Periphery Vision Menu System: Expanding the Interaction Tools of Head Mounted Displays for Virtual Reality

by

Peter Mitchell

Thesis
Submitted to Flinders University
for the degree of

Doctor of Philosophy

College of Science and Engineering
25/01/2022

Abstract

As technology adapts and becomes ubiquitous in our daily lives, there is a need to examine how our interactions can best control the systems and information presented. Head-mounted displays are an example of interaction technology that has become popular for virtual and augmented reality use in recent years. Their toolkits have a varied approach to interaction depending on the headset and application. Currently, many services for head-mounted displays are game related. As the cost and size of these devices continue to decrease and technological improvements increase, the expectation is that a broader consumer base will start to embrace the technology. Understanding and then developing appropriate interaction techniques for these devices is essential. As part of this research, an investigation was conducted to determine if an approach to interaction could be utilised generically between all headsets, regardless of additional interaction tools. This research presents a Periphery Vision Menus System developed to provide interface interaction exclusively with head gestures. The design of the menus allows for contextual support to many types of applications. By only utilising the orientation sensor's data, it was possible to provide useful interactions within a virtual reality system. While the system emphasises head interaction as a standalone solution, there is scope to combine with additional input techniques to provide further layers of engagement.

Throughout three experiments, participants provided their evaluations of the interface and its interactions. The tasks in the initial experiment focused on forms of simple volume object manipulation. The later experiments, still looking at object manipulation, implemented a tower defence game designed to provide the participant with an engaging and exciting experience. A serious games research methodology was employed to improve the interest in research participation. Results from the experiments indicated that the functionality of the Periphery Vision Menu System was accessible and required little training. The activation of menus and subsequent selection of items became natural and straightforward for the participants. Feedback from participants was positive toward the system between the experiments. A desire to use the technique in the future was expressed by participants as well.

It is hoped the approach will become a tool used in many future applications for head-mounted displays. The technique used in this research is beneficial when using additional input methods is difficult or where there is a desire to hide menus intuitively. This research contributes to human-computer interaction, virtual reality, augmented reality, head-mounted displays and serious games. The focus for testing was on virtual reality, but the techniques presented in this research are transferable to any form of head-mounted display application.

Acknowledgements

Before moving on to the body of the dissertation, it is necessary to acknowledge those who provided ongoing support while developing this work from beginning to end. I want to acknowledge the continued support and feedback provided by my principal supervisor Brett Wilkinson. I would also like to acknowledge the patience and motivation provided by family and friends throughout completion. I want to acknowledge the financial support from the Australian Postgraduate Award from 2014 to 2016 and the Australian Government Research Training Program Scholarship in 2017. Finally, I would like to thank the participants who assisted in the experiments conducted by helping to provide the data necessary to complete this work.

Declaration

I certify that this thesis:

1. does not incorporate without acknowledgment any material previously submitted for a degree or diploma in any university; and

2. to the best of my knowledge and belief, does not contain any material previously published or written by another person except where due reference is made in the text.

Signed: ___________________

Table of Contents

Table of Figures

Table of Tables

Table of Listings

List of Abbreviations

Abbreviation	Meaning
0D	0 Dimensions (Classification of a marker or QR code that is static. (Normand et al., 2012))
2D	2 Dimensions (A grid or something that can be mapped to a grid, e.g. GPS)
2D + θ	2 Dimensions + Orientation
3DOF	3 Degrees of Freedom (typically referring to rotational movement with only yaw, pitch and roll)
6D or 6DOF	6 Dimensions or 6 Degrees of Freedom (6D is used concerning AR (Normand et al., 2012), both represent 3-axis movement for each of directional and rotational (yaw, pitch and roll).)
AR	Augmented Reality
ARET	Augmented Reality Exposure Therapy
ASR	Automatic Speech Recognition
AV	Augmented Virtuality
CAAP	Computer-Aided Assembly Planning
CAD	Computer-Aided Design
FPS	Frames Per Second
GPS	Global Positioning System
GUI	Graphical Use Interface
HMD	Head-Mounted Display
IPD	Interpupillary distance (distance between the centre of eyes)
LOD	Level of Detail
MR	Mixed Reality
NPS	Net Promoter Score (used for SUS)
NUI	Natural User Interface
PVMS	Periphery Vision Menu System
SUS	System Usability Scale
SVM	Support-vector machine (a type of supervised learning model)

TUI	Tangible User Interface
WIMP	Windows, Icons, Menus, Pointers
VR	Virtual Reality
VRET	Virtual Reality Exposure Therapy

1 Introduction

The use of virtual reality (VR) for expressive, engaging experiences is still in its infancy and consideration for effective interaction processes are essential to ensure the success and usability of the technology beyond research laboratory or gaming situations. VR is defined as a computer simulation in 3D space typically presented to the user while they wear a Head-Mounted Display (HMD). When thinking about interactions with VR, it is typical to assume the user will be using their hands to engage with the system, either through handheld controllers or hand tracking technologies. This type of interaction may not be possible in some situations, requiring further investigation of input modalities.

To illustrate this, consider a future scenario involving a surgeon performing a remote procedure through a VR interface. For the work of a surgeon, it is necessary to focus on complex, hand-related tasks, where there is typically a tool, device, or surgical instrument always in the surgeon's hand. With the improvements to technology, it is foreseeable for VR or augmented reality (AR) to be used to assist with the visualisation of tasks for surgery. AR is similar to VR, but it focuses on creating a composite view of virtual elements and the natural world using camera input from suitable devices. Some surgery tasks may be related to visualising overlays of scans from the patient, life sign data, guided steps for completing the surgery, or any other relevant data that may be of use to the surgeon. Typically, a surgeon's hands will be scrubbed in and only directly interact with the medical equipment. Due to the hands being occupied by instruments in this scenario, it could be difficult for the surgeon to modify elements of their virtual environment. How can a user of such a system engage with the virtual elements without controllers or hand tracking? Rather than using their hands, a user could, with simple head movements, provide suitable input into a VR system to support their needs during surgery. The research will cover a foundation that developers could apply to such scenarios as part of this dissertation.

1.1 Research Motivation

Research toward this dissertation began in early 2013 as Oculus Rift and HTC Vive were beginning to inspire consumer popularity in accessible and affordable VR. The original research focused on investigating the types of interaction techniques used for controlling head-mounted displays. Most of the applications released relied on controller inputs that had differences between consoles. With integration into Unity and Unreal Engine 4, game development became more accessible for easy deployment to multiple consoles and HMDs. When investigating this area, it was apparent the existing techniques each have their place for specific activities. Through investigation, head interactions were identified as underutilised as an area of research beyond the action of looking around. All HMDs at the time tracked the rotation of the user's head while wearing the device. Rotation tracking as a consistent feature of HMDs inspired the improvement of techniques for HMD Only type interactions. Investigation and prototyping led to developing a prototype system named the Periphery Vision Menu System (PVMS).

The Periphery Vision Menu System is a technique that uses the orientation of a head-mounted display to provide access to hidden periphery menus. Developers can use the menus to offer functionality through head-mounted only interaction or as a different technique for applications with controllers to provide more options to users. By utilising head-only interaction, users are free to use their hands for other activities or handle situations where it is impossible to use hands. The use of head-only interactions designed in this way meant developers could transfer the technique to any head-mounted display where the device rotation is accessible to an application.

1.2 Research Focus

In the previous section, the motivations around the research focus were explored from a broad view introducing the topic area. When beginning to investigate what area may benefit from additional research, it started with questions more generally such as "Can VR interaction be usable without controllers?". Looking for existing works that investigated this area, much of the material discovered focused more on techniques for incorporating independent controllers that a user would typically control via the hands. However, as the HMD has to be worn by users to engage with the VR environment, there was a consideration that the tracking sensors could provide input data to an application. The HMD devices have already utilised orientation sensors to determine where users are looking based on their head rotations. When considering this across multiple different types of HMDs, including how future iterations may work, it raised an important question "Can the development of an interaction system that utilises just HMD systems be usable across multiple platforms and applications?".

The ability to determine orientation as a rotation within applications could be assumed with these HMDs, so it was decided to pursue this as a research focus. When considering how the proposed system could benefit users, there were other considerations regarding comparing against the existing solutions. Specifically, for example, considering “Can the use of head driven interaction provide the same functionality as a traditional controller-based VR system?”. From reviewing related works, both research papers and technical demonstrations of products, it was determined the design and prototyping of the PVMS would be a significant area to investigate. The background leading to identifying this research gap is most prominently discussed in sections 2.4 and 2.5. Based on this area of focus, a range of research goals was formulated, which aided in defining specific research questions.

1.2.1 Research Questions

Before introducing the research questions, the following goals were developed to help guide the research, product iterations, and define the research questions. These goals headed the primary direction for investigation, informed by the background reading and initial experimentation.

- To investigate how user interfaces can be improved for head-mounted displays—specifically looking at the ease of interaction, tools of interaction, and presentation of interactive responsiveness.
- To develop applications demonstrating prototypes of behaviours for head-mounted interactions with a variety of input sources that improve the ease and usefulness of interaction.
- To collect user feedback from a collective of people who experience using the applications to improve the methods of interaction and interfaces.
- To draw conclusions about the usefulness, usability and other features of the proposed interfaces and interactions based on the user feedback.

The research was initially guided by a primary research question (RQ1), and then as the experimentation was conducted, additional questions were defined to focus the scope (RQ2, RQ3, and RQ4). The research questions are listed below.

- **RQ 1:** How can head rotations, captured through the orientation sensors of a HMD, be used to increase the user interaction capabilities with virtual worlds?
- **RQ 2:** Does using a head-mounted display with head-only interaction for menu navigation provide a similarly useful experience compared to integration with a tool for instant selection?
- **RQ 3:** Can a hidden menu provide a viable and useful tool for interaction with head movement as the mechanism for revealing it?
- **RQ 4:** Does a hidden menu revealed and interacted with using the head provide a useful experience compared to menus appearing at a fixed place in a virtual world?

Each research question is discussed in the summary of results for each experiment and the final discussion chapter (Chapter 8). RQ1 was directly investigated in all three experiments. RQ2 was investigated in the first and third experiments. RQ3 and RQ4 were investigated as part of both the second and third experiments.

1.3 Execution

Three experiments were conducted to meet the goals and evaluate the research questions to provide necessary quantitative data. At the beginning of the research, the plan was to develop a head-mounted AR solution to explore improvements to interface development. An initial review of research focused on the AR domain with significant insight and interface guidance sourced from the investigation. From the initial investigation, the focus moved toward testing and implementation in VR to use available hardware.

The first experiment was conducted in early 2015; details about the first experiment are covered in Chapter 3. This experiment used object manipulation while comparing head-mounted only interaction against a mouse or mobile device input. 18 participants took part in the experiment. Data was collected through a combination of pre- and post-experiment questionnaires, including a System Usability Scale (SUS) for each input method. Other survey questions were used to collect data on how effective or difficult the participants found the system. In addition to the questionnaires, application log data was collected for each individual who participated during the experiment.

The second experiment was conducted in early 2016, which consisted of three different tasks to evaluate the first prototype of the PVMS (Chapter 5). The tasks were designed with a gameoriented approach compared to the first experiment. The main two tasks focused on using the prototype PVMS to customise a tower and then play a tower defence game. A total of 23 participants took part in the experiment. Data was collected from their participation through similar means to the first experiment. This included the pre- and post-experiment questionnaires along with data collected within the application. Data was collected about how participants found the experience of using the PVMS.

The third experiment was conducted in late 2016 and iterated on the prototype interactions in the second experiment to capture additional data for evaluation of the system. Chapter 6 details the process and results of this experiment. The tasks in the third experiment involved object matching using a similar style to the first experiment and an improved version of the tower defence game from the second experiment. 26 participants provided feedback used to evaluate the experiment outcomes. Data collection was similar to the other experiments, with some minor changes to questions and improvements to the data collected from within the application.

From the three experiments, a significant amount of data was collected from the participants and used to perform an analysis to form the combined results in Chapter 7 and a discussion in Chapter 8.

1.4 Contributions

The primary contribution from this dissertation is the prototype Periphery Vision Menu System (PVMS). The work demonstrates ways this interaction technique can exist on its own for head-mounted only interaction and alongside other interaction input controllers to provide further utility. Using a technique that can exist universally between any type of HMD is an important user interaction approach allowing users to transition between platforms and applications with ease. The design for this system is explained throughout the dissertation. Providing example cases for both theoretical case studies and directly through experimentation in the second and third experiments. With a supporting analysis and comparative discussion of results collected from participants across three separate experiments. Through the various experiments, there is a demonstration that the technique can improve many VR applications through its inclusion. The success of the implementation and testing contributed to the validation of the experiment approach through serious games as a testing platform for research. Through a serious game platform, participants' motivation to aid in research seemed to increase, and it helped to provide context and understanding for what participants should be doing. The game-based approach allowed for exploration of the PVMS's complexity within a familiar and engaging environment for participants. To be clear, the serious game approach used in this was not to develop a serious game targeted toward health or other serious areas of investigation. The process used games for the serious purpose of recruitment, encouraging additional participants and advertising using game trailers; therefore, using games and game technologies for a non-entertainment purpose.

The first experiment provided a baseline for comparison regarding investigating the use of HMD Only and the addition of a separate piece of hardware for selection. This baseline was compared against the second and third experiments, where the focus was more heavily on testing the PVMS against different factors. Results from the first experiment contributed data on the general usability experience in VR and the usability concerns of those who participated.

The second experiment evaluated the first publicly tested version of the PVMS. Generating data to validate the viability of the proposed system and collected data to have available for future improvement enabled a contribution to the perception of usability testing and development of VR focused interaction processes. The second experiment also validated the serious games approach used for research testing as a viable means for participant recruitment and made tasks relatable through known types of game interactions.

The third experiment deployed an improved version of the PVMS taking the development and results of the second experiment and making small but important improvements to the system. Notably, the addition of the two-step widget to reduce the impact of accidental error states contributed to the understanding and design of effective, usable and accurate interaction sequences. In addition to the iteration and changes demonstrated as part of the third experiment also provided additional data that could be used to continue further fine-tuning of the system. While also validating the approach to both testing through serious games and demonstrating the viability of the proposed system.

Preliminary data from the second experiment was presented at the 28th Australian Conference on Human-Computer Interaction (OzCHI2016) (Mitchell and Wilkinson, 2016). The paper established the fundamental concepts of the system and concisely encapsulated participant feedback. The data collected from the other two experiments and subsequent discussion from this dissertation are being consolidated for future papers.

1.5 Document Structure

The document has been structured to provide an overview of the PVMS in order of design and development. Chapter 2 Background begins by providing a discussion of existing research in both VR and AR, as they both have applications with this research. Investigation continues into display technologies, user interfaces, interactive controllers, serious games, and a selection of examples looking at the breadth of uses for the described technologies. The review of this work identified the gap in the research and situated the work presented in this dissertation, its contribution and where it opens future possibilities of application and continued investigation.

Chapter 3 First Experiment provides a detailed breakdown of the first experiment. In this chapter, a significant discussion is devoted to the methodology. The intention was to present suitable detail to allow recreation of experiment apparatus and execution for comparative studies and future work. The results and a summary of the results have been included in the chapter. Supporting content for this chapter is included in Appendix A, detailing code and project structures to enable repetition.

Chapter 4 Periphery Vision Menu System details the technology overview of the interaction system. This chapter examines the specific details of the design, construction, and significance of the system. It provides details to clearly define the system and the approach to implementation for experimentation during the second and third experiments.

Chapter 5 Second Experiment and Chapter 6 Third Experiment follow. In these chapters, the prototype PVMS is exercised through experiments. Each experiment investigated and iterated on the design to evaluate and establish the viability of the system. These chapters are presented in a similar structure to Chapter 3, focussing on methodology, followed by the presentation of results for each specific experiment. Additional materials for each of the experiments can be found in Appendix B (second experiment) and Appendix C (third experiment).

Chapter 7 Combined Results examines the combination of results and the knowledge gained from additional experiment data analysis. Results from all three experiments are discussed in contrast to explain and justify findings from the research.

Chapter 8 Discussion presents a review of the research questions to establish how the results demonstrate the success of the PVMS. Furthermore, the chapter compares against more recent evolutions in VR systems, contrasting how the PVMS compares against examples of other available interaction technologies. The chapter additionally provides a discussion to present a framework for how developers or other researchers could take the PVMS and iterate or integrate it.

The final chapter (Chapter 9) presents a summary to conclude the dissertation and identify contributions. The chapter discusses areas of limitation in research approach, feature expansion, technology improvement, and future work to direct researchers and developers toward ideas for potential directions of further investigation.

Appendices at the end of the document provide additional detail on the construction of the software, the experiment tools and surveys utilised, the data collected throughout the experiments and the final appendix shows where to access the original experiment code for all three experiments on GitHub.

2 Background

This chapter will review existing literature, providing a background for the dissertation and identifying the research gap. The review will focus on topics used for the basis of the research and those used as inspiration. The overall theme will be a discussion on the state of VR and AR. The applications of the research presented in this dissertation overlap both fields. The prototyping and experimentation were conducted purely with VR, but the techniques are seen as interchangeable. The specifics of the PVMS are discussed further in Chapter 4.

The key topics discussed in this chapter will begin with an overview of VR and AR, followed by a discussion on display technologies and their relationship with VR and AR. Then a look at the evolution of interfaces and how they have advanced from command lines to natural interfaces with newer interaction technologies. The discussion leads to looking at the interaction technologies and the types of controllers, providing a variety of ways to immerse the users in new ways. Games are an area providing a wider commercial reason for the development of VR and AR. As part of the experiments conducted during this research, games have been explored to provide research data and provide an interesting element for participant recruitment. For this reason, a section discussing serious games and how games can be used beyond entertainment has been included. Finally, the last sections of the background are dedicated to examples of the diverse uses for VR and AR.

2.1 Virtual and Augmented Reality an Overview

VR can be described as a form of escapism. Current technology allows access to VR primarily using HMDs, with interaction provided typically through sensor-driven hand-held controllers. When defining VR, it is typically defined by the experience you would expect to encounter. Sherman and Craig defined four key elements of the VR experience (Sherman and Craig, 2002).

1) Virtual World: The representation or design of what is to be experienced by the user.
2) Immersion: This deals with the users' perception of the virtual world. The feeling that you have become part of the virtual world. For example, becoming a wizard who can cast spells.

3) Sensory Feedback: Sensory Feedback creates the feeling that you are impacting how you are in the world or the world is causing an impact on you. This feedback, for example, could be simply the ability to orient your head or move yourself within the world, ideally in a way that feels natural.
4) Interactivity: The user's actions have an impact on the world. In an extension of the world providing feedback, the interactivity element will generally offer the means of changing the world's state to progress the user further toward some goal.

AR differs from VR in the way it is presented to the user. VR focuses on providing a total immersion within a virtual world, but AR will, in most cases, attempt to use the physical world as a basis for adding virtual content instead. AR takes a perception of the world that can be physical or audible and alters it before it is shown to a user. The prevalence of increasingly powerful mobile computing technologies capable of video and audio capture allows for more widespread AR use. As defined by Azuma, AR combines real and virtual worlds in interactive real-time and is registered in 3D (Azuma et al., 1997; Azuma, 2004). The virtual world that is brought in through AR is intended to supplement the information from the real world. It is important that this information is spatially related to the position and orientation of the user and that of the content being viewed.

In an AR-related gaming survey, Thomas identified three features of AR that support computer games compared to VR (Thomas 2012). The three features identified were:

1) The field of view is typically limited, therefore not requiring an entire virtual world to be populated for an application to provide necessary visual information. AR does not set limits on how much of the world can be populated with virtual elements. The lack of a limit allows environments to be overlayed entirely with AR elements, thus forming a synthetic world. The completeness of how much of the environment is shown as AR is highly dependent on the purpose and scope of the applications developed.
2) AR still allows view and interaction with the physical world. The interaction with the physical world enables users to still engage with or avoid physical features and people. It would be hazardous to expect a user to navigate blindly in an entirely virtual world where the world is not related to the physical world. The interaction with the environment and people provides extension beyond simplistic isolation and can provide a more enriching engagement.

3) As identified with the previous point, physically moving within an open environment can be hazardous for an entirely virtual environment. In VR applications, physical movement expectations are limited to minimise the risk of blindly walking around. Physically moving within an AR environment could be manifested through moving outside or simply moving inside. As the user moves through the environment, the AR interface will update in real-time to render the latest view of the world. The real-time view can also include interaction with objects, such as lifting an object and then perhaps even identifying the object to play a more important role in the virtual environment.

Milgram et al. (1995) presented the continuum between VR and AR as a spectrum that moved from experiencing the real environment to a completely virtual environment. AR can fall into a broad domain of different deviations commonly referred to as Mixed Reality (MR) that encompasses incorporating AR into the visualisation of a real environment, all the way through to virtual environments that exist with an element of Augmented Virtuality (AV). These definitions for VR and AR can be applied in varying levels to the different applications and systems discussed in this chapter. The definitions more recently, with the newer generations of HMDs, are slowly becoming blurred together. The main difference dividing the two will continue to be the reliance on seeing the real world as part of an AR experience. As Milgram et al. (1995) presented, there can be a broad continuum in defining how closely a reality can be associated with the real environment or virtual environment. Considering the overlaps is important for the systems this research presents because the approach is interchangeable between the two technologies, as will be discussed in future chapters.

2.2 Display Technologies

The technologies that will be given an overview in this section are used with VR or AR. There will be four different types, covering a broad range of applications in how they can be used. The types of displays that will be briefly described are Head-Up Displays (HUDs), HMDs, Tabletop/Projector Displays, Holograph Displays, and Mobile/Tablet Displays. Each falls into a niche for how they are used, making them desirable in varying situations. For this research, the focus has been on using HMDs; therefore, the focus will be on those, but it is important to understand the other types of displays, particularly those that impact VR and AR.

2.2.1 Head-Up Displays

Head-Up Displays are typically fixed in place, as is the case with windshields on cars or aircraft. In these cases, navigation information can be placed in front of the user to make decisions with more spatial awareness. A prime example of this type of system is the Taxiway Navigation and Situation Awareness (T-NASA) (Foyle et al., 2005). In this type of system, the information about aircraft navigation, including the position and direction of runways, is overlayed on a display by projecting onto the windshield to give pilots up-to-date information. More recently, this type of display can be found in some cars to provide information to the driver without looking away from the road. Figure 2.1 shows an example (Navdy, 2017).

Figure removed due to
copyright restriction.
See link for original:
[Link]

Figure 2.1: Navdy Head-Up Display[1]

2.2.2 Tabletop/Projector Displays

Projection onto a surface such as a tabletop, as seen in the Augmented Coliseum (Kojima et al., 2006), provides the augmentation using real-world objects and an overlay with additional information. In this case, explosions and other visual effects related to the interactions of small robot toys as seen in Figure 2.2 (over, left). Displays that use a projector can be used with a wider variety of surfaces. Projectors do not need to focus top-down; projectors can be used for projection onto a wall or arbitrary surfaces, i.e. projection mapping/shader lamps.

[1] Travel Skills 2016, "New device offers drivers a heads-up hands-free display", URL: https://travelskills.com/2016/11/04/new-device-offers-drivers-heads-display/, Last accessed 11/12/2021.

For example, this could be used to improve the scale of the view of CamBall developed by Woodward et al. (Woodward et al., 2004). CamBall is an AR table tennis game. Also related to this are see-through head-up displays that allow interaction with the surface. An example of this is the work on interaction with these displays by Olwal (2008). In Olwal's example, the content projection was onto a transparent display, providing an experience where the users could interact and still see through to elements behind the display.

Figure removed due to copyright restriction.
See Kojima et al. 2006 for original.

Figure removed due to copyright restriction.
See link in footnote for original.

Figure 2.2 (Left) Augmented Coliseum (Kojima et al., 2006), (Right) Microsoft PixelSense[2]

Projectors are largely very generic in the commercial market, but Microsoft Surface provided one of the earlier examples of large interactive screens providing tabletop computing. The Microsoft Surface was more recently renamed the Microsoft PixelSense, differentiating it from smaller tablets using the previous Microsoft Surface name. The original tabletop was released in 2008 with a 30-inch display with 1024x768 resolution (Bowden, 2017). An example of the display can be seen in Figure 2.2 (right). With a starting price of $10,000 USD at the time, the device was not intended to be a product general consumers would obtain. With continued development into better screen technologies, it is expected that table size interaction surfaces will become more common.

2.2.3 Mobile/Tablet Displays

Mobile and tablet type displays are significant platforms for this type of blended experience, given how many individuals carry a smartphone or similar device around. Mobile screens or their components have been used in some HMDs. Specifically, for the Google Cardboard and

[2] Microsoft 2016, "Multitouch Tabletop Microsoft PixelSense Surface 1 Microsoft Surface Developer Hardware Unit – Black JUI-00041", URL: https://dresden-technologieportal.de/en/equipment/view/id/1375, Last accessed 11/12/2021.

Samsung Gear VR, as will be discussed further in section 2.2.4. Mobile devices typically utilise sensors to detect the location, orientation, voice, camera, and a range of other data about the environment that the user occupies. A discussion of mobile phone sensor technologies will be presented in section 2.5.1.

For AR, mobile devices provide a tool to freely observe augmented elements depending on the scope of the application. The free observation could be observing elements placed with fiducial markers (or another more natural marker), using player position within the world for games like Ingress in Figure 2.3 (Niantic, 2012), or Pokémon Go in Figure 2.4 (Niantic, 2016). For content being observed with a mobile device in AR or within a virtual world, the device itself is not restricted to being fixed in one place. The lack of restriction allows freedom of movement and also fits well with pervasive-type applications.

Figure removed due to copyright restriction. See footnote for Ingress content.

Figure 2.3: Ingress Example[3]

Figure removed due to copyright restriction. See footnote for Pokemon Go content.

Figure 2.4: Pokémon GO Example[4]

In the research presented in this dissertation, mobile devices have been used only as an interaction tool while not visible to the user, as they were wearing HMDs. Mobiles are particularly relevant due to their capacity for use as the HMD through combination with a holder (as is the case with Google Cardboard). The continued iterations of smartphone models make them constantly more desirable. Finding additional ways to use mobiles that allow users to reuse their personal hardware for more purposes benefits the user from increased utility and the developer from access to tooling options for more unique interactions.

[3] Niantic 2012, "Ingress Prime", URL: https://ingress.com/, Last accessed 11/12/2021.
[4] Niantic 2014, "Pokémon GO", URL: https://pokemongolive.com/en/, Last accessed 11/12/2021.

The initial direction of research for this dissertation did consider using mobile devices to provide a more complex menu interaction tool as part of the solution. The first experiment (found in Chapter 3) originally would have had a more comprehensive mobile input solution than the tap-to-select interaction. Mobiles sensors (as further discussed in section 2.5.1) provide many options that would have allowed for using a smartphone in a similar way to wand-type controllers. One primary advantage of a smartphone compared to the wand-type controllers is the addition of a touch screen that could be used for menu selection or other input. From the initial survey of the research areas and as work began on developing prototype solutions, it was evident that focusing on using the HMD as a sole interaction tool was a less explored area of investigation. This avenue of investigation led to using mobile as a simple interactive tool for initial comparison in the first experiment to help validate HMD Only type input. The following section will discuss HMDs concerning a history of devices that had emerged when the research was conducted and consider how HMDs evolution has improved prospects for AR and VR development.

2.2.4 Head-Mounted Displays

HMDs have, in recent years, become more viable as a consumer product due to the continued improvements to the hardware. A cornerstone in this improvement is graphics cards that iterate toward representing virtual environments as more immersive and engaging. While at the same time, display technologies improve toward higher resolutions, further increasing the amount of visual detail possible. Combining these technological advances in smaller form factors has led to various releases to the commercial market. With the advent of the Oculus Rift, a rapidly increasing number of competitor models have begun to appear for use in the consumer VR and AR market.

Several notable HMDs were conceptualised, prototyped, or commercialised preceding the Oculus Rift. The Sensorama (Heilig, 1962), developed in 1956 by Morton Heilig, led to the development of the first VR headset: the Telesphere Mask, in 1960 (Heilig 1960). Other products of note included: Headsight in 1961 (Comeau and Bryan, 1961), the Sword of Damocles in 1968 (Sutherland, 1968), the Virtual Environment Workstation Project at NASA in 1985 (Fisher et al., 1986), the unreleased Sega virtual headset in 1993 (Machkeovech, 2020), and the Nintendo Virtual Boy in 1995 (Boyer, 2009). The Nintendo Virtual Boy only supported red/black colours and lacked software, among other difficulties. This led to the production being halted in 1996 and limited interest outside of research for HMDs until 2011, when the Oculus Rift Kickstarter was released (Oculus, 2012). Early HMDs struggled with the hardware of their times to meet users' expectations. The evolution of graphics and computing hardware has enabled immersive experiences with sensory feedback and interactivity. These are defined by Sherman and Craig (2002) as key factors for VR.

Virtual Reality HMDs

HMDs are distinct in how they sit on the front of users' faces to consume their entire field of view. Control over the field of view enables immersion through filling the visual sensory input. Further immersion through other senses and inputs can be made possible with additional devices, as discussed in 2.5.4. With the growing variety of HMDs available for VR and the newer emerging devices for AR, it is beneficial to list these devices. The following list identifies some significant consumer devices with brief details about their capabilities.

Figure removed due to
copyright restriction.
See link for original.
[Link]

Figure 2.5: Google Cardboard[5]

Figure removed due to
copyright restriction.
See link for original:
[Link]

Figure 2.6: Samsung Gear VR[6]

- **Google Cardboard**: The Google Cardboard in Figure 2.5 (Google, 2014) combined an inexpensive cardboard shell with a smartphone to provide one of the cheapest VR experiences. The quality of the experience was very dependent on the specific smartphone used. This smartphone variation made it more difficult to tailor application performance to the end-user.
- **Samsung Gear VR**: This was another variation of using smartphones as the screen for HMD as seen in Figure 2.6. This headset was designed to work with Samsung smartphones from 2015 onward. It was intended to have a 2560x1440 resolution (1280x1440 per eye) based on phone resolution with a refresh rate of 60 Hz and a field of view of 96°. The tracking supported everything the mobile device contained with 3

[5] Google 2014, “Google Cardboard”, URL: https://vr.google.com/cardboard/, Last accessed 11/12/2021.

[6] Samsung 2015, “Samsung Gear VR”, URL: https://www.samsung.com/global/galaxy/gear-vr/, Last accessed 11/12/2021.

degrees of freedom (3DOF) to support orientation (Samsung, 2015). One of the early advantages for both the Samsung Gear VR and Google Cardboard was that the display contained the hardware responsible for processing application execution by using a smartphone as the visual display. The other HMDs described here required a separate computer and existed as an alternative screen instead of an all in one type solution. Although there are limitations on smartphone processing capabilities compared to a dedicated computer, the devices available now can provide a visually immersive experience with appropriate input and sensory feedback.

Figure removed due to copyright restriction.
See link for original:
[Link]

Figure 2.7: Oculus DK 1[7]

Figure removed due to copyright restriction.
See link for original:
[Link]

Figure 2.8: Oculus DK 2[8]

Figure removed due to copyright restriction.
See link for original:
[Link]

Figure 2.9: Oculus Consumer Model[8]

- **Oculus Rift**: The first version of the Oculus Rift came before other commercial HMDs (such as HTC Vive or Playstation VR) when it was released on Kickstarter in 2012 (Oculus, 2012). As part of the research presented in this dissertation, the Oculus Rift Dev Kit 1 and Oculus Rift Dev Kit 2 were used for running the experiments with participants. For this reason, it is worth comparing how the device changed between versions. As a baseline, all three versions (including the initial consumer model) supported at least 3DOF with a gyroscope, accelerometer and magnetometer (Rift Info, 2016; Oculus, 2018).

  The first model available via Kickstarter was the Oculus Rift Dev Kit 1, as seen in Figure 2.7. The display used a 7-inch LCD with 1280x800 resolution (640x800 per eye) with a refresh rate of 60 Hz and 110° field of view. The second iteration, Oculus Rift Dev Kit 2, as seen in Figure 2.8, used a smaller screen size (5.7-inc OLED) but had a higher resolution. The screen supported 1920x1080 resolution (960x1080 per eye) with a selectable refresh rate of 60 Hz, 72 Hz, or 75Hz. Due to the smaller screen size, the

[7] Oculus 2012, “Kickstarter: Oculus Rift: Step Into the Game”, URL: https://www.kickstarter.com/projects/1523379957/oculus-rift-step-into-the-game, Last accessed 11/12/2021.
[8] Oculus 2018, “Oculus Rift”, URL: https://www.oculus.com/, Last accessed 11/12/2021.

device only provided a 100° field of view. The device natively supported 3DOF but could use 6 degrees of freedom (6DOF) by attaching an included camera. The consumer version further iterated the design, as seen in Figure 2.9. Dual OLED panels provided a 2160x1200 resolution (1080x1200 per eye) with an increased refresh rate to 90 Hz and 110° field of view. The consumer model device provided 6DOF and included in-built headphones for a more immersive experience. After the initial release of the consumer model, controllers became available separately with similar capabilities to those of the HTC Vive's controllers. The iteration in development moved toward high refresh rates and higher resolutions with support for 6DOF to provide a robust platform for software iteration in VR.

Figure removed due to
copyright restriction.
See link for original:
[Link]

Figure 2.10: HTC Vive[9]

Figure removed due to
copyright restriction.
See link for original:
[Link]

Figure 2.11: PlayStation VR[10]

- **HTC Vive**: The HTC Vive, as seen in Figure 2.10, provided a direct competitor to the Oculus Rift models with its release in April 2016 (compared to March 2016 for the Oculus Rift first consumer version). The base HTC Vive model provided an identical resolution (2160x1200), refresh rate (90Hz) and field of view (110°) to the Oculus Rift consumer version. The HTC Vive Pro increased the resolution to 2880x1600 (1440x1600 per eye). The significant difference for the HTC Vive was that it used a laser position sensor to pair with base stations for detecting 6DOF of both the headset and the included pair of hand-held controllers (Vive, 2018).
- **Playstation VR**: As seen in Figure 2.11, the PlayStation VR was developed by Sony for the Playstation 4. It contained a 5.7-inch OLED with 1920x1080 resolution (960x1080 per eye), support for 90 Hz and 120 Hz refresh rates, a 100° field of view and 6DOF (Sony, 2018). The primary advantage of the Playstation VR was that it targeted

[9] Vive 2018, "HTC Vive", URL: https://www.vive.com/au/product/, Last accessed 11/12/2021.

[10] Sony 2018, "Playstation VR", URL: https://www.playstation.com/en-au/explore/playstation-vr/, Last accessed 11/12/2021.

hardware that was the same for all users. The other HMDs either could have variable support from the smartphones used (for the Google Cardboard and Gear VR) or from the ability of a separate computer to run applications smoothly (Oculus Rift and HTC Vive).

The HMDs listed above are all primarily designed for use with VR. They are similar in their general properties with some variance in the screen quality, resolution, refresh rate and field of view. HMDs with 3DOF typically only support movement in the form of rotation. In contrast, those in most newer headsets support 6DOF, allowing translation and rotation. The general trend for HMDs has seen a shift to higher resolutions and refresh rates.

The trend in increased quality comes from general improvements to displays over time to provide a richer experience and to deal with cases where users experience motion sickness. Motion sickness from VR in a HMD environment comes from a sensory mismatch where the user expects something, but their visuals do not match expectations (Dennison and D'Zmura, 2018). This mismatch could be experienced in cases where the display does not keep up with the user's movements (either from refresh rate or the application not being performant) or from virtual forces applied to the user that moves their view in unexpected ways. The ergonomics associated with motion sickness was a key factor in the display of content and the environment where the users for the experiments presented in this thesis were conducted. Refresh rate has been shown to be a factor but equally important is the physical environment and user pose. Merhi et al. (2007) demonstrated a lower rate of motion sickness while using HMDs when sitting compared with standing.

Another topic of ergonomics to consider for HMDs is the fatigue that users may feel. Guo et al. (2017) investigated the effects of visual fatigue in virtual environments by comparing it against a smartphone. Their study found that eye strain, general discomfort, focus difficulty, and headache factors had statistically significant differences favouring the HMD over the smartphone. Zhang et al. (2020) evaluated fatigue for longer-form content using a 40-minute video comparing a HMD against an iPad with measurements taken every 10 minutes. Both samples demonstrated some visual fatigue over time. The HMD appeared to cause greater visual fatigue than the iPad, but HMD fatigue was lower and less serious than expected.

Augmented Reality HMDs

AR headsets differ from VR headsets as they seek to overlay the virtual world onto a representation of the real world. The user's view of the world can be seen as either a video see-through or optical see-through experience. A video feed is captured for video see-through, and AR elements are incorporated before rendering the video to the user. Optical see-through uses information from the video feed or other sensors to situate the augmented objects in the real world and then subsequently render them on a semi-transparent pane. This pane still provides a view of the physical world with an overlay of virtual objects. Video see-through can reliably present objects consistently by directly rendering the AR objects onto the video feed. The main difficulty with video see-through is the reliance on image quality and handling of responsiveness while moving through the physical world. For video see-through AR on a mobile device, it does not necessarily matter if there is a momentary pause in output as the handheld mobile device does not completely consume the user's view. However, when wearing a HMD, this disruption in viewing the real world could lead to hazards. The disconnect between the physical world and augmented objects in optical see-through can make users feel the objects are artificial (Medeiros et al., 2016). The following list explores some examples of HMDs for AR that were developed at a similar time to the previously listed VR HMDs. Other HMDs developed prior to the following for research can be found in the trend discussion by Kiyokawa (Kiyokawa, 2012).

Figure removed due to copyright restriction.
See link for original:
[Link]

Figure 2.12: Google Glass[11]

Figure removed due to copyright restriction.
See link for original:
[Link]

Figure 2.13: Microsoft HoloLens[12]

Figure removed due to copyright restriction.
See link for original:
[Link]

Figure 2.14: Magic Leap One[13]

- **Google Glass**: Seen in Figure 2.12. The Google Glass was released in 2013 as a technology you could take everywhere. The device was a part of the glasses and included a mini screen always visible in one corner. With a 640x360 resolution display, a camera, touchpad, and voice control, the device provided a natural interface for

[11] Google 2013, "Google Glass", URL: https://www.google.com/glass/start/, Last accessed 11/12/2021.

[12] Microsoft 2016, "Microsoft HoloLens", URL: https://www.microsoft.com/en-us/hololens, Last accessed 11/12/2021.

[13] Magic Leap 2018, "Magic Leap One", URL: https://www.magicleap.com/magic-leap-one, Last accessed 11/12/2021.

interaction (Google, 2013). The small screen and interaction methods made it more of a utility device, varying its usefulness between individuals based on their personal use cases.

- **Microsoft HoloLens**: Seen in Figure 2.13. The HoloLens provided a mixed reality experience with sensors to allow users to engage with an AR experience. These sensors included an accelerometer, a gyroscope, a magnetometer, a depth-sensing camera, a photographic camera, microphones, and ambient light sensors (Microsoft, 2016).
- **Magic Leap One**: Seen in Figure 2.14. The Magic Leap One was like the Microsoft HoloLens using mixed reality. This HMD had not been released yet at the time of writing, but some preview models have been sent to reviewers. One significant advantage this HMD had over the others is the smaller size with the focus on representing them as goggles (Magic Leap, 2018).

Concerning applicability to this research for the presented AR HMDs, the Google Glass does not offer a large enough display to work with for the proposed PVMS. The Microsoft HoloLens and Magic Leap One both offer 6DOF and, therefore, could benefit from a hidden menu approach with a gesture trigger. For both VR and AR, the presentation of information is within a constrained space, limited by the number of pixels seen.

Many examples of display technologies have been explored, demonstrating how the area of display technologies are evolving to meet the requirements of future applications in many different domains. The Oculus Rift Dev Kit 1 and Dev Kit 2 were used to prototype and test the proposed techniques for this research. The development and data collection preceded many of these other newer HMDs. The techniques discussed as part of this dissertation apply to HMDs broadly for most, if not all, devices identified. In the following section, examples from many domains are explored and discussed, including their influence on the research.

2.3 Example Uses for Virtual and Augmented Reality

This section will focus on examples from a wide variety of domains looking at how VR and AR have been used. Many of these examples have been used as inspiration for the techniques developed as part of this dissertation. The research presented here also provides interesting elements to draw from in demonstrating the breadth of research conducted in these areas. The list of topic areas covered in this section can be seen below.

- Health and Assistive Technologies
- Books and Education
- Drawing/Artwork
- Tourists
- Collaborative Computing
- Assisted Construction
- Pervasive Applications
- Mixed Reality Gaming Research
- Commercial Gaming

With the devices available and the continued improvements to technology, new ideas will continue to emerge. In the coming years, the fields of VR and AR will continue to have more affordable and powerful hardware. The number of developers with access to these technologies will increase opportunities for widespread, consumer-focused applications.

2.3.1 Health and Assistive Technologies

Combining the medium of AR or VR with ideas that improve an individual's wellbeing or give assistance to medical staff in dealing with patients is an important area of research. Many VR and AR applications could be considered entertainment, but in many cases, they are being repurposed to go beyond entertainment and bring something useful to those who use them. Assistive technologies are applications that can be used to provide support for those who may have a condition that prevents them from engaging in typical behaviour. Support for conditions could assist a speech or hearing impairment or a cognitive issue, such as dementia. Providing tools that can aid these people can improve their quality of life if the systems they are given are not so complicated that it becomes a burden for them to use. Headings separate each health and assistive section to distinguish the core topics with a summary at the end to condense the presented information.

Dementia Treatment with VR

For example, in the case of dementia, there have been studies looking at how those affected can be assisted by using VR. Flynn et al. (2004) found that the experiences within the virtual environment were beneficial for the persons with dementia who participated in their study. The experience for participants involved navigation of a virtual park with trees, benches, fences and other objects you would expect to encounter. The participants navigated the environment using a joystick input. With only six participants, the small sample size meant they could not generalise their findings for all persons with dementia. The authors suggested that for persons with dementia and their carers to benefit from developments in VR, more development and evaluation would be necessary. Namely, with a focus on cognitive assessment, cognitive rehabilitation, and therapeutic activity.

Another study related to dementia was performed by Hodge et al. (2018); it included seven participants and looked at the use of VR environments for people with dementia. The discussion in the paper brought out three key themes around the experiences of the participants. Those included feeling foolish and free, seeking to share new worlds, and blending the old with the new. From the themes discussed, future directions to improve VR for dementia focused on careful physical design, making room for sharing, utilising all senses, personalisation, and positioning the person with dementia as an active participant. A final dementia-related example was performed by Moyle et al. (2018) as a small study looking at the use of a VR forest to enrich the lives of those with dementia. They found the experience positively impacted the pleasure and alertness of those with dementia who participated. The experience was reported to be most appropriate for people living with mid-stage dementia.

Exercise Gaming

Exergaming is a type of gaming designed by Görgü et al. (Görgü et al. 2010, 2012). They combined the words “Exercise” and “Gaming” to form this word. In their work, they outlined some principles for combining exercise with gaming. Requiring stages of “warm-up” and “cool down” as part of applications developed with this methodology, for example. Existing applications like Dance Dance Revolution, Wii Fit, Microsoft Kinect, and Playstation Move were cited as examples of existing exergaming relying on indoor virtual environments. Freegaming was designed to be mobile (e.g., outdoors), augmented (using AR to provide immersion), collaborative (providing multiplayer support), and adaptive (changing to meet the current environmental conditions).

Stroke Rehabilitation with VR

Rehabilitation is an important area where many different scenarios vary depending on exercise requirements. One place that has been investigated, for example, with VR, is dealing with the effects of strokes. Laver et al. (2015) had 37 randomised trials with 1019 participants where VR was used to assist in improving upper limb functions after a stroke. The VR solution was found to be significantly more effective compared to that of the conventional therapy solutions. Another separate study related to the rehabilitation of those who had suffered a stroke was conducted by Gamito et al. (2014). In Gamito et al.'s study, a sample of 99 stroke patients was put through a VR-based rehabilitation program. The program included exercises to train attention and memory abilities. These activities were found to be beneficial for the participants.

Food Intake Control with AR

Controlling the amount of food consumed through the illusion with AR was researched by Narumi et al. (Narumi et al. 2012). In their work, they identified many factors that modify how much food someone will eat. By adjusting how the food appears, they hoped to cause a reduction in food intake and improve the nutritional value of the food consumed. Their prototype captured images of their hand and the food. The hand and food would then be separated and rescaled. They tested with shrinking, leaving unchanged, and enlarging the appearance of food through the AR interface. The system in the small trial did provide positive results. The study indicated that education would play a significant role in how effective it could be in the longer term for users to make intelligent decisions.

Dentistry AR Tooth Analysis

Dentists performing cosmetic dentistry traditionally had to interpret shades manually. Qiao et al. designed a system that used AR to compare teeth shades against those expected (Qiao et al., 2011). There were two parts to their system. The first part of the system analysed and presented shades related to the patient's teeth. The second was to render a photo-realistic virtual tooth as part of an overlay of the original tooth. The system could then be used to evaluate and test the quality with other teeth. They only had a small study with mostly positive feedback regarding the benefits of the system.

Exposure Therapy with VR and AR

VR exposure therapy (VRET) has been explored to assist with the treatment of anxiety. Linder et al. (2017) presented a suggested list of factors to help those developing/evaluating these technologies that should be considered. The suggested factors included the use of gaming to match with existing technologies to improve their accessibility, making use of unique features of VR to add additional stimuli, exploring the VRET self-help format for providing evidence-based solutions, making use of observational fear extinction learning, making use of inhibitory learning exposure, exploring the use of controllers for VR including hand motion sensors and eye-tracking, and adopt the same intervention evaluation standards used for other behavioural interventions. Maples-Keller et al. (2017) discusses the use of VR to treat a wide variety of different psychological related issues, including anxiety, phobias, social anxiety disorder, post-traumatic stress disorder, panic disorder and agoraphobia, obsessive-compulsive disorder, schizophrenia, pain management, addiction, autism, and other forms of treatment.

Similar to the use of VRET, AR exposure therapy (ARET) is a form of AR that can assist in mental health (Wrzesien et al., 2011). In the research of Wrzesien et al., the ARET system was used to provide interactive exposure by a therapist to a client. An example was cockroach phobia; the therapist would control how the cockroaches appeared to the client through various factors, including size, quantity, and behaviour. The client would be observed while viewing the scene through an AR HMD to gauge how they reacted to the exposure. The observation allowed for a controlled environment to monitor clients' reactions to different forms of exposure. The system was evaluated based on anxiety, avoidance, belief, and a behavioural avoidance test. In the few cases tested, all these areas had positive results for improvement of the clients from before they started the session to after the session was over.

Speech Recognition with AR for the Hard of Hearing

An example of a useful assistive technology is the use of AR and automatic speech recognition (ASR) for assisting people who are deaf or hard of hearing (Mirzaei et al., 2012). Through audio-visual speech recognition, Mirzaei et al. showed that their system was preferred over text and sign language communication. The system would use advanced facial expression and speech processing technology to determine what the person speaking was saying. Then this would be displayed as text on the screen to the user. People who were deaf voted that sign language (80%) as a form of communication was nearly as interesting to them as the AR+ASR system (90%). The similarity in preference was perhaps because those who were deaf had already spent a reasonable amount of time learning sign language, so it was familiar to them.

In this section, there has been a selection of different health and assistive technologies reviewed, representing a sample of how VR and AR can be used to benefit patients. Dementia treatment has been investigated through providing experiences in VR parks (Flynn et al., 2004), forests (Moyle et al., 2018), and foolish freeing experiences that can be shared (Hodge et al., 2018). Exercise gaming (or Exergaming) can help provide structured exercise with warm-up and cool-down stages as both VR and AR experiences (Görgü et al. 2010, 2012). Rehabilitation of stroke patients using VR was shown by Laver et al. (2015) to improve upper limb functions post-stroke. Gamito et al.'s (2014) VR-based stroke rehabilitation application helped improve patient outcomes with exercise and memory tasks. Narumi et al. (2012) investigated the use of AR to create illusions that helped reduce food intake by shrinking or enlarging the visual appearance of food. As an example use of image analysis and AR, Qiao et al. (2011) demonstrated an application for dentistry that compared shades of teeth to test for quality. Therapy through exposure to fears or other stimuli were investigated in VR (Linder et al., 2017; Maples-Keller et al., 2017) and AR (Wrzesien et al., 2011) with demonstrated improvement for clients. Finally, speech recognition combined with AR was used by Mirzaei et al. (2012) to assist the deaf and hard of hearing with demonstrated improvement.

Any new developments in technology are worth considering in terms of how they may benefit people dealing with different physical or psychological needs. The user base should also be considered for an application to determine if accessibility accommodations need to be made, such as colour-blind modes or other commonly utilised solutions. The following section provides examples of AR and VR used to improve the domain of books and education.

2.3.2 Books and Education

Books and education provide important information (Ott and Feina, 2015; Liou et al., 2017) to those going through a traditional school system. Not every individual finds it beneficial to learn in one specific way. It is beyond the scope of this review to cover the types of learning. Still, VR and AR can expand a different dimension to how visual learning can be used, helping to focus on specific elements (McNamara, 2011) of interest or providing a more interactive experience to provide teaching points (Buhling et al., 2012). The following cases show examples where VR or AR can bring learning alive and potentially make it enjoyable for those who could otherwise find the interactions a dull experience. The first examples demonstrate the application of VR and AR for science with anatomy and physics visualisations, then interactive books, gamified education, and finally a model to apply when considering the appropriateness of using VR for education.

Science Learning Visualisation Through VR and AR

Science education contains many topics where a visual aid can help explain a concept, process or otherwise provide understanding. Seo et al. (2007) demonstrated a VR representation of a canine skeletal system. The various canine anatomy structures could be taught by constructing the skeleton within a VR environment. This representation of anatomy could teach more complicated anatomy, including that of other animals, or perhaps as part of teaching health-related topics.

Physics is an area of science that can be described or shown through images, but sometimes seeing a process in action makes understanding a process much more understandable. In the research of Dunser et al., they worked on creating an interactive physics education book that relied on AR (Dunser et al., 2012). The book allowed visualisation of things like a DC motor viewed as a 3D model coming off the page instead of the static image. Or interaction simulating magnets to demonstrate induced magnets. The visualisation was achieved by providing a handheld AR device that the user would use to view the book. The trials run found that AR aided in learning 3D concepts taught by the modified books. Similarly, although not directly related to science education, the following example demonstrates another example of adding different visualisation to books with AR.

Interactive Books with AR

Grasset and Billinghurst created a visually augmented illustrative children's book (Grasset and Billinghurst, 2008). The book had elements including 3D smoke, a 3D model house with immersion through clouds, cinematic effects, and text boxes that would appear based on gaze interaction. Their main limitation was identified as the hardware as it could not always keep up with the ways users tried to interact with the book. The close proximity of multiple text boxes also caused some issues for the application users where too many text boxes would be shown, causing confusion. Users from their study tried to interact with quick scanning and close up viewing. These were identified as the main areas for improvement.

Gaming Education with AR

Separate to the science and book examples used so far, simple game-like methods can augment the learning experience for the alphabet, spelling, or practising memory. Fiducial markers give the user something tangible they can manipulate without requiring the user to always observe a scene through the filter of an AR interface, which can be advantageous. Han et al. designed a marker system using the English alphabet (Han et al., 2011). Their system allowed combining markers to form words through proximity to other markers. Combining the markers to form words could then be used to represent an object through an AR. For example, combining letters to form the word “SHIP” made a ship model appear over the word. This type of interaction could be used with children to reinforce spelling activities.

Read-It is a tabletop collaborative game designed to assist children aged 5 to 7 with their reading ability (Sluis et al., 2004). The game was designed with a play style like that of “Memory”, where pairs of pictures are matched. In this case, users would need to identify the word and match it. The game used both audio and visual cues. The physical cards were tagged such that they could be interpreted as either face-up or face down. The game space was projected onto a table surface with projected content for the cards. The virtualisation of the game allowed for mitigation of cheating and provided appropriate error feedback for turning too many cards face-up or turning the same card multiple times in succession. These two game-like examples demonstrate the viability of using AR for enhancing learning. The application of AR, or more specifically VR, may not always be appropriate for all situations.

A Model for Determining if VR is Suitable for Education

Pantelidis (2009) discussed reasons for using VR in education with a model to determine appropriate use. Some of the reasons for promoting VR use in education included adding new forms of visualisation to provide alternatives to traditional mediums, motivating students through active participation, and the opportunity for experience by students beyond regular class time to learn at their own pace. Some of the suggested scenarios to use VR for education included use of simulations, the teaching of dangerous/difficult activities, environment-specific training, where the interaction can be comparable to the real activity, as a reduction of cost for travel/logistic costs, to share experiences, unique opportunities for information visualisation, to make activities more fun, providing access to activities for people with disabilities, and giving opportunities to make mistakes without consequences. The research provided a 10-step model to determine whether VR should be used in a specific education scenario. The steps started by identifying the specific course objectives, then considering the advantages/disadvantages of developing a simulation for the scenario. The list of possible simulations could be refined and evaluated for the level of realism required and the type of immersion/presence necessary. Based on the requirements, hardware solutions could be selected, and a virtual environment designed to suit the requirements for use on the hardware. The developed prototype should then be evaluated with a pilot study and a repeating evaluation until satisfactory. The environment could be then tested on the target population and evaluated with continual changes until it was suitable to the needs of a scenario.

This section has discussed a selection of examples where technology can be used to improve education. Pantelidis (2009) identified that the difficulty with using solutions such as VR for education could involve needing to commit upfront costs, time to teach educators how to use the software effectively, dealing with health and safety effectively, and reluctance in adoption for curriculum use. The potential benefits of incorporating VR into learning with appropriate scenario evaluation could provide additional tools for educators to engage with their students. The examples in this section demonstrated the benefits of VR and AR in science education for anatomy observation (Seo et al., 2007) and physics simulations (Dunser et al., 2012). More generally, for books, Grasset and Billinghurst (2008) demonstrated making books come more to life through AR. Typical games-like methods used to teach the alphabet, spelling (Han et al., 2011), and memory training (Sluis et al., 2004) were shown to improve variety in education with technology. The following section considers the art medium as part of its uses in AR and VR.

2.3.3 Drawing/Artwork

Drawing and artwork allow users to express themselves within a space. Many different art mediums exist, even within the domain of digital art. Listing all the types of digital art is outside the scope of this background. The purpose of this section is to provide examples of simple but important applications where art can be created in a way uniquely suited for AR and VR.

Laviole and Hachet have designed a system that uses AR to assist in some forms of physical drawing (Laviole and Hachet, 2012). Their system uses spatial AR, a technique sometimes used for advertising through projection mapping onto large buildings. They took this concept and set up a system that allows for direct manipulation of the digital content to set it up so that it can be drawn onto paper. The system enables easier tracing of the source material and could be used for shade mapping by comparing against a digital overlay.

One of the applications used to show off the abilities of the HTC Vive HMD was the Tilt Brush application (Google, 2016). Ars Technica referred to it as one of their "killer apps" (Machkovech, 2016). It is possible to draw in 3D with many utilities to make the art immersive using an application. Figure 2.15 shows an artist drawing a piece of clothing. Figure 2.16 shows an artist creating terrain from the ground up. Additional examples of the application and the type of interactions can be seen in Figure 2.17 (over) and Figure 2.18 (over).

Figure removed due to copyright restriction. See footnote for video source.

Figure 2.15: Tilt Brush Example A[14]

Figure removed due to copyright restriction. See footnote for video source.

Figure 2.16: Tilt Brush Example B[14]

[14] Google 2016, "Tilt Brush: Painting from a new perspective", URL: https://www.youtube.com/watch?v=TckqNdrdbgk, Last accessed 11/12/2021.

Figure removed due to copyright restriction. See footnote on previous page for video source.

Figure removed due to copyright restriction. See footnote on previous page for video source.

Figure 2.17: Tilt Brush Colour Chooser[14]

Figure 2.18: Tilt Brush Creature Drawing[14]

Figure 2.17 demonstrates a capture of the type of menu interaction used in the Tilt Brush application. The user would have a menu always attached to the left-hand controller represented in the virtual space. The menu contents could be swapped or otherwise interacted with by using the right-hand controller. The controls provided in the menu included features such as the colour wheel seen in Figure 2.17. A 2D capture does not do these scenes justice artists have created impressive scenes within the application[15].

This section has presented two examples from the domain of art, with an example of an AR tracing assistant (Laviole and Hachet, 2012) and VR Tilt Brush (Google, 2016) applications. Both instances present early use of VR and AR to enhance their respective mediums. As developers continue to innovate, there are likely to be many variations that seek to improve the applications further to empower artists with freedom of creative expression. The following section considers tourism and the impact of AR and VR on it.

2.3.4 Tourism

As long as there have been points of interest where people want to visit, tour guides have shown people around and marketing to drive people to those locations (Griffin et al., 2017). The role of a tour guide is to guide people through content and give them interesting information about the content they are observing. This role can be used in the context of walking through a museum using systems like the books described in section 2.3.2. The representation allows for additional imagery relevant to an observed element or more focused information related to the context. The following example demonstrates a sample of what can be used to improve a tourist's experience.

[15] Google 2017, "Tilt Brush: Art of Wonder with Liz Edwards", URL: https://www.youtube.com/watch?v=EUYJSxjUmYg, Last accessed 11/12/2021.

Arbela Layers Uncovered (ALU) is an AR application designed by Mohammed-Amin et al. (Mohammed-Amin et al., 2012). The purpose of the application is to provide a visitor experience for an archaeological site. The site is located in the Kurdistan region of Iraq, where there is a heritage site with some history. The application provided three types of viewing content: a 3D view with content based on where the camera is viewing, a 2D overview from above with a map showing pins with points of interest, and a reconstructed view showing how the building was in its original state. This type of tour with these three types of views could be applied to many tour locations to provide a more engaging experience.

The application of systems for tourists can be similarly applied to everyday scenarios such as using a shopping centre (Olsson et al., 2011). In the work of Olsson et al., the use of a mobile AR service was considered. They investigated the aspects where potential users could benefit from mobile AR within a shopping centre environment. This research's expected experience and design requirements can be found reiterated from the more general classifications found in section 2.4.1 (Olsson et al., 2012). Visual tagging with possible use for commodity identification was also explored by Mohan et al. using imperceptible visual tags (Mohan et al., 2009).

In summary, the examples shown related to tourism have demonstrated examples of expanding visualisation of archaeological sites (Mohammed-Amin et al., 2012), assisting navigation of people touring places like shopping centres (Olsson et al., 2011), and visual tagging for providing information (Mohan et al., 2009). It is plausible from these examples that both AR and VR could be used to offer targeted experiences for tourists or similar users. These technologies provide an opportunity to give users different perspectives showing additional contextual information, or the transition of states over time (such as before/after of a building), or otherwise guide users through curated experiences. In the context of this research, by reducing the required equipment for an immersive experience, i.e., implementing a system with just the HMD, exploration of interaction and immersion could improve the experience of these types of scenarios. Section 4.2.2 explores a related use case directly concerning the proposed PVMS technology for museums and sightseeing. The following section explores some examples of collaborative computing that allow multiple users to participate in the same virtual space using AR and VR.

2.3.5 Collaborative Computing

Having multiple actors engaging within a single VR or AR environment can provide a more enriching experience and expand what can be done beyond a single user experience. When using AR or VR, the content does not necessarily have to be located in the same place (same room or even country) and can instead be shown via telepresence. Telepresence is a process where multiple users view the same content but are in separate locations. The experience could be projected onto a tabletop environment or shown through a HMD. The discussion presents a couple of examples of research in collaborative AR and VR, starting with AR.

Collaboration with AR

AR provides a medium ideal for collaborative activities, particularly in 3D. The activities can be in the form of face-to-face collaboration, remote collaboration, or multiscale collaboration as described by Billinghurst and Kato (Billinghurst and Kato, 2002) with examples of each type. Face-to-face collaboration allows multiple users to view the same content simultaneously within the same physical space. Remote collaboration can be in the form of remote conferencing, where people can appear to be present despite not physically being in the same room. Multiscale collaboration is slightly different because it shifts from the content viewed in the same way by all users to providing different views based on a user's needs. These types of collaboration can be directly compared to VR. The big difference for purely VR is the lesser requirement for collaborators to be physically present in the same space.

Prince et al. (2002) demonstrated a different take on video conferencing. In their work, they used fiducial markers to position a collaborator in arbitrary locations. The person performing some kind of content would be captured by 15 cameras and placed onto the fiducial marker through the end user’s AR view relative to the position/orientation of the marker. This collaboration presentation would be less practical for most users due to needing cameras all around the target being captured. More realistically, a single viewing angle could be captured and placed in an arbitrary place within the end user’s world. Either in an AR world using a fiducial marker or perhaps even as some form of custom placement within a VR environment.

Collaboration with VR

Theoktisto and Fairén (2005) presented a VR collaboration system by converting an existing ALICE VR Navigator tool. They added collaboration tools to support inspection and manipulation of complex models. Their research discusses the considerations that had to be made to convert the existing system. The original system offered stereoscopic visualisation, user position and orientation tracking, different VR modes to support different types of VR displays, and multiple interaction devices. The examples of collaborative demos produced included inspection and navigation of the interior of a ship (to discuss the environment remotely), training in medicine (for a surgeon to demonstrate a scalpel for incisions to students), and inspection and modification of architectural design. The results showed their framework integrated well with the existing system and performed well.

Mütterlein et al. (2018) investigated the driving factors of VR-based collaboration with 102 participants. Immersion was found to be the primary driver of users' intentions for collaboration during their study, with interactivity also found to be an important factor. The representation of team members through optimal avatars was found to be less important. They suggested the focus should be the collaboration aspects centred around interaction and the virtual environment to promote immersion.

This section has presented a few examples of collaborative computing that include the use of AR or VR. Billinghurst and Kato (2002) and Prince et al. (2002) demonstrated examples of discussion around the use of collaborative computing for AR. Theoktisto and Fairén (2005) presented benefits for navigation, training, and inspection in VR, and Mütterlein et al. (2018) found that immersion was the primary driver of user intentions for collaboration. Collaboration on activities becomes more important to ensure activities or work can be shared regardless of the distance with an increasingly more connected world through the internet. These experiences could be added to by incorporating the proposed PVMS to aid in various utility actions dependent on the applications. The following section considers the use of AR and VR for assisted construction.

2.3.6 Assisted Construction

Assisted construction includes any experience where parts are combined virtually to prototype something that could be created as a physical system. An example experience could consist of visualising buildings in a VR walkthrough to experience how the end product will look or prototyping designs with moving parts that may need simulation. Or, as in the following example, direct a user with contextual information to complete the construction of an object with AR assisted guidance.

Barna et al. designed a system that assists with a construction process (Barna et al., 2012). Their system understood fixed locations and the appearance of pieces that needed to be combined to form a completed gearbox. This type of system could be applied to a broader range of assembly-oriented applications. The AR interface could show colour coded markers for where elements needed to be placed. Once the elements were placed, the object could be recognised, indicating that the next part needed to be inserted. The ability to view how a piece of hardware will be placed when shown directly in real-time in the main field of view reduces the errors introduced through the interpretation of an assembly manual.

The advantages of VR for product prototyping and assembly have also been discussed by Seth et al. (Seth et al., 2011). They presented the importance of planning in product development to improve production efficiency and product quality as a cost reduction measure and shortening the time to reach the market. Current techniques are discussed, including computer-aided design (CAD) and computer-aided assembly planning (CAAP). The challenges of adapting and improving the existing techniques become incorporated into a VR environment are discussed. Some of those considerations include collision detection (suggesting that computer models should allow for realistic collision detection), inter-part constraint detection and management (the assemblies could be constructed with pre-defined constraints to reduce issues with physics simulations), and physics-based modelling (where realistic simulations can enhance immersion/interactivity).

The presented examples of assisted construction, such as the construction assistant by Barna et al. (2012), could be potentially enhanced by incorporating the proposed PVMS from this research. Using a hidden menu while observing virtual objects freely to provide additional creation/manipulation options could improve these software solutions. The following section considers examples of pervasive applications that utilise a user’s location as a primary interaction component.

2.3.7 Pervasive Applications

Pervasive AR refers to the ability of the user to freely move through the real world, engaging with the application where points for interaction are real-world locations. Moving to different physical interaction points is particularly popular in games such as Ingress (Niantic, 2012) or Pokémon GO (Niantic, 2014). Ingress is a massively multiplayer online role-playing game that uses GPS. Multiple players engage with each other through controlling landmarks (referred to as “portals”) as part of a faction. The players' actions create an ongoing attempt to control the highest number of world areas portrayed in the AR environment. The examples presented in this section demonstrate three examples of outdoor, followed by one indoor example to explore how researchers have used pervasive applications.

Outdoor Pervasive Applications

Tracking within the pervasive environment can be complicated to perform accurately based on how reliable the reception of the GPS is. Guan et al. presented a method of dealing with extreme, large-scale areas within an AR environment (Guan et al. 2011). In their system, pre-processing at locations of interest would occur. This pre-processing involved taking images to generate a 3D point cloud and then matching that with real-world coordinates and clustering the point clouds. Then to determine the location and camera pose of the user, an image would be captured, partitioned and combined with a rough GPS location to perform a search and calculate the pose based on the existing images. Their results indicated this system performed well; however, many clusters would be needed in a complicated environment for large-scale applications. Thus, causing overhead in pre-processing.

Zarzycki created an urban environment game called “Mystery Spaces” (Zarzycki A. 2012). In this game, the user’s goal was to explore an urban environment to increase awareness of underappreciated public spaces. The application allowed users to follow routes with places marked as points of interest. This application demonstrated the geo-caching and gamification of traditional sightseeing. Geo-caching and gamification could be applied more broadly to any tour or travel between different locations if the content was set up suitably using geo-caching.

While in a pervasive environment using AR, it may be necessary to identify or move toward specific points of interest that already have information associated with them or where a user intends to contribute data. Lu et al. investigated subtle cueing for visual search (Lu et al., 2012). The example scenario considered proposed use in aiding a paramedic during earthquake rescue, with features such as identifying obstacles and assisting locating people needing aid.

Indoor Pervasive Applications

Indoor navigation within a building is traditionally more complicated due to the reduced accuracy of GPS and other systems that allow determining a position. One way to get around this issue is to have markers placed within the buildings that an application supports to provide visual context for the tracking system. This approach was used by Mulloni et al. for indoor navigation and pathfinding (Mulloni et al., 2011). Their application could suggest a route based on the known locations when you arrive at a point. The main downside of this approach was the requirement to prepopulate the environment with markers to orient the application.

In summary, the examples have demonstrated pervasive application uses, including games with Ingress and Pokémon GO, position estimation with GPS (Guan et al., 2011), "Mystery Spaces" as a tourist activity in an urban environment (Zarzycki A. 2012), search and rescue (Lu et al., 2012), and indoor navigation of buildings (Mulloni et al., 2011). Pervasive applications are, by their definition, skewed toward the AR end of the continuum between AR and VR that was presented by Milgram et al. (1995). An important consideration regarding their use is that although it may be hazardous to use the VR while moving freely with the world actively, this can change once reaching points of interest. Upon reaching a point of interest driven by a pervasive AR application, the user could switch into a VR mode. Consider, for example, if the user is visiting a museum or ruins of some sort. This scenario will be discussed further with an example in section 4.2.2. The following section looks at examples of non-commercial games researchers have developed for mixed reality.

2.3.8 Mixed Reality Gaming Research

This section discusses three examples of mixed reality gaming research that were inspirational in the early stages of the investigation. Each instance is concluded with brief text discussing their significance.

ARQuake takes the game Quake by ID Software and brings it into AR. (Piekarski and Thomas, 2002; Thomas et al., 2000; Thomas et al., 2002). The game was developed to be played in the physical world with the freedom to move around. The game's view is determined solely by the orientation and position of the user's head. The game is experienced as AR using a transparent HMD, and the game is controlled through real-life props and metaphors to make it easier to understand. The haptic feedback of the gun players could use provided additional immersion. The player's position was inserted into the game with an absolute tracker position and orientation. Their demo contained mapping of 30 buildings within a 350m by 450m area. Although they believe the system could be used anywhere if maps were adequately constructed. The setup for this experiment indicated the ability to track the orientation and position of the user's head. Access to orientation tracking would allow for the use of the PVMS proposed in this dissertation. A user could engage with the world playing the game and modify the parameters of their game state by engaging with the PVMS.

Human Pacman combined the well-known game Pacman with an AR system and multiplayer component to provide a different way of playing the game (Cheok et al. 2003, 2004; Magerkurth et al., 2005). In Human Pacman, the players were put on two opposing teams. The Pacman team had two pacmen with two helpers, and the ghost team had two ghosts and two helpers. The world was viewed through a HMD. The pacmen would move through the world collecting plain and special cookies. The ghosts would attempt to "devour" a pacman by tapping them on a sensor. The ghosts could be "devoured" as well if the pacman trying to do so had the correct powerup. The setup in this experiment is interesting because the helpers were viewing a VR version of the game world viewed from anywhere physically. Meanwhile, the players were participating by using AR. An interesting consideration to be made here is how elements could be introduced more with the overlap between players in an entirely VR environment introducing elements to the players participating in the AR environment. A similar type of experience was also found in the NetAttack game from the following example.

Broll et al. designed a system titled MORGAN and used it to develop three outdoor example games (Broll et al. 2006). The first game was called NetAttack. It was a scavenger hunt game where the goal was to destroy a central database. They would collect objects within the game world and combine them to access the database and destroy it. The game had two teams, with one indoor and one outdoor person per team. The indoor person would aid their teammate in locating the objects they were seeking (from a computer interface showing the positions of objects), and the outdoor player would view the world through a HMD. The second game, Epidemic Menace, puts players in a team-based world where viruses have been released. As players progress through the game, their score could give access to upgrades that allow them to see and destroy the viruses using AR. An important part of this game was the viruses moved dependent on real-time weather data. They would move relative to the wind direction and strength. The third and final game was titled TimeWarp. The game was not complete at the time of paper publication but included localisation with virtual characters and building elements that a player could interact with within a town.

The examples described all used HMDs as part of an AR experience. ARQuake (Piekarski and Thomas, 2002; Thomas et al., 2000; Thomas et al., 2002) demonstrated a translation of a traditional shooter experience converted into AR. Human Pacman (Cheok et al. 2003, 2004; Magerkurth et al., 2005) presented an overlap between using a VR interface for one user and the other players using AR. MORGAN (Broll et al. 2006) combined three AR games, including a scavenger hunt, a team-based fight against viruses, and virtual interaction with characters. These show a sample of experiences that could have improvements to menu interactions added for initiating additional actions in the context of the applications. The following section covers gaming from the commercial angle with games from the VR commercial market.

2.3.9 Commercial Gaming

HMDs have recently become cheaper and popular, with more significant VR experiences and games being released. As the research for this dissertation began, the Oculus Rift had many applications developed for it and the other devices that have followed. One place to view game-specific applications is the VR section on Steam, titled Steam VR (Valve, 2018). These range from Minecraft (a block-based survival/building game) to horror-themed games such as Slender. In the case of horror games, the newer HMDs provide immersion, increasing the sense of fear when playing horror games.

Development of experiences for HMDs has seen an increase in interest and ongoing support provided through development environments, such as Unity and Unreal Engine. The ongoing support has allowed the market to flourish with many innovative and unique creations for the VR platform (Pallavicini et al., 2017). The consumer demand for VR experiences has grown in recent years due to cost-effective, accessible, and portable hardware. The interest in these technologies and the experiences provided has seen a wide variety of content developed for the commercial market, moving beyond the niche research applications that have previously been seen. This section will highlight a few prominent or interesting examples showing the capabilities of VR within gaming.

Figure removed due to copyright restriction. See Steam store page in footnote for original.

Figure 2.19: Beat Saber Gameplay[16]

Figure removed due to copyright restriction. See Steam store page in footnote for original.

Figure 2.20: Beat Saber Main Menu[16]

Beat Saber (Beat Games, 2018) is a type of rhythm game. Rhythm games use a musical or audible beat and require the player to perform actions synchronised with the beat. In the case of Beat Saber, this involves using two controllers in the hands to slash oncoming blocks in VR. Figure 2.19 shows typical gameplay where the user completes the game by cutting coloured blocks with the correct coloured lightsabre controller. The blocks and other forms of obstacles are displayed based on music. With typically faster-paced beats in songs showing significantly more blocks to destroy. Figure 2.20 shows the way menus appear during the main menu. The user is positioned looking at three static interface panels that update to represent the current menu state. Beat Saber came out after the experiments for this research were completed. The menus shown in Figure 2.20 can be compared to the menus from experiment 1 (Chapter 3), with their layout reflecting a similar setup. The menus can also be compared to the interfaces used to present experiments 2 and 3 (Chapter 5 and 6). These similarities from successful commercial products after the research was conducted further validate the choices in interface design.

[16] Beat Games 2018, “Beat Saber”, URL: https://store.steampowered.com/app/620980/Beat_Saber/, Last accessed 11/12/2021.

Figure removed due to
copyright restriction.
See Steam store page
in footnote for original.

Figure 2.21: Job Simulator Example A[17]

Figure removed due to
copyright restriction.
See Steam store page
in footnote for original.

Figure 2.22: Job Simulator Example B[17]

Figure removed due to
copyright restriction.
See Steam store page
in footnote for original.

Figure 2.23: Superhot Example A[18]

Figure removed due to
copyright restriction.
See Steam store page
in footnote for original.

Figure 2.24: Superhot Example B[18]

Figure removed due to
copyright restriction.
See Steam store page
in footnote for original.

Figure 2.25: Arizona Sunshine Example[19]

Figure removed due to
copyright restriction.
See Steam store page
in footnote for original.

Figure 2.26: VRChat Example[20]

The above selection shows three other games that were briefly reviewed while looking at the VR game market. Job Simulator (Owlchemy Labs, 2016), seen in Figure 2.21 and Figure 2.22, had the user complete food orders for customers where they controlled hands with VR controllers to lift objects in the world. Superhot (Superhot Team, 2017), seen in Figure 2.23 and Figure 2.24, had enemies that only moved when the player moved. The interaction technique allowed for a unique strategy for dodging bullets and otherwise interacting with

[17] Owlchemy Labs 2016, "Job Simulator", URL: https://store.steampowered.com/app/448280/Job_Simulator/, Last accessed 11/12/2021.

[18] Superhot Team 2017, "Superhot VR", URL: https://store.steampowered.com/app/617830/SUPERHOT_VR/, Last accessed 11/12/2021.

[19] Vertigo Games 2016, "Arizona Sunshine", URL: https://store.steampowered.com/app/342180/Arizona_Sunshine/, Last accessed 11/12/2021.

[20] VRChat Inc 2017, "VR Chat", URL: https://store.steampowered.com/app/438100/VRChat/, Last accessed 11/12/2021.

the game. Arizona Sunshine (Vertigo Games, 2016), seen in Figure 2.25, is a zombie shooter where the player would traverse the world in VR searching for supplies while surviving zombie attacks. VRChat (VRChat Inc, 2017), seen in Figure 2.26, is very much a player-driven game. The game allows players to become almost any avatar they wish in a 3D world where they can speak and interact with other players within VR. The world can provide experiences the community of players want to develop. These virtual worlds can then be shared with many other players to provide unique situations for interactions. The game offers an immersive collaborative experience with a community of thousands of players.

The games presented in this section have shown some of the examples of VR games that were observed to see the types of experiences being offered by developers in commercial VR made available through devices such as the Oculus Rift and HTC Vive.

In summary, many examples have been shown to capture the breadth of interactive experiences. The purpose of this section at its outset was to survey and understand how incorporating AR and VR impacts domains. Health and assistive technologies were investigated in section 2.3.1, with research examples showing how VR could be used to help people. Examples included assisting people with dementia (Flynn et al., 2004; Moyle et al., 2018; Hodge et al., 2018), providing rehabilitation for stroke patients (Laver et al., 2015), and using exposure therapy in VR (Linder et al., 2017; Maples-Keller et al., 2017) and AR (Wrzesien et al., 2011). Section 2.3.2 presented example uses for books and education, including enhancing science for anatomy (Seo et al., 2007) and physics (Dunser et al. 2012), the visualisation of books with AR (Grasset and Billinghurst, 2008), and learning for spelling (Han et al., 2011) and memory (Sluis et al., 2004). Following the section on books and education, section 2.3.3 explored the use of applications such as Tilt Brush (Google, 2016) that demonstrate new alternative ways to create art.

Examples of tourism were shown in section 2.3.4, including an AR archaeological site experience (Mohammed-Amin et al., 2012) and navigation of a shopping centre (Olsson et al., 2011). Collaborative computing combines multiple users, and as discussed in section 2.3.5, could be used for applications such as the ALICE VR Navigator Tool to navigate ship interiors, train people in medicine and inspect or modify architectural designs as part of a collaborative experience. Mütterlein et al. (2018) found that immersion was the most important factor in the experiences in collaborative computing. Assisted construction was presented in section

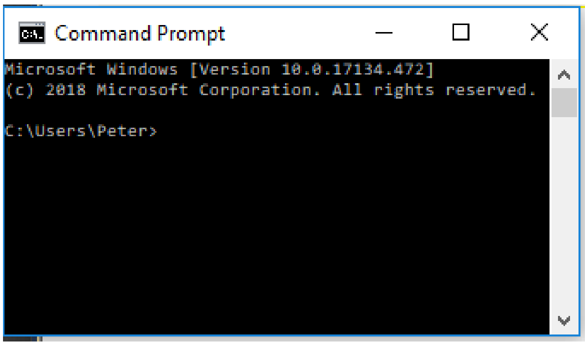
2.3.6 and showed an example of an assembly assistance application using AR (Barna et al., 2012) and discussed the advantages of using VR for prototyping products and assembly (Seth et al., 2011).

The section on pervasive applications (2.3.7) showed examples such as Ingress (Niantic, 2012) and Pokémon GO (Niantic, 2014) as mobile games, the use of a "Mystery Spaces" application for exploration of urban environments (Zarzycki A. 2012), and indoor navigation (Mulloni et al., 2011). The section discussing mixed reality gaming research (2.3.8) presented three examples of games titled ARQuake (Piekarski and Thomas, 2002; Thomas et al., 2000; Thomas et al., 2002), Human Pacman (Cheok et al. 2003, 2004; Magerkurth et al., 2005) and MORGAN (Broll et al. 2006). Each of the examples demonstrated different examples of using AR for gaming. In the case of Human Pacman, it combined the AR world with a separate user observing from VR. This final section (2.3.9) considered examples from commercial gaming, including presenting overlaps in how more recent games such as Beat Saber (Beat Games, 2018) have shown their menus. The examples discussed are a limited number of the contributions to AR and VR but demonstrate the breadth of their respective domains. The following section will look at user interfaces and their evolution, leading to a more focused discussion on AR and VR.

2.4 Evolution of User Interfaces

User interfaces utilise displays to present data to the user. The way this is represented has changed significantly from the origins of computing, with the improvements to computer hardware, user needs for the types of data a computer needs to represent, and an evolving attitude to how computers are used. Concerning the PVMS's menu layout, a specific discussion is later provided as part of section 4.3.1 with a direct connection to the influence of mobile application development on its design. This section will briefly look at a few areas concerning how interfaces have evolved and are used with VR and AR. Presented here are some of the milestones of computing user interfaces that form the foundation of AR and VR interfaces, discussed in the subsequent sections.

Command-line and text interfaces were the common method of computer interaction with early computers with a display (Liu, 2010). A limited number of pixels, colours, and computer speed, meant the first interfaces were limited for technical reasons. For their time, these interfaces were still a powerful tool, which sped up user activities compared to conventional approaches of the time. Figure 2.27 shows an example of a simple command-line interface that is still used today for some activities.



```
Microsoft Windows [Version 10.0.17134.472]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\Peter>
```

Figure 2.27: Windows Command Prompt

Figure removed due to copyright restriction. See video in footnote for original.

Figure 2.28: First Computer Mouse[21]

The addition of a simple mouse device seen in Figure 2.28 (Engelbart and English, 1968; SRI International, 2018) made interactions with a graphical user interface (GUI) possible. Before the mouse, the primary input was the keyboard alone. There has been continual evolution to make the most of newer interaction controllers. For the research presented in this dissertation, various types of interaction controllers are considered in section 2.5.

Early GUIs began to adopt a WIMP (Windows, Icons, Menus, Pointers) model and were often very simplistic in option variety and choices (van Dam, 2000). As this has moved toward current computer systems, most applications still follow a WIMP model for user familiarity. The accessibility of WIMP interfaces opened up computer technology to a broader, consumer-level market, moving away from specialised usage scenarios. The demands of the novice user and the widespread adoption of personal computing led to further enhancements and accommodations associated with WIMP interfaces which have seen the mouse and keyboard become universally accepted interaction devices.

By the nature of their platform, some devices have necessitated an evolution of the WIMP interface paradigm. For example, with mobiles and tablets, it is typically the case that only a single application will be visible to the user (even if there are many in the background). Screen space is important because of the physical and resolution properties available. The advent of touch-sensitive screens allowed users to interact naturally with their devices without the

[21] SRI International 2018, "1968 'Mother of All Demos' by SRI's Doug Engelbart and Team", Youtube URL: https://www.youtube.com/watch?v=B6rKUf9DWRI, Last accessed 11/12/2021.

need for additional external tools like the mouse or a stylus. This natural interaction between the users' fingers and the screen surface allowed for simple tap interaction and accommodated gestures for a range of operations, one of which was to make content appear from the sides of the screen depending on the context (Jacob et al., 2008). This notion of bringing applications, additional content, or other features into the user's view of the device led to the early concepts of the periphery menu system designed as part of the research presented in this dissertation.

Natural User Interfaces (NUI) take this sort of interaction a step further and attempt to utilise newer input forms to optimise how users can interact (Liu, 2010; Jacob et al., 2008). The interactions with NUIs are often device or application-specific. For example, requiring inputs supporting gestures or voice cannot be assumed for just any user. The advantages of using these additional interface tools can be used when the content has been targeted at a specific audience. Some of the significant devices that enable the development of NUIs include the Perceptive Pixel, Microsoft PixelSense (formerly called Microsoft Surface before Microsoft acquired Perceptive Pixel), and the Microsoft Kinect. The Perceptive Pixel and PixelSense function as large screens for interaction as a tabletop, with object recognition and other natural interface developments (Reisman et al., 2009; Banes 2009). The Microsoft Kinect primarily tracked movements within 3D space, allowing gesture detection through full-body tracking of observed actors (Marin et al., 2014).

Similar to NUIs is the development of Tangible User Interfaces (TUI), which use objects as metaphors or interaction points, otherwise referred to as tangible bits. One of the earliest such systems using this type of interface was that presented by Aish (1979). In Aish's work, a physical model of a building design could be constructed and then be evaluated by the computer to turn it into an architectural production drawing. Further to the work of Aish, two notable researchers in the area of TUIs are Ishii and Ullmer (Blackwell et al., 1995; Ishii and Ullmer, 1997; Ullmer and Ishii, 2000; Piper et al., 2002; Homquist et al., 2004; Blackwell et al., 2007). Ishii and Ullmer (1997a; 1997b) presented a metaDESK that used physical objects as metaphors for GUI elements. Some examples included a lens representing a window, a physical handle representing a GUI handle, and an instrument representing a widget such as a slider. Alongside the metaDESK they also investigated using an ambientROOM (Ishii et al.,

1998) to provide background awareness through subtle light, sound, and movement, and transBOARD as a digitally enhanced whiteboard.

Fundamental to interface design and menu development considerations were two design principles: Fitts' Law and Hick's Law. The first law that has seen significant use throughout research since its inception is Fitts' Law (Fitts and Posner, 1967; Scott MacKenzie, 1992). Fitts' Law presented the connection between distance to and size of a target. Applied, this means that further distances and smaller targets will impact the time it takes a user to complete interactions. The interactive elements presented as part of all the experiments have used suitably large objects to make them easy to select with anticipation that the user will cover interactions over varying distances depending on their current context. The second principle that was significant in determining how the content was presented on the menus was Hick's Law (Hick, 1952). Hick's Law deals with a different aspect of time for interaction compared to Fitts' Law. Hick's Law deals with choice when faced with varying amounts of stimuli. Providing more options increases the time it takes for a user to decide. Therefore, to mitigate the issues defined by Hick's Law, the options and menu design were presented with a minimum number of choices for quick selection incorporating short word complexity of options.

The following subsections cover some of the additional research related to AR and VR focused on user interfaces. In section 2.4.1, AR and VR interfaces are discussed to identify design considerations. Section 2.4.2 moves the discussion to HMDs to focus on the type of interaction used in the experiments presented in this dissertation. And finally, section 2.4.3 presents some considerations about representing contextual information.

2.4.1 Interfaces for Augmented and Virtual Reality

Designing interfaces for AR and VR has a significant distinction from the traditional desktop computing display due to how they immerse the user. These mediums provide a view into either an altered representation of the real world or a completely virtual world that a user can interact with using appropriate hardware. The ability to view the environment within the virtual worlds, focusing on the content presented, is important, so overlaying WIMP-like interfaces all over the screen is not practical. Furness et al. (1995) identified the following list of ideal attributes that should be considered for allowing the user to coexist with their virtual environment.

- Matches the sensory capabilities of human.
- Easy to learn.
- High bandwidth bridge to the brain.
- Dynamically adapts to the needs of the task.
- Can be tailored to individual approaches.
- Natural semantic language.
- Organisation of spatial/state/temporal factors.
- Macroscopic vs microscopic view.
- High bandwidth input.
- Information clustering.
- Information filtering.
- Unambiguous.
- Does not consume reserve capacity.
- Easy prediction.
- Reliable.
- Operates when busy.
- High semantic content (simple presentation).
- Localisation of objects (movement, state, immediacy).
- Sense of presence.

When considering each of the elements in the list, it is evident why they are important. Matching the sensory capabilities is primarily influenced through providing stimulus either onto the user with visual/auditory/tactile/scent-based stimuli or onto the system itself by using the user as an input source. Being easy to learn makes interfaces and systems easier for novice users to pick up. A high bandwidth bridge to the brain implies the application's capability to show a lot of information when needed by the user. Dynamically adapting to the user's needs can be crafted to guide the user in an expected way or otherwise present information contextually (as is discussed further in 2.4.3). Tailoring to individual approaches can be directed toward either expanding the variety of application types or an extension adapting to users' needs. Providing a natural semantic language refers to the way objects can be described within the environment. The organisation of spatial/state/temporal factors allows a user to modify elements within the world or otherwise shift their perspective and world view. Macroscopic vs microscopic view is a comparison of scale; in some contexts, a user may need to view the whole picture of something or some individual detail with a definition of how this transition between views is shown.

High bandwidth input allows for a high frequency of changes within the state of a VR application. Information clustering and information filtering relate to how information can be condensed in different ways to make it easier for visual consumption. For interfaces to be unambiguous, it should be evident through a short period of observation the intended purpose of any action. The benefits of unambiguity are reiterated through elements of interfaces having easy prediction. The reliability of a VR environment ensures that interactions remain predictable. Operates when busy refers to the environment continuing to present itself fluidly even when there is a lot of content present. High semantic content implies that content should be presented simply at a glance, with the most important details presented simply. Localisation of objects with movement, state, and immediacy can be manifested throughout VR applications as interaction points. And finally, a sense of presence implies that the user feels immersed within the presented environment.

Olsson et al. (2012) performed a survey of user experience expectations of mobile AR. The outcomes could be applied more broadly as the goals of both VR and AR applications. Keywords related to the expected experience from mobile AR applications were identified as seen in the following list.

- Captivation (e.g., immersion within the application/world)
- Collectivity (participation as part of a larger community)
- Connectedness (awareness of others who are participating)
- Creativity (creative self-expression)
- Efficiency and accomplishment (time or effort saving through ease of information)
- Empowerment (ability to reach new goals)
- Increased awareness and knowledge (increased insight into events and information)
- Inspiration (a sense of encouragement, perhaps to try new things)
- Intuitiveness (natural feeling to use)
- Liveliness (constant new content to provide a dynamic, vivid experience)

- Playfulness and entertainment (joy, amusement, or playfulness from learning new content)
- Surprise (ability to receive unexpected information or have expectations surpassed)

Not all these characteristics of experiences may manifest in all VR and AR applications. There should, to an extent, be the ability to use these characteristics for evaluation for these types of interfaces. Additionally, Olsson et al. defined several design requirements for mobile AR systems (Olsson et al. 2012). The design requirements can be equally applied to develop for VR and AR, as seen in the following list.

- Easy and flexible access
- Distinct affordances (cues about augmented content that are subtle when they need to be)
- Privacy and control
- Reactivity
- Relevance
- Reliability

The two lists from Olsson et al. are directed toward a mobile AR application but can be considered in the context of a VR system. First, consider the initial list of expected experiences in the context of a VR application. Providing captivation keeps a user invested in their use of the system. Collectivity and connectedness are seen through the collaborative experiences and avatar-based interactions being implemented across several VR applications and platforms. Creativity and inspiration allow users of VR to come up with unique solutions beyond being guided through a set sequence. Efficiency and accomplishment through presenting interfaces in an organised and logical way to save the user’s time spent completing activities in VR. Empowerment is demonstrated in VR by providing guidance and instruction within the application to direct the user toward their goal. Increased awareness and knowledge should be clearly illustrated through stimulus relevant to the type of information being engaged within the world. Intuitiveness improves a VR experience by promoting ease of use. Liveliness can be shown through objects within the VR world reacting as expected to promote playfulness or entertainment. Not all information has to be expected, certainly in the context of a game, where surprise is important to create different outcomes that a user needs to adjust to as part of that experience.

Considering the second shorter list of design requirements in the context of VR, making VR elements easy and flexible to access is important within VR because it promotes understanding for the user that can be rapidly learned. Distinct affordances are just as important in VR as they are in AR. The visual space is limited, so designing elements with easily identifiable cues saves visual space and promotes mental interaction models. Privacy and control are important characteristics of any software system, especially VR. The sense of comfort and security a user has with their digital presence is essential for continued use. Concerning privacy and control, the users should be given the ability to control the access to their information if it is necessary. Reactivity is implied through VR by users viewing a changing world directly and can engage in-situ and react to changes in real-time. Relevance comes back to the distinct affordances; information should not be presented if irrelevant to the user. And finally, reliability is important in VR because it breaks the user's immersion if interfaces or actions to interact with them are not consistent.

Another more general set of interface design rules are the 8 Golden Rules conceptualised by Schneiderman (1992). The following list summarises the rules.

- Strive for Consistency
- Enable Frequent Users to Use Shortcuts
- Offer Informative Feedback
- Design Dialogue to Yield Closure
- Offer Simple Error Handling
- Permit Easy Reversal of Actions
- Support Internal Locus of Control
- Reduce Short-Term Memory Load

The concepts are similar to those from the previous lists. Consistency, in this case, is intended to not just be within a single application but to more broadly strive for consistency that users can understand between different applications. Shortcuts provide a means for skipping straight to action instead of following a more rigid sequence of steps. For dialogue to yield closure, it should tell the user the activity has finished (e.g. proof of purchase receipt). Simple error handling should provide a fool-proof experience to walk users through precisely what they need to do. Providing easy reversal of actions can relieve anxiety by allowing a user to understand changes can be undone; therefore, inviting exploration of options. Supporting internal locus of control lets users initiate actions and gives agency over how the system behaves. Reducing short-term memory load relates similarly to Hick’s Law (Hick, 1952). Presenting fewer options will allow a user to assess the current state and efficiently choose behaviour to progress their state.

When designing interfaces and interactions, the points presented by Furness et al. (1995), Olsson et al. (2012) and Schneiderman (1992) should all be considered in how they are being managed within AR and VR. When prototyping different interfaces, it is important to visualise how they will appear. In the case of AR and, to some extent, VR, this can be done with video prototyping (de Sa and Churchill, 2012; de Sa et al., 2011). Video prototyping was not used for this research. Still, it seems like a helpful approach for conceptualising how content should appear in a more dynamic world when the user can navigate with 3DOF or 6DOF.

In summary, the PVMS and interfaces designed for the experiments drew from each presented list with varying considerations. Some list elements relate more to the design of an entire application than the function of a menu. In designing the PVMS, attributes like allowing a user to reverse their actions are not something the menu system itself would handle but could be provided as a menu option. The PVMS is intended to be a general-purpose menu system. Some of the key points relevant to the intent in design for the PVMS from Furness et al. (1995) were that it be easy to learn and dynamically adapt by providing appropriate context. Furthermore, it should be reliable to the extent of availability when the user needs it, and it should give a sense of presence. From the two lists provided by Olsson et al. (2012), some key factors are the user's empowerment by providing an engaging menu with intuitive behaviour. Additionally, including ease and flexibility of access with relevant options delivered reliably. Finally, Schneiderman’s (1992) rules offer key points useful to this work in providing a consistent experience for the user with support to allow agency over when interactions and behaviours occur (internal locus of control) and simple menu options to reduce short-term memory load.

This section has provided an overview of design requirements for AR and VR interfaces. Awareness of, consideration for, and application of these design principles were key when designing the VR interface experience used for the experiments conducted as part of the research presented in this dissertation. The following section will give additional considerations specifically for integrating interfaces into HMDs.

2.4.2 Interfaces for Head-Mounted Displays

HMDs have been a significant part of development into AR for many years. Work on improving the response of HMDs has been worked on by people such as Azuma (Azuma 1994, 1995). They have had a significant impact on the field of AR through much of their research by improving static registration in see-through HMDs and using inertial sensors to predict head motion. Feiner demonstrated early methods of presenting 2D windows inside of 3D AR (Feiner 1993). In their studies, they demonstrated the ability to show contextual information that was placed within the 3D environment.

Zhou et al. (2008), in a review of past AR tracking, interaction, and display, identified a few expectations regarding the user interfaces. The following dot points summarise the three main areas identified as important for interaction and user interface research.

- Ubiquitous computing: refers to how a user views and interacts with the environment from devices not constrained to a single location (e.g. desktop computer). Ubiquitous computing was first defined by Weiser (1991). For example, Law et al. (2012) considered the evaluation of environmental analysis. That same sort of analysis and element identification can be brought visually into a HMD, potentially anywhere outdoors. The form factor of head-mounted interfaces means they are reaching a point where they do not all need a constant connected power supply. Instead, they can run on a battery system and become ubiquitous without the need to tether to a desktop computer.
- Tangible bits: otherwise referred to as TUIs, leverage unique affordances of objects in the real world to connect with the virtual world. With a HMD that could be worn constantly, there is potential with improved object detection to identify any object automatically. The next step after automatic object identification would logically be to determine relationships between objects automatically. This relationship is dependent mainly on a large data source such as the internet or a more specialised

tool to determine purpose rapidly. Many researchers have explored TUIs, including Ishii and Ullmer (1997a; 1997b).

- Sociological reasoning to interaction problems: related to social, cultural and psychological phenomena, this paradigm will become a significant part of society as content platforms (e.g. Facebook) integrate user information into AR systems. With a considerable source of user-created content and the ability to contextually link that information to the world, it presents a breadth of possibility for sharing knowledge, experiences and expressive content.

Considering more specifically the types of interfaces that can be used as part of a VR interface, a major difference is how these interfaces are positioned within the virtual world. The following list identifies examples of how interfaces are used and adapted from reviewing existing examples of applications. The list illustrates how example interfaces react to user stimuli to determine the relationship between the interface and its relative position to a user.

- Static Overlay: This type of interface attaches to the camera. This type of interface can be overly obstructive when viewing virtual environments because they detach the user from the expected head rotation interactions.
- Static Interface in World: This is the simplest type of interface where the interface is prepositioned within a virtual world and will always appear in the same place.
- Static Interface attached to Object: This interface type might be hidden and shown when some action is completed. This interface is still static relative to the object’s location within the world.
- Dynamic Interface attached to Character: This interface is directly attached to the person playing the game or another user of the world. This interface type allows the user to take the interface with them by attaching the interface to a portion of the body. This attachment may be observed when looking down at the user’s chest or found attached to hand controllers.
- Dynamic Interface attached to Object: Interfaces that must move with the user as they control a moving object may need to be attached to an object in the virtual world. For example, while a user is travelling in a virtual vehicle, and they cause an interface to appear, it would be necessary to move the interface with the object in a relative way.

These different types of interfaces are not a one size fits all approach for any application. An application could use some or all of these features to provide diverse functionalities as required for the end-user to perform actions. Whatever the type of interaction in a HMD, it is necessary to consider how an application would be used outside of a traditional desktop computer environment. There needs to be something gained from performing the actions within a virtual environment. When comparing desktop to HMD use with navigation tasks, Santos et al. found participants performed better on average with the desktop (Santos et al., 2009). Except for participants who seldom played computer games performing better with the HMD.

All interfaces prototyped and developed as part of the experiments (Chapter 3, Chapter 5, and Chapter 6) are presented within a HMD. The choices for how to present these interfaces have been based on the information detailed throughout this section and influenced by the research/examples shown as part of this background research. The following section presents some additional observations and discussions about displaying information contextually.

2.4.3 Representing Contextual Information

Displaying information contextual to a situation within the HMD environment is essential to reducing overall clutter and directing application purpose. Determination of what information is relevant at a given time separates suitable applications from those that make the users suffer high downtime or difficulty interacting. Areas where this context can come from vary from speech, object recognition, marker recognition, or other sensors to determine the viewports relative position to points of interest. Each example using AR is detailed and then discussed with context to how they could relate to VR for presenting contextual information, starting with a flow to providing information.

Ajanki et al. developed a simple application that responded to different contexts (Ajanki et al. 2011). They provided a graphic that summarises the flow of AR interaction. The flow of control in the graphic was shown to connect in the order: Inferring Relevance, Context-sensitive Information Retrieval, Augmented Reality, Display Devices, Interaction, and then back to Inferring relevance. From this cycle, the relevance of the interaction is extracted, and context-sensitive information is retrieved. Then the information is produced as an overlay for the environment and displayed before waiting to prompt for further interaction. Their system demonstrated facial recognition, poster graphic recognition, and marker-based recognition features. The application understood the face of the principal researcher and was able to display contextual information alongside detailing their name and a note to go with it. As part of the poster summarising the research, one or more additional contextual patterns could be recognised. These showed a contextual tooltip with more detail. And finally, as another example, they also used a marker to show real-time reservations in a room. The process of inferring relevance and providing context-sensitive information could be directly applied to VR, thereby reducing information overload by only showing contextual information about a user's current activity related to objects considered to be in current focus.

Geospatial tags as a different type of specific contextual point are based on the proximity by GPS typically to determine the relevance of individual tags. In a space where many tags and AR information elements appear, there can quickly become a lot of clutter. Choi et al. suggested one way to simplify this issue (Choi et al. 2011). In their work, they worked on a clustering system. The system assumed that an interface could be simplified by grouping elements and fading out those deemed less relevant as a group. The study showed that grouping elements did reduce the overall processing required by the system. They trialled three different approaches: no grouping, manual grouping, automatic grouping. The ease of use was ranked very poorly (3/10) on average for no grouping, still quite poorly for the manual grouping (~4/10), and around a reasonable rating (6/10) for the automatic grouping. The concept of grouping was supported by the research of Dedual and Feiner (2013). A minimisation of the screen real estate through an AR interface was done by clustering information based on buildings. The information was directly associated with a building to give information a fixed point in space. Geospatial tags have a less direct link to VR as users are typically not walking around as freely compared to AR. Geospatial tags could be used to provide contextually relevant experiences where the user engages a VR system when they execute a VR experience within a range of a tag. The considerations toward clustering of information are also significant as it presents a way that could also be utilised to reduce information overload within a VR environment. The two examples represent existing scenarios in a much broader set of prior work by researchers. Labelling and clustering of data in this research context represent broadly how contextual information can be presented to users through the condensing of information. The small number of menu options used in the

PVMS experiments did not warrant any data shrinking. If necessary, appropriate techniques could be used to generate menu options if the need arose for more complex applications.

The representation of information or images in an accurate way that is useful to the user is essential. Khademi et al. considered the case of presenting information to the user and comparing how the size of that information is related to user preference (Khademi et al. 2012). The study aligned information directly related to a visibly recognised individual next to their face. In the first case, the information was presented at the same size regardless of distance from the target. In the second case, the information was scaled relative to the target distance. 80% of users felt it was more natural for the elements overlayed should be relative to the distance of the content on which they are providing information. The perception of depth, in this case, could be determined based on the relative size of expected dimensions for a facial region. The presentation used in the study is useful to consider when presenting information in VR. Letting users approach elements to discern their contextual relevance allows for automatic presentation based on distance from the user. The scaling levels of presented content is perhaps comparable to a level of detail (LOD) in 3D graphics where visual complexity is reduced based on distance (Luebke et al., 2003).

Users may wish to provide their own context to a situation. Sano demonstrated an example system where a user could create 3D rectangular objects by showing a camera the objects (Sano 2011). Those objects were assigned within the system to a displayed marker. Then whenever the marker was viewed, the user-generated 3D model would appear. A particular pair of markers was held up while capturing objects to accomplish this. These markers allowed the determination of the relative size to capture objects. Then once recorded, the object could be displayed through AR when viewing the other marker. In this case, the objects were inserted into a web database so that multiple users could use them. 17 of the 25 users said they found the process easy and enjoyable. Comments the users had about this system for improvements included: 10 said shapes needed more variety, size was not always accurate, 5 said scaling could help, 8 said it was difficult to understand the procedures, some found it excessive for the amount of time you had to stay still holding objects, 3 found it difficult to hold objects straight, four found printing the markers was inconvenient. Allowing users within virtual environments to control their own contextual information and perhaps share it with others could be used to drive a unique way to collaborate.

These examples of research have demonstrated a selection of considerations that can be included when developing VR applications. The proposed systems described in this research over the following chapters discuss the PVMS, which can offer contextual menu options. These could be further coupled with contextual elements within the world made up of the discussed qualities to provide relevant options to the user in a useful, organised way.

In summary, this broader section has presented an investigation of user interfaces and their evolution. The section has presented many aspects influencing what should be considered part of interface development for VR and AR in a HMD environment. The introduction explored the iteration from WIMP to NUIs and TUIs. The target area of this research is focused on the development of a NUI by controlling a HMD. Fitts' Law (Fitts and Posner, 1967; Scott MacKenzie, 1992) was influential in how interactive elements are presented as the user navigates between them.

Similarly, Hick's Law (Hick, 1952) was used in considering the complexity and number of options presented to the user. Section 2.4.1 discussed lists from three different papers influential on the design leading to the PVMS and other interface elements (Furness et al., 1995; Olsson et al., 2012; Schneiderman, 1992). The design considerations and rules impacted the intentionally easy to learn, reliable, consistent, and empowering interface exhibited by the final prototype solution presented in later chapters (Chapter 4 and experiments two and three). Due to the user's field of view being completely encompassed by the screen of the HMD, the types of interfaces discussed in 2.4.2 dictated that the PVMS should use world-space menus instead of more traditional 2D screen-space menus. And finally, in this last section for user interfaces, the condensing of information to show contextual information has been presented to establish the importance of only showing critical elements to the user given their current context. Some additional considerations about the design of the PVMS interface have been included as part of the PVMS technology overview chapter in section 4.3.1. The following section moves the discussion from interfaces to the tools for interacting with interfaces looking at the types of interactive controllers.

2.5 Interactive Controllers

Many different types of sensors exist to support different kinds of user input. Some examples of sensor categories are listed as identified by Zhou et al. (2008): Magnetic, Acoustic, Inertial, Optical, Mechanical.

Sensors can detect various forms of interaction, including movement, audible cues, changes in the environment, orientation, and GPS location. Together, they provide the input for common commercial products, including HMDs as described in 2.2.4 or those used for many gaming console-specific inputs. For AR, Normand et al. classified some examples of AR applications based on their tracking types (Normand et al., 2012). Classifying applications into 0D, 2D, 2D + θ, and 6D. Where 0D is purely based on features such as QR codes, 2D fit into location-based services, 2D + θ combine location with an orientation, and 6D allow full positional/rotational control in 3D space.

In this section, an overview covers some types of interactive controllers. The different types of controllers include handheld controllers, body-worn controllers, external controllers, and head-mounted controllers. The various controller types are significant to consider because they show how interaction has been provided alongside HMDs for VR or in ways appropriate for use with VR or AR. The discussion culminates into section 2.5.4, where the focus on 3DOF as a shared feature of HMDs is discussed, and then examples of how other researchers have implemented gestures as part of section 2.5.5. The following section presents some of the most prevalent types of controllers that users can hold in their hands to provide varying degrees of interaction.

2.5.1 Handheld Controllers

Handheld controllers cover a spectrum of devices that users would pick up and use with their hands. Examples discussed in this section include mobiles as controllers, glove controllers, the Wii remote, and other haptic handheld controllers. Before introducing examples from research, the following list broadly identifies examples of the different types of handheld controllers a user might interact with as part of an application.

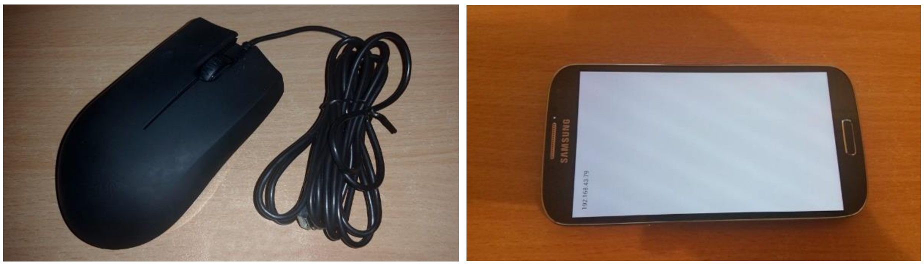


Figure 2.29: Computer Mouse and Mobile from Experiment 1 (Chapter 3)

- Computer Mouse (Figure 2.29)
- Mobile/Tablet (Figure 2.29)

Figure removed due to copyright restriction. See link for original: [Link]

Figure 2.30: Xbox 360 Controller[22]

Figure removed due to copyright restriction. See link for original: [Link]

Figure 2.31: PS4 Controller[23]

Figure removed due to copyright restriction. See link for original: [Link]

Figure 2.32: Wii Remote[24]

- Gaming Console controllers: Xbox Controller (Figure 2.30), PS4 Controller (Figure 2.31), Wii Remote (Figure 2.32) (and attachments e.g. nunchuk), etc.
- HMD handheld controllers: HTC Vive Controller (Figure 2.33), Oculus Touch (Figure 2.34)

Figure removed due to copyright restriction. See link for original: [Link]

Figure 2.33: HTC Vive Controller[25]

Figure removed due to copyright restriction. See link for original: [Link]

Figure 2.34: Oculus Touch Controller[26]

All the listed devices demonstrate examples of commercial products used for interaction. As part of the research presented in this dissertation, the computer mouse and a mobile device,

[22] Microsoft 2021, “Controllers & Remotes”, URL: https://www.xbox.com/en-AU/accessories, Last accessed 11/12/2021.

[23] Sony 2021, “Playstation 4”, URL: https://www.playstation.com/en-au/ps4/, Last accessed 11/12/2021.

[24] Nintendo 2021, “Nintendo”, URL: https://www.nintendo.com.au/, Last accessed 11/12/2021.

[25] Vive 2018, “HTC Vive”, URL: https://www.vive.com/au/product/, Last accessed 11/12/2021.

[26] Oculus 2018, “Oculus Rift”, URL: https://www.oculus.com/, Last accessed 11/12/2021.

as seen in Figure 2.29, were used in experiment 1 (Chapter 3). The Xbox controller seen in Figure 2.30 was used in experiment 3 (Chapter 6). These were all used for instant selection to contrast against hover-to-select interactions. The early stages of experiment design were considering utilising the power of a mobile device for many more of its sensors to be comparable to a Wii remote with the benefits of a dynamic touch screen button layout. This type of approach to utilising sensors in people's mobiles was removed to prevent scope creep. It is still worth considering the functionality a mobile device can provide, given that many people will have a mobile phone. Utilising this existing powerful technology could aid in reducing the barrier to entry to software. The following examples consider the feature set of mobile devices.

Mobile Devices as Controllers

Mobiles with their camera input, orientation sensors, GPS sensor, and widespread use make them significant devices for AR development. They package all the tools you could want for most simple AR applications. The accuracy of managing these sensors have been investigated by Guan et al., who, in their work, looked at using image recognition from a mobile device for determining a user's position in a large area (Guan et al., 2012). Or in other cases providing additional inputs for the device, as was the case in adding a camera to track the 3D movement of a mobile device in Pahud et al.'s work (Pahud et al., 2013). This section will primarily look at a few examples of places where mobile interaction has been explored as they incorporate most of the different sensor types into a convenient utility. These examples demonstrate additional ways that could be used to reference the position/orientation of a mobile device that may be useful for integration into a VR environment.

Bai and Lee investigated mobile touch screen interaction for AR (Bai and Lee, 2012). Their work suggested that a freeze view touch method could be used. They indicated users could have issues holding a device steady while interacting with a fixed pointing direction for the camera. To resolve this, the act of freezing the image when an interaction is beginning meant movement of the device could be for a short time independent of the camera without causing tracking issues. Another suggestion in their research was that automatic zooming could be used for feature manipulation, like the freeze-frame type system. Another more recent study was conducted by Vincent et al., who tried to reduce the jitter through filtering instead of using a freezing technique (Vincent et al., 2013). Their study found they could improve management of the artificial jitter, but it did not help significantly in any other form of jitter management. Freezing the screen may not be useful directly in the context of VR. Still, the idea of freezing objects within VR to perform modifications provides an interesting consideration when moving around a virtual world.

Chun and Hollerer investigated a different form of hand interaction using mobile phones; instead of using the touch screen for object interaction, they used hand tracking (Chun and Hollerer, 2013). Using the camera in a smartphone, they detected the fingers and demonstrated using them to translate and scale a virtual object displayed in the AR space. Object translation was achieved through the hand entering the scene from a direction, indicating a push from the entry direction. Scaling the object was done through the commonly used method for scaling on touch screens of pinching and un-pinching. These interactions could be considered a possible extension for interaction in VR if mobile were to be used as a primary interaction device.

Henrysson et al. investigated a method of object manipulation using mobile phones where the object can be tied to the mobile phone's position (Henrysson et al., 2005). In this study, an AR object with translation input was compared through being tied to the position of the phone, use of the keypad, and bilateral control. Some participants in their study indicated the object being tied to the mobile device made it feel like they were holding the object. The problem the researchers cited as a reason not to use this approach (at least in its current implementation) was that to perform a rotation instead of just a translation; the user would be required to move themselves around the object's position. Attaching virtual objects to the position of the mobile device within VR could provide a means of easily moving objects around.

Sambrook and Wilkinson presented a system titled HARATIO (Sambrooks and Wilkinson, 2016), where a mobile device could be used to interact with an AR environment anchored to a QR code. A freezing technique allowed independent modification and creation of objects with a radial menu and scripting language. The designs were constructed to enable novice users of AR and programming to create scripted scenarios within an AR environment. Inspiration for the radial menu used in experiment 1 and experiment 3 (Chapter 3 and Chapter 6) came from observing this research.

A selection of examples for mobiles has been presented to show how smartphones can extend interaction as a handheld input device. The examples included determination of a mobile’s position (Guan et al., 2012; Pahud et al., 2013), freezing of the device’s camera view for applying interactions (Bai and Lee, 2012), interpreting hand interactions as inputs via the camera (Chun and Hollerer, 2013), and object manipulation (Henrysson et al., 2005; Sambrooks and Wilkinson, 2016). The following moves from mobile examples to consider glove controllers as a different type of handheld controller that preceded the smartphone.

Glove Controllers

It is worth considering some of the controllers that were part of early VR interaction. Glove controllers were utilised in many of the earlier systems. Sturman and Zeltzer (1994) presented a survey of the different glove controllers. The glove controllers discussed in the paper included the Sayre glove, MIT LED glove, Digital Data Entry Glove, DataGlove, Dexterous HandMaster, Power Glove, CyberGlove, and Space Glove. The gloves provided a range of different input types for detecting the position of a hand and other types of input. Glove controllers have never taken off as commercial, consumer interaction devices, but the iterative developments associated with them have led to the implementation of other currently used controllers.

Dipetro et al. (2008) also surveyed glove controllers and discussed the appropriateness of using glove devices and the limitations of the technology. One of the primary considerations presented was to evaluate whether another comparable device could provide the glove's function. Many earlier glove inputs offered similar functionality to a 3D joystick controller. Three questions suggested by Dipetro et al. when determining if a glove would be appropriate for a context included: “Are there natural ways to use the hand in the application?”, “Are there many different tasks to switch between?” and “Do the tasks require coordination of many degrees of freedom?”. Limitations were suggested to include the portability, the limitation of a user’s haptic sense and naturalness of movement, poor robustness, poor durability, need for calibration, and high cost. Many of the features offered by a glove controller can be generalised to use detection via devices such as the Leap Motion controller. Glove controllers would likely benefit a user most where highly accurate tracking of hand inputs is necessary, or it is desirable to experience haptic feedback. For these types of controllers to be accepted by consumers, the cost would need to be proportional to capabilities with applications designed to support them.

Wii Remote Controller

The Wii Remote (released in 2006) is an example of a popular commercial device capable of detecting changes in motion, allowing it to be used for gesture recognition and development toward the controllers used with the HTC Vive and Oculus Rift. An example of research using a Wii remote was conducted by Oda and Feiner (2012). They used a modified Wi Remote with optical tracking markers to determine a fixed position in space (Oda and Feiner, 2012). This system allowed users to point with the device at physical objects and have the camera track the direction where the remote was being aimed. Their study focused on object selection, but a tool demonstrated in this research could be used for further object manipulation in other uses.

Haptic Handheld Controllers

The last few examples of handheld controllers will discuss some of the haptic handheld controllers that have been developed. Haptic controllers can enhance immersion by creating feelings when interacting with objects in the virtual world. The first example is Benko et al. (2016), who developed two separate handheld controllers. The first was a controller titled NormalTouch; this controller sensed the force of input from touch to change how a tiltable and extrudable platform would be manipulated. The second was called TextureTouch that provided a tactile surface made up of a 4x4 array of actuated pins. There were no significant differences found in comparing the accuracy between the two controllers. Both offered successful accuracy for interacting with targets in virtual environments.

Lee et al. (2019) investigated the use of a VR controller for in-hand, high-dexterity, finger interactions titled TORC. The controller allowed for precise manipulation of objects through position and rotation changes. Some functional scenarios included grasping and releasing virtual objects, object deformation for object elasticity based on squeezing, textured feeling, and precise manipulation by sensing changes in finger motion. The controller had the user holding a wand-style controller with specific placement of two fingertips into a Velcro grip. The thumb was placed on the other side to control pressure and force while moving over a 2D trackpad.

Another device also provided a similar type of experience in collaboration with many of the same researchers in the research by Choi et al. (2018). The CLAW haptic controller provided an experience where a user would place their index finger into a grip with a force sensor attached. Once mounted, the index finger could be used to grasp and touch objects within a 3D environment. The device could also perform shoot actions, with the interaction being similar to squeezing the trigger on a gun.

Kovacs et al. (2020) presented a device titled PIVOT. This device varied from the other two previous examples where the haptic element was likely always touched even if an object was not currently interacting within the application. PIVOT utilised a wrist-worn controller, allowing hand tracking with on-demand haptic feedback. The haptic feedback was provided through a grip that would pivot from further down the arm into the hand. Once in hand, it could be clasped to give feedback from the interaction. The position of the haptic element left the hands free for other actions, with the advantage of, when necessary, providing haptic forces to simulate gravity, inertia, and air drag. Examples of interactions included catching a falling apple, touching and clasping objects, catching and throwing objects, and feeling the wiggling of objects (such as holding a rabbit).

Handheld controllers provide a tangible object for the user to grasp, making them feel like they are engaging directly with a virtual world. Haptics delivered through rumble in console controllers or feedback from the examples of specific haptic controllers from the research discussed help immerse users. Fingers are capable of many different types of interaction that can be performed through touch. A tangible button press gives a reliable experience for the user knowing their input has been entered compared to a gesture. As development moves toward NUIs, there have been an expanding number of alternative niche controllers. This dissertation considers using input that moves away from the highly saturated research with handheld inputs to use a HMD as input instead. As part of the first experiment, two types of handheld controllers (mouse and mobile) were used to provide direct contrast for a physical selection action against using the head as an interaction tool. Similarly, the third experiment used an Xbox controller to compare HMD Only input against a tap-to-select type interaction with the proposed PVMS. Sambrooks and Wilkinson (2016) inspired the implementation of a radial menu (referred to as a circular menu) used in the first and third experiments. The menu acts distinctly differently as a world-space VR menu compared to the screen-space mobile implementation in the paper. The circular menu is used to compare against the PVMS. Additionally, the circular menu is considered an example menu that can exist alongside the PVMS to expand the options for interaction.

The handheld controllers demonstrated in this section have shown a small selection of the many different types of controllers developed to innovate new ways for VR interaction. In the next section, additional controllers will be explored that interact directly with the body but are not held in a user's hands.

2.5.2 Body-worn Controllers

The few controllers identified as examples for body-worn types in this section represent vast fields of study. If the research focus were on any of the areas these devices are attributed to, they would warrant separate extensive discussion. Further to the haptics for handheld controllers presented in the previous section, the first controllers explained here identify haptics and interaction directly with the body that is not related to the HMD nor from handheld controllers. As separate extensions to the HMD controllers discussed later in section 2.5.4, the addition of other head-worn controllers can be considered body-worn and distinct from the HMD context because they are not universal for HMDs. Specifically incorporation of hand detection via attaching an extension to the body (Leap Motion controller) and combining a brain-computer interface with a HMD.

Figure removed due to
copyright restriction.
See link for original:
[Link]

Figure 2.35: Virtuix Omni[27]

There are a growing number of devices designed to improve the immersive elements of VR. One is the extension to allow movement (locomotion) in virtual space without actually moving in real life. Typically, this sort of action could be accomplished with a joystick or other similar

[27] Virtuix 2015, "Virtuix Omni", URL: http://www.virtuix.com/, Last accessed 11/12/2021.

input to control player movement. Devices such as the Virtuix Omni have been developed (Virtuix, 2015) to provide users with a more immersive and engaging interface. The Virtuix Omni allows players to simulate moving around their environment by physically moving their feet. An example of this device can be seen in Figure 2.35. The user is centred within the device, and foot movement in a direction can be used by applications to translate into motion within the virtual world.

Figure removed due to
copyright restriction.
See link for original:
[Link]

Figure 2.36: Hardlight Suit[28]

Simple haptic feedback in VR systems can be accommodated through vibration in handheld controllers. Some technology is designed to provide this haptic feedback using equipment worn that will make the user feel like something has happened. An example of this is the Hardlight Suit (Sinko and NullSpace VR, 2018) in . This body armour extends the body's tracking beyond just the head's position and any supported controllers to the chest and arms. Not only this, but the suit’s body armour will allow the user to experience haptic feedback in games. It becomes possible to feel a sensation when you are shot or hit with a sword within the virtual world. Another example of haptic feedback for VR is the research by Lopes et al. (Lopes et al., 2017). Their study tested haptic technology that allows you to feel feedback when interacting with virtual walls or heavy objects with electrical stimulation. The two types of stimuli, delivered via pads on the arms to represent walls, were a soft surface technique and repulsion technique. The soft surface was represented as a magnetic field that provided electrical stimulus to allow penetration of the surface but stimulated a desire to remove the hand gradually. The repulsion type walls were presented electrified walls that would jolt the user’s muscles combined with visual stimulus to make the user know they should not touch

[28] Sinko M. and NullSpace VR 2018, “Kickstarter: Post mortem report and the conclusion of Hardlight”, URL: https://www.kickstarter.com/projects/morgansinko/hardlight-vr-suit-dont-just-play-the-game-feel-it, Last accessed 11/12/2021.

there. The work also demonstrated interaction with objects, including lifting, punching and throwing. Each involved electrical stimulus delivered similarly to make the user feel the weight of their interaction with the application. The remaining examples shift from physical movement detection and haptic feedback to extensions of inputs that can be attached to a user's head.

Despite the similar name, the LEAP Motion controller exists separately from the Magic Leap One HMD (Magic Leap, 2018). The LEAP Motion controller was developed to provide hand tracking (LEAP Motion, 2013), as shown in Figure 2.37. The LEAP Motion controller could be considered a body-worn controller as it could be attached to a HMD. The controller could also be left to sit freely in a separate location. The controller would allow tracking of the hands for use with VR or other applications.

Figure removed due to
copyright restriction.
See link for original:
[Link]

Figure 2.37: LEAP Motion on Oculus DK2, and separately (bottom right)[29]

Figure removed due to
copyright restriction.
See link for original:
[Link]

Figure 2.38: Emotiv EPOC[30]

Another example of a body-worn controller would be a device that allows brain-computer interaction. The Emotiv EPOC controller (Emotiv, 2018) shown in Figure 2.38 is one such device. Salisbury et al. (2016) combined VR with brain-computer interfaces (using the Emotiv EPOC) to assist in neurorehabilitation of patients who had suffered spinal cord and brain injuries. Their study suggested some feasibility but spoke of the cost-based difficulties of establishing proper testing, how it factored into costs for training and indicated the future success of this type of use would come from additional future testing.

This section has presented examples of body-worn controllers. These show how the field is advancing by combining additional controllers attached in different places to the body that

[29] Leap Motion 2016, "Leap Motion VR Mount + Oculus Rift CV1", URL: https://www.youtube.com/watch?v=OUdL3y-mrFM, Last accessed 11/12/2021.

[30] Emotiv 2018, "Emotiv Epoc+", URL: https://www.emotiv.com/epoc/, Last accessed 11/12/2021.

can augment a user’s experience within virtual worlds. The controllers that have been presented in this section all could exist alongside other types of controllers to create a rich interactive experience. The following section considers controllers that are separate from the body as a different type of engagement that differs from the formerly discussed handheld controllers and the body-worn controllers of this section.

2.5.3 External Controllers

External controllers include any controller where the user is either not directly in contact with a device or has a much larger interaction surface not attached to the user. Examples of external controllers include the Microsoft Kinect and the base stations included with the HTC Vive (Vive, 2018). The Microsoft Kinect and other similar devices track users with depth cameras, infrared sensors or other methods of detecting elements of interest. In the case of the Microsoft Kinect, one of the ways it could be used would be to track the skeleton of users. This skeleton could allow the user’s physical actions to translate into the application. In the case of the HTC Vive, tracking accuracy is based on many smaller sensors on the HMD and controllers. The base stations sweep over the area created by the user interacting with the sensors on the devices, allowing accurate tracking within the defined area. This tracking form is known as the Lighthouse technique (Vive, 2018). As part of this technique, two base stations sweep the area by emitting infrared pulses at 60Hz detected by the HTC Vive headset and handheld controllers.

Tabletop systems could be considered external controllers due to their separation from being attached to the user. As part of the background research, their consideration was driven by the initial desire to represent VR space for experimentation as a virtual tabletop experience. The following few examples illustrate some of this inspiration. The first experiments layout (in Chapter 3) was influenced by a tabletop as an interaction surface but was not continued into any significant representation for the later experiments. There is the possibility of introducing overlap between a user controlling a virtual space with VR while interacting on a tabletop controller either with others in VR or so that external users can still provide inputs or observation.

Tabletop systems are particularly suited for real-time telepresence interactions between multiple people. Like a VR environment, a tabletop provides a fixed position that can be used as a relative point for all participants. Tabletop interfaces can include overlays on existing physical interfaces. For example, Liu et al. designed a hand-held mobile application that could be used to view a MIDI controller (a type of sound interface) with an AR overlay (Liu et al., 2012). The overlay of this tabletop system gives visual cues indicating the modifications to settings that should be implemented. The main advantage of applications like this is that they can reduce many user errors that could be present or provide training to show the correct way to configure systems.

The combination of physical elements can be used in multiplayer tabletop games as well. Morde et al. designed a game system for playing chess using tabletop AR (Morde et al., 2004). A novice user would play on a physical game board in their system and view the other player's pieces through AR. The expert player would play from an entirely virtual environment application. Moving the physical pieces would translate over into VR and AR, demonstrating an example of determining a relative position for game objects that indicates how they can perform additional moves within the context of the game. Since all components were on a chessboard, it gave the system a relative positioning and scaling method.

Rodrigues et al. developed a multitouch tabletop type system (Rodrigues et al., 2012). In their system, they used markers that could be manipulated on the surface of the projected tabletop. The markers could be translated, rotated, and scaled to control object presentation through the AR interface. This marker manipulation technique could be used with multiple users to expand the potential capabilities of the devices for more applications.

Lee et al. designed a tangible AR interface using occlusion (Lee et al., 2004). In their work, interactions could be performed with a grid or line of markers. The marker's visibility was used to determine a state for that cell, changing the game state. Actions such as selection, drag and drop, and object pushing were demonstrated using the grid of markers.

Tracking hands within tabletop interaction environments is important for making the most of available inputs. Corbett-Davies et al. created a system using the Microsoft Kinect to track objects that could be interacted with hands to manipulate virtual objects (Corbett-Davies et al., 2012). In their system, the depth sensor from the Microsoft Kinect was used to detect the depth of elements and enabled hand interaction with these objects.

Each of these examples related to tabletop controller systems could overlap with VR. Primarily by providing a shared interaction/observation surface for multiple users with one or more using HMDs and some users observing the projected tabletop version. Alternatively, by presenting an entirely virtual tabletop system, it could be virtualised with haptic reactions to simulate touch with the advantage of offering interaction in a familiar setting. Although it was not utilised for later experiments, presenting a virtual tabletop environment was considered part of the first experiment's design. The following section returns to a more specific type of body-worn controller directly related to the VR prototyping performed in this dissertation focusing on the head-mounted controller.

2.5.4 Head-Mounted Controllers

In section 2.2.4, the HMDs relevant when this research was conducted were identified, including details about interaction provided by the HMDs. Importantly the devices all provided at least 3DOF with rotation, with the more recent models providing 6DOF. 3DOF is principally offered in HMDs by combining a gyroscope and accelerometer. Gyroscopes measure orientation and angular velocity, and accelerometers measure acceleration forces acting on an object. The following list identifies the differences between the different devices separated by 3DOF and 6DOF, including how the devices sense the interaction.

- **3DOF**:
  - **Google Cardboard** and **Samsung Gear VR** (Google, 2014; Samsung, 2015): Mobile devices used for these HMDs contain at least an accelerometer and gyroscope.
  - **Oculus Rift Dev Kit 1**: Provided the 3DOF with a combination of a gyroscope, accelerometer, and magnetometer sensors (Rift Info, 2016); the magnetometer added another metric for validating orientation sensing by measuring magnetic fields like a compass.
- **6DOF**:
  - **Oculus Rift Dev Kit 2**: 3DOF from the same sensors as Dev Kit 1 with the option to attach an infrared sensor to swap to 6DOF (Rift Info, 2016).
  - **Oculus Rift (consumer version)**: 3DOF from the same sensors as Dev Kit 1 and 2. In this version of the Oculus Rift, the “Oculus Sensor” was used to determine

the additional degrees of freedom. Additionally, the hand-held Oculus Touch controllers provided an extra dimension of input (Oculus, 2018).

- **HTC Vive**: 3DOF from a gyroscope and accelerometer increases to 6DOF using laser position sensors to pair with base stations. The HTC Vive also came with a pair of hand-held controllers that used the same approach to provide consistent tracking (Vive, 2018).
- **Playstation VR**: 3DOF from an accelerometer and a gyroscope, with 6DOF via an optical 360^0 LED system like the Oculus Rift (Sony, 2018).

The essential information from this summary is the support for at least 3DOF from each device. Interaction techniques discussed in this dissertation will rely on the degrees of freedom as a tool for enabling any HMDs to benefit from this research. The shared property presents a significant area for investigating interaction when considering interacting with the HMD as an independent device. Researchers have investigated head interactions, but studies found while surveying the field focused on specific interactions without providing general-purpose solutions. More commonly, the investigations focused on incorporating additional controllers. The PVMS presents a prototype solution using 3DOF to take advantage of the shared properties of HMDs that could be used universally by developers for HMDs. The rest of this section will explore additional examples of research related to using head-mounted controllers with consideration for the user of controllers and VR sickness.

While using head-mounted controllers that partially or fully obscure a user's vision, it is important to consider usability aspects while developing software. McGill et al. considered three different situations that could impact usability (McGill et al., 2015). The three usability situations considered were: keyboard use with a HMD on, varying levels of blending with reality comparing a range of minimal, partial, and full blending, and the effect of the presence of others in the vicinity. Newer HMDs are more likely to be more reliable from iterative design. During the first experiment (Chapter 3), some participants experienced errors with rotation as a usability issue. The rotation not lining up with the user's expectation was considered by Zhang and Kuhl (Zang and Kuhl, 2013). These issues did not continue with the second experiment using Oculus Rift Dev Kit 2 instead of version 1.

While HMDs provide an immersive experience required for VR, the use of HMDs, for some users, has also caused adverse physical reactions, namely VR sickness. VR sickness is similar to the response some people have to motion sickness. Tanaka and Takagi (2004) described the symptoms of VR sickness could include headache, vertigo, and nausea. Their research investigated how the velocity and visual angle of content related to VR sickness. They applied a neural network model that improved the worst case for recovery from VR sickness down from 60 minutes to 5 minutes. Chang et al. (2020) summarises many factors influencing VR sickness and describes some emerging approaches to reducing VR sickness. These include using a dynamic depth of field, adding additional sensory information (auditory, olfactory, and tactile), and improving visual fidelity. Munafo et al. (2017) observed the effects of VR sickness from their tests using the Oculus Rift disproportionately negatively affected females. The negative impact observed on females by Munafo et al. was not supported with statistical significance. Still, it is essential to consider the implications of VR sickness in each application's design.

In all three experiments detailed in Chapter 3, Chapter 5, and Chapter 6, the HMD was used significantly as a controller, focusing on HMD Only interaction compared with hand-held controllers. The discussion in this section has broadly presented the tools for interaction using HMD controllers, where the most important part of this research are the overlaps in orientation tracking. 3DOF has been identified as a shared feature of all HMDs and, therefore, can be utilised to provide interaction across any HMD devices. For this research, the prototyping was conducted using the Oculus Rift Dev Kit 1 (first experiment) and Oculus Rift Dev Kit 2 (second and third experiments). The research area of gestures was investigated to use the 3DOF in a useful way. The following section explores this use of gestures and identifies inspirational work that led to the PVMS.

2.5.5 Gestures

In this section, some of the different examples of gestures are explored in research. Gestures provide a specific subset of interactions with many different input methods. Due to the types of interactions discussed in this dissertation, they have been included here separately from other types of interaction. Gestures vary in complexity depending on the application. Most gestures involve tracking a moving interaction of one or more points through 2D or 3D space until a condition is reached. The examples begin with full-body and hand gestures, then some discussion about using dwell time and gaze tracking, finishing with head gestures. A summary is provided at the end specific to gestures and a broader summarising of the entire interactive controller section and its impact on the research.

Full Body and Hand Gestures

Roupé et al. used full body gestures controlled through the Microsoft Kinect to navigate in VR (Roupé et al., 2014). Body poses included a calibration pose, forward and backward leaning for movement, right and left turning by rotating shoulders, and swapping modes by holding out an arm. The pose-based gestures used in this research were reported to be easy and not demanding when compared to a standard keyboard and mouse input. Full body type detection requires monitoring devices positioned to view all body parts necessary for detecting actions. With space limitations and the ability to move freely as considerations, this leads to focusing on hand gestures as a lower barrier for developers and users to use gestures for interaction.

Marin et al. investigated the use of hand gestures detected by the Leap Motion and Microsoft Kinect sensors (Marin et al., 2014). The two sensors were used simultaneously to compare hand gesture detection for accuracy. The research found that the Leap Motion sensor provided a higher level of data with limited description compared to the full depth map of the Kinect. The Leap Motion sensor provided data on fingertip distances, angles and elevations. The Kinect sensor provided data, including the 3D structure of the detected hand, but with less accuracy. For gesture recognition used in the study, the different devices were evaluated for accuracy independently and together. The best accuracy for Leap Motion was 80.86%, using all three finger features, and 76.07% for fingertip distances as the best single indicator. The Kinect performed better individually with 87.28% using a curvature descriptor evaluation, and when combining both curvature and correlation, 89.71% accuracy was achieved. Combining both sensors to evaluate gesture recognition had a 91.28% accuracy. This result demonstrates an example of how combining feature sets from multiple devices can improve gesture detection.

Petry and Huber combined the Oculus Rift and Leap Motion controllers to play omnidirectional videos with hand gesture inputs (Petry and Huber, 2015). Omnidirectional videos are panoramic and can cover up to 360° of a user's view. In this research, hand gestures were used to allow for temporal navigation. The head was used independently for spatial navigation. The authors indicated the interactions were only mapped to a single hand and acknowledged that tracking a second hand could add a lot of other potential possibilities for interaction.

Colaco et al. built a hand gesture-based device called Mime (Colaco et al., 2013a; Colaco, 2013b). The Mime sensor is claimed to have the advantages of being small, supporting daylight sensitivity, with low power consumption, and using low-cost components with comparisons drawn against the Microsoft Kinect, LEAP Motion controller, and HMDs. The gesture controller was incorporated into smart glasses, using 3D position estimation to determine hand gestures for interaction with 2D interface elements.

Serrano et al. tested the use of hand to face input as a form of interaction with head-worn displays (Serrano et al., 2014). The user would use their finger on their cheek to create tactile actions of panning, pinch zooming, cyclo zooming, and rotation zooming. The acceptance of different positions for gestures was measured on the participants using a mock device, followed by an implementation of the finger to face interaction with testing that utilised the cheek as an interaction surface for gestures.

A variation on hand gestures compared to directly mapping hands or fingers for interpretation can include holding a controller or camera that acts as a controller. Lagerstam et al. experimented with pseudo mobile AR interactions with children and an animated character (Lagerstam et al., 2012). This research used a USB camera connected to a laptop as a gesture control tool. A marker would represent where the animated dog character would appear. Five types of responsive gesture input caused different reactions from the animated character. These were: the dog's head would always face the camera when the camera got too close, the dog would scratch the ground and growl, vertical shaking would cause an animated sequence of jumping while barking, and circular motion would trigger a rollover animation. This study is interesting to consider due to the issues encountered in gesture recognition. The recognition of gestures used image recognition that caused difficulty in detection at some times from loss of the optical tracking from motion blur or moving too far from the marker used for detection. The other type of issue experienced in the study was the difficulty in adapting to different movement styles of users, given the freeform control of the camera. It presented difficulty distinguishing between normal camera navigation with movements intended as gestures.

Hand gestures provide a natural way for users to indicate an action. Incorporating cameras and other sensors suitable for hand detection into AR HMDs (e.g. HoloLens) means the ability for users to input actions with hand gestures can provide a convenient NUI experience. Full body type gestures can be practical with sensors such as the Microsoft Kinect (Roupé et al., 2014), but space is necessary if actions require a lot of movement. For most users, it is not practical to combine multiple external sensors like was done by Marin et al. (2014) with the Leap Motion and Microsoft Kinect. More realistic would be to combine the features used for recognition into a single device to simplify the experience for users. Hands provide a useful tool for scrolling through content such as a temporal sequence, as Petry and Huber (2015) demonstrated, in addition to using head movement for spatial navigation. Colaco et al. (2013a; 2013b) showed that development is ongoing to create improved sensors for hand interaction. Hands can be used as a gesture against the face with appropriate detection, as shown by Serrano et al. (2014). Lagerstam et al. (2012) demonstrated a handheld controller that, with a camera, detected gestures. This type of interaction is similar to wearing a camera attached to a HMD, but with the freedom to move independently of a user's body. It is necessary to include additional sensors to track hand interactions and therefore is not always available for all HMDs. Hand interactions could be considered additive for selection and interaction if available, but for this dissertation, the focus is on providing a HMD Only type experience.

Dwell Time and Gaze Gestures

Dwell time, and gaze tracking can be classified as a type of gesture in how a user interacts. Many researchers have investigated their uses for interaction (Chennamma and Yuan, 2013; Stiefelhagen et al., 1997; Morimoto and Mimica, 2005; Zhu and Ji, 2004). Dwell time is used as a command without a click by applying focused visual attention at a specific point of interest for a short, defined period of time (Hansen et al., 2003). The use of gaze tracking extends to applications of behavioural analysis with tracking what a user is reading (Kim et al., 2014). Gaze tracking is not as relevant to the proposed PVMS because the HMD is used as a pointer controlled by the entire head with 3DOF instead of individual eyes with gaze tracking. Dwell time is fundamental to how gaze tracking works and is therefore relevant to consider. More specifically, concerning timings and button sizes with dwell times, Penkar et al. (2012) found that it was better to have large buttons with long dwell times (around 1 second) when placing text on buttons.

Špakov and Majaranta combined gaze pointing and head gestures to improve dwell time (Špakov and Majaranta, 2012). The study suggested that nodding as a gesture for interaction provided more convenient interaction than the standard dwell operation. The paper indicated a more direct comparison between dwell time and the nodding gesture would be conducted in future.

Dwell time is fundamental to the selection used with HMD Only type interaction in this dissertation. While gaze tracking is relevant for consideration historically, HMDs do not natively track gaze. The mechanism of turning a user's head to aim the forward-facing direction centrally focused on an element is comparable to that of using eyes to look directly at an object. For selection within HMD Only systems, a user can perform a look at action by turning their head to face directly at the element they wish to select and then dwell for some time to confirm the selection of that element. A nod for confirming interaction, as suggested by Špakov and Majaranta (2012), would likely not perform as well in the context of the PVMS because it is preferable to keep the user's focus near a point of interest. The nod could be considered comparable to the proposed reveal mechanic for the PVMS with a rapid head turn to reveal the menu (discussed further in section 4.4.1). A nod is typically up and down, but for the PVMS, it can support interactions up, down, left and right. The following examples consider some additional interactions specifically using head gestures.

Head Gestures

Morency and Darrel investigated a prototype head gesture detection device for dialog box confirmation and document browsing (Morency and Darrel, 2006). A stereo input with an SVM-based classifier was used to recognise gestures. The study found users benefitted from the proposed system over conventional alternatives. When participants were given the freedom to choose out of the head gesture, mouse, or keyboard inputs in the last step of the experiment, 60% preferred the head gesture for dialog box confirmation. Participants preferred the keyboard for document browsing, with 45.8% selecting the keyboard and 31.2% choosing the head gesture, with participants citing that they desired more control when performing that interaction. Their results showed favourably that the head interaction was suitable and preferred for simple interactions.

Hirsch et al. developed a smart textile neck brace for detecting head gestures (Hirsch et al., 2014). The prototype sensor hardware was tested with 15 different types of gestures. Using combinations of nod, tilt, look, circle, and woodpecker (forward head movement) type gestures. This neck brace could capture the type of interactions similar to HMDs without the screen being incorporated.

Jackowski et al. used head gestures as a tool for hands-free control of a robot (Jackowski, 2016). In their work, the automation of a robot arm was manipulated with 4 types of control groups. Head gestures would control open/close gripper operations, orientation changes, vertical plane movement, and horizontal plane movement. Five steps were used for gesture recognition, including pre-processing, segmentation, feature extraction, dimensionality reduction, and classification.

Ruban and Wood investigated head gesture interactions using k-Nearest-Neighbour and Dynamic Time Warping (Ruban and Wood, 2016). The k-Nearest-Neighbour algorithm is used for classification and regression by considering the k closest training examples. Dynamic Time Warping is used to measure similarity between two temporal sequences. In their experiment, the Oculus Rift DK2 was used to detect yes, no, or null responses with measurements of acceleration, angular velocity, and rotation. A nod was used to represent a yes response, and shaking the head horizontally would represent a no. The way this system was designed would allow for other additional head gestures to be defined by users in the future.

Head gestures provide an interesting domain for investigation because as users adopt HMDs with continually increased capabilities, the HMD itself presents an option for interaction. Morency and Darrel (2006) found that participants preferred to use HMD gestures for simple tasks (e.g. confirmation compared to document navigation). Hirsch et al. (2014) and Jackowski et al. (2016) demonstrated gestures to perform specific operations with various types of head movement. And as an example of the Oculus Rift DK2 for a head gesture tool, Ruban and Wood (2016) used the properties of the HMD to detect a nod or headshake for confirmation.

At the outset of the investigation into the use of HMDs and using the head as an interaction tool, it was evident that the knowledge gap was in providing a combination of menu system with a gesture. The information found from surveying research indicated examples of general use of HMDs with adding handheld controllers or other types of controllers. The interactions found specific to HMDs were targeted applications that dealt with detecting a gesture as an independent evaluation (e.g. Ruban and Wood, 2016; Morency and Darrel, 2006) rather than as part of a suitable menu solution. Based on the review of the discussed gesture related material and other interaction controller material, it was decided that the gesture should be a quick turn to the left or right with additional support for up or down.

The gestures presented have demonstrated a range of examples relevant to determining how gestures could be used for HMD Only interaction. Starting with more focus on separate devices such as the Microsoft Kinect and leading to techniques for using gestures for interacting within HMD environments. The interactions presented were all considered in designing the PVMS's gesture technique and its focus on supporting developers and users by restricting required hardware to just the sensors on the HMD.

In summary, many controllers have been explored for the interactive controllers section as a whole. The first section (2.5.1) explored handheld controllers. Users are used to tangible inputs that they can grasp, and handheld controllers give the user precise agency over the timing of a trigger. As shown in the section, many handheld controllers are targeted toward game consoles and can be used as familiar input devices as part of other domains. Mobile, mouse and Xbox controllers were used for comparison in the experiments as tap-to-select type interactions. The circular menu was inspired by the radial menu used as part of an AR phone application designed by Sambrooks and Wilkinson (2016). In section 2.5.2, the discussion moved to present body-worn controllers as examples of extensions that can provide haptic auxiliary interaction or extension through head-mounted hand detection or brain-computer interfaces combined with a HMD.

Section 2.5.3 showed examples mostly related to tabletop interaction systems as external controllers. The main message from that section was discussing an initial development direction of condensing an experience into a virtual tabletop. This type of experience was used to design the first experiment's experience in Chapter 3. The discussion for types of interaction controllers surveyed culminated in section 2.5.4 that began by summarising the interaction experience of 3DOF and 6DOF provided by the HMDs described earlier in section 2.2.4. Significantly now, as gestures have been discussed, the basis for interaction was focused on utilising 3DOF to detect gestures with rotation. Using this specific common under-utilised technique increases the number of options available to HMD users.

Further to the discussion about motion sickness in section 2.2.4, section 2.5.4 continued the discussion about the experience of users of HMDs as essential considerations for the prototype experience. Finally, more specific examples from this section on gestures were presented to establish the types of interaction that other researchers have investigated. In particular, examples of using a nod (Morency and Darrel, 2006) or headshake (Ruban and Wood, 2016) to trigger an action. The interactions with head gestures surveyed were limited to specific use case examples and did not appear to be investigated for more significant HMD Only type interaction with menus. Combining the gesture of a head turn to make a menu appear and then a dwell to select (referred to as hover-to-select) menu options provided a unique area for investigation in combination with NUI techniques discussed in section 2.4. The following section presents a secondary area of investigation considered as part of this dissertation in how research can be improved with serious games.

2.6 Serious Games

For the research presented in this dissertation, games have been used as a tool for data collection about user interactions and to incentivise participants toward participation. Rather than provide just mundane tasks, the experiments were designed to provide a game experience to make participation more engaging. The use of games for a purpose other than entertainment is referred to as serious games.

Figure removed due to copyright restriction.
See Laamarti et al. 2014 for original.
Figure showed categories with Application area, Activity, Modality, Interaction Style, and Environment. Each with sub-elements.

Figure 2.39: Serious Games Taxonomy (Laamarti et al., 2014)

In an overview of serious games by Laamarti et al., a taxonomy is presented, breaking up the features that may go into examples of serious games (Laamarti et al., 2014). In (previous page), the taxonomy presented from the article by Laamarti et al. is shown. The taxonomy broadly demonstrates an example of how varied these serious games can be in their features. Serious games are defined by their overlaps between entertainment, multimedia, and experience (Laamarti et al., 2014).

Education and training can be facilitated by providing an engaging environment to immerse users in the topic they are learning (De Gloria et al., 2014). This learning approach offers an alternate approach to simply reading about an activity. Romero et al. found that many current serious games do not contribute broadly to 21st-century skills, but instead, they are domain-specific (Romero et al., 2014). An example of a domain-specific area could be historical education. In work by Mortara et al., the domain of learning cultural heritage with serious games was explored (Mortara et al., 2014). Their work focused on cultural awareness, historical reconstruction, and heritage awareness, exploring how serious games can be applied to these areas.

Figure removed due to
copyright restriction.
See footnote for original.

Figure 2.40: OrbIT Controller[31]

In the health domain, there are many applications for games as a tool for rehabilitation (Lange et al., 2012) and improving the wellbeing of users. In an article by Fleming et al., some areas benefiting mental health are explored (Fleming et al., 2017). Some of those areas covered were: exergames (exercise gaming), VR games, cognitive behaviour therapy games, biofeedback games, cognitive training games, and entertainment games. An example of a health targeted serious game tool would be the OrbIT (Henschke et al., 2012). The OrbIT, shown in , is a controller designed for users with cerebral palsy and can assist with the therapy

[31] Hobbs D. 2017, “Game therapy: serious video games can help children with cerebral palsy”, TheConversation, URL: https://theconversation.com/game-therapy-serious-video-games-can-help-children-with-cerebral-palsy-72950, Last accessed 11/12/2021.

of users with this condition. The controller aids in treatment by encouraging tactile engagement with both hands and has been found to improve the non-dominant hand use in cerebral palsy users who participated in this study.

Figure removed due to
copyright restriction.
See link for original:
[Link]

Figure 2.41: America's Army Game[32]

An example of using games as an advertisement is America's Army (Parkin, 2015; Zyda, 2005), shown in . This game was developed as a recruitment tool for the US Army and was released in 2002. The use of games for recruitment is only one area where they can be found within the military domain. The area of training in many different fields can be aided by serious games (Lim and Jung, 2013). Serious games allow the simulation of activities that may otherwise put users in dangerous or impossible situations. VR and AR improvements continue to improve the quality of immersion possible for these training situations and, more broadly, for all serious games.

Using games as a motivator and recruitment tool for research participation has been implemented as part of the research presented in this dissertation. The use of serious games for recruitment applies mostly to the second (Chapter 5) and third experiments (Chapter 6). The specifics of how serious games were used is discussed in each chapter. It is believed that the use of serious games to help drive research goals will aid future research projects more broadly within the research community. This topic has presented the last area of investigation for the background. The following section will conclude the background, summarising the information presented throughout the chapter.

[32] American's Army 2021, "America's Army", URL: https://www.americasarmy.com/, Last accessed 11/12/2021.

2.7 Conclusion

The background has presented many examples of research and technologies. The examples informed and influenced the research, design and implementation of the experiments and prototyping of the PVMS presented in this dissertation. The chapter began by introducing VR and AR to provide a foundation for the research that was conducted. Display technologies were surveyed to establish the types of presentation techniques used, which informed the HMDs used as the focus for development. This section was followed with many different examples that have been categorised based on the type of activity, including health, education, art, collaboration, computer-assisted activities, pervasive applications, and finishing with some examples of games in research and the commercial markets. Interfaces were evaluated from a historical standpoint and discussed how interfaces presented within HMDs could provide experiences suitable to the technologies. After identifying HMD technologies, the background led to a discussion on the types of interactive controllers that could interact with interfaces, focusing on virtual environments and gestures relevant to the PVMS. Serious games were discussed to provide context to their use in experiment 2 and experiment 3, both as part of the experiment design and how the YouTube trailers were used to drive recruitment.

The research gap has been identified from summaries across the background sections. Section 2.2 presented the different types of displays and demonstrated that HMDs have become more viable as tools for VR and AR. Devices such as the Oculus Rift and HTC Vive as commercial HMDs presented experiences superior to previous VR supported by other newer computing hardware to drive immersion. Significantly, the Oculus Rift, HTC Vive and other HMDs discussed all had 3DOF at a minimum. The 3DOF represents a shared feature necessary for determining the orientation for correct display and can be used for gestures, as discussed in later sections.

After establishing the examples in section 2.3, the two critical sections for showing the gap in the research discussed interfaces in section 2.4 and interaction in section 2.5. The evolution of interfaces was examined to illustrate the iteration from traditional WIMP interfaces to the more recent NUI interfaces. Fitts' Law (Fitts and Posner, 1967; Scott MacKenzie, 1992) and Hick's Law (Hick, 1952) were significant considerations for design. Section 2.4.1 identified design considerations for interfaces (Furness et al., 1995; Olsson et al., 2012), including

Schneiderman’s 8 Golden Rules of interface design (Schneiderman, 1992). Section 2.4.2 identified the significance of designing for HMDs. The section considered how the screen takes up the user’s entire field of view, making it easier to present interfaces in world-space instead of screen-space. The section on interfaces concluded with a discussion on representing contextual information in section 2.4.3 to identify the significance of showing contextual options as part of the solution.

With the types of interfaces for VR and HMDs identified in section 2.4, section 2.5 presented interaction as the most significant area for investigation. The discussion started with the most common type represented by handheld controllers. Handheld controllers are significant for their reliability but are already very prevalent in research compared to HMDs solely as an interaction tool. Other types of controllers were identified in sections 2.5.2 and 2.5.3 to show the evolving interaction methods. Section 2.5.4 focused on HMDs, reiterated the significance of 3DOF, and considered motion sickness impacts (Tanaka and Takagi, 2004; Chang et al., 2020, Munafo et al., 2017) further to those discussed in section 2.2.4. Gestures were considered in section 2.5.5 to use the 3DOF shared by HMDs in a useful way to further HMD interaction. In discussing gestures, examples were first presented using hands as the gesture. To implement HMD Only type interaction, there needed to be a technique for selection, so the history of gaze (Chennamma and Yuan, 2013; Stiefelhagen et al., 1997; Morimoto and Mimica, 2005; Zhu and Ji, 2004) and dwell time interaction were considered. A HMD can be used for gazing at elements with a dwell time (Hansen et al., 2003; Špakov and Majaranta, 2012) to trigger a selection command. For head gestures, four examples were presented. Hirsch et al. (2014) used a neck brace to detect head gestures. Jackowski et al. (2016) controlled a robot using head gestures. Morency and Darrel (2006) showed that participants preferred head gestures with a nod for simple actions (e.g. dialog box confirmation) compared to using a keyboard or mouse. Ruban and Wood (2016) also used head gestures with a nod for yes and a head shake for no. Section 2.6 presented a final section to identify a delivery method for testing the proposed solution as part of experiments two and three with a serious games approach.

From surveying the work of researchers, a gap in research was defined by identifying the limited use of HMDs as a sole interaction device. Most solutions for interaction with HMDs relied on additional external devices for detecting interaction or other external controllers.

3DOF, as a shared feature of HMDs presented a source for meaningful interaction through the use of a gesture to reveal a menu in world-space. Like a nod or a head shake, a user could perform a quick head turn either left or right and from 3DOF detect the gesture to reveal a menu. By hiding a menu until the gesture is performed, the options can be modified at the time of revealing to provide appropriate contextual commands. With a menu revealed, the user could gaze by turning their head to face an option and then perform a dwell operation to complete the selection. Each topic for this gap in research was presented in the background to give necessary insight to the research preceding this dissertation.

