pCubee

Evaluation of a Tangible Outward-facing Geometric Display

by

Billy Shiu Fai Lam

B.ASc., The University of British Columbia, 2011

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

MASTER OF APPLIED SCIENCE

in

The Faculty of Graduate Studies

(Electrical and Computer Engineering)

THE UNIVERSITY OF BRITISH COLUMBIA

(Vancouver)

April, 2011

Abstract

This thesis describes the evaluation of pCubee, a handheld outward-facing geometric display that supports high-quality visualization and tangible interaction with 3D content. Through reviewing existing literatures on 3D display technologies, we identified and examined important areas that have yet to be fully understood for outward-facing geometric displays. We investigated the performance of a dynamic visual calibration technique to compensate for tracking errors, and we demonstrated four novel interaction schemes afforded by tangible outward-facing geometric displays, including static content visualization, dynamic interaction with reactive virtual objects, scene navigation through display movements, and bimanual interaction. Two experiments were conducted to evaluate the impact of display seams and pCubee's potential in spatial reasoning tasks respectively. Two stimuli, a path-tracing visualization task and a 3D cube comparison task that was similar to a mental rotation task, were utilized in the experiments. In the first experiment, we discovered a significant effect on user performance in path-tracing that was dependent on the seam thickness. As seam size increased beyond a thickness threshold, subjects relied less on multiple screens and spent longer time to trace paths. In the second experiment, we found that subjects had significant preference for using the pCubee display compared to a desktop display setup when solving our cube comparison problem. Both time and accuracy using pCubee were as good as using a much larger, more familiar desktop display. This proved the utility of outward-facing geometric displays for spatial reasoning tasks. Our analysis and evaluation identified promising potential but current limitations of pCubee. The outcomes from our analysis can help to facilitate development and more systematic evaluations of similar displays in the future.

Preface

The pCubee display has been a collaborative research effort at the Human Communication Technologies Laboratory by Professor Sidney Fels, Dr. Ian Stavness, Master's student YiChen Tang, undergraduate student Ryan Barr and me. The work reported in this thesis has resulted in three publications.

1. Billy Lam, Ian Stavness, Ryan Barr, and Sidney Fels. 2009. Interacting with a personal cubic 3D display. In *Proceedings of the 17th ACM international conference on Multimedia* (MM '09). ACM, New York, NY, USA, 959-960. Awarded Best Technical Demonstration.

2. Ian Stavness, Billy Lam, and Sidney Fels. 2010. pCubee: a perspective-corrected handheld cubic display. In *Proceedings of the 28th international conference on Human factors in computing systems* (CHI '10). ACM, New York, NY, USA, 1381-1390.

3. Billy Lam, Yichen Tang, Ian Stavness and Sidney Fels. A 3D Cubic Puzzle in pCubee. In Press. *Symposium on 3D User Interfaces 2011*, IEEE. Awarded 3DUI Contest second place.

Portions of publications 1 and 2 have been modified for Chapters 3 and 4 of this thesis. Publication 3 has been modified for Chapter 5 and Appendix C. Professor Fels and Dr. Stavness contributed ideas on the development and evaluation of pCubee and provided editing and revisions for the publications. Dr. Stavness participated in writing the initial draft for publication 1. Mr. Tang assisted in conducting the evaluation of using pCubee to solve a 3D cubic puzzle as described in publication 3. Mr. Barr was responsible for developing portions of the pCubee software concerning the integration of the rendering and physics simulation engines that is presented in publication 1.

I made the following significant contributions to the research and writing for the three publications.

- ideas on the design and development of the pCubee display reported in publication 2;
- assembly of the pCubee display hardware and implementation of the pCubee software, including calibration and prototyping the applications that are reported in publications 1 and 2;
- design, implementation and execution of all of the experiments that are reported in publications 2 and 3;
- writing initial drafts for publications 2 and 3 and subsequent editing of all of the publications.

Professor Fels and Dr. Stavness actively supervised the research project and contributed ideas on the development and evaluation of the pCubee display. All content reported in this thesis was part of my research during my participation in the project, including the implementation of the calibration algorithm and prototyping of the interaction techniques, the design and execution of all pilot studies and formal user studies, and the analysis of all data that resulted from the experiments.

The pCubee research project was funded by the Networks of Centres of Excellence of Canada through GRAND, the Graphics, Animation and New Media NCE, and by the National Sciences and Engineering Research Council of Canada. All experiments reported in this thesis have been approved by the UBC Behavioural Research Ethics Board (Certificate Number H08-03005).

Table of Contents

Appendices

List of Tables

List of Figures

List of Code Snippets

Acknowledgements

Thank God for all the opportunities, successes and failures given to me throughout the course of working on this thesis. It has been my privilege to take part in the pCubee project and be guided by some of the brightest minds I have come to know in the field.

To my supervisor Sid: thank you for your supervision and guidance. Your insights have greatly broadened my limits and perspectives.

To Ian: thank you for backing me up and keeping me out of trouble so many times. You have been invaluable to everything written in here.

To my HCT and MAGIC friends: you guys made the lab time that much more slack and enjoyable. Best wishes to your ongoing research and future endeavors.

Thank you mom and dad for your love and patience. Love you all. (Bro, thank you for not being here, I needed your room and the quiet time)

A big thank to all my friends for journeying with me, supporting me, praying for and with me. I would not be here without you guys, seriously.

Chapter 1

Introduction

Understanding the design and performance of three-dimensional (3D) display technologies has become increasingly important and relevant to our interaction with digital 3D information. Due to rapid advances in computer graphics and capture systems in the recent years, 3D data sources are growing more abundant and accessible, along with an increasing number of display technologies that allow us to visualize and interact with them.

In addition to commercially-available stereoscopic displays, different viable 3D display technologies have been proposed, including volumetric displays and head-tracked, perspective-corrected displays. While there has been significant focus on achieving implementations of the highest possible technical standards, less emphasis has been placed on formal evaluation of these displays. This prevents us from being able to fully understand and compare them with respect to their designs and how well they support 3D perception and interaction in specific tasks.

This is especially true for a class of multi-screen 3D displays, which we refer to as outward-facing geometric displays. They extend the concept of traditional Fish Tank Virtual Reality (FTVR) displays by arranging multiple flat panel screens outwardly to form a tiled geometric shape (hence the name geometric), such as a cube. By tracking the user's head position and correcting the perspective of each screen correspondingly, outward-facing geometric displays can render 3D scenes to make virtual content appear as if real objects were contained within the boundary enclosed by the display. Unlike more sophisticated hardware requirements for other 3D display techniques, outward-facing geometric setups take advantage of existing high-quality flat panel screens, making the hardware mostly self-contained and usually compact enough to be directly manipulated while held in a user's

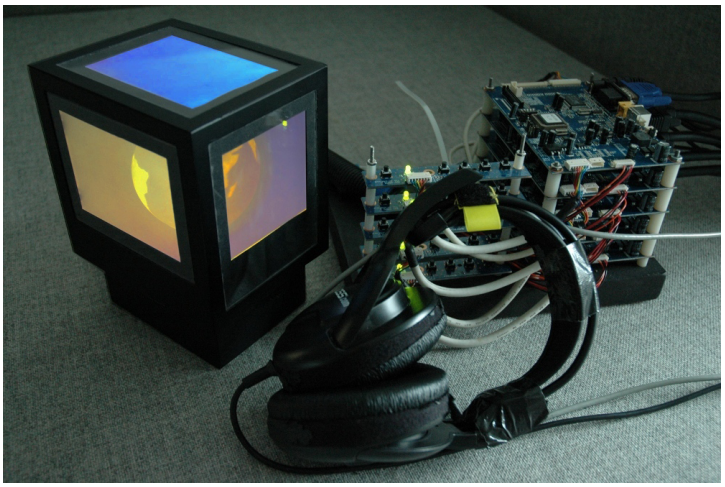


Figure 1.1: The pCubee cubic display

hands. The capability to couple manipulation and visualization together in a tangible system provides an interaction experience that is similar to how we interact with physical objects in our hands. As 3D visualization become more prevalent and more interactive technologies such as touch-screen and gesture-based controls are made available, we see great potential in outward-facing geometric displays to become effective tools in a variety of application domains involving 3D content, such as scientific data visualization, computer-aided design (CAD), biomedical applications, and other virtual reality tasks such as artistic virtual painting and animation.

The introduction of outward-facing geometric displays has brought forth a number of research problems yet to be thoroughly explored. One of the main design challenges with these geometric displays is the presence of physical seams, or bezels, at the joining edges between display panels, which can disrupt the visualization experience when a user's gaze moves across multiple screens. Another problem is that head-tracked displays require accurate tracking of the user's viewpoint to generate perspective-corrected images for the user's viewpoint. Distorted or slow tracking leads to visual mismatches that are readily apparent when viewing multiple screens.

In terms of interaction techniques and 3D task performance, the tangible nature of outward-facing geometric displays offers promising potential to support more intuitively perception and interaction with 3D objects. Exploring all of these issues is a crucial step towards fully understanding the capabilities and supporting the future adoption of outward-facing geometric displays in real-world 3D applications.

To address the lack of empirical work on these issues, we conduct an evaluation on a tangible outward-facing geometric display, pCubee, which arranges five small flat panel screens into a cubic shape without the bottom (Figure 1.1). Through an exploratory analysis of pCubee, we explore the strengths and weaknesses of the display, including its system design and tracking calibration, and we demonstrate different interaction techniques that it affords. We perform two controlled user studies in which we investigate (i) the effect of physical seam occlusions in pCubee on user performance in a path-tracing visualization task, and (ii) how users perform a 3D cube comparison task, which was similar to mental rotation, using pCubee compared to a conventional desktop setup. In this thesis, we report the outcomes and findings from our analysis and evaluation of pCubee.

1.1 Contributions

The research presented in this thesis provides the following contributions with respect to the evaluation of outward-facing geometric displays.

Evaluation on the Impact of Seam Size

Using a 3D path-tracing experiment, we discovered that user performance and interaction behaviors were dependent on the level of visual discontinuity caused by physical seam occlusions of the pCubee display. Our results revealed that path-tracing is an unsuitable task for current outward-facing geometric displays because of both seam occlusions and the task's apparent inability to take advantage of wide range of perspectives.

Evaluation on Spatial Reasoning using pCubee

Using a 3D cube comparison experiment based upon existing mental rotation literature, we found that users significantly preferred using pCubee compared to a desktop-and-mouse setup to perform the task. While physical seam occlusions remained an issue in the current pCubee hardware, our results confirmed the usability of tangible outward-facing geometric displays in this type of spatial reasoning and perception task.

Novel 3D Interaction Schemes

We proposed and showcased four interaction schemes in pCubee, including static visualization, dynamic interaction, large scene navigation and bimanual stylus-based interaction. These techniques are unique to tangible outward-facing geometric displays and show the capabilities of this new display technology to support novel methods for interacting with 3D content.

1.2 Thesis Structure

The remainder of this thesis is structured as follow: Chapter 2 surveys previous literature on existing 3D display technologies closely related to pCubee; Chapter 3 provides an analysis of the pCubee system with respect to its design, tracking calibration and interaction techniques; Chapter 4 reports a first experiment on evaluating the effect of physical seam occlusions in pCubee on path-tracing tasks; Chapter 5 reports a second experiment on evaluating the spatial reasoning capability using pCubee for 3D shape comparison tasks; Chapter 6 concludes with future directions our research suggests.

Chapter 2

Related Work

Research and development of 3D display technologies date back to as early as the 1960s (I.E. Sutherland developed what was known to be the first head-mounted immersive display in 1968 [54]); however it was not until the last two decades that more variety of concepts could be realized. Due to rapid improvements in computer graphics and display technologies, recent 3D display development boasts both significantly higher fidelity and more sophisticated design. These emerging high quality implementations have allowed researchers to better explore and evaluate various aspects of interaction techniques and task performance.

In this chapter, we review previous literature on 3D display technologies that are similar to pCubee. The first part of the chapter surveys existing 3D displays to provide an analysis of different implementation techniques and the strengths and weaknesses of each. Because formal evaluations of outward-facing geometric displays are few, our goal for the review is to extract important issues for 3D displays identified by other researchers in the past, which we can reference for our evaluation of pCubee. In the second part of the chapter, we summarize empirical findings associated with 3D displays in the context of a number of topics important for outward-facing geometric displays.

2.1 Classification of 3D Displays

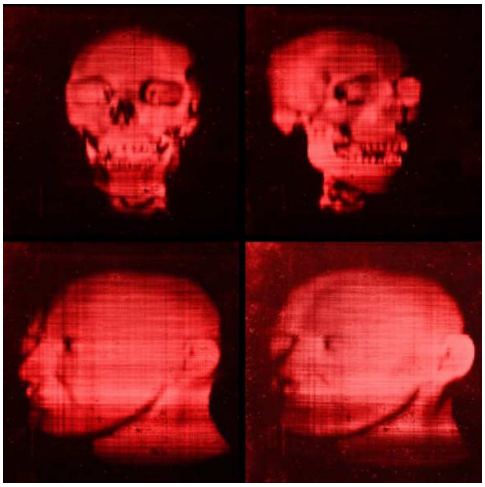
3D display technologies convey 3D information by mimicking one or a combination of depth cues used by the human visual perception system. These include monocular cues such as motion parallax, occlusion, perspective, lighting and shadows, and binocular cues such as stereopsis and convergence (see

chapters 2 and 3 of the report by Wickens et al. [60] for an in-depth survey of these visual cues and their interactions). While conventional 2D displays support a number of depth cues such as perspective, occlusion, lighting and shadows, 3D displays utilize more salient depth cues, including stereopsis and motion parallax, to deliver compelling 3D effects to their users. Existing 3D display implementations that are similar to pCubee can be classified into two categories: (i) volumetric displays, in which 3D information is rendered to its corresponding physical location; and (ii) geometric displays, in which visualization is dependent on the arrangement of multiple screens and a correct perspective projection onto each.

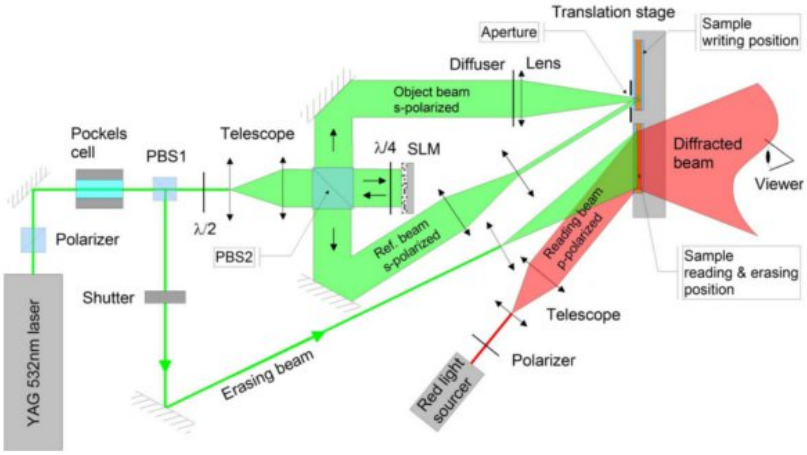
2.1.1 Volumetric Displays

Volumetric displays, also known as "true 3D" displays, illuminate 3D points in their physical spatial locations, which allows them to render perceptually rich 3D content by satisfying all visual depth cues just as real-world objects do. 3D effects generated by volumetric displays are achieved without requiring users to wear special glasses or other hardware, a particularly desirable property for 3D displays which is referred to as autostereoscopic. The fact that these displays can be viewed by a multitude of users with independent viewing perspectives makes them ideal tools for collaborative 3D tasks. However, volumetric displays are difficult to realize at the present time because of a number of technical challenges that have to be overcome. Volumetric displays are in general limited in resolution, brightness, and compactness due to constraints with the optical elements used or the large number of simultaneous views they have to render. Opacity can also be a problem for translucent volumetric systems that diffuse lights in all directions.

Existing implementations of volumetric displays include holographic [56], static-volume [17], swept-volume [23, 53] and multi-view [33, 34] techniques. From the perspective of output, these displays all provide true 3D image visualization, but their underlying designs and strategies vary widely.



(a) Visualization



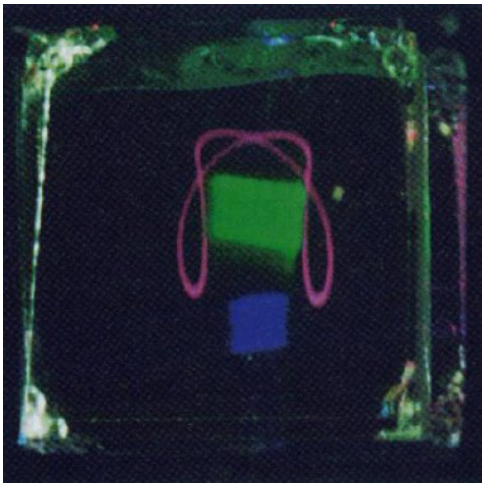
(b) Schematic

Figure 2.1: Visualization and schematic for a holographic display. Figures adapted from [56].

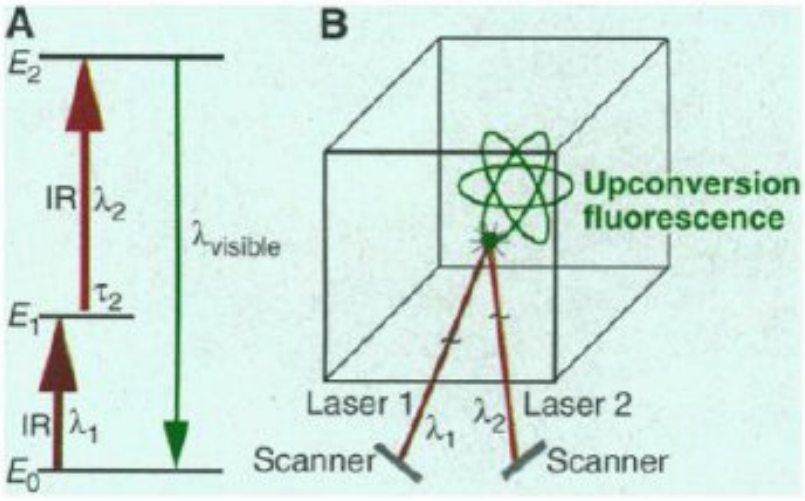
Holographic Displays. Holographic displays record the light scattering patterns of physical objects and reconstruct them in a display medium. The viewer can view these 3D recordings within a certain viewing range as if they have been imprinted inside the medium. Traditionally, holographic images are recorded in permanent materials such as silver halide films, dichromated gelatin or photopolymers. Displaying dynamic 3D content with holographic displays remains technologically challenging: interactive holographic displays are still far from achieving real-time speed. The most recent implementation by Blanche et al. [56] (Figure 2.1) requires a minimum 1 minute recording time on a 4in. x 4in. hologram.

Static-volume Displays. Another class of volumetric displays, referred to as static-volume or cross-beam displays, share similarities with holographic displays in the sense that they too render volumetric data within solid-state display mediums. Instead of recording light scattering patterns, however, static-volume displays utilize active optical elements that can be excited to emit light. Using dual intersecting infrared laser beams at locations representing the voxel data, these systems can display 3D objects by drawing out their shapes rapidly within the display medium. Figure 2.2 illustrates a cross-beam, static-volume display by Downing et al. [17]

Swept-volume Displays. These displays rely on the persistence of vision to allow users to stitch together 3D images from a sequence of 2D slices. Different architectures, including rotational and translational systems, have been implemented to display these 2D slices using high-speed projectors or displays. A rotational system is described by Grossman and Balakrishnan [23] (Figure 2.3). It utilizes a rotating, omnidirectional diffuser screen to project lights from 2D slices in all directions in 3D space. On the other hand, a translational system, such as the DepthCube [53], uses multi-planar optical elements that can rapidly shut off and let through light to project stacks of 2D slices at their corresponding planes to achieve depth.

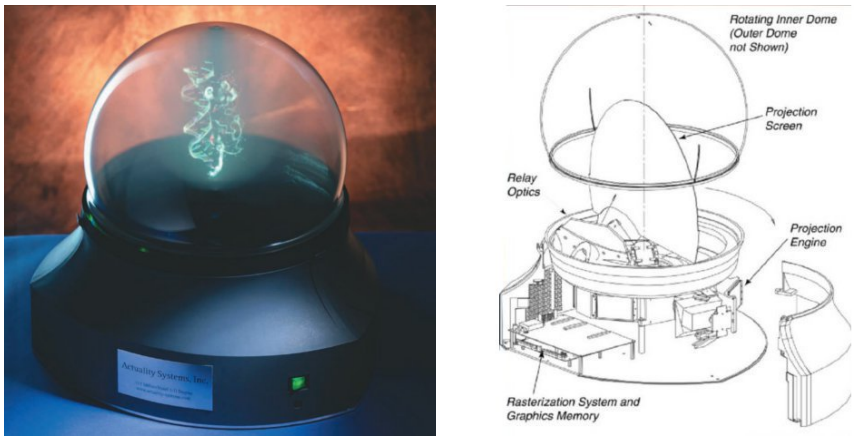


(a) Visualization



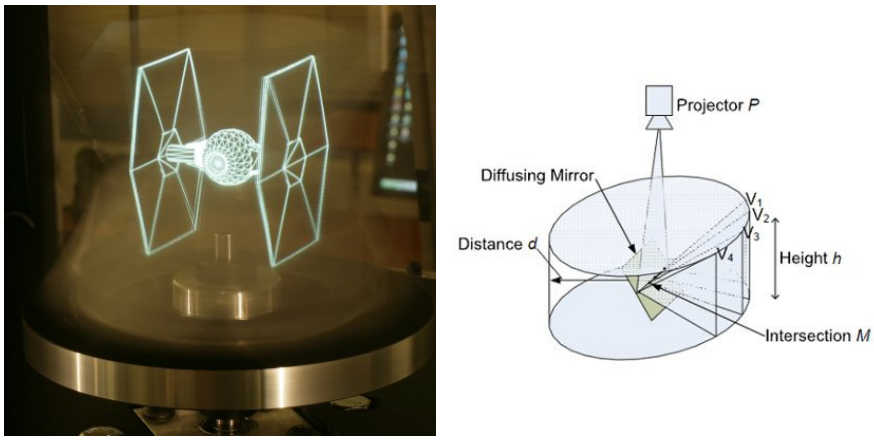
(b) Schematic

Figure 2.2: Visualization and schematic for a static-volume display. Figures adapted from [17].



(a) Visualization (b) Schematic

Figure 2.3: Visualization and schematic for a swept-volume display. Figures adapted from [19].



(a) Visualization (b) Schematic

Figure 2.4: Visualization and schematic for a multi-view display. Figures adapted from [34].

Multi-view Displays. Multi-view displays are a special class of volumetric displays. Similar to swept-volume displays, they reconstruct 3D images by displaying a large number of 2D projections through high-speed projection such as in the light field system by Jones et al. [34] (Figure 2.4, or "special-purpose" LED arrays such as in the *RayModeler* by Ito et al[33]. However, instead of rendering and diffusing the data points at their physical spatial locations, multi-view displays project the 2D images through a surface directed at each image's corresponding viewing direction. These configurations provide autosterescopic horizontal-parallax views in 360 degrees, but in general they neglect 3D information in the vertical direction and also do not take into account the user's viewing distance (the light field system by Jones et al. required head-tracking for per-user vertical parallax effects).

2.1.2 Geometric Displays

Alternative approaches to volumetric displays are geometric displays, which rely more heavily on motion parallax to deliver 3D effects to their users. By rendering images on multiple 2D screens with perspectives corrected to the user's point of view, these displays can establish the illusion of 3D on a 2D surface, a technique we refer to as head-coupled perspective rendering.

Geometric displays are extended from the original concept of head-tracked desktop virtual reality displays, also Fish Tank Virtual Reality (FTVR) displays as described by Arthur et al[2]. Traditionally, FTVR displays often consist of a single screen coupled with a head tracker and LCD shutter glasses to generate stereo images at the users perspective. These include small-scale desktop systems as described by Deering [13] and McKenna [42], and also large-scale systems that support multiple regions of user-specific perspectives for better collaboration as described recently by Maksakov et al. [41] While simple and fairly effective, these systems offers a limited viewing angle and are hindered by occlusion mismatches when virtual objects rendered in front of the screen are cut off by the screen boundary. By arranging multiple screens together, geometric displays effec-

tively overcome the viewing angle limitations, allowing objects to be in front and behind the screens with proper occlusion cues depending on the screen configurations.

