[62] Wikipedia. Polarized 3D Glasses. <http://en.wikipedia.org/wiki/Polarized_3D_glasses>, 2010. [Online; accessed 3-March-2010].

[63] Wikipedia. Snell's Law. <http://en.wikipedia.org/wiki/Snells_law>, 2010. [Online; accessed 10-March-2010].

[64] Kenneth Wittlief. Stereoscopic 3d film and animation: getting it right. *SIGGRAPH Comput. Graph.*, 41(3):2, 2007.

[65] Xuewu Xu, Sanjeev Solanki, Xinan Liang, Shuhong Xu, Ridwan Bin Adrian Tanjung, Yuechao Pan, Farzam Farbiz, Baoxi Xu, and Tow-Chong Chong. Dynamic display of 3d objects in real and virtual spaces with computer-generated holography. In *VRCAI '08: Proceedings of The 7th ACM SIGGRAPH International Conference on Virtual-Reality Continuum and Its Applications in Industry*, pages 1–6, New York, NY, USA, 2008. ACM.

[66] John M. Zelle and Charles Figura. Simple, low-cost stereographics: Vr for everyone. In *SIGCSE '04: Proceedings of the 35th SIGCSE technical symposium on Computer science education*, pages 348–352, New York, NY, USA, 2004. ACM.