Design and Evaluation of Trajectory-Based Tasks in a Thin-Seam Perspective-Corrected Cubic Display

by

Yichen Tang

Bachelor of Science, Fudan University, 2010

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF
THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF APPLIED SCIENCE

in

The Faculty of Graduate and Postdoctoral Studies

(Electrical and Computer Engineering)

THE UNIVERSITY OF BRITISH COLUMBIA

(Vancouver)

January 2016

Abstract

This thesis describes the design and evaluation of wire-tracing task in pCubee, an improved version of hand-held perspective-corrected display that allows the user to observe and interact with 3D content visualized inside the cubic system. In order to overcome visual discontinuity issues identified from previous works, we redesigned pCubee system using OLED panels and FPGA-based display controller to achieve reduced seam size and compact form-factor. We investigated user performance with the new system using a trajectory-based wire-tracing task where users were asked to move a ring along wires. Experiments were conducted to evaluate the impact of ring radius, wire length and curvature. Analysis of results revealed that a linear model similar to the steering law for 2D tunnel task applies to 3D trajectory-based task in pCubee as well, exhibiting an increase of task completion time when smaller ring or longer wire is used. Our study complemented the theory that 3D interaction in virtual reality system follows existing principle for 2D tasks, and also identified a potential method to evaluate interaction designs for geometric displays. This work could help motivate future development of pCubee and guide interaction design for similar systems.

Preface

All of the work presented in this thesis was conducted in Human Communication Technologies Laboratory at the University of British Columbia. All experiments and associated methods were approved by the University of British Columbia's Research Ethics Board (Certificate Number H08-03005).

The hardware design in Chapter 3 was done by myself, with OLED panels available in HCT lab and development kit sponsored by Altera. Display mechanical design was based on the original pCubee project by I. Stavness, B. Lam and S. Fels (2010). Modifications were made to accommodate different dimensions and structure of the new system. J. Wang was involved in the design of 3D printed chassis and contributed to configuration and tuning of the printing device. I owned FPGA development for display controller, referenced datasheets and application notes provided by display panel manufacturer, and designed customized features to support multiple displays. Except for the DVI-HSMC transceiver card purchased from Terasic and Arduino Micro development board, all other peripheral circuits were designed by myself and printed in a local PCB house.

The software system for new pCubee was modified from B. Lam's original code. B. Lam implemented head-tracking feature including off-axis projection, matrix transformations and calibration which were reused in this work. I updated some of the original models, added an abstraction layer in order to support later versions of PhysX engine, and introduced touch input device into the scene control.

I was the lead investigator responsible for user studies in Chapter 4. I.

Stavness, B. Lam and A. Ho contributed to early stages of system development and experiment design, including the execution of user studies, data gathering and analysis. Professor S. Fels was actively involved throughout the project in concept formation, design iterations and manuscript edits.

Part of this work has been published in CHI 2014 Interactivity session. Yichen Tang, Ian Stavness, and Sidney S. Fels. 2014. The new pCubee: multi-touch perspective-corrected cubic display. In CHI 2014 Extended Abstracts on Human Factors in Computing Systems (CHI EA 2014). ACM, New York, NY, USA, 419-422.

This research project was funded by the Networks of Centres of Excellence of Canada through GRAND, the Graphics, Animation and New Media NCE.

Table of Contents

Appendices

List of Tables

List of Figures

Acknowledgements

I would like to thank my supervisor Professor Sidney Fels for guidance and supervision, especially for his invaluable suggestions for me to overcome challenging tasks and fight against various distractions. I would not have been where I am without his patient and continous encouragement.

Thanks to Gregor Miller, Ian Stavness, Billy Lam, Antonio Sanchez, Andrew Ho, Abir Al Hajri, Matthew Fong, Johnty Wang and all HCT and MAGIC friends for support and feedback. Best wishes to your future endeavors.

Finally, thank you mom and dad for your love and support.

Chapter 1

Introduction

A Virtual Reality (VR) system, as defined by Cruz-Neira et al. [7], is one which provides real-time computer-generated head-tracking perspective with interactive control and binocular display that allows users to interact with alternate reality. Researchers have exploited various technologies in order to provide convincing virtual reality experiences. Each solution excels at different aspects. Some focus on immersive experiences, like the CAVE Automatic Virtual Environment system [7]. With images projected onto walls of a room, it allows users to step into a surrounding virtual space, see it from different angles and navigate around the world. Head-Mounted Display (HMD) provides similar immersive feelings as well as extraordinary image quality in a smaller form-factor. Another solution, referred to as volumetric display [20], aims at illuminating actual physical points in the real world using advanced technologies like laser beams and spinning films, producing realistic viewpoint-independent virtual Three-Dimensional (3D) worlds.

Perspective-corrected geometric display technology, which extends the concept of Fish Tank Virtual Reality (FTVR) display by arranging multiple flat panels together to form a geometric shape (e.g. a cube) to provide head-coupled perspective, stands out as a promising solution with its various advantages. By tracking the viewer's head position and adjusting the image perspective accordingly on each screen, geometric display offers a strong 3D clue with motion parallax, creating an illusion of realistic objects being contained within the physical area enclosed by display panels. Unlike other VR display solutions, no sophisticated circuitry or excessive mechanical setup is required. Therefore, the display can be designed to be

as compact and lightweight as a small personal hand-held device. Geometric display also inherits advantages of flat LCD or OLED display technology, including the slim form-factor, low power requirement, continuously improving image quality and native support for touch input.

The unique form-factor of hand-held geometric display has introduced challenges in interaction design. In pCubee system [20], our cubic perspective-corrected geometric display, a virtual cubic volume overlaps with the physical area formed by display panels, creating a strong illusion of realistic object existing within this space. The alignment between virtual volume and the actual physical geometric shape brings opportunities for designers to create consistent visual experience that expands to both virtual and real world. There are many questions to be answered for this novel display before effective interaction designs can be implemented. For example, among multiple methods for 3D manipulation tasks (e.g. direct mapping, virtual widgets, or ray-based pointing), which one would be more suitable for the scenario in a perspective-corrected cubic display is currently unknown. It would also be useful to understand the difficulty levels of various tasks that can be performed in pCubee and identify which tasks are better supported by this type of system. Furthermore, unlike traditional display which serves only as an output by providing feedback according to the user's input from other devices (e.g. mouse or trackball), the geometric display itself can be manipulated and used as a six Degree-of-Freedom (DOF) input device, which also presents a unique challenge of combining both input and output effectively in a single equipment. Lastly, it is worthwhile to understand how user perform differently on the same task implemented separately in virtual environment and real world, which can provide valuable information for designers to create more realistic interactions to fill the gap.

To start with a small step, in this thesis we would like to study the fundamental principles of user interactions in perspective-corrected geometric display system. For linear pointing and selecting task, Fitts' Law [11] is a robust regularity describing the mathematical relationship between user

performance and task difficulty. Steering law extends the model to 2D trajectory-based movement task. If a similar regularity could be discovered for 3D tasks in pCubee, we would have a model that can both describe user performance on our device and gauge effectiveness of interaction design objectively.

Inspired by 2D tunnel task used for steering law evaluation, we investigated a trajectory-based 3D manipulation task where the user have to move a ring along a given wire without the two objects colliding with each other in virtual 3D environment visualized in pCubee. We designed this wire-tracing task to evaluate three potential factors that may affect task completion time in user studies. Two experiments were conducted in order to test whether a linear relationship similar to the steering law model proposed by Accot et al. [1] applies in 3D trajectory-based tasks. The outcome from our analysis of data can help us gain better understanding of user performance on 3D manipulation tasks in hand-held geometric displays, and motivate further development on novel interactions around this technology.

Previous work on the old pCubee system revealed a visual discontinuity issue that affects user performance in multiple tasks. As an effort to address this issue before conducting our experiment, a redesigned pCubee system was implemented to take advantage of latest OLED display technologies, customized FPGA-based display controller as well as a lightweight 3D-printed chassis. The new system features thinner display seams and better image quality in a smaller form factor. We will continue identifying advantages and limitations of the system in order to provide improved 3D experience.

1.1 Contributions

The research presented in this thesis provide the following two major contributions.

1.1.1 Evaluation of Trajectory-Based Tasks in pCubee

Trajectory-based task such as following a curve in virtual 3D environment is challenging due to multiple reasons such as limited feedback of depth on virtual reality systems. It is valuable to understand whether 2D interaction principles apply in virtual 3D environment. We investigated two sets of trajectory-based tasks where user moves a ring along wires in our perspective-corrected display. The research question we tried to answer from this study is whether a model similar to the steering law for 2D tunnel task can be identified to describe user performance for 3D trajectory-based tasks, if such a model exists, what the contributing factors were for wire-tracing tasks, and what modifications were required on the original linear model in order to correctly reflect user performance in our scenario. Furthermore, this research presented a potential method of investigating how trajectory-based task and the linear model can be applied to evaluate the effectiveness of interaction designs in geometric displays.

Along with our study of wire-tracing task performance, a qualitative survey was also conducted to gauge user experience and preference on interaction mechanisms used in pCubee. User feedback revealed limitations of our system that can help guide future iterations of hardware development and interaction design.

1.1.2 A New Hand-Held Perspective-Corrected Cubic Display System Design

In order to ensure our evaluation of trajectory-based tasks is not undermined by the visual discontinuity issue in pCubee identified previously, we completely redesigned the perspective-corrected display system incorporating OLED display panel, FPGA display controller and 3D-printed chassis. The new pCubee brought several improvements over existing system as described below.

- With thinner OLED panels we were able to reduce screen bezel width

from 23 mm to 6 mm. The thinner seam width helped reduce visual discontinuity caused by occlusion when the user changes viewpoint from one screen to another, which discourages cross-screen view change and limits how the user interacts with the display. As previous study indicated, smaller seam will motivate user to utilize more than one screen and change viewpoint more frequently, which helps them to accomplish visualization tasks such as 3D path-tracing faster and more accurately.

- The new system with lightweight 4.3-inch OLED panels and 3D printed plastic chassis was smaller and lighter than the previous one with wooden frame and 5-inch LCDs. New design allowed it to fit better in one hand, making the whole display easier to manipulate and reducing frustration when prolonged bi-manual interaction tasks are conducted.

- Higher contrast and wide viewing angle introduced by OLED panels gave the user better visibility when the screen is tilted, resulting in consistent visual experience and stronger 3D illusion.

- Resistive touch panels were attached to all our displays to enable tracking of single touch behavior on each panel. Combining information received from all five panels, we were able to create cross-screen multi-touch gestures for manipulating virtual objects within the display.

- Last but not the least, a FPGA-based display controller was designed to replace 5 independent VGA-LVDS conversion board, eliminating the need of multiple VGA output from PC as well as several peripheral accessories. Hardware complexity of new pCubee system was significantly reduced to enable easier carrying and setting-up the portable system.

1.2 Thesis Structure

The remainder of this thesis is structured as follows: Chapter 2 surveys previous literature on geometric displays and related interaction designs; Chapter 3 outlines hardware and software design and analysis of the new pCubee; Chapter 4 covers user studies we conducted to evaluate user performance on new pCubee and corresponding results; Chapter 5 concludes with discussions on our findings and suggestions on future research directions.

Chapter 2

Related Work

As personal computing devices become more and more portable and powerful, there has been increasing attention in nontraditional user interfaces that create intuitive experiences. With the advent of virtual environments, augmented reality and mixed reality technologies, the 3D user interface is becoming a critical area for researchers and developers to understand. One of the recognizable barriers is the limitation of display technologies. While computer graphics has improved drastically in past years, allowing realistic images to be generated, most users still interact with 2D projections of 3D virtual worlds, using the same basic principles and designs for traditional desktop interfaces. Increasingly, efforts have been made to investigate 3D display technologies and interactions supported by these new technologies.

In this chapter, we review previous literature in two major categories. First, we summarize our survey on development of geometric 3D display technologies and existing systems similar to the new pCubee, identify the unique characteristics of this type of display and discuss factors that prompted our redesign of the pCubee system. Then we explore previous studies on interaction designs and point out remaining challenges for this hand-held display.

2.1 Geometric 3D Display Technologies

Continuous efforts have been put into the development of 3D display devices that can visualize realistic 3D objects in physical space, classified as volumetric displays [20]. By illuminating corresponding physical points (static

or swept-volume displays) or providing autostereoscopic horizontal-parallax views in multiple degrees (multi-view displays), volumetic displays are able to present vivid 3D images to multiple viewers at the same time. However, these benefits come at a cost of reduced resolution, slower refresh rate and limited image quality. Volumetric displays also require complicated setup that make them only suitable for desktop applications.

Research has been done to explore the possibilities of utilizing flat-panel display technology in 3D display design, hoping to inherent the benefits of this mature technology and bring better image quality to 3D displays. The early work on this type of display technology took place in 1980s when Fisher [10] used a monitor with head tracking to provide different views to the user by displaying pre-computed images. A similar system was developed by Venolia et al. [32] to provide pre-rendered stereoscopic images with head-coupled perspective. Deering [8] described mathematical details about generating high-quality head-coupled stereoscopic images on a CRT monitor, taking the screen curvature and glass thickness into account. McKenna [25] used a 3D target reaching task to compare three different types of displays, including a fixed-display monocular system with head-tracking, a mobile display monocular system, as well as a hand-held display, and reported best results on the head-tracked system.

Arthur et al. summarized similar approaches referred to as Fish Tank Virtual Reality (FTVR) technology in their work [3]. A typical FTVR system requires a head tracker to determine the user's real time point of view. Based on this information, computer graphics software adjusts rendered images on a flat screen to present the corrected view from the user's perspective. Unlike volumetric displays that provides 3D effects by illuminating actual physical voxels in space, FTVR displays rely heavily on motion parallax in order to create their 3D effect, because they essentially only generate 2D images. Motion parallax is a monocular dynamic 3D cue delivering motion-induced depth perception [33]. It causes objects closer to the viewer to move more quickly across the visual field and object farther away to move more

slowly. Combining other auxiliary depth cues including relative sizes, occlusion, shadows and texture gradient, FTVR display can deliver realistic 3D experience to the user with a simple and lightweight setup. The flexibility of this technology enables it to be supported by commercial mobile products such as Amazon Fire Phone and Nintendo 3DS consoles.

While being simple and fairly effective, single FTVR display systems only support limited viewing angles. Virtual objects will be cut off by screen border if the user moves beyond certain angle, causing mismatches that hinder 3D effect. Geometric display effectively overcome this issue by arranging multiple FTVR displays into a geometric shape such as a cube, allowing the user to continue observing virtual object beyond screen boundaries. While creating competitive 3D effects as volumetric displays, geometric 3D display inherits all benefits from FTVR technologies, such as high resolution, wide color range, low latency as well as mechanical flexibility. However, in most cases geometric display only supports one single user at a time, as the perspective can only be adjusted to accommodate one view point. This limits the display's potential for collaborative tasks and public scenarios, but has less impact for personal usage in either hand-held or desktop setup.

One of the earliest geometric display system is the CAVE Automatic Virtual Environment [7], which uses the walls of a room as back projection screens. The system allows a user to walk into the inward-facing vitual environment and navigate around to interact with the immersive 3D world. Cubby, a scale-down version of the CAVE system using similar inward-facing mechanism was presented by Djajadiningrat et al. [9]. In these displays, virtual 3D space overlapped with real physical cubic area in front of the panels, thus creating an immersive illusion allowing the user to reach into virtual 3D world and even "touch" virtual objects. However, inward-facing geometric display has to be seamless otherwise the occlusion caused by screen borders will severely impact 3D effect. As a result, back-projection seems to be the best solution suitable for this type of display but also makes it difficult to implement in a small and portable form-factor.

On the other hand, outward-facing geometric display has better tolerance on screen borders since virtual 3D environment appears to be confined inside the area constructed by multiple screens so appropriate occlusion can be accepted. For example, Inami [17] demonstrated MEDIA CUBE with four display panels mounted to sides of a cubic box. gCubik [22] proved the possibility of utilizing lens arrays to filter pre-rendered images of multiple perspectives to a selection of viewing angles at a sacrifice of display resolution. Cubee [31] arranged five desktop LCD monitors into a cubic frame that hung from a truss to allow easier manipulation. pCubee [30] was a smaller version of Cubee that replaced desktop LCD screens with 5-inch LCD panels, allowing the whole display to be held and manipulated in one hand. Lam [19, 20] evaluated this hand-held perspective-corrected cubic display and identified how visual discontinuity resulting from different seam size impacts path-tracing task performance and discourages users from moving viewpoints from one screen to another. Their results indicated that a seam width between 3 mm and 13 mm seems to be the most effective size for a similar screen-to-seam ratio condition, resulting in reduced mean response time and mean error rate.

As a drawback from inverting the screen arrangement, outward-facing display provides weaker immersive experience, and limits any interaction with virtual objects outside of the screen boundary. Movement mapping or projection is required for 3D manipulation tasks in virtual world, increasing the challenges in interaction design.