An advantage of geometric displays over volumetric displays is in the ongoing advances in projection and display technologies to create bright, high-resolution images in increasingly lighter form factors. The arrangement of multiple screens into different geometric shapes can establish a compelling illusion similar to volumetric displays but show more detailed imagery. However, geometric displays are typically only valid for one perspective, which significantly hampers collaboration tasks, as opposed to volumetric displays that by definition provide multiple simultaneous correct views. While special eyewear such as shutter or polarized lenses can be used for stereo viewing or potentially multiplexing perspectives between two or more users, the additional hardware presents its own issues such as reduced brightness, ghosting and discomfort. We distinguish geometric displays into two categories, inward-facing and outward-facing setups.

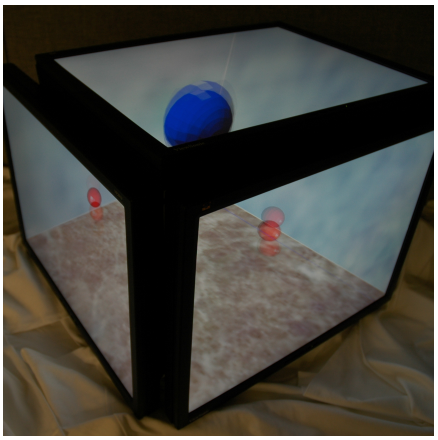
Inward-facing Displays. Inward-facing geometric displays utilize different combinations of projector and projection screen arrangements to generate correct 3D perspectives to the users on otherwise flat surfaces. One of the earliest displays in this category is the CAVE system [11] (Figure 2.5a), which uses the walls of a room as inward-facing back projection screens. The system allows the tracked user to walk around and receive correct perspectives of a surrounding scene, in addition to providing stereoscopic views through shutter glasses. CAVE systems provide strong 3D effects and an immersive experience because users are located inside the virtual reality projected from all sides; however, these systems are both large and expensive to set up.

A number of other inward-facing geometric displays have also been proposed. Cubby [15] (Figure 2.5b) uses three small rear-projection screens and shows compelling monocular head-coupled 3D effect through the large motion-parallax afforded by the multi-screen setup. The "virtual showcases" as described by Bimber et al. [5] (Figure 2.5c) demonstrate cubic or cylindri-

(a) CAVE

(b) Cubby

(c) Virtual Showcase



(d) Cubee

Figure 2.5: Visualizations in different geometric display designs.

cal arrangements of rear-projection systems and transflective surfaces that can produce a similar effect to outward-facing displays. As with the CAVE system, one of the biggest advantages of projection-based geometric displays is that they can be made seamless, however, it is challenging to make them tangible given their large size compared to their outward-facing counterparts.

Outward-facing Displays. Outward-facing geometric displays also extend the FTVR concept and arrange flat panel screens to face outwardly. The effect is exactly opposite of that generated by CAVE: instead of having a virtual environment that surrounds their users or objects that are in front of the screens, outward-facing displays enclose the virtual objects within a physical volume to be viewed from outside. Inami [30] showcased the first outward-facing geometric display prototype, which he described as an "object-oriented" display, called MEDIA CUBE. The system renders correct perspectives onto four display panels representing four sides of a cubic shape. This portrays a compelling 3D effect of objects contained inside.

A number of other outward-facing geometric displays similar to pCubee have been implemented in the past, all of which draw many parallels with MEDIA CUBE. Cubee [52] (Figure 2.5d) is a large-scale version of pCubee, which was assembled from five desktop monitor-sized screens compared to five-inch screens used in the current system. Cubee was supported with ropes from an overhead truss to allow for direct user manipulation. A more sophisticated design, such as a five-screen cubic prototype gCubik [39], utilizes special lens arrays to precisely divide integral images containing multiple perspectives to a wide range of viewing angles. The lenses achieve an autostereoscopic effect similar to the multi-view 3D displays as described previously in Section 2.1.1, although they significantly degrade the resolution at each perspective, as shown in Figure 2.6. As well, real-time interaction currently remains a problem due to the large number of simultaneous views the system needs to render.

2.2 Evaluation of 3D Displays

Various explorations and evaluations of 3D displays have been reported in the past. Many of these focused on topics that are common to most 3D displays, such as different approaches for interacting with 3D content and performing 3D tasks. There are also less-explored areas that we consider to be especially significant for pCubee and other geometric displays, such as

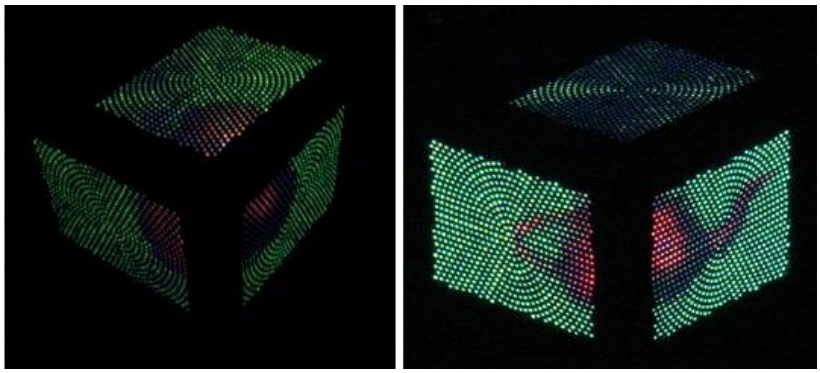


Figure 2.6: gCubik autostereoscopic cubic display showing limited resolutions. Figure adapted from [39].

the requirements of tracking and calibration and the discontinuity of visualization created by the display seams or bezels. Given the lack of formal evaluations on outward-facing geometric displays, we survey existing empirical findings on 3D display technologies in the context of these various issues in order to support a systematic approach to our analysis and evaluation on pCubee.

2.2.1 Tracking Calibration

The accurate tracking of the user's viewpoint relative to the display is essential for head-tracked systems to render perspective-corrected images to the user. Especially for multi-screen geometric displays, the 3D effect is compromised when there are mismatches between the virtual scene and the physical arrangement of the screen panels due to tracking errors. Therefore, it is important to understand and characterize the tracking and calibration techniques employed to achieve the performance desired for these particular systems.

A number of alternatives for position tracking have been utilized for head-tracked displays in the literature, including mechanically linked systems such as the Shooting Star Technology ADL-1 tracker [2], ultrasonic systems as described by Deering [13], and electromagnetic systems such as

the Polhemus Fastrak used in a number of geometric displays [11, 30, 52]. Our review focuses specifically on issues surrounding electromagnetic tracking systems which are most often used, as is the case for pCubee. We categorize these issues into spatial and temporal calibration problems.

Spatial Calibration. In general, electromagnetic tracking has many favorable characteristics, such as its acceptable resolution without line-of-sight problems, but they are notorious for their sensitivity to magnetic field distortions resulting from metal objects and electronic equipment in the environment. Further, their spatial accuracy falls off rapidly as the distance between the transmitter and the sensors increases. These factors add up to significant errors in position and orientation data (up to more than 50 cm in location errors and 15 degrees in orientation errors as observed in [27]). These tracking errors can be corrected through a two-step calibration process: (i) data acquisition to characterize the distortions in the work space, and (ii) error compensation using different numerical methods during system usage to correct for distortions.

Numerical methods used in the error compensation stage can be classified into two categories: global methods that consider all data points to derive the most accurate global mapping function, and local methods that only use neighboring data points to obtain a localized compensation. Global methods described in previous work include high-order polynomial fit [7, 35], Hardy's Multi-quadric method [61] and also neural network-based method [36]. While global techniques provide continuous mapping over the entire tracked space and thus better error compensation than local methods, they are more challenging to implement. On the other hand, local methods, such as tri-linear [38] and tetrahedral interpolations [18], are simpler to adapt but suffer from discontinuities (of zeroth order) when the interpolated function crosses from one data grid to another. The deficiency could be improved upon by using a higher number of surrounding data points to provide a more continuous gradient throughout the tracked space [7].

Different calibration approaches as mentioned above have been shown to reduce tracking errors to the order of less than 2 cm for position and 3

degrees for orientation for electromagnetic trackers. In most scenarios, an assumption these techniques make is that the display configuration remains constant, and therefore only one cycle of data measurements is necessary to characterize the distortions. This assumption can be a problem for pCubee and other tangible head-tracked displays, which we describe in further details in the following chapter.

Temporal Calibration. Besides spatial distortions, temporal artifacts such as latency in the tracking, software and hardware systems can be problematic for head-coupled perspective displays. As shown previously by Arthur et al. [2], lag can disrupt motion-based 3D effects and task performance (lag over 210 msec was found to be worse than static viewing for a simple path-tracing task). Deering [13] also suggested that critical values for perceived lag are similar to motion fusion and should be no more than 50-100 msec. Previously, head-tracked systems mitigate the effect of latency by using predictive tracking techniques, such as simple linear interpolation (Deering [13]) or higher order interpolators such as Kalman's filter (Friedmann et al. [**?**]).

It is important to note that predictive tracking introduces additional undesirable artifacts into the tracking data, including overshooting during high acceleration and amplifying sensor noise. Overshooting can be notorious especially for tangible outward-facing geometric displays like pCubee because of the high degree of movements available to the displays compared to the user's relatively limited head motion. Recent head-tracked system development are less concerned with temporal calibration because of improvements in tracking and graphics technologies that reduce lag to well below threshold values identified above. For our evaluation of pCubee, we focus only on spatial calibration to correct for distortions in the tracking system as lag was rarely noticed to be an issue in the system.

2.2.2 Interaction Techniques

Interacting with 3D content requires mechanisms different from conventional desktop-and-mouse setups. Actions such as pointing and selection remain challenging problems that need to be investigated for different display configurations. Various interaction schemes and mappings have been proposed for existing prototypes and for high fidelity mock-up displays. They have been shown to offer a more engaging and intuitive means of interaction with 3D content as compared to traditional 2D display setups. Here we summarize these proposed interaction schemes with respect to volumetric, inward-facing and outward-facing geometric displays.

Volumetric Display Interaction. For volumetric displays, interaction requires indirect manipulation techniques that are from outside the visualization space because users cannot directly reach into the bounded virtual content. Balakrishnan et al. [3] explored a variety of possible interaction schemes using Wizard-of-Oz prototypes. Implementations that extend from these ideas include ray-tracing-based selection using a 3D ray cursor (Grossman et al. [21]) and gesture-based rotation using over-the-surface interaction (Ito et al. [33]). In particular, Grossman and Balakrishnan identified a 3D ray cursor metaphor to be a better design choice than a point cursor in 3D space for selection, improving movement time, error rates and input device footprints [21]. With touchscreen technology becoming common-place, multi-touch interaction can also be incorporated with volumetric displays. Selection and manipulation with multi-touch gestures, such as zooming, offer interesting possibilities that have been explored for volumetric displays [23].

Inward-facing Geometric Display Interaction. For interaction with inward-facing geometric displays, one challenge is the occlusion problem, similar to how stereo images can be blocked by screen borders. As the virtual objects are "floating" in front of the displays as opposed to being behind them, reaching into the display space can lead to visual mismatches when

users attempt to move behind the virtual objects. A virtual tip extension on a physical stylus reaching into the visualization space has been demonstrated with Cubby [16], which partially alleviates the occlusion issues. A one-to-one mapping of a 3D input device, such as a wand, is also often used to allow more direct manipulation of virtual objects (Vickers et al. [57], Demiralp et al. [14]).

Outward-facing Geometric Display Interaction. For outward-facing geometric setups, a tangible interaction scheme can be employed to allow users to directly manipulate the display hardware, which is not possible with static systems. While tangible 3D input devices have been explored in previous evaluations, as was done by Hinckley et al. [29] and Ware and Rose [59], outward-facing geometric displays couple manipulation and visualization in a unified workspace. This enables a novel interaction scheme of having simulated physics for virtual objects based on the tracked movement of the display, as was shown in Cubee [52]. Touch-screen rotation proposed for volumetric displays have also been demonstrated with gCubik [39], showing potential interaction development in over-the-surface, gesture-based control for outward-facing setups. Further, other interaction techniques available to volumetric displays can be applied to outward-facing setups for a single user seeing the correct viewpoint.

2.2.3 Visual Discontinuity

Physical artifacts inherent in different 3D display designs cause disruptions to the experience that we refer to as visual discontinuity. For tasks in which the continuity of the rendering is important, such as scientific data visualization, these artifacts are particularly undesirable. Similar to understanding temporal artifacts such as frame rates and lags, identifying performance limits due to physical spatial artifacts have important implications to the future development of 3D display technologies.

While visual discontinuity is a less-explored area, it is present in many of the 3D displays discussed earlier in this chapter. For outward-facing

geometric displays such as pCubee, a large portion of the virtual scene can be occluded by the presence of physical seams at the joining edges between screens. Seams between multiple monitors have been investigated in the past for 2D information such as texts and lines (Mackinlay and Heer [40]), though their effects have not been examined in multi-screen geometric 3D displays. Understanding the impact of seams can lead to insights regarding content or tasks that are suitable for these displays. For other displays such as swept-volume or multi-view systems, the effect of different levels of sacrificed resolutions or intervals between 2D slice sequences can shed light on the capability and usability of those systems.

2.2.4 3D Task Performance

One of the most important areas in 3D display evaluations is understanding whether and how visualization and interaction schemes can better support users performing 3D tasks. Past evaluations of task performances focused on comparisons between different 3D displays and depth cues they afford in a number of task domains. We can these categorize into visualization and reasoning tasks.

3D Visualization. The extra depth information provided by 3D displays allows users to better explore 3D data for tasks such as scientific visualizations where the spatial relationship between the data are important. Graphs, path-tracing and visual search are such tasks that have been investigated with 3D displays in the past. Arthur et al. [2] compared a one-screen FTVR setup to a monitor-based desktop system and reported that the FTVR setup significantly improved performance in a path-tracing task. They found that benefits gained by head coupling were more than those gained from stereoscopic effect alone. Ware et al. [58] reaffirmed these trends in a graph visualization study and also noted that any structured motion cues in general, including head-coupled rendering, hand-guided motion or automatic rotation, led to similar performance improvements. In more recent comparison study, Demiralp et al. [14] compared the performance of the CAVE and

a one-screen FTVR display using visual search tasks and concluded that users performed better with and preferred FTVR displays. However, Prabhat et al. [47] reported contrasting results in their comparison of the same virtual environments in more complex scientific visualization tasks. Further, comparisons between CAVE-like setups and standard desktop workstations in statistical visualization [1] and oil well path planning [24] have revealed users' preference towards the immersive system. The divergent results from these studies suggest that user performance and preference on different 3D display implementations can be very task-dependent.

3D Reasoning. An alternative to pure scientific visualization tasks are 3D reasoning tasks that involve the perception and understanding of 3D space and shapes. These include tasks such as collision judgments and 3D rotation which have also been evaluated with 3D displays in the past. Grossman and Balakrishnan [22] conducted a comparison between traditional stereo FTVR system and a swept-volume display and found that the latter allowed better perception in both depth and collision judgment tasks. Prabhat et al. [48] evaluated a desktop setup and a CAVE system for learning hypercube rotations and showed that users were more accurate and learned more about the geometries using the CAVE. Other researchers have used or proposed mental rotation as a task stimulus for 3D display or input system evaluation in the past (Booth et al. [6], Hinckley et al. [29]), which is a natural fit in the 3D reasoning task space. We see mental rotation as an interesting area to explore because 3D rotation has been shown to be a difficult task [44], and few empirical evaluations have been done that utilized 3D displays in this area.

2.3 Summary

In this chapter, we reviewed a number of existing 3D display configurations that are similar to pCubee, including volumetric and geometric displays. Given that there has been no reported empirical work on outward-facing geometric displays, we presented a review of evaluations with other existing

implementations to provide insights on the current research landscape of 3D display technologies.

Through our review, it becomes apparent that no one existing display implementation is best for all 3D interaction techniques and tasks given all the design trade-offs needed to be taken into consideration. While volumetric displays such as holographic and swept-volume displays offer true 3D rendering and are suitable for multi-user collaboration tasks, they have limitations in terms of resolution, brightness, compactness and opaqueness. On the other hand, geometric displays can offer higher quality visualization but are limited to a single user's perspective. These tradeoffs are supported by the diverse results and preferences we identified from previous studies. Further, given the specificity of some tasks involved in past evaluations, such as oil well path planning and complex scientific visualization, existing results are not easily generalizable to allow fair comparisons between different display technologies. These findings further strengthen our desire for a systematic approach towards evaluations of various 3D displays.

We categorized previous empirical results into a number of issues important for outward-facing geometric displays, including tracking calibration, interaction techniques, visual discontinuity and task performance. In our investigation in subsequent chapters, we examine these four issues for the pCubee system, either through exploratory analysis or formal, controlled user studies.

Chapter 3

Analysis on Design, Calibration and Interaction Techniques

pCubee is designed to be a compact cubic 3D display. The system integrates input manipulation and output visualization to support tangible interaction such as tilting and shaking the display to make virtual objects react. By coupling an additional input device, such as a 3D stylus, pCubee enables bimanual manipulation of both the display and the stylus. These factors make pCubee a compelling 3D display that affords novel interactions and tasks that are closer to how users interact with physical objects.

In this chapter, we provide an exploratory analysis of the pCubee display to understand its system design and issues regarding tracking calibration and interaction techniques. For tracking, we examine a dynamic visual calibration technique for pCubee, through which we provide an analysis of its performance, strengths and limitations. Regarding interaction techniques, we showcase a number of novel interaction schemes that can take advantage of the unique, tangible nature of pCubee and other outward-facing geometric displays.

3.1 Display Hardware

The hardware design of pCubee is diagrammed in Figure 3.1. The display consists of five 5-inch VGA resolution (640x480 pixels) LCD panels [32] that are mounted onto a wooden box-shaped frame. The panels are arranged and

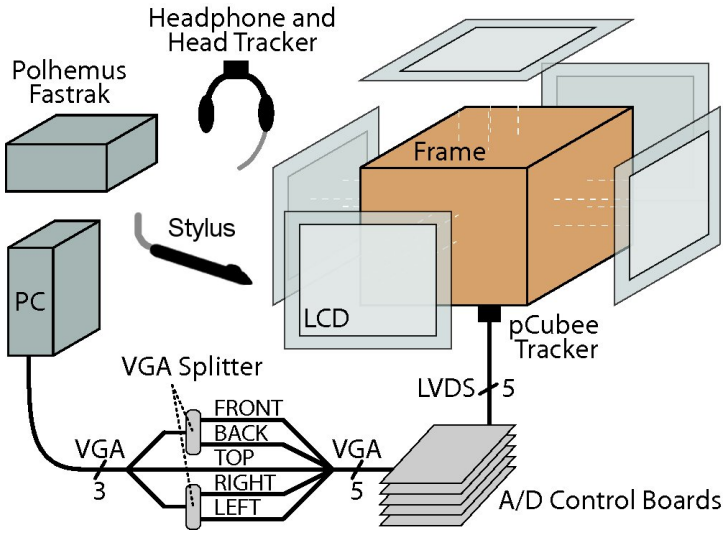


Figure 3.1: Hardware components of pCubee.

aligned to create even seams on all sides of the display; two side panels are oriented vertically to fit them evenly with the top screen. The bottom portions of the vertical panels are covered up to create even bottom seams on all sides, leaving an empty area that can also be used to grasp the display without blocking the screens. The bottom side of the box is left open for ventilation and cables with a small 120x96x36 mm base to make the entire display box easier to grasp, and the total weight of the frame, base, and screens is measured to be 1.3kg (2.87lbs). Excluding the base, the display box measures 145x120x145mm.

Figure 3.2 illustrates the screen configuration we used in pCubee. Small physical seams are difficult to realize with LCD panels because the border is dependent both on the thickness of the panel and the width of the bezel. The virtual arrangements shown in the figure, with the screen edges actually connected, measures 133.90x110.15x133.90 mm, compared to the prototype's actual dimensions reported above. Brightness and color consistencies across the viewing range of the screen panels are also important factors to take into consideration because users can view from all around

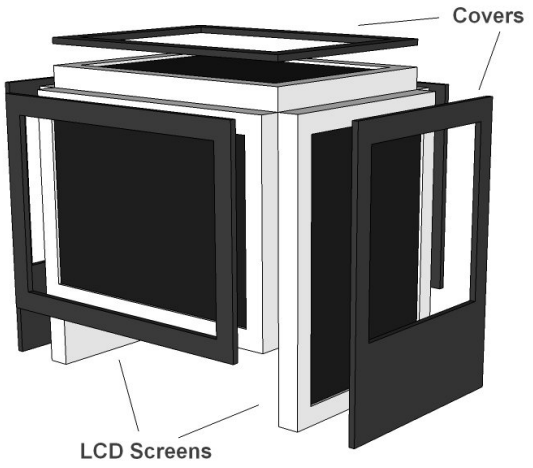


Figure 3.2: Screen arrangement of pCubee.

an outward-facing geometric display. While screen brightness is consistent with the current LCD panels, there are noticeable color distortions at oblique viewing angles, producing either a blue or yellow tint depending on the side of the screen users are viewing from.

Three graphics signal outputs are used to drive pCubee because users can only see three sides of the box at any given time. A host computer, (Intel Quad Core 3.0GHz processor, Windows XP) with two dual-output Nvidia GeForce 9800 GX2 graphics cards, generates three VGA signals. The distribution of separate rendering contexts to graphics card outputs is done using multi-monitor support in the Nvidia graphics driver. The VGA signals for opposite sided screens (front and back, left and right) are routed through signal splitters to get five video signals total. Each VGA signal is converted to low-voltage differential signaling (LVDS) video with an analog-to-digital (A \D) control board [31] and connected to a timing control board on the backside of the LCD panel as shown in Figure 3.3. The five A/D control boards are housed in a pedestal and connected to pCubee with a bundle of five 1-meter LVDS cables. A stylus is incorporated to allow for precise manipulation of content inside the display.

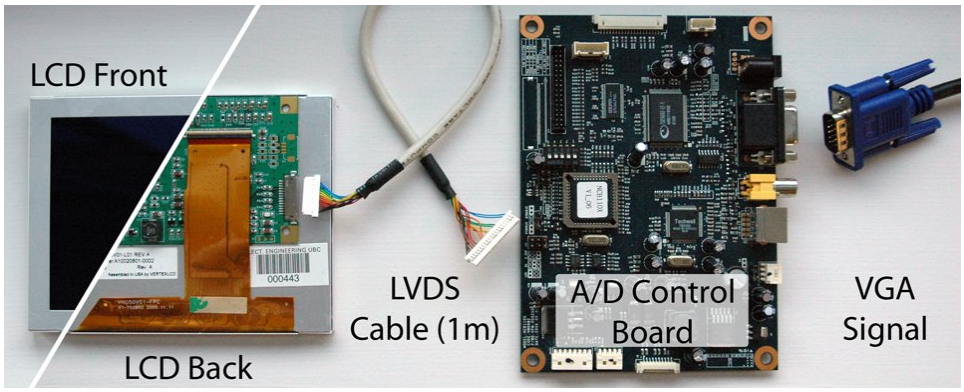


Figure 3.3: pCubee electronics: showing the LCD panel, the controller board and the LVDS and VGA cables.

3.2 Software Components

The pCubee software is built upon existing open source engines, including OpenSceneGraph (OSG) [43] for rendering, Nvidia PhysX [10] for physics simulation, and the FMOD toolkit [20] for sound simulation.