The research conducted to survey the provided background influenced the establishment of research questions. RQ1 was influenced by viewing interactions used for controlling head interaction and not finding significant research into this area. Most surveyed interaction devices focused on using the hands or other external devices without directly benefitting from the sensors otherwise provided by HMDs. RQ2 was necessary to compare the proposed HMD Only interaction against a more traditional handheld instant selection tool. RQ3 was influenced by considerations of what could happen after a gesture had occurred. From reviewing materials about the representation of information in virtual worlds, the hidden behaviour of the PVMS was presented with the two-step behaviour to be discussed in Chapter 4 used to augment the amount of visual space occluded. RQ4 was then necessary to compare the menu presented in the PVMS against existing menu styles for validation.

This section concludes the background section of the dissertation. Many different related topic areas have been discussed related to VR, AR, HMDs, and associated or inspiring research. This chapter has laid the basis for the discussion leading into the dissertation's experimentation and prototyping of the PVMS. The first experiment will be covered in the next chapter, looking at interactions related to HMDs. The first experiment was used to experiment with HMDs and VR. The experiment captured useful data to guide how to carry out future experiments.

3 First Experiment: Object Manipulation via HMD Interaction

This chapter will cover the first of three experiments conducted during the research toward this dissertation. The experiment was run between May and June of 2015 and was designed to look at the use of HMD based VR with a variety of user input devices. The experiment was used as a starting point to determine development requirements, engine and workflow structures, and participant interest and, therefore, viability around conducting future experiments. There was enough interest from this experiment to continue for the following two experiments.

The chapter will cover an overview of the experiment. First looking at the methodology with a breakdown of participant recruitment, the technologies used, and descriptions of each task for the participants to complete. This experiment used pre-and post-questionnaires to collect demographic and user perception data and in-application data for usability. The types of information collected from these two sources will be covered before identifying the results. There will be a summary of discussions around the results in this chapter, further comparative discussion across results from all experiments will be covered in Chapter 7. For details about accessing the experiment code on GitHub, see Appendix D.

3.1 Experiment Overview

This experiment received ethics approval from the Flinders University Social and Behavioural Research Ethics Committee under research project number 6776. The following points were the primary goals set out to be answered as part of this experiment.

- Determine whether any of the three input methods (Computer Mouse, Mobile, or HMD Only) provided a better overall experience in terms of usability.
- Determine the user's perceived usability concerns associated with the head-mounted interaction experience specific to interaction with the Oculus Rift (Dev Kit 1).
- Determine the interest of participants in the future use of similar technologies.

A set of eight tasks were designed with varying levels of difficulty to accomplish these goals. All were designed to be simple for a novice user of the hardware, as participants with HMD experience would have been hard to find at that time. Each task was intended to test something slightly different. The user tasks were focused on the general theme of object selection and evaluating specific user interactions with object property adjustments related to position, size, and colour. The pre-experiment questionnaire was designed to gauge experience and perceptions before completing the experiment. The post-experiment questionnaire addressed the user experience across all the tasks and input methods. Further discussion and detail related to the questionnaires are continued in section 3.2.12.

3.2 Methodology

This methodology section covers the important features of how the experiment was run. The methodology has been broken down into many smaller sections. Section 3.2.3 Experiment Tasks details some of the general functional information related to tasks overall, leading to a brief discussion for each task on how they worked within the experiment. The task titles convey what to expect for a general overview of how the experiment was paced. After identifying experiment tasks, two sections cover the methods of data collection used in this experiment with the questionnaires and the data collected from within the application.

3.2.1 Recruitment

Recruitment was conducted over a couple of weeks for this experiment and ran between May and June of 2015. An email was sent out to all students in the College of Science and Engineering to drive recruitment. Subsequently, students were made aware of the email with a brief appearance at some lectures for topics that had many students. The email invited students to follow up for further information. Participants were all volunteers with no monetary compensation for their time. The Oculus Rift had only released Dev Kit 1 two years prior, and Dev Kit 2 was not released for nearly another year. The lack of availability to experience the hardware meant many students had not experienced the opportunity to use an Oculus Rift yet and presented an ideal way to drive interest in participation.

Once a potential participant had made contact via email to express their interest, a brief formal process was completed. A response to their interest was sent back to thank them for their interest and provide additional information. At this time, potential participants were sent via email three documents:

- A letter of introduction: introducing the project and researcher from the supervisor.
- An information sheet: giving brief details explaining what would occur during the experiment.
- A consent form: to show what they would be consenting to for participation. The form established the participant was of appropriate age (17+), understood experiment requirements/expectations, indicated they would not directly benefit from the research, indicated they were free to withdraw at any time and confirmed they would be deidentified to remain confidential in any publications.

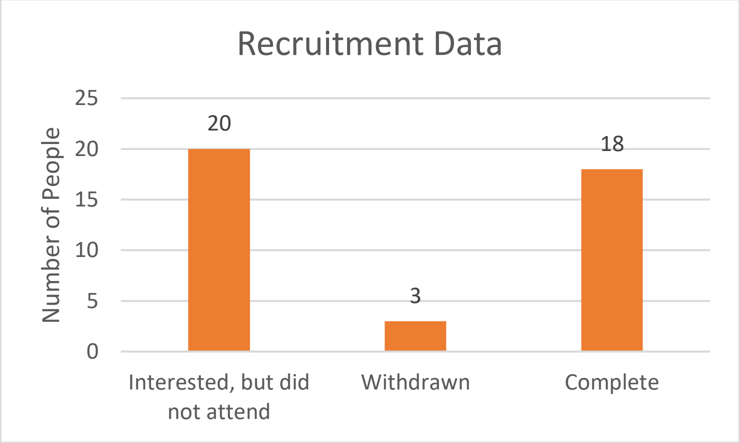


Figure 3.1: Experiment 1 Participation Quantities

Figure 3.1 shows the number of people who interacted via email. Of the 41 people who showed interest, 20 never followed up with an indication they wished to continue with participation after being provided with the additional documentation. The three who withdrew did so for various reasons. Two decided to withdraw when they arrived at the experiment and found it difficult to interact because they required glasses to see. The other withdrawal was from someone who allocated time to participate but failed to turn up. As shown in the figure, this left a total of 18 participants who completed the experiment. All materials related to recruitment and experiments can be found in Appendix A.2. Screening for vision issues was not done, and participants had to decide if they could handle the vision requirements. Additionally, Interpupillary distance (IPD) was not measured for participants to optimise the HMD as this was not a common practice till more recent HMDs made it more accessible.

3.2.2 Hardware and Software APIs

A selection of different hardware and software APIs were needed to develop and run the experiment. The following list identifies all the tools used, followed by a discussion about their importance to the experiment.

- Oculus Rift Dev Kit 1 with Oculus Rift SDK 0.6.0
- Mobile Device: Samsung Galaxy S4 running a custom Java App.
- Computer Mouse: Razer Abyssus
- Unity version 5.0.1f1
- Laptop Computer

Oculus Rift Dev Kit 1 with Oculus Rift SDK version 0.6.0

Figure removed due to
copyright restriction.
See link for original:
[Link]

Figure 3.2: Oculus Rift Dev Kit 1

Seen in Figure 3.2 is the Oculus Rift Dev Kit 1. The researcher acquired this device through the original Kickstarter campaign. The Dev Kit 1 was the first iteration and had a reasonably low resolution compared to future hardware with only 1280x800 resolution (640x800 per eye). This version of the HMD also caused some minor technical difficulties, as were experienced during the experiment. The issues caused the Oculus Rift Dev Kit 1 to not interact correctly with drift causing the user’s vision to move on its own (later discussed in 3.3.2 concerning Task Completion Rates). Compared to anything else available when conducting the first experiment, no HMD VR device could compete with this. Oculus Rift SDK version 0.6.0 was used throughout this experiment as it was the most up to date version at the time. Shortly after this experiment was conducted, SDK version 0.7.0 was released.

Mobile Device: Samsung Galaxy S4 running a custom Java App

The mobile device used for this experiment could effectively have been any small device with a touch screen for the way it was used. The Samsung Galaxy S4 was chosen because it was a researcher's spare phone that was not currently used for any other purpose.

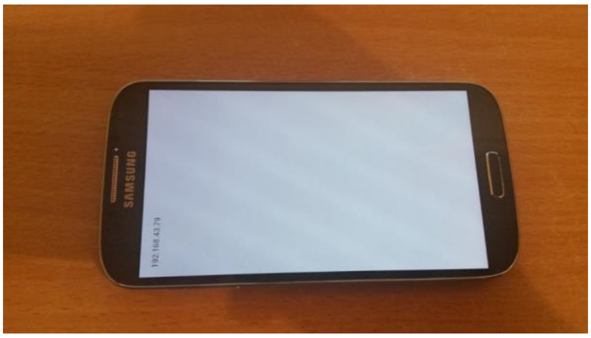


Figure 3.3: Samsung Galaxy S4 Running Custom Application

In Figure 3.3, the Samsung Galaxy S4 used is shown with the application running. Two types of messages would be sent from the mobile device to the Unity application. Either an “Alive” message to notify the phone is still functioning correctly or a tap counter to indicate the screen had been tapped and therefore an interaction should be attempted. The IP address shown on the phone is the PC connected to for debugging purposes. As the participants were wearing a HMD while using the interaction tool, the participant did not need the content displayed on screen for this experiment.

Computer Mouse: Razer Abyssus

Figure 3.4: Razer Abyssus Computer Mouse

Computer mice are mostly very generic, and this could have been any mouse as the left click was the only interaction point with this piece of hardware. The Razer Abyssus, as seen in Figure 3.4, was used for this experiment as it was a spare mouse that could be used and provided a similar feel to holding most other typical mice.

Unity version 5.0.1f1

The Unity game engine was an obvious choice for rapid experiment prototyping and development. Unity provides a simple game object-based approach to building scenes while dealing with rendering and providing utility. More significantly, as a point toward using Unity, the Oculus Rift SDK was provided with a Unity version for easier integration. VR video output was not a standard feature when this experiment was run, so the SDK integration was a welcome point of assistance for quick prototyping with the HMD. In the newer version of Unity used for the second and third experiments, the need for an SDK was removed and replaced with a generic checkbox to dictate the output to a VR headset. This feature specifically makes Unity applications quickly deployed to various VR headsets without manually loading the correct libraries as a developer. Code was written in C# within Unity.

Laptop Computer

A laptop computer was used to develop and run this experiment with the following specifications.

- CPU: Intel Core i7-4810Q 2.8GHz 4 cores
- Motherboard: P15SM-A/SM1-A
- RAM: 16GB DDR3 (2x8GB) 1600MHz
- GPU: NVIDIA GeForce GTX 980M
- OS: Windows 10 Pro

3.2.3 Experiment Tasks

The three different input methods were similar in how they were operated during task interaction. The primary distinction in what changed was how the apparatus or device would be held. The following list briefly summarises the nuances of each interaction method and shows the order completed during the experiment.

- **HMD Only:** The user would not have an additional input device to “click” with this method. A hover-to-select action was used to facilitate selection interactions. While looking at interaction elements or menus, a selection process would commence. Staying hovered over the same interaction point for 0.5 seconds would complete the selection operation. The 0.5 seconds dwell time was longer than necessary. It was chosen to ensure those who had little practice with HMDs would not perform accidental selections. Given the hover-to-select operation, it was difficult to define a deselection operation; for example, in the last two tasks (Task 6 and 7, sections 3.2.10 and 3.2.11), it was required for the participant to place cubes in a specific location. To overcome this issue, once the selected cube was moved roughly to the centre by a participant, it would become deselected and snap to the correct solution.

- **Mouse with HMD:** Interaction with the mouse did not change the process of looking at elements for manipulation. Instead of a hover with a delay before selection could occur, the left mouse button could be used to select instantly. When moving objects around, the snap to the location was disabled for use with the mouse. Instead, the participant would use a second left click to drop the currently held object.
- **Mobile with HMD:** The mobile interaction was roughly the same when compared to the mouse. A tap on the screen provided the equivalent of a left-click operation. The main difference with the mobile is how it can be held in both hands like a controller. Initially, there was some consideration to having a multi-button layout on the screen, but it was unnecessary for the tasks.

The application used a simple crosshair to show the point of interaction. The crosshair can be observed in Figure 3.5 (over). It may look small in the picture, but it was appropriately sized when viewed from within the Oculus Rift. Some participants found it too large, and the scale was decreased for future experiments. It served the primary purpose of clearly marking where the point of interaction was central to the participant's view. When developing this experiment, it was still necessary to use the Oculus Rift SDK with the associated camera rig game object in Unity. The camera rig in Unity consisted of a left and right camera used to generate the outputs for each eye in the HMD. A third camera was added centrally on the rig to use as a point for raycasting interactions as this simplified the code without using an offset from the left or right camera. This visually hidden third camera was also used for viewing/recording replay data and could be used in debugging mode. The position of the hidden camera was relevant concerning the crosshair because the crosshair used in these experiments is a mesh placed at a fixed constant offset from the camera rig. All three cameras shared a single parent object causing any translations or rotations of the HMD to cause synchronised movement of all cameras and components.

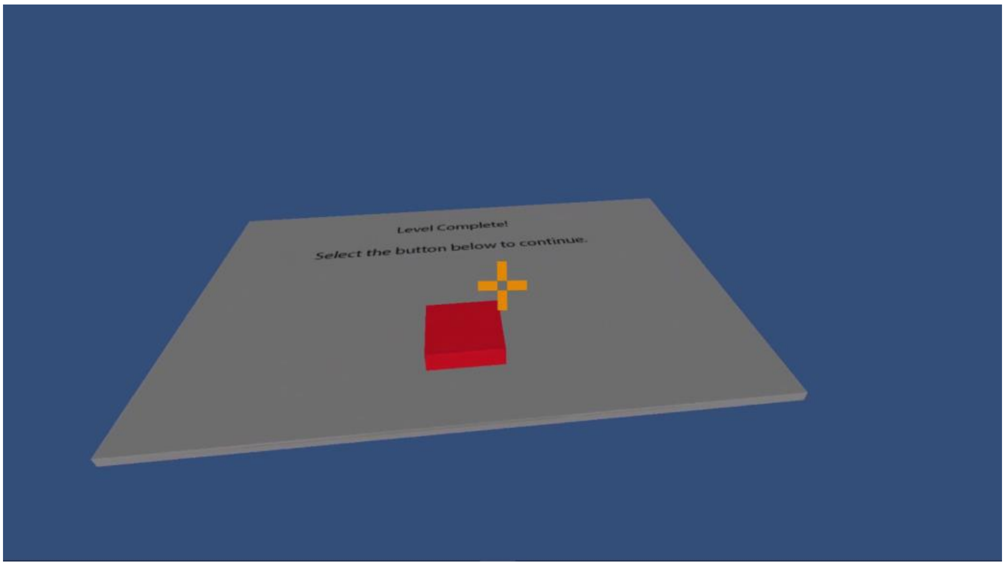


Figure 3.5: Cursor Example with a Level Complete Interface

For this experiment, an approach was used to simulate a primitive environment. This environment was dubbed “The Table Task Approach”. Originally the idea was that the system would combine the experiment tasks with an AR marker system. The markers would glue the interactions as if they were being completed directly on a tabletop. After some initial implementation testing, the addition of AR seemed unnecessary for the interaction testing. It was likely to cause more issues than necessary with the extended development time and possible failed marker recognition that was sometimes occurring. The user was placed in a perspective that would give the feeling of looking down on a table to provide a more reliable user experience. The table is represented by the grey surface consisting of one or more thin planes representing a surface for objects to exist on. The user was seated at a desk to reinforce their perception of the tabletop interaction. Being seated was not relevant to the data that was likely to come out of the experiment. It was hoped to aid in the immersion provided by the pseudo-reality approach.

An important experiment control was introduced in the form of staggering between the experiment tasks. Tasks were all timed, so it was important to ensure the participant was ready to begin as soon as they entered the tasks. The task intermission can be seen in Figure 3.5; it allowed for the occasional adjustment of the HMD, if necessary, between tasks. It consisted of text either indicating the first task was going to begin or that the previous task had been completed. Each followed by the red cube acting as a “button” to initiate the next task. Once tasks were initiated, the participants were prompted with an information panel above the tasks, describing what the goal was in as few words as possible. For example, in Figure 3.6, the text can be seen saying, “Task 6: Match the size, colour and position of the cube.” The participants were provided with a sheet showing what the tasks looked like in advance of the experiment. It cannot be expected that any participant would remember what was expected for any of them. The aid of in-headset prompts meant reduced researcher explanation was necessary to ensure the participants were on track. The participant’s progress could be observed from the laptop screen it was running from in case they did run into issues. Each of the tasks relied on cubes in some form; Appendix A.1.1 discusses how the cube script was designed and presents the game management strategy and operations.

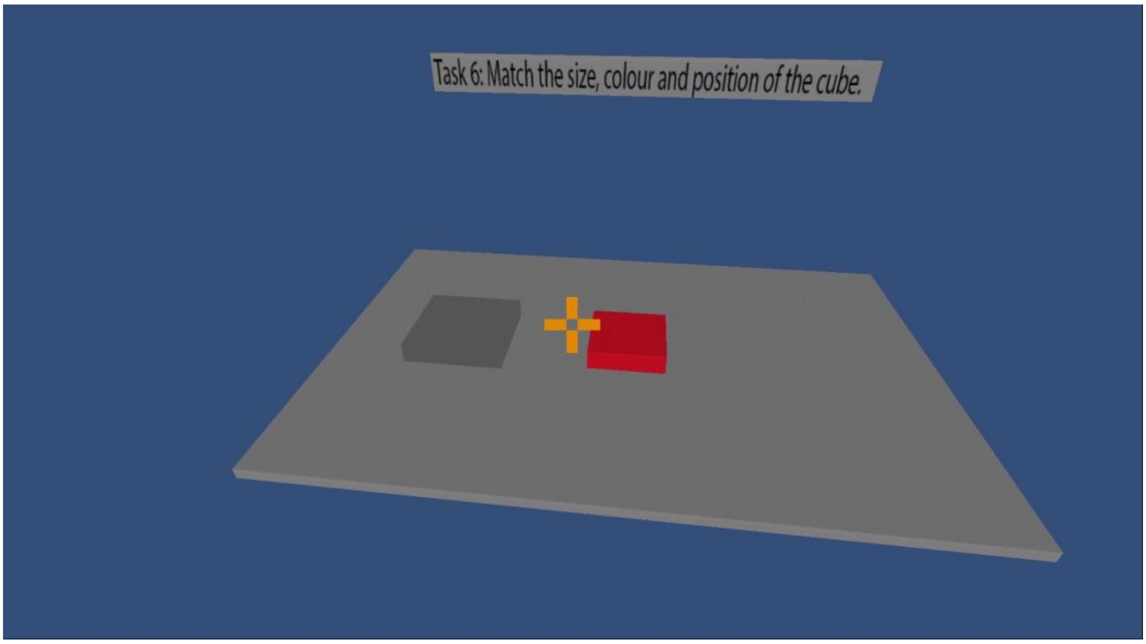


Figure 3.6: Information Panel Example

Task Design Philosophy

The overall philosophy of why the tasks were chosen should be discussed before describing them with more specific implementation detail and images to show how they appeared to the participants. The three types of input were selected to provide a distinct comparison between input with only the HMD (HMD Only), the tap-to-select instant interaction of a familiar mouse device, and an adaption of a mobile device to act in the same way as the mouse but exist as an untethered input. In the initial design for the experiment, there were more grandiose ideas for branching variation in types of input. It was intended to make use of a Microsoft Kinect or LEAP Motion sensor as a fourth input. These would have been used to contrast with the direct interaction of virtualised hands in the game world or gestures detected by the sensors. These inputs were scrapped to prevent too much scope creep. Another consideration for the mobile input was to go beyond a simple tap to interact and instead virtualise the mobile to become part of the game world using sensors or subdivide the screen to provide contextually appropriate buttons. Similarly, the extended use of mobile in this way for the experiment was abandoned to prevent too much scope creep. The considerations of how a mobile device might extend interaction in VR is part of how tasks 6 and 7 with their menus led to the PVMS.

For the individual tasks, they fall into four categories for design philosophy. The tasks were designed to evaluate effectiveness, difficulty, and fatigue using the Oculus Rift Dev Kit 1 as base data for VR interaction, focusing on comparing tap-to-select and dwell-to-select. The four categories for the design included simple baseline selection tasks with static objects, selection of moving objects, performing finely detailed movements with accuracy, and more complex object manipulation using two types of custom menus to match objects. The following list extrapolates the considerations of each.

- **Selection of Static Objects:** For Tasks 1, 2, 2 (part 2), and the level complete screens between each task, the participant was presented with one or more static cubes that had to be selected. These represented a goal to select all the available elements. Principally these tasks were achieving two fundamental tests. The first was to evaluate a situation related to Fitts' Law (Fitts and Posner, 1967; Scott MacKenzie, 1992), looking at how effectively participants could navigate the virtual world with an increasing number of required selections. The second was that these tasks introduced the mechanics to the participants by starting them off with a selection that would change how they had to accomplish it using the different input methods, therefore, preparing them for the later, more complex tasks.
- **Selection of Moving Objects:** For Tasks 3 and 4, the participants were presented with moving cubes instead of static ones. The purpose of these tasks was an interaction still considering Fitts' Law (Fitts and Posner, 1967; Scott MacKenzie, 1992), with the additional consideration that it is not always possible for elements to be static when performing actions. A user can not expect everything to be static in a VR world as moving elements provide contextual enrichment. The design of these tasks required

the participant to track a target using the cursor attentively. The cube targets gradually slowed once a user hovered over them to simplify the interaction and sped up if the participant stopped targeting with the cursor.

- **Fine Detail Object Interaction:** For Task 5, the participants had to select and use the tracking they had learned in Task 3 and 4 to navigate a hazardous maze course. The maze was short, but the participant was punished with a task reset if the participant collided with a wall. The purpose of the task was to evaluate the effectiveness of the more precise movement in VR with head interaction. Participants were expected to struggle with at least a couple of mistakes.
- **Object Matching with Custom Menu Interaction:** The last two tasks had participants complete multi-step actions to accomplish object matching. In both tasks, they were asked to perform similar functions with different menus. Participants were provided with an initial cube state. They would alter the size, colour, and position until matching a faded out requirement. These types of matching tasks demonstrate a clear understanding of using the tools if they can be accomplished efficiently. The two different menus evaluated included a circular menu that appeared at the location of an interacted object and was compared against a pair of static menus that were fixed to a world position. These menus were designed to be simple and provide a baseline for interaction going into future experiments.

The following sections present additional detail about the tasks that the participants were asked to complete.

3.2.4 Task 1: Single Object Interaction

The first task provided a simple entry point for participants to ensure they understood how selection worked with the current interaction method. Before completing this task, the participant would have already completed this same task by confirming they were ready (seen in Figure 3.5). The task introduced the participant to the information panel describing the task goal of "Select the cube." as seen in Figure 3.7 (over). The participant would use the HMD to hover over the single cube using the crosshair to complete this task. The crosshair has been removed from the following screenshots to focus on the task content. The crosshair can be seen in Figure 3.6 or other earlier screenshots if further reference is needed. With the crosshair over the cube, the participant would initiate interaction differently depending on the current device in use. With HMD Only, this would be completed by hovering for 0.5 seconds. With the mouse, it was completed with a mouse click while hovering over the cube. And for the mobile device, a screen tap while hovering over the cube would complete the task. After completing one of these interactions successfully, the participant would automatically transition to the level complete screen.

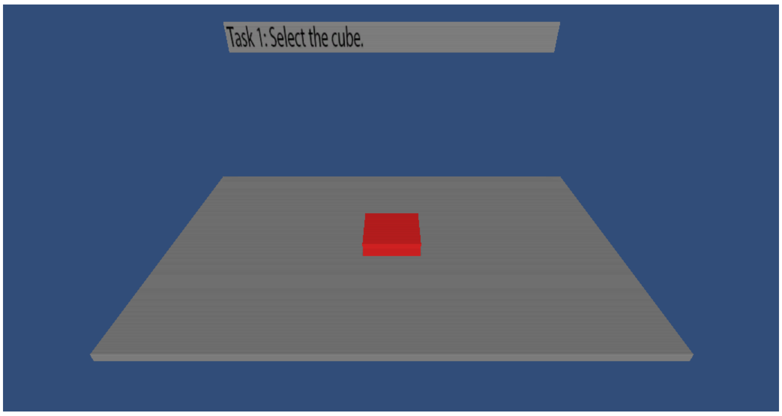


Figure 3.7: Task 1

3.2.5 Task 2: Multiple Object Interaction on a Single Plane

In the second task, the participant was presented with the task of "Select all the cubes". This task presented multiple cubes simultaneously for user selection, as seen in Figure 3.8, to evaluate multiple selection operations. All the cubes are visible from the moment the task began without the participant needing to look in any different directions. The cubes were also all immobile. The main difference for participants completing this task was how quickly they could swap between active interaction points. The cubes could be selected in any order, and successfully turning them all green would take the participant to the level complete screen.

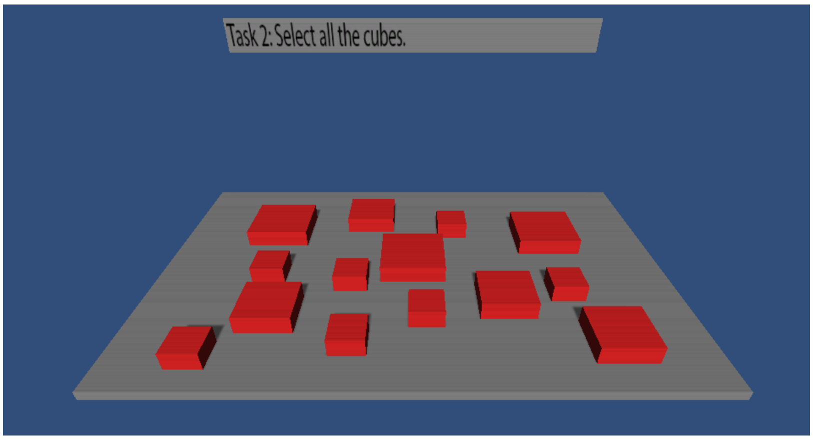


Figure 3.8: Task 2

3.2.6 Task 2 (part 2): Multiple Object Interaction on Multiple Planes

Although this task was labelled Task 2 (part 2), it should have had its own number for the experiment. The main differences for this task were a shuffling of cubes on the main task area (compared to Task 2), and more significantly, the addition of two more regions with cubes. The addition of two peripheral task areas forced the participant to interact with objects further apart and outside of the initial viewed area. The cubes could be selected in any order, although participants typically completed the central cubes first as they are the first visible. Selecting all the cubes in any order would complete the task and take the participant to the level complete screen. This task can be seen in Figure 3.9.

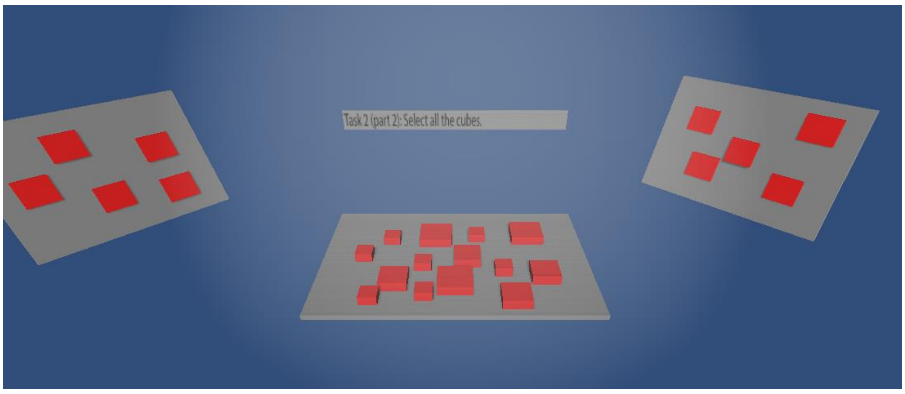


Figure 3.9: Task 2 (part 2)

3.2.7 Task 3: Multiple Object Interaction with Moving Objects

Task 3 provided the first task where cubes were not static. In Figure 3.10, the black box shows the outline path where both cubes would travel. A slowing down feature was implemented to make this experience more engaging, less frustrating, and improve the experience, particularly for HMD Only interaction.

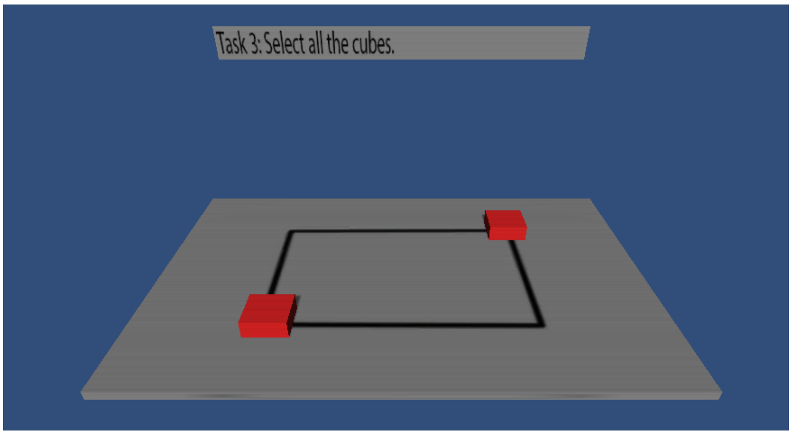


Figure 3.10: Task 3

```
curMoveSpeed = (maxMoveSpeed - minMoveSpeed)
                * (1 - selectionProgress) + minMoveSpeed;
```

Listing 3.1: Current Speed of Cube

Listing 3.2 shows the code used to calculate cube move speed. For this formula, the values **minMoveSpeed** was set to 1.5 and **maxMoveSpeed** was set to 3. When the participants' focus was on one of the cubes, this part of the script would execute. By the time selection progress had reached 100%, the object movement speed would be halved. The interaction gave the impression of the cubes' motion slowing down when actively engaged with the participant. Cancelling selection by moving off the cubes would reset this progress and return the cubes to their maximum speed. The goal was to select both moving cubes, as was the case with the previous tasks. An example of how the cubes for this task were configured can be seen in Appendix A.1.6.

3.2.8 Task 4: Multiple Object Interaction with More Moving Objects

Task 4 followed on with a second example of interacting with moving objects. Instead of just two objects, this time, there were four cubes for the participant to select. Their properties were the same as with Task 3. The two new cubes were placed on a slightly larger square to cycle around the inner square. The additional cubes can be seen in Figure 3.11. Once all four cubes were selected, the user would be taken to a level complete screen.

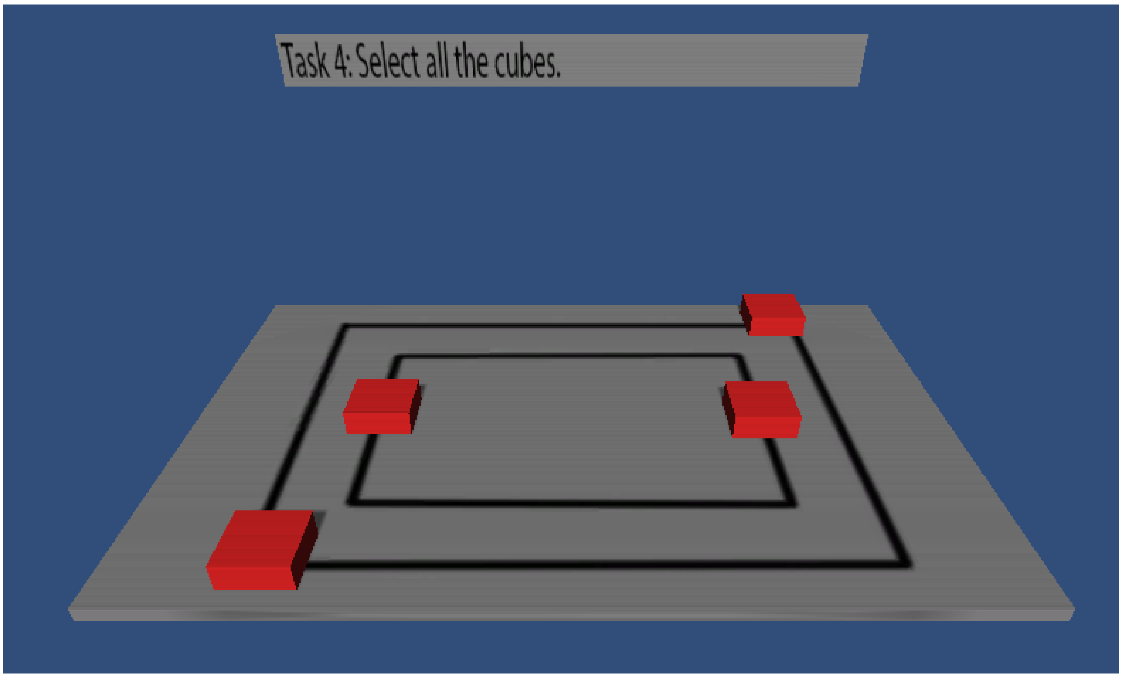


Figure 3.11: Task 4

3.2.9 Task 5: Fine Detail Object Manipulation with Hazard Avoidance

Task 5 was the only task with a soft failure state. This task was designed to test participants' ability to navigate a path without hitting any obstacles using the HMD as the primary controller. The navigation was made more difficult by the camera angle from where participants would be viewing the scene. The participant's perspective of the task can be seen in Figure 3.12.

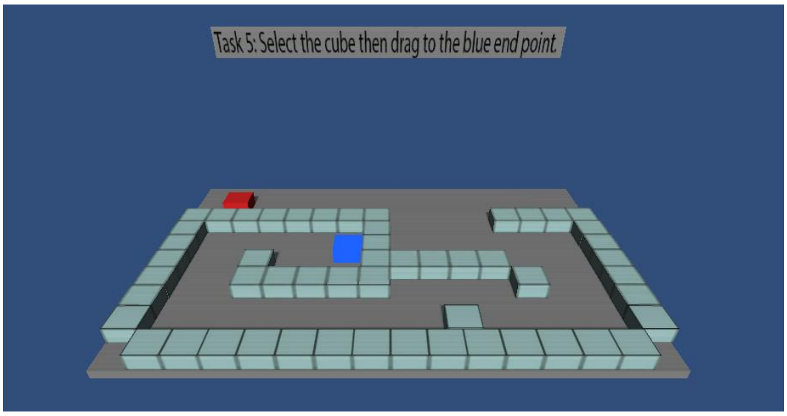


Figure 3.12: Task 5

To succeed in this task, the participant had first to select the red cube. Then move the red cube through to the blue cube without touching any of the walls. The path is shown in Figure 3.13. The selection of the red cube was completed in a similar way to selecting any of the previous red cubes; by hovering over the cube followed by selection over time with HMD only, a click of the left mouse button to select, or a tap of the mobile screen to select. The cube was configured in the application by using a "Draggable" property for the cube move type. When the red cube touched the blue cube at the end, in any way, the task would complete and take the user on to the level complete screen.

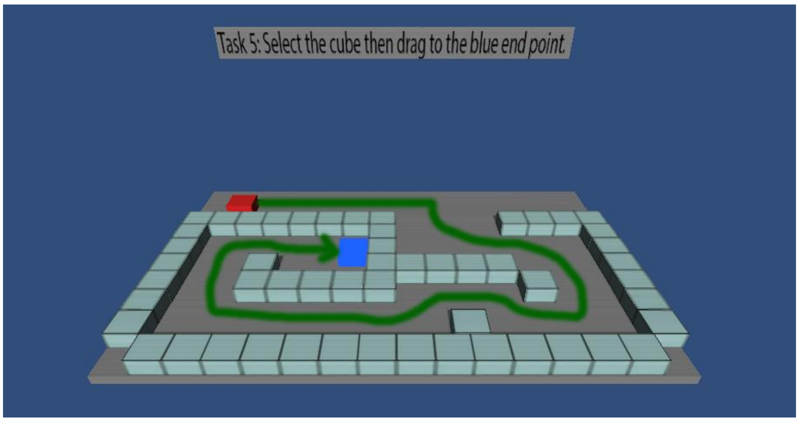


Figure 3.13: Task 5 Path

The primary difficulty of this task came from what happened when the red cube touched any of the walls. In Figure 3.14, a top-down view of the task is shown. The cube has a small clearance of 1.5 to 2 cubes through the area that must be navigated. Once the red cube touched any wall, a reset would be triggered, forcing the red cube to return to the start location and was deselected again.

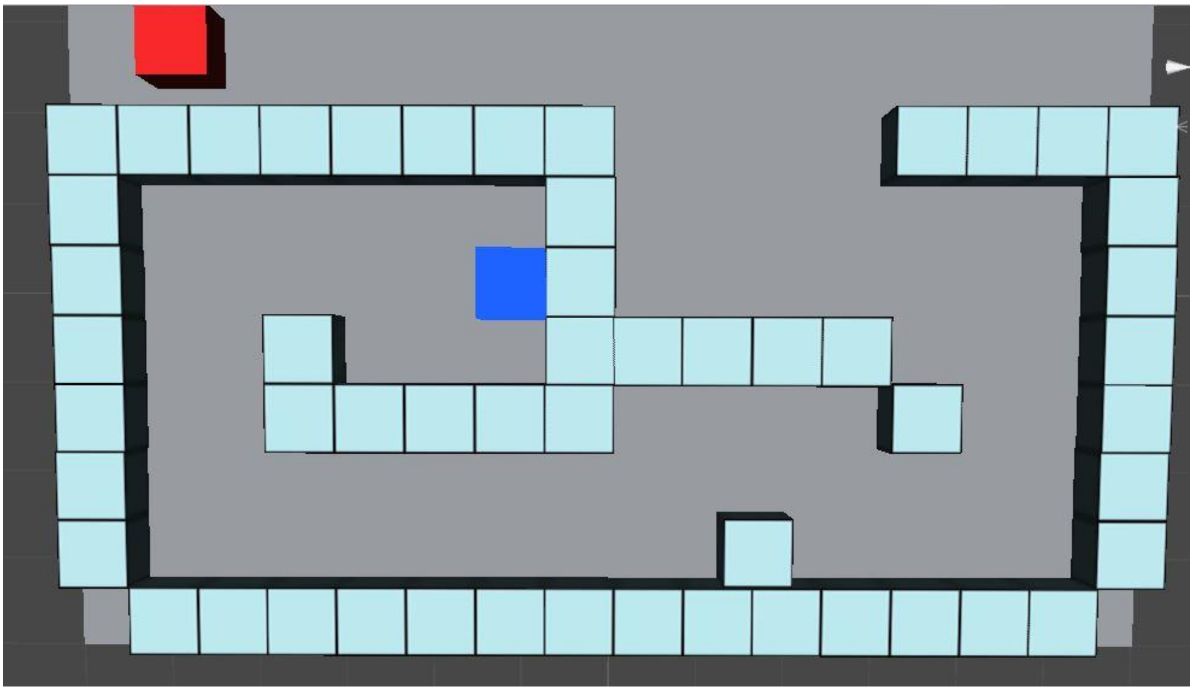


Figure 3.14: Task 5 Bird Eye View

A failsafe mechanic was introduced to make this frustrating mechanic manageable within an experiment. In Figure 3.15 (over), the object ids for all the walls are visible. There are four walls with a red tag (excluding the End Point). These walls were selected as the failsafe. If the participant failed by colliding into these walls five times, that specific wall would disappear and no longer cause collisions. As will be discussed later in the results, this may not have been generous enough. Perhaps more appropriate would have been hitting any of the special walls 5 times would make all the special walls disappear. The difficulty is something that could only be tuned after watching participants struggle. All wall collisions were saved with a counter of collisions per wall id. The counter made it possible to see the walls causing the most problems for participants. There were very few participants who triggered the failsafe.

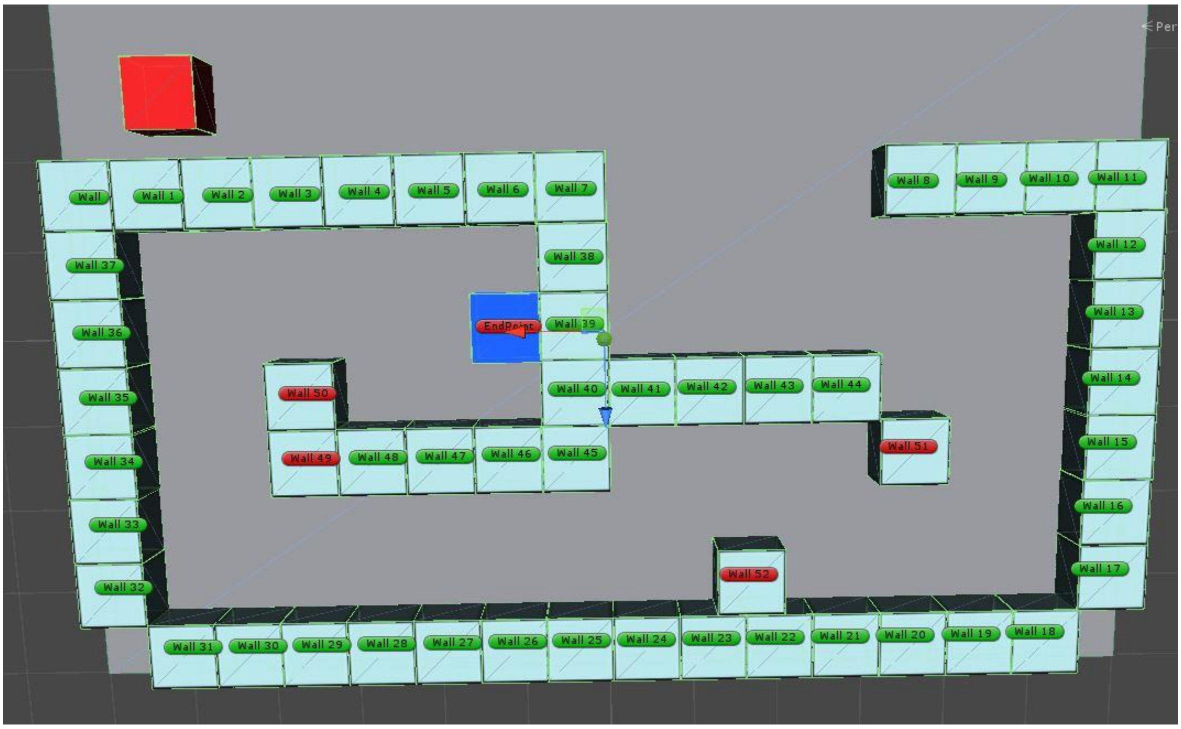


Figure 3.15: Task 5 Object IDs

3.2.10 Task 6: Singular Object Matching with Circular Menu

Task 6 was the first of two final tasks where the participant had to set a cube's size, colour, and location properties. For Task 6, this was completed using a simple circular menu. The pattern to be matched would always be in the same location, require changing colour to green, and size to large. The initial state the user would observe when they first entered the task can be seen in .

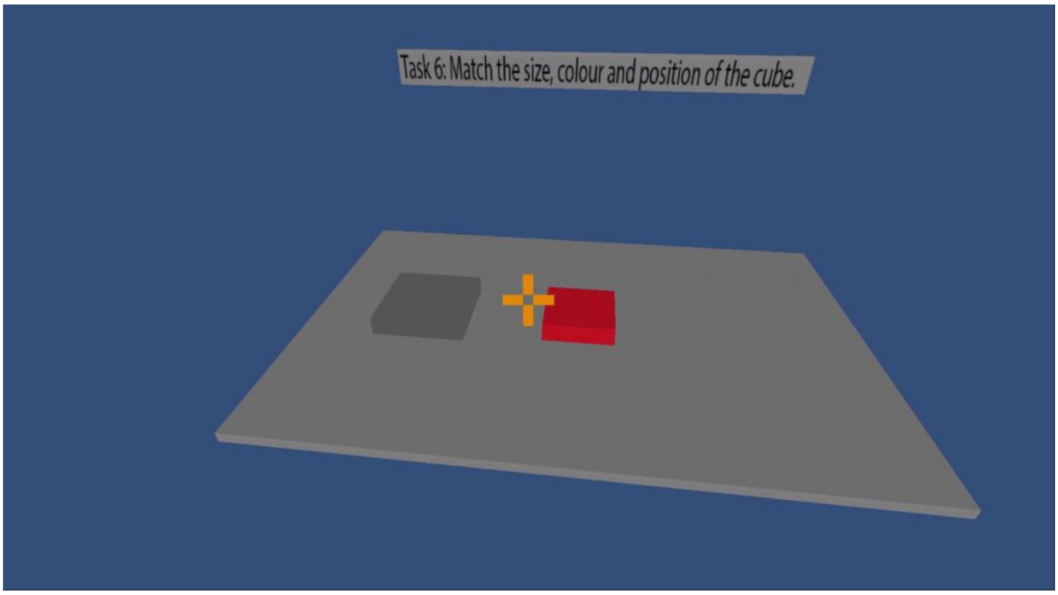


Figure 3.16: Task 6

As with the other tasks, selecting the cube was completed by hovering over the cube, then completing the selection operation depending on the input method. Once successfully selected, two distinct actions would occur. Firstly, a circular menu would appear around the cube with four possible options. Secondly, the objective to be completed would begin to cyclically change its alpha to make the large green finish point appear to glow. These can both be seen from comparing (previous page) to Figure 3.17.

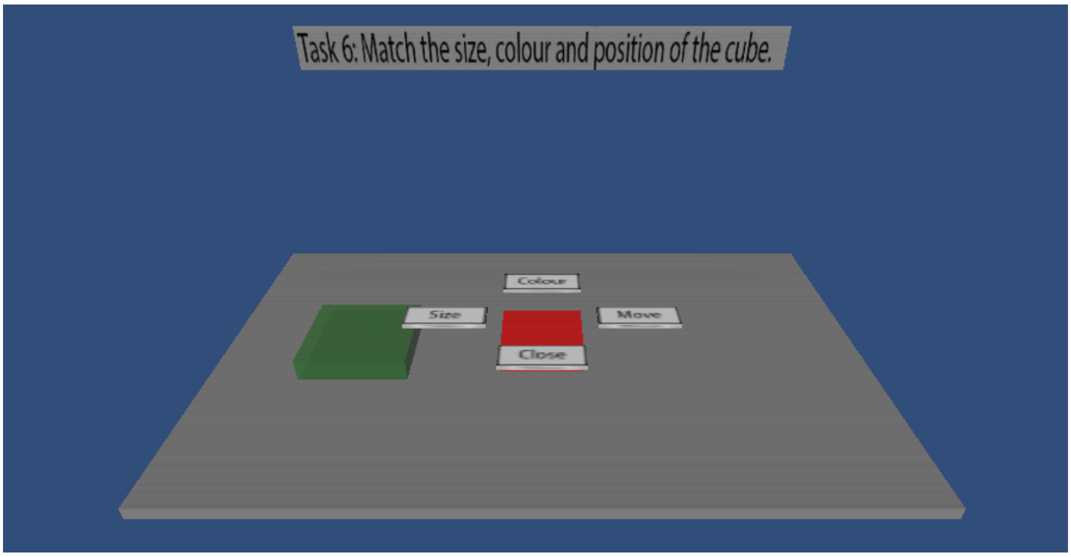


Figure 3.17: Task 6 Menu Selected

The structure of these objects within Unity can be seen in Appendix A.1.7.

In Figure 3.17, the user has selected the Move option. The Move option changes the cube to be "Draggable", and as the cube is already selected, it will stay attached to the crosshair's location. To drop the cube for this task, the participant needed to move the cube inside the target cube (**`MatchCube`** in the game script) to match the same location within 0.1 units in any direction. Completing the move would leave the cube dropped, as seen in Figure 3.18 (over). There are still two properties that need matching to achieve the end goal. In this case, the next property to be changed was the size. Once again, the participant would have selected the cube, which opens the same menu as seen in Figure 3.17. Then choosing the "Size" menu option would give the menu seen in Figure 3.19 (over). In this figure, the "Large" size has already been applied, and the user is selecting the "Back" option to return to the previous menu. This way, the participant could try different sizes to determine which size was the correct option without the menu closing on them until they used the "Back" option.

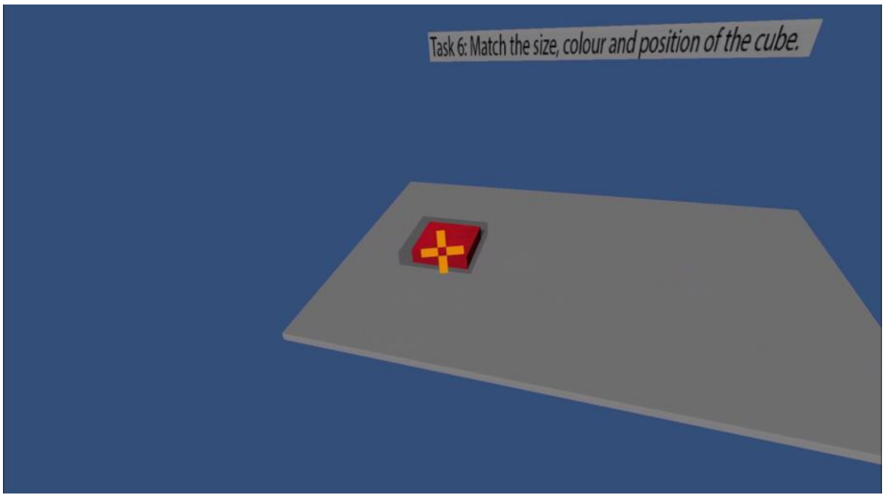


Figure 3.18: Task 6 Cube Moved to Finish (With Wrong Size/Colour)

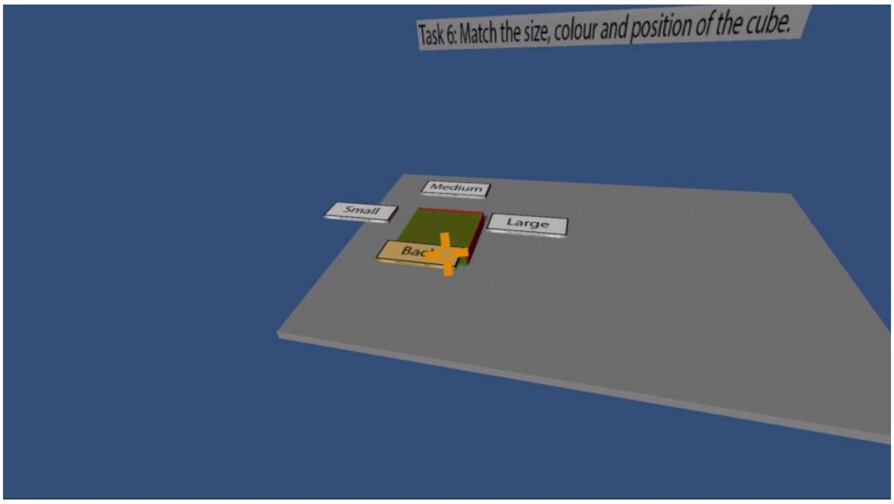
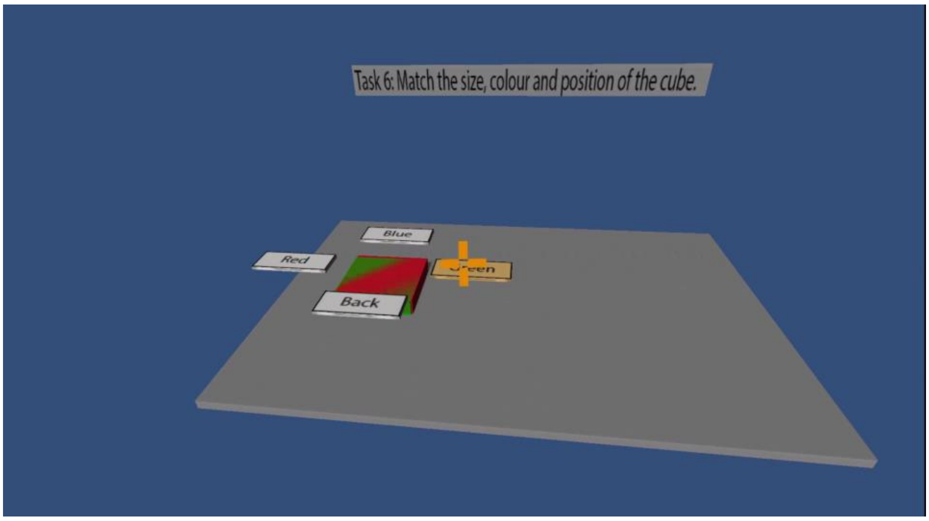


Figure 3.19: Task 6 Cube Size Options

The final step in completing this example of the task would be to set the colour to green, matching the glowing colour. In Figure 3.20 (over), the red and green are shown mixed together because of the clash between objects showing at the same position and size. It was still clear what colour should be selected even if the colour was set last. After the "Back" option was used in Figure 3.19, the original menu from Figure 3.17 would be shown. Then selecting "Colour" brings the participant to the menu shown in Figure 3.20 (over). Selecting "Green" would complete the task and take the participant to the completion screen. This sequence of actions was one of a few ways to solve the task. For example, the participant could solve it in the order Colour->Size->Position. Or any other combination. When different participants completed the tasks, there was a variance in how different people approached the solution.

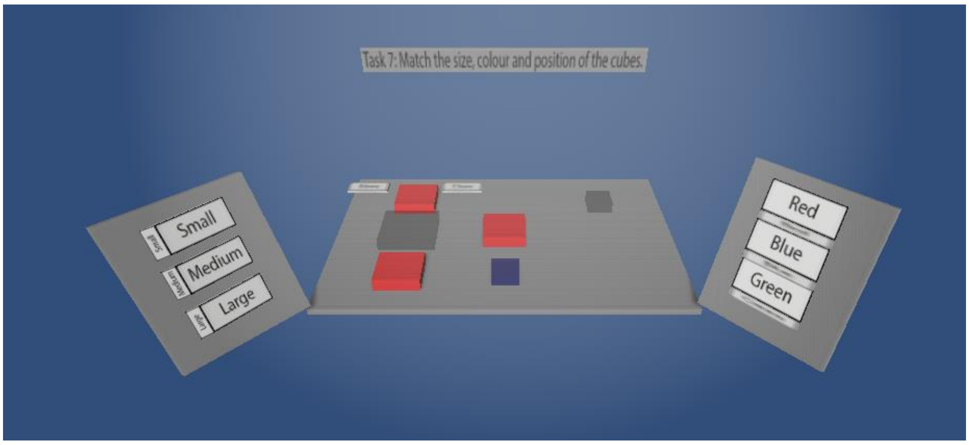


Task 6: Match the size, colour and position of the cube.

Figure 3.20: Task 6 Cube Colour Options

3.2.11 Task 7: Multiple Object Matching with Static Periphery Menu

Task 7 was the most complicated due to how many individual actions had to be completed for the task to finish. In this task, the goal was to match three cubes to the correct **`MatchCube`** properties. This task used an approach referred to as a static periphery menu for most of the state changing interactions on cubes. The menus' locations relative to the participant's perspective required turning the head to look either left or right at the menus for changing either size or colour. In Figure 3.21, the full view of all elements in this task can be seen. The periphery menus would only display while a cube was selected along with two additional buttons directly on the cube with "Move" and "Back" options. These two buttons were included specifically to make it obvious which object was currently selected to aid the user in making decisions. In this figure, the top left cube is selected, and the **`MatchCube`** in the bottom middle showing as a small blue cube would be the end destination required.



Task 7: Match the size, colour and position of the cubes.

Figure 3.21: Task 7 (zoomed out for full view)

To complete this task, the pairs that were configured as the solution were:

- Top Left Medium Red Cube would solve as a Small Blue Cube on the Bottom Middle.
- Middle Medium Red Cube would solve as a Large Green Cube on the Left Middle.
- Bottom Left Medium Red Cube would solve as a Small Red Cube on the Top Right.

In every case, when a cube was selected to be modified or moved, the associated **`MatchCube`** would glow to indicate what the result should look like and where it should be. One possible order of completion can be found in the following figures. In Figure 3.22, the participant has opted to modify the middle cube and changed the properties from red to green and size from medium to large.

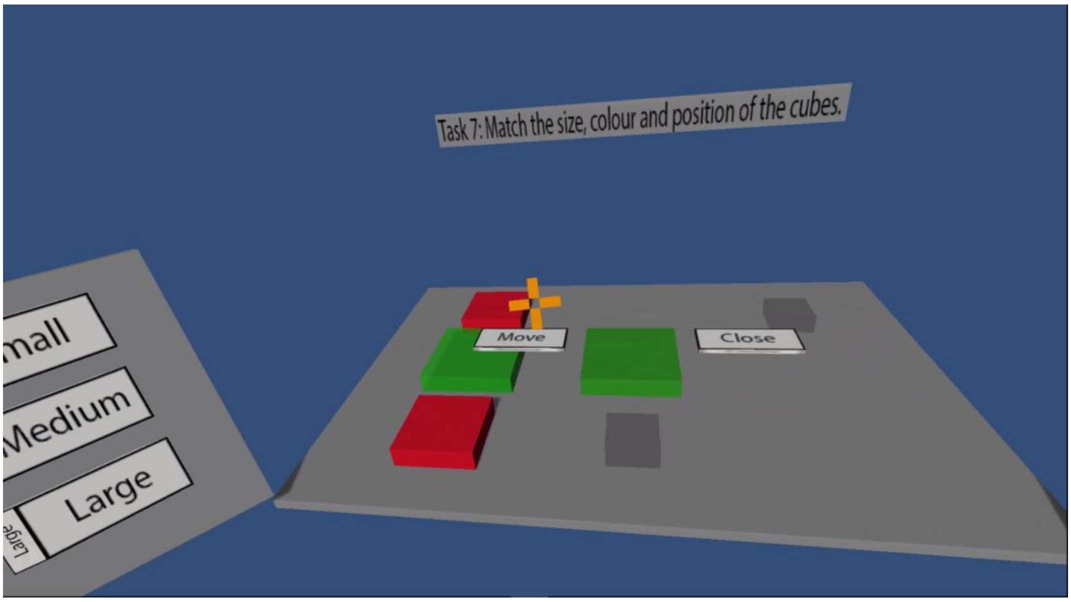


Figure 3.22: Task 7 Cube Selected with None Solved

In Figure 3.23 (over), further steps have been taken. The large green cube from Figure 3.22 has moved to the final location on the left. Additionally, the top left medium red cube has been modified. The top left cube was changed from red to blue and from medium to small. Then moved to the final location in the bottom middle. This sequence of actions would leave the state as seen in Figure 3.23 with the last cube selected. The last two steps to complete the task would be to set the size from medium to small and move the cube to the top right. After these steps are complete, the participant would be taken to a screen indicating that all tasks for the current input method have been completed. As with Task 6, this was just an example order of how the task may be completed. Participants could opt to solve the task in any way they wished.

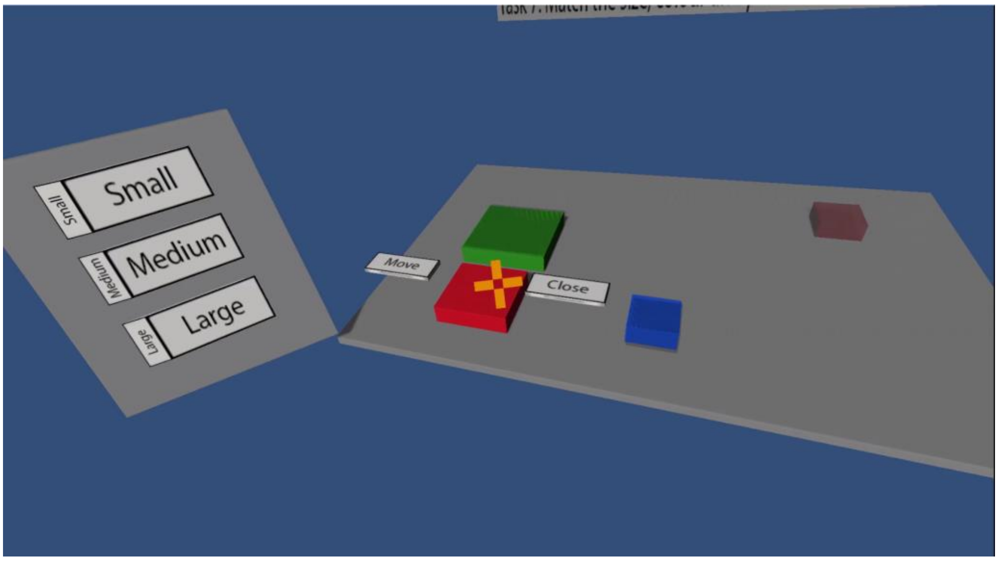


Figure 3.23: Task 7 Near Complete

Additional information about how this task was configured can be found in Appendix A.1.8.

3.2.12 Pre and Post Experiment Questionnaires

As part of the experiment, each participant completed two questionnaires to evaluate their feedback related to the experiment and demographic data. The wording, presented concisely here, has been modified for the question summaries, but the complete questionnaires can be found in Appendix A.2.

One question that does warrant an explanation for why it was included is question 18 in the pre-experiment questions. When developing and designing the first experiment, there was still a consideration for how the testing of future experiments may be targeted. One of those considerations was how the PVMS might be useful for the elderly. In the interest of more general usability testing, this was not continued as an avenue for future experiments but has been included in the results for completeness.

Pre-Experiment 20 questions

The pre-experiment questionnaire focused on the prior experience of the participant. Some general non-experiment questions were added to assess participant thoughts on using HMDs in public and use for elderly assisted living. The pre-experiment questions were finished with questions to determine participants' perceptions about the effectiveness of each input method. The following will cover a summary of the questions in the pre-experiment questionnaire.

- Questions 1 to 5: Personal and Academic Background covering questions related to Age, Gender, Education Status, Area of Study/Teaching/Research.
- Questions 6 to 11: Previous Participation in Experiments or Personal Use of Augmented Reality, HMDs, and Mobile/Tablet Computing.
- Questions 12 to 14: Use of Computers, hours per week using a computer, playing video games, and the types of platforms that games were played on.
- Question 15: Interest in using HMDs for non-gaming activities.
- Question 16: Importance of social weight a device is to the participant for wearing in public.
- Question 17: The significance of factors on the participant wearing a HMD in public, ranking each of the size, weight, comfort, outward visual appeal, and usefulness of the device.
- Question 18: How useful a HMD is perceived to be for the assisted living of the elderly.
- Question 19: Choice of preferred input method before conducting the experiment and why.
- Question 20: Ranking each input method for perceived effectiveness without having participated in the experiment yet.

Post-Experiment 14 questions

All questions in the pre-experiment questionnaire were single response questions. Many of the questions here were multiple part responses broken up by the input methods and the tasks. In total, there were 112 responses asked of the participant. The number of questions was, in hindsight, too many. It may have been more appropriate to ask the questions related to each input method after the tasks were completed with the input method instead. This change would have broken up the responses significantly. The primary metrics used in this questionnaire focused on the effectiveness, difficulty, and fatigue caused by the input methods. Question 12 mostly followed a System Usability Scale (Sauro, 2011) but only had 9 of 10 questions. The one question left out was related to requiring a technical person to use the system. The questions still provided a useful data source to represent the individual responses and an overall inferred score for the System Usability Scale.

- Question 1: Rate how effective each input method was for every task. 1 = extremely ineffective, 5 = extremely effective. 3 input methods times 7 tasks = 21 responses.
- Question 2: Rate how difficult each input method was for every task. 1 = very difficult, 5 = very easy. 3 input methods times 7 tasks = 21 responses.
- Question 3 to 8: List up to 3 most enjoyable and 3 most difficult aspects for each input method.
- Question 9: Preferred input method and why the input method is preferred.
- Question 10: Use of Augmented Reality HMDs in future.
- Question 11: Rate how fatigued each input method was for every task. 1 = No Fatigue, 5 = Extreme Fatigue. 3 input methods times 7 tasks = 21 responses.
- Question 12: For each input method, the following questions used a scale of 1 = strongly disagree, and 5 = strongly agree. 9 questions times 3 input methods = 27 responses.
    - "I think that I would like to use this input method frequently."
    - "I found the input method unnecessarily complex."
    - "I thought the input method was easy to use."
    - "I found the various functions in this input method were well integrated."
    - "I thought there was too much inconsistency with this input method."
    - "I would imagine that most people would learn how to use this input method very quickly."
    - "I found the input method very cumbersome to use."
    - "I felt very confident using the input method."
    - "I needed to learn a lot of things before I could get going with this input method."
- Question 13: Any other comments about the experiment.

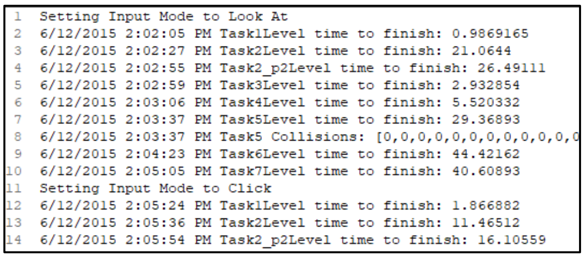
3.2.13 Application Logged Data

Data logs were captured for each session. Three different data files were generated for each participant with the following naming formats (where the start is a date/time format).

- yyyy_M_d__hh_mm_extra.dat: Text logs of events.
- yyyy_M_d__hh_mm_network.dat: Text log of all network events.

- yyyy_M_d__hh_mm.dat: Replay data binary file.

These logs provided additional metrics to use alongside the feedback from the questionnaires. shows an example of the extra data log. The extra data log primarily showed the task completion times for each task and the collisions occurring during Task 5.



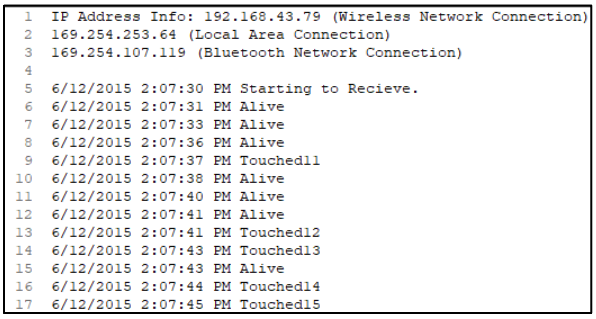
```
Setting Input Mode to Look At
6/12/2015 2:02:05 PM Task1Level time to finish: 0.9869165
6/12/2015 2:02:27 PM Task2Level time to finish: 21.0644
6/12/2015 2:02:55 PM Task2_p2Level time to finish: 26.49111
6/12/2015 2:02:59 PM Task3Level time to finish: 2.932854
6/12/2015 2:03:06 PM Task4Level time to finish: 5.520332
6/12/2015 2:03:37 PM Task5Level time to finish: 29.36893
6/12/2015 2:03:37 PM Task5 Collisions: [0,0,0,0,0,0,0,0,0,0,0,0
6/12/2015 2:04:23 PM Task6Level time to finish: 44.42162
6/12/2015 2:05:05 PM Task7Level time to finish: 40.60893
Setting Input Mode to Click
6/12/2015 2:05:24 PM Task1Level time to finish: 1.866882
6/12/2015 2:05:36 PM Task2Level time to finish: 11.46512
6/12/2015 2:05:54 PM Task2_p2Level time to finish: 16.10559
```

Figure 3.24: Extra Data Event Log

The second type of output file was the network data event log, as seen in Figure 3.25. The network log showed four important types of data.

- The connection info to verify everything was correct in case of a wrong address when the log started.
- Events for starting and stopping receiving updates from the mobile device to indicate if the device was disconnected from the network.
- An “Alive” message every two seconds to allow error handling by Unity if the mobile device lost connection.
- And most importantly, network events in the form “TouchedN”, where N is the counter of how many touch events have occurred as registered on the mobile side.

The file was generated to allow quick debugging and error handling if the experiment functioned incorrectly while a participant was interacting.



```
IP Address Info: 192.168.43.79 (Wireless Network Connection)
169.254.253.64 (Local Area Connection)
169.254.107.119 (Bluetooth Network Connection)

6/12/2015 2:07:30 PM Starting to Recieve.
6/12/2015 2:07:31 PM Alive
6/12/2015 2:07:33 PM Alive
6/12/2015 2:07:36 PM Alive
6/12/2015 2:07:37 PM Touched11
6/12/2015 2:07:38 PM Alive
6/12/2015 2:07:40 PM Alive
6/12/2015 2:07:41 PM Alive
6/12/2015 2:07:41 PM Touched12
6/12/2015 2:07:43 PM Touched13
6/12/2015 2:07:43 PM Alive
6/12/2015 2:07:44 PM Touched14
6/12/2015 2:07:45 PM Touched15
```

Figure 3.25: Network Data Event Log

The last type of file output was the binary file capturing the entire experiment for the current participant. Figure 3.26 and Figure 3.27 (over) show the definition of classes used to generate the binary file through serialization. When replayed, this custom replay system would accurately represent the participant's actions as they completed when wearing the HMD during the experiment. The game object representation for some of this content can be found in Appendix A.1.5.

The replay system was designed to be very simple to minimise data inconsistencies while replaying the session. Only the minimum amount of data was stored per frame, with every frame containing the same data size to provide a suitable consistent replay system. The data for each frame was represented by the **`ReplayEvent`** class, as seen in Figure 3.26. The variable **`deltaTime`** represented the length of that frame. The camera was represented by a **`Quaternion`** and **`Vector3`** for rotation and position. The **`triggerEdge`** represented if there was a tap or click action that would be applied on that frame. And the **`KeyActionEvent`** used a set of enumerated values to indicate if a task jump or input mode change was occurring. A **`ReplayEvent`** was used to store the action whether the participant triggered a task change by task completion or skipped using a keyboard shortcut. Serialization occurred via the methods included with the class and not directly serializing the variables because not all the data types could be serialized.

<table>
<tr><th colspan="2">[Serializable] ReplayEvent</th></tr>
<tr><td>public enum KeyActionEvent { None = 0, NextLevel = 1, PreviousLevel = 2, FirstLevel = 3, RestartGame = 4, SetInputMode_Look, SetInputMode_Cursor, SetInputMode_Mobile}

// All variables are [NonSerializedAttribute]
+ float deltaTime;
+ Quaternion rotation;
+ Vector3 position;
+ bool triggerEdge;
+ KeyActionEvent keyActionEvent;</td><td>+ ReplayEvent()
+ ReplayEvent(float deltaTime, Quaternion rotation, Vector3 position, bool triggerEdge, KeyActionEvent keyActionEvent)
+ void GetObjectData(SerializationInfo info, StreamingContext context)
+ ReplayEvent(SerializationInfo info, StreamingContext ctxt)</td></tr>
</table>

Figure 3.26: ReplayEvent Class Definition

The **`ReplayEvent`**s were stored in a **`ReplayDatabase`,** as seen in Figure 3.27. The key part here is the first variable with a List object containing **`ReplayEvent`**s. This array is what would be serialized to store into the binary file. The other variables indicated the file status, including the current frame ID for replaying at the current index in the **`replayEvents`** List, and the **excessDeltaTime** to maintain time consistency, using the **deltaTime** from the **ReplayEvent** objects and real-time to keep the frames taking the same amount of time. It should be noted, while not having a HMD as part of the Unity application, the frame rate would increase a lot. Therefore, limiting the frames per second (FPS) was necessary to be the same relative rate as during the experiment. The application ran at a sufficiently high frame rate for it not to be noticeable as a concern for the participants.

ReplayDatabase	
+ List<ReplayEvent> replayEvents + int replayEventID + float excessDeltaTime - string databaseCreation - string fileOpen	+ ReplayDatabase() + ReplayEvent getNextEvent() + void saveDatabase(string filename) + void loadDatabase(string filename) + getCreationData()

Figure 3.27: ReplayDatabase Class Definition

3.3 Results

The results section of this chapter will present the data collected from this experiment. Further analysis and comparison with other experiments' data are discussed in Chapter 7. The questionnaire results are presented in 3.3.1, then the results from the application data in 3.3.2. Discussion about the results and their application to the goals of this research are presented in section 3.4.

3.3.1 Questionnaire Results

This section will summarise participant responses to the two questionnaires. The results have been grouped based on the content and the order of appearance in the questionnaires. As indicated previously, the complete surveys can be found in Appendix A.2.

General Background

As discussed in section 3.2.1, 18 participants volunteered for this study. Of the 18 total participants, 10 were male, and 8 were female. Ages for participants included 3 participants under 21, 14 between 21 and 30, and 1 between 31 and 40. All participants were students currently studying at Flinders University. Students identified many different areas of study for their degrees, including Information Technology, Science and Education, Mechanical Engineering, Biomedical Engineering, Computer Science, Arts/Humanities, Robotics Engineering, and Digital Media. Only two participants had used AR before as part of other research. Two had used mobiles or tablets as part of other research participation. These were all noted as being positive experiences. Five participants had previously used AR before in a personal capacity. Four participants had used a HMD previously. Two had used the Oculus Rift before, one had used the Google Cardboard, and one the GearVR.

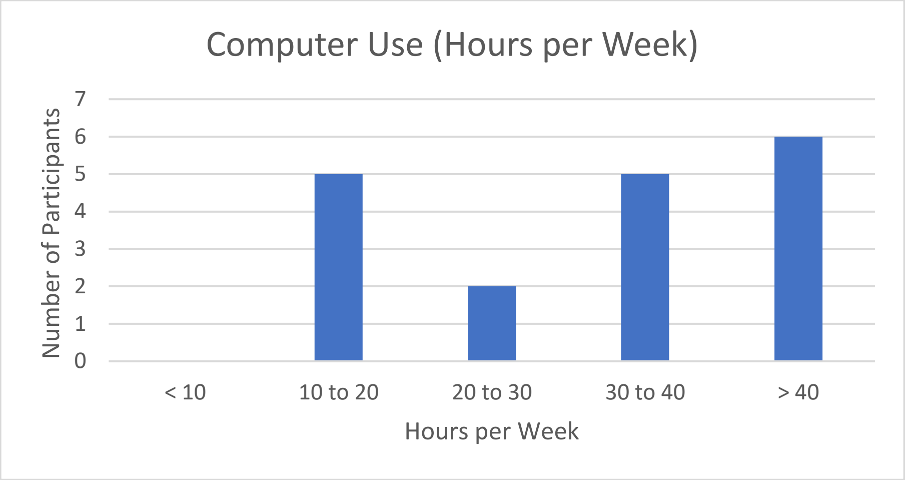


Figure 3.28: Computer Use (Hours per Week)

Figure 3.28 shows the computer use by participants. Given the nature of the participants, primarily technology-focused students, the computer usage per week was expected to be somewhat high for each participant. From the data captured, most people were using computers for at least 20 hours per week. From this value, we could surmise that most participants were comfortable with the use of computers. Figure 3.29 shows the time spent playing games per week. Most participants claimed to be playing games less than 10 hours a week. The answers may have been lower because all the participants completed the responses during a semester.

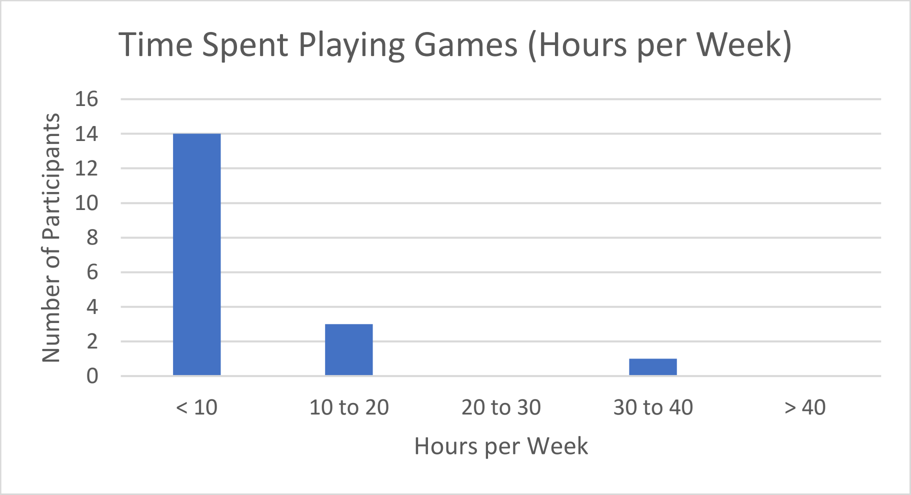


Figure 3.29: Time Spent Playing Games (Hours per Week)

Head-Mounted Display General Questions

From the pre-experiment questionnaire, a few non-gaming related HMD questions were asked. When asked about their interest in using HMDs for activities other than gaming, participants on average responded with a score of 7.5 (SD = 1.9, ranking on a Likert scale out of 10). Indicating on average, most were interested in some capacity. When asked more specifically about the usefulness for assisted living for the elderly or disabled, participants responded with 7.2 (SD = 1.9). Participants were asked their thoughts on the importance of social perception of wearing a HMD in a public setting; the average response was 6.6 (SD = 1.9). Several different factors affecting the wearing of a HMD in public were presented for the users to rank on a scale of 1 to 5, where 1 was the highest factor influencing HMD usage in public. Figure 3.30 shows an ordered set of average rankings. The figure represents the averaged priority ranking by participants for the factors affecting the use of a HMD. These factors were ranked from most important to least important. A lower average value indicates a more important factor of the users to the use of HMDs. The results mean, on average, comfort and usefulness of a device were more important to the participants than the size and outward appearance.

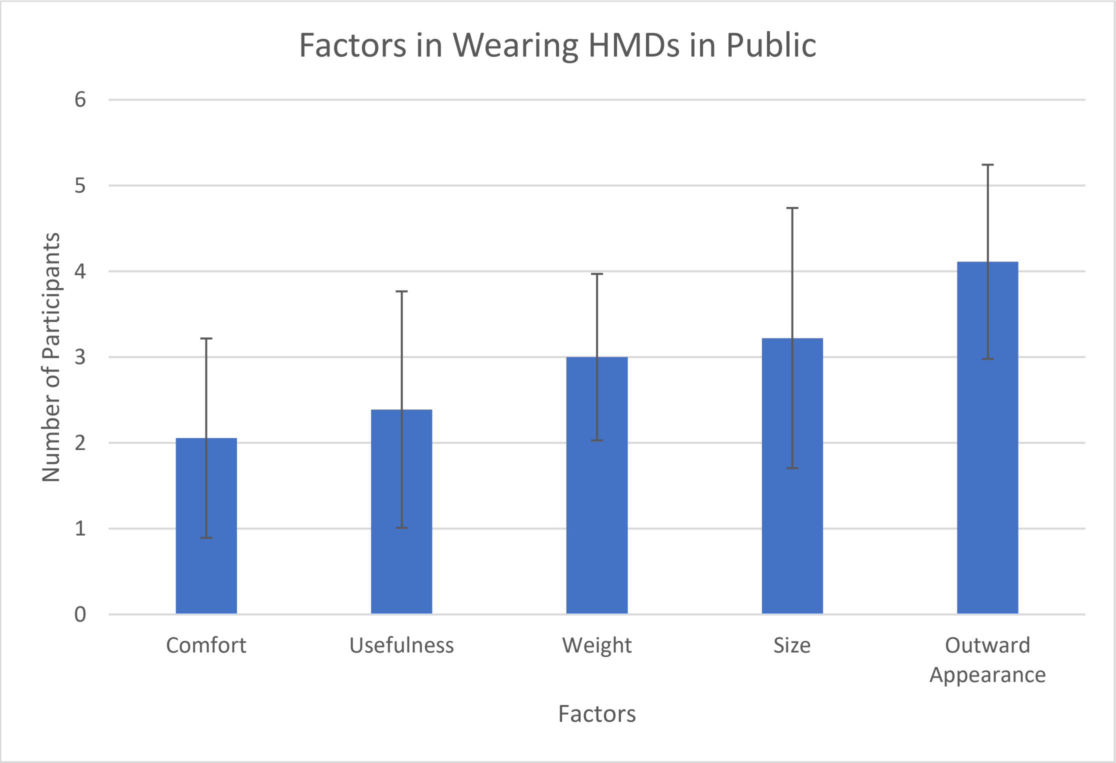


Figure 3.30: Factors in Wearing HMDs in Public

Input Method Effectiveness

The input method effectiveness was scored on a scale of 1 to 5 for every task. Where 1 was extremely ineffective, and 5 was extremely effective. Figure 3.31 (over) shows an overall average for input methods between all tasks. The data shows that participants felt the mouse was slightly more effective than the mobile input (t(17)=-3.31, p = <0.01). The HMD Only input fell slightly behind the other two (HMD Only vs Mouse was significant t(17)=4.46, p = <0.01, but HMD Only vs Mobile was not).

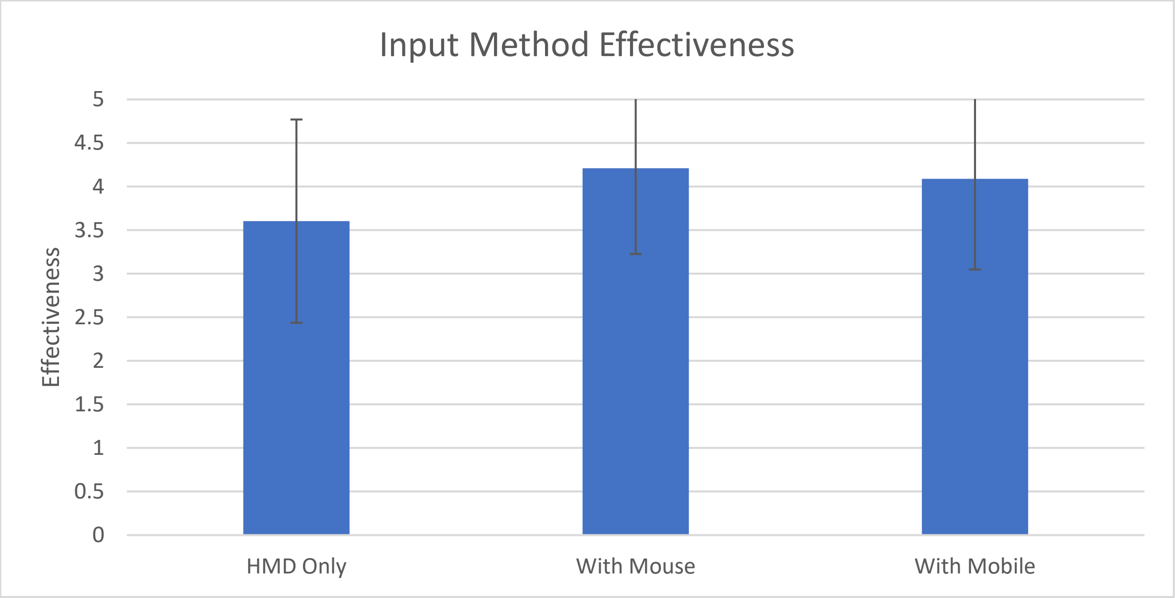


Figure 3.31: Input Method Effectiveness

In Figure 3.32 (over), the effectiveness of each input method is shown. From the figure, participants, on average, felt the level of effectiveness was the lowest for Task 5 for all input methods. The effectiveness had a similar average for other tasks with the Mouse and Mobile inputs with some slightly lower results for the HMD Only. The averages for each task can be compared to the complexity of each task. Task 5 with the maze was the most complex because it had a failure state and benefited the least from any advantage of using the HMD. Task 5 was almost functionally identical between the different input methods, so that it may be an example of bias toward a preferred input method. A similar discrepancy, perhaps due to bias, can be observed in the difficulty average for Task 5 in Figure 3.34.

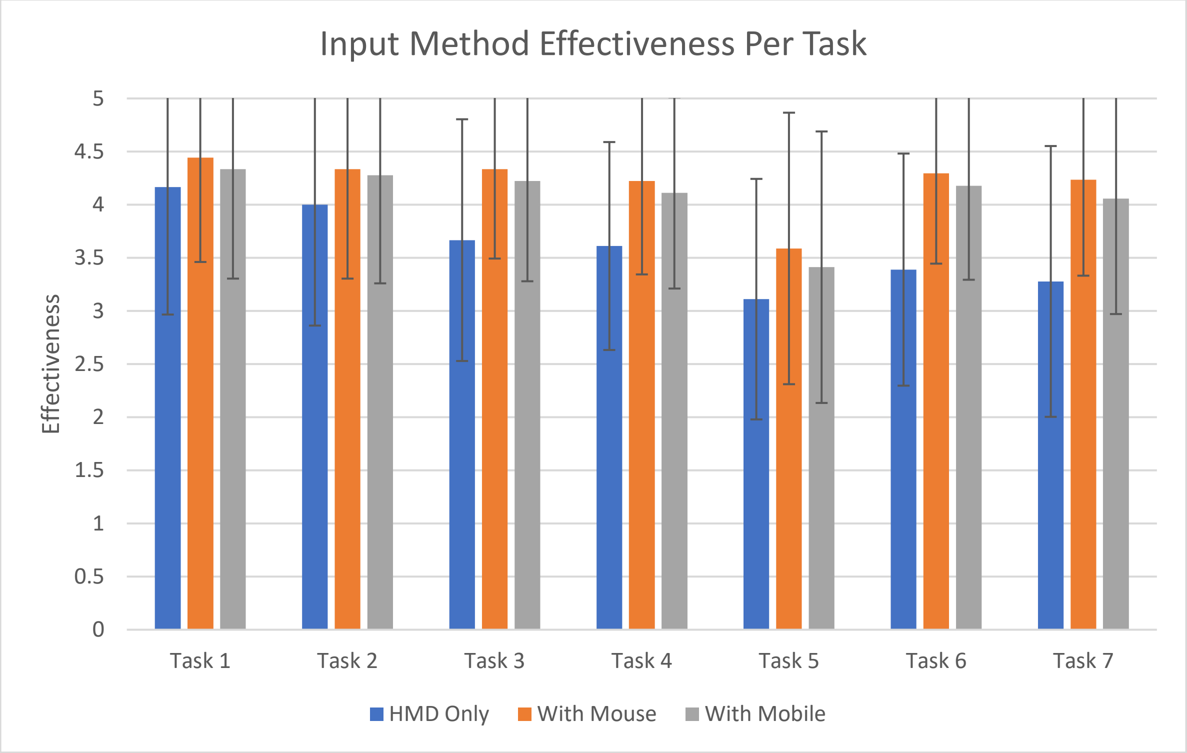


Figure 3.32: Input Method Effectiveness Per Task

The statistical significance was measured between each input for each task using a one-tail t-Test. Tasks 1, 2, and 5 showed no statistical significance between any comparison per task. For Task 3, the HMD Only data had a mean of 3.67, the Mouse had 4.33, and Mobile had 4.22, with HMD Only against Mouse (t(17)=-2.46, p = 0.01) and HMD Only against Mobile (t(17)=-1.86, p = 0.04) showing significance. For Task 4, only the HMD Only with a mean of 3.61 showed statistical significance (t(17)=-2.17, p = 0.02) when compared against Mouse with a mean of 4.22. Task 6 had statistical significance when comparing HMD Only (mean of 3.35) against Mouse (mean of 4.29, t(16)=-3.57, p = <0.01) and Mobile (mean of 4.18, t(16)=-3.04, p = <0.01). Finally, for Task 7, there was also statistical significance when comparing HMD Only (mean of 3.24) against Mouse (mean of 4.24, t(16)=-3.27, p = <0.01) and Mobile (mean of 4.06, t(16)=-2.53, p = 0.01).

Input Method Difficulty

The input method difficulty was scored on a scale of 1 to 5 for every task. Where 1 was very difficult, and 5 was very easy. In Figure 3.33 (over), participants found very similar difficulty with the mouse and mobile inputs, while the HMD Only was slightly more difficult.

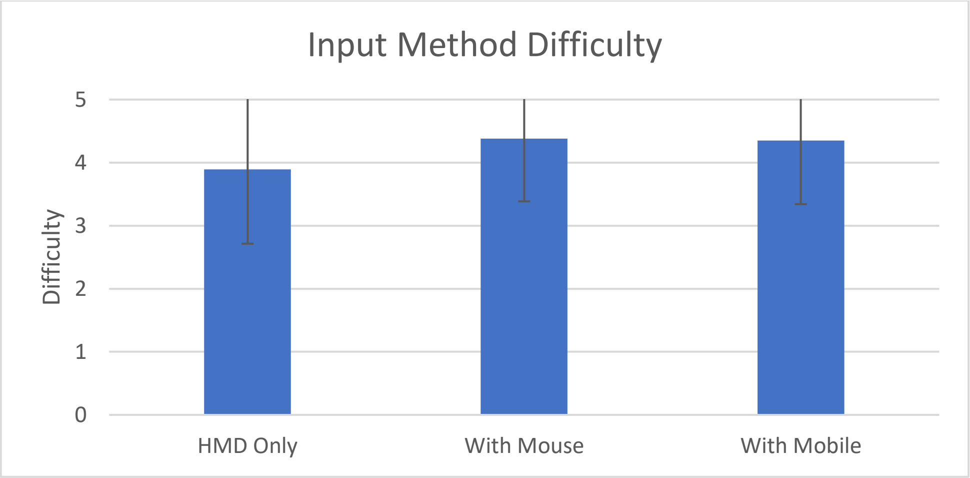


Figure 3.33: Input Method Difficulty

In Figure 3.34, the input method difficulty for each task is shown. When comparing this figure to Figure 3.32, it is interesting to note that the mobile and mouse inputs swap. Task 6 and Task 7 show higher effectiveness for the mouse input and slightly greater difficulty. Overall, the difficulty and effectiveness are very similar when compared between the figures. The HMD Only type input method was consistently scored slightly worse than the other input methods for difficulty.

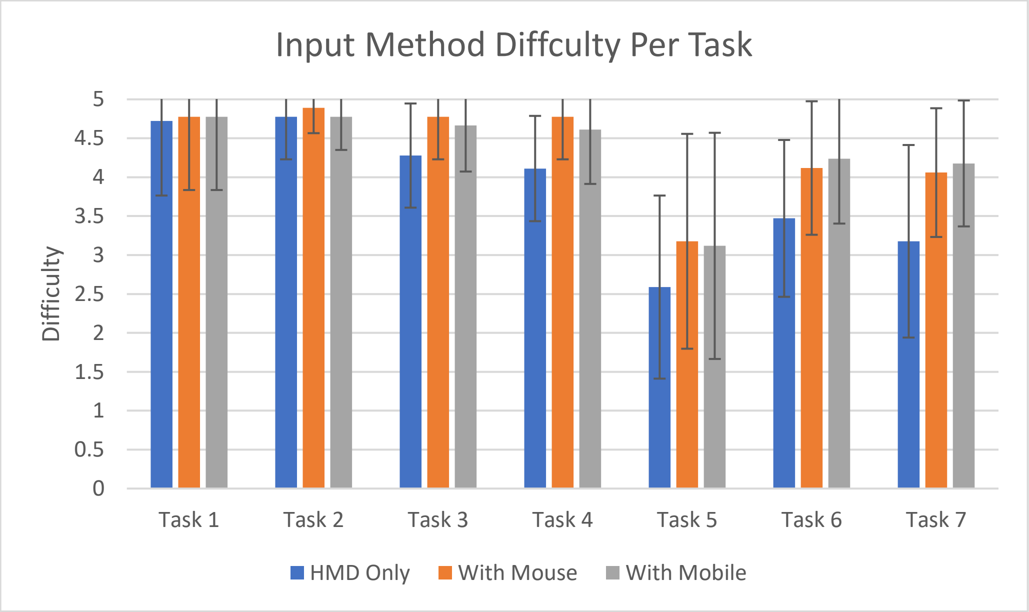


Figure 3.34: Input Method Difficulty Per Task

The data was compared for each task between each input with a one-tail t-Test. Most comparisons were found to be statistically significant. All comparisons for Task 1 were found to be statistically significant, with HMD Only (mean of 4.72) against Mouse (mean of 4.78, t(17)=-1.9, p = 0.04), HMD Only against Mobile (mean of 4.78, t(17)=-1.9, p = 0.04) and Mouse against Mobile were identical responses. No comparisons for Task 2 were found to be statistically significant. For Task 3, HMD Only (mean of 4.28) against Mouse (mean of 4.78, t(17)=-3.3, p = <0.01) was significant, and HMD Only against Mobile (mean of 4.67, t(17)=-2.67, p = 0.01) was also significant. For Task 4, HMD Only (mean of 4.11) against Mouse (mean of 4.78, t(17)=-4.43, p = <0.01) was significant, and HMD Only against Mobile (mean of 4.61, t(17)=-2.97, p = <0.01) was significant. For Task 5, HMD Only (mean of 2.59) against Mouse (mean of 3.18, t(16)=-1.86, p = 0.04) was significant. Tasks 6 and 7 were statistically significant between all input comparisons. For Task 6, HMD Only (mean of 3.47) against Mouse (mean of 4.12, t(16)=-3.34, p = <0.01) was significant, HMD Only against Mobile (mean of 4.24, t(16)=-3.72, p = <0.01) was significant, and Mouse against Mobile (t(16)=-2.08, p = 0.03) was significant. For Task 7, HMD Only (mean of 3.18) against Mouse (mean of 3.06, t(16)=-4.14, p = <0.01) was significant, HMD Only against Mobile (mean of 4.18, t(16)=-4.08, p = <0.01) was significant, and Mouse against Mobile (t(16)=-2.08, p = 0.03) was significant.

Input Method Fatigue

The input method fatigue asked participants to score how much fatigue they felt while completing each of the tasks with each input. These were scored on a scale of 1 to 5. Where 1 was no fatigue, and 5 was extreme fatigue. None of the tasks required excessive physical activity. All tasks were completed by the participant standing with the input device (where applicable) held in their dominant hand. As seen in Figure 3.35, the fatigue scores from participants overall for all tasks were very similar between all input methods.

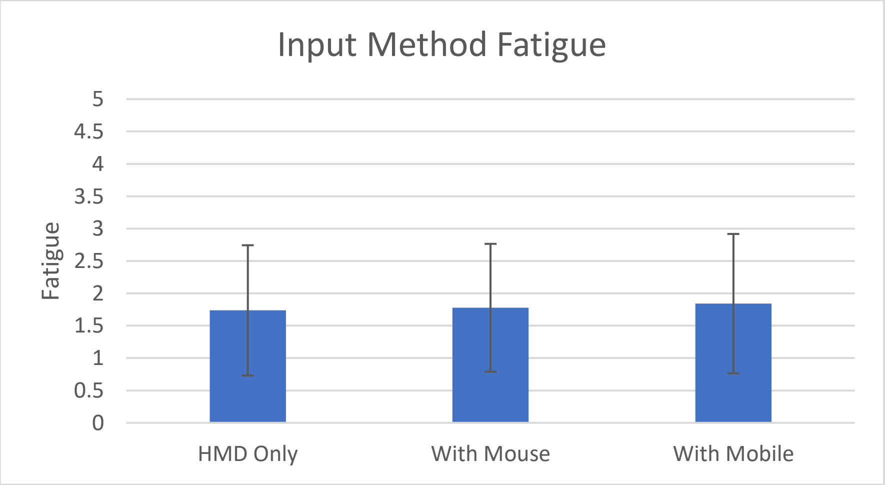


Figure 3.35: Input Method Fatigue

In Figure 3.36, the fatigue score for each task is shown. As expected, Task 5 had the highest fatigue due to the stress of navigating the maze with precise movement. Participants reported lower fatigue for Task 1 to 4 with the HMD Only and slightly higher for Task 6 and 7. Overall the fatigue averages between roughly 1 to 2 indicate there was low fatigue from the experiment, on average.

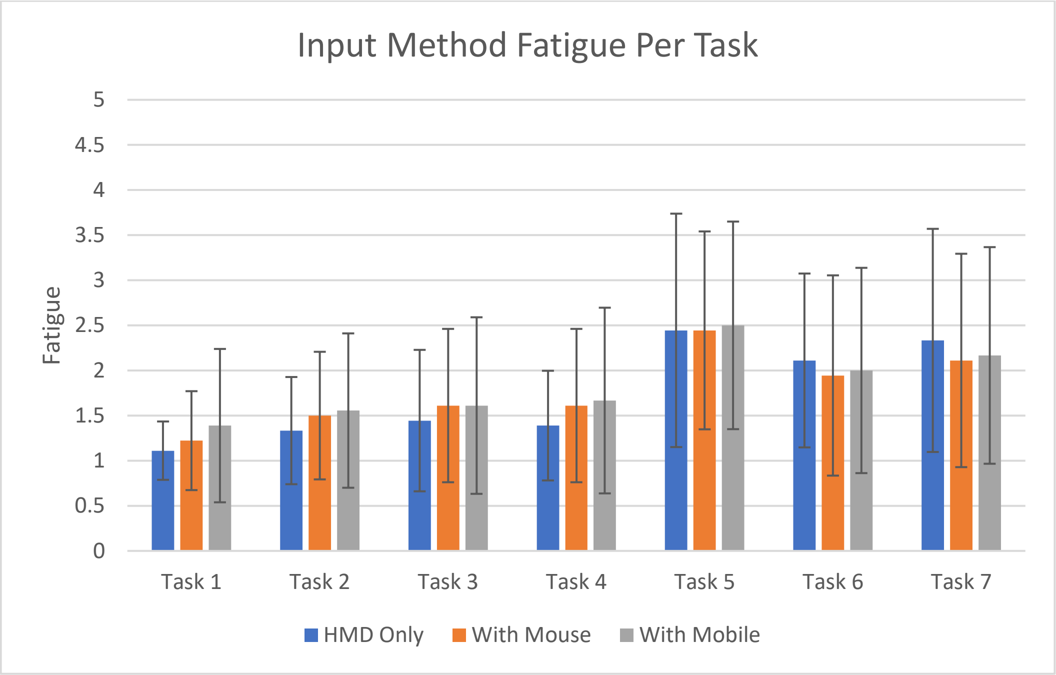


Figure 3.36: Input Method Fatigue Per Task

Similar to the Effectiveness and Difficulty, the Fatigue was measured for statistical significance between the different input types. Task 1 was statistically significant for all three comparisons, with HMD Only (mean of 1.11) against Mouse (mean of 1.22, t(17)=-2.11, p = 0.02) was statistically significant, HMD Only against Mobile (mean of 1.39, t(17)=-1.85, p = 0.04) was significant, and Mouse against Mobile (t(17)=-1.79, p = 0.05) was significant. Tasks 2, 3, 4 and 5 had no statistical significance when comparing inputs. Task 6 had statistical significance when comparing Mouse (mean of 1.94) and Mobile (mean of 2, t(17)=-1.9, p = 0.04). Task 7 had statistical significance when comparing Mouse (mean of 2.11) and Mobile (mean of 2.17, t(17)=-1.9, p = 0.04).

Input Method Preference

Participants dramatically shifted their opinions about their input preferences between the pre-experiment questionnaire and post-experiment questionnaire. The opinion shift is shown in Figure 3.37. Before the experiment was conducted, 14 participants were expecting to prefer using the mouse. After completing the experiment, only 6 participants of the 18 preferred the mouse. With an equal number of 6 preferring the Oculus Rift by itself. Leaving 4 preferring the mobile input and 2 preferring to indicate "other". Participants preferring the HMD Only type input preferred it for the simplicity, not needing another device, increased focus on the visual interactions, and the fun. Participants preferred the mouse input for the increased speed of selection, ease of use, and familiarity with the device. Participants preferring the mobile input preferred it for comfort, smooth interactions, and speed of interactions.

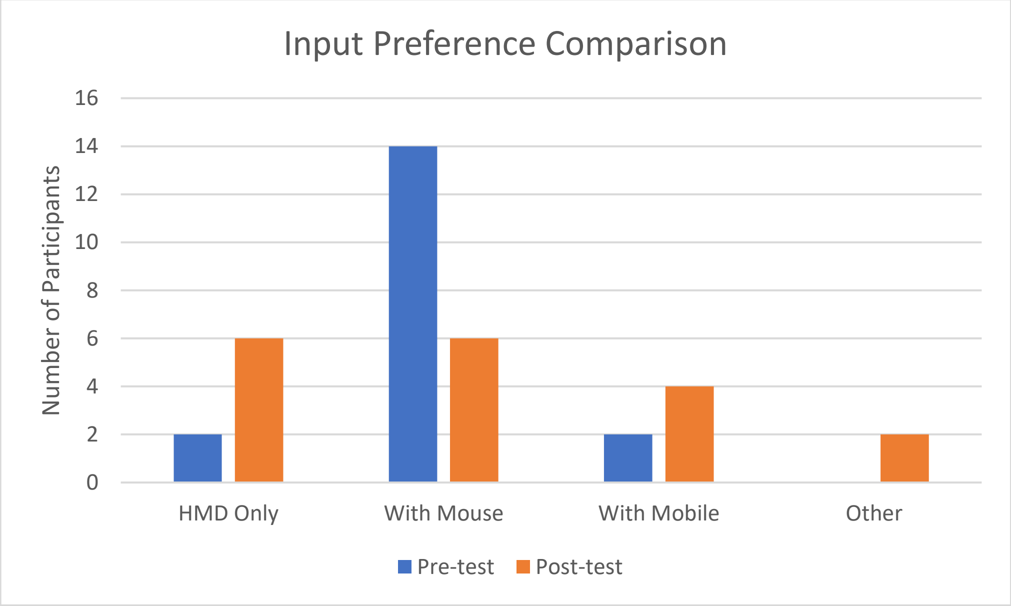


Figure 3.37: Input Preference Comparison

Only two participants said they would prefer to use some other form of input method, but others also provided some ideas of what they may wish to see. The following lists options for alternate input methods suggested by participants.

- Voice Control
- Oculus + Gamepad
- Mouse Only (independent to head control)
- Full Touch Screen
- Mobile with Haptic Feedback
- Eye Tracking

System Usability Scale Type Questions

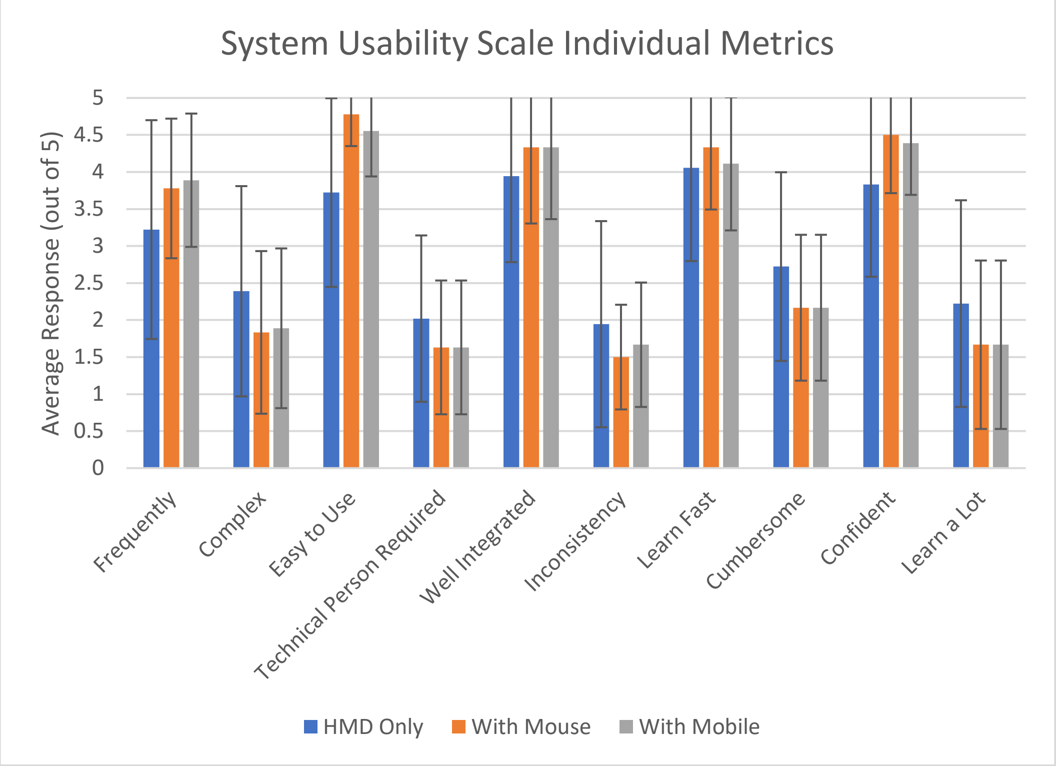


Figure 3.38: System Usability Scale Individual Metrics

Figure 3.38 shows the individual metrics used to calculate the System Usability Scale (SUS) score. These are ordered as they appeared in the questionnaire. It should be noted that the "Technical Person Required" value is an average of the complex and "learn a lot" values, as it was not included in the original questions.

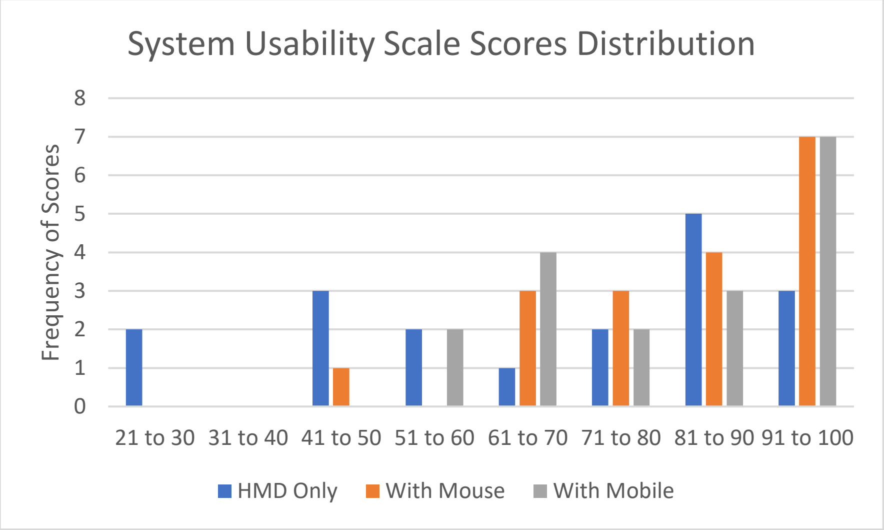


Figure 3.39: System Usability Scale Scores Distribution

Figure 3.39 (previous page) shows the distribution of SUS scores for each of the different input methods. A few significantly lower scores led to the overall average drop for the HMD Only type interactions compared to the gradual climb to the slightly extreme high scores for the mouse and mobile type inputs.

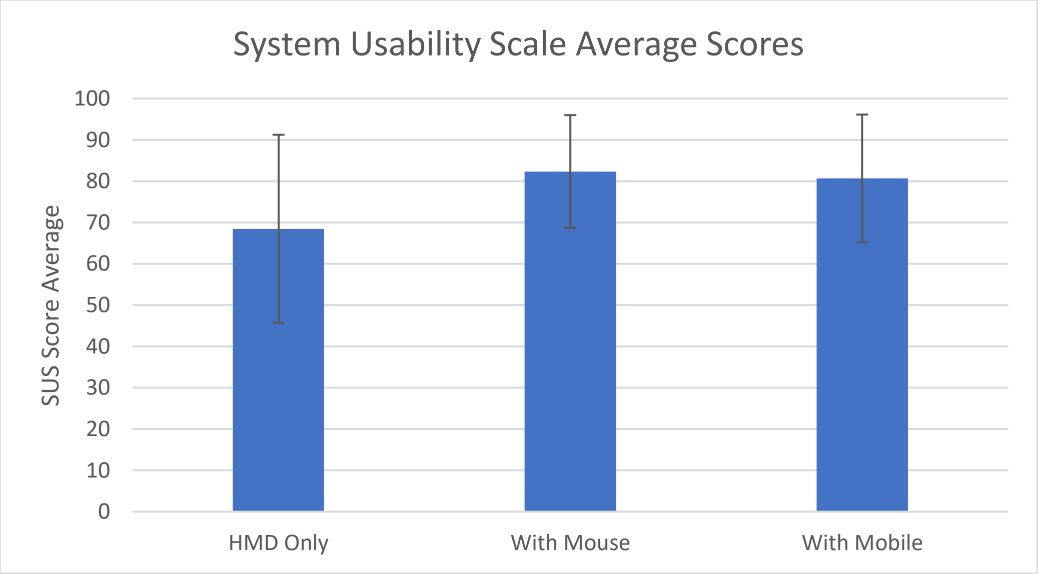


Figure 3.40: System Usability Scale Average Scores

Figure 3.40 shows the average scores by interaction type. HMD Only had a score of 68.47 (SD = 22.79), With Mouse had a score of 82.36 (SD = 13.65), And With Mobile had a score of 80.69 (SD = 15.45). HMD Only's score falls within a C grade and can be considered OK, with a Marginal level of being acceptable and is considered a Passive score for the Net Promoter Score (NPS). Both the With Mobile and With Mouse average SUS Scores can be considered A grade representing Excellent scores that are both Acceptable and are Promoters using the NPS. The raw averages indicate the HMD Only was worse received, but it is still within an acceptable range. The scores were compared with a one-tail t-Test, the comparisons between HMD Only and Mouse (t(17)=-2.89, p = 0.01) and HMD Only and Mobile (t(17)=-2.49, p = 0.01) were shown to be statistically significant. Many additional comparisons against the SUS score have been made in section 3.3.3.

3.3.2 Application Data Results

The results covered in this section are data captured from the application while being used by the participant. Some of the data was also calculated from using the replay system as described in 3.2.13. However, this only applies to aspects for Task 7. Most of the data collected looked at how long tasks took to complete, with some data related to the ways participants failed Task 5 and how Task 7 was handled.

Task Completion Rates

Before looking at other relevant application data, it is important to show the completion rates for each task, as not all tasks were successfully completed by every participant. In Table 3.1, the completion rates for each task with each input method are shown.

Table 3.1: Task Completion Rates

	Task 1	Task 2	Task 2_2	Task 3	Task 4	Task 5	Task 6	Task 7
HMD Only	18	18	18	18	18	16	17	8
With Mouse	18	18	18	18	18	14	16	7
With Mobile	18	18	18	18	18	13	15	6

Problems with the hardware were the main causes of non-completion of tasks, as seen in various tasks not having a total of 18 completions recorded. The HMD's orientation sensor often failed during the experiment requiring troubleshooting and restarting of the application. When this happened, the participant would experience the virtual space continuing to rotate without any interaction (i.e., head movement) from them. In some cases, the participants opted to stop the experiment once they were very close to the end. Any time this hardware fault occurred, an intervention would occur to restart the application, and the participant was skipped straight back to the final stage. Some participants misunderstood Task 7 and did not understand that the cubes could only be matched to specific locations. Some attempted to place the cubes in any location. The way participants executed the tasks will be discussed in greater detail once all application data results have been presented in this chapter.

Task Completion Times

Task completion times are shown as averages in Table 3.2. All values are in seconds. Figure 3.41 shows a diagram with the data represented.

Table 3.2: Mean Average Task Completion Times (seconds)

	Task 1	Task 2	Task 2_2	Task 3	Task 4	Task 5	Task 6	Task 7
HMD Only	4.5 (SD = 2.7)	18.7 (SD = 4.7)	33.2 (SD = 10.3)	5.3 (SD = 2.2)	7.3 (SD = 1.6)	93.8 (SD = 81.6)	64.3 (SD = 33.6)	119.8 (SD = 74)
With Mouse	7.4 (SD = 6.8)	18.9 (SD = 16)	22.1 (SD = 5)	3.5 (SD = 1.5)	5.5 (SD = 2.1)	80.8 (SD = 53.6)	23.6 (SD = 12.9)	55.6 (SD = 57.2)
With Mobile	4.3 (SD = 4)	11.6 (SD = 3.7)	21.8 (SD = 4)	3 (SD = 1.3)	5.5 (SD = 2.2)	45.7 (SD = 34.7)	18.2 (SD = 5)	28 (SD = 4)

In Figure 3.41, the difference in completion time is very evident. Tasks 1 to 4 each were quick to complete as they only involved the selection of cubes. The HMD Only type interaction was slower because of the selection time mechanic for each cube. There is a particularly high variance seen for Task 5 and Task 7 where participants were learning, having technical difficulties, or struggling with the mechanics of the tasks.

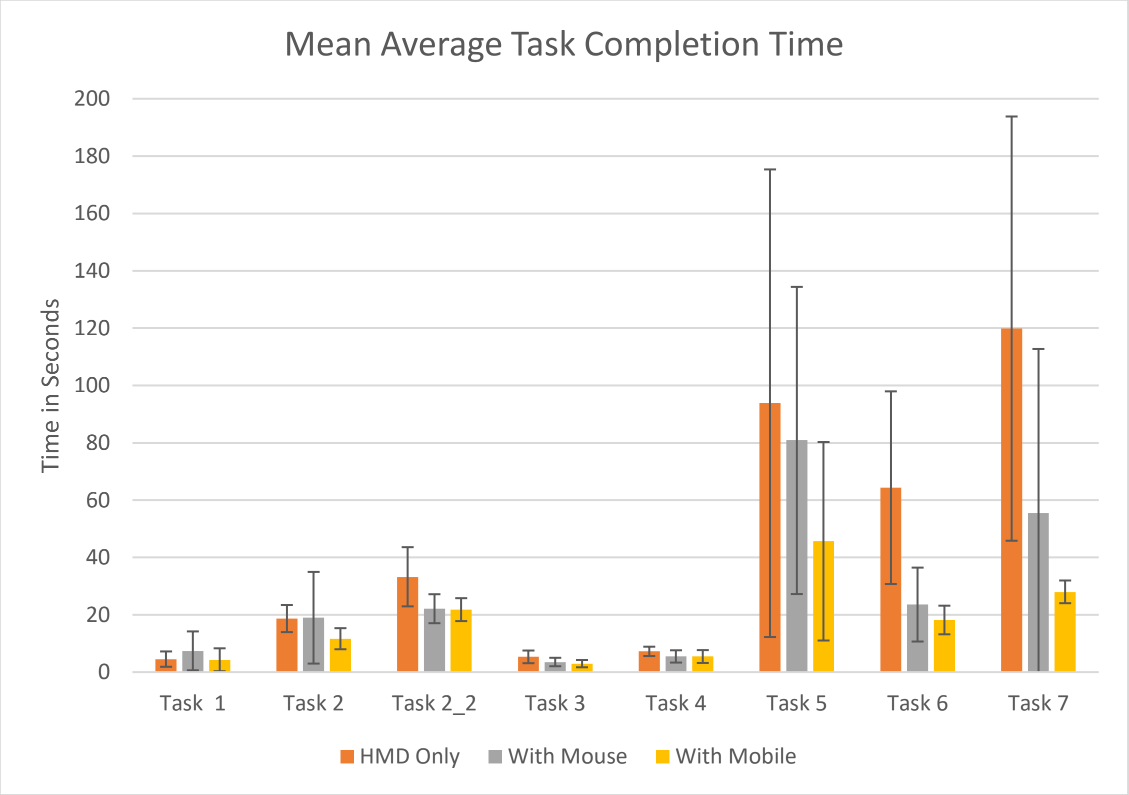


Figure 3.41: Mean Average Task Completion Time

The following sets of data will show how the times compared to a minimum action count to provide an alternative view of the task completion times. The optimal action counts have been generated for this data. Only Task 5, 6, and 7 are likely to have varying action counts depending on how participants selected to solve or fail to solve the tasks. For comparison purposes, the optimal action count was used. The following list will summarise the expected action counts with how they are reached.

- Task 1: Select 1 Cube (1 action)
- Task 2: Select 14 Cubes (14 actions)
- Task 2_2: Select 24 Cubes (24 actions)
- Task 3: Select 2 Cubes (2 actions)
- Task 4: Select 4 Cubes (4 actions)
- Task 5: Select 1 Cube, Move Cube to End (2 actions)
- Task 6: Select Cube, Select Colour Menu, Select Green, Select Back, Select Size, Select Large, Select Back, Select Move, Complete Movement (9 actions)
- Task 7: Select Cube 1, Select Green, Select Large, Select Move, Complete Move, Select Cube 2, Select Blue, Select Small, Select Move, Complete Movement, Select Cube 3, Select Small, Select Move, Complete Movement (14 actions)

Table 3.3: Mean Average Task Time / Min Action Count

	Task 1	Task 2	Task 2_2	Task 3	Task 4	Task 5	Task 6	Task 7
Min Action Count	1	14	24	2	4	2	9	14
HMD Only	4.53	1.34	1.38	2.66	1.81	46.91	7.15	8.56
With Mouse	7.40	1.36	0.92	1.76	1.36	40.42	2.62	3.97
With Mobile	4.30	0.83	0.91	1.48	1.36	22.83	2.02	2.00

Table 3.3 shows the modified time per task data as a time per action. This type of representation does not work as well for Task 5 because of the failure state. Many participants triggered a failure state, as will be discussed in the next section. For that reason, Task 5 was removed from Figure 3.42. In Figure 3.42 (over), it can be observed that the time per action was mostly similar despite the 0.5 second selection time per cube for the HMD only interactions. It is evident for Task 6 and Task 7 that participants took many additional actions,

both from the time taken to work out how to complete the necessary actions and making mistakes leading to additional actions required to fix mistakes.

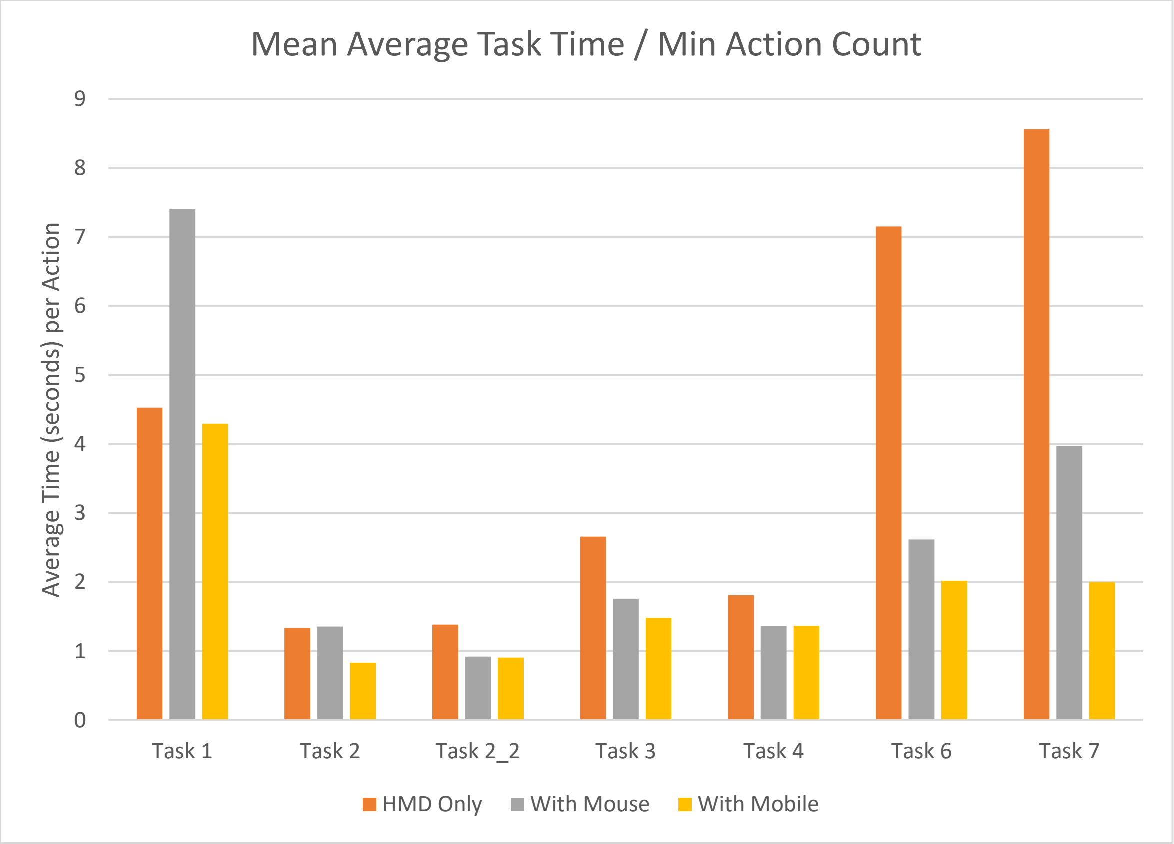


Figure 3.42: Mean Average Task Time / Min Action Count

The spike seen in Figure 3.42 for Task 1 can be explained because the first task only requires a single action. Participants would take a few seconds to adapt to each new input method. It was not recorded for the use of data at the time, but some may have adjusted their headset or been finding where the mouse was blindly during Task 1, adding a couple of seconds average. The first task introduced the action that was then used across Task 2 to Task 4, so Task 1 averages were likely an anomaly based on the number of actions and becoming used to the action.

Task 5 Wall Collisions

As was indicated in sections 3.2.9 and 3.2.13, data was collected detailing which walls participants were colliding with during the task. Figure 3.43 (over) provides a duplicate of the earlier figure to help with understanding where the walls were located for the data. The four light blue wall cubes numbered 49 to 52 (with red labels) had the easy mode state available. The easy mode state would remove the wall after 5 collisions with that same wall.

There are two types of data to consider when looking at what walls were collided with during the task: total collisions and unique hits. Total collisions are the total number of times, across all participants, that a wall was collided with, either with a specific input method or overall. Unique hits are the number of times the wall was uniquely collided with by a participant. Viewing the two together shows the walls repeatedly hit by only a few participants compared to those hit by many participants. Table 3.4 (over) shows the summary of collision data for each input type, separated into total hits and unique hits. As can be seen from the bottom right, five easy modes were triggered, all on the same cube. Wall 51 is the cube first encountered as a slightly more difficult point for navigation. Four of these easy modes were triggered on the HMD Only input method and one during the mouse input method.

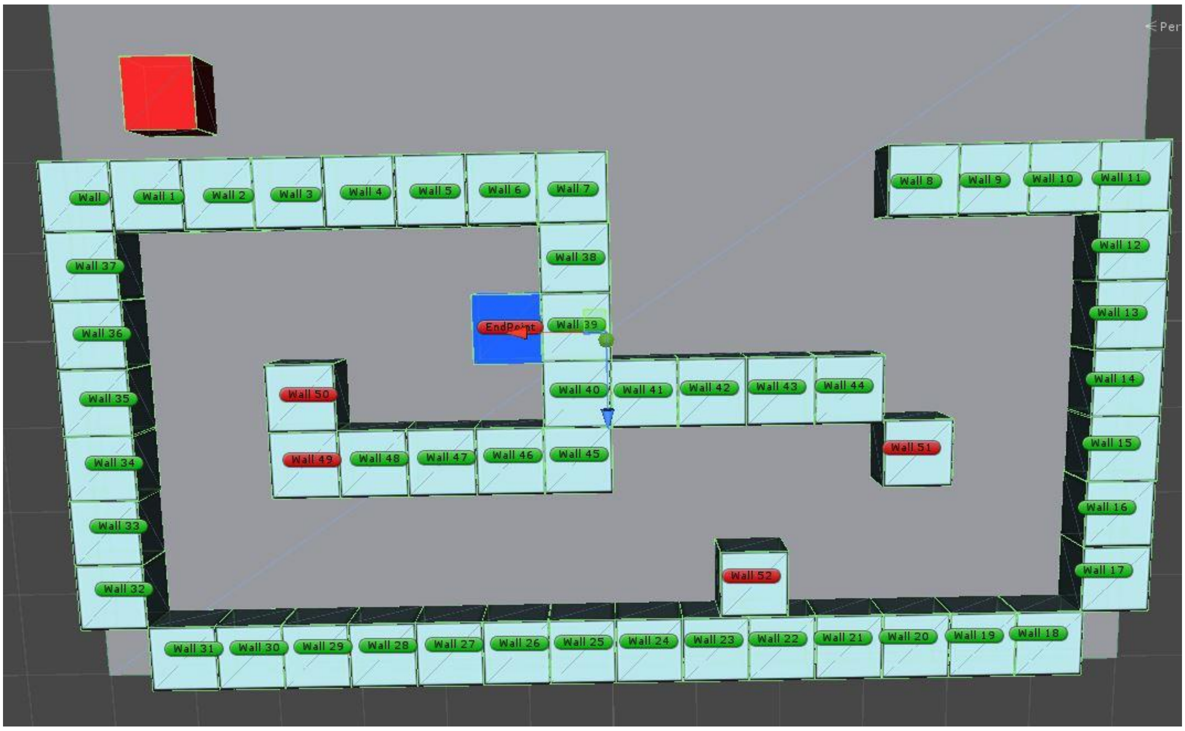
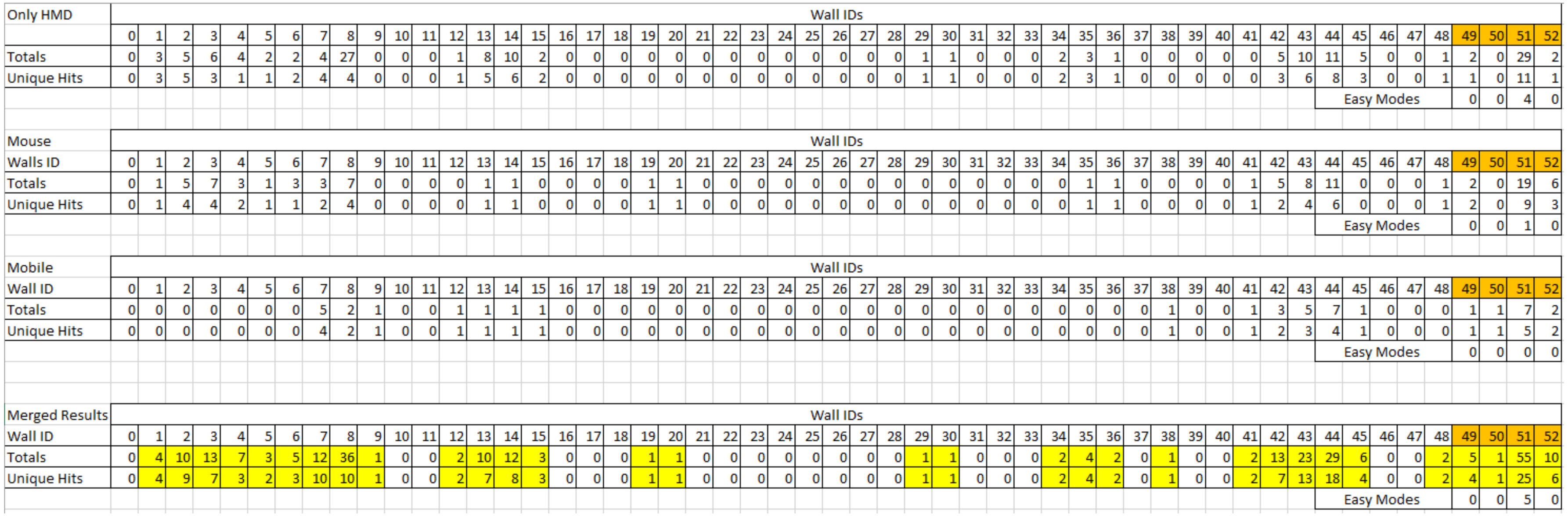


Figure 3.43: Task 5 Wall Object IDs

Table 3.4: Wall Collision Data



Only HMD	Wall IDs																																																				
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52
Totals	0	3	5	6	4	2	2	4	27	0	0	0	1	8	10	2	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	2	3	1	0	0	0	0	0	5	10	11	5	0	0	1	2	0	29	2
Unique Hits	0	3	5	3	1	1	2	4	4	0	0	0	1	5	6	2	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	2	3	1	0	0	0	0	0	3	6	8	3	0	0	1	1	0	11	1
																																												Easy Modes					0	0	4	0
Mouse	Wall IDs																																																				
Walls ID	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52
Totals	0	1	5	7	3	1	3	3	7	0	0	0	0	1	1	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	1	5	8	11	0	0	0	1	2	0	19	6
Unique Hits	0	1	4	4	2	1	1	2	4	0	0	0	0	1	1	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	1	2	4	6	0	0	0	1	2	0	9	3
																																												Easy Modes					0	0	1	0
Mobile	Wall IDs																																																				
Wall ID	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52
Totals	0	0	0	0	0	0	0	5	2	1	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	3	5	7	1	0	0	0	1	1	7	2
Unique Hits	0	0	0	0	0	0	0	4	2	1	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	2	3	4	1	0	0	0	1	1	5	2
																																												Easy Modes					0	0	0	0
Merged Results	Wall IDs																																																				
Wall ID	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52
Totals	0	4	10	13	7	3	5	12	36	1	0	0	2	10	12	3	0	0	0	1	1	0	0	0	0	0	0	0	0	1	1	0	0	0	2	4	2	0	1	0	0	2	13	23	29	6	0	0	2	5	1	55	10
Unique Hits	0	4	9	7	3	2	3	10	10	1	0	0	2	7	8	3	0	0	0	1	1	0	0	0	0	0	0	0	0	1	1	0	0	0	2	4	2	0	1	0	0	2	7	13	18	4	0	0	2	4	1	25	6
																																												Easy Modes					0	0	5	0

Figure 3.44 presents this data as a comparison of total collisions versus unique collisions. It shows a rough trend in the relationship between unique and total collisions. As more unique collisions occur, there appears an associated increase in total collisions.

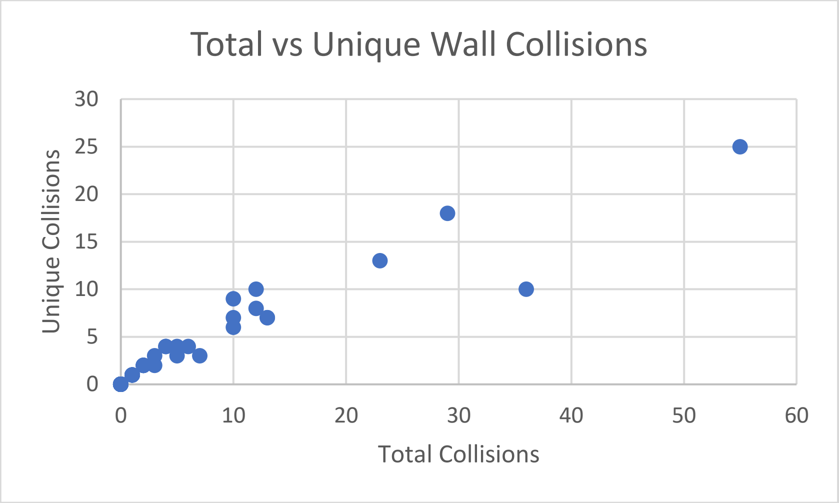


Figure 3.44: Total vs Unique Wall Collisions

Task 7 Interactions

Task 7 exhibited some issues during testing, which led to the win condition not always successfully registering, even though the puzzle was correctly solved. In some cases, the participant had issues with the HMD malfunctioning with a continuous rotation issue, as outlined previously. The participants suffering from this issue often found it impossible to complete the task in full. To still provide some data from how participants fared in this task, some smaller subsets of the data were used to calculate the results. Figure 3.45 shows the completion of Task 7 based on three different types of conditions.

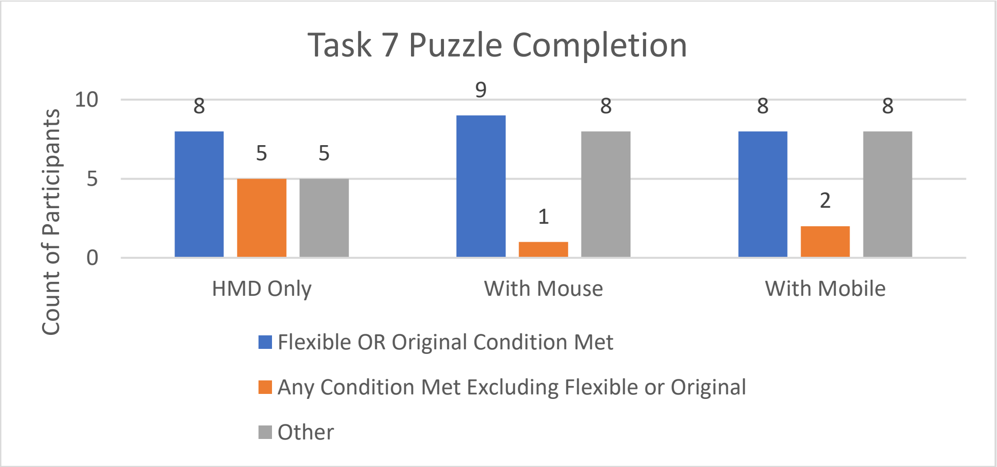


Figure 3.45: Task 7 Puzzle Completion

Meeting a Flexible or Original Condition meant it was automatically estimated from log data that the participant had successfully completed the task in full. The Any Condition Type indicated a partial solution by participants, where either the participant had not followed through, or the solution was not finished. And Other typically indicates an error occurred, resulting in data that could not apply to the other two categories. All the remaining results will be based on the subset of those who completed the puzzle either by detection of the original win condition or estimated (flexible) win condition.

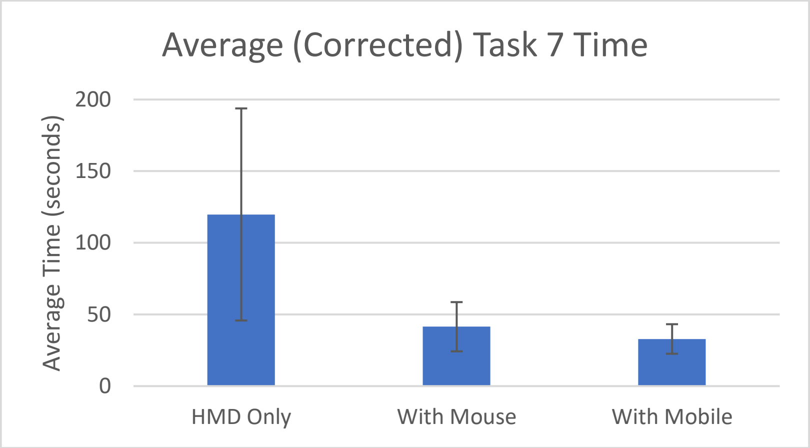


Figure 3.46: Average (Corrected) Task 7 Time

shows the average time for completing the task. The standard deviation is very high for the HMD Only type input, showing that some participants completed the task quickly. Many participants who took significantly longer misunderstood that the objects to match had to be moved to specific locations. Once properly understood, the time to complete the puzzle dropped dramatically with the other two input types.

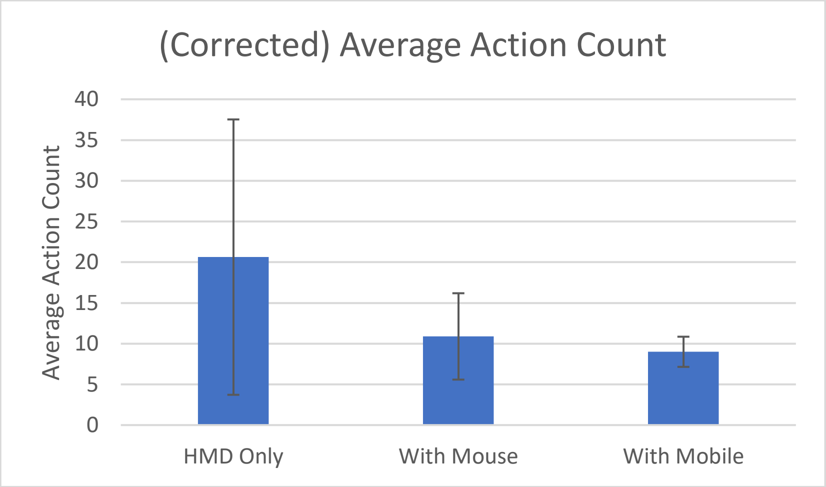


Figure 3.47: (Corrected) Average Action Count

Figure 3.47 (previous page) shows the number of actions taken on average to complete the tasks. The action count data correlates with the amount of time taken in (previous page). A perfectly executed puzzle completion would take 8 actions to complete. The second and third times this was completed with the other interaction methods had most participants approaching this number. Figure 3.48 shows the combination of average time per average action to represent the time taken on average to complete each action.

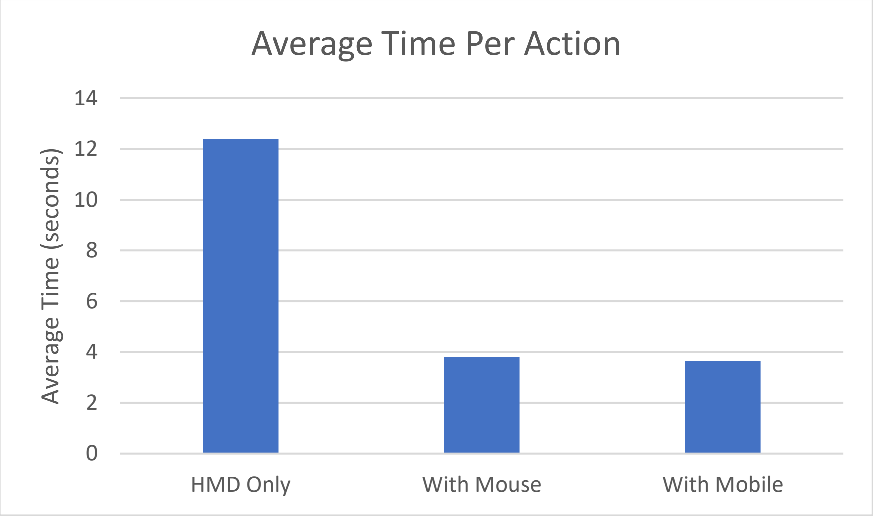


Figure 3.48: Average Time Per Action

3.3.3 Result Analysis

This section analyses specific comparisons within the data for the first experiment. For data analysis against the other experiments, see Chapter 7. The comparisons here look at the relationship between principally SUS scores against categories the participants selected.

SUS Scores per Age for Each Input Method

Figure 3.49 (over) shows the relationship between age and SUS scores. The 3 participants in the less than 21 age had average SUS scores of 50 (SD = 25.37) for HMD Only, 81.66 (SD = 14.64) for Mouse, and 83.33 (SD = 22.4) for Mobile. Suggesting those participants preferred the Mouse and Mobile by a large margin. The 14 participants in the 21 to 30 age bracket had average SUS scores of 70.89 (SD = 21.43) for HMD Only, 81.96 (SD = 14.35) for Mouse, and 79.46 (SD = 15) for Mobile. The single participant in the 31 to 40 age bracket had SUS Scores of 90 for all three inputs. An ANOVA comparison for each input method did not reveal any significance in comparing age against SUS for any input method. With most of the population data bracket skewed toward the 21 to 30 population, this is expected.

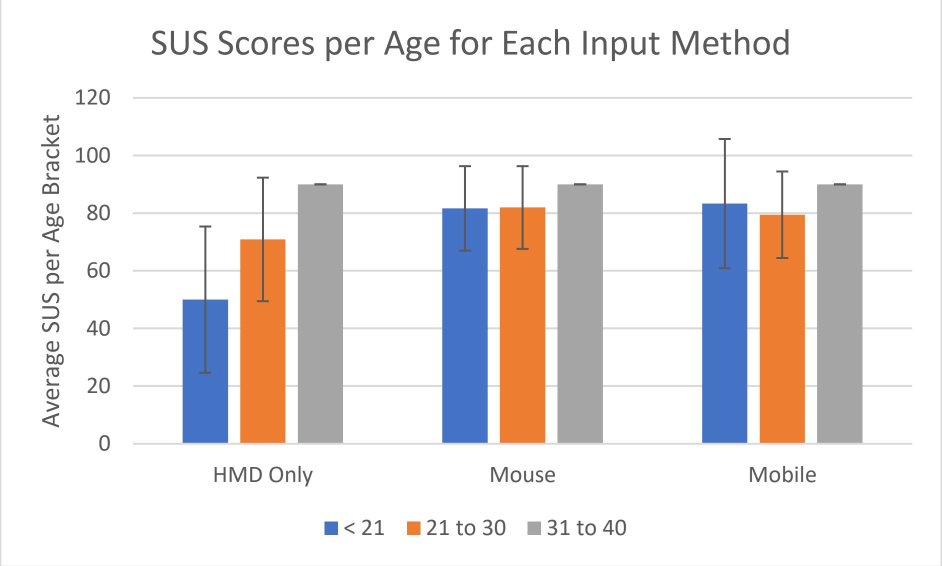


Figure 3.49: SUS Scores per Age for Each Input Method

SUS Scores per Gender for Each Input Method

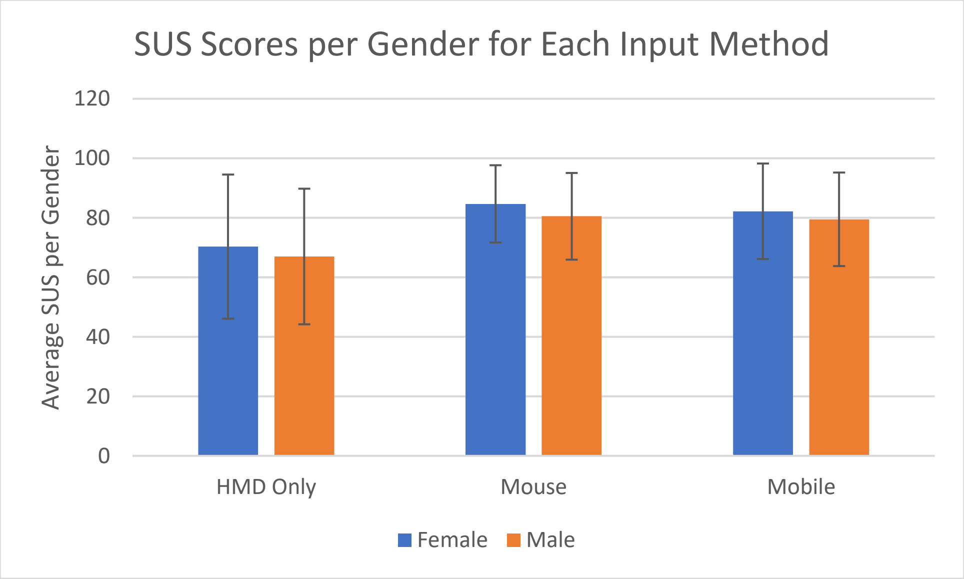


Figure 3.50: SUS Scores per Gender for Each Input Method

Figure 3.50 shows a comparison between SUS scores based on the gender of the participants. There is a reasonable observable comparison with the relatively well-split population, including 8 female and 10 male participants. Both genders from observation provided similar average responses for each input method. Female participants provided SUS scores of 70.31 (SD = 24.22) for HMD Only, 84.69 (SD = 12.99) for Mouse, and 82.19 (SD = 16.06) for Mobile. Male participants responded with slightly lower averages for all inputs, with 67 (SD = 22.79) for HMD Only, 80.5 (SD = 14.57) for Mouse, and 79.5 (SD = 15.71) for Mobile. An ANOVA analysis for each input mode did not yield any significance relating to the input modes for gender use.

SUS Scores per Computer Use (per week) for Each Input Method

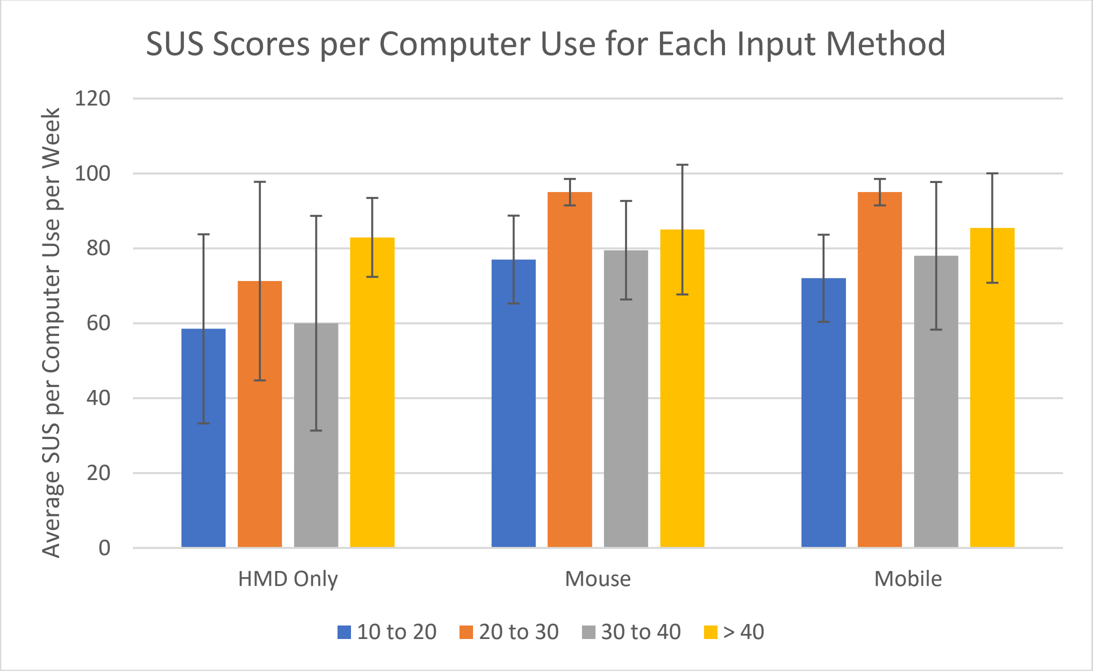


Figure 3.51: SUS Scores per Computer Use for Each Input Method

Figure 3.51 shows a comparison between the time participants reported they use computers weekly against their calculated SUS scores. The 5 participants who reported spending 10 to 20 hours per week on the computer had marginally lower SUS scores on average across all input methods. Those in the 10 to 20 hours per week range had averages of 58.5 (SD = 25.25) for HMD Only, 77 (SD = 11.73) for Mouse, and 72 (SD = 11.64) for Mobile. Those in the 20 to 30-hour range had a high variance for HMD Only SUS scores compared to their very high Mouse and Mobile scores. The 2 participants in this group had average SUS scores of 71.25 (SD = 26.5) for HMD Only and 95 (SD = 3.53) for both Mouse and Mobile. The 5 participants in the 30 to 40-hour range had average SUS scores of 60 (SD = 28.67) for HMD Only, 79.5 (SD = 13.16) for Mouse, and 78 (SD = 19.72) for Mobile. The remaining 6 participants in the greater than 40-hour range had average SUS scores of 82.92 (SD = 10.54) for HMD Only, 85 (SD = 17.32) for Mouse, and 85.42 (SD = 14.61) for Mobile. An ANOVA comparison for each input method on the SUS data compared to computer use did not reveal any significance.

SUS Scores per Game Use (per week) for Each Input Method

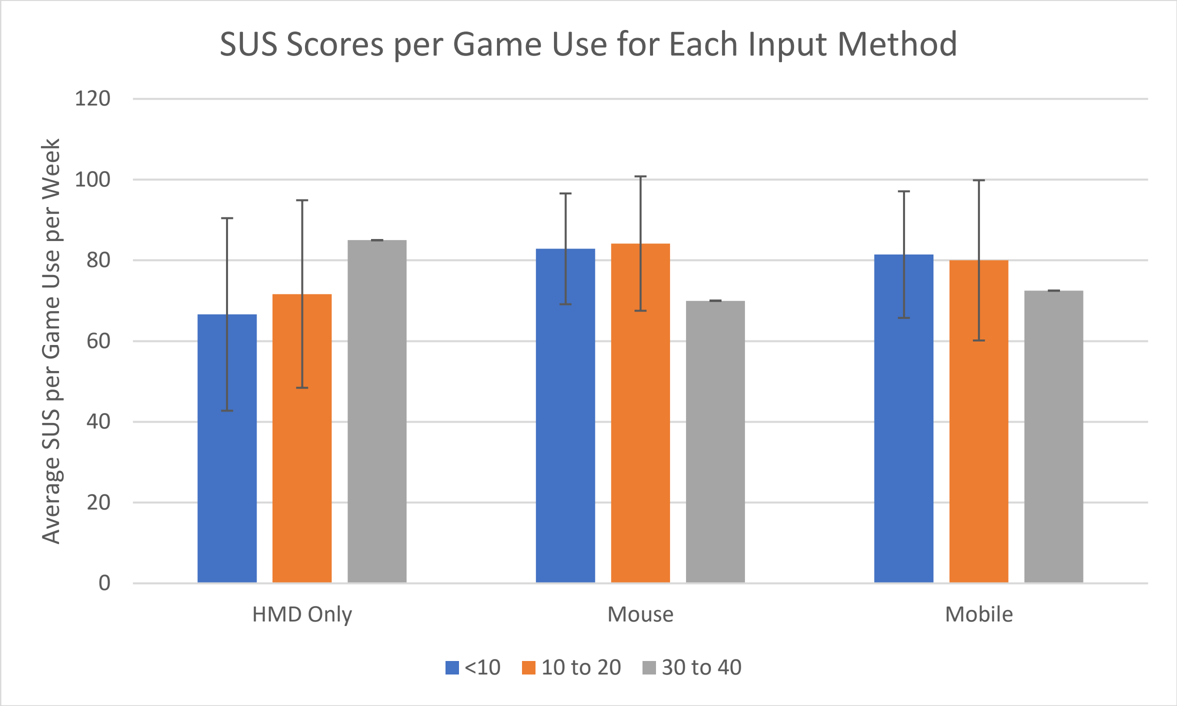


Figure 3.52: SUS Scores per Game Use for Each Input Method

Figure 3.52 shows participants' SUS scores categorised based on game use per week for each input method. Most participants (14) reported less than 10 hours per week with average SUS scores of 66.6 (SD = 23.85) for HMD Only, 82.86 (SD = 13.72) for Mouse, and 81.4 (SD = 15.68) for Mobile. Three participants reported 10 to 20 hours of games per week with average SUS scores of 71.67 (SD = 23.23) for HMD Only, 84.17 (SD = 16.65) for Mouse, and 80 (SD = 19.84) for Mobile. The single participant in the 30 to 40-hour range had SUS scores of 85 for HMD Only, 70 for Mouse, and 72.5 for Mobile. An ANOVA comparison did not reveal any significance when comparing game use to the SUS scores reported for each input method.

Pre-Experiment and Post-Experiment Input Preferences Compared to SUS Data

Figure 3.53 (over) shows bars representing the categories of participants who selected each input preference and how they responded to the SUS for each input during the experiment. The two participants who thought they would prefer the HMD Only input had average SUS scores of 82.5 (SD = 14.14) for HMD Only, 83.75 (SD = 12.37) for Mouse, and 80 (SD = 21.21) for Mobile. The two participants who had a preference for the mobile input had average SUS scores of 50 (SD = 3.53) for HMD Only, 85 (SD = 10.6) for Mouse, and 77.5 (SD = 21.21) for Mobile. The largest group (14 participants) were those who had a preference toward the

Mouse with average SUS scores of 69 (SD = 24.07) for HMD Only, 81.79 (SD = 14.89) for Mouse, and 81.25 (SD = 15.53) for Mobile. An ANOVA test comparing the SUS scores for each input based on the participant's pre-experiment preferred input method was not significant. Given that almost the entire sample of data was in one category, this result is not surprising.