2.2 User Interface Design for Geometric 3D Display

While traditional user interfaces like mouse and keyboard are still prevalent, there are more and more non-traditional devices and interface components

emerging. In the case of geometric display, it is already confirmed by multiple researchers [15, 20, 25] that traditional mouse is not an effective interface for object manipulation in virtual 3D environment. In order to identify the benefits brought by the unique hand-held geometric display and achieve a deeper understanding of interaction designs for similar display technologies, Lam et al. conducted multiple studies on previous pCubee system. They first studied visual discontinuity on head-coupled display using a radial spanning tree-tracking task [20] and identified the effect of display seam size. It is reported that small-range structured rotation with high quality visualization, such as a single-screen head-tracked display, would be sufficient for path-tracing visualization tasks, reaffirming the findings of Ware et al. [34]. Then they designed a spatial reasoning task where the user had to compare 3D cubes using mental rotation. They compared task performance under three different conditions (desktop+mouse, desktop+pCubee and pCubee-only) and reported differences in mean error rates and response times. The results indicated that pCubee did not significantly outperform the desktop display, primarily due to seam occlusions and restraints on manipulating the device resulting from the cables, the size and weight of pCubee. However, subjects' feedback revealed that pCubee is highly preferred and intuitive choice for performing the cube comparison task. Further experiments [19] were carried out on more complex tasks such as solving 3D puzzles that require coordination of selection, translation, rotation and placement tasks. Although users were significantly faster in solving physical puzzle on a desk compared to doing an identical virtual task within pCubee, this work demonstrate the potential and challenges of using hand-held geometric display as an interactive platform for real-world tasks. Throughout all these studies, it is pointed out that display seam size is one of the major issues impacting user performance on different tasks and therefore should be taken care of in future generations of the system.

Aside from visual discontinuity issue, it is reported that people often find it inherently difficult to understand 3D spaces and interact with virtual objects in 3D environment simulated by computer [14]. Although we

live in a 3D physical world and interact with 3D objects all the time in our daily life, there are lots of constraints and affordances on which we can base our actions [28]. Objects around us are designed to have their own moving trajectory so that the user only needs to spend minimum effort to make the object move as it should. For example, a door swings only around the hinges and a key can only be inserted straight into the lock. Any violation of the rule will cause confusion. However, in free 3D space, a user doing tasks such as free translation and rotation is usually not able to benefit from these extra visual constraints as well as haptic feedback. Therefore the user has to spend more efforts in order to accomplish similar tasks in virtual space. Because of the described difficulties in 3D interaction, proper system design is required in order to provide an experience close to real-world tasks.

Efforts have been made to explore suitable interaction designs for 3D manipulation tasks [4]. Smith et al. tried to apply 2D constraints to 3D scene to assist the user in placing tasks [28]. Martinet et al. also evaluated possibilities of separating DOF [23, 26]. Additional hand-held devices [21, 27, 35] or metaphors [29] were also introduced for virtual object manipulation. Despite user input method, the 3D user interface components should also be carefully designed [18], especially for different types of display technologies [13].

Before proposing any new interaction designs, we have to first understand whether any fundamental regularities exist for user interactions with geometric 3D display, so that a model independent from tasks can be used to describe and evaluate the effectiveness of user interaction designs. For example, Fitts' law [11] is a widely adapted and robust model to describe the relationship between cursor travel time and object dimension for target acquisition tasks. Previous work [6, 12, 24, 36] has already shown that the time taken to select an object using a 3D point cursor in volumetric display follows Fitts law, and will thus be a function of the travel distance. The popularity of Fitts' law is also because of its Index of Performance (IP) parameter, which describes the effectiveness of a certain system and can be used to compare the quality of different designs. If a similar regularity

can be found for tasks other than pointing and selecting in pCubee, it may be used to motivate and guide further interaction designs with usability in mind.

2.3 Summary

In this chapter, we summarized the development of geometric display technology and related usability studies. We identified limitations of previous systems and discussed design challenges for both display itself and interactions with the system. Main issues impacting user experience such as seam size were highlighted, which motivated the design of new pCubee system. We also pointed out that in order to design proper interaction mechanisms for pCubee, fundamental rules for user interactions in head-coupled geometric display have to be investigated.

Chapter 3

Design and Analysis of the New pCubee System

As a major revision of our previous system, the new pCubee is designed with primary focus on resolving existing issues discovered by studies using the earlier prototype. With various improvements on both visual and haptic experience, the new pCubee system allows us to carry out experiments to take a deeper look into how user interacts with virtual world in the display without worrying about distractions or discontinuities that impact user performance. Besides, with the addition of interaction mechanisms including cross-screen multi-touch support, the new pCubee extends its function from only a hand-held cubic 3D display to becoming a virtual reality interactive platform. This chapter will cover detailed designs of the new system, critical decisions made during the development, and discussions about trade-offs and next steps.

3.1 Hardware Design

The most distinctive change in new pCubee hardware is the replacement of LCD screens using five AMOLED panels in favor of their better display quality and thinner bezels. But this change actually pushed us to pursue a complete redesign of the whole system. Because driver technologies and required signals for OLED are different, existing display solution could not support OLED panels. We gave up on using multiple independent VGA to LVDS signal converters to forward video signal directly from PC to display panels because of two reasons. First, instead of using a bundle of wires to

transmit differential LVDS signal pairs, the OLED panel only supports parallel signals transmitted by a fine-pitch (0.3 mm) Flexible Flat Cable (FFC). In order to maintain acceptable signal quality, the length of FFC cable has to be much shorter, making it difficult to separate signal converter board and OLED panels like before. On the other hand, if OLED signal converter board is used, we would end up building a large box of five converter boards as well as two VGA signal splitters, because each panel requires an independent input and driver board. It is impossible to fit all these boards into the cubic area without attaching a much larger base. Therefore, instead of following similar solution from previous pCubee, we designed a customized FPGA-based video controller that drives all OLED panels directly, while taking only one input from PC. Table 3.1 provides more detailed differences between two display solutions.

Except for changing video controller, the smaller form-factor of OLED also requires a redesign of the chassis. A detachable 3D printed chassis is used to replace the wooden frame on our previous pCubee, reducing weight to make it easier to manipulate the display without sacrificing robustness. It is also possible for us to utilize resistive touch panels mounted on top of OLEDs to support touch input on all 5 sides of the cubic display, which further expands the possible applications of new pCubee.

Items	Old pCubee	New pCubee
Panel type	LCD	OLED
Panel dimension (inches)	5	4.3
Viewing angle (degrees)	100	176
Panel thickness (mm)	6	3
Panel bezel size (mm)	10	2
Final bezel size (mm)	23	6
Video signals	LVDS	parallel or serial RGB
Number of controller boards required	5	1
Power supplies	5 power supply units for each LCD	1 ATX power supply
Controller board	VGA to LVDS signal converter	DVI to multiple parallel controller
Controller source	Purchased with LCD	Designed in-house
Controller chip	proprietary CPLD-based solution	Altera Stratix III FPGA
Number of separate video inputs required	5	1
Frame per second	60	30
Touch overlay	None	4-wire resistive

Table 3.1: Comparison between old and new pCubee display solutions.

The hardware design of pCubee contains four major modules: (1) OLED screen controller, which drives each OLED screen with appropriate clock and pixel data; (2) peripheral circuits and connections, including DC/DC circuitry which provides appropriate power supply for FPGA and OLED dual voltage diode; (3) touch controller, which actively scans and detects single finger touches on all panels and reports coordinates via USB; and (4) 3D printed chassis which defines the device dimension and accurately holds all boards, panels, and sensors in their designed positions in pCubee. We will go through details of each module respectively in following sections.

3.1.1 OLED Screen Controller

The display controller of new pCubee is a highly hierarchical design. Major challenges in designing the new system include:

- Instead of using existing signal converter boards in our previous prototype, we have to manually design a display signal transmitter that can provide appropriate signals required by OLED panel driver IC, including specific timing signals, formatted RGB data and SPI control commands.
- Instead of taking care of one screen, our controller should have enough bandwidth and processing power to be capable of driving at least three independent displays simultaneously.
- The controller is expected to receive images for all five screens simultaneously via a single video output from PC.

The core of OLED screen controller is a FPGA-based video signal processing system implemented on an Altera DE3 development board with a Stratix III series EP3S340F1152C2 FPGA chip. The system takes input from a PC or laptop's graphics card through a single DVI-D connector, generates appropriate video streams for each OLED panel by picking from a large 1280 * 1024 input stream and adjusting pixel formats and timing, then drives OLED panels with corresponding data.

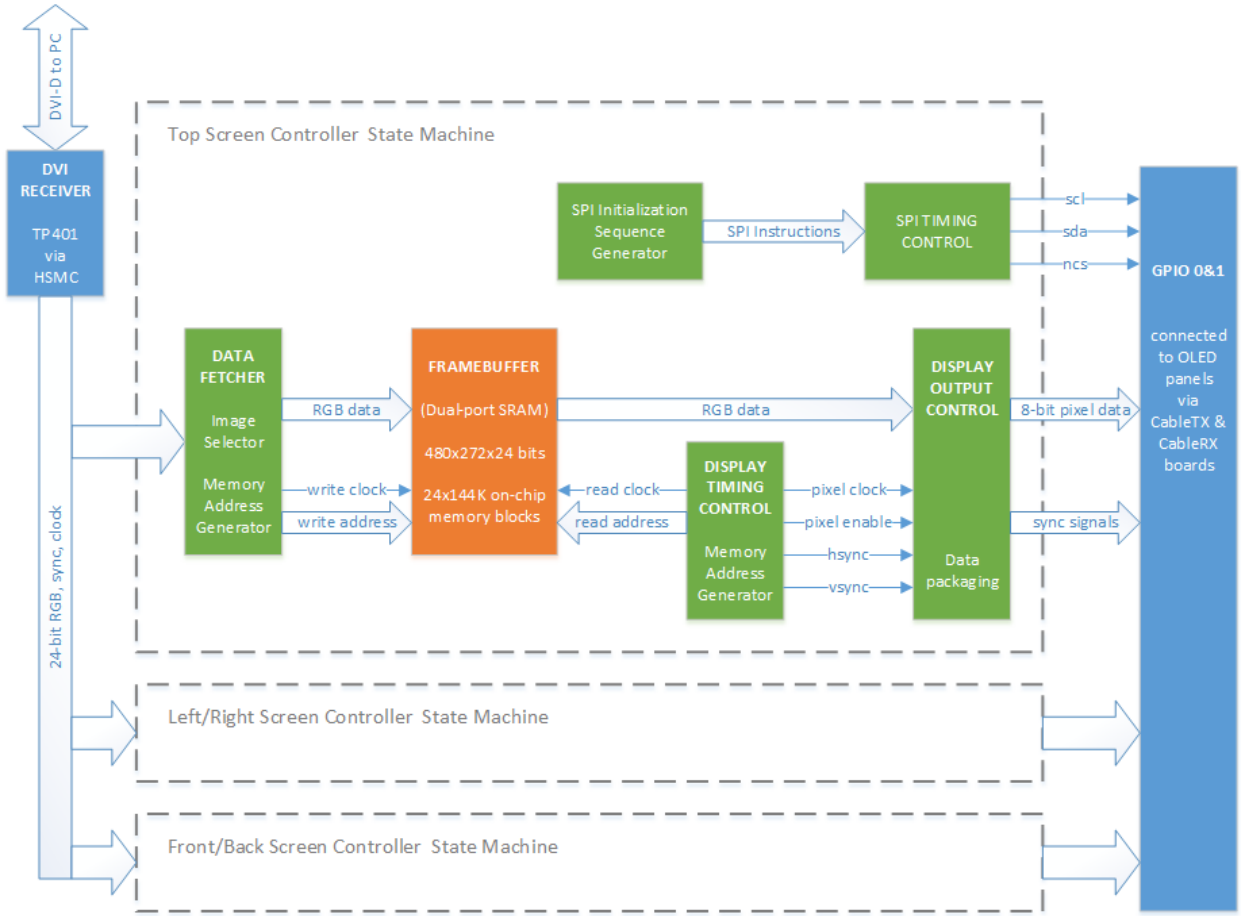


Figure 3.1: System block diagram of FPGA-based OLED screen controller.

The controller also handles panel driver power-on/off, initialization sequences and image/color testing on OLED panels. All these features implemented in Verilog and programmed onto the FPGA development platform.

The system block diagram 3.1 illustrates different software components of the OLED screen controller system. We have three groups of modules to provide three simultaneous video output to screens on top, left and front side of pCubee. To save FPGA resource and reduce output bandwidth, images on right and back screens are directly duplicated from left and right screen respectively. Actual image will be switched based on real time viewpoint. pCubee software determines which screen is not visible depending on data from head tracker and flip images when necessary.

Except for the FPGA development platform itself and a DVI transceiver daughter board, all other components illustrated below are developed and

DVI receiver

Video stream generated by PC graphics card is transmitted to the receiver via only one full-digital DVI-D connector. On PC/laptop side, the output is configured as a secondary 1280 * 1024 (or higher resolution, maximum 1600 * 1200 or 1920 * 1080 at 60 Hz) display that can either mirror with primary display (for debugging purposes) or act as a extended desktop so that primary display can be used to present other information such as an overview of virtual world. A Texas Instruments TFP401 chip on DE3 daughter board detects the input, synchronize and decode the data stream, then transmit via High-Speed Mezzanine Card (HSMC) to FPGA pins.

DVI data fetcher

Decoded RGB pixel data and synchronization signals (VSYNC, HSYNC, DE and clock) are shared by three DVI data fetcher connected to the same input pins. Each data fetcher synchronizes with data stream by searching for

appropriate HSYNC/VSYNC edges, compares with predetermined location information (pixel counts) and pulls corresponding data out of input stream and fills into framebuffer. For example, the OLED screen on top side of pCubee is designed to show the image data in a 480 * 272 rectangular area between (482, 100) and (961, 371) on the large virtual display output. The data fetcher first locates valid row by skipping first 100 rows of a new frame by counting VSYNC pulses in input stream. It then ignores the first 2 pixels on each following row and start forwarding valid RGB pixel data from input stream to framebuffer.

At the same time, a group of three counters keep track of the location of current pixel by recording VSYNC, HSYNC and DE (pixel enable) pulses. This information is then used to calculate the RAM address for the pixel. DE signal is also used to control the clock for RAM write operations.

Framebuffer

Three framebuffers are implemented using FPGA's internal on-chip memory blocks, storing RGB data for top, left and front screens respectively. Each framebuffer is configured as a 24-bit wide 2-port RAM that can fit in a whole frame of image data for the panel with a native resolution of 480 * 272 pixels. Memory address signal is 17-bit wide (supporting up to 131,072 addresses) which is sufficient for all 130,560 pixels. Each address provides access to 8-bit red, green and blue value for the corresponding pixel.

On the input port of the RAM, write clock, write address and input data bus are connected to the memory address generator of corresponding DVI data fetcher module. RAM write clock is controlled by pixel enable (DE) signal as well as conditions defined by image selector to ensure only pixels for the specific panel are passed through and saved in framebuffer. During RAM write cycle, 17-bit address signals are calculated based on three counter values in memory address generator. RAM address space is flattened and accessed as a 1 * 130560 list.

For the output port, pixel data are read out from framebuffer under a

different clock and sent directly to display output control module for final alignment and packaging. Read clock and RAM read address signals are inputs controlled by another memory address generator in display timing control module, which coordinates with display output control module to make sure correct data sequences are being sent to OLED panels.

From a resource perspective, because we are saving whole frames into the buffer, each framebuffer is relatively large: 480 columns * 272 rows * 3 colors * 8 bits = 3,133,440 bits (3060 Kbits), let alone other resources spent for managing the buffer structure and input/output port drivers. A minimum of approximately 9 Mbits memory space is required for the system to support three independent screens. There are two memory solutions each featuring some advantages and limitations in different aspects. First, we could implement all framebuffers using on-chip memory blocks of the FPGA. The internal memory block is much faster than any other alternatives and easier to design using IP modules provided by Altera. However, due to much higher cost, these memory blocks are usually very small and may not be sufficient for holding a large number of data. Alternatively, separate DRAM chips with plentiful storage capacity can be introduced to solve this issue, but we have to cope with a number of other design challenges, including extra latency, communication overhead, signal integrity risks and sophisticated DRAM management (such as byte format conversion and periodic refreshing). It is a less preferred solution considering all the sacrifices as well as the difficulties in implementation because a System on Programmable Chip (SoPC) design will be required.

Altera Stratix III FPGA family provides two types of configurable internal memory blocks, organized in 9 Kbits and 144 Kbits groups respectively. There are 1040 M9K memory blocks and 48 M144K blocks in the EP3SL340 chip on our development board, providing a total of 16,272 Kbits user memory. This is enough for us to implement three framebuffers while leaving enough extra for other purposes. In our final design, left and front framebuffers are implemented using all M144K blocks, each taking 24 blocks; top framebuffer uses nearly 40 percent of all M9K blocks.

If DVI data fetcher could not receive any valid data from PC (for exam-

ple, when DVI cable is not connected or PC display setting is not correctly configured), all framebuffers will instead load pre-programmed test images showing color gradients for debugging purposes. The memory initialization file (.mif) can be loaded and changed using IP configuration tool in Quartus II IDE.