3.2.1 Rendering Software

The OSG engine is used to render high quality graphics in pCubee, including shadows, multi-texturing and shading, for compelling depth cues and realism. To generate perspective-corrected images on each screen of pCubee, a standard off-axis projection scheme as described by Deering [13] is implemented. This is done in OSG by creating three *View* objects that correspond to the three visible screens on pCubee. The camera for each *View* is located at the users real-world eye position, oriented perpendicular to its corresponding virtual screen, and given a view frustum that passes through the screen corners as shown in Figure 3.4. The near-clip plane is set to be coincident with the screen plane in order to prevent rendering of virtual objects that are outside of the pCubee boundary, which would cause occlusion issues at the screen edges (i.e. objects that are in front of the display seams would not be seen). Figure 3.5 illustrates how the skewed images

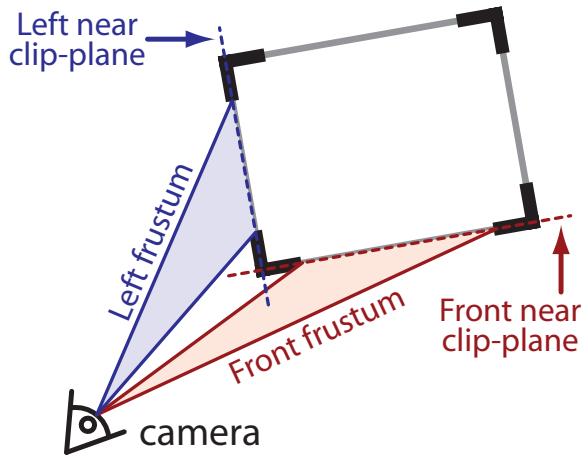


Figure 3.4: View frustum calculation for each pCubee screen.

generated with off-axis projections fuse when viewed obliquely on the sides of the cubic display. The multiple rendering contexts are contained within a single *CompositeViewer* object and the camera parameters in each *View* are updated before a single call is made to *CompositeViewer* to update all the *Views* simultaneously.

A virtual pCubee frame is added to the 3D scene to enhance occlusion cues and the illusion of looking into a box. At oblique viewing angles, the real seams along the cube edges occlude virtual objects within the cube, and the virtual objects occlude the virtual frame that is rendered behind. pCubee shows only monocular views due to current synchronization limitations with the LCD panels, which is most noticeable at perspectives where the virtual frame only aligns with one eye but not the other. Stereoscopic rendering could be added to pCubee with stereo-capable flat panel displays and synchronized shutter glasses to alternate between left-eye and right-eye views for more accurate occlusion cues. However, rendering objects to appear outside of the display boundary, especially across multiple panels, remains a problem and should be avoided.

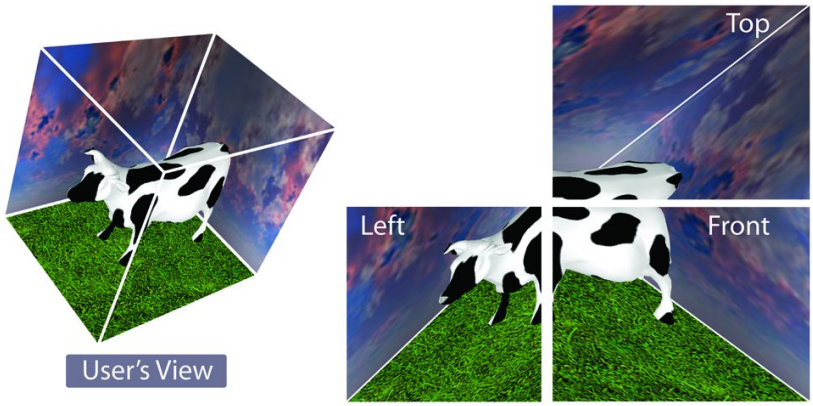


Figure 3.5: Images generated with off-axis projections. Note how each image looks skewed when viewed directly but produces the proper 3D effect when arranged geometrically.

3.2.2 Physics Software

A physics simulation engine is integrated with the OSG renderer to create different ways for users to interact with 3D content in pCubee. In the current pCubee software, Nvidia PhysX engine is used for real-time simulation of rigid body, deformable body, and particle system dynamics. Each virtual object in pCubee is represented both in the rendering scene as an OSG *Geode* object and in the physics simulation scene as a PhysX *Actor*. For rigid body models, the representations are often the same polygonal mesh; however for more detailed 3D objects, a high-resolution polygonal mesh could be used for the OSG *Geode* while its convex hull is used as the PhysX *Actor* to achieve faster simulation. For soft-body models, a coarse tetrahedral mesh could be used as the physics *Actor*, which is linked to a higher resolution polygonal mesh for rendering. Objects in the scene can be either static or dynamic. Static objects appear "attached" to the display because their positions are updated based on the display tracking sensor before each simulation step. The virtual pCubee frame, virtual transparent walls surrounding the frame and ground plane are static objects and move with the physical display. Dynamic objects appear to move freely within the box, i.e. they fall downward

under gravity relative to the real world, because their positions are updated by the physics engine after each simulation step. Collisions are computed between dynamic objects and between dynamic and static objects, making dynamic objects appear to bounce off the virtual inner walls as if pCubee was a glass box with real objects inside.

The FMOD toolkit is used to generate collision sound effects and ambient sounds that blend with the virtual scenes in pCubee. Currently, sound effects are pre-recorded and played at a volume corresponding to the magnitudes of the collision events. More realistic collision sounds could be synthesized directly from the collision objects.

3.2.3 Integration and Simulation

The pCubee software uses an object-oriented design. The *CubeeModel* class, which is the base class of all objects rendered inside the display, integrates the functionalities of the rendering, physics and sound engines to facilitate agile scene development. Classes extending from *CubeeModel* are used to generate models of different properties, including convex meshes, triangle meshes as well as softbody models. Both the OSG *Geode* and PhysX *Actor* representations of virtual objects are managed within inherited or derived classes of *CubeeModel*. Functions to set various model properties, including collision sounds, materials and textures, are also implemented. Appendix A documents the application programming interface (API) of the *CubeeModel* class and Appendix B illustrates the code required to create a simple virtual scene containing a dynamic soccer ball.

The pCubee system can achieve a 60Hz update rate for dynamic scenes with a small number of rigid bodies (e.g. 50 rigid body cow models each with a 5800-triangle *Geode* for rendering and a 125-triangle convex hull as its PhysX *Actor*). For more complex physics simulation, such as soft bodies and particle systems, the system achieves a 40Hz update rate for modest sized scenes appropriate for the scale of the display (e.g. two soft body cow models with 1700 tetrahedra each). The simulation loop for pCubee is a six-step process as outlined below:

Display Dimension	145x120x145 mm
Weight	1.3 kg (2.87 lbs)
Total Resolution	5x640x480 pixels (5xVGA)
Number of Simultaneous Views	1
Display Color	24-bit Full Color
Max. Update Rate	40 Hz with stylus, 60Hz without

Table 3.1: Specifications of the pCubee system

1. Obtain latest display, head and stylus (if used) positions;
2. Update the positions of static objects in both their OSG *Geode* and PhysX *Actor* representations;
3. Perform a single time step simulation to update the position of dynamic objects in the physics simulation scene;
4. Update dynamic object positions in OSG scene graph based on physics simulation results;
5. Update OSG *View* frustum parameters based on relative position of the user's viewpoint to the display.
6. Render scene and play collision sound effects (if any).

3.3 System Specifications

We summarize the specifications of the pCubee system in Table 3.1 using similar metrics as used by Ito et al. [33]

3.4 Tracking Calibration

To render a perspective-corrected scene, the pCubee system requires the position of the users eye relative to the display. In addition, it requires the position and orientation of te display in space to allow the physics engine to

determine velocity and acceleration information to simulate a virtual scene that reacts to display movements.

pCubee relies on a wired electromagnetic tracking system (Polhemus Fastrak [46]) to achieve low latency tracking. With two sensors attached, the tracking update rate is 60Hz with a reported latency of 2-3msec. To estimate the user's eye position in space, a head-tracking sensor (referred to henceforth as head sensor) is embedded in the top of a pair of headphones, making the wired sensor less intrusive as users listen to sound effects and music while using pCubee. A tracking sensor is also embedded on the base of the box (referred to henceforth as display sensor) to track the movement of pCubee in 6 degrees-of-freedom. A pre-computed offset and rotation for each LCD screen relative to the display sensor is used for calculating perspective-corrected view frustums. pCubee also incorporates a tracked 3D stylus (referred to henceforth as stylus sensor) to allow users to directly interact with virtual objects inside the display. However, the tracking update rate is slowed to 40Hz with the additional stylus sensor.

3.4.1 Dynamic Visual Calibration

As discussed in Chapter 2, electromagnetic systems have distortion issues that require calibration to improve the tracking accuracy. Although in outward-facing displays such as pCubee, the physical seams provide added occlusion cues, the effect of having a physical boundary around the virtual scene is compromised when the perspective is incorrect due to tracking errors, resulting in a rendered virtual frame that does not align with the physical frame.

We implemented and tested a dynamic visual calibration technique based on the line-of-sight method described by Czernuszenko et al. [12] to explore its effectiveness in calibrating the pCubee system. The line-of-sight method measures tracking errors as the amount of displacement between virtual and physical objects placed in front of the display that should appear superimposed when viewed from a particular perspective. We see the technique as a natural fit for pCubee because of the presence of physical seams that

can be used "for free" to visually align with the virtual frame to perform perspective adjustments. Thus, it is unnecessary to superimpose additional physical objects over the display surface during the calibration process. The advantages of this technique are two-fold: i) users intuitively know they need to input additional data points when there are visual mismatches between the physical and virtual frames, and ii) since the physical and virtual frames are always present, the system allows for quick re-calibration during usage.

The calibration procedures begin with the initialization of offsets for both the display sensor relative to the center of pCubee and the head sensor relative to the user's eye. The display sensor's offset can be physically measured and remains constant with the pCubee's design, while the head sensor's offset is user-dependent and needs to be visually obtained at the start of each system usage. This is achieved by asking the user to align the physical and virtual frames from any single perspective. The obtained head sensor's offset is added to the final interpolated correction vector to achieve a corrected perspective at any given point in space.

After the offsets are initialized, the user can interactively manipulate pCubee and generate correction vectors at locations where they see mismatches between the physical and virtual frames. The user adjusts the virtual perspective using keyboard commands until the virtual frame in the scene is aligned with the physical display, creating a new correction vector at that specific location. For each location $\mathbf{P}$, the correction vector v($\mathbf{P}$) is obtained by the following equation (adapted from [12]):

$$v(\mathbf{P}) = \mathbf{V} - \mathbf{V}_o \tag{3.1}$$

where $\mathbf{V}_o$ is the head sensor offset initialized for each independent user, and $\mathbf{V}$ is the user-adjusted offset to provide the correct perspective at location $\mathbf{P}$. Correction vectors are stored and used to create a lookup table (LUT), a rectilinear 3D grid of 200x200x200 units (1 unit is equivalent to 1 cm in real world space) that characterizes the effects of these vectors (f($\mathbf{Q}$)) at each grid point $\mathbf{Q}$ according to the equation (adapted from [12]):

$$f(\mathbf{Q}) = v(\mathbf{P}_i) \tag{3.2}$$

if there exists i such that $dist(\mathbf{P}_i, \mathbf{Q}) = 0$; or:

$$f(\mathbf{Q}) = \sum_{i=1}^{n} \frac{w_i}{\sum_{j=1}^{n} w_j} v(\mathbf{P}_i) \tag{3.3}$$

if $dist(\mathbf{P}_i, \mathbf{Q}) \neq 0$ for all possible i: $n \geq i \geq 1$. n is the total number of correction vectors used, and w_j is a weight based on distance:

$$w_j = \frac{1}{dist^2(\mathbf{P}_j, \mathbf{Q})} \tag{3.4}$$

We chose an exponential factor of two for the distance ($dist$) because we tested it to be adequate for the pCubee system setup; a higher or lower factor can be used to increase or decrease the area of effect by each correction vector. Once the LUT is established, we perform linear interpolation with the closest eight data points surrounding the tracked location of the user's eye. The perspective of the virtual scene can then be rendered based on the corrected location, which is the summation of the initial head sensor offset and the interpolated correction vector. Every time a new correction vector is introduced, the LUT is updated in real-time until the user obtains visually correct viewpoints all around the display.

3.4.2 Calibration Results

To understand the performance of the dynamic visual calibration technique, we asked two volunteer subjects to test the calibration process on pCubee. While our goal was not to precisely characterize the calibration outcomes with such a small number of subjects, we expected it to be a quick validation on whether error reductions using the technique would be comparable to what was reported in the past when applied to pCubee.

Following a procedure similar to that described in [12], subjects generated correction vectors at pre-determined locations, and we measured the magnitudes of these vectors. These included locations about 30 cm away ex-

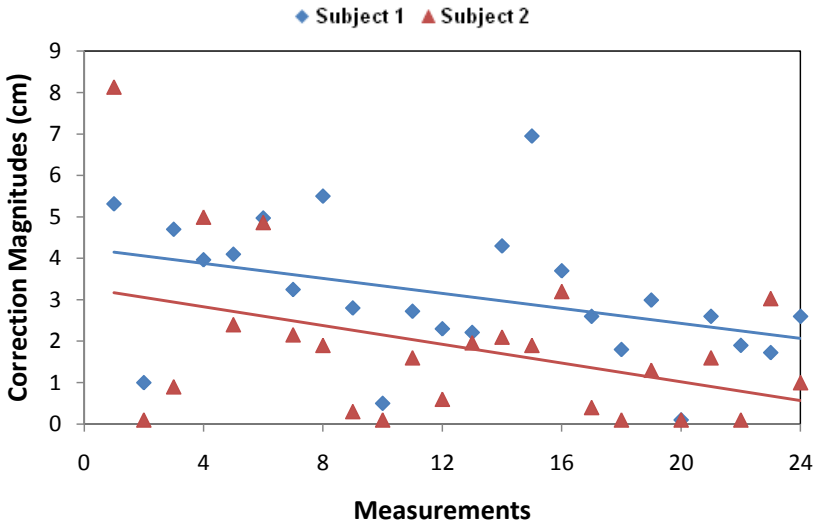
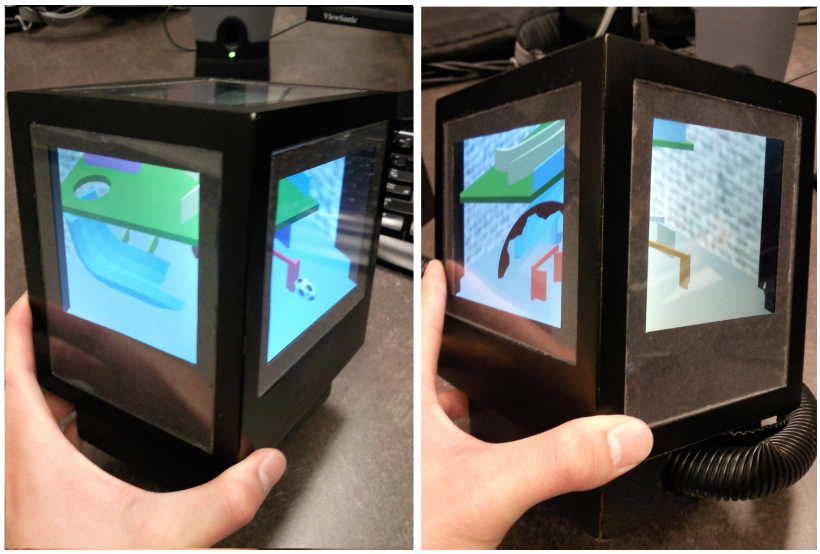
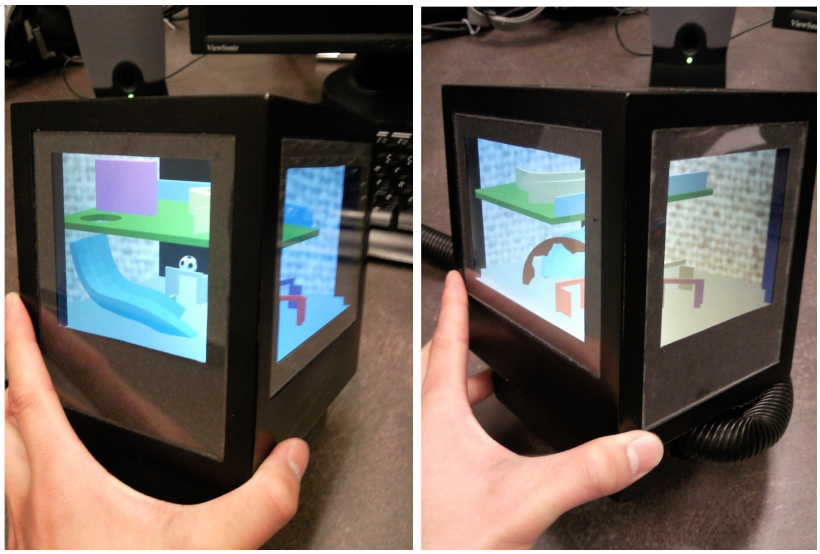


Figure 3.6: Magnitudes of the 24 correction vectors each subject generated using the dynamic visual calibration technique. Note the decreasing trend lines indicating a decrease in residual errors.

tending from the centers of the four side screens and the four side edges at 3 different heights (i.e. above the display, parallel to the display and below the display); this resulted in eight locations surrounding pCubee at each height for a total of 24 correction vectors. To ensure the adjusted perspectives were consistent at each point, we asked subjects to perform the calibration using monocular-only viewing by covering one eye. Based on previously reported results, we had expected magnitudes to gradually decrease as the number of correction vectors increases, which would indicate a decrease in residual errors in the tracked space. Figure 3.6 illustrates the magnitudes of the 24 measurements each subject generated. As shown, the magnitudes of the correction vectors gradually decreased throughout the calibration process down to an order of less than 3 cm, which implies that the user-perceived errors were smaller than 3 cm. These results are very similar to the corrections achieved by Czernuszenko et al. in their experiment. Figure 3.7 shows the visualization on pCubee before and after calibration.



(a) Before Calibration



(b) After Calibration

Figure 3.7: Visualization on pCubee before and after calibration. Note the virtual contents were skewed across multiple screens and are mismatched against the physical frame before calibration. The virtual contents aligned visually with the physical frame after calibration.

There are currently a number of limitations to the dynamic visual calibration approach when applied to pCubee, including a potentially inadequate assumption about the dynamic system, a lack of depth reference when calibrating from directly centered and in front of each screen, and uncorrected angular errors.

As discussed previously, an assumption most calibration techniques make is that the display system will remain static, which is true for most fish tank or inward-facing geometric displays reported in the past. The main difference with tangible outward-facing geometric displays is that both the system and the user can be dynamic during usage. Display movements can cause additional distortions and make data acquisition and error compensation more challenging. In our calibration implementation, we made a similar assumption that the user's head position would remain static because pCubee allows the user to perform most manipulations in their hands and to move their heads only slightly. While this is somewhat true, the assumption is violated when the user adjusts their head position, resulting in additional errors that are not accounted for in our current technique. We believe these additional errors resulted in the fluctuations of a number of correction vectors shown in the figure, such as in the 1st and 15th measurements. A more comprehensive calibration approach would be to use lookup tables that represent the absolute positions of both the display and the head sensors. This would require a different and more complex calibration method, involving over a few hundred data measurements [27] compared to only 24 in our current approach.

Our volunteer subjects commented that calibrating from a front view of each screen was difficult due to a lack of depth reference. Given that pCubee only supports monocular viewing, subjects were unable to accurately judge depth using the physical frame when viewing only one screen. Subjects indicated that calibration was the easiest when viewing from the corners and edges of the cubic display, which allowed them to effectively compare whether the physical and virtual frames were parallel. Moreover, our calibration technique only corrects for position errors, not angular errors, which is another important factor in pCubee due to the orientation changes that

could occur with a dynamic display. An additional correction table could be built by measuring orientation difference at each grid point to compensate for angular errors.

3.5 Interaction Techniques

pCubee enables a number of tangible interaction techniques that are distinct from static volumetric and geometric display systems. We explored four novel interaction schemes suitable for tangible outward-facing geometric displays. These include static visualization of 3D models, dynamic interaction with virtual objects through simulated physics, walk-through and fly-through scene navigation, and bimanual stylus interaction to support selection and other interesting applications.

3.5.1 Static Visualization

A natural technique for viewing a 3D scene within pCubee is to rotate it to look into different sides of the box, a common interaction scheme that is also demonstrated in other outward-facing displays [30, 39, 52]. The interaction metaphor requires small sized or miniature virtual objects that fit within the bounds of the physical box, but the visualization effect is as compelling as if the users are observing a physical object through a glass box.

In this case, while objects in the scene are static (i.e. stationary within the display), the perspective of the scene is constantly changing corresponding to the movements of the display and the user's head. Complex 3D shapes can be viewed from different sides in a tangible manner. High quality real-time rendering and the visual quality of the LCD panels allow for highly detailed representations of different types of 3D data including CAD, architectural, or anatomical models. Static information visualization can also be an important application for observing virtual artifacts, such as in museums or in schools, at which the equivalent physical models cannot be presented.

We demonstrate this interaction technique with a 3D model of a Japanese

Figure 3.8: Static visualization of a Japanese Noh Mask artifact.

Noh mask[1], as shown in Figure 3.8. The artist's signature stamp on the backside of the Noh mask is clearly visible if the user looks into the backside screen.

3.5.2 Dynamic Interaction

Extending the metaphor of virtual objects inside the box, we can make them dynamically react to the movement of the display with simulated physics supported by the software implementation. In this case, objects in the scene are dynamic and move within the display due to simulated forces, including gravity, collisions with the inner sides of the display box and also other virtual objects. This interaction scheme was also demonstrated in Cubee [52], but the display movement was restrained due to the large and heavy display. The pCubee prototype allows for finer and more responsive control due to its small size. The interaction between the user and the virtual objects is indirect: the user moves the box, and the box moves the objects through downward sliding under gravity or colliding with the walls of the box.

Reactive object interaction is well-suited for games or entertainment ap-

[1]We thank Xin Yin from Ritsumeikan University Computer Vision Lab for providing the Japanese Noh Mask model.

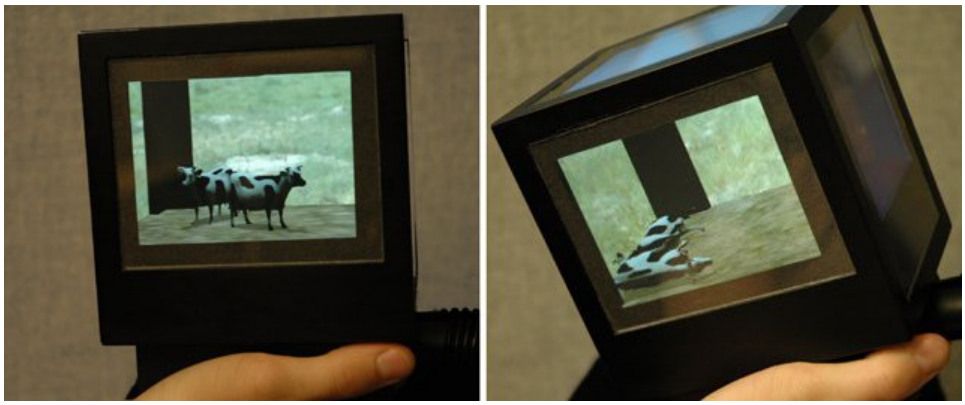


Figure 3.9: Dynamic interaction with cow models.

plications where dynamic 3D content can be fun to play with, as with many existing physics-based toys and games. A tangible outward-facing geometric display like pCubee can serve as a platform to implement similar toys and games in the virtual reality domain. We demonstrate dynamic interaction with virtual cows tipping and bouncing inside pCubee, as shown in Figure 3.9.

3.5.3 Large Scene Navigation

Similar to volumetric displays, larger virtual scenes that extend outside the bounds of the physical box present a problem in navigating to see distal parts of the scenes. We propose an interaction scheme for navigating 3D landscapes in pCubee in which the viewpoint translates in the direction that the display is tilted. This interaction technique is similar to using a joystick in virtual cockpit flight simulation games.

We achieve this effect by placing a ball with simulated gravity inside the scene that reacts to the user's tilting motion, serving as a virtual "navigator". By centering the virtual cameras on the "navigator", the user can explore around the scene as it rolls through the landscape. By adjusting simulated gravity (or other effects), we have control over the effect the display tilt has on the traversal speed, which is like a control-to-display ratio.