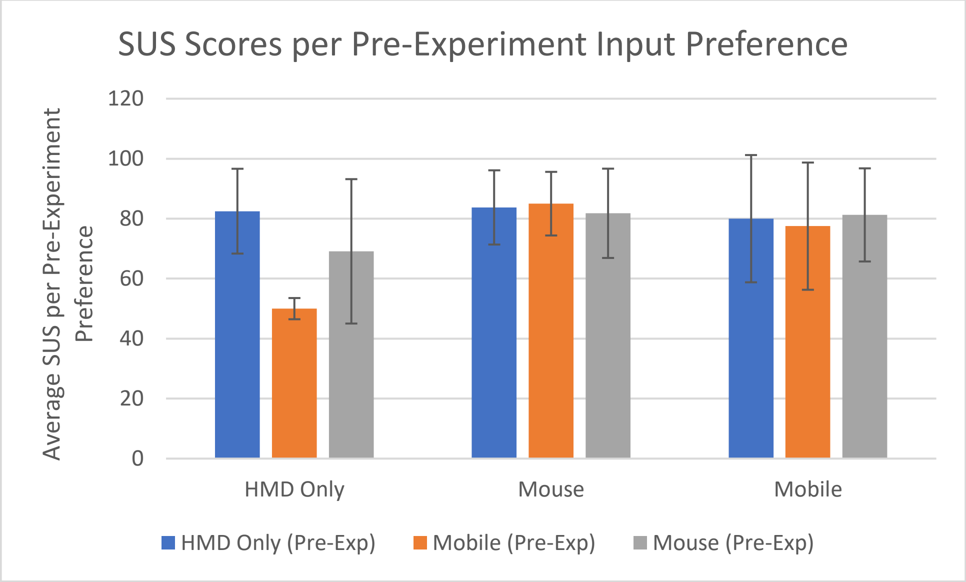


Figure 3.53: SUS Scores per Pre-Experiment Input Preference

Figure 3.54 (over) shows a similar comparison to the previous figure, except this time comparing the SUS scores against the post-experiment preferences. The post-experiment response allowed participants to state a preference as "other". Strangely, the participants who responded with "other" rated the input methods higher in the SUS scoring compared to the average response based on other selections. The six participants who selected HMD Only had average SUS scores of 82.5 (SD = 11.18) for HMD Only, 78.33 (SD = 17.15) for Mouse, and 77.5 (SD = 15.24) for Mobile. The four participants who selected mobile had average SUS scores of 60 (SD = 25.33) for HMD Only, 91.88 (SD = 3.15) for Mouse, and 95 (SD = 2.04) for Mobile. The six participants who selected mouse had average SUS scores of 51.67 (SD = 19.15) for HMD Only, 75.83 (SD = 11.14) for Mouse, and 68.75 (SD = 10.69) for Mobile. The two participants who selected "other" had average SUS scores of 93.75 (SD = 1.77) for HMD Only, 95 (SD = 3.54) for Mouse, and 97.5 (SD = 3.54) for Mobile. From performing an ANOVA comparing SUS scores for each category, the HMD Only (F(3,14) = 4.72, p = 0.02), and Mobile (F(3,14) = 6.1, p = 0.01) scores were significant. Indicating that the preferences toward higher SUS scores seen by participants who also selected the same input as a preference for HMD Only and Mobile are significant.

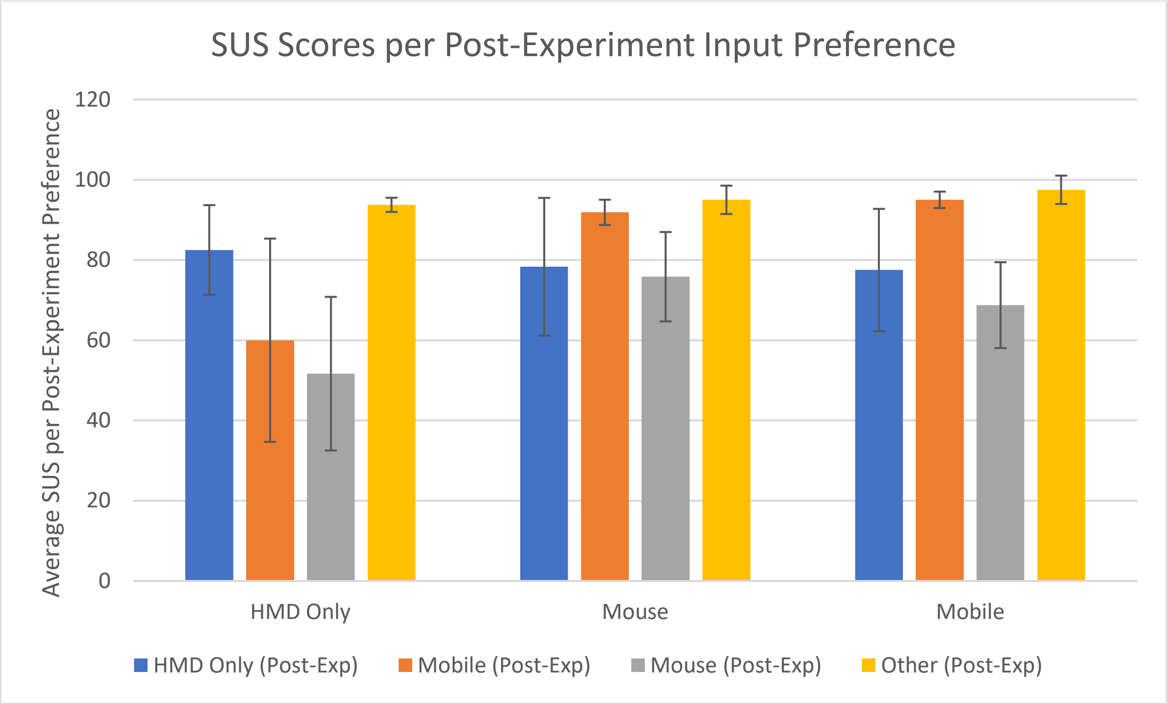


Figure 3.54: SUS Scores per Post-Experiment Input Preference

Figure 3.55 shows the data from both Figure 3.53 and Figure 3.54 combined to view both pre-experiment and post-experiment data side by side.

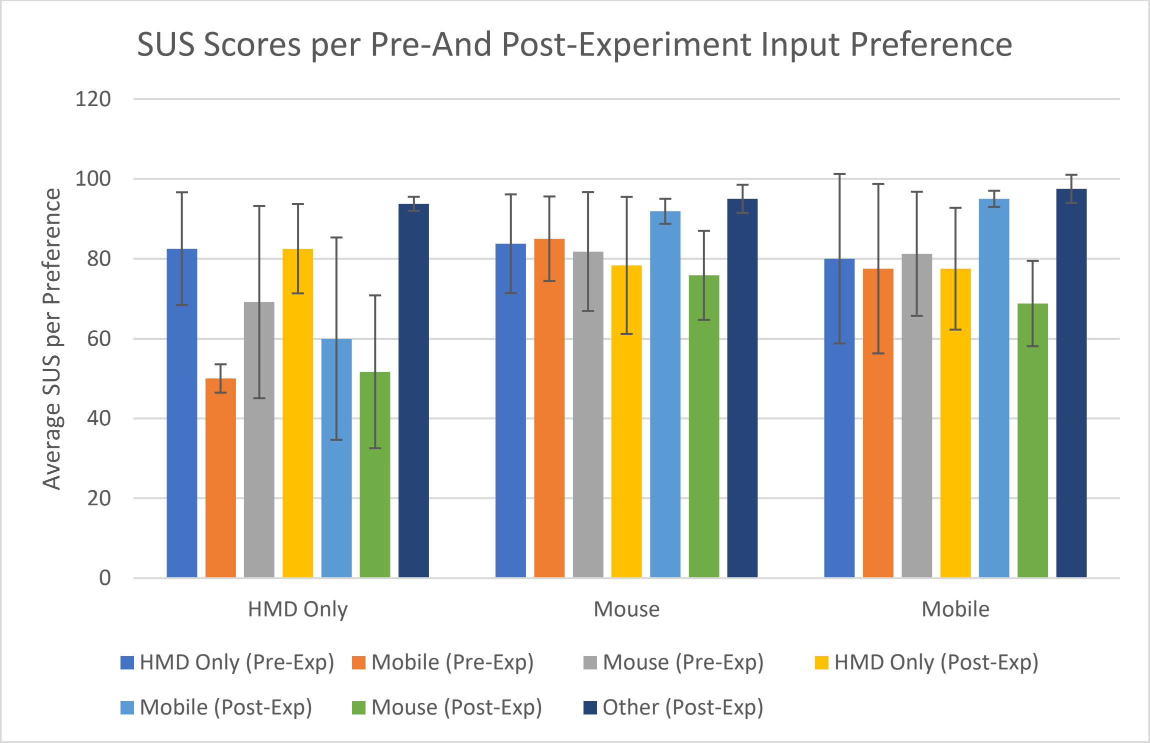


Figure 3.55: SUS Scores per Pre-And Post-Experiment Input Preference

Effectiveness compared against SUS for Each Input Method

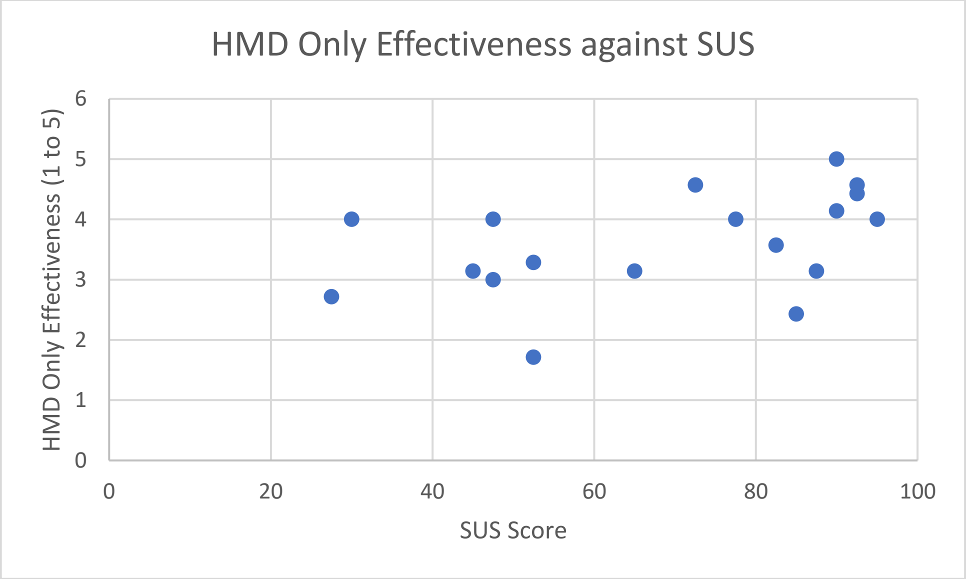


Figure 3.56: HMD Only Effectiveness against SUS

Figure 3.56 shows a plot of the average HMD Only input effectiveness across all tasks reported by participants compared against their calculated HMD Only SUS scores. A one-tail t-Test was performed by scaling the effectiveness scores with a multiplication by 20. The t-Test did not show any significance between the SUS and Effectiveness ratings for HMD Only.

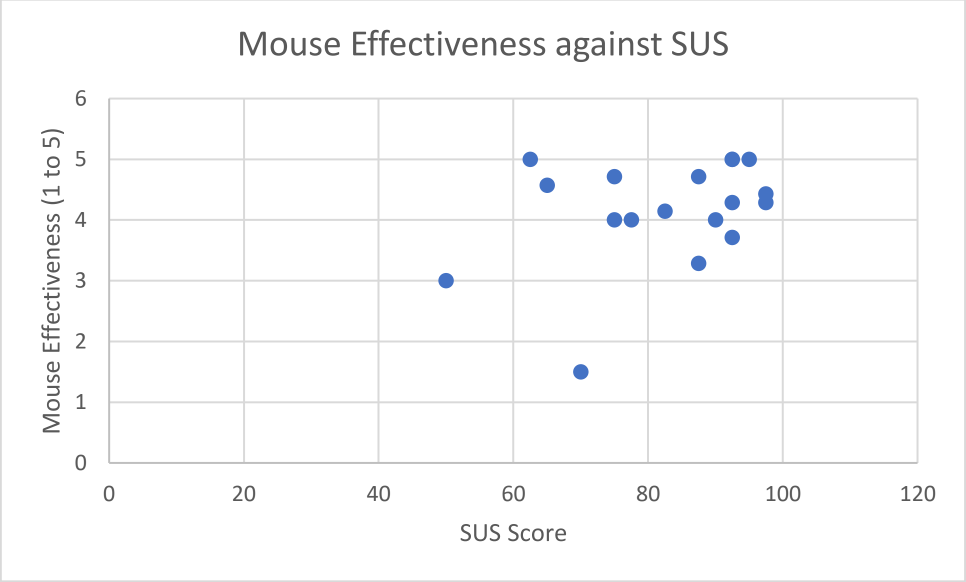


Figure 3.57: Mouse Effectiveness against SUS

Figure 3.57 compares average effectiveness across all tasks for the Mouse input against the Mouse SUS scores. Similarly, Figure 3.58 (over) compares the effectiveness and SUS for the

Mobile input. Performing a t-Test for each did not show any significance when comparing the data.

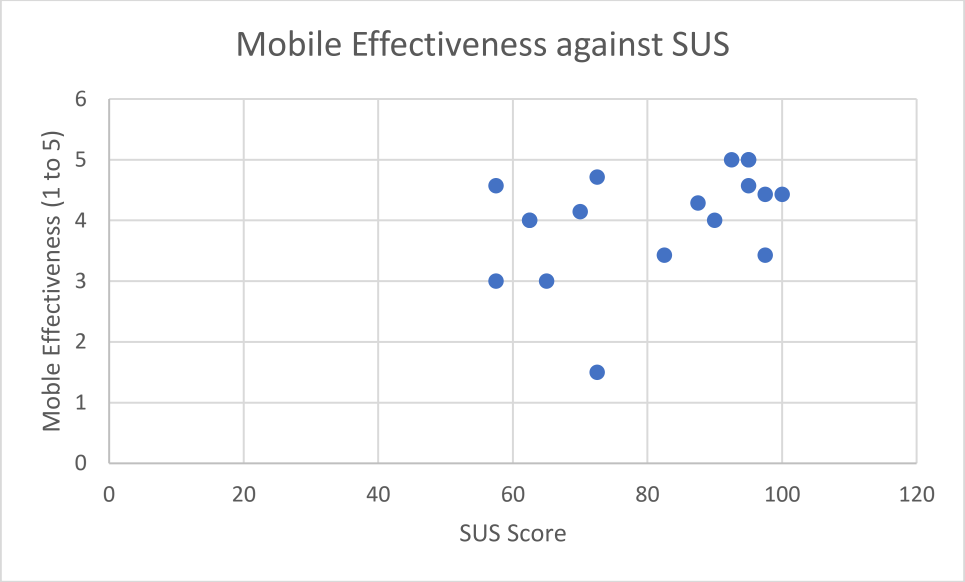


Figure 3.58: Mobile Effectiveness against SUS

Difficulty compared against SUS for Each Input Method

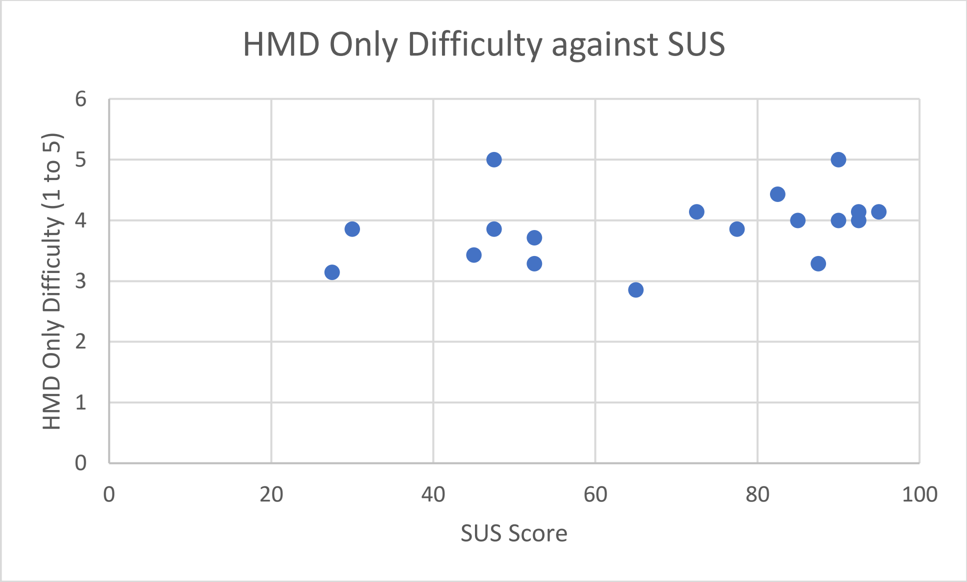


Figure 3.59: HMD Only Difficulty against SUS

Figure 3.59 compares the average difficulty across all tasks against the SUS score for the HMD Only input. A one-tail t-Test demonstrated significance in comparing the data t(17)=-1.86, p = 0.04.

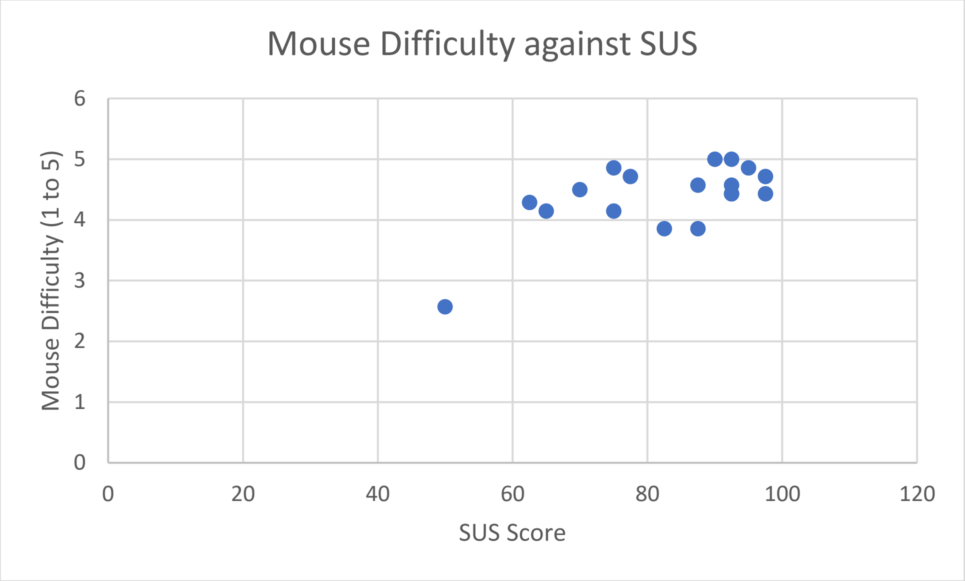


Figure 3.60: Mouse Difficulty against SUS

Figure 3.60 does the same type of comparison for the Mouse input and was found to be significant t(17)=-2.1, p = 0.03.

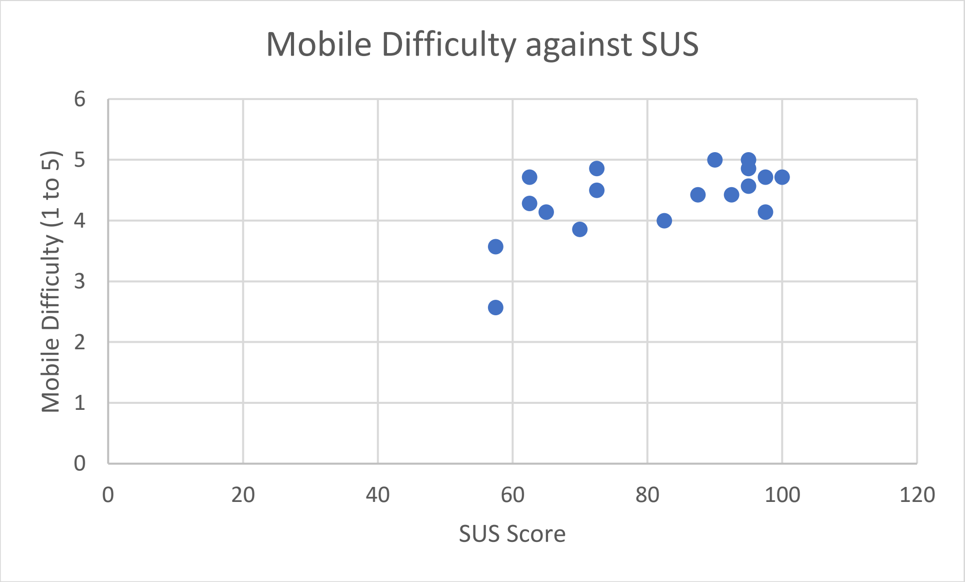


Figure 3.61: Mobile Difficulty against SUS

Figure 3.61 presents the last comparison for input types with the Mobile input with average difficulty across all tasks against the SUS score. Similar to the other results for the difficulty, this was also significant t(17)=-2.14, p = 0.02. The average difficulty compared to the matching SUS score was significant for all three different inputs.

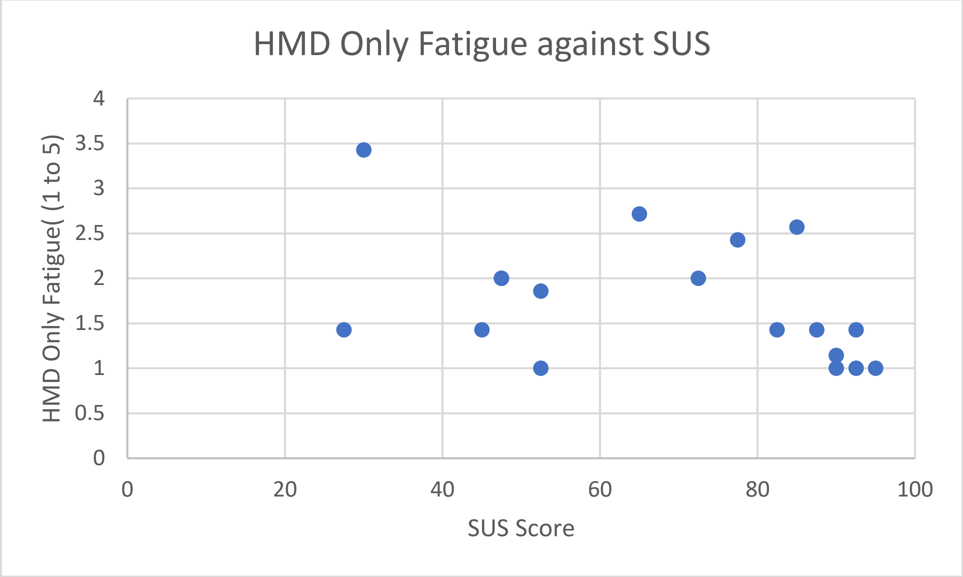


Figure 3.62: HMD Only Fatigue against SUS

Figure 3.62 shows the HMD Only fatigue against each participant's HMD Only SUS score. The inverse of the fatigue (5 - fatigue) was scaled by 20 to give comparable data. The inverse was used for the calculation because fatigue was ranked negatively, with SUS scoring positively. A one-tail t-Test for HMD Only fatigue did not show any significance.

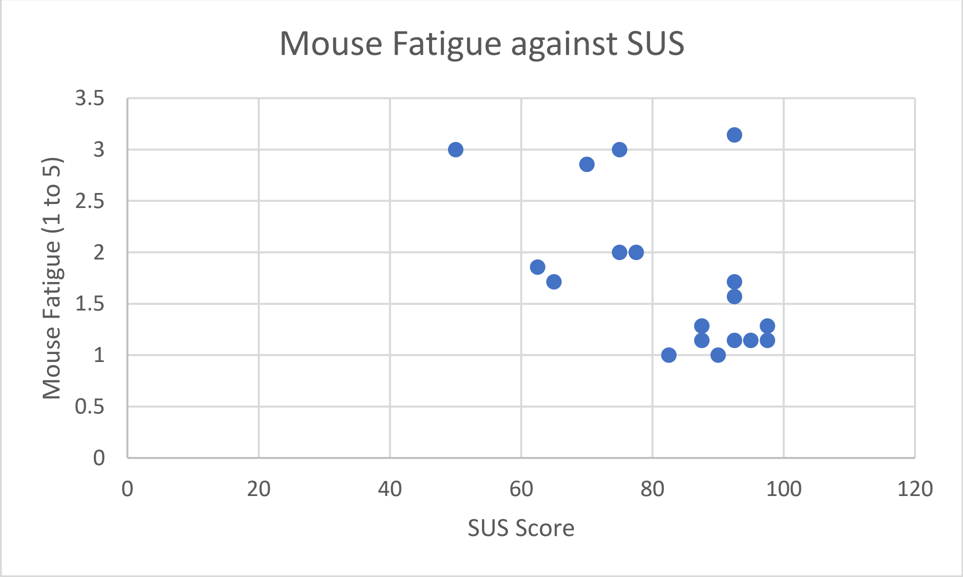


Figure 3.63: Mouse Fatigue against SUS

Figure 3.63 shows the Mouse input fatigue scores against the Mouse input SUS data. Comparing the data was found to be significant t(17)=5.64, p = <0.01.

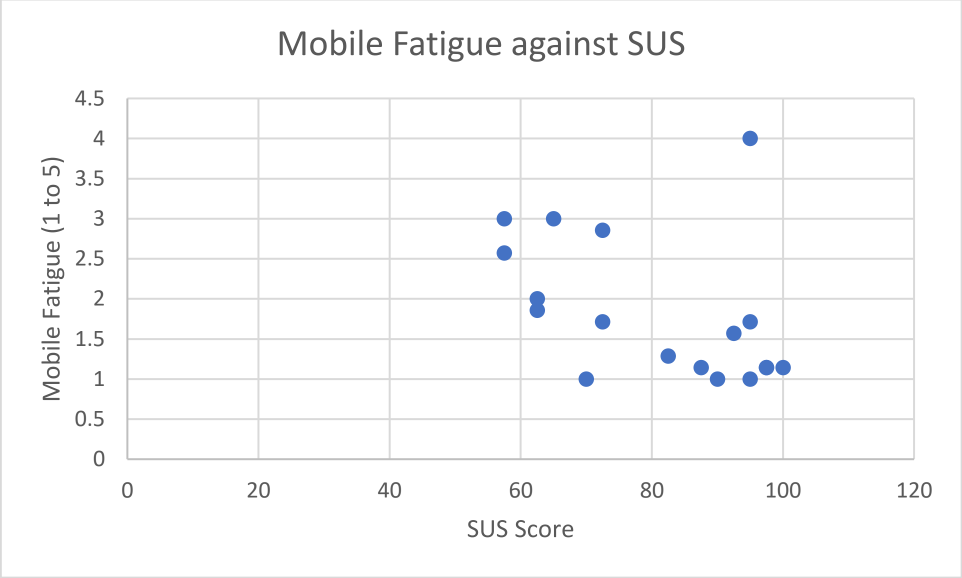


Figure 3.64: Mobile Fatigue against SUS

Figure 3.64 shows the Mobile input fatigue scores against Mobile input SUS data. Similar to the Mouse input, the Mobile input was also significant when compared t(17)=4.15, p = <0.01. These results show that participants who felt lower fatigue were also more likely to give a higher result on the SUS scoring for both the mouse and mobile inputs.

Effectiveness, Difficulty, and Fatigue against Age and Gender

Each of the Effectiveness, Difficulty, and Fatigue were split up into categories based on Age and Gender to evaluate statistical significance. An ANOVA test did not find any significance when comparing any of these factors based on the participant responses.

3.4 Summary of Results

The following section discusses the implication of these results on the goals of the experiment. For comparison against experiments 2 and 3, see Chapter 7. In section 3.1, the experiment overview for this first experiment outlined the following goals.

1) Determine whether any of the three input methods (Computer Mouse, Mobile, or HMD Only) provided a better overall experience in terms of usability.
2) Determine users' perceived usability concerns associated with the head-mounted interaction experience specific to interaction with the Oculus Rift (Dev Kit 1).
3) Determine the interest of participants in the future use of similar technologies.

This section will briefly draw from the many individual results to show the most relevant information for each of these goals.

Following the discussion of the three experiment goals, a consideration of how the findings from this experiment apply to and inform the research questions will also be covered. For this experiment, RQ1 and RQ2 were both investigated. The questions have been repeated below for reference.

- **RQ 1:** How can head rotations, captured through the orientation sensors of a HMD, be used to increase the user interaction capabilities with virtual worlds?
- **RQ 2:** Does using a head-mounted display with head-only interaction for menu navigation provide a similarly useful experience compared to integration with a tool for instant selection?

Goal 1: Input Method Usability Experience

Refer back to Figure 3.31, showing a comparison of input effectiveness between input types and Figure 3.33, showing a comparison of difficulty. They offer two metrics representing part of the overall usability of the techniques used in this experiment. HMD Only interaction was rated slightly lower for both effectiveness and difficulty. For both figures, a higher score was preferred. The mouse input was slightly preferred over mobile and HMD Only. The scores on average for HMD Only, as the lowest-rated input, was still around a score of 4 of a possible 5. Indicating the experience was overall favourable, but preferences fell toward the familiar input device where a tap-to-select type interaction was used.

This familiarity is partially indicated in the way participants responded to the SUS type questions, as seen in Figure 3.38 from the results section. Participants' two most highly rated metrics for the mouse were ease of use and confidence. As a result of the non-randomised presentation of input methods to participants, there was a learning effect for completing the experiment tasks while also learning how to manipulate HMD Only interaction. The mouse and mobile's faster selection with tap-to-select type interaction also simplified the experience. These together can be assumed to have impacted participants' thoughts when responding to questions. One thing to notice from the figure is standard deviation shown as the error bars for HMD Only are larger, indicating a wider mix of reactions between the positive and negative. The responses for the questions were consistent in the difference, indicating participants felt the experience was a little worse than when a tap-to-select interaction was present.

Figure 3.40 in the results section summarised the results for the SUS results. These results showed that mouse was preferable over mobile, with the HMD Only least preferred. From these figures, it is evident the usability based on the given metrics was principally toward the mouse as a form of input. Leaving mobile very close behind and the HMD Only least preferred. These values, though, can be contrasted against Figure 3.37 in the results section, where, pre-experiment, it is evident participants were drawn toward their preconception of preferring the mouse input. The interesting shift in this figure indicates participants shifted away from the mouse input in their preferences, ending with an equal split between HMD Only and the mouse type input in the post-experiment responses. The shift toward HMD Only indicates there is an interested group of people who felt they could embrace HMD Only interaction as a form of input.

Goal 2: Usability Concerns Associated with Head Mounted Interaction

Participants responded with many different variations of similar themes when explaining either their choice of preferred input preference or what was found enjoyable/difficult during each input method. Table 3.5 contains the summarised responses from the participants' questionnaire data.

Focusing on the negatives that have been summarised for the HMD Only responses, there were three primary issues identified. The first issue regarding lower accuracy can be assumed to be about Task 3 and Task 4. In those tasks, the participant was required to select moving objects by hovering for a period, compared to the mouse/mobile where they could tap immediately to select once hovering. The difficulty of selecting moving objects was mitigated by slowing moving cubes based on how long a participant had hovered over them. The participants still considered the difficulty noticeable and is a trade-off when using a setup with no physical button. Considerations of the selection lead to the second point about longer selection time. The amount of time to select elements was static when hovering. The selection time could be adjusted to suit a user's preferences in a published application. The speed of selection was chosen to be deliberately a bit slow to make it easy for people new to the interaction technique. The final issue with difficulty in precision tasks such as Task 5 was expected. It does emphasise the importance of targeting correctly to the experience of using a HMD. Fine-tune mechanics that are punishing for minor movements in the wrong way do not fit well and should be used sparingly, if necessary, based on this feedback.

Table 3.5: Positives and Negatives of Interactions

	HMD Only	Mouse / Mobile
Positives	• Simplicity and ease of use. • More immersion from tasks while only focusing on looking. • Reduced need for other input devices providing a hands-free experience. • Intuitive. • Fun.	• Tactile feeling is familiar. • Faster selection. • Smooth experience. • Accurate.
Negatives	• Lower accuracy from having to spend longer interacting. • Longer selection time. • Difficulty with precision tasks such as the maze in Task 5.	• Distraction from immersion. • Less enjoyable, including dissatisfaction from the number of clicks required. • The cursor was not independent of the HMD. • Maze task was not made easier with mouse/mobile.

One of the usability concerns was the issue of fatigue. Figure 3.35 in the results section showed that the scores were rated very similarly between the different input methods. Indicating overall fatigue was not a large factor during this experiment. The data on fatigue suggests that the fatigue did increase relative to the complexity of the tasks with very low fatigue for simple repetitive selection. The fatigue approached a medium level of fatigue when performing tasks requiring precision (such as Task 5 discussed in 3.2.9) or more complex steps requiring observation of multiple directions frequently. From this, it could be suggested that keeping the user focused with interaction elements near the point of interest and minimising the number of interactions may reduce fatigue and improve usability.

Goal 3: Interest in Future Use

There are already two figures that can be used to derive participants' interest in the future use of the input techniques used in this experiment. Figure 3.37 showed the input preferences. Preference toward a specific input method suggests an interest in future use. From this, it can be drawn there was similar interest in the HMD Only and mouse type interactions. Another element for consideration is the reported frequency of participants expected to use the interaction techniques from the SUS data in Figure 3.38. When contrasted against participant preferences, this would suggest the mouse input would be used more frequently in the future. Despite this outcome, all three input techniques used the HMD for control and positively responded to questions.

It is important to consider what it means for how participants view the HMD interactions compared to the mouse and mobile inputs when reflecting on the overall outcomes of this experiment. In most of the metrics used, the HMD Only input fell behind by a small margin. The margins of difference indicated the area of HMD interaction was worth further investigation. Overall, the responses to this experiment were positive and provided experience in running a VR type experiment.

3.4.1 Research Question Discussion

RQ 1: How can head rotations, captured through the orientation sensors of a HMD, be used to increase the user interaction capabilities with virtual worlds?

In this experiment, the interface elements demonstrated clearly that interaction with HMD Only type use was possible. The interfaces and points of interaction were primitive, with fixed locations within the VR environment. The types of interface elements presented were similar to other existing systems where the interface elements are placed at predesignated locations within a VR environment. The interface experiences of the experiment set a baseline, looking at differences when using the HMD Only type interaction compared to using an additional selection tool in the form of the mouse or mobile device for a tap-to-select action. Supported by the experiment results, the hover-to-select type interaction is simple to understand and provides a usable experience. It has been demonstrated there was potential in investigating the use of the head as a tool for interaction, supporting the validity of the proposed research direction, as discussed in the chapters to follow.

RQ 2: Does using a head-mounted display with head-only interaction for menu navigation provide a similarly useful experience compared to integration with a tool for instant selection?

Most of the leading discussion on this can be reviewed concerning Goal 1. The instant selection tools used in this experiment were the mouse and mobile input devices where a click or tap would perform the instant selection based on where the participant was looking. The discussion from Goal 1 justifies that the technique with hover-to-select in the HMD Only interaction provided a similarly useful experience compared to that of the two instant selection methods. Participants did comment on the selection time as a problem for them, as shown in the discussion around Goal 2. The interaction experience was shown to be useful for the types of interactions demonstrated within the experiment and suggests the merits of further investigation to improve the features available to those who would wish to use or develop with HMD Only type interactions.

3.5 Conclusion

This chapter discussed the first experiment that was conducted as part of the research of the dissertation. The experiment showed there was potential for HMD interactions as a stand-alone tool. With slightly lower scores for HMDs compared to mobile and mouse inputs, the input method did fall behind in one regard. However, some participants indicated they had a preference toward the HMD Only type input. This feedback accomplished the goals of the experiment and provided insight for future experiments. The next chapter will discuss the system designed following this experiment for use in the final two experiments focusing on how the PVMS works and its merits.

4 Periphery Vision Menu System: Technology Overview

This chapter will discuss the core technology developed during this research. The chapter breaks down the core components and discusses some of the prototype code. The discussion is separated into sections dedicated to the prototype user interface, the interaction technique for revealing the interface, and a discussion around using the system for supporting contextual interactions. Before discussing the specifics for each part of the system, the first sections will provide a history that led to the chosen implementation, an overview of the system, and some example use cases where this technology would be useful.

4.1 Initial Design History

The first experiment in the previous chapter demonstrated that HMD Only type input was viable as either an alternative or in addition to other interaction technologies. The results suggested that further investigation was suitable to evaluate how interaction could be improved for HMD Only. The menu used for the final task in the first experiment presented a starting point to seek improvement methods. Investigation into the use of gestures, particularly with head interactions, as detailed in section 2.5.5, revealed examples of using nods or other head movements to represent actions. The types of scenarios these were typically used for included a static interface and focused on actions such as accepting a dialog box (Morency and Darrel, 2006). Many examples of interfaces for the newly released HMDs (Oculus Rift Dev Kit 1 and 2, and HTC Vive) focused on using additional controllers. Hand interactions with either a wand-type device held by the user or detection of a user's hands to put them in the VR world. The HMD as a VR device provided position and rotation information but seemed underutilised by developers for interactions focusing on the head as an interaction tool. Some additional preliminary considerations in favour of maximising the potential for HMD Only type interactions were from considering the downsides of separate controllers. Both situations where controllers were purchased separately, therefore, incurring extra cost, and considering conditions where a person may not have the ability to use one or both hands for their inputs.

Most applications observed while evaluating the area of VR presented either static menus to the user that would appear in a fixed place as part of the world or menus attached to some other element of the world. In cases such as Tilt Brush (Google, 2016), the attachment was to the controller's position in the world. In other cases, interaction points could be attached to either other places on a user's virtual body or other objects in the world. These types of menus did not make full use of the HMD as an interaction device.

From designing the final task of the first experiment, a concept was born to create a type of menu that would be just out of a user's view but had the menu move with the user. Humans have a natural field of view with two forward-facing eyes that provide a mostly shared perspective. The area not shared in the overlap is referred to as periphery vision. The initial concept was to use this periphery vision by rendering the menu partially into the edges of the HMD display. A user's focus is expected to primarily face forward to the area where a user's vision overlaps with both eyes. Putting a menu into the periphery would uniquely hide a menu. Initial attempts to implement this type of menu ran into a few issues that meant it would be necessary to re-evaluate how the menu would appear. The first issue was the output resolution of the Oculus Rift Dev Kit 2 used for the second and third experiments. In combination with the distortion applied to visuals (due to the Oculus SDK), the resolution made menu text in the periphery harder to read. It was necessary to make the text much larger and dominate users' peripheral vision. The second issue from the larger display requirements was that it became evident that it would be preferable only to have the menu context come into view as users began to turn their heads toward that context. Revealing the menu as a shift into the screen based on turning the head was complex due to necessitating clear visual interface elements based on the first issue. Therefore, as a result of these two issues, the focus of prototyping moved to use turning into the periphery as a gesture to create a menu in VR world space.

The prototype menu presented a suitable initial version that allowed for creating menus at any arbitrary point in world space. The system was designed to make the menus contextually based on the direction of a gesture, including left/right and up/down. The second experiment was designed to provide testing and evaluation of the menu by letting participants experience the hidden menu system provided by the PVMS. Based on the feedback collected during the second experiment, the design was modified to improve issues experienced by participants.

A simple version of the original concept was also evaluated with the menu names displayed in the periphery as part of the final task. The third experiment evaluated the updated prototype and validated the changes implemented to improve the user experience. This chapter discusses an overview of the interface created as part of the second and third experiments. Most of the functionality and interaction was the same between the two versions, with minor but important differences that improved the experience in the third experiment. The second and third experiments are presented to validate the PVMS following this chapter.

4.2 Concept Overview

In this section, a general overview of the technology is described. The overview will introduce each aspect of the interaction scheme and then is explained in specific, relevant detail in later chapters. After defining the concept, interaction presentation and functionality is established. The following examples illustrate how this system would be broadly used and where caution should be used:

- Museums and Sightseeing
- Game Interactions
- Military and Emergency Response
- Within Any Application
- Where the System Should Not Be Used

Each example will discuss why the system is beneficial to the scenario and a short example of how the integration might function. The techniques are designed to be very transferable between different HMD type applications, so a more general use case has been included to emphasise the various benefits. The PVMS is designed with scope for use in any application that supports a HMD.

4.2.1 What are Periphery Vision Menus?

The Periphery Vision Menu System (PVMS) provides a technique for revealing a hidden menu using head interaction. The periphery refers to the interaction and visualisation used for revealing the menus. The menus, once revealed, provide a means for contextual interactions dependent on the application and situation where they are being utilised. The primary advantage of this system is that it can be entirely controlled with head interaction. The focus on the head as an interaction tool allows interactions with the use of the head where it would otherwise be impractical to hold additional input controllers. The system can also be used alongside other interaction techniques as an additive feature of user control. The system was evaluated through two experiments detailed in Chapter 5 and Chapter 6.

The menu system was born from investigating interactions for use in the first experiment. The focus of the first experiment was to evaluate interactions with HMDs. Task 7 was used as inspiration for developing the PVMS from the success of the first experiment. At the time of development and research, no system functioned in the way this was designed. Most applications focused on using additional external controllers as the form of input, with either static interface elements or interfaces attached to a controller. It was seen as an important distinction for the PVMS that key interaction mechanics did not interfere with the user's primary task by only showing and interacting with the menus when necessary.

The PVMS can be broken into three stages. The first stage is when a user intends to reveal the menu. A head gesture would be completed to initiate the reveal interaction, based on application-defined configurations. In the case where this head rotation gesture was expressive enough to trigger an interaction, this would initiate the second stage. In the second stage, a small interaction element, titled a two-step widget, is shown to the user. Hovering the user’s focus over this element initiates the third stage of revealing the menu for user interaction. The following list provides some additional information relating to each of these features.

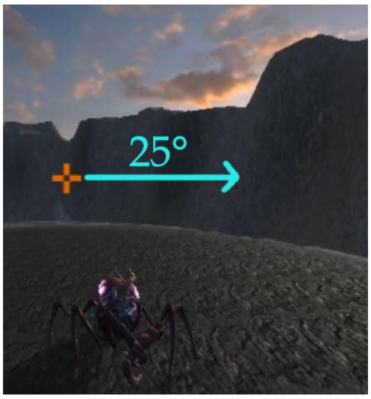


Figure 4.1: Direction Head Rotation

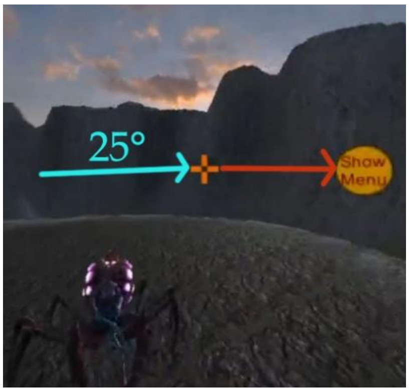


Figure 4.2: Reveal Two-Step Widget

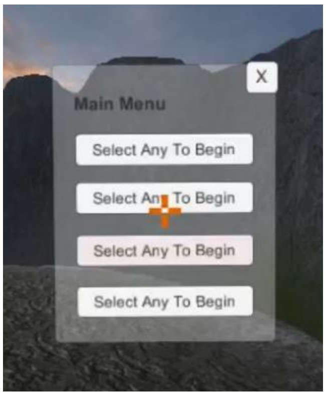


Figure 4.3: Contextual Menu

- **A directional head rotation to reveal the menu** (**Figure 4.1):** Head interactions were tracked using Euler angles. An algorithm evaluated Euler angles over time against configurable thresholds to provide a choice of interaction sensitivity. The specifics of how the algorithm worked with prototype code will be discussed in section 4.4.1 on Revealing the Menu.

- **An interface to reveal the Two-Step Widget (Figure 4.2):** The two-step widget was added in the third experiment to improve upon the testing from the second experiment. When participants of the second experiment revealed the core menu with a gesture and decided they did not want the menu, it became unnecessarily intrusive on the participant's vision. Creating a visually smaller menu revealing button as a two-step process mitigated the impact of gesture mistakes. The two-step widget will be discussed further as part of 4.3 concerning the interface, and the technical information can be found in 4.4.2 on interacting with the menu.
- **An interface showing contextual options or information (Figure 4.3):** This section considers the menu revealed by user interactions. A simple menu interface was used for prototyping to provide consistency and rapid learning for participants who had limited VR use. The appearance of the menu could be changed to the form required by an application. The menu consisted of a title and four option buttons configured through contextual states by the application. As part of development for the third experiment, a close button was also added to the dialog for the same reason as the two-step widget was added, providing a way to opt-out of making a choice and closing the menu. In the second experiment, the only way to complete this action was to not interact with the dialog for a defined period of time. The interface visuals are discussed in 4.3 on the user interface, and technical information will be discussed in 4.4.2 on interacting with the menu.

With the core concepts identified, the specific features related to each part of the interaction system will be discussed in sections 4.3 and 4.4. The following subsections present various conceptual use cases to illustrate where the PVMS could be used. These proposed scenarios proved context to show the inherent usefulness of this style of interaction paradigm.

4.2.2 Example Use Scenario: Museums and Sightseeing

Some museums and sightseeing locations provide their visitors with audio guides (The British Museum, 2018). Often these provide interesting extra details about exhibits beyond what could be written or presented by a human guide. Some perceived advantages of audio guides are that the devices stand-alone, are lightweight, and have minimal cost. For attractions wishing to provide a visually interactive experience, this may be accomplished with the aid of HMDs in the future. As the cost and technology improve, there will likely be a time when these same advantages would be feasible to apply to HMDs.

In this scenario, the typical use would be a version of AR for safety and presence within the exhibit. The user could approach exhibits wearing the HMD and observe additional information about the exhibits. The information may include historical videos, pictures, stories, serious games or other forms of engagement. Additional controllers may not be practical to offer to visitors wishing to use this type of interaction. They add additional objects, risk of damage (to the devices and the exhibits), maintenance, or loss. Therefore, it would be practical to develop this type of scenario to use primarily head interactions. The addition of the PVMS in this situation would allow intuitively hiding many options. Typically changing settings in one of these situations would be done just once or infrequently. Some of the options a hidden menu could provide may include:

- Language Options: Providing a way for users to change the language would increase the accessibility for a wider group of people.
- Accessibility Options: For enabling features such as reading aloud content being viewed for someone with poor vision or magnification to perform a similar function.
- Feature Options: Some visitors may wish to tailor their experience constantly to show specific types of content or never to show some kinds of content. The options may include changing between a representation intended for a child and one designed for an adult, tailoring the experience to fit the desired experience. The experience for a child could incorporate faster access to games at many exhibits to encourage learning.
- Execute an activity: Allows a user to start a specific activity, make selections in an interactive quiz, implement navigation choices (play, pause, skip, back) for displayed content or other types of content focused interactions.

All these options could be hidden using the PVMS to allow the user to change their preferences at any time.

4.2.3 Example Use Scenario: Game Interactions

Broadly games within the VR space could benefit from the inclusion of PVMS. Typically, fast-paced games require many player actions per minute or fine accuracy where the user cannot look away to perform other interactions. Other games slow gameplay down to match the user's desired speed, with a mix of other types of games in between. Games with a slower pace or moments for a pause between hectic gameplay would benefit most from the inclusion of the PVMS.

Across all types of games, there are common features that normally appear as part of the game. These features could include the main menu, game settings, initiating communication with other players, joining multiplayer games, viewing gameplay statistics, or submitting bug reports. All these features could be controlled with a PVMS. In the case of the faster-paced games, this would typically be during downtime. Perhaps between games while in a lobby or quickly changing something while hiding behind an object in the virtual world.

Games with a slower pace or specific downtime points where actions would be completed could make additional contextual use of the PVMS. For example, in a typical online multiplayer competitive game, there is usually a period at the start of the game where the player can define their character or inventory for the coming match. A VR equivalent version of this game could utilise the PVMS for this type of player preparation feature.

Strategy games or games where there is a choice for spawning objects and making decisions about those objects fit with the PVMS too. This interaction function has been demonstrated during the second and third experiments of this research, using object manipulation tasks and the implemented tower defence game. These types of games can be designed to accommodate HMD only interactions for decision making and strategic choices. The second and third experiments, discussed in Chapter 5 and Chapter 6, will describe in more detail how the menu system has been used for game interactions.

4.2.4 Example Use Scenario: Military and Emergency Response

Military scenarios would be an especially sensitive area, as equipment needs to be responsive and not impede the ability to perform in potentially deadly scenarios. HMDs have been investigated for their use as an AR tool for surveying environments, identifying friendlies, communication, distribution of visual plans with route information or many other forms of tactical improvement. These types of interactions for activating or configuring controls could be provided through PVMSs. By using head-mounted interactions for these activities, it becomes possible to be carrying or interacting with other objects at the same time independently.

Marking important objectives, communicating, and providing information directly to emergency response workers may also fulfil a similar role. Some examples may include coordination of fire fighters for forest fires, prioritising search and rescue in natural disasters between teams (floods, tornados, etc.), or accessing on-screen emergency medical information to identify and respond to unfamiliar symptoms. The applications would be very domain-specific in how the menu system could be applied. Through this speculative exercise, it could be suggested that there is a breadth of potential applications for hands-free head-mounted interaction in critical situations.

4.2.5 Example Use Scenario: With Any Application

Many of the uses for the PVMS fall into similar categories of interaction. Rather than providing long-form descriptions for every type of example, this section describes scenarios where the system could be used with any application. The described scenarios have been separated into two parts. The first list will provide some examples of generic uses where the PVMS could be applied. The second list will cover examples used in the post-experiment questionnaires of the second and third experiments. These examples were cases where participants were asked to rank situations they would most want to use the PVMS.

Uses with Any Application

- **Main Menu:** As a recurring theme for the second and third experiments, the PVMS was used as a mock main menu system. In these cases, all menu options would take the participant to the same place because the experiments were linear in the types of tasks to be completed. For most user scenarios, an application's main menu does not need to be constantly accessed or visible. The main menu not requiring constant visibility makes it a good candidate for the PVMS. When contextually appropriate within an application, the main menu could be made available to the user. Main menus for applications are often used for navigating between different activities. Examples of menu actions could include actions such as: swapping between game modes (multiplayer, single-player, and many variants), changing settings for the application, creating a new project or performing other actions to change the high-level state of an application.

- **Advanced Options:** Some applications have features or options that the standard user would rarely use. These features may include debugging tools, streamer chat integration tools, real-time data visualisation tools, or other settings to enable additional hidden features within an application. These types of features, when provided via the PVMS, would allow a simple opt-in type approach when the user desires additional functionality.
- **Object Creation:** Much of the testing during experiments with this menu system has included object creation. In the experiments, this was used to create towers in the tower defence game and spawning shapes during object matching tasks. This type of spawning of objects to be used within a virtual environment could be applied to many types of tasks. Particularly for populating the virtual world and providing a way to customise existing objects or prototype new environments. This creation and customisation could apply to both a developer utility for assisting in designing elements or for end-users to make choices about what appears within their own environments.
- **Object Manipulation:** Another common feature of the menu used in the experiments was for object manipulation. Upon selecting an existing object, the properties of this object could be modified through a PVMS interface. The examples of interactions from the experiment provide options for repairing, moving, and changing objects' colour/size/shape. The properties exposed to the menu in this way could be as detailed or as simplistic as appropriate for the specific application. In addition to modifying the characteristics of any particular object, such a menu could also provide a way to attach other forms of interaction to elements.

Questionnaire Scenario Examples

- Virtual Cinema: For viewing movies, TV shows or any other type of visual video medium. The interaction could be used as a virtual remote control providing common playback functionality or navigation between video sources.

- Constructing Models: Following the similar approach used in the experiments of spawning in objects to place them into the world. Constructing models could fall into either creation of single models or the broader world population. Single creation could exist by combining multiple components, such as putting together different parts, for example, making a car from various defined components. In the second experiment, a rudimentary version was included with stacking blocks together to form a larger block. In contrast, the world population is more along the lines of placing multiple different objects together. Building design falls between the two of these scenarios where the architecture is being designed as a single entity made of multiple smaller entities within a virtual world.
- Operating Systems: Within a virtual operating system, the start menu or other forms of system options could be included using this menu system.
- Messaging: Instant Messaging can include voice, video, or text type data. With the use of the PVMS, the initiation of communication could be started. Once a communication has been started, the menu could provide options related to a user's preferences for presenting the user or how others are presented.
- Browsing the Internet: The PVMS could be used for functions commonly on the toolbar region of a web browser within a VR interface. Functions such as home, back, refresh, bookmarks, or sharing URLs.

4.2.6 Example Use Scenarios: Where the System Should Not Be Used

The examples described above speculated various scenarios where the interaction technique could be potentially beneficial if incorporated. There are some scenarios where the technique, due to its nature, should be avoided. The way the menu is triggered by turning partially away from the focus point could be hazardous if the activity requires a wide field of view or high attentiveness. Similarly, any situation where the menu would directly impede vision of likely dangers could also cause problems. The following two examples show just two of the possible situations where there is a definite risk involved. In any situation where risk is involved, great care should be taken to mitigate where possible or choose alternative technical solutions.

- During operation of moving vehicles: Imagine the future where it is plausible that head-mounted interaction becomes a part of daily life, for example, while controlling vehicles. Any time a user is performing manual control, it would be reasonable to conclude the act of looking away to open and manipulate a menu would incur a risk of an accident. Ways to mitigate this situation could include either the assumption of self-driving cars or only allowing menu interactions when the vehicle is stationary.

- During operation of hazardous equipment: Heavy machinery, delicate machinery, or dealing with hazardous elements incur an expectation of the user to focus on the task. Frequent interactions with a periphery menu would be impractical for a safety assessment in these conditions. These types of interactions could be through a virtual interface where the user is controlling a device remotely. As with the vehicle example, this type of interaction technique would only be practical when operations are paused. In this way, the menu interaction technique could still be used to initiate the configuration and connection to the target device.

4.3 User Interface

It is important to consider the aspects that will define this user experience, having presented a range of motivating scenarios for using the head rotation style of interaction. The following section discusses the various design and development decisions that have been implemented to realise the PVMS.

The user interface provides the visual representation of information to the user and the canvas for interaction. In VR and AR, these interfaces can be rendered as part of the world space. As used for most applications on a computer screen, traditional interfaces can only render directly to the screen, and interface design needs to accommodate this “always-on” display mode. In the case of using a HMD the screen is directly in front of a user’s eyes, so rendering a constant interface (such as menus or common interaction widgets) directly on the screen could make it hard to read or obscure the core target functionality of the application. This consideration makes it necessary to handle interfaces differently from the traditional desktop paradigm and detach the user from the interfaces to allow them to view interfaces from varied angles while moving within a virtual environment while not obscuring the focus of the application.

The Unity User Interface tools were used throughout implementing the various prototypes used for testing to simplify interface construction. The interface presentation techniques designed for the experiments would be suitable for use with the menu system. The main requirement is that whatever canvas is used, it would be rendered in the world space. The loose restriction of only requiring world space rendered elements leaves it mostly open-ended for an application's developer to decide what interactions are necessary for their menus. This section will focus purely on the visual aspects of the interfaces. The technical information and interaction logic will be discussed in section 4.4.

4.3.1 The Importance of Visual Simplicity

Monitors and TV screens continue to push to larger sizes and higher resolutions. With a HMD the difficulty is providing small screens with high resolutions without high financial cost. Improvements in the area of smartphone resolutions have helped toward this, with consumer-level HMDs being developed with 2K resolution per eye. Examples of HMD using these resolutions include the Oculus Quest 2[33] or HP Reverb G2[34]. Unlike sitting back to view all the small details of interfaces from afar on a computer monitor, these interfaces directly take up visible space in the virtual worlds. Small details make it vastly more important for visualisation to be quickly recognisable with a low impact on other features, thereby providing visual simplicity.

Mobile devices applications typically need to deal with a similar problem to fit necessary information onto a small screen. The standard model of interaction on mobile devices is the use of fingers for touch input. This means interface elements need to be sized large enough to distinguish them apart with a touch operation. As a result, mobile UI designers need to create elements distinct enough in size and layout for interaction. A similar methodology can be applied when considering VR. Whether using head only interactions or a controller to point within the virtual world, the elements need to be distinct enough to make interactions obvious. This constraint leads to a desire for visual simplicity to ensure interface elements have designs consistent with those recommended in Schneiderman's 8 Golden Rules of interface design (Schneiderman, 1992). These rules are supported by ensuring elements have a consistent purpose, clear layout, and the ability to quickly determine the desired action within an interface to reduce short-term memory load.

[33] Oculus Quest 2, https://www.oculus.com/quest-2/
[34] HP Reverb G2, https://www8.hp.com/us/en/vr/reverb-g2-vr-headset.html

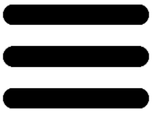


Figure 4.4: Hamburger Widget

Simplicity can be used beyond how the elements are organised by using recognisable visual cues or gestures. One visual example is the hamburger widget seen in Figure 4.4 used to draw parallels for this work to those used in mobile interface development. This icon represents a menu that can be expanded commonly to provide settings or other functionality in mobile applications. The two-step widget that will be described in the following sections follows a similar principle. The gesture used to reveal this widget also draws a parallel from mobile development, where a swipe across the screen can be captured to perform an action. In the case of this work, it has been used as a gesture to simplify the visual space by making it only visible when desired.

4.3.2 Interface Layout

In Figure 4.5, the final layout for the PVMS is shown; this was the style used in the third experiment. The interface was designed with visual simplicity as a focus. There were four different menu selection targets. A menu title provided direct feedback to the user about the type of menu they were interacting with at the time. The other interaction component is the close button, which was added in the third experiment to provide an additional way to disengage from the interface.

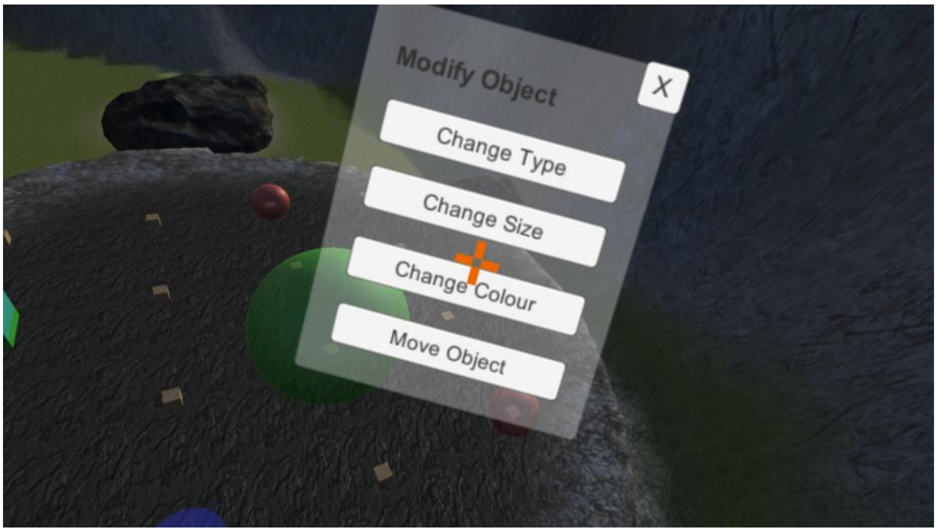
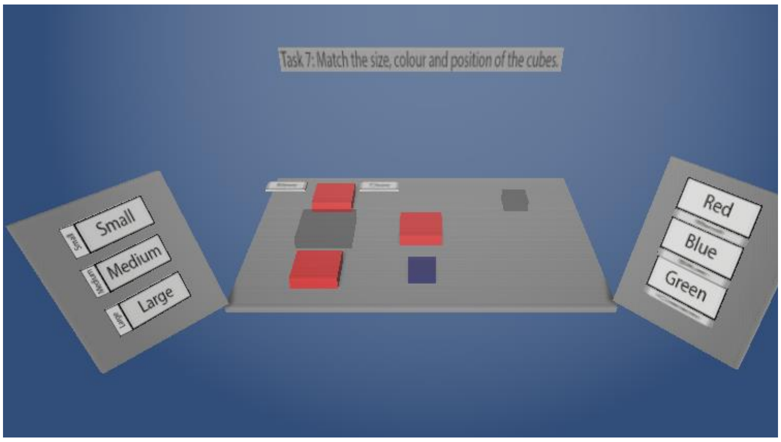


Figure 4.5: Experiment 3 Example Interface

Informing the final design of the PVMS interface were the design and results from the first and second experiments. The first experiment used a static interface; this forced taking up screen space and did not support the user being able to access the menus from any direction. As seen in Figure 4.6, the layout of the buttons was similar. During the first experiment, the buttons were cubes with textures rendered on them. After this initial design and implementation, a refined, extensible solution was developed using the Unity Interface tools.


Task 7: Match the size, colour and position of the cubes.

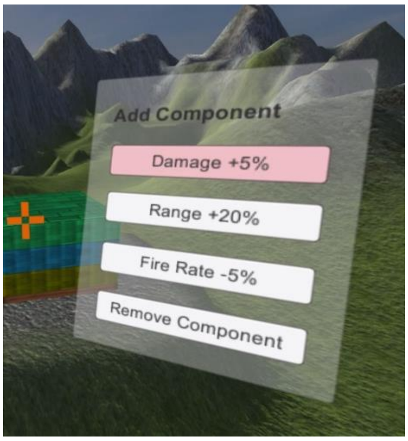


Figure 4.6: Experiment 1 Example

Figure 4.7: Experiment 2 Example

The second experiment iterated on the static interface and shifted the design toward a prototype interface that always positions relative to the user. The positioning of the menu will be discussed further in 4.3.3. Figure 4.7 shows an example of the menu from the second experiment. The interface looks cleaner than the first experiment's version and is very deliberate in the visual spacing. Each interactable element has a clear amount of space in between, providing room to hover around the menu if taking longer to decide on an action and reducing any mistaken selections by the user. Menus would automatically disappear after 4 seconds when the user either did not interact with the menu's canvas or an element on the menus. Hovering over any part of the canvas would reset the automatic hiding countdown.

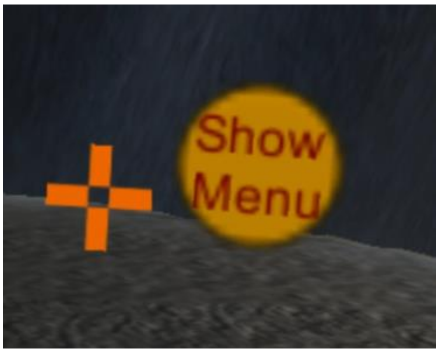


Figure 4.8: Two-step Widget

The “clunky” feel of menus not disappearing fast enough was remedied through the iterative design of the system for the third experiment. A combination of two simple visual elements was used to provide an improved experience. When revealing the menu in the second experiment, the menu would appear immediately (e.g. as seen in Figure 4.7). Sometimes this was not desired for an action to be performed. Participants would check the menus during downtime to see if they could purchase towers despite insufficient currency (Section 5.1 explains what towers are and how they were used in the tower defence game). Sometimes participants also accidentally triggered the menus when it was not intended with a head rotation while surveying the game space. An accidental trigger left a menu in their view or the world that they had to wait to disappear. A two-step widget operation was implemented to overcome this, as shown in Figure 4.8. For the third experiment, this was shown instead of the menu immediately. Hovering over the icon would immediately replace the icon with the appropriate menu. This interaction did not impact time for interaction but meant ignoring the smaller icon for a short time could be done instead of waiting for a full-size menu dialog to disappear. A close button was added to the menu interface to provide an additional method for forcing an immediate closure of the menu.

4.3.3 Positioning of the Interface Relative to the User

Static menus traditionally suffer from two problems: obstruction of content and fixed positioning. The first problem is that they continuously take up visual real estate, reducing room for broader visualisation of the main focus of the application. The second problem is that static menus are not designed to move with the user. There are certain situations where a minimal static menu makes sense in a VR environment. For example, when the user is kept in a single place to provide menu inputs leading to the entry of a game. Or for the novelty of needing to move up to a menu to interact as if it is a virtual interaction screen within the virtual environment. Often, a user wishes to interact with menus on the go and returning to a static interface may not be practical.

To deal with this issue, the PVMS displays menus in a way that is relative to the user; this is realised in two ways: world location and rotation. The first concerns where menus are displayed, and the second is that menus should always face the camera. Figure 4.9 shows a visual indicator of where the menus would appear in the form of blue cubes. The cubes seen in this figure represent hidden game objects. These game objects are used to reference an offset from the camera at the time when a menu is triggered. The gesture would then result in a two-step widget appearing, as seen in Figure 4.10. A gesture to the left would create the widget at the left node, and similarly, a gesture to the right would result in the widget rendering on the right node. This widget is then displayed, always facing the camera. This presentation results in widgets and menus that are always fully visible and easy to read without encountering problems where a menu could be facing at an angle that is not viewable from the user.

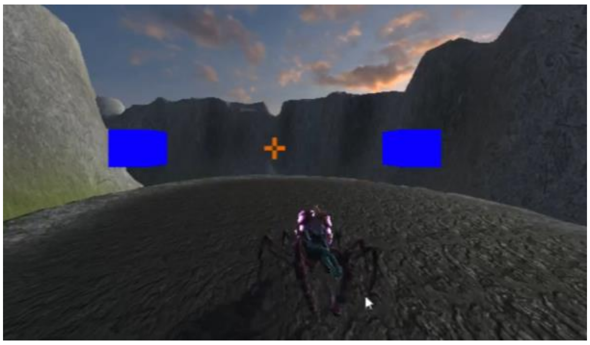


Figure 4.9: Visual Indicators of Menu Spawn

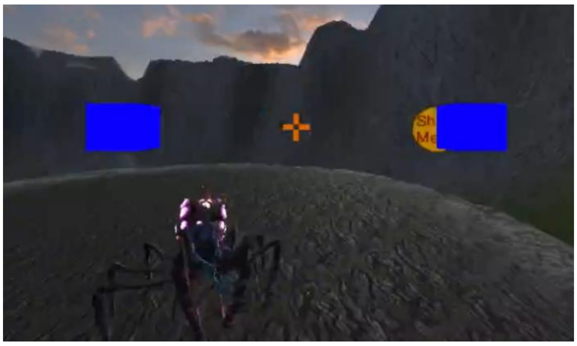


Figure 4.10: Triggered Right Menu with Visual Spawn Locations

This sequence of actions means that once a user creates their Periphery Vision Menu, it transitions from a two-step widget created by the periphery action to a temporary menu. For the experiments discussed in Chapter 5 and Chapter 6, the menus are presented in a fixed place relative to the user until interacted with or automatically closed after a non-interaction period.

4.4 Menu Interaction

This section separates two of the areas for technical discussion. Section 4.4.1 will explore the code and process behind revealing the menus. Much of this includes exposing the prototype code and explaining how the prototype system worked. Following the discussion on revealing the menu is a discussion on how the menus were interacted with once revealed, presented in section 4.4.2.

4.4.1 Revealing the Menu

This section will discuss the code implementation used for the prototype system used in experiments two and three, including the class structure of the Periphery Behaviour class and the code used for the core methods. The complete code can be found via GitHub as described in Appendix D. The process is generalised in this section by presenting it as a figure with an example implementation shown with code exerts. Before discussing the code, the following figures introduce the process used for revealing the Periphery Vision Menu. Figure 4.11 (over) shows the four states of interaction required to present the menu. Figure 4.11 (A) is the initial state when the user is currently using a HMD and just looking around the virtual environment. Figure 4.11 (B) demonstrates that the user has rotated their head 25 degrees while not rotating too far on any other axis of rotation, causing a trigger event. Figure 4.11 (C) shows the two-step widget as it is being selected. Once the two-step widget has been interacted with, the menu is revealed, as seen in Figure 4.11 (D).

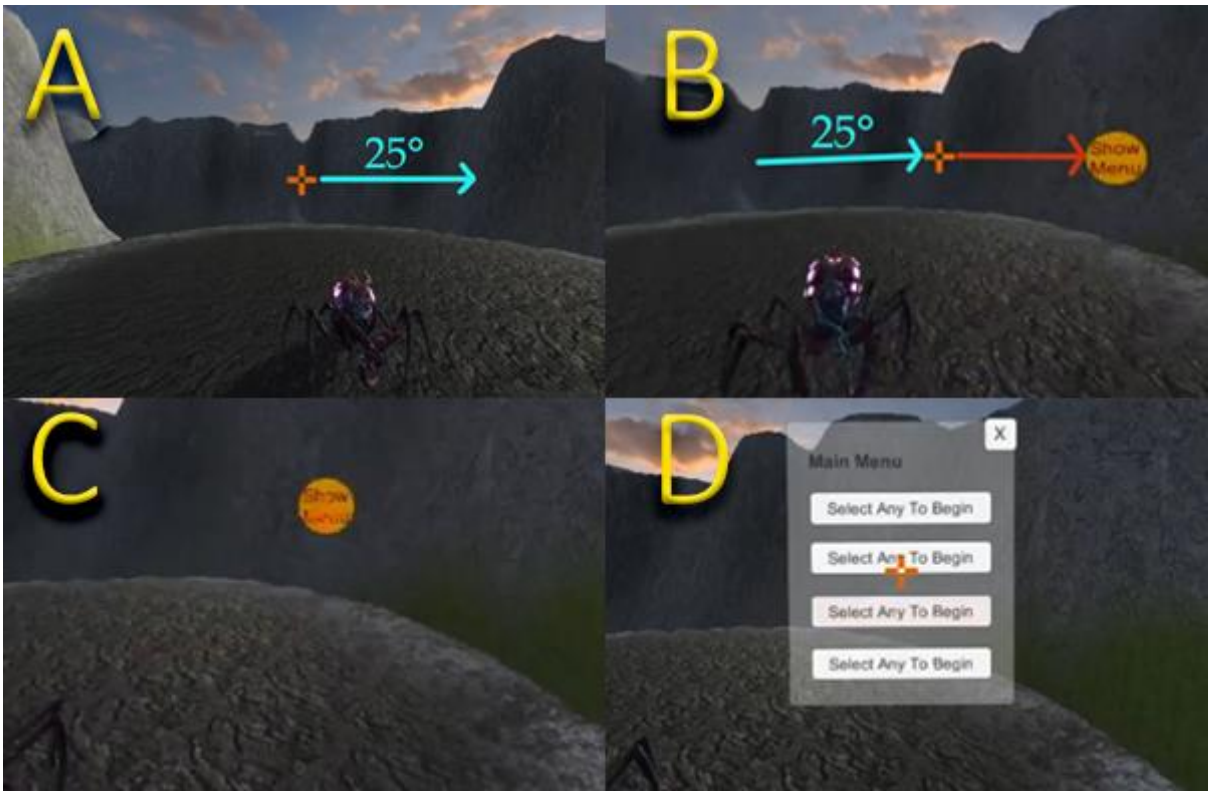


Figure 4.11: Example of rotation. A: No state, B: Turn with Two-Step Triggered, C: Selecting Two-Step, D: Menu Visible

(over) shows the logic used as part of the **`updateRotationTrigger`** method. The approach used for determining if a menu trigger occurred involved keeping a history of rotation data over time. Delays between triggering new menus were introduced to handle cases where a user needed to rotate their head a significant amount rapidly. Without some form of delay check, the menu would repeatedly trigger, which was a use case deemed unlikely the intention of the user. These delays are set at the bottom of the diagram. If there was a menu associated with the direction of rotation, then a longer period for pause is applied. And if no menu existed, then there was no context available for that interaction at the time, so a short pause is applied to save operation time. There was no use of up or down triggers for the prototype, so these would always incur a short delay.

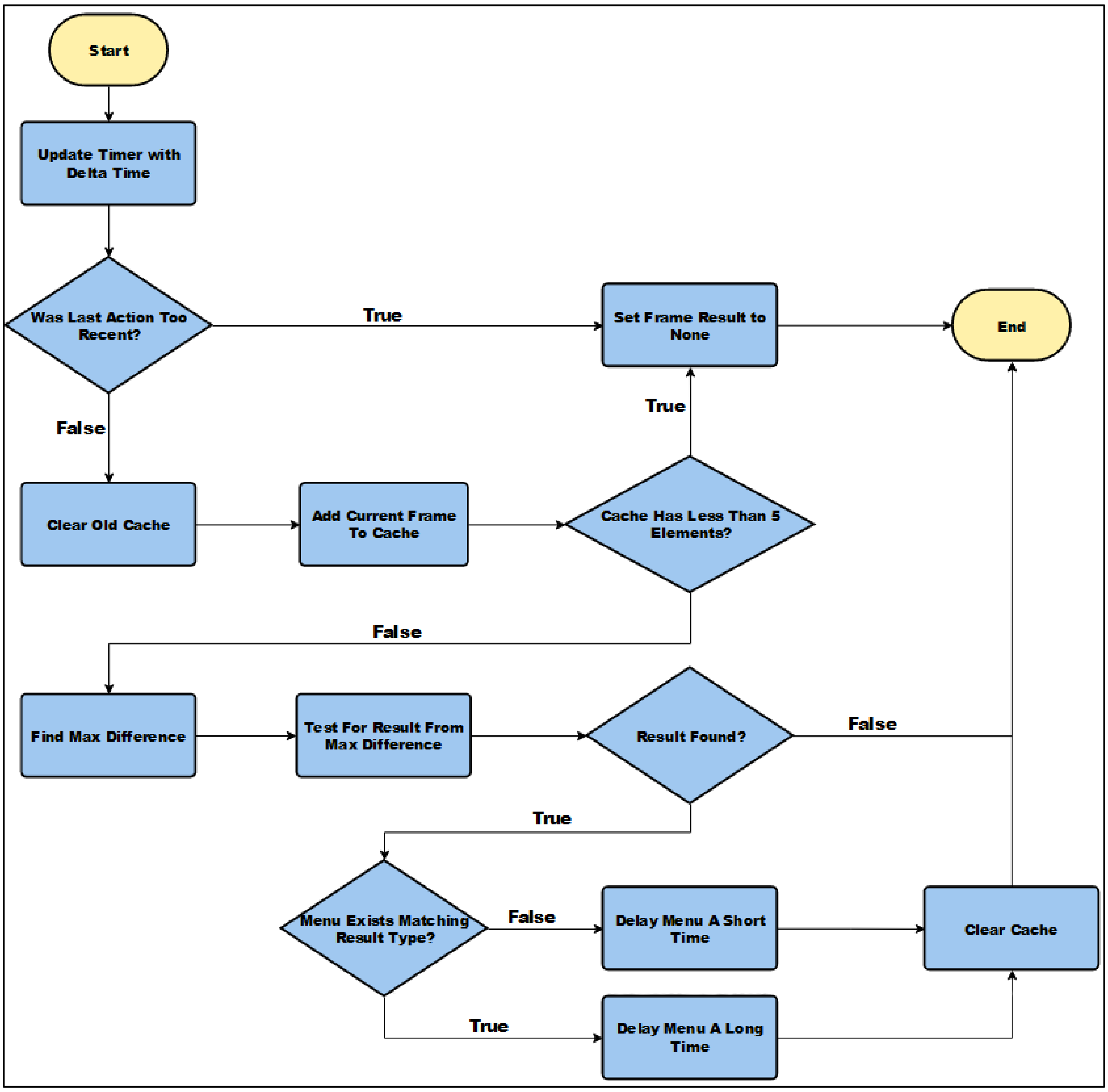

Start
Update Timer with Delta Time
Was Last Action Too Recent?
True
Set Frame Result to None
End
False
True
Clear Old Cache
Add Current Frame To Cache
Cache Has Less Than 5 Elements?
False
Find Max Difference
Test For Result From Max Difference
Result Found?
False
True
Menu Exists Matching Result Type?
False
Delay Menu A Short Time
Clear Cache
True
Delay Menu A Long Time

Figure 4.12: Process for Menu Reveal

The cache of rotational data is represented by a simple list with four numbers stored in `Vector4` type objects. Including the Euler angles for X, Y, Z and the time associated with the interaction. The time between frames is not constant, so this was tracked using the delta time between updates. This code would be run every update as part of an update loop. When elements in the history became too old, they were removed. This element expiry was determined based on the period of time for a trigger to occur. There is no reason to keep the information longer than the maximum time for performing a gesture. Removal of history, or when no new history has been added, leads to situations where the cache will have one or very few elements as it begins to repopulate with events.

For this reason, a minimum cache size of 5 was required to perform the logic used to detect a rotation trigger. During testing on hardware, there would typically be close to 17 elements in this list during the experiments. Once the cache was validated, the next step when determining if a trigger has occurred was to check for the maximum difference between the event history. This maximum difference was then compared against sensitivity configuration settings to determine if the event occurred.

```
PeripheryBehaviour (Experiment 3)
```

```
public enum Sensitivity { Low, Medium,
High }
public enum RotationResult { None = 0, Up
= 1, Down = 2, Left = 3, Right = 4 }

// Predefined Configurations
+ SensitivityConfig lowSensitivity
+ SensitivityConfig mediumSensitivity
+ SensitivityConfig highSensitivity

// Current Configuration
+ Sensitivity curSensitivity
+ float thresholdAngle
+ float thresholdZAngle
+ float thresholdOtherAngle
+ float thresholdTime

// Logic Variables
+ List<Vector4> history
+ float timerBetweenEvents = 2.0f
+ float timerBetweenEventsNoMenu = 0.1f
+ float ignoreTimesUntil = 0
+ RotationResult curFrameResult =
RotationResult.None
- float curTime = 0

// Menu Variables
+ GameObject leftNode
+ GameObject rightNode
+ GameObject menuObject
+ MenuBehaviour menuBehaviour
+ string leftMenuDef
+ string rightMenuDef
```

```
// Other Experiment Variables
+ bool showDebug
+ int[] lookCounter = new int[4]
+ ExtraDataRecorder extraDataRecorder
+
ExtraDataRecorder.ExtraDataCollection
peripheryEventLog
+ ExperimentState experimentState

// Methods
+ void Start()
+ void update(float deltaTime)
+ void updateRotationTrigger(float
deltaTime)
+ void
updateMenuObjects(floatDeltaTime)
+ void addCurFrame(float curTime)
+ void clearCache(float curTime)
+ Vector3
getMaxDifference(List<Vector4>
history)
+ RotationResult
detectResultFromDifference(Vector3
difference)
+ void ignoreEventsForNext(float time)
+ void setSensitivity(Sensitivity
newSensitivity)
+ void
setSensitivity(SensitivityConfig
config)
+ float fixRotation(float rotation,
float normalPoint)
- void generateLogData(Vector3
difference)
```

Figure 4.13: PeripheryBehaviour Class Definition

Figure 4.13 shows the structure used for the **`PeripheryBehaviour`** class. Variables have been separated to show areas they were related to within the overall structure. The predefined configurations with low, medium and high were used for the second experiment. Outside of one task where the sensitivity settings were compared, the other tasks for experiments two and three used medium sensitivity. The current configuration would be stored in the **curSensitivity**, and threshold variables when a sensitivity configuration was set. The logic variables include the history with all the rotation data over time, the set of timers for ignoring events, and the **RotationResult** to indicate a trigger has occurred. The menu variables give references to objects for positions of where menus would be placed, the menu to create, and the contextual menu definitions for each direction. Only left and right triggers have been implemented with spawning functionality as part of this code.

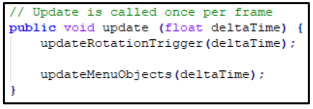
```
                  PeripheryBehaviour.SensitivityConfig (Experiment 3)
+ float thresholdAngle
+ float thresholdZAngle
+ float thresholdOtherAngle
+ float thresholdTime

// Constructor
+ SensitivityConfig(float thesholdAngle, float thresholdZAngle, float
thresholdOtherAngle, float thresholdTime)
```

Figure 4.14: SensitivityConfig Class Definition

The Sensitivity was defined with four different variables, as seen above in Figure 4.14. The variables were used in a couple of different ways to provide thresholds. The most important one for an event occurring was the **thresholdAngle**. This angle, represented in degrees, is the minimum rotation required for an interaction to occur in the left, right, up, or down directions. It is not enough to just look at a single rotation when considering if an event has occurred. Looking “wildly” in many different directions should not cause an event to trigger – this may be the player or user scanning the scene. The variables **thresholdZAngle** and **thresholdOtherAngle** represent maximum rotations. The Z angle rotation could be considered tilting of the head from side to side. When too much of this rotation occurs, it would indicate a menu was likely not desired. The **thresholdOtherAngle** was used to handle the opposite rotation direction based on the **thresholdAngle** direction. For example, if testing for **thresholdAngle**s in the left/right directions, the **thresholdOtherAngle** would restrict movement in the up/down direction. The opposite would also be true. The **thresholdTime** was used to indicate the maximum amount of time during which a trigger event could occur. The definitions of these sensitivities can be seen in Table 4.1 (over).

Table 4.1: PVMS Sensitivity Configurations

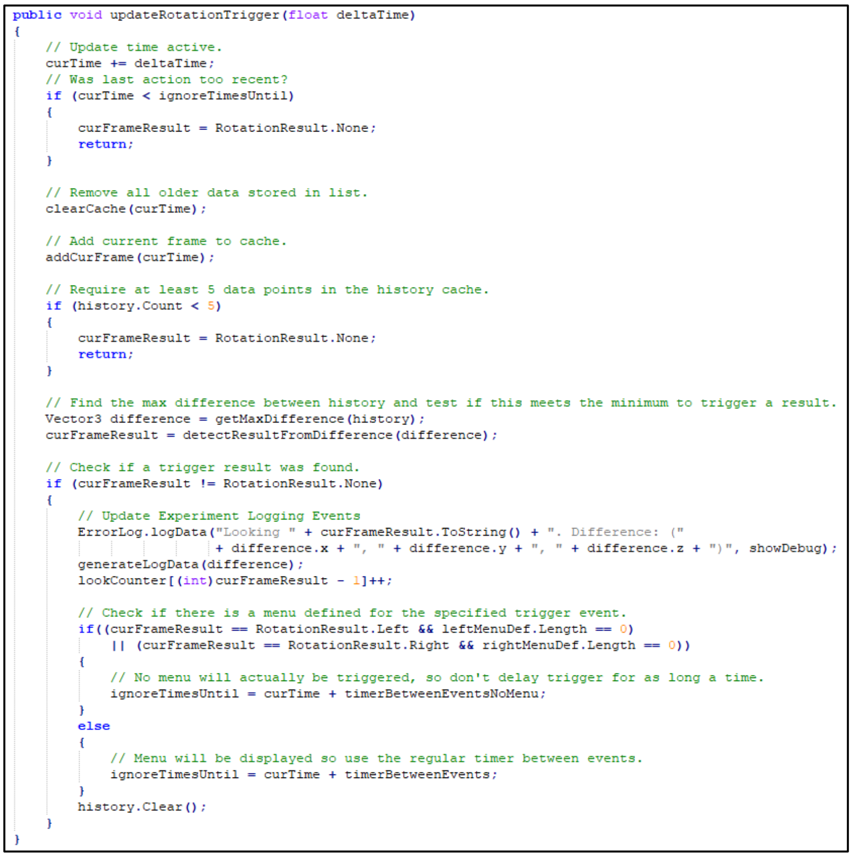
Name	ThresholdAngle	ThresholdZAngle	ThresholdOtherAngle	ThresholdTime
Low	25	30	30	0.35f
Medium	25	30	30	0.3f
High	25	25	25	0.25f



```
// Update is called once per frame
public void update (float deltaTime) {
    updateRotationTrigger(deltaTime);

    updateMenuObjects(deltaTime);
}
```

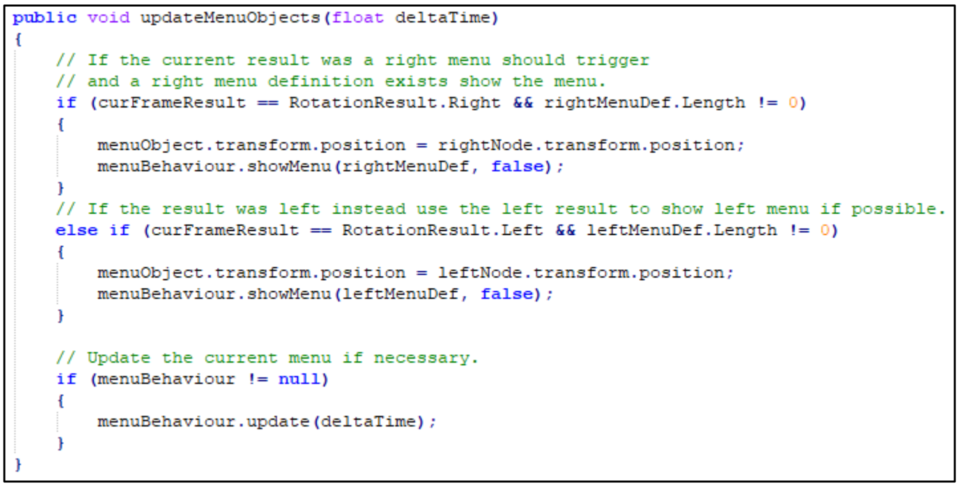
Listing 4.1: Periphery Behaviour Update Method

The remainder of this section will show the code used to define the interactions with brief explanations. Most of this will focus on the processes used as part of the update loop; the update function is shown in Listing 4.1. This method initiates the update of trigger events, then once any new trigger states have been confirmed, the menu objects related to the event are updated. Listing 4.2 shows the code version of the earlier flow diagram from . The comments further explain how each part functions. For the complete code, see Appendix D for details about accessing it on GitHub.



```
public void updateRotationTrigger(float deltaTime)
{
    // Update time active.
    curTime += deltaTime;
    // Was last action too recent?
    if (curTime < ignoreTimesUntil)
    {
        curFrameResult = RotationResult.None;
        return;
    }

    // Remove all older data stored in list.
    clearCache(curTime);

    // Add current frame to cache.
    addCurFrame(curTime);

    // Require at least 5 data points in the history cache.
    if (history.Count < 5)
    {
        curFrameResult = RotationResult.None;
        return;
    }

    // Find the max difference between history and test if this meets the minimum to trigger a result.
    Vector3 difference = getMaxDifference(history);
    curFrameResult = detectResultFromDifference(difference);

    // Check if a trigger result was found.
    if (curFrameResult != RotationResult.None)
    {
        // Update Experiment Logging Events
        ErrorLog.logData("Looking " + curFrameResult.ToString() + ". Difference: ("
                                    + difference.x + ", " + difference.y + ", " + difference.z + ")", showDebug);
        generateLogData(difference);
        lookCounter[(int)curFrameResult - 1]++;

        // Check if there is a menu defined for the specified trigger event.
        if((curFrameResult == RotationResult.Left && leftMenuDef.Length == 0)
            || (curFrameResult == RotationResult.Right && rightMenuDef.Length == 0))
        {
            // No menu will actually be triggered, so don't delay trigger for as long a time.
            ignoreTimesUntil = curTime + timerBetweenEventsNoMenu;
        }
        else
        {
            // Menu will be displayed so use the regular timer between events.
            ignoreTimesUntil = curTime + timerBetweenEvents;
        }
        history.Clear();
    }
}
```

Listing 4.2: Periphery Behaviour Update Rotation Trigger Method

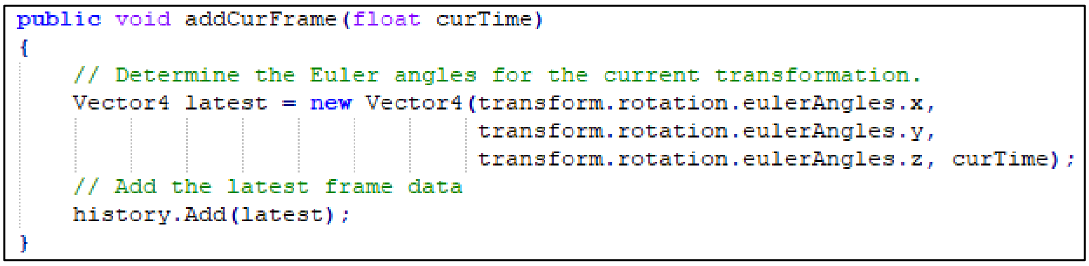


```
public void updateMenuObjects(float deltaTime)
{
    // If the current result was a right menu should trigger
    // and a right menu definition exists show the menu.
    if (curFrameResult == RotationResult.Right && rightMenuDef.Length != 0)
    {
        menuObject.transform.position = rightNode.transform.position;
        menuBehaviour.showMenu(rightMenuDef, false);
    }
    // If the result was left instead use the left result to show left menu if possible.
    else if (curFrameResult == RotationResult.Left && leftMenuDef.Length != 0)
    {
        menuObject.transform.position = leftNode.transform.position;
        menuBehaviour.showMenu(leftMenuDef, false);
    }

    // Update the current menu if necessary.
    if (menuBehaviour != null)
    {
        menuBehaviour.update(deltaTime);
    }
}
```

Listing 4.3: Periphery Behaviour Update Menu Objects Method

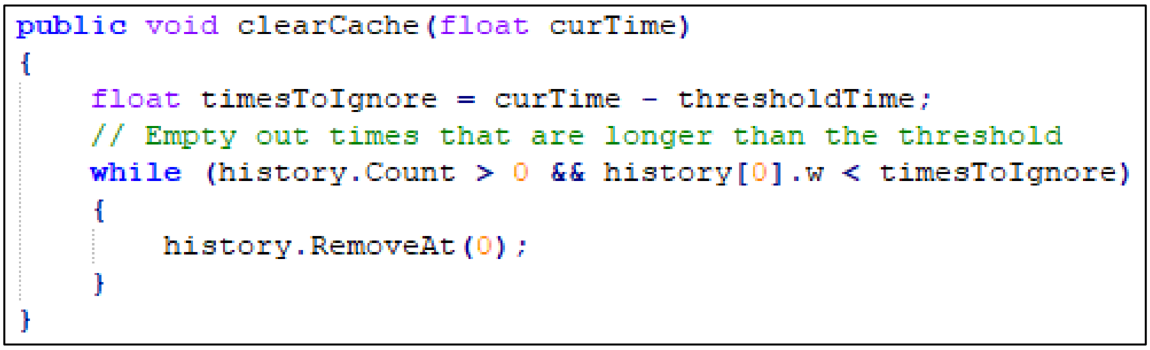
Updating menu objects fell into two different types of updates, as seen in Listing 4.3. At first, a check was made to determine if a **`RotationResult`** had been triggered by the prior method. If the right or left trigger had occurred, this code would position the menu based on the transform of the right or left node objects attached to the camera. These nodes can be seen in section 4.3.3 detailing the positioning of the menu relative to the user. After setting the updated position, the text and other menu definition information were updated with a call to **`showMenu`**. The second parameter used for **`showMenu`** as false indicates whether the two-step widget should be skipped. Meaning this will show the two-step widget initially, and then the two-step widget would use a similar call but with true to make the menu appear.

The **`PeripheryBehaviour`** was attached to the camera's game object to access the position and rotation representing the game object's transform. When Unity has the HMD interaction enabled, this transform is automatically updated based on the device's sensor information. The rotation object is a Quaternion type but provides a way to represent the data as a Euler angle, as seen in Listing 4.4. The rotation information is combined with current time into the **`Vector4`** and added to the history cache.



```
public void addCurFrame(float curTime)
{
    // Determine the Euler angles for the current transformation.
    Vector4 latest = new Vector4(transform.rotation.eulerAngles.x,
                                 transform.rotation.eulerAngles.y,
                                 transform.rotation.eulerAngles.z, curTime);
    // Add the latest frame data
    history.Add(latest);
}
```

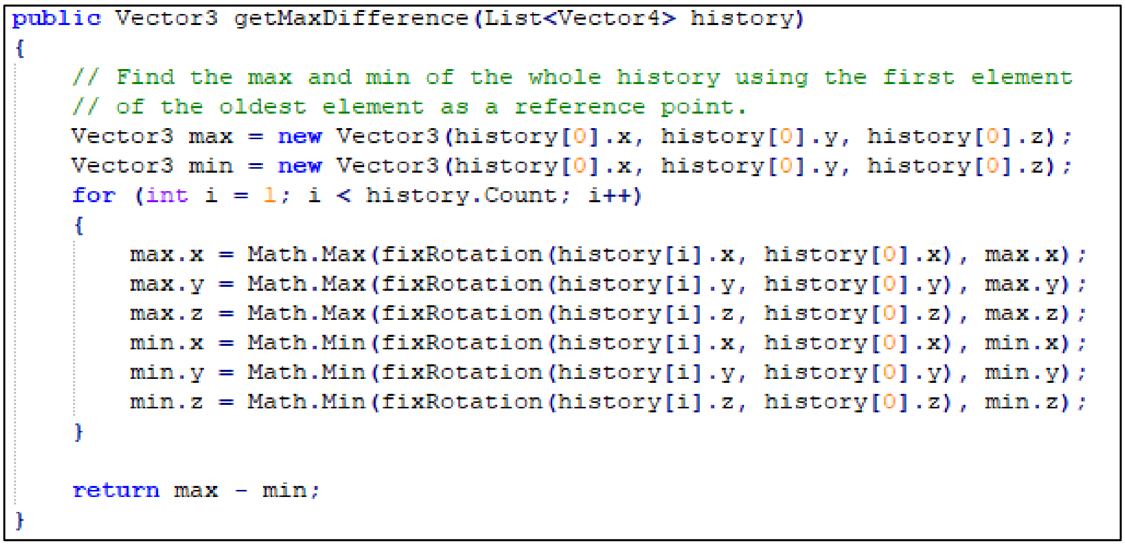
Listing 4.4: Periphery Behaviour Add Current Frame Method



```
public void clearCache(float curTime)
{
    float timesToIgnore = curTime - thresholdTime;
    // Empty out times that are longer than the threshold
    while (history.Count > 0 && history[0].w < timesToIgnore)
    {
        history.RemoveAt(0);
    }
}
```

Listing 4.5: Periphery Behaviour Clear Cache Method

Clearing the cache used the **`thresholdTime`** to determine the maximum relevant time period to consider, as shown in Listing 4.5. The code pops elements from the list while any at the start of the list do not meet the criteria.

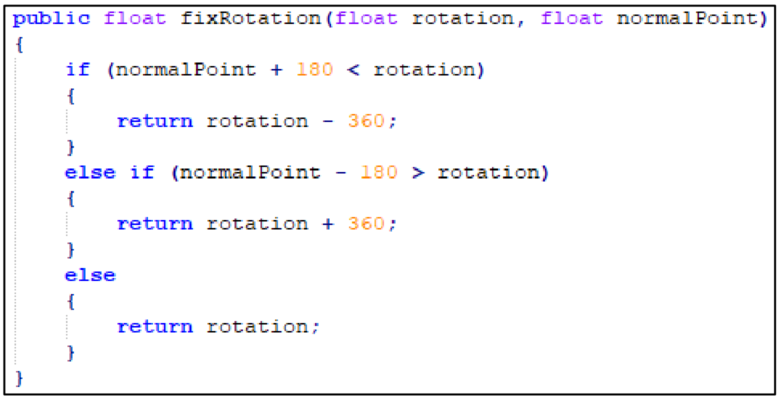


```
public Vector3 getMaxDifference(List<Vector4> history)
{
    // Find the max and min of the whole history using the first element
    // of the oldest element as a reference point.
    Vector3 max = new Vector3(history[0].x, history[0].y, history[0].z);
    Vector3 min = new Vector3(history[0].x, history[0].y, history[0].z);
    for (int i = 1; i < history.Count; i++)
    {
        max.x = Math.Max(fixRotation(history[i].x, history[0].x), max.x);
        max.y = Math.Max(fixRotation(history[i].y, history[0].y), max.y);
        max.z = Math.Max(fixRotation(history[i].z, history[0].z), max.z);
        min.x = Math.Min(fixRotation(history[i].x, history[0].x), min.x);
        min.y = Math.Min(fixRotation(history[i].y, history[0].y), min.y);
        min.z = Math.Min(fixRotation(history[i].z, history[0].z), min.z);
    }

    return max - min;
}
```

Listing 4.6: Periphery Behaviour Get Max Difference Method

Listing 4.6 shows how the primary metrics used for rotation detection were calculated. The maximum and minimum for each of X, Y, and Z were calculated. These were recalculated every time in full for the prototype to ensure everything was working from a functional and usability point of view. This code could be simplified by further caching parts of this calculation. The main point of complication was with how angles are represented and therefore required the **`fixRotation`** method. This method will be discussed next. The output from this method provided the difference between maximum and minimum angles. These could be used for determining if angles were under or over the threshold sensitivities.

Listing 4.7 demonstrates the functionality of **`fixRotation`** method. The goal for this method was to shift rotations to become a continuous line. In Unity, the numbers provided by the Euler angles are between 0 and 360. Figure 4.15 shows a visual indication of what this means. On the left is a circle that is broken at the top. Angles start at 0 degrees and can go up to 360 degrees. This number range becomes a problem when the user rotates over the line represented by the gap. Values would fluctuate wildly from 359 to 1, for example. When applying a difference between these two numbers, it would come out as 358 when it is actually a difference of 2 degrees.



```
public float fixRotation(float rotation, float normalPoint)
{
    if (normalPoint + 180 < rotation)
    {
        return rotation - 360;
    }
    else if (normalPoint - 180 > rotation)
    {
        return rotation + 360;
    }
    else
    {
        return rotation;
    }
}
```

Listing 4.7: Periphery Behaviour Fix Rotation Method

Values were fixed to provide continuous data to solve this problem. An assumption was made that a rotation would never be more than 180 degrees in a single event. Performing a 180-degree rotation in 1/60th of a second would likely cause neck injury and, therefore, a safe assumption. The yellow additions to Figure 4.15 visually demonstrate this extension. By adding or subtracting 360 degrees from angles in the blue section, they could be shifted to a range of 180 to 540 degrees or -180 to 180 degrees. This shift used the oldest element in the event history, representing the likely starting point for a rotation during the current frame.

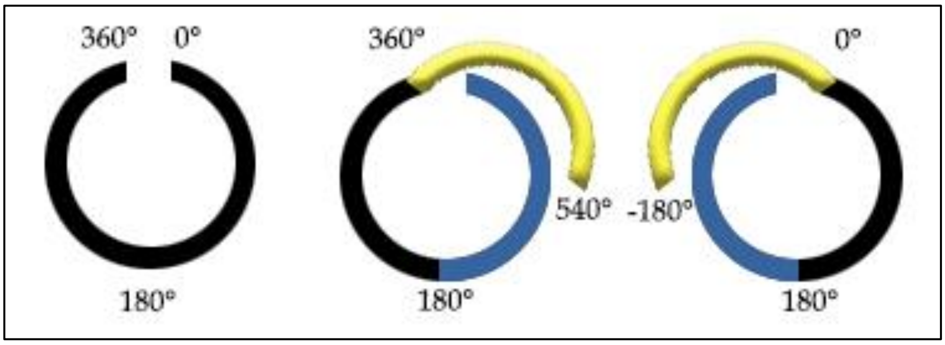
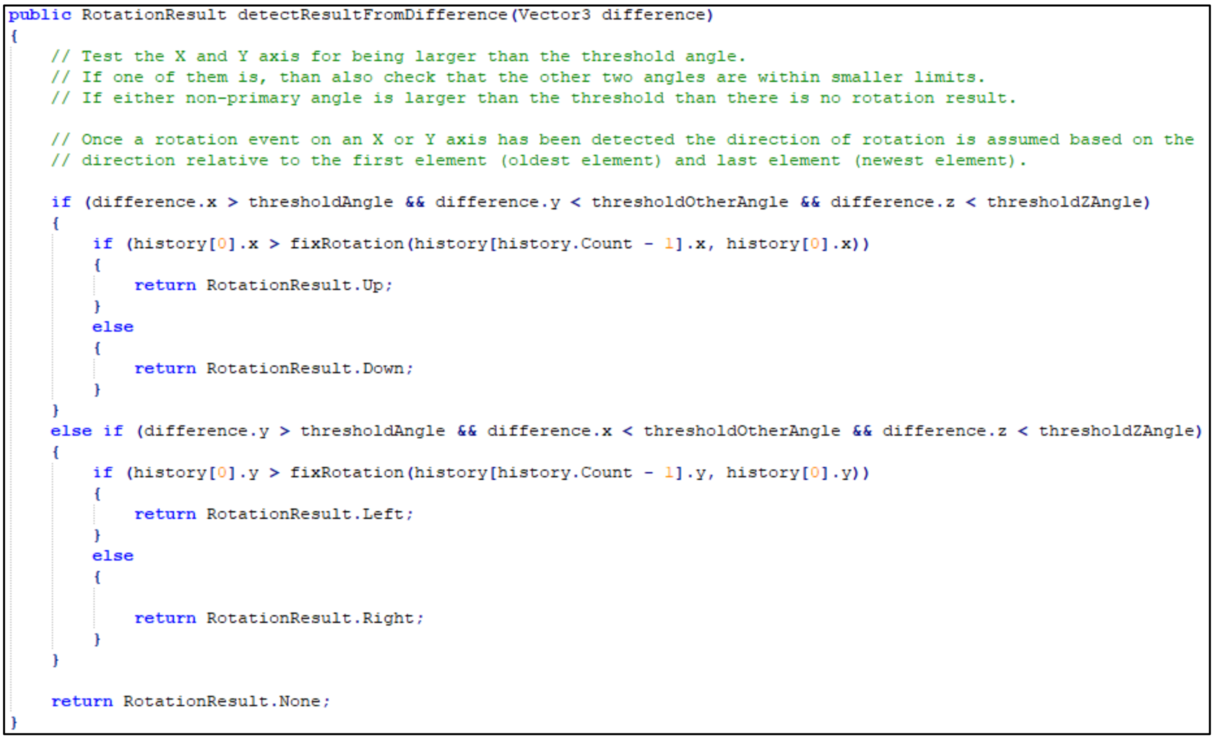


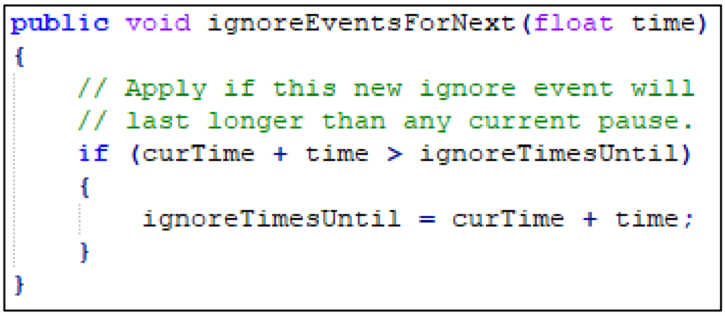
Figure 4.15: Fix Rotation Visual Representation

Listing 4.8 shows the method that uses the calculated difference data and sensitivity thresholds to determine if an event has occurred. When determining which result should be used, the difference and thresholds are compared. Then if these are found to match a solution for either up/down or left/right, the direction is determined based on the first and last elements in the cached history. This deterministic model for selecting rotation results will keep giving consistent results for each situation where conditions are met in the same way.



```
public RotationResult detectResultFromDifference(Vector3 difference)
{
    // Test the X and Y axis for being larger than the threshold angle.
    // If one of them is, than also check that the other two angles are within smaller limits.
    // If either non-primary angle is larger than the threshold than there is no rotation result.

    // Once a rotation event on an X or Y axis has been detected the direction of rotation is assumed based on the
    // direction relative to the first element (oldest element) and last element (newest element).

    if (difference.x > thresholdAngle && difference.y < thresholdOtherAngle && difference.z < thresholdZAngle)
    {
        if (history[0].x > fixRotation(history[history.Count - 1].x, history[0].x))
        {
            return RotationResult.Up;
        }
        else
        {
            return RotationResult.Down;
        }
    }
    else if (difference.y > thresholdAngle && difference.x < thresholdOtherAngle && difference.z < thresholdZAngle)
    {
        if (history[0].y > fixRotation(history[history.Count - 1].y, history[0].y))
        {
            return RotationResult.Left;
        }
        else
        {

            return RotationResult.Right;
        }
    }

    return RotationResult.None;
}
```

Listing 4.8: Periphery Behaviour Detect Result from Difference Method



```
public void ignoreEventsForNext(float time)
{
    // Apply if this new ignore event will
    // last longer than any current pause.
    if (curTime + time > ignoreTimesUntil)
    {
        ignoreTimesUntil = curTime + time;
    }
}
```

Listing 4.9: Periphery Behaviour Ignore Events for Next Method

Any time it was necessary to pause interactions with the PVMS, the `ignoreEventsForNext` method could be applied. Listing 4.9 shows an implementation of this method where time would not continue to stack longer for pausing if there was already a longer pause. This version of the method was mostly used for dealing with quick swapping between sensitivity settings so that no events happened for a minimum time after the change.

4.4.2 Interacting with the Revealed Menu

As part of interacting with the revealed menus, there were three different aspects to consider; the way selection occurs, how menus are hidden, and the two-step widget. The general theme for interaction focused on providing users with a way they could interact with only their heads. The way this was primarily achieved was through a delayed selection (dwell or hover operation) progress to ensure users were interacting with the element they wanted to.

Hover-to-Select

Using the head as an interaction tool with hover selection is possible using only the HMD with no assumption of additional controller devices. Figure 4.16 shows an example of the changing cursor colour. This cursor colour was used to provide immediate visual feedback for the selection progress. The point of interaction was defined using the screen's width and height divided by two. This interaction point is lined up with the visual cursor's centre. After moving the cursor (the focus of the user's field of view) over a button, the time to select for most buttons in experiments two and three was set to 1.5 seconds. Moving out of a button too early would cancel this selection progress and require it to be started again.

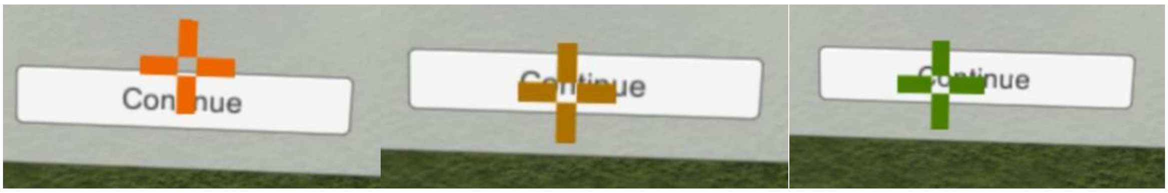
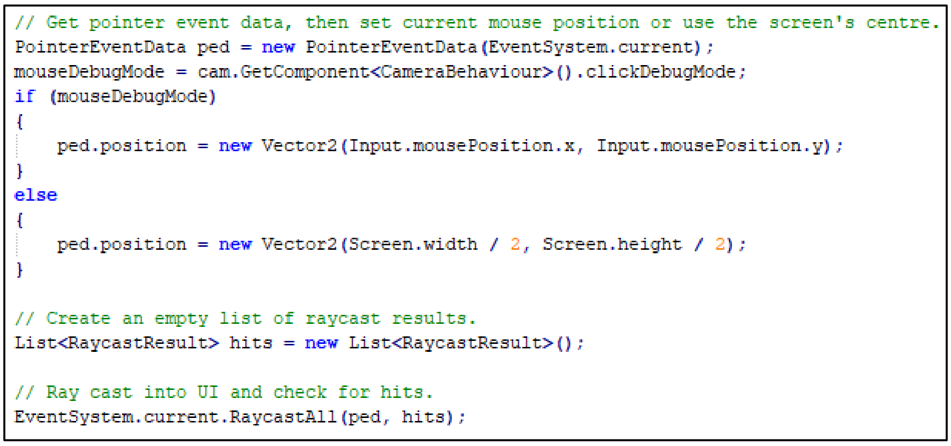


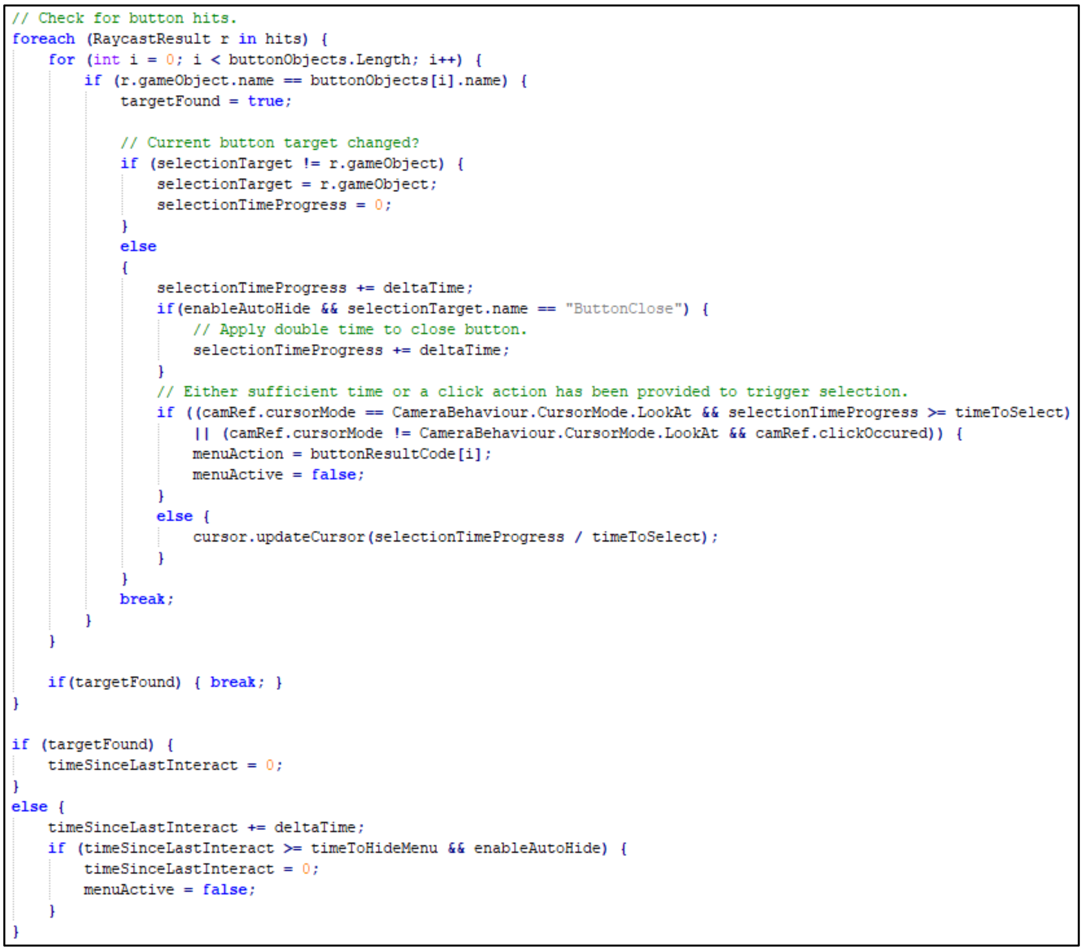
Figure 4.16: Cursor Colour Change

The Unity interface event system provides a way to perform ray casting for hit detection. The interface elements are all represented as game objects, meaning when a ray cast is completed, the returned list will provide all intersected game objects along a ray. Listing 4.10 (over) shows the example code used to determine intersected elements.



```
// Get pointer event data, then set current mouse position or use the screen's centre.
PointerEventData ped = new PointerEventData(EventSystem.current);
mouseDebugMode = cam.GetComponent<CameraBehaviour>().clickDebugMode;
if (mouseDebugMode)
{
    ped.position = new Vector2(Input.mousePosition.x, Input.mousePosition.y);
}
else
{
    ped.position = new Vector2(Screen.width / 2, Screen.height / 2);
}

// Create an empty list of raycast results.
List<RaycastResult> hits = new List<RaycastResult>();

// Ray cast into UI and check for hits.
EventSystem.current.RaycastAll(ped, hits);
```

Listing 4.10: Collecting Ray Cast Data



```
// Check for button hits.
foreach (RaycastResult r in hits) {
    for (int i = 0; i < buttonObjects.Length; i++) {
        if (r.gameObject.name == buttonObjects[i].name) {
            targetFound = true;

            // Current button target changed?
            if (selectionTarget != r.gameObject) {
                selectionTarget = r.gameObject;
                selectionTimeProgress = 0;
            }
            else
            {
                selectionTimeProgress += deltaTime;
                if(enableAutoHide && selectionTarget.name == "ButtonClose") {
                    // Apply double time to close button.
                    selectionTimeProgress += deltaTime;
                }
                // Either sufficient time or a click action has been provided to trigger selection.
                if ((camRef.cursorMode == CameraBehaviour.CursorMode.LookAt && selectionTimeProgress >= timeToSelect)
                    || (camRef.cursorMode != CameraBehaviour.CursorMode.LookAt && camRef.clickOccured)) {
                    menuAction = buttonResultCode[i];
                    menuActive = false;
                }
                else {
                    cursor.updateCursor(selectionTimeProgress / timeToSelect);
                }
            }
            break;
        }
    }

    if(targetFound) { break; }
}

if (targetFound) {
    timeSinceLastInteract = 0;
}
else {
    timeSinceLastInteract += deltaTime;
    if (timeSinceLastInteract >= timeToHideMenu && enableAutoHide) {
        timeSinceLastInteract = 0;
        menuActive = false;
    }
}
```

Listing 4.11: Ray Collision Detection and Automatic Hiding

Listing 4.11 (over) shows how the generated ray cast information is then used to iterate over the buttons in the menu. Once a button is found to have been intersected, the selection logic takes over and updates progress until a successful selection is made. As part of selected tasks in experiments two and three, interactions were included for instant selection with a click, tap, or button press on additional handheld interaction devices. This code would also allow these interactions if the input mode allowed for it. The input mode is represented by the `CursorMode` enum where `CursorMode.LookAt` refers to HMD Only interaction.

Menu Hiding

There are three situations where a menu would be hidden. The first is when a successful button interaction occurs, resulting in a stored button code and the menu's active property is set to false. The second is when the close button is interacted with by the user. The third experiment added this close button to provide an alternative for menu closing. The close button could be activated twice as fast as any other button; the close button can be seen in Figure 4.17. Due to the small size of the close button, the faster selection time allows directed, fast termination to remove undesired menus. The third way was to automatically close menus after a period of no interaction. The final else clause from Listing 4.11 covers this case. If no target was found, then the period of no interaction starts again, but otherwise, the timer will continue to tick until the dialog is closed. This period was set to 4 seconds in both experiments. An important distinction is that interacting with the menu's background also deliberately counted toward ray cast interaction.

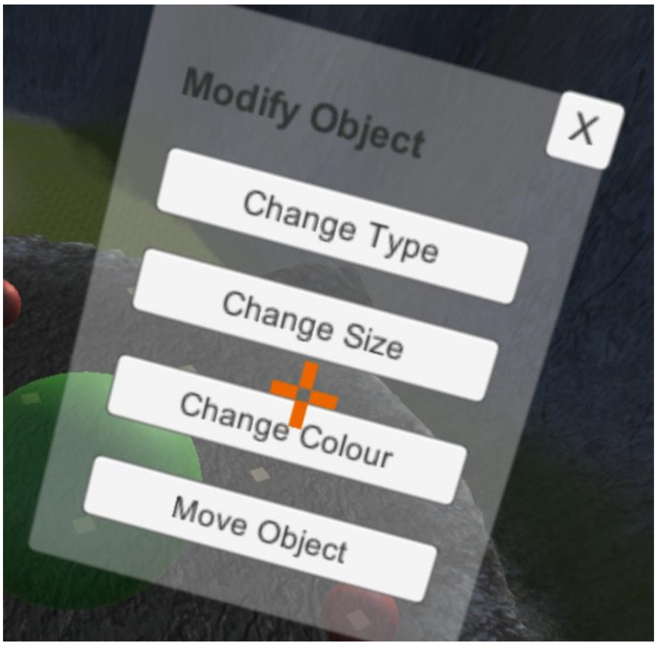


Figure 4.17: Example Interface with Close Button

The menu was not hidden immediately when the user turned their head away for two reasons. In the case of the second experiment, the user did not make the menu appear at the current facing, so the user had to turn to the menu, causing it not to make sense to hide the menu immediately if it wasn't being observed in this situation. The other reason related to both experiments was that by leaving the menu visible despite looking away from it, users could observe the current task context and then return to an already opened menu when ready. Developers could either reduce the time to hide or remove it entirely if appropriate to an application.

Two-Step Widget

The third experiment added the two-step widget to reduce the impact of unintended menu activation. In cases where the menu was triggered accidentally, or a user's mind had changed about opening the menu, this in-between step provided more control. Figure 4.18 shows an example of the button that would appear for representing this process. Interacting with the button by hovering anywhere over it would instantly trigger the display of the real menu with no delay on selection time, leaving the cursor already in the middle of the newly opened menu. Due to the far smaller impact of the two-step widget, this icon was automatically set to hide after a period of 8 seconds.

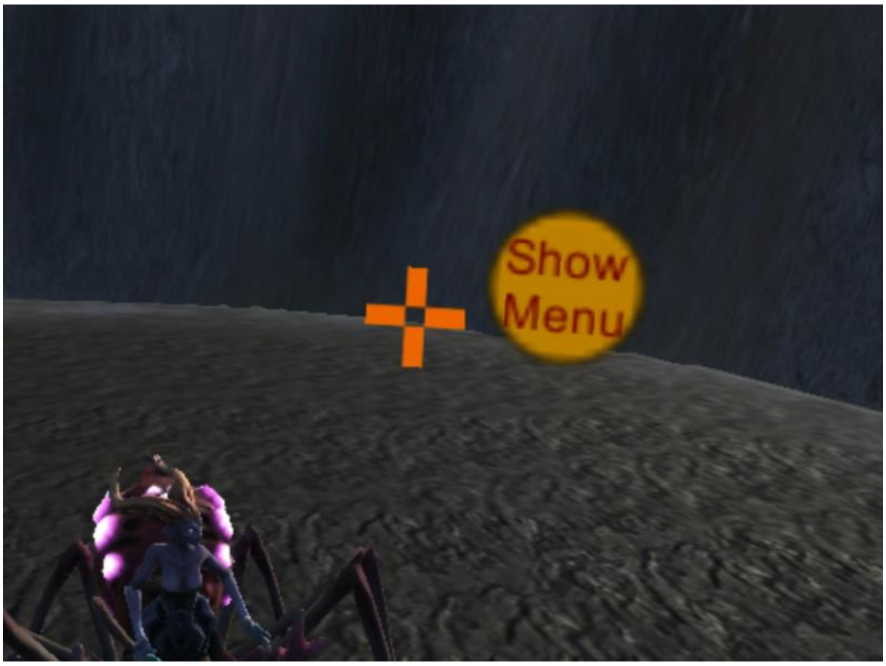


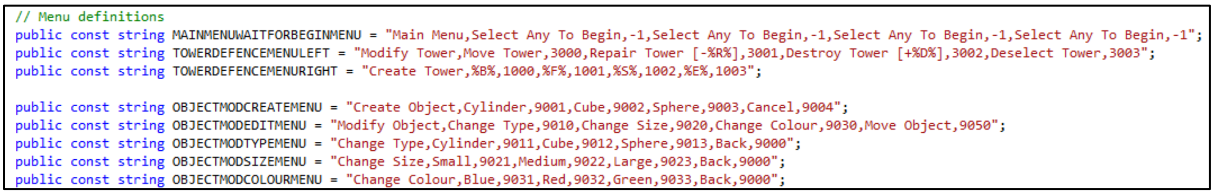
Figure 4.18: Two-Step Widget

All the timings implemented were designed to be used by novice users of the system, i.e., where the users may not have much experience with VR. The numbers could be easily modified to provide a faster interaction experience for users, depending on their personal preferences. For experienced users, the timings could perhaps be halved to give a significantly more responsive experience.

4.5 Contextual Menu Support

Contextual menus are important because of the connection between any specific situation and the expectation by the user that an action will occur. These situations do not need to have complicated scenarios for most situations. For example, while in a game lobby, the menus expected would be main menus or perhaps an inventory allowing customisation, as discussed in section 4.2.3. Or, while selecting an object, the menu could provide options for modifying the selected object. This approach is analogous to the "right-click" menu presentation in typical Windows applications; for example, right-clicking on a cell in the Excel application will display a menu that displays interaction items for that specific cell. The most important feature is that the situations these appear in are consistent; a hidden menu system needs this consistency to remain usable. All menus used in the experiments were designed and created with a clear goal and were explained to the participants in advance to make them aware of the contexts they would encounter.

The model used for this research to provide contextual support was based on a set of state machines. Each high-level game state could have separate left and right menu definitions. Leaving these fields blank would result in no menu for that trigger case. The managers for each task respond to the result codes from menu interactions to determine what function should be performed. Some example menu definitions can be seen in Listing 4.12.



```
// Menu definitions
public const string MAINMENUWAITFORBEGINMENU = "Main Menu,Select Any To Begin,-1,Select Any To Begin,-1,Select Any To Begin,-1,Select Any To Begin,-1";
public const string TOWERDEFENCEMENULEFT = "Modify Tower,Move Tower,3000,Repair Tower [-%R%],3001,Destroy Tower [+%D%],3002,Deselect Tower,3003";
public const string TOWERDEFENCEMENURIGHT = "Create Tower,%B%,1000,%F%,1001,%S%,1002,%E%,1003";

public const string OBJECTMODCREATEMENU = "Create Object,Cylinder,9001,Cube,9002,Sphere,9003,Cancel,9004";
public const string OBJECTMODEDITMENU = "Modify Object,Change Type,9010,Change Size,9020,Change Colour,9030,Move Object,9050";
public const string OBJECTMODTYPEMENU = "Change Type,Cylinder,9011,Cube,9012,Sphere,9013,Back,9000";
public const string OBJECTMODSIZEMENU = "Change Size,Small,9021,Medium,9022,Large,9023,Back,9000";
public const string OBJECTMODCOLOURMENU = "Change Colour,Blue,9031,Red,9032,Green,9033,Back,9000";
```

Listing 4.12: Experiment Three Menu Definitions

By defining menus in a simple reusable way, the quick definitions can be reused for simple tasks. In cases where more complex menus are desired with more unique layouts for situations, it would be more appropriate to store a menu reference to the predefined menu instead of storing properties in the way used here. One potential change that could be applied to the PVMS is variable button quantities. For testing, a static quantity of four buttons was used. A variable number of buttons could provide appropriate interaction options for a needed activity.

4.6 Definition of Core Components and Recommendations

This section reiterates the core components of the PVMS as they were used in the experiments to define clearly the prototype implementation that has been developed. Some of the elements are recommendations and could be altered to become more suitable for a target application. For a discussion about the framework with reflection on the second and third experiments, see section 8.3.

Interaction Detection Requirements

The PVMS has been designed to function with no calibration if a user chooses to use the default configuration. To accomplish this, a target HMD, whether that be for AR or VR, should support at least 3DOF to track the yaw, pitch, and roll. The mechanism for how these values are collected is not necessarily important. Provided the data generated by any HMD is continuous and accurate enough that a user will experience no significant latency in the application's rendering to match changes when turning their head. In the case of the Unity development environment setting a camera object to perform as a VR element will automatically use the data supplied by the HMD. The experiments attached the necessary scripts as a component under the VR camera in the Unity hierarchy to simplify accessing the orientation data that was automatically updated from the HMD. This type of nested attachment would not be necessary for it to function. It would also be suitable to modify the provided framework as an independent observer operating from a separate object to detect triggers. Another benefit of nesting as part of the hierarchy is that it is necessary to reference an offset from the camera to make the menus appear. So the framework could be packaged together with the offset. In the implementation, these offsets were handled using invisible objects (see Figure 4.9 and Figure 4.10) that were also translated and rotated by input from the HMD as a child object to the VR camera. The values used for the offsets are detailed in Table 4.2 (over). It would be possible to calculate offsets from a current orientation when the trigger occurred if it was not practical or desirable for an application to use the invisible object approach. A minimum viable requirement for interaction detection would only require a HMD with 3DOF and access to that information.

Table 4.2: Offsets

Offset Name	Offset Value Used (X, Y, Z) Offsets are from Camera position (0,0,0) with (0,0,0) rotation and (1,1,1) scale.
Left Menu Spawn Node	Position translation of (-3.84, 0, 5.84).
Right Menu Spawn Node	Position translation of (3.84, 0, 5.84).
Cursor	Position translation of (0,0,2)

Calibration

Calibration has already been discussed as part of this chapter concerning proposed sensitivity configurations with specific values used for each pre-selected variation, as seen in Table 4.1. The calibration elements include the **`ThresholdAngle`**, **`ThresholdZAngle`**, **`ThresholdOtherAngle`**, and **`ThresholdTime`**. The **`ThresholdAngle`** is the most significant element for detection as it defines how far the user must turn left/right or up/down to trigger a menu. Only the left/right triggers were used for the experiments, but the system is suitably functional to support the up/down triggers. The angle is the minimum rotation required to trigger the menu. The minimum of **`ThresholdAngle`** had to be reached within **`ThresholdTime`**. The **`ThresholdZAngle`** represents a maximum restriction on tilting your head (roll) when triggering a menu, and **`ThresholdOtherAngle`** represents a maximum restriction on the pitch angle for a left/right detection or yaw for up/down. Calibration using these metrics can be selected by providing a series of options for each metric or a scaling change that modifies all of them. The medium sensitivity was selected for use in the experiments to provide a suitable experience that felt appropriate to the experience provided by testing by the researcher before experimentation with participants. Similar testing could be applied for applications using the PVMS to provide pre-set options for users to choose their preferred experience quickly.

When selecting or defining a calibration, it is essential to consider what the user will be performing as part of an application. An application requiring a lot of frequent head movement may need a less lenient calibration than one that does not have much movement. This consideration is due to how the thresholds work; the stricter the thresholds are, the stricter the application is about an exact left/right gesture without deviation. If the thresholds are too broad, the menu will always appear, or a trigger will never occur if the thresholds are too tight. The **`ThresholdTime`** is one easy way to adjust the sensitivity because it controls how quickly the interaction can be. Reducing the time below 0.25 seconds used for the high sensitivity (or 0.3 seconds for others) would force the user to be more conscious about their gestures. Lowering the time too low or making the interaction requirement too large would likely cause significant fatigue to a user. Participants during the experiments did not appear to suffer significant fatigue for the 15 to 20 minutes they were exposed and interacting. Based on the experiences observed and data collected during the first experiment, the fatigue presented by the PVMS was suitably low. No significant stress test was performed to evaluate long use fatigue.

Cursor Guidelines

The cursor in the context of the experiments was represented as a crosshair, as seen in Figure 4.16. The crosshair served two purposes: first, showing the exact position where a raycast from the HMD would be performed for selection, and second, acting as a progress indicator. The PVMS could be used with no cursor, but if using HMD Only type interactions where the orientation of the HMD is used to detect a selection point, it will make the target point more apparent to a user. The cursor used in the application was a 3D model attached to the camera rig to keep all elements part of the scene instead of separating into a 2D overlay interface layer. There is nothing special about the choice of shape for the crosshair. The constraints considered as part of choosing the crosshair were to make it visible enough from size to see where it was aimed and with sizing enough to observe the colour gradient change. The colour gradient seen in Figure 4.16 was selected as an easy way to observe that a change was occurring. For all the selection operations in the application, there was a distinct change in state when the selection ended. It was not important for a participant to discern the exact progress of their selections. Suppose selection progress was a necessary value to be aware of; in that case, alternate progress visuals could be used. For example, a spinner that changes how far it loops around the cursor based on progress shows a shape change and may be easier to distinguish than a colour change as it can be measured from observation.

Timings

There are a variety of different timings that have been presented concerning the menu revealing, menu selection, and hiding after a period of no interaction. The timings were selected to give experiment participants an experience that did not burden them with overly rapid interaction requirements. Timings were mainly designed to allow participants to interpret and understand what they were experiencing as novice users of the systems. An experienced user could alter the timings to be faster based on their requirements, or a user with mobility issues could adjust the timings to be even more forgiving. The timings are summarised below in Table 4.3, except for `ThresholdTime`, which was discussed separately as a part of the calibration.

Table 4.3: Timings

Timing Name	Timing Value	What the Time Represents
Selection Time	1.5 seconds	The amount of time a user would have to aim the cursor at an interactable object (e.g. menu button) for a selection to complete.
Selection Time (Close Button)	0.75 seconds	The amount of time a user would have to aim at the close button to force a menu to hide.
Two-step Widget Hide Time	8 seconds	The amount of time before the two-step widget would automatically hide if there were no interaction.
Menu Hide Time	4 seconds	The amount of time before the revealed menu would automatically hide if there were no interactions. It would reset the timing based on hovering over any part of the interface's canvas.
Two-step Widget to Menu Transition	Instant	The time for transition when hovering over the two-step widget to make it transition into the options.
Delay Long	2 seconds	The minimum time until the next menu could be created with a gesture after a successful trigger.
Delay Short	0.1 seconds	The minimum time until the next menu if a successful trigger occurred when no menu could be created.

The two timings for hiding menus were deliberately high to enable a user to create the menu and continue observing their surrounding for a moment before making a choice. The two-step widget was made to be significantly longer before a hide occurred with double the time. The two-step widget was intentionally made small and likely to not be in the way. A user wishing to make it go away faster than the 8 seconds could hover to transition instantly to the menu and then use the close button. As the transition is instant when moving between the two-step widget and menu, it is ideal not to have an option selectable at the cursor's position after the transition completes. The long and short delays after a successful gesture could be reduced in time if more frequent gestures are required. Typically from the use in experiments, the delays were not noticeable and reduced the computational workload.

Two-step Widget

The two-step widget was only used as part of the third experiment but demonstrated from the evaluation of the data in the following chapters that it provided mitigation of user errors. The implementation used in the third experiment showed the text “Show Menu”, as seen in Figure 4.8. The text could be changed to tell a user what menu would be revealed through interaction. Any text used should be short because the widget should be kept small for its primary purpose. “Show Menu” presents two words that are clear and short, but if the context of the menu were to be different for different situations would not indicate which menu would be revealed. Table 4.4 presents the properties used for the two-step widget to allow replication.

Table 4.4: Two-Step Widget Visual Properties

Property	Value
Button Size	50px by 50px
Button Background Colour	RGBA: 195, 133, 0, 255
Button Text Font	Font: Arial, Style: Normal, Font Size: 14, Alignment: Centred, Horizontal Wrap
Button Text Colour	RGBA: 116, 0, 0, 255

PVMS Menu

The last content to define is the menu presenting options to the user. A similar menu appearance was used for both the second and third experiments, with the primary visual difference being the close button as part of the third experiment. The visual style of elements was derived principally from the defaults provided by Unity's interface components (including features like the rounded corners on buttons). The linear type menu designed for the second experiment considered both Hick's Law (Hick, 1952) and Fitts' Law (Fitts and Posner, 1967; Scott MacKenzie, 1992). Considering Hick's Law makes it ideal from menu design to present a small number of options with easy to understand choices. At the same time, Fitts' Law deals with travel to a specific target leading to suitably large elements for interaction. When the PVMS menu is revealed in the second experiment, a user from their offset to the menu creation location can direct their cursor to interact with any menu option without touching any other menu option.

The size of the buttons did contribute a larger surface for cursor interaction which is relevant as per Fitts' Law. As additional consideration for button size, the head as an interaction tool with hover-to-select required a user to hover over a button for the period defined in timings. Due to the dwell time, it was necessary to make sure buttons were large enough to account for any variation in movement. The size of elements can also be considered relative to the offsets as they were defined. A menu appearing closer to the user may need smaller components and sizing. The transparency of the menu allowed for continued minor see-through of the menu to observe any events occurring in the background (e.g. observing the movement of enemies in the tower defence games). The dead space between buttons is not significant except in the middle of the menu when considering the addition of the two-step widget as part of the third experiment. When the two-step widget transitions to the menu, it is centrally placed relative to the widget and puts the cursor into the dead space between the four buttons. The user can rapidly move either up or down to select an option, use the close button, or let the menu hide automatically. The layout of the menu options was kept the same for both experiments, but it would have been suitable to test an alternative circular layout with all menu options equidistant. This menu type was used in contrast to the PVMS in the third experiment, with selections of already created objects. It was suitable to keep the menu layout consistent with letting the third experiment focus on evaluating the gesture, improvements to user error handling with the two-step widget, and the close button. Table 4.5 presents the settings used for defining the PVMS menu's visual elements.

Table 4.5: PVMS Menu Visual Properties

Property	Value
Canvas Size	Width: 200px Height: 250px
Canvas Transparency	Background Image with RGBA: 255, 255, 255, 100
Element Size (Text and Buttons)	Width: 160px Height: 30px
Title Properties	Font: Arial, Style: Bold, Size: 16, Alignment: Left, Colour: Black
Button Text Properties	Font: Arial, Style: Normal, Size: 14, Alignment: Centre, Colour: Black
Button Colour Properties	Background: White, Border: Black (1px)
Close Button Size	Width: 30px, Height: 30px
Close Button Text Properties	Font: Arial, Style: Bold, Size: 16, Alignment: Centre, Colour: Black
Element Positions	Title: (0, 83, 0) Button A: (0, 49, 0) Button B: (0, 6, 0) Button C: (0, -41, 0) Button D: (0, -87, 0) Close Button: (86.6, 111.9, 0)

4.7 Conclusion

This chapter has provided the necessary details to understand the proposed PVMS. The details included discussing potential use cases for how it could benefit interactions in various areas of VR and AR applications. With the system's operation and construction definitions discussed, the subsequent chapters (Chapter 5 and Chapter 6) will build on this knowledge and awareness to indicate the application of the menu system. These chapters will discuss the execution of the experiments, the methodology applied, and the results gathered. The results are summarised in these chapters, with a comparative discussion on the results from all experiments in Chapter 7.

5 Second Experiment: Periphery Vision Menus in Practice

This chapter will discuss the approach, structure, execution, and results of the second of three experiments conducted across this research. The experiment was conducted between March 2016 and May 2016. The experiment took what was learned from conducting the first experiment and built on the initial ideas—improving upon the experimentation techniques focusing on providing a testing platform for the Periphery Vision Menus System (PVMS). The participant feedback received during this experiment was used to improve the system before conducting the third experiment. A paper has been published that documents the approach and findings from this experiment (Mitchell and Wilkinson, 2016).

This chapter will provide an overview of the experiment. Starting by looking at the methodology, including the recruitment methods details of each task, and a breakdown of the data collection methods used. After the methodology, a discussion of the data and the importance of the results is presented. These result summaries will be further examined in Chapter 7 by comparing results across all three experiments. For details about accessing the experiment code on GitHub, see Appendix D.

5.1 Experiment Overview

As required to carry out the experiment and validate results, the Flinders University Social and Behavioural Research Ethics Committee granted ethics approval under research project number 7103. The following points were the primary goals set out to be addressed by this experiment:

- Determine whether a PVMS is viable as a tool for interaction within this application and more broadly for other applications.
- Determine whether improvements could be made to the proposed menu system in the way it is calibrated.

These goals were investigated with four different tasks. Each task was designed with a different outcome in mind. The focus overall for this experiment was using a HMD as the only type of interaction device. When initially designing the experiment, the plan was to have the participants use three or four different input methods (HMD Only, mouse controller, mobile controller, MagicLeap controller), similar to experiment one, detailed in Chapter 3. To focus on the key validation requirements, the final version of the experiment would take 15-20 minutes of wearing the HMD with just one input method. The experiment duration led to focusing on just the one input method to provide testing of the interactions for the interface more than the tool for interacting.

For the design of the experiment, there were four separate tasks embedded within a tower defence game scenario that the participants were asked to complete. Task 1 was used to determine how participants felt about three different calibration settings for the interface interactions. Task 2 had the participant construct their personalised towers. Task 3 took the personalised towers for use in the tower defence game. Task 4 provided an alternate representation of the interface to gauge reactions from participants. A pre-experiment questionnaire queried prior experience and perceptions, and the post-experiment questionnaire explored the participant's experience while conducting the experiment. Further details about the data collection methods can be found at the end of the methodology section (5.2) about the questionnaires and the data collected by the application.

Tower defence games incorporate several core elements to create the commonly established game experience. The genre typically has the player defending one or more positions and will lose if the protected location is destroyed. Enemies spawn in sequenced waves to assault the player while following a path to the defended base and damage the base on arrival. Enemies in this game are represented as spider-like creatures with differences, including health, speed, and if they can retaliate against towers. The player controls creating towers to place at defined locations along the path enemy units traverse. Towers are player-controlled units that can be created, moved, and repaired. These towers can have different functions such as slowing enemies, single target attacks, area of effect damage, rapid-fire or many other functions. To control how many towers a player can control at any time, the player has to manage currency, with towers each having a cost dependent on their utility. As is the case with these experiments, the player also sometimes needs to repair them as they are damaged either by enemies or general wear and tear. The player is awarded currency based on progress through the waves as both fixed amounts as entire waves end, and each enemy unit they defeat. As part of the evaluation of results, the way participants decided to spend currency, and their general performance is evaluated to consider the success of the game's deliberate decisions about challenge and evaluation of the PVMS using a serious games approach. The specifics of the implementation for this experiment and the third experiment are described in the associated task sections.

5.2 Methodology

This methodology section will cover the important features of how the experiment was run. The section begins by detailing the recruitment process before looking at the experiment equipment and required software. A description of the experiment tasks and structure follows. Then the various questionnaire and survey apparatus are presented. A summary of results follows, and then an evaluation of the results related to the experiment aims.

In designing this experiment, each of the points discussed in Chapter 2, section 2.1 about the key elements for providing a VR experience were considered as defined by Sherman and Craig (2002). These elements were: virtual world, immersion, sensory feedback, and interactivity. The virtual world is implied from using HMDs. Immersion was a focus for using the PVMS to keep the user focused on the elements they were interacting with and allow them to access the menu when required. Sensory feedback was provided with a natural action of turning the head to reveal a periphery menu and the visual and auditory feedback provided by the game mechanics and interactions. Finally, interactivity was provided by direct feedback from the menu to change how the world acts. The design of the menus to be presented as smaller points of interaction in the virtual environment drew on concepts from the domain of mobile interface development (Hoober and Berkman, 2011). In mobile development, there are similar hurdles where development constrains what is possible in a smaller context. The concepts of making the interaction process easy, flexible, reactive, relevant, and reliable were considered (Olsson et al., 2013). The approach for evaluating the interactions was based on an intent to use serious games as a medium for motivation and engagement to draw participants in. Considering the taxonomy of serious games presented by Laamarti et al., this experiment could be either classified as a combination of education and training or its own research type (Laamarti et al., 2014).

5.2.1 Recruitment

This second experiment was approved by the Social and Behavioural Research Ethics Committee under project number 7103. Following this approval, the experiment recruitment began in March 2016 and concluded in May 2016. Much of the fundamental approach to recruitment of participants was conducted in the same way as the first experiment. The main deviation for this experiment was the use of a promotional video. The video and the role it played will be discussed later in this section. The participants were provided with no monetary compensation, as was the case with the first experiment.

Two rounds of emails were sent out to advertise the experiment to students in Computer Science, Engineering, and Mathematics at Flinders University. The principal researcher attended a selection of university lectures to advertise the experiment, research project and how students could volunteer to participate. During the spiel given at lectures, the promotional video was shown. Interested individuals were asked to make contact via email to express their interest.

Once a potential participant had made contact via email to express their interest, a brief, formal process was completed. A response to their interest was sent back to thank them for their interest and provide additional information. At this time, potential participants were sent three documents:

- A letter of introduction: introducing the project and researcher from the supervisor.
- An information sheet: giving brief details explaining what would occur during the experiment.
- A consent form: to show what they would be consenting to by participating. The form established the participant was of appropriate age (17+), understood what the experiment would require them to do, indicated they would not directly benefit from the research, indicated they were free to withdraw at any time and confirmed they would be deidentified to remain confidential in any publications.

In addition to these documents providing additional information, a link to a Google Sheet was provided. This sheet provided a simple calendar view showing the times available, filled, or unavailable. Anyone interested in participating after this additional information had been sent could reply and request any available time. The dynamic calendar provided a straightforward way for students to quickly see when they could fit the research around their studies.

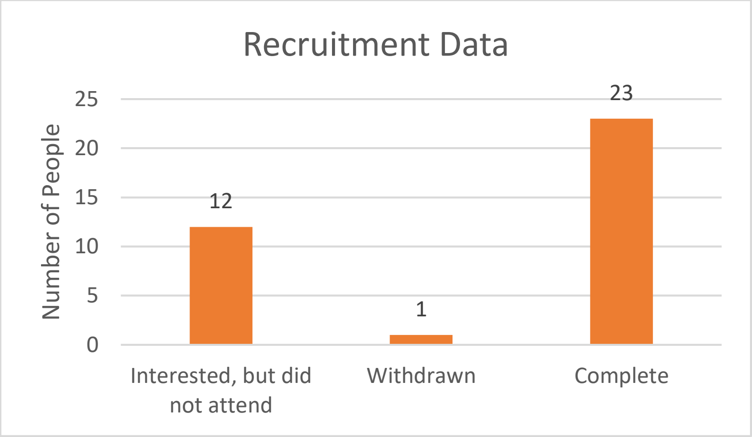


Figure 5.1: Experiment 2 Participation Quantities

In Figure 5.1, the number of people who interacted via email can be seen. A total of 36 students made contact via email. While this was less than the total of 41 from the first experiment, more showed interest and then went on to complete the second experiment. 12 people who were sent the additional information never followed up to request a time. 23 participants completed participation in the experiment, and 1 student withdrew from participation due to difficulties with interaction while not wearing their prescription glasses. Like the first experiment, vision issues were left to the participants to decide if they were capable, and the HMD was not optimised with IPD for each participant.

The following section will discuss the promotional YouTube Video concerning the storyboard, followed by some of the data collected from viewing statistics provided by YouTube.

YouTube Trailer

Images showing a storyboard for the trailer can be found in B.2.1 as Figure B.8. A narrative approach was used to generate interest in the research and to try and increase participation. The video started by introducing the researcher and the purpose of the project. The video led into a premise of a standard tower defence game, the player's home was under attack by enemy forces. The village was symbolised by the single large house on the game map, which, through camera motion, was zoomed into view. The method of defence is introduced by saying, "Designs for the towers have been passed down for generations…", followed by a panning shot of the towers used in the game. A customisation example was shown with a quick sequence of coloured blocks stacking together to finally resolve into the (blue) tower to show that these towers involved more interaction than placement. The game footage finished with a panning shot over an active battle sequence with a wave of enemies spawning and the towers firing projectiles at them. The video concluded with a fade to black wipe followed by a final text message to indicate the participants time and assistance would be appreciated.

The goal of the trailer was to provide a straightforward introduction to the game aspect of the research. The video was created using Windows Movie Maker and some automated camera movements within Unity. Segments of the video were given roughly equal amounts of time except for the scene panning across the active battle so that the most exciting scene would have the longest run time. Having the text stand alone as separate segments of time did extend the length of the video. The video sequence has been described in full but can be viewed on YouTube at:

https://www.youtube.com/watch?v=7R2cSl9IyD8.

YouTube Watch Data

YouTube provides a lot of tools[35] for helping content creators to observe how their content is being viewed. This allowed collection of some additional data related directly to recruitment. This data is included to show how much interaction there was from recruitment by showing the number of views and how many of the views translated into actual participants. The trailer was shown to multiple people simultaneously when promoting the content in situations such as lectures where it was advertised. The following list details some of the key metrics of interest that were collected.

- Duration: 1 minute 4 seconds
- Total Views: 129
- Total Watch Time: 87 minutes
- Average View Duration: 40 seconds
- Average Percentage Viewed: 63%
- Viewing by Device:

[35] Tool provided by YouTube for Analysing Video Watch Data: "YouTube: Measure Audience Retention", URL: https://support.google.com/youtube/answer/9314415?hl=en, Last accessed 11/12/2021.

- Computer: 67 minutes
- Mobile: 19 minutes
- Tablet: 1 minute

- Viewing by Gender: 99.3% male, 0.7% female.
- Viewing by Age:
  - 18-24 years: 48%
  - 25-34 years: 45%
  - 35-44 years: 4.7%
  - 65+ years: 2.2%

The average view duration ends right after the tower customisation sequence, around 40 seconds. With this average time, it means viewers, on average, did not see the most active sequence where the combat was occurring.

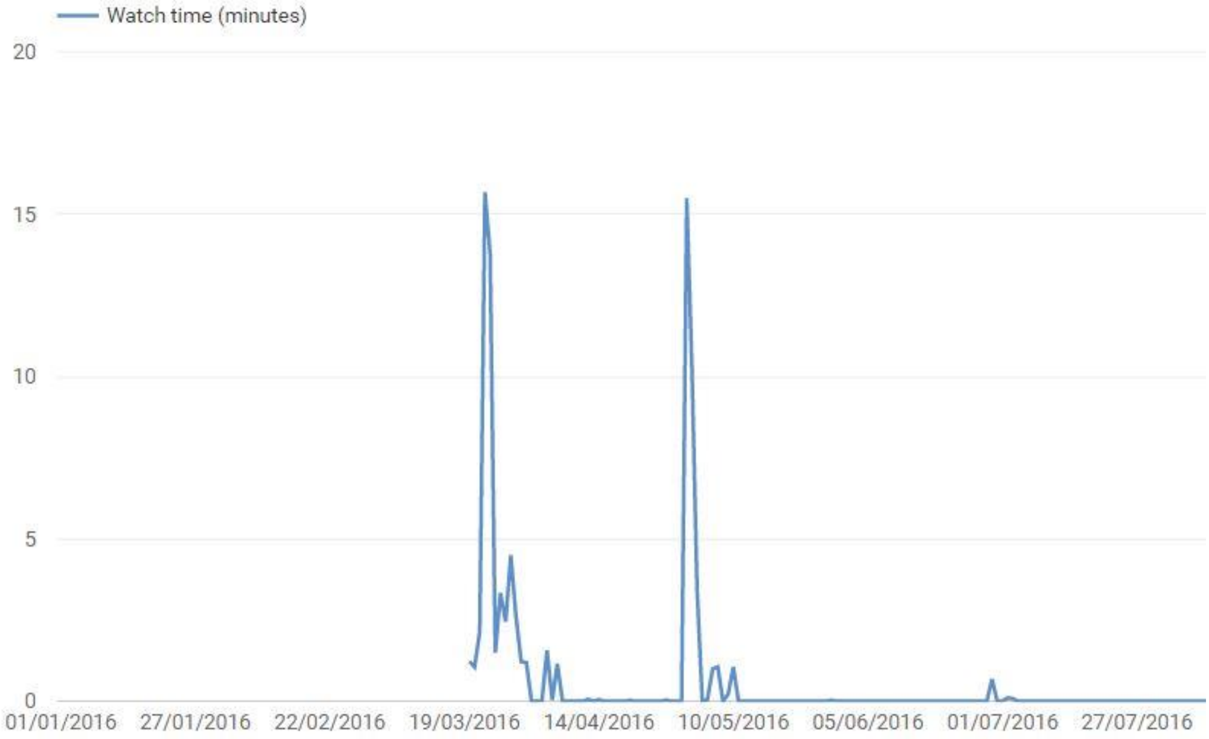


Figure 5.2: Video Watch Time by Date

In Figure 5.2, the watch time distribution is clearly shown from the graph provided by YouTube. The two peaks indicate when emails were sent out. With a rush of initial views each time, followed by a smaller flow of later viewers who viewed the email link at a later time.

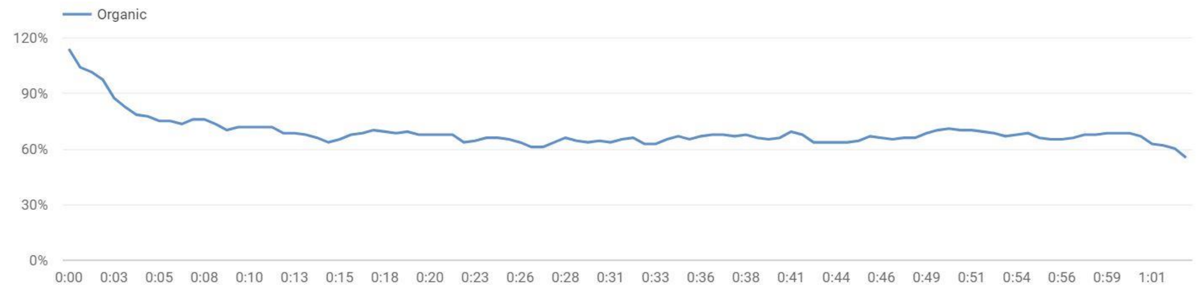


Figure 5.3: Video Absolute Audience Retention

The graphs, as seen in Figure 5.3 and Figure 5.4, were also provided by the YouTube data. These show the audience retention concerning time with two different views. Figure 5.3 shows the absolute audience retention. The figure represents a percentage of viewers who continued to watch, showing the percentage of viewers who made it to a time code. From observation, this shows more accurately what the 63% average viewing time means. Viewers who watched past the first 5 seconds were far more likely to watch till the final text. Retention is represented in Figure 5.4, showing the increasing audience retention after passing 5 seconds.

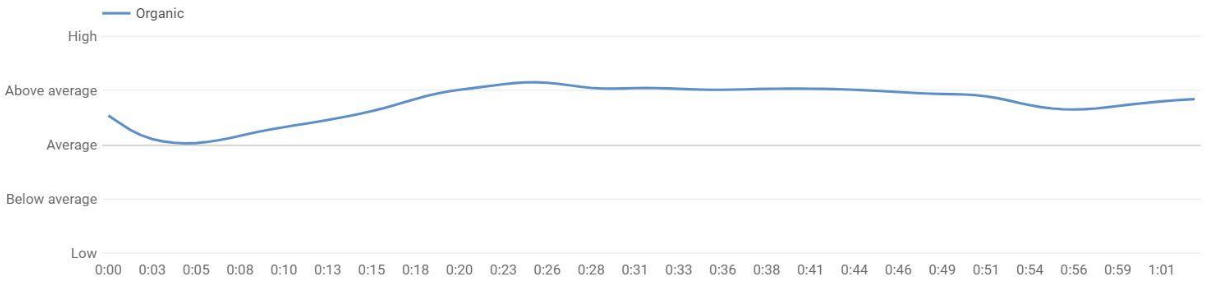


Figure 5.4: Video Relative Audience Retention

5.2.2 Hardware and Software APIs

This experiment focused on using just the Oculus Rift Development Kit 2 as a HMD for interactions. The experiences of developing the first experiment were used to iterate and augment the experience for testing the PVMS. Initially, there was a comparison planned against other input devices within this experiment, but these were excluded as the development reached a final stage. The decision to exclude additional inputs was to reduce the duration of participant time and focus on the experiment's core evaluation of the PVMS. Pilot sessions determined this experiment would take 20 to 25 minutes on average, including the questionnaires. The following list summarises the hardware and software APIs used as part of developing the experiment. Appendix B.1.5 details the 3rd party art assets incorporated to aid in the production of the experiment.

- Desktop Computer
- Oculus Rift Dev Kit 2
- Unity version 5.3.2

Desktop Computer

The decision to use a desktop computer over a laptop for this experiment mostly came down to the ability to run multiple participants simultaneously if necessary. The access to multiple computers capable of running Oculus Rift Dev Kit 2 devices meant the experiment could be standardised in a quick way to set up. The specifications of the desktop computers were as follows.

- CPU: Intel Core i7-5820K 3.3GHz 6 cores
- Motherboard: Alienware Area-51 R2 0XJKKD-A01
- RAM: 8GB
- GPU: NVIDIA GeForce GTX 960
- OS: Windows 10 Pro

Oculus Rift Dev Kit 2

The Oculus Rift Dev Kit 2 seen in Figure 5.5 provided an easier configuration experience to the Oculus Rift Dev Kit 1. No additional, separate runtime applications had to be loaded, making the device far more of a plug-and-play experience. The device also used an improved screen with a 1920x1080 resolution (960x1080 per eye), among other features. At the time of the experiment, the HTC Vive as a primary contender was not released yet (released 5th of April 2016). The integration with Unity was also significantly improved with this release.

Figure removed due to
copyright restrictions.
See link for original:
[Link]

Figure 5.5: Oculus Rift Dev Kit 2

Unity version 5.3.2

One of the recent updates to Unity at the time of development for this experiment was the addition of support for HMDs. As were used for the first experiment, the previous version required importing the libraries and associated objects to support these interactions. Instead, with this version, a simple check box was added to the game settings that could be used to toggle between a standard camera type and one for a HMD type device.