Display timing control

This module contains a finite state machine that switches between different states and generates synchronization signals with appropriate sequences and delays for the display panel according to specifications in application notes. At the same time, a synchronized memory address generator calculates the memory location of the pixel to be displayed, then sends a read request to framebuffer. RGB data of current pixel from framebuffer and four synchronization signals (DCLK, VSYNC, HSYNC and DE) will be sent to display output control module for final assembly.

Ideally, pixel RGB values should be ready on data bus right at the moment when synchronization signals are triggered. However, due to an extra step of reading data from framebuffer, the delay in memory access creates some design challenges. To eliminate the possibility of sending incorrect data to next module, a few possible solutions were explored. At each negative edge of pixel clock signal, Instead of reading current pixel data, we request RGB values for the next pixel in framebuffer. When the next clock edge arrives, framebuffer returns correct data for the pixel, which align well with corresponding synchronization signals. However, this method may introduce a problem: for the first pixel on each line, the controller is not able to determine the time to send read request in advance, therefore no valid data will be read from framebuffer, resulting in a black pixel. In order to correct this, the controller monitors counts in Horizontal Back Porch (HBP) defined by clock counts between HSYNC signal and the first valid pixel. At the last pulse in HBP, controller starts requesting a new pixel from framebuffer. One cycle afterwards, when DE becomes positive indicating that

valid pixel data should be ready for read, framebuffer is able to provide correct RGB values for the first pixel on each line.

Together with corresponding initialization commands via SPI bus, display timing control module determines operating mode of OLED panel by adjusting data format or signal edges. This module also enables us to change the OLED frame rate and modify pixel clock (DCLK) frequency for later tuning on image and signal quality.

Display data control

This module is a short buffer that ensures all signals are properly aligned and packaged as required by OLED driver IC. Depending on actual OLED operation mode chosen, it converts parallel RGB data into serial signals and align them with synchronization signals if necessary.

For each screen, display data control module exports 8-bit RGB pixel data (under serial RGB mode) and four corresponding synchronization signals (DCLK, VSYNC, HSYNC and DE) to GPIO buffers. Two 40-pin parallel cables are used to transmit signals for all three screens to customized CableTX board.

SPI command sequence generator

Before OLED panel starts to display data received through parallel interface, it has to be properly configured in order to work as expected. For example, it has to understand the format of pixel data, polarity of clock signal, blackout delays etc. to correctly interpret data from parallel interface before anything else can be done.

The OLED panel allows us to use a SPI interface to send control commands to the driver IC in order to control display modes and provide configuration information before turning on the display. Functions supported by SPI commands include:

- Control power on/off sequence of different components on OLED

- Put the panel into standby mode when necessary
- Select initialization parameters including scan mode, RGB color format, display resolution, clock polarity
- Enable or disable optional pins and choose serial RGB mode or parallel RGB mode
- Provide gamma correction for color tuning

All acceptable SPI commands are stored within the module. Based on chosen configuration, appropriate lines of commands will be drawn and the sequence generator will iteratively trigger SPI timing control module to parse and send out each command one by one.

Under normal operating conditions, we could also send a few limited SPI commands to adjust display parameters or to put the panel into standby.

SPI timing control

The AMOLED driver IC HX5116 offers a 3-wire serial interface that requires a customized SPI controller. The serial peripheral interface uses three signals including chip select line (NCS), serial transfer clock line (SCL) and serial input/output data (SDA). Bi-directional command bitstream is transmitted over a 2 MHz clock. Each command consists of 7-bit register address, 1-bit read/write flag and 8-bit data. We implemented a SPI transceiver using Algorithmic State Machine (ASM) method to match the data format and timing requirements. Internally, the controller works at 40 MHz system clock. A state machine controls delays and timing, while its data path generates serial signals.

Each OLED panel is connected to a SPI timing control module through GPIO, which allows us to configure different screens separately.

Screen controller state machine

Three groups of above modules are implemented inside FPGA chip in order to support multiple screens at the same time. All modules within each group are coordinated by a high-level screen controller state machine that controls the system flow and provides user interface to monitor system status and manually switch between modes.

3.1.2 Peripheral Circuits

As shown in Figure 3.2, FPGA development boad is used to generate appropriate signal for OLED panels, but its outputs are limited to on-board GPIOs. In order to properly communicate with OLED driver IC mounted on each panel and display expected images, three customized printed circuit boards are designed to interface the signals and provide extra functions. Except for the FPGA development platform itself and a Terasic DVI transceiver daughter board, all other components illustrated below are developed and built in house.

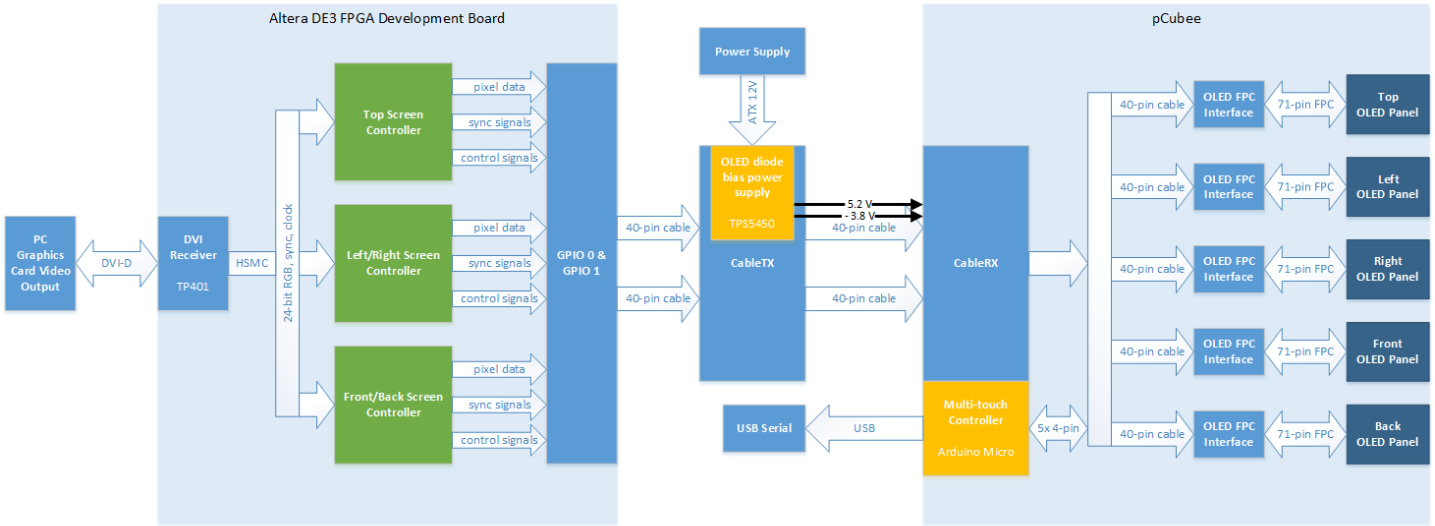


Figure 3.2: Peripheral circuits for FPGA-based OLED screen controller.

CableTX

This transmitter board takes input from two GPIO connectors on FPGA development board, and forward them to CableRX board on pCubee through a pair of 40-conductor cables.

In addition to signal forwarding, two DC-DC converters are also placed on this board to provide power for diodes in OLED panels, which requires a pair of specific voltages different from driver IC's 3.3 V VCC. A TPS5450 buck converter takes in 12 V from extra pins on the ATX power supply for FPGA, step down the voltage to 5.2 V to drive VSSP line. Another converter uses an inverting buck-boost topology to generate -3.8 V VSSN line. Both lines are added to 40-conductor cables connecting with CableRX board. Multiple pins are used to support enough current. All power lines are isolated from digital signals to avoid interference.

CableRX

CableRX is the main board located inside pCubee. It gets input signals and power lines from CableTX through two 40-pin cables, then generates signals for each OLED panel and transmits to five OLED FPC Interface boards via 40-pin cables as well. Due to dimension limits inside pCubee and relatively large number of connectors, routing on a two-layer CableRX board is a challenging task. To reduce the difficulty and get better signal quality, signals are rearranged on CableTX board before being transmitted, allowing less wire crossings and shorter lengths on CableRX.

To support resistive touch panels on each screen, 4-wire touch signals are also included in each 40-pin cables and routed to connectors on the back side of CableRX, allowing an Arduino Micro board to be connected to communicate with all five touch panels. Touch signal are processed by Arduino board and sent to PC via a separate USB cable.

OLED FPC interface

A small interface board is attached to each of the five OLED panels to convert power, pixel data, synchronization, display control and touch signals from 40-pin cable to 71-pin 0.3 mm-pitch FPC connectors. Power line filters and decoupling capacitors are placed on the board as suggested by OLED driver reference design.

3.1.3 Touch Controller

Each OLED panel is overlaid with a transparent 4-wire resistive touch sensor panel. The sensor includes two conductive layers separated by insulating spacer dots. When the screen is pressed, the top layer will bend downwards, pushing spacer dots away and creating an electrical connection between the two layers. By applying different voltages on opposite sides of one layer and measuring the voltage sensed on the other layer, we will be able to tell the distance between touch point and sides of the screen by comparing the voltages. Next, we do a similar measurement in the other direction, then we can get both horizontal and vertical positions of the touch.

Status	X-	X+	Y-	Y+
idle	N/C	N/C	N/C	N/C
read X	0	VCC	ADC	Hi-Z
read Y	ADC	Hi-Z	0	VCC

Table 3.2: 4-wire resistive touch signals.

Table 3.2 demonstrates the signals required on all four wires (X-, X+, Y-, Y+) during the process. As we can see, at least two wires (X-, Y-) have to be connected to ADC inputs in order to measure the voltage on that layer, but the other two (X+, Y+) only need to switch between VCC and Hi-Z mode so digital I/O pins should be sufficient. Considering that we have to support five touch screens, an Arduino Micro board providing 12 available analog input channels is chosen to be implemented as a touch

controller.

Once powered on, the controller consecutively reads touch coordinates on all five panels. 8-bit analog values are converted to pixel coordinates and transmitted to PC via USB serial interface. Due to communication delays and limitation of processing power in micro controller, the final scanning rate is 10 Hz, resulting two samples on a panel per second.

Due to the limitation of resistive touch technology, only single touch point can be detected on each screen. Gestures such as pinching and zooming on a screen can not be supported, however, it is possible to utilize touches on different screens to support cross-screen multi-finger gestures.

Capacitive touch solutions have been explored. Transparent touch overlays are available but they have at least 5 mm thick bezels to allow sensor traces to be routed, making them impossible to be mounted on top of our OLED panels.

3.1.4 3D Printed Chassis

On previous pCubee, all LCD panels were mounted on a wooden box with a smaller base for users to hold. It was designed to be a small form-factor hand-held device, but unfortunately the device was still not small and light enough for users comfortably manipulate in one hand. It was observed that the user tend to hold the box with both hands and rotate it around to see virtual objects displayed on screens. When bimanual tasks are conducted where the user have to hold the display in one hand and stylus in the other, subjects easily get tired after a short period of time. Also, the thick edges of wooden box added extra seam width which is critical to visual discontinuity issue.

With lighter and thinner OLED panels, we were able to build a even smaller cubic display without sacrificing too much display area. Our OLED panel has 3 mm bezel width between active area and outer case. If minimized seam size has to be achieved, we would require 3 mm * 3 mm rods around all panels to hold them in place. A wooden external frame at this thickness

would be too fragile to provide any support to other components. Therefore, a 3D printed frame is used instead.

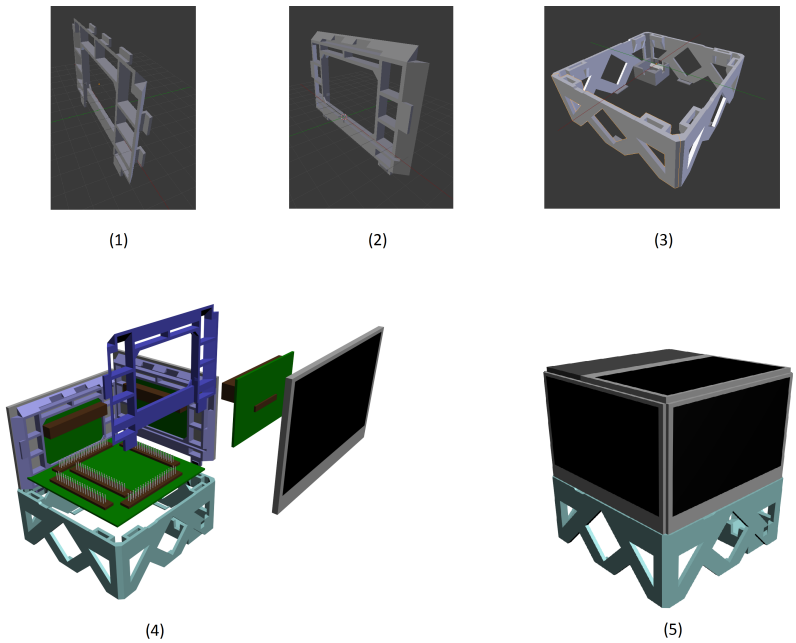


Figure 3.3: 3D printed chassis of new pCubee: (1) top module, (2) left module, (3) base module, (4) assembly demonstration, (5) final assembly

We designed a modularized 3D printed chassis that can be detached and reattached easily as shown in Figure 3.3. Each screen has a frame piece that holds circuit boards, cables and OLED panels together. Five panel modules are assembled together on a square base to become a cubic display. CableRX board, touch controller and magnet sensor are also held in place inside the cube. All chassis parts are 3D printed using ABS material.

Extra caution was taken when designing the chassis model, as the material tend to bend due to uneven temperature during cooling process. To allow heat to dissipate, more holes are opened while not affecting supporting strength. Wall thickness is also adjusted to ensure rigid and accurate parts get printed successfully.

3.2 Software System

While the software framework remains similar to our previous system, two major additions are made in new pCubee.

First, an abstraction layer is added between our own code and physics engine, isolating pCubee control logic and updated Nvidia PhysX engine. If the API changes in future PhysX releases, only minimum code revision on the abstraction layer will be required. Moreover, this addition creates the possibility of implementing other real time physics simulation engines into our system, allowing us to incorporate pCubee system with other larger platforms without changing internal structures and interfaces.

Second, with the addition of touch screens and customized controller, new pCubee is able to support single touches on all five screens. Touch coordinates are generated by the controller and sent to PC via USB serial, a separate thread is implemented to translate coordinates and recognize gestures. Figure 3.4 illustrates all gestures currently supported by pCubee.

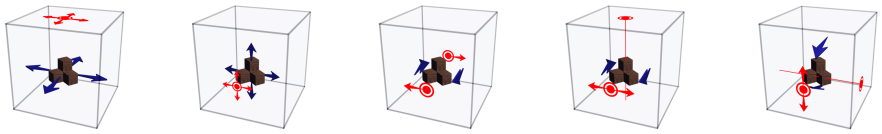


Figure 3.4: Cross-screen multi-touch gestures supported by pCubee, from left to right: translation along horizontal plane, translation along vertical plane, rotation when swiping at opposite directions, fixed-axis rotation along a vertical axis, fixed-axis rotation along a horizontal axis.

1. One finger movement on any screen moves the object accordingly on the same plane.

2. Two finger swiping in opposite directions on opposite screens causes the object to rotate towards the swiping direction.

3. One finger holding steady on a screen while another finger swiping in an adjacent screen will cause the object to rotate along axis given by the steady finger.

A demo is created to allow user to manipulate objects visualized inside pCubee using these touch gestures. As discussed in earlier sections, if capacitive touch sensors are used, more multi-touch gestures will be available, allowing multiple ways to interact with virtual objects by touching pCubee surface.

Due to function limitations of resistive touch technologies discussed above and based on the interaction task design, cross-screen multi-touch function was not evaluated in later user study sections. However, it still demonstrates a novel way of interacting with virtual reality world in pCubee.

3.3 Discussion

3.3.1 Design Review

With the primary focus on improving user experience for interacting with pCubee, the new design has achieved five major goals:

1. Reduced seam size of pCubee from 23 mm to 6 mm.
2. Replaced the wooden box with a detachable 3D-printed chassis.
3. Replaced LCD screens with OLED panels.
4. Provided cross-screen multi-touch support.
5. Implemented a FPGA-based video controller instead of relying on multiple signal conversion boards.

Combining these changes, the new pCubee design overcomes several issues discovered in previous studies on old prototype. For example, the

smaller seam size is proven to encourage multiple screen usage as well as to reduce response time and error rates for path-tracing task. From our previous studies on visual discontinuity of pCubee, it is suggested that a seam size for better user performance lies between 3 mm and 13 mm [20]. Considering the native bezel width of OLED panels available, the minimum seam size is achieved to alleviate its impact on user performance. During our actual user study using new pCubee, it is noticed that user easily changes viewpoints from one screen to another when necessary, as the occlusion is no longer creating difficulties for following an object or a certain trace.