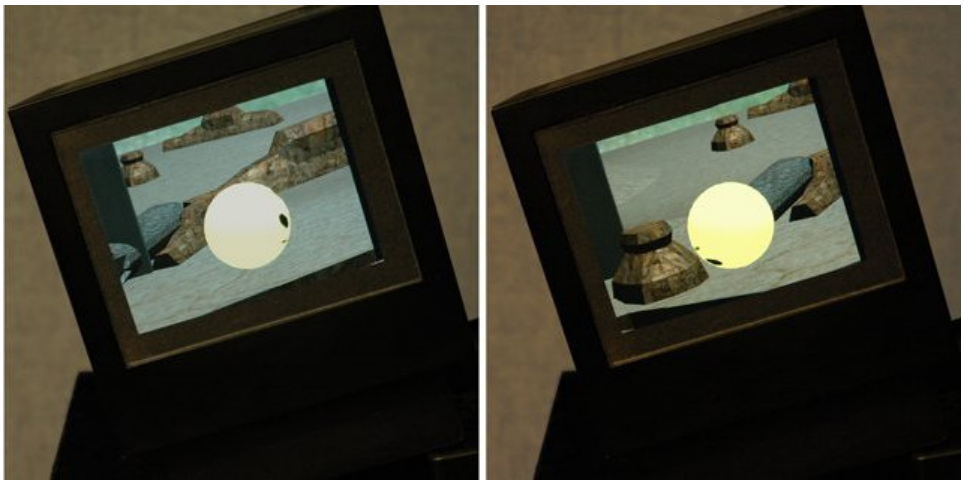


Figure 3.10: Navigation through a virtual landscape using display motion.

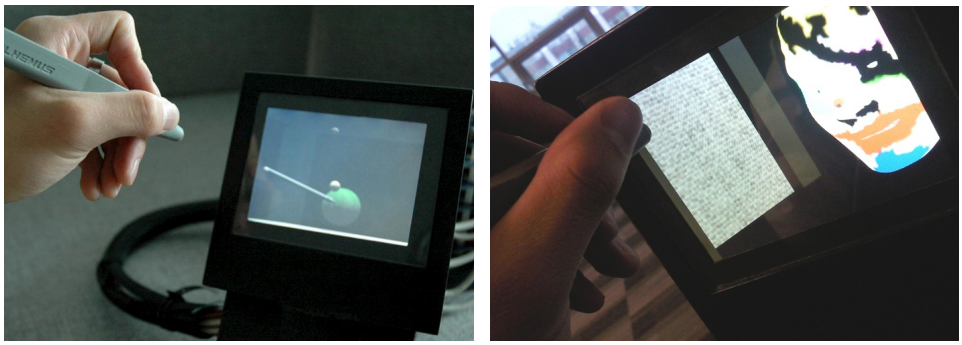
By using simulated earth gravity, the user feels they are adjusting the tilt of a hill for the ball to roll downward, which is quite natural for pCubee. Figure 3.10 depicts a desert navigation application we prototyped.[2]

An alternative can be a "fly-through" style navigation, in which the displacement of pCubee from its original position constitutes its velocity and the rotation constitutes its angular velocity. These types of navigation interactions may be useful for virtual museums, where the user can bring distal exhibits into their perspectives, and also gaming, where users need to go to different places on a large-scale map to accomplish different objectives. With outward-facing displays made wireless and portable, it is also possible to create an augmented reality (AR) application that unifies a virtual world within the real-world space, in which the user walks around a physical room to observe objects in museums or treasure-hunt-style games.

3.5.4 Bimanual Stylus Interaction

Direct selection and manipulation of objects are needed in applications that require fine-grained user control, such as 3D widget interaction, CAD de-

[2]We thank Team Timeless from Simon Fraser University for providing the desert scene.



(a) Object selection (b) Virtual painting

Figure 3.11: Bimanual interaction with pCubee using the Polhemus stylus.

sign, virtual sculpting or painting. Traditionally, direct 3D manipulation techniques are implemented with either stylus or tracked fingers to unimanually reach into the virtual space with static setups [16, 21]. The tangle nature and small form factor allow the user to use pCubee and an additional 3D input device in tandem to support the theoretical model of bimanual motor behavior proposed by Guiard [25]. Guiard's Kinematic Chain (KC) model suggests that two hands work as a pair of asymmetric motors assembled in a serial linkage, based on observations in motor tasks such as page manipulation by the non-dominant hand in handwriting.

3D bimanual control has been explored with tangible interfaces (Hinckley et al. [28]) and dual mouse setups (Balakrishnan and Kurtenbach [4]) to control the view with the non-dominant hand and cursor or cut-plane manipulation with the dominant hand. Results from these studies have demonstrated that bimanual interaction can be more natural and faster than unimanual mouse input in certain 3D tasks. Coupled manipulation and visualization in pCubee allows the user to hold and view the display in the non-dominant hand for tangible, contextual movements (i.e. rotate the display to the desired perspective), while the dominant hand can be used for finer movement to control and manipulate objects inside the display.

Using the Polhemus stylus, we experimented with multiple schemes to manipulate virtual content inside pCubee such as by creating a virtual ex-

tension to the physical pointer, as was done with Cubby [16]: when the user positions the stylus near pCubee, the virtual tip appears inside the display to provide an extension. The virtual stylus allows the user to select objects that the virtual tip comes into contact with. Figure 3.11a shows a scenario where we demonstrate pointing into and interacting with the scene with the virtual extension, creating a bimanual interaction metaphor similar to using a physical pointer to pick and poke objects held in ones hand.

Another approach is the line-of-sight selection technique as was done by Grossman et al. [21], which lets the user point the stylus into the pCubee space at a distance t select objects that are in line with the direction of the stylus. We demonstrate the interaction scheme with a 3D virtual painting application on the Japanese Noh Mask model, as shown in Figure 3.11b. The user can point into the scene and press the stylus button to spray different colors onto the 3D model, which updates the model's texture in real time. The area of effect of the spray can be controlled by the distance between the stylus and the painting surface. The interaction scheme draws parallels to holding and spray-painting a physical object.

The above stylus interaction schemes require precise stylus calibration for a consistent visual match between the physical and virtual styluses, which is currently a challenge for pCubee. Due to monocular-only viewing in the current system, it is difficult to achieve the desired sense of smooth transition between the physical and virtual domains. Given these limitations, we explored a third interaction scheme in which the user can manipulate virtual content within pCubee by using the stylus in a physical workspace that is decoupled from the visualization, much like how it is done currently with existing 3D input devices. We implemented the interaction scheme in a 3D cubic puzzle task that involves pointing, selection, manipulation and placement of 3D objects. We will describe the 3D cubic puzzle task and the interaction schemes in the context of spatial reasoning tasks in Chapter 5.

3.6 Summary

In this chapter, we reported our analysis on the pCubee display with respect to its system design, tracking calibration and interaction techniques. First, We described the hardware components of pCubee, in which we discussed the design challenges that should be taken into consideration for future display development. The current limitations in the hardware design of pCubee, including the thick seams and color distortions at oblique viewing angles, require better display technologies to overcome them. We also described the software components of the system, including the OpenSceneGraph, PhysX, and FMOD engines and how they are integrated in the *CubeeModel* class. The software architecture adheres to object-oriented design and facilitates agile content development for pCubee.

For tracking calibration, we explored a dynamic visual calibration technique adapted from the previous work by Czernuszenko et al. [12] and tested its performance on pCubee. The technique allows the user to naturally look around the display and adjust perspectives at locations where there are mismatches between the virtual content and the physical display frame. We validated that the calibration technique alleviates the visual mismatch problem by using only 24 correction measurements, as compared to a much larger number of data points required by other calibration techniques.

We also described a number of novel interaction techniques that can be supported by pCubee, including static visualization, dynamic interaction, scene navigation and bimanual stylus interaction. The interaction techniques are novel because of the tangible nature of outward-facing geometric displays. We prototyped these interaction schemes in a number of virtual applications, which helped demonstrate interesting applications possible with this unique type of 3D display technology.

Chapter 4

Evaluation of Visual Discontinuity: Effect of Seams

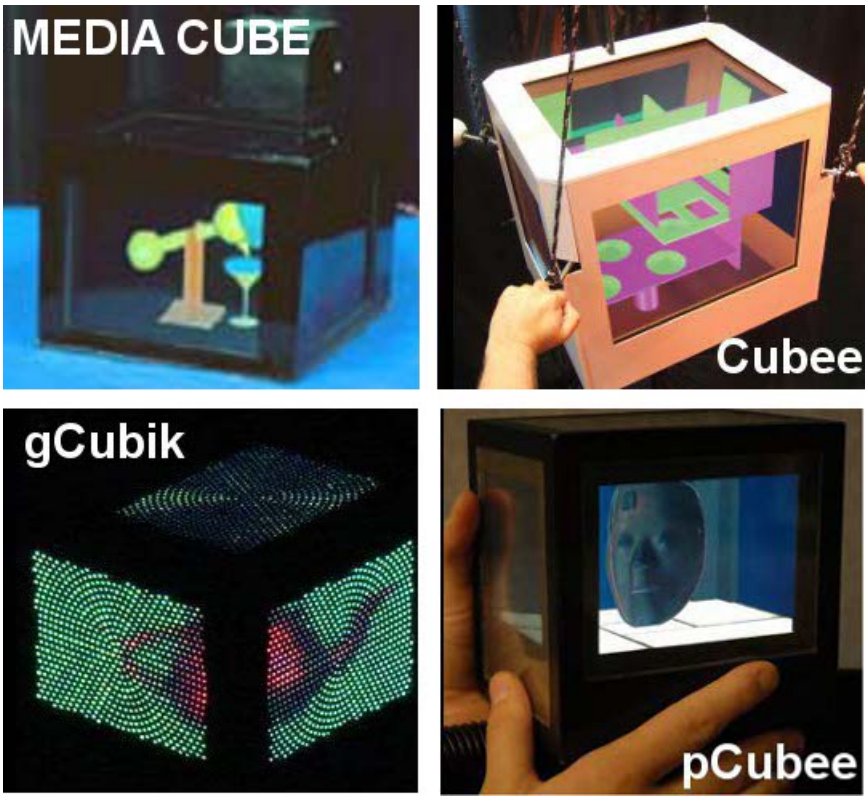


Figure 4.1: Seam occlusion problem with existing outward-facing geometric displays.

The design of multi-screen configurations leads to thick seams at edges between the flat-panel screens. These seams result in disruptions of the 3D visualization when users switch views from one screen to another, which we refer to as visual discontinuity. While the problem is apparent in all previously described outward-facing geometric displays, as illustrated in Figure 4.1, no formal evaluation has measured its effect on the user's viewing experience or 3D task performance.

We conducted an experiment to evaluate the effect of pCubee's seam size in a 3D path-tracing visualization task that has been commonly used in classic evaluations with single screen FTVR displays. Visualization tasks such as path-tracing are suitable stimuli because they can be very vulnerable to visual discontinuity; seam occlusions prevent users from maintaining continuous visibility and thus hinder their ability to resolve the paths. We confirmed this vulnerability in a pilot experiment prior to the user study, in which we compared the path-tracing performance of pCubee and a 2D desktop display. In this chapter, we first report our main study and related findings on the impact of seam size in path-tracing tasks; we then discuss our pilot comparison study and the lessons learned from it, which were valuable for identifying the design parameters in our main experiment.

4.1 Path-tracing Tasks

Path-tracing tasks require users to examine path or graph structures in space, such as to determine the root of a tree branch that is tangled with other branches as illustrated in Figure 4.2. Investigating user behaviors and performances in path-tracing provide implications to real-world applications, including visualization of scientific data such as vector fields and biomedical data such as blood vessels. Traditionally, evaluations relied on path structures of different designs, including top-down trees [2, 51] and information nets [58]. Trends identified in these previous studies are that both structured motion-based and stereo 3D views are better than conventional 2D views, but motion cues are more beneficial than stereo cues alone regardless of the structures that were observed. While head-coupled motion

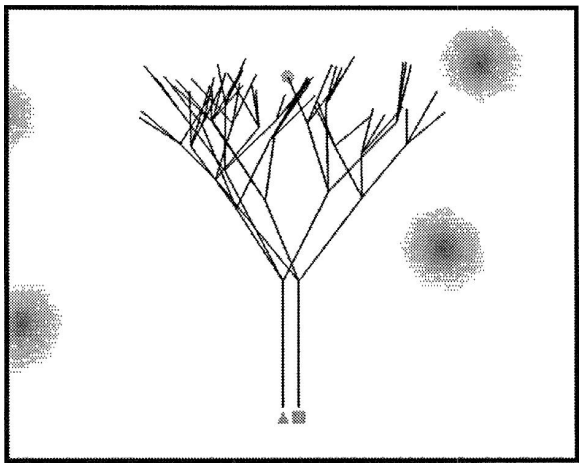


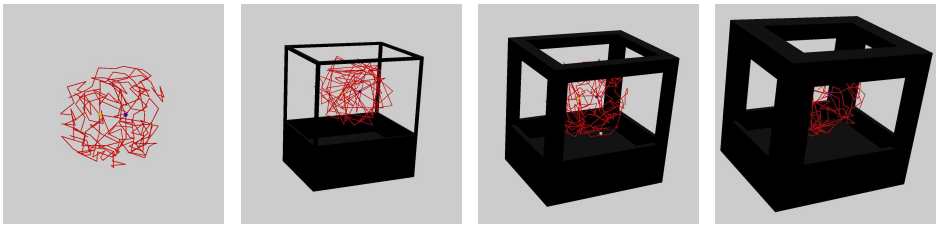
Figure 4.2: A path-tracing task showing two trees interleaving one another. Subjects were asked to identify whether the circle belonged to the triangle or square root at the bottom. Figure adapted from [2].

in pCubee should lead to similar performance advantages, we used variations of these path stimuli to examine issues surrounding the multi-screen aspect and seam occlusions of the display.

4.2 Apparatus

Both the pilot and the main studies were conducted using the workstation and the pCubee device described in Chapter 3. For the desktop display, we used a non head-tracked 24-inch LCD monitor (1920x1200 resolution, 0.27mm pixel pitch) for the pilot study and a 20-inch ViewSonic VP201b display (1600 x 1200 resolution, 0.26 pixel pitch) for the main study. Neither of the displays was stereo-capable due to limitations with the LCD panels.

The experiments were set up on a desk where the subjects performed the tasks while seated. We used a conventional keyboard interface for subjects to enter their responses. The mouse that was used for manipulation in the experiment was set up on the right side of the subject while the pCubee display was set up on the left. Subjects were allowed to freely interact



(a) 0-unit frame (b) 0.3-unit frame (c) 1.3-unit frame (d) 2.3-unit frame

Figure 4.3: Virtual frame structures of varying seam sizes used in the seam size experiment.

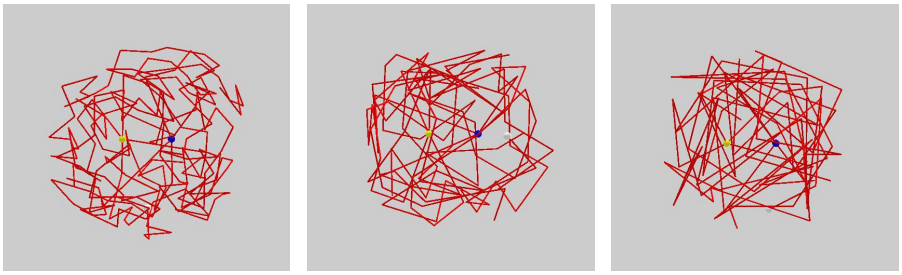
with the pCubee display as they chose (all subjects performed the pCubee conditions seated except one, who chose to walk around pCubee instead of picking it up in the pilot study, and we discarded that data).

Head-tracking for pCubee was done using a pair of headphones with the mounted head sensor, which subjects had to wear for the duration for the study. We used a fixed head-to-eye offset (10 cm below, 5 cm in front and 3 cm to the left to approximates monocular viewing with the left eye) for head-coupled perspective rendering on pCubee. In the experiment, 1 "unit" in the virtual space was analogous to 1 cm in the real world. No calibration of the tracking data was performed in this study.

4.3 User Study: Effect of Seam Size

To investigate the effect seam size has on visualization tasks, we evaluated user performance in path-tracing under the presence of different frame occlusions.

Because we do not have physical prototypes different than the current pCubee, the experiment was conducted on a desktop setup. We created four virtual frames to simulate the different versions of pCubee that would have been built using panels of different seam sizes. As shown in Figure 4.3, we included 0-unit, 0.3-unit, 1.3-unit and 2.3-unit virtual frames to create different occlusion levels around the path stimuli. The 0-unit frame represented the best possible scenario when no frame was present, and the other three



(a) 1.3-unit segment (b) 2.6-unit segment (c) 3.9-unit segment

Figure 4.4: Spherical structures of varying segment lengths used in the seam size experiment.

frames were designed to be within the possible minimum and maximum ranges that could be physically constructed (the seam size on the current physical pCubee is 2.3 cm, and we anticipated that the next pCubee prototype would be constructed with thinner display panels that create 1.3-cm seams). The virtual frames were generated to have viewing windows that were analogous in size as the physical pCubee display.

We designed a set of spherical path structures (see Figure 4.4) to more accurately represent searching in 3D space compared to stimuli used in past evaluations. Each structure was constructed by spanning three paths from a center node (referred to henceforth as the root), with each path containing multiple segments. The paths were placed around the surface of a 3-unit radius "shell" centered at the root. In total, three different spherical structures were generated using varying segment lengths to create the paths, including 1.3-unit, 2.6-unit and 3.9-unit segments as illustrated in Figure 4.4. We were interested in whether varying segment lengths alone at each frame occlusion level would affect users in their viewpoint selection behaviors when performing path-tracing. We suspected that users would switch across multiple screens more if segments were long enough to appear on more than one screen simultaneously. In our design, 3.9-unit segments were easily viewable across two screens under all virtual frame occlusions, 2.6-unit segments were mostly blocked by only the 2.3-unit frame, and 1.3-unit segments were completely blocked by both 1.3-unit and 2.3-unit frames.

Nodes used to connect the segments were randomly generated on the 3-unit shell with a 0.8-unit minimal node separation distance. To create more irregular and random path structures, we varied the distance between a node and the 3-unit shell ($\pm$0.4 unit variance from the surface radius) and also the length of each individual segment (a random factor of 0.7 to 1.3). We also constrained the total length of each path to be approximately 30 units to balance the difficulty across stimuli of different segment lengths (i.e. 1.3-unit structures had 23 segments on each path; for 2.6-unit, 11 segments; and for 3.9-unit, 7 segments).

With traditional path structures, we discovered in our pilot study that subjects were able to narrow down the task by ignoring large portions of the stimuli where the trace target was not present. Whereas in spherical path structures, each path occupied a large region around the spherical surface, and subjects would not be able to easily isolate the paths from a single perspective. Compared to the top-down tree used by Arthur et al. [2] and the "radial-spanning tree" used in our pilot study, the spherical path stimulus was a more spatially 3D design similar to "information nets" [58]. We expected that subjects would be inclined to stay on the side of the stimulus where occlusions to the path being traced were minimal, and thus they could take advantage of large viewpoint changes to rotate and maintain the most desirable views.

The task involved two spherical path structures placed side by side in the center of the scene in each trial, separated horizontally by 1 unit between the two roots that were represented by a blue sphere and a yellow sphere respectively. A target node, represented by a white sphere of the same size, was randomly placed at the end of one of the paths in one structure. The goal of the task was to determine which root the target node was connected to. Subjects indicated their answers by pressing keys that were color-coded blue and yellow to correspond to the colors of the roots. To control for performance consistency, we used a fixed set of random seeds to generate the same set of path structures for each subject, but the order of presentation was randomized. We tested and iterated on the path generation algorithm to ensure the chosen design made the path-tracing task non-trivial yet not

so difficult that a typical subject would have more than a 10 to 15 percent error rate.

4.3.1 Condition Design

In total, we tested four conditions of subjects performing the spherical path-tracing task. Each condition involved a different virtual frame rendered over the spherical path structures, which we refer to as *0 Frame*, *0.3 Frame*, *1.3 Frame* and *2.3 Frame*. The four conditions were run on the desktop monitor, and we showed a fixed perspective projection of the virtual scene by placing the virtual camera 60 units from the path stimulus to simulate a typical desktop viewing distance. Subjects were allowed to use the mouse to rotate the spherical path structures along with the virtual frame; we mapped horizontal mouse movements to yaw (z-axis) rotation and vertical mouse movements to roll (x-axis) rotation. We reset the mouse cursor to be at the center of the screen after each mouse click to avoid subjects losing track of the mouse at the monitor's edges.

4.3.2 Method

The experiment used 4x3 within subjects design to evaluate performance across the four frame conditions and the three segment lengths. Subjects were instructed to complete the task as fast as possible while keeping errors to a minimum. At the beginning of each condition, subjects were given five practice trials to familiarize themselves with the presence of different frames, during which the system provided auditory feedback on whether they answered correctly. No feedback was given during the actual experiment.

For each condition, there were 15 consecutive trials of the path-tracing task for a total of 60 trials through the experiment. Within each condition block, we generated five spherical path structures for each of the three segment lengths and randomized their orders across the fifteen trials. Each trial was initialized with the viewpoint as if the subject was looking from the front side of pCubee. Upon completion of the 60 trials, we asked subjects to fill out a short questionnaire on their approaches and challenges

with respect to the seam size and segment length variations. Prior to the experiment, an instruction sheet was given to the subject outlining the task and the procedures as described above.

10 subjects (6 males, 4 females) were recruited to participate in the study with compensation. Due to the limited subject pool, the presentation of the four conditions was counterbalanced such that no condition would be in the same order more than 3 times between all subjects (i.e. each condition would be presented first no more than 3 times, and so forth). The principal dependent variables for the experiment were response times and error rates, in which response times represented the durations from the stimulus onset to a keyboard response and error rates represented the percentages of correct responses provided. Throughout the experiment, we also recorded the locations from which they viewed the path structures, which was done by capturing the virtual camera location relative to the centre of the virtual scene in each frame (i.e. 1 camera location data every 25 msec for our experiment software running at 40 frames per second). This allowed us to analyze how subjects manipulated their viewpoints under different conditions, as described below.

4.3.3 Results

Tables 4.1 and 4.2 shows the statistics for the mean response times and error rates. Two-way repeated measures analysis of variance (ANOVA) was carried out on both variables across the twelve seam size and segment length combinations [3]. We only considered response times from correctly answered trials, because incorrect responses could mean a missed trace that resulted in shortened or lengthened trace times.

For the mean response times, degrees of freedom (DOF) were corrected using Greenhouse-Geisser estimates of sphericity (epsilon = 0.652), and the results showed that response times across seam sizes differed significantly

[3] We removed outliers from the data if the response time of an individual trial was 3 times the inter-quartile range (IQR) away from the 1st and 3rd quartiles of the mean response times within the same condition. In total, we removed 15 outliers out of 600 data points spread across four subjects in all four seam size conditions; *0 Frame*: 4 outliers; *0.3 Frame*: 3 outliers; *1.3 Frame*: 2 outliers; *2.3 Frame*: 6 outliers.

	Seam Size			
Segment Length	0 Frame	0.3 Frame	1.3 Frame	2.3 Frame
1.3	17.85s	17.70s	25.28s	27.41s
2.6	20.16s	20.84s	31.08	30.92s
3.9	17.14s	17.50s	32.32s	30.61s
Average	**18.38s**	**18.68s**	**29.56s**	**29.65s**

Table 4.1: Statistics of mean response times in the seam size study.

	Seam Size			
Segment Length	0 Frame	0.3 Frame	1.3 Frame	2.3 Frame
1.3	10.00%	6.00%	6.00%	8.00%
2.6	4.00%	8.00%	4.00%	10.00%
3.9	2.00%	0.00%	11.00%	8.00%
Average	**5.30%**	**4.70%**	**7.00%**	**8.70%**

Table 4.2: Statistics of mean error rates in the seam size study.

($F(1.956,17.603) = 9.171$, $p = 0.002$). However, no significant differences were found between segment lengths ($F(2,18) = 1.428$, $p = 0.266$) nor interaction between seam sizes and segment lengths ($F(6,54) = 0.806$, $p = 0.570$). Pairwise t-tests with Bonferroni adjustments revealed that mean response times were significantly different ($p<.05$) between the *0 Frame* and *1.3 Frame* conditions, between the *0.3 Frame* and *1.3 Frame* conditions, and between the *0 Frame* and *2.3 Frame* conditions.