5.2.3 Experiment Tasks

The tasks included in this experiment were each designed with specific individual purposes to test an aspect of using the PVMS. All were designed with the novice user in mind. The tasks were presented to all participants in the same order and accessible to someone with limited VR or gaming experience. Instructions were given before the user commenced each task to mitigate the learning curve required to complete the tasks. The in-application instructions allowed participants to cycle through text and visual instructions to see what would occur before attempting the process themselves. After each task was complete, the participants were also asked one or more questions in the application. The questions allowed for an immediate response based on their current perception, removing any recall issues or obfuscation. These questions are identified for each task as part of the task-specific sections. The following is a summary list introducing the required activities for each task involved and how they fit into the experiment. More detailed descriptions of each task follow later in their respective sections.

- **Task 1 Calibration**: (presented in section 5.2.6) This task aimed to test three different activation configurations of the PVMS. These were referred to as low, medium, and high sensitivities. The sensitivities were randomised into a set of 24 steps alternating between the different sensitivities to complete a task. The name for this task is a little deceptive as it did not calibrate for the current experiment; its purpose was to capture data for future review. The medium sensitivity was used for the remaining tasks to standardise everything. From development testing before use by participants, the medium sensitivity appeared a safe choice for general users. The data from this task could be used to look at the usability of different configurations for future tool iterations.

- **Task 2 Tower Construction**: (presented in section 5.2.7) This task provided a scenario with no time pressure to complete the task. The task was focused on providing an experience where the participant would spawn in objects and place them in a stack. The stacks represented augmentations of the attributes of the towers. These customised towers were then used in Task 3.
- **Task 3 Tower Defence Game**: (presented in section 5.2.8) The tower defence game was designed to be simple for novice users while creating a situation where there would be perceived time pressure. The game used two different forms of menus—one menu to spawn in new towers and one for interacting with existing towers. By using the two different menus, the participant would control the game. With the use of four different towers, the participant could spawn them based on different costs. The goal for the participant, as with any typical tower defence game, was the minimise the number of enemy units reaching the player's "home" after passing through a maze of towers.
- **Task 4 Periphery Context Preview**: (presented in section 5.2.9) This task provided a visual question to the participants. The technique presented in this task demonstrated a primitive representation of a different idea for how the menu system could be further improved.

Task Design Philosophy

Each of the tasks evaluated a distinctly different part of experiencing the PVMS. The following list further summarises the decisions for why tasks were selected for inclusion in the experiment design.

- **Task 1**: The task did not perform an actual calibration as mentioned in the prior list. Calibration would typically be the process of either training based on some data or directly using settings to approximate the best calibration. The PVMS does not need training data but has settings that can be tuned to improve a user's experience. Therefore, what is being tested as part of the first task is comparing three different pre-selected configurations that all provided suitable parameters for interaction, but where the medium sensitivity was expected to be close to ideal. The task completely randomised the order of sensitivities, left/right, and whether the participant had to reveal a menu or perform a turn without revealing a menu. Randomising the order of these operations with additional repeats for a total of 24 actions gave a suitable collection of data to validate the pre-selected calibrations.

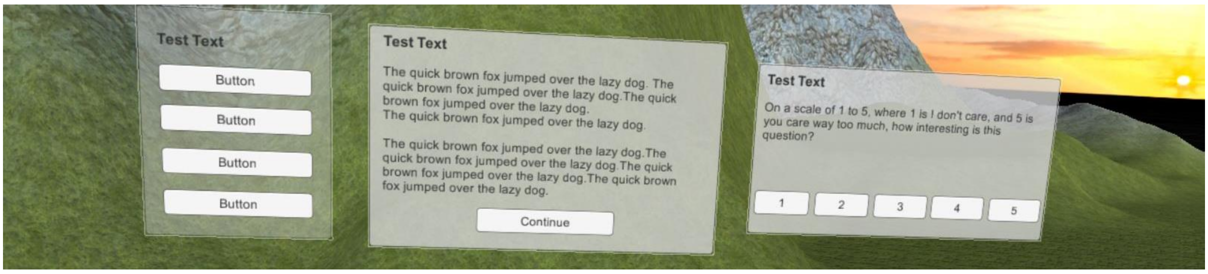
- **Task 2**: The task, as mentioned previously, was intended to give participants an untimed situation to experience the PVMS that was more directed than Task 1. Untimed here refers to the lack of pressure from an urgency to perform an action. The task has been referred to as a "Tower Construction" task. The implication here is that the construction is comparable to assembly, creation, and combining components. The focus was directed on the menu use and not the complexity of the object creation. The simplistic stacking of coloured contextual blocks could be similar to building something out of LEGO. The objects could be swapped for more domain-specific applications to create suitable objects for any creation experience. The block choices changed the properties of towers in Task 3 to give the participants a direction for object choice.
- **Task 3**: The task presented a Tower Defence game. In contrast to Task 2, this task was designed to show a time-pressured situation where the participant would need to quickly make decisions about what towers to create, where to place the towers, how to spend their currency, and adjust based on unknown numbers of enemies. The task was intended as a significant point for recruitment with the trailer. The design of the task enabled participants who used a balanced strategy to succeed while experiencing a substantial amount of PVMS interaction.
- **Task 4**: The final task was less a sequenced task but a question to evaluate thoughts about the earlier design for the PVMS as described in section 4.1. The purpose was to gauge the reaction of participants to the additional overlay.

The tasks are discussed with implementation details in later sections. Before further discussing the specifics of the tasks, the different interfaces and tutorials are presented as they are used across all tasks.

5.2.4 Interface Element Types

As part of this experiment, a set of dialogs were designed for simple generic use between all different tasks. The standardisation of dialogs helped create an overall theme within the application and some consistency within the experiment. For all interfaces used while testing the PVMS, the Unity interfaces system was used. Each interface consisted of custom-configured objects, including panels, labels, and buttons. Through visual simplicity, the interaction itself could be focused on during experimentation.

There were three different types of interaction dialogs created for this experiment. These are shown in Figure 5.6. On the left is the PVMS interface, providing a place for the title and four menu button options. This was only visible for interaction through PVMS interactions and in no other place. The middle type of dialog was used for providing information to the participant, including space for a title, a moderate amount of text, and a continue button. This middle dialog was primarily used to provide the introduction text before each task. The dialog on the right is the question dialog. It presented a question to the participant with a title, question text, and options 1 to 5 allowing the participant to respond.



The quick brown fox jumped over the lazy dog. The quick brown fox jumped over the lazy dog.The quick brown fox jumped over the lazy dog.
The quick brown fox jumped over the lazy dog.

The quick brown fox jumped over the lazy dog.The quick brown fox jumped over the lazy dog.The quick brown fox jumped over the lazy dog.The quick brown fox jumped over the lazy dog.

On a scale of 1 to 5, where 1 is I don't care, and 5 is you care way too much, how interesting is this question?

Figure 5.6: Template Interface Dialogs

Some of the other dialogs used in the experiment were purely for providing a form of information to the participant and did not allow for any specific user interaction. Figure 5.7 shows the three different dialogs displayed for the tower defence task. The details of the information represented on them will be explained as part of the tower defence game section, section 5.2.8. This figure shows how the information important to the participant could be clearly shown during the tower defence game.

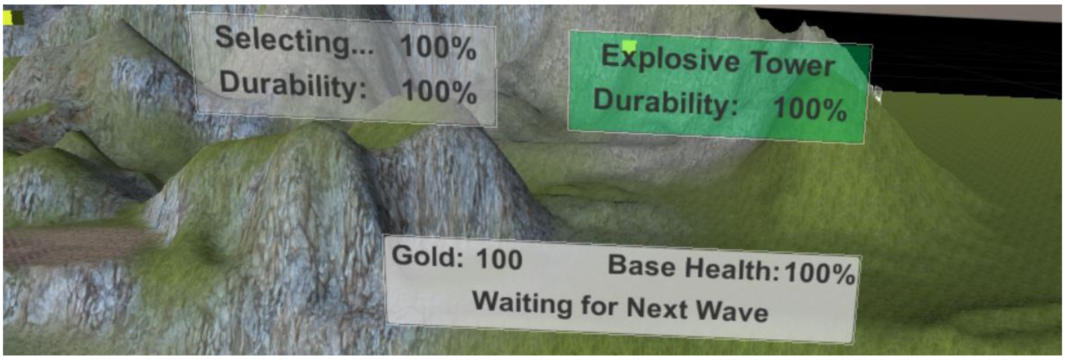


Figure 5.7: Tower Defence Interface Dialogs

The biggest improvement to the interface elements of the project was the implementation of the cursor. During the first experiment, the cursor was larger and always remained the same tone of orange regardless of context. For this experiment, the cursor was shrunk to reduce the amount of screen space it occupied. The colour was also set up to have a gradient. In Figure 5.8, the transition of colours can be seen. On the left is the cursor with no current selection target. Then as the cursor begins to hover over a button, it begins to transition in colour until it reaches the green shown on the right. This colour change was used to represent the selection time visibly using the cursor. The colour change was calculated using a Lerp between the orange and green colours with selection progress as a value between 0 and 1. Much like the first experiment, the selection was through a hover-to-select mechanic.

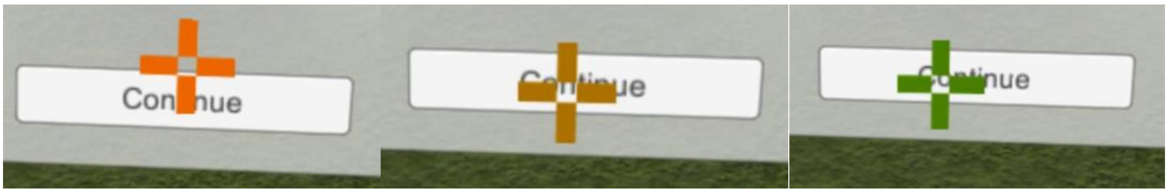
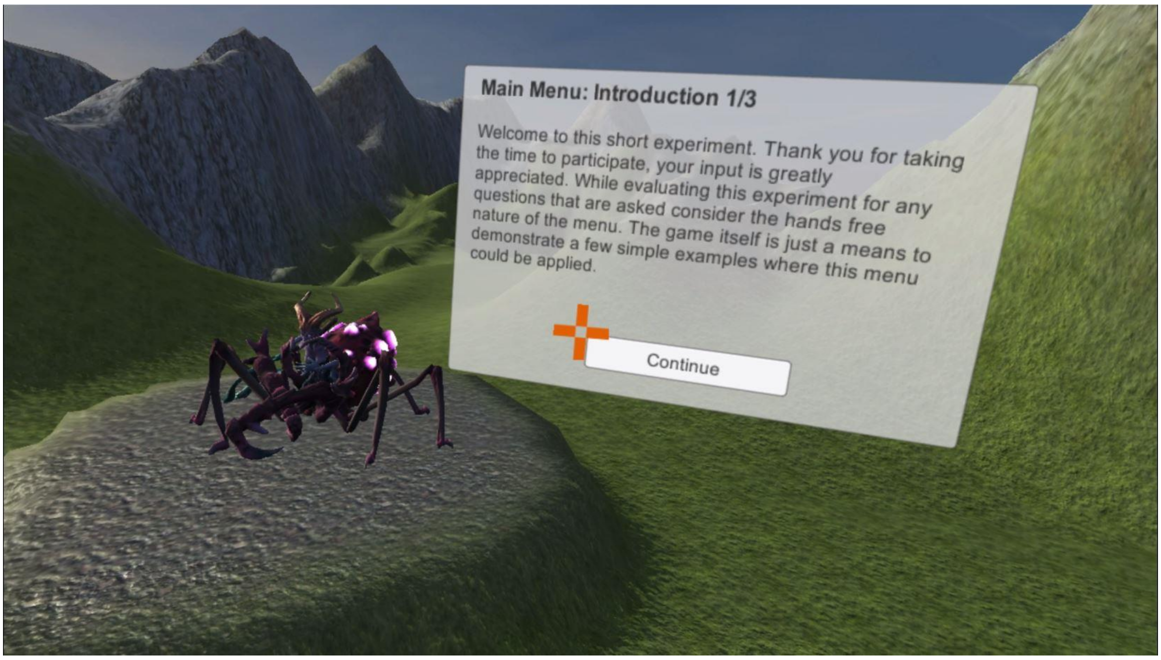


Figure 5.8: Cursor Colour Change

5.2.5 Use of In Application Tutorials

While running the first experiment, some participants asked how to perform parts of the required tasks. For the first experiment, participants were given limited instructions. The limited instructions were not a burden because the tasks were completed three times to use each input device. When it came to this second experiment, the tasks were only to be completed once. The single experience of the tasks meant it made sense to explain to participants what to expect during the play session without showing them a perfect way to complete the tasks that could potentially skew results. The solution to this problem was to include information dialogs in the virtual environment, explaining what the participant needed to know as they continued.

Each of the experiment sections following will include their associated introduction dialogs. The ones included here do not fit with any others because they introduced the beginning of the experiment. (over) shows the first dialog a participant would see when they entered the experiment with one of the enemy characters animated next to the dialog. Once the participant selected the Continue button, the dialog would continue to the next screen. In all cases, feedback was provided to the user by displaying the number shown in the title to indicate how many dialogs would appear before there was something to do.

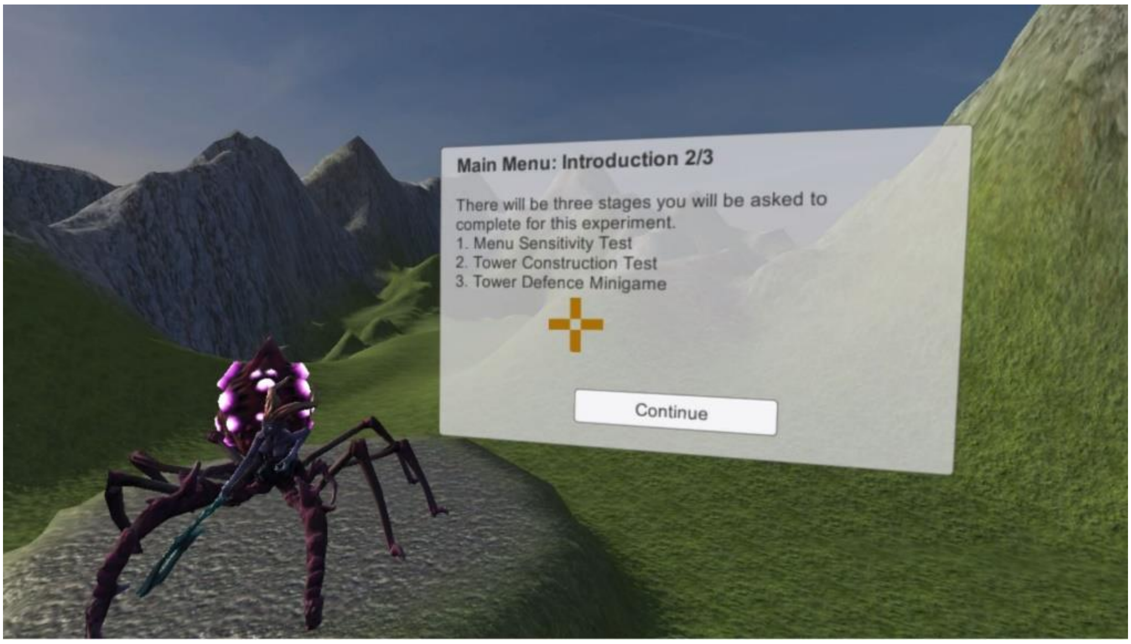


Main Menu: Introduction 1/3

Welcome to this short experiment. Thank you for taking the time to participate, your input is greatly appreciated. While evaluating this experiment for any questions that are asked consider the hands free nature of the menu. The game itself is just a means to demonstrate a few simple examples where this menu could be applied.

Figure 5.9: Main Menu Introduction Screen 1

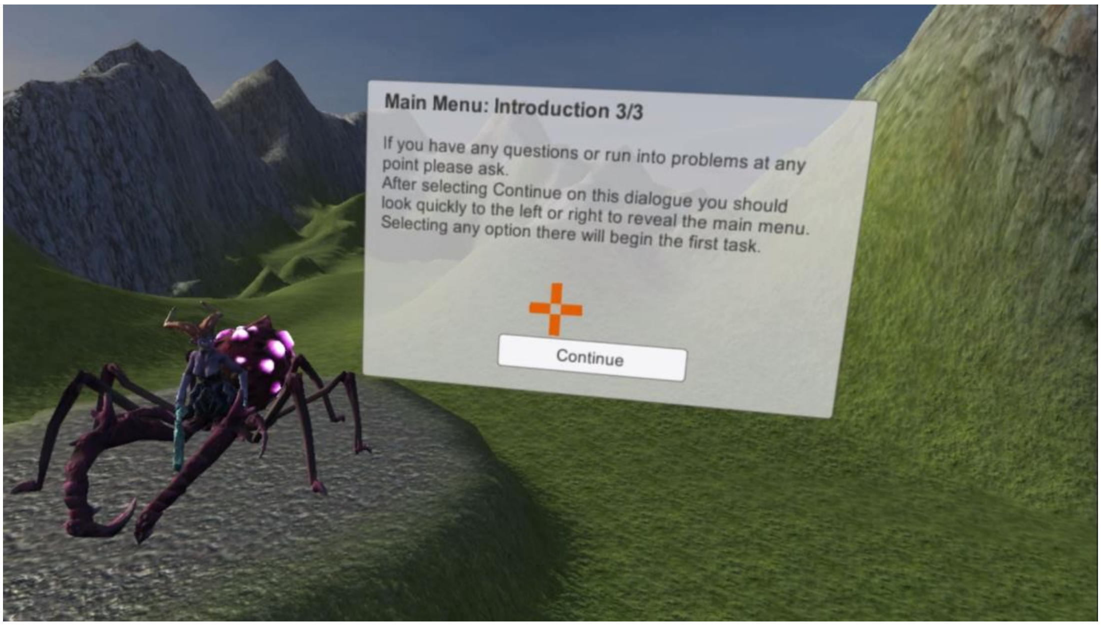
In Figure 5.10 and Figure 5.11 (over), the other two dialogs are shown for the main menu introduction. One notable exclusion from Figure 5.10 is Task 4, partly due to only being added late in development. Also, because the task only required observation and not completing a series of goals like the others.



Main Menu: Introduction 2/3

There will be three stages you will be asked to complete for this experiment.
1. Menu Sensitivity Test
2. Tower Construction Test
3. Tower Defence Minigame

Figure 5.10: Main Menu Introduction Screen 2



Main Menu: Introduction 3/3

If you have any questions or run into problems at any point please ask.
After selecting Continue on this dialogue you should look quickly to the left or right to reveal the main menu.
Selecting any option there will begin the first task.

Figure 5.11: Main Menu Introduction Screen 3

After hitting Continue in Figure 5.11, the participant was left with no instruction window, and they would have to follow the last instruction shown, performing the reveal interaction for the PVMS. Completing this interaction would initiate the appearance of the main menu, seen in Figure 5.12. This dialog existed less as any form of actual main menu and more to introduce participants to how the interaction would be performed before using it for the following task. All the menu options were named the same and performed the same action to change the game state to Task 1.

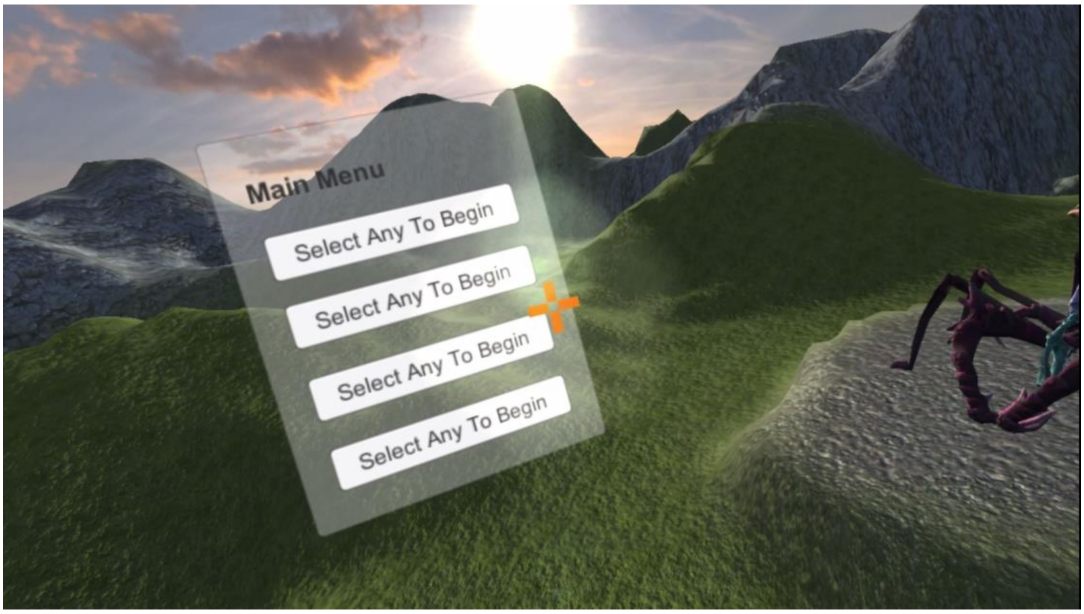
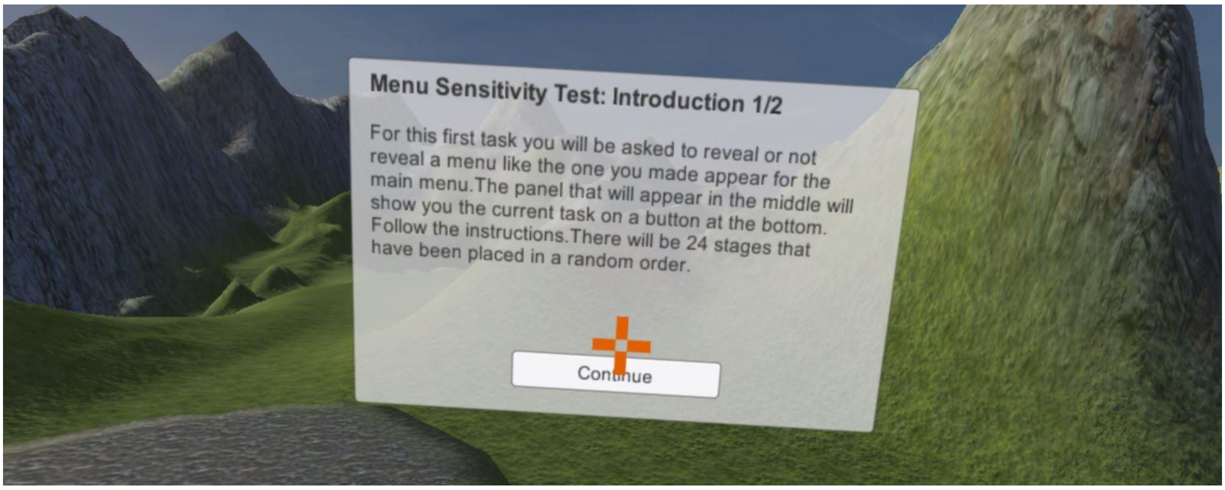


Figure 5.12: Main Menu via Periphery Menu

5.2.6 Task 1: Calibration

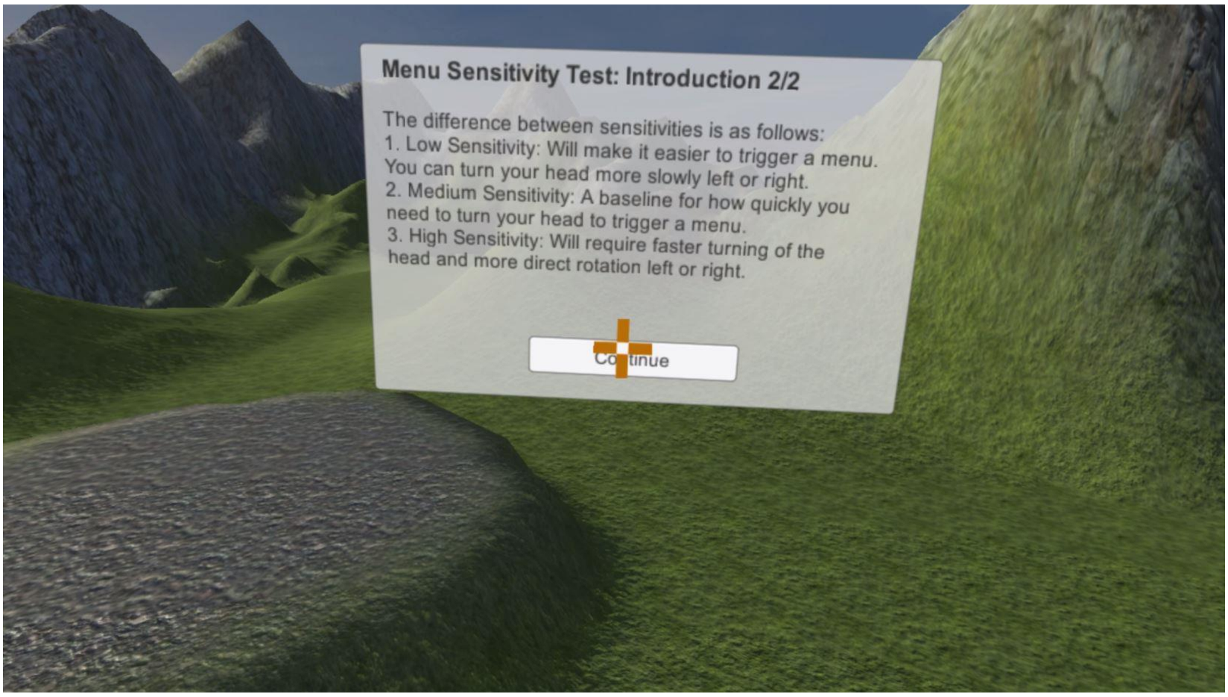


Menu Sensitivity Test: Introduction 1/2

For this first task you will be asked to reveal or not reveal a menu like the one you made appear for the main menu.The panel that will appear in the middle will show you the current task on a button at the bottom. Follow the instructions.There will be 24 stages that have been placed in a random order.

Figure 5.13: Calibration Task Introduction 1/2

The calibration task was referred to as a "Menu Sensitivity Test" to the participant, mainly because the calibration would not impact the later tasks of the current experiment and collect data for evaluation only. The importance of this task was to test three slightly different configurations of the PVMS. A low, medium, and high sensitivity. The task was explained to participants as seen in Figure 5.13 and Figure 5.14, describing the task in general and expectations for the sensitivities.



Menu Sensitivity Test: Introduction 2/2

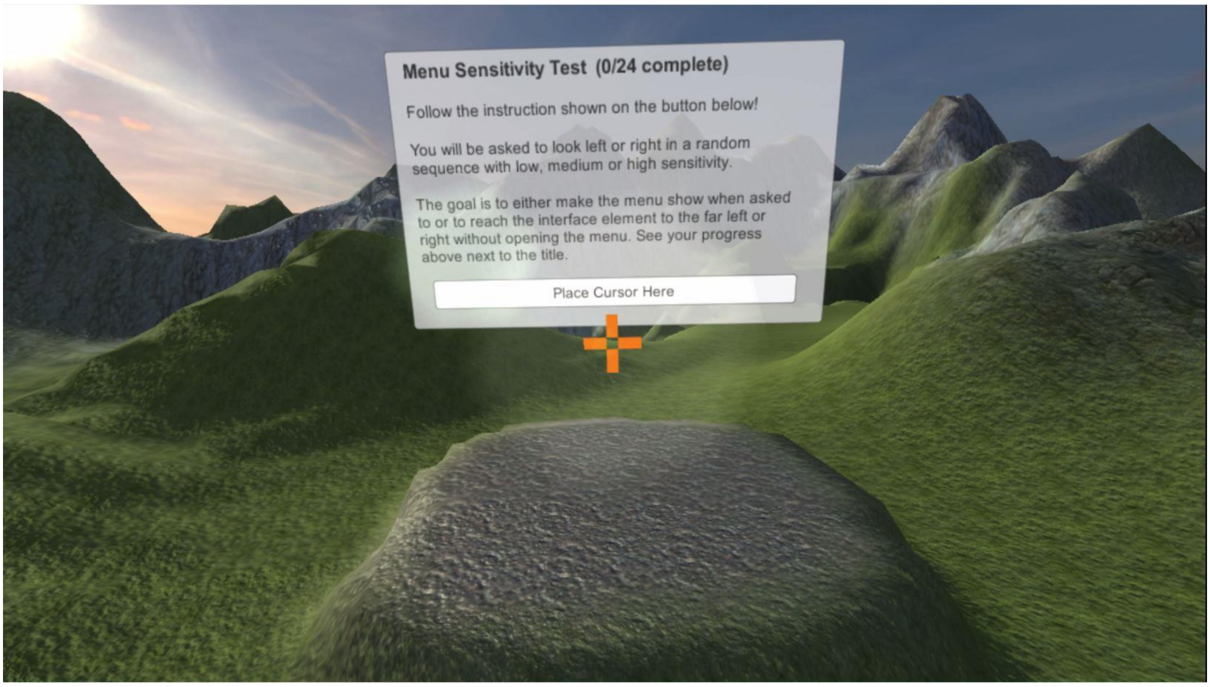
The difference between sensitivities is as follows:
1. Low Sensitivity: Will make it easier to trigger a menu. You can turn your head more slowly left or right.
2. Medium Sensitivity: A baseline for how quickly you need to turn your head to trigger a menu.
3. High Sensitivity: Will require faster turning of the head and more direct rotation left or right.

Figure 5.14: Calibration Task Introduction 2/2

The sensitivities were defined with the properties seen in Table 5.1. The primary difference used for this experiment was a change in time and a slightly more restrictive setting for high. The discussion about what these values mean is presented in Chapter 4. The researcher found the medium settings to "feel good" to use from testing leading up to this experiment. Their feeling good led to them being used as a baseline configuration, with minor changes to increase and decrease the thresholds to test against two different configurations. The main change for the low sensitivity was a slightly longer time to reach the threshold angle, meaning you could turn your head slower and still trigger the menu. The high sensitivity setting increased the turning speed of the head and made the rotation of the head in other directions stricter.

Table 5.1: PVMS Sensitivity Configurations

Name	`ThresholdAngle`	`ThresholdZAngle`	`ThresholdOtherAngle`	`ThresholdTime`
Low	25	30	30	0.35f
Medium	25	30	30	0.3f
High	25	25	25	0.25f



Menu Sensitivity Test (0/24 complete)

Follow the instruction shown on the button below!

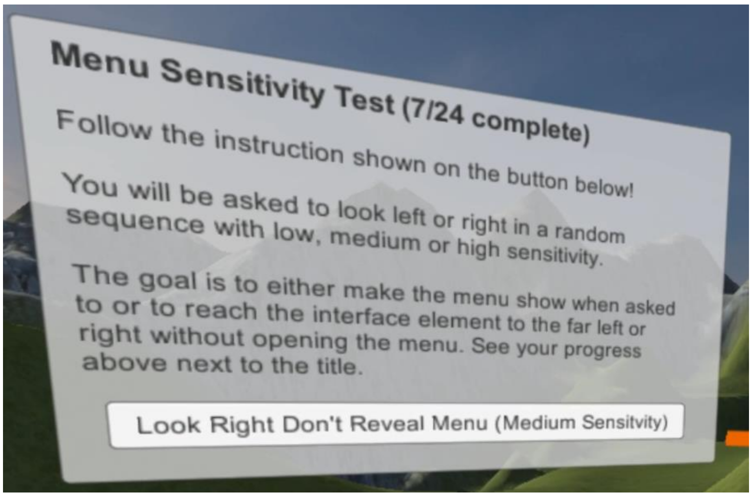
You will be asked to look left or right in a random sequence with low, medium or high sensitivity.

The goal is to either make the menu show when asked to or to reach the interface element to the far left or right without opening the menu. See your progress above next to the title.

Figure 5.15: Calibration Task Central Dialog

The central dialog shown in Figure 5.15 (previous page) was shown after the two introduction dialogs were viewed. The task instructions were shown throughout the completion of the task in case the participant wished to review what the goal was. The most important instruction was the "Follow the instruction shown on the button below!". The buttons in this task automatically updated to show relevant text related to what had to be done next. A total of 24 different steps would be completed while doing this task, as shown in the title of the central dialog.

A total of 24 tasks were equally split between the three different sensitivity configurations as already defined. For each of these sensitivity configurations, two types of interactions were completed. One was to force the PVMS to reveal, and the other was to reach the panels without triggering the menu. The two interaction types were completed twice for each side (left and right). To summarise this: 3 (sensitivities) * 2 (reveal and do not reveal) * 2 (left and right) * 2 (repeat each action twice) = 24 stages. These stages were in a randomised order for every participant.



Menu Sensitivity Test (7/24 complete)

Follow the instruction shown on the button below!

You will be asked to look left or right in a random sequence with low, medium or high sensitivity.

The goal is to either make the menu show when asked to or to reach the interface element to the far left or right without opening the menu. See your progress above next to the title.

Figure 5.16: Calibration Task Instruction Example

In Figure 5.16, an example instruction is shown. The button on this dialog would show "Place Cursor Here" if the participant were required to return to the centre to initiate the next task. The "Place Cursor Here" text can be seen in Figure 5.15. An instruction such as "Look Right Don't Reveal Menu (Medium Sensitivity)" would mean the participant would have to look far enough right till they see the dialog shown in Figure 5.17. The act of looking to the direction would be done without performing an action that would reveal the PVMS. It was accomplished by the user turning their head in a slow rotation. If the current task were to reach the Right Panel in this example, the dialog would appear like the left side of Figure 5.17. After hovering over the button, it would instantly change to show the message "Return to Centre", as shown on the right side of Figure 5.17. The interaction would cause a return to a state similar to Figure 5.15 while waiting for the next task.

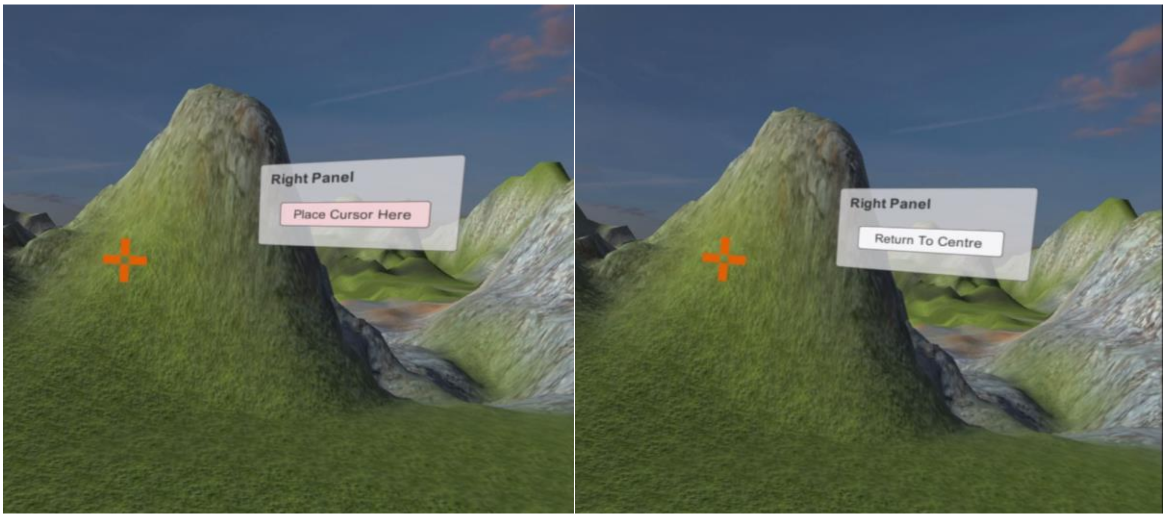


Figure 5.17: Calibration Task Right Panel Example

The other type of interaction was to perform an action such as "Look Left Reveal Menu (Low Sensitivity)". Figure 5.18 shows an example completion of this where the PVMS has been revealed to the left. Selecting any of the options on the menu would cause the menu to disappear. It was not necessary to choose any option from the menu before continuing; a participant could ignore the menu and carry out the instructions appearing on the left and right panel buttons. Most participants, however, did perform an interaction to follow the instruction of selecting any button to hide.

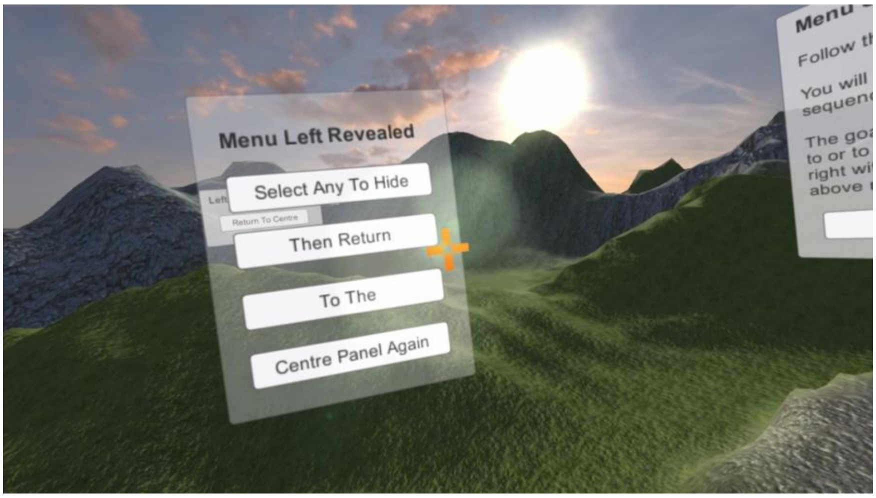
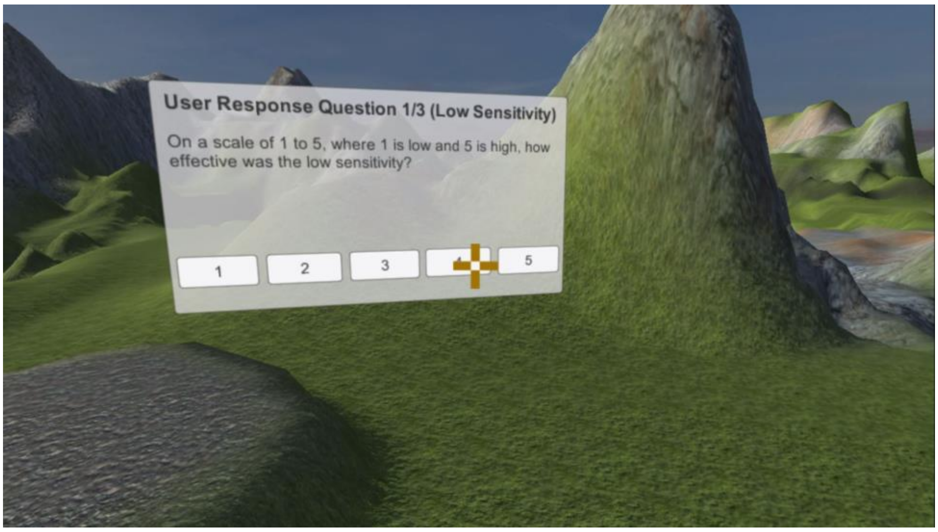


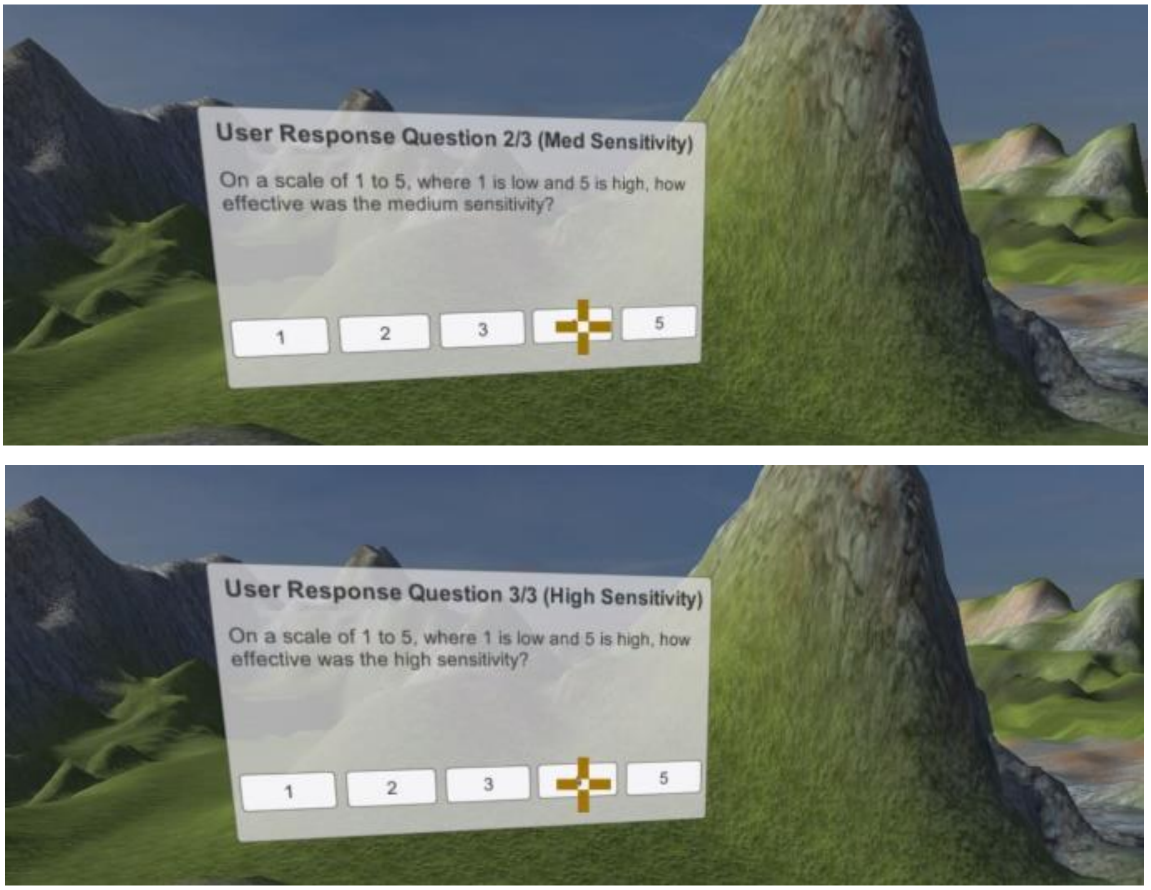
Figure 5.18: Calibration Task Menu Revealed Example


User Response Question 1/3 (Low Sensitivity)

On a scale of 1 to 5, where 1 is low and 5 is high, how effective was the low sensitivity?

Figure 5.19: Calibration Task User Response Question

After completing the 24 stages of Task 1, the participant would be automatically taken to the dialog shown in Figure 5.19. They would be asked to provide feedback on the low sensitivity. Ranking how effective they perceived the configuration to be—followed by the other two questions seen in Figure 5.20 concerning the medium and high sensitivities.


User Response Question 2/3 (Med Sensitivity)

On a scale of 1 to 5, where 1 is low and 5 is high, how effective was the medium sensitivity?

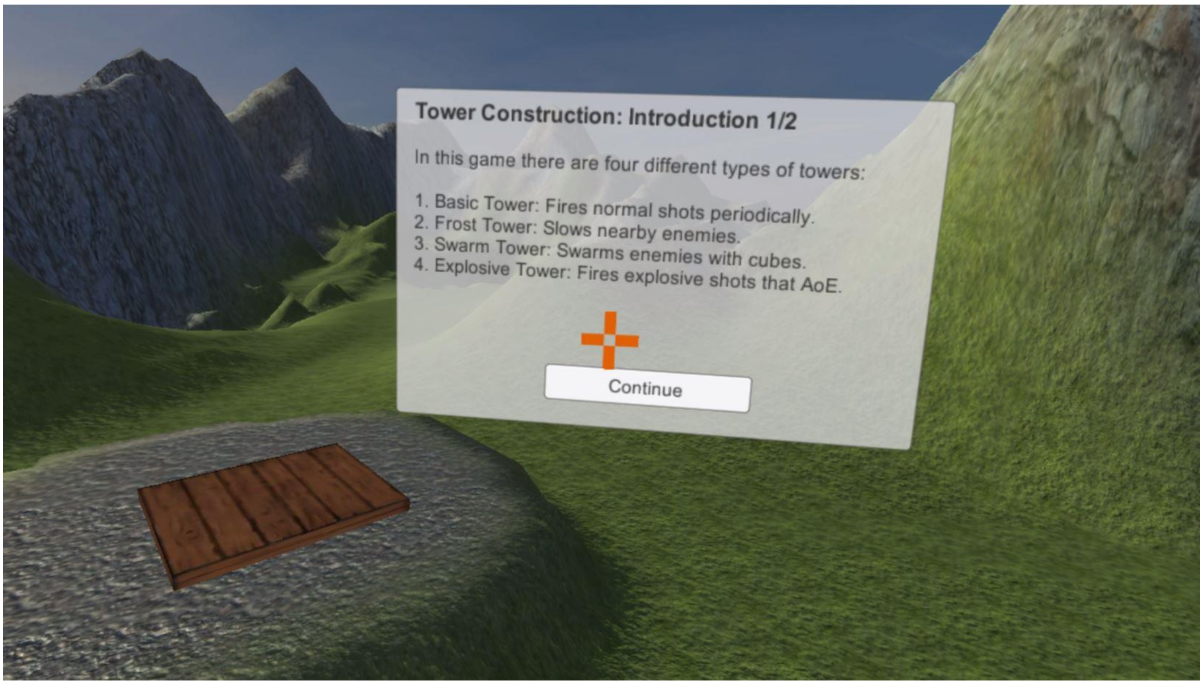
User Response Question 3/3 (High Sensitivity)

On a scale of 1 to 5, where 1 is low and 5 is high, how effective was the high sensitivity?

Figure 5.20: Calibration Task Other User Response Questions

5.2.7 Task 2: Tower Construction

The tower construction task was designed to be a simple drag-and-drop of construction task, where the participant would make decisions about how they wanted to change the towers for use in the tower defence game. There were four different types of towers, each with a unique type of projectile. The participant could modify three properties during this task. These were: damage, range, and rate of fire. Each tower consisted of 6 property modifiers that could be applied as any combination of the three property types. Damage modifiers increased projectile damage by 5%. Range modifiers increased tower range of attack by 20%. The rate of fire modifier decreased the time between spawning projectiles by 5%, causing a faster attack speed. Some participants did not understand the language used for decreasing fire rate and thought this was making the attack rate slower. Some indicated verbally to the researcher that they avoided it for this reason. Realistically the choices made during this task allowed for any combination with minimal repercussions to gameplay. During the later task, the economic decisions of tower types with strategic positioning were more important than what modifiers were used. Mathematically the fire rate would provide slightly better damage, but the repair function was more expensive in contrast. Repairs will be discussed in section 5.2.8.



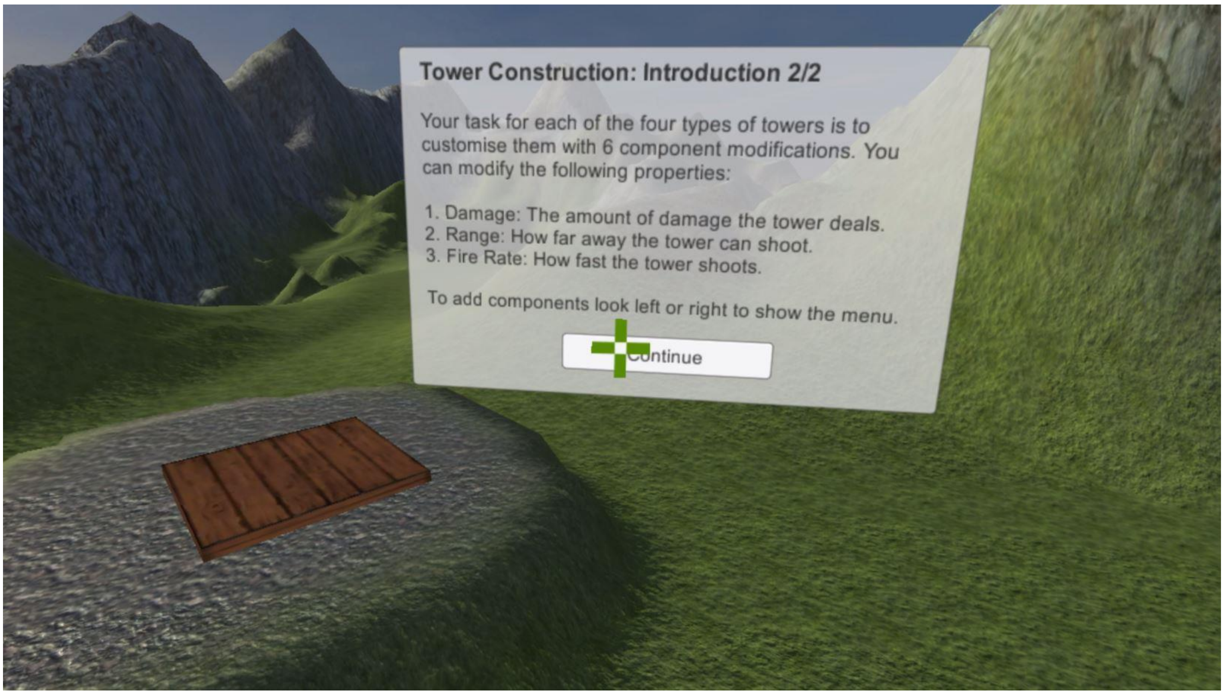
Tower Construction: Introduction 1/2

In this game there are four different types of towers:

1. Basic Tower: Fires normal shots periodically.
2. Frost Tower: Slows nearby enemies.
3. Swarm Tower: Swarms enemies with cubes.
4. Explosive Tower: Fires explosive shots that AoE.

Figure 5.21: Tower Construction Task Introduction 1/2

In Figure 5.21 (previous page) and Figure 5.22, the introduction was shown as a tutorial to participants, briefly introducing each tower's unique functionality. Subsequent explanation described the types of modifiers for participant choice.



Tower Construction: Introduction 2/2

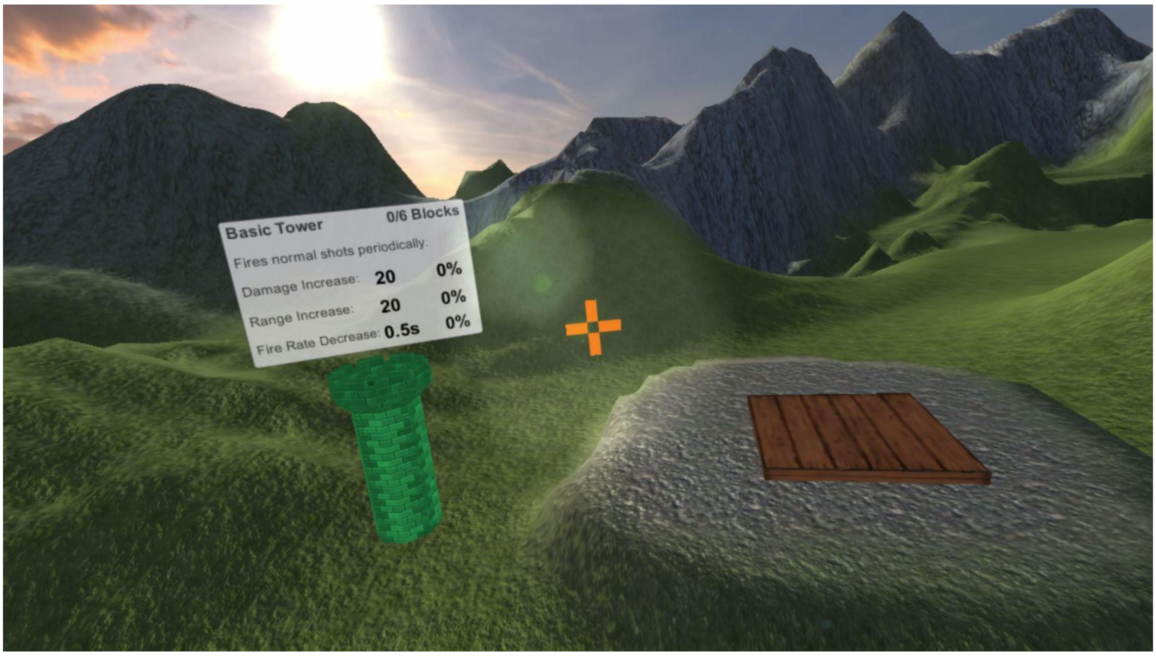
Your task for each of the four types of towers is to customise them with 6 component modifications. You can modify the following properties:

1. Damage: The amount of damage the tower deals.
2. Range: How far away the tower can shoot.
3. Fire Rate: How fast the tower shoots.

To add components look left or right to show the menu.

Figure 5.22: Tower Construction Task Introduction 2/2

Figure 5.23 shows the default state of elements in the construction task. For additional implementation details about this, see Appendix B.1.6.



Basic Tower 0/6 Blocks

Fires normal shots periodically.

Damage Increase: 20 0%

Range Increase: 20 0%

Fire Rate Decrease: 0.5s 0%

Figure 5.23: Tower Construction Task First Tower Starting View

The default values for each type of tower are shown in Table 5.2. One important difference to note is what damage meant for the frost tower. Unlike the other towers, this represented a slowing modifier percentage. From the Base Damage column in the table, the slow feature would by default apply as a 60% speed reduction on affected enemies.

Table 5.2: Default Tower Properties

Tower Name	Base Damage	Base Range	Base Rate of Fire
Basic Tower	20	20	0.5s
Frost Tower	0.6	15	1s
Swarm Tower	20	20	0.2s
Explosive Tower	35	20	0.8s

The PVMS could be triggered by looking either left or right, and either rotation would show identical menus with the same menu options. These options can be seen in Figure 5.24 (over). Selecting one of these options would attach a block to the camera that could be dropped onto the stack of blocks at the **`ConstructionSnap`**. In Figure 5.24, there are seen three different blocks already dropped. Each block colour represents a different modifier type. Originally a more detailed appearance was going to be shown to make the construction appear as more of a tower. This final appearance was used to simplify asset creation as it was deemed out of scope to create additional 3D model assets.

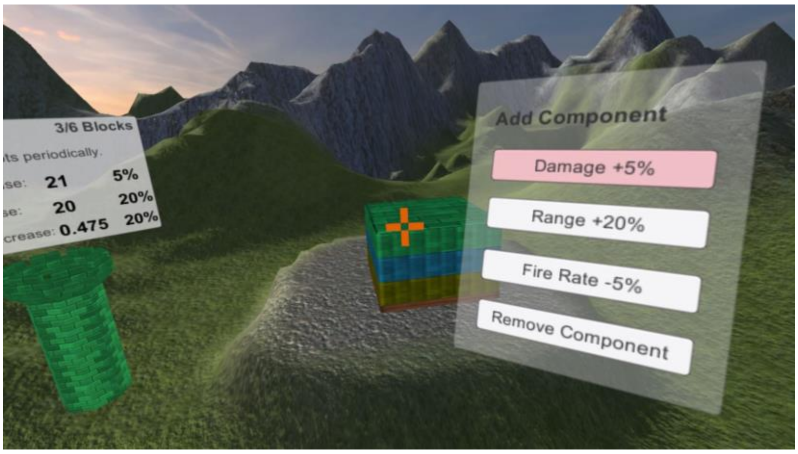


Figure 5.24: Tower Construction Task PVMS Options

The component menu also provided a remove component option. The remove component option was rarely used by participants, as they were typically happy with their first choices each time. Those who opted to use the remove component feature typically appeared to be testing to see what it did more than needing it to undo a mistake. Also seen in Figure 5.24 is the updated set of values for the dialog on the right to reflect modifiers applied. Once completed, the stack could look similar to Figure 5.25, where two of each modifier have been used.

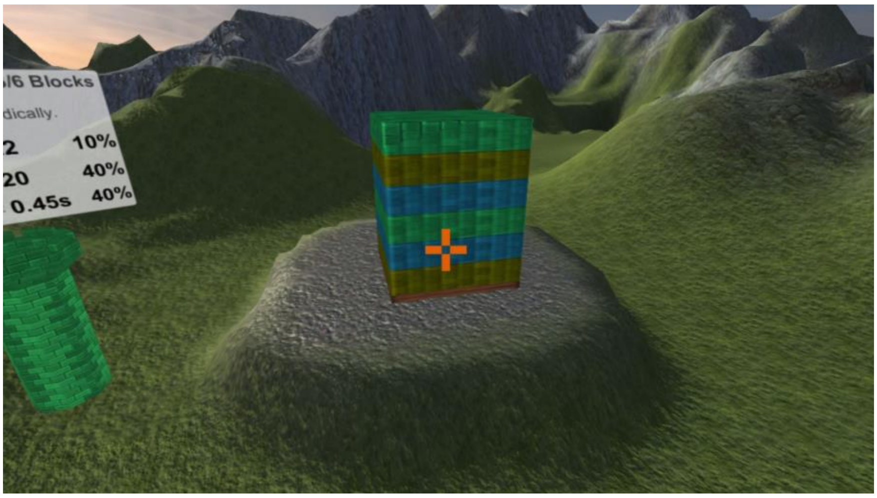


Figure 5.25: Tower Construction Task Example Completed Stack

The game used the visual effect shown in Figure 5.26 to demonstrate that a task had been completed. The effect would show two seconds after the sixth block of a tower had been placed. It was designed to give the feeling of watching the tower being teleported away with a glowing effect and a swirl circling closer to the tower's blocks. Even though it existed as a simple effect, the effect sequence provided a visually appealing feature to watch from within the HMD.

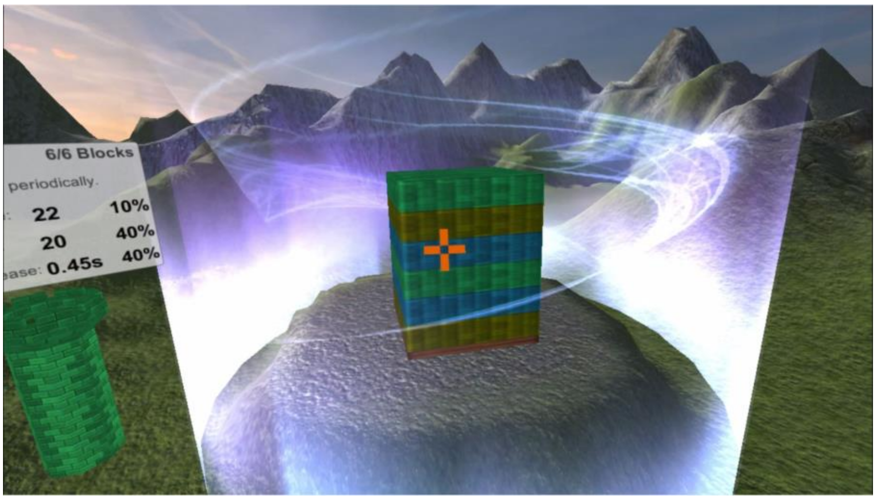
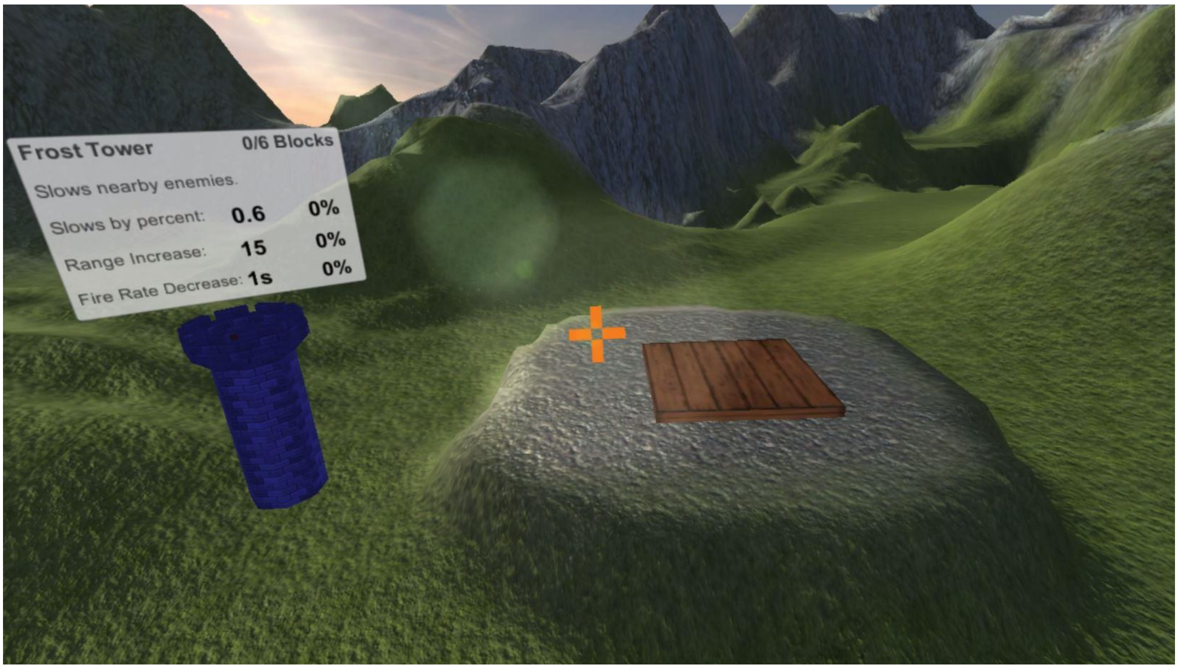


Figure 5.26: Tower Construction Task Visual Effect on Completion

As soon as the visual effect from Figure 5.26 ended, the scene would transition to show the next tower to be modified. Figure 5.27 shows the frost tower. This tower did not deal any damage and instead existed as a slowing aura. The 15-unit range is slightly smaller than the default 20 range of all the other towers. The range difference was set to balance the tower slightly for how dramatically the range could be quickly increased with modifications. Typically, the most potent property for the frost tower was range because the participant would only want one or two frost towers. By increasing the range using modifiers, a player could keep a slow aura active for a longer time for each enemy with careful placement. The slow duration was far longer than the fire rate. The slow was applied for a duration of *fire rate x 4*. For this reason, modifying the fire rate was an overall wasted modifier for this tower if used.



Frost Tower 0/6 Blocks

Slows nearby enemies.

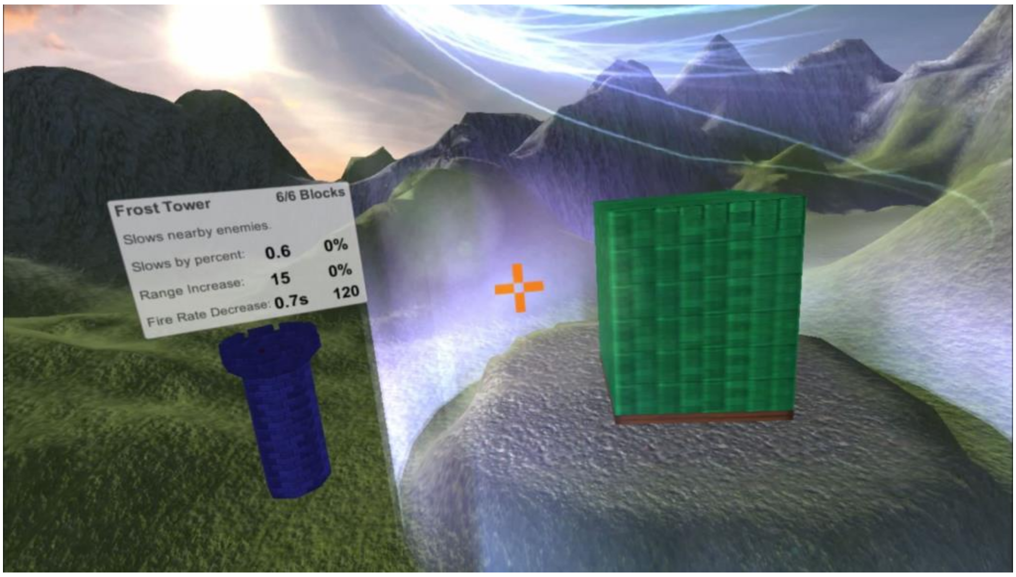
Slows by percent: 0.6 0%

Range Increase: 15 0%

Fire Rate Decrease: 1s 0%

Figure 5.27: Tower Construction Task Frost Tower

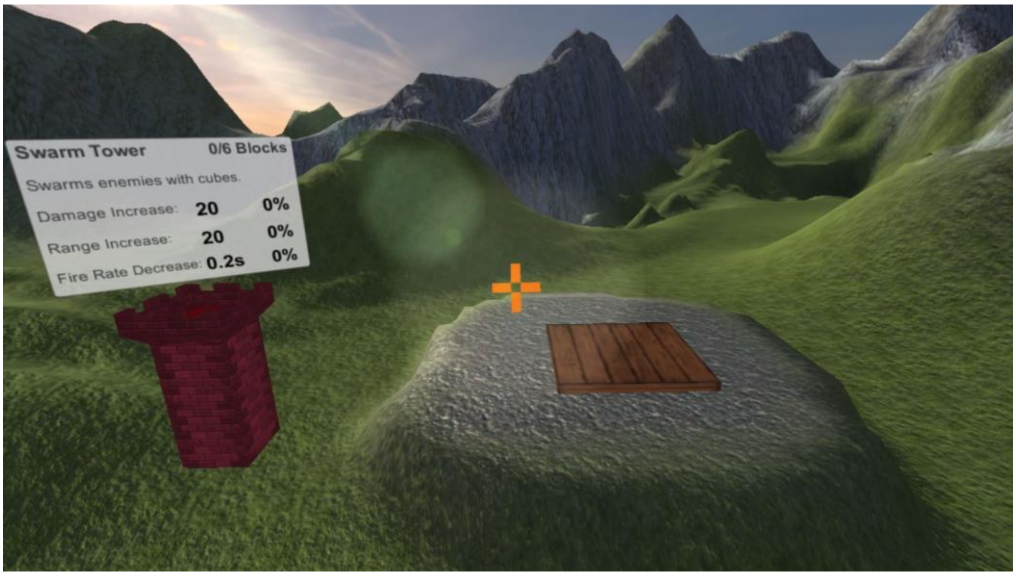
Figure 5.28 (over) shows an example case where all the same modifiers were selected. If this had been stacked with all range modifiers, then 15*(1+6*0.2) = 33 range. Or if these were all damage modifiers, the value could be 0.6*(1+6*0.05) = 78% slow effect. And as shown in the example case, 1*(1+6*-0.05) = 0.7s fire rate.



Frost Tower 6/6 Blocks
Slows nearby enemies.
Slows by percent: 0.6 0%
Range Increase: 15 0%
Fire Rate Decrease: 0.7s 120

Figure 5.28: Tower Construction Task Stacked Stats Example

Figure 5.29 shows the swarm tower. This tower had two unique properties when compared to the basic tower. The basic tower had a rate of fire of one projectile every 0.5 seconds, while this tower would fire a projectile every 0.2 seconds, causing far more frequent damage. The name for this tower comes from the path of the projectiles. The basic tower and explosive tower would fire projectiles directly at enemy units. The swarm tower would fire in a way that continually tracks and moves toward the enemy until within a threshold distance. The movement was updated with the two lines of code shown in Listing 5.1 (over). Causing a slow ramp up as projectiles approached, but as they began to hit a target, the damage would occur quickly.



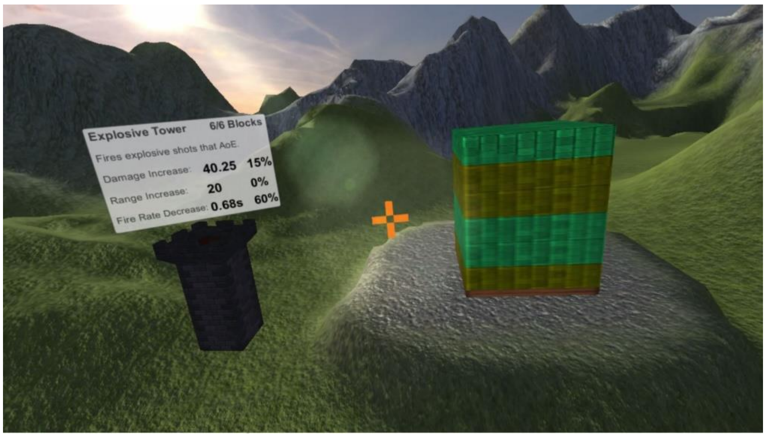
Swarm Tower 0/6 Blocks
Swarms enemies with cubes.
Damage Increase: 20 0%
Range Increase: 20 0%
Fire Rate Decrease: 0.2s 0%

Figure 5.29: Tower Construction Task Swarm Tower

```
transform.rotation = Quaternion.Slerp(myTransform.rotation,
                       Quaternion.LookRotation(target.position - myTransform.position),
                       rotationSpeed * deltaTime);
transform.position += myTransform.forward * moveSpeed * deltaTime;
```

Listing 5.1: Projectile Movement Code

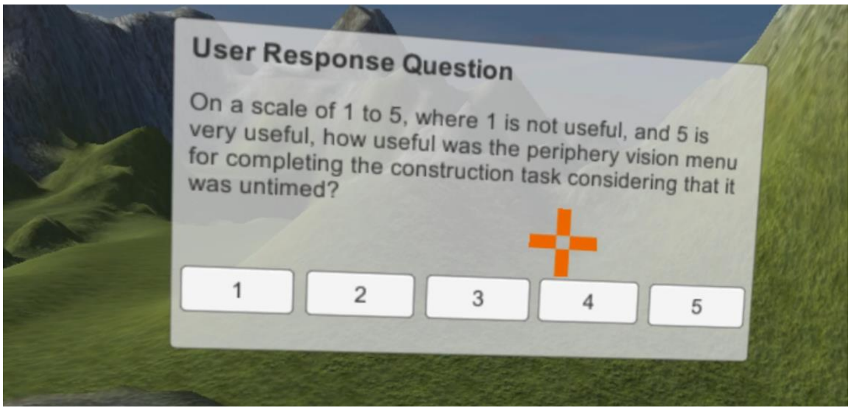
The final type of tower was the explosive tower shown in Figure 5.30. The projectiles from this tower, when reaching targets, would cause equal damage to all targets within a range of 5 units around the target. Slowing enemies with the frost tower would cause them to group up more, allowing significantly increased cleave damage from this tower. The damage dealt against single targets was considerably lower than the basic or swarm towers because of the slow fire rate.



Explosive Tower	6/6 Blocks	
Fires explosive shots that AoE.		
Damage Increase:	40.25	15%
Range Increase:	20	0%
Fire Rate Decrease:	0.68s	60%

Figure 5.30: Tower Construction Task Explosive Tower

After completing the explosive tower as the fourth tower, the participant was taken to a final question. As seen in Figure 5.31, the participant is asked how useful they felt the menu interaction was for this type of task.



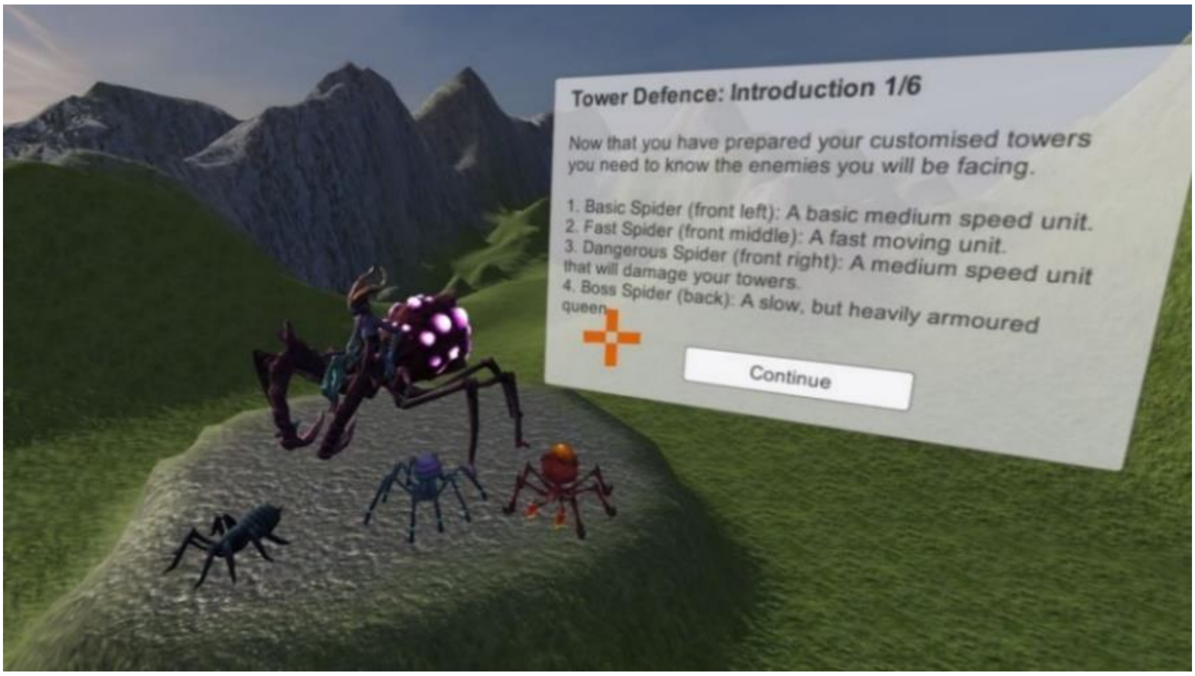
User Response Question

On a scale of 1 to 5, where 1 is not useful, and 5 is very useful, how useful was the periphery vision menu for completing the construction task considering that it was untimed?

Figure 5.31: Tower Construction User Response Question

5.2.8 Task 3: Tower Defence Game

The tower defence game was designed to motivate and attract participants to be interested in testing the PVMS. Compared to the other tasks, this one was the most complicated. The task combined two different menus with some economic management. The goal for the participant was to prevent enemies from reaching their base by creating and maintaining towers. The tower configurations created by the participants' choices in the previous task were copied over to be part of the game experience. The class files shown in 5.2.4 showed the scope of how many different scripted elements there were for this part of the experiment. Before looking at some of the features used for this experiment, the following series of 6 figures will show the introduction text and objects as they were used to explain the game experience to the participants.



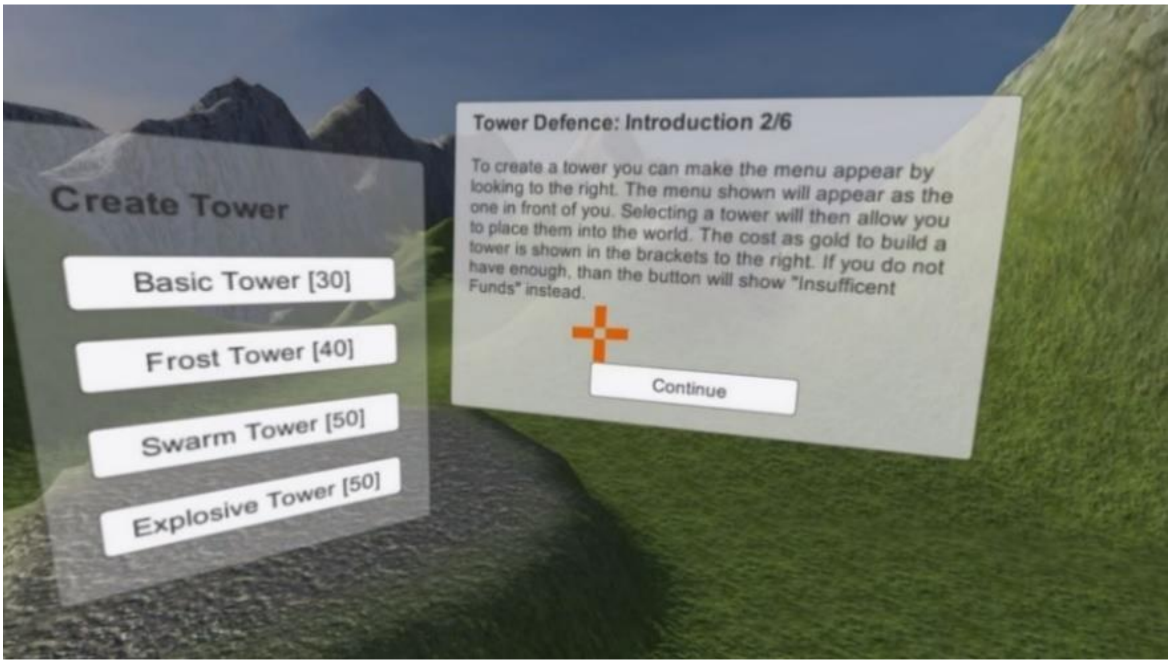
Tower Defence: Introduction 1/6

Now that you have prepared your customised towers you need to know the enemies you will be facing.

1. Basic Spider (front left): A basic medium speed unit.
2. Fast Spider (front middle): A fast moving unit.
3. Dangerous Spider (front right): A medium speed unit that will damage your towers.
4. Boss Spider (back): A slow, but heavily armoured queen.

Figure 5.32: Tower Defence Task Introduction 1/6

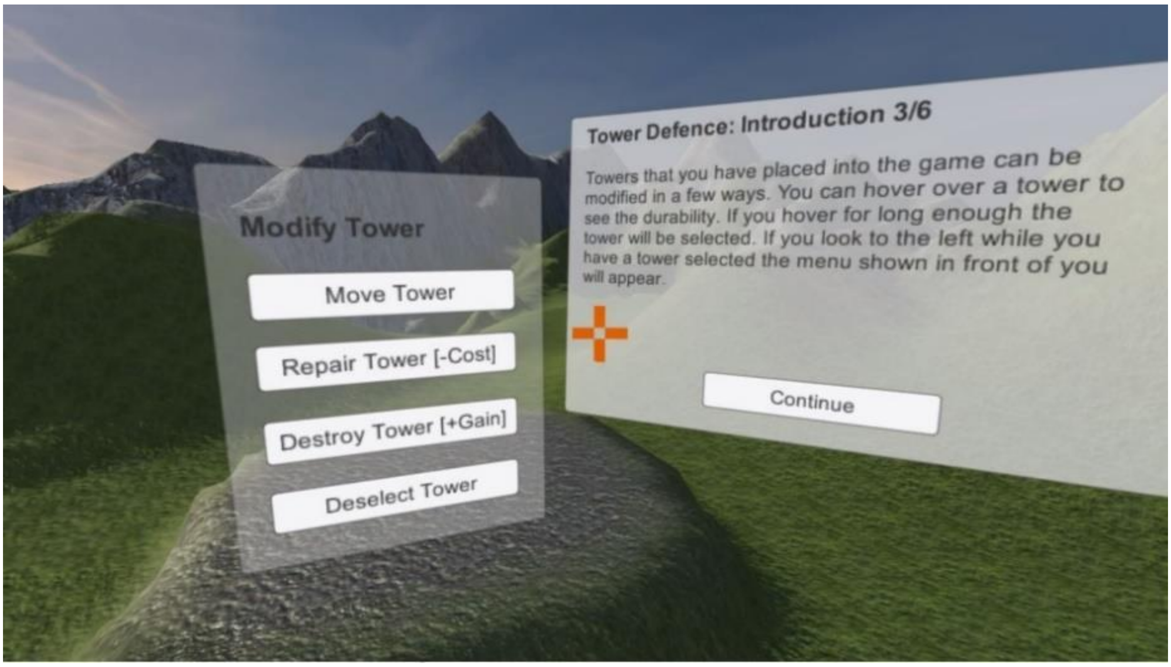
The introduction pages shown to the participants were designed to provide relevant information for understanding what was about to happen without going into excessive detail so as not to overwhelm them. Figure 5.32 began by introducing what participants would be facing as an adversary. From the previous task, participants had already been introduced to the concept of the different types of towers. Figure 5.33 introduced a currency in exchange for the defined towers. This currency would come from defeating enemies and as bonuses awarded between waves.


Tower Defence: Introduction 2/6

To create a tower you can make the menu appear by looking to the right. The menu shown will appear as the one in front of you. Selecting a tower will then allow you to place them into the world. The cost as gold to build a tower is shown in the brackets to the right. If you do not have enough, than the button will show "Insufficent Funds" instead.

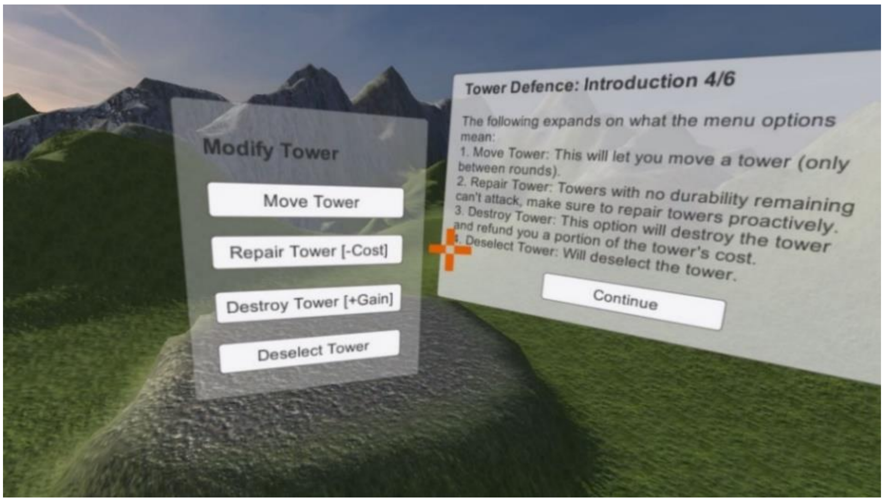
Figure 5.33: Tower Defence Task Introduction 2/6

The next step once towers were placed would then require some maintenance. The tower management was explained using Figure 5.34 and (over) by firstly describing how to make the menu appear. Then a brief explanation of what each menu option meant. Typically, the most common option needed regularly from this menu would be the repair option. The destroy tower option was never really a good choice and only included to help participants recover from planning mistakes they may have made.


Tower Defence: Introduction 3/6

Towers that you have placed into the game can be modified in a few ways. You can hover over a tower to see the durability. If you hover for long enough the tower will be selected. If you look to the left while you have a tower selected the menu shown in front of you will appear.

Figure 5.34: Tower Defence Task Introduction 3/6



Tower Defence: Introduction 4/6

The following expands on what the menu options mean:
1. Move Tower: This will let you move a tower (only between rounds).
2. Repair Tower: Towers with no durability remaining can't attack, make sure to repair towers proactively.
3. Destroy Tower: This option will destroy the tower and refund you a portion of the tower's cost.
4. Deselect Tower: Will deselect the tower.

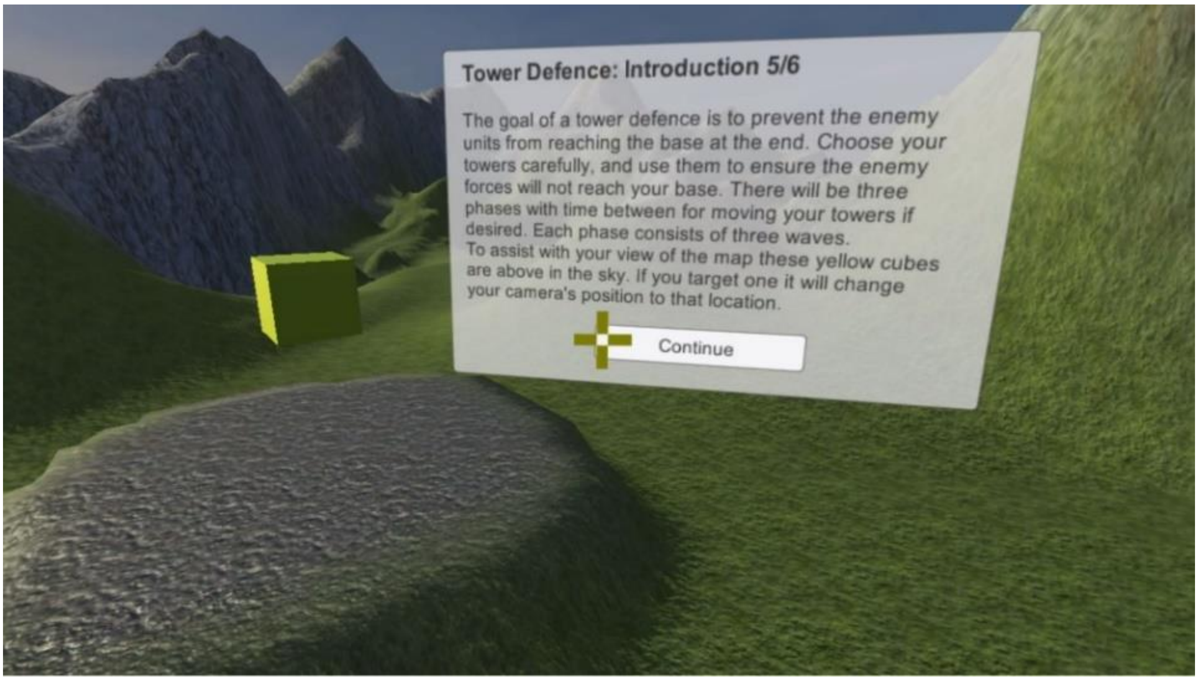
Figure 5.35: Tower Defence Task Introduction 4/6

The values for repair cost and destruction refund were calculated using the formulas shown in Listing 5.2. Both results were cast to **`int`** to drop any decimal places and were based on a combination of the original tower cost and the durability, where the durability was a number between 0 and 1. The durability will be discussed later in this section. The durability was changed from firing attacks and receiving damage from the dangerous type of enemy.

```
Repair cost = (int)(towerCost * (0.5 * (1 - durability)))

Destroy refund = (int)(towerCost * (0.3) * (1+durability))
```

Listing 5.2: Tower Cost Calculations

The next tutorial introduction page in Figure 5.36 (over) continued to explain how the game was to be played and indicated how long the player had to survive before the end. An example of the yellow cubes is also shown. These cubes were spaced out around the outer edge of the playing area. The cubes represented camera snap locations (later referred to as **`CameraSnapNodes`**) from which the participant could view the game to provide a different perspective. The original starting location was considered an optimal place to begin, but the use of these snapping locations gave a player choice. It also demonstrated a way for moving around a VR game of this type without having some other input for moving. These followed the same interaction procedures as buttons with a dwell to interact mechanic. While the player continued to look at them, they would grow to provide visual stimulus as feedback and inform the player that the interaction was occurring.

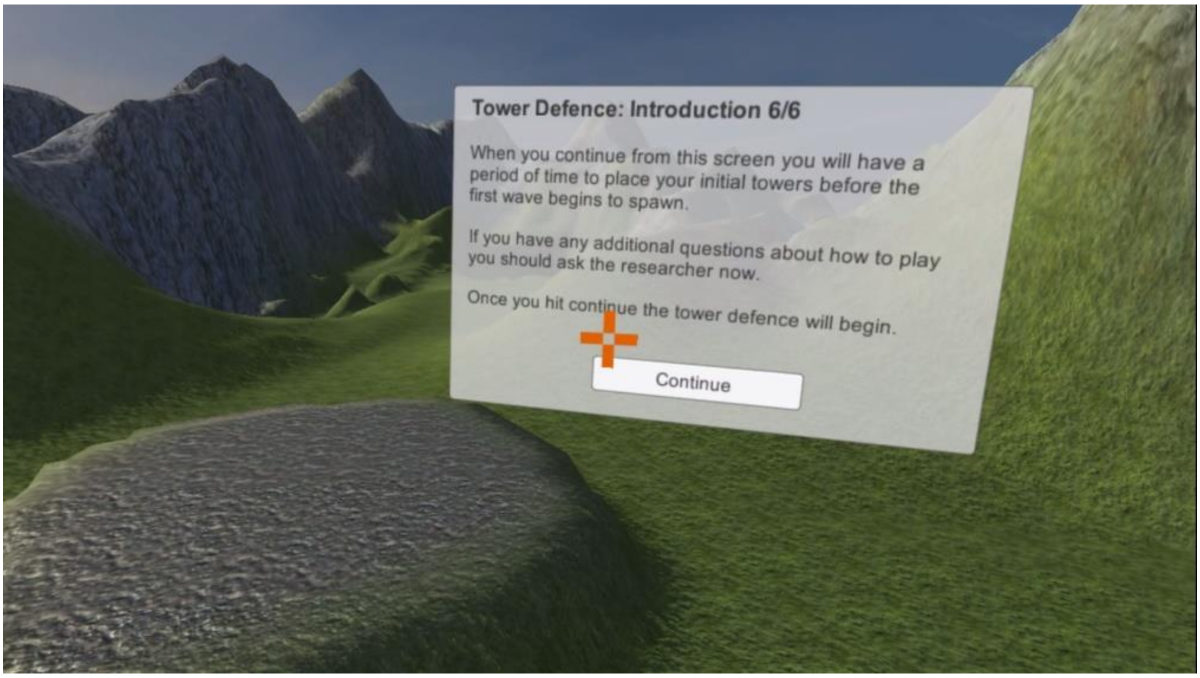


Tower Defence: Introduction 5/6

The goal of a tower defence is to prevent the enemy units from reaching the base at the end. Choose your towers carefully, and use them to ensure the enemy forces will not reach your base. There will be three phases with time between for moving your towers if desired. Each phase consists of three waves.
To assist with your view of the map these yellow cubes are above in the sky. If you target one it will change your camera's position to that location.

Figure 5.36: Tower Defence Task Introduction 5/6

The final introduction page seen in Figure 5.37 gave the participant a pause before starting into the game. This page was necessary as it gave the player a chance to reflect on the many instructions and provided them with an opportunity to ask any questions. Most participants did not feel they needed to ask anything, but a few had general questions about their expectations.



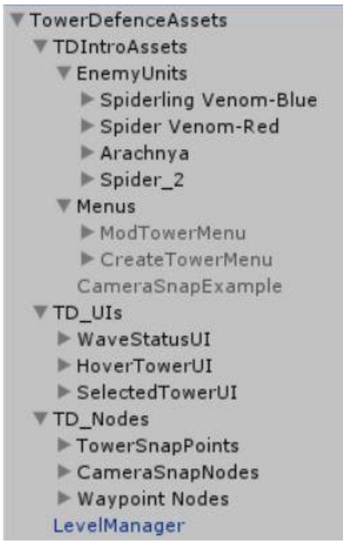
Tower Defence: Introduction 6/6

When you continue from this screen you will have a period of time to place your initial towers before the first wave begins to spawn.

If you have any additional questions about how to play you should ask the researcher now.

Once you hit continue the tower defence will begin.

Figure 5.37: Tower Defence Task Introduction 6/6



```
▼ TowerDefenceAssets
  ▼ TDIntroAssets
    ▼ EnemyUnits
      ▶ Spiderling Venom-Blue
      ▶ Spider Venom-Red
      ▶ Arachnya
      ▶ Spider_2
    ▼ Menus
      ▶ ModTowerMenu
      ▶ CreateTowerMenu
      CameraSnapExample
  ▼ TD_UIs
    ▶ WaveStatusUI
    ▶ HoverTowerUI
    ▶ SelectedTowerUI
  ▼ TD_Nodes
    ▶ TowerSnapPoints
    ▶ CameraSnapNodes
    ▶ Waypoint Nodes
  LevelManager
```

Figure 5.38: Tower Defence Task Unity Object Hierarchy

Once the introduction slides were completed, the game would commence. Figure 5.38 shows the general structure of the game object hierarchy. Everything under **`TDIntroAssets`** were non-functional elements used to represent the various examples of dialogs and objects. The three other supporting interface elements were used to show the status of the individual towers or the overall game state. Three types of nodes were used. The **`TowerSnapPoints`** where all locations towers could be snapped to after creating or while moving. **`CameraSnapNodes`** were the yellow cubes to which participants could change their camera point of view by aiming their cursor. **`WaypointNodes`** were the points for the enemies to move between in sequence. All of this was then controlled by the **`LevelManager`** object.

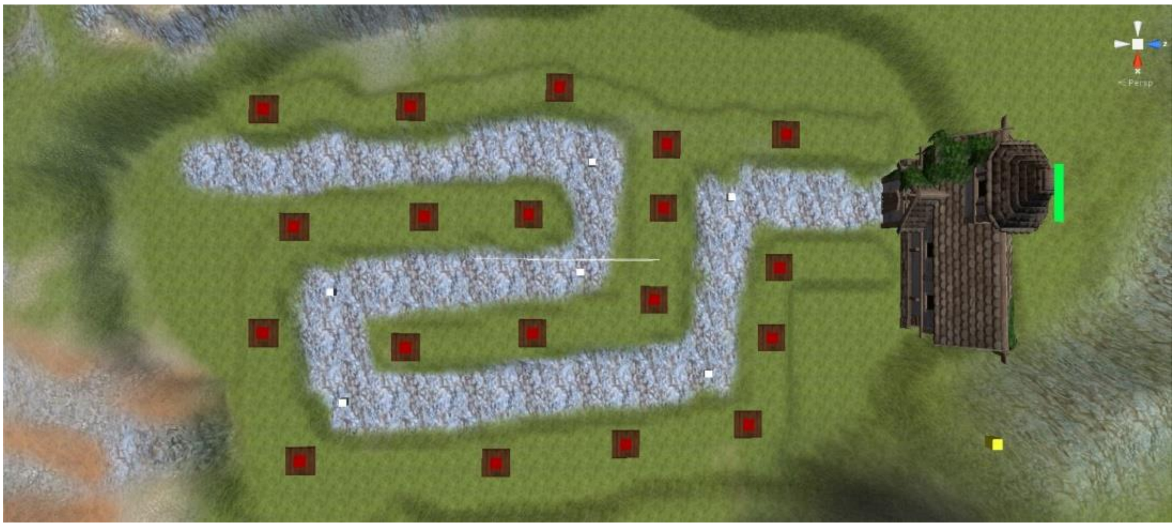


Figure 5.39: Tower Defence Task Top-Down View

Figure 5.39 (previous page) shows a top-down view of the tower defence game. All three different types of the node game object are visible in this figure. The brown bases with red marks in the middle are the **TowerSnapPoints**. The white cubes on the grey path are the AI navigation nodes that are set to hide when the game is running. One of the yellow **CameraSnapNodes** can be seen in the bottom right below the player's base. Another element of interest seen in this figure is the green bar displayed by the player base building. The green bar was used to represent the player's base health and would shrink to represent health loss. Enemies followed a similar type of health bar but used a red colour.

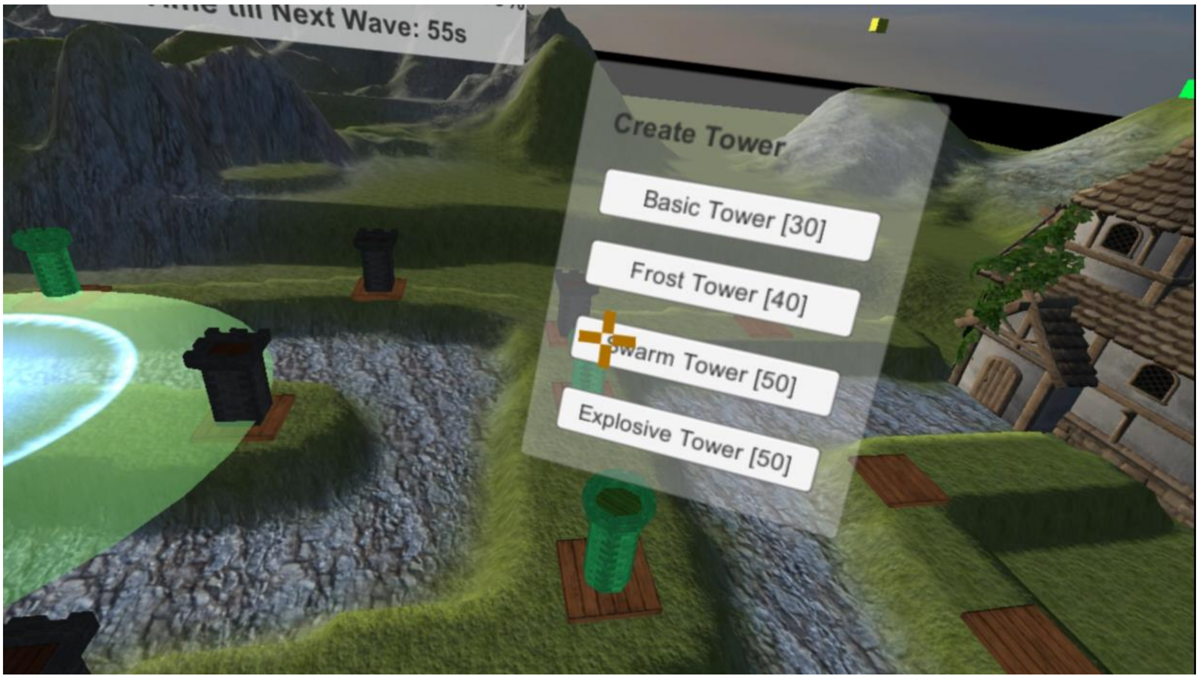


Figure 5.40: Tower Defence Task Create Tower Menu

The following series of figures will demonstrate examples of interactions within the game. Towers could be purchased from the PVMS by looking to the right. The menu shown in Figure 5.40 would look like this if the participant had 50 or more currency. Any menu option with insufficient funds to purchase the associated tower had the text replaced with "Insufficient Funds".

After purchasing a tower from the PVMS, the new tower would be attached to the cursor, as seen in Figure 5.41. All the **TowerSnapNodes** would transition to show the red region to make it obvious all the places where the new tower could be snapped to in the game. The red region would also show when a participant used the Move Tower option from the other menu. The range of a tower was visible while moving the tower with a pulsating transparent green circle. The visible range also helped while deciding where to place towers, so it was unnecessary to guess how far away enemy targets would be hit.

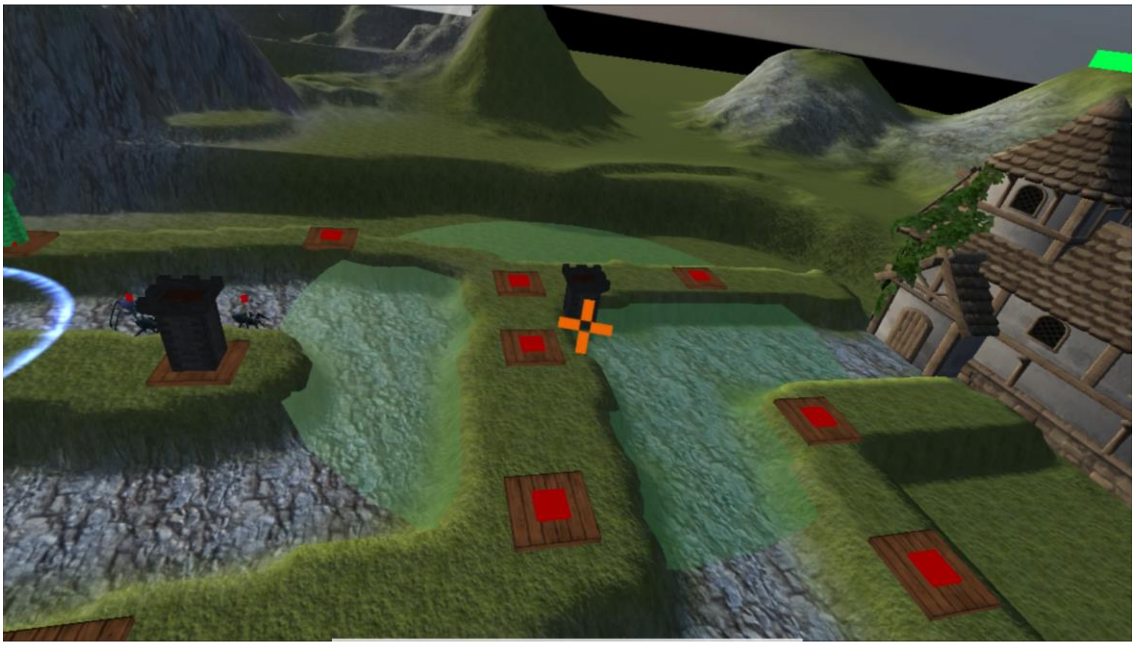


Figure 5.41: Tower Defence Task Creating/Moving Tower

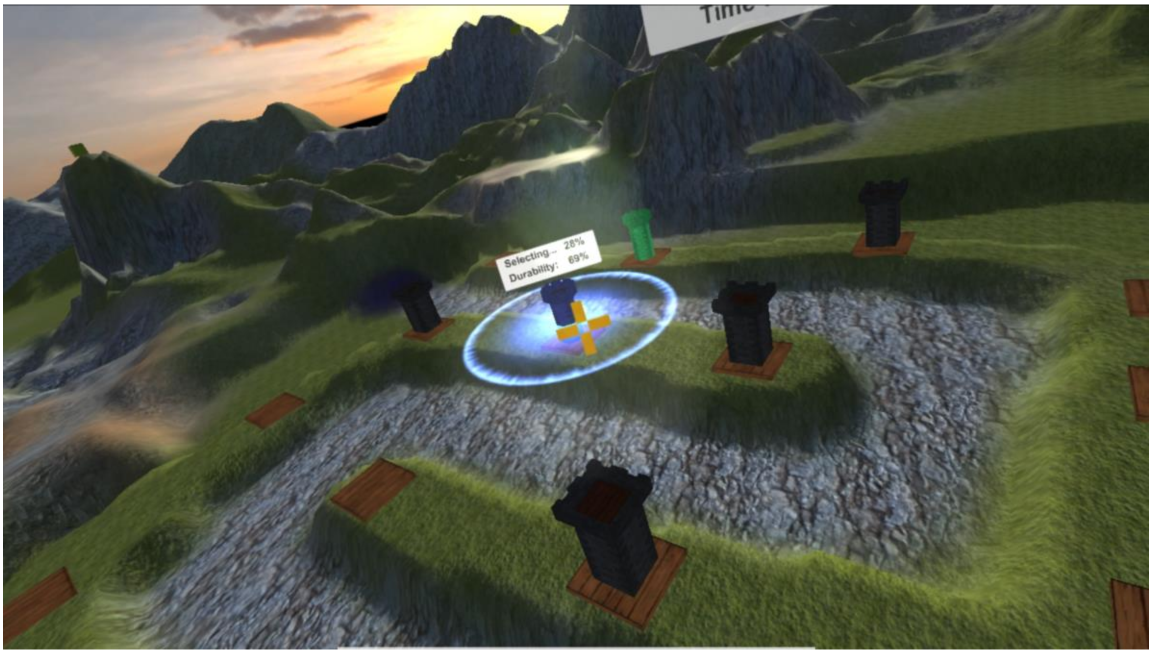


Figure 5.42: Tower Defence Task Selecting Tower

While selecting towers, there were two distinct types of dialogs used to help. Figure 5.42 shows a dialog displayed while a tower was in the process of being selected. Whenever a participant hovered over a tower, the dialog would appear and show the inspected tower's selection progress and current durability. Towers, while not selected, would hide the range effect to minimise screen clutter. However, the ring observed around the frost tower is part of the continuous visual effect of the slowing aura. Figure 5.43 shows how a tower would look after a selection has occurred. The text saying selecting is replaced with text showing the tower's name, a green background, and durability.

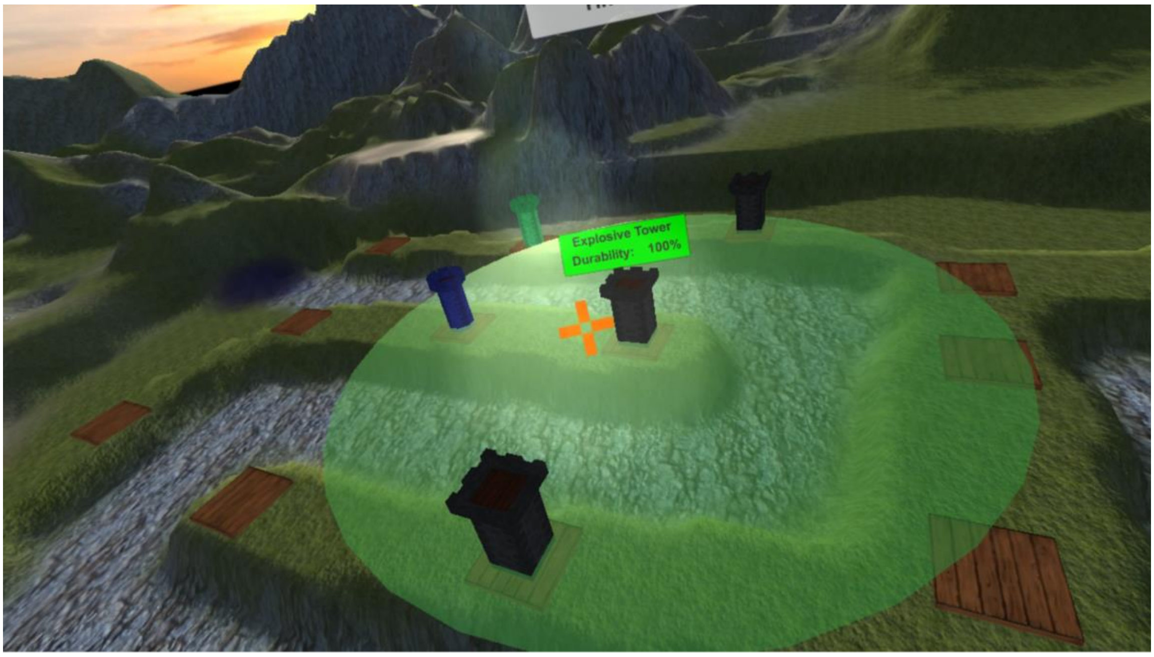


Figure 5.43: Tower Defence Task Tower Range Effect and Selected Dialog

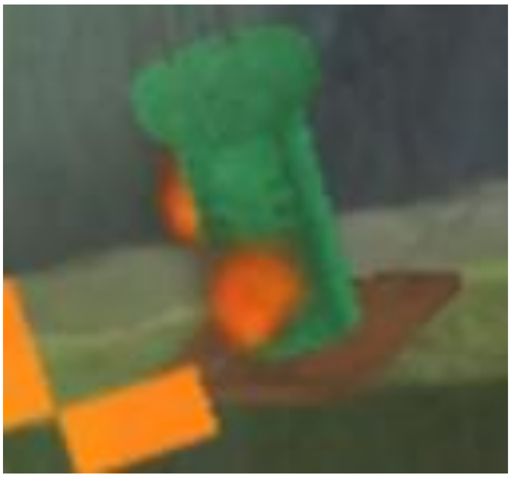


Figure 5.44: Tower Defence Task Damaged Tower Needing Repair

The durability of the towers was one of the main reasons for needing to use the Modify Tower menu. Figure 5.44 shows an example of a damaged tower. Fire effects were used to show how significantly damaged the tower was. Predetermined fire would spawn at locations on the towers at 50%, 25%, and 10% durability. To repair this damage, the tower would first have to be selected. Then the participant would trigger the PVMS by looking left.

Figure 5.45 shows an example of what this PVMS menu may look like with some flames visible. At the time of revealing this menu, there was a current wave of enemies approaching. Any time this was the case, the Move Tower option was replaced with Move Tower Unavailable. The option was replaced to prevent an exploit where towers could be moved to maximise damage as the enemies move through the level by dragging the towers to keep pace with enemies. The way durability was handled based on the activity of the tower, for every projectile fired (or for the frost tower it pulsed with a target in range) 0.5% was removed from the durability (represented as 0.005 `float`). The dangerous type enemies would fire projectiles at towers dealing 2% durability damage.

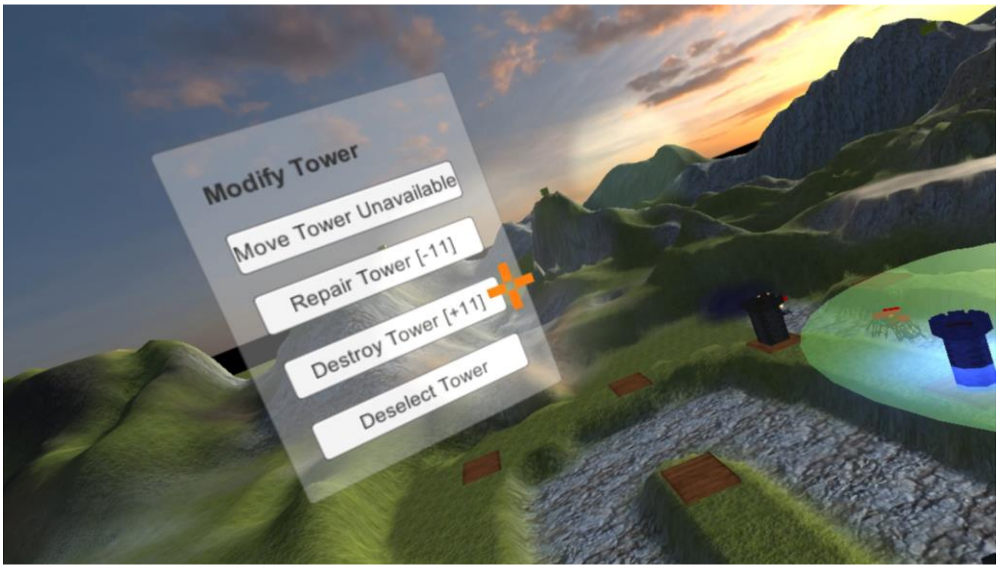
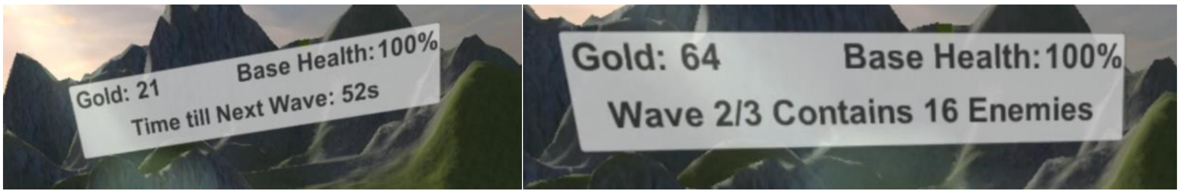


Figure 5.45: Tower Defence Task Modify Tower Menu


Gold: 21 Base Health: 100%
Time till Next Wave: 52s

Gold: 64 Base Health: 100%
Wave 2/3 Contains 16 Enemies

Figure 5.46: Tower Defence Task Wave Status UI

The `WaveStatusUI` shown in Figure 5.46 shows different examples of game states. The first state shown on the left indicates a downtime period where the participant would have a chance to rest. This time could also be used for repairing towers, constructing new towers or any of the other discussed actions. On the right is an example of what would appear during each of the waves of enemies. A wave counter showing what stage was currently being completed and the number of enemies gave the participant an understanding of what to expect from each stage as it happened. The other details shown in this dialog were the base health on the top right. Most participants did not have trouble with this value as it would typically be high. In the event of hitting 0%, it would not result in a loss for the experiment, but it was almost impossible to reach this. On the top right is shown the gold. Gold was gained in two ways. The first gold acquisition type was automatically given based on stage progress, with 100 earned at the start and 20 after each round. The other was from killing enemies. Enemies granted 4 gold for killing a basic spider, 3 gold for killing a fast spider, 6 gold for killing a dangerous spider, and 20 gold for killing a boss spider.

The waves of enemies were predefined using formatted strings of numbers. Each **`WaveCommand`** was separated by a semi-colon (;). Each **`WaveCommand`** consisted of a **`timeToWait`** and a **`unitIDToSpawn`** separated by a colon (:) between them. For example, **`timeToWait:untiIDToSpawn`** could be 1:0 to spawn unit type 0 after 1 second. -1:-1 was used to represent the end of a wave. For simplicity here, only a simple definition has been provided. Additional specifics of the **`WaveCommand`** data structure and configuration are found in Appendix B.1.7.

- Stage One Definition (3 waves)
    - Wave 1: 5 basic
    - Wave 2: (2 basic, 2 fast) twice
    - Wave 3: (4 basic, 4 fast) twice
- Stage Two Definition (3 waves)
    - Wave 1: 5 dangerous
    - Wave 2: (2 fast, 1 dangerous) three times
    - Wave 3: (1 basic, 1 dangerous, 1 fast) three times
- Stage Three Definition (3 waves)
    - Wave 1: 12 fast
    - Wave 2: (2 basic, 4 fast) twice, then 5 dangerous
    - Wave 3: (2 basic, 2 dangerous, 2 fast) three times, then 1 boss
- Boss Stage Definition (1 wave)
    - boss, dangerous, boss, dangerous, boss, boss, dangerous, boss, dangerous, boss

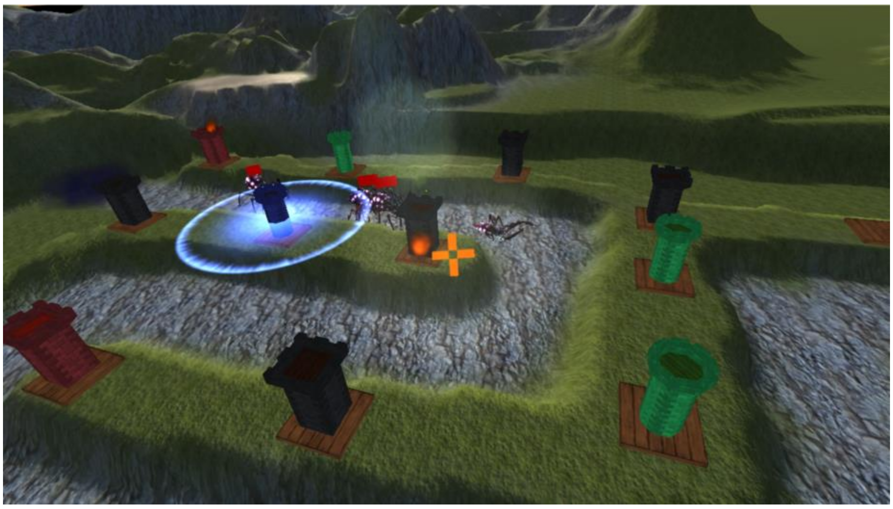
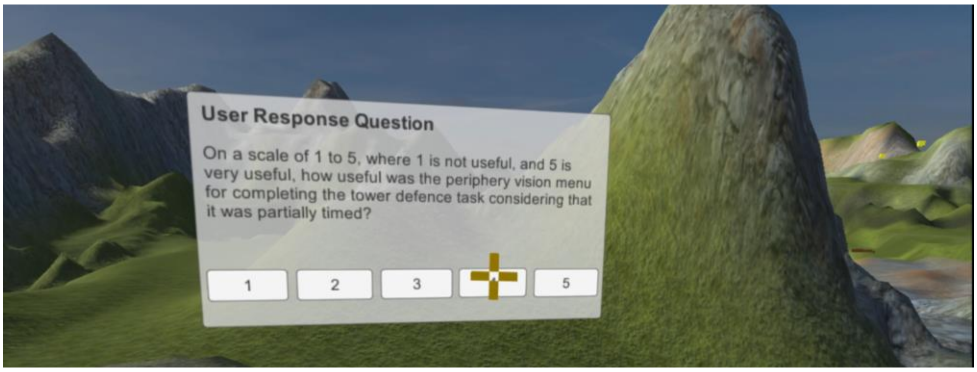


Figure 5.47: Tower Defence Task End Game Example

Figure 5.47 shows an example of how a balanced set of towers may look at the end of the experiment. Typically, one frost tower around the area where it has been placed was enough to slow the enemies down sufficiently. The slow from the frost tower would group enemies up, and the four explosive towers could perform an area of effect attack. Then for single target damage, the three basic towers and two swarm towers could provide some burst. Normally it was best to avoid the swarm towers because the cost associated with repairs from faster attacks made them prohibitively expensive to maintain. Any combination of 1-2 frost towers and the remaining towers as a combination of basic and explosive towers would generally perform well.



User Response Question

On a scale of 1 to 5, where 1 is not useful, and 5 is very useful, how useful was the periphery vision menu for completing the tower defence task considering that it was partially timed?

Figure 5.48: Tower Defence Task User Response Question

After completing the final boss stage of the tower defence task, the participant was taken away from the playing area and returned to the original interaction area. The participants were presented with the question seen in Figure 5.48. The question focused on the timed pressure to perform during the tower defence game compared to the previous task.

Table 5.3: Enemy Properties

Enemy Name	Basic Spider	Fast Spider	Dangerous Spider	Boss Spider
Move Speed	5	11	7	3
Rotation Speed	9	15	9	3
Damage to Base	1	1	1	1
Max Health	300	200	400	1500
Fire Rate	N/A	N/A	4 seconds	N/A
Max Range	N/A	N/A	15	N/A
Damage	N/A	N/A	0.02 (2%)	N/A

Table 5.3 shows the attributes of each type of enemy. Move speed indicates how fast the enemy could move forward, and the rotation speed is how fast they could turn toward a new objective. A gradual turning sequence made the enemies appear less robotic. All enemies would deal 1 damage to the end base if they reached it. If this were to become a commercial game, the damage would likely be changed to have at least the boss deal significantly more. The health made the largest difference in difficulty between the different enemies. The boss enemy was very slow-moving, so the 1500 health was not impossible. Only the dangerous enemy could fire projectiles and cause additional durability damage to a nearby tower every 4 seconds within a 15-unit distance.

Figure 5.49 (over page) shows the definition used for the **AIWayfinder** behaviour. The **AIWayfinder** was used to control all the enemy AI logic. Including movement/targeting decision making and changes between animation states. The AI Wayfinder class was responsible for accepting incoming damage from tower projectiles when an applyDamage call was made.

AI Wayfinder	

public enum UnitType { Basic = 0, Fast = 1, Dangerous = 2, Boss = 3}
// Unit Information
+ UnitType unitType;
+ String unitName;
// Navigation
+ GameObject waypoint;
- Transform target;
+ WayPointBehaviour targetBehaviour;
+ float moveSpeed;
+ float rotationSpeed;
// for waypoint completion
+ float targetReachThreshold;
- Transform myTransform;
// for slow aura
+ float moveSpeedMultiplier;
// slow aura duration
+ float moveSpeedTimer;
+ float dealsDamageToEndWayPoint;
// self-destruct
+ bool destroyOnNoTarget;
// Unit Health
- HealthBarBehaviour healthBar;
+ float maxUnitHealth;
+ float unitHealth;
+ Texture2D healthBarTexture;
+ GameObject explosionPrefab;
+ bool isDead;
// delay destruction for animation time
+ float deathTime;
+ AnimationClip deathAnimation;
- float timeTillDeath; // uses deathTime
	// Movement and Attacking Properties
	+ AnimationClip walkAnimation;
	+ AnimationClip attackAnimation;
	+ GameObject projectilePrefab:
	// one projectile at a time
	+ GameObject projectile;
	- float timeSinceLastShot;
	// firing rate
	+ float shootCooldown;
	+ float maxRange;
	+ float weaponDamage;
	+ Transform firingPosition;
	+ bool isProjectileActive;
	// Methods
	+ void Awake()
	+ void Start()
	+ void update(float deltaTime)
	- void updateWayPoint()
	+ void applyDamage(float amount)
	+ void applySpeedModifier(float amount, duration)
	- void updateMoveSpeedMod(float deltaTime)
	+ int getGoldValueOnKill()
	- GameObject spawnProjectile()
	- GameObject findNextTarget(GameObject thatIsNotThis = null)
	- void fireProjectile()
	+ void resetProjectile()

Figure 5.49: AIWayfinder Class Definition

Tower Behaviour	

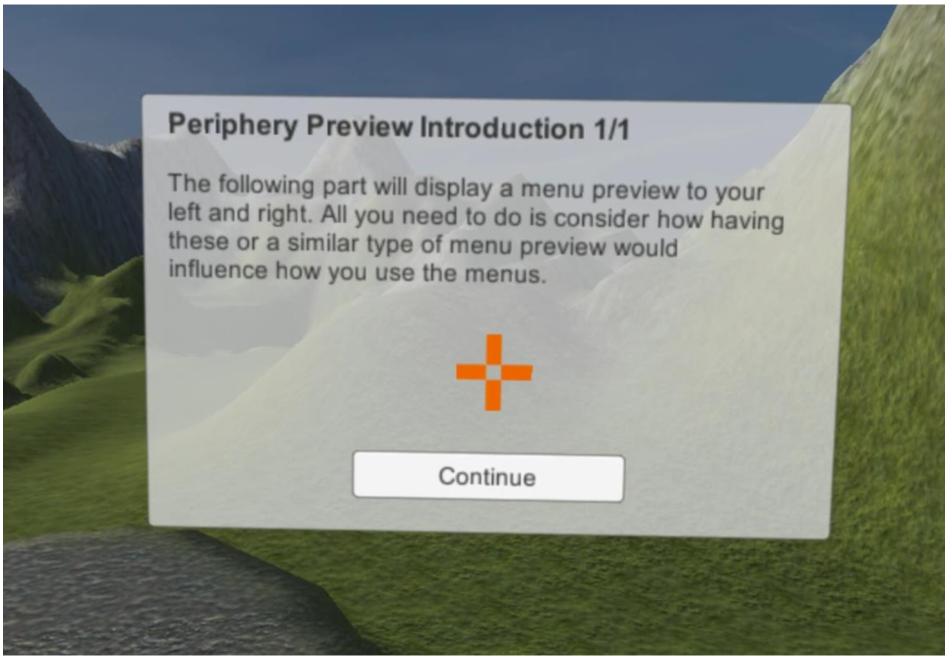
public enum TowerType { Basic = 0, Swarm = 1, Sniper = 2, Explosive = 3, Frost = 4 }
public enum TowerState { TowerPlacement, TowerAIControlled, TowerUserControlled, TowerInactive }
 	
// Tower properties
+ TowerType towerType;
+ float shootCooldown; // fire rate
+ float maxRange;
+ float towerDamage;
+ float durability;
 	
// Projectile properties
+ Transform firingPosition;
+ GameObject projectilePrefab;
// prespawned
+ List<GameObject> offProjectiles;
// active projectiles
+ List<GameObject> onProjectiles;
- float timeSinceLastShot;
 	
// Tower state
+ GameObject currentTarget;
// unused feature
+ GameObject playerTarget;
+ TowerState towerState;
// range indicator material
+ Material mat;
+ bool matAlphaIncreasing;
	// Camera
	// unused
	+ Transform cameraControlPoint;
	- CameraBehaviour cameraRef;
	
	// Child references
	+ GameObject towerRange;
	- TowerRangeBehaviour towerRangeScript;
	// durability visuals
	+ GameObject[] fireParticiles;
	- GameObject frostTowerEffect;
	
	// Methods
	+ void Awake()
	+ void Start()
	+ void update(float deltaTime)
	- fireProjectile()
	- slowAllTargetsInRange(float range, float speedModifier)
	- GameObject findNextTarget(GameObject thatisNotThis = null)
	- void spawnProjectile()
	+ void resetProjectile(GameObject projectile)
	+ void setTowerState(TowerState state)
	+ void applyDamage(float amount)
	+ void repairTower()
	+ void destroyTower()
	- void configCustomStats() // apply construction task properties

Figure 5.50:TowerBehaviour Class Definition

Figure 5.50 shows how towers were defined; each property is separated into sections with a commented header showing its purpose. Some of the properties show other features that were considered and dropped during development. The “Sniper” tower was left out as it did not feel unique enough to exist on its own, and the game felt better with just the four. Part of the possible uniqueness would have been the addition of controlling towers individually as player-controlled weapons. In a debug mode of the game, it was possible to take control of a tower and have the camera repositioned sitting on top of a tower. Then the player’s targeting would override the automatic target selection of the tower. The alternate view from on a tower could have had a special ability unique to the different towers. The mechanic felt like something you would use later in a game and not during a beginner level. The "**`TowerUserControlled`**" property and **`cameraControlPoint`** variable provided this functionality.

5.2.9 Task 4: Periphery Context Preview

The Periphery Context Preview task was initially intended to be a feature included as part of the regular testing. The goal was to have previews of the menus you would reveal with the PVMS. These would always appear in the periphery in this example task. The resolution on the Oculus Rift Dev Kit 2 felt limiting in how these could be rendered. The way the visual output was rendered for use with the HMD meant that the content had to be quite large to read objects in the periphery. The size of the objects impacted and reduced the visual real estate too much. One of the other considerations was to have the previews fade in as the menu activation threshold was being approached. The visual effects necessary to make this work well were deemed outside of the scope for the initial evaluation of this menu system. To still evaluate participants thoughts, the content was still presented in this task as a quick question to ask participants with a visual example. The introduction to this task given to the participant is seen in Figure 5.51.

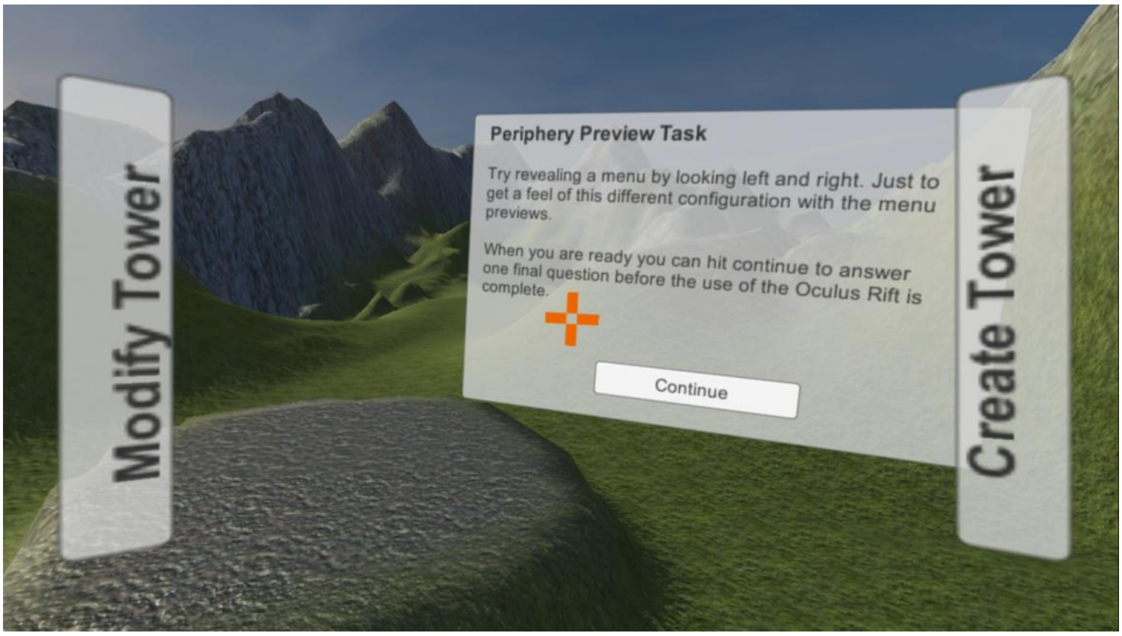


Periphery Preview Introduction 1/1

The following part will display a menu preview to your left and right. All you need to do is consider how having these or a similar type of menu preview would influence how you use the menus.

Figure 5.51: Periphery Context Preview Task Introduction

After continuing, the participant would see a screen as shown in Figure 5.52. The way it would be observed in a HMD is difficult to show clearly with a screenshot, as the capture does not have the visualisation changes applied when rendering to a HMD in Unity.



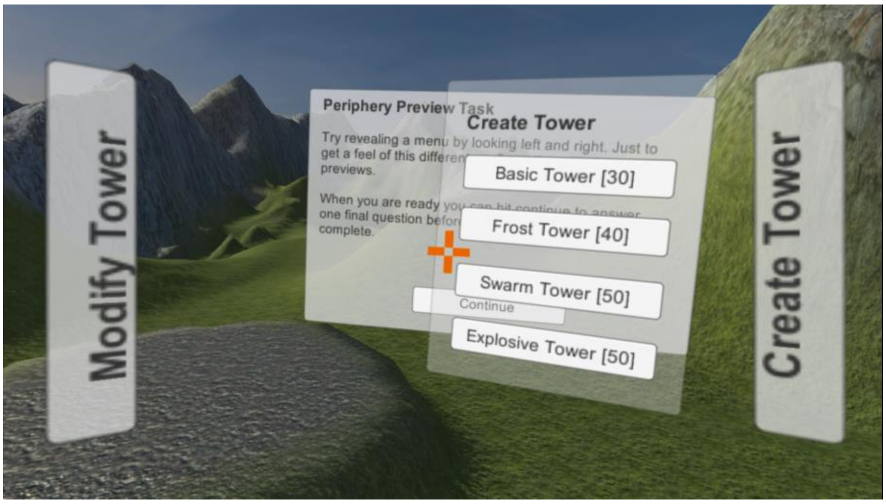
Periphery Preview Task

Try revealing a menu by looking left and right. Just to get a feel of this different configuration with the menu previews.

When you are ready you can hit continue to answer one final question before the use of the Oculus Rift is complete.

Figure 5.52: Periphery Context Preview Active

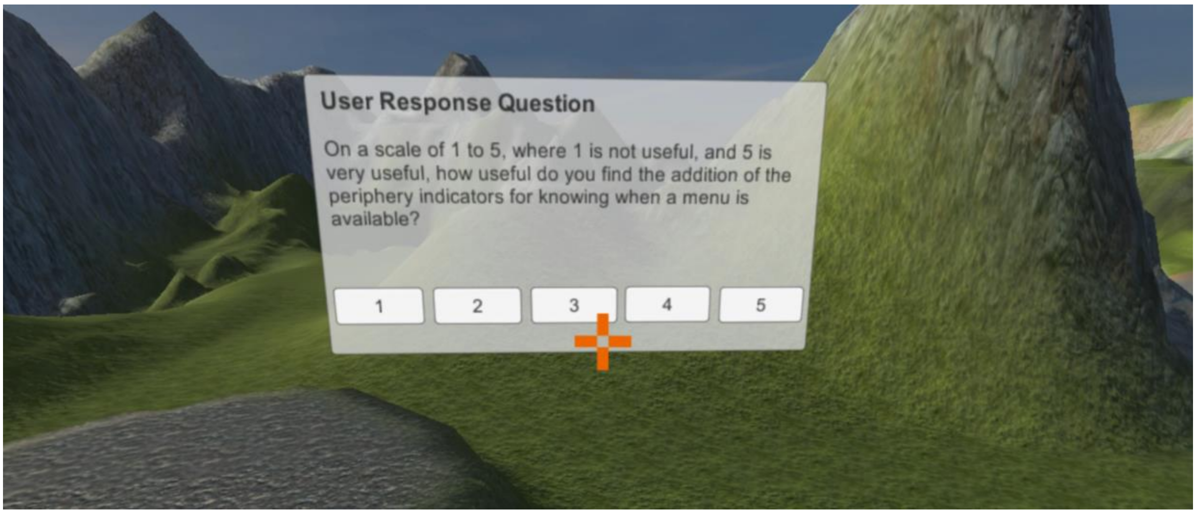
The screenshots make the elements to the left and right appear far more visible than they were. Part of the trick is that each eye would only see one part each to make them always outside the participant's focus. For this example, both types of menus from the tower defence task could be triggered, as shown in Figure 5.53. The options on the menu were disabled for the task. It was only a visual representation of the process associated with these menu systems.



Modify Tower
Periphery Preview Task
Try revealing a menu by looking left and right. Just to get a feel of this differen
previews.
When you are ready you can hit continue to answer one final question befor
complete.
Continue
Create Tower
Basic Tower [30]
Frost Tower [40]
Swarm Tower [50]
Explosive Tower [50]
Create Tower

Figure 5.53: Periphery Context Preview Menu Active Example

The participant was free to experiment and explore this feature. After previewing this addition to the PVMS, the participant could use the Continue button to take them to Figure 5.54. Here they were asked to rate how useful they thought the addition of this feature would be to the overall system.

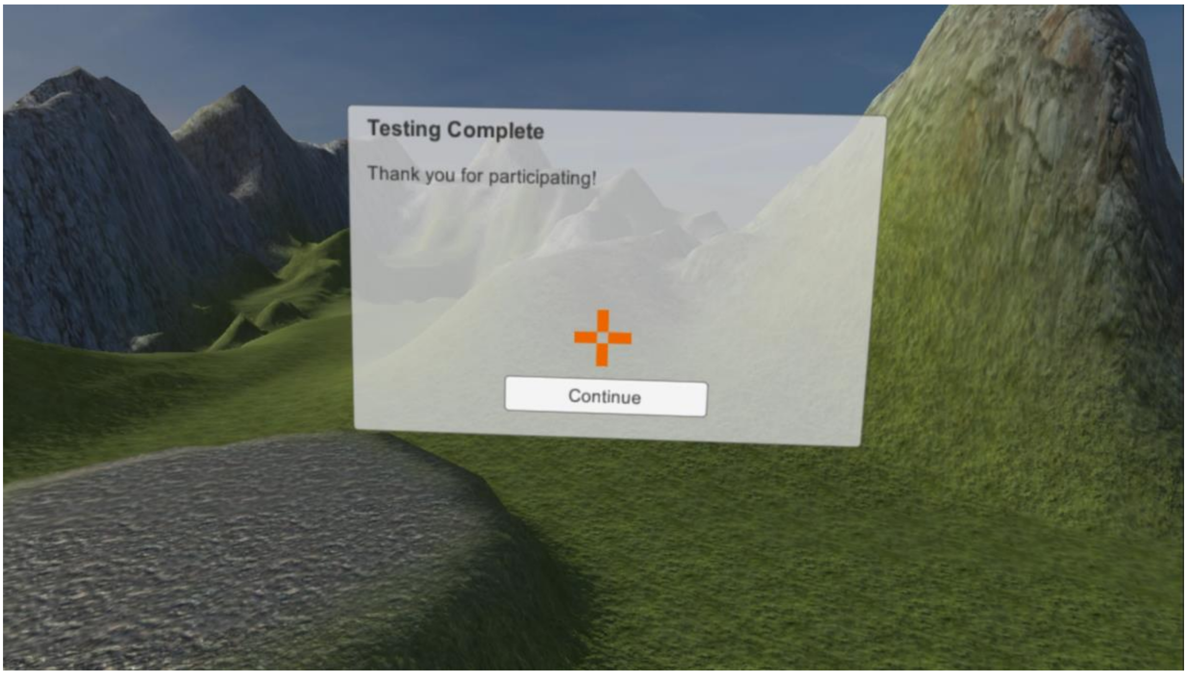


User Response Question

On a scale of 1 to 5, where 1 is not useful, and 5 is very useful, how useful do you find the addition of the periphery indicators for knowing when a menu is available?

Figure 5.54: Periphery Context Preview User Response Question

After answering the last user response question, the participant was taken to a testing complete dialog shown in Figure 5.55. At this point, they could remove the HMD and continue to complete the post-experiment questionnaire.



Testing Complete

Thank you for participating!

Figure 5.55: Experiment Complete Dialog

5.2.10 Pre and Post Experiment Questionnaires

Each participant completed two questionnaires to evaluate their feedback related to the experiment and surrounding topics as part of the experiment. The wording is modified for the question summaries to represent the point of the questions without the language targeting participants where appropriate. The questions used in this experiment were reduced in quantity from the first experiment, focusing more on the quality of questions over the quantity of data. The complete surveys with questions as they were asked can be found in Appendix B.2.

Pre-Experiment 16 questions

- Questions 1 to 4: Personal and Academic Background covering questions related to Age, Gender, Student/Other, Area of Study/Teaching/Research.
- Questions 5 and 6: Participation in research or personal use with VR, AR, HMDs, or Mobile/Tablet Computing. With space to list related relevant projects or applications.
- Questions 7 to 9: Use of Computers
    - How many hours a week would you use a computer on average?
    - How many hours a week would you spend playing video games on average?
    - Examples of computer games or consoles typically played on.
- Question 10: Interest in using HMDs for non-gaming activities.
- Question 11: Frequency of having played tower defence games. 1 = Never, 10 = Very Often.
- Question 12: The significance when considering the usability of user interfaces for HMDs with either AR or VR. Ranking from 1 to 4 the features: speed of accessing features, the accuracy of accessing features, simplicity of physical interactions, the visual appeal of the interface. With the option of listing any other features that were felt to be important.
- Question 13: Rank from 1 to 4 personal preference toward input methods: Oculus Rift by itself, Oculus Rift with a computer mouse, Oculus Rift with a mobile device, Oculus Rift with the LEAP or Microsoft Kinect Sensors. Each of these was explained with a sentence to clarify how these would work for the participant.

- Question 14: List any other preferred devices or interaction methods not included in question 13 for use with HMDs for AR or VR.
- Question 15: On a scale of 1 to 10 (1 = no influence, 10 = high influence), how much did the game trailer influence desire to participate in the research.
- Question 16: On a scale of 1 to 10 (1 = no influence, 10 = high influence), feeling toward visual presentations such as the game trailer's influence on the desire for future research participation.

Post-Experiment 21 questions

- Question 1: Found the periphery menu system useful in the way it appeared? Yes or no, and why.
- Question 2: How accurately the periphery menu system responded when wanting to make it display. Scale of 1 to 10. 1 = not accurate, 10 = very accurate.
- Question 3: How often the periphery menu system displayed at the wrong times or when not meaning to display it. Scale of 1 to 10. 1 = not often, 10 = very often.
- Question 4: How often the periphery menu system displayed at the correct times. Scale of 1 to 10. 1 = not often, 10 = very often.
- Question 5: How useful the gesture felt of rotating the head to make a menu appear. Scale of 1 to 10. 1 = not useful, 10 = very useful.
- Question 6: Any suggested changes for how the periphery menu system worked.
- Question 7: How likely would it be for wanting to use the interaction again in the future. Scale of 1 to 10. 1 = not likely, 10 = very likely.
- Question 8 and 9: Three aspects found most enjoyable and most difficult while using the periphery vision menu.
- Question 10: System Usability Scale (Sauro, 2011) with all questions using 1 to 5 scales of strongly disagree to strongly agree. All questions used language focusing on the menu system so that participants would understand they were about the menus and not the application in general.
    - "I think that I would like to use this menu system frequently."
    - "I found the menu unnecessarily complex."
    - "I thought the menu was easy to use."

  - “I thought that I would need the support of a technical person to be able to use this menu.”
  - “I felt that options presented by menus were well integrated.”
  - “I felt that there was too much inconsistency with the menu.”
  - “I would imagine that most people would learn how to use these menus very quickly.”
  - “I found the menus very cumbersome to use.”
  - “I felt very confident using the menus.”
  - “I needed to learn a lot of things before I could use the menus.”
- Question 11: Rank from 1 to 5 activities based on where the menu system would be most desirable. 1 = most desired, 5 = least desired. Options: Games, Viewing a Movie, Constructing Models, Instant Messenger or Voice Chat, Operating System Controls/Menus.
- Question 12: Any other scenarios not listed in 11.
- Question 13: How useful the menu interaction would be suited for AR. Scale of 1 to 10. 1 = not suited, 10 = very suited.
- Question 14: Preference of VR, AR, or no preference for the use of the menu interaction and why.
- Question 15: The significance when considering the usability of user interfaces for HMDs with either AR or VR. Ranking from 1 to 4 the features: speed of accessing features, the accuracy of accessing features, simplicity of physical interactions, the visual appeal of the interface. With the option of listing any other features that are felt to be important. This question is a repeat of question 12 from the pre-experiment questionnaire.
- Question 16: Rank from 1 to 4 personal preference toward input methods: Oculus Rift by itself, Oculus Rift with a computer mouse, Oculus Rift with a mobile device, Oculus Rift with the LEAP or Microsoft Kinect Sensors. Each of these was explained with a sentence to clarify how these would work for the participant. This question is a repeat of question 13 from the pre-experiment questionnaire.

- Question 17: List any other preferred devices or interaction methods not included in question 16 for use with HMDs for either AR or VR. This question is a repeat of question 14 from the pre-experiment questionnaire.
- Question 18: Rate how much fatigue was felt while using the Oculus Rift for performing the menu interactions. Scale of 1 to 10. 1 = low fatigue, 10 = high fatigue.
- Question 19: "Do you feel the Oculus Rift by itself with the provided functionality provides enough functionality to stand alone?". Scale of 1 to 10. 1 = strongly disagree, 10 = strongly agree.
- Question 20: Any other thoughts about the experiment.

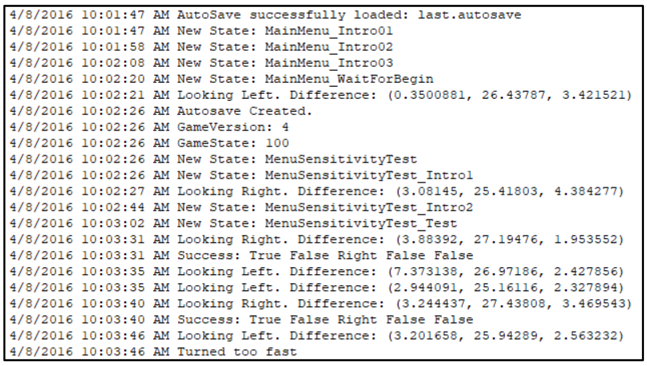
5.2.11 Application Logged Data

The participant responses to the two questionnaires were only one aspect of the data collected during this experiment. While the Unity application was running, several types of automatic data collection were occurring. For each participant, four different data files were generated with the following naming formats (where the start is a date/time format).

- yyyy_M_d__hh_mm.log: Text logs of events.
- yyyy_M_d__hh_mm.autosave: AutoSave serialised data.
- yyyy_M_d__hh_mm.dat: Replay data binary file.
- yyyy_M_d__hh_mm.csv: Categorised extra data log.

Text Log

Text logs contained details of state change events, PVMS events, user response questions and the randomised task order information. An example exert from the event text log can be seen in Figure 5.56.



```
4/8/2016 10:01:47 AM AutoSave successfully loaded: last.autosave
4/8/2016 10:01:47 AM New State: MainMenu_Intro01
4/8/2016 10:01:58 AM New State: MainMenu_Intro02
4/8/2016 10:02:08 AM New State: MainMenu_Intro03
4/8/2016 10:02:20 AM New State: MainMenu_WaitForBegin
4/8/2016 10:02:21 AM Looking Left. Difference: (0.3500881, 26.43787, 3.421521)
4/8/2016 10:02:26 AM Autosave Created.
4/8/2016 10:02:26 AM GameVersion: 4
4/8/2016 10:02:26 AM GameState: 100
4/8/2016 10:02:26 AM New State: MenuSensitivityTest
4/8/2016 10:02:26 AM New State: MenuSensitivityTest_Intro1
4/8/2016 10:02:27 AM Looking Right. Difference: (3.08145, 25.41803, 4.384277)
4/8/2016 10:02:44 AM New State: MenuSensitivityTest_Intro2
4/8/2016 10:03:02 AM New State: MenuSensitivityTest_Test
4/8/2016 10:03:31 AM Looking Right. Difference: (3.88392, 27.19476, 1.953552)
4/8/2016 10:03:31 AM Success: True False Right False False
4/8/2016 10:03:35 AM Looking Left. Difference: (7.373138, 26.97186, 2.427856)
4/8/2016 10:03:35 AM Looking Left. Difference: (2.944091, 25.16116, 2.327894)
4/8/2016 10:03:40 AM Looking Right. Difference: (3.244437, 27.43808, 3.469543)
4/8/2016 10:03:40 AM Success: True False Right False False
4/8/2016 10:03:46 AM Looking Left. Difference: (3.201658, 25.94289, 2.563232)
4/8/2016 10:03:46 AM Turned too fast
```

Figure 5.56: Event Text Log

Extra Data Logs

The following list shows all the automatically generated data that was stored in a CSV file for easy loading. Most of this data existed to expose the stats showing players had been doing during the game portions.

- Menu Behaviour Log: Timestamp, Game State, Menu Definition, Menu Up Time, Menu Hide Reason, Button Name, Button ID, Button Result Code.
- PVMS Events: Timestamp, Game State, Rotation Result, Show Menu, Sensitivity Setting, **Diff_X** (Pitch), **Diff_Y** (Yaw), **Diff_Z** (Roll), History Count, Full History of Quantity Count with X;Y;Z;DeltaTime with oldest to newest events.
- Time Log: Timestamp, Game State, Time Since Last State
- User Response Questions: Timestamp, Question, Response
- Construction Task Data: Timestamp, Tower ID, Version ID, Damage Modifier Count, Range Modifier Count, Fire Rate Modifier Count, Removed Modifier Count.
- Sensitivity Test Data: Timestamp, Left/Right, Menu/No Menu, Sensitivity Setting, Task Time, Left Menus Triggered, Right Menus Triggered, Left Panels Used, Right Panels Used.
- Tower Defence Stats: Timestamp, Level State, Tower Basic Built, Tower Frost Built, Tower Swarm Built, Tower Explosive Built, Tower Repaired, Tower Destroyed, Units Killed, Failed Tower Buy, Failed Repair, Camera Snap Used, Base Health.

Replay System

The replay system was designed to be very similar to what was used in the first experiment with a few simple improvements. For details on the system used in the first experiment, refer to section 3.2.13. The database class, as seen in Figure 5.57, is identical to what was used for the first experiment.

ReplayDatabase	
+ List<ReplayEvent> replayEvents	+ ReplayDatabase()
+ int replayEventID	+ ReplayEvent getNextEvent()
+ float excessDeltaTime	+ void saveDatabase(string filename)
- string databaseCreation	+ void loadDatabase(string filename)
- string fileOpen	**+ getCreationData()**

Figure 5.57: ReplayDatabase Class Definition

The primary changes came in the **`ReplayEvent`,** as seen in Figure 5.58. Specifically, the use of a **`KeyActionEvent`** to define level transitions and key interactions was replaced with two different variables. **`ExperimentState.GameState`** represented the current game state and took over tracking when states changed. A separate variable for state allowed separating anything to do with input. The other addition was a variable list of **`KeyCode`**s to track multiple key events during any single frame. The key tracking support was mainly only relevant for debugging because no inputs were used other than the HMD input.

[Serializable] ReplayEvent	

// All variables are [NonSerializedAttribute]
+ float deltaTime;
+ Quaternion rotation;
+ Vector3 position;
+ bool triggerEdge;
+ ExperimentState.GameState gameState;
+ List<KeyCode> keyEvents
	+ ReplayEvent()
	+ ReplayEvent(float deltaTime, Quaternion rotation, Vector3 position, bool triggerEdge, ExperimentState.GameState gameState, List<KeyCode> keyEvents)
	+ void GetObjectData(SerializationInfo info, StreamingContext context)
	+ ReplayEvent(SerializationInfo info, StreamingContext ctxt)

Figure 5.58: ReplayEvent Class Definition

The problem with the replay system that made it unreliable for both this experiment and the third experiment was the use of physics interactions. Enemy units used forward vectors to update their position and obeyed object collisions with other enemy units and the terrain. The use of direction vectors and collisions was used to make enemies stay spread out even when slowed. Minor changes within the physics system between experiment participation and when the system was running a replay caused some synchronisation problems. The errors were relatively rare but made it unreliable for any post-processing on the replay data and resulted in a novel feature rather than an analysis tool. Retrospectively to both experiments, one possible solution to this problem would have been to store some of the object state information to correct for errors periodically.

5.3 Results

The results section of this chapter will broadly cover the spread of data collected from this experiment. Firstly, looking at the questionnaire results in 5.3.1 and then the results collected from the application data directly in 5.3.2. For a discussion about the results overall for the experiment, see section 5.4.

5.3.1 Questionnaire Results

This section will summarise participant responses to the two different questionnaires. The results have been grouped based on the content and the order of appearance in the questionnaires. Complete examples of the questionnaires are found in Appendix B.2.

General Background

For this experiment, there was a total of 23 participants who completed the entire experiment. A single participant completed the pre-experiment questionnaire but withdrew before completing the experiment tasks due to vision issues when not using glasses. Their data has been included as part of the pre-experiment data. Of the participants, there were 23 males and 1 female. Ages for participants included 9 participants under 21, 11 between 21 and 30, 4 between 31 to 40. All participants were students studying in the areas of computer science, engineering, information technology, and marketing.

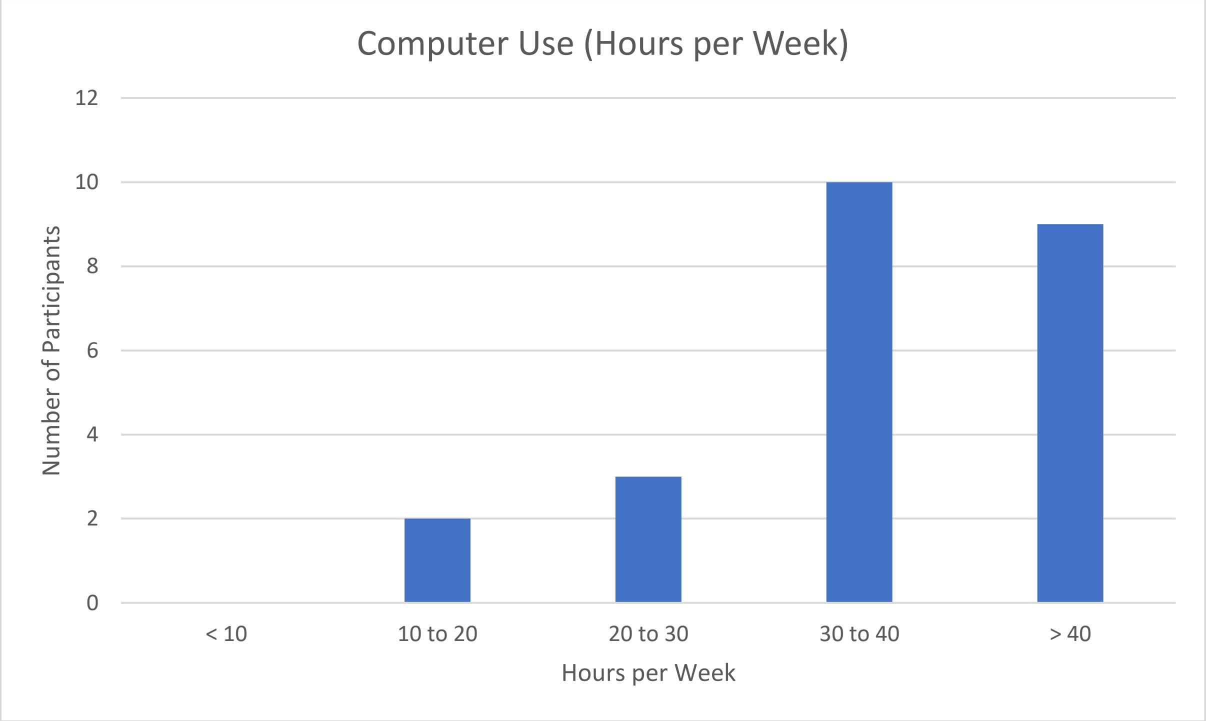


Figure 5.59: Computer Use (Hours per Week)

Figure 5.59 shows computer use by participants. Most participants indicated they spent 30 or more hours per week using a computer. Figure 5.60 (over) then shows the time spent playing games during an average week. The experiment was conducted during a university semester, and therefore the numbers reflected current perceived use and not an average across a year.