Aside from seam size change, visual discontinuity is also mitigated by improved image quality, especially when observing from a sharp angle with the display. Traditional LCD panels may suffer from color distortion and brightness degradation at a sharp viewing angle, which hinders user experience as the changed object color on another screen makes it difficult for user to perceive them as different parts of a same object, causing the illusion of realistic 3D objects existing inside pCubee to fail. The wide viewing angle provided by OLED panel ensures consistent brightness, contrast and color balance at various angles, hence strengthens the geometric 3D illusion.

Other than visual discontinuity, it is observed that the form factor and length of signal cable in previous prototype discouraged subjects to rotate the display and utilize multiple screens to perform tasks. Although designed to support bimanual interactions, users are discouraged from performing such actions due to the weight of the display. With reduced number of cables and a smaller, lighter chassis, the new pCubee is made easier to handle with only one hand for a long time.

3.3.2 Limitations and Alternate Solutions

While the new design resolved several issues presented in our previous system, there is still numbers of areas that can be further improved to create an intuitive and flexible interface.

As pointed out by 3 subjects during our experiment, the cable attached to pCubee is still preventing them from manipulating the display freely.

We explored the possibilities of eliminating external cables and creating a standalone device without sacrificing existing features. Taking a look at all signals transmitted through the cable in our current design:

- 3 groups of video signals for OLED panels
- OLED power lines
- Touch controller USB
- Polhemus Fastrak sensor

The most difficult ones are high-speed digital video signals. It is difficult to wirelessly transmit customized high-speed signals without using existing reliable solutions. But there are some approaches we may consider: (1) Embed FPGA video controller inside pCubee and transmit conventional video signals from PC to an receiver, then forward to FPGA; (2) Build a small form factor computer within pCubee and manage all rendering and signal processing locally. Either case more circuitry has to be added to pCubee in order to boost the processing power on the device to interface with outside. This could result in increased weight and power consumption.

Another concern is the wired magnet sensor. Magnet tracking can provide very low latency and accurate tracking, but the sensors have to be physically connected through a cable. Optical tracking seems to be the best alternative wireless method which provides comparable accuracy and latency that meets our requirements, but it also creates inconvenience when the user rotates the display. Optical tracking relies on direct line of sight, but user's hand or the display itself may occlude optical trackers, therefore multiple marker has to be placed in order to cover different angles. Additional markers mounted onto the display in return make it difficult for user to handle the device.

Although we have to rely on cables between pCubee display and controller boards for now, it is possible for us to build wireless versions in future. Based on available technologies up to date, a possible path for newer versions of pCubee can be proposed. Several iterations are listed based on development efforts required.

1. Create a compact customized FPGA board that fits inside pCubee, which eliminates CableTX and CableRX boards. PC display output can be directly connected to the device.

2. Replace magnet tracking method with wireless tracking technologies including: optical tracking, combining gyroscope and accelerometer, IR tracking or active radar.

3. The last step, build a portable system inside pCubee, render all images without help of a PC and display them on pCubee. Attach a battery module for power supply, then external connections are no longer required.

On the other side, as the new pCubee is such a highly customized design, extensibility is very limited. For example, the framebuffers are designed to exactly fit our 4.3-inch 480 * 272 OLED screen, if other resolution or pixel format is used, the framebuffer module as well as all memory address generators have to be redesigned according to the new panel. One of the future directions would be making a general FPGA driver that supports different types of panel and different number of display.

Investigation into other display panel and interconnect technologies reveals several alternative solutions available. If applied appropriately, these technologies could benefit pCubee with less connections, higher resolution and framerate, as well as improved signal quality.

A commonly used interface for non-monitor LCD/OLED panels is Display Serial Interface (DSI) specification by the Mobile Industry Processor Interface (MIPI). DSI defines a high-speed differential signaling point-to-point serial bus and a communication protocol between the host and the device. It could be used to replace the parallel connection between FPGA and OLED panels by adding DSI transmitter and receiver at each end. Compared to our current solution, DSI interface allows longer cables with improved signal quality, less connections and higher bandwidth, all thanks to its differential signaling serial bus. Design effort is also reduced as many

new display panels in the market have MIPI DSI receivers integrated with the module already. However, the OLED panels used in our current system require customized signals which causes difficulties designing DSI receiver board, therefore parallel connections are our only choice due to this limitation, unless new display panels with built-in DSI were used.

Starting from DisplayPort version 1.2, the protocol supports daisy-chain connection with multiple monitors called Multi-stream Transport, allowing the user to stream independent video displays from a single video output. Based on bandwidth limits, up to five 1680 x 1050 displays (or even more with lower resolution) can be daisy-chained to a single DisplayPort output. Although the technology is yet to mature with very limited controller chips supporting such a feature, it does offer the possibility of eliminating the additional FPGA video controller in pCubee for separating video streams. Video for all panels could be transmitted via only one DisplayPort connection from the host and selected by each panel driver internally, resulting in greatly reduced number of components, routing complexity and power requirements.

3.4 Summary

In this chapter, we described our design of the new pCubee hardware including OLED screen controller, peripheral supporting circuits, touch controller and 3D printed chassis. Several improvements over old pCubee were made to provide the user a better experience when interacting with the system. We also discussed the trade-offs and limitations in different aspects of the new hardware as well as possible alternative future solutions.

On software side, the new pCubee incorporats an abstraction layer for real time physics simulation, which provides the possibility of switching to different physics engines if necessary. A selection of simple cross-screen multi-touch gestures are also supported, allowing the user to use multiple fingers to translate and rotate a 3D object.

With issues discovered in previous version of system being addressed, the new pCubee design enabled us to further investigate novel interaction techniques made possible with this unique type of geometric 3D display.

Chapter 4

Evaluation of Trajectory-Based Tasks in pCubee

Despite identifying improvements in various aspects in our redesigned pCubee as a display, we would also like to find out whether the user can benefit from the new system when interacting with objects in virtual environment. While pCubee supports various tasks ranging from basic pointing and selection to complicated interactive tasks such as virtual painting and scene navigation, this thesis will focus on a common task that is used frequently in 3D environment and can reflect unique characteristics of our hand-held geometric 3D display.

As Accot et al. points out in [1], trajectory-based interactions such as drawing curves, navigating through nested menus, and moving in 3D worlds, are common tasks in modern virtual reality interfaces. They designed a 2D tunnel task to model user's performance when moving a cursor through a tunnel without colliding with it. They also explored other cases including narrowing tunnels, curved tunnels, as well as spiral tunnels to further derive a general description of the law. The resulting model for 2D trajectory-based tasks called the steering law can be further used to evaluate and compare different input devices [2].

Evaluating trajectory-based interaction in 3D becomes a bigger challenge because of additional efforts required to control in the third dimension in order to follow a desired trajectory. While people move objects in 3D all the time effortlessly in real world, manipulating objects to navigate along

an existing 3D path is not as simple. Accurate and comprehensive understanding of the target trajectory is required. Close feedback to make path corrections based on real time observation is also essential, making the task particularly difficult in virtual reality interfaces due to insufficient depth feedback from traditional display. Furthermore, most users are more experienced working with 2D input devices on a day-to-day basis, such as regular mouse and trackpad, but are rather new to 3D input devices and would require reasonable amount of time to familiarize the hand movement and visual feedback. This also adds difficulties for general users to work with 3D in virtual environment [5, 15, 16, 28].

Considering the benefits brought forth by pCubee such as the flexibility of manipulating the display and virtual scenes inside, an intuitive way of changing perspective, as well as the strong motion parallax effect, we would like to investigate how 3D trajectory-based task is accomplished in pCubee.

In this chapter, we discuss the design of a wire-tracing task in pCubee and two studies we conducted to investigate user performance with our device. Statistical effects of three factors–ring radius, wire length and wire curvature are tested. Results for each study are reported and discussed respectively. A derived steering model inspired by Accot's work [1] is proposed for 3D trajectory-based interaction based on our findings in these studies.

4.1 Wire-Tracing Task

Target acquisition task such as pointing and positioning is one of the most common task used in virtual reality environments. Fitt's law [11] describes the relationship between acquisition time of a target and the target's dimension and distance which reflects the efficiency of particular input device.

$$T = a + b\log_2(\frac{A}{W}) \tag{4.1}$$

where:

- T is the time to complete the movement.

- a and b are regression coefficients.
- A is the distance from the cursor starting location to the center of the target.
- W is the width of the target measured along the axis of motion.

In addition to target acquisition, trajectory-based manipulation are also commonly used when controlling objects in virtual reality systems such as pCubee. However, Fitts' law, while being a robust regularity for pointing tasks, could not be applied to model user's performance in these trajectory-based tasks. Accot et al. [1] proposed the steering law for 2D tunnel tasks based on Fitt's model:

$$T = a + b\frac{A}{W} \tag{4.2}$$

where the tunnel has length of A and width of W, a and b are regression coefficients, T is the time to complete the movement. Index of Difficulty (ID) is defined as $\frac{A}{W}$. Using this new model, Accot et al. evaluated several input devices [2] and compared their Index of Performance (IP) defined as $IP = \frac{1}{b}$.

To evaluate trajectory-based task performance in 3D display such as pCubee, we could not simply extend the 2D tunnel task to 3D. It is difficult to visualize a 3D cursor inside a tunnel without affecting depth perception– rendering a translucent tunnel or subtracting the front half of the tunnel wall will create difficulties for user to judge the distance between 3D cursor and the wall, therefore can cause frustrations and unexpected failures.

On the other hand, although very challenging, the tunnel task is a good abstraction to represent a common type of tasks where a constrained trajectory has to be followed. For example, in 2D, navigating through cascaded sequences of menu items, or keeping a target within moving boundaries. 3D examples include some medical surgeries where a pill or something else has to be delivered to certain target without touching anything on its way.

Wire maze (Figure 4.1) is a popular toy that requires the user to move

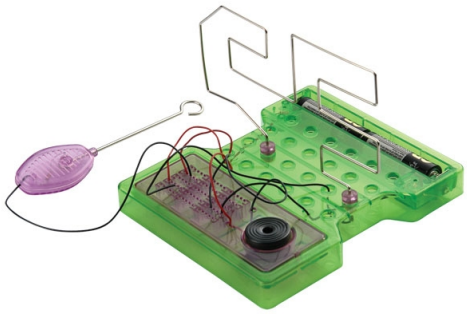


Figure 4.1: An example of wire maze game device.

a metal ring along a curved wire without the two objects touching each other. Whenever a collision is detected (usually by detecting electrical short between the ring and wire), warning light or buzz sound will go off as a notification. It is a fun game designed to practice one's ability to coordinate visual and hand movement. The task may seem to be easy, but it is in fact very challenging to avoid collision when the ring is small enough and the wire becomes complicated.

Inspired by the physical wire maze game, we design a constrained wire-tracing task to replace tunnel task in 3D based on the interaction model of the physical game: the user controls a virtual ring to move it along a given wire without both objects touching each other in 3D space visualized inside pCubee. Figure 4.2 shows an example of the task. A similar real world task would be adding beads or rings to a thread.

As Figure 4.3 indicates, if we imagine a flattened ring around mouse cursor, the original tunnel task with a tunnel width of D is equivalent to a 2D version of wire-tracing task with a $D/2$ radius ring around a zero-thickness

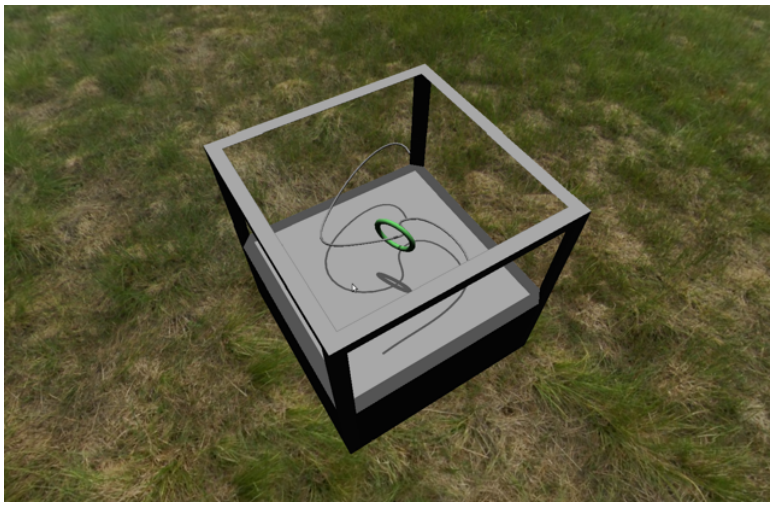


Figure 4.2: Wire-tracing task visualized in pCubee

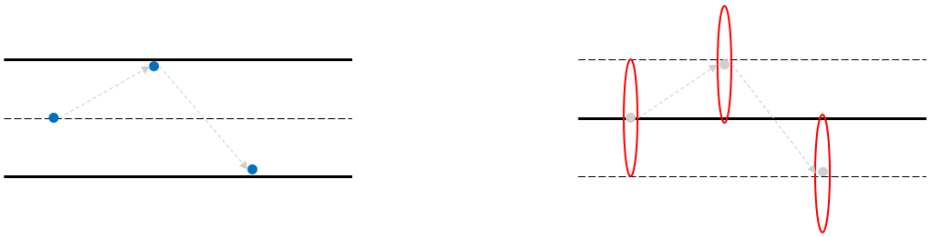


Figure 4.3: Comparison between 2D tunnel task and wire-tracing task.

wire. Therefore we would like to test if findings from Accot's 2D tunnel task are still valid for 3D wire-tracing task and whether any modification is needed. To be exact, in our studies we would like to test the effects of three potential factors on user performance: (1) radius of the ring, (2) length of the wire, and (3) curvature of the wire.

4.2 Experiment Design

4.2.1 Apparatus

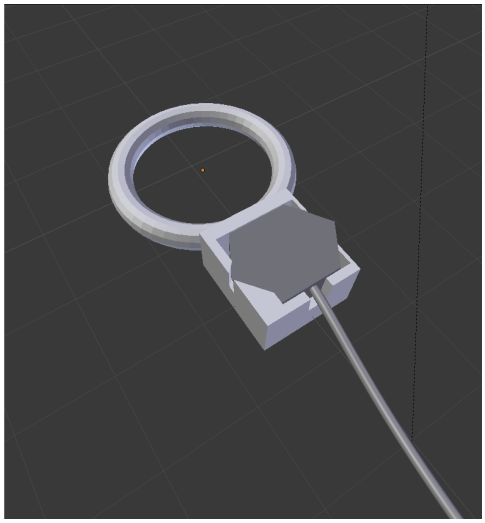


Figure 4.4: 3D print model of physical ring attachment on magnet sensor. User controls virtual ring by directly manipulating the sensor.

All our studies are conducted using the new pCubee system as described in Chapter 3. The experiments are set up on a desk where the subjects performed the tasks while seated as they were using a PC, with elbows rested against desk. The users are allowed to freely interact with the pCubee display. Head tracking is done by a magnetic sensor mounted on top of a cap that the subject wears for the duration of the experiment. A keyboard interface is used for the experimenter to control the tasks. A traditional LCD monitor is used to visualize the scene from user's perspective for the experimenter to observe the interaction and provide guidance during training session.

In wire-tracing tasks, the wire is fixed in position with outer frame of pCubee that can be moved around with the user's non-dominant hand, while the ring being controlled by an extra sensor on the user's dominant hand

moves freely in space within the boundary of pCubee. A physical ring fixture is attached to the sensor to mimic wire maze game and indicate sensor orientation as shown in Figure 4.4.

4.2.2 Procedure

The experiment procedure for each subject is illustrated in Figure 4.5.

When a task starts, the user is allowed to pick and move the ring around in pCubee freely. Timing will start as soon as the user begins to move the ring along given wire. This is triggered by detecting the intersection between wire segments and an invisible flat disc located in the ring with the same radius and orientation. When intersection ends at the other end of wire, a new time stamp will be marked and the duration of current trial along with all collected data will be recorded.

Collision between ring and wire is detected by traversing through the scene and reporting intersection between any triangle of the ring geometry and line segments of the wire. Based on user feedback, collision detection is temporarily turned off for the first few seconds when a new task is loaded to avoid unexpected collision when object position resets.

If any collision are detected during the movement, current trial will stop with the wire and floor of pCubee illuminated in red to indicate the failure. The incomplete record as well as conditions of current trial will still be reported but marked as a failed attempt. After 2 seconds delay, the system will automatically load a new task. User can choose whether they would like to be given another chance to repeat the same trail again or move forward to another one and come back at a later time.

If the user finds the task too difficult to accomplish, he or she could request to skip current trial instead of spending extra amount of time figuring out the task in frustration. A failed attempt rather than an extraordinary long task time will be reported, so that our statistical analysis on task completion time won't be affected.