For the mean error rates, the data showed no significant difference between the four seam sizes ($F(1.536,13.826) = 0.821$, $p = 0.494$) with DOF corrected using Greenhouse-Geisser estimates of sphericity (epsilon = 0.512) nor between the three segment lengths ($F(2,18) = 0.604$, $p = 0.557$). Further, no interaction effect was found between the two factors ($F(2.706,24.366) = 1.569$, $p = 0.225$).

Figures 4.5 and 4.6 show the plots for the mean response times and mean error rates, respectively, across the test conditions.

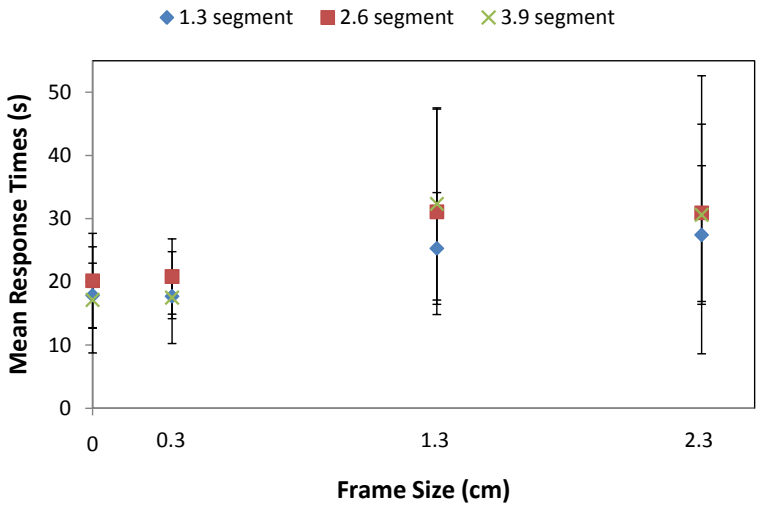


Figure 4.5: Plot of mean response times in the seam size experiment. Note the clear gap between the 0.3 Frame and 1.3 Frame conditions.

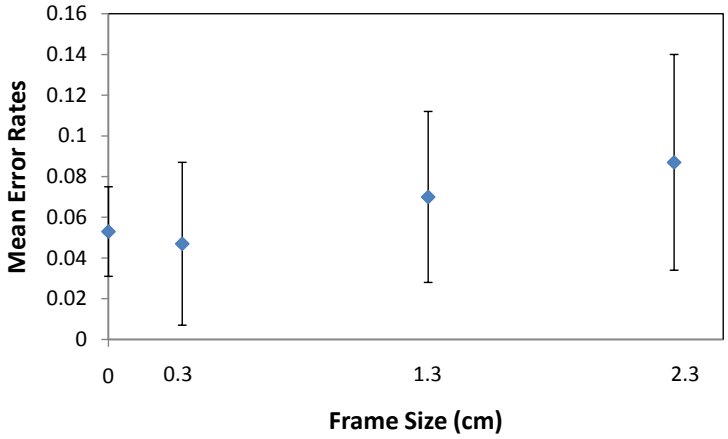


Figure 4.6: Plot of mean error rates in the seam size experiment. Data shown was averaged across all segment lengths within each frame condition.

4.3.4 Discussions of Results

From the statistically significant effects found on mean responses times and the additional viewpoint data that was collected, we observed important trends regarding the effect of seam size in path-tracing tasks. Although the tasks were performed using a desktop interface, we suspect user behaviors in the physical pCubee display would be similar to these results if given the same seam size conditions.

Mean Response Times

As illustrated in Figure 4.5, a significant division in the mean response times between the different seam size conditions can be observed. The *0 Frame* and *0.3 Frame* conditions produced similarly fast response times, while the two thicker *1.3 Frame* and *2.3 Frame* conditions produced similarly slower response times. The gap between the two condition groups were surprisingly large (10 seconds) despite a relatively small sample. Speed-accuracy trade-off was not present to be a factor, as subjects also made an increased number of errors as the seam size increased (though not significantly). The fact that response times of the *0.3 Frame* condition was not significantly different from the *0 Frame* condition suggests that occlusion was not a problem with the thinnest virtual frame. In the *0.3 Frame* condition, a spatial reference with minimal occlusion was commented to be helpful to subjects in the task; one subject felt that the *0.3 Frame* "was the easiest" and "no frame was surprisingly hard" because they did not have a "concrete reference point".

These results indicate that seam sizes that are currently possible mechanically for pCubee (*2.3 Frame* condition) are too big to be effective, as was also pointed out by subjects in the questionnaire responses. However, the results also revealed promise for future display development because small seams are mechanically feasible with better screen panels, but seamless geometric displays are difficult to achieve and require a different type of display technology. From the data, it appeared that seams only have to be below a certain threshold point (in between *0.3 Frame* and *1.3 Frame*) for the display to be effective. There might also be a screen-to-seam ratio that

tolerates thicker seams with increased display real estates. Further investigation into the performance gap will be important for deriving more precise design guidelines for outward-facing geometric displays.

Specific lengths of the path segments did not affect user performance. Mean response times were not significantly different, and no noticeable patterns could be observed from the error rates. Subjects indicated longer segment lengths appeared more difficult due to more self-occlusions within the path structures themselves but not due to the different frame occlusion levels. We compensated for the difficulty difference by adjusting the number of segments, which was reflected in similar response times across all segment lengths used. In general, subjects did not exhibit any changes in the strategies used to solve the task with respect to different segment lengths. These results suggest that the seams' impact might be independent of the size of the virtual content involved.

Viewpoint Analysis

By analyzing the recorded virtual camera positions, we can better understand user interaction with the stimulus and whether there were any notable trends between the conditions.

We defined frustums within the virtual space that represent six viewing regions (i.e. front, back, left, right, top and bottom) in order to visualize how subjects observed the path structures throughout the trials. We binned the recorded virtual camera position data into these six regions using two metrics: i) per-screen usage pattern, which represents the average number of virtual camera data points in each of the six viewing regions across all trials, and ii) multi-screen usage pattern, which represents how many regions were viewed and how long each was viewed in an average trial. Tables 4.3 and 4.4 summarize the virtual camera data using these two metrics; we calculated both as percentages of the total number of data points recorded in each frame condition. We also explored screen usage patterns across different segment lengths but found they were similar, which further confirmed that segment lengths had no effect on how subjects interacted with the scene, as

Seam Size	Front	Back	Left	Right	Top	Bottom
0 Frame	45.61%	9.93%	13.97%	15.41%	8.61%	6.46%
0.3 Frame	59.95%	8.38%	12.70%	12.20%	6.76%	0.01%
1.3 Frame	56.51%	11.73%	14.62%	9.54%	7.60%	0.01%
2.3 Frame	63.97%	10.54%	6.73%	11.73%	7.02%	0.02%

Table 4.3: Per-screen usage pattern in the seam size study

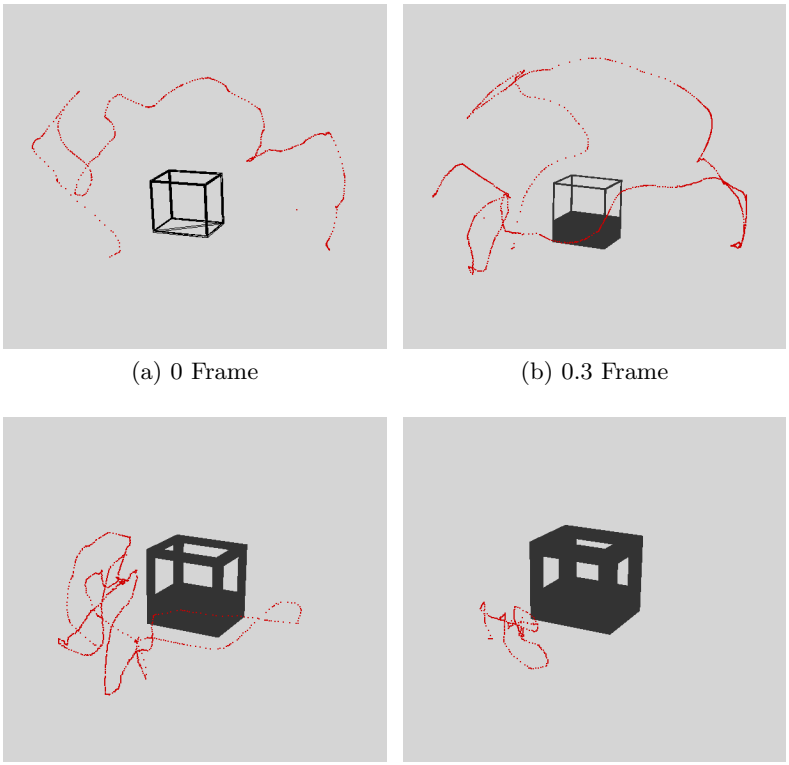
Seam Size	First	Second	Third	Forth	Fifth	Sixth
0 Frame	63.52%	24.22%	9.34%	2.38%	0.47%	0.06%
0.3 Frame	72.72%	20.43%	5.52%	1.08%	0.24%	0.00%
1.3 Frame	75.02%	17.65%	5.60%	1.57%	0.15%	0.00%
2.3 Frame	78.78%	16.49%	3.58%	0.95%	0.19%	0.00%

Table 4.4: Multi-screen usage pattern in the seam size study

was discussed previously.

From the per-screen usage pattern, subjects spent the majority of their viewing from the front, which can be attributed to the fact that trials were initialized from the front view. All other viewing regions occupied no more than 15% each of the total camera data, compared to the 45% to 65% that were spent on the front viewing region. Subjects took advantage of the bottom view in the *0 Frame* condition when there was no occlusion from the bottom, which resulted in the more evenly spread viewpoint data across the six viewing regions compared to all other conditions.

From the multi-screen usage pattern, there was a steady increase in the amount of time subjects spent on one screen in the task as the seam size increased. Close to 80% of the time in a single trial was spent on one screen in the *2.3 Frame* condition. Furthermore, subjects devoted to only two of the six viewing regions close to 90% of their time even in the best scenario, *0 Frame* condition. These patterns revealed that even with the spherical path stimulus that was designed specifically for multi-screen viewing, subjects were able to solve them within small viewing regions with motion, especially in the presence of thick seams that discouraged subjects switching views.



(a) 0 Frame (b) 0.3 Frame

(c) 1.3 Frame (d) 2.3 Frame

Figure 4.7: Viewpoint movement visualization for the seam size experiment. The black dots represent the virtual camera positions where the subject was viewing from around the pCubee frame throughout the trial.

The observed patterns were good indicators to why subjects took more time to solve the task in the thicker frame conditions, because they had to resolve additional segment occlusions when viewing mostly from a single region.

By plotting the virtual camera positions in 3D space, we could see more clearly how subject behaviors changed corresponding to the seam size. Figure 4.7 illustrates the behaviors of how a subject's viewpoint moved around the scene in a single trial in each condition. As shown, in the extreme cases, most movements were in front of one side of the structure when frame occlusions were the largest (*1.3 Frame* and *2.3 Frame* conditions); whereas when the virtual frame was minimal, there was little constraint on where

the subject would view from, and the camera positions spread more evenly around the scene (*0 Frame* and *0.3 Frame* conditions).

Our viewpoint analysis also provided insights regarding the usability of outward-facing geometric displays in path-tracing visualization tasks. As discussed above, our path-tracing task was solvable through motion in a single viewing region; this is consistent with the findings in Ware's study [58] that any structured motion cues, including head-coupled rendering, hand-guided or automatic motions, lead to similar performance improvements in path-tracing. We conclude that there would only be small benefits to using pCubee for rotation compared to using a mouse for the task even without any seam blockages. This also explains why using pCubee and mouse a bimanually offered about the same performance as using just the desktop and mouse in our pilot study, as described below. While path-tracing visualization tasks are still feasible with outward-facing geometric displays with small enough seam size, we argue there are 3D tasks that can better take advantage of the large range of tangible manipulation supported by pCubee.

4.4 Pilot Study: Radial Spanning Tree

In a pilot study which preceded our main study, we explored different visualization and manipulation techniques afforded by pCubee in another path-tracing task, compared with a desktop display and mouse. Instead of the conventional desktop setup which would be the current status-quo solution to solving the task, we were interested in whether pCubee can offer performance benefits to users.

We designed a path structure that spanned outwardly in all directions (see Figure 4.8), which we refer to as radial spanning trees, to better utilize the visualization space inside pCubee. Each of the tree structures contained three levels of branching: the first level extended seven branches from the root; the two subsequent levels after the first-level branches each extended randomly either three or four branches, resulting in a total of 63 to 112 branches per tree. To avoid cluttering and ambiguity due to overlapping

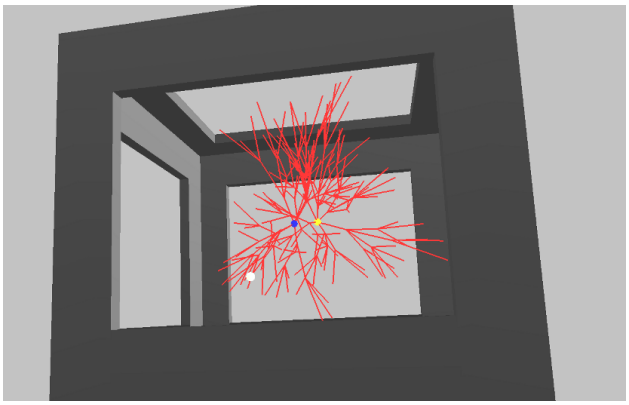


Figure 4.8: Radial spanning tree stimulus used in the path-tracing pilot study.

Condition	Visualization	Rotation Input
pCubee-only	pCubee	pCubee
pCubee-and-Mouse	pCubee	Mouse(bimanual)
Desktop-and-pCubee	Desktop	pCubee
Desktop-and-Mouse	Desktop	Mouse

Table 4.5: Test conditions in the path-tracing pilot study

paths, each branch was of a random length from 0.5 to 1.5 units, with a minimal separation distance of 0.5 unit between the end points of the branches. The spanning direction of each branch was randomly generated, as long as it satisfied the separation distance requirement.

Similar to the main study, the pilot study task was to search through two overlapping trees to determine connections between the roots and a target node, which was randomly placed at the tip of one of the last-level branches in one of the tree structures. The subjects indicated their answers by pressing keys that are color-coded blue and yellow. The tree structures were randomly generated for each trial.

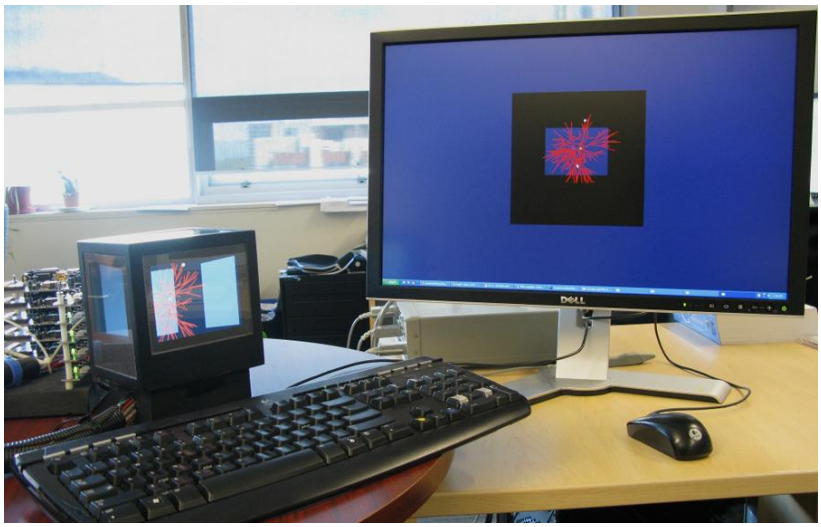


Figure 4.9: Experimental setup for the path-tracing pilot study.

4.4.1 Condition Design

We tested four conditions involving different combinations of manipulation and visualization with pCubee, the desktop monitor and the mouse. In the first condition (*pCubee-only*), only pCubee could be used to visualize the trees; the subjects could pick up the display and look into different sides. In the second condition (*pCubee-and-Mouse*), visualization inside pCubee was coupled with the mouse that could be used to rotate the trees relative to the display for a bimanual interaction scheme. In the third condition (*Desktop-and-pCubee*), pCubee was used as an input device, and the visualization of the trees was decoupled and displayed on the desktop monitor; the rotation of pCubee was mapped with a one-to-one ratio onto the visualization on the desktop monitor. In the forth condition (*Desktop-and-Mouse*), the mouse was used to rotate the visualization on the desktop monitor. The four conditions are summarized in Table 4.5, and Figure 4.9 illustrates our experiment setup.

To ensure consistent task difficulty across the four conditions, we included a virtual pCubee frame on the desktop monitor visualization, as

illustrated in Figure 4.8 previously, to provide the same occlusion as would be observed on pCubee. For conditions involving the desktop monitor for visualization, we showed a fixed perspective projection of the virtual scene by placing the virtual camera 60 units away to simulate a typical desktop viewing position. We also adjusted the size of the visualization on the desktop monitor to match what would be physically seen on pCubee. Due to the pixel pitch difference between the LCD panels, visualization on pCubee is sharper and offers better image quality than the desktop monitor.

For conditions that involved pCubee for visualization, we turned off the desktop monitor to avoid disruption. For conditions that involved the mouse for rotation, the mapping was the same as what was used in the main experiment. Because the mouse cursor was reset to be at the center of the screen after each mouse click, subjects would not be able to see the cursor to appear inside pCubee in the *pCubee-and-mouse* condition.

4.4.2 Method

We measured response time and accuracy to evaluate performance between the four conditions. All subjects were first given verbal instructions about the task and how they could interact with pCubee. They were instructed to be accurate as they could while completing the task as fast as possible. After the briefing, all subjects performed 10 consecutive trials of the path-tracing task for each condition (for a total of 40 trials) and were allowed practice trials before each condition. Upon completion of the 40 trials, we conducted a short interview session with each subject, in which they were asked to rank their preferences for the different interaction schemes. We also asked some general questions regarding what the subject liked or disliked about pCubee. In total, ten subjects (7 males, 3 females) were recruited to participate in the study with compensation.

4.4.3 Results

Repeated measures ANOVA carried out on the two dependent variables[4] showed significant difference in response time ($F(3,27) = 9.395$, $p = 0.004$) but not in error rate ($F(3,27) = 2.444$, $p = 0.081$). Pair-wise t-tests with Bonferroni adjustments on mean response times revealed that the *pCubee-only* condition was significantly slower than the *pCubee-and-Mouse* condition and the *Desktop-and-Mouse* condition. Given the small sample size, no significant ordering effect was found between the conditions using two-factor (univariate) ANOVA ($F(3,27) = 0.473$, $p = 0.704$). Figures 4.10 and 4.11 show plots of mean response time and error rate for each condition.

For the preference ranking, nine out of ten subjects indicated they preferred the *pCubee-and-Mouse* condition most. Table 4.6 summarizes the number of votes each condition received for each rank (equal ranks were permitted). A CHI-square was unsuitable for our analysis because our subject pool was limited and the vote count in each category was small.

4.4.4 Discussions of Results

Despite the small sample size, a number of important lessons was learned about users interacting with the pCubee display which inspired our main study. Here we summarize the results with respect to the mean response times and error rates, and also qualitative feedback and observations we obtained during the pilot study.

Mean Response Times and Error Rates Subjects were the fastest in the *pCubee-and-Mouse* condition, which suggests that the bimanual interaction scheme with pCubee offers benefits in accurately choosing viewpoints and tracing information in 3D scenes. This would accord with the findings by Balakrishnan and Kurtenbach [4], in which bimanual interaction techniques for camera control offered faster performance. A contributing factor

[4]Using IQR range, we removed 35 outliers out of 400 data points spread across 7 subjects in all four conditions; no visible patterns could be drawn. (*pCubee-only*: 6 outliers; *pCubee-and-mouse*: 8 outliers; *LCD-and-pCubee*: 10 outliers; *LCD-and-mouse*: 11 outliers).

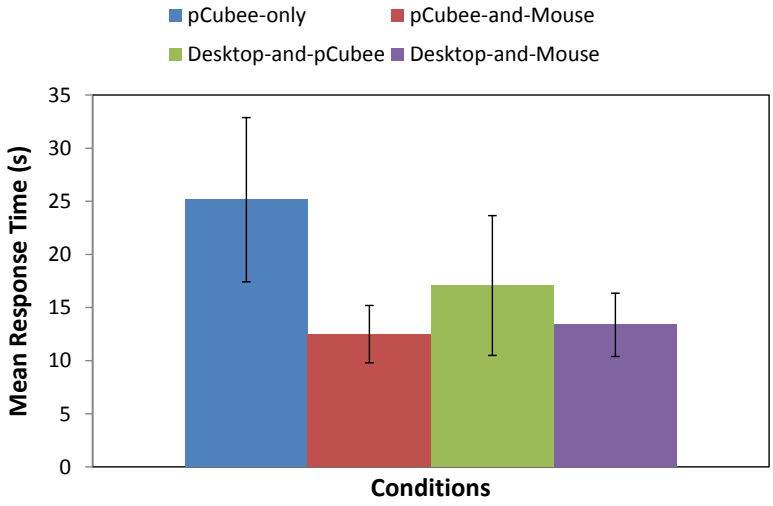


Figure 4.10: Plot of mean response times in the path-tracing pilot study.

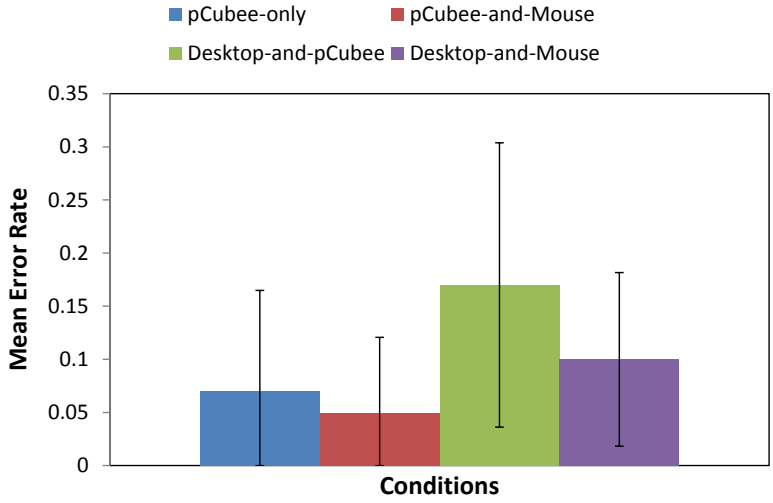


Figure 4.11: Plot of mean error rates in the path-tracing pilot study.

to the faster response times may have been that the path structures could be rotated independently of the virtual frame in the *pCubee-and-Mouse* condition, which reduced the impact of virtual frame occlusions. A surprising

Condition	First	Second	Third	Forth
pCubee-only	2	5	2	1
pCubee-and-Mouse	9	1	0	0
Desktop-and-pCubee	0	1	8	1
Desktop-and-Mouse	0	3	1	6

Table 4.6: Preference ranking in the path-tracing pilot study.

result was that the *pCubee-only* condition was relatively slower versus all other cases, including the *Desktop-and-pCubee* condition which was least-preferred. A lack of user-specific calibration could have been a problem that hindered subjects from selecting their desired viewpoints. The extra keyboard acquisition time (1-2 seconds) when using the *pCubee-only* condition could have biased the response time data, but the amount is small compared to the overall time differences. Additional occlusions may also have played a role, as it was observed that subjects would commonly grasp the sides of the box despite the small base on pCubee's bottom that was intended for holding, and the hand movements could have blocked their view during the task.

In terms of accuracy, subjects made noticeably more errors when they were using the desktop display. Speed-accuracy trade-off was not a dominant factor between the conditions, as subjects performed the fastest in the *pCubee-and-Mouse* condition while making the least errors (the *Desktop-and-Mouse* condition was almost just as fast but with double the error rate). We attributed this difference in accuracy to the pixel pitch difference which resulted in shaper images on pCubee. This was a deficiency we corrected for in our subsequent experimental designs to balance the number of pixels the test stimuli would occupy.