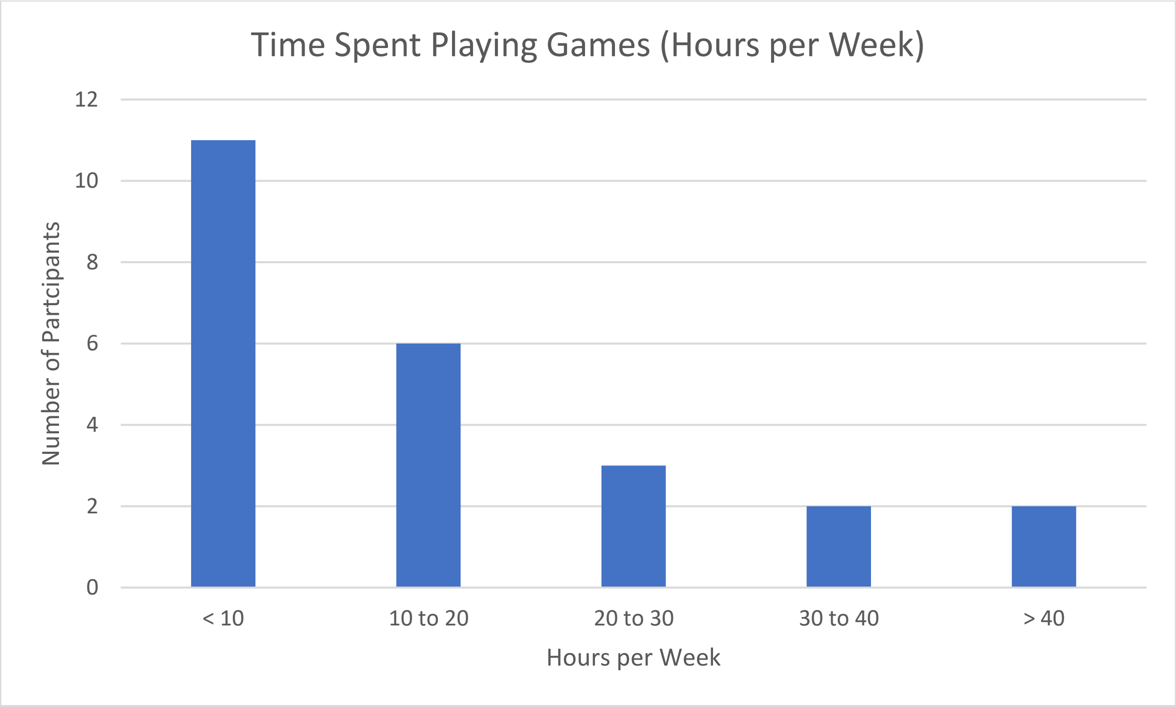


Figure 5.60: Time Spent Playing Games (Hours per Week)

Figure 5.61 shows the numbers of participants who had previously participated in research with specific types of devices. Only three participants had previously used VR, AR or HMDs in a research setting. Five participants had used mobile or tablet type devices previously as part of other research.

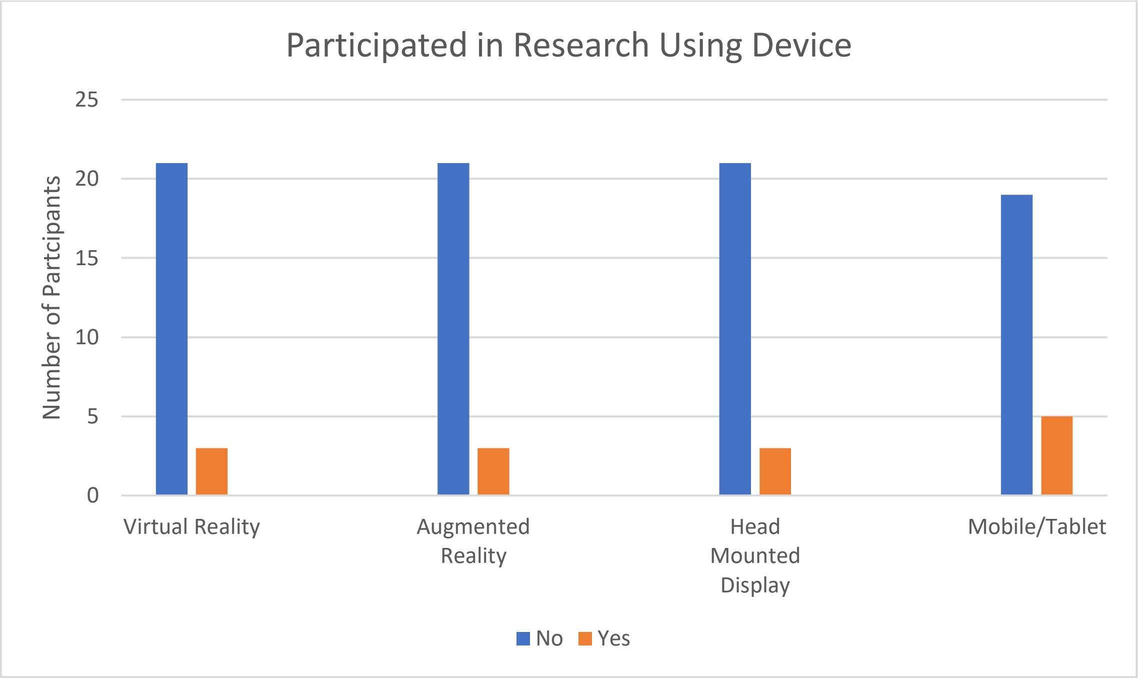


Figure 5.61: Previous Participation in Research

Figure 5.62 shows slightly more participants had used devices for personal use compared to previous research. Five participants had previously used VR, four had previously used AR, and three had previously used HMDs. As expected, a higher result of personal use for mobile devices was found, with 17 participants indicating they had previously used mobile/tablet devices. It is suspected participants misunderstood the mobile/tablet response. It is reasonable to assume every person would have a smartphone for students studying a technology-focused discipline at a tertiary institution at the time of the experiment.

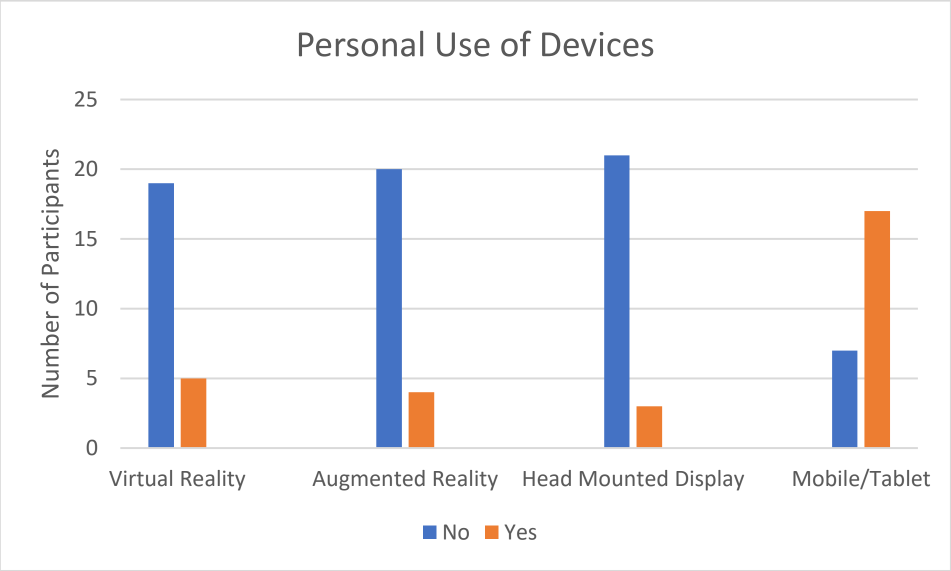


Figure 5.62: Previous Personal Use of Devices

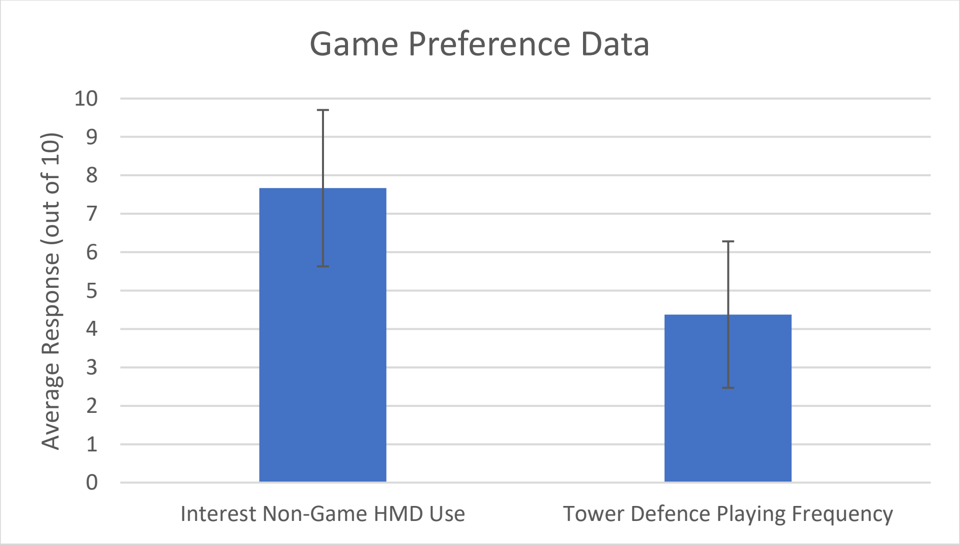


Figure 5.63: Game Preference Data

Figure 5.63 shows additional metrics related to participants' activity preferences. Participants responded, indicating a positive interest in using HMDs for non-gaming uses with a response of 7.67 (SD = 2.04, ranking on a Likert scale out of 10). As participants were going to play a tower defence style game during this experiment, the amount of time they had played this type of game was requested. The response of 4.38 (SD = 1.9) indicates participants, on average, occasionally play this type of game.

Interface Usability

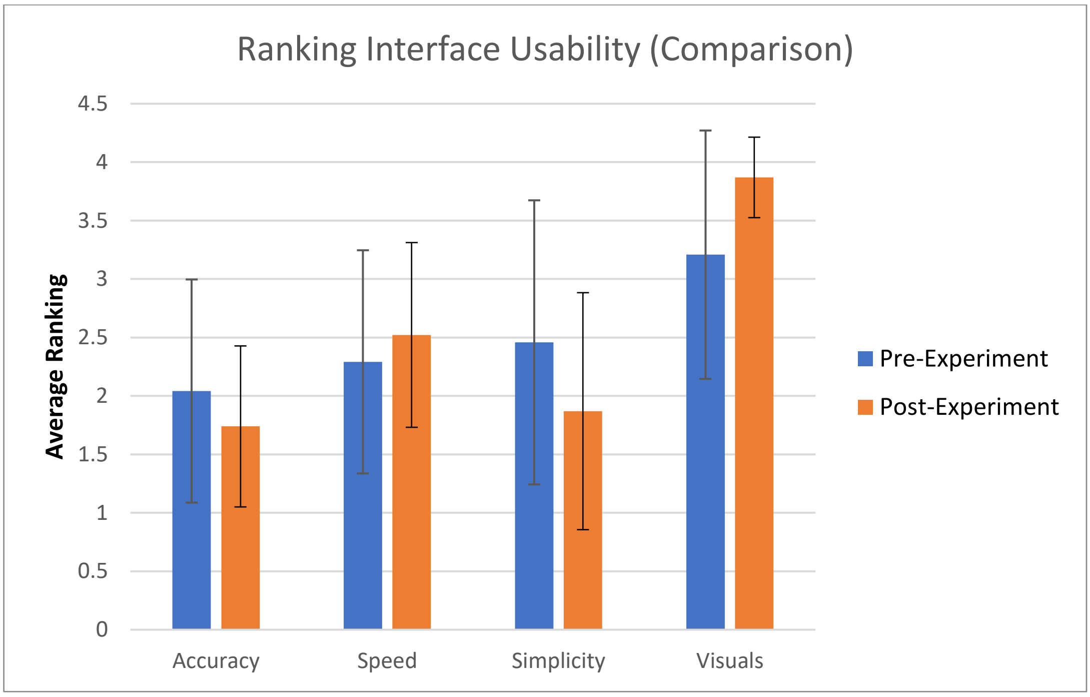


Figure 5.64: Ranking Interface Usability

Figure 5.64 shows the responses from both the pre-experiment questionnaire and post-experiment questionnaire alongside each other. Lower numbers in the responses meant they were considered more important in preference because a rank of 1 was what the participant considered the most important aspect. Between pre-and post-responses, it can be seen the order did not change except in the middle two. The ranking was specifically considering HMDs. Participants thought accuracy of accessing features was the most important feature. The accuracy is followed by the speed of accessing features and the simplicity of interaction required. Overwhelmingly participants agreed visual appeal of the interface was the least important trait.

Additionally, the error bars showing standard deviation show a lower variation for choosing visuals as the least important to usability during the post-experiment versus the pre-experiment. The changes in ranking for accuracy and speed were not statistically significant from performing a one-tail t-Test comparing each of the pre-and post-experiment changes for each attribute. The change in simplicity ranking with a mean of 2.39 in the pre-experiment and 1.87 in the post-experiment was significant t(22)=1.95, p = 0.03. This change suggests a strong preference shift toward simplicity. The change in visuals ranking was also statistically significant, with a mean of 3.26 pre-experiment and 3.87 post-experiment (t(22)=-3.06, p = <0.01), reinforcing that visuals were considered least important.

Device Preferences

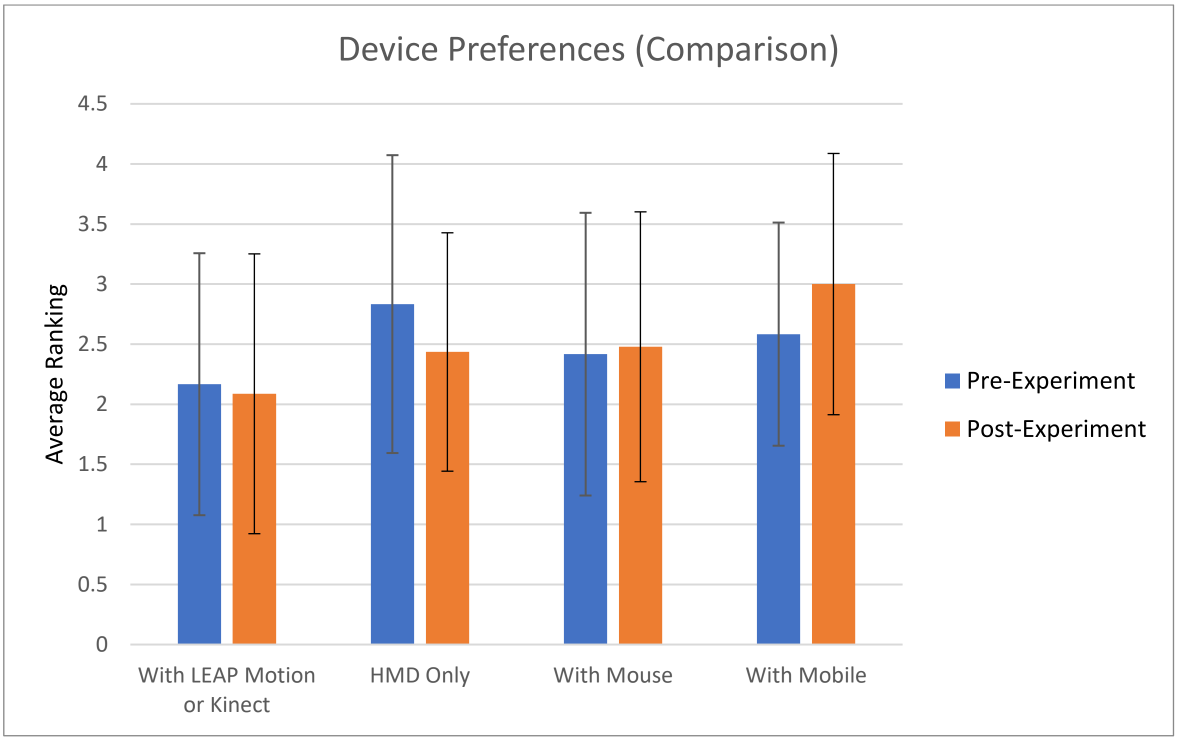


Figure 5.65: Device Preferences

Participants were asked in the pre-and post-experiment questionnaires about their preferences for device use for interaction with HMDs. The LEAP Motion and Kinect sensors were explained as the use of hand gestures for performing interactions. The participant may not have had personal experience with the devices. The original experiment design was intended to test one or both inputs in addition to the rest. Participants ranking the LEAP Motion and Kinect sensors still let them speculate on how they felt about using those devices for the PVMS. As with interface usability, the lower number indicates a greater desire to use the interaction device. Figure 5.65 shows that participants preferred the idea of using their hands for interactions with gestures. Interestingly for these preferences, the option for HMD Only type interaction started as least desired in the pre-experiment questionnaire but shifted to the second most desired in the post-experiment. With a change from 2.83 (SD = 1.24) to 2.09 (SD = 1.16). Comparing each change in ranking with a t-Test did not find the changes were statistically significant except for the increase in preference for “With Mobile” from a mean of 2.57 to 3 (t(22)=-2.15, p = 0.02).

Game Trailer Responses

Participants were asked in the pre-experiment questionnaire about the trailer. It would have been beneficial to include a direct question here to ask if the trailer had been viewed to separate responses from those who had watched it and those who had not. Some participants indicated they had not watched the trailer, and this may have skewed the numbers as they responded as it not having influenced their desire.

When asked trailer’s influence on their desire to participate, participants responded with an average of 5.54 (SD = 2.3, ranking on a Likert scale out of 10). The average response regarding the influence on future participation was 7.38 (SD = 1.2). The data indicates participants felt there was some influence on their decision to participate, but that a trailer may in the future increase their desire more than the impact of this experiment. Comparing the influence of the trailer for experiment two against influence on future participation was statistically significant t(23)=-3.83, p = <0.01.

General Feedback Metrics

Participants were asked if they found the PVMS useful; 20 responded yes, and three responded no, as seen in Figure 5.66 (over). Participants indicated the parts they found enjoyable included: the ease of use, the effectiveness of what it was trying to accomplish, the speed of use, the accuracy, and improvement to immersion through hiding menus. Those who had issues indicated their grievances included: the menu sometimes appearing when they did not want to view it, issues with the sensitivity configuration, and some soreness of the neck from overly aggressive head-turning. Asking whether the PVMS was useful was significant to ask because the audience of participants had all used computing technology in their past. The question relied on users’ understanding of interactive technologies to form their response about whether they found the technique and menu system useful compared to their other technology experiences.

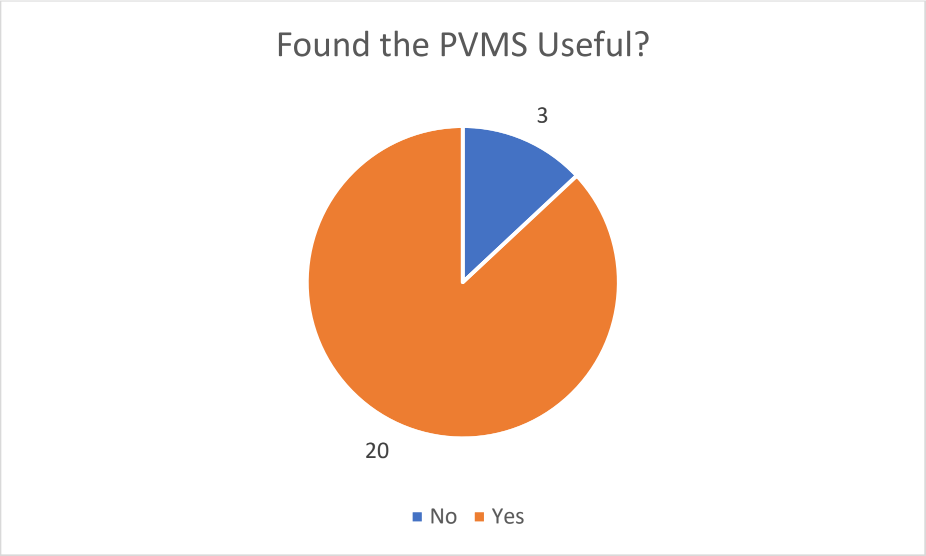


Figure 5.66: Found PVMS Useful?

Figure 5.67 (over) provides some of the general questions related to the overall use of the PVMS. Accuracy was intended to be associated with showing the correct menu. Participants indicated that the menu system showed far more at the correct times than the wrong times. The number of wrong times was reported to be very high, so this was an area to be addressed in the following experiment. Participants responded, indicating the gesture was useful on average, and they were likely to use the system in future. Participants were asked to rate the fatigue they felt while completing the experiment. On average, participants responded with 4 (SD = 1.93, ranking on a Likert scale out of 10), indicating fatigue was mostly on the lower end but meant there was room for improvement. When asked about the experience provided with no additional controller, participants responded with a positive rating of 7 (SD = 1.93).

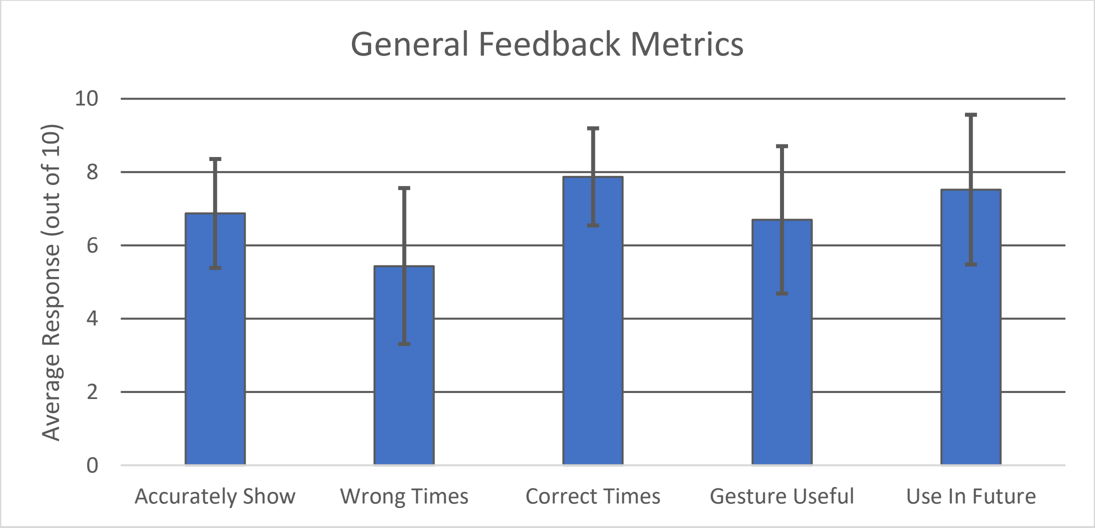


Figure 5.67: General Feedback Metrics

Comparisons between this data using a t-Test indicated they were statistically significant. Wrong Times (mean of 5.43) was statistically significant compared to Correct Times (mean of 7.87) t(22)=-4.1, p = <0.01. Wrong Times was statistically significant compared to Gesture Useful (mean of 6.7) t(22)=-1.9, p = 0.04. Correct Times was statistically significant compared to Gesture Useful t(22)=2.74, p = 0.01. Comparing the responses to Accurately Show (mean of 6.87) against Gesture Useful was not significant, but against Wrong Times was significant (t(22)=2.17, p = 0.02), and against Correct Times was significant(t(22)=-3.73, p = <0.01).

System Usability Scale

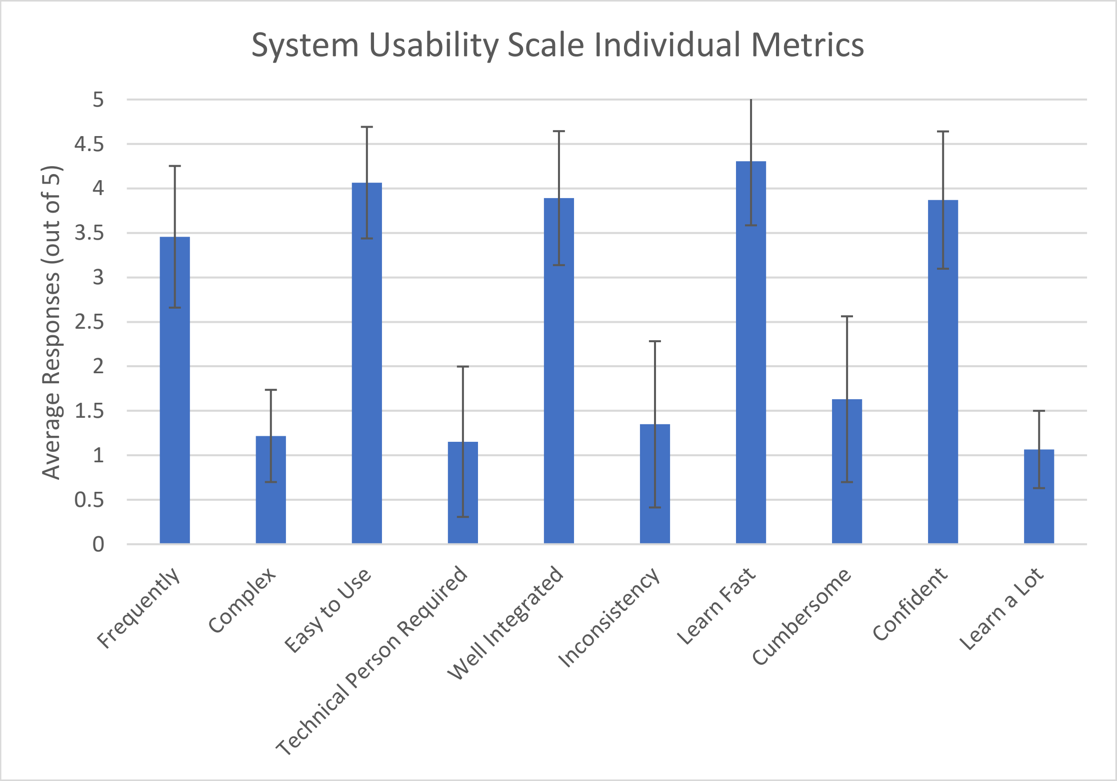


Figure 5.68: System Usability Scale Individual Metrics

As part of the collected data were questions related to the System Usability Scale (SUS). Five of the questions assess positive aspects of a system, and five assess the negative aspects. They are combined to calculate a score out of 40 that can then be converted to a score out of 100. Figure 5.68 shows the results from the questions. The values are ordered by how they appeared in the questionnaire.

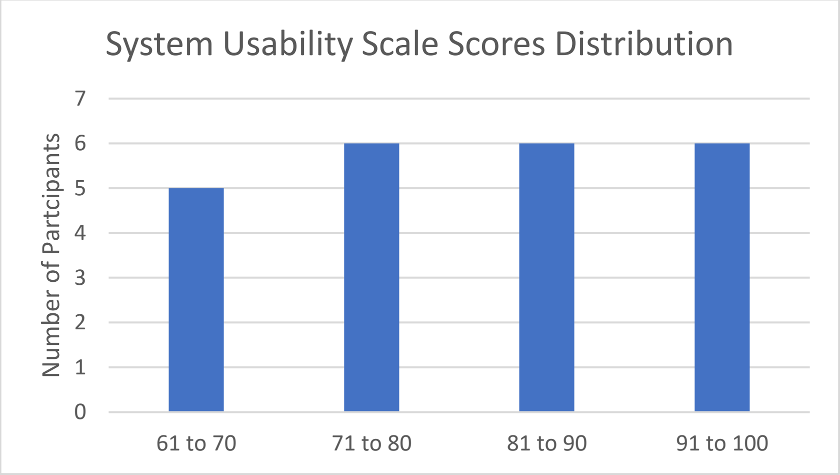


Figure 5.69: System Usability Scale Score Distribution

The overall score result from the SUS was 82.17 (SD = 11.26). In terms of descriptors used when talking about the SUS, this would be rated as an A grade, deemed excellent, acceptable, and a promoter for the Net Promoter Score (NPS). The distribution of scores is shown in Figure 5.69, with an equal distribution mostly between 61 to 100. Additional analysis comparing the SUS results against other data in this experiment can be found in 5.3.3.

System Use Preferences

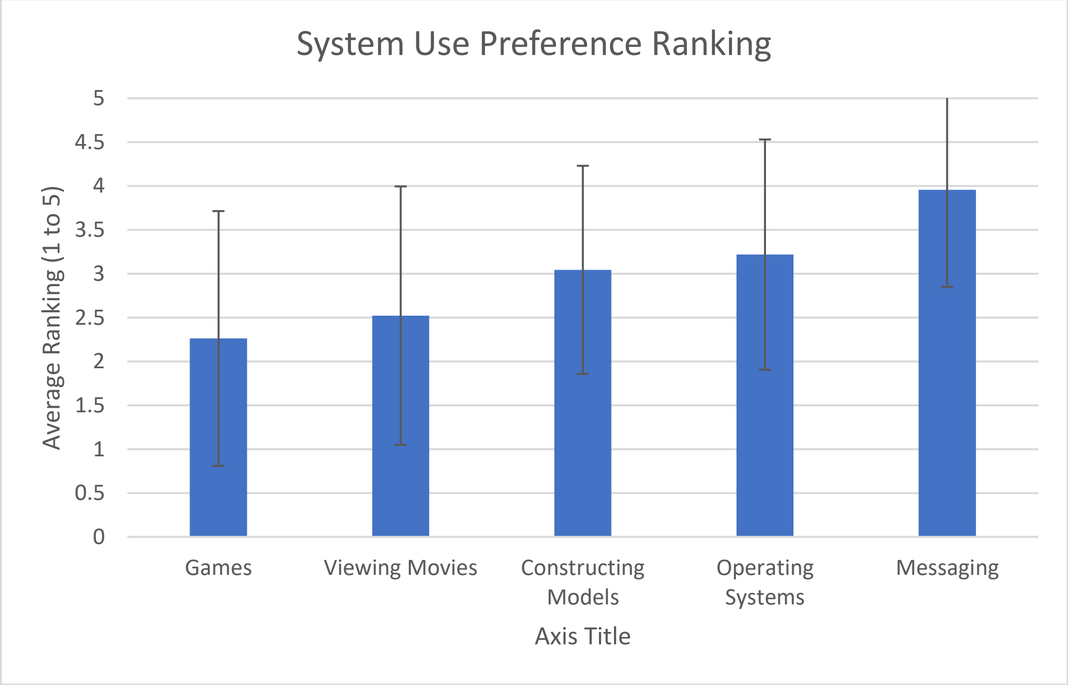


Figure 5.70: System Use Preference Ranking

Figure 5.70 shows the ranking of where participants would prefer to use the PVMS. Lower numbers indicate a stronger preference. Games were the highest preferred use, with messaging the least preferred. In hindsight, after completing all the experiments, this line of questioning to participants may have better focused on individual types of interactions instead of system use. Although these systems all incorporate many different menu tasks, this information is still useful to observe. It may be considered based on preferences where participants would be willing or interested to use most functionality via the PVMS.

5.3.2 Application Data Results

This section will cover application data results that have been derived from the logs collected during the experiment. The tables of data that were generated have been summarised in section 5.2.11. The results have been grouped based on the types of information.

PVMS Use

Different tasks provided different experiences for the choice of opening a menu on the left vs the right. Figure 5.71 shows the total occurrences of menus by direction. The main menu, construction task, and periphery task provided no difference in what would occur from the left or right interaction. In an ideal case, the sensitivity task results would be equal for both directions, and any difference would be due to mistakes. The tower defence game menus were distinctly different, with most associated operations being triggered by a right movement. The difference was very similar otherwise between left and right use.

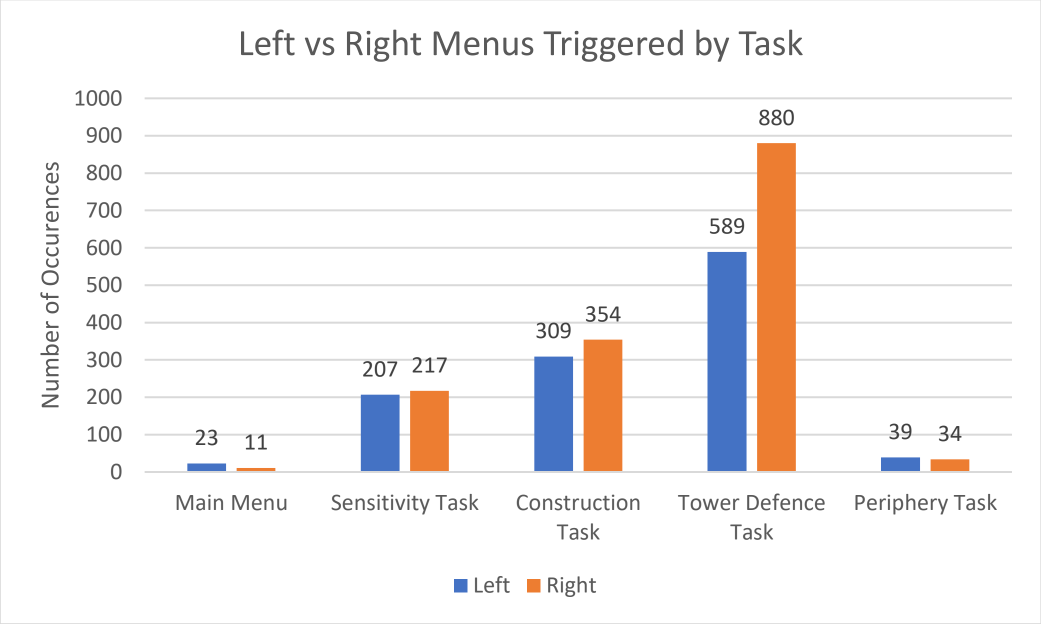


Figure 5.71: Left vs Right Menus Triggered by Task

Figure 5.72 shows two-thirds of the difference data used when checking for thresholds, as described in the technology overview chapter. The difference on the Y-axis is not as useful to look at because the threshold worked as a minimum leading to most values appearing slightly over 25. `Diff_X` refers to the vertical angle of looking up and down (also known as pitch). `Diff_Z` is the rotation from tilting the head (also known as roll). This data demonstrates the maximum thresholds for the sensitivity settings could be lowered further to reduce unnecessary menu creation. The tower defence task involved far more looking around than the other tasks where the focus was on a smaller region and is believed to be the cause for the large average angle difference. When generating the difference data, an average history cache size was around 22.66 (SD = 2.21) elements.

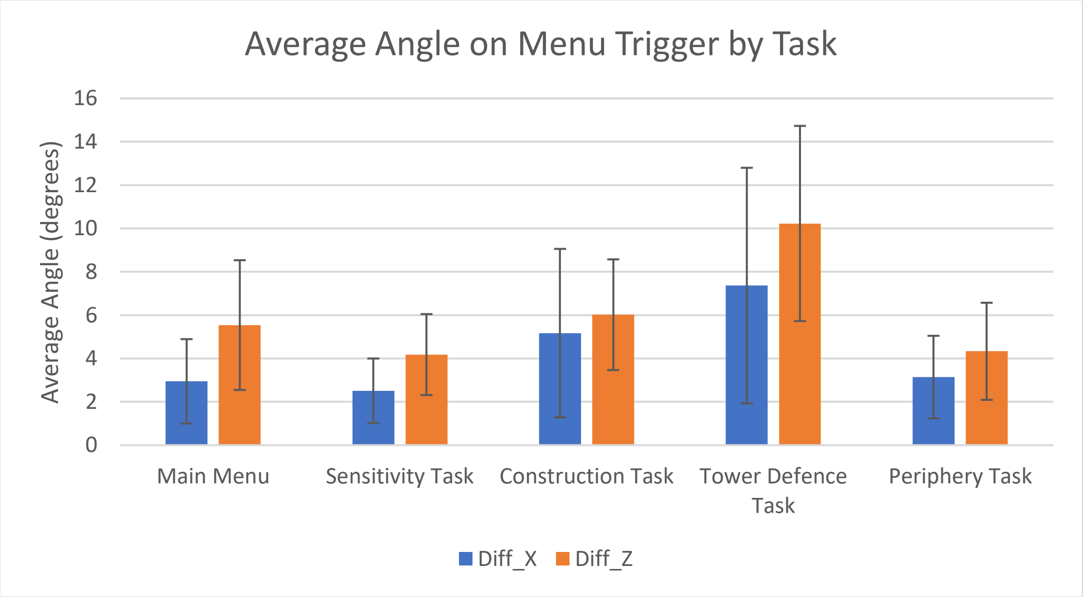


Figure 5.72: Average Angle on Menu Trigger by Task

Menu Interactions

Figure 5.73 (over) shows the events that caused the PVMS to close. A "Button Event" indicated any interaction with elements in the menu. "Hide From Inactivity" was triggered automatically after a period of 4 seconds if the participant did not at least hover within the menu's region. "Replaced While Active" means the menu was replaced with another menu from another PVMS trigger. Some of the causes for these last two included: participants would trigger menus to check options during downtime, participants would not decide in time, participants would decide they wished to perform a different action, or the menu was wrongly triggered when they did not mean to for their current situation. The menu sensitivity test did not require any button interaction once the correct menu was revealed, leading to the high numbers of these occurrences once participants realised this. The events during the tower defence task are believed to result from pressure to perform well during the game and could be attributed to how participants responded regarding the number of wrong times the menu appeared during use (see Figure 5.67).

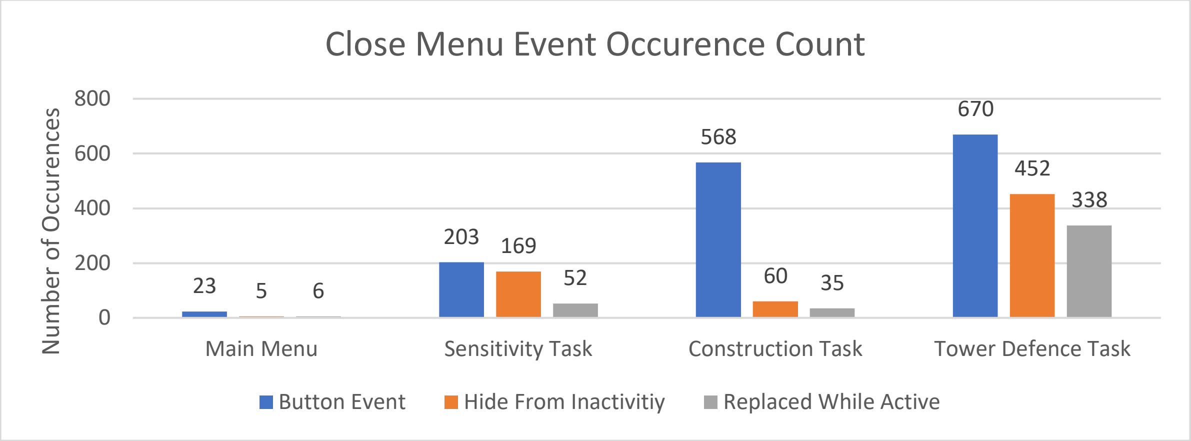


Figure 5.73: Close Menu Event Occurrence Count

In Figure 5.74, the average time menus were open is shown. The events causing the closing of the menu are those listed in Figure 5.73. Importantly the time to select was set at 1.5 seconds which inflates the time on successful interactions for the purpose of testing. The average time for closure from inactivity was over 4 seconds in all cases. Indicating the menus were sometimes interacted with and then left to close with no button option chosen.

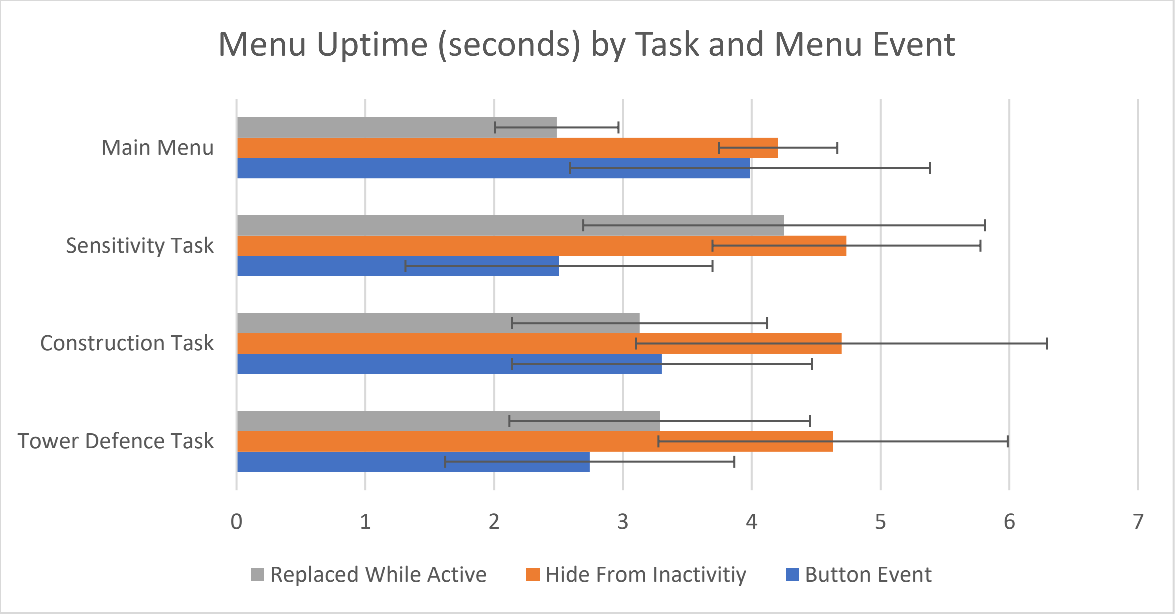


Figure 5.74: Menu Uptime (seconds) by Task and Menu Event

Task Completion Time

Figure 5.75 shows the average time it took participants to complete the tasks. The tasks mainly were participant-driven for how fast completion could occur. The length of time spent observing the in-application tutorials has also been included. The efficiency of their strategy determined the duration of time spent completing the tower defence task. Defeating all enemies quickly without letting any through defences would indicate they selected more appropriate towers building on the strengths of their selections during the construction task.

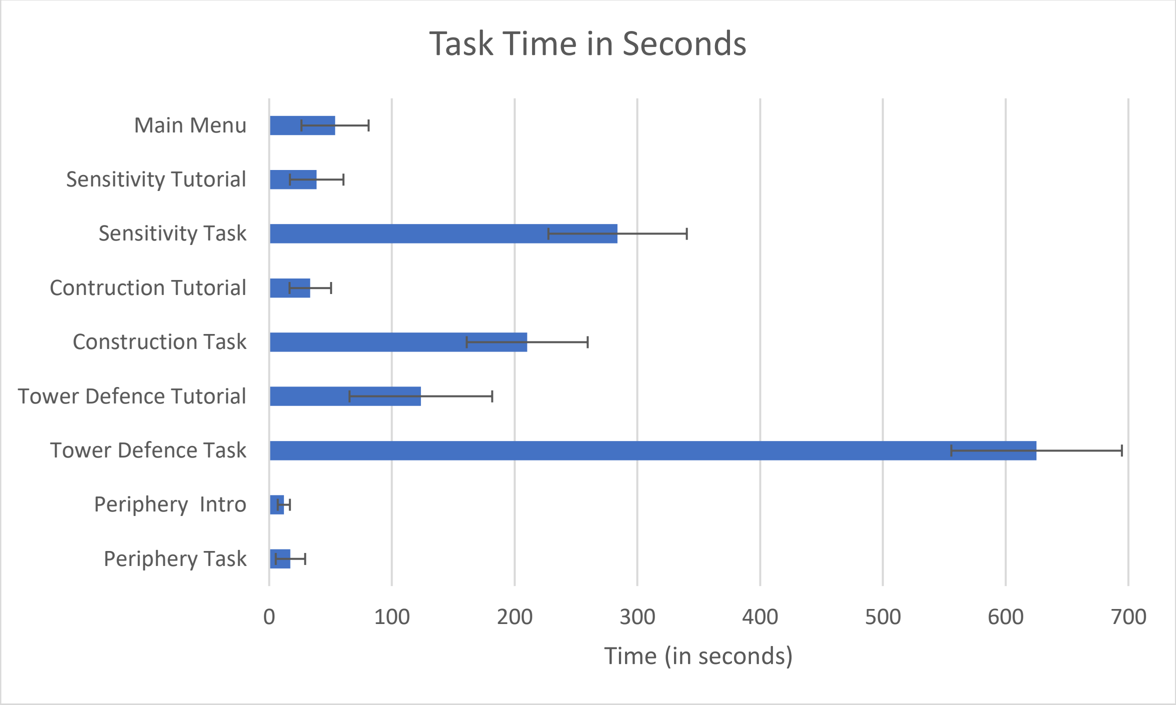


Figure 5.75: Task Time in Seconds

In Application Responses

After each task, participants were asked about their thoughts regarding the PVMS's usability for the task they had just completed. Figure 5.76 (over) shows the responses to these questions. The questions related directly to the type of task. Questions for each type of sensitivity for the Menu Sensitivity Task, an untimed task question for the construction task, a timed question relating to the time pressure of the tower defence game and finally, the periphery preview example. Participants slightly preferred the medium sensitivity and slightly preferred the usability for untimed tasks. Responses to these questions were, in general, all positive. Comparing the pairings of low/med/high and untimed/time with a t-Test found no statistical significance between the categories of questions.

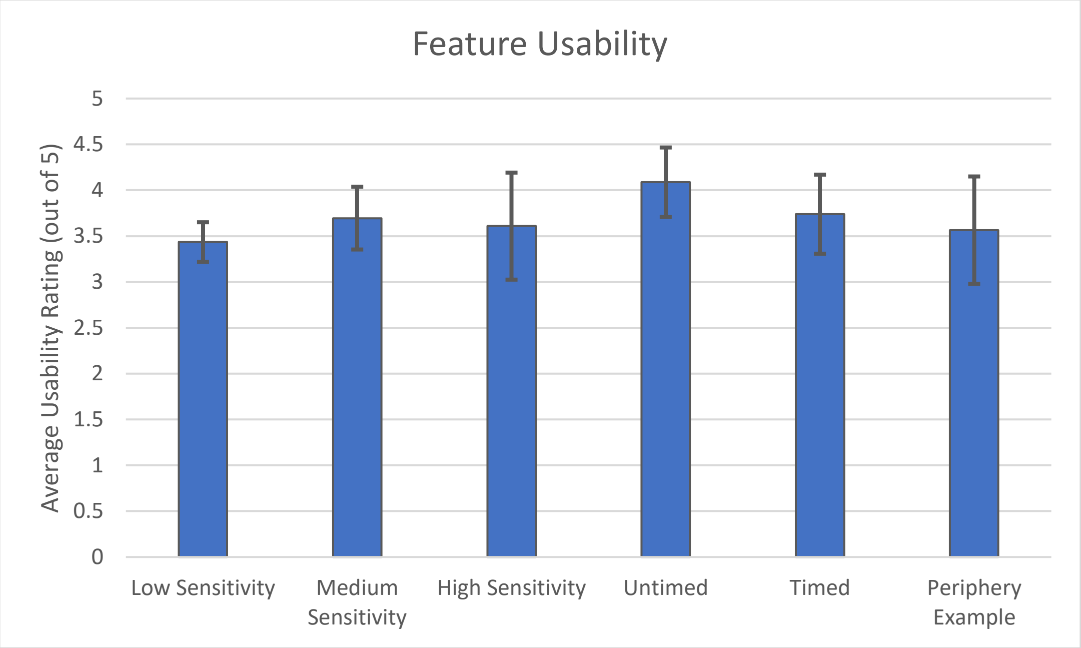


Figure 5.76: Feature Usability

Menu Sensitivity Task Data

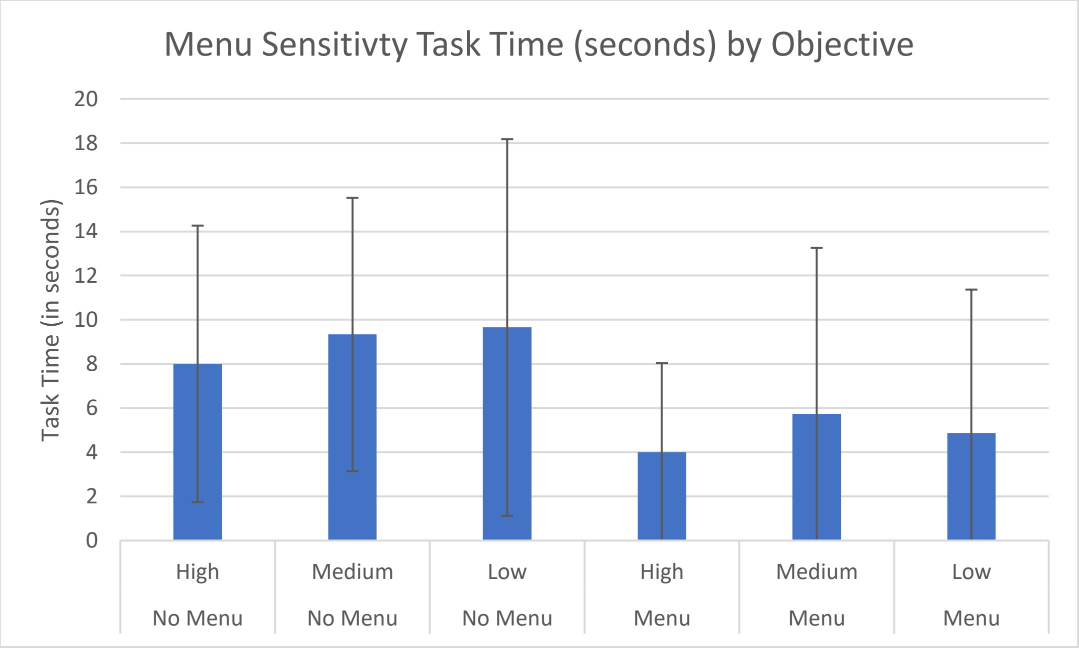


Figure 5.77: Menu Sensitivity Task Time (seconds) by Objective

Figure 5.77 shows the average completion times for the different types of tasks. Each participant would have completed the tasks four times, which relates to the combinations of left twice and right twice head interactions. The higher completion time when no menu had to be revealed was largely down to a combination of mistakes from turning too quickly and becoming overly cautious after making mistakes. For all tasks, the time was increased from determining what the next task was. Figure 5.78 shows all these mistakes made during the experiment. The revealing no menu had the most errors, with PVMSs accidentally or incorrectly revealed. There was also a case of looking toward the wrong sides (Left Panel/Right Panel). Errors when the goal was to reveal the menu were more varied, with cases of the wrong menu being displayed and cases where the participant went too far and selected the left or right panel instead.

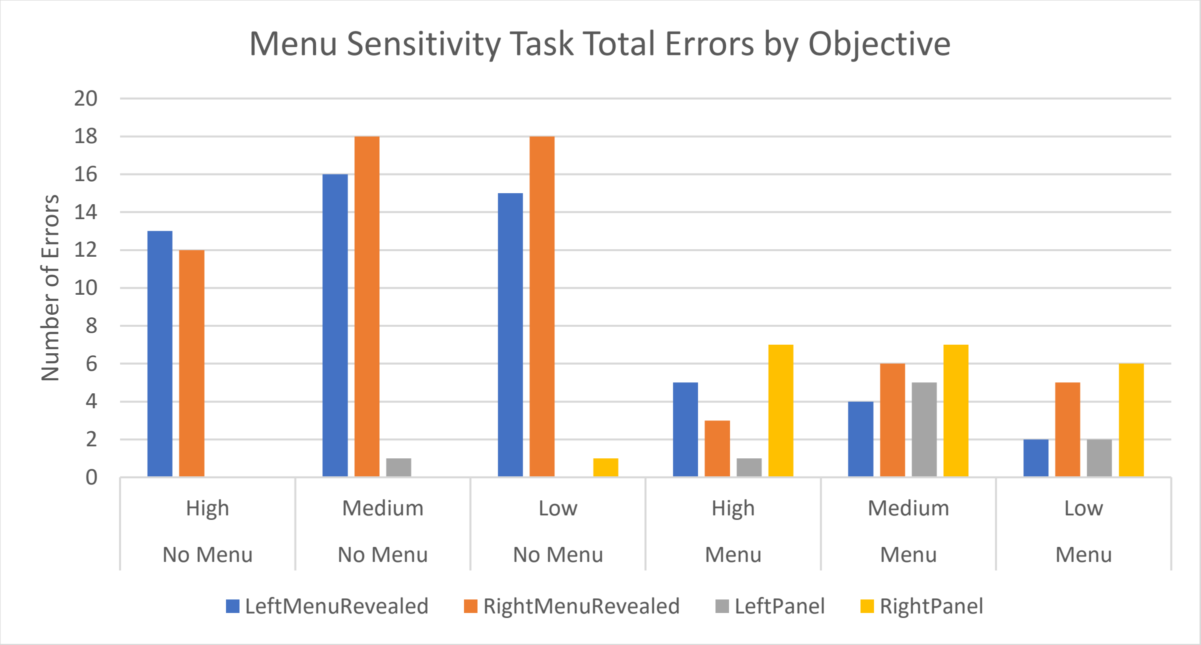


Figure 5.78: Menu Sensitivity Task Total Errors by Objective

Tower Construction Task Data

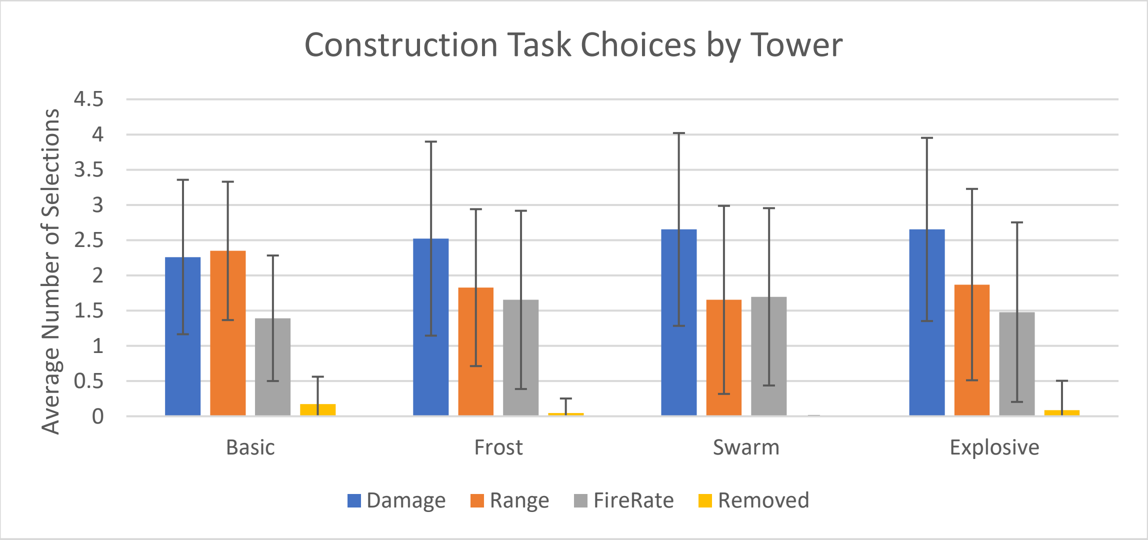


Figure 5.79: Construction Task Choices by Tower

Figure 5.79 (previous page) shows the average choices of modifiers applied to each tower. The Basic tower had a nearly equal selection between increasing damage and range options, while the other three had participants prefer choosing the damage modifier. Participants hardly used the "Remove" menu option.

Tower Defence Task Data

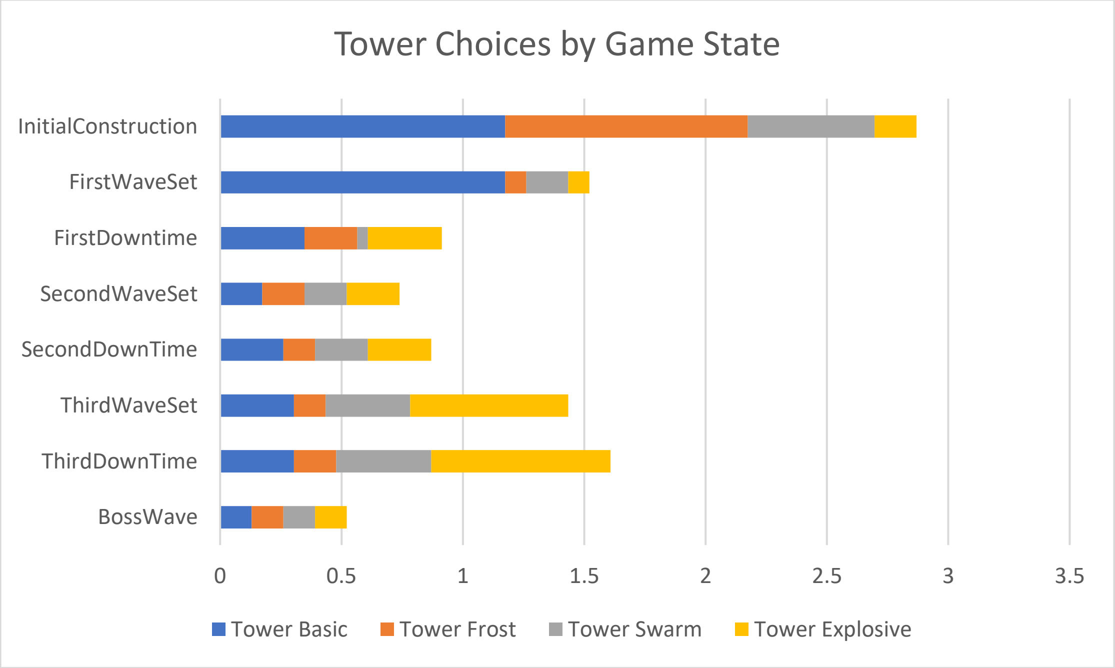


Figure 5.80: Tower Choices by Game State

Figure 5.80 shows how participants were choosing to build their strategy while completing the tower defence game. Most participants started by building at least one basic tower and one frost tower during the initial construction phase. Then one basic tower normally during the first wave of enemies. After this choice, there was a lot of variances as they began spending their gold on repairs. Toward the end game, many participants began building the more expensive explosive towers, showing an interesting dynamic in the shift from starting with cheap towers and moving toward purchasing the more costly towers late in the game.

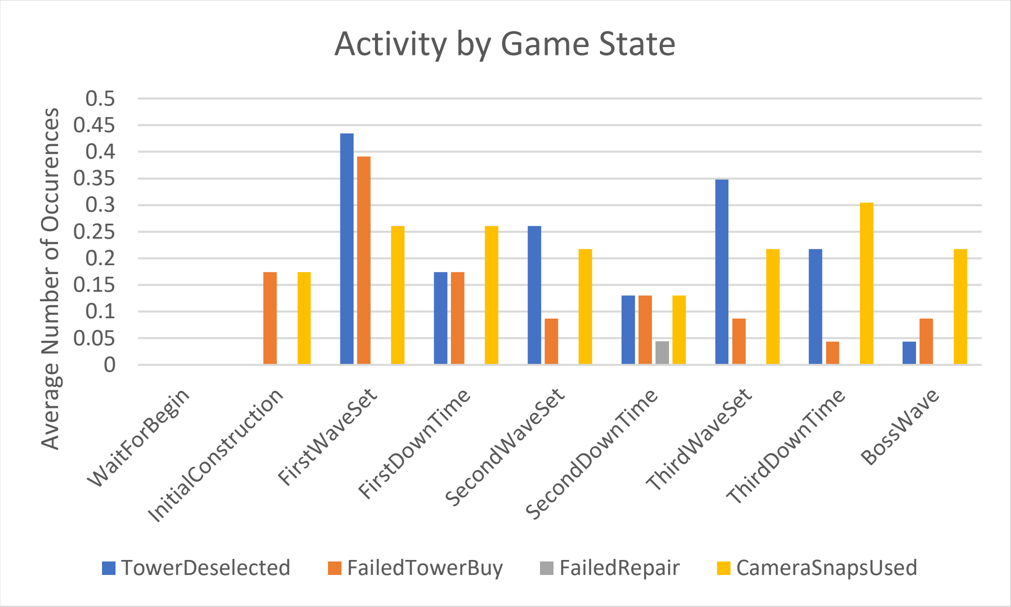


Figure 5.81: Activity by Game State

Figure 5.81 shows some of the average activity related to user actions. These were all very low, indicating that not many participants engaged with positive actions such as tower deselection via PVMS or camera snap nodes (represented as CameraSnapsUsed). The “Failed Tower Buy” and “Failed Repair” events would occur when the participant had insufficient funds, but they attempted to make the action occur regardless.

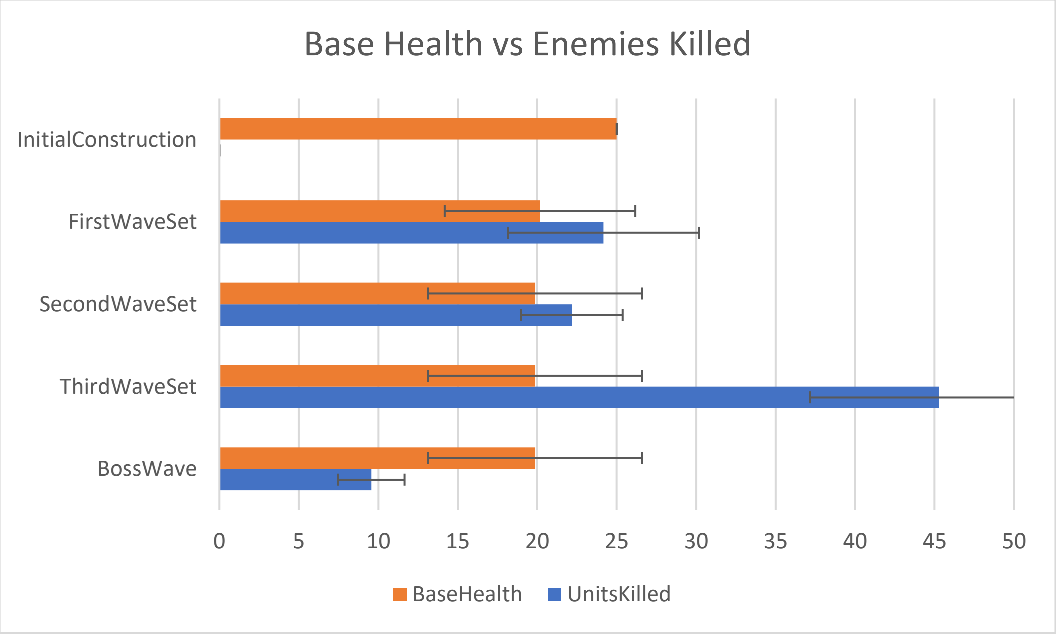


Figure 5.82: Base Health vs Enemies Killed.

Figure 5.82 shows some values related to the performance of participants. Base health would start at 25, so most participants would, on average, fail to defeat 5 of the enemies during the first wave, but then stabilise and, on average, defeat most if not all remaining enemies. From this data and the information shown in the previous figures, it is clear that the difficulty needed to be increased slightly for the average participant if this were a product to be released commercially. As the focus was on using the PVMS, the challenge to the participant could be considered equivalent to a tutorial level and is appropriate for them to understand the PVMS without being overwhelmed with potentially unfamiliar gameplay.

5.3.3 Result Analysis

This section will consider the statistical significance of data by comparing it based on the categorical data provided by participants in their questionnaire responses. For analysis comparing this experiment against the other experiments, see chapter 7.

SUS Scores against Age

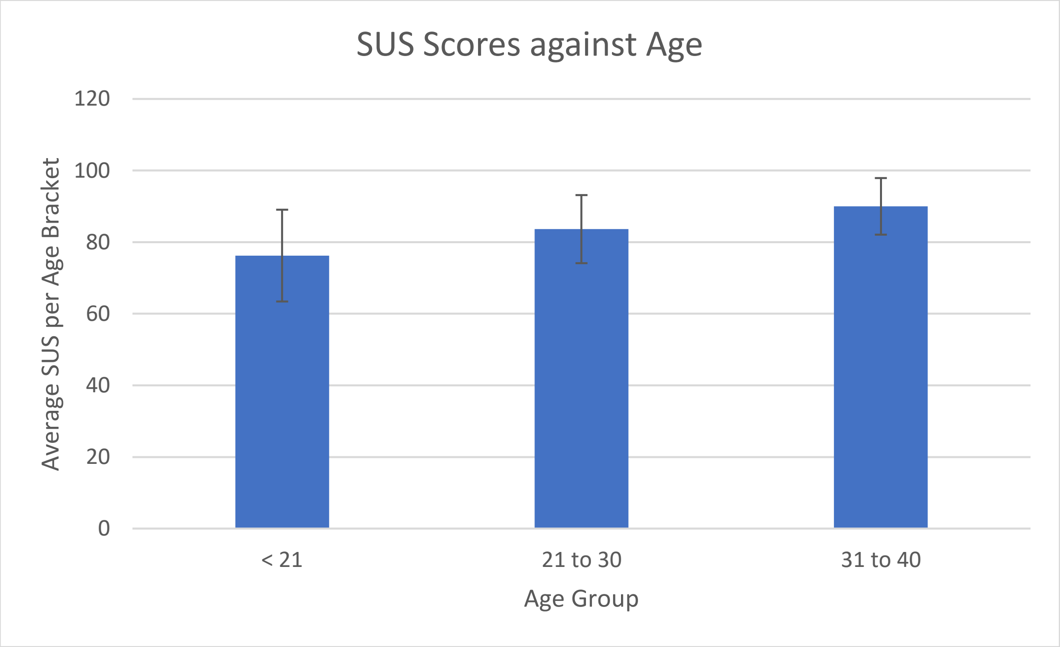


Figure 5.83: SUS Scores against Age

The average SUS scores for each age bracket are seen in Figure 5.83. An ANOVA test comparing the categories did not find any statistical significance between the results of different age groups. The less than 21 age group had a mean of 76.25 (SD = 12.82, n = 8), the 21 to 30 age group had a mean of 83.64 (SD = 9.51, n = 11), and the 31 to 40 age group had a mean of 90 (SD = 7.91, n = 4). A comparison against genders was not practical compared to the age, as there was only a single female participant.

SUS Scores against Computer Use

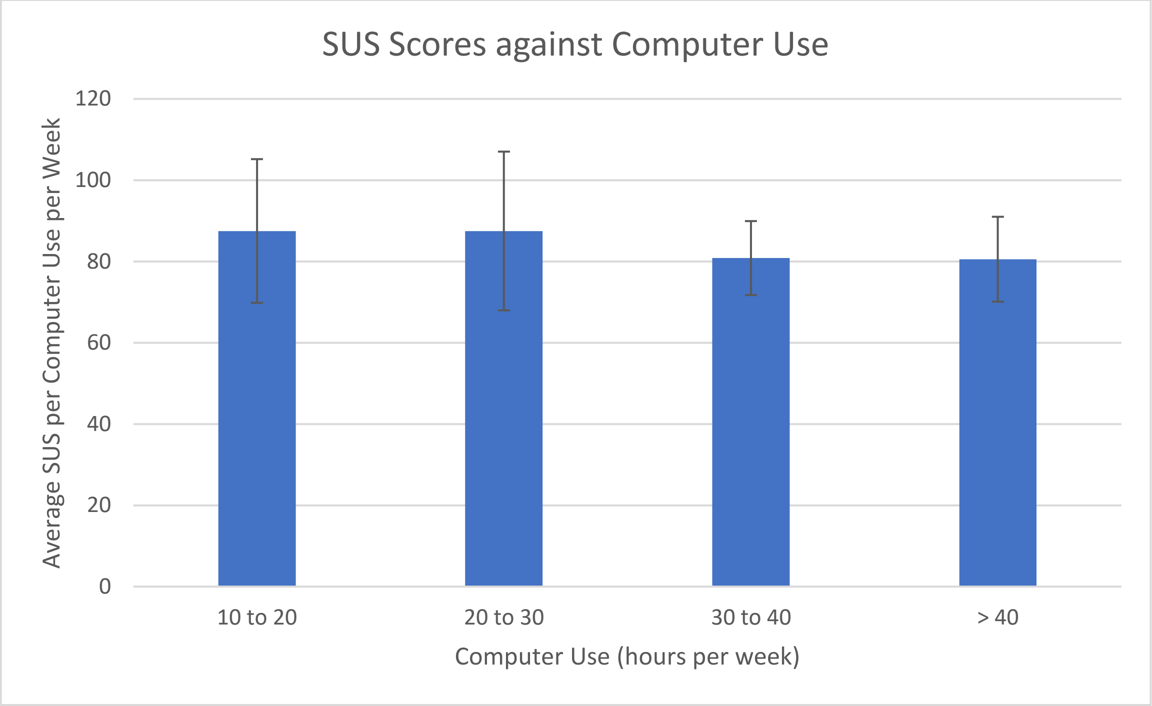


Figure 5.84: SUS Scores against Computer Use

Figure 5.84 shows a breakdown of the SUS scores averaged for categories based on participants computer use (in hours per week). No statistical significance was found for these categories when performing an ANOVA test. The 10 to 20-hour group had a mean of 87.5 (SD = 17.68, n = 2), the 20 to 30-hour group had a mean of 87.5 (SD = 19.53, n = 3), the 30 to 40-hour group had a mean of 80.83 (SD = 9.1, n = 9), and the greater than 40 hours group had a mean of 80.56 (SD = 10.44, n= 9).

SUS Scores against Game Use

Figure 5.85 (over) shows average SUS scores for each category of average game use (in hours per week). No statistical significance was found from comparing the categories using an ANOVA test. The less than 10 hours group had a mean SUS of 84.77 (SD = 11.7, n = 11), the 10 to 20 hours group had a mean of 86 (SD = 9.45, n = 5), the 20 to 30 group had a mean of 74.17 (SD = 3.82, n = 3), the 30 to 40 group had a mean of 65 (SD = 0, n = 2), and the greater than 40 hours group had a mean of 87.5 (SD = 7.07, n = 2).

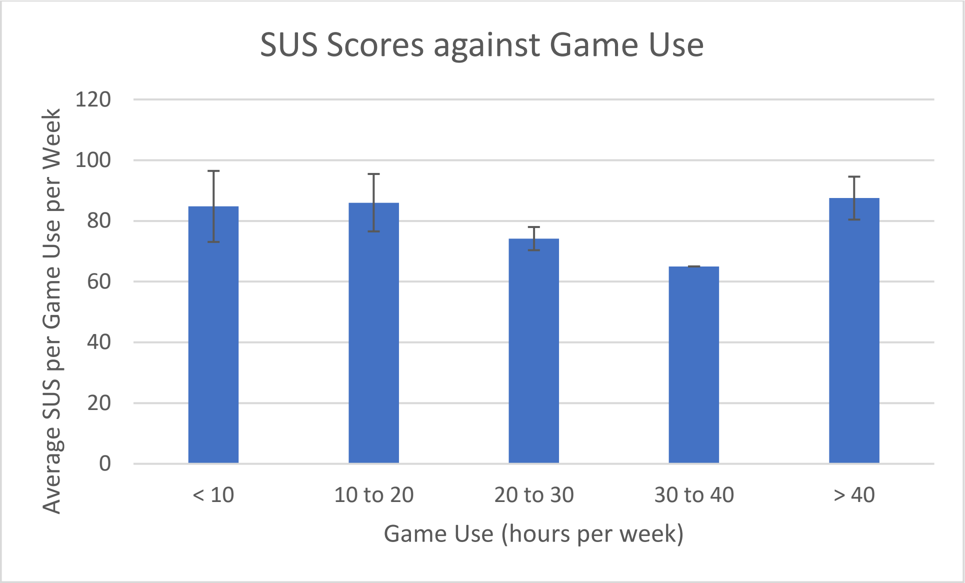


Figure 5.85: SUS Scores against Game Use

SUS Scores against Untimed Response

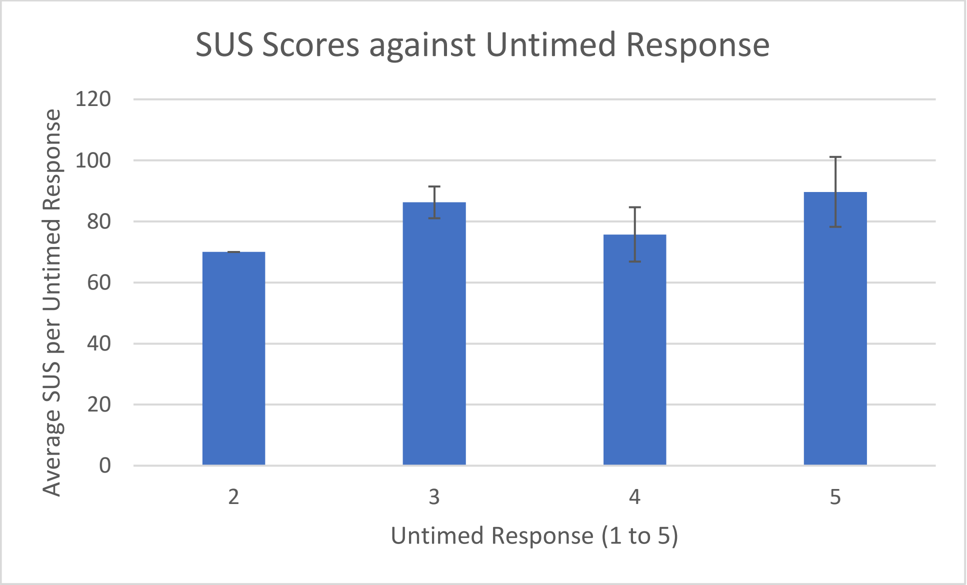


Figure 5.86: SUS Scores against Untimed Response

Participants were asked how they felt using the PVMS for Untimed activities (such as the Tower Construction task). Figure 5.86 categorises responses from how participants responded to show average SUS scores relative to how participants responded to the question. An ANOVA test comparing based on the categorisation showed statistical significance based on the rating given for Untimed and the SUS score (F(3,19) = 3.99, p = 0.02). Participants responding to Untimed with 2 out of 5 had a SUS mean of 70 (SD = 0, n = 1), for

3 out of 5 had a mean of 86.25 (SD = 5.2, n = 4), 4 out of 5 had a mean of 75.75 (SD = 8.9, n = 10), and 5 out of 5 had a mean of 89.69 (SD = 11.45, n = 8).

SUS Scores against Timed Response

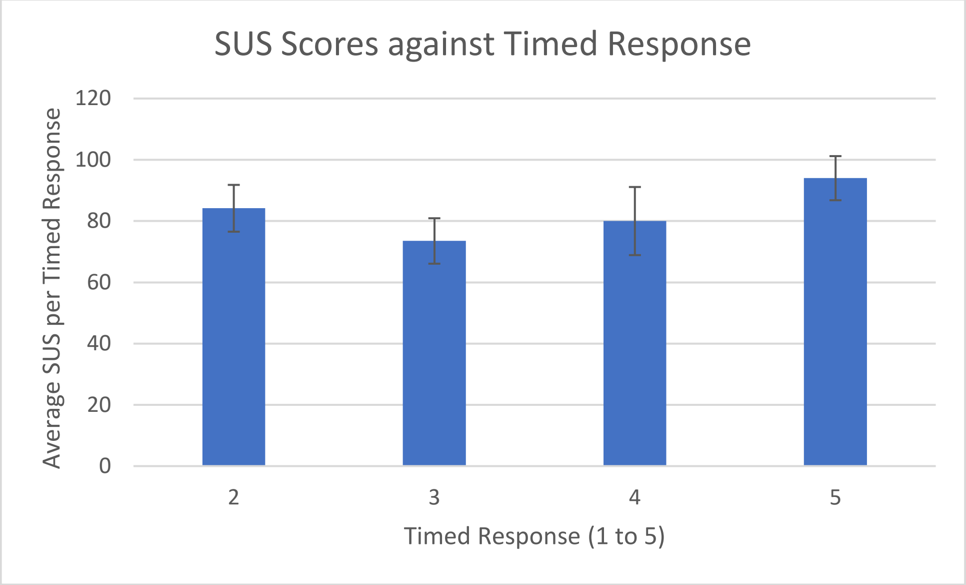


Figure 5.87: SUS Scores against Timed Response

Similar to the Untimed comparison just presented, the Timed situation question response was also evaluated, as shown in Figure 5.87. The Timed responses scaled to make them comparable to SUS scores with a multiplication of 20 and evaluated with a t-Test showed the data comparison was statistically significant (t(22)=1.93, p = 0.03). The data for the responses were also evaluated using the categories with an ANOVA test that also showed it was statistically significant (F(3,19) = 4.34, p = 0.02). Participants who responded with 2 out of 5 had a mean average SUS score of 84.17 (SD = 7.64, n = 3), 3 out of 5 had a mean of 73.5 (SD = 7.42, n = 5), 4 out of 5 had a mean of 80 (SD = 11.12, n = 10), 5 out of 5 had a mean of 94 (SD = 7.2, n = 5).

SUS Scores against Gesture Useful Response

Figure 5.88 (over) shows a plot of the SUS scores against the Gesture Useful question responses. The Gesture Useful scores were scaled with a multiplication by 10 to make them the same scale as SUS scores. A t-Test comparing the scaled values against SUS scores found the comparison was statistically significant t(22)=3.77, p = <0.01. Comparisons were also made to compare for the categories of age, computer use, and game use for each of the

Accurately Show, Wrong Times, Correct Times, and Gesture Useful responses, but no statistical significance was found.

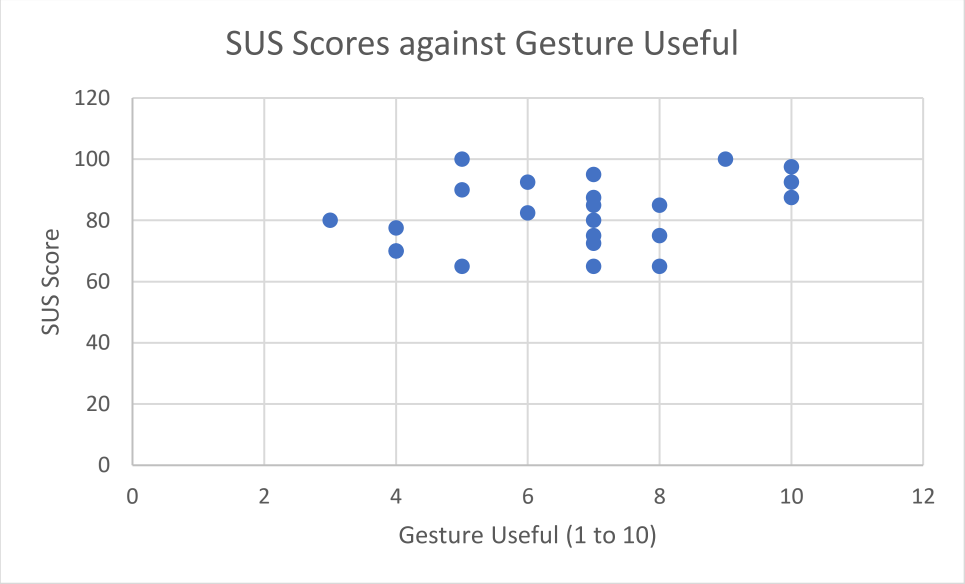


Figure 5.88: SUS Scores against Gesture Useful

5.4 Summary of Results

In this section, the focus returns to the goals of this experiment. The relevance of this experiment's results will be compared to the other experiment's results in Chapter 7. In section 5.1, the experiment overview outlined the following goals for each of these goals. This section will briefly draw from the many individual results to show the most relevant information.

1) Determine whether a PVMS is viable as a tool for interaction within this application and more broadly for other applications.
2) Determine whether improvements could be made to the proposed menu system in the way it is calibrated.

In addition to these two goals, research questions RQ1, RQ3, and RQ4 were investigated as part of this experiment. Following the discussion about how the experiment's goals were met, a discussion linking to the research questions is provided.

- **RQ 1:** How can head rotations, captured through the orientation sensors of a HMD, be used to increase the user interaction capabilities with virtual worlds?

- **RQ 3:** Can a hidden menu provide a viable and useful tool for interaction with head movement as the mechanism for revealing it?
- **RQ 4:** Does a hidden menu revealed and interacted with using the head provide a useful experience compared to menus appearing at a fixed place in a virtual world?

Goal 1 Viability of the PVMS

The primary goals for this experiment were to determine if the experience worked with the prototype menu system. While at the same time, providing a testing system that would promote participants to engage and suggest answers directed at improving the overall experience. The viability of the system can be drawn out from a selection of the results. The first question on the post-experiment questionnaire was one of the more binary responses used. Figure 5.66 in the results section showed the overwhelming number of participants who found the PVMS useful.

Figure 5.67 in the results section showed a few different metrics related to the menu system specifically. The perceived proportion of wrong times for showing the menu was always going to be reasonably high. The sensitivity settings were left less constrained so that capture of data useful to calibration could be used. Additionally, the impact of not having the option to dismiss the incorrectly shown dialogs immediately increased the likelihood that participants would respond negatively in this regard. The most important features of this figure were the showing at correct times being rated highly and the desire to use in future falling just behind. These indicate interest from the participants and some viability in the technique.

The results shown in Figure 5.76 from the results section came from the questions posed to participants immediately following each task as part of the application. All the responses to these questions provided positive feedback on the tasks. Specifically, participants felt the menu system worked best in the untimed scenario where they were asked to construct the towers.

The SUS distribution in Figure 5.69 of the results section showed that all the scores for the system overall were very high. The SUS scores provided additional positive indications for the viability of the system.

As a final point on the system's viability, Figure 5.65 in the results section showed the shift in preferences by participants between the pre and post-experiment questionnaires. Participants began by indicating on average that they would least prefer the HMD Only interaction. These scores were ranks where a 1 was the most preferred meaning lower numbers would be better from the participants' perception. The HMD Only type interaction shifted from the last place to second place when comparing the averages. Therefore, this shift in preference can be taken as a vote of confidence in the ability to provide HMD Only type interactions utilising menu techniques like those shown in this experiment.

Goal 2 Improvements for the PVMS

All the questions and types of data collection were designed to derive evidence toward the system's effectiveness, usability, and viability. Some aspects of the data collected were more valuable than others for directly addressing the goal of improvement. To summarise how this goal was met, there are two significant areas to look at for evaluation. The first is the sensitivity configuration, and the second is to look at the descriptive feedback provided by participants.

The data shown in Figure 5.72 in the results section provided useful information for modifying the sensitivity configuration directly. An average interaction threshold and standard deviation provide unbiased data about how the system was used. The maximum thresholds were far above those shown in the figure. Indicating they could be reduced to somewhere closer to these values. For experiment 3 (to be discussed in Chapter 6), the calibration was kept the same as the medium setting to allow direct comparison between the two experiments. Participants most commonly experienced the unnecessary menu triggers during the tower defence game. As seen from the figure, the values for the tower defence game were higher than the others.

Table 5.4 (over) shows a summary of the enjoyable and difficult features participants reported. The enjoyable features show many of the good merits of this menu system for how it was used during the experiment. The enjoyable features are good for validating the system's viability from positive feedback, but it is important to identify how the difficult features impact negatively. Menus appearing at the wrong times was always going to happen to some extent, with the sensitivity configuration deliberately left more responsive than necessary.

Participants were all experiencing the system for the first time during this experiment, which meant there was also a small learning curve to using the menu system's triggering via the gesture for all participants as they progressed through tasks. Not having a way to remove menus quickly and the impact of incorrectly opened menus was addressed in experiment three as a major change point. The selection speed was left very long at 1.5 seconds to ensure novice participants could not make mistakes easily. This duration for the selection timer could be modified based on user preference, thereby addressing the participants' concerns. The size of the interface was left very large to counter any issues with readability from within the HMD interface. As resolutions of screens inside HMDs continue to improve, this is less of a problem. Most participants did not have this type of issue with reading. Difficulty reading did not appear to impact participants' actions during the tasks significantly. Higher physical demands will be an issue when using an interaction technique driven by a HMD, as they inherently assume some form of head movement. The demands of HMD use can be contrasted against staring at a traditional computer screen and just using slight hand movement to perform actions. The immersive world and interactive tools contribute to the direct usefulness of VR and make the demands a worthwhile trade-off.

Table 5.4: Enjoyable vs Difficult Features

Enjoyable Features	**Difficult Features**
• Ease of use. • Menus out of the way (unobtrusive) and only visible when required. • Quick, responsive and accurate. • Good for hiding complexity. • Immersive and natural interaction. • Fun.	• Menus appearing at wrong times (too sensitive). • No way to quickly remove incorrectly opened menus. • Speed of selection. • Reading of menu options (difficulty with eyesight – perhaps needed a different lens). • Higher physical demands than when using traditional input devices.

5.4.1 Research Question Discussion

RQ 1: How can head rotations, captured through the orientation sensors of a HMD, be used to increase the user interaction capabilities with virtual worlds?

This experiment has demonstrated validation of the PVMS as a technique that can increase the number of available interaction tools available to developers. The approach harnesses the inherently available information from the mechanism the headset uses to determine its own orientation. By utilising only the information that could be assumed as part of any HMD available as a consumer product, the likelihood of using the technique more broadly increases.

RQ 3: Can a hidden menu provide a viable and useful tool for interaction with head movement as the mechanism for revealing it?

The PVMS iterated from the fixed in place menus, demonstrated as part of the first experiment, and provided a menu experience that presented contextual menus wherever it was necessary. These menus remained hidden until the PVMS determined that they should be shown using the gesture type mechanism described throughout this chapter and Chapter 4. The PVMS has been demonstrated as viable and useful through the experiment tasks, demonstrating the future potential for iteration of this system to make improvements as described in the discussion on Goal 2. This was further explored after the development iteration of the PVMS tool for the third experiment, covered in Chapter 6.

RQ 4: Does a hidden menu revealed and interacted with using the head provide a useful experience compared to menus appearing at a fixed place in a virtual world?

The previous discussion concerning RQ3 shows that the fixed in place type menus do have their place. Some of the fixed in place menus used for this experiment used for contrast include the tutorial interfaces, post-task feedback dialog, and many of the status dialogs. The PVMS did provide a useful experience compared to the fixed in place elements. It does not entirely replace the need for these types of interfaces when used appropriately. The purpose of interfaces and their elements should be considered when determining where it is appropriate to present the information and actions to a user. Placing fixed elements in an expected location when the user is not expected to stray from that location, as seen in the tutorials, makes sense to use an element fixed in place. For menus, like those demonstrated with the PVMS for making contextual actions that can change depending on the situation or are subject to other factors, the benefits of revealing a menu that takes this into account is clear.

5.5 Conclusion

This chapter provided a detailed explanation of the many systems that went into the methodology of the second experiment. The experiment provided four different tasks designed to test specific aspects of functionality for the PVMS. From the results, the tasks were received positively and demonstrated validity for this research. The feedback taken during this second experiment was combined with ideas for future improvements and developed through iteration for the following experiment: experiment three. The specifics of how the PVMS worked in experiments 2 and 3 have been left to the Technology Overview chapter (Chapter 4). As with this chapter, the next chapter (Chapter 6) will focus on the experimentation used to test the refined system and the fitness of the prototype PVMS for use.

6 Third Experiment: Improving on Periphery Vision Menus

The third experiment was designed to test further improvements to the Periphery Vision Menu System (PVMS). This final experiment, as part of this research, was conducted between August 2016 and December 2016. Much of the codebase and experiment design was reused from experiment two for this experiment. The second experiment focused on features of the menus related to making them appear, ensuring the experience felt intuitive and identifying issues perceived by participants. Building on the feedback, improvements to the core menu system were applied. Testing conducted during this experiment focused on comparing various scenarios, returning to some of the techniques of the first experiment with a pattern-matching puzzle and the tower defence game from the second experiment.

As was the case with the other experiment chapters, the discussion presented in this chapter will follow a similar style. The first sections discuss the experiment methodology. Some of these sections will reference the second experiment where appropriate overlaps in the content are discussed. The later sections present the results and a summary of the results. A discussion of the combined results from all experiments can be found in Chapter 7. For details about accessing the experiment code on GitHub, see Appendix D.

6.1 Experiment Overview

This experiment received ethics approval from the Flinders University Social and Behavioral Research Ethics Committee under research project number 7375. The following points were the primary goals used to guide this experiment.

- Determine whether an input method provides a better overall experience in terms of usability with HMDs by comparing HMD Only or combining with a controller.
- Determine whether a PVMS is viable as a tool for interaction within this application and more broadly for other applications.
- Determine whether improvements could be made to the proposed menu system in the way it is calibrated.

These goals were investigated with four different tasks. Task 1 and Task 2 required the participant to complete an object matching task. Firstly, with just the PVMS. Then a radial-style menu complemented the PVMS, which was paired to show how the menu could exist alongside other techniques. Changes were also made to the tower defence game in Task 3 and 4 to improve the experience. Overall, this experiment was given a vastly more appealing visual style to enhance the participant's experience, primarily driven by the completely new experiment map created from scratch. Each task has a section discussing how they were designed as part of the methodology, along with other supporting elaboration on the structure of the experiment and questionnaires providing direct feedback from participants.

6.2 Methodology

This methodology section will cover the important features of how the experiment was run. The methodology has been broken down into many smaller sections. Section 6.2.3 Experiment Tasks details some of the general functional information related to tasks overall, leading to a brief discussion for each task on how they worked within the experiment. The other sections in the methodology will provide relevant information necessary to understand the scope of this experiment. After identifying experiment tasks, two sections cover the methods of data collection used in this experiment with the questionnaires and the data collected from within the application.

Around the time between the second and third experiments, many applications were released commercially for VR. The applications gave some insight into how developers were approaching interface design from a product perspective. One application was the Tilt Brush application (Google, 2016) that demonstrated techniques for multi-dialog interchangeability using the HTC Vive controller. These interfaces encapsulated utility in simple small interfaces. The interfaces' simplicity was comparable to the interfaces used in the experiments when many novice users were beginning to experiment with VR. The framework used in the experiments was designed to utilise a serious games approach to research by fitting in, where possible, to serious games taxonomies (Laamarti et al., 2014). The experiment used games and puzzles as a medium for useful data collection while changing how participants would interact. This serious game approach extended from the techniques used for recruitment as a means of marketing to participants completing the tasks.

6.2.1 Recruitment

Recruitment began in August 2016 and concluded with the final participants in December 2016. Recruitment was conducted in an almost identical way to the second experiment. No monetary compensation was provided to any participants. Advertising was conducted with the aid of a new trailer via YouTube. The primary distribution method of advertising was email. As with experiment two, select university lectures were attended to promote awareness for the study by showing the trailer and briefly describing the research. This left individuals to follow up based on their interest in participation. When participants followed up with an expression of interest either in response to the email or from the lectures, a response email would be sent with three documents:

- A letter of introduction: introducing the project and researcher from the supervisor.
- An information sheet: giving brief details explaining what would occur during the experiment.
- A consent form: to show what they would be consenting to for participating. The form established the participant was of appropriate age (17+), understood the experiment requirements, indicated they would not directly benefit from the research, indicated they were free to withdraw at any time and confirmed they would be deidentified to remain confidential in any publications.

Participants could view these documents and respond with preferred times based on a Google Sheet. The sheet was updated to reflect available times, so everything could be efficiently conducted around the times that suited each participant.

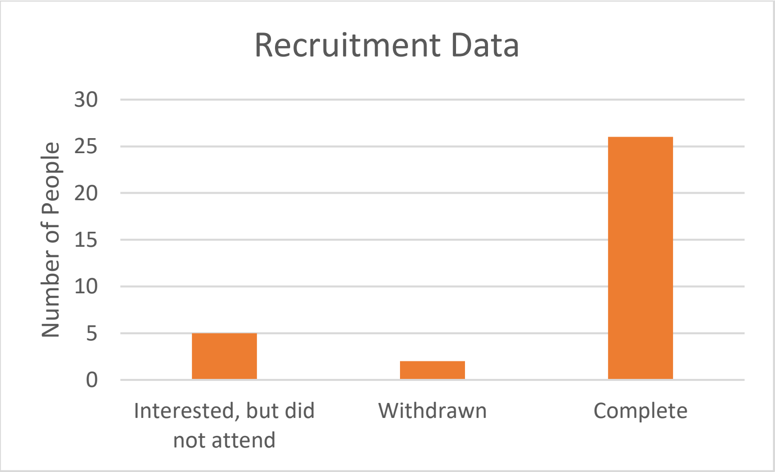


Figure 6.1: Experiment 3 Participation Quantities

In Figure 6.1, the communications with people interested in the experiment are summarised. 5 people were sent the additional information and never followed up. 2 people withdrew from the experiment after selecting a time to participate. 26 people participated in the experiment to completion. No participants experienced issues with vision in this experiment, and like experiments 2 and 3, the IPD was not measured or used.

YouTube Trailer

The visual example of the storyboard for the trailer can be found in Appendix C.2.1. One of the major flaws of the second experiment trailer had been too much fluff with little exciting happening on screen. The arguably most interesting sequence had been tucked away right at the end after a series of less visually engrossing shots. The slow build-up thematically drove the design process of this storyboard. Due to the decision to edit with Windows Movie Maker, there were certainly many limitations on how content could be cut together because the application did not support multiple concurrent video channels. The number of unnecessary text sections was cut down, leaving more time to show some flashy visual effects to capture attention. The cuts were made to keep the trailer to the point with a short runtime.

The trailer started the same way as the previous experiment, with a title text introducing the content. The trailer then began forming a narrative to connect this experiment to the previous one. The narrative for this experiment was one of defeating the enemies who had invaded the presumably human settlement. The trailer showed the final boss of the previous experiment collapsing to death. The viewer was then told that the way these creatures were reaching the humans was still open. Suggesting there were more coming and posed an imminent threat to be dealt with and required a response from the player to put up a defence.

The next scene showed a swirling blue portal effect over a teleporter pad where the spider creatures had been teleporting through to attack. The camera slowly panned backwards, giving time to see the effect and show the endpoint defended during the experiment. Then text introduced the idea of having travelled through the portal for a pre-emptive strike.

After introducing the idea of fighting back against the threat, a rotating panning shot showed a selection of different towers spawning in with the yellow visual effects making them appear to teleport in. A camera transition was used to show the entire route from the end teleporter pad to the cave where the enemies would be spawning from to attack. The scene that followed was a quick horizontal panning shot that presented a glimpse of gameplay as the towers and spider enemies fired at each other.

The focus changed to a cave where the spiders had been coming from during the gameplay. Meanwhile, an audible drumbeat was heard to emphasise something sinister was coming, followed by an ominous creature roar. An explosion was shown, causing many boulders to fly through the air and come to a rest. With the climax of an unseen creature, the trailer ended with text indicating this was a research project seeking participants who, if interested, could contact the researcher for more information. The last text was left up for about 20 seconds to provide enough time to see the email address if it was unknown.

The video can be viewed at: https://www.youtube.com/watch?v=bVtB0wj8ehI.

YouTube Watch Data

As presented with the second experiment's trailer, the YouTube data has been provided to show how successful the trailer reached interested parties. Compared to the second experiment, the metrics showed similar total watch times: 87 minutes for the second experiment trailer versus 85 minutes for the third experiment. Although the trailer for the third experiment was longer by 2 seconds, only the first 45 seconds had to be viewed to see all the content. So, with an average view duration of 44 seconds, this generally indicates those viewing watched the entire content. Viewing devices saw an increase in mobile and a reduction in desktop computers; 67 minutes down to 58 minutes for computer, and 19 minutes up to 27 minutes for mobile. There was an increase in the number of female viewers for this trailer compared to the second experiment. Up from 0.7% to 7.5%. There were also fewer in the range of 18-24 years (48% down to 35%) and increased distribution for 35 years onwards.

- Duration: 1 minute 6 seconds
- Total Views: 113
- Total Watch Time: 85 minutes
- Average View Duration: 44 seconds
- Average Percentage Viewed: 68%
- Viewing by Device:
    - Computer: 58 minutes

    - Mobile: 27 minutes
    - Tablet: 0 minutes
- Viewing by Gender: 92.5% male, 7.5% female.
- Viewing by Age:
    - 18-24 years: 35% (91% male, 8.6% female)
    - 25-34 years: 46% (90% male, 9.7% female)
    - 35-44 years: 13% (100% male)
    - 45-54 years: 4.3% (100% male)
    - 55-64 years: 1.6% (100% male)

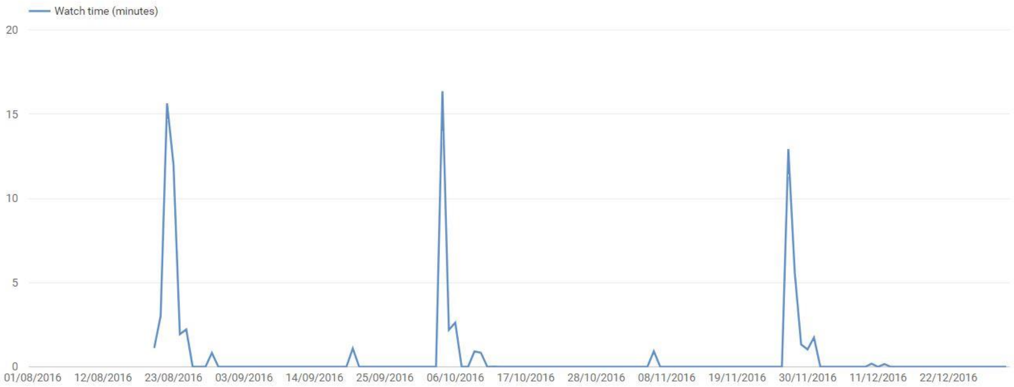


Figure 6.2: Video Watch Time by Date

Throughout recruiting, three separate waves of emails were sent out. The waves of recruitment can be seen from the viewing data in Figure 6.2. The first two sets had nearly equal peak viewing with a faster drop off for the second. And a slightly smaller amount of viewing when the third round was completed. It should be noted the final round was sent out near the end of year exam times, so the number of people engaging with recruitment emails was expected to be lower.

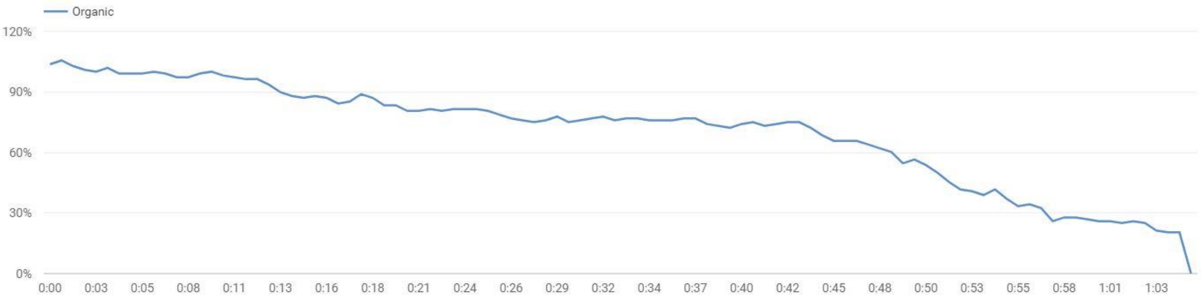


Figure 6.3: Video Absolute Audience Retention

Figure 6.3 (previous page) and Figure 6.4 show a more detailed representation of how the videos were viewed on average. Absolute retention fell off after 45 seconds when the text remained up for the rest of the video. The fall-off is emphasised by Figure 6.4, showing the number of viewers watching past this point was below average.

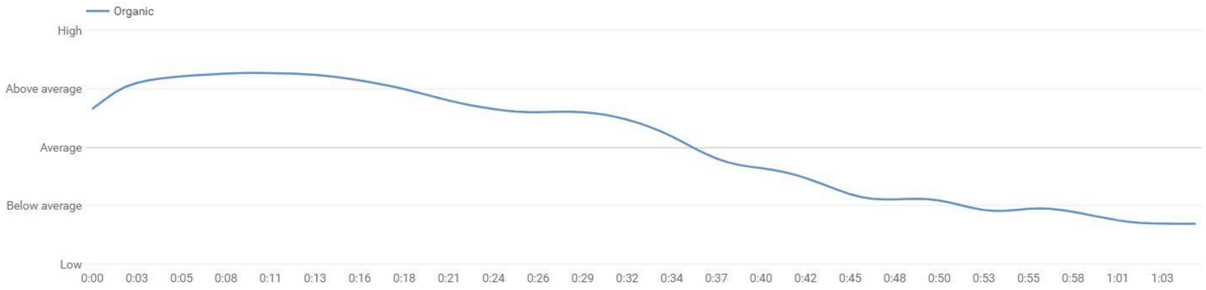


Figure 6.4: Video Relative Audience Retention

The questionnaire included questions related to the trailer material, so further discussion about the success of the trailers will be covered in the results summary (section 6.3.1) and the combined results in Chapter 7.

6.2.2 Hardware and Software APIs

Only one significant change to hardware and software was made following the second experiment with the addition of an Xbox controller. An Xbox controller is designed to naturally sit in a user's hands while interacting with provided buttons. The controller was selected as a substitute for either mouse or mobile as were tested during the first experiment. The controller was used purely for selections during the second iteration of the tower defence game. For drawing comparisons between having the controller input as an extra utility versus the experience of using HMD Only interactions. To summarise, the hardware and software used as part of this experiment were the following.

- Desktop Computer (same as was used for the second experiment)
- Oculus Rift Dev Kit 2
- Unity version 5.3.2
- Xbox 360 wired controller (Figure 6.5 over page)

There were additional updates released for Unity during the time of development. It was decided to keep a consistent version based on what had worked for the second experiment. The decision to maintain consistency was made to ensure no unnecessary fixes had to be applied for unforeseen changes with a codebase that already worked from conducting the second experiment.

Figure removed due to
copyright restrictions.
See link for original:
[Link]

Figure 6.5: Xbox 360 Wired Controller

6.2.3 Experiment Tasks

The tasks used for this experiment could be considered improved versions of tasks used for previous experiments. The object matching task was used from the first experiment and the tower defence game from the second experiment. Each provides a different experience to enable testing for the iterative changes applied. Systems from the second experiment in the form of tutorials were utilised again to reduce the learning curve. The types of interaction with the object matching for the first two tasks were only completed once each. The randomised puzzle solutions introduced the concepts while at the same time not providing a solution that could be repeated identically between tasks. A single question followed each task to evaluate the participant’s immediate perception. The following list introduces each task with more detailed explanations following in later sections of the methodology.

- **Task 1 Object Matching Menus with PVMS Only**: (presented in section 6.2.6) This task and Task 2 emulated tasks from the first experiment. The goal was to use the provided menu system to match five objects. Creating these objects was completed with one menu and modifying the created objects with another. Objects to be matched could have one of three shapes, one of three colours, one of three sizes, and form many different positional variations. The puzzles were randomly generated for each participant to be unique in a way that required a minimum number of interactions.
- **Task 2 Object Matching Menus with Circular and PVMSs**: (presented in section 6.2.6) The second task was almost the same as Task 1, except the menu types were changed.

Object creation was still completed using the PVMS, but object modification was completed with a circular menu similar to the first experiment’s sixth task.

- **Task 3 Tower Defence Game with HMD Only**: (presented in section 6.2.7) This task provided a new way to experience the HMD Only interactions of the tower defence game from the second experiment. Some of the significant changes from the previous iteration included: removing tower destruction, shifting the tower move/repair/deselect to appear above the tower while selected, and rebalancing to support a larger playing area.
- **Task 4 Tower Defence Game with Xbox Controller**: (presented in section 6.2.7) The first experiment had compared interactions with the aid of an additional interaction tool against using HMD Only interactions. This task provided a final comparison between Task 3 and Task 4 by providing an experience of using the improved PVMS with the added selection tool. The Xbox controller was decided to be a natural interaction tool without having one of the hand-held Oculus Rift (or HTC Vive) controllers.

Task Design Philosophy

Further to the listed task descriptions, this list presents some additional discussion about the philosophy for designing the tasks. This experiment was primarily designed to have comparable overlaps to the first two experiments while still testing new improvements to the PVMS.

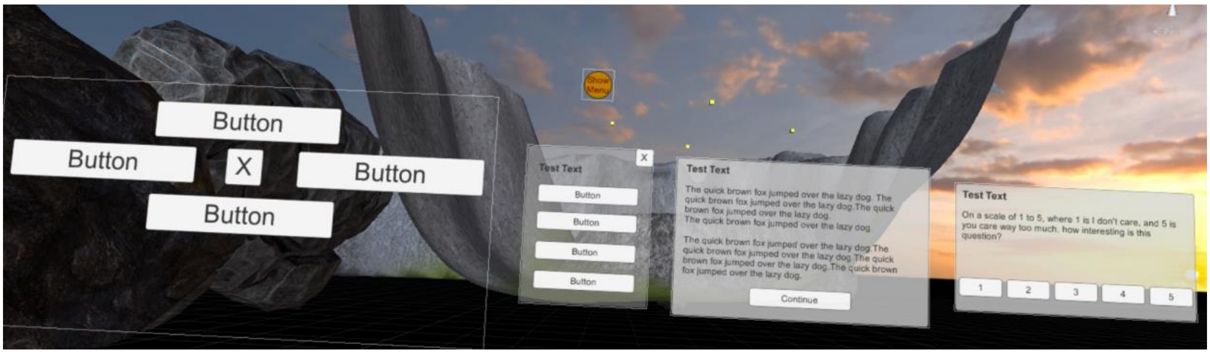
- **Tasks 1 and 2**: These tasks evaluated an improved implementation of the last two tasks from the first experiment. The tasks also overlapped with the “Tower Construction” task in experiment 2. Both tasks for this experiment did not impose time pressure on the participant while they engaged by performing object creation and manipulation to reach a specific goal state. The implementation provided context cues with a tick to indicate success to the participant instead of being less clear for individual step completion as in the first experiment. Between the two tasks, the design goal was to evaluate how using the PVMS for all operations compared to using the PVMS for object creation and the circular menu for manipulation. The purpose of this was to evaluate how the PVMS worked alongside another menu type. It would be possible to create objects with the circular menu, but the difficulty for the user comes from selecting a position to create the object initially. The circular menu presented a menu at the location where a participant intended to perform an action relative to the object of interaction, instead of the PVMS that could be placed anywhere by the user. Using these two types of menus provided a comparison of menus for object manipulation.

- **Tasks 3 and 4**: The last two tasks were designed to contrast against Task 3 of the second experiment directly by using a similar implementation with improvements to provide a better experience for the participant. As stated in the design philosophy for the Tower Defence game as part of the second experiment, the experience provided an experience that utilised time pressure to make the participant engage actively. The tasks were also highlighted as the key motivation for research participation from the trailer used for advertising. The difference between the tasks was designed to evaluate instant selection for the PVMS compared to the first experiment's evaluation of different input modes.

The tasks for the experiments and their design philosophy have now been presented. The implementation details are further discussed in the following sections. Before discussing the task implementation further, the interface elements and the use of tutorials for this experiment are presented similar to how they were for the second experiment.

6.2.4 Interface Element Types

The core dialogs used for this experiment are shown in Figure 6.6 (over). The main changes seen here were the addition of a circular menu (left), the two-step widget (top middle), and the addition of the close button to the PVMS (middle). The circular menu is a different layout of the PVMS, constructed with four buttons and a close button, matching the same number of interactable elements except with no title. The consistent composition of elements allowed the menu to reuse the same code for populating menu buttons and providing button interactions. For discussion about the two-step widget and the addition of the close button, refer to 4.4.2 in the Technology Overview chapter.



Test Text

Test Text

The quick brown fox jumped over the lazy dog. The quick brown fox jumped over the lazy dog.The quick brown fox jumped over the lazy dog.
The quick brown fox jumped over the lazy dog

The quick brown fox jumped over the lazy dog.The quick brown fox jumped over the lazy dog The quick brown fox jumped over the lazy dog.The quick brown fox jumped over the lazy dog.

Test Text

On a scale of 1 to 5, where 1 is I don't care, and 5 is you care way too much, how interesting is this question?

Figure 6.6: Menu Interface Elements

Only one of the tower defence interfaces saw a dramatic change. The tower selected dialog seen with the green background in Figure 6.7 has up to three buttons around it. Deselect would always be visible to deselect the current tower. Repair would be visible if the tower had 1 or more cost to repair, (shown in the figure where "Repair [25]" is). And the move tower option was visible during down time between waves in the same way this option was usable during the first experiment.

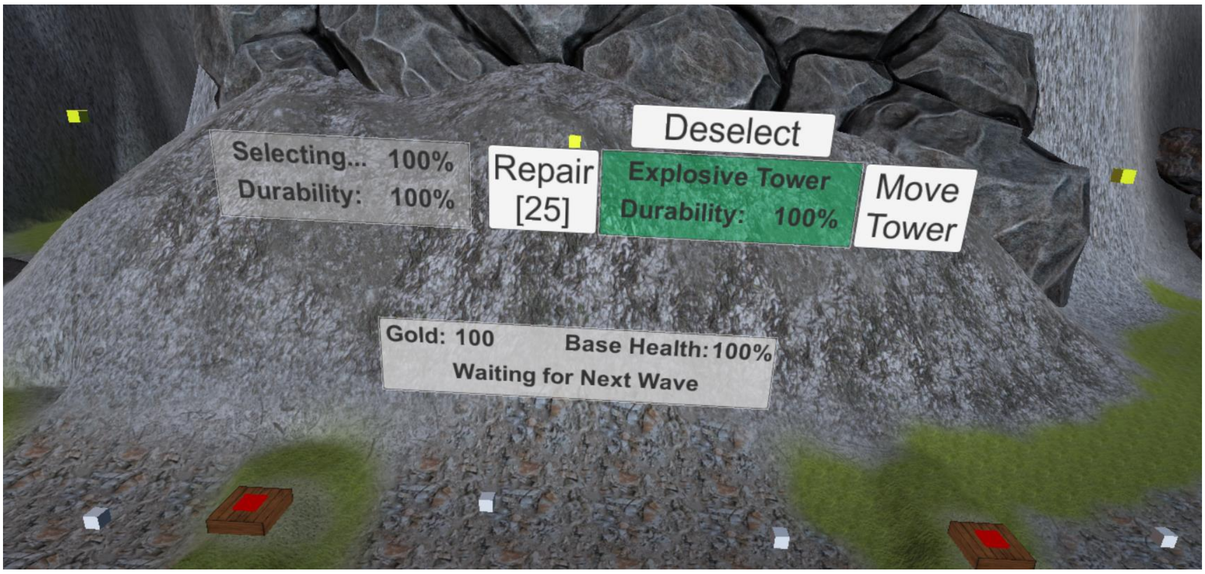
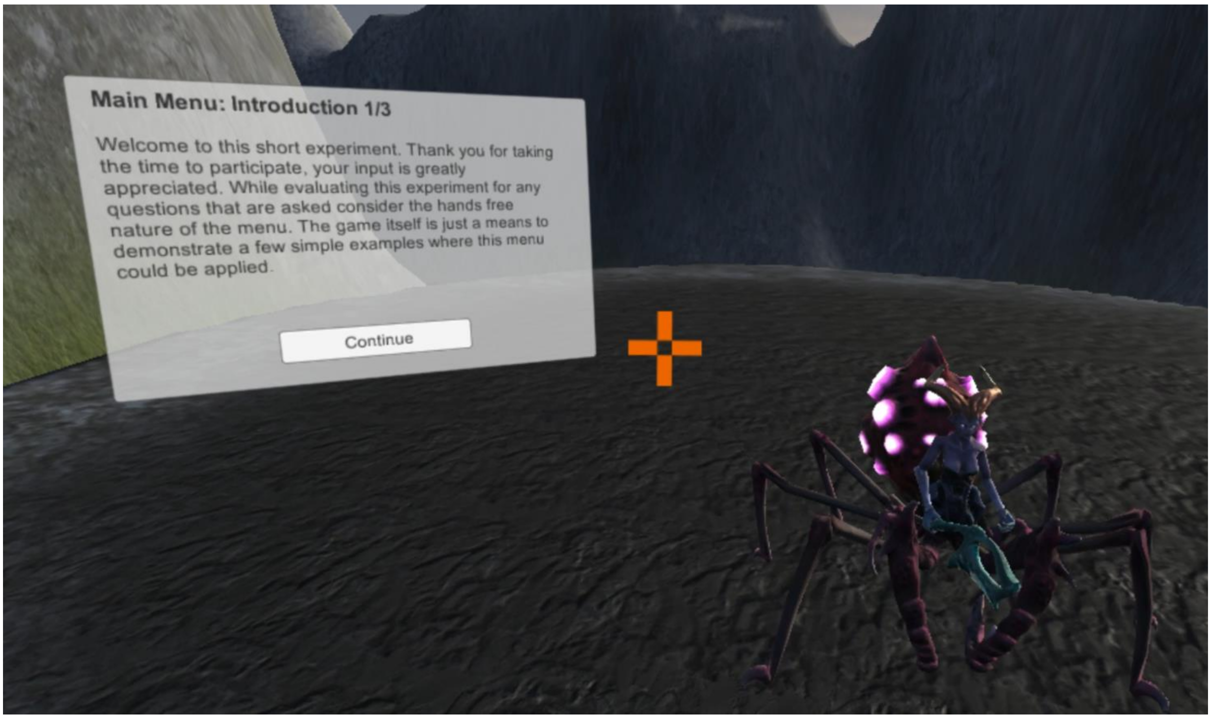


Figure 6.7: Tower Defence Interface Elements

6.2.5 Use of In Application Tutorials

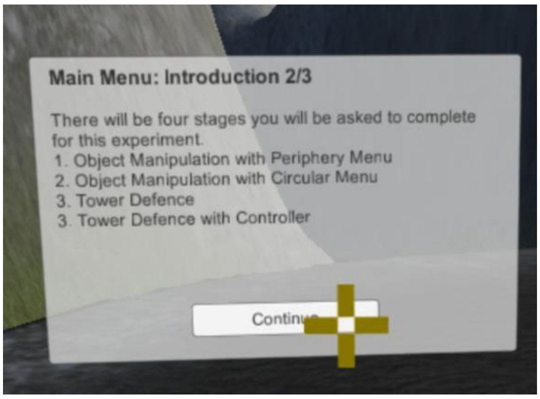


Main Menu: Introduction 1/3

Welcome to this short experiment. Thank you for taking the time to participate, your input is greatly appreciated. While evaluating this experiment for any questions that are asked consider the hands free nature of the menu. The game itself is just a means to demonstrate a few simple examples where this menu could be applied.

Figure 6.8: Main Menu Introduction 1/3

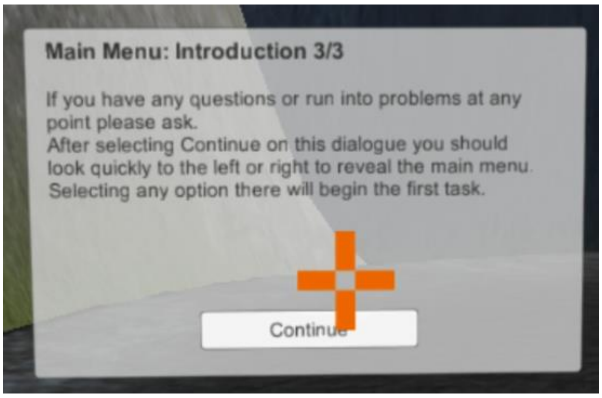
As was the case in the second experiment, introductory tutorials were used before each task to introduce concepts. The tutorial goals were to introduce what to expect from the experiment (Figure 6.8), provide a list of tasks that would occur during the experiment (Figure 6.9), provide a time for questions (Figure 6.10), and introduce the basics of how to reveal the PVMS. The tutorial slides for each task are included in the following sections to pair with the relevant content. The main menu introduction slides had one of the animated boss enemies, as seen in Figure 6.8, to provide an additional visual element.



Main Menu: Introduction 2/3

There will be four stages you will be asked to complete for this experiment.
1. Object Manipulation with Periphery Menu
2. Object Manipulation with Circular Menu
3. Tower Defence
3. Tower Defence with Controller

Figure 6.9: Main Menu Introduction 2/3



Main Menu: Introduction 3/3

If you have any questions or run into problems at any point please ask.
After selecting Continue on this dialogue you should look quickly to the left or right to reveal the main menu. Selecting any option there will begin the first task.

Figure 6.10: Main Menu Introduction 3/3

A similar process was used to introduce the participant to the PVMS interaction to conclude the main menu introduction. After closing the introduction in Figure 6.10 (previous page), the participant had to reveal the menu by turning to the left or right. Performing the action would show the two-step widget seen in Figure 6.11; then, by hovering over this element, they would reveal the menu seen in Figure 6.12. Requiring the action to continue ensured participants had used a basic version of the interaction before completing any tasks with it.

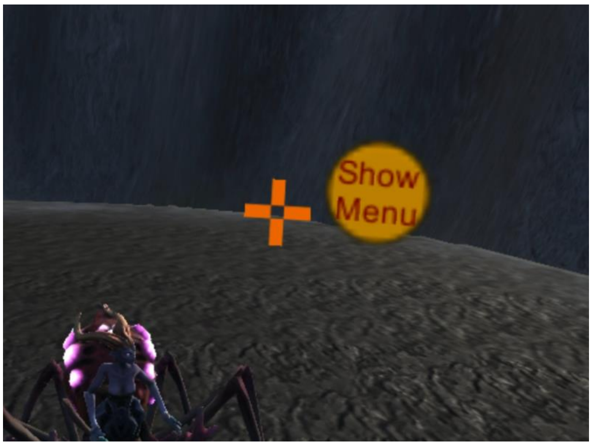


Figure 6.11: Two-Step Widget Example

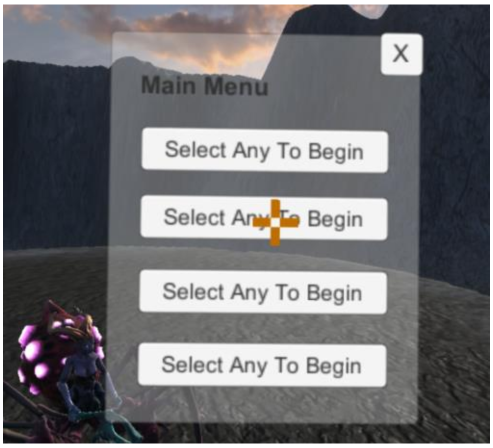


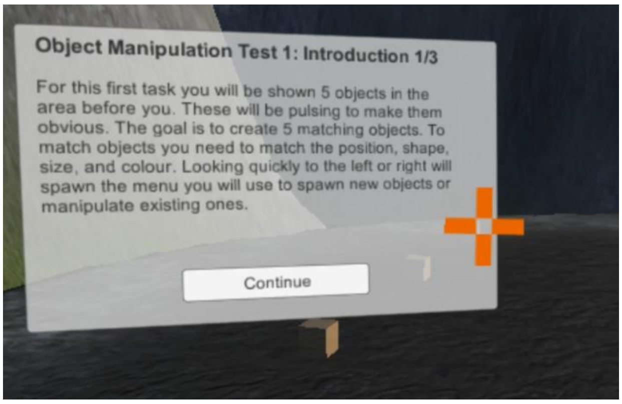
Figure 6.12: Main Menu

6.2.6 Task 1 and 2: Object Matching Menu Comparison

The object matching puzzles required the completion of a sequence of steps to reach of final matched condition. This pair of tasks were based on Task 6 and Task 7 from the first experiment. Utilising a similar approach with a few significant improvements formed the basis for these tasks. The first notable improvement was the shift from using static interface elements to incorporating the PVMS. The other significant improvement was the shift to dynamic puzzle creation and adding shape as a new variable to be considered. This section will begin with Task 1's PVMS Only approach showing the tutorial, layout, and an example. Following at the end will be another set of tutorial slides and an example for Task 2 where a circular type of menu was incorporated.

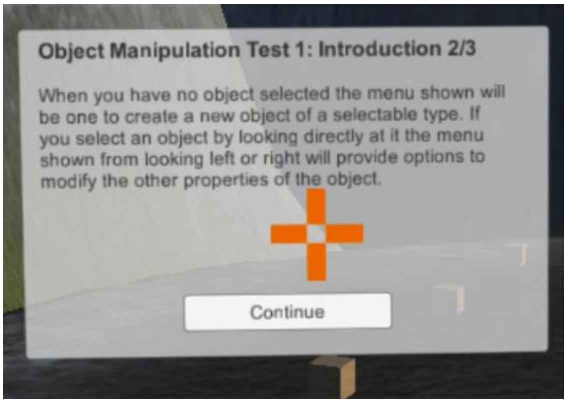
Tutorial

In Figure 6.13 (over), the participant is introduced to the primary goal for completing the upcoming Task 1and has explained how the PVMS will be used to complete the actions required for the puzzle. Figure 6.14 (over) goes into more detail, explaining the difference between the two menus.



Object Manipulation Test 1: Introduction 1/3

For this first task you will be shown 5 objects in the area before you. These will be pulsing to make them obvious. The goal is to create 5 matching objects. To match objects you need to match the position, shape, size, and colour. Looking quickly to the left or right will spawn the menu you will use to spawn new objects or manipulate existing ones.



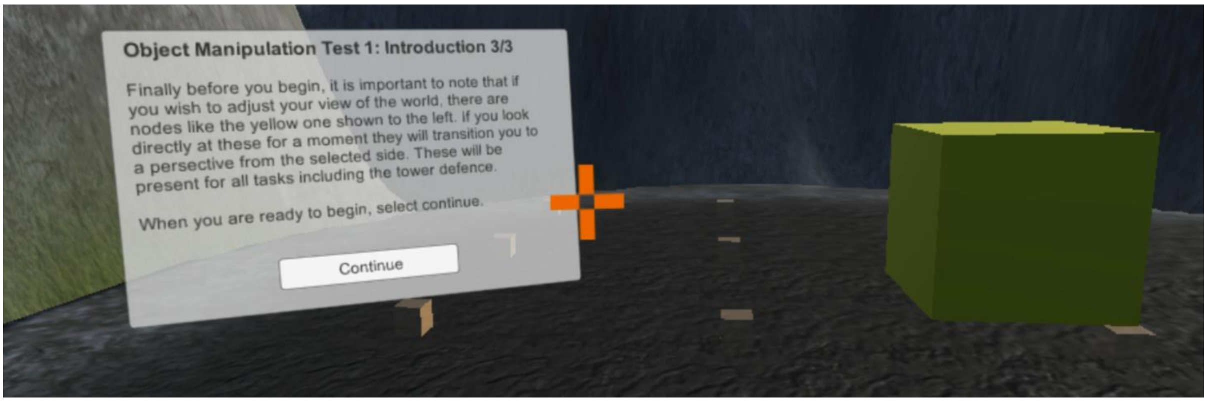
Object Manipulation Test 1: Introduction 2/3

When you have no object selected the menu shown will be one to create a new object of a selectable type. If you select an object by looking directly at it the menu shown from looking left or right will provide options to modify the other properties of the object.

Figure 6.13: Object Matching Puzzle Introduction 1/3

Figure 6.14: Object Matching Puzzle Introduction 2/3

Figure 6.15 explains how to use the optional feature with the yellow blocks. These functioned in the same way as with the second experiment. For this task, there were four yellow blocks placed around the area. These were not necessary to complete the tasks but provided a way to change the perspective.



Object Manipulation Test 1: Introduction 3/3

Finally before you begin, it is important to note that if you wish to adjust your view of the world, there are nodes like the yellow one shown to the left. If you look directly at these for a moment they will transition you to a persective from the selected side. These will be present for all tasks including the tower defence.

When you are ready to begin, select continue.

Figure 6.15: Object Matching Puzzle Introduction 3/3

Task 1 with PVMS Only

Figure 6.16 shows an example of how the puzzle may first appear to the participant. The slightly angled top-down view gave a suitable perspective for seeing all the objects regardless of how they spawned. The solution elements seen in Figure 6.16 include a: small green cube, small red sphere, large green sphere, small red sphere, and small blue cylinder. Each of these had transparency to make them appear different from the objects created by the participant.

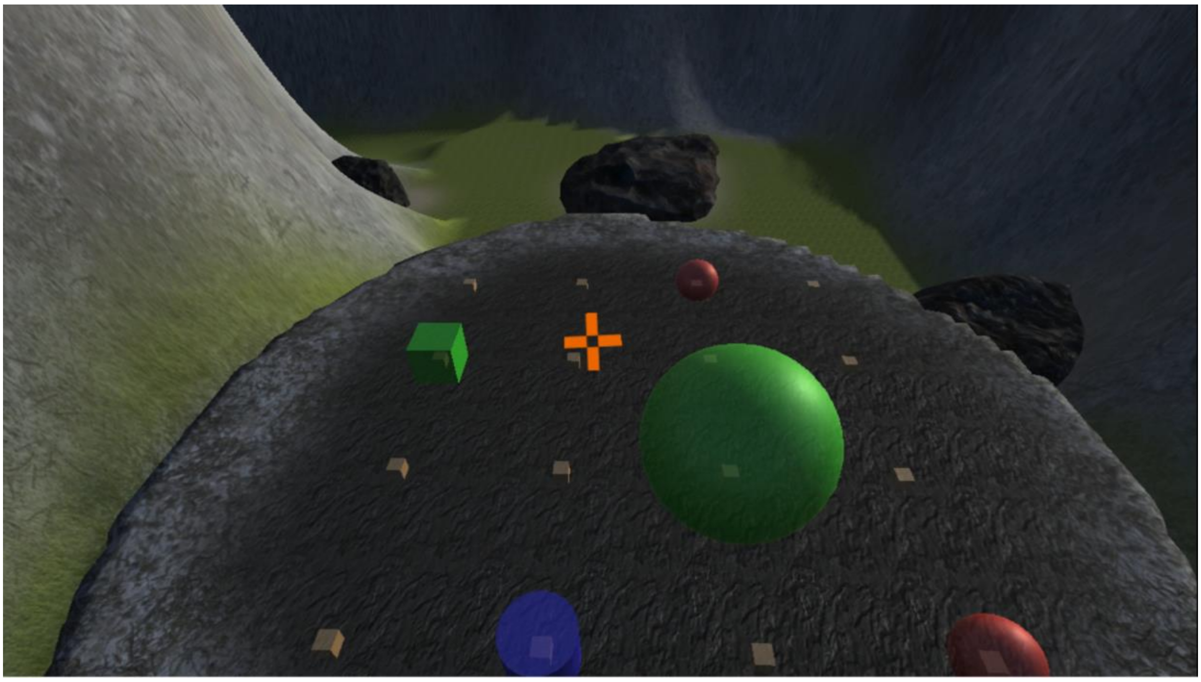


Figure 6.16: Object Matching Puzzle with Transparent Solution

These solution objects were spawned by the **`MatchObjectManager`** using a **`spawnSolutionNodes(int count)`** method. The steps used to generate the random solution were as follow.

1) Reset all nodes to their default state to clear the previous puzzle.
2) Reset the minimum operation count and operation count arrays used to track the minimum operations versus how many the participant will take to complete the task.
3) Loop for count times:
    a. Find a unique unclaimed node on the grid, not next to any other node.
    b. Select a random shape represented as 0 to 2.
    c. Set the colour based on the number of objects spawned so far. Forcing the existence of at least one object for each colour. Colours represented as 0 to 2.
    d. If the number of objects spawned currently is larger than 2, select a random colour with a random value between 0 and 2.
    e. Set the size to a random number between 0 and 2 to represent the three different sizes.
    f. If the random size is type 0 and a random number between 0 and 100 is greater than 70, change the size to a random number between 1 and 2.
    g. Spawn a solution object with the generated randomised properties.

h. Attach the solution object to the appropriate node selected in step a.
i. Increase the minimum operation counts as needed for the entire solution.

The way these solutions were randomly generated is significant to ensure participants did not have all the same single type of object with the same properties. When creating objects, they would default to type 0 for size and colour. For this reason, steps (d) and (f) of the procedure were used to enforce a minimum amount of difference. The forced randomisation of variations was unnecessary for the shapes to the same extent because the participant manually selected these from the menu. Typically, natural randomisation gave a good distribution between the 5 objects.

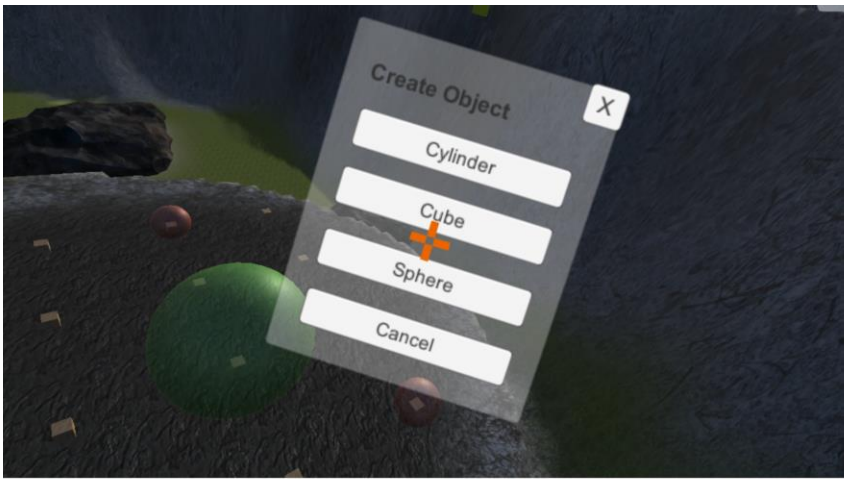


Figure 6.17: Object Matching Puzzle Periphery Menu for Creating New Objects

The first step for the participant to solve the puzzle was to reveal the PVMS, as seen in Figure 6.17. With no object selected, the menu would show options to create a new object. The options enabled the creation of a Cylinder, Cube, or Sphere. Selecting any of these three would spawn an object of that type. Figure 6.18 (over) shows an example where the cube object was selected. Blue and small were the defaults for the other properties. Spawning a cube would attach the small blue cube to the cursor. The cube could then be moved across the plane of the solution area. Moving into one of the nodes would trigger a snap type of interaction, removing control from the cursor.

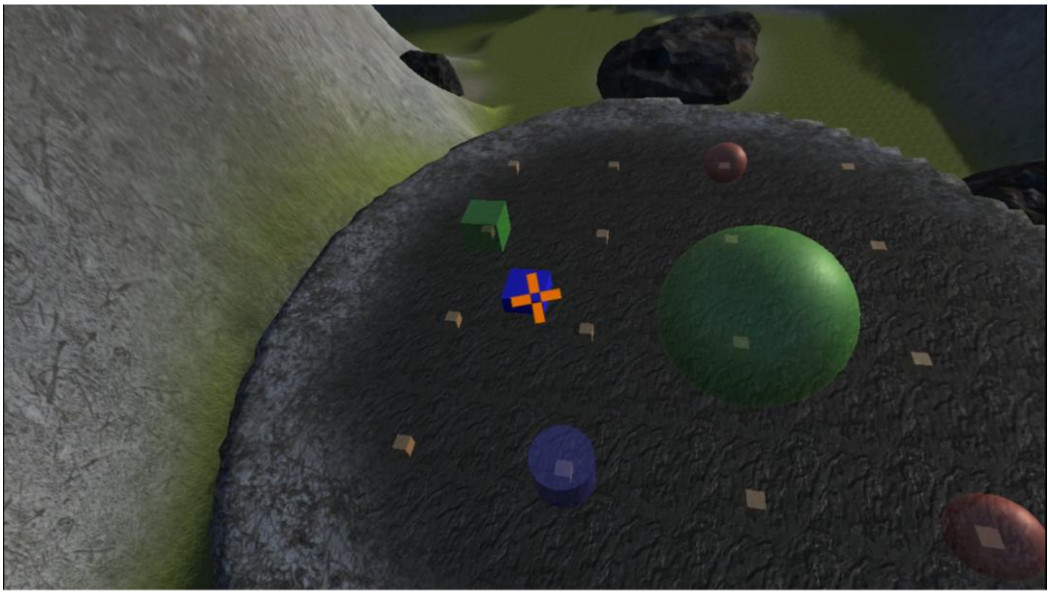


Figure 6.18: Object Matching Puzzle Spawned Cube Moving to Snap

By default, the object would not be selected after placing an object. By using the cursor to look directly at a spawned object, the targeted object could be selected. Targeting an object can be seen in Figure 6.19, where the blue cube has been selected. The solution at this node is a small green cube. The last step in this example would be to change the colour to green, therefore completing the solution for that object. The transparent solution is slightly offset in position to make it always visible when inside another object. During Task 1, when an object is selected, the Deselect button is shown over the object. Hovering over the button would deselect the current object.

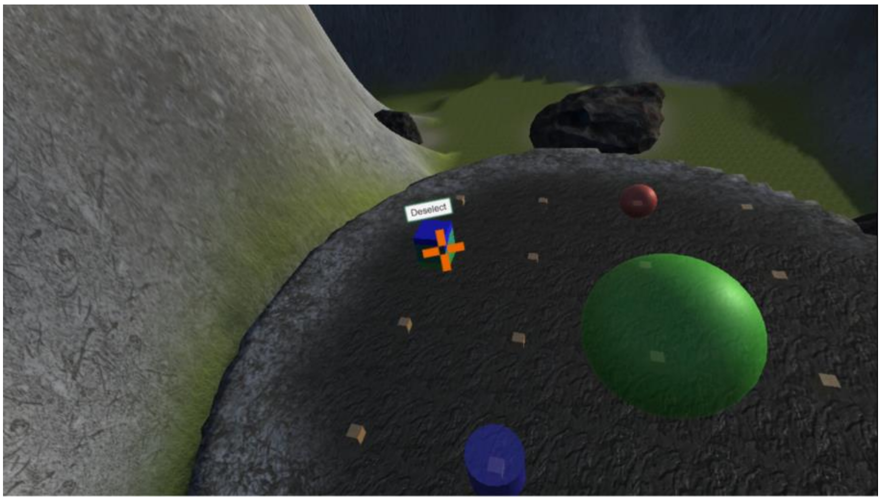


Figure 6.19: Object Matching Puzzle Snapped and Selected Cube

While an object is selected, the PVMS changes the context to provide a Modify Object menu when triggered. The menu displayed in Figure 6.20 (over) offers four options for the Shape

Type, Size, Colour, and Move. The first three options change the menu to show a sub-menu. The changing shape menu is like the create object menu. The other two options will be shown in examples to follow.

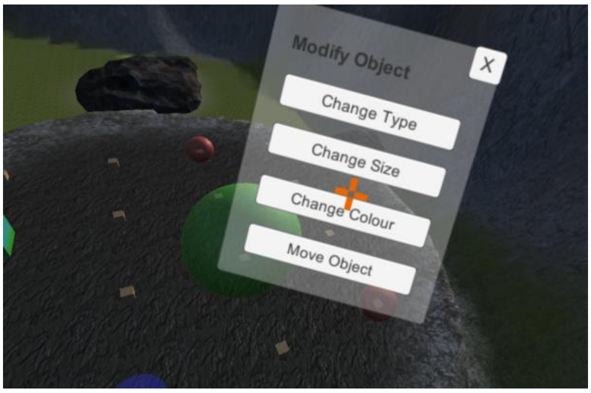


Figure 6.20: Object Matching Puzzle Modify Selected Object Menu

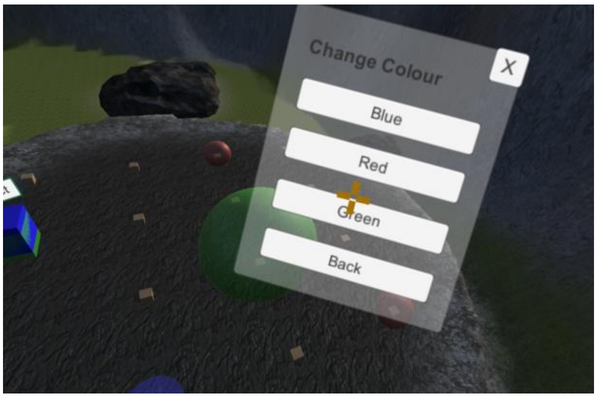


Figure 6.21: Object Matching Puzzle Change Colour Sub Menu

Figure 6.21 shows the colour sub-menu. Selecting any of these options will change the currently selected object’s colour. In this example, the Green option was selected, resulting in Figure 6.22. Once the green menu option was selected, the object was automatically deselected because the solution for that node was complete. The tick is shown above the object to confirm to the participant that the node task was complete. Figure 6.23 shows an almost completed solution for Task 1. To reach the state shown in the figure, the participant would have created two small blue spheres, changed them to a red colour, and created a small blue cylinder and a small blue sphere. The final step was to change the last small blue sphere with a size operation to large and a colour operation to green.

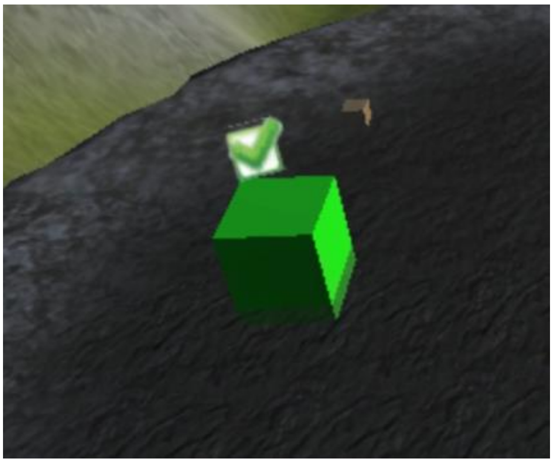


Figure 6.22: Object Matching Puzzle Matched Object

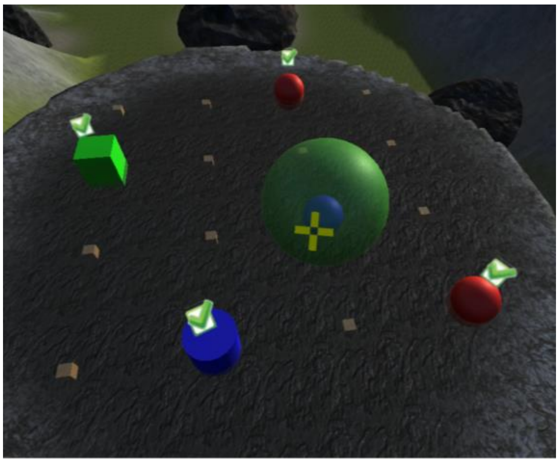


Figure 6.23: Object Matching Puzzle Multiple Complete with Circle Object Spawned

The size menu is shown in ; the colour was already changed to green in this example. When the last step of switching from small to large was completed, the task would end. The puzzle would become hidden and replaced with a user response question seen in (over).

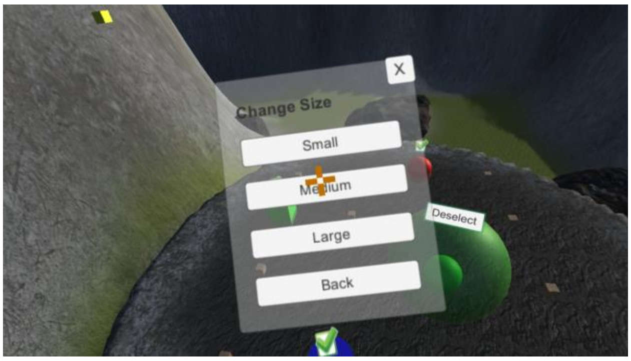
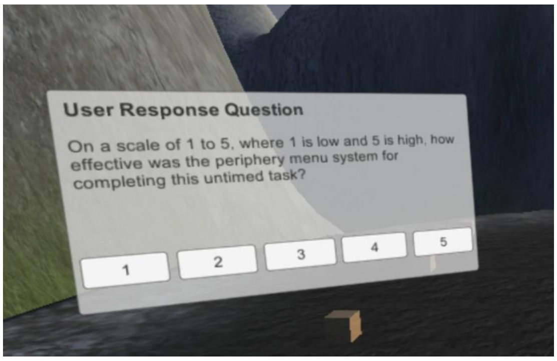


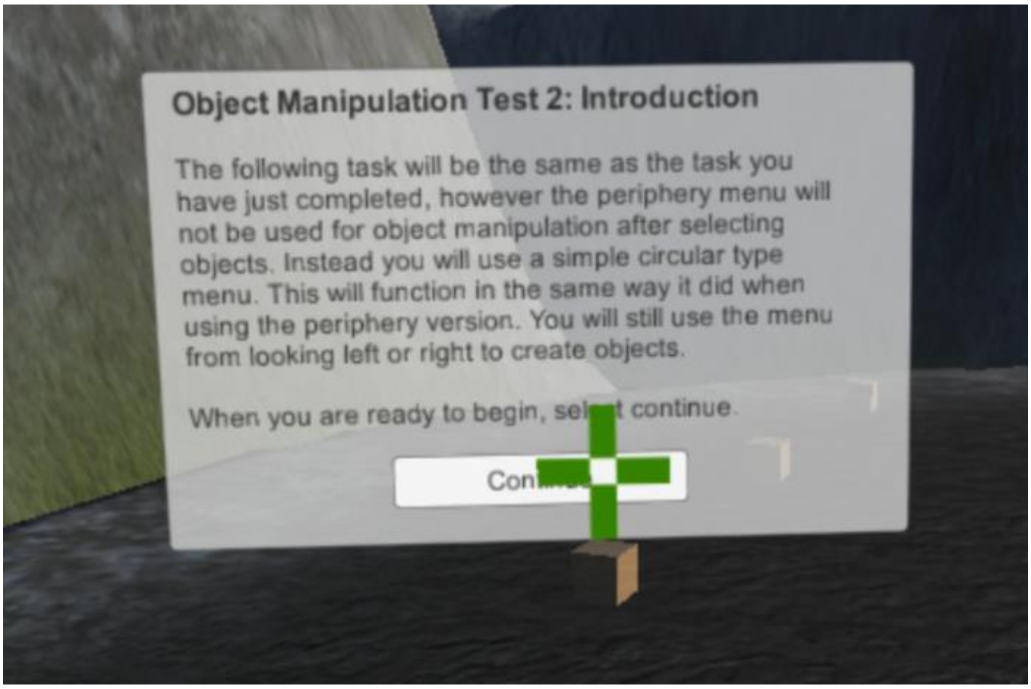
Figure 6.24: Object Matching Puzzle Change Size Sub Menu


User Response Question

On a scale of 1 to 5, where 1 is low and 5 is high, how effective was the periphery menu system for completing this untimed task?

Figure 6.25: Object Matching Puzzle Task 1 User Response Question

Task 2 Circular Menu with PVMS


Object Manipulation Test 2: Introduction

The following task will be the same as the task you have just completed, however the periphery menu will not be used for object manipulation after selecting objects. Instead you will use a simple circular type menu. This will function in the same way it did when using the periphery version. You will still use the menu from looking left or right to create objects.

When you are ready to begin, select continue.

Figure 6.26: Object Matching Puzzle Task 2 Introduction

Task 2 provided an almost identical experience regarding the type of puzzle being solved. The primary difference was the use of a circular menu. The design for this menu followed the type of circular menu used in the first experiment’s Task 6, with changes to improve the visual appeal and build the menu into the generic menu management code used in the PVMS. The menu itself could be an alternate representation of the PVMS as it is primarily a visual style change. The primary difference with this menu is that it is used as an object attached menu instead of a Periphery triggered menu. Two menu types are used to demonstrate an example of the types of menus coexisting for different purposes. Spawning in objects is used as an infrequent activity with no point of interaction, so it benefits from remaining hidden in the PVMS. Contrasted against the circular menu's use for more frequent actions on already defined objects with a fixed location in the world. After completing the user response question from Task 1, the participant would be taken to (previous page). Where the differences between the Task 1 and Task 2 were explained.

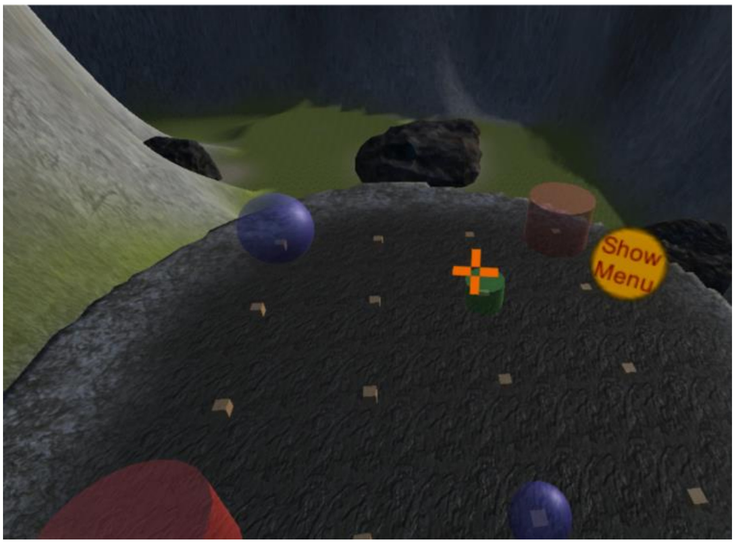


Figure 6.27: Object Matching Puzzle Alternate Puzzle

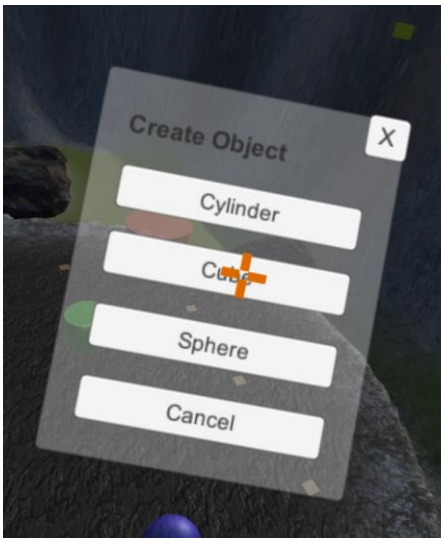


Figure 6.28: Object Matching Puzzle Create Object Menu

Figure 6.27 shows how the puzzle could have spawned differently for this task, using the same random logic from Task 1 to provide a different random experience. The solution objects shown are a medium blue sphere, small green cylinder, medium red cylinder, large red cylinder, and small blue sphere. As with Task 1, an example is shown in the following steps for solving a single object. As normal, the first step would be to create an object using the PVMS seen in Figure 6.28. In this case, a small blue cylinder will be spawned to match the small green cylinder. Once the small blue cylinder was spawned and placed at the preferred node, it could be selected.

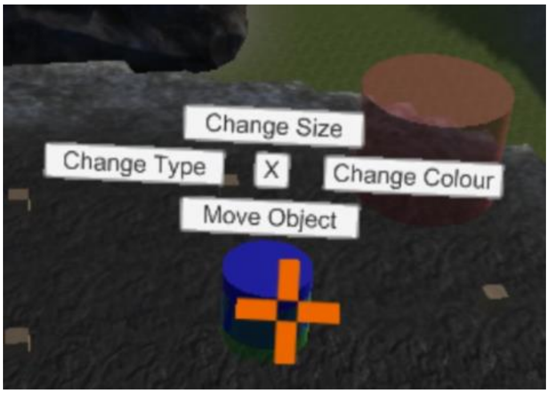


Figure 6.29:Object Matching Puzzle Circular Menu

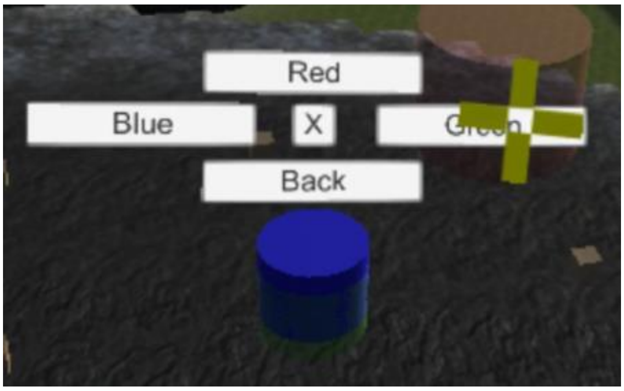


Figure 6.30: Object Matching Puzzle Circular Colour Sub Menu

Figure 6.29 shows the circular menu that appears when an object is selected. This replaces the "Deselect" option from Task 1. The X button functions as the Deselect, and the other menu options function the same as the modification menu did for Task 1. As seen in Figure 6.30, when the "Change Colour" menu was opened, the "Green" option was selected to finish the matching solution for this node.

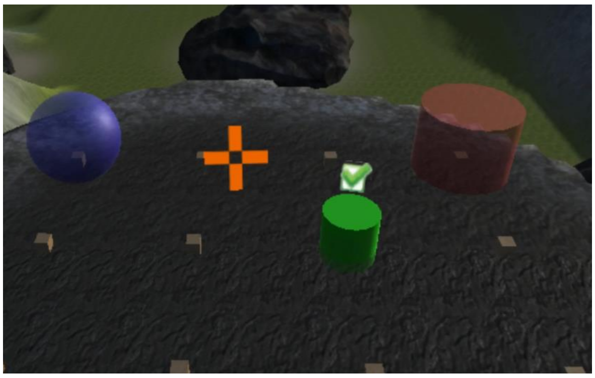


Figure 6.31: Object Matching Puzzle Matched Example

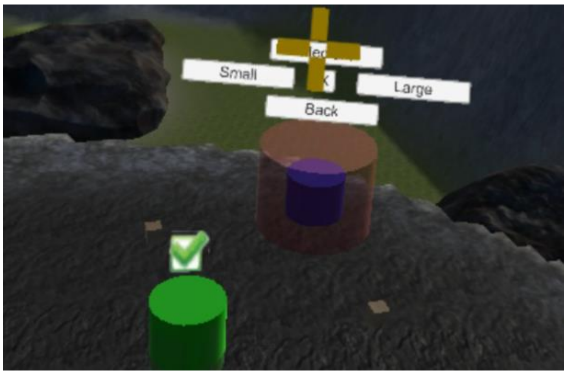


Figure 6.32: Object Matching Puzzle Circular Size Sub Menu

After the "Green" option was selected, the object is shown as completed in Figure 6.31. Figure 6.32 shows one more view of the circular menu demonstrating the size sub-menu. The shape sub-menu followed a similar type of interaction too. Unless the participant made a mistake choosing the type of shape they needed, this sub-menu was unnecessary. Figure 6.33 (over) shows the nearly completed puzzle with a single medium blue sphere to be constructed. Once the final object was completed, the participant would be taken to Figure 6.34 with a user response question. Used to evaluate the immediate post-task reaction to the circular menu.

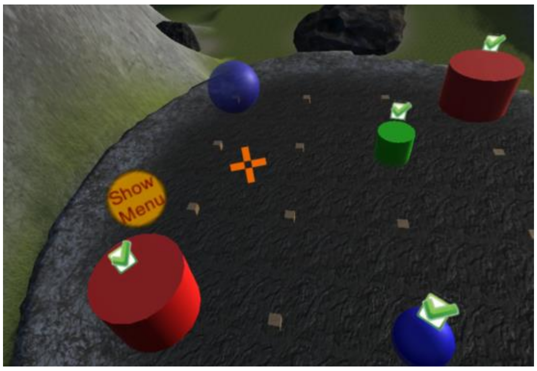
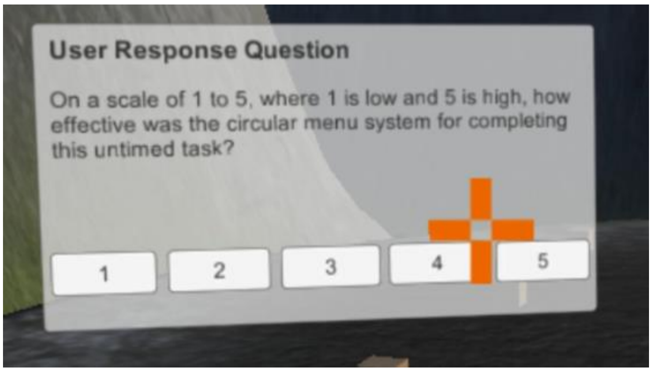


Figure 6.33: Object Matching Puzzle Near Completion


User Response Question

On a scale of 1 to 5, where 1 is low and 5 is high, how effective was the circular menu system for completing this untimed task?