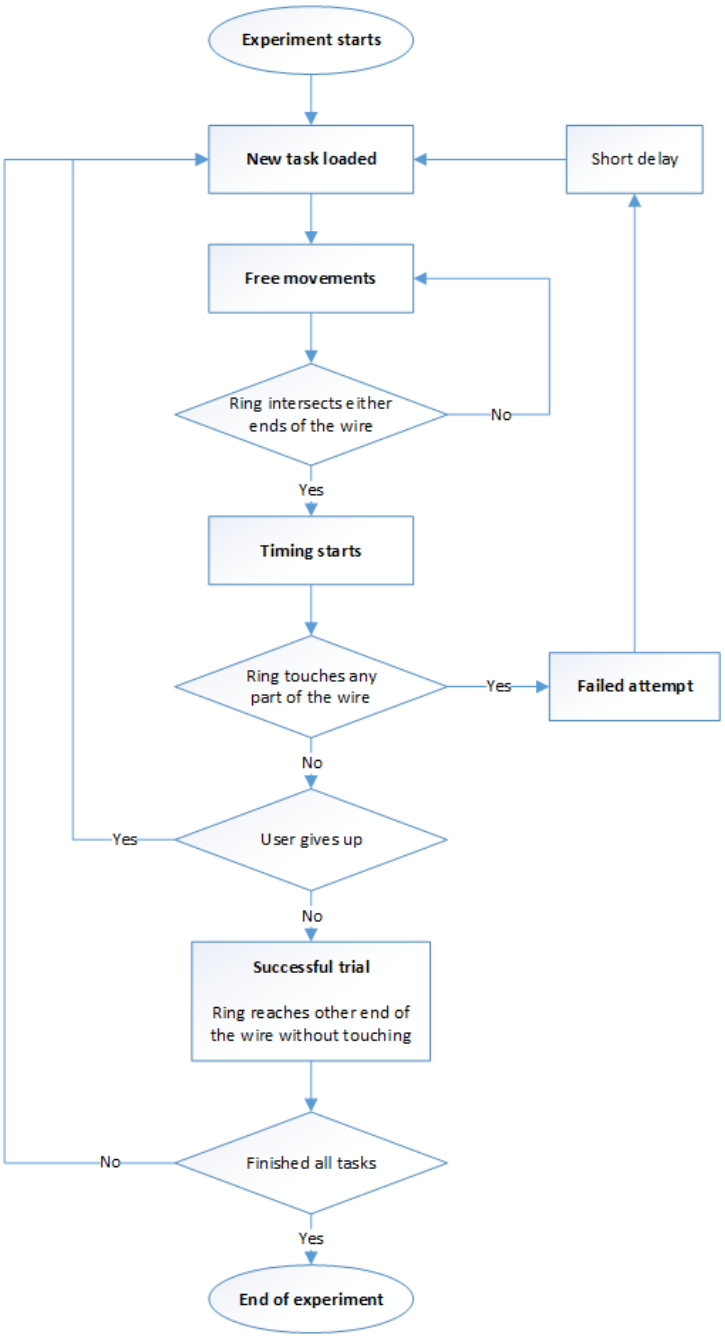


Figure 4.5: Flow chart of experiment procedure.

Only when user successfully moved the ring all the way along given wire without any collision detected during the process, pCubee floor will illuminate in green to indicate a successful task completion. System will notify end of experiment when target number of conditions are all tested.

4.3 User Study 1: Effect of Ring Radius

From experience in real-life wire maze game, we know the task is more difficult when a small ring rather than a larger one is used because more careful control is required. We would like to see if the same happens in virtual environment and further understand how ring size affects task performance. To investigate the statistical effect of ring radius on completion time for wire-tracing tasks, we conducted a controlled study where subjects are asked to move rings of different size along the same wire without collision as fast as possible. In this study, other major factors (length and curvature of wire) that we believe may affect user performance stay at appropriate fixed values. Orientation of the ring is also locked at perpendicular position to the wire regardless of any tracker rotation. Only 3-DOF positional data from the 6-DOF tracker is used to control the ring. This will ensure that the effective radius (determined by projected radius on the perpendicular plane of wire) remains the same without being affected by ring rotation.

4.3.1 Conditions

As discussed above, fixed values are chosen for length and curvature of the wire to avoid effects of other factors. We decided to use a straight 80 mm wire placed horizontally in the middle of pCubee for all the tasks in this study, as it is (1) long enough to ensure the tasks are not too easy, and in the same time (2) allows enough space for subjects to move and align the ring without accidental collisions.

The form-factor of new pCubee determines that our study will be limited

within a 100 mm * 100 mm * 60 mm 3D environment. Comparing to Accot's tunnel task which is done on a 19-inch (482.6 mm diagonally) 2D monitor and 18 * 25 inches tablet, object movement in our device becomes more constrained. This creates a challenge to choose appropriate values for ring radius: a too small ring will make it extremely difficult to avoid collision, while a too large one will suffer from very limited movement inside our cubic display.

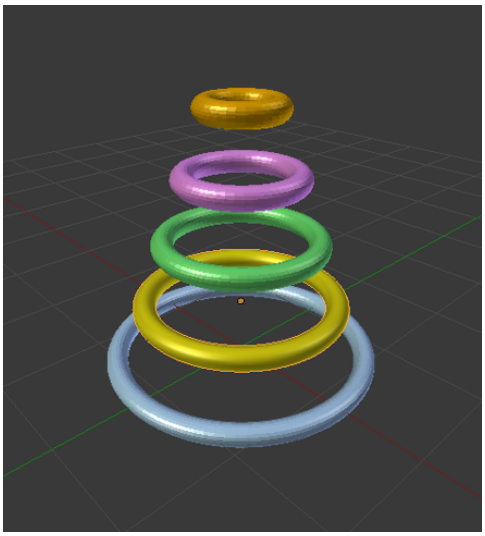


Figure 4.6: Five rings with different radius used in our user study.

In order to find the right balance, we started a pilot study testing 5 different ring radius: 3 mm, 5 mm, 7 mm, 9 mm and 12 mm. As shown in Figure 4.6, rings are rendered in gold, purple, green, yellow and blue respectively to be easily distinguished. We recorded number of failed attempts for each radius and compared them with total number of attempts. As indicated from table 4.1, a 3 mm ring causes as much as 71.5% of failed attempts which is significantly higher than other conditions (all below 40% with an average of 32%). After careful consideration we decided to drop 3 mm radius condition as it appears to be too difficult for pilot users and could not reliably reflect their performance.

Ring radius	Total attempts	Failed attempts	Failure rate
3 mm	330	236	**71.5%**
5 mm	307	114	37.0%
7 mm	384	131	34.2%
9 mm	339	101	29.8%
12 mm	339	93	27.4%

Table 4.1: Comparison of failure rate of 5 different ring radius.

On the other hand, a 12 mm radius ring is already large enough because we noticed that pilot users have to carefully move the ring around when not crossing with wire, especially when the ring is below the wire, as the 30 mm space barely allows a 24 mm diameter ring to pass.

In conclusion, based on results and observations from pilot study, we decided to use rings with 5 mm, 7 mm, 9 mm and 12 mm radius for actual user study.

4.3.2 Method

The experiment used a within subjects design due to limited amount of participants available. Independent variable, or within subjects factor, is radius of the ring. The principal dependent variable is measured task completion time. The experiment consists of 15 trials per condition and 4 different radii, resulting 60 total trials for each participant. Tasks are randomly ordered to counterbalance any learning effect.

We instruct the subjects to perform the tasks as fast as possible without failing. The experiment typically takes 20-40 minutes to complete depending on subject's performance. A 5-10 minute training session is included before actual measurements are taken, where subjects can familiarize themselves with controlling both pCubee display and the extra sensor. Tasks of various difficulties are available during the session for them to try out. The experimenter also helps clarify any questions or confusions.

A total of 7 subjects with little or no experience with 3D input devices

participated in the experiment. After completion of all tasks, post-study questionnaires were given to our participants to collect subjective feedback regarding their interaction experience. We designed 11 7-point Likert scale (-3 to +3) questions to gauge subject feedback as listed below. Open-ended questions (item 6, 7 and 8) are also included to survey opinions and suggestions on new pCubee design.

1. How would you rate the overall difficulty of this task?
2. How would you rate your performance on this task?
3. Difficulty and intuitiveness of:
   (a) Visualizing the wire maze in virtual environment
   (b) Manipulate virtual ring using physical sensor
   (c) Moving the ring
   (d) Rotating the ring
   (e) Handling pCubee
4. Do you find following feature helpful:
   (a) Collision highlighting
   (b) Shadows
   (c) Bimanual manipulation
   (d) Motion parallax
5. What do you think contributes the most to the difficulty of the task? (wire length, curvature, or ring radius)
6. What feature do you think should be added to better help you complete the task?
7. What improvements would you expect on next version of pCubee?
8. Other comments.

4.3.3 Results

Within subjects one-way ANOVA was carried out to test whether ring radius has any effects on task completion time. Each radius group consisted of 7 samples, which is the average time of 15 trials for each participant. A statistical significant difference was found between radius groups ($F(3, 24) = 3.8800, p = 0.0215$). Post-hoc pair-wise test revealed that mean time of 12 mm radius group ($M = 827.24, SD = 340.41$) is significantly shorter than 5 mm group ($M = 2419.41, SD = 1331.98$), but no significant differences between other pairs were found.

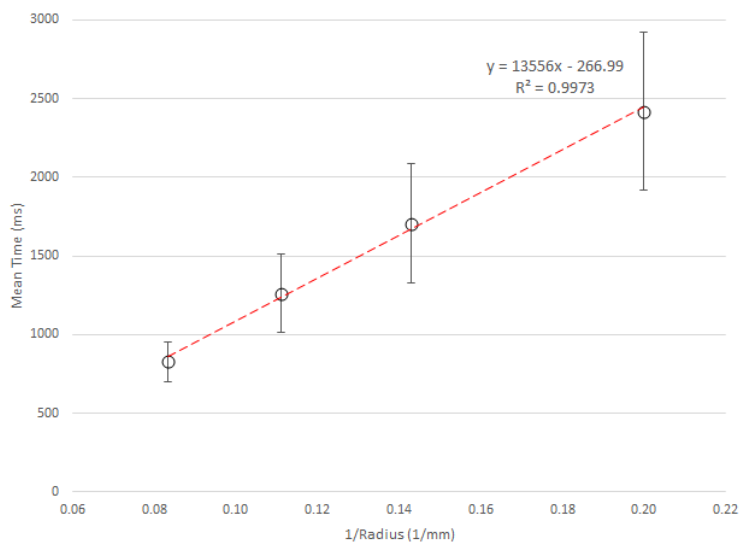


Figure 4.7: Linear regression of Mean Time v.s. 1/Radius.

Linear regression is performed on $1/R$ vs Mean time 4.7, results indicate a linear relationship $MT = 13556 * 1/R - 266.99$, with correlation coefficient $r^2 = 0.9973$.

Table 4.2 summarizes all objective questionnaire responses.

4.3.4 Discussions

Although we successfully proved the effect of ring radius on subject's average task completion time, the statistical significance observed in our study is not very strong ($p = 0.0215 < 0.05$). The small number of samples (only 7 participants) should be one of contributing issues. We also noticed that the variance between subjects are relatively large, indicating a large individual difference on task completion time. For example, Subject 2 takes averages of 1331, 1852, 3151 and 4838 ms to complete the task for each condition, while subject 3 only spends around 1/3 of the time (438, 539, 824, 1341 ms respectively) to do the same job. This huge variance as well as limited number of samples both impacts our analysis results, although an obvious within subjects trend can be observed.

We conducted our post-hoc analysis using two methods, pair-wise t-test with Bonferroni correction and Tukey HSD. Bonferroni correction tends to be more conservative and gives more false negatives in most cases. But for our study, both method provide the same result showing that only 5 mm group and 12 mm group have significant difference. It also indicates that variance between conditions is not large enough compared to between-subjects variance. Choosing conditions further apart from each other may help. However, as discussed earlier, the radius range is limited in our study therefore it is impossible to test conditions with larger or smaller rings.

It is also worth noting that between-subjects variance increases as radius gets smaller, demonstrating that user performance varies more for harder tasks but appears less diverse for easier ones, which may indicate that factors such as personal experience or task strategy plays a more important role as the task difficulty rises.

On the other hand, linear regression shows clear linear relationship between $1/R$ and Mean time:

$$T \propto \frac{1}{R} \tag{4.3}$$

suggesting that ring radius R might be equivalent to tunnel width W in steering law model.

Questionnaire results

Questions	Subject#1	#2	#3	#4	#5	#6	#7	Average score
Q1	-3	-3	-3	-2	-3	-2	-1	-2.43
Q2	-3	1	-1	0	-2	0	1	-0.57
Q3.1a	-3	1	2	1	-2	2	1	0.29
Q3.1b	-1	1	1	1	-2	1	1	0.29
Q3.2a	-2	-1	0	-2	-2	1	-1	-1.00
Q3.2b	0	3	0	0	2	1	2	1.14
Q3.3a	1	2	2	1	-2	2	2	1.14
Q3.3b	1	3	1	1	1	2	2	1.57
Q3.4a	-1	-1	2	-2	-1	2	-2	-0.43
Q3.4b	-1	3	2	1	2	2	2	1.57
Q3.5a	-2	-1	-1	2	-2	1	0	-0.43
Q3.5b	-1	3	-1	2	2	1	2	1.14
Q4.1	3	3	3	-1	2	2	2	2.00
Q4.2	3	2	1	3	1	3	1	2.00
Q4.3	-1	-2	-1	2	-2	0	1	-0.43
Q4.4	2	2	1	-2	-2	1	2	0.57

Table 4.2: Response and average score for each 7-point Likert scale question, ranging from -3 to 3.

Responses of questions in our survey also reveal some useful information.

- All participants consider the task to be challenging.
- Three participants are not satisfied with their performance on the task.
- Manipulating virtual ring using the physical sensor is intuitive, but also difficult.
- Moving and rotating the ring is intuitive.
- Participants also found object shadows and collision highlighting very helpful.

According to their feedback in open-ended questions, the factors contribute most to the difficulty of the task include:

- Cable attached to sensor and pCubee
- Offset between virtual ring and the sensor
- High curvature of the wire

As for improvements and suggestions, 6 out of 7 participants mentioned that they would like a wireless sensor to manipulate. Two participants also requested highlighting contact point between wire and virtual disc on the ring to assist determining distance.

4.4 User Study 2: Effects of Wire Length and Curvature

Despite ring radius, we observed that wire length and curvature may also be factors that determine task difficulty and impact user performance on physical wire maze games. In order to investigate their influence on wire-tracing task in pCubee, a second experiment under slightly different conditions was conducted to test the effect of wire length and curvature on task completion time. We are not able to test all three factors in a single experiment because the combination of three factors will result in too many conditions, and between-subjects design is not possible due to limited time and resources available. Within-subjects design may lead to lengthy studies that easily causes fatigue and frustration for subjects, especially when the task itself is both mentally and physically challenging. Therefore, we conducted two experiments to test the factors separately so that both experiments would be feasible.

4.4.1 Conditions

To rule out effects of ring radius, a certain value has to be chosen for all tasks. We selected 9 mm as the ring radius used in this study for following reasons:

- As demonstrated by previous study, a larger ring makes the task easier and gives us smaller between-subjects variance. It will cause less distraction for us to identify the impact of actual factors being tested.
- At the same time, we want to maintain an appropriate difficulty level in our tasks. If the ring is too large that makes it effortless to cross in conditions with shorter wires, it may affect test results. Considering our shortest wire is only 31 mm, a 9 mm ring is preferred than a 12 mm one as the latter takes as short as 0.3 second to cross entire wire based on data from previous study.

Figure 4.8: Nine wires with different length and curvature used in our user study.

Three wire lengths are selected: 10π (31.4) mm, 20π (62.8) mm and 30π (94.2) mm. The longest one requires some rotation to fit into our 100 mm * 100 mm * 60 mm cubic virtual space. All other wires with different length and curvature are randomly rotated and positioned within the space, while leaving sufficient area for the ring to align and travel through.

Curvature of the wire is defined by the radius of osculating circle (ROC). For simplicity we only tested smooth curves in our study, which means the wire has only one osculating circle of a certain radius and is always part of this osculating circle. For example, a 20π mm length wire with 40 mm ROC is actually quarter of a 40 mm radius circle. To ensure our study covers nearly straight as well as more curved conditions, we selected 40 mm, 80

Wire Number	Length (mm)	ROC Curvature (mm)
1	10π	160
2	10π	80
3	10π	40
4	20π	160
5	20π	80
6	20π	40
7	30π	160
8	30π	80
9	30π	40

Table 4.3: User study: wire conditions.

mm and 160 mm ROC to be combined with different lengths, creating 9 total conditions. All wires used in this study are shown in Figure 4.8.

4.4.2 Method

The experiment uses a 3 * 3 within subjects factorial design for three lengths and three curvatures. The subject will conduct at least 10 trials under each condition, resulting in a total of 90 trials for each participant. All trials are randomly ordered to counterbalance learning effect. Task completion time is measured the same way as previous user study. All failed attempts are recorded and another trial will be provided at a later time, unless the subject explicitly requests to skip the condition. Upon completion of all trials, we conducted a short interview session with each subject and asked general questions to gather their feedback. In total, 8 subjects were recruited to participate in this study. A small amount of compensation was provided for their time and effort.