Feedback and Observations During the interview sessions, most subjects commented favorably on the high degree of control available to them in the *pCubee-and-mouse* condition. The bimanual interaction scheme allowed them to first choose their viewpoints through rapid movement of their head

and display and then fine-tune the rotation of the path structures with the mouse to get the most desired view. A number of subjects commented that the interaction was intuitive to them as if they were holding real objects in their hands. On the contrary, subjects felt that the *Desktop-and-pCubee* condition was unintuitive and cumbersome.

While in general subjects disliked the weights and cables which made pCubee difficult to manipulate, we observed the thick seams to be an especially problematic factor that affected how subjects interacted with the cubic display. Subjects indicated the thick display seams were a major impediment for being able to perform the task. Initially, we were concerned that subjects would rely heavily on the mouse input in the *pCubee-and-Mouse* condition, as they were most accustomed to mouse interaction. While subjects utilized both pCubee and the mouse bimanually, most manipulation with pCubee was performed at the early stage in a trial when subjects were searching for the best initial viewpoint to perform the task with subsequent mouse rotations. We seldom observed subjects to switch across the multiple screens of pCubee after engaging the mouse, but instead they used slight head movements to "wiggle" their viewpoint back and forth. These evidences suggest that the radial spanning tree was solvable even with a single head-tracked display as long as subjects were allowed mouse control; but more importantly, the presence of the seams discouraged users to better take advantage of the multi-screen aspect of pCubee.

4.5 Summary

In this chapter, we reported an experiment on the visual discontinuity of pCubee. We investigated the effect of seam size by testing user performance in path-tracing tasks under four different levels of seam occlusions. A division of user performance in the task was discovered: subjects were similarly fast when the seam size of the virtual pCubee frame was less than 0.3 unit and similarly slow when the seam size was greater than 1.3 units, which suggests a physical threshold that exists in between.

By analyzing viewpoint data on how subjects interacted with the scene,

it was shown that thicker seams discouraged subjects to utilize multiple screens to perform the task. This led to additional self occlusions of the path structures that subjects had to solve, thus resulting in increased response times with thicker seams. Further, it was also observed that varying segment lengths in the path structures had little effects on user performance and interaction behaviors. This revealed that subjects were mainly affected by how much information was blocked by the thick seams as opposed to whether or not the information could be carried across multiple screens. These observations accord with our pilot study in which we explored different visualization and manipulation techniques afforded by pCubee. Although we identified benefits of using pCubee bimanually with a mouse, the tangible display was used limitedly to perform initial coarse rotation, and subjects mostly focused on rotating with the mouse by using only one screen.

From the studies' results, we concluded that path-tracing is not a suitable task space for pCubee or other tangible outward-facing geometric displays in which seams remain a significant issue. Even in the ideal case with a seamless frame, we observed little benefits in the utilization of the otherwise compelling multi-screen aspect of the cubic display. On the other hand, small-range structured rotation with high quality visualization, such as a single-screen head tracked display, would be sufficient for such tasks, which reaffirmed the findings of Ware and Franck [58]. However, our studies motivate future evaluations to identify the observed seam size threshold more precisely, as it remains unclear how much a screen-to-seam ratio played a role. Further, similar evaluations in other visualization task domains, such as change blindness tasks in 3D, would strengthen our understanding regarding the impact of visual discontinuity in pCubee and similar outward-facing geometric displays.

Chapter 5

Evaluation of Task Performance: Spatial Reasoning

Compared to existing 3D display technologies, pCubee allows users to interact with high quality virtual content in a unified workspace similar to how we manipulate real-world physical objects. Despite a number of limitations we identified in previous chapters, we see potential in pCubee to better assist users in spatial perception and reasoning tasks, such as comparing shapes and performing 3D rotation.

We conducted an evaluation on pCubee to characterize user performance in 3D reasoning. Specifically, we compared pCubee to a desktop display in a shape comparison task similar to mental rotation, exploring interaction and manipulation techniques afforded by the different display setups. This involved the design of a 3D cube stimulus based on previous literature and pilot studies. Building upon the experiment, we also implemented a more complex 3D task that required users to solve a cubic puzzle inside pCubee. In this chapter, we report the evaluation and related findings on the utility of pCubee in our shape comparison task and also in more practical problem solving such as the cubic puzzle task.

5.1 Mental Rotation Tasks

Mental rotation tasks generally require the comparison of two 3D shapes oriented at two different angles (see Figure 5.1), which demands users to

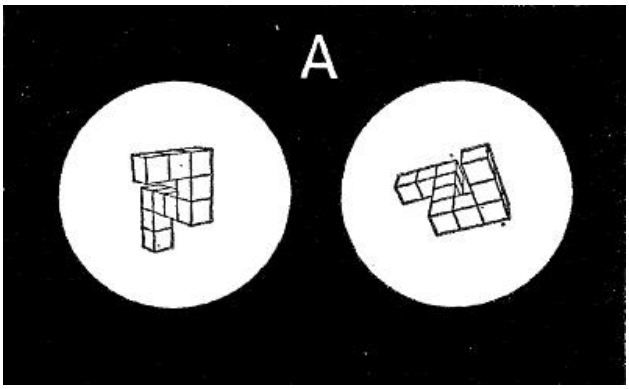


Figure 5.1: A mental rotation task sample in which subjects need to determine if two shapes were identical. Figure adapted from [50].

mentally rotate and match the models in order to determine whether they are identical. Shepard and Metzler [50] pioneered studies in this domain and discovered that the response time for users to decide whether two 3D shapes matched was linearly proportional to the angular difference between the two shapes. In further research, Shepard and Cooper [49] identified that user behaviors for rotations within the depth plane (2D rotations) and in depth (3D rotations) were similar, concluding that the task was sensitive not to the axis of rotation but rather only to the angular difference between the shapes.

More recent research has been in the area of cognitive psychology, identifying trends in learning effects [55], where users were found to perform equally fast in familiar orientations with practice, and gender differences [37, 45], where a large performance advantage in favor of the males was clearly shown. There is a large body of work on mental rotation with 2D shapes and letters [8, 9]; however our evaluation focuses on 3D shapes in virtual space.

Compared to the path-tracing visualization tasks evaluated in the previous chapter, shape comparison tasks such as mental rotation have a number of characteristics that make pCubee a suitable device to perform the task. First, mental rotation tasks involve stimuli that more closely resemble phys-

ical objects. Users can learn their features to facilitate comparisons through manipulation and view point changes, which is compatible with the interaction scheme provided by pCubee. Second, stimuli in mental rotation tasks also have simpler geometries that we suppose are less vulnerable to display seam occlusions. Unlike path-tracing tasks in which the continuity of the information has been proved to be crucial, the nature of mental rotation tasks requires users to learn and maintain good spatial models of 3D shapes throughout the comparison process. Given the tangible, coupled interaction provided by pCubee, we expect it to offer users a strong spatial reference and more intuitive approach to solving similar tasks compared to conventional 2D display setups.

5.2 User Study: 3D Cube Comparison

We performed a comparison study to evaluate the performance of different visualization and manipulation combinations afforded by the pCubee display, a conventional desktop display and a mouse in a 3D cube comparison task that was similar to mental rotation. We used a standard desktop monitor and mouse setup as this would be the current approach users have to solve the task. We were also interested in using the desktop display for visualization and the pCubee device just as an input device for rotation to investigate the impact of the tangible nature of a 3D device rather than a mouse for manipulating orientation. Thus, we evaluated three different conditions: (i) *Desktop-and-Mouse*, (ii) *Desktop-and-pCubee*, and (iii) *pCubee-only*.

For the experimental stimulus, we used a pair of dice-like cubic shapes. Instead of standard casino dice faces, we rendered different symmetrical icons on the cube faces, including spheres, cylinders, squares, diamonds and stars (see Figure 5.2). As opposed to traditional mental rotation stimuli that were usually evaluated in one fixed viewpoint, our cubic shapes were designed so that subjects can take advantage of large perspective changes.

Given the limitations regarding the seams being thick relative to the screen size in the current pCubee hardware, our stimulus design forced users

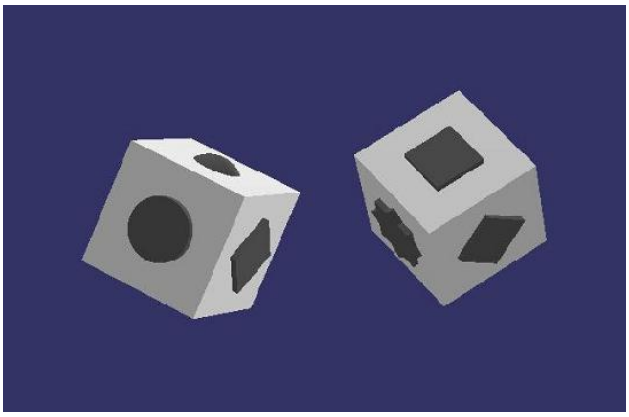


Figure 5.2: A 3D cube comparison stimulus. The two cubes are rotated at different angles and could contain the same or some different face icons.

to rotate pCubee and use multiple screens to examine and compare the 3D cubes. This helped to mitigate the seams' impact from the viewpoints encountered when performing the task. While users would potentially learn the shapes better through interactive viewpoint changes as opposed to a fixed viewpoint, the nature of the task was similar to mental rotation as users were not allowed to rotate each cube independently.

We centered the pair of 3-unit cubes, separated horizontally by 5.5 units, within the virtual scene (pCubee itself is 14.5 units wide). We used five icons to map on to the six faces of each cube, allowing for one duplicated icon because we found the comparison to be trivial if all six faces were unique. The arrangements of the icons on the cubes were randomly generated across all trials. In total, there were four angular differences we tested between the cube pair, including 45, 90, 135 and 180 degrees (we found the 0-degree comparison to be trivial). Each cube pair was first rotated together to a random, arbitrary orientation (a random axis of rotation and a random angle) and then a target angular difference was applied to one of the cubes along the same random axis.

The goal of the cube comparison task was to identify whether the two cubes presented on the display were "identical" or "different"; subjects in-

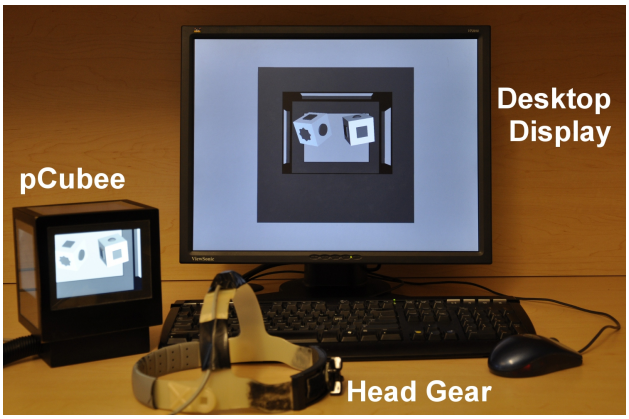


Figure 5.3: Experimental setup for the cube comparison experiment.

put their response on the keyboard by pressing either "y" for the identical cases or "n" for the different cases. For the "identical" cases, the icon arrangements were equal between the cube pair, and it would be possible to rotate one cube along some arbitrary axis to exactly match the icons on the other cube. For the "different" cases, we swapped the icons on two random faces on one of the cubes, and it would be impossible to rotate one cube along any axis to produce the same icon arrangement as the other cube. We ensured that the swapping in the "different" cases would not reproduce identical cubes (i.e. swapping two faces containing the same icons).

5.2.1 Apparatus

Figure 5.3 shows our experimental setup. Both the workstation and pCubee software used were similar to what was described in the experiment in the last chapter. For test conditions involving the desktop display for visualization, we used a non head-tracked LCD monitor (a 20-inch, ViewSonic VP201b display panel with 1600 x 1200 resolution). Again, we did not test any stereo conditions due to limitations with the LCD panels on both pCubee and the desktop setup.

Differing from the last experiment in which we used a pair of headphones, we mounted the head sensor on a head gear that could be adjusted to fit into

Condition	Visualization	Rotation Input
Desktop-and-Mouse	Desktop	Mouse
Desktop-and-pCubee	Desktop	pCubee
pCubee-only	pCubee	pCubee

Table 5.1: Test conditions in the cube comparison study.

different users' heads. We calibrated the head-sensor-to-eye offset for each subject at the beginning of the experiment to ensure they received the most accurate viewpoints possible. We applied error compensation in the head sensor data based on correction vectors generated using the dynamic visual calibration approach prior to the experiment. To keep all test conditions balanced, we fixed the head gear on the subjects for the duration of the experiment.

The experiment was set up on a desk where the subjects performed the cube comparison task while seated. We used a conventional keyboard interface for subjects to enter their responses. The mouse used in the experiment was set up on the right side of the subject while the pCubee display was set up on the left.

5.2.2 Condition Design

The three conditions in our evaluation are summarized in Table 5.1.

In the *Desktop-and-Mouse* condition, subjects used a conventional mouse interface to rotate the two cubes together in the desktop display. We used a trackball implementation provided within OSG to allow 3D view manipulation, which was similar to standard viewpoint manipulation interfaces used in existing 3D modeling tools. The rotation mapping of the mouse to the 3D scene was based on its movement on the desktop display. We reset the mouse cursor to the center of the desktop display after each mouse click to avoid the cursor to go outside of the rendering contexts when it was used for rotation.

In the *Desktop-and-pCubee* condition, subjects used the pCubee display

to manipulate the rendered content in the desktop display. We used a one-to-one mapping between the rotation of pCubee and the visualization on the desktop display, so that the orientation of pCubee matches with what was seen in the desktop visualization. As with other tangible 3D input devices, this condition allowed subjects to perform rotation using pCubee in the workspace decoupled from the visualization.

In the *pCubee-only* condition, subjects used the pCubee display to manipulate and visualize the 3D cube stimulus rendered inside. In this case, visualization and manipulation were coupled in a unified workspace, which we believe would offer the most natural interaction scheme for the comparison task. The cube pair remained fixed relative to pCubee, so subjects could look around the display to compare them.

For conditions involving the desktop display, we showed a fixed perspective projection of the 3D cube stimuli by placing the virtual camera 60 units (analogous to 60 cm in real space) away to simulate a typical desktop viewing position. Compared to the previous experiment in which we attempted to maintain the same physical size between display conditions, we corrected for the difference in display parameters between pCubee and the desktop display by scaling the models by a factor of the pixel pitch difference between the two displays. Thus, the cubes appeared bigger on the desktop display but occupied approximately the same number of pixels as in pCubee.

To further balance the visualization qualities for both displays, we rendered a virtual pCubee frame in the desktop visualization to maintain the same levels of occlusions in all conditions. We piloted a short study to show that seams remain a significant factor in our task, which we will briefly discuss in the context of our main evaluation results later in this chapter.

5.2.3 Method

The experiment used a 3x4 within subjects design for the three display conditions and four angular differences. The experiment consisted of 20 consecutive trials of the cube comparison task per display condition for a total of 60 trials. Within the conditions, we generated five cube pairs for

each of the four angular differences and randomized their orders across the 20 trials. In between completion of the 20-trial blocks, we allowed subjects to rest and practice 5 trials for the next condition.

We instructed the subjects to be as accurately as possible when performing the task by providing additional compensation for more accurate results. The system provided auditory feedback on whether they answered correctly throughout the experiment, and it displayed the accumulated score (i.e. accumulated number of errors made) to the subjects upon completion of each 20-trial block.

Post-task questionnaires were given to our participants to collect subjective feedback regarding their interaction experiences. Using 7-point semantic differential questions, we designed 11 questions with which subjects rated on specific aspects of the interaction and mental process involved in the task for each condition. We also asked open-ended questions on what subjects liked and disliked about each interaction scheme. The 11 questions are listed below:

- Q1: Using *condition* to do the task was (Very Challenging\Very Easy)
- Q2: Using *condition* to do the task was (Very Unintuitive\Very Intuitive)
- Q3: Using *condition* to do the task was (Very Unenjoyable\Very Enjoyable)
- Q4: Using *condition* to rotate to the view I wanted was (Very Challenging\Very Easy)
- Q5: Using *condition* to rotate to the view I wanted was (Very Unintuitive\Very Intuitive)
- Q6: Using *condition*, rotating the cubes in my mind was (Very Challenging\Very Easy)
- Q7: Using *condition*, rotating the cubes in my mind was (Very Unintuitive\Very Intuitive)

	Display Conditions					
	Error Rates			Response Times		
Angles	D+M	D+P	P	D+M	D+P	P
45	16.67%	17.50%	20.42%	24.66s	26.36s	30.39s
90	22.78%	16.67%	18.33%	32.88s	34.23s	33.49s
135	18.33%	20.00%	18.33%	32.31s	37.41s	36.87s
180	25.42%	25.00%	17.08%	34.00s	39.10s	32.82s
Average	**20.80%**	**19.79%**	**18.54%**	**30.96s**	**34.27s**	**33.39s**

Table 5.2: Mean error rates and response times in the cube comparison study. (D+M = Desktop-and-Mouse; D+P = Desktop-and-pCubee; P = pCubee-only)

- Q8: Using *condition* to do the task, the cubes appeared (Not at all 3D\Very 3D)
- Q9: Using *condition* to do the task, the cubes appeared (Not at all Real\Very Real)
- Q10: Using *condition*, I felt I performed (Very Slowly\Very Fast)
- Q11: Using *condition*, I felt I performed (Very Inaccurately\Very Accurately)

The experiment took one and a half hours on average, and 12 subjects (7 males, 5 females) participated in the experiment. The principal dependent variables for the experiment were response time and accuracy, and the three test conditions were counterbalanced between subjects to account for ordering bias. We recorded the virtual camera positions from where subjects viewed the scenes as in our previous experiment to analyze how they manipulated viewpoints in different test conditions.

5.2.4 Results

Table 5.2 shows the descriptive statistics for the error rates and response times [5]. Two-way repeated measures ANOVA was carried out on the two variables across the twelve display and angular difference combinations. No significant main effects of the type of displays used was found for either mean response times ($F(2,22) = 1.499$, $p = 0.245$) or error rates ($F(2,22) = 0.177$, $p = 0.839$). A significant difference was found between angular differences in mean response times ($F(3,33) = 7.510$, $p = 0.011$). Pair-wise t-tests with Bonferroni adjustments revealed that 45-degree ($M = 27.136$s, $SD = 2.409$) was significantly faster than 180-degree ($M = 35.304$s, $SD = 2.316$). No interaction effect was found between the display used and the angular differences. We also performed an analysis using linear regression on the two variables; the correlation coefficients r^2 are summarized in Table 5.3. Figures 5.4 and 5.5 illustrate the mean error rates and response times plotted for the different conditions.

Table 5.4 summarizes the 7-point Likert scale responses for the 11 semantic differential questions in the post-task questionnaire. Repeated measures ANOVA showed significant differences between the *Desktop-and-Mouse* and *pCubee-only conditions* on Q6,Q7 and Q8 (F statistics are $F(2,20) = 4.327$, $p = 2.745$E-2; $F(2,20) = 4.462$, $p = 2.498$E-2; $F(2,20) = 9.633$, $p = 1.176$E-3 respectively) and also between all conditions on Q9 in which we asked how realistic the cubes appeared to be in the visualization ($F(2,20) = 13.093$, $p = 2.318$E-4).

Condition	Error Rates	Response Times
Desktop-and-Mouse	0.49	0.69
Desktop-and-pCubee	0.79	0.89
pCubee-only	0.87	0.27

Table 5.3: Correlation coefficients (r^2) in the cube comparison study.

[5] Using IQR range as in the previous experiment, we removed 26 outliers in total out of 720 data points spread across six subjects; *Desktop-and-Mouse* condition: 12 outliers; *Desktop-and-pCubee* condition: 4 outliers; *pCubee-only* condition: 10 outliers.

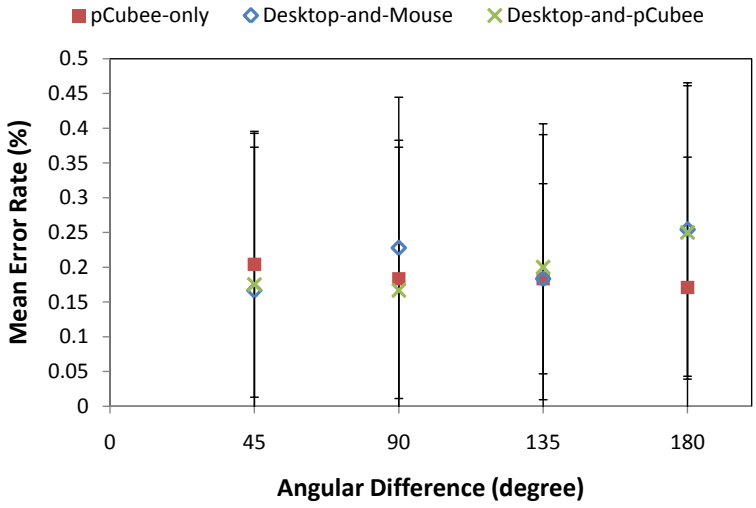


Figure 5.4: Plot of mean error rates in the cube comparison experiment.

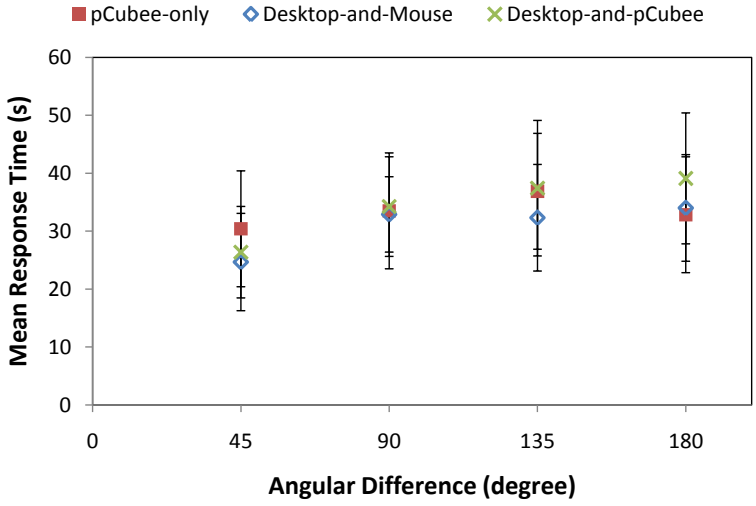


Figure 5.5: Plot of mean response times in the cube comparison experiment.

5.2.5 Discussions

We observed interesting results and trends revealing the advantages and current limitations of pCubee in spatial reasoning tasks. Here we discuss

Questions	D+M	D+P	P
Q1	-0.167	0.333	**0.833**
Q2	-0.0833	0.750	**1.250**
Q3	-0.500	0.500	**1.333**
Q4	0.667	0.917	**1.167**
Q5	0.417	0.583	**1.250**
Q6	-0.417∗	0.417	**0.917**∗
Q7	-0.250∗	0.417	**0.750**∗
Q8	0.250∗	1.000	**2.083**∗
Q9	-0.917∗	0.417∗	**1.833**∗
Q10	-0.667	**0.750**	0.583
Q11	-0.333	**0.833**	0.583

Table 5.4: Questionnaire responses from the cube comparison experiment. Values in bold are the highest in their categories. Values with asterisks (∗) indicate there is a significant difference to at least one other condition.

our main evaluation results in the context of error rates, response times, subject preferences and viewpoint usage.

Mean Error Rates and Response Times

Surprisingly, both mean error rates and response times were very similar in the three conditions, which was contrary to our belief that the coupled workspace in pCubee would better support the task. This was also contrary to findings in a previous study by Ware and Rose [59] on 3D spatial rotation, in which they noted the importance of collocation of the physical rotational input and the virtual object being rotated. Compared to traditional mental rotation results, both response times and error rates were noticeably higher in our experiment; this could be due to the nature of our stimulus which required subjects to view multiple sides to perform comparison instead of viewing from a fixed perspective.

By comparing these data with results from a pilot study we conducted with the same 3D cube stimulus, the characteristics of using pCubee in the task space could be more clearly shown. In the pilot study, we tested only

the *Desktop-and-Mouse* and *pCubee-only* conditions, but we did not render virtual frame occlusions in the desktop display condition. Figures 5.6 and 5.7 show the mean error rates and response times of this pilot study.