Figure 6.34: Object Matching Puzzle Task 2 User Response Question

6.2.7 Task 3 and 4: Tower Defence HMD Only and Controller Inputs

From observing how participants played the tower defence task in the second experiment, it was evident that some changes could be made. The second experiment's unit versus tower scaling and a small area to place towers made the overall task relatively easy. Typically, a single frost tower to slow enemy units combined with a mix of basic and explosive towers would handle the enemy waves. From a gameplay perspective, it seemed beneficial to increase the combat time and increase the size of the area to add a slightly more positional strategy. The interactions within this task provide a good example of real-time use for the PVMS with the changes after the second experiment. There were many smaller changes to the gameplay to provide quality of life improvements. Some of these included:

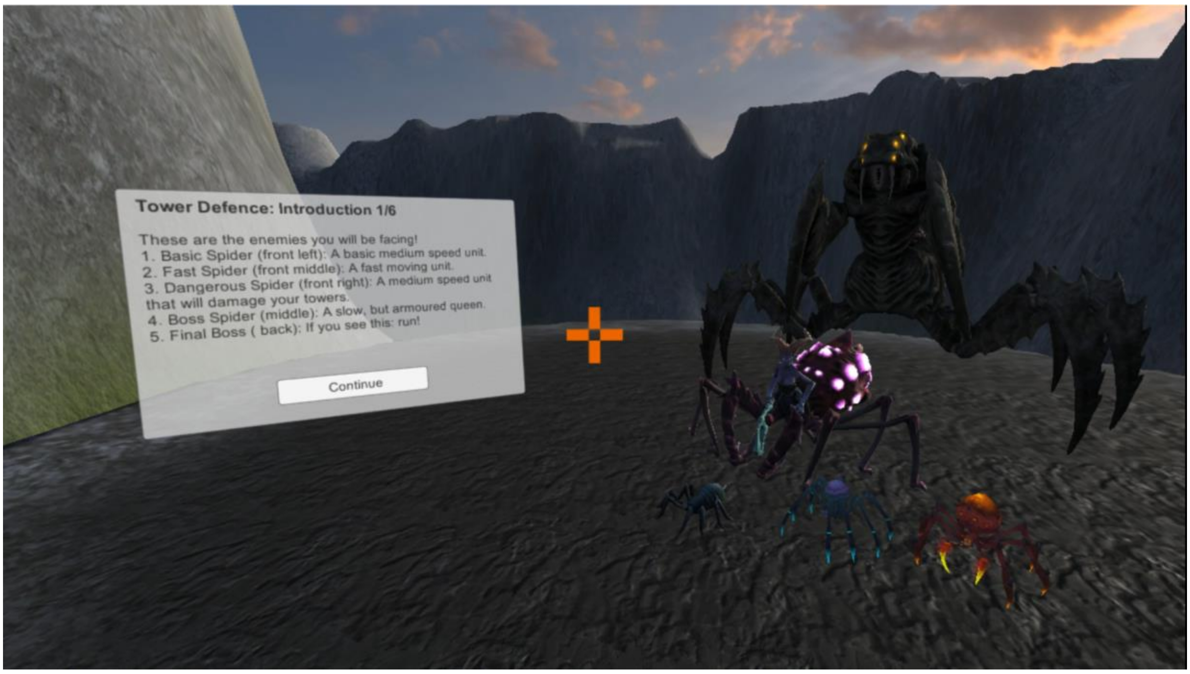
- Adding additional visual effects when important actions happen (spawning towers and the end boss spawning),
- Rebalancing the enemies as well as adding a new one for an improved active combat uptime, and
- Changes to the interface to present information or actions to the participant in a responsive way.

One of the most notable removals was the removal of the destroy tower feature. Choosing to destroy a tower was seldom a good idea in the second experiment's tower defence game. The normal time a function like this becomes important to use in a game like this is when all places for towers are filled, and you are looking to upgrade to more powerful towers. As this is just a single level with a reasonably short length, this was never something that would be necessary. Therefore, it was removed to focus on resource management, killing enemies for currency and then building/repairing towers with the currency.

The tower defence game was used to compare the PVMS with both a HMD Only type interaction and a controller. The controller was only used for providing an instant selection instead of the normal hover-to-select with a delay. The two tasks were always done in the same controller order. The repetition of the same task meant participants would have more experience approaching the task when reaching Task 4 with the controller. The following figures will introduce the introduction tutorial text provided to participants for Task 3 and Task 4.

Tutorial

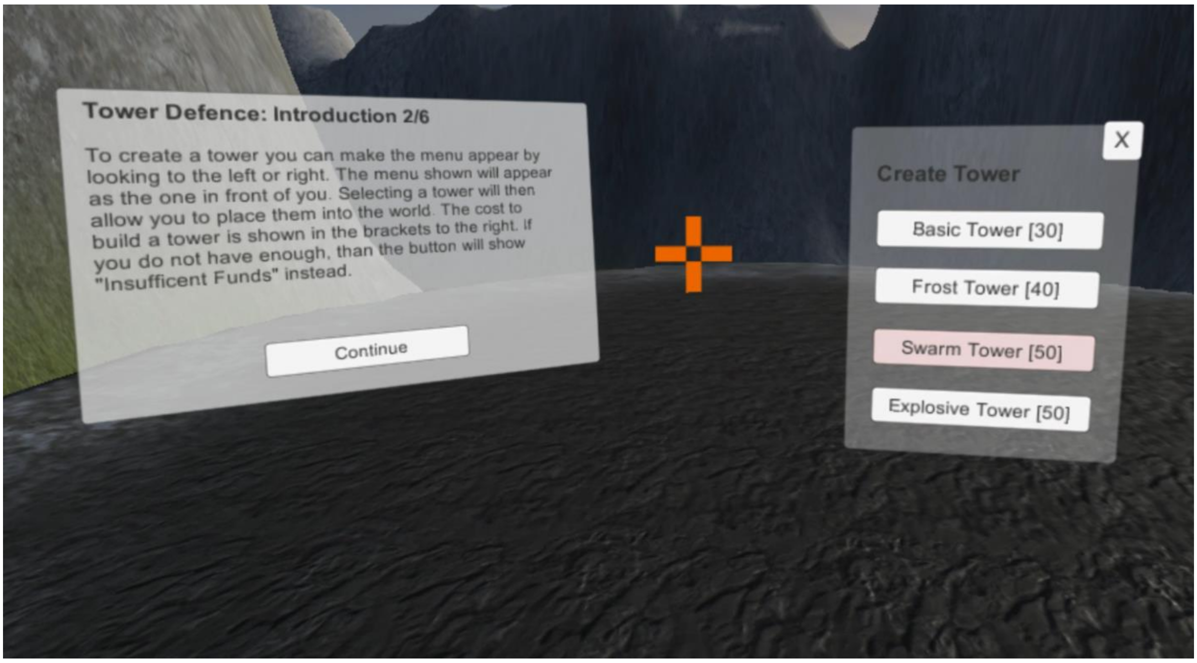
Figure 6.35 introduced the enemies the participant would be facing, including the new enemy. The sheer size of the final boss at the back is supposed to indicate how difficult this enemy would be to defeat. Figure 6.36 (over) introduces how participants will create towers with the Create Tower menu. As can be seen here, the costs for each tower were left the same as the second experiment.



Tower Defence: Introduction 1/6

These are the enemies you will be facing!
1. Basic Spider (front left): A basic medium speed unit.
2. Fast Spider (front middle): A fast moving unit.
3. Dangerous Spider (front right): A medium speed unit that will damage your towers.
4. Boss Spider (middle): A slow, but armoured queen.
5. Final Boss (back): If you see this: run!

Figure 6.35: Tower Defence Task Introduction 1/6

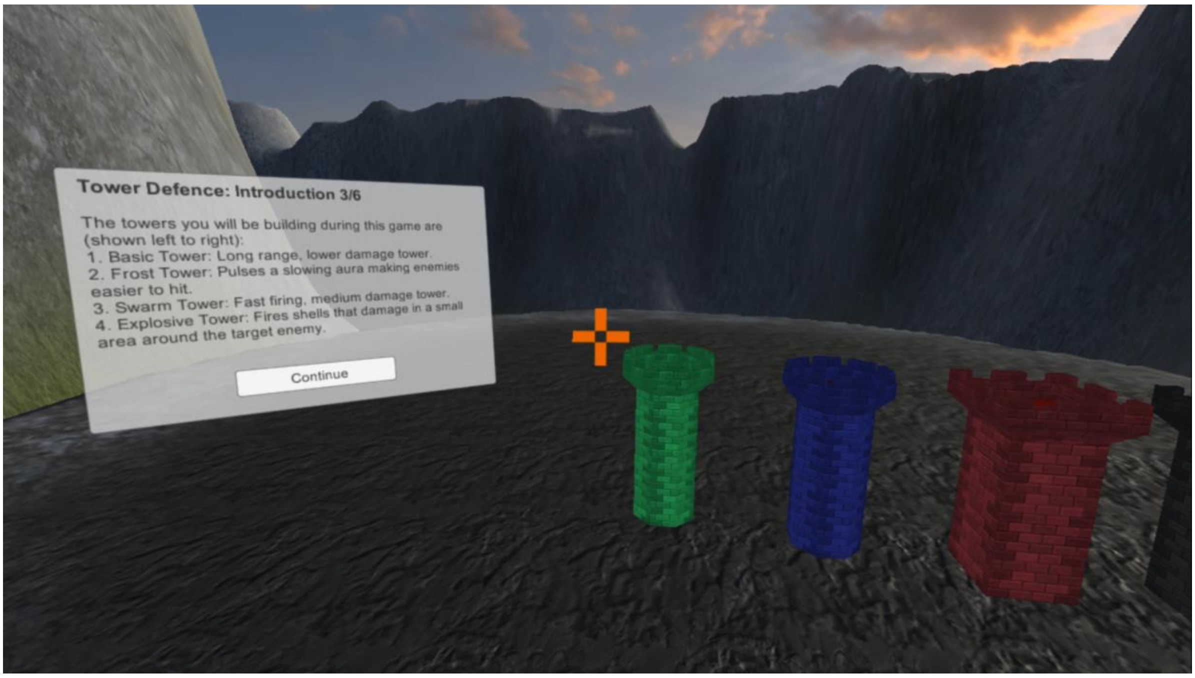


Tower Defence: Introduction 2/6

To create a tower you can make the menu appear by looking to the left or right. The menu shown will appear as the one in front of you. Selecting a tower will then allow you to place them into the world. The cost to build a tower is shown in the brackets to the right. If you do not have enough, than the button will show "Insufficent Funds" instead.

Figure 6.36: Tower Defence Task Introduction 2/6

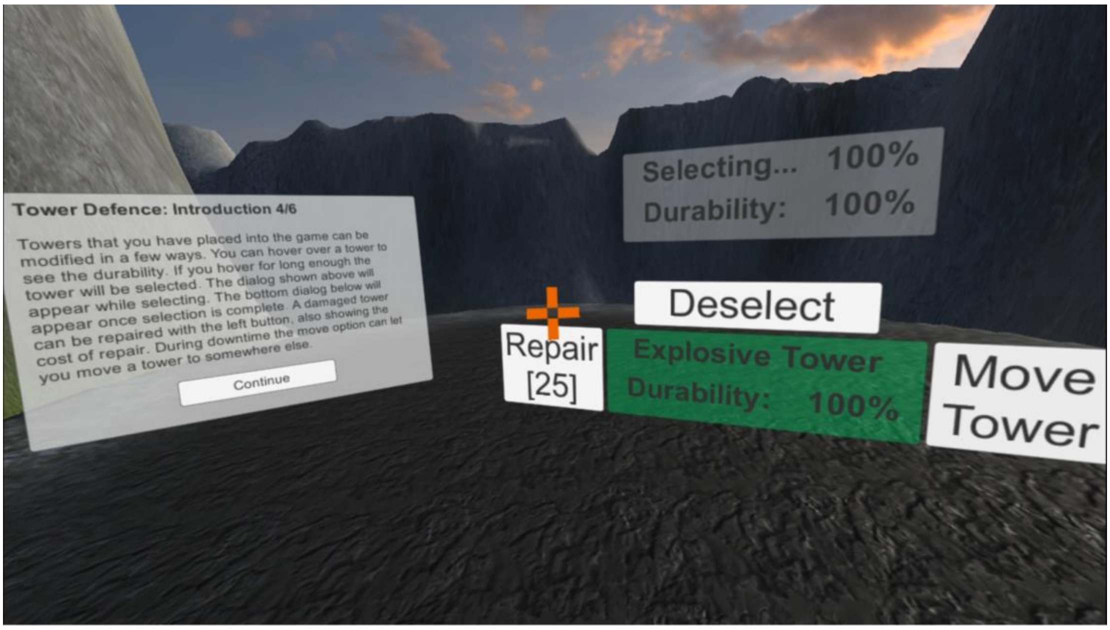
Figure 6.37 introduces each of the towers and their basic properties. The tower properties were introduced during the construction task in the second experiment because that was the first time the participant saw the towers. There was no construction task this time, so it was necessary to explain as part of this introduction.



Tower Defence: Introduction 3/6

The towers you will be building during this game are (shown left to right):
1. Basic Tower: Long range, lower damage tower.
2. Frost Tower: Pulses a slowing aura making enemies easier to hit.
3. Swarm Tower: Fast firing, medium damage tower.
4. Explosive Tower: Fires shells that damage in a small area around the target enemy.

Figure 6.37: Tower Defence Task Introduction 3/6

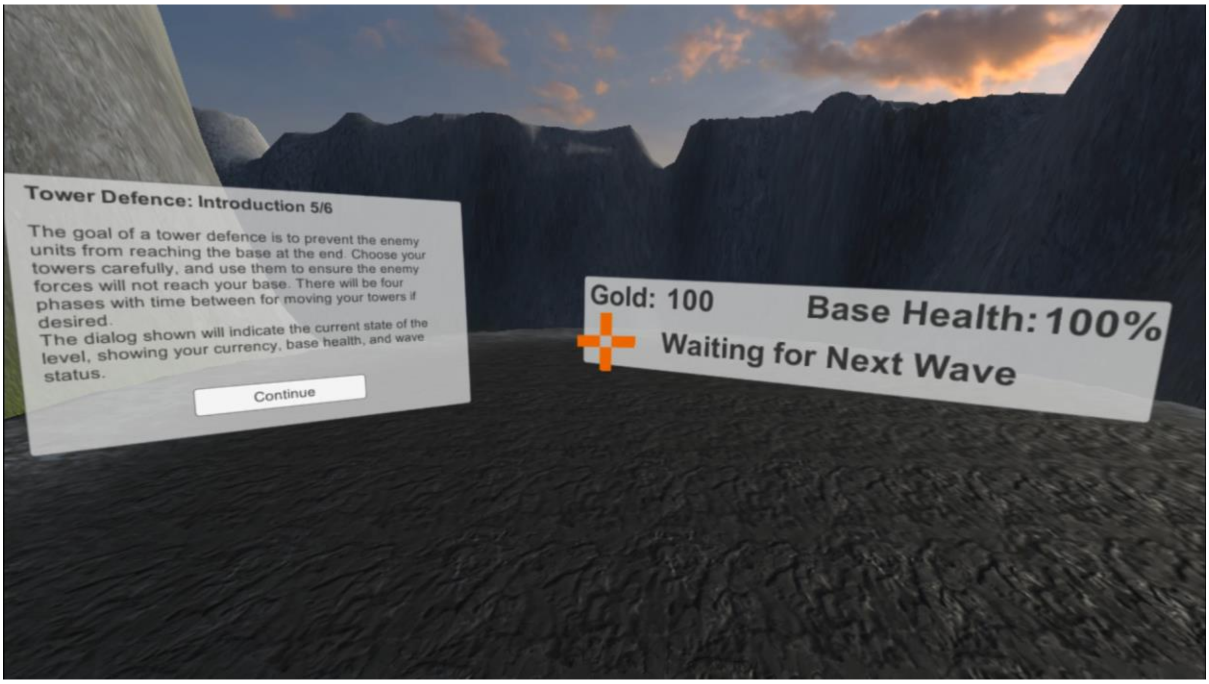


Tower Defence: Introduction 4/6

Towers that you have placed into the game can be modified in a few ways. You can hover over a tower to see the durability. If you hover for long enough the tower will be selected. The dialog shown above will appear while selecting. The bottom dialog below will appear once selection is complete. A damaged tower can be repaired with the left button, also showing the cost of repair. During downtime the move option can let you move a tower to somewhere else.

Figure 6.38: Tower Defence Task Introduction 4/6

Figure 6.38 shows the interfaces related to tower interaction. The information slide, visible here, explained how this functionality could be used for each of the Repair, Deselect, and Move Tower buttons. Repairs used the same formula for calculating repairs as the second experiment (section 5.2.8). The Deselect button would be visible all the time when a tower was selected. The Repair button was visible when the cost to repair was higher than 1 currency, and they had enough to pay. Move Tower was visible when there was scheduled downtime between waves. There was a bug in the game with the Move Tower button. A participant hovering where this button would have existed could still activate the button during waves. Only one participant realised this was the case, but it did not significantly impact the outcomes of their gameplay. It occurred because only the visible part of the button had been disabled, and the collision was still being checked.

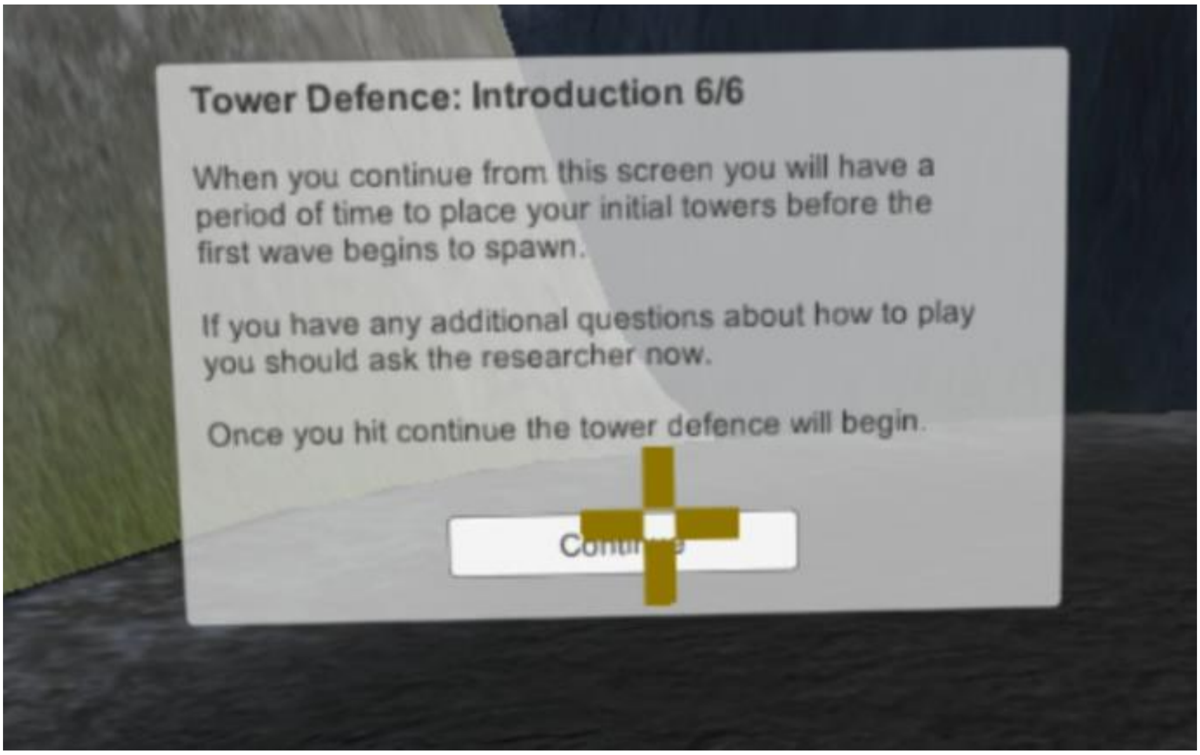


Tower Defence: Introduction 5/6

The goal of a tower defence is to prevent the enemy units from reaching the base at the end. Choose your towers carefully, and use them to ensure the enemy forces will not reach your base. There will be four phases with time between for moving your towers if desired.
The dialog shown will indicate the current state of the level, showing your currency, base health, and wave status.

Figure 6.39: Tower Defence Task Introduction 5/6

The interface showing current currency remained the same for this experiment, as seen in Figure 6.39. The figure shows the currency as gold in the top left, the base health in the top right, and the current wave status information in the bottom middle.



Tower Defence: Introduction 6/6

When you continue from this screen you will have a period of time to place your initial towers before the first wave begins to spawn.

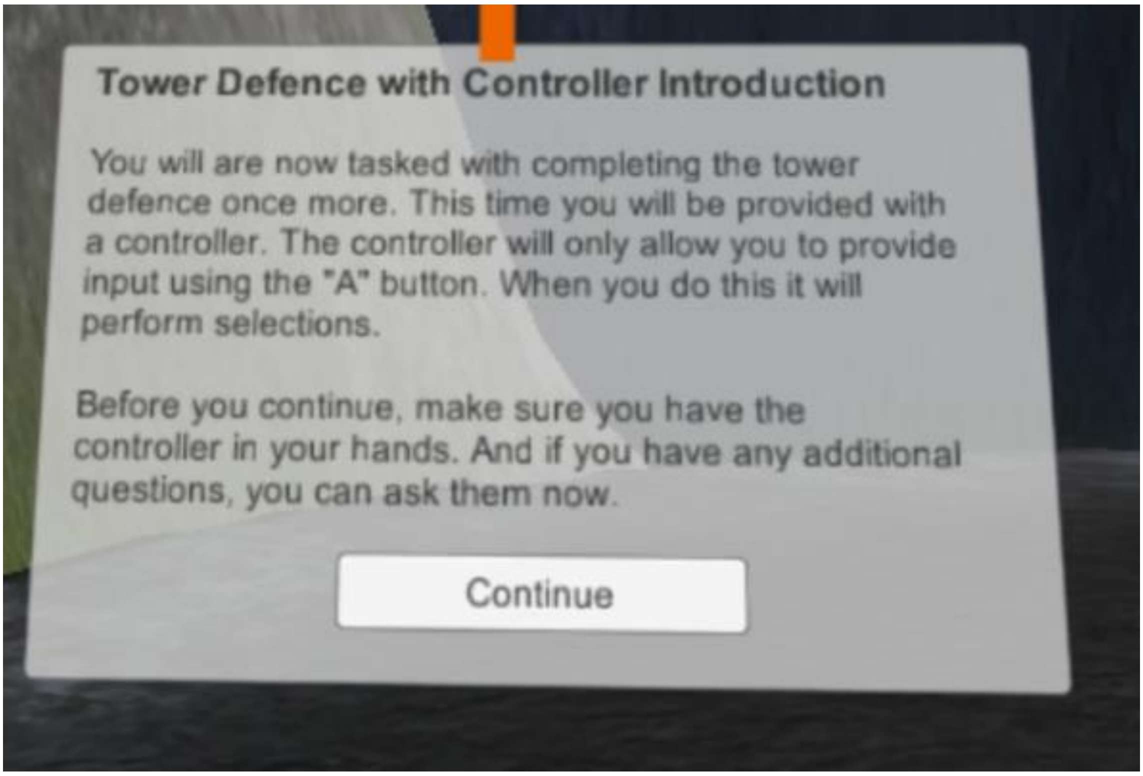
If you have any additional questions about how to play you should ask the researcher now.

Once you hit continue the tower defence will begin.

Figure 6.40: Tower Defence Task Introduction 6/6

Figure 6.40 (previous page) was the final introduction text provided to the participant. Information in these introductions was improved from the second experiment to use direct language, focusing on the important information while ensuring everything was apparent when it came to playing the game. As was the case in the second experiment, the final text left a time for the participant to ask any questions before the task commenced. Most participants jumped straight in with no questions asked.

Figure 6.41 shows the introduction text for Task 4. This introduction panel has been included here alongside the Task 3 content because the examples of the tasks would be nearly identical. The only other specific visual for Task 4 was the user response question. The question will be discussed after the example discussions. Two minor changes were hidden to participants during Task 4. The first was that a new tower that had not been yet placed could be cancelled with cost returned with the "B" button on the controller. The other change was to increase the health of all enemies by 10%. The health increase was included to increase the difficulty, making the difficulty feel comparable to the first time they had completed it, increasing the importance of tower placement and tower choice that came from completing Task 3.



Tower Defence with Controller Introduction

You will are now tasked with completing the tower defence once more. This time you will be provided with a controller. The controller will only allow you to provide input using the "A" button. When you do this it will perform selections.

Before you continue, make sure you have the controller in your hands. And if you have any additional questions, you can ask them now.

Figure 6.41: Task 4 Tower Defence with Controller Introduction

Example Walkthrough of Gameplay

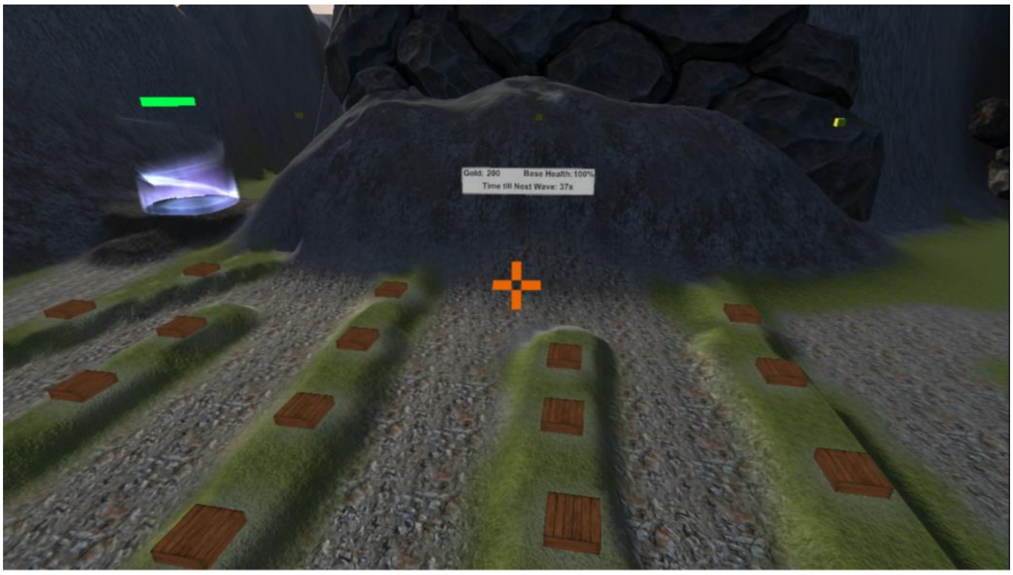


Figure 6.42: Tower Defence Task Starting Perspective

Figure 6.42 shows an example of what would be visible once the participant had completed the introduction. A starting amount of 200 gold was given to purchase the initial number of towers that could be built. The starting amount increased compared with the gold provided in experiment two and was intended to balance against increased health and a larger number of enemies that needed to be dealt with at once. After each wave of enemies, an additional 100 gold was automatically awarded as a bonus. The time between waves for this experiment decreased the time from the second experiment. A downtime period of 40 seconds before the first wave of enemies and 30 seconds before every other wave of enemies was given to provide time to build towers, move towers, or repair towers. Figure 6.43 shows an example of the opened Create Tower menu.

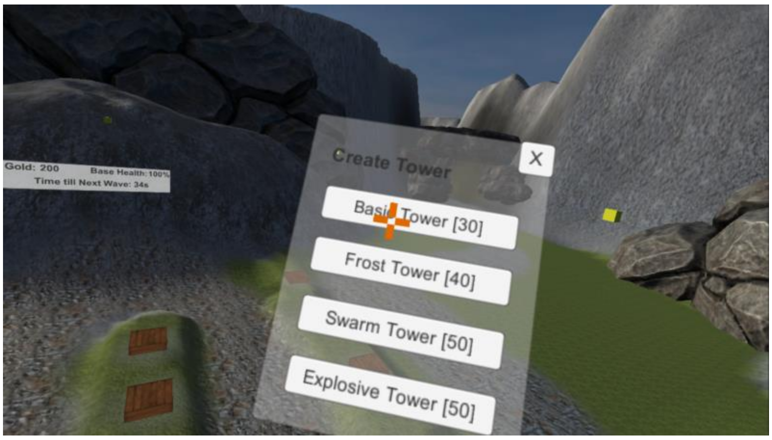


Figure 6.43: Tower Defence Task Create Tower Menu

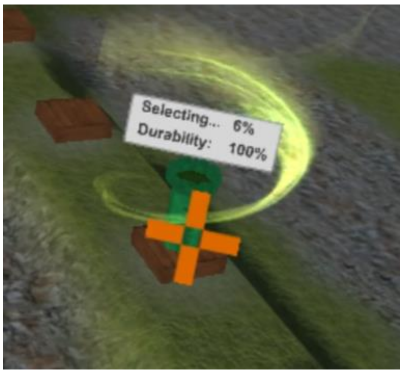

Selecting... 6%
Durability: 100%

Figure 6.44: Tower Defence Task Tower Creation Effect and Selection

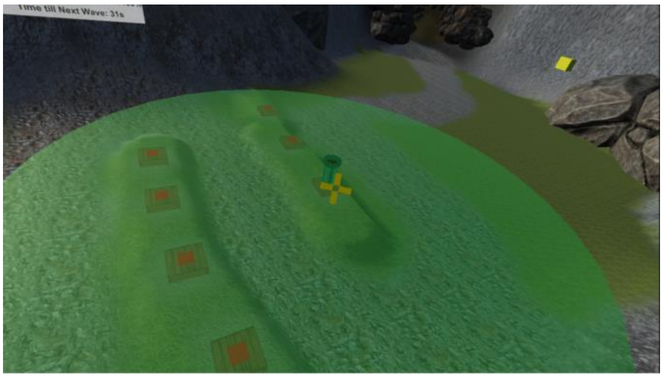

Time till Next Wave: 31s

Figure 6.45: Tower Defence Task Tower Range Example

Once a tower was created, a visual effect would swirl around to make the creation move visually stimulating. Figure 6.44 shows the yellow tinted swirl that would spiral down to the tower base and then despawn after a second. Figure 6.44 also shows an example of the selecting dialog. The selecting dialog was visible any time a tower was hovered over, showing the current durability and the selection progress. Figure 6.45 shows an example of the range for the basic tower. In this specific example, the tower is currently being snapped to a location for placement. Showing the range was useful during placement to provide player feedback and enable planning and strategy.

Figure 6.46 shows an example of having insufficient funds. The player has 10 gold remaining, so all the tower purchase options cannot be met. Figure 6.47 shows a view of the cave as the first wave of enemies are approaching. The wave status dialog has been updated to show "Wave 1/4 Contains 29 Enemies" to give the participant an idea of what to expect.

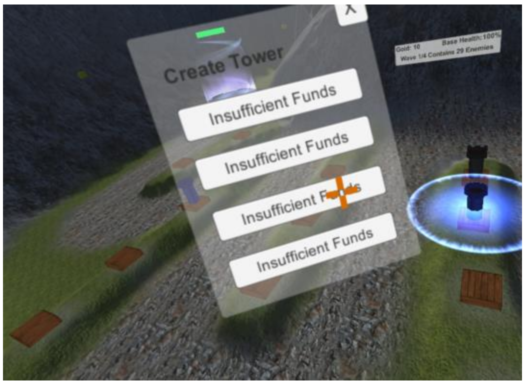

Create Tower
Insufficient Funds
Insufficient Funds
Insufficient Funds
Insufficient Funds
Gold: 10 Base Health: 100%
Wave 1/4 Contains 29 Enemies

Figure 6.46: Tower Defence Task Example of Insufficient Funds

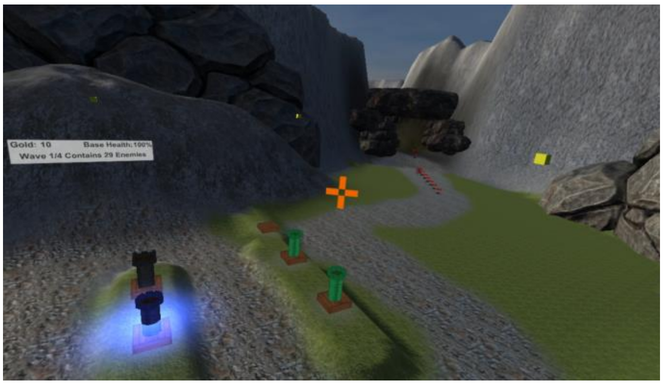

Gold: 10 Base Health: 100%
Wave 1/4 Contains 29 Enemies

Figure 6.47: Tower Defence Task Enemies Approaching

The **`CameraSnapNodes`** were used more frequently in this experiment than the others. This was assumed due to the larger area and a desire to get different perspectives. The starting viewpoint was considered the easiest to control everything from, but sometimes controlling towers was made easier by moving closer. Figure 6.48 shows the cursor interacting with one of these **`CameraSnapNodes`** next to the teleport pad. The teleport pad demonstrates the reused effect from the tower construction task of experiment two. Figure 6.49 shows the updated position and perspective after the camera was transitioned to the new location. The participant could at any time look to another yellow cube to reposition themselves again.

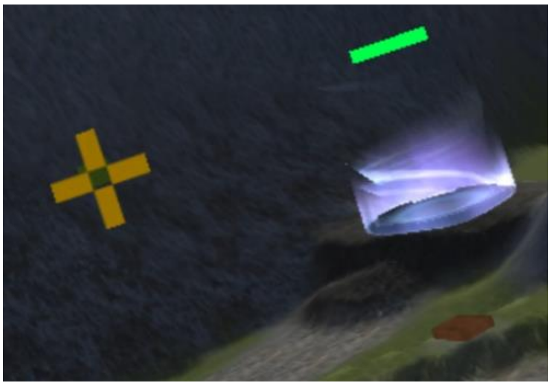


Figure 6.48: Tower Defence Task Using CameraSnapNode and Teleporter Pad

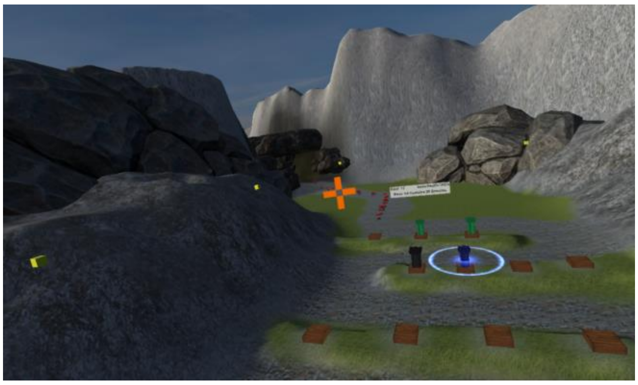


Figure 6.49: Tower Defence Task Alternate Perspective

The change from having tower modification options as a PVMS to the hovering over tower type interface was made to provide a different experience to the second experiment. Task 1 had already used contextual menus based on selection, so it was less necessary to test this in all tasks. The idea for changing this was to blend the interfaces naturally into gameplay with two distinct interfaces between the creation and modification interactions. Figure 6.50 shows what was visible most of the time with this interface, as was previously discussed in the introduction for the tasks. Figure 6.51 shows the interface with all buttons visible for when there is downtime between waves.

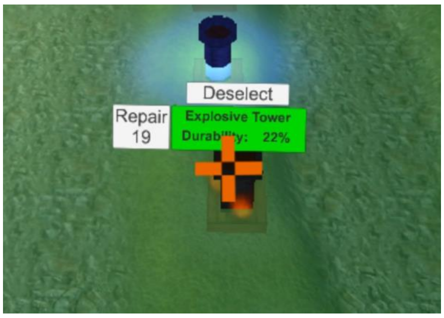


Figure 6.50: Tower Defence Task Selected Tower Example

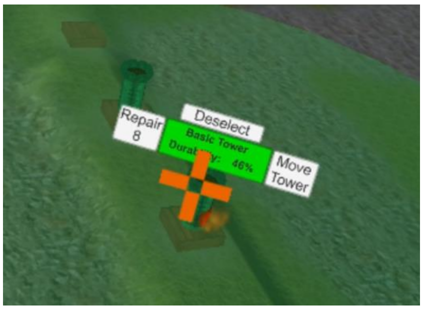


Figure 6.51: Tower Defence Task Selected Tower All Buttons

The background colour would change on the dialogs to provide a context cue. Figure 6.52 shows how the ordinarily grey dialog is rendered with a red background when the durability is 0% to alert the participant to the importance of that tower. Figure 6.53 shows how this would then look when the tower has been successfully selected. The selection is shown with a yellow background informing the player of a problem with the tower that needed attention. For more discussion about the main gameplay interactions, refer to the second experiment's tower defence task discussions (section 5.2.8).

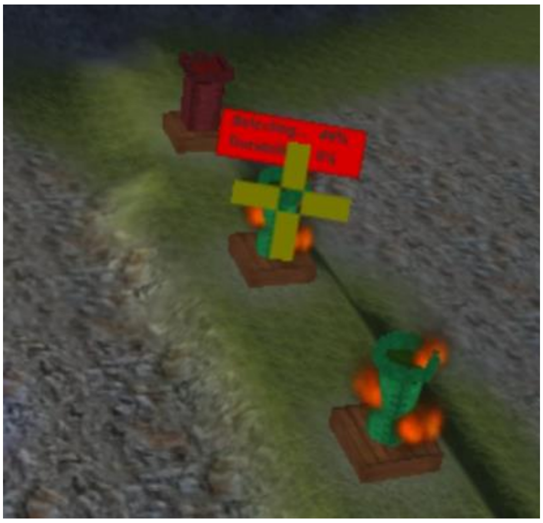


Figure 6.52: Tower Defence Task Destroyed Tower Selecting

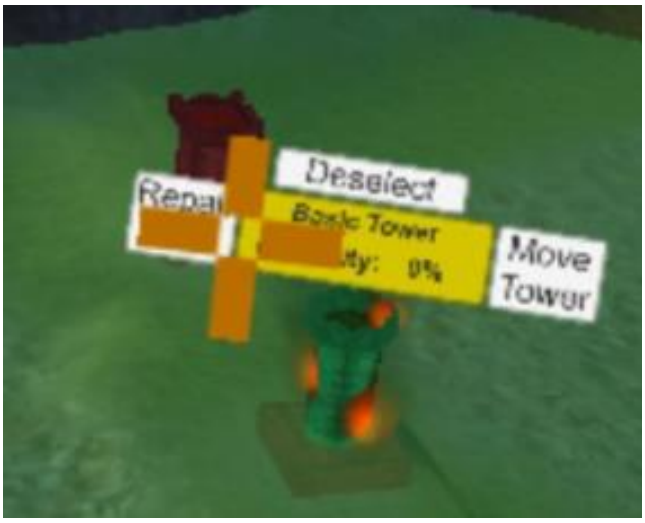


Figure 6.53: Tower Defence Task Destroyed Tower Selected

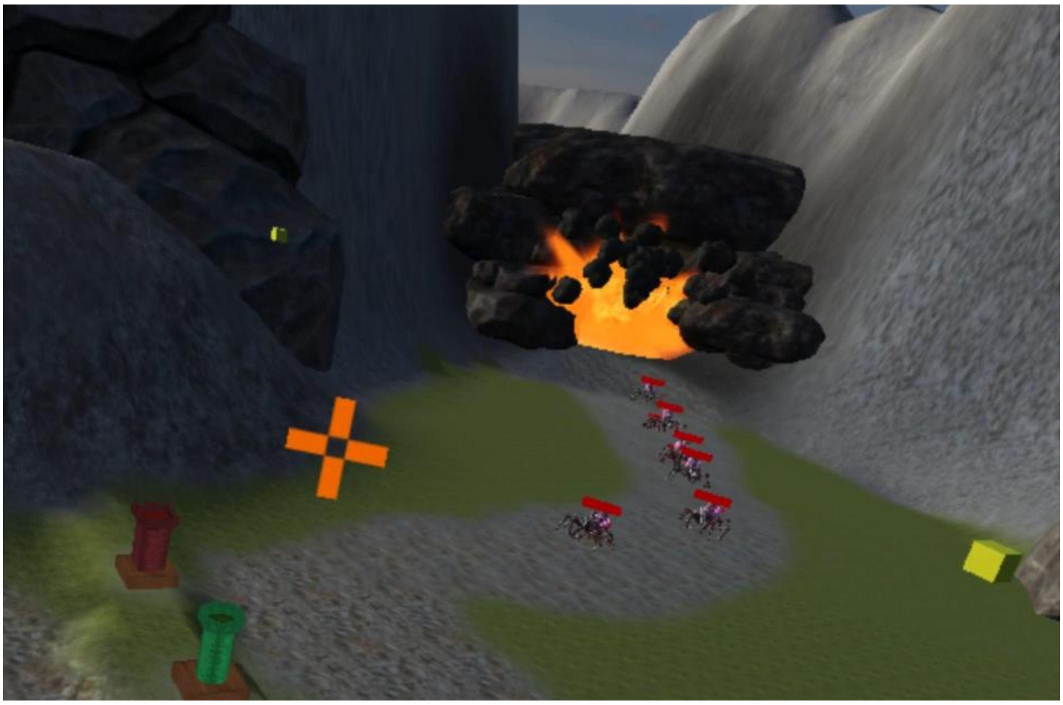


Figure 6.54: Tower Defence Task Cave Explosion

After making the trailer for YouTube, the visual effect of the cave exploding felt stimulating and motivating for a player. As a result, it was included in the spawning sequence for the final boss. The effect involved an explosion visual effect and animation of all the rockets launched by the explosion. The effect can be seen in Figure 6.54 (previous page) as the rocks begin to fly. The effect is further described earlier in the recruitment section about the YouTube video (section 6.2.1).

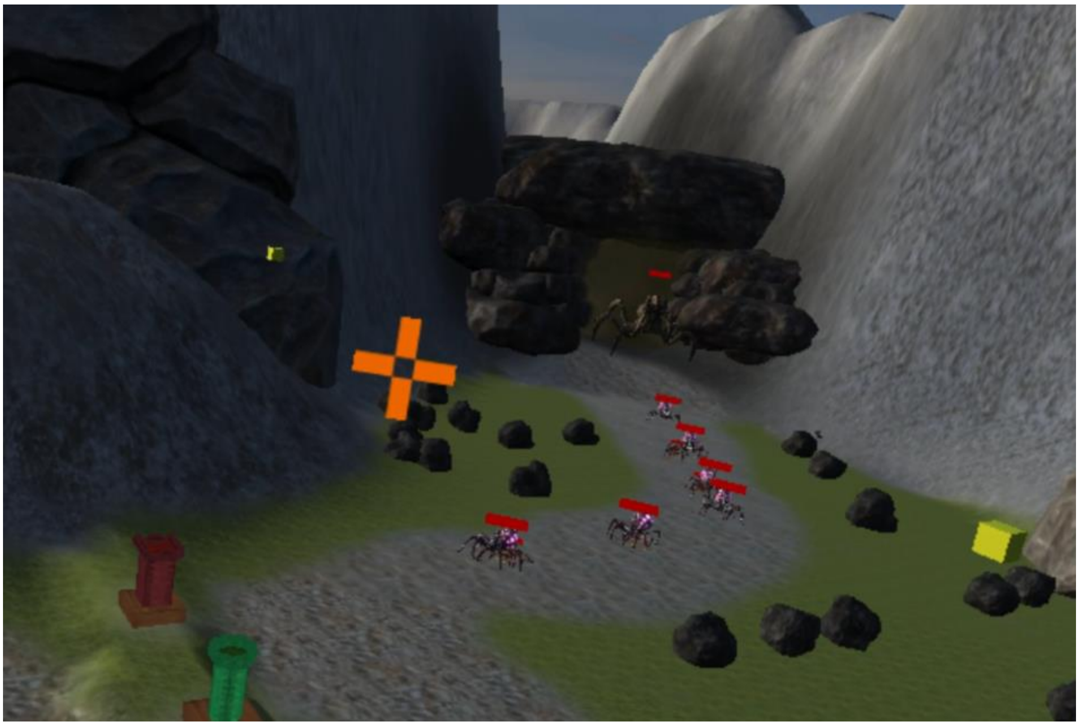


Figure 6.55: Tower Defence Task Final Boss Spawned

Figure 6.55 shows the rocks settled in their final resting positions as the wave of comparably smaller boss enemies approaches, with the final boss following behind as the explosion settles. It was possible to defeat the boss enemy with smart gameplay. Very few participants successfully defeated it before the enemy reached the teleport pad. The difficulty in defeating this enemy came down to two factors. The comparably large amount of health it had compared to other enemies made it survive longer against attacks. The creature's attacks would instantly destroy any tower it was in the range of on the path. A good balance of towers was necessary to defeat the enemy combined with repairing towers only after the creature was out of range. Many participants tried to repair towers while the boss was still in range, resulting in their towers being instantly destroyed again. A perspective of the final boss performing an attack is shown in Figure 6.56 (over).

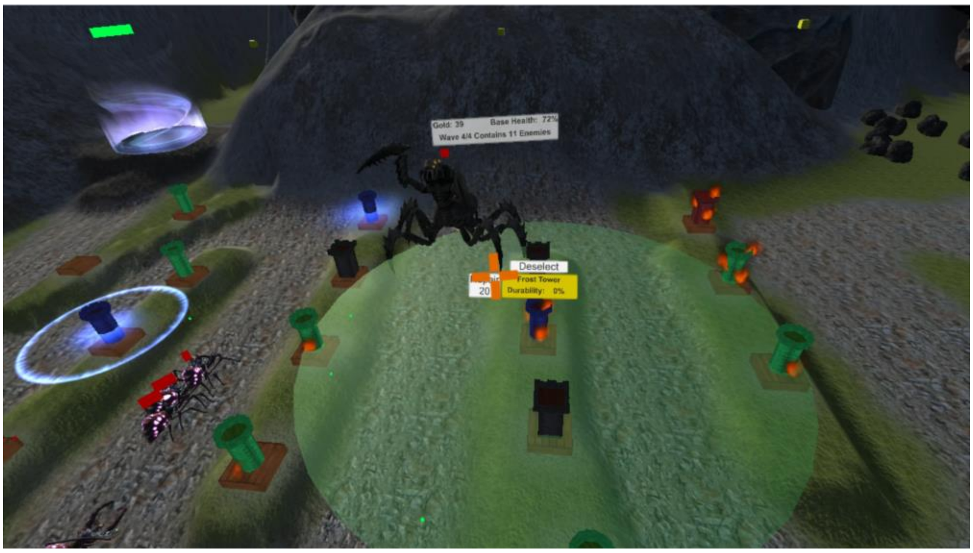
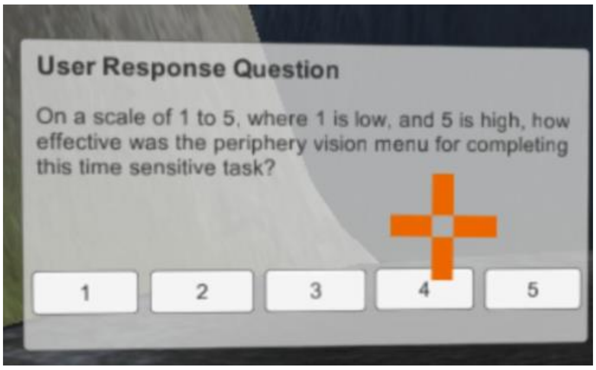
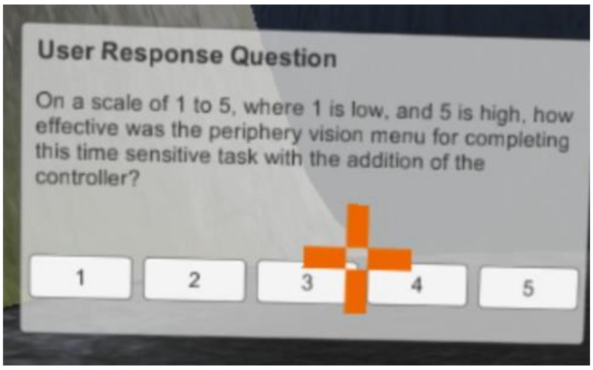


Figure 6.56: Tower Defence Task Final Boss Attacking


User Response Question

On a scale of 1 to 5, where 1 is low, and 5 is high, how effective was the periphery vision menu for completing this time sensitive task?

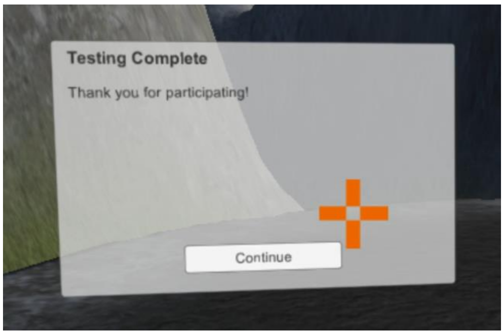

User Response Question

On a scale of 1 to 5, where 1 is low, and 5 is high, how effective was the periphery vision menu for completing this time sensitive task with the addition of the controller?

Figure 6.57: Task 3 Tower Defence with HMD Only User Response Question

Figure 6.58: Task 4 Tower Defence with Controller User Response Question

After completing Task 3 with the final boss wave, the participant was provided with the question shown in Figure 6.57. Similarly, after completing the final boss wave of Task 4, the participant was asked the question in Figure 6.58. Finally, after completing the response question for Task 4, the player was presented with the slide seen in Figure 6.59 to indicate the experiment was complete.


Testing Complete

Thank you for participating!

Figure 6.59: Experiment 3 Complete Dialog

Experiment Math and Properties

Some of the data used to populate the properties of this experiment have been included here to provide additional context. The first data in Table 6.1, is showing the enemy properties. The values were rebalanced to where they felt competitive against the different layout and different tower attack values. During Task 4, the health of all enemies was increased by 10% to increase challenge based on the learning effect gained from the first time through.

Table 6.1: Enemy Properties

Enemy Name	**Basic Spider**	**Fast Spider**	**Dangerous Spider**	**Boss Spider**	**Final Boss***
Move Speed	7*	11	7	6*	5*
Rotation Speed	9	15	9	3	3*
Damage to Base	1	1	1	1	1*
Max Health	450*	400*	1000*	3000*	15000*
Fire Rate	N/A	N/A	2 seconds*	N/A	3*
Max Range	N/A	N/A	15	N/A	15*
Damage	N/A	N/A	0.2 (20%)*	N/A	20*

* = value changed from second experiment.

Wave definitions are shown below for all the stages. Each stage, except for the final boss stage, had three individual waves. The text briefly indicates the types of enemies spawned with quantities. The wave structure notations used for the creation of these waves are discussed in section 5.2.8. The waves of enemies for this experiment used the same enemies as the second experiment except for the boss stage. A single final boss enemy was spawned 5 seconds after the others to create a pause for the destructive entry with the cave explosion. Although the waves were defined the same, they felt different from gameplay. The difference in feel was because, in the second experiment, enemies could be attacked immediately after spawning. In this experiment, there was a marching period before they came into the range of towers. Different enemy speeds would result in different timings for towers to deal with enemies. The full expanded data definition with **`WaveCommand`** format can be found in Appendix B.1.7 and Appendix C.1.8.

- Stage One Definition (3 waves)
  - Wave 1: 5 basic
  - Wave 2: (2 basic, 2 fast) twice
  - Wave 3: (4 basic, 4 fast) twice
- Stage Two Definition (3 waves)
  - Wave 1: 5 dangerous
  - Wave 2: (2 fast, 1 dangerous) three times
  - Wave 3: (1 basic, 1 dangerous, 1 fast) three times
- Stage Three Definition (3 waves)
  - Wave 1: 12 fast
  - Wave 2: (2 basic, 4 fast) twice, then 5 dangerous
  - Wave 3: (2 basic, 2 dangerous, 2 fast) three times, then 1 boss
- Boss Stage Definition (1 wave)
  - boss, dangerous, boss, dangerous, boss, boss, dangerous, boss, dangerous, boss, final

Table 6.2: Tower Properties

Tower Name	Damage	Range	Rate of File
Basic Tower	20	40*	0.5s
Frost Tower	0.5*	25*	1s
Swarm Tower	20	30*	0.2s
Explosive Tower	30*	30*	0.8s

* = value changed from base values of the second experiment.

Table 6.2 shows the values used for this experiment. These were rebalanced to account for the far larger area of gameplay. The frost tower in the second experiment was a little too strong, so the slow was reduced from 60% base to 50%. The frost tower was given a range of 25, increased from 15 in experiment two to make up for the reduced slow. The explosive tower had been a little too strong, so the base damage was decreased to 30 from 35. The rate of fire property remained unchanged because they felt right relative to the types of towers. The following data elaborates on the cost-effectiveness data used when balancing the towers. The following list defines the formulas as they are used in Table 6.3 below.

- DPS = (damage x fire rate)
- DPS Effectiveness = ((DPS * Range) / 10)
- Max Repair Cost = (TowerCost / 2)
- Shots Till Repair = (1/0.005) = 200 shots
- Time Between Repairs = (ShotsTillRepair / (1 / FireRate))
- Cost Effectiveness = (DPSEffectiveness / Tower Cost)
- Cost for 300 Seconds = (TowerCost + RepairCost*300/TimeBetweenRepairs)
- Damage over 300 seconds = (300 * DPS)
- Damage per gold over 300 seconds = (DamageOver300Seconds / CostFor300Seconds)

Table 6.3: Tower Balancing Math

Tower Name	**Basic Tower**	**Frost Tower**	**Swarm Tower**	**Explosive Tower**
DPS	40	N/A	100	1 target = 37.5 2 targets = 75 3 targets = 112.5 4 targets = 150
DPS Effectiveness	160	N/A	300	1 target = 112.5 2 targets = 225 3 targets = 337.5 4 targets = 450
Tower Cost	30	40	50	50

Repair Cost	15	20	25	25
Shots Till Repair	200	200	200	200
Time Between Repairs	100	200	40	160
Cost Effectiveness	5.3333	N/A	6	1 target = 2.25 3 targets = 6.75
Cost for 300 Seconds	75	70	237.5	96.875
Damage over 300 Seconds	12000	N/A	30000	1 target = 11250 3 targets = 33750
Damage per Gold over 300 Seconds	160	N/A	126.32	1 target = 116.13 3 targets = 348.39

The numbers shown in Table 6.3 indicate the numbers used to ensure towers were effective and not under or overpowered. When considering the cost-effectiveness of any single tower, it was necessary to consider the cost against how much damage it would be doing with optimal uptime. From the damage per gold over 300 seconds, it is evident that the basic tower is far superior with the most economical cost per damage for a single target. At 237.5 cost per 300 seconds, the swarm tower is challenging to maintain with available currency in the game. When it comes to the area of effect splash damage, the explosive tower loses out on 1 and 2 targets to the basic tower, but once there are 3 or more enemies in the splash radius, the cost-benefit begins to look positive. For this reason, a good balance between basic towers for defeating the final boss and explosive towers for defeating the smaller enemies was a good mix with some cheap frost towers to increase the uptime for the attack of the other towers.

6.2.8 Pre and Post Experiment Questionnaires

As part of the experiment, each participant completed two questionnaires to evaluate their feedback related to the experiment and associated topics. The questions are summarised to represent their point without the language targeting participants where appropriate. The questions are similar to the second experiment, focusing on the added features between the experiments. Complete surveys with original question wording are available in Appendix C.

Pre-Experiment 15 questions

- Questions 1 to 4: Personal and Academic Background covering questions related to Age, Gender, Student/Other, Area of Study/Teaching/Research.
- Questions 5: Mark all that apply. Participated in the first experiment, second experiment, other VR/AR research (specify), other HMD research (specify).
- Questions 6 to 8: Use of Computers
    - How many hours a week would you use a computer on average?
    - How many hours a week would you spend playing video games on average?
    - Examples of computer games and consoles typically played on.
- Question 9: Interest in using HMDs for gaming activities. 1 = Not Interested, 10 = Very Interested.
- Question 10: Interest in using HMDs for non-gaming activities. 1 = Not Interested, 10 = Very Interested.
- Question 11: Frequency of having played tower defence games. 1 = Never, 7 = Very Often.
- Question 12: Specify any examples of using the head as an interaction tool.
- Question 13: Select the option that most applies: (space for additional comments)
    - I am planning on buying a HMD (specify model).
    - I already own a HMD (specify model).
    - I am undecided and waiting to see more of VR/AR before deciding.
    - I am not planning on buying a HMD.
- Question 14: On a scale of 1 to 7 (1 = no influence, 7 = high influence), how much did the game trailer influence desire to participate in the research.
- Question 15: On a scale of 1 to 7 (1 = no influence, 7 = high influence), feeling toward visual presentations such as the game trailer influencing desire to participate in future research.

Post-Experiment 13 questions

- Question 1: How often the PVMS displayed at the wrong times or when not meaning to display it. Scale of 1 to 7. 1 = not often, 7 = very often.
- Question 2: How often the PVMS displayed at the correct times. Scale of 1 to 7. 1 = not often, 7 = very often.
- Question 3: How useful the gesture felt of rotating the head to make a menu appear. Scale of 1 to 7. 1 = not useful, 7 = very useful.
- Question 4: How likely would it be for wanting to use the interaction again in the future. Scale of 1 to 7. 1 = not likely, 7 = very likely.
- Question 5 and 6: Three aspects found most enjoyable and most difficult while using the PVMS.
- Question 7: System Usability Scale with all questions using 1 to 5 scales of strongly disagree to strongly agree. All questions used language focusing on the menu system so that participants would understand they were about the menus and not the application in general.
    - "I think that I would like to use this menu system frequently."
    - "I found the menu unnecessarily complex."
    - "I thought the menu was easy to use."
    - "I thought that I would need the support of a technical person to be able to use this menu."
    - "I felt that options presented by menus were well integrated."
    - "I felt that there was too much inconsistency with the menu."
    - "I would imagine that most people would learn how to use these menus very quickly."
    - "I found the menus very cumbersome to use."
    - "I felt very confident using the menus."
    - "I needed to learn a lot of things before I could use the menus."
- Question 8: Rank from 1 to 7 activities based on where the menu system would be most desirable. 1 = most desired, 7 = least desired. Options: Games, Viewing a Movie/TV Shows/Virtual Cinema, Constructing Models, Instant Messenger or Voice

Chat, Operating System Controls/Menus, Virtual Tour Guide, Browsing the Internet. Additional space to list any other scenarios not listed in this question.

- Question 9: The significance when considering the usability of user interfaces for HMDs with either AR or VR. Ranking from 1 to 4 the features: speed of accessing features, the accuracy of accessing features, simplicity of physical interactions, the visual appeal of the interface. With the option of listing any other features that were felt to be important.
- Question 10: How useful did you find the circular menu fixed to an object as compared to completing actions with the PVMS. Scale of 1 to 7. 1 = not useful, 7 = very useful.
- Question 11: How useful did you find the PVMS as compared to completing actions with the circular menu. Scale of 1 to 7. 1 = not useful, 7 = very useful.
- Question 10: How did you find the experience of interreacting with the addition of the Xbox controller as a selection tool compared to using the head alone as a selection tool. Scale of 1 to 7. 1 = worse experience, 4 = similar experience, 7 = worse experience.
- Question 13: Any other thoughts or suggested changes about the experiment.

6.2.9 Application Logged Data

As with the other experiments, while the Unity application was running automatic data collection was occurring. For each participant, four different data files were generated with the following naming formats (an explanation of these formats can be found in section 5.2.11). The only file with significant differences was the CSV file due to the different tasks conducted during this experiment.

- yyyy_M_d__hh_mm.log: Text logs of events.
- yyyy_M_d__hh_mm.autosave: AutoSave serialised data.
- yyyy_M_d__hh_mm.dat: Replay data binary file.
- yyyy_M_d__hh_mm.csv: Categorised extra data log.

Extra Data Logs (CSV)

The following list extends from data collated into CSVs from the second experiment, tracking all the important elements from the second experiment that carried over while extending the data stored based on the needs of new tasks. The data logging that was the same as the second experiment has been noted in the list. New additions include data stored to track the two-step widget, the circular menu, and metrics from the matching puzzle.

- Menu Behaviour Log (Same as Second Experiment): Timestamp, Game State, Menu Definition, Menu Up Time, Menu Hide Reason, Button Name, Button ID, Button Result Code.
- PVMS Events (Same as Second Experiment): Timestamp, Game State, Rotation Result, Show Menu, Sensitivity Setting, **`Diff_X`** (Pitch), **`Diff_Y`** (Yaw), **`Diff_Z`** (Roll), History Count, Full History of Quantity Count with X;Y;Z;DeltaTime with oldest to newest events.
- Time Log (Same as Second Experiment): Timestamp, Game State, Time Since Last State
- User Response Questions (Same as Second Experiment): Timestamp, Question, Response.
- Two-Step Menu Log: Timestamp, Game State, Menu, Menu Uptime, Menu Hide Reason.
- Tower Defence Stats (Same as Second Experiment): Timestamp, Level State, Tower Basic Built, Tower Frost Built, Tower Swarm Built, Tower Explosive Built, Tower Repaired, Tower Destroyed, Units Killed, Failed Tower Buy, Failed Repair, Camera Snap Used, Base Health.
- Circular Menu Log: Timestamp, Game State, Menu, Menu Uptime, Menu Hide Reason, Button Name, Button ID, Button Result Code.
- Periphery Match Puzzle Metrics: Variable (MinimumOperationCount, OperationCount, Difference), Spawn Cylinders Count, Spawn Cubes Count, Spawn Spheres Count, Set Small Count, Set Medium Count, Set Large Count, Set Blue Count, Set Red Count, Set Green Count, Set Cylinder Count, Set Cube Count, Set Sphere Count.
- Circular Match Puzzle Metrics: Same as Periphery Match Puzzle Metrics.
- Tower Defence Stats with Controller: Same as Tower Defence Stats.

6.3 Results

Like previous sections, the results will be presented to show the spread of data collected from this experiment. The questionnaire results in section 6.3.1 cover the questions listed in section 6.2.8. The application data results in section 6.3.2 cover the relevant metrics pulled from the data described in section 6.2.9. Following on from this section is a discussion summarising the results in section 6.4.

6.3.1 Questionnaire Results

The participant responses have been collected and combined to generate the data in this section. The order of presentation is based on where it was presented to the participants in the questionnaires. For a summary of the questionnaire questions, it can be seen in section 6.2.8 or for the complete questionnaires as they were presented to the participants, see Appendix C.2.

General Background

A total of 26 participants completed the entire experiment. The participants included 24 males and two females. 21 participants were students, and 5 were affiliated with the university as teaching staff or involved in research. Figure 6.60 shows the wide age distribution of participants. Participants' areas of study included Computer Science, Engineering, Biomedical, Psychology, Forensic Biology, and Artificial Intelligence. Three individuals participated in all three experiments. Three additional individuals participated in both experiments 2 and 3. Six participants had previously participated in VR/AR research unrelated to this research. One participant had participated in HMD research unrelated to this research.

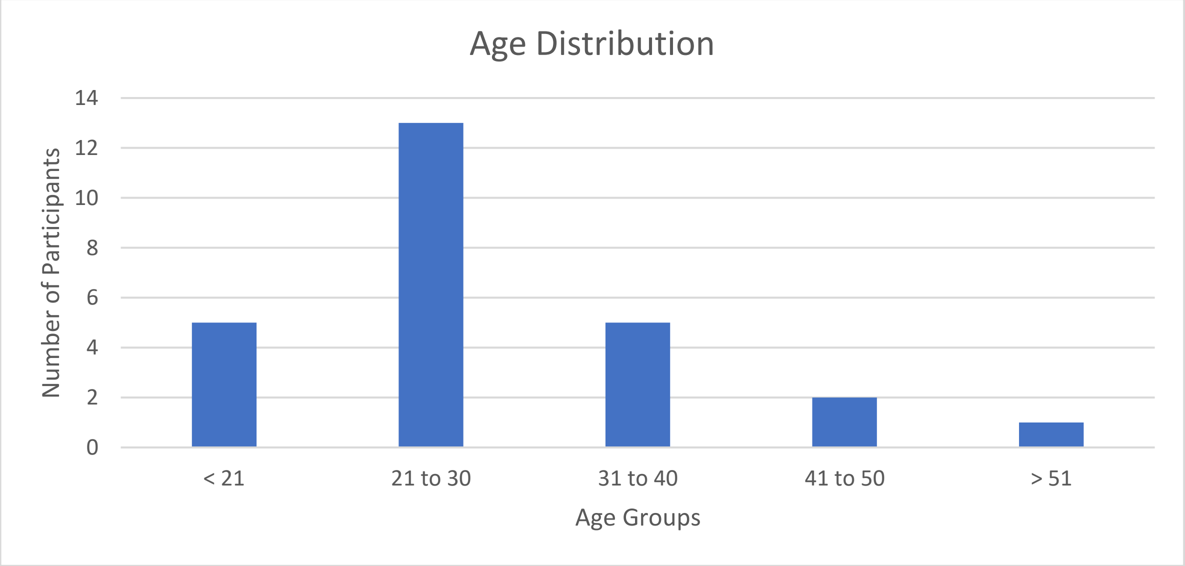


Figure 6.60: Age Distribution

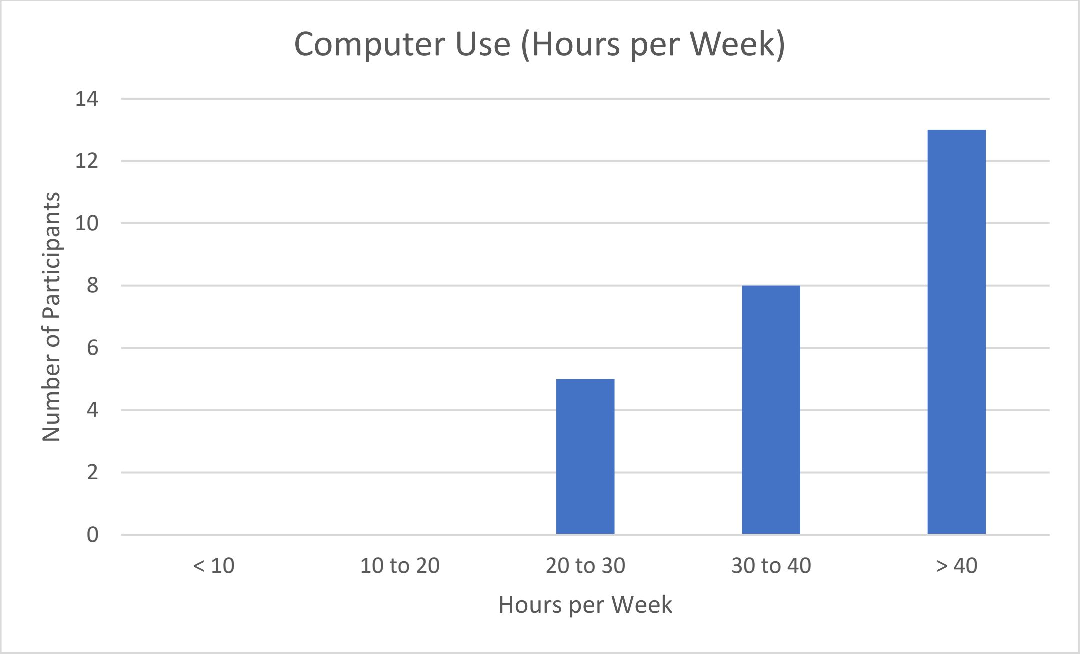


Figure 6.61: Computer Use (Hours per Week)

Figure 6.61 shows the distribution of reported time spent using a computer by participants. Most participants indicated a high amount of computer use, which again seems logical given most participants' predominant technology career focus. Figure 6.62 shows how much time participants reported playing games. As with the other experiments, it may have been more useful to contrast semester time activity against semester break activity. Most participants indicated a low amount of time playing games per week.

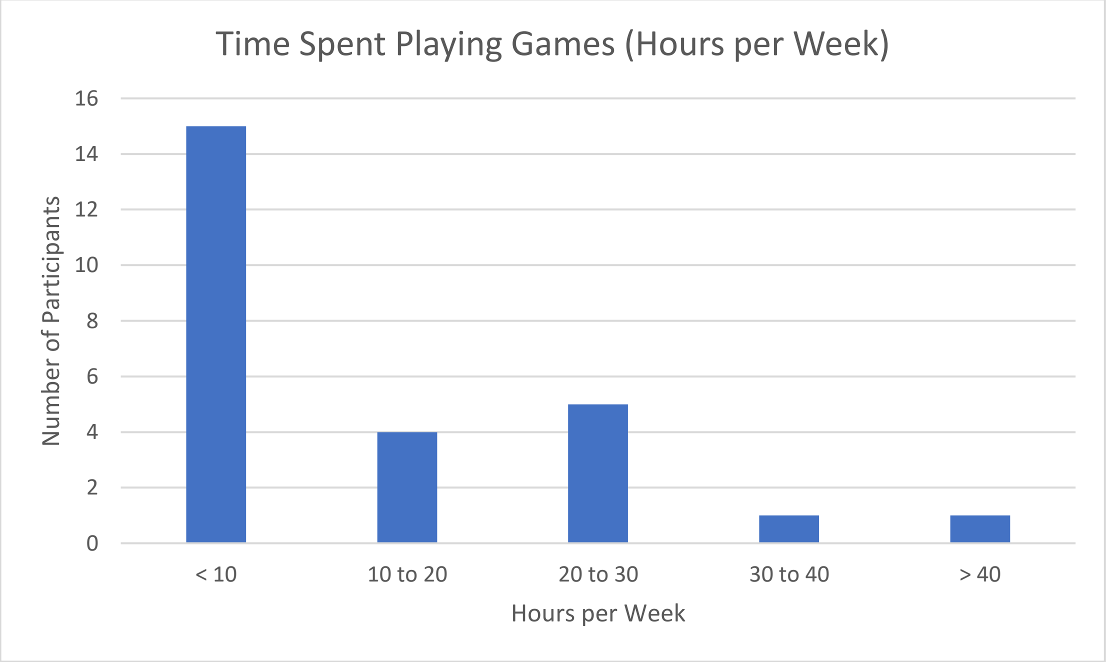


Figure 6.62: Time Spent Playing Games (Hours per Week)

HMDs

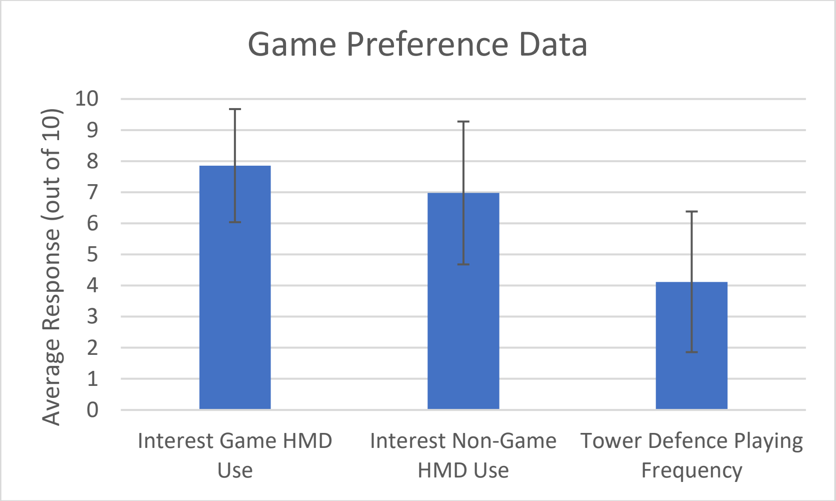


Figure 6.63: Game Preference Data

Figure 6.63 shows participants general interests toward types of HMD use and tower defence games. Participants, on average, were more interested in using HMDs for playing games. However, the interest in non-game activities was comparable. The reported frequency of playing tower defence type games was similar to the results for experiment two, with a high enough frequency to indicate occasional play of tower defence type games. Comparing the interest in HMD game use with a mean of 7.86 (out of 10) and non-game use with a mean of 6.98 was statistically significant t(27)=2.09, p = 0.02, indicating the preference for use with games was significant.

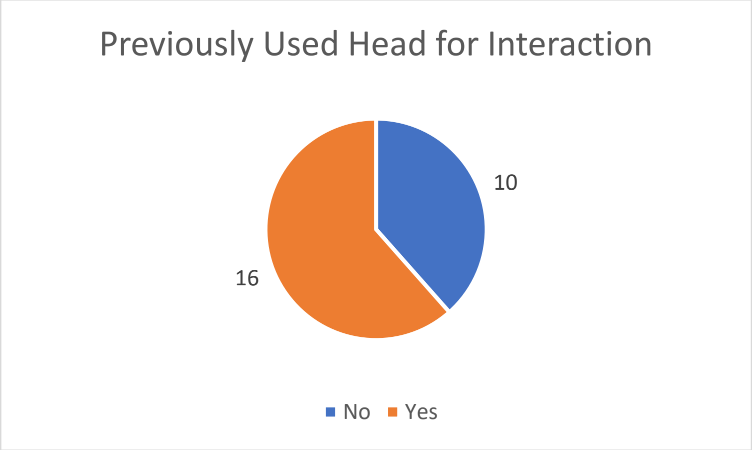


Figure 6.64: Previously Used Head for Interaction

Participants were asked if they had previously used their head as an interaction control technique, with some examples cited. Figure 6.64 shows 16 of 26 participants had previously used their heads in this way. Some of the examples indicated by participants included: this study (as part of a previous experiment), the Microsoft Kinect, other HMDs, and the GearVR.

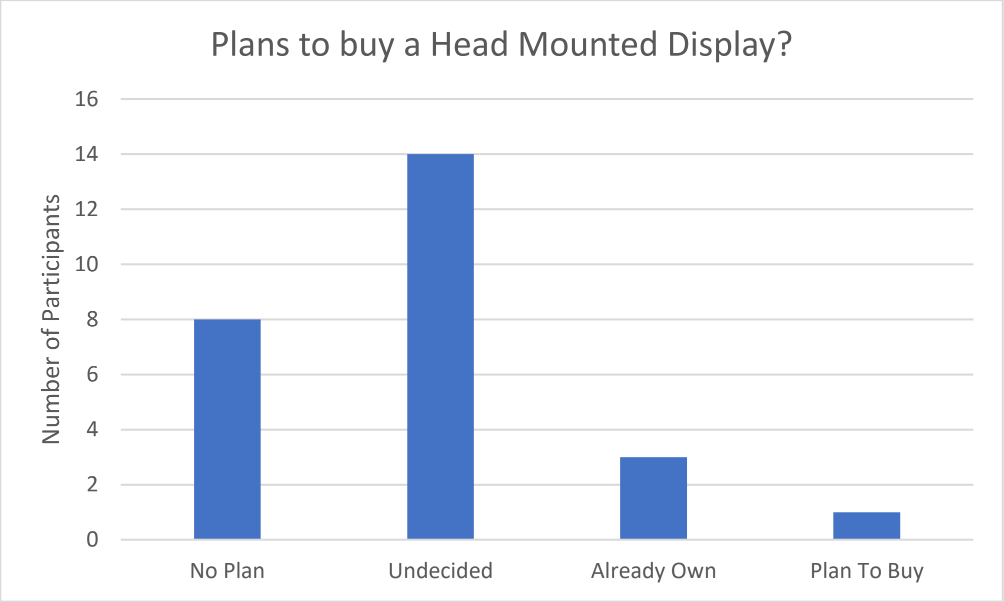


Figure 6.65: Plan to buy Head Mounted Display?

More directly concerning HMDs, participants indicated they were mostly undecided about purchasing a HMD, as seen in Figure 6.65. Three participants already owned a HMD and only one indicated a plan to buy. Comments about their responses indicated a desire to wait for a cheaper cost or an application that would provide features significant enough to draw them in.

Game Trailer Responses

Participants were asked about the influence of the game trailer on their decision to participate. These questions needed a control question to check if participants had viewed the trailer as some had not, and this did change the data as it reduced influence. Participants responded with an average of 5 (SD = 2.4, ranking on a Likert scale out of 10) when asked about the trailer's influence on their desire to participate. And an average of 6.65 (SD = 2.32) when asked about the expected trailer influence on future participation. The averages saw roughly a 10% drop in perceived average influence compared to the second experiment's responses with a larger variance on the standard deviation for future participation. The difference in comparing the data with a one-tail t-Test showed it to be statistically significant t(25)=-4.71, p = <0.01.

General Feedback Metrics

Figure 6.66 shows some general metrics related to the PVMS. Responses indicate participants found the menu appeared almost always at the correct times and somewhat at the incorrect times. Participants found the gesture interaction somewhat useful, and there was some positive interest in future use. A more detailed comparison will be conducted in the discussion chapter (Chapter 7), but comparatively, the correct times were reported to be higher in this experiment. The "Wrong Times" were reported about the same with a lower variation. The "Gesture Useful" and "Use in Future" responses were slightly lower than the second experiment.

Each feedback metric was compared with a t-Test to evaluate the significance of comparing them. Wrong Times with a mean of 5.44 compared to Correct Times with a mean of 8.85 was statistically significant t(25)=-9.99, p = <0.01. Wrong Times compared to Use in Future with a mean of 6.7 was statistically significant t(25)=-2.1, p = 0.02. Correct Times compared to Gesture Useful with a mean of 5.99 was statistically significant t(25)=7.39, p = <0.01. Correct Times compared to Use in Future was statistically significant t(25)=4.97, p = <0.01. Gesture Useful compared to Use in Future was statistically significant t(25)=-3.01, p = <0.01.

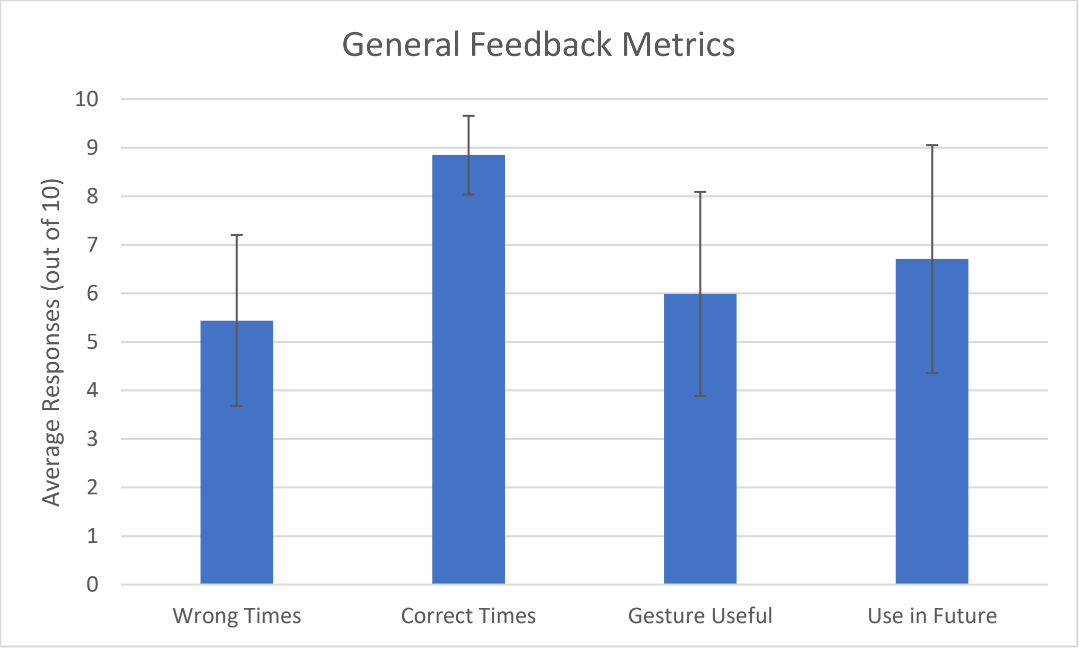


Figure 6.66: General Feedback Metrics

Participants were asked for elements they found enjoyable and difficult about using the interactions. Participants reiterated similar statements from the second experiment. Indicating they found the interaction was quick, responsive, intuitive, novel, easy to use. The general difficulties people responded with were related to some fatigue or when they accidentally triggered the menu when they had not intended to reveal it.

System Usability Scale

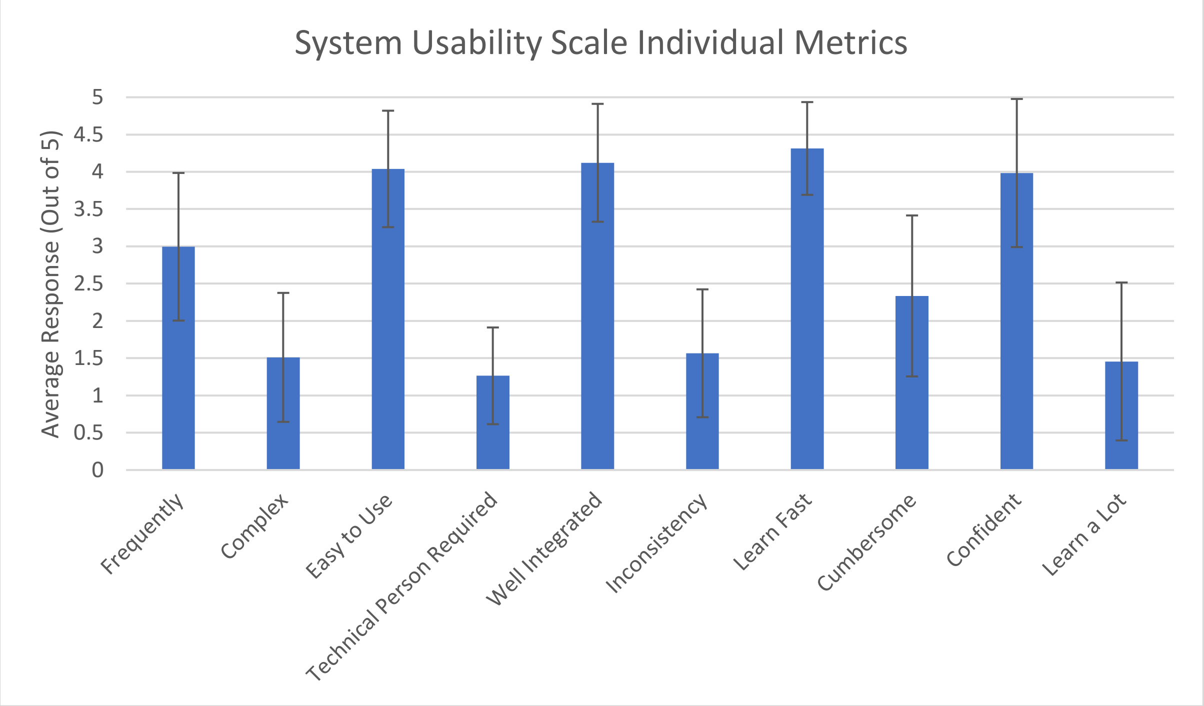


Figure 6.67: System Usability Scale Individual Metrics

As part of the collected data were questions related to the System Usability Scale (SUS). Five of the questions assess positive aspects of a system, and five assess the negative aspects. They are combined to calculate a score out of 40 that can then be converted to a score out of 100. Figure 6.67 shows the results from the questions. They are ordered by how they appeared in the questionnaire.

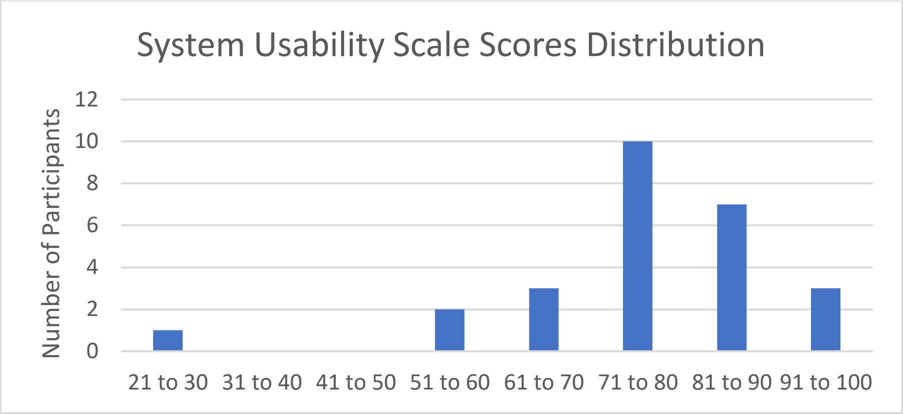


Figure 6.68: System Usability Scale Scores Distribution

The overall score result from the SUS was 78.3 (SD = 15.8). This score is a reasonable score. However, it is mostly offset by one very low score, as seen in Figure 6.68. 20 of the scores were higher than 70. For metrics related to the SUS score evaluation, this would be a B+ grade, fall in the good range, acceptable, and just in the passive range for the Net Promoter Score (NPS). Data for the SUS is compared further in 6.3.3 against responses to other questions.

System Use Preferences

Participants were asked to rank places where they would wish to use the PVMS. Lower numbers indicate a higher preference. Additional options were included with this experiment compared to the previous experiment with improved explanations for each item. Participants indicated on average in Figure 6.69 they would prefer to use the menus as a utility while viewing movies or as part of a tour guide (as described in the technology overview chapter's use cases in Chapter 4). Internet browsing was the least preferred on average.

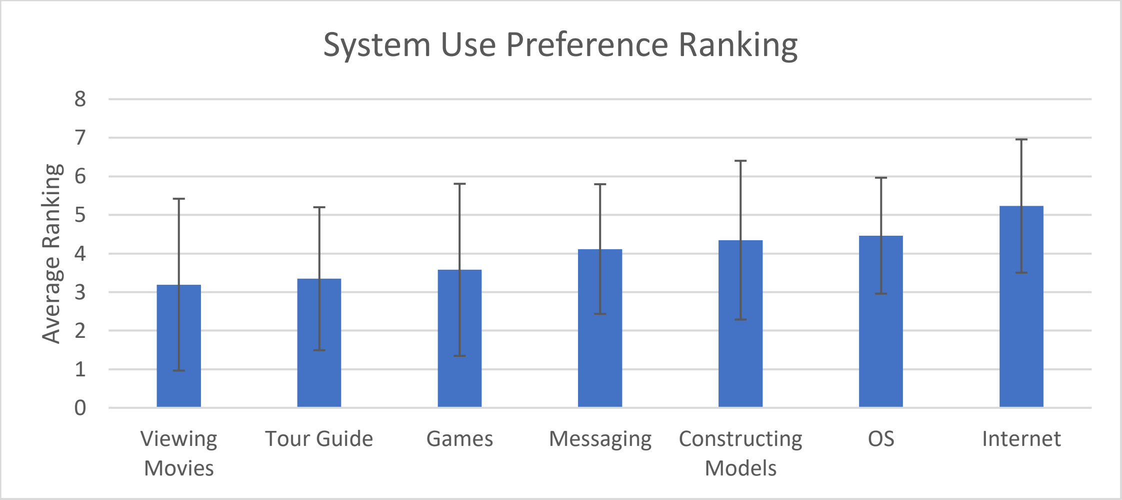


Figure 6.69: System Use Preference Ranking

Interface Usability

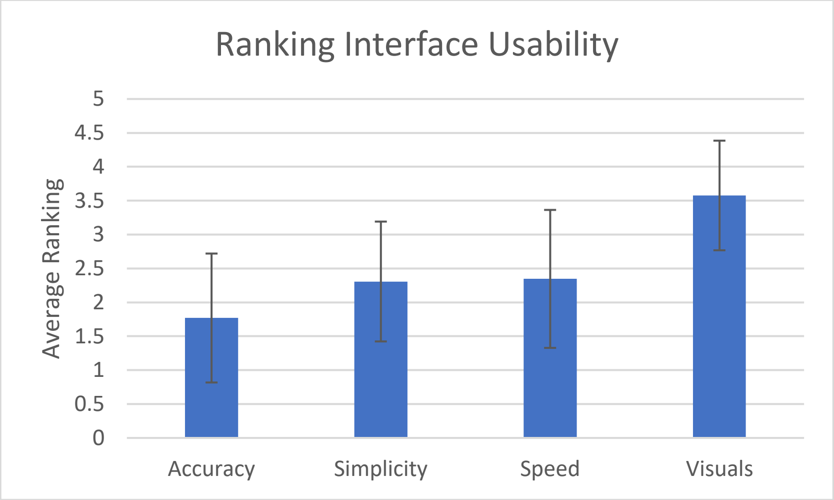


Figure 6.70: Ranking Interface Usability

Participants provided a similar response when asked to rank interface usability for HMDs compared to the second experiment. Lower numbers indicate higher preference. Figure 6.70 shows the accuracy of accessing features was considered the most important factor of interface usability. Speed and simplicity of interactions were rated around the same. Overwhelmingly visuals were considered the least important factor for usability.

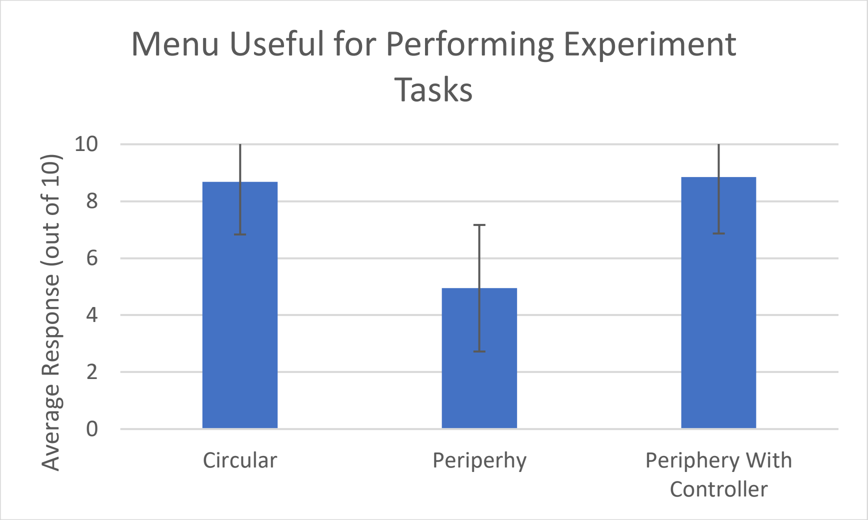


Figure 6.71: Menu Useful for Performing Experiment Tasks

Participants were asked to rate their experience as a comparison between the different types of interactions experienced during the experiment, as shown in Figure 6.71. Participants indicated preference toward the circular menu where it was used in comparison to the PVMS Only. And a preference toward using the additional controller input with the PVMS as compared to HMD Only interaction. In a larger application, a PVMS could be used alongside multiple types of menus where each menu is suitable for the type of interactions required. The PVMS does not need to be used for every scenario if a more appropriate solution exists. Still, it provides a foundation for alternate techniques in the area of menu interaction. Comparing the results for statistical significance found the circular against periphery with a controller to not be statistically significant. Circular with a mean of 8.68 against periphery with a mean of 4.95 was statistically significant t(25)=5.52, p = <0.01. Similarly, periphery with a controller with a mean of 8.85 was statistically significant compared to the periphery without a controller t(25)=-5.89, p = <0.01.

The circular menu solution functioned as the modification tool for objects already created in the world alongside the periphery menu simultaneously used for object creation. The statistical significance indicates participants felt strongly about using appropriate menus for the situation. For HMD Only input, this suggests a mix of menu types is appropriate where menus are created at the point of interest when directly interacting with an element. The PVMS menu should be used for creation and other interactions unrelated to a direct interaction point. Although the comparison between circular menu use and the PVMS with a controller was not statistically significant, the similarity between the data from observation indicates adding a tap-to-select interaction makes the PVMS more competitive as a utility.

6.3.2 Application Data Results

This section will cover application data results that have been derived from log data collected during the experiment. The tables of data that were generated have been summarised in section 6.2.9. The results have been grouped based on the types of information.

PVMS Use

All the experiments' tasks provided the same contextual menus for left and right interactions with the PVMS. Figure 6.72 (over) shows the breakdown grouped by task for participant choices between left and right menu operation. Overall, the numbers are very similar across all tasks. During the main menu and object matching tasks, the participants slightly preferred using the left side menu. The two variations of tower defence tasks had participants choosing to use menus on the right side more often. One reason for this may have been partially due to enemy waves coming from the right side of the default camera position. Meaning a menu on the right would still typically provide a view of the enemies as they entered.

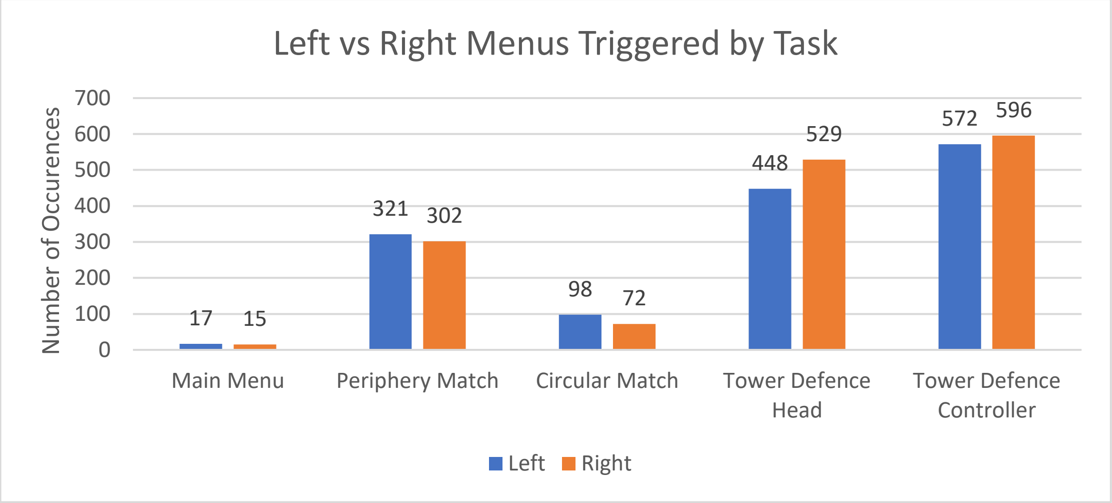


Figure 6.72: Left vs Right Menus Triggered by Task

Figure 6.73 shows the X and Z angle difference data used to determine if the PVMS should display a menu. `Diff_X` refers to the up and down angle (Pitch), and `Diff_Z` is the rotation from tilting the camera (Roll). `Diff_X` saw little difference on average between the tasks. The values for `Diff_Z` on average show distinctly similar values dependent on the task being completed. The similar values is likely due to the camera angle typically used. The more angled the camera's view for viewing current activities more likely it leads to higher variance on the Z-axis while turning around (Yaw rotation). The average history cache size during this experiment was around 22.76 (SD = 1.48) elements.

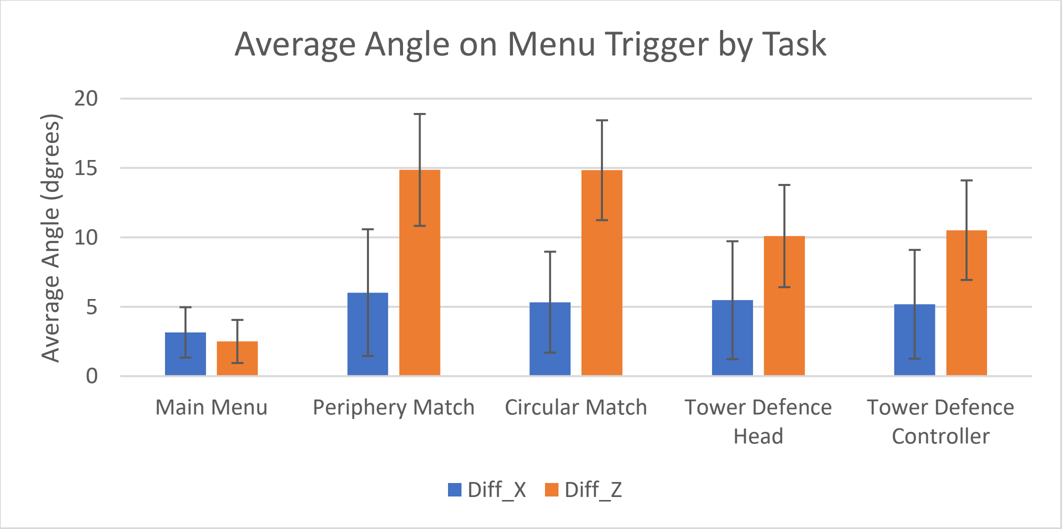


Figure 6.73: Average Angle on Menu Trigger by Task

Menu Interactions

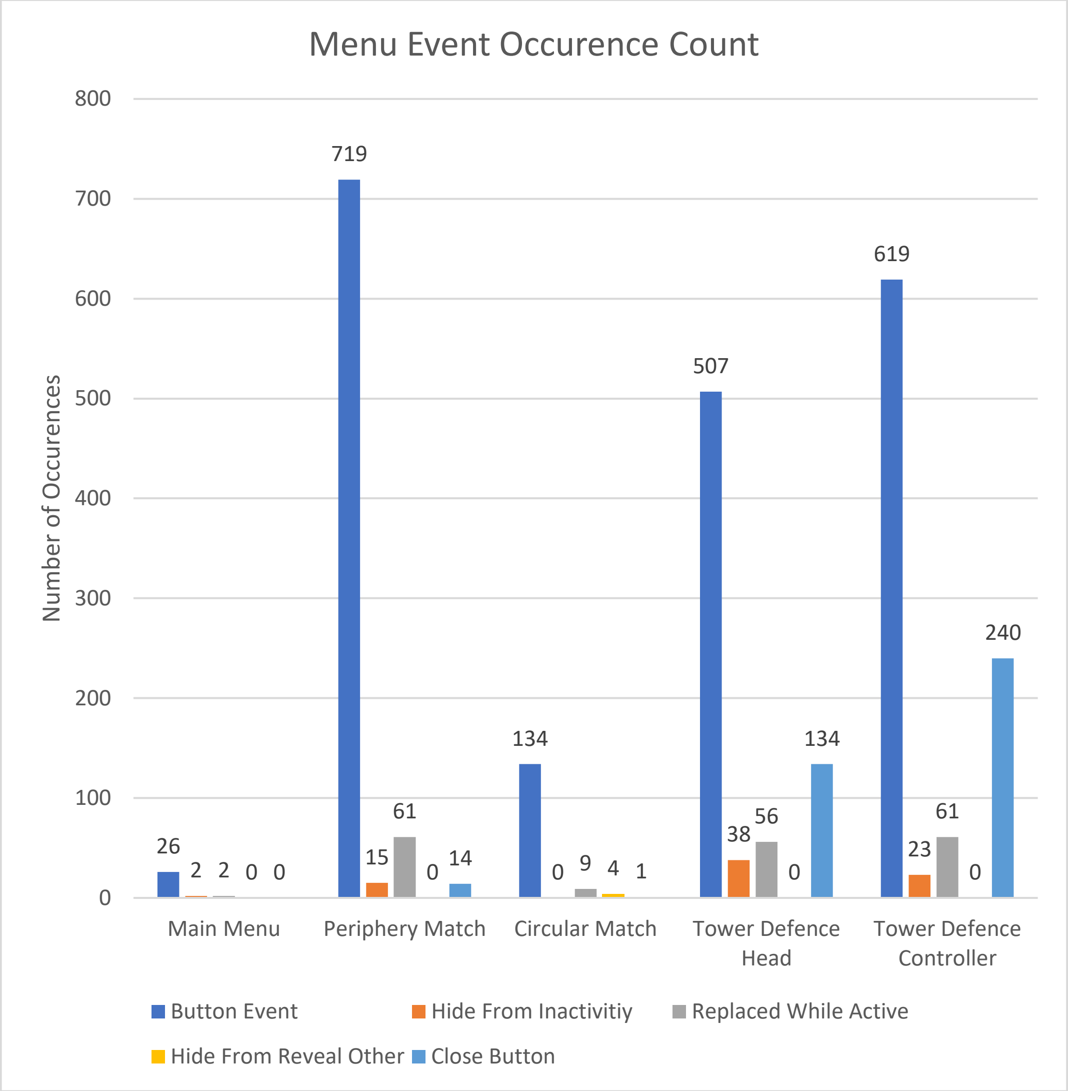


Figure 6.74: Menu Event Occurrence Count

Figure 6.74 shows the number of occurrences for the different types of events resulting in hiding a PVMS. A "Button Event" indicates an action was taken within the menu resulting in the menu being successfully used. The total for button events does include the use of the close button. The close button has been recognised additionally as a separate element to show the use of the new feature in this experiment. Most of the close button use was during the tower defence games showing more use when a controller was used. A "Hide from Inactivity" occurred if the participant opened the menu but did not interact for a period of 4+ seconds. Replaced While Active indicated the menu had been replaced with a different two-step widget for a new menu. Hide from Reveal Other was related to a rare case for checking overlapping interactions between the Circular and Periphery Menus.

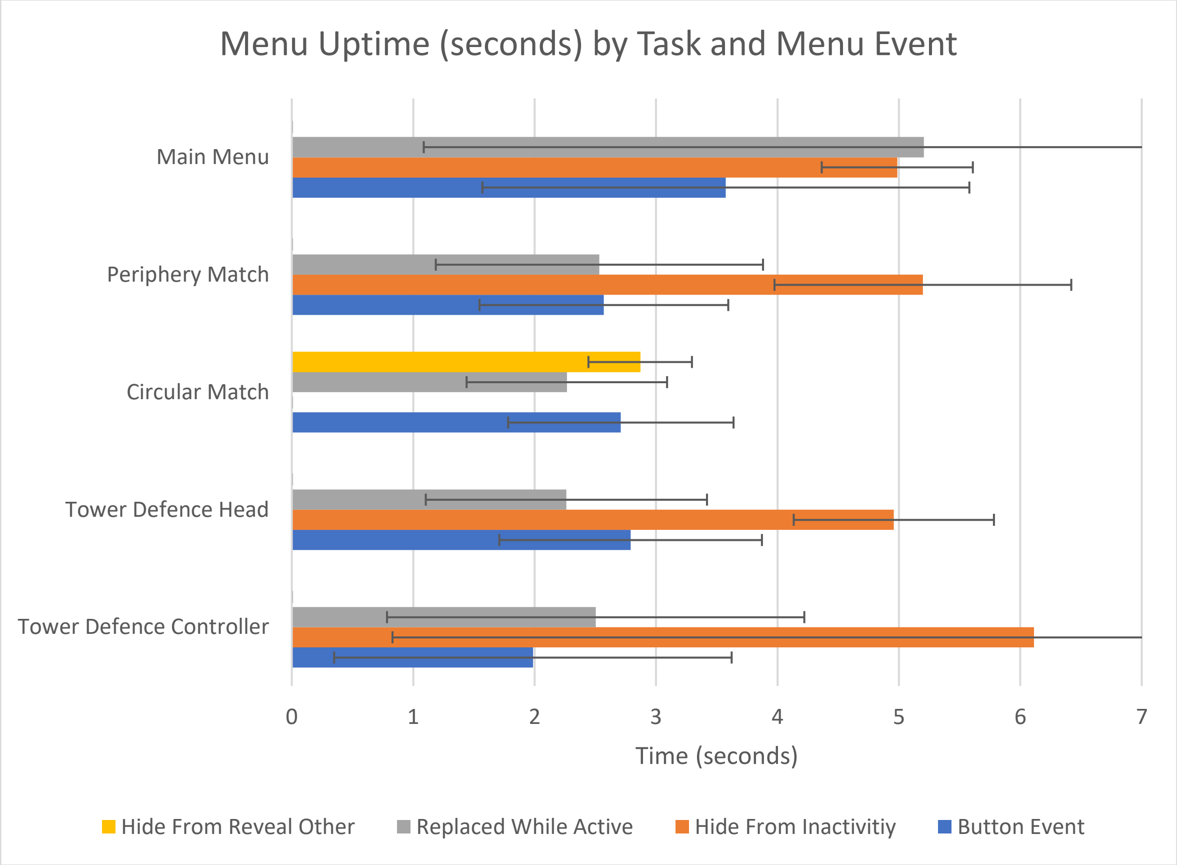


Figure 6.75: Menu Uptime (seconds) by Task and Menu Event

Figure 6.75 shows the average time in seconds that menus were open during each task related to the type of interaction used to close the menu. A successful menu interaction would take at least 1.5 seconds to perform a hover selection unless the interaction was with the close button, where it would take 0.75 seconds. Participants experiencing the "Hide from Inactivity" type event on average interacted with the interface for 1 to 2 seconds before leaving them untouched for 4+ seconds to hide automatically. Interaction time with a controller for instant selection did not appear to significantly speed up button event type interaction speed compared to the HMD Only type speed.

Other Menu Interaction Data

Figure 6.76 (over) shows the number of occurrences for each type of menu event for the two-step widget. The comparison of occurrences illustrates that the "Show Menu" button was almost always interacted with to show the PVMS in full. Most of the hide from inactivity events occurred during the tower defence game when participants were panning around the scene more than other tasks from the experiment and either did not intend to make the menu appear or changed their mind before interacting with the button.

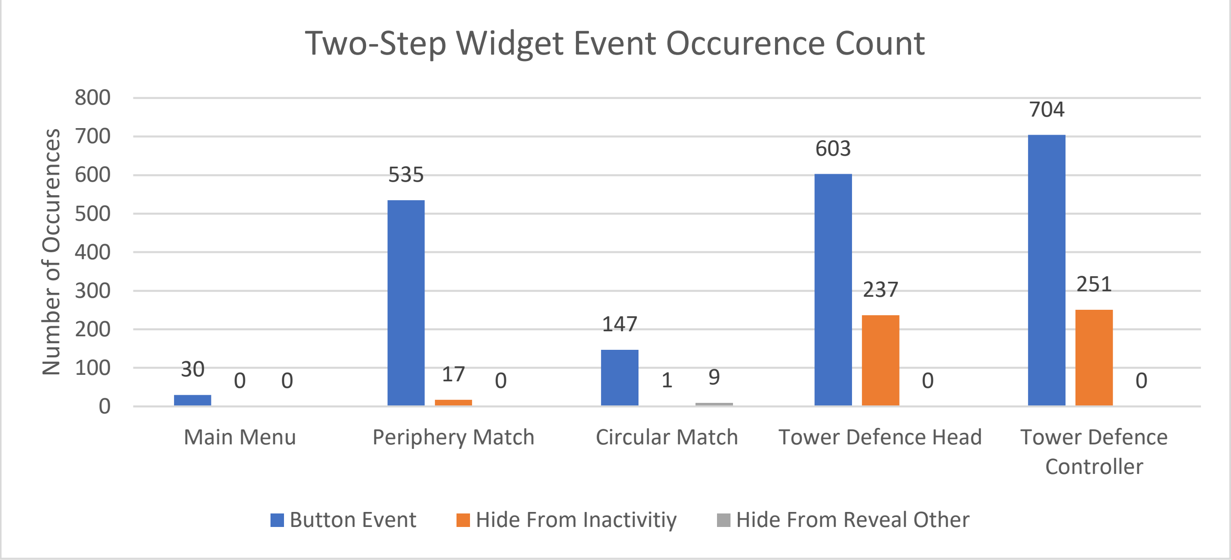


Figure 6.76: Two-Step Widget Event Occurrence Count

Figure 6.77 shows the menu uptimes for the two-step widget. The uptime shows most interactions occurred in a short period, with the faster durations less than 0.1 seconds. Related is the circular menu's average button event uptime of 2.8 (SD = 1.27) seconds. The "Hide From Inactivity" was constant at 8 seconds because that would only occur due to no other action taken for the 8 seconds.

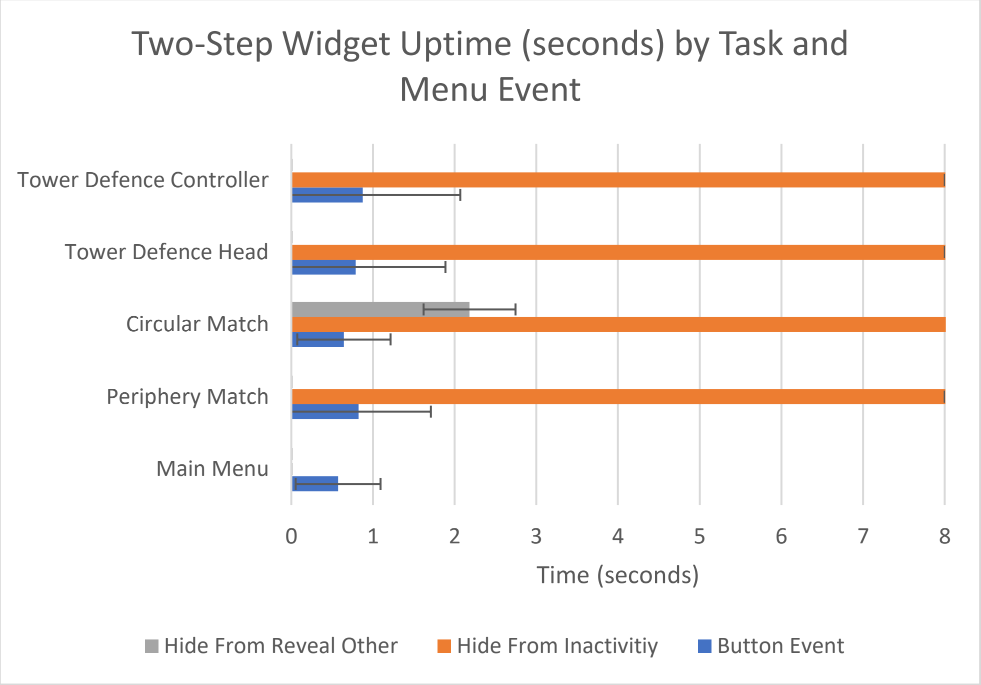


Figure 6.77: Two-Step Widget Uptime (seconds) by Task and Menu Event

Task Completion Time

Figure 6.78 shows the average time spent on each task, including time spent on the tutorials appearing before each task. The average task completion time for the object matching tasks shows the PVMS Only type interactions taking, on average, a longer period. The participant did have a learning effect advantage the second time as they knew how the task was to be completed correctly. The learning effect will be shown with lower total errors from task-specific data later in this section. The time spent on playing the tower defence game had minimal variance between HMD Only interaction and the addition of the controller interaction.

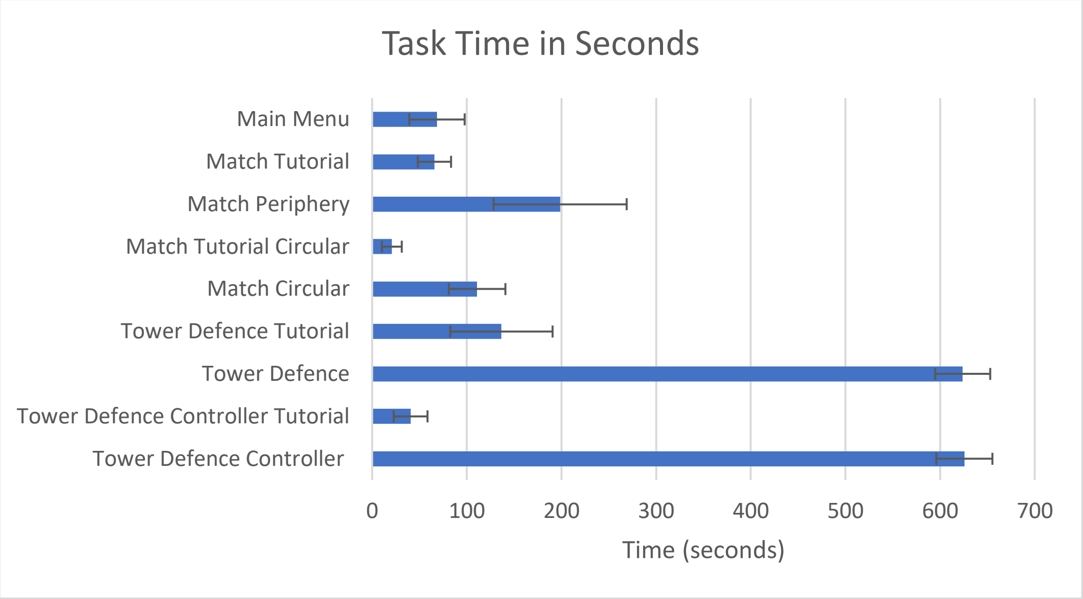


Figure 6.78: Task Time in Seconds

In Application Responses

Figure 6.79 shows participants' responses after each experiment task when asked about the effectiveness of the menus used in that task. Participants indicated a preference toward the circular type of menu for the object matching tasks. And a preference toward using the controller for the tower defence task. Overall, the scores indicate participants felt the menus were in the medium to high range of effectiveness. Match Periphery compared to Match Circular was statistically significant t(25)=-7.3, p = <0.01. Match Periphery compared to the HMD Only Tower Defence was not statistically significant. Match Periphery compared to the Controller version of the Tower Defence was statistically significant t(25)=-3.74, p = <0.01. Comparing the Match Circular to the HMD Only Tower Defence was significant t(25)=4.61, p

= <0.01, but comparing against the Controller version was not. Finally, comparing the HMD Only and Controller Tower Defence responses were statistically significant t(25)=-3.53, p = <0.01.

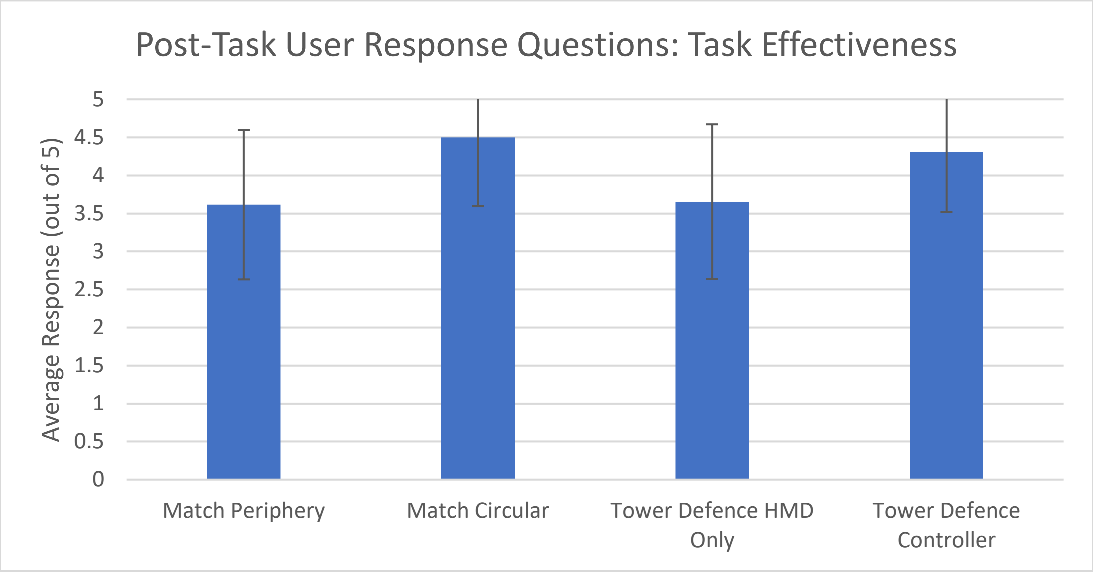


Figure 6.79: Post-Task User Response Questions: Task Effectiveness

Periphery Menu Object Matching Data

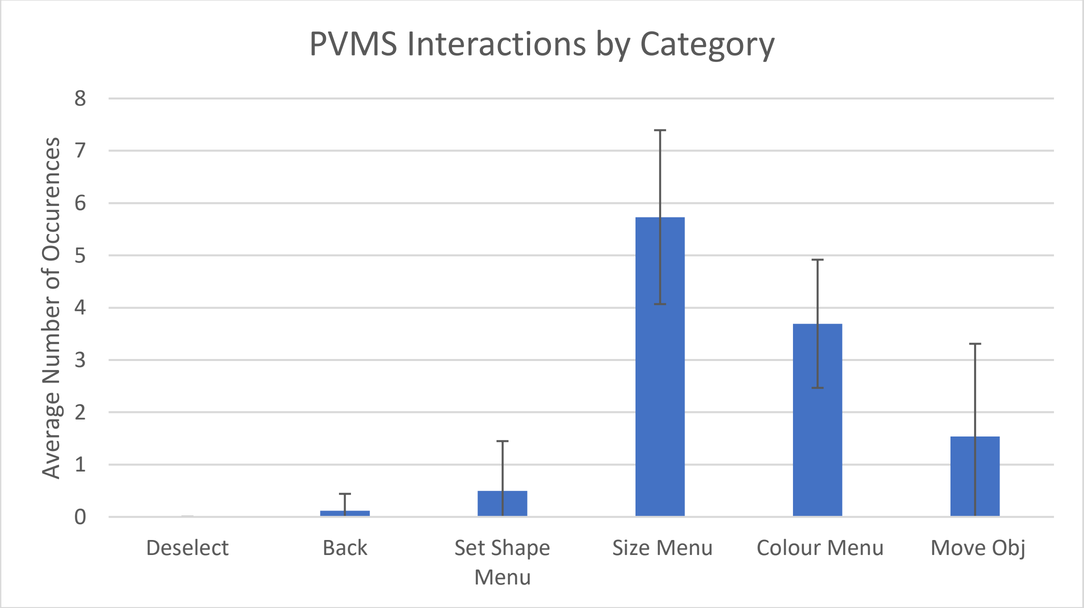


Figure 6.80: PVMS Interactions by Category

Figure 6.80 shows the average number of interactions applied using the menu during the PVMS focused version of the object matching task. No participant used the deselect feature in the menu. The back feature and set shape features were used very little. Not using these features was good as it shows participants were not incorrectly selecting an option very often. When participants completed puzzles correctly, it was never necessary to use any of the deselect, back or set shape menus.

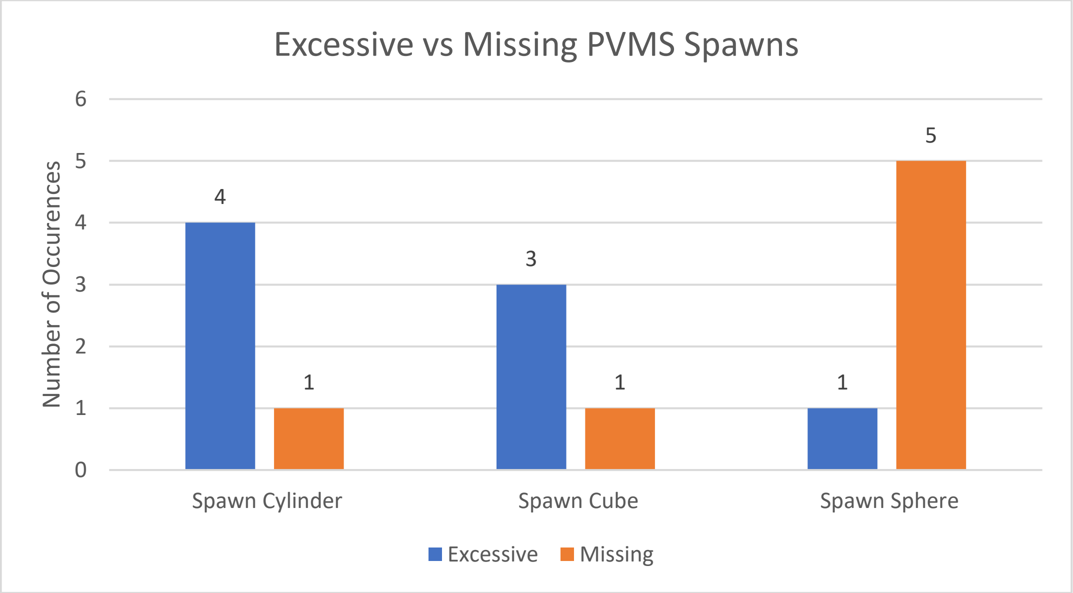


Figure 6.81: Excessive vs Missing PVMS Spawns

The object matching puzzles were designed to have a minimum number of operations required to complete. The first step of completing any single object match was to spawn in a needed shape. Figure 6.81 shows participants spawned in unneeded shapes (Excessive) and then had to use the set shape menu to change them back into the needed shape (Missing).

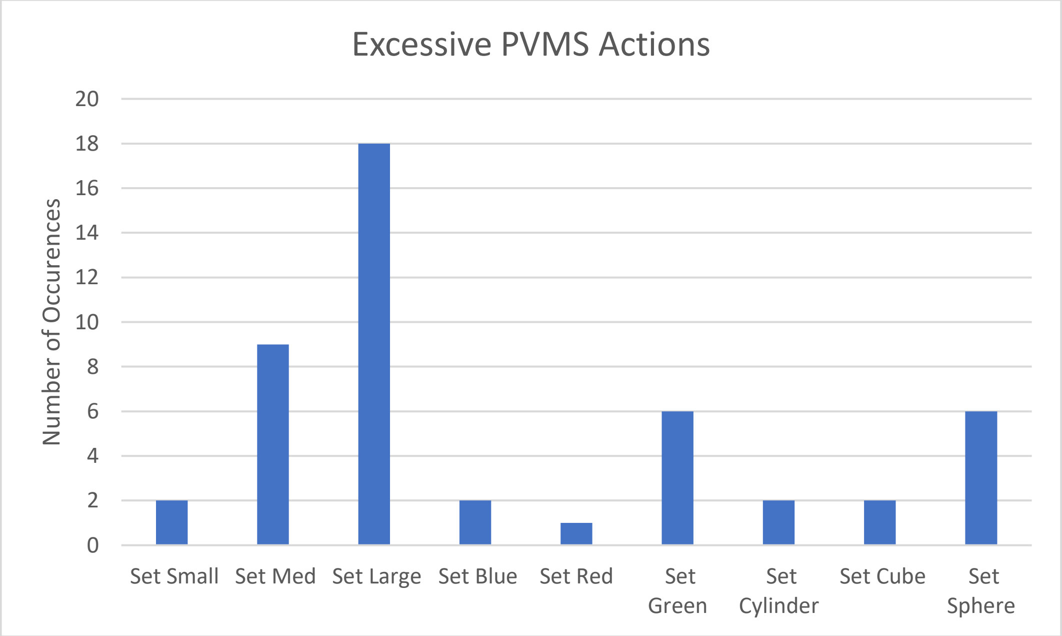


Figure 6.82: Excessive PVMS Actions

Figure 6.82 (previous page) shows the other types of additional actions participants had to take when correcting mistakes they had made while matching objects. The most common incorrect actions were setting objects to large or medium sizes. Likely when the shape should have been the opposite size and participants were unsure of the correct size.

Circular Menu Object Matching Data

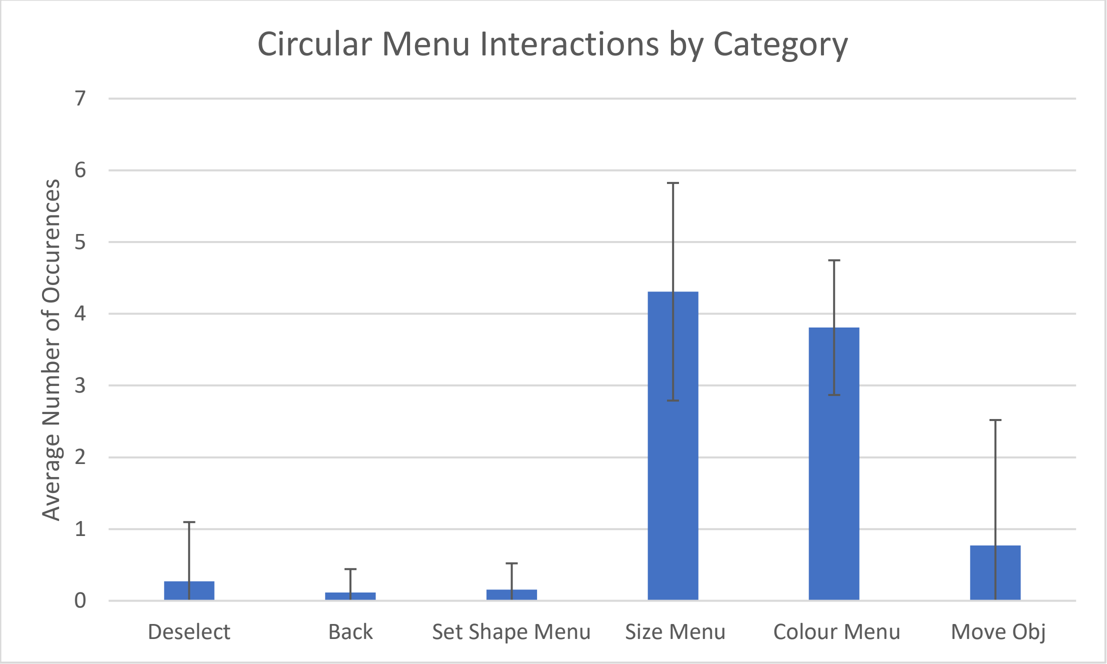


Figure 6.83: Circular Menu Interactions by Category

Figure 6.83 shows there was more use of the deselect and back functionality with the circular menus when compared to interactions with the PVMS. There were fewer errors to correct with the set shape menu. It also appears the randomisation of puzzles required more colour menu interaction and slightly less size menu interaction. Participants also used the move object function less, meaning they placed the newly created objects in the correct location where they intended more often. As discussed previously, this could result from the learning factor between the two separate tasks. The participant became familiar with creating and modifying objects in Task 1 and used that knowledge to place the objects first go in Task 2 correctly.

Figure 6.84 (over) shows that there was only a single error made with choosing the wrong shape across all participants. The participant who made the error had selected a cube but needed a sphere to complete their current objective. A single error is a significant improvement over the errors from Figure 6.81 in the previous task.

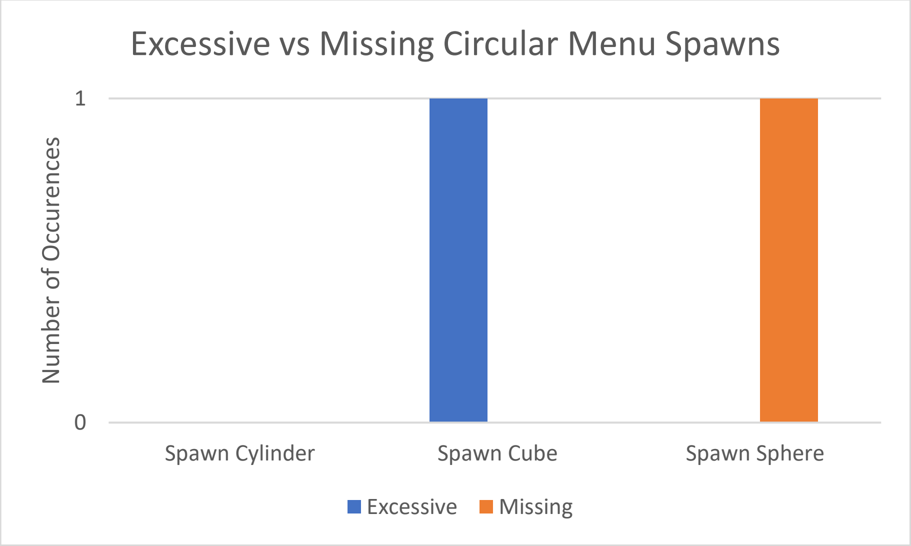


Figure 6.84: Excessive vs Missing Circular Menu Spawns

The other excessive actions were also less overall in Figure 6.85 than they were in Figure 6.82. There were still some issues with selecting the correct size and perhaps unintentionally changing shapes. The most common problem was setting elements to blue when they did not need to become blue.

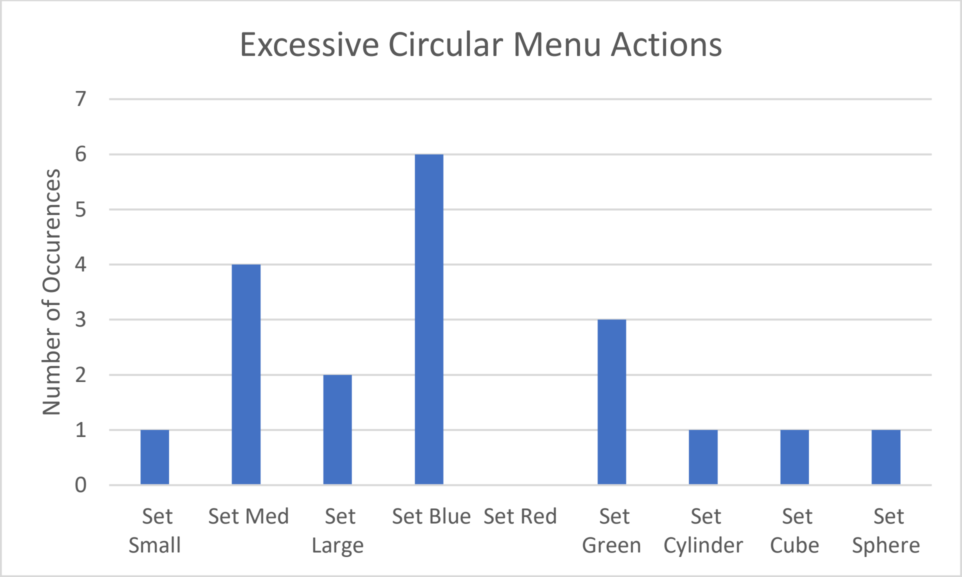


Figure 6.85: Excessive Circular Menu Actions

Tower Defence Task HMD Only Data

Figure 6.86 (over) shows the choices for currency spending on towers during the HMD Only version of the tower defence game. On average, participants built at least one basic tower and one frost tower during the initial construction. And on average, two basic towers during the first wave. With an additional one during the second wave. Purchases of other towers were all small across all participants for any single game state. The variations shown in the figure indicate participants were purchasing the different types of towers at varying times.

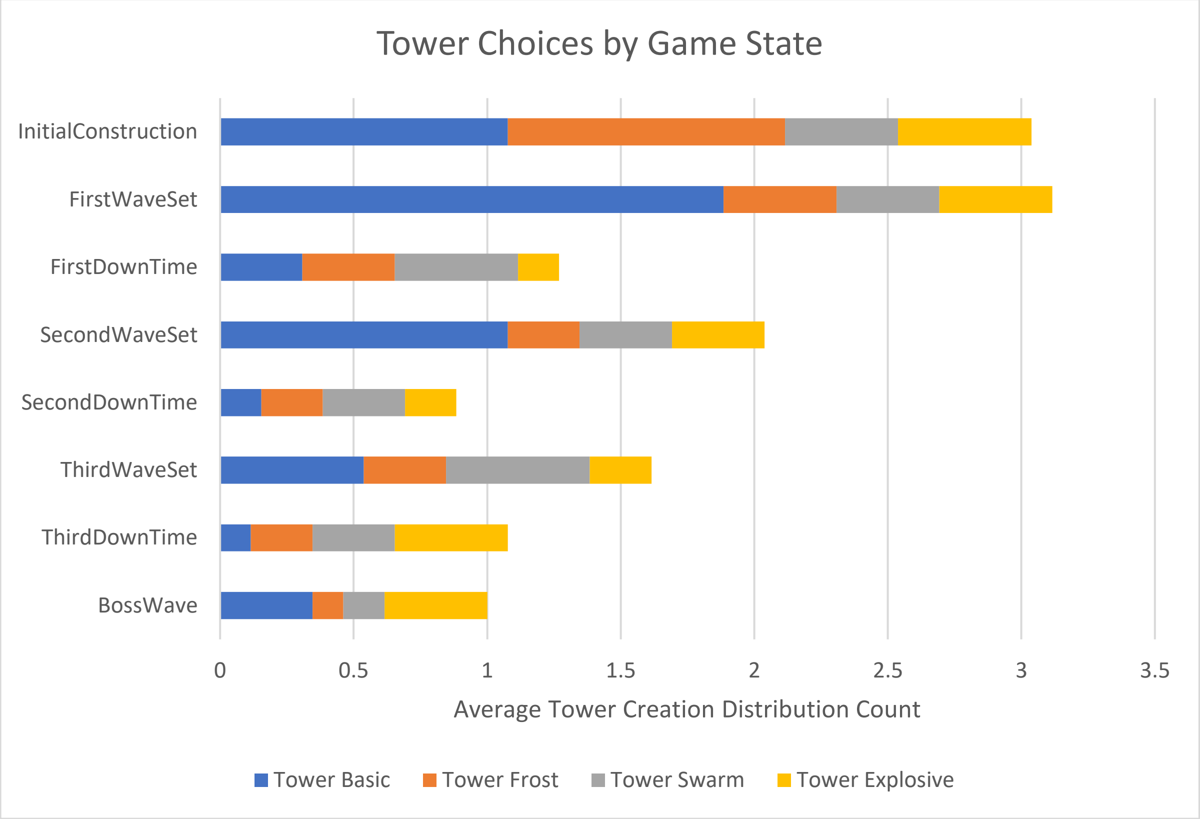


Figure 6.86: Tower Choices by Game State

Figure 6.87 (over) shows participants, on average, tried the camera snap nodes (Camera Snaps Used) during the initial downtime but then did not interact with this mechanic much during the waves to change their perspective. Deselection was increasingly used relative to the number of total waves that had passed, indicating a lot of tower selection. The increase of tower selections was likely to check the current health values of towers to initiate repairs. The value for failed repairs during the boss was excluded from the figure because the value of 26 dwarfs all other values. It was discovered post-experiment that there was a bug in the code causing repairs to be repeatedly requested while continuing to hover over the menu option resulting in the error constantly occurring when there was insufficient currency at the time to perform the action.

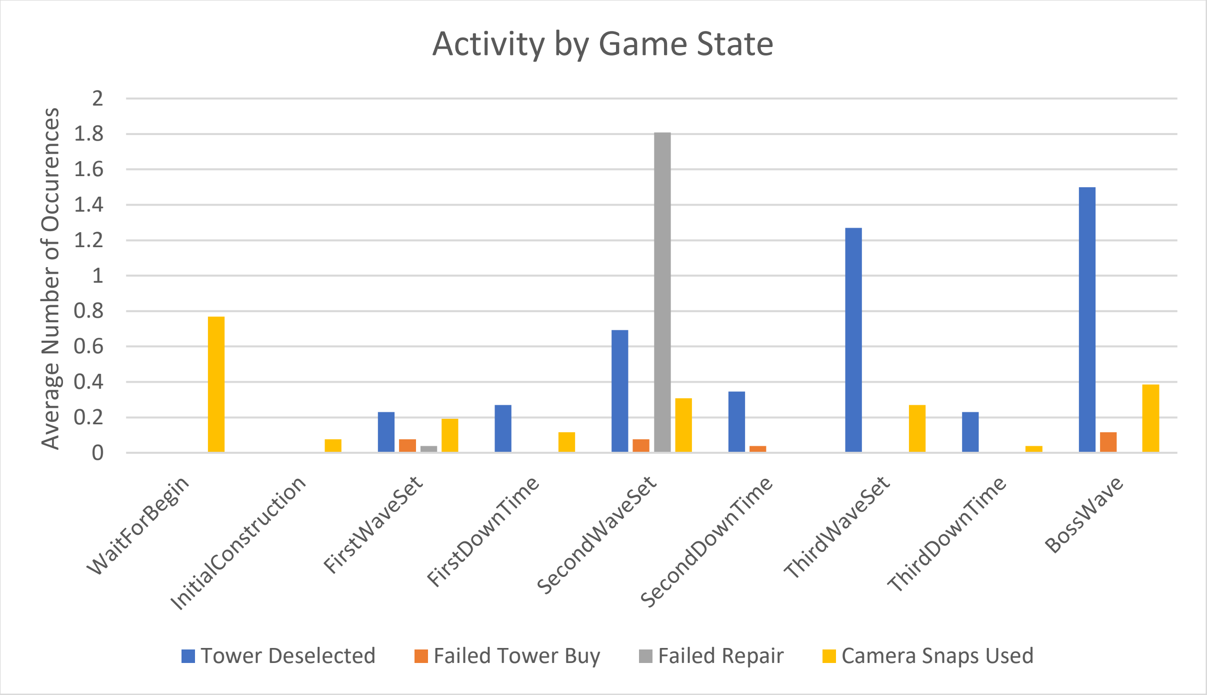


Figure 6.87: Activity by Game State

Figure 6.88 shows the base health and enemies killed for each wave. The player base defended during this experiment had 40 health up from the 25-health used in experiment two. Participants, on average, let through a larger number of enemies during the first wave and continued to let through a few enemies on subsequent waves gradually. Indicating on average, the difficulty in strategy continued past the first wave, unlike the second experiment.

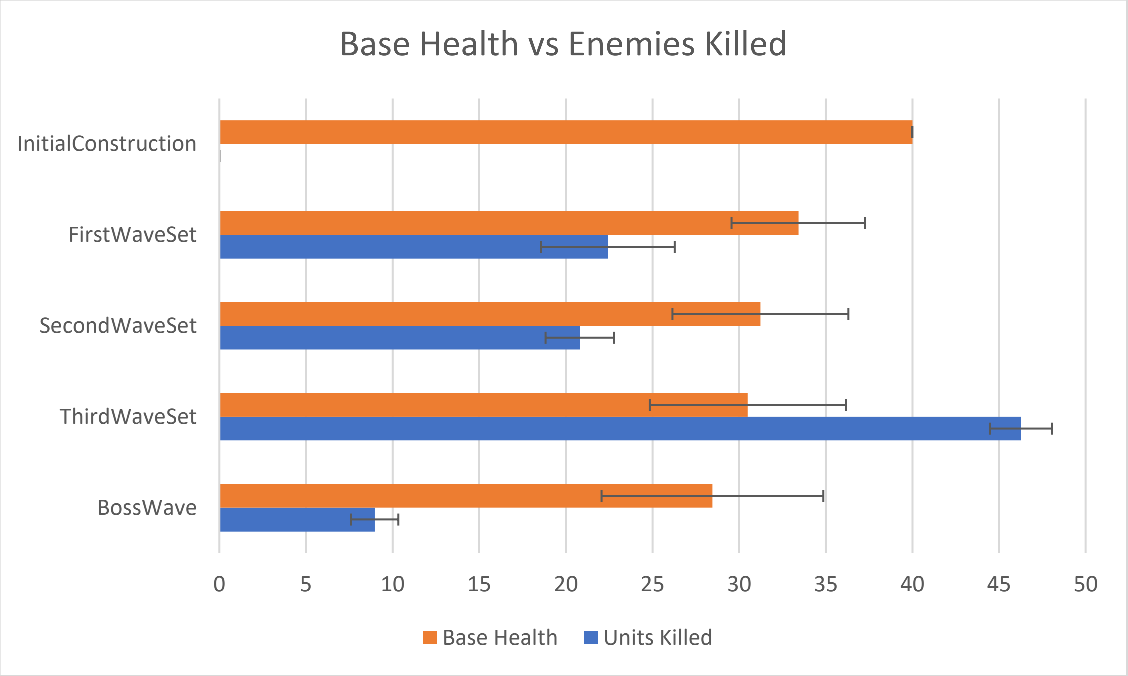


Figure 6.88: Base Health vs Enemies Killed.

Tower Defence Task with Controller Data

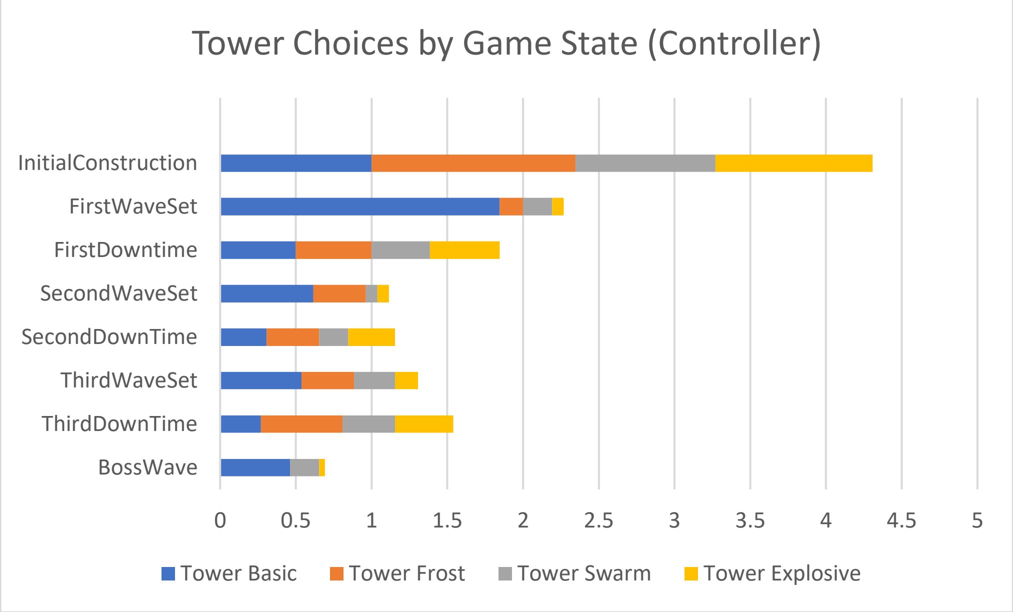


Figure 6.89: Tower Choices by Game State (Controller)

Figure 6.89 shows participants had far more idea about what they wanted to build at the start during the initial construction state. During the HMD Only type interaction, participants were often still deciding on towers after the first wave had begun to march. Participants shifted more currency toward explosive and swarm towers as part of their initial tower choices but then selected those options less during the subsequent states.

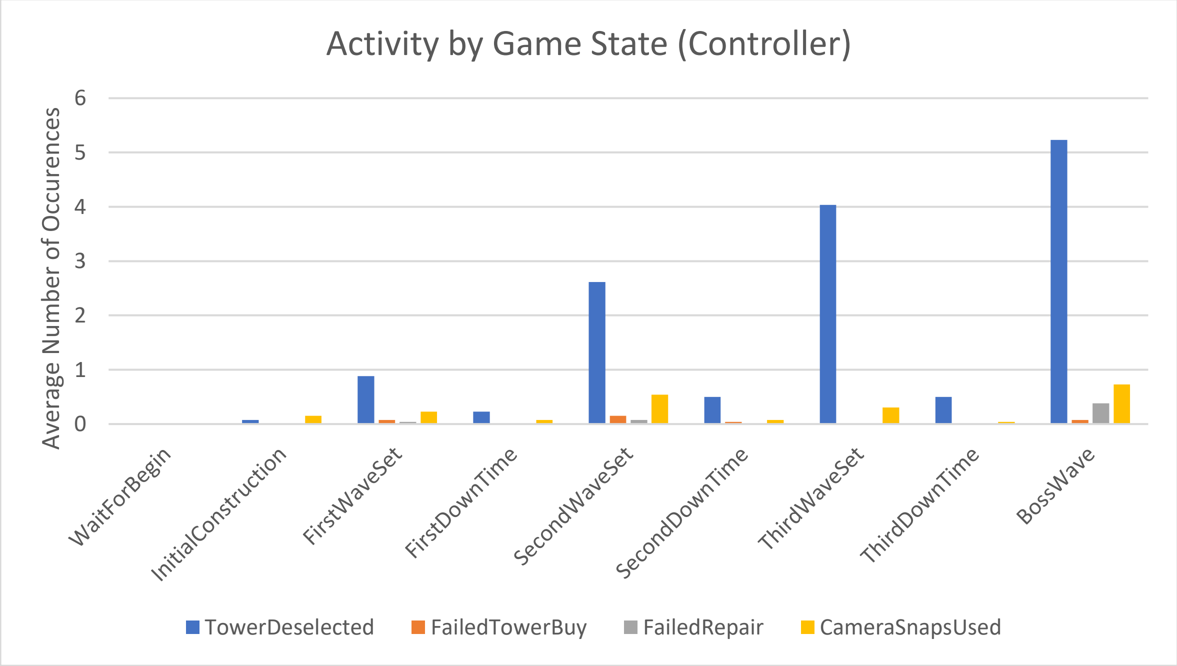


Figure 6.90: Activity by Game State (Controller)

Figure 6.90 (previous page) shows a far larger number of deselection type operations compared to the HMD Only interactions. The deselections show an increase relative to the number of waves and should be expected due to a larger number of towers available to interact with as the game progresses. Camera snap use was far less than the first time through the task, with the most use shifting to the final boss wave likely to spectate after nothing else could be done before the end of the game.

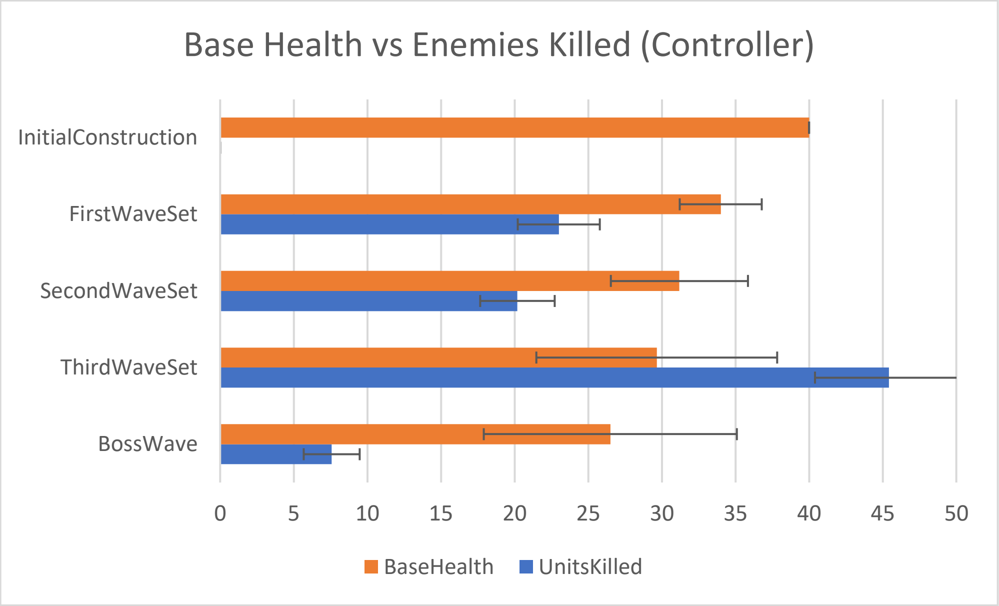


Figure 6.91: Base Health vs Enemies Killed (Controller)

The enemies had 10% more health during the second time through this task, so it was expected to be slightly more difficult. The difficulty is evident in Figure 6.91, with slightly lower average base health. This lower average health was due to a small number of additional enemies breaching the tower defence. The variation suggests that, on average, participants were making smarter choices in their strategies to handle the increased difficulty.

6.3.3 Result Analysis

Similar to the result analysis sections for the first two experiments, this section will present a comparison specific to the third experiment with a comparison between elements that made up the data collected as part of the experiment. For analysis comparing the different experiments, see Chapter 7.

SUS Scores against Age

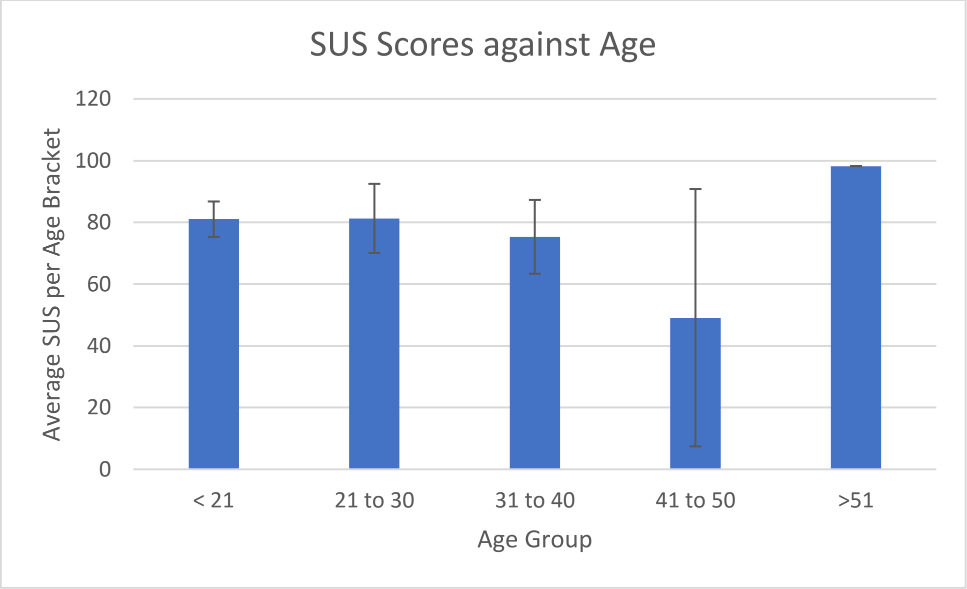


Figure 6.92: SUS Scores against Age

Figure 6.92 shows a comparison of average SUS scores for each age group. Participants with ages less than 21 had an average SUS score of 81.07 (SD = 5.73, n = 5), those in the 21 to 30 group had a mean of 81.32 (SD = 11.19, n = 13), the 31 to 40 group had a mean of 75.36 (SD = 11.94, n = 5), the 41 to 50 group had a mean of 49.11 (SD = 41.67, n = 2), and greater than 51 group had a single participant with a score of 98.21. Comparing the SUS scores between categories with an ANOVA test found significant differences (F(4,21) = 3.07, p = 0.04). The age groups can be observed from the figure to have a gradual decreasing mean from less than 21 to the 31 to 40 group where most participants were. The standard deviation increased gradually as the means decreased with the 41 to 50 group, demonstrating an extreme differential between two opposite scores.

SUS Scores against Computer Use

Figure 6.93 (over) shows the SUS scores categorised by the computer use of participants. The 20 to 30 hours per week category had an average SUS score of 77.86 (SD = 13.98, n = 5), the 30 to 40 group had a mean of 73.44 (SD = 22.54, n = 8), and the greater than 40 group had a mean of 81.46 (SD = 11.49, n = 13). There was no statistical significance found from comparing these categories.

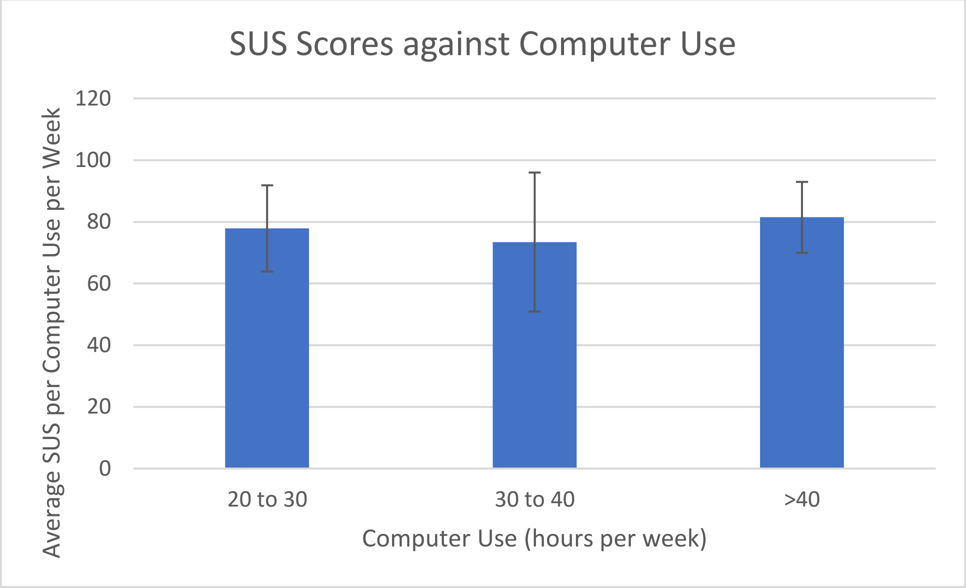


Figure 6.93: SUS Scores against Computer Use

SUS Scores against Game Use

Figure 6.94 compares the average SUS scores against game use by participants. Participants who said they played less than 10 hours a week of games had average SUS scores of 75.24 (SD = 19.14, n = 15), those playing 10 to 20 hours had a mean of 79.02 (SD = 12.23, n = 4), those playing 20 to 30 hours had a mean of 87.14 (SD = 4.79, n = 5), the single participant in the 30 to 40-hour category had a score of 73.21. The single participant in the greater than 40 category had a score of 82.14. No statistical significance was found when comparing SUS scores for these categories.