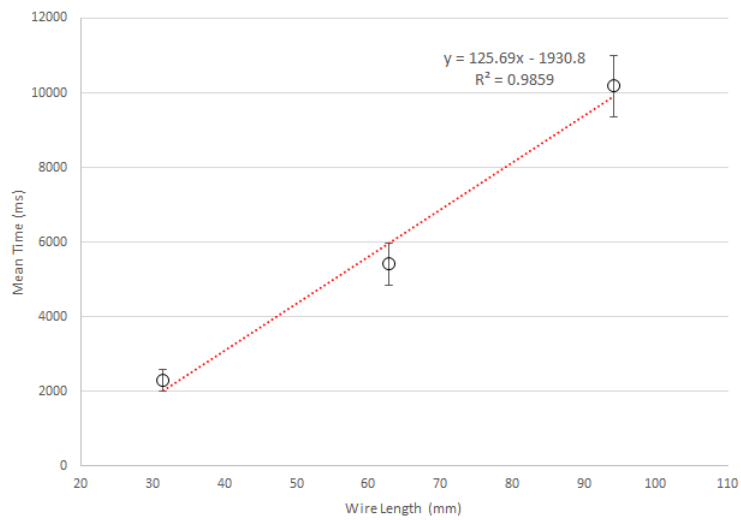


Figure 4.9: Linear regression of Mean Time vs Wire Length.

4.4.3 Results

Two-way within subjects ANOVA was conducted on 9 length and curvature combinations. A significant main effect of length was found ($F(2, 63) = 45.6459, p < 0.0001$), however there are no significant effect of curvature ($F(2, 63) = 2.9685, p = 0.0586$) or interaction effect between length and curvature ($F(4, 63) = 0.3719, p = 0.8278$). Post-hoc test using Tukey HSD revealed that 10π mm length group ($M = 2288.18, SD = 1393.32$) is significantly faster than 20π mm group ($M = 5418.57, SD = 2781.66$). Both groups are also significantly faster than 30π mm group ($M = 10181.80, SD = 3969.51$).

Linear regression (Figure 4.9) was conducted on length factor which demonstrated a significant effect. The correlation coefficient $r^2 = 0.9859$.

Table 4.4 summarizes percentage of failures under each length and curvature condition based on all data collected during the experiment.

Length	Total attempts	Failed attempts	Failure rate
10π mm	284	33	11.6%
20π mm	338	87	25.7%
30π mm	457	228	49.9%
ROC	**Total attempts**	**Failed attempts**	**Failure rate**
40 mm	369	131	35.5%
80 mm	373	123	33.0%
160 mm	337	94	27.9%

Table 4.4: Comparison of failure rate of each length and curvature condition.

A clear trend can be observed that as the wire gets longer, percentage of failed trials increases as well. Also, as the wire gets less curved, percentage of failure slightly drops.

4.4.4 Discussion

As we have learned from our previous experiment, between-subjects variance is relatively high due to factors that we could not control in this study. Through our observation and follow-up discussion with participants, we would like to identify the following noninclusive list of possible factors that may be related:

1. Although all participants claim that they have little or no experience with 3D input devices before, some of them are apparently more comfortable working with pCubee and using the sensor to control accurately while others spend quite a long time struggling with the device to do what they expected. One participant pointed out that it may be the result of previous training on certain musical instruments or other crafting skills that requires accurate hand movement and close hand-eye collaboration.

2. Some participants can easily learn to use the system and get good at manipulating the ring in a short time, while others still have trouble

understanding the direct mapping relationship until the end of experiment. This personal difference in learning practice is very difficult to capture before the experiment. This factor may be eliminated if a long enough training period is introduced but time and resource constraints would not allow such a plan.

3. Different subjects use different strategies on completing the task. We noticed that some subjects take full advantage of free viewpoint by carefully observing the 3D scene from various angles before making a continuous move, while others are more conservative and prefer a stable viewpoint and figuring out object's 3D relationship by trial and error with small movements. Some even combine both methods by first looking for a comfortable viewpoint, stay there and do a small move, then change to a new viewpoint and move again, etc. Compared to tunnel task study, wire-tracing task in pCubee gives user more flexibility to interact with virtual objects at expense of greater between subjects variation in performance measurements.

Despite the similar between-subjects variation as we witnessed in our first study, the wire length factor still exhibits a strong effect and linear relationship to task completion time. In contrast, wire curvature shows a weak effect at 0.05 significance level. The result actually matches Accot's steering law model, where wire length L and ring radius R appear equivalent to tunnel length A and width W respectively. Also, as pointed out by Accot et al. [1], for a curved tunnel, $T_C = a + b \int_C \frac{ds}{W(s)}$, where T_C is the time to travel across the tunnel and C is the path, if tunnel width $W(s)$ is constant and equals to W, we have $T_C = a + b \cdot \frac{ds}{W(s)}$, telling that curvature of the tunnel does not affect the time to steer through path C.

Combining results from both studies, we can confirm that wire-tracing task follows a model similar to 2D steering law:

$$T = a + b \cdot \frac{L}{R} \tag{4.4}$$

where L is length of the wire, R is radius of the ring, and T is the time for

the ring to travel through wire.

4.5 Summary

In this chapter, we reported our experiments of trajectory-based task performance in redesigned pCubee. Effects of three potential factors – ring radius, wire length and wire curvature – were investigated by testing user performance in a wire-tracing task visualized inside pCubee. We carried out two experiments with different conditions to limit the total number of trials in one user study to make it feasible. Subjective feedback was reported in post-study questionnaires.

Analysis of recorded user performance data revealed that radius of the ring and length of curved wire both have significant effects on task completion time, while curvature of the wire shows little effects in our study. It was also proven that average task completion time is proportional to both the reciprocal of radius and the length of wire respectively. Although the statistical effects were identified, we observed large between-subjects variance in our data. Possible issues such as prior personal experience not related to 3D manipulation, task strategy and effects of learning curve were explored and discussed. Results from our survey also indicated that the task was very challenging that half of the participants were not satisfied with their performance. Feedback about the task pointed us to difficulties caused by the attached cable and the offset of physical sensor.

Based on these results, we confirmed that 3D trajectory-based interaction follows a similar model as Accot's 2D steering law, in virtual 3D space visualized in perspective-corrected geometric display.

Chapter 5

Conclusions

In the first part of this thesis, we discussed the design and implementation of new pCubee system. We learned from previous studies about limitations of old system and designed a new version of hand-held perspective-corrected cubic display system to bring in thinner borders and better display quality. The new design helped effectively reduce visual discontinuity that impacts user experience when interacting with virtual objects inside pCubee.

We investigated 3D manipulation tasks supported by new pCubee, evaluated user performance on a trajectory-based wire-tracing task implemented inside the virtual environment. By varying ring radius and wire length, we identified a linear relationship between movement time and these factors, suggesting a model similar to the steering law for 2D tunnel task also applies in 3D trajectory-based tasks. The experiment proved that certain 2D interaction principle applies to 3D tasks in perspective-corrected geometric displays as well, providing an effective model for researchers to further investigate user interaction designs and hardware revisions for similar display technologies.

In this chapter, we discuss the potential areas where the new pCubee system can be used to fill the gap between virtual environment and real world. We also summarize a number of challenges we encountered throughout the research and promising directions for future investigation.

5.1 Potential Applications

Being a hand-held system, pCubee not only has all the benefits of a perspective-corrected geometric display, but also demonstrate unique advantages such as easier manipulation and portability. This enables the system to be used in a variety of scenarios where other solutions might not be effective.

Personal Entertainment

Gaming is one of the dominating areas where virtual reality technology is widely used and continuously improved. While most gaming system still stick to a display+controller model, more and more devices bring in new interactive features including perspective correction. For example, Nintendo 3DS used camera sensor to determine the user's viewpoint and provided a head-coupled gaming experience. The form-factor of pCubee makes it a promising candidate for personal hand-held consoles, bringing a stronger illusion of virtual world as well as a more intuitive method of interacting with objects in games such as spacial puzzles, or navigation-based role playing.

Design and Prototyping

Designing in 3D space is a challenging task using 2D user interfaces. Multiple widgets have to be utilized to build constraints and map inputs. pCubee allows designers to create or modify 3D content directly in a realistic 3D space. Using direct mapping, the user would be able to control a virtual stylus to do 3D drawing, sculpturing and assembling. Most 3D model file types are supported by pCubee and can be easily visualized inside the cubic display. Even novice designers can preview 3D model using pCubee, make necessary adjustments directly without going back to complicated model creation tool, and finally publish the design or send it out for 3D printing.

Scientific Simulation and Visualization

Highly sophisticated scientific data, for example, protein structure, medical model of human body, network model etc., are difficult to be visualized in

traditional 2D displays. Researchers can take advantage of head-coupled perspective of pCubee system and navigate through these data intuitively without spending a lot of effort controlling their point of view. Furthermore, they can select, mark and manipulate the 3D data with direct mapping, turning pCubee into a interactive simulation platform for tasks such as surgery operations.

Collaborative Working

Although pCubee itself does not support multiple viewers due to limitations in perspective-corrected display technology, it still demonstrates great potential for collaborative content creation and editing. Multiple users can share a common virtual work space visualized in each of their own pCubee devices, while individual user working with their unique view point and leveraging all the benefits of this portable display.

5.2 Future Directions

Here we summarize a number of challenges we encountered and limitations we identified throughout the design, implementation and evaluation of the new pCubee system. Based on our current work, we would also like to point out promising research directions for further investigation.

5.2.1 Tracking Technology

While the visual discontinuity problem from previous pCubee is taken care of in new pCubee system, there are still remaining issues that impact user experience that need to be accounted for.

One of the major issues is calibration of the tracking system. The Polhemus magnetic tracker we used provided great accuracy and minimum latency, but as with all systems operating on magnetic fields, it suffers from interference of objects within its working area, especially metals which can cause distortion in the field. As the system becomes more compact, less

space between the sensor and metal materials in circuit boards and display panels can be guaranteed. On the other hand, to maintain signal integrity, large ground planes or shielding cans have to be placed to ensure proper grounding. We tried placing the sensor on the furthest corner of the display and using a 3D-printed plastic frame, but we could still observe some jitter in tracking data as a result of interference from multiple sources.

In addition to jitter, calibration is also affected by the position mapping from magnetic tracker to actual eye position. We can only estimate subject's eye position by applying a fixed offset from the tracker position located on a cap worn by the subject. Due to individual difference, the offset needs to be adjusted for each participant through an additional calibration process.

Moreover, it is reported from multiple participants that the cables caused difficulties when manipulating both pCubee and the sensor, especially when rotating to certain angles. A wireless system would be helpful to eliminate this issue.

A possible solution for these issues would be a different tracking system which may directly tell user's eye position without using a magnetic sensors. Depth camera or multiple camera arrays seem to be promising substitutions, but their tracking accuracy, latency as well as refresh rate must be carefully evaluated before further implementation.

5.2.2 Cross-Screen Multi-Touch

In this system, a preliminary multi-touch interface was implemented supporting very limited number of gestures for translation and rotation tasks. It demonstrated the possibility of interacting with virtual objects by directly touching their images on the display. Only a single touch can be detected on each panel due to the resistive touch panel available, preventing more complicated and useful gestures to be implemented. Our work set an initial step on investigating this unique and promising way of interacting with virtual environment.

Camera-based finger tracking or capacitive touch panel could be consid-

ered to overcome the limitation of resistive touch technology, bringing in smooth and accurate multi-touch experience. It is also worthwhile to further explore how this unique multi-touch interface may help manipulating objects in pCubee.

5.2.3 Trajectory-Based Tasks

As pointed out previously, tasks involving free movement in virtual 3D environment are inherently difficult due to lack of constrains and affordances. Same conclusion was reaffirmed in our wire-tracing task, as a large number of participants consider the task difficult to be accomplished. In fact, our preliminary test also showed that even for real-world physical wire puzzle game, users still had to make a lot of efforts to complete successfully, as a result of inadequate movement constrains. It would be useful to investigate both virtual and physical trajectory-based tasks with varying difficulty levels, and understand any performance difference as well as factors causing the gap. This could lead to deeper understandings on how people interact with objects when very limited movement constrains were available. Furthermore, it may inspire future researchers to identify the challenges in related interaction design and explore mechanisms that may help overcome these difficulties.

5.2.4 Comparison of Geometric Display Technologies

As steering law model has been confirmed for 3D trajectory-based manipulation task in pCubee, it is possible for us to expand the study to similar systems and understand the effectiveness of each design using the Index of Performance parameter from the model. It proves researchers a tool to explore and evaluate designs that better support trajectory-based tasks, and identify factors that contribute to an effective design.

Bibliography

[1] Johnny Accot and Shumin Zhai. Beyond fitts' law: models for trajectory-based hci tasks. In *Proceedings of the ACM SIGCHI Conference on Human factors in computing systems*, CHI '97, pages 295–302. ACM, 1997.

[2] Johnny Accot and Shumin Zhai. Performance evaluation of input devices in trajectory-based tasks: an application of the steering law. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, CHI '99, pages 466–472. ACM, 1999.

[3] Kevin W. Arthur, Kellogg S. Booth, and Colin Ware. Evaluating 3d task performance for fish tank virtual worlds. *ACM Trans. Inf. Syst.*, 11(3):239–265, July 1993.

[4] Doug A Bowman and Larry F Hodges. An evaluation of techniques for grabbing and manipulating remote objects in immersive virtual environments. In *Proceedings of the 1997 symposium on Interactive 3D graphics*, pages 35–ff. ACM, 1997.

[5] Doug A Bowman, Ernst Kruijff, Joseph J LaViola Jr, and Ivan Poupyrev. *3D user interfaces: theory and practice*. Addison-Wesley, 2004.

[6] Yeonjoo Cha and Rohae Myung. Extended fitts' law for 3d pointing tasks using 3d target arrangements. *International Journal of Industrial Ergonomics*, 43(4):350–355, 2013.

[7] Carolina Cruz-Neira, Daniel J. Sandin, and Thomas A. DeFanti. Surround-screen projection-based virtual reality: the design and imple-

mentation of the cave. In *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '93, pages 135–142, New York, NY, USA, 1993. ACM.

[8] Michael Deering. High resolution virtual reality. In *Proceedings of the 19th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '92, pages 195–202, New York, NY, USA, 1992. ACM.

[9] J.P. Djajadiningrat, G.J.F. Smets, and C.J. Overbeeke. Cubby: a multiscreen movement parallax display for direct manual manipulation. *Displays*, 17(3C4):191 – 197, 1997.

[10] Scott Fisher. Viewpoint dependent imaging: An interactive stereoscopic display. volume 0367, pages 41–45, 1983.

[11] Paul M Fitts. The information capacity of the human motor system in controlling the amplitude of movement. *Journal of experimental psychology*, 47(6):381, 1954.

[12] Tovi Grossman and Ravin Balakrishnan. Pointing at trivariate targets in 3d environments. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '04, pages 447–454, New York, NY, USA, 2004. ACM.

[13] Jonna Häkkilä, Maaret Posti, Olli Koskenranta, and Leena Ventä-Olkkonen. Design and evaluation of mobile phonebook application with stereoscopic 3d user interface. In *CHI '13 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '13, pages 1389–1394, New York, NY, USA, 2013. ACM.

[14] Kenneth P. Herndon, Andries van Dam, and Michael Gleicher. The challenges of 3d interaction: A chi '94 workshop. *SIGCHI Bull.*, 26(4):36–43, October 1994.

[15] Ken Hinckley, Joe Tullio, Randy Pausch, Dennis Proffitt, and Neal Kassell. Usability analysis of 3d rotation techniques. In *Proceedings*

of the 10th Annual ACM Symposium on User Interface Software and Technology, UIST '97, pages 1–10, New York, NY, USA, 1997. ACM.

[16] Stephanie Houde. Iterative design of an interface for easy 3-d direct manipulation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '92, pages 135–142, New York, NY, USA, 1992. ACM.

[17] Masahiko Inami. Media3: The virtual hologram. In *ACM SIGGRAPH 97 Visual Proceedings: The Art and Interdisciplinary Programs of SIGGRAPH '97*, SIGGRAPH '97, pages 107–, New York, NY, USA, 1997. ACM.

[18] Jaeho Kim, Sungwook Jung, Jaeho Choe, and Eui S Jung. A study on user interface design according to 3d menu type and control type for 3d displays. 30(4):551–562, 2011.

[19] Billy Lam, Yichen Tang, Ian Stavness, and Sidney Fels. A 3d cubic puzzle in pcubee. In *Proceedings of the 2011 IEEE Symposium on 3D User Interfaces*, 3DUI '11, pages 135–136, Washington, DC, USA, 2011. IEEE Computer Society.

[20] Billy Shiu Fai Lam. *pCubee: evaluation of a tangible outward-facing geometric display*. University of British Columbia, 2011.