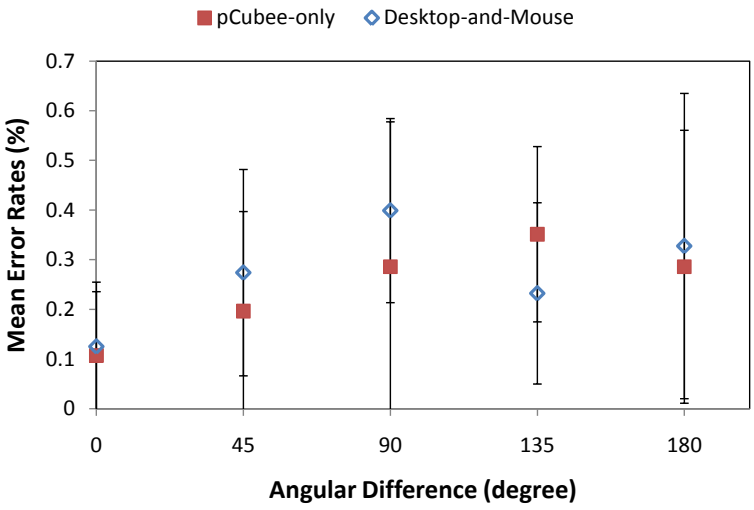


Figure 5.6: Plot of mean error rates in the cube comparison pilot study.

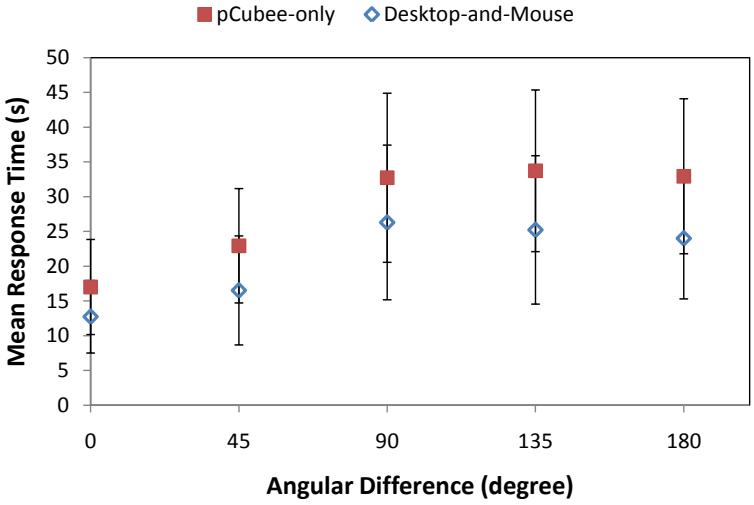


Figure 5.7: Plot of mean response times in the cube comparison pilot study.

From the pilot study results, subjects performed significantly faster using the desktop display than with pCubee when no virtual frame was present. Compared to our main evaluation results, we concluded that seam occlusions remain a significant factor for pCubee even in spatial reasoning such as our comparison task. As in the path-tracing pilot study, we attributed a portion of the time difference to the extra keyboard acquisition time when pCubee was used for manipulation, which was a limitation in the experimental design we failed to address methodologically.

By examining the linear regression coefficients in Table 5.3, the mean response times in our main evaluation were in line with the linear relationship between judgment speed and angular difference previous reported with traditional mental rotation tasks ($r^2 = 0.69$ for the *Desktop-and-Mouse* condition and $r^2 = 0.89$ for the *Desktop-and-pCubee* condition), except for the *pCubee-only* condition ($r^2 = 0.27$). Large standard deviations were present in the error rate data with no noticeable trends; we suspected this is due to the difficulty of the task itself but also possibly the random nature in the generation of the 3D cube stimuli, which resulted in large between-subject variance (unlike Chapter 4, where order was randomized but not the stimuli). Given the relatively small sample size, a larger participant pool would be needed to confirm trends between response times, error rates and angular differences in our stimulus.

An encouraging insight drawn from these results is that seam occlusions accounted for majority of the time difference between our main evaluation and the pilot study. Most subjects disliked the cables, the size and weight of pCubee which restrained their ability to manipulate the device. These are technical factors that could be improved with engineering efforts for outward-facing geometric displays. We argue that with a smaller, lighter and potentially wireless prototype, the manipulation factor could be greatly improved to allow pCubee to outperform the desktop display.

Subject Preferences and Feedback

From the questionnaire, pCubee received the highest average scores in 9 out of 11 questions on performing the cube comparison task. Most importantly, in questions regarding the difficulty and intuitiveness of rotating the cubes mentally, scores given to the *pCubee-only* condition were significantly higher than the *Desktop-and-Mouse* condition. These results revealed qualities of pCubee that made it the "preferred" and intuitive choice for performing our comparison task.

Subjects' feedback were also in agreement with these qualitative advantages pCubee provides: "I rotated it (pCubee) in my hands a lot, which was nice. I was less familiar with this (pCubee) than rotating on a monitor, which was challenging. I tried to visualize pCubee as a box containing the cubes suspended in clear gelatin, and to see into the glass box. This seemed to help." Another subject identified that "it (pCubee) was initially slower, having to move around a physical object, but I got used to it and I had more confidence in my answers." Furthermore, subjects commented on the unintuitiveness of the mouse-based rotation: "it (mouse) took a lot of clicks, and I sometimes lost my frame of reference", and "it was difficult to get the view I wanted because it was less hands-on than actually rotating the cube."

However, from the experimental data, we could not judge whether there was a difference between coupled *pCubee-only* and decoupled *Desktop-and-pCubee* visualization and manipulation in our cube comparison tasks. The questionnaire responses reflected that subjects felt they were faster and more accurate, though not significantly, in the *Desktop-and-pCubee* condition, and they also indicated their preferences for a larger viewing area with the desktop monitor. These results may be due to the fact that objects appeared larger on the desktop display after the pixel pitch adjustment, and they contradicted with previously reported findings in a similar task space regarding the advantages of a unified workspace [59]. It would be worthwhile to further investigate the benefits of coupled and decoupled visualization and manipulation for outward-facing geometric displays.

Display	Front	Back	Left	Right	Top
D+M	26.17%	22.87%	10.46%	11.44%	28.90%
D+P	50.66%	7.50%	5.94%	10.83%	24.90%
P	45.56%	13.08%	6.22%	8.67%	26.34%

Table 5.5: Mean per-screen usage pattern from the cube comparison experiment. The bottom side is not shown since it is occluded in the visualization and was almost never used.

Display	First	Second	Third	Forth	Fifth
D+M	51.89%	24.16%	13.61%	7.15%	3.10%
D+P	57.31%	25.23%	11.34%	4.79%	1.29%
P	58.16%	24.22%	11.39%	4.70%	1.52%

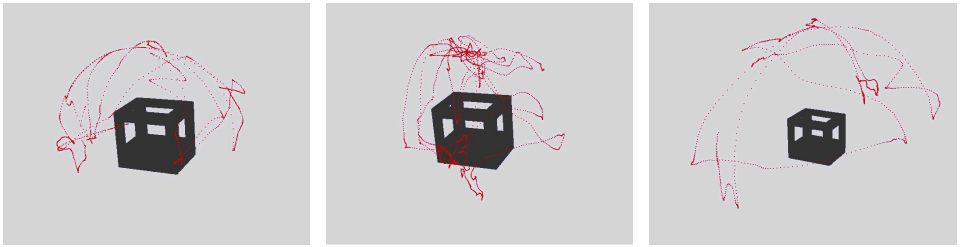
Table 5.6: Mean multi-screen usage pattern from the cube comparison experiment. The sixth screen is not shown because the bottom screen was almost never used.

Viewpoint Analysis

Table 5.5 and 5.6 summarize the virtual camera data using the same two metrics of per-screen usage and multi-screen usage patterns as described in the seam size experiment.

From the per-screen usage pattern, viewpoint usage was spread evenly across different sides especially in the *Desktop-and-Mouse* condition, in which there were little constraints on the mouse manipulation. For pCubee, the front and top screens were most heavily used, which could be attributed to the experimental setup that placed pCubee to have the front and top screens most easily accessible.

More noteworthy trends could be drawn from the multi-screen usage pattern when compared with the seam size experiment. In our cube comparison tasks, subjects made use of at least three viewing regions, which accounted for about 90% of the virtual camera recordings in all three test conditions. To visualize subjects' interaction behaviors more explicitly, Figure 5.8 depicts how one subject manipulated their perspective in each of the



(a) Desktop-and-Mouse (b) Desktop-and-pCubee (c) pCubee-only

Figure 5.8: Viewpoint movement visualization in the cube comparison experiment. The black dots represent the virtual camera position where the subject was viewing from around the pCubee frame throughout the trial.

three conditions in a single trial. As shown in these data, the virtual camera viewpoints surrounded the pCubee frame from all five sides; these interaction behaviors were drastically different from the restrained behaviors due to seam size which were observed in the path-tracing studies. These evidences showed that our stimulus design was successful in persuading subjects to perform a larger range of view manipulation.

5.3 Applications in Spatial Reasoning Tasks

Results from our study have important implications that can aid the design and development of tangible, outward-facing geometric displays in challenging, real-world applications, such as 3D rotation and docking [26] (see Figure 5.9). Imagining rotation in 3D has been proven to be a difficult task [44], and precisely rotating and positioning 3D objects, such as in a 3D Tetris and puzzle applications, remain as open research problems.

Building upon results from the cube comparison experiment, we explored the utility of pCubee in a 3D cubic puzzle task. Specifically, we were curious whether or not pCubee could support more complex rotation and problem solving with its novel and tangible interaction schemes. The 3D cubic puzzle, as shown in Figure 5.10, involved a "working space" where the puzzle pieces were scattered and a 3x3x3 "solution space" into which users had

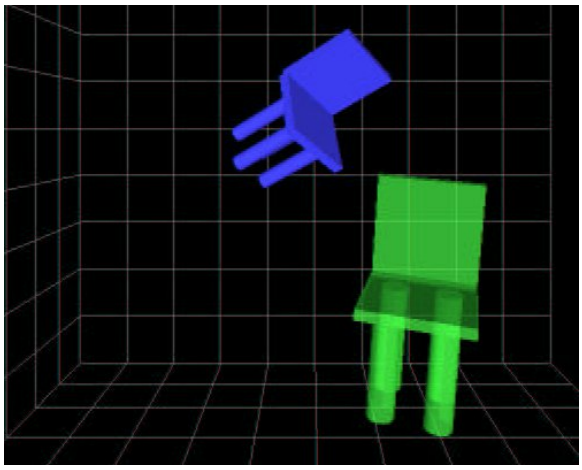


Figure 5.9: A 3D docking task requiring the user to superimpose one chair on top of the other. Figure adapted from [26].

to fit pieces. Using the physical stylus, users were asked to control a virtual stylus inside pCubee to complete the puzzle, which involved pointing, selection, manipulation and placement of the puzzle pieces in 3D space. Different mechanisms were implemented to assist users in performing the task, including visual guides for collisions and snapping in the "solution space". Refer to Appendix C for a detailed description of the interaction schemes that were implemented, as well as the experimental analysis.

We recruited 10 volunteer subjects to attempt the puzzle task, in which we compared user performance in solving a physical cubic puzzle on a desk and a virtual version of an identical puzzle inside pCubee 5.11. We counterbalanced the order of the physical and virtual puzzles and measured the total completion time of each. In a post-task questionnaire, we asked the volunteer subjects about the benefits and weaknesses doing the puzzle in each domain.

All subjects were able to complete the puzzles in both domains, and average completion times were measured to be 147.8s and 327.3s for the physical and virtual puzzles. In the questionnaire, subjects preferred the interactions (selection, manipulation, placement and correction) available

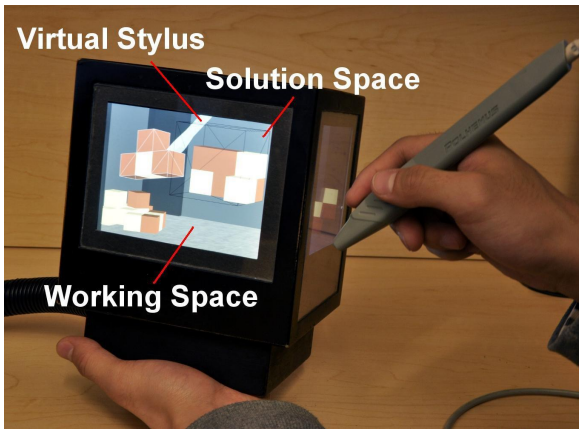


Figure 5.10: pCubee showing stylus interaction with a 3D cubic puzzle.

(a) Physical Puzzle (b) Virtual Puzzle

Figure 5.11: Physical and virtual puzzles used in the cubic puzzle task.

to them in the physical puzzle over the virtual puzzle, but perceived performance was equal for both domains (+1 scores on 7-point Likert scales). Despite their strong preferences for solving the puzzle physically, subjects were only about twice as slow solving the virtual puzzle, which we considered to be acceptance performance given the well known challenges of 3D interaction. These results were encouraging, and they confirmed the practical application potential of pCubee. However, there remains a significant gap between physical manipulation and virtual manipulation with 3D interfaces.

5.4 Summary

In this chapter, we reported an experiment on the capability of pCubee to support 3D reasoning. In our study, we compared the performance of pCubee to a desktop display in a cube comparison task that was similar to mental rotation. We explored different interaction techniques including using pCubee for coupled visualization and manipulation, using the desktop display for visualization and pCubee for decoupled manipulation, and also a conventional desktop and mouse setup.

Due to limitations of thick seams in the current pCubee hardware, we designed a stimulus that required subjects to view from multiple sides to solve the task. The viewpoint data recorded during the experiment showed that subjects utilized multiple viewing regions which was different than what was observed in the previous seam size experiment. While seam occlusions remained to be a factor that significantly affected user performance, we identified that subjects significantly preferred using pCubee over a desktop setup. Both time and accuracy in using the preferred pCubee device were as good as using the larger, more familiar desktop display.

We further examined pCubee for more complex tasks such as solving 3D puzzles that require selection, manipulation and placement. We compared user performance in solving an identical pair of a physical puzzle on a desk and a virtual puzzle inside pCubee. While users were significantly faster in solving the physical puzzle, results from using pCubee were better than what we had expected, which further confirmed the usability of pCubee for spatial reasoning tasks. These results motivate the ongoing evaluation of tangible outward-facing geometric displays to identify their associated benefits in similar domains, such as in 3D visual search tasks.

Chapter 6

Conclusions

In this thesis, we evaluated pCubee, a tangible outward-facing geometric display, to address the lack of empirical work reported on this class of 3D display technology. Through identifying important issues for 3D display evaluations based on previous literature, we analyzed and investigated multiple aspects of pCubee, including its design, tracking calibration, interaction techniques, seam size occlusions and 3D task performance. In this concluding chapter, we revisit the main contributions of this work related to outward-facing geometric displays. Further, we suggest a number of future directions in the context of continual development and evaluation on display technologies that are comparable to pCubee.

6.1 Contribution Summary

Evaluation of the Impact of Seam Size

Through an evaluation on different virtual pCubee frame occlusions in path-tracing visualization tasks, we identified a significant division on response times and interaction behaviors that were dependent on the widths of the seams. Our results suggested a threshold on the physical size of seams that would hinder task performance and discourage users from viewing across multiple screens. For the current size and configuration of pCubee, we discovered that the display could be effective as long as the seams were below a threshold that exists between a 3-13 mm window. More importantly, we demonstrated that path-tracing tasks failed to take advantage of multi-screen viewing and are unsuitable tasks for tangible outward-facing geometric displays.

Evaluation of Spatial Reasoning using pCubee

In an evaluation of a cube comparison task that was similar to mental rotation, we identified that subjects significantly preferred the pCubee display over a desktop setup to perform the task. Both response times and error rates using the pCubee system were shown to be similar to the bigger and more familiar desktop setup, despite the current limitations of pCubee. Our evaluation confirmed the usability of tangible outward-facing geometric displays for 3D spatial tasks and proved them to be a promising approach for improving users' 3D manipulation and reasoning abilities.

Novel 3D Interaction Schemes

We explored interaction schemes afforded by tangible outward-facing geometric displays that are different from existing static 3D systems and are more similar to how we interact with physical objects in the real world. We proposed four novel interaction schemes, including static 3D content visualization, dynamic interaction, large scene navigation and bimanual stylus-based interaction, to demonstrate the application potential for outward-facing geometric displays.

6.2 Future Directions

Throughout the course of this research, a number of challenges related to outward-facing geometric displays have been encountered and are promising directions for future investigation. Here we summarize these future directions with respect to display development and each of the area regarding pCubee that we have investigated.

Display Development

Much of the problems associated with the properties of flat panel displays in the current pCubee system can be overcome with better display technologies. Organic Light Emitting Diode (OLED) display panels can make a significant improvement over LCD panels in both seam thinness and wide viewing angle

acuity. Square panels would also create a true cubic display to have equal viewing real estate around all sides, but square screens are not commercially available. Transmitting all images in one single graphics signal output will be a significant step towards reducing the amount of cabling involving in the current system, allowing the display to be less constrained and more easily manipulated. We foresee future outward-facing geometric displays are capable of providing better 3D visualization and interaction for the purposes of real-world applications and more in-depth evaluations.

Tracking Calibration

Outward-facing geometric displays present a calibration problem that requires corrections on both the display and the user. With the current magnetic tracking system, distortion arises from display movements is a challenge that existing calibration techniques fail to compensate for. Developing a comprehensive tracking technique that addresses position and orientation errors of both the display and the user would be an important step towards a more immersive 3D experience. Other tracking technologies less prone to distortion, such as inertial and optical techniques, could be considered, but accuracy and latency constraints are high for outward-facing geometric displays, like pCubee.

Interaction Techniques

Beyond visualization tasks, precision interaction with virtual content, such as selection and manipulation, remains a difficult problem for 3D display technologies regardless of their designs. The bimanual interaction scheme afforded by pCubee is a unique approach to manipulation with 3D objects that have yet to be thoroughly investigated in this work. It is worthwhile to further explore 3D selection and manipulation techniques that can be supported by pCubee's bimanual, unified workspace.

Visual Discontinuity

While adequate seam occlusions were shown to provide correct occlusion cues and spatial reference, we propose to identify a more precise seam threshold for outward-facing geometric displays. From our evaluation, it remains unclear whether a threshold is absolute or relative to the size of the display panels, though we suspected it could be dependent on a screen-to-seam ratio or the amount of perspective change that is involved in the discontinuity. Given the severity of the effect seam size had in our task performance analysis, deriving a design guideline on the level of visual occlusions would be a significant contribution to future display development.

3D Task Performance

Through evaluation of our cube comparison task, we discovered that pCubee was the preferred choice despite its performance was no better than the desktop display. Future evaluations with an improved prototype (i.e. untethered manipulation, thinner seams) would facilitate better comparison between the systems and confirm any performance benefits in using pCubee. Also, migrating existing 2D spatial psychology experiments to 3D, such as change blindness or visual search tasks, are avenues of future evaluation few has addressed in the past.

Applications

Throughout this thesis, we demonstrated the capability of pCubee to support a number of interesting applications, including artistic painting and 3D Tetris or cubic puzzles. While pCubee is naturally a gaming and entertainment platform, it can be utilized in practical and novel tasks especially in the area of virtual presence. For virtual museums, physical objects can be brought into the display for closer examination and manipulation; also for teleconferencing, 3D representations of the persons can be shown inside the display boundary for a "talking-head" interaction experience.

6.3 Concluding Remarks

We foresee outward-facing geometric displays to become a new paradigm for visualizing, interacting and communicating with digital 3D content, making them an effective tool in practical applications including both education and entertainment. As thinner and higher quality display technologies like OLED panels reach the market, these displays will continue to improve in fidelity and create even more compelling 3D effects.

The evaluations of the various aspects of pCubee reported in this thesis are the first such analyses applied to the domain of outward-facing geometric displays. We believe this work be an important first step towards realizing their full potential in wide-reaching real-world applications as described above. Our aim is that outcomes from our evaluations will provide a reference and a systematic framework for analyses on similar display technologies in the future to facilitate deeper understanding of these displays.

Bibliography

[1] L. Arns, D. Cook, and C. Cruz-Neira. The benefits of statistical visualization in an immersive environment. volume 0, page 88, Los Alamitos, CA, USA, 1999. IEEE Computer Society.

[2] K.W. Arthur, K.S. Booth, and C. Ware. Evaluating 3D task performance for fish tank virtual worlds. *ACM Trans. Inf. Syst.*, 11:239–265, July 1993.

[3] R. Balakrishnan, G. Fitzmaurice, and G. Kurtenbach. User interfaces for volumetric displays. *Computer*, 34(3):37 –45, mar 2001.

[4] R. Balakrishnan and G. Kurtenbach. Exploring bimanual camera control and object manipulation in 3d graphics interfaces. In *Proceedings of the SIGCHI conference on Human factors in computing systems: the CHI is the limit*, CHI '99, pages 56–62, New York, NY, USA, 1999. ACM.

[5] O. Bimber, B. Frhlich, D. Schmalstieg, and L.M. Encarnao. The virtual showcase. *IEEE Computer Graphics and Applications*, 21:48–55, 2001.

[6] K.S. Booth, M.P. Bryden, W.B. Cowan, M.F. Morgan, and B.L. Plante. On the parameters of human visual performance: an investigation of the benefits of antialiasing. In *Proceedings of the SIGCHI/GI conference on Human factors in computing systems and graphics interface*, CHI '87, pages 13–19, New York, NY, USA, 1987. ACM.

[7] S. Bryson. Measurement and calibration of static distortion of position data from 3D trackers. In *Proceedings of SPIE Conference on Stereoscopic Displays and Applications III*, pages 244–255, 1992.

[8] L.A. Cooper. Mental rotation of random two-dimensional shapes. *Cognitive Psychology*, 7:20–43, 1975.

[9] M.C. Corballis, N.J. Zbrodoff, L.I. Shetzer, and O.B. Butler. Decisions about identify and orientation or rotated letteres and digits. *Memory & Cognition*, 6(2):98–107, 1978.

[10] Nvidia Corporation. Nvidia PhysX Engine : http://www.nvidia.com/object/physx_new.html/. Technical report, Nvidia, 2010.

[11] C. Cruz-Neira, J.D. Sandin, and T.A. DeFanti. Surround-screen projection-based virtual reality: the design and implementation of the CAVE. In *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '93, pages 135–142, New York, NY, USA, 1993. ACM.

[12] C. Czernuszenko, D. Sandin, and T. DeFanti. Line of sight method for tracker calibration in projection-based VR systems. In *Procedings of 2nd International Immersive Projection Technology Workshop*, 1998.

[13] M. 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.

[14] C. Demiralp, C.D. C.D. Jackson, D.B. Karelitz, S. Zhang, and D.H. Laidlaw. CAVE and Fishtank virtual-reality displays: A qualitative and quantitative comparison. *IEEE Transactions on Visualization and Computer Graphics*, 12:323–330, 2006.

[15] J.P. Djajadiningrat, G. Smets, and C. Overbeeke. Cubby: A multi-screen movement parallax display for direct manual manipulation. In *Displays*, volume 17, pages 191–197, 1997.

[16] J.P. Djajadiningrat, P.J. Stappers, and C. Overbeeke. Cubby: A unified interaction space for precision manipulation. In *Medicine Meets Virtual Reality 2001*, pages 129–135. IOS Press, 2001.

[17] E. Downing, L. Hesselink, J. Ralston, and R. Macfarlane. A three-color, solid-state, three-dimensional display. In *Science*, volume 273, pages 1185–1189, 1996.

[18] S. Ellis, B. Adelstein, S. Baumeler, G. Jense, and R. Jacoby. Sensor spatial distortion, visual latency, and update rate effects on 3D tracking in virtual environments. *Virtual Reality Conference, IEEE*, 0:218, 1999.

[19] G.E. Favalora. Volumetric 3D displays and application infrastructure. *Computer*, 38(8):37 – 44, aug. 2005.

[20] FMOD. FMOD Toolkit : http://www.fmod.org/. Technical report, FMOD, 2010.