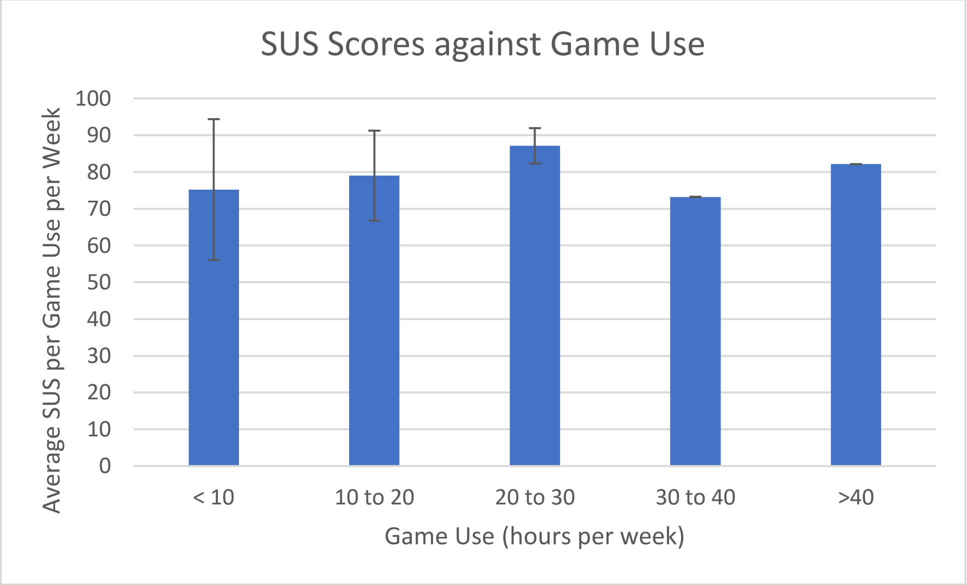


Figure 6.94: SUS Scores against Game Use

SUS Scores against Correct Times

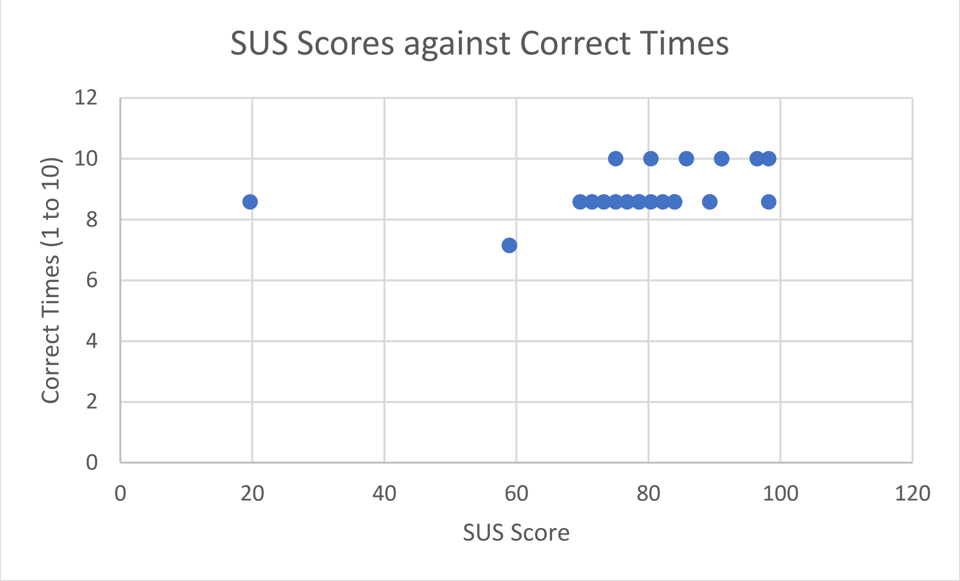


Figure 6.95: SUS Scores against Correct Times

Figure 6.95 compares SUS scores against the perceived accuracy in showing the PVMS correctly to the participant. The data was scaled to compare with a t-Test by multiplying the values by 10. The scaled Correct Times had a mean of 88.46, and SUS with a mean of 78.3 had a statistically significant difference t(25)=-3.77, p = <0.01.

SUS Scores against Wrong Times

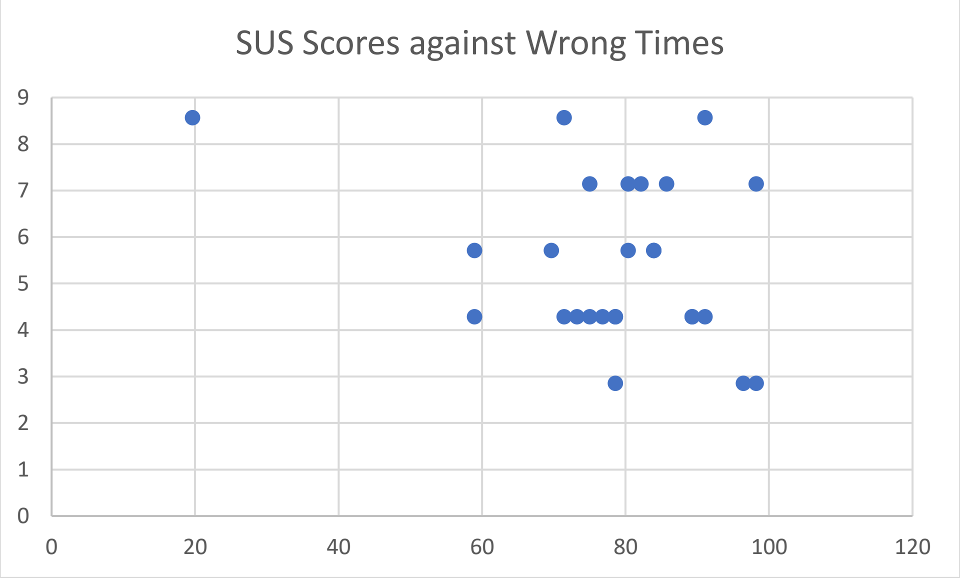


Figure 6.96: SUS Scores against Wrong Times

Figure 6.96 (previous page) compares the SUS scores against the perceived Wrong Times that participants indicated for the presentation of the PVMS. To compare the data using a t-Test, the Wrong Times was multiplied by 10 and inverted (100 – value). From this comparison, the comparison was statistically significant t(25)=8.46, p = <0.01.

SUS Score against Gesture Useful

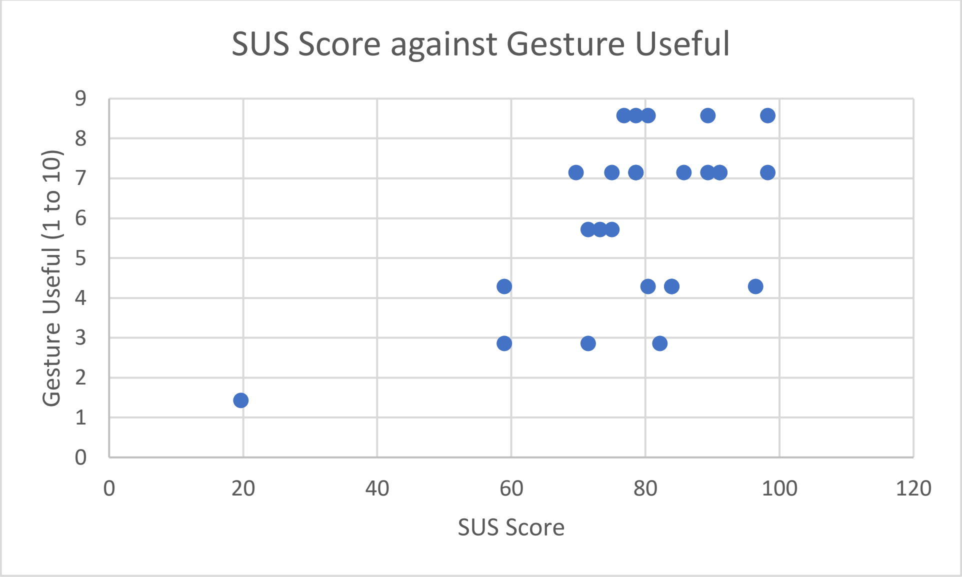


Figure 6.97: SUS Score against Gesture Useful

Figure 6.97 compares the responses to whether participants found the Gesture Useful against the SUS score. The Gesture Useful responses were scaled to the same range as the SUS scores and compared with a t-Test. They were statistically significantly different t(25)=5.23, p = <0.01. Comparing for other factors including age, computer use, and game use did not show any statistical significance.

SUS Score against Use in Future

Figure 6.98 compares expected future use against SUS Score. A similar method to the t-Test for Gesture Useful was used for evaluation. The results for Use in Future and the SUS score were statistically significant t(25)=3.05, p = <0.01. Age as a category for Use in Future was found to be statistically significant too, using an ANOVA test F(4,21) = 3.92, p = 0.02. Averages for Use in Future for each category were 8.86 for less than 21 years old (n = 5), 6.59 for 21 to 30 years old (n = 13), 6 for 31 to 40 (n = 5), 2.86 for 41 to 50 (n = 2) and 8.57 for 51 and greater (n = 1). No statistical significance was found when comparing Use in Future against Computer Use or Game Use.

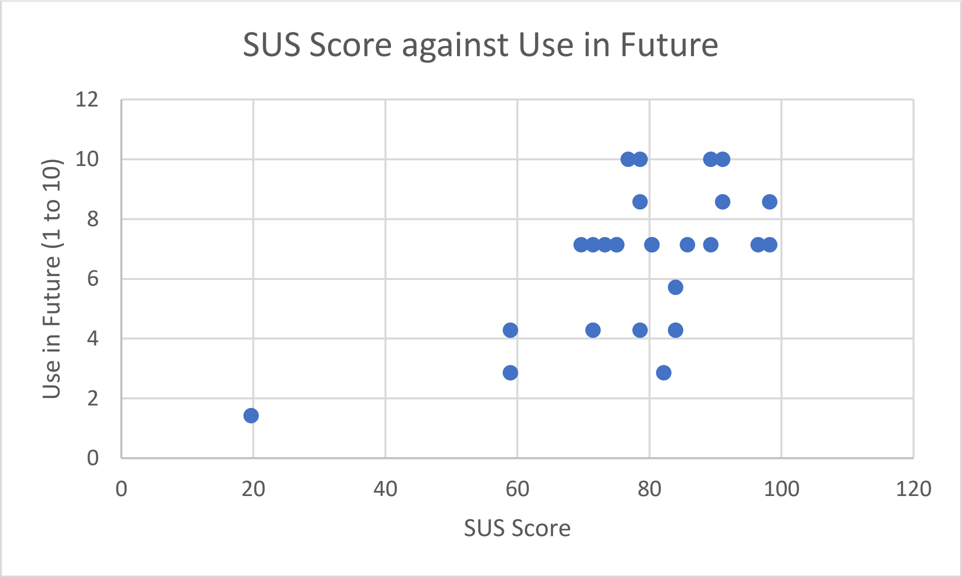


Figure 6.98: SUS Score against Use in Future

6.4 Summary of Results

The questionnaire and application data results have been presented in the previous sections and can now be considered concerning the goals outlined in section 6.1. For each of the goals, results have been drawn to demonstrate relevant information. As laid out in section 6.1, this experiment's goals were as follows.

1) Determine whether an input method provides a better overall experience in terms of usability with HMDs by comparing HMD only or combining with a controller.
2) Determine whether a PVMS is viable as a tool for interaction within this application and more broadly for other applications.
3) Determine whether improvements could be made to the proposed menu system in the way it is calibrated.

After discussing these points, there are additional brief discussions for each of the research questions and how they related to this experiment. This experiment included relevant material for all four of the following research questions as part of tying everything together.

- **RQ 1:** How can head rotations, captured through the orientation sensors of a HMD, be used to increase the user interaction capabilities with virtual worlds?
- **RQ 2:** Does using a head-mounted display with head-only interaction for menu navigation provide a similarly useful experience compared to integration with a tool for instant selection?

- **RQ 3:** Can a hidden menu provide a viable and useful tool for interaction with head movement as the mechanism for revealing it?
- **RQ 4:** Does a hidden menu revealed and interacted with using the head provide a useful experience compared to menus appearing at a fixed place in a virtual world?

Goal 1 HMD Only vs HMD with Controller

The mechanics and interactions for this experiment were informed by feedback and results from the previous experiments. One change of note was the inclusion of the gamepad controller for input as distinct from the mouse or mobile solutions from experiment one. The tap action could be completed with almost any controller with a single button. Figure 6.71 in the results section showed responses to questions regarding the usefulness between the different menus. From the figure, participants found the circular type of menu comparable to the PVMS with controller input. The additional head movements to interact with the PVMS, as described by Fitts' Law (Fitts and Posner, 1967; Scott MacKenzie, 1992), will impact time compared to the circular menu. The time saved by faster actions in the PVMS with a controller felt similarly useful to the circular menu.

The user responses (found in Figure 6.79) to the post-task single question confirm the user preference to the immediate and reduced movement interaction processes associated with the circular menu and gamepad interaction experiences. The results from these questions suggest that participants preferred the faster selection equally with controller input and the reduced movement associated with the circular menu.

As further discussed regarding goal 3, with Table 6.4, the selection time was identified as an issue. Tuning selection time to a user's preference should reduce the perceived cost of individual actions within the interface. The change to selection time would bring it more in line with the fast selections with a controller. Selection with delays will always be slower, but based on a user's confidence, they could set the interaction speed to any value.

Goal 2 Viability of the PVMS

Figure 6.66 in the results section showed some of the general feedback participants gave directly regarding the PVMS. The responses to showing the menu at the correct times were rated very highly. The sensitivity had not been altered from experiment two for this experiment, which meant the primary menu interaction changes were the two-step widget and close button. Wrong times were still very high, but this does not account for the lower impact from the two-step widget. Most of these events occurred during the tower defence tasks. The application data results show the number of times participants chose not to interact with these menus. Participants indicated the gesture was useful and there was interest in future use.

The SUS Distribution can be found in Figure 6.68 of the results section. As was discussed previously in 6.3.1, the SUS would be ranked as a B+ grade, good range, is acceptable, and a passive score for NPS. From the results of the SUS, functionality results on activation of menus and the successful completion of all experiment tasks, it is evident participants found the PVMS viable.

Goal 3 Improvements for the PVMS

Table 6.4: Enjoyable vs Difficult Features

Enjoyable Features	Difficult Features
• Easy to Use. • Intuitive. • Quick. • Felt natural. • Provided immersion. • Usable with no additional controller. • Hidden until needed and does not obscure the tasks. • Practice improved accuracy.	• Looking around too quickly made the menu appear too frequently. • False positives for menu triggering. • Positioning of menu in desired location. • Selection time. • Some fatigue/soreness of the neck.

Table 6.4 shows the enjoyable and difficult features participants reported from the questionnaire. For almost all the participants, this was the first time they had used the menu system, which inevitably meant there was a learning curve during this experiment about how best to position or trigger the menus. Looking around too quickly with this type of menu is difficult to adjust for in many cases. It may be possible to check for gestures that are too quick and ignore those. False positives were increased due to the more relaxed sensitivity configuration to gather more user data. Positioning the menu in specific places can be learned by users of the system over time. The menus will always appear in the same offset from the camera after the same angle of turning. The consistency makes it a matter of learning the gesture well enough to manipulate how it appears. Selection time was left high to ensure novice participants would not make accidental selections. The time could be dropped for a non-experiment application. Participants were largely people who had not previously used

HMDs much. Some participants would turn more aggressively than necessary to trigger the menus. Aggressive turning is something likely to cause more fatigue than necessary. Further use of HMDs would help train people to become more used to controlling applications with their head/neck; however, a system that utilised the participants' behaviour to tailor the activation of menus automatically would be beneficial. Such systems, like predictive text on mobile devices, could enable interactions that suit the user’s preferred interaction style.

6.4.1 Research Question Discussion

RQ 1: How can head rotations, captured through the orientation sensors of a HMD, be used to increase the user interaction capabilities with virtual worlds?

The third experiment evaluated a new iteration of the PVMS with improvements, particularly regarding the two-step widget. From the goal discussion, it can be concluded that the PVMS has been shown in this experiment to enhance the user experience. The interface is viable, usable and participants responded positively. Therefore, this final experiment has demonstrated a functional technique that can be added to the repertoire of interface developers to provide engagement for their users.

RQ 2: Does using a head-mounted display with head-only interaction for menu navigation provide a similarly useful experience compared to integration with a tool for instant selection?

As discussed in goal 1, participants reported a similarly useful experience between using the PVMS with HMD Only and a tool for instant selection. There will always be some trade-off in terms of time for selection when using HMD Only selection, as discussed in goal 3. This trade-off is justified, though, dependent on the context in which the interaction occurs. The use of the PVMS provides the option of a hands-free menu that can be called up whenever a user needs it.

RQ 3: Can a hidden menu provide a viable and useful tool for interaction with head movement as the mechanism for revealing it?

The PVMS demonstrated a hidden menu approach in this third experiment that improved the technique prototyped in the second experiment. Adding the two-step widget mitigated accidental reveals of the hidden interface by making the visual footprint minimal. With the simple head gesture, the demonstrated ability to call up the interface from its hidden state at will was consistent, viable, and useful. The interface can be adjusted to match the needs of applications by presenting contextually appropriate menu choices when the menu is requested. The mechanism for revealing can be adjusted to be optimal for specific applications based on the expected viewing angles or for individual users with how sensitive the PVMS is.

RQ 4: Does a hidden menu revealed and interacted with using the head provide a useful experience compared to menus appearing at a fixed place in a virtual world?

The participants were not directly asked about the comparison between the menus that were fixed in place. There can be a comparison drawn from responses such as those in Figure 6.92. The comparison between the PVMS version of the object matching puzzle (Task 1) against that of using the fixed in place circular menu (Task 2) had similar responses. Participants generally responded positively to the way the interface was presented, indicating that the culmination of different interface techniques provided a useful experience. This experiment demonstrated the PVMS as a hidden menu that could be placed freely, compared to the contextual fixed in place menus that would appear for the circular menu and other types of menus such as the tutorial menus. Each type of menu demonstrated a clear use case depending on the type of input and context required by a user.

6.5 Conclusion

This chapter presented the approach undertaken for the third experiment and outlined how it used feedback and data from the second experiment to improve the experience for the users. The four tasks conducted during this experiment provided a different element of additional data for further improvement of the PVMS. From the results, the feedback continued to show positive user perceptions toward the system. The following chapter will discuss results across the entire set of three experiments to discuss the outcomes of this research.

7 Combined Results

In previous chapters for each of the experiments, the data was presented from questionnaires and data captured during the experiment tasks, followed by short summaries directly covering the goals of each experiment. This chapter aims to further this discussion by comparing the results of all three experiments providing perspective broadly across the data. This chapter will cover a summary of the participants, look at the results and how they relate to the Periphery Vision Menu System (PVMS), consider the participant's views on interaction techniques and device preferences, investigate interface preferences, as well as the effectiveness of the experiment tasks. Finally, a discussion on the impact of the game trailer and what it meant for recruitment.

The section on the PVMS provides a lengthy discussion, as this is the key focus evaluated in this research. The other sections have been provided to introduce comparisons where interesting data points were observed between the different experiments more generally outside of the PVMS.

Each of the experiment chapters has covered how they addressed the research questions as part of their result summaries. The four research questions for reference were as follow.

- **RQ 1:** How can head rotations, captured through the orientation sensors of a HMD, be used to increase the user interaction capabilities with virtual worlds?
- **RQ 2:** Does using a head-mounted display with head-only interaction for menu navigation provide a similarly useful experience compared to integration with a tool for instant selection?
- **RQ 3:** Can a hidden menu provide a viable and useful tool for interaction with head movement as the mechanism for revealing it?
- **RQ 4:** Does a hidden menu revealed and interacted with using the head provide a useful experience compared to menus appearing at a fixed place in a virtual world?

The following list reiterates the goals for this dissertation from the introduction to evaluate the research questions. All these goals have been completed as part of the experiments and investigation of previous research. The discussion in this chapter will cover the important points used in evaluating and drawing conclusions about the success of this research.

- To investigate how user interfaces can be improved for head-mounted displays—specifically looking at the ease of interaction, tools of interaction, and presentation of interactive responsiveness.
- To develop applications demonstrating prototypes of behaviours for head-mounted interactions with a variety of input sources that improve the ease and usefulness of interaction.
- To collect user feedback from a collective of people who experience using the applications to improve the methods of interaction and interfaces.
- To draw conclusions about the usefulness, usability and other features of the proposed interfaces and interactions based on the user feedback.

7.1 Summary of Participants

Throughout three experiments, the participants were recruited to provide feedback and data for evaluating the implementation of the PVMS. This section summarises the data related to samples that participated in each experiment.

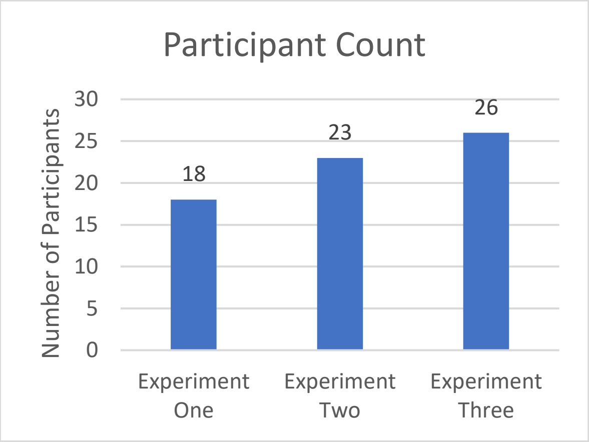


Figure 7.1: Participant Count

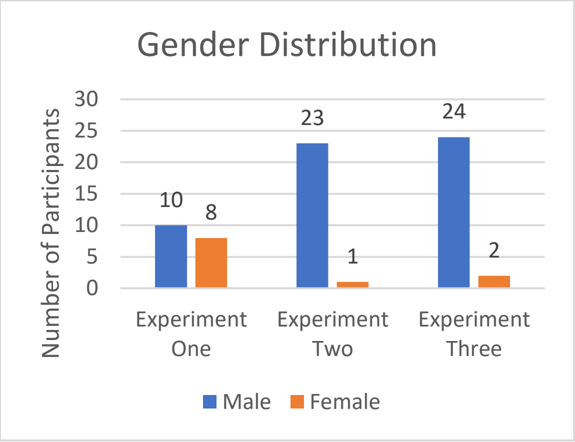


Figure 7.2: Gender Distribution

A total of 67 participants were used across the three experiments to collect data, as seen in Figure 7.1. With 18 participants in the first experiment, 23 in the second experiment (24 including one participant who only completed the pre-experiment questionnaire), and 26 in the third experiment. Gender distribution shifted dramatically after the first experiment. Figure 7.2 shows that almost half of the participants in the first experiment were female. Compared to the single female in experiment two and two in experiment three. The low number of female participants was comparable to the YouTube statistics data that indicated only 0.7% of the second experiment video viewers were female. In contrast, the third experiment video fared better with 7.5% of female viewers.

It was felt the increase in participants was due to several factors. The first experiment did not use a creative recruitment campaign to gather participants. It was believed that participants would be intrigued by the idea of trying out the Oculus Rift as an interesting activity to do while at the same time participating in research. A short game trailer was used to drive interest for experiments two and three. The trailer is believed to have helped make participating more appealing. Sources of participants were only from within the university and then primarily from those doing computing topics. The gender distribution for students in computing topics was heavily male-dominated. The gender distribution in the primary place of advertising does suggest why the number of participants for the experiments was primarily male. The rough equality of gender for the first experiment was unusual; however, those approached for the first experiment predominantly came from first-year topics. First-year topics typically have a larger pool of students from both genders to recruit. It should be considered, given the gender bias of the participants, that further study should be conducted to confirm the usability findings across all user groups.

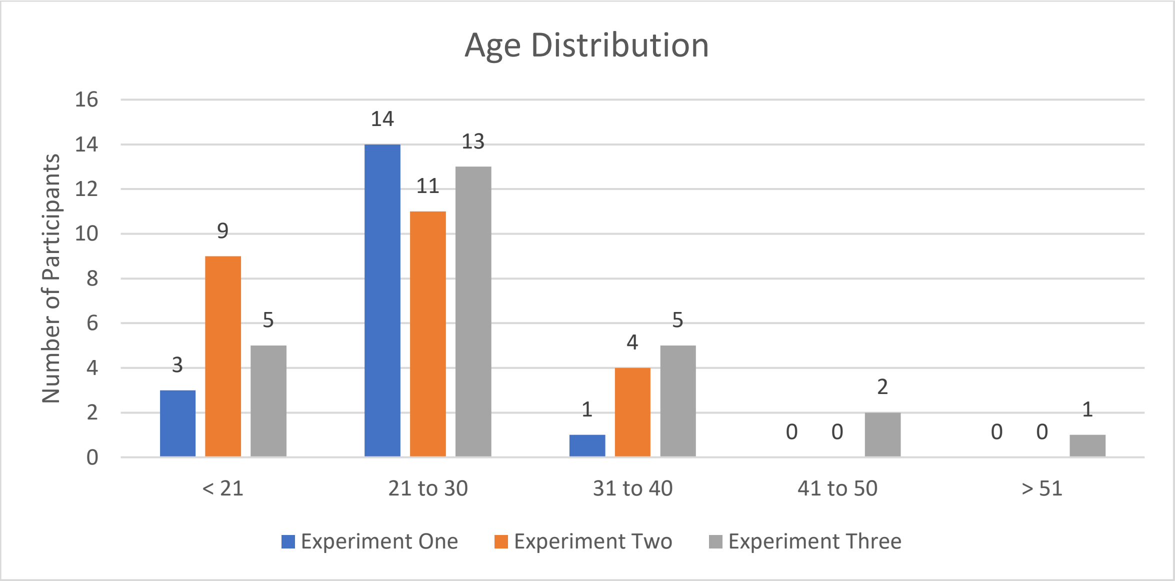


Figure 7.3: Age Distribution

Figure 7.3 shows the distribution of ages. The largest quantity of participants was between 21 and 30, followed by participants younger than 21 (older than 17 as per the ethics requirements). The quantity of participants 31 and older increased over the second and third experiments. The age ranges seem very typical of the average student who would be attending university. With many students coming directly from high school, many first-year students would be between 18 to 20 years of age.

Three participants attended all three experiments, and one participant attended both the second and third experiments. All other participants were unique for each experiment. Figure 7.4 shows a comparison of the recruitment data for all three experiments. Although each experiment had fewer total messages of initial interest for participation, the number completing was the reverse of those who had requested more information and not continued. It is believed the trailer contributed to what is seen here. The very low number of participants who participated in multiple experiments does suggest the participants may have been mostly interested in experiencing using the technology. Due to the experiments being conducted in separate semesters, it is also possible the participant's conditions changed in ways that made it harder for them to participate. Perhaps due to different topic loads, different other commitments, or where they had left university between the experiments.

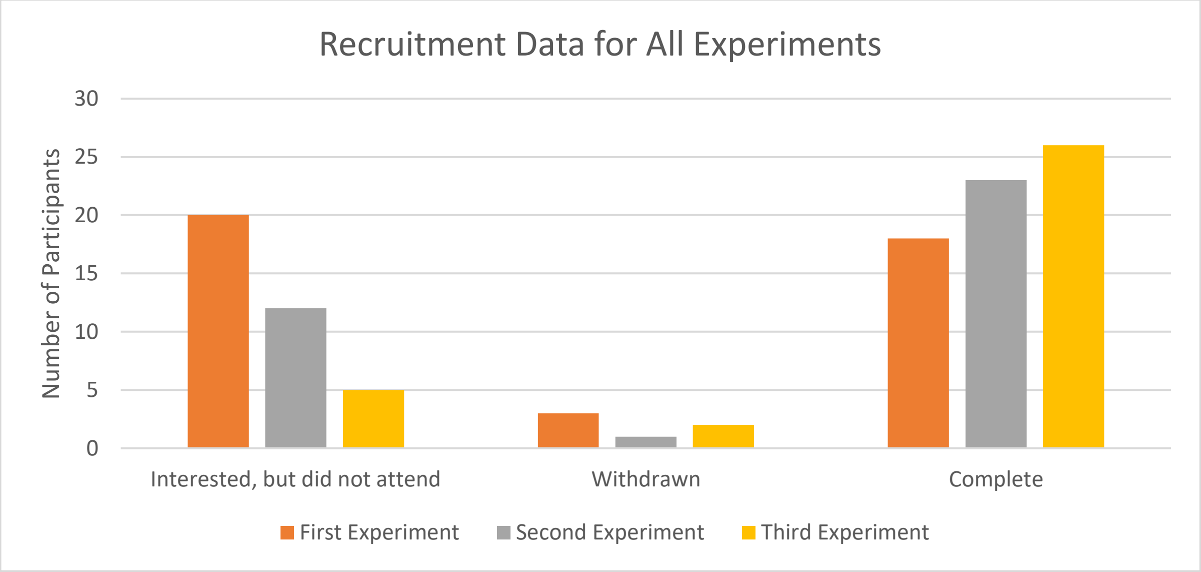


Figure 7.4: All Experiments Comparison of Participation

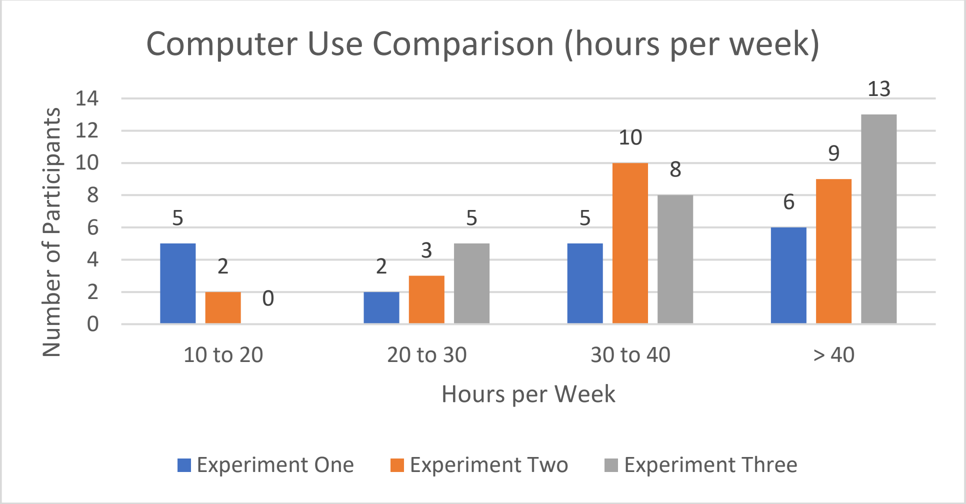


Figure 7.5: Computer Use Comparison (hours per week)

Participants mostly specified (Figure 7.5) that their weekly computer use was higher than 30 hours a week. The second and third experiments had higher quantities of participants who indicated higher computer use per week. Comparatively, participants specified across all three experiments an average of fewer than 10 hours per week were spent on playing games, as shown in Figure 7.6. The low time spent playing games is useful to consider because it shows that although the participants were using a computer for a high number of hours on average, those hours were spent not playing computer games. 10 hours is quite a lot for some to fit into a week for playing games. It may have been more appropriate to include some lower ranges such as up to 2, up to 5, or other similar amounts.

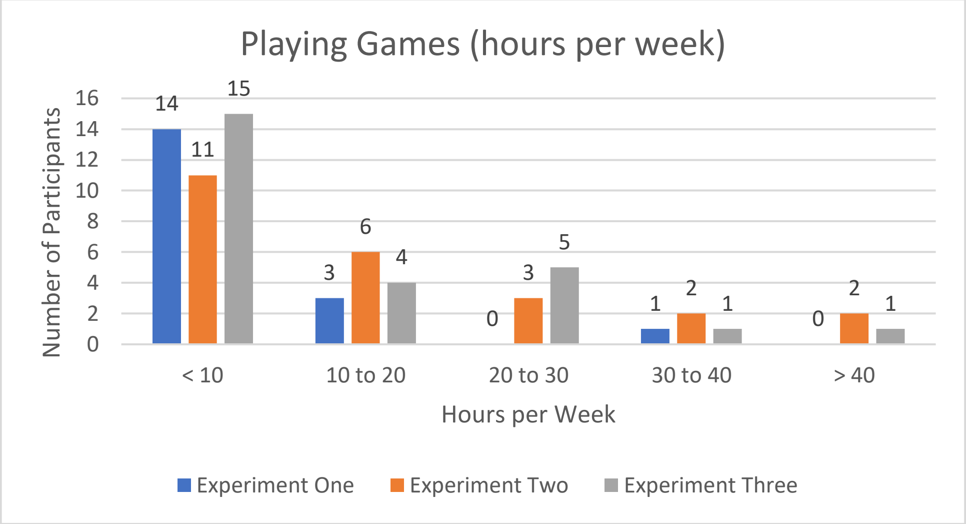


Figure 7.6: Playing Games (hours per week)

This section considered the numbers of participants and the distribution of their gender, age, computer use for those who participated. This information demonstrates that there was an increasing number of participants over the experiments, which is believed to be from an improved advertising campaign. The change in gender parity from experiment one to experiment two and three does suggest there could have been something improved to target more females. The overwhelmingly male numbers for experiments two and three may have introduced some bias to the data.

7.2 The Periphery Vision Menu System

This section has separated the many different comparative data related to the PVMS into specific sections. The data and analysis presented here will focus on the second and third experiments because they were the only experiments that directly evaluated the prototype system. This section will begin with a look at the participants' perception of the usefulness of the PVMS (section 7.2.1), followed by general feedback (section 7.2.2). Following this is a discussion of the post-experiment responses (section 7.2.3) and the findings from the System Usability Scale (section 7.2.4). Finally, the section will conclude with a look at the preferences for the use of the PVMS (section 7.2.5), the event types (section 7.2.6) and what the interactions were with the opened menu (section 7.2.7).

7.2.1 Periphery Vision Menu Useful

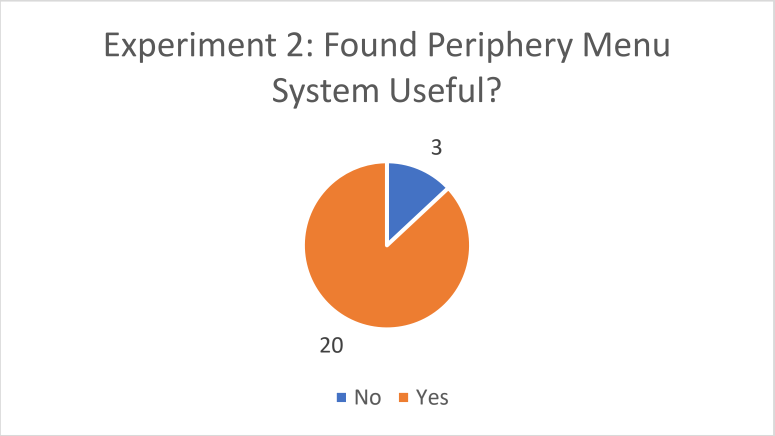


Figure 7.7: Experiment 2: Found Periphery Menu System Useful?

As part of the second and third experiments, there were two different measurements regarding the usefulness, providing an overall representation of how participants felt about the usefulness of the PVMS. In Figure 7.7, the participants in the second experiment overwhelmingly indicated they found their experience with the PVMS useful. In this experiment, all the tasks had been focused on using the HMD Only interactions with the prototype menu. The response for the second experiment can be compared to those in the third experiment seen in Figure 7.8. The data was represented differently to provide contrast between three different options with comparative type questions. In this experiment, the participant had experienced some alternate types of interfaces and the change of using an interaction controller. Participants preferred the circular-type menu augmented with a periphery menu for object creation compared to utilising the periphery menu alone.

The preference can be speculated to result from the method of controlling the object directly with the attached menu while keeping focus. This interaction method is contrasted against tasks where the periphery menu was controlled with a tap-to-select through the Xbox controller. Participants, in this case, felt the periphery menu was similar, with the high rating suggesting many of the issues experienced by participants were related to the time for selection, as this was the only real advantage given by the controller.

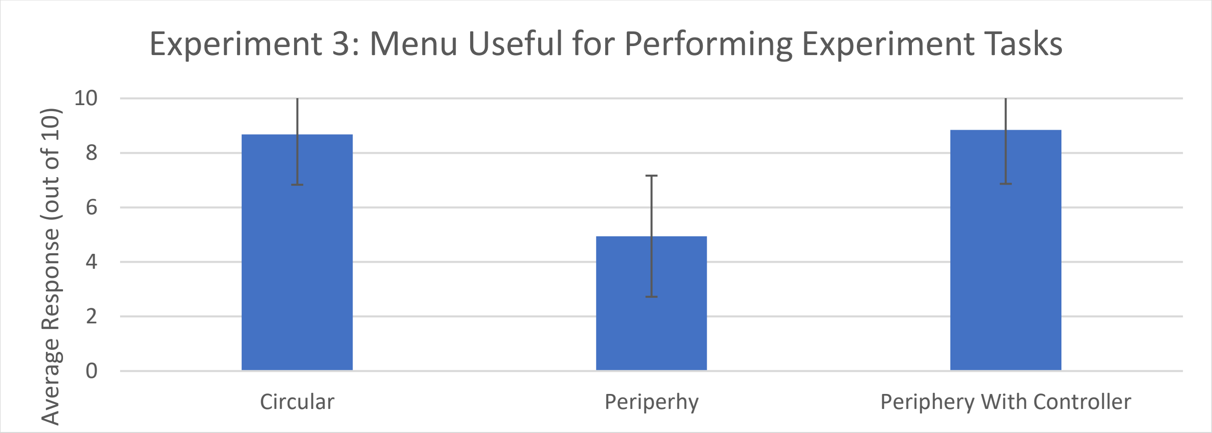


Figure 7.8: Experiment 3: Menu Useful for Performing Experiment Tasks

The data has a statistically significant difference for the values shown in Figure 7.8 for comparing two pairs of the different values. Comparing the Circular and Periphery values gives F(1,50) = 43.46, p = < 0.01. Similarly comparing Periphery against the Periphery with Controller gives F(1,50) = 44.67, p = < 0.01. However, when comparing the Circular response with Periphery with Controller, the result of F(1,50) = 0.1, p = 0.76 shows no significance. This result validates the significance of preference toward incorporation of the additional options for interaction.

This data validates that the PVMS was found to be useful and demonstrates that it worked alongside the other types of menus. It was also shown that the PVMS could be used with appropriate selection techniques based on the application, function and end user's desires.

7.2.2 General Feedback

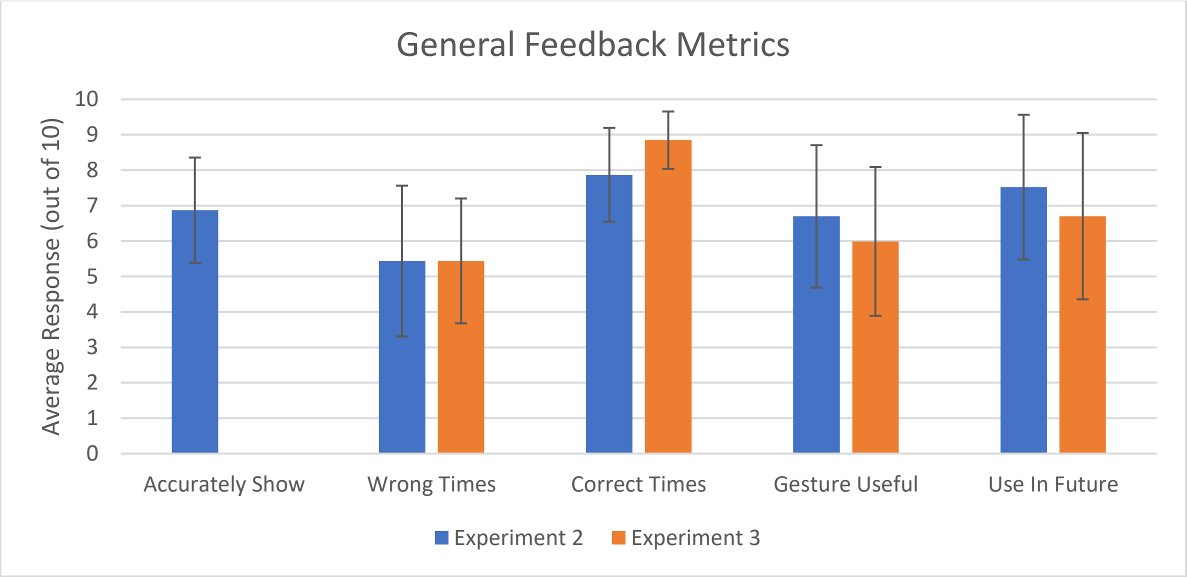


Figure 7.9: General Feedback Metrics

In the second and third experiments, participants were asked general questions about their thoughts on the Periphery Vision Menu System (Figure 7.9). The responses for both experiments were very similar to the data collected through observation. Participants indicated they felt the menu appeared at the wrong times about the same for both experiments. The correct times the menu was displayed was indicated to be higher with lower deviation. In contrast, there were slightly lower ratings for participants finding the gesture useful and the desire to use the system in the future. From the second to third experiment, it was hoped that participants would naturally lower the rating for wrong times based on the mitigation strategy deployed with the two-step widget. In 7.2.7, the wrong times will be directly compared to the actual number of times participants did not make an action with the interface.

In looking at the statistical significance of this data, the correct times response was found to show a statistically significant difference with a result of F(1,47) = 9.94, p = < 0.01. The other three attributes did not indicate a statistically significant difference when comparing the wrong times: F(1,47) = 0, p = 0.99, the gesture's usefulness: F(1,47) = 1.44, p = 0.24, and the desire to use in future: F(1,47) = 1.67, p = 0.2. Comparing within the individual experiments between the wrong and correct times, responses demonstrated a statistically significant difference of F(1,44) = 21.7, p = < 0.01 in the second experiment and F(1,50) = 80.22, p = < 0.01 in the third experiment.

This data suggests there is a correlation between how participants perceived the system's function and the ways it would react to them. The ratio of perceived times the PVMS triggered in the wrong and correct times will be explored further in section 7.2.7 by combining the data with actual use data taken from within the application.

7.2.3 Post-Experiment Task Responses

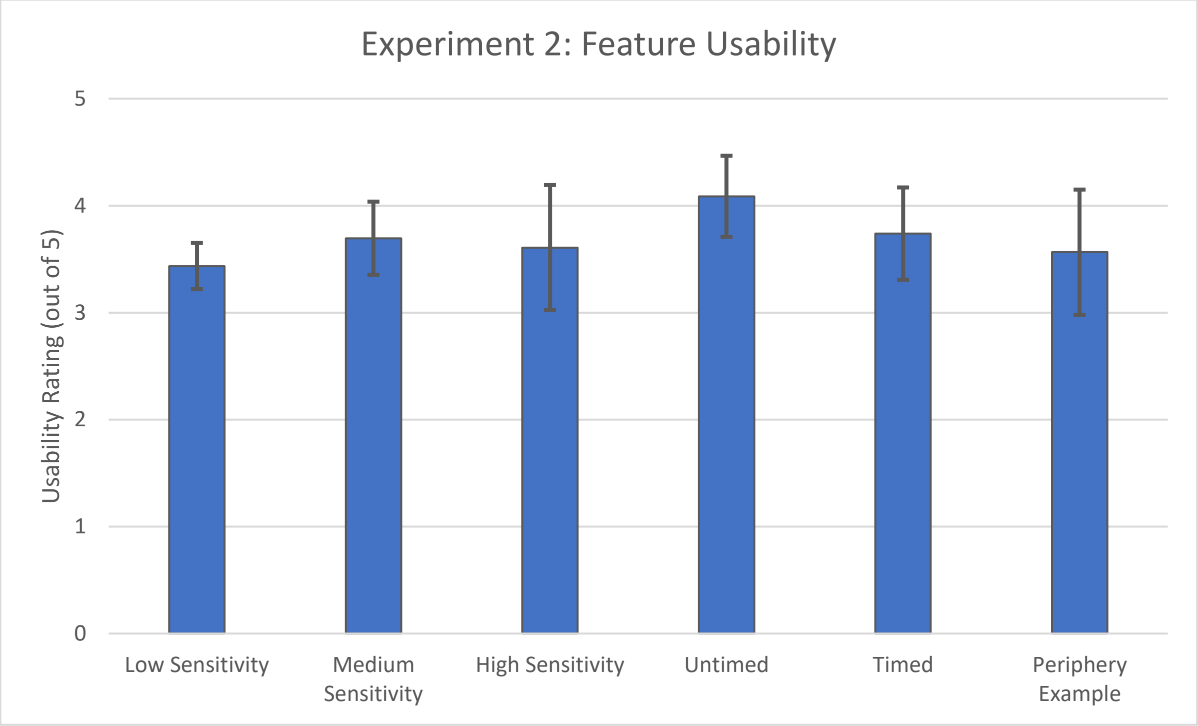


Figure 7.10: Experiment 2: Feature Usability

Both experiments two and three provided feedback points from participants in the moments right after they had completed the tasks. These questions evaluated the feedback using a different choice of wording between experiments two and three. In the first experiment, the goal was to determine if the system was usable. The results shown in Figure 7.10 are all at favourable average scores out of 5 with minimal variance. This minimal variance indicates participants felt each stage was usable in ways that worked for each task, where they indicated the untimed activity was the most usable from observation. In the third experiment’s responses, shown in Figure 7.11, there was a higher variance in responses, with very positive responses toward the circular and tower defence with controller tasks. For untimed this was 4.09 (SD = 0.38) in experiment two versus 3.62 (SD = 0.98) and 4.5 (SD = 0.9) in experiment three. For the timed tower defence game this was 3.57 (SD = 0.59) in experiment two versus 3.65 (SD = 1.02) and 4.3 (SD = 0.79) in experiment three.

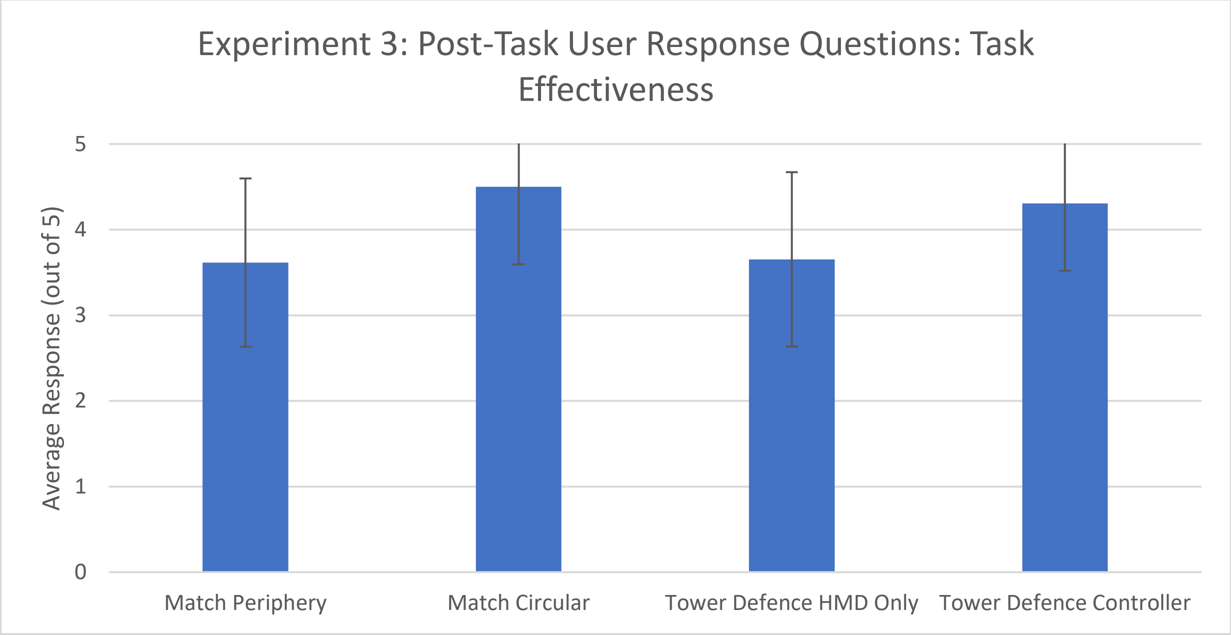


Figure 7.11: Experiment 3: Post-Task User Response Questions: Task Effectiveness

From general observation, the values from experiment 2 in Figure 7.10 all appeared similar, while deeper analysis supported this by revealing no statistical difference between them. Comparing the responses for low, medium, and high gave a result of F(2,66) = 0.36, p = 0.7. Comparing untimed vs timed gave a result of F(1,44) = 1.69, p = 0.2. The data in the third experiment (seen in Figure 7.11) comparing between similar tasks can be shown as statically significant. The matching task with a result of F(1,50) = 11.39, p = < 0.01, and the two tower defence games in the third experiment with a result of F(1,50) = 6.71, p = 0.01, indicate statistical significance. The main point for comparison between the experiments was the tower defence game. Comparing the results of the second experiment's tower defence game with each of the third experiment's shows no significance for the HMD Only input with a result of F(1,47) = 0.09, p = 0.77. The comparison between experiment two's tower defence and the use of a controller in experiment three was statically significant though as shown by F(1,47) = 5.15, p = 0.03.

This data does indicate that the addition of the controller was a noticeable factor. The controller did increase the perceived effectiveness by a small margin. Overall, the effectiveness was rated highly in all the presented scenarios, demonstrating validation of the system as an alternative interaction technique. The statistically significant increase for using the circular menu also suggests that the use of menus should be considered for how they relate to the type of activity interaction.

7.2.4 System Usability Scale

The System Usability Scale was utilised in all three experiments; therefore, the data for the first experiment has been included here as a point of comparison. This section will begin by comparing the individual metrics that go into the scale, then comparing the distributions of scores, and finally looking at the overall averages for all scores.

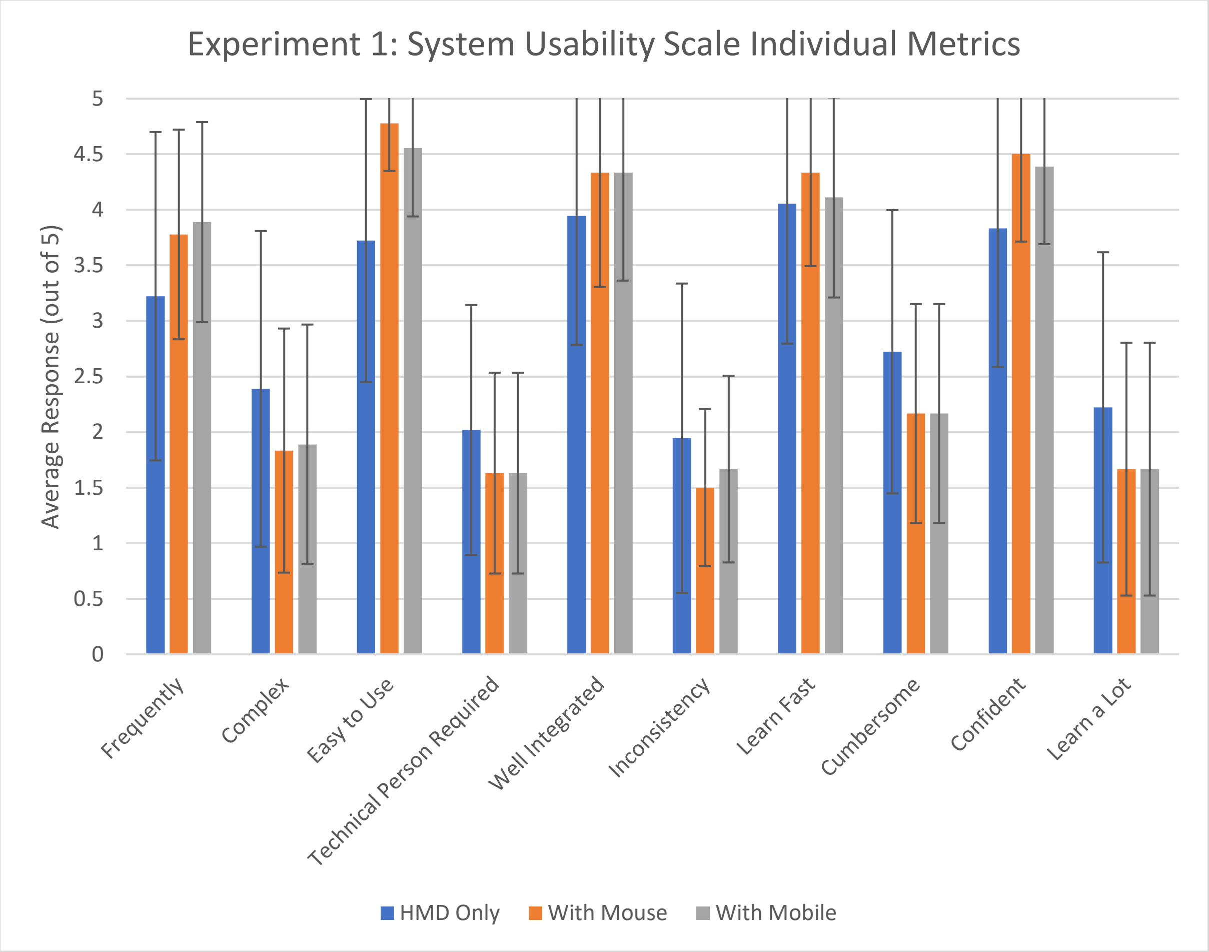


Figure 7.12: Experiment 1: System Usability Scale Individual Metrics

In Figure 7.12, the three separate input techniques are shown from experiment one. In the first experiment, interactions with the HMD Only type interactions were more limited and did not benefit from the improvements seen in experiments two and three. The SUS data for experiments two and three can be seen in Figure 7.13 (over). Specifically, the addition of the PVMS and other improved menus for making the experience more cohesive. The order of ranking was similar. In all experiments focusing on HMD Only interactions, the highest metric was “learn fast”. “Well Integrated”, “Easy to Use”, and “Confident” ranked around the same values. The “Use Frequently” response was consistently the lowest metric compared with the other positive intent questions in the SUS. Participants responded that the systems felt more cumbersome than the other negative worded questions listed for all experiments. All the other SUS questions where a lower value indicated a better total score had consistent low values indicating positive reception of the system.

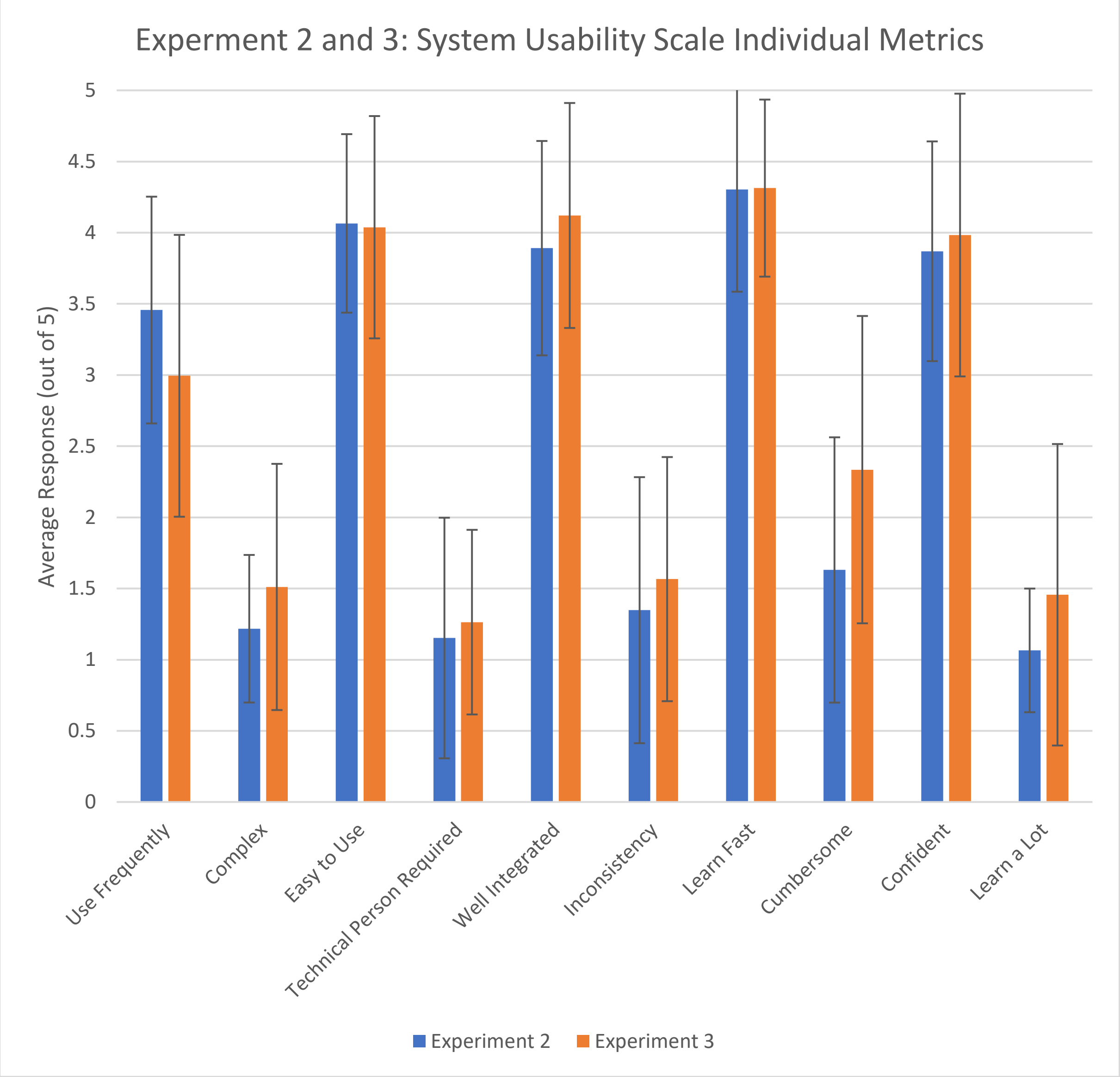


Figure 7.13: System Usability Scale Individual Metrics

Table 7.1 (over) shows that only the cumbersome metric shows a statistically significant difference when compared between the two experiments. As this was the highest average negative phrased question, participants felt most strongly that improvement was needed for this aspect of the system. This aspect of targeted improvement will be a significant area for work in future.

Table 7.1: Comparison for Statistical Significance (SUS Individual Metrics)

Metric	ANOVA Comparison between Experiment 2 and Experiment 3
Learn Fast	F(1,47) = 0, p = 0.96
Well Integrated	F(1,47) = 1.08, p = 0.3
Easy to Use	F(1,47) = 0.02, p = 0.9
Confident	F(1,47) = 0.2, p = 0.66
Use Frequently	F(1,47) = 3.18, p = 0.08
Cumbersome	F(1,47) = 5.9, p = 0.02
Inconsistency	F(1,47) = 0.73, p = 0.4
Complex	F(1,47) = 2.01, p = 0.16
Learn a Lot	F(1,47) = 2.72, p = 0.11
Technical Person Required	F(1,47) = 0.27, p = 0.6

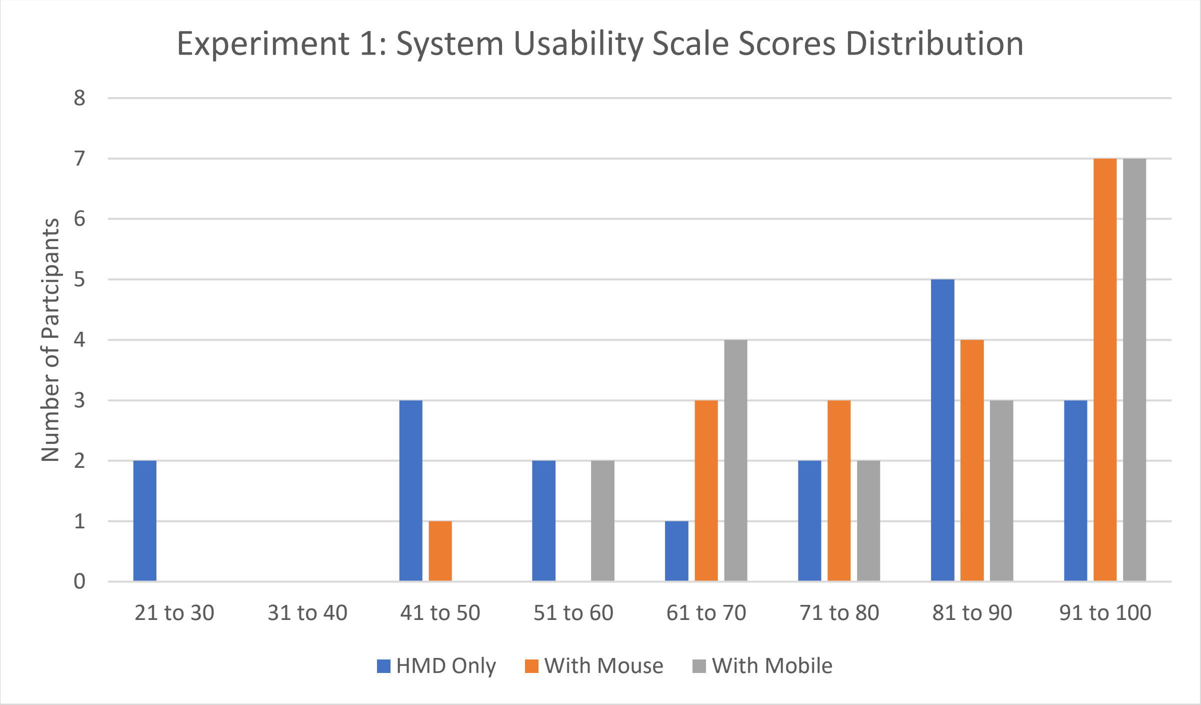


Figure 7.14: Experiment 1: System Usability Scale Scores Distribution

Figure 7.14 shows the distribution for each interaction technique from the first experiment compared to the second and third experiments in Figure 7.15 (over). The System Usability Scale scores in the first experiment do not demonstrate any perceived consistency, with some low scores mostly for the HMD Only type input technique. A score of 68 is considered average

(Sauro, 2011; Bangor et al., 2009). Most scores are observed to be over this average from the categorisation distribution for all three experiments. This being above the average is especially true for the second and third experiments, with most scores above 68.

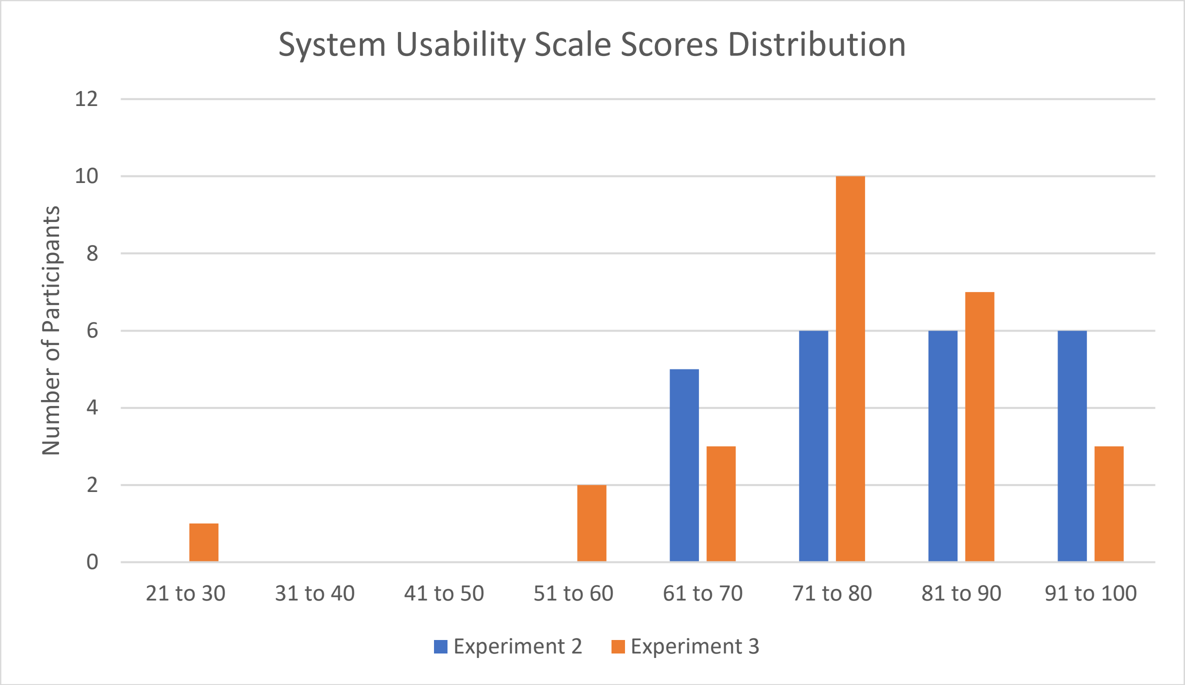


Figure 7.15: System Usability Scale Scores Distribution

Figure 7.16 (over) shows the overall comparison of average scores for the System Usability Scale between all the data from the three experiments. Experiment one had scores of 68.47 (SD = 22.79) for HMD Only, 82.36 (SD = 13.65) for Mouse, and 80.69 (SD = 15.45) for Mobile. Experiment two had a score of 82.17 (SD = 11.26), and experiment three had a score of 76.41 (SD = 14.75). These results show the high variance from experiment one for the HMD Only type input. All the scores were around or above the average SUS score of 68. Experiment three's one outlier dropped the score below the second experiment. The score was left in the data for all calculations to show that not all participants were entirely happy with the system. A review of these scores suggests that they positively evaluate the success of the PVMS and the shift from a lower score for HMD Only type interaction to those used in the second and third experiments. Comparing these results, experiment 2 showed a statistically significant difference against the HMD Only version of experiment 1 with a result of F(1,39) = 6.36, p = 0.02. Comparing within experiment 1 between the HMD Only system and mouse was also statistically significant with the result F(1,34) = 4.92, p = 0.03. Comparing the other data did not show any statistically significant difference: experiment 1 (HMD Only) against experiment

3 F(1,42) = 1.97, p = 0.17, experiment 1 (HMD Only) against (mobile) F(1,34) = 3.55, p = 0.07, mouse against mobile F(1,34) = 0.12, p = 0.73, and experiment 2 against experiment 3 F(1,47) = 2.32, p = 0.13.

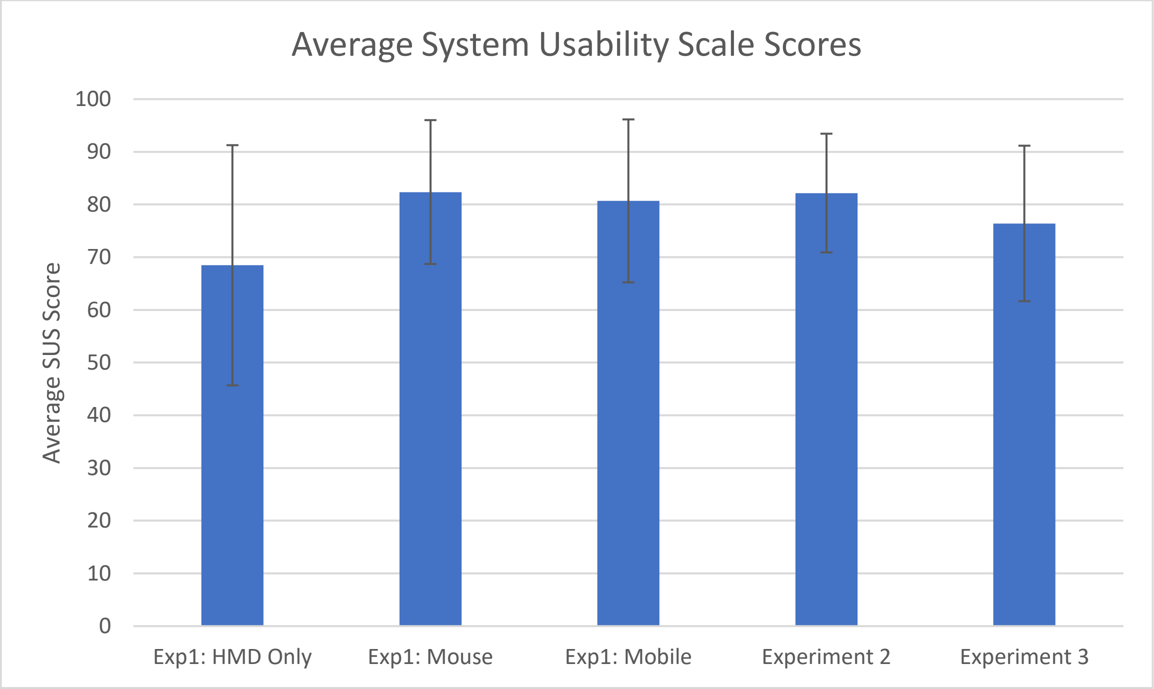


Figure 7.16: Average System Usability Scale Scores

From the significance and other represented data, it can be summarised that the system saw a significant improvement to HMD Only interaction between experiment 1 and experiment 2. The lack of statistically significant difference between experiment 2 and experiment 3 indicates no dramatic change between the response to experiments. The lack of dramatic change is good because the changes were minimal, indicating the system was effective in users finding the usability similar across the two experiments. The shared support between experiments means from this data that the design approach is effective and usable.

7.2.5 Preferences for Periphery Vision Menu Use

In the second and third experiments, participants were asked to rank their preferred uses for the Periphery Vision Menu System shown in Figure 7.17 (over), where a lower score indicated the activity was preferred. Participants were asked to rank on a scale of 1 to 7 for experiment 3 and 1 to 5 for experiment 2, where 1 represented the most desired use, and the highest numbers (5 or 7) represented the least desired use. The third experiment included two additional scenario options to provide a wider array for participants to consider. On average, games were preferred more than constructing models or controlling operating systems. The third experiment saw a shift toward preferring two other usage scenarios over games on average, with the use as a virtual cinema and tour guide (like the use case described in Chapter 4) functionality coming out ahead. The most dramatic shift in rankings was participants in the third experiment considering messaging as an activity they could foresee using the system for over constructing models, operating systems or browsing the internet. When compared to results from the second experiment, it was ranked last by a substantial margin. The variance, particularly for the third experiment, shows that different participants had varying desires for individually prefer to use the techniques. The variance in preference suggests that participants had different agendas regarding how they would use the PVMS, whether that be as an entertainment, professional, or utility integrated technique.

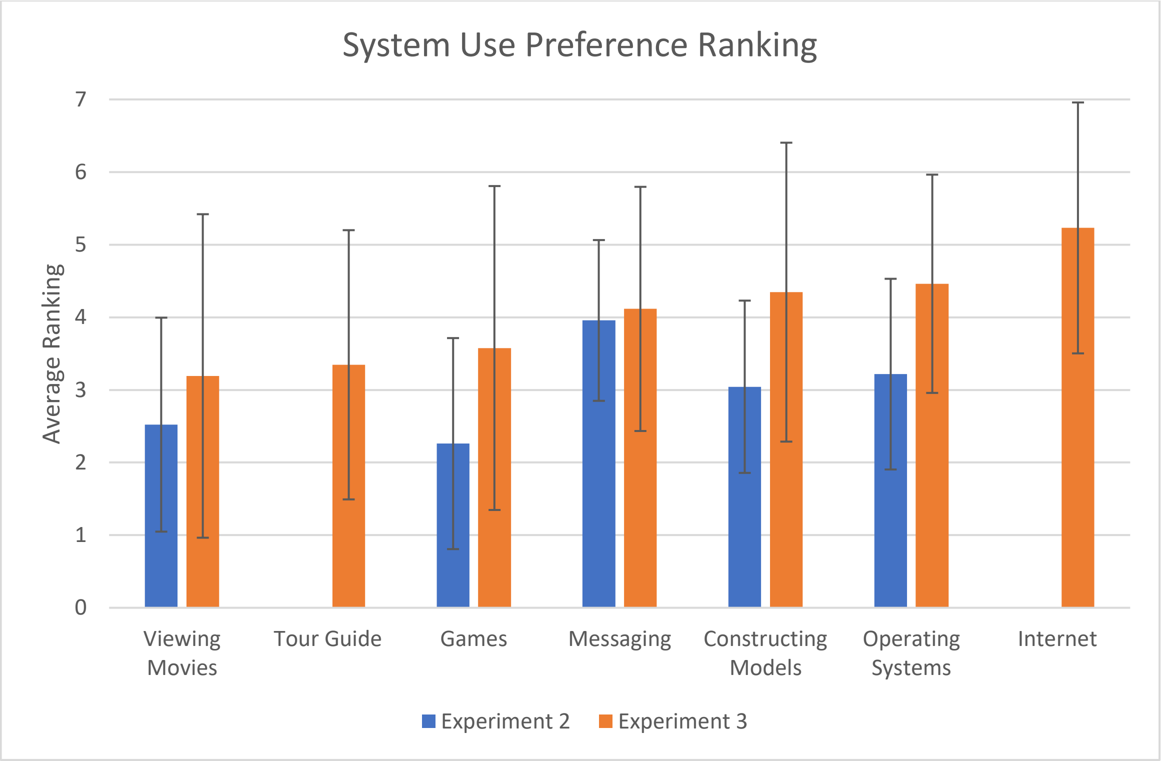


Figure 7.17: System Use Preference Ranking

7.2.6 Periphery Vision Menu Event Types

In this section, a summary of how participants triggered menus is explored. The way they were triggered is important because it shows concisely how the PVMS was used with the gesture-based triggers. The trigger data evaluated includes the primary trigger from a left or right rotation and how the secondary conditions related to **`Diff_X`** (Pitch) and **`Diff_Z`** (Roll) affected the interactions.

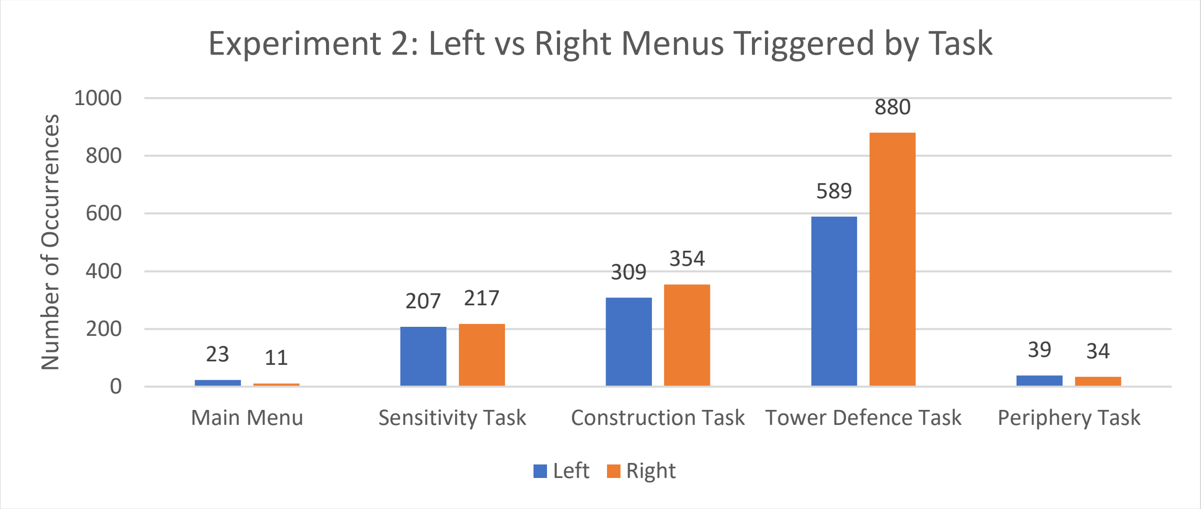


Figure 7.18: Experiment 2: Left vs Right Menus Triggered by Task

For most tasks, there was a limited difference between opening a menu on the left or the right for the type of menu that would appear. The main exception for this was the tower defence game during the second experiment. During the second experiment, a menu on the left would provide functionality for modifying a selected tower, and a menu on the right would show a create tower menu. The data relating to triggers shown in Figure 7.18 and Figure 7.19 (over) indicate participants were opening menus in both directions close to equal in most scenarios. These data do not account for mistakenly opened menus or those where no action was taken. These two types of menu results will be discussed separately in section 7.2.7. The data collected does not directly provide a way to determine the cause for the equal spread. One possible reason for the distributions may be due to the benefit from surveying during continuous movement. Once a participant had opened a menu, then performed an action, they would typically return their focus to the central task or otherwise begin to survey the area for what they wished to do next. Participants may have found it easier to continue turning their heads in the opposite direction to the previous menu. Another possible reason may have been due to individual user preferences, similar to how people are normally preferential toward either a left or right hand. The primary takeaway from this left vs right data is that there is no reason to suggest menus appearing on only the left or right will be preferred universally.

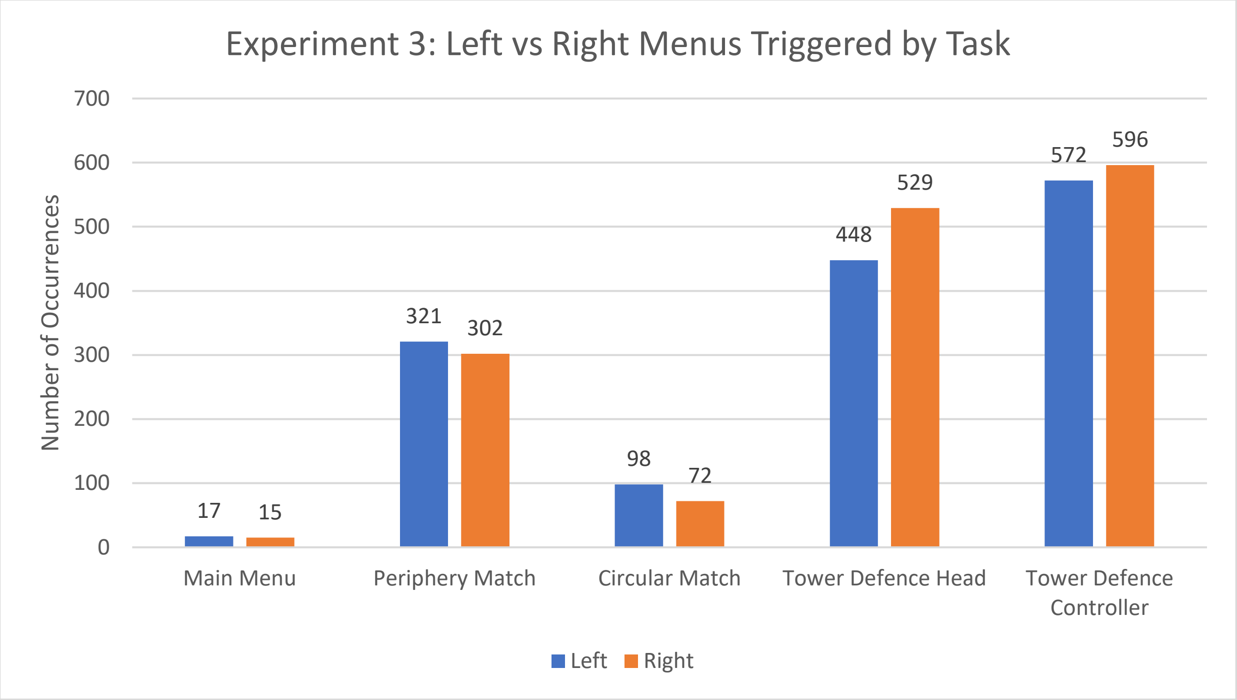


Figure 7.19: Experiment 3: Left vs Right Menus Triggered by Task

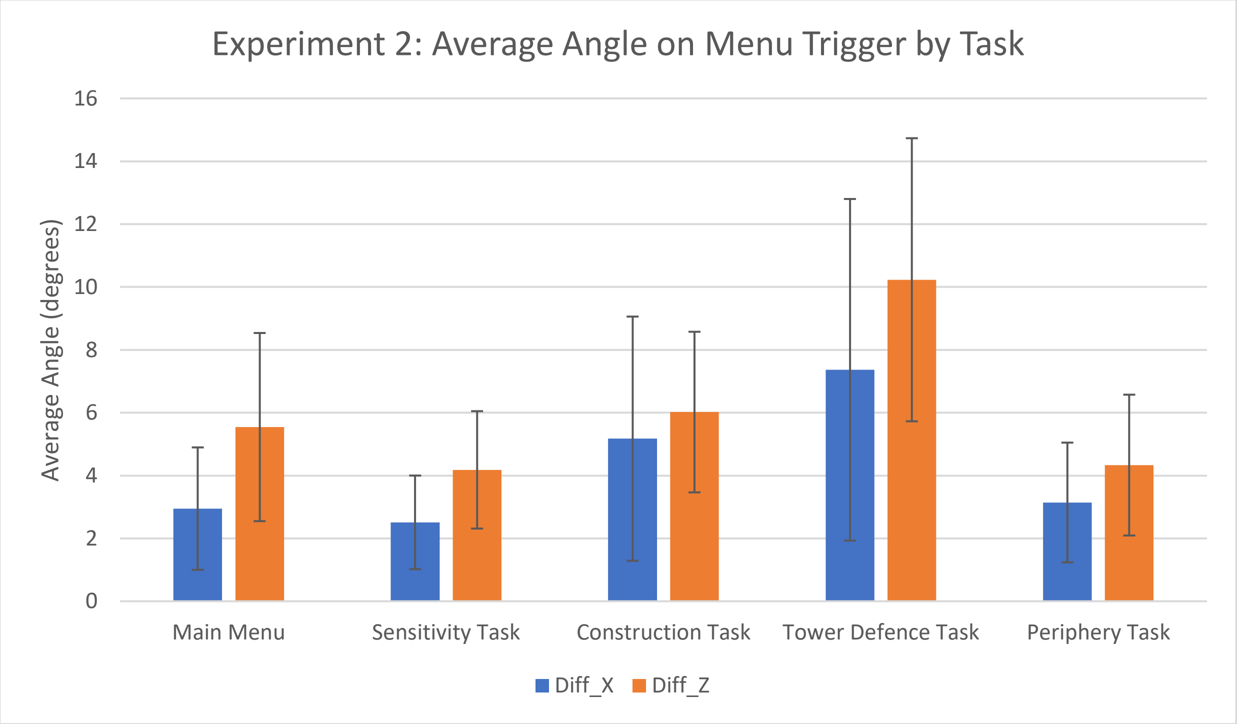


Figure 7.20: Experiment 2: Average Angle on Menu Trigger by Task

Figure 7.20 and Figure 7.21 (over) show the average angles for each task used to generate the PVMS events. The Diff_Z (Roll) values showed a higher variance across tasks and consistently higher values based on the angle of interaction. The angle of interaction can be seen throughout the screenshots presented in Chapter 5 and Chapter 6. The tasks of the second experiment were all completed while looking in mostly a "forward" direction, except for the tower defence task where larger rotating angles were required to look around at controlled towers effectively. The variance in angles is then shown more dramatically in the third experiment (Figure 7.21), with similar averages for each type of task. The higher values are consistent relative to the amount a participant would need to be viewing the tasks at a downward angle. The required perspective to view tasks for the third experiment required a larger initial downward angle than those used in the second experiment. The difference in angles can be observed from the screenshots from the related tasks. The Diff_X (Pitch) values were very consistent across all tasks for the third experiment with some variance in the second experiment with a higher value, particularly for the tower defence task.

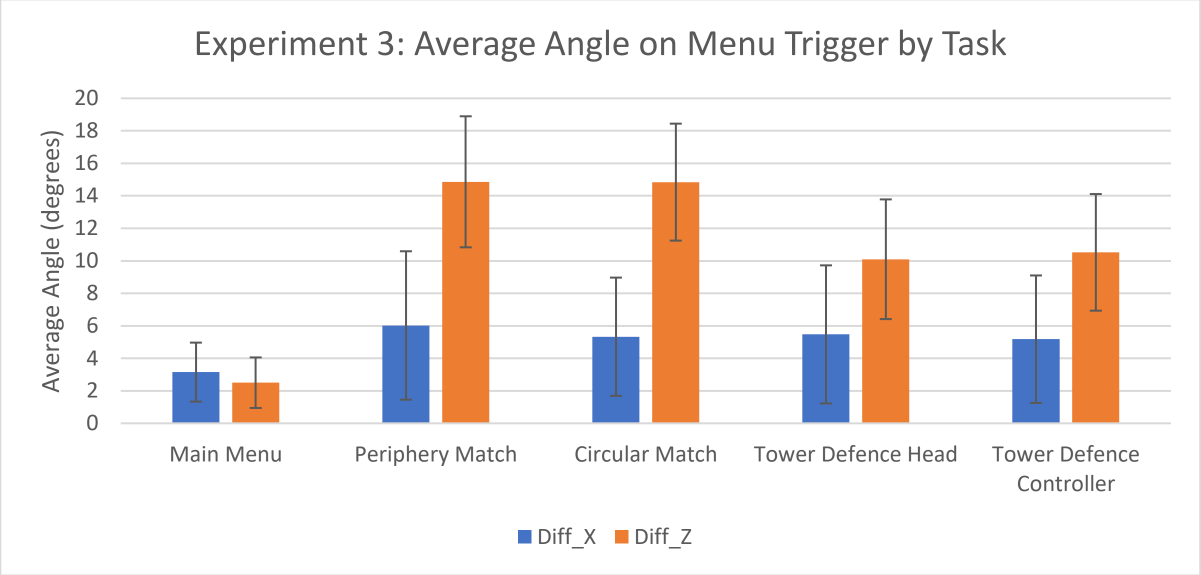


Figure 7.21: Experiment 3: Average Angle on Menu Trigger by Task

The maximum values for **`Diff_X`** in each experiment were: 7.37 (SD = 5.43) in experiment two and 6.02 (SD = 4.57) in experiment three. The maximums for **`Diff_Z`** in each experiment were: 10.23 (SD = 4.5) in experiment two and 14.86 (SD = 4.03) in experiment three. For this data, it would be reasonable to suggest sensitivity settings could be based on these numbers. Looking at one standard deviation above the average would give values of 12.8 and 10.59 for **`Diff_X`** and values of 14.73 and 18.89 for **`Diff_Z`**. The test values were around 25 to 30 degrees for the **`Diff_X`** and **`Diff_Z`** values in the different sensitivities used for experiment two. They were both set to 30 for the medium sensitivity used across all experiments. This data suggests values closer to 13 for **`Diff_X`** and 19 for **`Diff_Z`** would be more appropriate. These lower thresholds could significantly reduce the number of wrong times menus appeared for participants. As is shown in the data from the different tasks, it is necessary to consider the types of activities being completed with the system. Sensitivity configurations may be configured separately for different activities, evaluating how they are best suited for use with different sets of thresholds to improve the benefit for slower expected actions, compared against higher frequency actions.

From this data, the activation process for how participants triggered the menus has been shown. This evaluation has considered the direction of menu creation and the average additional constraints from **`Diff_X`** and **`Diff_Z`**, looking at task-dependent changes. From the data, it can be observed that the **`Diff_Z`** angle required a higher allowed variance for processing the detection of the gesture from actual usage, with values for each angle proposed as potentially viable defaults where they can be tuned to any specific application.

7.2.7 Interactions with the Opened Menu

Having looked at the way participants triggered the menu in the previous section, this section will compare what participants did once the menu was open. Starting with a review of the event occurrences discussed in the experiment chapters (3, 5, and 6), then showing a simplified view of this data and contrasting the results against how participants responded to the perceived number of wrong executions in the questionnaires. Then finishing with a comparison of the active menu times across the experiments.

Figure 7.22 (over) shows the total occurrences for each event interaction with the PVMS in experiment two. “Button Events” indicate any action where a participant directly interacted with the menu. An event type of “Hide from Inactivity” meant the menu was left open until it automatically closed. “Replaced while Active” meant there was already a menu open, and a new menu was opened, resulting in the original being overridden. During the Menu Sensitivity Test task, participants found that it was not necessary to interact with the menu. This figure shows that most of the participants still applied a button event to close the menu. Participants successfully applied a “Button Action” with nearly every opened menu during the tower construction task. Then during the tower defence task, participants had a high number of both “Hide from Inactivity” and situations where the menu was “Replaced While Active”. The speculated reasons for these numbers being so high were a combination of the tuning on the sensitivity and participants evaluating their options. Participants often opened menus to either create a tower with not have enough currency or opened one menu but decided they wanted the other menu before making an action. These types of choices are difficult to represent accurately. Participants were not asked why they used the different menu actions. The use of the close button was observed to result from changing their mind about needing to use the menu or realising they could not currently perform a useful action given their current situation.

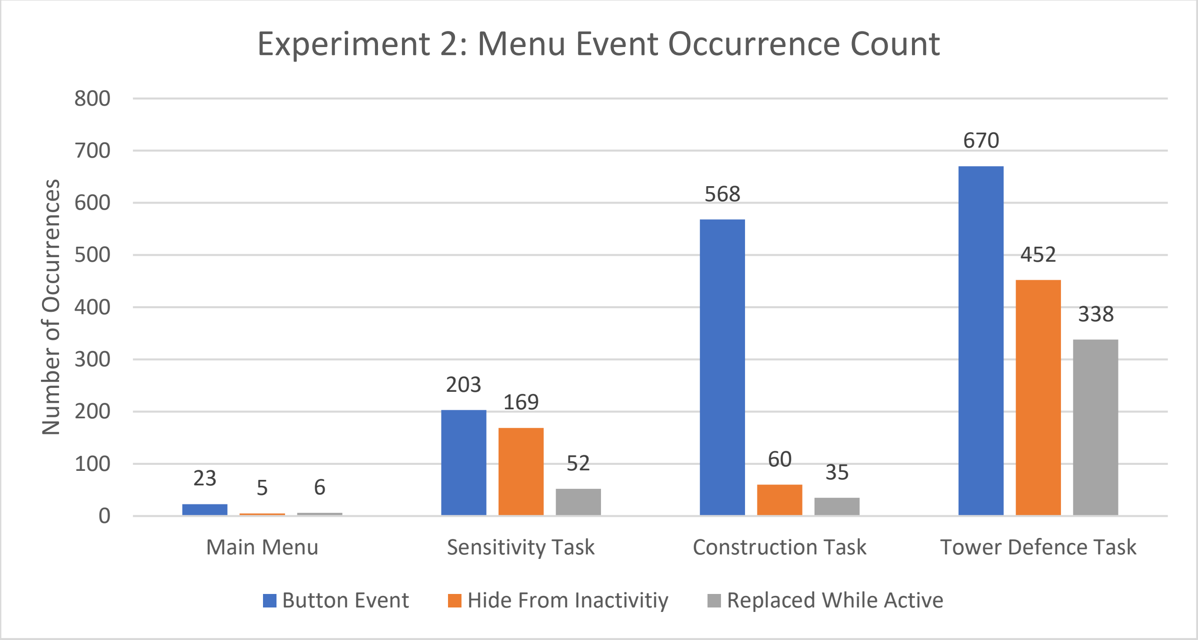


Figure 7.22: Experiment 2: Menu Event Occurrence Count

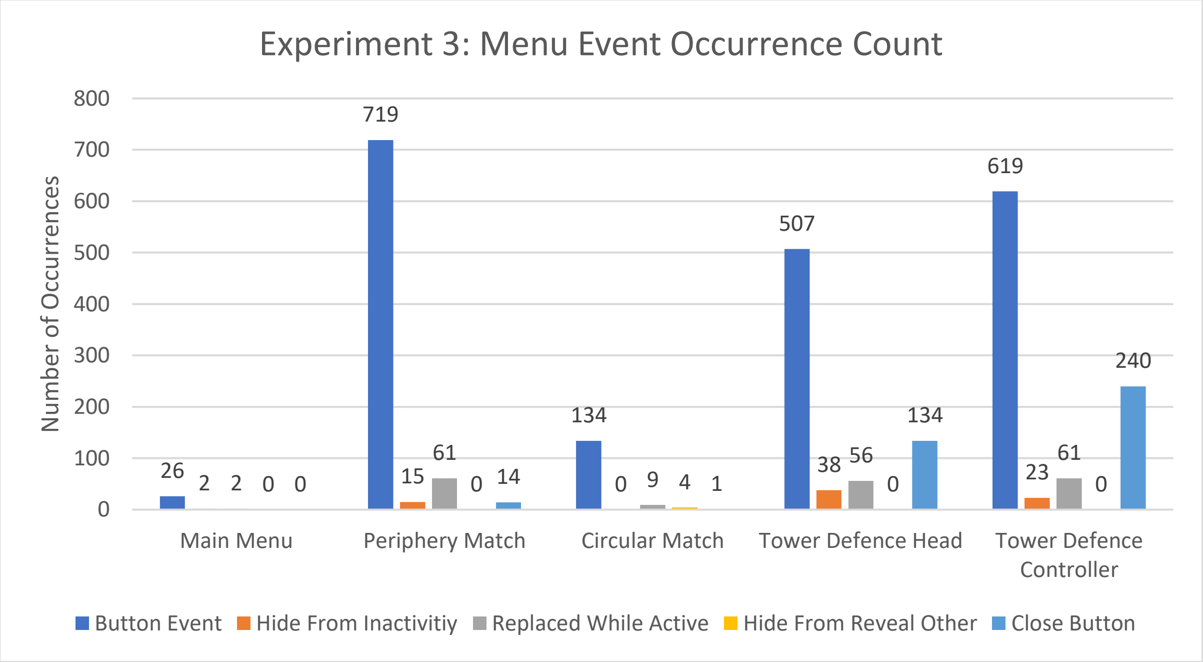


Figure 7.23: Experiment 3: Menu Event Occurrence Count

During the third experiment, the menu events were divided into two different stages with the addition of the two-step widget. A successful button action had to be completed with the two-step widget to reach the opened menu. There were also two new types of events to track in this experiment. "Hide from Reveal Other" referred to opening the circular menu, forcing the closure of any open Periphery Vision Menu. The other action was the use of the "Close Button". Figure 7.23 (previous page) illustrates the count of the various activities associated with the PVMS. From this data, we can see that once a menu was successfully opened, the number of "Hide from Inactivity" events was almost none, compared to the total number of "Button Events". It was far more common for the participant to use the "Close Button". The ease of use and the short interaction time for terminating a menu with the controller were the main influences for the high jump in the tower defence with the controller test. Participants knowing this were more likely to open a menu to check if they could make a tower and then choose to terminate the menu easily.

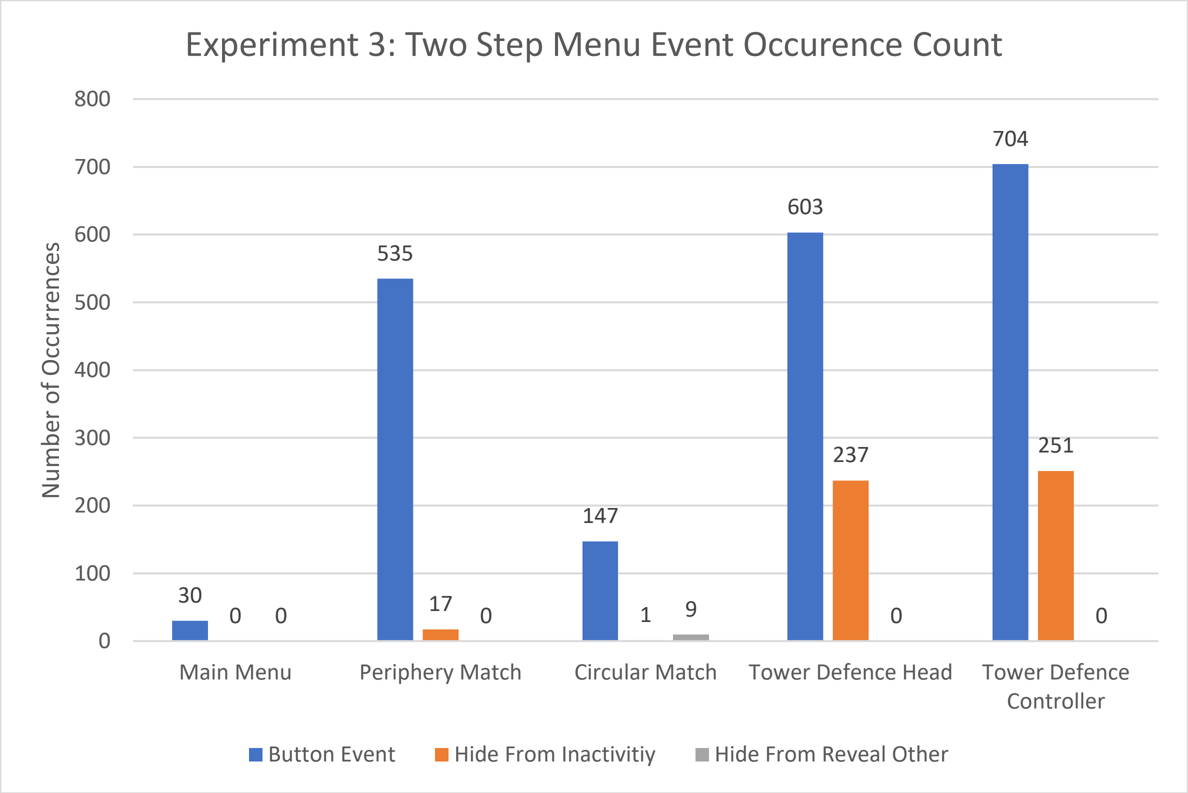


Figure 7.24: Experiment 3: Two-Step Menu Event Occurrence Count

The data from experiment three's PVMS can be compared against the data for the two-step widget button, as seen in Figure 7.24. Every "Button Event" is a successfully opened menu, and every "Hide from Inactivity" indicates when participants let the two-step widget automatically hide. Almost all the hide events occurred during the tower defence tasks with similar quantities. These were quite lower than the number of accepted actions. Comparing this back to Figure 7.22 and Figure 7.23, there is a change in how participants engaged with the relatively high button interaction counts compared to hide/replace type events for the third experiment. The variation in event types indicates that there was mitigation of user error in experiment three compared to experiment two by giving the user a choice to close the interface or let the two-step widget time out.

The figures that follow (Figure 7.25, Figure 7.26, Figure 7.27, and Figure 7.28) will show the total uses of the menu system calculated using menu events. These figures will summarise the difference between successfully used menus and menus where “No Action” was taken. For experiment two, the single entry point for the menu makes it simpler to summarise than experiment three. The averages in Figure 7.25 (over) were calculated using “Button Action” divided by 23 (where n=23 was the number of participants) for “Button Action” and the sum of “Hide from Inactivity” and “Hide from Reveal” other divided by 23 for “No Action”. It is useful to consider this data because it shows the actual use of the PVMS relative to the types of tasks performed.

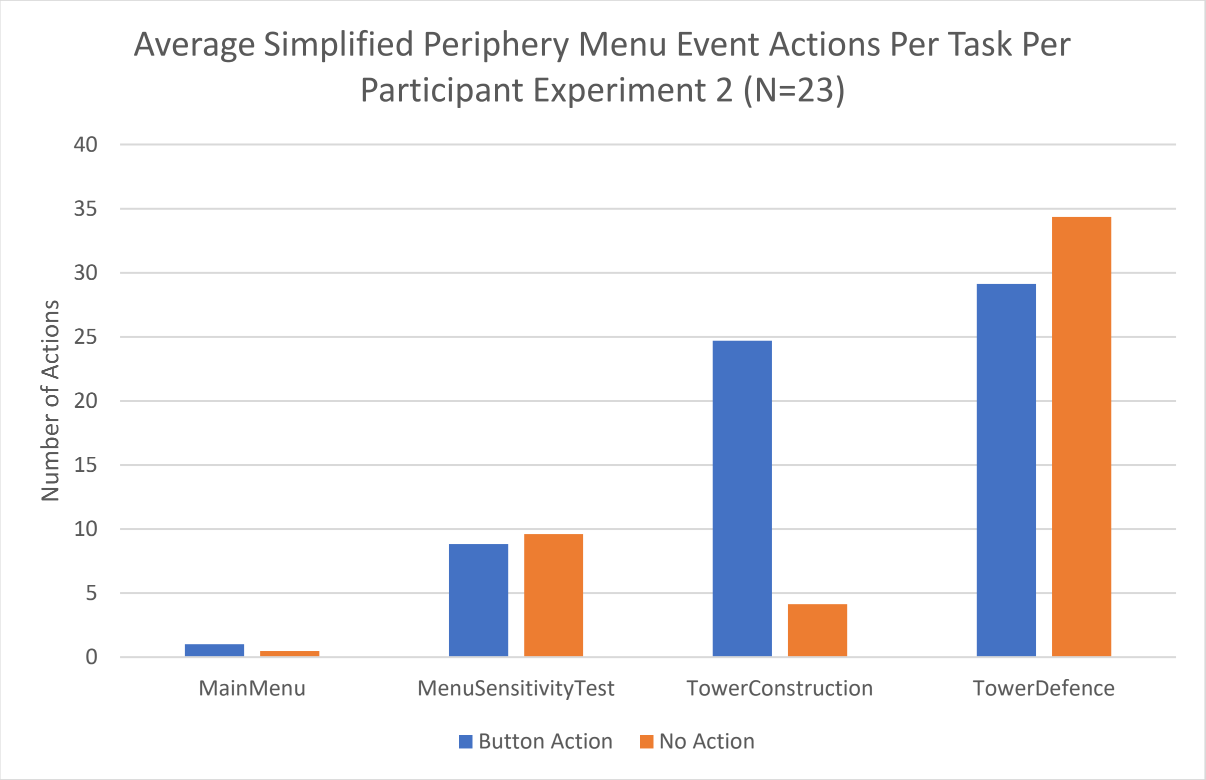


Figure 7.25: Average Simplified Periphery Menu Event Actions Per Task Per Participant Experiment 2

The third experiment was expanded to include reporting for “Minimal” user engagement and is calculated as an average per participant, as seen in Figure 7.26. “Button Actions” are the button events from the equivalent two-step widget (from the Periphery menu) with “Hide from Inactivity”, “Hide from Reveal Other”, “Replaced while Active”, and “Close Button” subtracted. The “No Action” value is calculated using the sum of Periphery menu values for “Hide from Inactivity”, “Replaced while Active”, “Hide from Reveal Other”, and “Close Button”. “Minimal” uses the sum of “Hide from Inactivity” and “Hide from Reveal Other” using two-step widget numbers. “No Action” refers to any situation where the participant partially completed an action but decided not to continue the action to completion. The “Minimal” value indicates only the two-step widget was shown, indicating that although the participant revealed the widget, they chose not to reveal the menu. The overall visual impact is mitigated in this scenario, making a possibly incorrect menu trigger a negligible issue.

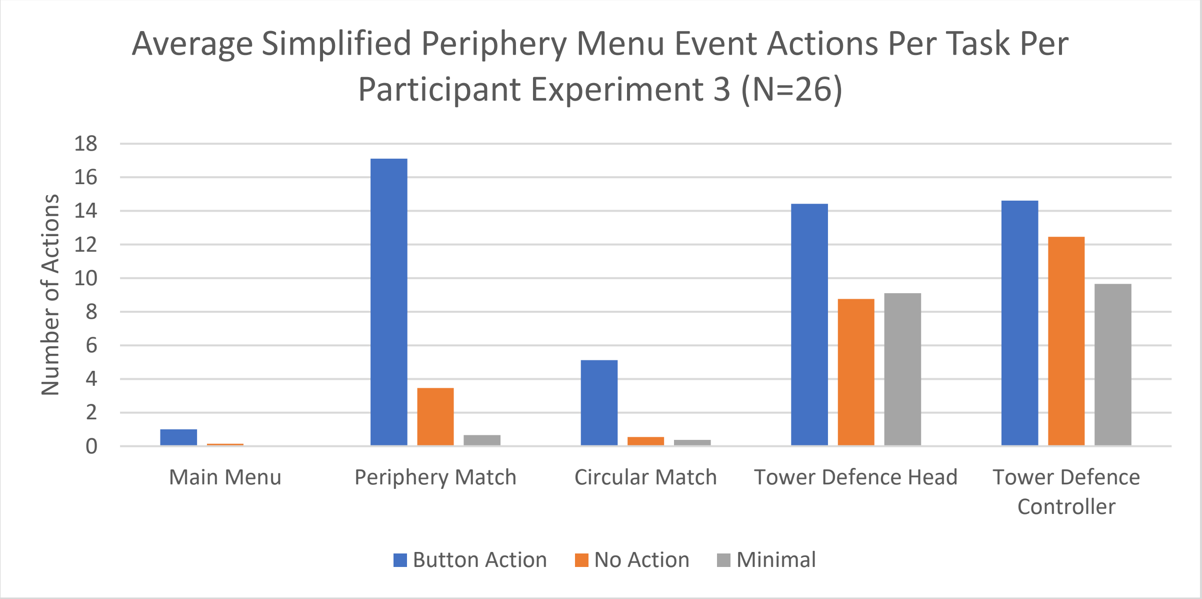


Figure 7.26: Average Simplified Periphery Menu Event Actions Per Task Experiment 3

This data suggests participants did open the menu a substantial number of times during the tower defence games where no continued action was taken. These instances were considered to have minimal impact on continued actions. A similar number of average menu actions were conducted between the two different tower defence tasks. The lack of direct action to remove accidental menus indicates that the PVMS does not suffer significantly from unintentional menu reveals by the user on an average basis.

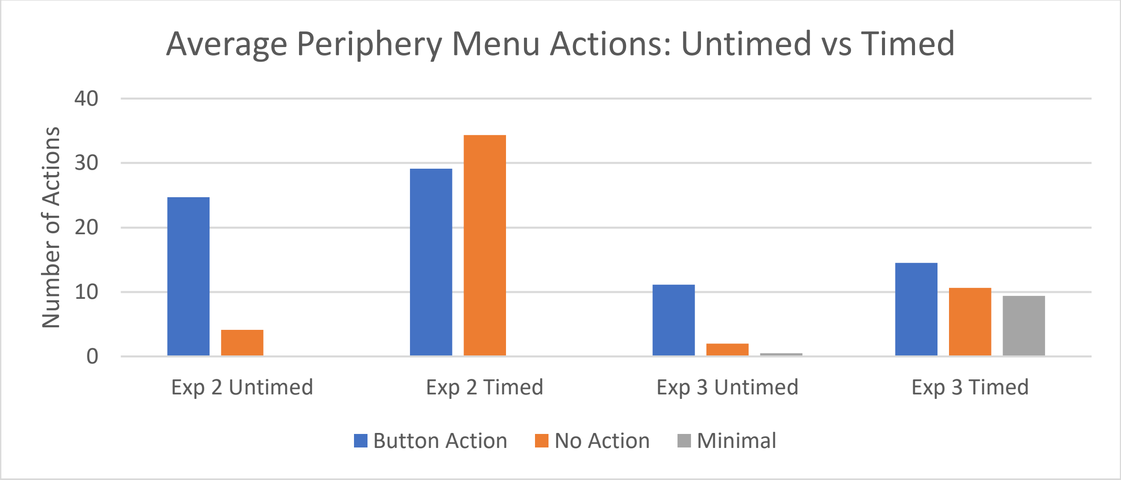


Figure 7.27: Average Periphery Menu Actions: Untimed vs Timed.

Figure 7.27 shows a comparison between timed and untimed activities for each experiment. The values reported are the average values for each type of menu interaction event discussed above. For experiment three, this is an average between the two related untimed and timed tasks. The main takeaway from this figure is that menus very rarely had "No Action" applied for both experiments for untimed tasks. We can speculate that this may have indicated a lesser feeling of pressure to view the menus by the participants resulting from more calculated choices for menu use. In timed tasks with the different versions of the tower defence game, participants opened more menus with "No Action" than when a "Button Action" was used. In the third experiment, this was mitigated significantly by the smaller two-step widgets. The lesser number of overall actions in experiment three was significantly down to restructuring experimentation with circular menus and the placement of tower repairs.

Figure 7.28 (over) shows an average of each classification of PVMS actions for each experiment compared against a "Weighted Wrong" value. The "Weighted Wrong" values are calculated from the questionnaire responses using the following formula: **`Weighted Wrong = (Wrong / Correct) * Button Action`**. This calculation assumes a relationship between the number of successfully completed "Button Actions" as a relationship to the perceived mistakes the participants believe they made, captured from their questionnaire. This figure shows a visible correlation between the "Weighted Wrong" values and the "No Action" results. In the second experiment, this was close with a lower quantity for the weighted wrong result. The lower result suggests the overall impact was lower for wrong cases in the third experiment than the second when related to participants' perceived rating for the number of wrong actions they believe they executed.

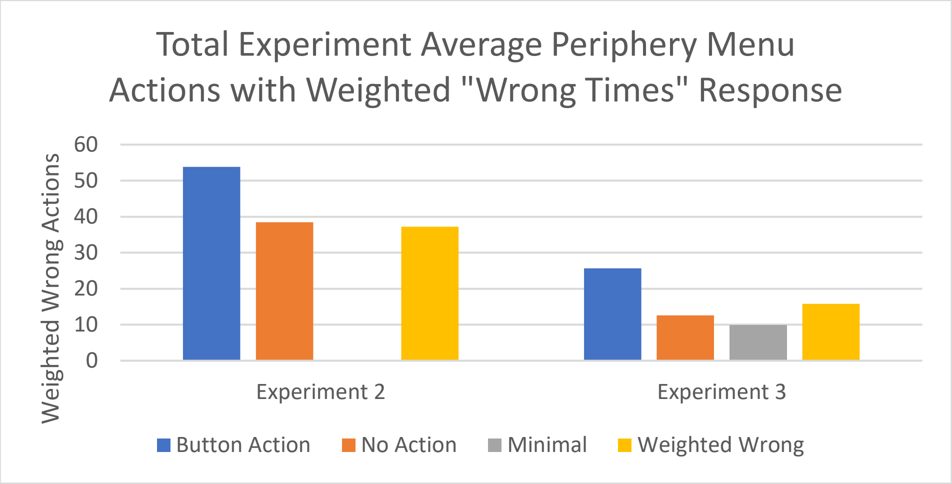


Figure 7.28: Total Experiment Average Periphery Menu Actions with Weighted "Wrong Times" Response

A number of observations can be made from comparing the amount of time the menus were open in Figure 7.29 and Figure 7.30 (over). Participants, on average, interacted with the main menu as a button event faster in the third experiment. Interactions during the tower defence game were similar for HMD Only in the third experiment and experiment two. The second time participants played through with a controller; the interactions were faster on average. The difference is not significant when considering the faster speed of interaction from the instant selection. The similarity suggests that the interactions with the PVMS were acceptable and did not unnecessarily hinder the ability of the participants to perform the tasks with the demonstrated approach.

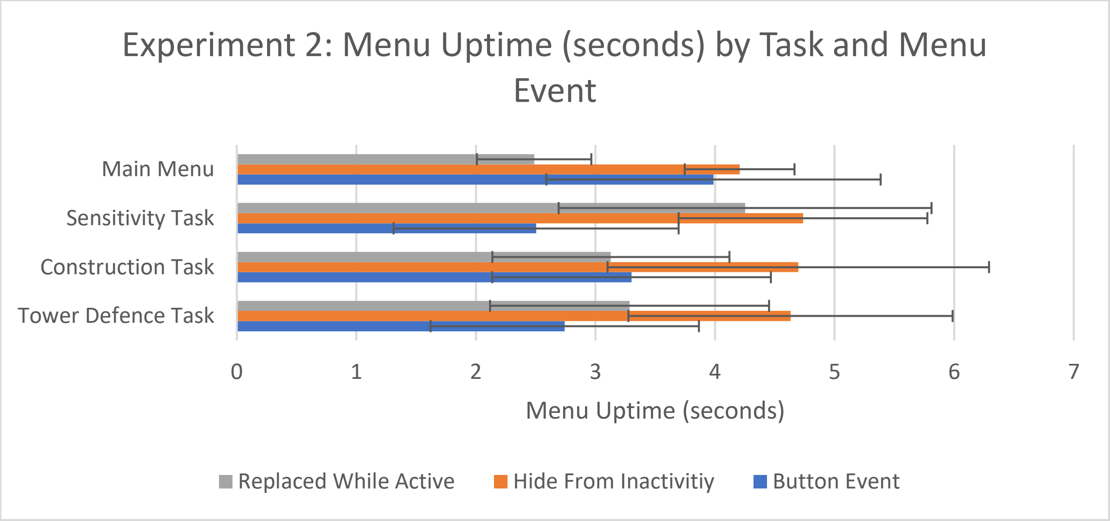


Figure 7.29: Experiment 2: Menu Uptime (seconds) by Task and Menu Event

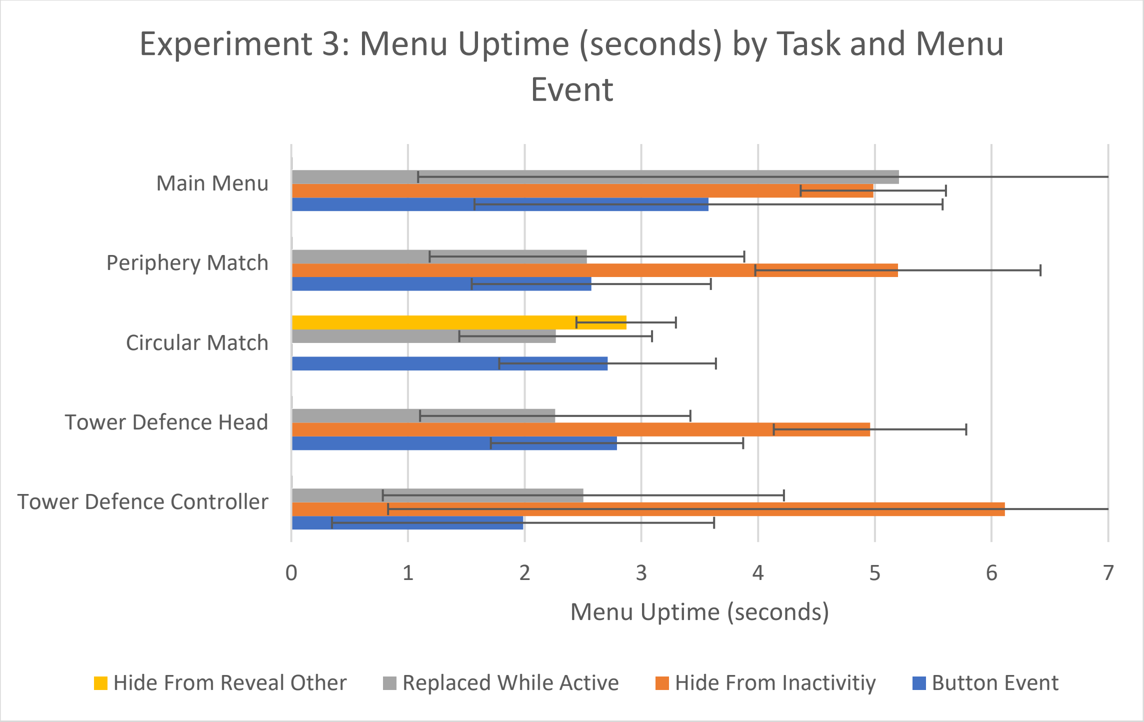


Figure 7.30: Experiment 3: Menu Uptime (seconds) by Task and Menu Event

Figure 7.31 shows the other related menu uptime information for just the time when the two-step widget was visible, showing how quickly participants interacted with the two-step widget. The wide variance shows some participants were interacting almost instantly with the button. Some took much longer to perform the action, where this could have been any length of time until a hide from inactivity occurred. In the PVMS, it was possible to stall the "Hide from Inactivity". The "Hide from Inactivity" for the two-step widget was always locked to be a maximum of 8 seconds.

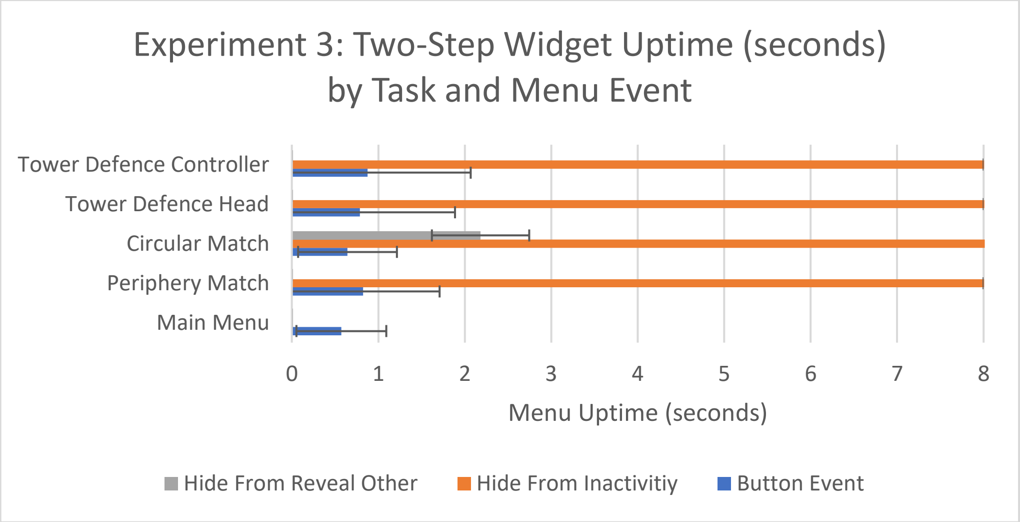


Figure 7.31: Experiment 3: Two-Step Menu Uptime (seconds) by Task and Menu Event

This section has broadly explored how participants interacted with the opened menu. It began by showing the occurrences of different actions performed by the participants to interact with the menus. The evaluation was then continued to show a simpler view of the data that represented the actions as averages for participants for each task. The evaluation related to actions showed how participants were using the menus but importantly demonstrated the impact of the two-step widget by showing how many times it could be considered a minimal impact. Then the timed and untimed situations were compared to show the difference in use with the impact from pressure to perform under time constraints. A weighted formula was used to compare the data to demonstrate that the number of perceived "wrong times" for the menu appearing was not as significant in the third experiment. Finally, the amount of time spent choosing menu options or letting the menu time out as part of using the PVMS was shown at the end. These have all demonstrated in their own way the positive attributes of the PVMS, highlighting that it is a system that was beneficial for use in the experiments.

7.3 Interaction Techniques and Device Preferences

This section considers some user preferences concerning HMDs, interaction techniques, AR vs VR, and device preferences.

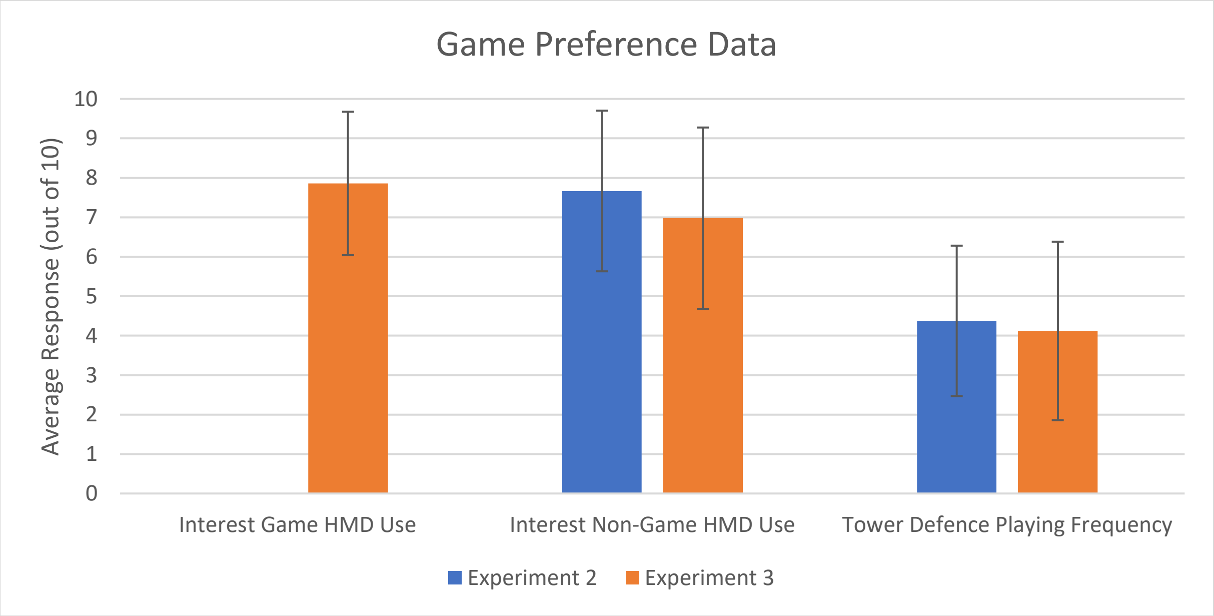


Figure 7.32: Game Preference Data

Figure 7.32 shows a similar average frequency associated with playing tower defence games between the participants in experiments two and three. This difference was not statistically significant (F(1,48) = 1.25, p = 0.27). From the third experiment by observation, participants felt they were more interested in using HMDs for game-related activities on average. The score for non-game HMD use during the second experiment was higher than the third experiment. The second experiment's higher preference for non-game use may have been influenced by the lack of an associated question directly about game use. The combination of questions may have led participants to consider the comparison between the two types of activities more thoughtfully. The interest in both cases was still indicated as high with averages of 7.67 (SD = 2.04) in experiment two and 6.98 (SD = 2.3) in experiment three for interest in non-game HMD use. The difference was not statistically significant (F(1,48) = 0.18, p = 0.67).

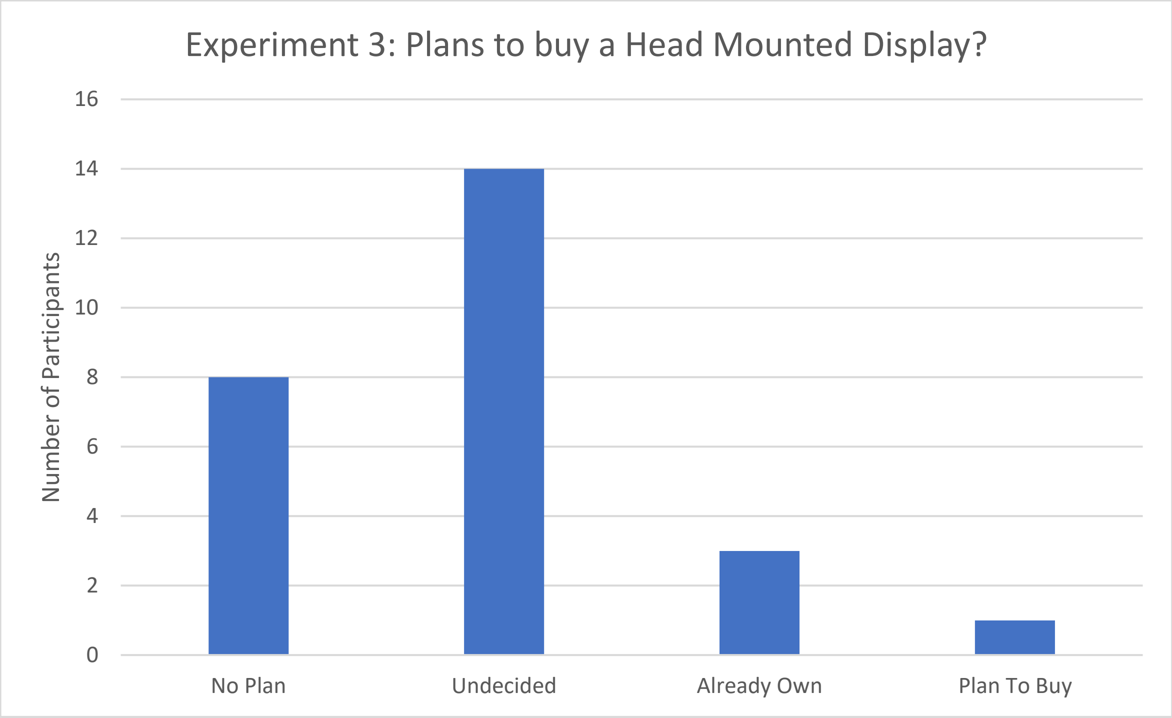


Figure 7.33: Experiment 3: Plans to buy a Head-Mounted Display?

As part of the data captured during the third experiment, the participant’s desire to become a personal user of HMDs was evaluated. Figure 7.33 shows how participants responded to this question. Participants overwhelmingly indicated a desire to wait and see if the price would drop or if some application they desired became available on a device. When the experiment was conducted, HMDs in the market were comparable to buying a video game console with a lesser quantity of possible applications. Gradual technology improvements will continue to make the devices more affordable, as has been the case with many other forms of technology. This type of development takes time, and much work is being done now has been laying the groundwork for future iterations in recent years. The evolution of VR with HMDs is evident with the low-cost but high performance of recent headsets like the Oculus Quest 2.

In each experiment, there were some questions related to preferences toward input devices. The first experiment has a broad inconsistent spread of data for each type of input preference. Most of this can be referred to in the first experiment chapter (Chapter 3). One of the better representations from the first experiment regarding input preferences came from the before and after preferred inputs. Participants were asked to select their preferred input device for interacting with the system before and after the experiment, as shown in Figure 7.34. Mouse had been overwhelmingly indicated as the preferred device for input before the first experiment was completed. Once participants had completed the experiment and had been exposed to the different interaction techniques, this preference was dramatically decreased. As seen in Figure 7.34, after the experiment, participants suggested that the HMD Only and mobile would have been suitable interaction devices compared to the mouse.

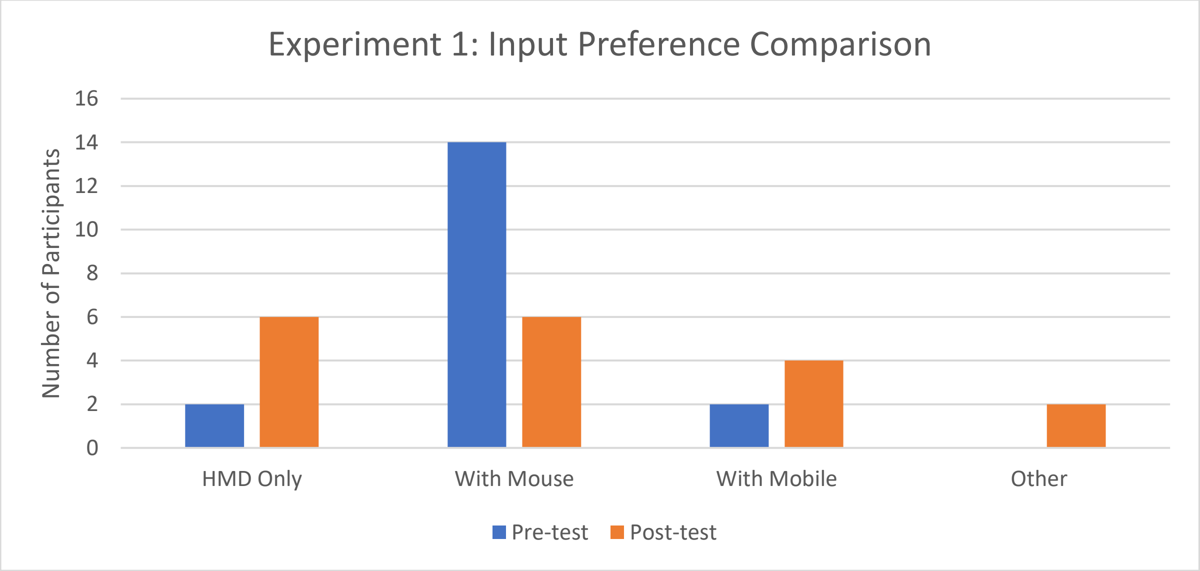


Figure 7.34: Experiment 1: Input Preference Comparison

In the second experiment, pre and post questions were posed again, as seen in Figure 7.35 (over). For this experiment, participants were asked to rank their preferences on different devices for input, where a lower number was a more preferred device (The ranking was on a 1 to 4 scale, where 1 was the most desired, and 4 was the least desired). During this particular experiment, participants were provided with just the HMD Only input device type, so there were not expected to be many dramatic shifts in preferences. The options for providing gestures with hand interaction were, on average, preferred in the combination of LEAP Motion or Microsoft Kinect sensors. This style of hand tracking is an accepted feature in current HMDs (for example, Oculus Quest and Hololens) and reinforces the user views from these experiments. The mouse and mobile type options were included here as well. Before the experiment, participants indicated the HMD Only type interaction was their least preferred. After the experiment was completed and the participants had been exposed to the interaction functionality, this shifted to participants ranking HMD Only interactions in second place. The shift toward HMD Only indicates participants felt their experience demonstrated enough viability as an interaction technique to change their preference order.

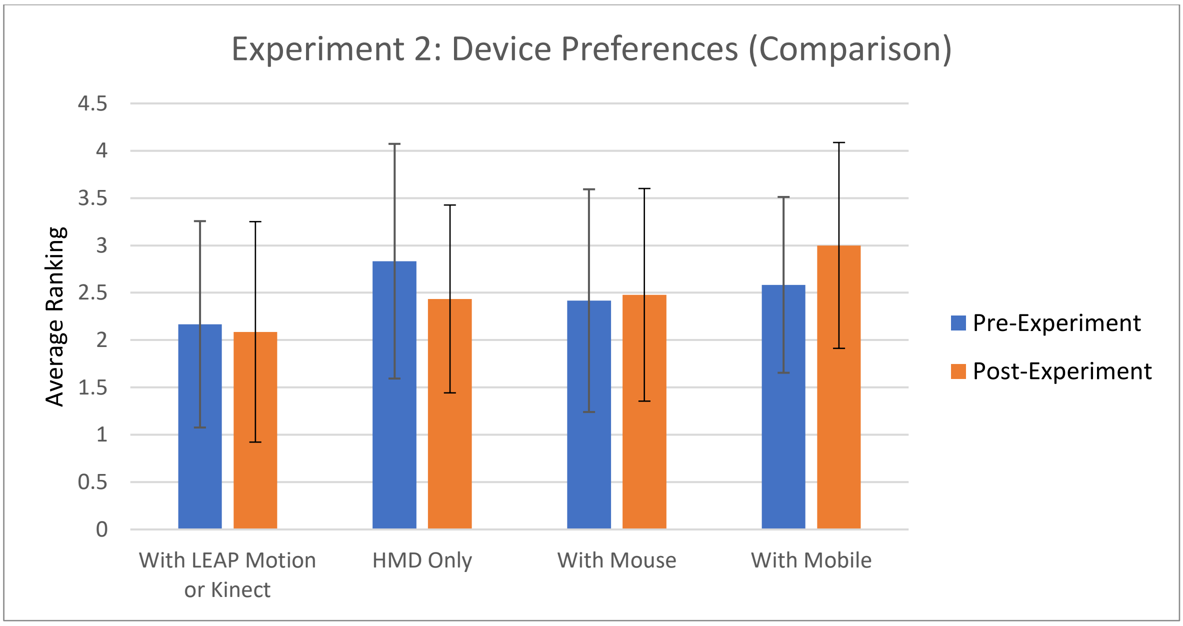


Figure 7.35: Experiment 2: Device Preferences (Comparison)

The third experiment looked less broadly when it came to device preferences. The second experiment demonstrated the viability of a HMD Only interaction mechanic; therefore, the third experiment sought to investigate the potential for input navigation systems that supported the HMD Only interaction. Thus, the third experiment had a focus on the participants' preference toward the use of PVMS. Participants were asked to compare their preferences using HMD Only interactions against adding a controller with tap-to-select. As shown in Figure 7.36, participants preferred to use the instant selection with tap-to-select. With average responses of 4.95 (SD = 2.22) for Periphery and 8.85 (SD = 1.98) when the controller was included. These results show participants felt the menu was useful, but the selection time was a big factor in how the participants scored their preferences. The selection time was intentionally longer for the experiments to reduce errors from novice users.

For this reason, it was expected the controller would come out ahead. Responses to this question confirmed the expectation and, when contrasted against the first experiment's responses, reiterates that participants prefer faster selection where possible. It is important to consider the way these menus are designed. The menus will work for both input types (HMD Only and any selection device), allowing users to interact with either type of selection, dependent on availability or the kind of content being interacted with using an application. As discussed in section 7.2.1, the Circular is statistically different to Periphery and Periphery statistically different to Periphery with Controller.

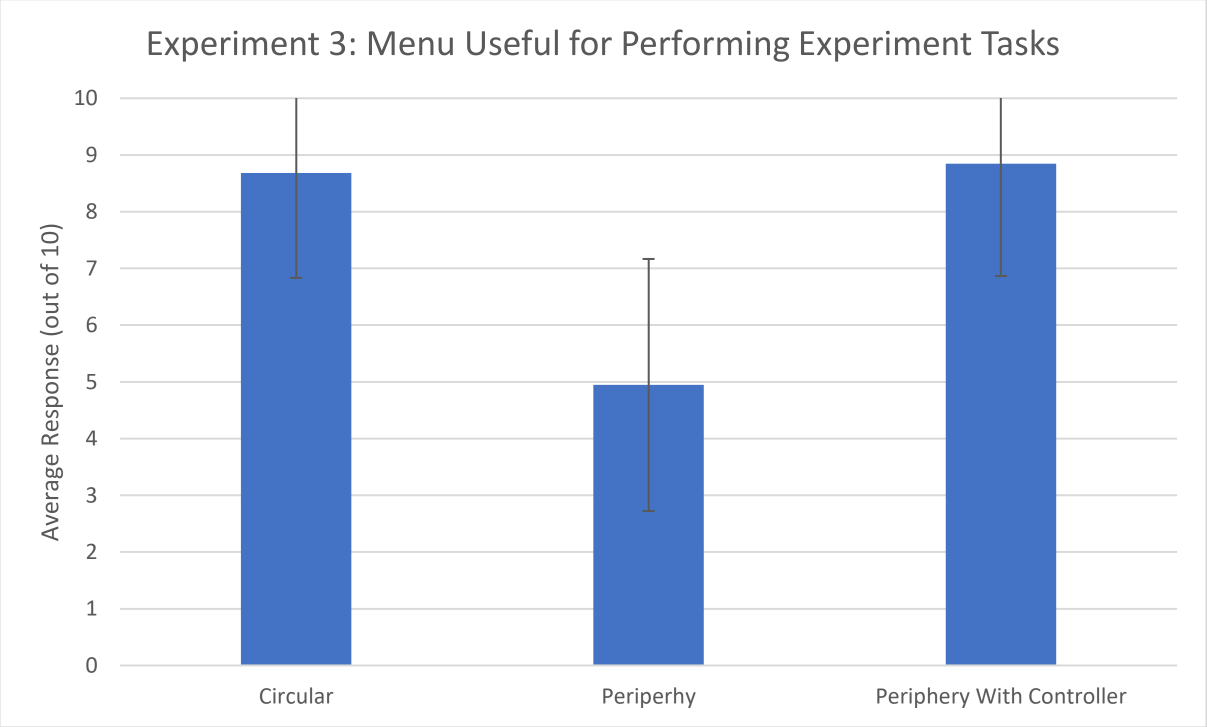


Figure 7.36: Experiment 3: Menu Useful for Performing Experiment Tasks

While answering the question for Figure 7.37 (over), some participants queried the difference between VR and AR to clarify. Nothing in the experiment directly showed the differences between these two. As VR was the tested experience during the experiment, there was likely to be a higher preference toward VR. This data could be contrasted in future when testing against use on an AR experience.

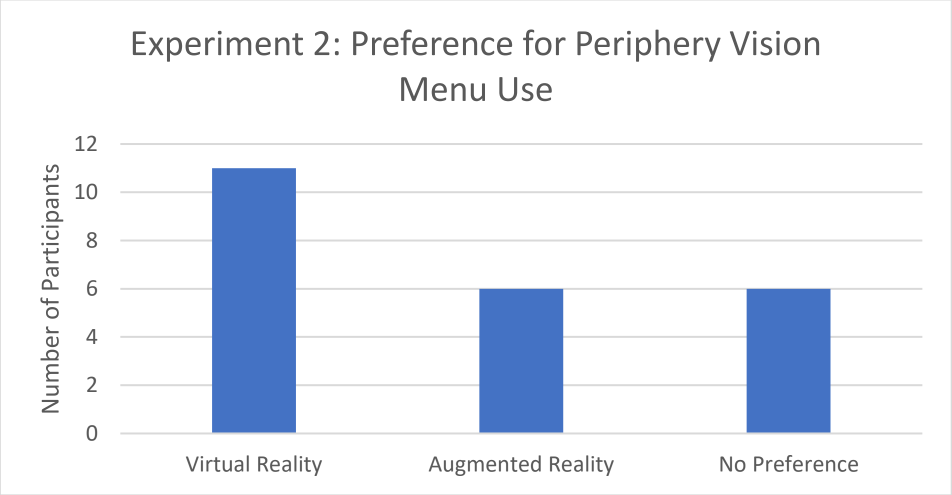


Figure 7.37: Experiment 2: Preference for Periphery Vision Menu Use

7.4 Interface Preferences

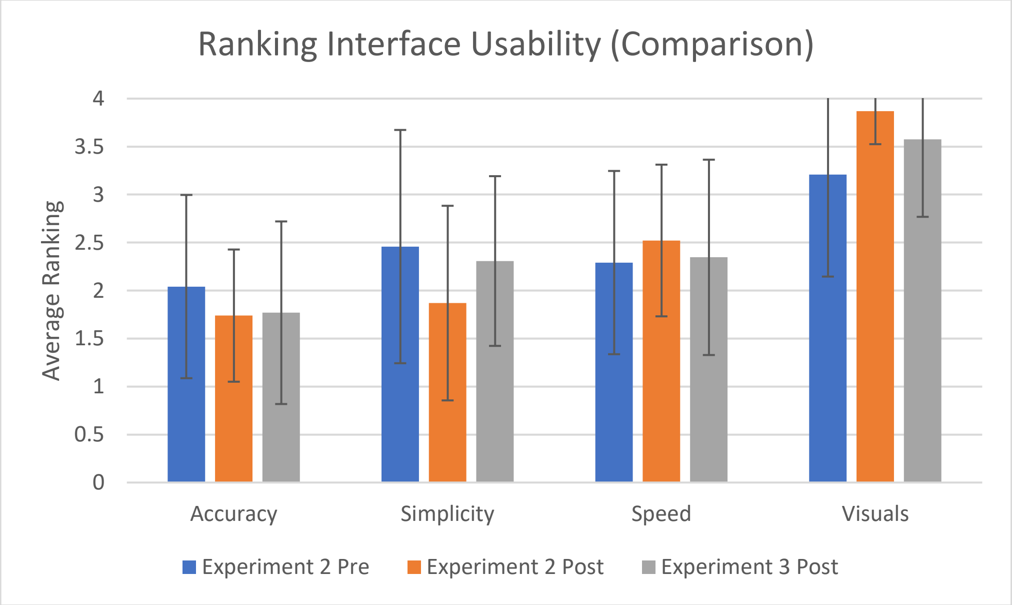


Figure 7.38: Ranking Interface Usability (Comparison)

During the second and third experiments, participants were asked to rank their preferences regarding the attributes of interface usability, specifically concerning HMDs. The rankings were asked both in the pre and post-experiment for experiment two and post-experiment for experiment three. The purpose of including this in the third experiment was to re-validate the findings from the second experiment. Lower scores indicate a higher ranking (participants ranked on a scale of 1 to 4, with 1 as the most important feature and 4 as the least important). As seen in Figure 7.38, the accuracy of content shown to the user was considered the most desirable attribute of usability across all three surveys, with more participants preferring accuracy in the post experiments. The other constant between the responses was visuals being the least important factor for usability. The ranking of visuals in this way indicates participants strongly felt that the functionality features were far more important to the experience of working with HMDs than how visually appealing they were. Simplicity and speed, when considering all three sets of responses, were mixed with reasonably close average rankings. Participants in the second experiment swapped their preferences between simplicity and speed in the pre and post-experiment analysis. With simplicity moving to a higher average rank from 2.45 (SD = 1.22) in the pre-experiment to 1.87 (SD = 1.01) in the post-experiment. And the speed rank changed from 2.29 (SD = 0.95) in the pre-experiment to 2.52 (SD = 0.79) in the post-experiment. While in the third experiment, participants considered Simplicity and Speed to be almost equal in rankings, with Simplicity preferred at 2.31 (SD = 1.22) and Speed just behind at 2.35 (SD = 0.95).

For the PVMS, each of these interface usability attributes was considered as part of the design. Accuracy is demonstrated through the consistent activation of the interface. In the second experiment, this was not always optimal due to the higher impact of the interface appearing at unintended times. With a similar configuration in experiment three, the interface did still activate unintentionally. The handling of unintentional triggers was improved by using the two-step widget to create a smaller visual cue. Simplicity was a significant goal of the interface design. For the purpose of both visual and interactive simplicity, the interface achieved these goals. With a simple gesture to reveal, followed by four menu options (and an exit button), the interface allowed the user to choose their required action quickly. Interfaces could be made as simple or complex as necessary to suit an application. The speed of interaction was an important consideration across the experiments. Due to the nature of participants still learning to use the interface, the delay for hover-to-select interaction was longer than a typical application may use. The experiments demonstrated variation between using instant selection with the controller in experiment three and hover-to-select in all three experiments. The visuals were rated the least important but are still necessary to consider as well. In the case of interfaces used for testing, they were not polished consumer interfaces. They all had a simplistic feel to them. There is room for developers choosing to use the PVMS interaction technique to make their interfaces fit the design aesthetics of their application. Many games have a style used across all their in-game menus. The PVMS does not restrict this in any way significant, allowing freedom for artistic creation for games or a formal approach for other types of application.

7.5 Effectiveness of the Experiment Tasks

Looking retrospectively at each of the experiment tasks, participants reacted positively to the content of each task. The following lists will briefly identify some of the observations from participants completing the experiments. It was overall more difficult to view how participants were interacting during the first experiment as the only video output was to the HMD. The other two experiments output video to the computer monitor allowed observation quickly if participants had any questions.

Experiment One

- Task 1 to 4: These tasks were designed to gradually increase the number of selections required to complete each task. Participants expectedly found these trivial. It was observed that some participants, when it came to the fourth task, took a moment to realise there were two separate, additional panels with more blocks to select after they had completed with the central panel.
- Task 5: The task was designed to cause frustration. Most participants were able to move through the maze relatively quickly. Only a few participants had more difficulty in avoiding blocks. Normally this was due to attempting faster movement when slowing down, and being careful was a more reliable way to complete the task.
- Task 6: In this task, there was only one cube to configure using an early version of the circular menu properly. Once participants had explored the functionality of the menu the first time, the task was quickly completed on subsequent devices.
- Task 7: Some participants completed this task with no problems. Others were able to complete the task, but it took longer due to misunderstanding matching of objects was not fluid (specific objects needed to be matched to specific locations). The aspect causing misunderstanding was improved when it came to the third experiment with not requiring specific pairing. At the time, it was easier to code the cubes to check against only a single case instead of matching to any. If this experiment were rerun, it would be preferable to make it more like the version in the third experiment. The other issue participants had during this task was with the HMD itself. The issue was

never identified, but participants who did not complete the task at all were typically due to auto-rotation, i.e., where the HMD thought the user was continually turning. The auto-rotation made the completion of tasks almost impossible when it did occur. There was no code written as part of the application that would try to update the position manually, and it seemed to be an API or hardware malfunction.

Overall, the tasks from the first experiment were well received and effectively demonstrated basic interactions within a HMD, providing a comparison between the different interaction techniques. For future versions of testing this experiment, it would have been more appropriate to randomise the order of interaction devices.

Experiment Two

- Task 1: The sensitivity calibration task provided a solid basis for introducing participants to how much interaction was needed to show or not show the menu. The difference between low, medium and high sensitivity configurations could have been more dramatic, with greater distinctions between each classification. Some participants realised faster than others that it was not necessary to interact with a revealed menu. The purpose of the testing was not to interact or select from the menu options; the task specifically focused on the action of revealing the menu. Overall, the task was successful.
- Task 2: Tower construction was designed to be a simple user-driven task, demonstrating a repeating series of actions that provided an illusion of choice as an untimed activity. Participants typically thought more about their choices on the first tower and then formed opinions faster about their choices for the following three towers. The task was functional and provided the expected experience. The main difference that could be accommodated for possible future iterations of the testing would be providing a better visual experience. Originally the stack of blocks was intended to be a tower that would form as the user placed their options. Given development time and minimal access to model development, leaving the visuals as larger blocks at the cost of improved illusion was decided.
- Task 3: The tower defence game in this experiment was mostly well-received. Participants successfully completed the game with little or no damage taken to their

end base. The minimal damage to the end base indicated the task was a little too easy and led to increased difficulty as part of experiment three. The timed waves of enemies created time pressure to perform activities necessary with building and repairing towers to keep up with the game state.

The second experiment established that the PVMS was a viable form of providing interactive menus as a prototype through these tasks. Each task looked distinctly at different attributes of the PVMS to test how the prototype worked under different conditions.

Experiment Three

- Task 1: For most participants, while completing the matching puzzle using the PVMS, this was the first time they had used the menu. After completing the first few actions, most participants appeared to become quickly used to the type of interaction. The tasks seemed to be performed far more effectively from an ease-of-use perspective compared to the last task of experiment one.
- Task 2: This task provided an alternate interaction experience using the PVMS for object creation alongside a circular menu, as seen in task 6 of the first experiment. The task was well-received and demonstrated an example of the menus co-existing. Participants responded positively to this type of menu and the interactions between the different menus.
- Task 3: The updated tower defence game subtly increased the difficulty over the second experiment through longer enemy wave phases and the number of enemies that would be dealt with simultaneously. Participants mostly made intelligent decisions when it came to tower selection leading to success.
- Task 4: The difficulty was further increased compared to the previous task by increasing the health of all enemies by 10%. This change offset prior experience with the expectation of better choices the second time around. With the addition of faster reaction times using a controller for instant selection, participants responded very positively to this task and preferred the way this task felt compared to task 3.

Over the four tasks in experiment three, further testing the modified PVMS prototype gave multiple experiences with untimed and timed situations. The tasks provided interesting but simple tasks to reduce any difficulty for novice users, leading to an overall effective experiment.

7.6 Game Trailer for Recruitment

It was expected that the inclusion of promotional material (recruitment YouTube video) for the system would result in a greater number of participants. For a full discussion about the YouTube data, see the relevant sections in sections 5.2 and 6.2 on experiments two and three. Despite improved viewing statistics for the third experiment trailer, participants indicated lower influence on the desire to participate and influence on future participation, as shown in Figure 7.39. A limitation of the survey regarding gauging feedback on the game trailers was that participants had not necessarily watched the trailer before attending. This limitation does mean that some participants may have answered more generally on how they felt it would influence instead of how it did cause them to be influenced. To mitigate this, it would be useful as part of the experiment design to show the trailer before the pre-experiment questionnaire to ensure it has been viewed with the choice to opt-out of viewing if they had already seen it. Comparing between the two experiments on the independent variables found no statistically significant difference for either of influence desire (F(1,46) = 0.31, p = 0.58) or influence future desire (F(1,46) = 1.08, p = 0.3). When comparing between values for each experiment internally though both are seen to be statistically significant when comparing influence desire and influence future desire in experiment two (F(1,46) = 11.93, p = < 0.01) and experiment three (F(1,50) = 6.35, p = 0.02).

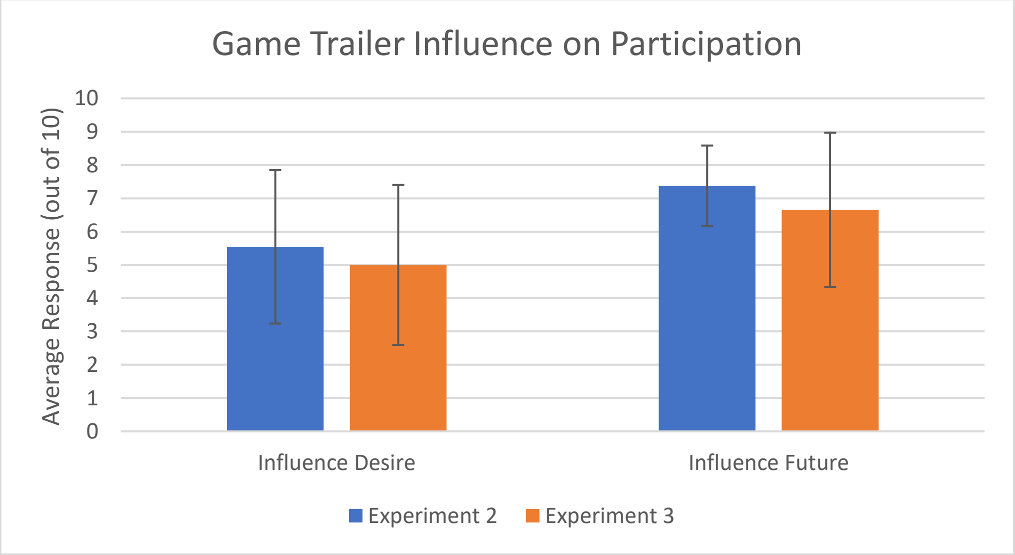


Figure 7.39: Game Trailer Influence on Participation

From this data, we can conclude that trailers to influence participation in research is beneficial and worth exploring for use where possible. As something that can help drive interest in research participation, the game trailers could bring in participants that may not have been engaged if they had been exposed to traditional research advertising.

7.7 Conclusion

Many positive points indicate the success of the PVMS from reflecting on the experiments conducted and the results generated. The results from the System Usability Scale indicated participants reacted positively with scores of 82.17 (SD = 11.26) in the second experiment and 76.41 (SD = 14.75) in the third. Participants during the third experiment showed a higher proportion of correct interactions. Participants reported the number of perceived wrong times for the menu triggering with a similar average as experiment two. The discussion in section 7.2.7 showed the interaction was mitigated by the implied lesser impact of the two-step widget. The close button saw a significant amount of use where participants could then open the menu to decide or check on making choices in the menu. With the data captured during the use of the menu system, the number of occurrences of wrong actions could be dropped by modifying the sensitivity. Section 7.2.6 investigated this through the data for triggered menus and suggested that the values around 13 for **`Diff_X`** and 19 for **`Diff_Z`** would be more appropriate. The values used for configuration within any individual application may need to be different based on the types of activities, as was shown by the variance during different types of tasks across the experiments. With 20 of 23 participants during the second experiment saying they found the PVMS useful, it provided a good indicator for future development of this system.

The next chapter will discuss the research questions, provide a framework for developers to use the PVMS and generally discuss the outcomes of this research.

8 Discussion

This chapter presents a discussion on the PVMS and its benefits for future researchers, developers, and users. Section 8.1 provides a short discussion about each of the research questions. Section 8.2 considers the work conducted by others to evaluate how the PVMS compares as a tool. The topics of comparison include papers that have cited the research published from this dissertation, other papers with similar overlaps, menus available via the Unity store for VR, and commercial games. Finally, section 8.3 presents a framework for implementing the PVMS with topics to consider in extension to the content of other chapters.

8.1 Research Question Discussion

The research questions have previously been discussed at each experiment's end of each chapter. This chapter will review the research questions in the context of all the experiments to prepare for conclusions in the next chapter. The questions are repeated for context and followed by a short overview summarising the combination of experiments.