[21] Hai-Ning Liang, Cary Williams, Myron Semegen, Wolfgang Stuerzlinger, and Pourang Irani. User-defined surface+motion gestures for 3d manipulation of objects at a distance through a mobile device. In *Proceedings of the 10th Asia Pacific Conference on Computer Human Interaction*, APCHI '12, pages 299–308, New York, NY, USA, 2012. ACM.

[22] Roberto Lopez-Gulliver, Shunsuke Yoshida, Sumio Yano, and Naomi Inoue. gcubik: real-time integral image rendering for a cubic 3d display. In *ACM SIGGRAPH 2009 Emerging Technologies*, SIGGRAPH '09, pages 11:1–11:1, New York, NY, USA, 2009. ACM.

[23] Anthony Martinet, Gery Casiez, and Laurent Grisoni. Integrality and separability of multitouch interaction techniques in 3d manipulation tasks. *Visualization and Computer Graphics, IEEE Transactions on*, 18(3):369–380, 2012.

[24] Michael J. McGuffin and Ravin Balakrishnan. Fitts' law and expanding targets: Experimental studies and designs for user interfaces. *ACM Trans. Comput.-Hum. Interact.*, 12(4):388–422, Dec 2005.

[25] Michael McKenna. Interactive viewpoint control and three-dimensional operations. In *Proceedings of the 1992 Symposium on Interactive 3D Graphics*, I3D '92, pages 53–56, New York, NY, USA, 1992. ACM.

[26] Miguel A Nacenta, Patrick Baudisch, Hrvoje Benko, and Andy Wilson. Separability of spatial manipulations in multi-touch interfaces. In *Proceedings of Graphics interface 2009*, pages 175–182. Canadian Information Processing Society, 2009.

[27] Norio Nakata, Naoki Suzuki, Asaki Hattori, Naoya Hirai, Yukio Miyamoto, and Kunihiko Fukuda. Informatics in radiology: intuitive user interface for 3d image manipulation using augmented reality and a smartphone as a remote control. *Radiographics*, 32(4):E169–E174, 2012.

[28] Graham Smith, Wolfgang Stuerzlinger, Tim Salzman, B Watson, and J Buchanan. 3d scene manipulation with 2d devices and constraints. In *Graphics Interface*, volume 1, pages 135–142, 2001.

[29] Peng Song, Wooi Boon Goh, William Hutama, Chi-Wing Fu, and Xiaopei Liu. A handle bar metaphor for virtual object manipulation with mid-air interaction. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '12, pages 1297–1306, New York, NY, USA, 2012. ACM.

[30] Ian Stavness, Billy Lam, and Sidney Fels. pcubee: a perspective-corrected handheld cubic display. In *Proceedings of the SIGCHI Con-*

ference on Human Factors in Computing Systems, CHI '10, pages 1381–1390. ACM, 2010.

[31] Ian Stavness, Florian Vogt, and Sidney Fels. Cubee: thinking inside the box. In *ACM SIGGRAPH 2006 Emerging technologies*, SIGGRAPH '06, New York, NY, USA, 2006. ACM.

[32] Dan S. Venolia and Lance Williams. Virtual integral holography. volume 1259, pages 99–105, 1990.

[33] Hans Wallach and DN O'connell. The kinetic depth effect. *Journal of experimental psychology*, 45(4):205, 1953.

[34] Colin Ware and Glenn Franck. Evaluating stereo and motion cues for visualizing information nets in three dimensions. *ACM Trans. Graph.*, 15(2):121–140, April 1996.

[35] Dongwook Yoon, Joong Ho Lee, Kiwon Yeom, and Ji-Hyung Park. Mobiature: 3d model manipulation technique for large displays using mobile devices. In *Consumer Electronics (ICCE), 2011 IEEE International Conference on*, pages 495–496. IEEE, 2011.

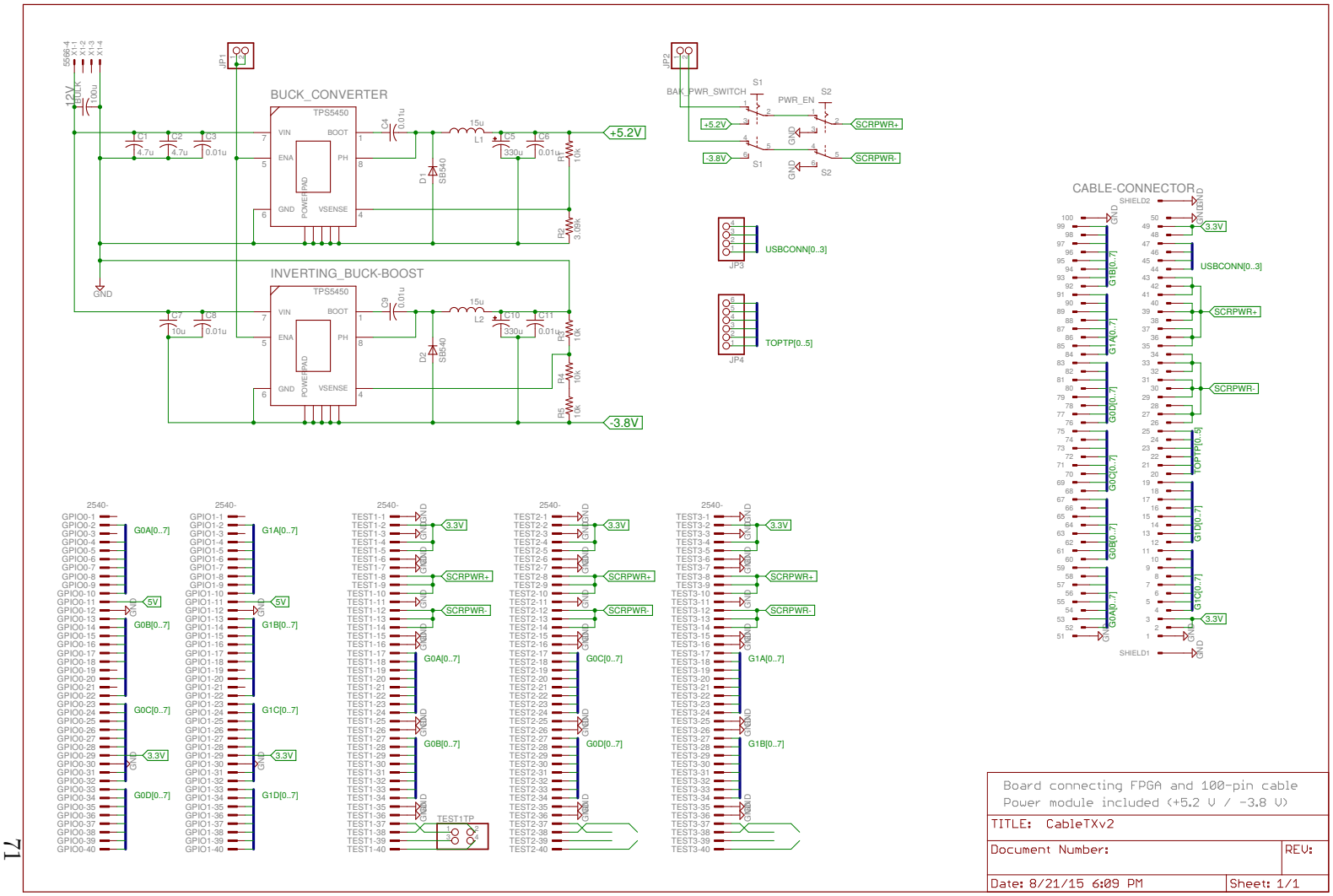
[36] Shumin Zhai, William Buxton, and Paul Milgram. The silk cursor: Investigating transparency for 3d target acquisition. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '94, pages 459–464, New York, NY, USA, 1994. ACM.

Appendix A

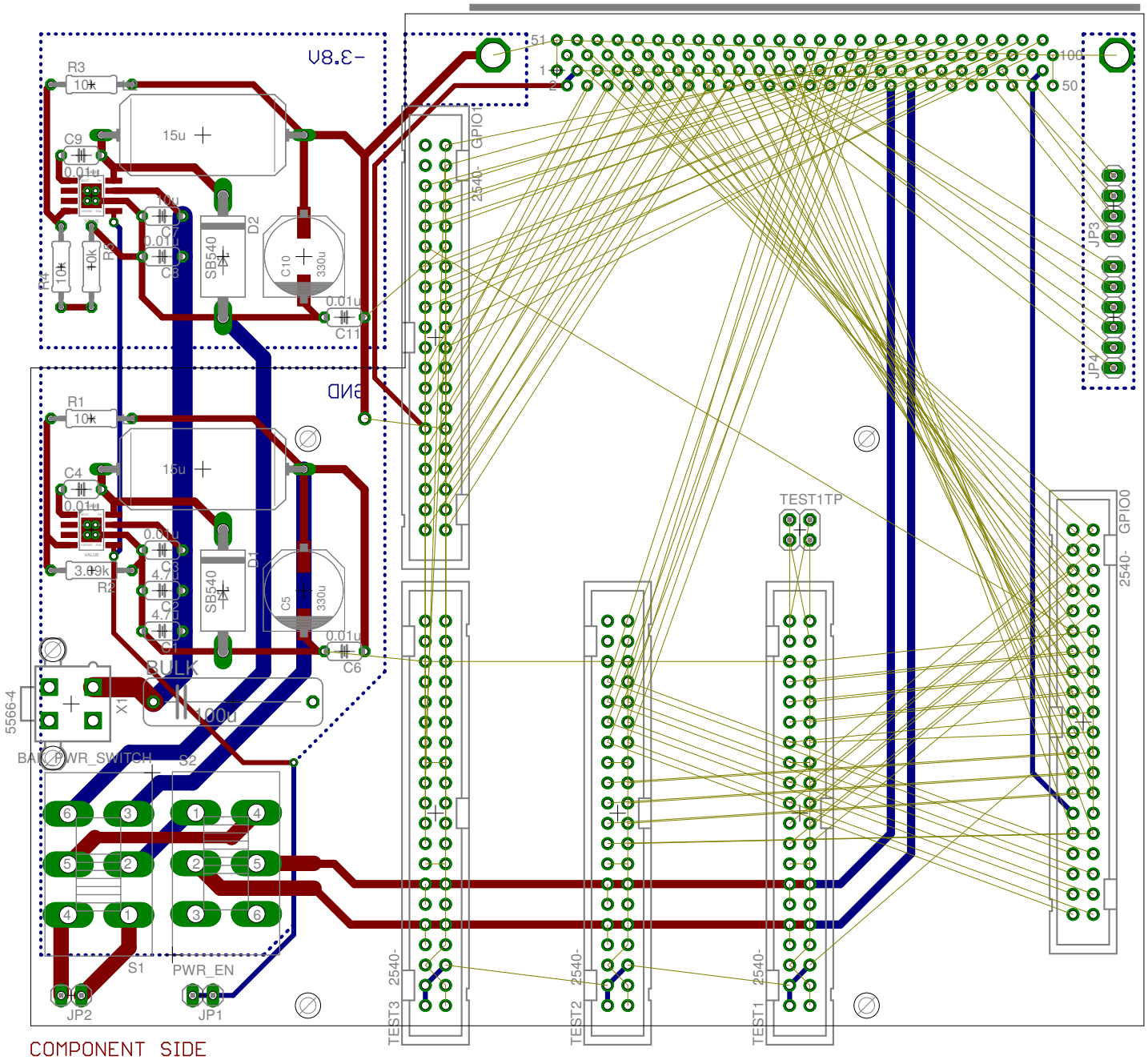
Board Schematics and Layout Designs

Following pages depict hardware design documents for the new pCubee display module.

1. CableTX board schematic design version A, featuring a 100-pin National Instrument cable for connection to display.
2. CableTX board layout design version A.
3. CableTX board schematic design version B. A compact design with two 40-pin ribbon cables for display connection. It allows easier manipulation of the display but as a trade-off, cable lengths are limited.
4. CableTX board layout design version B.
5. Final CableRX board schematic design. Breakout of ribbon cable signals to OLED interface boards. Including touch controller signals for all five panels.
6. Final OLED interface board schematic design.
7. Final OLED interface board layout design.

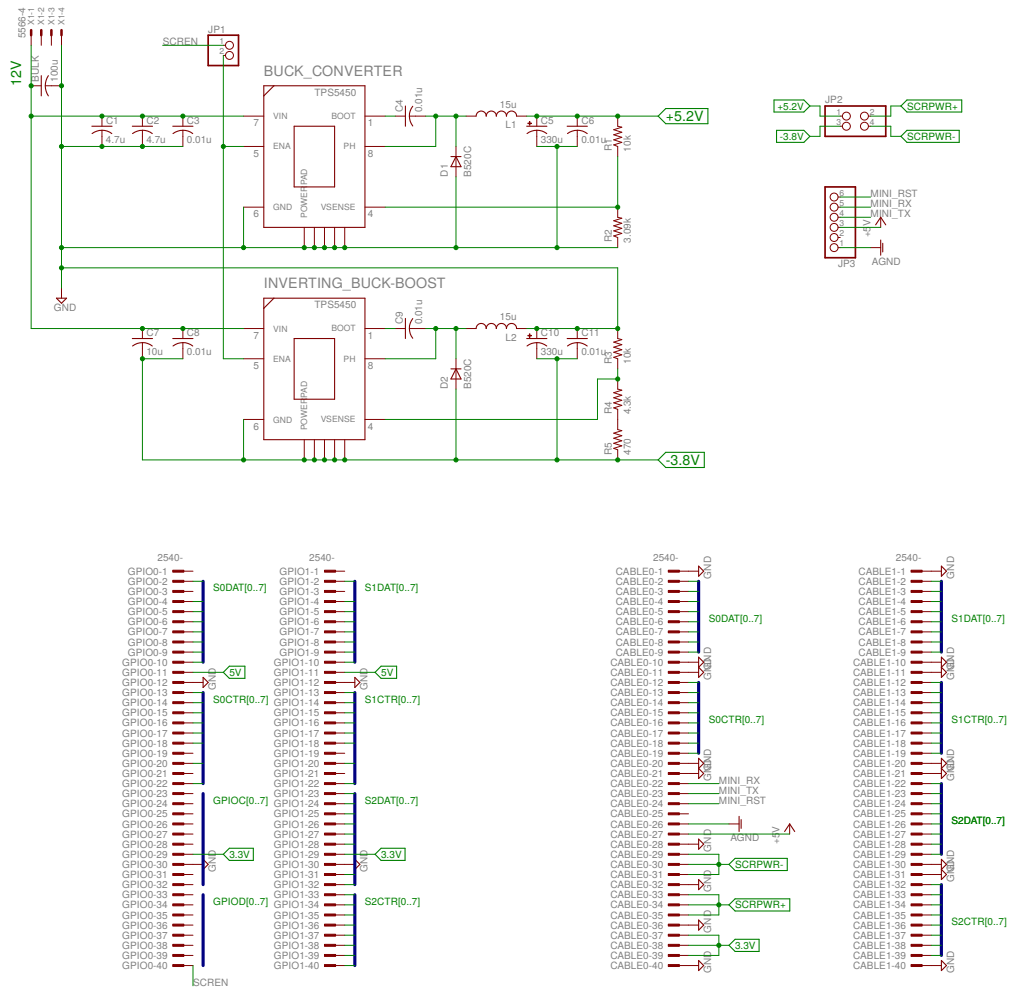


4/21/15 6:50 PM f=0.70 /home/yitang/git/eagle/CableTXv2/CableTXv2.sch (Sheet: 1/1)