[21] T. Grossman and R. Balakrishnan. The design and evaluation of selection techniques for 3D volumetric displays. In *Proceedings of the 19th annual ACM symposium on User interface software and technology*, UIST '06, pages 3–12, New York, NY, USA, 2006. ACM.

[22] T. Grossman and R. Balakrishnan. An evaluation of depth perception on volumetric displays. In *Proceedings of the working conference on Advanced visual interfaces*, AVI '06, pages 193–200, New York, NY, USA, 2006. ACM.

[23] T. Grossman, D. Wigdor, and R. Balakrishnan. Multi-finger gestural interaction with 3d volumetric displays. In *Proceedings of the 17th annual ACM symposium on User interface software and technology*, UIST '04, pages 61–70, New York, NY, USA, 2004. ACM.

[24] K. Gruchalla. Immersive well-path editing: Investigating the added value of immersion. *Virtual Reality Conference, IEEE*, 0:157, 2004.

[25] Y. Guiard. Asymmetric division of labor in human skilled bimanual action: the kinematic chain as a model. *Journal of Motor Behavior*, 19:486–517, 1987.

[26] M. Hachet, P. Guitton, and P. Reuter. The CAT for efficient 2D and 3D interaction as an alternative to mouse adaptations. In *Proceedings of the ACM symposium on Virtual reality software and technology*, VRST '03, pages 225–112, New York, NY, USA, 2003. ACM.

[27] J. Hagedorn, S. Satterfield, J. Kelso, W. Austin, J. Terrill, and A. Peskin. Correction of location and orientation errors in electromagnetic motion tracking. *Presence: Teleoper. Virtual Environ.*, 16:352–366, August 2007.

[28] K. Hinckley, R. Pausch, J. Goble, and N. Kassell. Passive real-world interface props for neurosurgical visualization. In *Proceedings of the SIGCHI conference on Human factors in computing systems: celebrating interdependence*, CHI '94, pages 452–458, New York, NY, USA, 1994. ACM.

[29] K. Hinckley, J. Tullio, R. Pausch, D. Proffitt, and N. 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.

[30] M. 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.

[31] VertexLCD Inc. LCD panel controller ADB1024-14LVK : http://www.vertexlcd.com/. Technical report, VertexLCD, 2010.

[32] VertexLCD Inc. LCD panel KM050V01-L01 : http://www.vertexlcd.com/. Technical report, VertexLCD, 2010.

[33] K. Ito, H. Kikuchi, H. Sakurai, I. Kobayashi, H. Yasunaga, H. Mori, K. Tokuyama, H. Ishikawa, K. Hayasaka, and H. Yanagiisawa. 360-degree autostereoscopic display. In *ACM SIGGRAPH 2010 Emerging Technologies*, SIGGRAPH '10, pages 1:1–1:1, New York, NY, USA, 2010. ACM.

[34] A. Jones, I. McDoWall, H. Yamada, M. Bolas, and P. Debevec. Rendering for an interactive 360° light field display. In *ACM SIGGRAPH 2007 papers*, SIGGRAPH '07, New York, NY, USA, 2007. ACM.

[35] V. Kindratenko. Calibration of electromagnetic tracking devices. *Virtual Reality*, 4:139–150, 1999. 10.1007/BF01408592.

[36] V. Kindratenko and W. Sherman. Neural network-based calibration of electromagnetic tracking systems. *Virtual Reality*, 9:70–78, 2005. 10.1007/s10055-005-0005-3.

[37] M.C. Linn and A.C. Petersen. Emergence and characterization of sex differences in spatial ability: A meta-analysis. *Child Development*, 56(6), 1985.

[38] M.A. Livingston and A. State. Magnetic tracker calibration for improved augmented reality registration. In *Presence: Teleoperators and Virtual Environments*, volume 6, pages 532–546, 1997.

[39] R. Lopez-Gulliver, S. Yoshida, S. Yano, and N. 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.

[40] J.D. Mackinlay and J. Heer. Wideband displays: mitigating multiple monitor seams. In *CHI '04 extended abstracts on Human factors in computing systems*, CHI EA '04, pages 1521–1524, New York, NY, USA, 2004. ACM.

[41] E. Maksakov, K.S. Booth, and K. Hawkey. Whale tank virtual reality. In *Proceedings of Graphics Interface 2010*, GI '10, pages 185–192, Toronto, Ont., Canada, Canada, 2010. Canadian Information Processing Society.

[42] M. 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.

[43] OSG. OpenSceneGraph : http://www.openscenegraph.org/. Technical report, OSG, 2010.

[44] L.M. Parsons. Inability to reason about an object's orientation using an axis and angle of rotation. *Journal of Experimental Psychology: Human Perception and Performance*, 21(6):1259–1277, 1995.

[45] T.D. Parsons, P. Larson, K. Kratz, M. Thiebaux, B. Bluestein, J.G. Buckwalter, and A.A. Rizzo. Sex differences in mental rotation and spatial rotation in a virtual environment. *Neuropsychologia*, 42, 2004.

[46] Polhemus. Polhemus Fastrak : http://www.polhemus.com/. Technical report, Polhemus, 2010.

[47] Prabhat, A. Forsberg, M. Katzourin, K. Wharton, and M. Slater. A comparative study of desktop, Fishtank, and CAVE systems for the exploration of volume rendered confocal data sets. *IEEE Transactions on Visualization and Computer Graphics*, 14:551–563, 2008.

[48] Prabhat, D.H. Laidlaw, T.F. Banchoff, and C.D. Jackson. Comparative evaluation of desktop and CAVE environments for learning hypercube rotations. In *CS-05-09*, 2005.

[49] R.N. Shepard and L.A. Cooper. *Mental Images and Their Transformations*. MIT Press, 1982.

[50] R.N. Shepard and J. Metzler. Mental rotation of three-dimensional objects. *Science*, 171(3972):701–703, 1971.

[51] R.L. Sollenberger and P. Milgram. A comparative study of rotational and stereoscopic computer graphics depth cues. In *Human Factors Society 35th Annual Meeting*, pages 1452–1456, 1991.

[52] I. Stavness, F. Vogt, and S. Fels. Cubee: thinking inside the box. In *ACM SIGGRAPH 2006 Emerging technologies*, SIGGRAPH '06, New York, NY, USA, 2006. ACM.

[53] A. Sullivan. DepthCube solid-state 3D volumetric display. In *Proceedings of SPIE 5291*, pages 279–284, 2004.

[54] I.E. Sutherland. A head-mounted three dimensional display. In *Proceedings of the December 9-11, 1968, fall joint computer conference, part I*, AFIPS '68 (Fall, part I), pages 757–764, New York, NY, USA, 1968. ACM.

[55] M.J. Tarr and S. Pinker. Mental rotation and orientation dependence in shape recognition. *Cognitive Psychology*, 21(2):233–282, 1989.

[56] S. Tay, P.A. Blanche, R. Voorakaranam, A.V. Tunc, W. Lin, S. Rokutanda, T. Gu, D. Flores, P.Wang, G. Li, P. St Hilaire, J. Thomas, R.A. Norwood, M. Yamamoto, and N. Peyghambarian. An updatable holographic three-dimensional display. *Nature*, 451(7179):694–698, 2008.

[57] Donald Lee Vickers. *Sorcerer's apprentice: head-mounted display and wand*. PhD thesis, 1972. AAI7310165.

[58] C. Ware and G. Franck. Evaluating stereo and motion cues for visualizing information nets in three dimensions. *ACM Trans. Graph.*, 15:121–140, April 1996.

[59] C. Ware and J. Rose. Rotating virtual objects with real handles. *ACM Trans. Comput.-Hum. Interact.*, 6:162–180, June 1999.

[60] C.D. Wickens, S. Todd, and K. Seidler. Three dimensional displays: Perception, implementation, and appliations. Technical Report CSERIAC Rep: CSERIAC-SOAR-89-001, Wright-Patterson Air Force Base, Ohio, 1989.

[61] G. Zachmann. Distortion correction of magnetic fields for position tracking. *Computer Graphics International Conference*, 0:213, 1997.

Appendix A

CubeeModel API

The following routines constitutes the *CubeeModel* API in the pCubee software:

A.1 Mutators

- **void setTexture(const string& texture, TextureCoord coord= CLAMP)**:

  Function: Apply a texture to the virtual model. Texture can be set to repeat or clamp to edge.

  Parameter *texture*: file name of texture to load

  Parameter *coord*: OpenGL parameter set to repeat or clamp textures

  Return: void

- **void setSpecular(float red, float green, float blue, float shine)**:

  Function: Change the specular and shine values of the model's default material

  Parameters: red, green, blue, and shine values

  Return: void

- **void setDiffuse(float red, float green, float blue)**:

  Function: Change the diffuse value of the model's default material

  Parameters: red, green, blue values

  Return: void

- **void setAlpha(float alpha)**:

  Function: Change the alpha value of the model's default material

  Parameters: alpha value

  Return: void

- **osg::Material∗ getDefaultMaterial(**:

  Function: Get and modify the material that is applied to this model when it is not highlighted by the stylus

  Return: A pointer to the model's default material

- **osg::Material∗ getHighlightedMaterial(**:

  Function: Get and modify the material that is applied to this model when it is highlighted by the stylus

  Return: A pointer to the model's highlighted material

- **void setPosition(osg::Vec3 pos)**:

  Function: Set the position vector of the model relative to pCubee

  Parameter *pos*: relative position vector of the model

  Return: void

- **void setRotation(osg::Matrix rot)**:

  Function: Set the rotational matrix of the model relative to pCubee

  Parameter *rot*: relative rotational matrix of the model

  Return: void

- **void makeStatic(osg::Matrix∗ XReference)**:

  Function: Set the model to be fixed to a reference matrix, could be the pCubee display's matrix or the stylus' matrix

  Parameter *XReference*: reference matrix for the model to be fixed to

  Return: void

- **osg::Geode∗ getGeode()**:

  Function: Get the OSG's *Geode* object of the model

  Return: OSG's representation of the model

- **NxActor∗ getActor()**:

  Function: Get the PhysX's *Actor* object of the model

  Return: PhysX's representation of the model

A.2 Accessors

- **float getScale()**:

  Function: Get the OSG's *Geode* object of the model

  Return: scale value of the model

- **osg::Vec3 getPosition()**:

  Function: Get the position vector relative to pCubee

  Return: relative position vector to pCubee

- **osg::Matrix getRotation()**:

  Function: Get the rotational matrix relative to pCubee

  Return: relative rotational matrix to pCubee

A.3 Selection Support

The following routines are included to support the *NodeVisitor* that handles stylus-based selection.

- **bool isSelectable()**:

  Function: Check whether the model is selectable by the stylus

  Return: boolean value for the query

- **void setSelectable(bool val)**:

  Function: Set the model to be selectable or not by the stylus

  Parameter *val*: boolean value indicating whether model can be selectable

  Return: void

- **bool isSelected()**:

  Function: Check whether the model is selected by the stylus

  Return: boolean value for the query

- **void setSelected(bool val)**:

  Function: Set the model to be selected or not by the stylus

  Parameter *val*: boolean value indicating whether model is selected

  Return: void

- **bool isHighlighted()**:

  Function: Check whether the model is highlighted by the stylus

  Return: boolean value for the query

- **void setHighlighted(bool val)**:

  Function: Set the model to be highlighted or not by the stylus

  Parameter *val*: boolean value indicating whether model is highlighted

  Return: void

A.4 Sound Support

The following routines are included to support collision sound playback

- **void setCollisionSFX(FSOUNDSAMPLE* sfx, CubeeAudio* cubeeAudio)**:

Function: Set the model to be selectable or not by the stylus

Parameter *sfx*: sound track to be played for the model upon collision

Parameter *cubeeAudio*: the *CubeeAudio* object that encapsulates the FMOD API

Return: void

- **void playCollisionSFX(const int vol)**:

  Function: play the sound track that has been attached to the model

  Parameter *vol*: volume value for the sound to be played

  Return: void

Appendix B

Sample Hello World Scene

Code Snippet B.1 shows a sample "hello world" scene that initializes a soccer ball that bounces within pCubee

CodeSnippet B.1 Sample scene initialization code to create a soccer ball that bounces within pCubee.

```
/* attaches a light to the scene */
root->addChild(new Light(root.get(), &XCubeeWorld));

/* adds the virtual pCubee frame to the scene */
osg::ref_ptr<Cubee> cubee= new Cubee(
  cubeePhysics->getScene(), 1.0f, osg::Vec3(0,0,0),
  osg::Matrix::identity(), &XCubeeWorld,
  "../geometry/cubeeFrame_normals.obj");
root->addChild(cubee.get());

/* adds a dynamic sphere to the scene */
osg::ref_ptr<Sphere> sphere= new Sphere(
   cubeePhysics->getScene(), osg::Vec3(0,0,0), 5, 5);
sphere->setTexture("../textures/SoccerBall.bmp");
root->addChild(sphere.get());

/* sets a trigger to play a bouncing sound at collision */
sphere->setCollisionSFX(cubeeAudio->load2dSample
  ("../audio/boink1.wav"), cubeeAudio);
cubeePhysics->getScene()->setActorPairFlags(
  *cubee->getActor(), *sphere->getActor(),
  NX_NOTIFY_ON_START_TOUCH | NX_NOTIFY_FORCES);
```

Appendix C

A 3D cubic Puzzle in pCubee

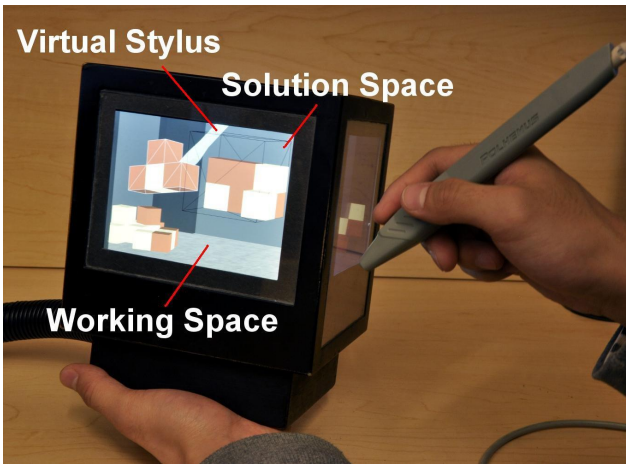


Figure C.1: pCubee showing stylus interaction with a 3D cubic puzzle.

Taking part in the IEEE 3DUI contest in 2011, we explored a decoupled bimanual interaction scheme of using pCubee and the stylus and conducted two informal studies to evaluate its performance in 3D cubic puzzle tasks. The following describes the interaction design, the experimental protocol and the result analysis regarding the cubic puzzle task.

The 3D cubic puzzle, as shown in Figure C.1, involves a "working space" where the puzzle pieces were scattered and a "solution space" where subjects had to fit the pieces into. Subjects could hold pCubee in their non-dominant hand and the stylus in their dominant hand to manipulate the puzzle pieces in a bimanual fashion but in a non-unified workspace.

C.1 Interaction

The interaction of the 3D cubic puzzle task can be classified into four stages, including direct selection and manipulation, large rotation, placement and correction.

C.1.1 Direct Selection and Manipulation

We render a virtual stylus in the 3D scene within pCubee and employ a one-to-one physical-to-virtual stylus mapping that is offset by a user-modifiable distance. Users manipulate the physical stylus in a workspace in real space detached from the "working space" while visualizing the interaction inside pCubee. The orientation mapping is direct: the virtual stylus's orientation is "attached" to the physical stylus. This allows for bimanual rotation control of the virtual stylus within pCubee: users can rotate both the physical stylus and/or pCubee to adjust the orientation of the virtual stylus within the "working space". The position mapping is also direct, but we impose certain constraints to better support multi-screen visualization and interaction with pCubee and a detached stylus. First, we constrain the virtual stylus to remain within pCubee's boundary, which allows users to drag the physical stylus workspace to any desired locations as needed. Second, we constrain the relative position of the virtual stylus to be fixed to pCubee, which prevents users from losing track of the virtual stylus due to unintended translations when rotating pCubee to view different sides.

The virtual stylus acts as a 3D cursor in the "working space" for selection. When the tip of the stylus intersects with a puzzle piece, the piece is outlined with a green wireframe to indicate that it is ready to be selected. Users tap the stylus button once to select the piece and enter a direct manipulation mode. The selected puzzle piece is attached to the tip of the stylus and is directly manipulated as described using the mapping above. To release a selected piece or to place it onto the "solution space", users tap the stylus toggle button once.

C.1.2 Large Rotation

Performing large rotation using a one-to-one mapping with the physical stylus is challenging due to the attached cable and limited wrist rotation. We provide a drag-and-clutch mechanism to allow users to rotate a selected piece in the direction that the stylus is being dragged while the stylus button is held down. A virtual arcball is rendered over the piece to indicate the large rotation mode.

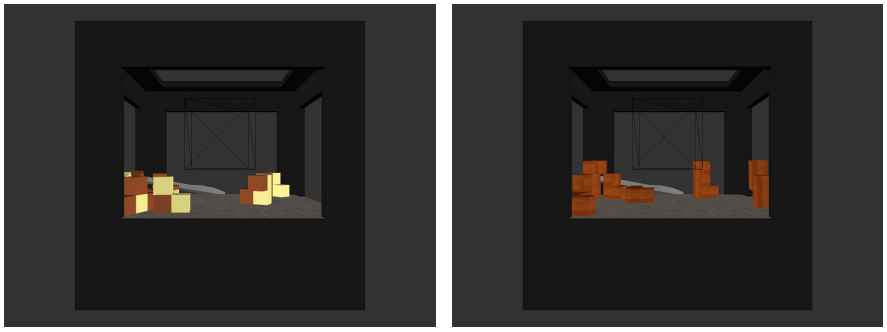
Both position and orientation of the piece remain fixed relative to pCubee when performing large rotation. The clutching mechanism allows users to release the button while repositioning the stylus and pCubee as needed to rotate the piece along any desired axes. Users can either drop the piece by tapping the stylus button or select the piece again by intersecting the piece and holding down the stylus button until it is outlined in green, then releasing the button.

C.1.3 Placement

We implement a snapping mechanism to assist users to place pieces in the "solution space". First, we identify the closest axis-aligned orientation of the selected puzzle piece. Then, we find the closest empty slot in the "solution space" that the axis-aligned piece can fit within. If the distance between the closest empty slot and the center of the puzzle piece is within 1 unit (each puzzle piece is formed with 1-unit cubes), we render a white wireframe in the "solution space" to indicate the possible placement location. The piece is snapped onto the location if the stylus button in tapped while the visual guide is shown.

C.1.4 Correction

Pieces that have been placed onto the "solution space" can be selected and removed; users can also rapidly shake pCubee to reset and drop all the placed pieces back onto the "working space".



(a) Contest Puzzle (b) Google Puzzle

Figure C.2: Two puzzles evaluated in the cubic puzzle experiment.

C.2 Experiments

We conducted two informal user studies to elicit understanding on our stylus interaction design in performing the 3D cubic puzzle tasks. Here we describe the two experiments and their results.

C.2.1 User Study 1: Standard Puzzle

10 novice subjects, who were not familiar with 3D interfaces, and 5 expert subjects, who were familiar, were recruited for a study that required them to solve a "standard" puzzle (see Figure C.2a). All subjects were introduced to the interaction schemes and were allowed a 1-minute practice session. Novice subjects performed only a single trial while the expert subjects did three trails of the "standard" puzzle. In each trial, we measured the completion time and also the time spent on each interaction stage. Post-task questionnaires were used to solicit feedback about the difficulty and intuitiveness of the interactions, and whether or not features such as highlighting, rotation widgets and snapping were helpful.

Results

Figure C.3 shows the times spent on different interactions for trials that were successfully completed by our subjects. In total, 7 novices completed

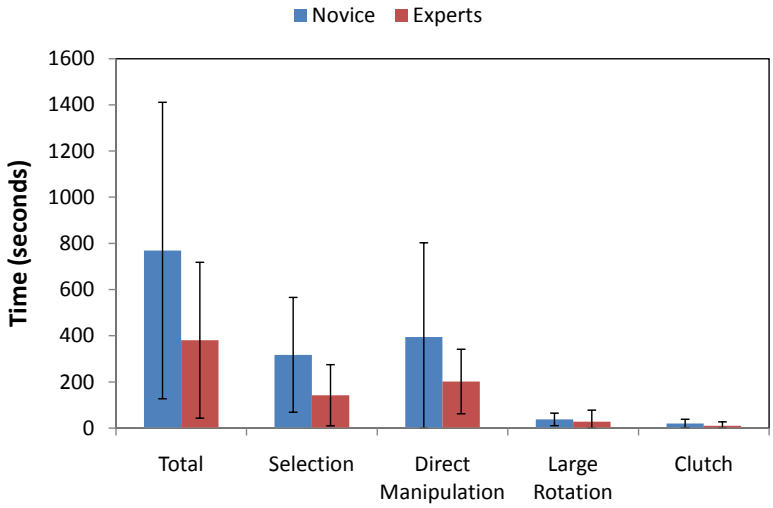


Figure C.3: Time spent on different stages of the standard puzzle.

the puzzle; 2 experts failed to complete the initial trial while another gave up after failing the first two trials.

C.2.2 User Study 2: Google Puzzle

The same 10 novice subjects took part in a second study that compared the performance of a real, physical Google puzzle and a virtual Google puzzle in pCubee (see Figure C.2b). We counterbalanced the order of the physical and virtual puzzles and measured total completion time of each; in the questionnaire, we also asked subjects about the benefits and weaknesses doing the puzzle in each domain.

Results

Figure C.4 shows the times spent on different interactions for trials that were successfully completed. All subjects were able to complete the trials, and average completion times were 147.8s and 327.3s for the physical and virtual puzzles. In the questionnaire, subjects preferred the interactions (selection, manipulation, placement and correction) available to them in the physical

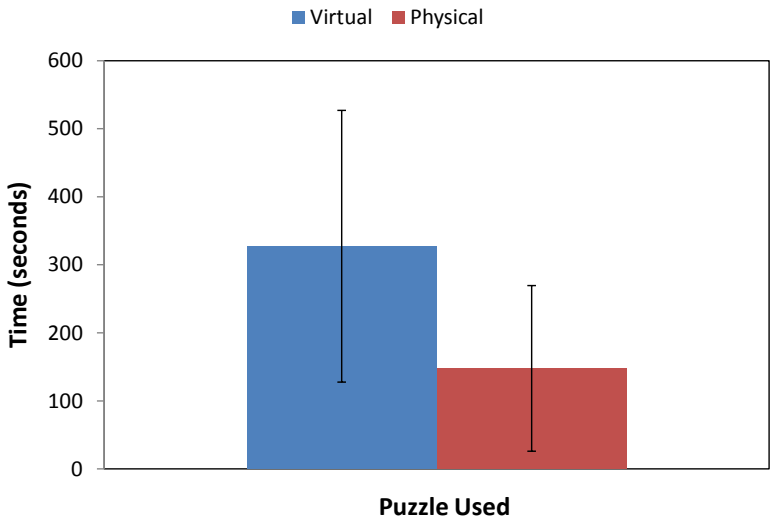


Figure C.4: Completion times for physical and virtual Google puzzles

puzzle over the virtual puzzle, but perceived performance were equal for both domains. (+1 scores on 7-point scales).

C.2.3 Overall Results

We identify the selection scheme which we designed to be a bottleneck, which took averages of 41% (317.3s) and 43% (140.6s) of the total completion times for the standard and Google puzzles respectively. In the qualitative feedback, subjects indicated that the stylus cable limited their ability to freely manipulate the stylus; they also commented on the number of functions that were implemented onto a single button was confusing. We believe that enhancing the stylus's capabilities or using a different 3D input device will give us much improved results. An interesting observation is that subjects used pCubee's physics capability extensively to bring pieces closer to their views.

Surprising to us was the seldom usage of the drag-and-clutch rotation mechanism, which we found to be important during our initial design iterations. Subjects negatively rated the difficulty, intuitiveness and the rotation

widget in the questionnaire. We suspect that the large rotation mechanism was difficult to use because imagining rotation axes is a difficult task [44]. As a result, subjects instead resorted to direct manipulation to perform both translation and rotation. All other interaction stages and helping features, including highlighting and snapping, received neutral to positive scores.