COMPONENT SIDE

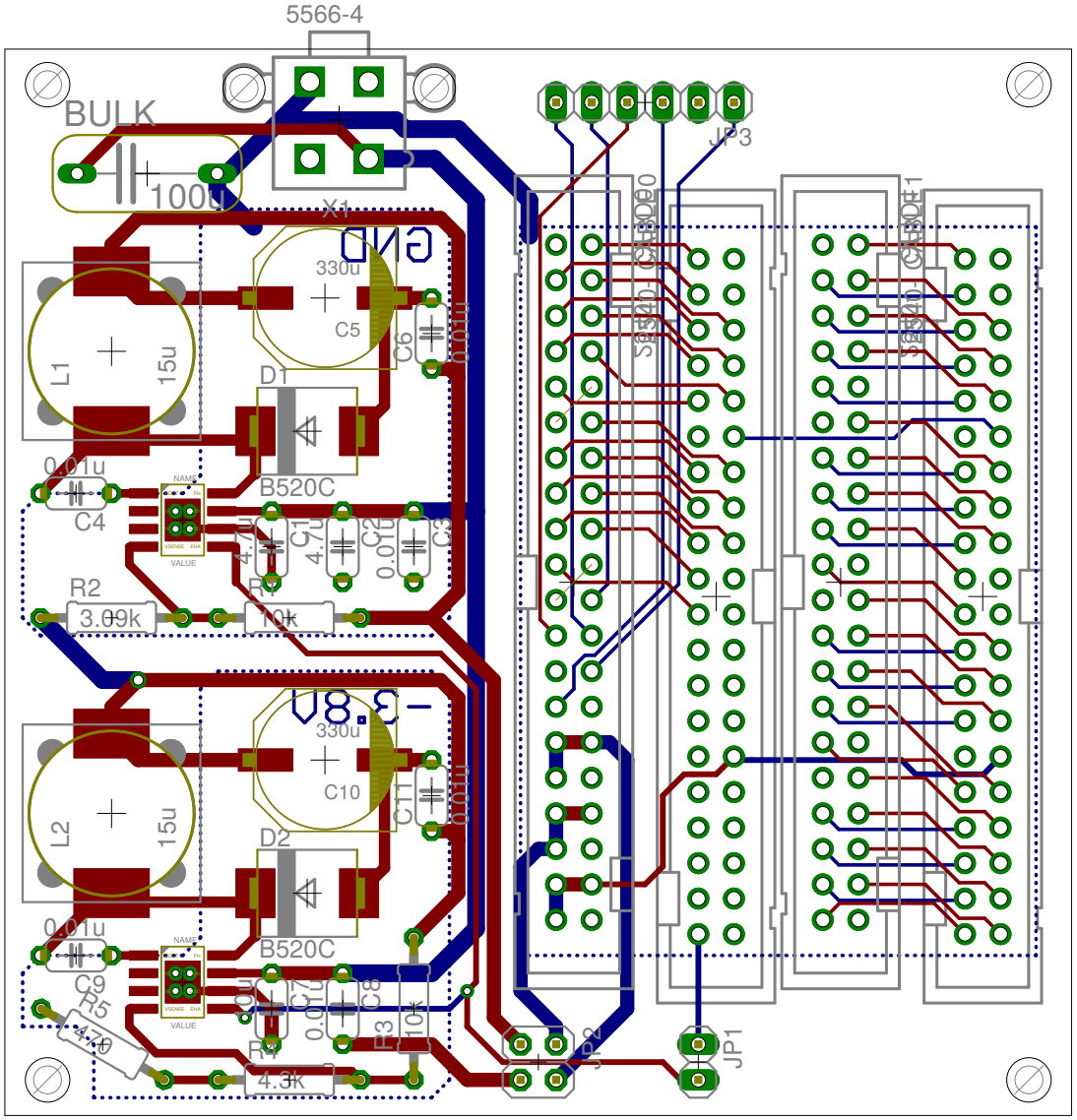
4/21/15 6:51 PM f=1.40 /home/yitang/git/eagle/CableTXv2/CableTXv2.brd



Board connecting FPGA and 2x 40-pin cable
Power module included (+5.2 V / -3.8 V)

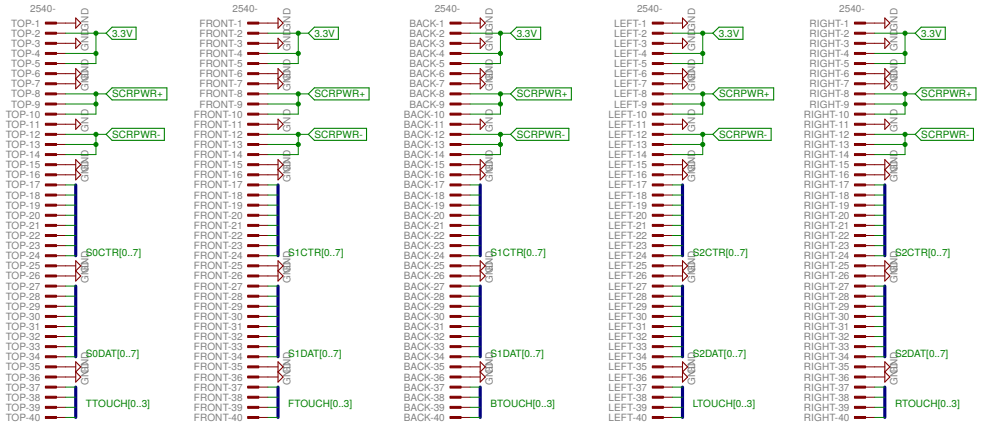
TITLE: CableTXv3r1	
Document Number:	REV:
Date: 4/21/15 6:43 PM	Sheet: 1/1

4/21/15 6:45 PM f=0.70 /home/yitang/git/eagle/CableTXv3/CableTXv3r1.sch (Sheet: 1/1)

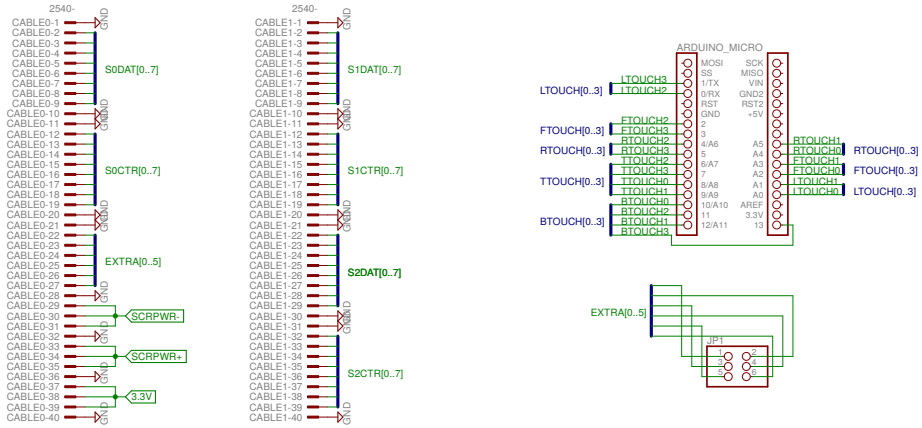


COMPONENT SIDE

4/21/15 6:46 PM f=2.00 /home/yitang/git/eagle/CableTXv3/CableTXv3r1.brd



CAUTION: CTRL and DATA buses on OLEDInterface boards are inverted (MSB first).

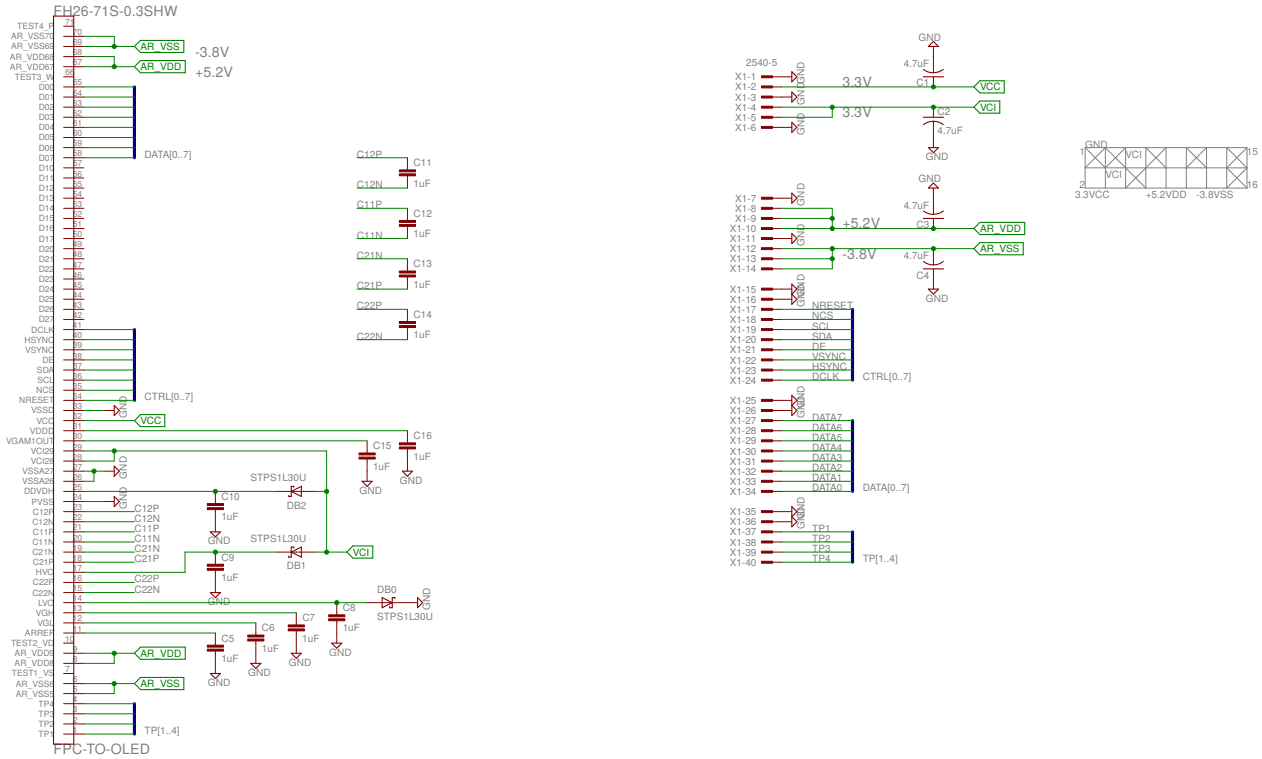


Board connecting 2x 40-pin cable & OLED Interface boards (with Arduino Micro)

TITLE: CableRXv4r2

Document Number: REV:

Date: 4/21/15 6:41 PM Sheet: 1/1



4.3-inch OLED Interface Board

71-pin FPC to 40-pin

TITLE: OLEDInterfacev8

Document Number:

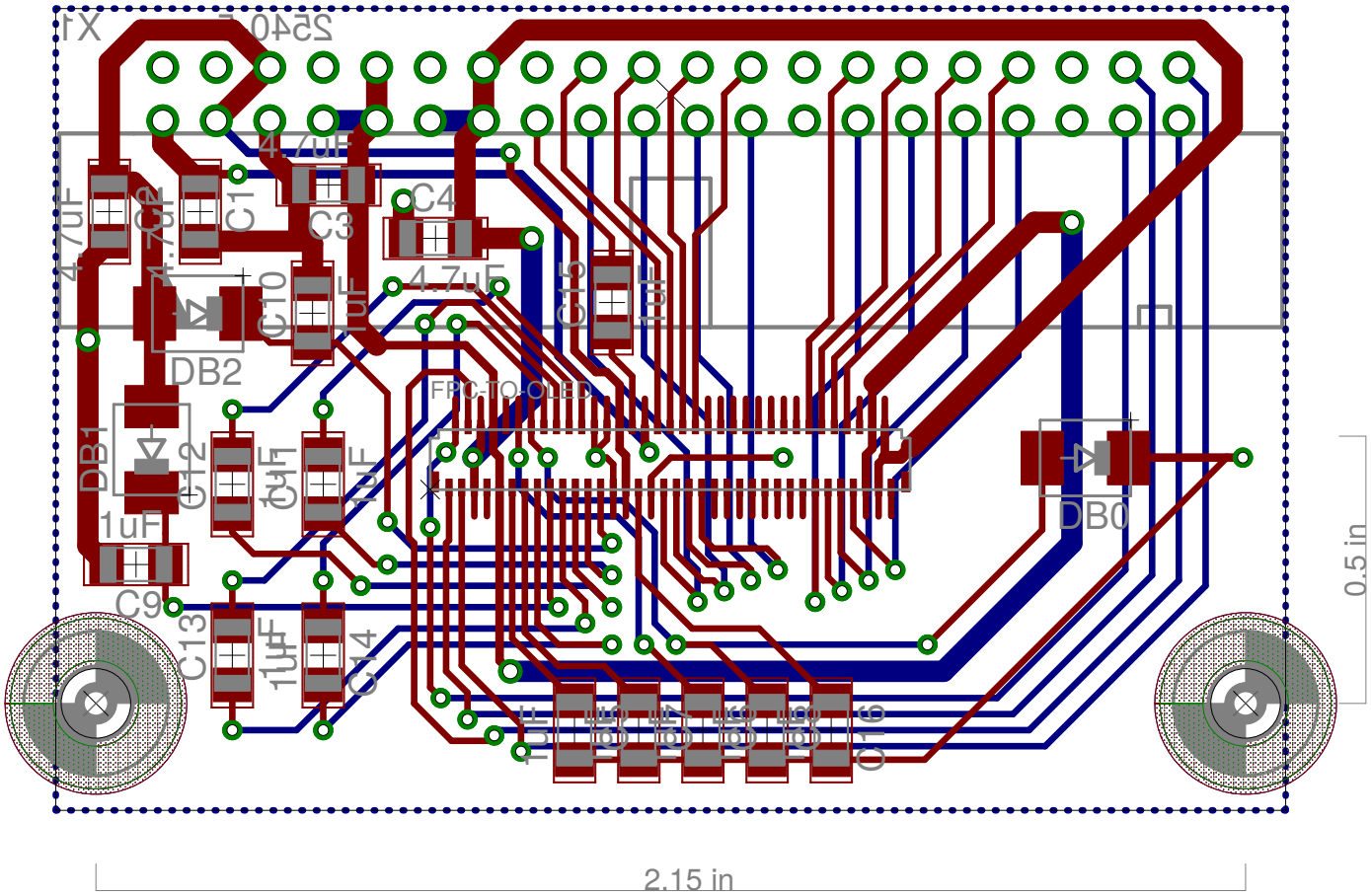
REV:

Date: 4/21/15 6:43 PM

Sheet: 1/1

4/21/15 6:47 PM f=0.70 /home/yitang/git/eagle/OLEDInterfacev8/OLEDInterfacev8.sch (Sheet: 1/1)

COMPONENT SIDE



4/21/15 6:48 PM f=3.00 /home/yitang/git/eagle/OLEDInterfacev8/OLEDInterfacev8.brd

Appendix B

User Study Questionnaire

User Experiment Questionnaire

Participant No. _____
Experiment Date _____
Evaluator _____

Section 1 - Basic information

Age:_____ Gender:_____

1.1 How would you rate your familiarity with 3D interfaces? - Choose one:

- ☐ No experience - never played with any 3D interaction device
- ☐ Little experience - played with wii/ps3move casually etc.
- ☐ Some experience - used Virtual 3D tools such as a 3D desktop
- ☐ Experienced - used 3D modeling software such as Maya, 3dsmax, SolidWorks, etc.
- ☐ Expert - used 3D input device like 3D trackballs, styluses, cyber gloves, etc.

1.2 Have you ever played with wire maze puzzle game before?

Section 2 - Evaluator observations

2.1 Bimanual on virtual maze? Time:
2.2 Bimanual on physical maze? Time:

Section 3 - Overall ratings

3.1 How would you rate the overall difficulty of this task?

	-3	-2	-1	0	1	2	3	
Very Challenging	☐	☐	☐	☐	☐	☐	☐	Very Easy

3.2 How would you rate the overall difficulty of the physical maze task?

	-3	-2	-1	0	1	2	3	
Very Challenging	☐	☐	☐	☐	☐	☐	☐	Very Easy

3.3 How would you rate your performance on this task?

	-3	-2	-1	0	1	2	3	
Very Poor	☐	☐	☐	☐	☐	☐	☐	Very Well

3.4 Please rate each of the following:

- Visualizing the wire maze in virtual environment

	-3	-2	-1	0	1	2	3	
Very Difficult	☐	☐	☐	☐	☐	☐	☐	Very Easy
Very Unintuitive	☐	☐	☐	☐	☐	☐	☐	Very Intuitive

- Manipulate virtual ring using the physical sensor

	-3	-2	-1	0	1	2	3	
Very Difficult	☐	☐	☐	☐	☐	☐	☐	Very Easy
Very Unintuitive	☐	☐	☐	☐	☐	☐	☐	Very Intuitive

- Moving the ring

	-3	-2	-1	0	1	2	3	
Very Difficult	☐	☐	☐	☐	☐	☐	☐	Very Easy
Very Unintuitive	☐	☐	☐	☐	☐	☐	☐	Very Intuitive

- Rotating the ring

	-3	-2	-1	0	1	2	3	
Very Difficult	☐	☐	☐	☐	☐	☐	☐	Very Easy
Very Unintuitive	☐	☐	☐	☐	☐	☐	☐	Very Intuitive

- Handling pCubee

	-3	-2	-1	0	1	2	3	
Very Difficult	☐	☐	☐	☐	☐	☐	☐	Very Easy
Very Unintuitive	☐	☐	☐	☐	☐	☐	☐	Very Intuitive
Very Bulky	☐	☐	☐	☐	☐	☐	☐	Very Light

3.5 What do you think contributes the most to the difficulties of the task?
(wire length, curvature, ring radius, etc.)

3.6 Do you find the following feature helpful?

- Collision highlighting

	-3	-2	-1	0	1	2	3	
Not Helpful at all	☐	☐	☐	☐	☐	☐	☐	Very Helpful

- Shadows

	-3	-2	-1	0	1	2	3	
Not Helpful at all	☐	☐	☐	☐	☐	☐	☐	Very Helpful

- Bi-manual manipulation

	-3	-2	-1	0	1	2	3	
Not Helpful at all	☐	☐	☐	☐	☐	☐	☐	Very Helpful

- Head-tracking (motion parallax)

	-3	-2	-1	0	1	2	3	
Not Helpful at all	☐	☐	☐	☐	☐	☐	☐	Very Helpful

3.7 What features do you think should be added to better help you complete the task?

3.8 What improvements would you expect for the system?

3.9 Other comments?