RQ 1: How can head rotations, captured through the orientation sensors of a HMD, be used to increase the user interaction capabilities with virtual worlds?

Across the three experiments, they have each demonstrated, in varying ways, the use of the head to provide useful experiences. The interactions tested focused on looking, tracking the gaze projected from the HMD’s orientation and the iterations of the PVMS alongside the variations of fixed in place menus and others such as the circular menu implementation. The examples of menus and interactions were all received positively by the participants. The PVMS provides an experience that is customisable to the user’s context. The data collected during the experiments demonstrated that the PVMS could provide a useful interface. Therefore, the PVMS and other examples (such as the static interfaces for introductions/feedback and circular menus) of interaction shown in this research can be considered for integration when developing any new VR system. Particularly where there is a desire to focus on HMD Only input and as an extension of other available functions that can be completed with the aid of other input devices.

RQ 2: Does using a head-mounted display with head-only interaction for menu navigation provide a similarly useful experience compared to integration with a tool for instant selection?

This research question was primarily investigated as part of the first and third experiments. The first experiment presented a preliminary simple interaction technique focusing on the hover-to-select in an environment with principally fixed in place visual elements. The first experiment demonstrated the viability of using this technique and led to prototyping the PVMS as an interaction technique. The third experiment iterated on the user experience, taking what was learnt from the first and second experiments to present an interface that harnesses the head as a gesture tool in a useful and simple way. When participants compared the use of HMD Only interactions to those where an instant selection tool was available, they did prefer to have the option of using the instant selection tool. The time delay during hover-to-select was set to a longer duration for understanding the process than would be used for an average user in a published application. The time for selection was the primary complaint, and this could be reduced to suit a user's preferred delay. The trade-off demonstrated of having the ability to control the PVMS with HMD Only interaction allows it to be useable in either scenario, making it a versatile interface.

RQ 3: Can a hidden menu provide a viable and useful tool for interaction with head movement as the mechanism for revealing it?

The second and third experiments demonstrated the capacity and capability of PVMS as a prototype for a hidden menu system. The PVMS as a hidden menu system, revealed by using head movement gestures, tracked with the orientation sensor data, was shown to be both viable and useful. The second experiment presented this as a menu that would appear with the options visible right away. When accidentally revealed through the head's normal movement, the impact was higher based on the extent of screen space consumed. The third experiment remedied this by adding the two-step widget and demonstrated the effects of minimisation of impact as discussed in section 7.2.7. The data showed that the weighting of how much accidental menu activations impacted the participants was alleviated, therefore, demonstrating validation of the improvement to the system.

RQ 4: Does a hidden menu revealed and interacted with using the head provide a useful experience compared to menus appearing at a fixed place in a virtual world?

The role of the PVMS in providing a hidden menu has been shown as viable in its own right. The PVMS from demonstrated examples through the experiments does not need to be implemented as the only user interaction technique. Targeting interfaces to the audience of the system and intended use cases is important. Not every application will need hidden menus, but where appropriate, the ability to reduce wasted space by concealing menus using the PVMS can be a viable answer. The clear benefit of the hidden menus as part of the PVMS instead of menus fixed in place as part of a VR world is that the user can control the appearance at will. Typical fixed in place menus as part of a VR world may require the user to move to the menu themselves or only have them available within a constrained space. The PVMS can bring the menus to the user, providing them with a useful experience based on their needs.

The PVMS was directly compared against the circular type menu for this research question. The questionnaire responses for circular with a mean of 8.68 against PVMS with a mean of 4.95 was statistically significant t(25)=5.52, p = <0.01. This significance was further demonstrated from the in-application questions showing for the object matching tasks, the Match Periphery (mean of 3.62 out of 5, SD = 0.98) compared to Match Circular (mean of 4.5, SD = 0.91) was statistically significant t(25)=-7.3, p = <0.01. The circular menu presented as a fixed in place menu appearing at the point of interest demonstrated that where appropriate, menu creation is preferable near the point of interest. As an experience for HMD Only type input, combining both types of menus would be recommended.

Some additional discussions of the research questions are presented in the next chapter as part of the conclusions. The remainder of this chapter compares the PVMS against other work and discusses the framework for developing with the PVMS.

8.2 Comparison Against Other Work

The purpose of this section is to evaluate associated work that has arisen with time since the PVMS was first designed, implemented, and evaluated. Experiments two and three were conducted during 2016, while the first iterations of the Oculus Rift and HTC Vive established a starting point for a new world of immersive VR. A continued iteration of hardware and software solutions aid in the design and development of new enriching capabilities for end-users and research. In this section, a selection of papers, commercial menu solutions available through the Unity store, and examples from games are described and contrasted against the PVMS.

Comparison Against Papers Citing this Research

Three papers have referenced the initial publication of the PVMS as presented for the second experiment at the 28th Australian Conference on Human-Computer Interaction (OzCHI2016) (Mitchell and Wilkinson, 2016). It is significant to evaluate how common themes exist between the more recent innovations and how they compare as solutions to the PVMS. The papers have common themes in their evaluation of menu techniques. The first two identify specific menu techniques and the third looks at more general menu interaction.

The first paper presented an AR solution titled HoloBar by Saidi et al. (2021) that combined the Microsoft HoloLens with a mobile device as a menu solution. The paper referenced the PVMS by describing the activation of menus using the head direction with quick head movements to create them at predetermined positions. The authors considered the existing identified collection of referenced 3D menu activation gestures to break the interaction flow. While the gesture for activation of the PVMS may be regarded as a deviation from the interaction flow, the disruption can be considered a minor trade-off. The PVMS can be used for discrete actions or kept present with transformation into alternate options for multi-level menus (as was used for object manipulation in the experiments). Therefore, the interaction flow of the PVMS does not significantly suffer. The menu solution presented as HoloBar in the paper worked by having a field of view based menu that combined specific positioning of the mobile device to correlate with choosing a menu option. When the mobile was within a specific activation zone, the smartphone display would update with a second-level of menu options. For example, if the user were to select a rotation option by aligning the phone with the "Rotate" option appearing at the bottom of their field of view, the phone would update with rotation options such as "Rotate Right 45^{o}". The HoloBar was found to take around 2.4 seconds to complete actions with up to 80 menu options available (10 top-level and 8 second-level on smartphone). From their experiments, the HoloBar was quicker than the alternatives they tested against (Air-Tap and Clicker). The menu system appears suitable for some use cases. The requirement to hold a mobile device at a specific location could become tedious with requirements for both head position and smartphone hand positioning compared to the PVMS's option for HMD Only interaction and tap-to-select with no positioning requirements. The HoloBar is specifically an AR-type solution because it requires vision through the transparent display to observe the secondary menu options on the smartphone. The PVMS demonstrates an advantage in its design compared to the HoloBar, allowing development with VR as was prototyped and AR.

The second paper presented by Iacoviello and Zappia (2020) used the Microsoft HoloLens HMD for a tourism application titled HoloCities. The discussion presented in the paper cites the PVMS incorrectly as a menu designed specifically for AR. Although the PVMS's applications extend to AR, it was principally demonstrated throughout this dissertation and described in the cited paper as a prototype through VR. Despite the misrepresentation of the PVMS as a source, the HoloCities application does overlap some of the considerations for use cases that were presented as part of this research. In sections 2.3.4 and 4.2.2, the tourism domain was considered and directly suggested that the PVMS would fit as a solution to menu interaction. The primary interface provided in the HoloCities application maps a semicircle menu with options to the floor. The menu was intended to follow the user while never unintentionally occupying a user's field of view. The application was targeted at two scenarios where the first had a guide identifying features to talk about, and the second was the navigation of a city without a guide. The conclusions did not quantify the effectiveness of the menu system but did discuss the enhancing of user experience from 3D manipulation with zoom, annotation, and collaboration. Compared to the PVMS, the HoloCities menu appearing on the floor provides a different experience. The menu system from the presented content does not provide contextual menu options depending on the situation and instead provides a consistent set of options. The other issue evident from observation of the work is the requirement of the user to look down would typically mean any menuing would be presented away from the point of interest. These features suggest that the PVMS has clear advantages of providing contextual menus and, more importantly, the ability to place the menus closer to points of interest. The HoloCities menu system seems suitable for the specific use case they describe and could be used as an additional menu option alongside the PVMS.

The third and final paper citing the PVMS to date was by Tu et al. (2019), which considered the human performance of ray-casting crossing for object selection in VR. The citation use, in this case, was very brief, with the PVMS presented as an example of a 2D plane type menu in 3D space. Separate from the connection by the paper to work on the PVMS as a 2D menu, the PVMS could theoretically be presented as a more complex 3D menu if it were to be required. The gesture action could be used to reveal any menu provided the menu was positioned based on similar offset rules as described in section 4.6. The paper explored using ray-casting to cross and point at 2D targets in 3D space with an Oculus Touch controller and compared it with a second experiment with 1D goals on a 2D plane. Four general design guidelines were presented as recommendations for crossing interface design and four practical design suggestions. The specifics for most of the guidelines are not relevant for comparing against the PVMS. One of the recommendations was to make surfaces crossing-friendly with 2D surfaces or 1D bars for improving interaction. The design of the PVMS would support this interaction paradigm as a 2D interaction surface. The primary use case with HMD Only type input would not benefit from the interaction described by the paper. With the addition of controllers, the PVMS could be moved or interacted in similar ways discussed in the paper with crossing-based interactions.

These three papers citing the PVMS have provided interesting comparisons to see how others have mentioned the work and the overlaps in menu approaches. The following section presents a comparison against examples of other research.

Comparison Against Other Research

This section considers recent research conducted by other researchers after the experiments for this dissertation. These provide three additional comparisons for discussion beyond the three papers that had cited the PVMS as a reference.

Wang et al. (2021) investigated fixed menus against handheld menus. The menu presented a grid of 4x4 3D shapes with equal numbers of four colours and four shapes to make each option different. The research compared HMD interaction between fixed and handheld menu variations and tested three interaction methods. The interactions were hand aiming with button press confirmation (referred to as Hand-BP), head gaze selection with button press confirmation (Head-BP) and head gaze selection with dwell time confirmation (Head-DW). The experiments used a dwell time of 780ms. A wheel visible in the VR background showed random interaction targets the participant had to select. Participants experienced all six variations of menu and interaction combinations and provided feedback via a questionnaire. The fixed in place menus were faster with a statistically significant difference from comparing selection time with all interaction methods. Head-DW was slightly faster for all comparisons with a single menu type after removing dwell time. Head-DW was rated higher than Hand-BP for ease of use. Hand-BP was rated higher for learnability and accuracy. Efficiency was roughly similar. The results showed that fixed menus were better than handheld menus overall. Considering the significance of this for the PVMS, it suggests that the PVMS's design is supported. The PVMS exists as a menu that combines the best of both fixed and dynamic menus by allowing the user to create the menu fixed in place at any desired location. The research supported similar findings for the impacts of using a button press compared to dwell time.

Pfeuffer et al. (2020) investigated five different interaction techniques with gaze-enhanced menus. Using a HTC Vive as the HMD, the researchers evaluated dunk brush, pointer, dwell time, gaze button, and cursor techniques. The evaluation task involved selecting a colour from a menu and drawing a line between two spheres. The dunk brush technique involved moving the pointer to touch the point on the menu for selection. The pointer technique was completed by pointing with a controller at the option and using a button. Dwell time used gaze tracking of the HMD with a 1 second selection time. The gaze button technique was similar to dwell time, except it used a button press to confirm the selection. The cursor technique used the controller to create a similar experience to a trackpad with button selection. Dwell time was noted by participants to feel slow, but it performed second fastest after the dunk brush technique. Dwell time and gaze button techniques had low coordination requirements compared to the others. The dunk brush technique had limitations because it required the most movement and relied on interactive objects being in range. The dwell time interactions were easier to learn but perceived as more eye tiring than hand alternatives. Lower performance was found with the gaze button technique, and the researchers suggested that more studies needed to be conducted on this issue. For the PVMS, these findings support the HMD Only interactions with dwell time as easy to learn. Developers could use each of the techniques presented for interaction with the PVMS. The dunk brush as a technique could require changing the offset distance of the menu to appear closer for interaction due to how the user must directly touch the options.

Safikhani et al. (2020) investigated UI for VR and identified guidelines for 3D UI based on surveying some of the higher-rated games on the Steam Store platform. The guidelines identified by the researchers are summarised below.

- Fit the environment.
- Intuitive and understandable based on interactions from daily life.
- Interactive objects should have similar interactions to others in the environment.
- Accessible anywhere or at a specific position for each level/scene.
- Error handling should be handled with two-step confirmation when it has serious consequences.
- UI should encourage exploration.

In the research, they developed a menu and inventory system, quest manager, and level selection portal to become part of a "WelcomeRoom" designed to introduce users to VR. The guidelines can be considered for how they relate to the PVMS. Fitting the environment involves matching aesthetics, which will change for each application. The PVMS menu, as presented in the experiments, could have suitable different textures applied to modify its appearance to match a visual style to improve immersion. The interactions concerning HMD Only with the PVMS are similar to actions performed in daily life. The act of looking to the side to see something is a common action, and then dwelling on a button for selection overlaps with a typical observation of interesting objects. The PVMS excels at allowing menus to be accessible anywhere through its hidden till needed reveal technique. Error handling is not specific to the PVMS, but suitable popups could be provided with another confirmation step if a change would have serious consequences. The PVMS does not directly encourage exploration, but it gives freedom to explore with the safety of having a menu available wherever the user goes in a virtual world.

These three papers have considered overlaps in interaction types and design considerations with how they impact the PVMS. The following continues the discussion by comparing against assets sold to streamline VR interaction development.

Comparison Against VR Menus Available via the Unity Store

The examples in this section will briefly present a selection of VR menus marketed toward VR that appear on the Unity Store[36] to incorporate into applications. Unity provides some basic tutorials[37] introducing the development of world-space menus for VR as part of their tutorials from at least the 2019 version.

The first example, titled "VR 3D Menu – Concept UI Design"[38] was first released in July 2020, and its latest release was in June 2021. The menu provides two different styles of 3D menu designed to be attached to handheld controllers. The menus support 3D pagination, hover effects, 3D text and buttons, drag and drop from the menu and some other features for interacting with objects not directly related to the menu. The menu layout has 3D buttons at the bottom to change between menu categories with optional additional pagination on the sub-menus demonstrated by showing a collection of objects with arrows to show different objects. The 3D menu for choosing objects that could be dragged and dropped into a scene presents an interesting overlap with the PVMS and its use in the experiments for similar actions. Developers could optionally use a similar 3D view to extend the PVMS and combine the best parts of this menu system with the reveal mechanic used for the PVMS. These described uses could be done with HMD Only input to remove the need for controllers.

"VR Ready Menu Room"[39] was released in 2017 as a virtual room with three curved screens ready to display. The scene is designed to enable control through the three static world-space screens. The store page suggests that the menu layout can be used for general menu navigation or as a command area for games such as a real-time strategy game. The use of static world space menus does present an easy option for developers but restricts the ease of access for users as they move through the virtual world. A combination of fixed in place menus (if necessary) and the PVMS would allow for freedom to perform actions while on the move.

[36] Unity 2021, "Unity Asset Store", URL: https://assetstore.unity.com/, Last accessed 19/12/2021.

[37] Unity 2019, "Creating a VR Menu", URL: https://learn.unity.com/tutorial/creating-a-vr-menu-2019-2, Last accessed 19/12/2021.

[38] Epibyte 2021, "VR 3D Menu – Concept UI Design", URL: https://assetstore.unity.com/packages/tools/gui/vr-3d-menu-concept-ui-design-144993, Last accessed 19/12/2021.

[39] Connor Wilding 2017, "VR Ready Menu Room", URL: https://assetstore.unity.com/packages/3d/vr-ready-menu-room-89921, Last accessed 19/12/2021.

"Cardboard VR TouchLess Menu Trigger"[40] was first released in 2016 and updated in 2018. This asset demonstrates an interesting comparison for dwell to select type interactions. The asset has hexagonal buttons that change colour when gazed at by the camera. Then with the cursor hovering over the buttons, the cursor shows a loading bar that forms a circle around the cursor. When it completes, an action can be triggered appropriate to that button. The software is designed to work with google cardboard's API. It presents a suitable alternative example of the cursor changing visually to show progress compared to the colour change used in the experiments. Another type of interaction for buttons by the same developer is titled "VR Advanced Touchless Triggers for VR"[41], which lets users gaze at buttons with specific different trigger types. It was released in September 2016 with a further update in 2020 and was designed for the Google Cardboard. These trigger types include one triggered while gazing at the button by covering the device's camera, issuing a voice command, and a dwell operation. These operations present alternatives to a dwell operation that could be used for rapid selection with support from additional camera or voice input inputs.

The "Tasty Pie Menu – Radial Menu VR Ready"[42] was first released in September 2016 and most recently updated in 2020. Similarly to the circular menu used in the experiments, this menu is graphically more appealing with an animated circle based on where a user is hovering. The menu appears to focus on icon-based menu items instead of text for its base appearance. Not all menus are suitable for icon-based options. Still, this option may be useful for simple actions that can take either classic metaphors or contextual iconography and represent menus succinctly. For use as an alternative appearance to the PVMS, this may be useful in some cases. An extension to the menu with optional text for each element that always shows next to the icons or only when they are selected would fix the noted issue.

[40] VR Cardboard Buddies 2018, "Cardboard VR TouchLess Menu Trigger", URL: https://assetstore.unity.com/packages/tools/gui/cardboard-vr-touchless-menu-trigger-58897, Last accessed 19/12/2021.
[41] VR Cardboard Buddies 2020, "VR Advanced Touchless Triggers for VR", URL: https://assetstore.unity.com/packages/tools/input-management/vr-advanced-touchless-triggers-for-vr-70312, Last accessed 19/12/2021.
[42] XAMIN Software 2020, "Tasty Pie Menu – Radial Menu VR Ready", URL: https://assetstore.unity.com/packages/tools/gui/tasty-pie-menu-radial-menu-vr-ready-70483, Last accessed 19/12/2021.

The “Gear VR 3D Menu”[43] was released in October 2016. The menu presented in this asset allows rapid development of horizontal or vertical menus that can either be clamped to show the options once or loop through the menu options. Any prefab can be attached as a menu item with an associated name and ID. The menu includes support for events, including swiping up/down/left/right, among others. The menu is not very visually appealing and states it is intended to allow developers to add a menu for testing their applications quickly. As a concept, the menu is suitable, but its use case appears similar to a static world-space menu for some limited types of interaction.

The “Mobile VR Interaction Pack”[44] was released in 2017. It presents a collection of interactive tools that can use gaze interaction with dwell time or instant selection with a button press. The tools include interaction distance to ensure objects are only interactable when close enough to them. The interface shown supported static world-space menus with buttons, toggle controls, dropdowns. Other elements were also included with scroll bars but required holding the Gear VR button to move them. The cursor would only appear when a user looks at an interactable object. Object outlines could also be applied to clarify the interaction target for interactable objects. The asset is limited in functionality for menus beyond the basic use of static world-space menus. This asset's default dwell time was very long at 3 seconds compared to the PVMS but could be modified to lower times.

These examples demonstrated surveyed assets from the Unity Store and have shown additions that could be added to the PVMS. The PVMS was intentionally designed to be a simple menu in its revealed form for the experiments. That could be changed and enhanced based on the needs of an application by adding 3D menu items or changing the layout to a radial menu with icons as identified from the examples. The following section continues to the last area for comparison with looking at the PVMS and menus for VR against commercial games.

[43] uDrawR 2016, “Gear VR 3D Menu”, URL: https://assetstore.unity.com/packages/tools/gear-vr-3d-menu-72558, Last accessed 19/12/2021.

[44] Ryan Zehm 2017, “Mobile VR Interaction Pack”, URL: https://assetstore.unity.com/packages/tools/gui/mobile-vr-interaction-pack-82023, Last accessed 19/12/2021.

Comparison Against Commercial VR Games

Commercial games have been discussed as part of the background in section 2.3.9. The examples are each reiterated here briefly to discuss how the PVMS could help those games individually. Additional popular or significant examples of more recent VR games and utilities are also presented to compare the PVMS and other experiences offered during the experiments. Each example has a footnote linking to their associated Steam Store[45] page.

Beat Saber[46] was released in 2019, and as discussed previously in section 2.3.9, is a rhythm game involving slashing oncoming blocks using controllers along to the sound of a musical beat. The main menu surrounds the user's front with static world-space menus to provide interaction and information effectively. These static world-space menus are effective for the type of game and validate the use of similar static menus used for showing tutorials and ratings in the experiments. The PVMS could be used in this game as either a tool to provide additional menu options (similar to how the later discussed OVR Toolkit does) or used as a quick pause menu that also pauses the game when triggered.

The other games identified in the background chapter included Arizona Sunshine, Job Simulator, VRChat, Superhot, and Tilt Brush. Arizona Sunshine[47] was released in December of 2016 as a zombie shooter. It does not have menus that appear during use, with world navigation via an aimed teleport and ammunition attached to the player's chest. The game's main menu is controlled by selecting and inserting game cartridges and then selecting menu options via a static world-space menu. The PVMS could help add a menu for letting the player change options while they traverse the world. Job Simulator[48] was released in early 2016 with experiences in workplaces such as a kitchen, office, and garage. The game is focused on using items around the environment to perform tasks. The PVMS could be used in this game to modify options or call in deliveries of items to add to the experience. Superhot VR[49] was released in early 2016 and provided a unique combat experience with slow-motion as a

[45] Valve 2021, "Virtual Reality on Steam", URL: https://store.steampowered.com/vr/, Last accessed 19/12/2021.
[46] Beat Games 2019, "Beat Saber", URL: https://store.steampowered.com/app/620980/Beat_Saber/, Last accessed 19/12/2021.
[47] Vertigo Studios 2016, "Arizona Sunshine", URL: https://store.steampowered.com/app/342180/Arizona_Sunshine/, Last accessed 19/12/2021.
[48] Owlchemy Labs 2016, "Job Simulator", URL: https://store.steampowered.com/app/448280/Job_Simulator/, Last accessed 19/12/2021.
[49] SUPERHOT Team 2016, "SUPERHOT VR", URL: https://store.steampowered.com/app/617830/SUPERHOT_VR/, Last accessed 19/12/2021.

puzzle-shooter. The PVMS could be utilised as a part of the experience during the slowed time to provide users with options. VRChat[50] is a multiplayer social experience released in 2017 with many personal customisations for players' avatars and the virtual world. The PVMS would be ideal in this experience to quickly access options via a hidden contextual menu to create new content, navigate the world, alter their appearance or other functions. As a final previously referenced game experience, Tilt Brush[51] was released in 2016, as discussed in section 2.3.3. The application provides a VR 3D drawing experience with menus attached to the controllers. The PVMS could increase the number of options by allowing users to select menus to attach to their controller. Other well-known art programs such as Photoshop on desktop computers have many options. Through further iteration, an application like Tilt Brush could similarly expand features to add more control to the user. The remaining examples will identify other newer games and consider how they have implemented menus and how they could benefit from adding the PVMS.

The Elder Scrolls V: Skyrim VR[52] exists as another version of Skyrim in a long list of re-releases onto different platforms since its original release in 2011. The VR version released in 2018 gives the user control of the menus by attaching the menu in world-space to one handheld controller while the other is used to interact with menu options. Fallout 4 VR[53], released in 2017, is similar to Skyrim as a rerelease of a game that began as non-VR in 2015 produced by the same studio. Fallout 4 VR attaches its menus primarily to the controller along with a compass that can be observed by looking down (like the downward menu from Iacoviello and Zappia (2020)). The classic "Pip-boy" controller that is part of the Fallout franchise appeared as a menu attached to the left controller. The Pip-boy's use from reviews[54] was "rough" and cumbersome from necessitating a lot of holding up of arms to interact. Both these games would benefit from revealing their menus using the PVMS gesture to reduce the need for one

[50] VRChat Inc. 2017, "VRCHAT", URL: https://store.steampowered.com/app/438100/VRChat/, Last accessed 19/12/2021.

[51] Google 2016, "Tilt Brush", URL: https://store.steampowered.com/app/327140/Tilt_Brush/, Last accessed 19/12/2021.

[52] Bethesda Game Studios 2018, "The Elder Scrolls V: Skyrim VR", URL: https://store.steampowered.com/app/611670/The_Elder_Scrolls_V_Skyrim_VR/, Last accessed 19/12/2021.

[53] Bethesda Game Studios 2017, "Fallout 4 VR", URL: https://store.steampowered.com/app/611660/Fallout_4_VR/, Last accessed 19/12/2021.

[54] IGN 2017, "Fallout 4 VR Review", URL: https://www.youtube.com/watch?v=SsZv2mQaIlc, Last accessed 19/12/2021.

or both hands having to interact. If necessary, there could be an option to grab the menu and move it around.

Phasmophobia[55] is a horror ghost hunting game released in 2020. The game presents the main menu as an interactive whiteboard that allows the same interaction between VR and non-VR. As a simplification for development in both types of interaction using the same assets, static in world menus make development easier. The other primary type of menu used in Phasmophobia is the journal. The journal lets the user make notes on evidence they have found in the game (from a set list of options), view photos they have captured during the game, and information about the types of ghosts they may encounter. This type of menu could be presented with the PVMS as the user moves through the world. The presentation of the menu as a journal that the user holds may be better suited to being grasped with controllers for the immersion. The PVMS gesture could make the journal appear then have the user grab it out of the air and interact with the hover-to-select type interactions.

OVR Toolkit[56], released in 2019, is an example of an application available through Steam that does not exist as a game. This utility has interesting overlaps with the PVMS in the toolkit's purpose. The toolkit provides access to desktop windows and features while inside other VR experiences. The windows can be attached to tracked devices or left stationary in the world. The primary interaction method involves using controllers with a trigger and grip, but the toolkit also supports voice and gaze interactions with keyboard input. The use cases this toolkit cover by providing views of a user’s desktop, chat platforms, and other windows could be further augmented by using the PVMS to reveal the windows. While including PVMS directly in an application would be ideal, this utility type could use a head gesture to reveal the PVMS as any window type needed.

Half-Life: Alyx[57] released in 2020 and continues the story of its Half-Life franchise from previous non-VR games. The menu system[58] used in the game shares some similarities to the

[55] Kinetic Games 2020, “Phasmophobia”, URL: https://store.steampowered.com/app/739630/Phasmophobia/, Last accessed 19/12/2021.
[56] Curtis English 2019, “OVR Toolkit”, URL: https://store.steampowered.com/app/1068820/OVR_Toolkit/, Last accessed 19/12/2021.
[57] Valve 2020, “Half-Life: Alyx”, URL: https://store.steampowered.com/app/546560/HalfLife_Alyx/, Last accessed 19/12/2021.
[58] Litruv 2020, “Half-Life: Alyx | Menu (Developer Reference)”, URL: https://www.youtube.com/watch?v=MejZ85cAgRc, Last accessed 19/12/2021.

visual sizing and options of the PVMS. Each menu level is shown as a concise small window similar to how the PVMS layout visually appears with room for six options and, where necessary, the ability to scroll and view more. The menu is interacted with using the handheld controllers to point. The menu supports multi-level menus by showing all levels that have been expanded. As each level is opened, the previous levels shift to the left. The new menu level appears central to where the user was interacting or to the right if the user interacted with a higher level panel. This type of menu interaction would enhance the PVMS for uses such as the object manipulation tasks used in experiment three by allowing viewing of multiple menu levels.

BONEWORKS[59] was released in 2019 as an experimental physics VR adventure. The player primarily uses VR handheld controllers to interact with objects in the environment, including physics weapons, tools and other objects to fight against enemies. The main menu is a virtual room where the player can move around, similar to other games (like Phasmophobia) with multiple menus and information dialogs shown as static world-space elements. As shown on the steam page in an example GIF, some physics-based movement requires one or both hands. The GIF shows the player dual-wielding crowbars held onto by hands that can then use physics interactions to hold onto objects. For a situation where a user has both hands engaged gripping objects, this limits the ability for a user to engage with secondary menus. In this situation, the PVMS would provide a HMD Only type interaction suitable to allow continued menu interaction while the user is actively holding onto virtual objects.

This section has considered comparisons against other work to describe the PVMS alongside the experiences created by other developers and researchers. The section began by citing examples of papers that cited the publication from this dissertation, compared against a few other examples of recent research, compared against menus available on the Unity store, and some examples from games. The following section presents a framework for using the PVMS to establish additional considerations for developers and researchers seeking to use it.

[59] Stress Level Zero 2019, “BONEWORKS”, URL: https://store.steampowered.com/app/823500/BONEWORKS/, Last accessed 19/12/2021.

8.3 Periphery Vision Menu System Framework

This section extends the discussion presented as part of Chapter 4. Section 4.6 defined the components with some recommendations. Further to the information given prior in the dissertation, this section focuses on the framework to describe the PVMS and its contribution to the broader community for development and research. The framework is presented by discussing a selection of topics that should be considered part of the implementation. Initially defining the components for clarity, then discussing the use of menus focusing on HMD Only use. Following those topics, questions are presented to consider when deciding if the PVMS is a correct menu type to use, then details about strategy for incorporation and selection of appropriate configurations. The section finishes by discussing use with 6DOF, AR, and other non-HMD benefits.

Components in Brief

The PVMS has been described throughout the thesis identifying components and the nuances that make up its use along with how they factor in HMD Only input. The core PVMS consists of the gesture, two-step widget, and menu presented with options. Each of these is listed below, with a concise description of their use as part of the framework.

- **The PVMS Gesture**: The gesture quintessentially involves a head turn to the left, right, up, or down for a minimum distance in a maximum time and maximum variation in rotation on the other axis. The gesture has been validated as a viable means for revealing a hidden menu.
- **The Two-Step Widget**: The two-step widget presents a small intermediary interaction waypoint for revealing the PVMS. Results have demonstrated its use helps minimise the impact of accidental reveals. The widget is shown as a result of the PVMS Gesture. A typical interaction with the two-step widget would involve instant revealing of the PVMS menu from directly looking at the widget. Ignoring the widget should cause it to disappear after a time. Timings are detailed in section 4.6.
- **The PVMS Menu**: The PVMS menu presents contextually appropriate options to the user depending on the user’s current situation. Menu revealing can be completed via the gesture (experiment 2) or the recommended interaction with the two-step widget (experiment 3). A typical interaction with the PVMS menu would involve looking

directly at the desired option and using a hover-to-select action. Alternatively, a user could ignore the menu for a time to hide it or use the close button (added in experiment three) to force it to hide.

Menu Types and Uses

Sections 5.2.4 and 6.2.4 identified the different interface elements used throughout experiments two and three. In section 8.2, the discussion presented newer variations of menus for VR. The individual parts of menus can be mixed and matched depending on the requirements of an application. The following list describes the menus used in this research and their uses.

- **PVMS Menu**: The PVMS menu provided four options to the user with a menu title and included a close button in the third experiment version. The PVMS menu provided contextual options dependent on the experiment state. The number of options, layout and visual style could be adjusted to the requirements of an application. A small menu region should be left empty to support hover-to-select type interactions. After revealing the menu, this region should be where the cursor will initially aim.
- **Information Dialog**: The information dialog was used to provide a static element conveying tutorials and experiment information. It provided a convenient and straightforward user experience by simply showing information with a confirmation button. Adding a back button may help improve the experience of using the dialog for non-experiment applications. A close button could be added to skip all information. This dialog was designed to be a static element as part of the world with pre-determined placing.
- **Rating Dialog**: The rating dialog was very similar to the information dialog. The dialog presented a question to the participant, and instead of a confirmation button, it had five buttons for designating a rating. As a variation on the information dialog the buttons could be changed to match the requirements of a question.
- **Circular Menu**: The circular menu was used by placing it based on where a user interacted. Participants reacted positively to the circular menu being used alongside the PVMS. The circular menu could be used as an alternative layout for the PVMS.

When creating menus related to a specific element in the world, the circular menu does provide a suitable experience to extend the PVMS with additional interaction.

All the menus described above provide experiences that support HMD Only. They all provide simple versions of the dialogs and have scope to expand with more detail and functionality.

Questions to Ask When Choosing the PVMS as a Solution

The following list of questions should be considered when deciding if the PVMS framework is appropriate for an application or use case. The PVMS is not necessarily suitable for all situations, and these questions generalise the determining of whether the system may be appropriate.

- **"Is the application using a HMD?"**
  The PVMS is designed for use specifically with a HMD. The PVMS may be used with any 3DOF rotation input as a gesture, but the technique primarily benefits situations where it is beneficial to hide menus. If not developing for a HMD, it is not recommended to use the PVMS.
- **"Would the application benefit from hidden menus or menus available anywhere?"**
  The PVMS, by the definition of its components, hides away the menu and allows the menus to be accessible anywhere. Hiding the menus minimises the impact of screen space used by menus during regular use. Having menus available at any location allows random access to contextually appropriate options, which is likely to be useful for many types of applications. If either hidden or available anywhere type properties are desired for menus in an application, then using the PVMS as part of a solution will improve the experience.
- **"Is HMD Only input desirable?"**
  Many applications for VR incorporate controllers of some form, so HMD Only input may not be the only type of interaction provided by an application. There may be situations by design where a user is required to use the controller for an independent action while still wishing to access a menu. Another use case may be to provide input for users who do not use their hands due to a disability or some other cause. As part of any HMD Only type interaction solution, the PVMS provides a useful experience to the user that enhances the capabilities with limited input requirements.

- **"Are there reasons why the PVMS menus would cause issues for the user?"**
  In section 4.2.6, the situations where the PVMS should not be used were discussed. If an application being developed involves distinct hazards, the PVMS may not provide a suitable interaction technique. If turning away for a moment to engage the menu is likely to be a serious issue, then a different solution should be considered.

These questions have generalised choosing to use the menu. It can be incorporated as either an independent interaction technique for some applications if they can be fully controlled with a HMD. The PVMS can also be paired with other types of input/menus to complement them or as a choice for the user.

Incorporating the PVMS

The incorporation of the PVMS will depend on the development environment used. The demonstrated examples in the experiments used Unity, but anything capable of deploying a VR or AR application should be suitable. In Unity, the PVMS scripts can be placed as a component that has identical translations and rotations applied to its object as performed by the HMD via the camera object. Suppose it is impossible to attach directly to the camera in an environment as part of a child-parent relationship; in that case, an observer model can be used to either observe the camera or directly observe the HMD orientation input. Access to the camera's position and orientation as a minimum is important to calculate offsets for placing the menus in the world. The menus can be stored as disabled objects until the detection of a gesture.

Selecting Appropriate Configurations

Calibrating the PVMS may vary depending on the type of application. Calibration of the PVMS has been discussed in section 4.6. It would be suitable to use a configuration such as the medium sensitivity used in the experiments if a plug and play experience is the only requirement for an application. It is recommended that developers incorporate some options to control the sensitivity of gesture recognition. One option is to allow direct access to a slider for one or more attributes and give users complete control. Some users may wish to have this much control, but it may depend on the target audience of an application. For example, there is a distinct difference between a professional domain-specific application requiring precision and a game designed for ease of access by children. Another option for presenting configuration to users would be in the form of presets. During the development of an application, a round of testing could be conducted to evaluate suitable configurations for different people. These options could be presented with a few descriptive words to users indicating the difference. Both approaches could be offered to users with presets and complete control. Providing a way to test changes as they are being changed would let users maximise their preferred use.

Using the PVMS with 6DOF

The examples demonstrated in experiments two and three used 3DOF and did not allow free movement with 6DOF. Therefore, it is worth considering how 6DOF could change the implementation and use of the PVMS. When moving from 3DOF to 6DOF, the additional degrees of freedom allows a user to move within the world. The menus were created using world coordinates. This use of world coordinates means that moving to 6DOF with the demonstrated implementation would result in users moving closer or further away from the menu as they navigate the world. This behaviour may be desirable depending on the scenario. It would allow an additional trigger to hide the menu if a user moved too far away. One example use case could have the described behaviour where the menu is created in a fixed location and continues to persist with optional hiding from proximity. A menu related to a specific feature with context to the current location may benefit from a fixed in place menu to prevent a user from performing an action somewhere unexpected.

Alternatively, it may be desirable for a menu to move with the user, either because the user is physically moving or has movement applied to their avatar. An example may be the user wishes to change options as they move within the world. The most straightforward approach would be to keep a consistent offset with the menu a constant distance away from the user. The menu in this situation would still hide based on no interaction after a period, just as it would in the other described 6DOF scenario. The menu would ideally not mirror rotations of the user and only translate position. When using this approach, it is essential to consider the impact of occlusion. Occlusion from accidentally moving the menu inside objects could be handled by rendering the menu as a separate layer in the camera’s view to make it always on top. Another option for handling the situation could be to scale and move the menu to make it appear consistent while keeping it in front of objects.

A final consideration for 6DOF is that of how gesture detection could change. Detection of the gesture may not change for movement in 6DOF, but there may be situations where it is desirable to restrict triggers. During significant translation around an environment in ways that also cause the user to rotate a substantial amount will likely trigger the menu using the default approach. Similar to how the thresholds are set for deviation on angles, it would be possible to impose restrictions on translation. It is plausible that some use cases would prefer a menu only appear while the user is at rest in a semi-stationary position or travelling at a minimal speed. The restrictions could be tied to different types of menus for specific contexts or broadly for all menus. In any case, the user should be told any requirements of interaction.

Use for Augmented Reality and Non-HMD Considerations

This dissertation has presented the PVMS with use in a VR environment. There is no significant reason why the PVMS could not be applied similarly for AR, provided the target HMD has the required minimum 3DOF to detect gestures. HMDs for both VR and AR would benefit the most from the PVMS because the primary benefit is how the menu is hidden until needed. Other forms of VR and AR do not take up a user's entire field of view, and therefore it is not as significant if a menu occludes the whole space. Use with non-HMD type devices would be possible, although it is unclear whether it would provide a useful experience compared to other alternatives. For example, using a mobile device to detect the gesture while moving in 6DOF would be possible. There may be a use case where this is beneficial as part of AR, but generally, the PVMS is intended to be for HMD interaction.

This chapter has presented additional discussion to identify the significance of the PVMS through the research questions. Following the research question discussion, the PVMS was compared against the work of others to acknowledge the passage of time, newer research and emphasise the relevance of the PVMS. Finally, this last section has presented a framework to help future researchers and developers use the PVMS. Before a conclusion to the overall dissertation, the next chapter will conclude the research questions and briefly discuss areas for future work. The chapter provides suggestions for further improvement using this work as a foundation. The recommendations consider the ease of development and how end users can best utilise the techniques.

9 Conclusion

Throughout this research and as presented in this dissertation, the Periphery Vision Menu System (PVMS) has been defined and established as a technique for improving HMD interactions. With a focus on providing an experience that does not interrupt the immersion and main flow of an application through the provision of menu interactions controlled directly with movements of the HMD. The system provided functionality that was easy to learn and useful. Selection techniques were implemented to operate with no additional input device (examples of input devices used in the experiments as comparison included mouse, mobile phone, and Xbox controller), relying on time delays with hover selection. This approach still provided an experience where participants could complete tasks successfully. The success of this interaction technique was evaluated in the experiments.

In the introduction as part of section 1.2.1, the dissertation established both research goals and research questions. The questions were discussed in each experiment chapter (3.4.1, 5.4.1, 6.4.1) and discussion (8.1). Before addressing conclusions to the research questions, the following were identified as research goals that contributed to identifying the research questions.

- To investigate how user interfaces can be improved for head-mounted displays—specifically looking at the ease of interaction, tools of interaction, and presentation of interactive responsiveness.
- To develop applications demonstrating prototypes of behaviours for head-mounted interactions with a variety of input sources that improve the ease and usefulness of interaction.
- To collect user feedback from a collective of people who experience using the applications to improve the methods of interaction and interfaces.
- To draw conclusions as to the usefulness, usability and other features of the proposed interfaces and interactions based on the user feedback.

The goals have each been considered as part of every experiment and the underlying background research. Investigation into the background was identified in Chapter 2 and then investigated further through testing and implementation. Three separate experiments were completed, demonstrating work toward improving the ease and usefulness of interactions, with a significant focus on the PVMS. Feedback was collected from the participants who took part in the three experiments. The data has been shown to validate the usefulness and usability of the proposed interface experience of the PVMS. The provided supporting evaluation demonstrates that the goals set out to be completed as part of the research have been achieved.

With the goals discussed, the next point to discuss is the research questions themselves. A summary of the research questions follows. For each question, the work from the dissertation as a whole is considered to provide conclusions.

- **RQ 1:** How can head rotations, captured through the orientation sensors of a HMD, be used to increase the user interaction capabilities with virtual worlds?
- **RQ 2:** Does using a head-mounted display with head-only interaction for menu navigation provide a similarly useful experience compared to integration with a tool for instant selection?
- **RQ 3:** Can a hidden menu provide a viable and useful tool for interaction with head movement as the mechanism for revealing it?
- **RQ 4:** Does a hidden menu revealed and interacted with using the head provide a useful experience compared to menus appearing at a fixed place in a virtual world?

9.1 Research Question Discussion

RQ 1: How can head rotations, captured through the orientation sensors of a HMD, be used to increase the user interaction capabilities with virtual worlds?

Participants provided a significant amount of data relating to the Periphery Vision Menu System usability from questionnaires completion and application use. During the second experiment, participants were directly asked if they found the technique useful with a response of 20 yes and 3 no. The dramatic bias toward finding the technique useful is important because it indicates that participants with their existing individual knowledge of interactive technologies formed an opinion that led them to deem it useful. This response was supported by analysing the other data based on the other responses by participants and their actions during the experiments demonstrating the technique worked as a concept. The System Usability Scale results presented a statistically significant difference from the original experience in the first experiment (HMD Only) with an average of 68.47 (SD = 22.79) when compared against the second experiment's average of 82.17 (SD = 11.26) with p = 0.02. Comparing the first and second experiment's System Usability Scale results against the third experiment did not show a statistically significant difference. The third experiment did demonstrate a statistically significant difference in the increased response to the correct times the interface was displayed. With results of 7.87 (SD = 1.32) in experiment two and 8.85 (SD = 0.81) in experiment three and p = < 0.01. The higher values for correct times against wrong times were significantly statistically different for both experiments with p = < 0.01. The response for wrong times was demonstrated from observation to be less impactful in the third experiment from the addition of the two-step behaviour. The primary factor that was shown to be significantly statistically different was the increase in participants views on the system being more cumbersome in the third experiment. Indicating future work is necessary to establish ways to improve from iterating on feedback. From this data, it can be concluded with confidence that the Periphery Vision Menu System provides a useful experience for functional interaction within a HMD environment.

From the demonstrated validation of the PVMS as a useful technique for use in VR with HMDs, the question (RQ1) has been satisfied with a significant prototype and analysis from participant evaluation. In addition to the PVMS, the research demonstrated the use of the head for more general interaction with hover-to-select. Hover-to-select as a more general technique was demonstrated as useful for interaction with any type of interface system when focusing on using the head as an interaction tool.

RQ 2: Does using a head-mounted display with head-only interaction for menu navigation provide a similarly useful experience compared to integration with a tool for instant selection?

Evaluation of the effectiveness and a comparison of additional input types was carried out in the first and third experiments. The first experiment compared HMD Only pointer movement with the addition of two inputs providing a tap-to-select experience with mouse and mobile phone. The third experiment compared head only interaction and the use of an Xbox controller for tap-to-select. Participants responded in both cases to indicate the tap-to-select interactions were more useful or usable. In the third experiment, participants responded with averages of 3.65 (SD = 1.02) for the head only Tower Defence game interaction compared to

4.31 (SD = 0.79) with a controller added, with p = < 0.01 showing these were significantly statistically different. Comparing the System Usability Scale results for the HMD Only interaction in the first experiment against the two other input types was not significantly statistically different. From observation of the average data, there was preference toward using the two different tap-to-select options. The second experiment asked participants to compare input preferences before and after the experiment. From observation, participants on average shifted their preferences from the perception that HMD Only interaction would be the worst to being on average the second-best preferred after Kinect/Leap Motion sensors. The change was not shown to have a statistically significant difference. From the collected data, it can be concluded that participants prefer to use additional interaction devices to augment their experience when available. The responses were still positive toward the system, though, so it can be argued that the experience is usable and useful when using the HMD Only experience. The main point participants reported as a negative toward the HMD Only experience compared to the tap-to-select interaction was selection time. For the experiments, this was intentionally left high to mitigate user error from novice users. One significant element for future work is to evaluate this impact of selection time.

In addressing RQ2, participants did find the addition of an instant selection technique to be preferable. The positive response to the HMD Only approach was rated slightly behind the instant selection tool. The positive response suggests that the proposed HMD Only approach can exist by itself, and where appropriate, it can be augmented to benefit from a button for instant selection.

RQ 3: Can a hidden menu provide a viable and useful tool for interaction with head movement as the mechanism for revealing it?

In Chapter 4, the design and implementation of the PVMS were presented and subsequently evaluated as part of the second and third experiments. The PVMS was shown as a viable and useful tool, as elaborated in the discussion from RQ1. The mechanism for revealing the PVMS was designed as a simple gesture to perform, allowing easy access to the menu. The initial version was evaluated in the second experiment, as presented in Chapter 5, where the complete menu was made to appear immediately. The third experiment, presented in Chapter 6, evaluated a subsequent iteration of the PVMS that sought to minimise the issues with occasional accidental reveals of the menu by using the two-step widget. The addition of the two-step widget was shown to improve issues found in experiment 2's version through the analysis in section 7.2.7 of the combined results. Therefore, this work has clearly shown an example that satisfies the question by showing that the PVMS can exist hidden and ready to use whenever a user needs it.

RQ 4: Does a hidden menu revealed and interacted with using the head provide a useful experience compared to menus appearing at a fixed place in a virtual world?

The merits of the PVMS have been discussed at length as part of the above questions and previous chapters. In answering this final question, it is important to consider the aim of interface elements. In the context of interruptive feedback, i.e., presenting a choice that must be completed before continuing with an application, a menu fixed in place as part of a VR world could be used. Revealing a menu demonstrates to the user that interaction is required when they approach it, or the user is notified of the need for interaction when the menu appears. The PVMS allows freedom to interact with a menu that can appear at any time in any place. The PVMS can provide contextual menus depending on the state of an application and does not need always to show the same menu. This functionality gives the user freedom to engage with the VR world with the ability to access menus as required. As suggested by the data captured from the experiments, the hidden menu provided by the PVMS does provide a useful experience alongside fixed place menus. It can exist alongside or as an alternative if necessary.

In conclusion, the Periphery Vision Menu System has been detailed and evaluated through analysis demonstrating statistical significance with improvements to the experience across the three experiments. From the analysis, it can be concluded that the research objectives have been completed, with the various research questions addressed. The experiments have demonstrated examples of use and provided a baseline experience with many areas to continue developing the future iterations. It is hoped this system will be integrated with future software to provide an improved experience in the scenarios suggested in the technology overview chapter (Chapter 4).

The remainder of this chapter will provide sections relevant to closing out the presented research. These begin by elaborating on the contributions made by this research, identifying the significance of this work. After the contributions, there follows the identification of the limitations affecting the research and how they impacted the design, experimentation and evaluation. The last main section covers future work, identifying areas where the presented work could be developed further outside the scope set for this dissertation and research. Finally, the chapter concludes with some final remarks.

9.2 Contributions

The contributions have already been discussed and identified as part of section 1.4. This section seeks to restate the contributions and represent them in the context of the totality of the content presented. The actual Periphery Vision Menu System software solution is the primary contribution and represents a significant addition to the available options for interaction in VR environments. Before discussing the contributions of the PVMS further, this section will elaborate on the contributions from each of the individual experiments and then lead into sections related to specific topics contributed as part of the PVMS.

Contributions from the First Experiment

The first experiment was targeted toward preparing an initial experience in VR demonstrating the viability of HMD Only interactions. These were contrasted against the inclusion of alternative input with the addition of the mouse/mobile phone inputs acting as a tap-to-select to vary from the hover-to-select used for HMD Only. The experiment demonstrated the viability of the HMD as an independent tool for interaction without the aid of instant selection provided through additional input devices. The experiment captured data related to the general usability experience and the concerns to be considered with the iteration into the second and third experiments. The data was evaluated and provided the results presented in section 3.4 of the first experiment chapter and compared to the other experiments throughout Chapter 7.

Contributions from the Second Experiment

The second experiment was designed to provide usability testing data to determine if the proposed PVMS provided participants with a viable, useful, and usable experience. The experiment contributed validation through demonstrated success from the data and results presented in section 5.4 and the comparison to contrast against the other two experiments in Chapter 7. The approach to experimentation embraced using games as a platform for testing as part of both the experiment and the advertising campaign seeking participants. The serious games aspects will be discussed later.

Contributions from the Third Experiment

The third experiment contributed an iteration improving on the PVMS experienced by participants in the second experiment. The addition of the two-step widget provided demonstrated improvement through the minimisation of impact from accidental errors. The experiment contributed comparison of the PVMS, considering situations where the additional input device for instant selection was compared against HMD Only. The comparison of inputs provided linked back to the initial testing as part of the first experiment. It combined the addition of matching puzzles similar to those in the first experiment with the continuation of a serious game focus primarily through the tower defence. By completing the experiments and collecting data from participants, the results contributed as part of the experiment in section 6.4, and the comparative discussion in Chapter 7 and Chapter 8 validated the PVMS.

Identification, Prototyping, and Evaluation of Suitable Interaction Mechanisms for Head-Mounted Virtual Reality Experiences

There were contributions related to the demonstration of interaction mechanisms to improve a user's VR experience in all three experiments. The focus of the evaluation mainly was toward establishing the significance of the PVMS. In doing this, other types of interaction techniques were used for comparison. More generally, the hover-to-select that was used extensively through every task was indirectly evaluated, demonstrating its usability success. Comparison between techniques was used for both the situations where the focus was on providing a HMD Only experience with hover-to-select and the instant selection provided by independent devices. The interfaces demonstrated included the PVMS menu, including the implementation of the two-step widget, the statically placed menus, such as those for the introduction of tasks, and the circular menus, represented as buttons circling around the interaction point which were displayed in context. These various menu interaction processes provided functional, usable and deployable examples of suitable tools for interaction with the HMD Only technique.

Considering the PVMS specifically, the PVMS interaction technique addresses the needs of end-users and the requirements of developers. The interaction technique identified was using a head gesture by rotating the head to either the left or right, thereby triggering an appropriate contextual menu activated through the use of the two-step widget. When combined, this experience highlights how users can be given access to menus with HMD Only interactions or provided to users who need menus made visible while moving through a VR world. The PVMS provides a suitable experience for users and a simple implementation for developers. Integration of the chosen interaction experience can be included for deployment to any HMD that utilises orientation sensor data to orient the users' view of the virtual world.

The evaluation conducted as part of this dissertation has demonstrated that the PVMS can work by itself if necessary or alongside other interfaces to provide an integral part of the experience for end-users. The evaluation data, captured across the three experiments and presented in their respective chapters (3, 5, and 6), were further evaluated for comparative insight as part of Chapter 7.

Demonstrated Integration of Serious Games as a Tool for Research

The second and third experiments contributed an approach to serious games that utilised games as a tool for driving interest in research. By incorporating the interactive experience to be evaluated into a game, the advertising benefited from a story-driven trailer for each experiment. Both experiments that utilised this approach saw increased numbers of participants. The recruitment is speculated to have been aided by the game trailers and the positive response to the trailers from participants who chose to participate. The game trailers drew participants in by demonstrating entertainment value. Additional data was collected about participant intent via watch data from these trailers. Once the participants were registered for the experiment, they were presented with non-game activities for calibration and object manipulation, which reinforced game mechanics within the tower defence games and the use of the PVMS. Encouraging participation in research is important for continued access to data generated from participants. Where appropriate, incorporating game elements to use a serious game for evaluation of research is beneficial and recommended to improve the engagement of the participants.

Publication of Preliminary Data

The final area of contribution was the publication of preliminary data from the second experiment. The data was documented in a paper presented at the 28th Australian Conference on Human-Computer Interaction (OzCHI2016) (Mitchell and Wilkinson, 2016). The paper presented and established the concepts making them available to the wider community for research. As the paper contribution only had the second experiment, it did not cover the improvements made in the third experiment and could only briefly describe the system. The subsequent data collection and further evaluation are also being consolidated for future papers.

As stated, the primary contribution was the design and implementation of the PVMS as an interaction technique to provide more options for VR development with the evaluation and analysis to support its significance. In the next section, the limitations are detailed to provide insight into the scope constraints encountered during the research.

9.3 Limitations

Limitations have been included to scope and structure the research. The limitations listed here did not inhibit the research but demonstrated areas where the work could have been improved with unlimited scope. The four areas to be covered are access to technology, the part-time nature of the research, population pools and availability of participants, and study design.

Access to Technology

When the experiments were conducted, the availability of modern HMD models was limited. The HMD devices used for this research were the Oculus Rift Dev Kit 1 and Oculus Rift Dev Kit 2. The Oculus Rift Dev Kit 1 was released on March 29th, 2013, acquired through the original Kickstarter crowdfunding and used for the first experiment between May and June of 2015. The Oculus Rift Dev Kit 2 was released on March 28th, 2016 and was acquired through pre-order and used for the second (March to May 2016) and third (August to December 2016) experiments. The direct competitor at this time for the HMD market was arguably the HTC Vive, released on the 5th of April 2016.

Testing with multiple different models of HMDs may have provided an extra element of nuance to allow for additional experimentation demonstrating the PVMS's portable architecture. In the initial stages of development, HMD libraries had to be included as a separate package for development in the Unity game engine. Around the time when Oculus

Rift Dev Kit 2 was released, Unity support added easier development for HMDs. Due to this, the developed software is anticipated to work on any HMD model supported by Unity with minimal modification if there is access to technology.

Part-Time Nature of the Research

A significant portion of the research was conducted and documented on a part-time candidature for various necessary reasons. The part-time aspect impacted the amount of time that could be devoted to completing the research in a timely manner. Aspects with similarity to the research presented in this dissertation may be present in newer works, with the published paper having already presented the concepts in 2016 (Mitchell P. and Wilkinson B., 2016). The presented results and proposed Periphery Vision Menu System are significant toward improving human-computer interaction with HMDs.

Population Pools and Availability of Participants

The primary source for participants were those sourced from Flinders University dictated by the sources defined from the ethics approval process. Participants were sourced based on their own interest, as can be found individually described in Chapter 3 First Experiment section 3.2.1 Recruitment, Chapter 5 Second Experiment section 5.2.1 Recruitment, and Chapter 6 Third Experiment section 6.2.1 Recruitment. The experiments would have benefitted from more participants to provide an increased amount of data for further validation. The number of participants was enough to provide the demonstrated results. Strategies for increasing participant pools could have been achieved from a variety of possible methods. Some of these methods may have included: increasing the incentive for participants to participate, increasing the number of times potential participants were made aware of the experiments, increasing the duration of advertising, advertising more broadly either within the university or expanding to external sources.

Study Design

The best example of change of study design that dictated how the structure of data was collected from participants is seen from the transition of the first experiment to the second experiment. In the first experiment, participants were confronted with an exhaustive number of data points to fill out. The exhaustive data points were primarily due to filling out perspectives based on each of the input techniques. The second experiment iterated on this to ensure the number of questions provided informative quantitative and qualitative data without overwhelming. Reducing the number of questions asked to the participant at the end was handled by collecting more data directly within the experiment. After each stage, the user response questions were used to provide in the moment feedback immediately after tasks had been completed. This approach was continued for the third experiment. In an ideal case for data collection with no limitations, there may have been many more questions to ask participants at the cost of keeping them answering for longer. It cannot be said whether exhaustive additional questions would have provided useful substantive data at the risk of losing participants’ attention span as they tried to remember what happened during the experiment.

9.4 Future Work

The work within this dissertation has led to an initial prototype for the Periphery Vision Menu System. However, there are many areas where this system can be further enhanced to provide a better experience for the end-user. The following sections are short explorations of some areas where future work would help improve the proposed system's quality, reliability, and usability. The sections covered for future work include long term goals, alternative menu layouts, algorithm optimisations, support for scaling sensitivity, modular systems for automatic contextual support, further iteration of the periphery preview functionality, and integration testing with additional development environments.

9.4.1 Long Term Goals

The other areas of future work all discuss more immediate areas of possible ways to improve the experience. Long term goals in this context are more related to the next 5 to 10+ years. Given the rapid technology changes, it is evident that technology will change in many potentially unexpected ways. The speed of development is likely to depend on market demand from businesses and potential end-users.

Smart Glasses

As digital components become more compact, the form factor of complex technology may become more mainstream for eyewear. If this eventuates, performing simple head interactions may be ideal for a primary way to perform many interactions. Use of the

Periphery Vision Menu System could adapt to provide hidden till necessary menus. These types of devices would be essentially very compact AR hardware.

Non-Head Mounted Uses

The concept of hidden menus triggered by looking rapidly in a direction could be applied to devices external to any head-mounted screen. Providing the orientation of a user's head can be tracked the technique could be modified to adjust to other forms of technology. For example, a car may have a transparent digital screen embedded as part of a car's windscreen. Performing head interactions, the user may be able to reveal a Periphery Vision Menu and perform interactions with that menu using either their head or other methods. In this specific scenario for safety, it would be ideal only to allow the interaction to occur when a car is at a complete standstill or if the vehicle is engaged in an autopilot mode.

Head-Mounted Virtual Reality and Augmented Reality

The devices like those existing now for both VR and AR are likely to still be the primary use of direct head-mounted visualisation for a long time yet. The Periphery Vision Menu System is expected to provide versatility to adapt to any new functional and structural designs. Changing perspective within a virtual environment requires tracking of the orientation at a minimum. This requirement for the displays to provide their baseline feature set enables the functionality of the techniques in this research.

9.4.2 Alternative Menu Layouts

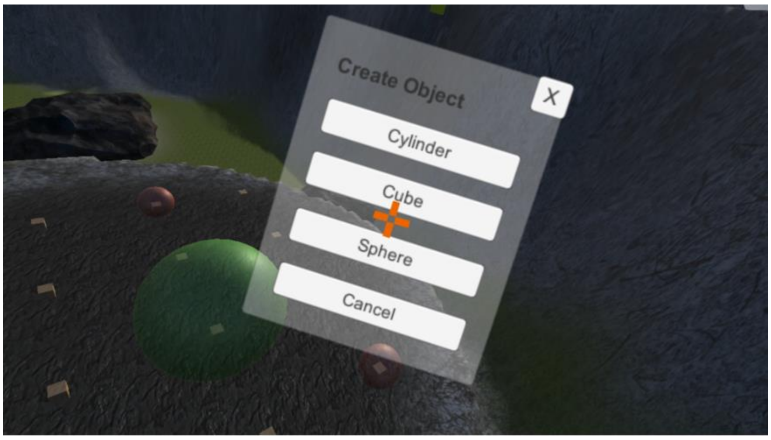


Figure 9.1: Example Periphery Menu Layout

As shown in Figure 9.1, the general layout used for this experiment presented a simple interface with four primary buttons stacked from top to bottom. This layout increases travel time for the cursor from opening the menu when reaching the very top or bottom options. One possible alternative layout could represent the menu in a similar way to the circular menu that was shown during experiment three. In this way, the menu options would be equidistant from the starting point of the cursor. Importantly the needs of each application will likely be different in factors such as the number of options, types of controls (non-button elements, for example), or custom visual themes. Application requirements changing to suit a use case suggests that the layout will vary dependent on those needs.

9.4.3 Algorithm Optimisations

As previously discussed in Chapter 4, the calculations used to determine if a trigger event had occurred were processed in full every update for the prototype to ensure everything was working properly. The process involved determining the maximum difference between angles on each axis over the maximum interaction time period.

One of the most significant increases to operational speed would be to cache the rotation angle shifts. The history could keep track of minimum and maximum values for all three axis rotations at the cost of a small amount of extra memory. These could be initialised at the time of object creation or the first time a corrected angle is requested, providing a way to perform an update only after a minimum difference has occurred instead of checking every frame. Another optimisation may be to cache the maximum and minimum for a running set of history. When new elements are added or removed, these running values could be used to determine if recalculation is needed.

9.4.4 Support for Scaling Sensitivity

There is often an option to control the mouse sensitivity in applications where a mouse is used accurately with rapid movement (particularly for games). Sometimes independently on each axis. That same type of functionality could be provided for the Periphery Vision Menus. Each property of the system can be represented as a scale with minimum and maximum threshold values to allow customisation. Default values like those tested in the completed experiments may fall somewhere in the middle of these ranges. The option for customisation could then give users control over how they want to interact with the application. Figure 9.2 (over) and Figure 9.3 (over) show some of the data useful in determining boundaries. Some of the values that could be opened to individual user choice could include those in the following list. For a more simplified configuration on the end user's side, a lesser number of sliders could be used to control multiple properties at the same time. An example would be a single sensitivity slider controlling all the thresholds simultaneously in a consistent way.

- **Threshold Angle**: The angle a user must at minimum rotate within an interaction time period to trigger the menu. The angle can be either yaw or pitch, depending on the configuration.
- **Z Angle**: The roll angle indicating the head is moving in a direction that is not up/down/left/right.
- **Other Threshold Angle:** The opposite of pitch or yaw used for the threshold angle. Used to determine the head is not rotation significantly on the secondary axis.
- **Interaction Maximum Time**: The maximum period during which a threshold angle is successfully rotated while meeting other conditions.
- **Menu Option Selection Time**: The time it takes to select menu elements.
- **The Direction of Menu Creation**: In cases where different menus are provided for each direction, the user could be given control to allow a preference if it is appropriate.

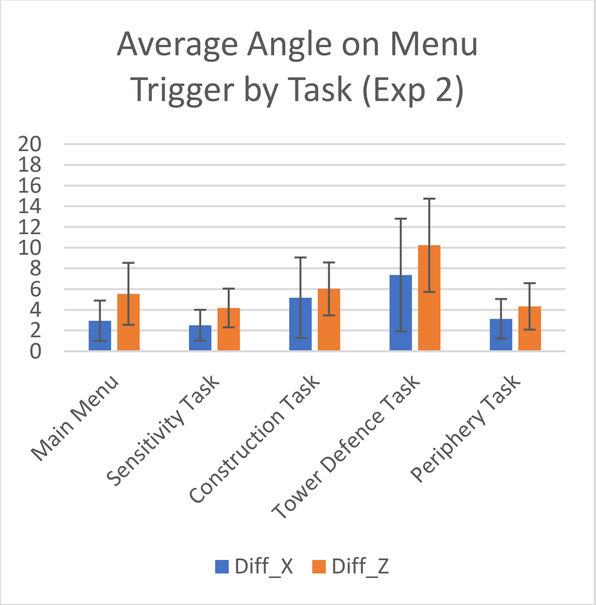


Figure 9.2: Experiment 2 Average Angle on Menu Trigger by Task

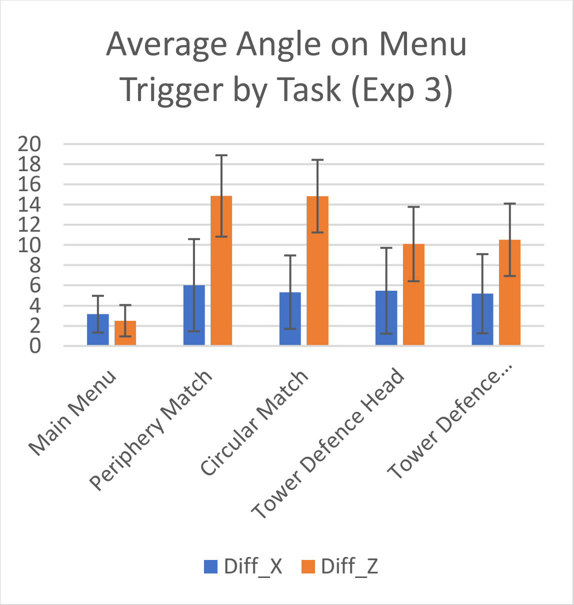
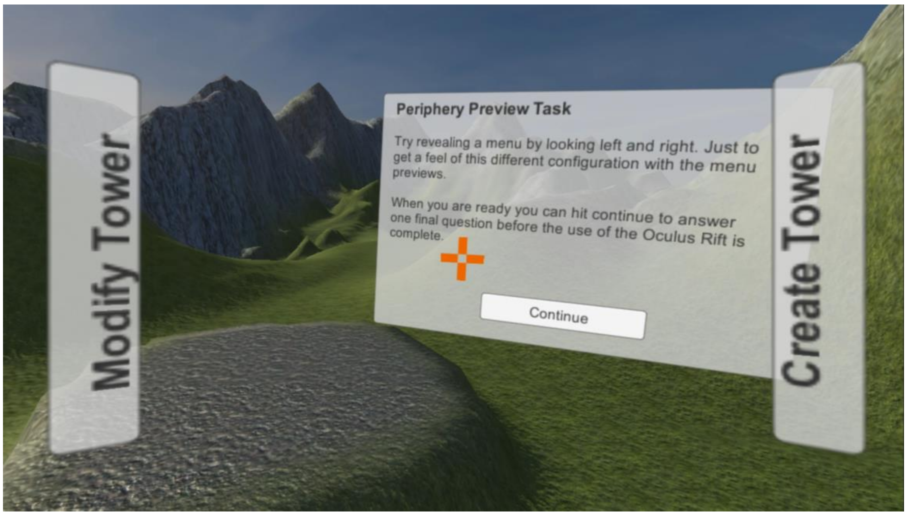


Figure 9.3: Experiment 3 Average Angle on Menu Trigger by Task

9.4.5 Modular Systems for Automatic Contextual Support

In the current prototype, updating the contextual menus is manually done using experiment states. The class can be expanded to house preconfigured definitions for menus tied to a state machine or other conditions. This extension would allow automatic shifts in the types of menus presented to users. With that said, contextual menus need to be sensible and consistent in their presentation to users. Continually changing the types of menus that appear is likely to confuse or frustrate users detracting from the overall product. This confusion leads to the next point that would help users understand the menus that would appear at any time.

9.4.6 Further Iteration of the Periphery Preview Functionality



Periphery Preview Task

Try revealing a menu by looking left and right. Just to get a feel of this different configuration with the menu previews.

When you are ready you can hit continue to answer one final question before the use of the Oculus Rift is complete.

Figure 9.4: Periphery Preview Example from Experiment 2

In the second experiment, a simple prototype of the Periphery Preview functionality was presented to participants (Figure 9.4). The purpose of this type of visual was for previewing the type of menu that would appear when the user turns their head in a specific direction. The issue with this was the interface felt a little too impactful. Further investigation is required to look at the impact of using more transparent effects, smaller marker indicators, or other types of identification for the menus that will appear.

9.4.7 Integration Testing with Additional Development Environments

The prototype was tested with the Oculus Rift Dev Kit 2 (and Dev Kit 1 in the first experiment). It would be beneficial to evaluate how the system performs on many different head-mounted displays. To identify any specific issues that manifest in untested situations. This system is hoped to be useful and usable to applications developed on any HMD for either VR or AR. Therefore, this type of testing is essential to show developers that this interaction technique will integrate with their targeted platforms. This platform integration also includes expansion from VR testing to working on integration with AR-type head-mounted displays.

9.5 Final Remarks

Head-mounted displays have become more accessible as hardware manufacturer competition drives the niche markets for uses in gaming and business, among many others. The future uses for the devices will continue to evolve and meet the demands of users. This dissertation has demonstrated the viability of head-based interactions with the Periphery Vision Menu System in VR. The techniques proposed in this dissertation have room to continue improving with future research alongside developers finding new innovative ways to make them appropriate for specific applications.

10 References

Aish R. 1979, "*3D input for CAAD systems*", Computer-Aided Design, 11(2), pp. 66-70.

Ajanki A., Billinghurst M., Gamper H., Jarvenpaa T., Kandemir M., Kaski S., Koskela M., Kurimo M., Laaksonen J., Puolamaki K., Ruokolainen T., and Tossavainen T. 2010, "*An augmented reality interface to contextual information*", Springer, Virtual Reality, 15(2-3), pp. 161-173.

Azuma R. and Bishop G. 1994, "*Improving Static and Dynamic Registration in an Optical See-through HMD*", in Proceedings of the 21st annual conference on Computer graphics and interactive techniques 1994, SIGGRAPH '94, pp. 197-204.

Azuma R. and Bishop G. 1995, "*A Frequency-Domain Analysis of Head-Motion Prediction*", in Proceedings of the 22nd annual conference on Computer graphics and interactive techniques, pp. 401-408.

Azuma R. 1997, "*A Survey of Augmented Reality*", Presence: Teleoperators and Virtual Environments, 6(4), pp. 355-385.

Azuma R. 2004, "*Overview of Augmented Reality*", SIGGRAPH2004 Course Notes.

Bai H. and Lee G. 2012, "*Demonstration: Interaction Methods for Mobile Augmented Reality*", in Proceedings of the 13th International Conference of the NZ Chapter of the ACM's Special Interest Group on Human-Computer Interaction, CHINZ'12, pp. 101, Dunedin, New Zealand.

Banes D. 2009, *"Microsoft Surface – a new approach to access and technology",* Journal of Assistive Technologies, 3(1).

Bangor A., Kortum P., and Miller J. 2009, *"Determining What Individual SUS Scores Mean: Adding an Adjective Rating Scale",* Journal of Usability Studies, 4(3), pp. 114-123.

Barna J., Novak-Marcincin J., Janak M., Marcincinova L., Fecova V., and Torok J. 2012, "*Open Source Tools in Assembling Process Enriched Elements of Augmented Reality*", in Proceedings of the 2012 Virtual Reality International Conference, pp. 1-8.

Beat Games 2018, "*Beat Saber*", URL: https://store.steampowered.com/app/620980/Beat_Saber/, Last accessed 11/12/2021.

Benko H., Holz C., Sinclair M., and Ofek E. 2016, "*NormalTouch and TextureTouch: High-fidelity 3D Haptic Shape Rendering on Handheld Virtual Reality Controllers*", in Proceedings of the 29th Annual Symposium on User Interface Software and Technology, UIST'16, pp. 717-728, Tokyo, Japan.

Billinghurst M. and Kato H. 2002, "*Collaborative Augmented Reality*", Communications of the ACM, 45(7), pp. 64-70.

Blackwell A., Fitzmaurice G., Holmquist L., Ishii H., and Ullmer B. 2007, "Tangible user interfaces in context and theory", in CHI'07 Extended Abstracts on Human Factors in Computing Systems, CHI EA '07, pp. 2817-2820, San Jose, California, USA.

Bowden Z. 2017, "*Retro review: Microsoft's 2008 Surface 'coffee table' in 2017*", URL: https://www.windowscentral.com/microsoft-surface-pixelsense-table, Last accessed 11/12/2021.

Boyer S. 2009, "*A virtual failure: Evaluating the success of Nintendo's Virtual Boy*", The Velvet Light Trap, (64), pp. 23-33.

British Museum 2018, "*Audio Guide The British Museum: your way*", URL: https://www.britishmuseum.org/visiting/planning_your_visit/audio_guides.aspx, Last accessed 11/12/2021.

Broll W., Ohlenburg J., Lindt I., Herbst I., and Braun A. 2006, "*Meeting Technology Challenges of Pervasive Augmented Reality Games*", in Proceedings of the 5th ACM SIGCOMM Workshop on Network and System Support for Games, Netgames'06.

Bühling R., Obaid M., Hammer S., and André E. 2012, "*Mobile Augmented Reality and Adaptive Art: A game-based motivation for energy saving*", in Proceedings of the 11th International Conference on Mobile and Ubiquitous Multimedia, MUM'12, pp. 1-2.

Chang E., Kim H., and Yoo B. 2020, "*Virtual Reality Sickness: A Review of Causes and Measurements*", International Journal of Human-Computer Interaction, 36(17), pp. 1658-1682.

Chennamma H. and Yuan X. 2013, "*A Survey on Eye-Gaze Tracking Techniques*", Indian Journal of Computer Science and Engineering, IJCSE, 4(5), pp. 388-393.

Cheok A, Fong S., Goh K., Yan X., Liu W., and Farzbiz F. 2000, "*Human Pacman: A Sensing-based Mobile Entertainment System with Ubiquitous Computing and Tangible Interaction*", in Proceedings of the 2nd Workshop on Network and System Support for Games, Netgame'03, pp. 106-117.

Cheok A, Fong S., Goh K., Liu W., Farzbiz F., Teo S., Li Y., and Yang X. 2004, "*Human Pacman: a mobile, wide-area entertainment system based on physical, social, and ubiquitous computing", Personal and Ubiquitous Computing, 8(2), pp. 71-81.*

Choi I., Ofek E., Benko H., Sinclair M., and Holz C. 2018, "*CLAW: A Multifunctional Handheld Haptic Controller for Grasping, Touching, and Triggering in Virtual Reality*", in Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems, CHI2018, pp. 1-13, Montreal, Canada.

Choi J., Jang B., and Kim G. 2011, "*Organizing and presenting geospatial tags in location-based augmented reality*", Personal and Ubiquitous Computing, 15(6), pp. 641-647.

Chun W. and Hollerer T. 2013, "*Real-time Hand Interaction for Augmented Reality on Mobile Phones*", in Proceedings of the 2013 International Conference on Intelligent User Interfaces, IUI'13, pp. 307-314.

Colaço A., Kirmani A., Yang H., Gong N., Schmandt C., Goyal V. 2013a, "*Mime: Compact, Low-Power 3D Gesture Sensing for Interaction with Head-Mounted Displays*", in Proceedings of the 26th Annual ACM Symposium on User Interface Software and Yechnology, UIST'13, pp. 227-236, St. Andrews, UK.

Colaço A. 2013b, "*Sensor Design and Interaction Techniques for Gestural Input to Smart Glasses and Mobile Devices*", in Proceedings of the Adjunct Publication of the 26th Annual ACM Symposium on User Interface Software and Technology, UIST'13, pp. 49-52.

Comeau C. and Bryan J. 1961, *"Headsight Television System Provides Remote Surveillance"*, Electronics, pp.86-90.

Corbett-Davies S., Green R., and Clark A. 2012, "*Physically interactive tabletop augmented reality using the Kinect*", in Proceedings of the 27th Conference on Image and Vision Computing New Zealand, IVCNZ'12, pp. 210-215.

de Gloria A., Bellotti F., Berta R., and Lavagnino E. 2014, "*Serious Games for education and training*", International Journal of Serious Games, 1(1).

de Sá M. and Churchhill E. 2012, "*Mobile Augmented Reality: Exploring Design and Prototyping Techniques*", in Proceedings of the 14th International Conference on Human-Computer Interaction with Mobile Devices and Services, MobileHCI'12, pp. 221-230.

de Sá M., Antin J., Shamma D., and Churchhill E.F. 2011, "*Mobile Augmented Reality: Video Prototyping*", in CHI'11 Extended Abstracts on Human Factors in Computing Systems, CHI'11, pp. 1897-1902.

Dedual N. and Feiner S. 2013, "*Addressing Information Overload in Urban Augmented Reality Applications*", GeoHCI Workshop at CHI2013.

Dennison M. and D'Zmura M. 2018, "*Effects of unexpected visual motion on postural sway and motion sickness*", Applied Ergonomics, vol. 71, pp. 9-16.

Dipietro, L., Sabatini, A., and Dario P. 2008, "*A survey of glove-based systems and their applications*", IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 38(4), pp. 461-482.

Dünser A., Walker L., Horner H., and Bentall D. 2012, "*Creating Interactive Physics Education Books with Augmented Reality*", in Proceedings of the 24th Australian Computer-Human Interaction Conference, OZCHI'12, pp. 107-114.

Emotiv 2018, "*Emotiv Epoc+*", URL: https://www.emotiv.com/epoc/, Last accessed 11/12/2021.

Engelbart D. and English W. 1968, *"A research center for augmenting human intellect"*, in Proceedings of the December 9-11, 1968, Fall Joint Computer Conference, Part 1, AFIPS'68, pp. 395-410.

Feiner S., MacIntyre B., Haupt M., and Solomon E. 1993, "*Windows on the World: 2D Windows for 3D Augmented Reality*", in Proceedings of the 6th Annual ACM Symposium on User Interface Software and Technology, UIST'93, pp. 145-155.

Fisher S., McGreevy M., Humphries J., and Robinett W. 1986, *"Virtual environment display system"*, in Proceedings of the 1986 Workshop on Interactive 3D Graphics, I3D'86, pp. 77-87.

Fitts P. and Posner M. 1967, "Human performance", CA: Brooks/Cole.

Fitzmaurice G.W., Ishii H., and Buxton W.A.S. 1995, "Bricks: laying the foundations for graspable user interfaces", in Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI'95, pp. 442-449.

Fleming T., Bavin L., Stasiak K., Hermansson-Webb E., Merry S., Cheek C., Lucassen M., Lau H., Pollmuller B., and Hetrick S. 2017, "*Serious Games and Gamification for Mental Health: Current Status and Promising Directions*", Frontiers in Psychiatry, vol. 7, pp. 215.

Flynn D., van Schaik P., Blackman T., Femcott C., Hobbs B., and Calderon C. 2004, *"Developing a Virtual Reality-Based Methodology for People with Dementia: A Feasibility Study"*, CyberPsycology & Behavior, 6(6), pp. 591-611.

Foyle D., Andre A., and Hooey B. 2005, "*Situation Awareness in an Augmented Reality Cockpit: Design, Viewpoints, and Cognitive Glue*", in Proceedings of the 11th International Conference on Human Computer Interaction, pp. 3-9, Las Vegas, NV.

Furness T. and Barfield W. 1995, *"Introduction to Virtual Environments and Advanced Interface Design"*, "Virtual Environments and Advanced Interface Design", Oxford University Press, pp. 3-13.

Gamito P., Oliveira J., Coelho C., Morais D., Lopes P., and Pacheco J. 2014, "*Cognitive training on stroke patients via virtual reality based serious games*", Disability and Rehabilitation, 39(4), pp. 385-388.

Google 2013, "*Gizmodo: Here Are Google Glass' Tech Specs by Chan C.*", URL: https://gizmodo.com/5994737/here-are-google-glass-tech-specs?IR=T, Last accessed 11/12/2021.

Google 2014, "*Google Cardboard*", URL: https://vr.google.com/cardboard/, Last accessed 11/12/2021.

Google 2016, "*Tilt Brush*", URL: https://www.tiltbrush.com/, Last accessed on 11/12/2021.

Görgü, L, Campbell A., McCusker K., Dragone M., O'Grady M., O'Connor N., and O'Hare G. 2012, "*Freegaming: Mobile, Collaborative, Adaptive and Augmented ExerGaming*", Mobile Information Systems, 8(4), pp. 287-301.

Görgü L., Campbell A., Dragone M., O'Hare G. 2010, "*Exergaming: A Future of Mixing Entertainment and Exercise Assisted by Mixed Reality Agents*", Computers in Entertainment (CIE), 8(4), pp. 1-3.

Grasset R., Dünser A., and Billinghurst M. 2008, "*Edutainment with a Mixed Reality Book: A visually augmented illustrative childrens' book*", in Proceedings of the 2008 International Conference on Advances in Computer Entertainment Technology, pp. 292-295, Yokohama, Japan.

Griffin T., Giberson J., Lee S., Guttentag D., Kandaurova M., Sergueeva K., and Dimanche F. 2017, "*Virtual Reality and Implications for Destination Marketing*", Travel and Tourism Research Association: Advancing Tourism Research Globally 2017 ttra International Conference, Québec, Canada.

Guan W., You S., and Neumann U. 2012, “*Efficient Matchings and Mobile Augmented Reality*”, ACM Transactions on Multimedia Computing, Communications, and Applications, TOMM, 8(3), pp. 1-15.

Guan W., You S., and Neumann U. 2011, “*GPS-aided Recognition-based User Tracking System with Augmented Reality in Extreme Large-scale Areas*”, in Proceedings of the 2nd Annual ACM Conference on Multimedia Systems, MMSys’11, pp. 1-10, San Jose, California, USA.

Guo J., Weng D., Duh H., Liu Y., and Wang Y. 2017, "*Effects of Using HMDs on Visual Fatigue in Virtual Environments*", in 2017 IEEE Virtual Reality (VR), pp. 249-250, Los Angeles, CA, USA.

Han S., Rhee E., Choi J., and Park J. 2011, “*User-Created Marker Based on Character Recognition for Intuitive Augmented Reality Interaction*”, in Proceedings of the 10th International Conference on Virtual Reality Continuum and Its Application in Industry, VRCAI 2011, pp. 439-440, Hong Kong, China.

Hansen J., Johansen A., Hansen D., Ito K., and Mashino S. 2003, "*Command Without a Click: Dwell Time Typing by Mouse and Gaze Selections*", in INTERACT, vol. 3, pp. 121-128.

Heilig M. 1960, *"Stereoscopic-Television Apparatus for Individual Use"*, US Patent No. 2955156.

Heilig M. 1962, *"Sensorama Simulator"*, US Patent No. 3050870.

Henrysson A., Billinghurst M, and Ollila M. 2005, “*Virtual Object Manipulation using a Mobile Phone*”, in Proceedings of the 2005 International Conference on Augmented Tele-existence, ICAT 2005, pp. 164-171, Christchurch, New Zealand.

Henschke M., Hobbs D., and Wilkinson B. 2012, “*Developing Serious Games for Children with Cerebal Palsy: Case Study and Pilot Trial*”, in Proceedings of the 24th Australian Computer-Human Interaction Conference, OzCHI’12, pp. 212-221, Melbourne, Australia.

Hick W. 1952, "On the rate of gain of information", Quarterly Journal of Experimental Psychology, 4(1), pp. 11-26.

Hirsch M., Cheng J., Relss A., Sundholm M., Lukowicz P., and Amft O. 2014, *“Hands-Free Gesture Control with a Capacitive Textile Neckband”,* in Proceedings of the 2014 ACM International Symposium on Wearable Computers, ICWC’14, pp. 55-58, Seattle, WA, USA.

Hodge J., Balaam M., Hastings S., and Morrissey K. 2018, *"Exploring the Design of Tailored Virtual Reality Experiences for People with Dementia"*, in Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems, CHI'18, pp. 1-13.

Holmquist L., Schmidt A., and Ullmer B. 2004, "Tangible interfaces in perspective", Personal and Ubiquitous Computing, 8(5), pp. 291-293.

Hoober S. and Berkman E. 2011, *“Designing Mobile Interfaces: Patterns for Interaction Design”,* O’Reilly Media.

Iacoviello R. and Zappia D. 2020, "*HoloCities: A Shared Reality application for Collaborative Tourism*", in Proceedings of the IOP Conference Series: Materials Science and Engineering, IOP Publishing, 949(1).

Ishii H., Wisneski C., Brave S., Dahley A., Gorbet M., Ullmer B., and Yarin P. 1998, "ambientROOM: Integrating Ambient Media with Architectural Space", in Proceedings of CHI'98 Conference on Human Factors in Computing Systems, CHI'98, pp. 174-175.

Ishii H. and Ullmer B. 1997a, "Tangible bits: towards seamless interfaces between people, bits and atoms", in Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems, CHI'97, pp. 234-241, Atlanta, GA, USA.

Jacob R., Girouard A., Hirshfield L., Horn M., Shaer O., Solovey E., and Zigelbaum J. 2008, "*Reality-based interaction: a framework for post-WIMP interfaces*", in Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI'08, pp. 201-210, Florence, Italy.

Jackowski A., Gebhard M., and Gräser A. 2016, *"A Novel Head Gesture Based Interface for Hands-free Control of a Robot", in* IEEE 2016 International Symposium on Medical Measurements and Applications, MeMeA, pp. 1-6, Benevento, Italy.

Khademi M., Hondori H., and Lopes C. 2012, "*Optical Illusion in Augmented Reality*", in Proceedings of the 18th ACM Symposium on Virtual Reality Software and Technology, VRST'12, pp. 195-196, Toronto, Ontario, Canada.

Kim Y., Hassan A., White R., and Zitouni I. 2014, "*Modeling dwell time to predict click-level satisfaction*", in Proceedings of the 7th ACM International Conference on Web Search and Data Mining, WSDM'14, pp. 193-202, New York, New York, USA.

Kiyokawa K. 2012, *"Trends and Vision of Head Mounted Display in Augmented Reality"*, in 2012 International Symposium on Ubiquitous Virtual Reality, pp. 14-17.

Kojima M., Sugimoto M., and Nakamura A. 2006, "*Augmented Coliseum: An Augmented Game Environment with Small Vehicles*", in Proceedings of the First IEEE International Workshop on Horizontal Interactive Human Computer Systems, TABLETOP'06, pp. 6.

Kovacs R., Ofek E., Gonzalez F. M., Sin A., Marwecki S., Holz C, and Sinclair M. 2020, *"Haptic PIVOT: On Demand Handhelds in VR"*, in Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology, UIST2020, pp. 1046-1059, Minneapolis, USA.

Laamarti F., Eid M., and Saddik A. 2014, "*An overview of serious games*", International Journal of Computer Games Technology.

Lagerstam E., Olsson T., and Harvianinen T. 2012, "*Children and intuitiveness of interaction: a study on gesture-based interaction with augmented reality*", in Proceedings of the 16th International Academic MindTrek Conference, MindTrek 2012, pp. 169-174.

Lange B., Koenig S., Chang C., McConnell E., Suma E., Bolas M., and Rizzo A. 2012, *"Designing informed game-based rehabilitation tasks leveraging advances in virtual reality, Disability and Rehabilitation"*, 34(22), pp. 1863-1870.

Laver K., George S., Thomas S., Deutsch JE., and Crotty M. 2015, "*Virtual reality for stroke rehabilitation: an abridged version of a Cochrane review*", European Journal of Physical and Rehabilitation Medicine, 51(4), pp. 497-506.

Laviole J. and Hachet M. 2012, "*Demo: Spatial Augmented Reality for Physical Drawing*", UIST'12, Cambridge, Massachusetts, USA.

Law C., Roe P., and Zhang J. 2012, "*Using Mobile Technology and Augmented Reality to Increase Data Reliability for Environmental Assessment*", in Proceedings of the 25th annual ACM symposium on User interface software and technology, OZCHI'12, pp. 9-10, Melbourne, Victoria, Australia.

LEAP Motion 2013, "*LEAP Motion Sensor*", URL: https://www.leapmotion.com/, Last accessed 11/12/2021.

Lee G., Billinghurst M., and Kim G. 2004, "*Occlusion based Interaction Methods for Tangible Augmented Reality Environments*", in Proceedings of the 2004 ACM SIGGRAPH International Conference on Virtual Reality Continuum and Its Applications in Industry, pp. 419-426.

Lee J., Sinclair M., Gonzalez-Franco M., Ofek E., and Holz C. 2019, *"TORC: A Virtual Reality Controller for In-Hand High-Dexterity Finger Interaction"*, in Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems, CHI2019, pp. 1-13, Glasgow, Scotland, UK.

Lim C. and Jung H. 2013, *"A study on the military Serious Game*", Advanced Science and Technology Letters, Games and Graphics, vol. 39, pp. 73-77.

Lindner P., Miloff A., Hamilton W., Reuterskiold L., Andersson G., and Powers M. 2017, "*Creating state of the art, next-generation Virtual Reality exposure therapies for anxiety disorders using consumer hardware platforms: design considerations and future directions*", Cognitive Behaviour Therapy, 46(5), pp. 404-420.

Liou H., Yang S., Chen S., and Tarng W. 2017, "*The Influences of the 2D Image-Based Augmented Reality and Virtual Reality on Student Learning*", Journal of Education and Technology and Society, 20(3), pp. 110-121.

Liu C., Huot S., Diehl J., Mackay W., and Beaudouin-Lafon M., 2012, "*Evaluating the Benefits of Real-time Feedback in Mobile Augmented Reality with Hand-held Devices*", in Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 2973-2976, Austin, Texas, USA.

Liu W. 2010, "*Natural user interface- next mainstream product user interface*", in 2010 IEEE 11th International Conference on Computer-Aided Industrial Design & Conceptual Design 1, vol. 1, pp. 203-205.

Lopes P., You S., Cheng L., Marwecki S., and Baudisch P. 2017, "*Providing Haptics to Walls & Heavy Objects in Virtual Reality by Means of Electrical Muscle Stimulation*", in Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems, CHI'17, pp. 1471-1482, Denver, Colorado, USA.

Lu W., Duh B., Feiner S. 2012, "*Subtle Cueing for Visual Search in Augmented Reality*", in IEEE International Symposium on Mixed and Augmented Reality 2012, ISMAR, pp. 161-166, Atlanta, Georgia, USA.

Luebke D., Reddy M., Cohen J., Varshney A., Watson B., and Hubner R. 2003, "*Level of Detail for 3D Graphics*", Morgan Kaufmann.

Machkovech S. 2016, "*ARS Technica: Learning how to VR with Tilt Brush, HTC Vive's killer app*", ARS Technica, published 4/6/2016, URL: https://arstechnica.com/gaming/2016/04/learning-how-to-vr-with-tilt-brush-htc-vives-killer-app/, Last accessed on 11/12/2021.

Machkeovech S. 2020, *"Lost 'Sega VR' game unearthed, made playable on modern VR headsets"*, URL: https://arstechnica.com/gaming/2020/11/lost-sega-vr-game-unearthed-made-playable-on-modern-vr-headsets/, Last accessed 11/12/2021.

Magerkurth C., Cheok A., Mandryk R., and Nilsen T. 2005, "*Pervasive Games: Bringing Computer Entertainment Back to the Real World*", Computers in Entertainment (CIE), 3(3), pp. 4.

Magic Leap 2018, "*Magic Leap One*", URL: https://www.magicleap.com/magic-leap-one, Last accessed 11/12/2021.

Maples-Keller J., Bunnel B., Kim S., and Rothbaum B. 2017, "*The use of virtual reality technology in the treatment of anxiety and other psychiatric disorders*", Harvard Review of Psychiatry, 25(3), pp. 103-113.

Marin G., Dominio F., and Zanuttigh P. 2014, "*Hand gesture recognition with Leap Motion and Kinect Devices*", in 2014 IEEE International Conference on Image Processing, ICIP'2014, pp. 1565-1569.

McGill M., Boland D., and Murray-Smith R. 2015, "*A Dose of Reality: Overcoming Usability Challenges in VR Head-Mounted Displays*", in Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems, CHI'2015, pp. 2143-2152, Seoul, Republic of Korea.

McNamara A. 2011, "*Enhancing Art History Education Through Mobile Augmented Reality*", in Proceedings on the 10th International Conference on Virtual Reality Continuum and Its Applications in Industry, VRCAI 2011, pp. 507-512, Hong Kong, China.

Medeiros D., Sousa M., Mendes D., Raposo A., and Jorge J. 2016, "*Perceiving Depth: Optical versus Video See-through*", in Proceedings of the 22nd ACM Conference on Virtual Reality Software and Technology, VRST'16, pp. 237-240, Garching bei Munchen, Germany.

Merhi O., Faugloire E., Flanagan M., and Stoffregen T. 2007, "*Motion sickness, console video games, and head-mounted displays*", Human Factors, 49(5), pp. 920-934.

Microsoft 2016, "*Microsoft HoloLens*", URL: https://www.microsoft.com/en-us/hololens, Last accessed 11/12/2021.

Milgram P., Takemura H., Utsumi A., and Kishino F. 1995, "*Augmented reality: A class of displays on the reality-virtuality continuum*", in Telemanipulator and Telepresence Technologies, International Society for Optics and Photonics, vol. 2351, pp. 282-293.

Mitchell P. and Wilkinson B. 2016, "*Periphery triggered menus for head mounted menu interface interactions*", in Proceedings of the 28th Australian Conference on Computer-Human Interaction, OzCHI'16, pp. 30-33, Tasmania, Australia.

Mirzaei M., Ghorshi S., and Mortazavi M. 2012, "*Using Augmented Reality and Automatic Speech Recognition Techniques to Help Deaf and Hard of Hearing People*", in Proceedings of the 2012 Virtual Reality International Conference, Laval Virtual VRIC'12, pp. 1-4, Laval, France.

Mohammed-Amin R., Levy R., and Boyd J. 2012, "*Mobile Augmented Reality for Interpretation of Archaeological Sites*", in Proceedings of the 2nd International ACM Workshop on Personalized Access to Cultural Heritage, PATCH'12, pp. 11-14, Nara, Japan.

Mohan A., Woo G., Hiura S., Smithwick Q., and Raskar R. 2009, "*Bokode: Imperceptible Visual tags for Camera Based Interaction from a Distance*", ACM SIGGRAPH 2009, pp. 1-8.

Morde A., Correa C., Hou J., Ganapathy S., Krebs A., Marsic I., Bouzit M., and Rabiner L. 2004, "*Asymmetric Collaboration through Tele-presence*", in Proceedings of the 2004 ACM SIGMM workshop on Effective Telepresence, ETP'04, pp. 57-58, New York, New York, USA.

Morency L. and Darrel T. 2006, "*Head Gesture Recognition in Intelligent Interfaces: The Role of Context in Improving Recognition*", in Proceedings of the 11th International Conference on Intelligent User Interfaces, IUI'06, pp. 32-38, Sydney, Australia.

Morimoto C. and Mimica M. 2005, "*Eye gaze tracking techniques for interactive applications*", Computer Vision and Image Understanding, 98(1), pp. 4-24.

Mortara M., Catalano C., Bellotti F., Fiucci G., Houry-Panchetti M., and Petridis P. 2014, "*Learning cultural heritage by serious games*", Journal of Cultural Heritage, 15(3), pp. 318-225.

Moyle W., Jones C., Dwan T., and Petrovich T. 2018, *"Effectiveness of a Virtual Reality Forest on People with Dementia: A Mixed Methods Pilot Study"*, The Gerontologist, 58(3), pp. 478-487.

Mulloni A., Seichter H., and Schmalstieg D. 2011, "*Handheld Augmented Reality Indoor Navigation with Activity-Based Instructions*", in Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services, MobileHCI 2011, pp. 211-220, Stockholm, Sweden.

Munafo J., Diedrick M., and Stoffregen T. 2017, *"The virtual reality head-mounted display Oculus Rift induces motion sickness and is sexist in its effects"*, Experimental Brain Research, 235(3), pp. 889-901.

Mütterlein J., Jelsch S., and Hess T. 2018, "*Specifics of Collaboration in Virtual Reality: How Immersion Drives the Intention to Collaborate*", PACIS2018 Proceedings, pp. 318.

Narumi T., Ban Y., Kajinami T., Tanikawa T., and Hirose M. 2012, "*Augmented Perception of Satiety: Controlling Food Consumption by Changing Apparent Size of Food with Augmented Reality*", in Proceedings of the SIGCHI conference on human factors in computing systems, CHI 2012, pp. 109-118, Austin, Texas, USA.

Navdy 2017, "*Navdy HUD*", URL: http://mynavdy.com.au/, Last accessed 24/12/2018.

Niantic 2012, "*Ingress Prime*", URL: https://ingress.com/, Last accessed 11/12/2021.

Niantic 2014, "*Pokémon GO*", URL: https://pokemongolive.com/en/, Last accessed 11/12/2021.

Normand J., Servieres M., and Moreau G. 2012, "*A new typology of augmented reality applications*", in Proceedings of the 3rd Augmented Human International Conference, AH'12, pp. 1-8, Megeve, France.

Oculus 2012, "*Kickstarter: Oculus Rift: Step Into the Game*", URL: https://www.kickstarter.com/projects/1523379957/oculus-rift-step-into-the-game, Last accessed 11/12/2021.

Oculus 2018, "*Oculus Rift*", URL: https://www.oculus.com/, Last accessed 11/12/2021.

Oda O. and Feiner S. 2012, "*3D Referencing Techniques for Physical Objects in Shared Augmented Reality*", IEEE International Symposium on Mixed and Augmented Reality 2012, ISMAR, pp. 207-215, Atlanta, Georgia.

Olsson T., Lagerstam E., Karkkainen T., and Väänänen-Vainio-Mattila K. 2011, "*Expected user experience of mobile augmented reality services: a user study in the context of shopping centres*", Mobile HCI 2011, 13th International Conference.

Olsson T., Kärkkäinen, Lagerstam E., and Ventä-Olkkonen L. 2012, "*User evaluation of mobile augmented reality scenarios*", Journal of Ambient Intelligence and Smart Environments, 4(1), pp. 29-47.

Olwal A. 2008, "*Unencumbered 3D Interaction with See-through Displays*", in Proceedings of the 5th Nordic Conference on Human-Computer Interaction: Building Bridges, NordiCHI 2008, pp. 527-530, Lund, Sweden.

Ott M. and Freina 2015, "*A literature review on immersive virtual reality in education: State of the art and perspectives*", Conference proceedings of eLearning and Software for Education (eLSE), pp. 133-141.

Owlchemy Labs 2016, "*Job Simulator*", URL: https://store.steampowered.com/app/448280/Job_Simulator/, Last accessed 11/12/2021.

Pahud M., Hinckley K., Iqbal S., Sellen A., and Buxton W. 2013, "*Toward Compound Navigation Tasks on Mobiles via Spatial Manipulation*", in Proceedings of the 15th International Conference on Human-Computer interaction with mobile devices and services, MobileHCI 2013, pp. 113-122, Munich, Germany.

Pallavicini F., Ferrari A., Zini A., Garcea G., Zanacchi A., Barone G., and Mantovani F. 2017, "*What Distinguishes a Traditional Gaming Experience from One in Virtual Reality? An Exploratory Study*", Springer, AHFE2017, Advances in Human Factors in Wearable Technologies and Game Design, pp. 225-231.

Pantelidis V. 2009, *"Reasons to Use Virtual Reality in Education and Training Courses and a Model to Determine When to Use Virtual Reality",* Themes in Science and Technology Education, vol. 2, pp. 59-70.

Parkin S. 2015, "*How VR is training the perfect soldier*", Wearable, published 31/12/2015, URL: https://www.wareable.com/vr/how-vr-is-training-the-perfect-soldier-1757, Last accessed 11/12/2021.

Petry B. and Huber J. 2015, "*Towards Effective Interaction with Omnidirectional Videos Using Immersive Virtual Reality Headsets*", in Proceedings of the 6th Augmented Human International Conference, AH'15, pp. 217-218, Singapore, Singapore.

Pfeuffer K., Mecke L., Rodriguez S.D., Hassib M., Maier H., and Alt F. 2020, "*Empirical Evaluation of Gaze-enhanced Menus in Virtual Reality*", in Proceedings of the 26th ACM Symposium on Virtual Reality Software and Technology, VRST'20, pp. 1-11, New York, New York, USA.

Piekarski W. and Thomas B. 2002, "*ARQuake: The Outdoor Augmented Reality Gaming System*", Communications of the ACM, 45(1), pp. 36-38.

Piper B., Ratti C., and Ishii H. 2002, "Illuminating clay: a 3-D tangible interface for landscape analysis", in Proceeding of the SIGCHI Conference on Human Factors in Computing Systems, CHI'02, pp. 355-362, Minneapolis, Minnesota, USA.

Prince S., Cheok A., Farbiz F., Williamson T., Johnson N., Billinghurst M., and Kato H. 2002, "*Real-Time 3D Interaction for Augmented and Virtual Reality*", in ACM SIGGRAPH 2002 Conference Abstracts and Applications, pp. 238.

Qiao C., Wang Y., Weng D., and Qi H. 2011, "*An Augmented Reality Based Teeth Shade Matching System*", in Proceedings of the 10th International Conference on Virtual Reality Continuum and Its Applications in Industry, VRCAI 2011, pp. 371-374, Hong Kong, China.

Reisman J., Davidson P., and Han J. 2009, *"Generalizing Multi-Touch Direct Manipulation"*, in SIGGRAPH 2009 Talks, pp. 1.

Rift Info 2016, "*Oculus Rift Specs – DK1 vs DK2 Comparison*", URL: https://riftinfo.com/oculus-rift-specs-dk1-vs-dk2-comparison, Last accessed 11/12/2021.

Rodrigues F., Sato F., Botega L., and Oliveira A. 2012, "*Augmented Reality and Tangible User Interfaces Integration for Enhancing the User Experience*", in Proceedings of the 11th ACM SIGGRAPH International Conference on Virtual Reality Continuum and Its Applications in Industry, VRCAI 2012, pp. 67-70, Singapore.

Romero M., Usart M., and Ott M. 2014, "*Can Serious Games Contribute to Developing and Sustaining 21st Century Skills?*", Games and Culture, 10(2), pp. 148–177.

Roupé M., Bosch-Sijtsema P., and Johansson M. 2014, "*Interactive navigation interface for Virtual Reality using the human body*", Computers, Environment and Urban Systems, vol. 43, pp. 42-50.

Ruban T. and Wood J. 2016, "*Recognizing Head Gestures for Head-mounted Displays*", Standford EE 267 Virtual Reality Course Report.

Safikhani S., Holly M., and Pirker J. 2020, "*Work-in-Progress--Conceptual Framework for User Interface in Virtual Reality*", in Proceedings of the 6th International Conference of the Immersive Learning Research Network (iLRN), pp. 332-335, San Luis Obispo, CA, USA.

Saidi H., Dubois E. and Serrano M. 2021, "*HoloBar: Rapid Command Execution for Head-Worn AR Exploiting Around the Field-of-View Interaction*", in Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems, CHI'21, Yokohama, Japan.

Sailsbury D., Dahdah M., Driver S., Parsons T., and Richter K. 2016, *"Virtual Reality and Brain Computer Interface in Neurorehabilitation"*, in Baylor University Medical Center Proceedings, 29(2), pp. 124-127.

Sambrooks L. and Wilkinson B. 2016, "*Designing HARATIO: a novice AR authoring tool*", in Proceedings of the 28th Australian Conference on Computer-Human Interaction, OzCHI'16, pp. 175-179, Launceston, Australia.

Samsung 2015, “*Samsung Gear VR*”, URL: https://www.samsung.com/global/galaxy/gear-vr/, Last accessed 11/12/2021.

Sano A. 2011, “*An Application for Creating Full-scale Augmented Reality Content without 3D Modelling Skills*”, in Proceedings of the 2011 ACM symposium on The role of design in UbiComp research & practice, RDURP’11, pp. 19-24, Beijing, China.

Santos B. S., Dias P., Pimentel A., Baggerman J., Ferreira C., Silva S., and Madeira J., “*Head-mounted display versus desktop for 3D navigation in virtual reality: a user study*”, Multimedia Tools Applications 2009, 41(1), pp. 161-181.

Sauro J. 2011, “*Measuring Usability with the System Usability Scale (SUS)*”, URL: https://measuringu.com/sus/, Last accessed on 11/12/2021.

Schneiderman B. 1992, *“Designing the user interface: strategies for effective human-computer interaction 2nd Edition”*, Addison-Wesley 1992.

Scott MacKenzie I. 1992, *“Fitts' Law as a Research and Design Toolin Human-Computer Interaction”*, Human-Computer Interaction, 7(1), pp. 91-139.

Seo J, Smith B., Cook M., Malone E., Pine M., Leal S., Bai Z., and Suh J. 2017, “*Anatomy Builder VR: Applying a Constructive Learning Method in Virtual Reality Canine Skeletal System*”, Springer, In International Conference on Applied Human Factors and Ergonomics, Advances in Human Factors in Training, Education, and Learning Sciences, AHFE2017, pp. 245-252.

Serrano M., Ens B., and Irani P. 2014, “*Exploring the Use of Hand-To-Face Input for Interacting with Head-Worn Displays*”, in Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI’2014, pp. 3181-3190, Toronto, ON, Canada.

Seth A., Vance J.M., and Oliver J.H., 2011, “*Virtual reality for assembly methods prototyping: a review*”, Springer, Virtual Reality (2011), 15(1), pp. 5-20.

Sherman W. and Craig A 2002, “*Understanding Virtual Reality: Interface, Application, and Design*”, Elsevier Science 2003.

Sinko M. and NullSpace VR 2018, “*Kickstarter: Post mortem report and the conclusion of Hardlight*”, URL: https://www.kickstarter.com/projects/morgansinko/hardlight-vr-suit-dont-just-play-the-game-feel-it/posts/2292833, Last accessed 11/12/2021.

Sluis R., Weevers I., van Schijndel C., Kolos-Mazuryk L., Fitrianie S., and Martens J. 2004, *“Read-It: Five-to-seven-year-old children learn to read in a tabletop environment”*, in Proceedings of the 2004 Conference on Interaction Design and Children: Building a Community, IDC 2004, pp. 73-80, College Park, Maryland, USA.

Sony 2018, “*Playstation VR*”, URL: https://www.playstation.com/en-au/explore/playstation-vr/, Last accessed 11/12/2021.

Špakov O. and Majaranta P. 2012, “*Enhanced Gaze Interaction Using Simple Head Gestures*”, in Proceedings of the 2012 ACM Conference on Ubiquitous Computing, UbiComp’12, pp. 705-710, Pittsburgh, USA.

SRI International 2018, *"1968 'Mother of All Demos' by SRI's Doug Engelbart and Team"*, Youtube URL: https://www.youtube.com/watch?v=B6rKUf9DWRI, Last accessed 11/12/2021.

Stiefelhagen R., Yang J., and Waibel A. 1997, "*A model-based gaze tracking system*", International Journal on Artificial Intelligence Tools, 6(2), pp. 193-209.

Sturman D. and Zeltzer D. 1994, *"A Survey of Glove-based Input"*, IEEE Computer Graphics and Applications, 14(1), pp. 30-39.

Sutherland I. 1968, *"A head-mounted three dimensional display"*, in Proceedings of the December 9-11, 1968, Fall Joint Computer Conference, Part 1, AFIPS'68, pp.757-764.

Superhot Team 2017, *"Superhot VR"*, URL: https://store.steampowered.com/app/617830/SUPERHOT_VR/, Last accessed 11/12/2021.

Tanaka N. and Takagi H. 2004, *"Virtual Reality Environment Design of Managing Both Presence and Virtual Reality Sickness"*, Journal of Physiological Anthropology and Applied Human Science, 23(6), pp. 313-317.

Theoktisto V. and Fairén M. 2005, *"Enhancing collaboration in virtual reality applications"*, Computers & Graphics, 29(5), pp. 704-718.

Thomas B., Close B., Donoghue J., Squires J., De Bondi P., Morris M., and Piekarski W. 2000, "*ARQuake: An Outdoor/Indoor Augmented Reality First Person Application*", In Digest of Papers Fourth International Symposium on Wearable Computers IEEE 2000, pp. 139-146.

Thomas B., Close B., Donoghue J., Squires J., De Bondi P., and Piekarski W. 2002, "*First Person Indoor/Outdoor Augmented Reality Application: ARQuake*", Personal and Ubiquitous Computing 2002, 6(1), pp. 75-86.

Thomas B. 2012, "*A Survey of Visual, Mixed, and Augmented Reality Gaming*", Computers in Entertainment (CIE), 10(1), pp. 1-33.

Tu H., Huang S., Yuan J., Ren X., and Tian F. 2019, "*Crossing-Based Selection with Virtual Reality Head-Mounted Displays*", in Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems, CHI'19, pp. 1-14, Glasgow, Scotland, UK.

Ullmer B. and Ishii H. 1997b, "The metaDESK: Models and Prototypes for Tangible User Interfaces", in Proceedings of the 10th Annual ACM Symposium on User Interface Software and Technology, UIST'97, pp. 223-232, Banff, Alberta, Canada.

Ullmer B. and Ishii H. 2000, "Emerging frameworks for tangible user interfaces", IBM Systems Journal, 39(3+4), pp. 915-931.

Valve 2018, "*Steam Storefront: Virtual Reality on Steam*", URL: https://store.steampowered.com/vr/, Last accessed 11/12/2021.

van Dam A. 2000, "*Beyond WIMP*", IEEE Computer Graphics and Applications, 20(1), pp. 50-51.

Vertigo Games 2016, "*Arizona Sunshine*", URL: https://store.steampowered.com/app/342180/Arizona_Sunshine/, Last accessed 11/12/2021.

Vincent T., Nigay L., and Kurata T. 2013, "*Handheld Augmented Reality: Effect of registration jitter on cursor-based pointing techniques*", in Proceedings of the 25th Conference on l'Interaction Homme-Machine, IHM'13, pp. 1-6, Bordeaux, France.

Virtuix 2015, "*Virtuix Omni*", URL: http://www.virtuix.com/, Last accessed 11/12/2021.

Vive 2018, "*HTC Vive*", URL: https://www.vive.com/au/product/, Last accessed 11/12/2021.

VRChat Inc 2017, "*VR Chat*", URL: https://store.steampowered.com/app/438100/VRChat/, Last accessed 11/12/2021.

Wang Y., Hu Y., and Chen Y. 2021, "*An experimental investigation of menu selection for immersive virtual environments: fixed versus handheld menus*", Virtual Reality, 25(2), pp. 409-419.

Weiser M. 1991, "*The computer for the twenty-first century*", Scientific American, 265(3), pp. 94-104.

Woodward C., Honkamaa P., Jappinen J., and Pyokkimies E. 2004, "*CamBall – Augmented Networked Table Tennis Played with Real Rackets*", in Proceedings of the 2004 ACM SIGCHI International Conference on Advances in Computer Entertainment Technology, ACE'04, pp. 275-276, Singapore.

Wrzesien M., Burkhardt J., Raya M., and Botella C. 2011, "*Mixing Psychology and HCI in Evaluation of Augmented Reality Mental Health Technology*", CHI'11 Extended Abstracts on Human Factors in Computing Systems, CHI 2011, pp. 2119-2124, Vancouver, BC, Canada.

Zarzycki A. 2012, "Urban Games: Application of Augmented Reality", in SIGGRAPH Asia 2012 Symposium on Apps, pp. 1, Singapore.

Zhang R. and Kuhl S. 2013, "*Human Sensitivity to Dynamic Rotation Gains in Head-Mounted Displays*", in Proceedings of the ACM Symposium on Applied Perception, SAP 2013, pp. 71-74, Dublin, Ireland.

Zhang Y., Yang Y., Feng S., Qi J., Li W., and Yu J. 2020, "*The Evaluation on Visual Fatigue and Comfort Between the VR HMD and the iPad*", in Proceedings of Advances in Physical, Social and Occupational Ergonomics, AHFE 2020, pp. 213-219.

Zhou F., Duh H., and Billinghurst M. 2008, "*Trends in Augmented Reality Tracking, Interaction and Display: A Review of Ten Years of ISMAR*", in 2008 7th IEEE/ACM International Symposium on Mixed and Augmented Reality, pp. 193-202, Cambridge, UK.

Zhu Z. and Ji Q. 2004, "*Eye and gaze tracking for interactive graphic display*", Machine Vision and Applications, 15(3), pp. 139-148.

Zyda M. 2005, "*From Visual Simulation to Virtual Reality to Games*", Computer IEEE 2005, 38(9), pp. 25-32.

A Appendix A: Additional First Experiment Details

Additional details were left out of the principal dissertation document for each experiment to keep content concise and on topic. These include some implementation-specific information not directly essential to discussing the Periphery Vision Menu System (PVSM). The information moved to these appendices may help replicate aspects of the experiments. This first appendix covers implementation information for the first experiment in A.1 and provides additional documents given to participants in A.2. You can find similar sections for the second experiment in Appendix B and the third in Appendix C.

A.1 First Experiment Additional Information

This section covers a specific set of additional implementation details that may be useful for replicating the experiment that was not necessary for understanding the experiments or their results.

The **`CubeBehaviour`** in A.1.1 defines the class used for controlling all functionality related to the Cube elements. All tasks for the first experiment involved using the Cubes, so they were an integral part of the experiments and merited a detailed class overview.

Level Management in A.1.2 introduces the three sections that came after it, discussing level swapping in A.1.3, a summary of most core code class files in A.1.4, and a summary of the hierarchy of game objects in A.1.5.

The following three sections discuss some of the specific implementations for tasks. An expansion of the specific implementation for Task 3 and Task 4 with the **`CubeBehaviour`** are shown in A.1.6. The Unity specific implementation of the circular menu used is expanded upon in A.1.7. A similar type of discussion is also provided for the static version of the menus in A.1.8.

A.1.1 Cube Behaviour

```
Cube Behaviour
```

```
+ enum CubeType { Static, MovingToPoint,
Draggable };
+ enum SelectionState { Waiting, Selecting,
Selected };
+ enum SelectionMode { Time, HoverSelect };

// Game Reference
+ GameObject levelObject;

// Colouring
+ Texture2D defaultTexture; // red texture
+ Texture2D selectingTexture; // orange
texture
+ Texture2D selectedTexture; // green texture
+ Texture2D[] extraColours;

// Object status and children
+ bool objectEnabled;
+ bool cubeObjComplete;
- Vector3 startPos;
- MenuBehaviour childMenu;

// Movement
+ CubeType cubeType;
+ GameObject nextMoveObject;
+ float minMoveSpeed;
+ float maxMoveSpeed;
- Vector3 nextMoveV3;
- float curMoveSpeed;
```

```
// Selection
+ bool canBeSelected;
+ float TIME_TO_SELECT;
- SelectionState selectionState;
- float selectionProgress;
- float progressTime;
- SelectionMode selectionMode;

// Methods
+ void Start()
+ void performUpdate(float
deltaTime)
+ void
beginSelection(SelectionMode mode)
+ void cancelSelection()
+ void removeSelection()
+ void completeSelection()
+ SelectionState
getSelectionState()
+ bool getObjComplete()
+ void setEnabledState(bool
newState)
+ bool getObjEnabled
+ void OnCollisionEnter(Collision
other)
+ void setExtraColourTex(int
textureID)
```

Figure A.1: CubeBehaviour Class Definition

The cube object prefab was used consistently throughout the first experiment, so it is worthwhile describing the properties that it could provide. The instance variables (and enum definitions) that made up the **`CubeBehaviour.cs`** class can be seen in Figure A.1. It has been broken down into these clearly marked sections: colouring, movement, selection, object status, and children. There were three colours indicating status: red (not selected), orange (hovering awaiting selection), green (already selected).

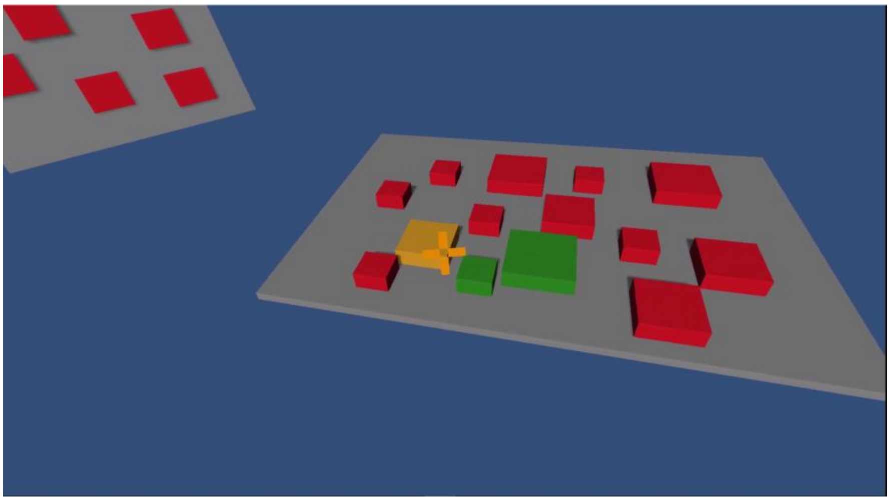


Figure A.2: Cube Colour Example

The three different colours can be seen in Figure A.2. These could also be described as the enumerated selection states described as **`waiting`**, **`selecting`**, **`selected`**. The **`CubeType`** was perhaps the most important attribute, as it determined if the cube was **`static`** (typically just a cube to be selected only), **`moving to point`** (meaning the cube would move between waypoint nodes, as seen in Task 2 and 3), or **`draggable`** (allowing the cube to be picked up in Task 5, 6, and 7). The third enumeration of **`SelectionMode`** clarified to the object if the selection would happen based on time or if it would be completed when a click/tap occurred. Another feature of the cube behaviour is the selection related to the movement speed for **`MovingToPoint`** type cubes. The variable **`selectionProgress`** differed from **`progressTime`** by representing progress as a percentage from 0 to 1. The formula to change move speed was applied once a selection was in progress and seen in Listing A.1.

```
curMoveSpeed = (maxMoveSpeed - minMoveSpeed)
                   * (1 - selectionProgress) + minMoveSpeed;
```

Listing A.1: Current Speed of Cube

This technique made it increasingly easier to keep track of selecting each cube while they moved. Moving away and failing to select a cube would reset the speed back to max again. Finally, because the cubes were being reused three times (once for each interaction method), they needed a way to reset. The reset was triggered when the objects were called to be enabled. The **`startPos`** would be used to reset the position along with other minor object defaults correctly.

A.1.2 Level Management

The following sections will look at a few of the interconnecting features for how moving between tasks was handled. These topics cover the tools for changing between levels, the class files used to give behaviours to all the game objects, and finally, an overview of how the scene's hierarchy was structured within Unity.

One important feature of the approach taken with level management was the use of a partial game loop. Game loops structure the order of how objects run within a scene. Depending on the type of application developed in Unity, game objects normally will exist independently with references to other objects. A game loop was used to order the update code between classes to ensure synchronisation for replay data and consistency between participants. **`MouseBehaviour.cs`** was the central controller with **`LevelManager.cs`** and **`LevelBehaviour.cs`** controlling many objects specific to levels.

A.1.3 Level Change Interactions

These level change interactions were configured to handle scenarios for debugging, completing the experiment as normal (with level complete and next level transitions), and to handle cases where anything may force the participant to pause the experiment or restart in cases of unexpected failure.

- Level Complete: The normal way to transition between levels. Participants could automatically transition between the levels and in between level complete screens at their own pace. After completing each input method, a manual keypress was required to continue to the end input method. Mostly to prevent the participant from not realising they needed to swap how they were interacting.
- Keyboard "F1": Next Level. It could be used if errors occurred requiring restart without needing to complete all the tasks again. Or for debug testing of specific levels.
- Keyboard "F3": Previous Level. Mostly for if F1 had been used too many times.
- Keyboard "F5": Reset to the First Level.
- Keyboard "F8": Set input mode to HMD Only.
- Keyboard "F9": Set input mode to Mouse.

- Keyboard “F10”: Set input mode to Mobile.
- Keyboard “F12”: Restart Game. Restarts the scene resetting everything.

The objects for levels were stored in a very far off place in the distance to facilitate level transitions seamlessly. The rendering on some objects was enabled and could be seen as a small blip in the distance. The objects in the distance could not be interacted with because of the disabling in the **`LevelBehaviour`** script.

1) Check new level is a valid level index. Do nothing if not.
2) Move the current level to the hidden coordinates (1000,1000,1000) and disable the **`LevelBehaviour`**’s script.
3) Move the next level to (0,0,0) and enable the **`LevelBehaviour`**’s script.
4) Show/Hide the information panel depending on available content to show for the level.

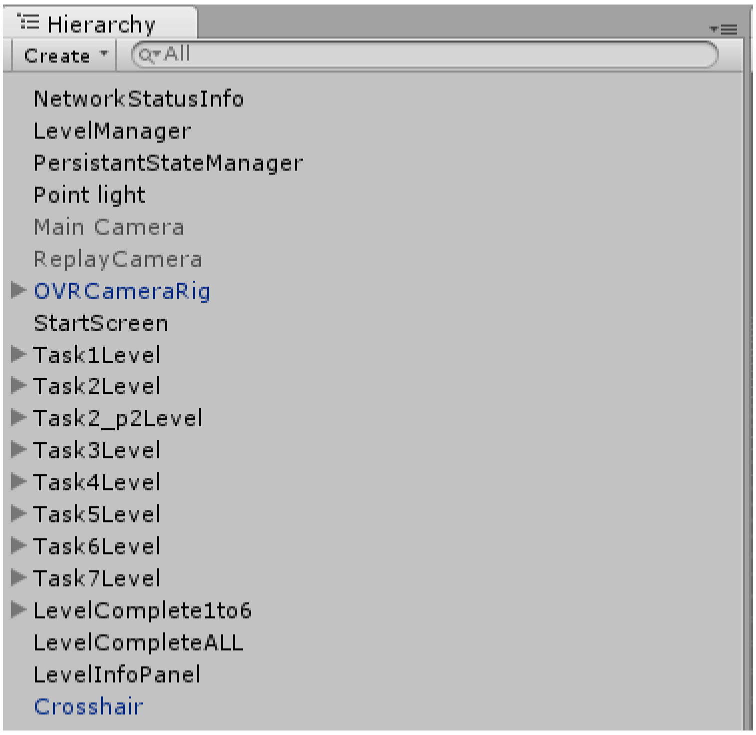
A.1.4 Class Files

- **`CubeBehaviour.cs`**: Used by Cube game objects. See section appendix A.1.1.
- **`CubeMatchBehaviour.cs`**: Used for Task 6 and 7 to pair with a **`CubeBehaviour`**.
- **`EndPointBehaviour.cs`**: Used for Task 5 to represent the end node.
- **`ExtraDataRecorder.cs`**: Used by **`LevelManager`** to record log files.
- **`LevelBehaviour.cs`**: Manages Cube and other objects related to a specific level.
- **`LevelManager.cs`**: Manages the collection of LevelBehaviours for all levels.
- **`MenuBehaviour.cs`**: Used for Task 6 and 7 to represent the menus.
- **`MenuBtnBehaviour.cs`**: Used for Task 6 and 7 by the **`MenuBehaviour`** for each button.
- **`ModeSelect.cs`**: Entry point for the program with command-line arguments to load replay, debug modes, or the version for participants to use.
- **`MouseBehaviour.cs`**: Primary controller for input attached to the camera. Tracked all forms of input and called appropriate other classes and objects to relay actions. A more appropriate name such as “InputBehaviour” may have made more sense, but this class started with the name before it was decided to use one class for

all three inputs.

- **PathCornerBehaviour.cs**: Used by Task 3 and 4 to move **CubeBehaviour** on a path.
- **PeristantStateBehaviour.cs**: Maintains the **ReplayDatabase** and **ExtraData** logs between scene transitions. The transitions would occur any time the input device was changed to reset all the levels to defaults.
- **ReplayDatabase.cs**: Stores a collection of **ReplayEvent**s. See section 3.2.13 for details.
- **ReplayEvent.cs**: Stores a single frames event/s. See section 3.2.13 for details.
- **ReplayReader.cs**: Modified version of **MouseBehaviour** using **ReplayEvents** instead of user input to allow replay.
- **ServerBehaviour.cs**: Listens for events from the mobile input for use by **MouseBehaviour**.
- **StartScreenBehaviour.cs**: Just provided some minor visual configuration.
- **WallBehaviour.cs**: Used by Task 5 for the wall collision interactions.

A.1.5 Unity Game Object Hierarchy



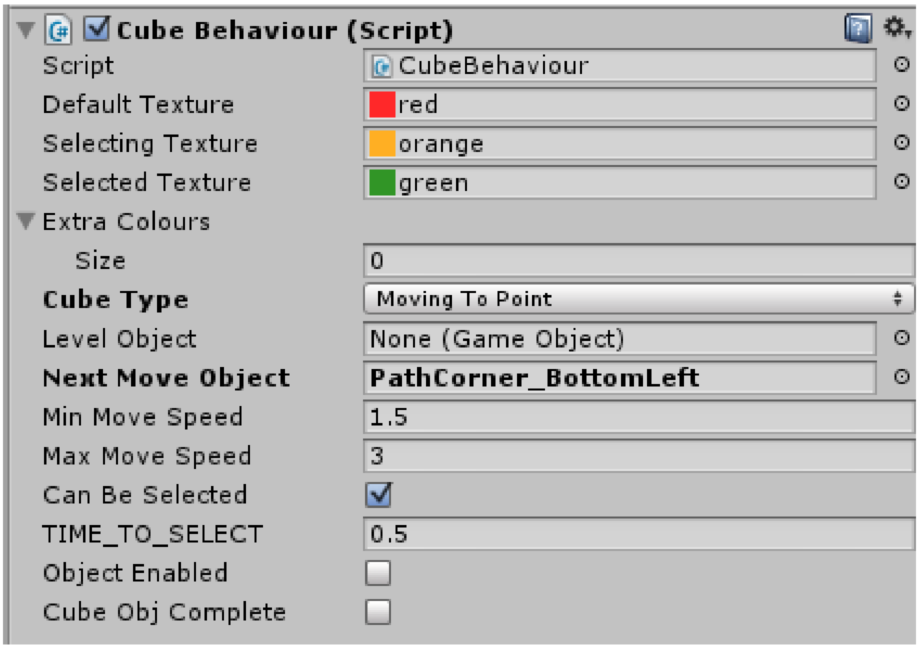
NetworkStatusInfo
LevelManager
PersistantStateManager
Point light
Main Camera
ReplayCamera
▶ OVRCameraRig
StartScreen
▶ Task1Level
▶ Task2Level
▶ Task2_p2Level
▶ Task3Level
▶ Task4Level
▶ Task5Level
▶ Task6Level
▶ Task7Level
▶ LevelComplete1to6
LevelCompleteALL
LevelInfoPanel
Crosshair

Figure A.3: First Experiment Unity Hierarchy

In Figure A.3, the hierarchy for the experiment can be seen. Each game object served a specific purpose within the scene. Having all the objects already loaded into the scene simplified, ensuring everything was active from the beginning of each experiment. The purpose of each object should be reasonably obvious from the naming. The objects related to levels included all the objects starting with "Task", **LevelComplete1to6**, **LevelCompleteALL**, **StartScreen**, and the **LevelInfoPanel**. The **LevelManager** object managed these. **NetworkStatusInfo** listened for messages from the mobile input. The **PersistantStateManager** was maintained between reloads to maintain all the information saved for experiments and remember which input method would be next after the reload. A single point light was used to illuminate the scene and create some minor shadows for immersion. These were rendered by one of the three cameras. **OVRCameraRig** was used whenever the HMD mode was active and had one camera for each eye and a third central camera to calculate ray casts for targeting. This object also used the **MouseBehaviour** class to run the whole application. The **Main Camera** was only used when a view mode was enabled for debugging without a HMD, and **ReplayCamera** was used for simulating replays from previous uses of the application.

A.1.6 Configuration of Cubes for the Multiple Object Interaction Tasks

Figure A.4 shows the pane that appears in the Unity inspector when any **CubeBehaviour** object is selected. Most of the settings here are standard for all cubes based on the prefabs. The prefab created for all cubes included the red, orange, and green textures, defaulted to being selectable with 0.5 second select time, the object-enabled flag defaulted to off, and by default not complete (meaning "selected" in most cases). The differences here for Task 3 are slightly bolded. Instead of a **Static** cube type, the **Moving To Point** meant the next move object property would be used. **PathCorner_BottomLeft** is another different game object. On each corner of the square are hidden objects with references pointing to the next ones forming the moving square when taken in sequence. Cube movement occurred by updating the position of cubes with the **curMoveSpeed** value as a movement toward the next corner objective. Once both cubes had been selected, this would take the user to the level complete screen.



☑ **Cube Behaviour (Script)**

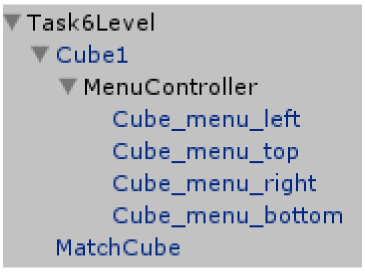
Script	CubeBehaviour
Default Texture	🟥 red
Selecting Texture	🟧 orange
Selected Texture	🟩 green
Extra Colours	
Size	0
Cube Type	Moving To Point
Level Object	None (Game Object)
Next Move Object	**PathCorner_BottomLeft**
Min Move Speed	1.5
Max Move Speed	3
Can Be Selected	☑
TIME_TO_SELECT	0.5
Object Enabled	□
Cube Obj Complete	□

Figure A.4: Task 3 Cube Behaviour Unity Inspector Example

A.1.7 Configuration of Object Matching for the Circular Menu Task

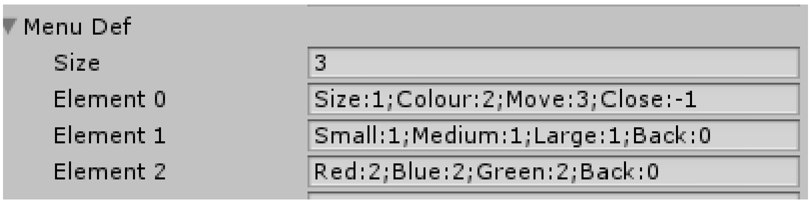
The object hierarchy used to represent each pair of cubes, and the final location it would end at can be seen in Figure A.5 (over). The **Cube1** object has a hidden **MenuController** used to reveal the four **Cube_menu** options. And a **MatchCube** for comparison between **Cube1** and **MatchCube** for determining if the task has been completed. As part of the **MenuController**, the buttons performed their functionality in the same way as the Cubes for selection. When any button was selected, the object would flag there had been a completed selection. Then the **MenuController** would determine the correct action to take based on a Menu Action and a Button Action. The **MenuController** assigned these properties with syntax seen in Figure A.6. Menu index 0 was the default when an object was selected. Then there were four pairs of data to define buttons with a String for the Button Action such as “Size” or “Blue”, a colon, then a Menu Action and semi-colon. When using the **applyAction** method, a special action would occur when the action was any of: “Small”, “Medium”, “Large”, “Red”, Blue”, “Green”, “Move”. These would change the properties of the parent Cube1 object. Then once any transformation had been applied, the **setMenu** method would be called to either change the menu to a different index from the

`MenuController`’s **`Menu Def`** or any number outside the range (0 to 2) of menu options would cause the menu to close.



```
▼ Task6Level
  ▼ Cube1
    ▼ MenuController
        Cube_menu_left
        Cube_menu_top
        Cube_menu_right
        Cube_menu_bottom
    MatchCube
```

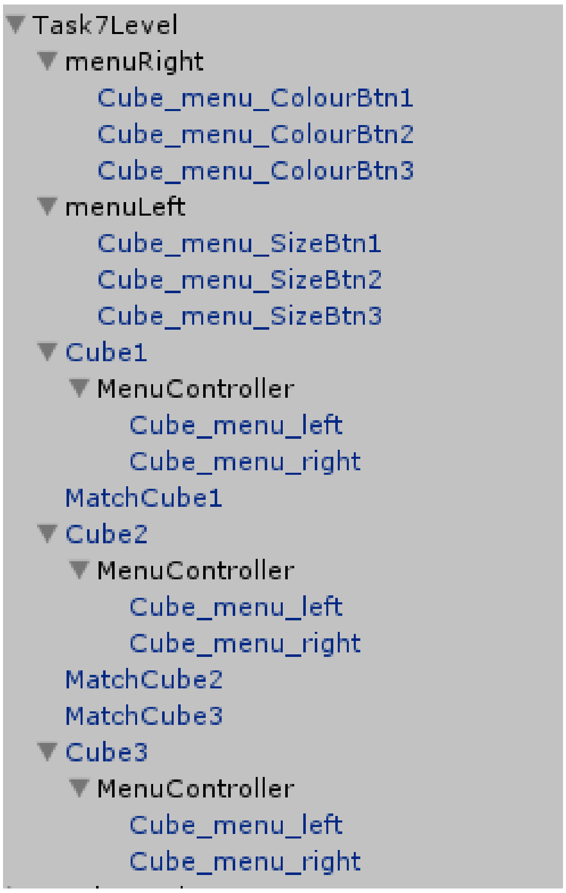
Figure A.5: Unity Hierarchy Task 6



▼ Menu Def	
Size	3
Element 0	Size:1;Colour:2;Move:3;Close:-1
Element 1	Small:1;Medium:1;Large:1;Back:0
Element 2	Red:2;Blue:2;Green:2;Back:0

Figure A.6: MenuController Menu Definitions

A.1.8 Configuration of Object Matching for the Static Periphery Menu Task



```
▼ Task7Level
  ▼ menuRight
      Cube_menu_ColourBtn1
      Cube_menu_ColourBtn2
      Cube_menu_ColourBtn3
  ▼ menuLeft
      Cube_menu_SizeBtn1
      Cube_menu_SizeBtn2
      Cube_menu_SizeBtn3
  ▼ Cube1
    ▼ MenuController
        Cube_menu_left
        Cube_menu_right
    MatchCube1
  ▼ Cube2
    ▼ MenuController
        Cube_menu_left
        Cube_menu_right
    MatchCube2
    MatchCube3
  ▼ Cube3
    ▼ MenuController
        Cube_menu_left
        Cube_menu_right
```

Figure A.7: Task 7 Unity Hierarchy

The functionality used within this task mirrored in a very similar way the code for Task 6. The **`MenuController`** object can be found attached to each separate cube in Figure A.7. There was a reference made to each of the buttons from each **`MenuController`** to reuse the periphery menu buttons from **`menuRight`** and **`menuLeft`** in the three **`MenuController`** objects. Then the properties would only apply the action if the cube were currently selected.

Therefore, avoiding the situation where using the shared-use buttons would apply to all cubes. Many features of the MenuController system written for this experiment would have benefitted from using shared objects for the menu elements, and Unity’s text rendering would have massively simplified some of the systems. For example, all the buttons with text would have three separate textures showing each text with three different background colours for selection. This approach could have been simplified with three colour textures and text rendered from the menu definition instead.

A.2 First Experiment Materials Provided to Participants

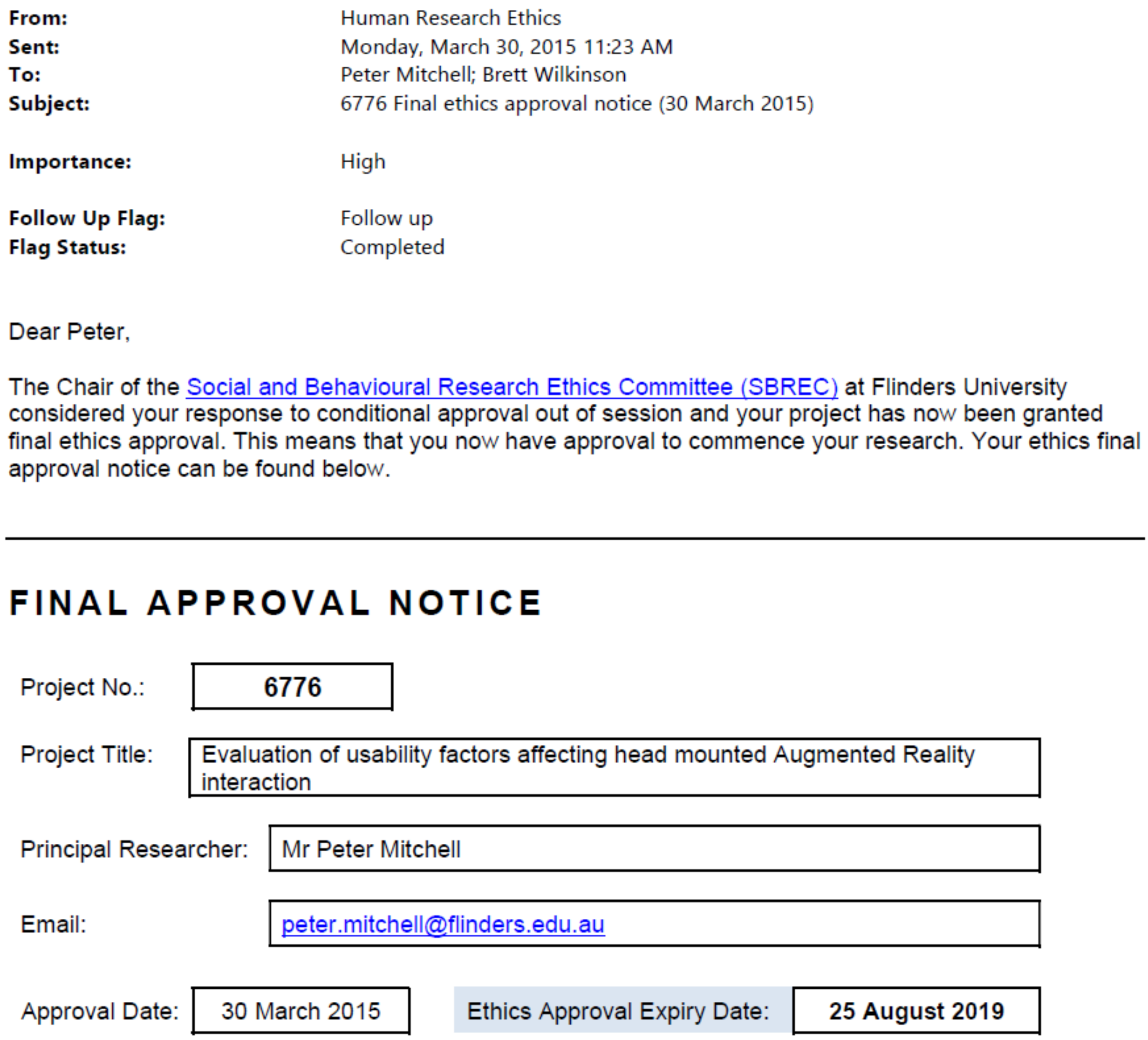
The following sections provide materials relevant to the interactions with participants. Except for the ethics approval email, the rest were provided directly to participants. Each of these sections only provides the material without elaboration. The following paragraphs briefly describe each section identifying where and how they were used in the experiments.

The ethics approval email in A.2.1 shows a confirmation email with the ethics approval for project 6776 used for the first experiment. This approval number was provided on all the material given to potential participants to identify the committee approved the project.

The email text listed in A.2.2 shows what was sent out to potential participants allowing them to contact if interested. Any individual who made contact would then be sent the information pack email in A.2.3 along with the information sheet in A.2.4, the letter of introduction in A.2.5, and the consent form in A.2.6. With available times included that they could request to participate.

If the individual followed up and arranged to participate, they would turn up at the arranged location. The participants were provided with a physical copy of the information sheet (A.2.4), consent form (A.2.6) and screenshots of the tasks (A.2.7). After signing the consent form, the participant would be provided with the pre-experiment questionnaire in A.2.8. After completing the experiment, they were given the post-experiment questionnaire in A.2.9.

A.2.1 First Experiment Ethics Approval Email



From:	Human Research Ethics
Sent:	Monday, March 30, 2015 11:23 AM
To:	Peter Mitchell; Brett Wilkinson
Subject:	6776 Final ethics approval notice (30 March 2015)
Importance:	High
Follow Up Flag:	Follow up
Flag Status:	Completed

Dear Peter,

The Chair of the Social and Behavioural Research Ethics Committee (SBREC) at Flinders University considered your response to conditional approval out of session and your project has now been granted final ethics approval. This means that you now have approval to commence your research. Your ethics final approval notice can be found below.

FINAL APPROVAL NOTICE

Project No.:	**6776**
Project Title:	Evaluation of usability factors affecting head mounted Augmented Reality interaction
Principal Researcher:	Mr Peter Mitchell
Email:	peter.mitchell@flinders.edu.au
Approval Date:	30 March 2015
Ethics Approval Expiry Date:	**25 August 2019**

A.2.2 First Experiment Recruitment Email

Hello {name},

My name is Peter Mitchell and I am a current PhD student in the School of Computer Science, Engineering, and Mathematics (CSEM) at Flinders University. My research interests are head mounted augmented reality (AR) and human-computer interaction (HCI).

I am currently investigating the usability issues of various input methods when used with interactive head mounted AR applications. Specifically, I am interested in highlighting the main differences between the different input methods and also determining whether a preferred (or significantly better) input method. The results of this investigation will lead into broader research I plan to conduct on head mounted AR content authoring.

To assist me with this research, I am seeking volunteers to participate in a brief experiment. The experiment will involve participants completing various interactive tasks using a head mounted display along with mouse and mobile type input devices. Three different input variations will be evaluated and the same set of tasks will be repeated for each input method in order to compare their differences. The tasks will involve controlled movements of the devices and/or interaction with the devices.

During the experiment, participants will also be observed and encouraged to voice their thoughts and feelings (out loud) with regards to the device and/or task being completed. Any comments made will be recorded, via digital voice recorder, and later transcribed for analysis. The reason this approach is being utilised is that I would like to capture participants' thoughts towards the different form-factors 'in the moment' rather than risk them being forgotten or confused with other thoughts at a later time. I will also ask participants to complete two questionnaires, one before attempting the tasks and one following their completion.

The study will be conducted in the IST building at Flinders University. The time commitment required is expected to be around 60 minutes. Participation in this study is completely voluntary and no penalties will be incurred by choosing not to participate. Your participation will be treated anonymously and you will be free to withdraw at any time without consequence. Supervisors and your lecturers will not know who has agreed to participate and who has not.

If you are interested in participating or would like further information, please reply to this email. I will then send you an information pack (via email) containing an information sheet, letter of introduction, and consent form. I will also include a list of available time-slots for participation. If for whatever reason you do not wish to participate, simply ignore this email.

Thank you for your time.

A.2.3 First Experiment Information Pack Email

Hello {name},

Thank you for requesting further information. Please find attached an information pack containing an information sheet, letter of introduction, and consent form. I have also included a list of available time-slots for participation in the experiment.

If, after reading the documents in the information pack, you would like to participate in the experiment, please reply to this email indicating your preferred times on the table below. I will confirm your participation by replying with a selected time from those indicated as well as the location of the experiment (the location will be in IST building at Flinders University). You will need to complete the consent form and bring it with you when you come along.

If for whatever reason you do not wish to participate, do not reply. This email in no way commits you to participate.

If you have any questions regarding any of the material in the information pack or the experiment, please don't hesitate to contact me.

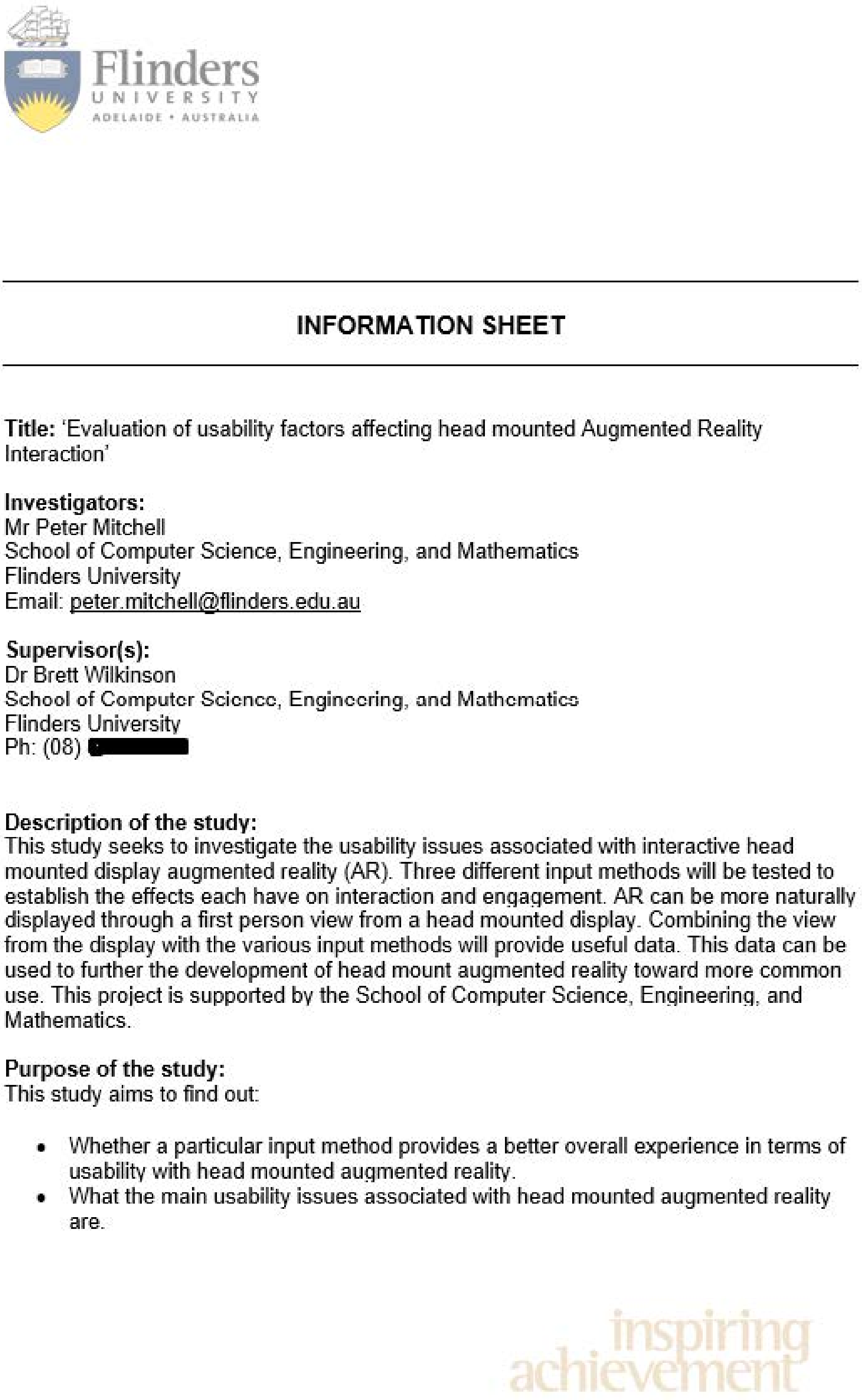
Thanks

Please indicate your preferred time-slots by ticking the appropriate boxes.

	Mon 20/10/14	**Wed 21/10/14**	**Thu 22/10/14**
10:00			
11:30	unavailable	unavailable	
13:00			
14:30			unavailable
16:00			unavailable

example dates and times

A.2.4 First Experiment Information Sheet



Flinders
UNIVERSITY
ADELAIDE • AUSTRALIA

INFORMATION SHEET

Title: 'Evaluation of usability factors affecting head mounted Augmented Reality Interaction'

Investigators:
Mr Peter Mitchell
School of Computer Science, Engineering, and Mathematics
Flinders University
Email: peter.mitchell@flinders.edu.au

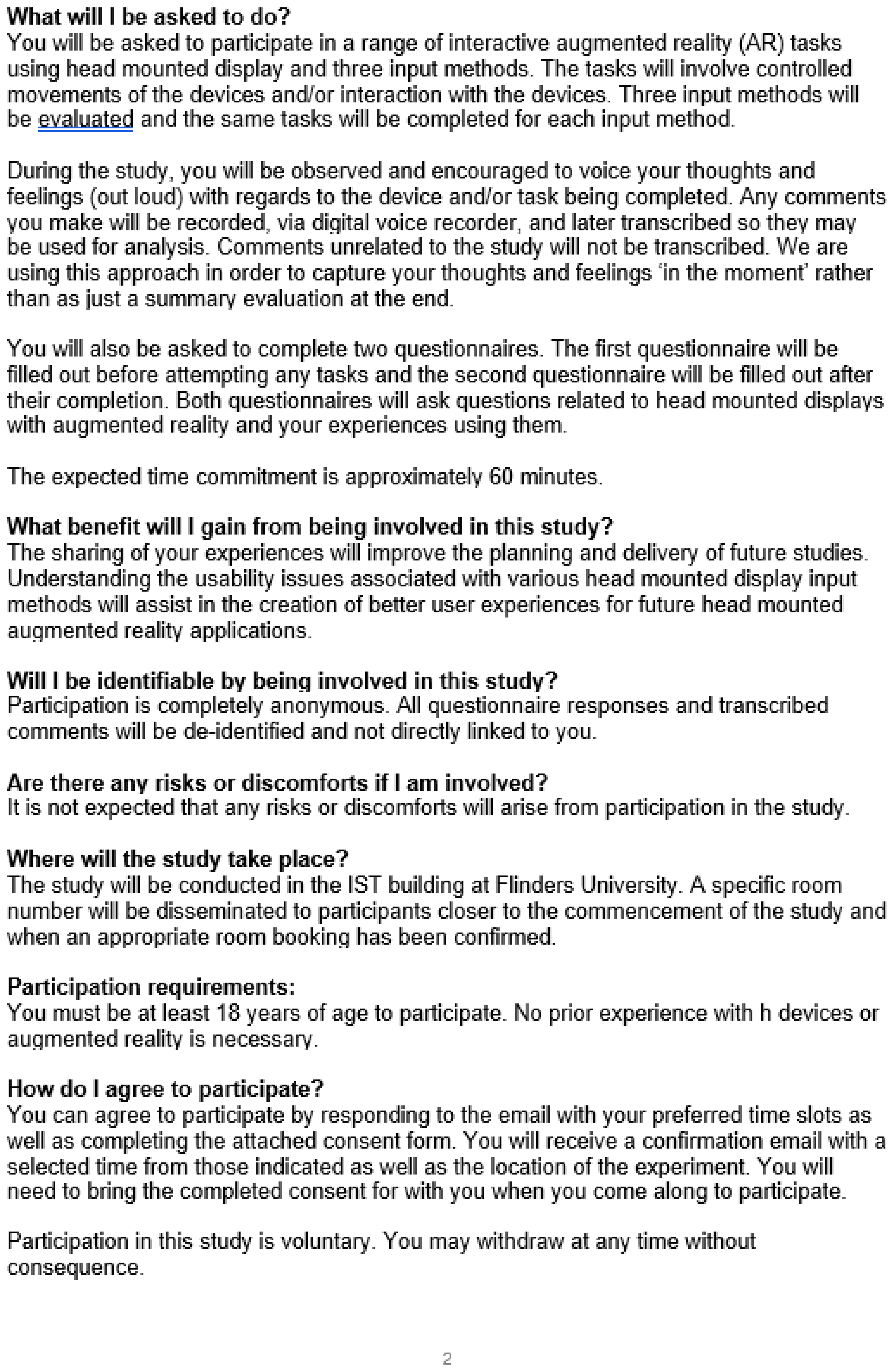
Supervisor(s):
Dr Brett Wilkinson
School of Computer Science, Engineering, and Mathematics
Flinders University
Ph: (08) [redacted]

Description of the study:
This study seeks to investigate the usability issues associated with interactive head mounted display augmented reality (AR). Three different input methods will be tested to establish the effects each have on interaction and engagement. AR can be more naturally displayed through a first person view from a head mounted display. Combining the view from the display with the various input methods will provide useful data. This data can be used to further the development of head mount augmented reality toward more common use. This project is supported by the School of Computer Science, Engineering, and Mathematics.

Purpose of the study:
This study aims to find out:

- Whether a particular input method provides a better overall experience in terms of usability with head mounted augmented reality.
- What the main usability issues associated with head mounted augmented reality are.

inspiring achievement



What will I be asked to do?
You will be asked to participate in a range of interactive augmented reality (AR) tasks using head mounted display and three input methods. The tasks will involve controlled movements of the devices and/or interaction with the devices. Three input methods will be evaluated and the same tasks will be completed for each input method.

During the study, you will be observed and encouraged to voice your thoughts and feelings (out loud) with regards to the device and/or task being completed. Any comments you make will be recorded, via digital voice recorder, and later transcribed so they may be used for analysis. Comments unrelated to the study will not be transcribed. We are using this approach in order to capture your thoughts and feelings 'in the moment' rather than as just a summary evaluation at the end.

You will also be asked to complete two questionnaires. The first questionnaire will be filled out before attempting any tasks and the second questionnaire will be filled out after their completion. Both questionnaires will ask questions related to head mounted displays with augmented reality and your experiences using them.

The expected time commitment is approximately 60 minutes.

What benefit will I gain from being involved in this study?
The sharing of your experiences will improve the planning and delivery of future studies. Understanding the usability issues associated with various head mounted display input methods will assist in the creation of better user experiences for future head mounted augmented reality applications.

Will I be identifiable by being involved in this study?
Participation is completely anonymous. All questionnaire responses and transcribed comments will be de-identified and not directly linked to you.

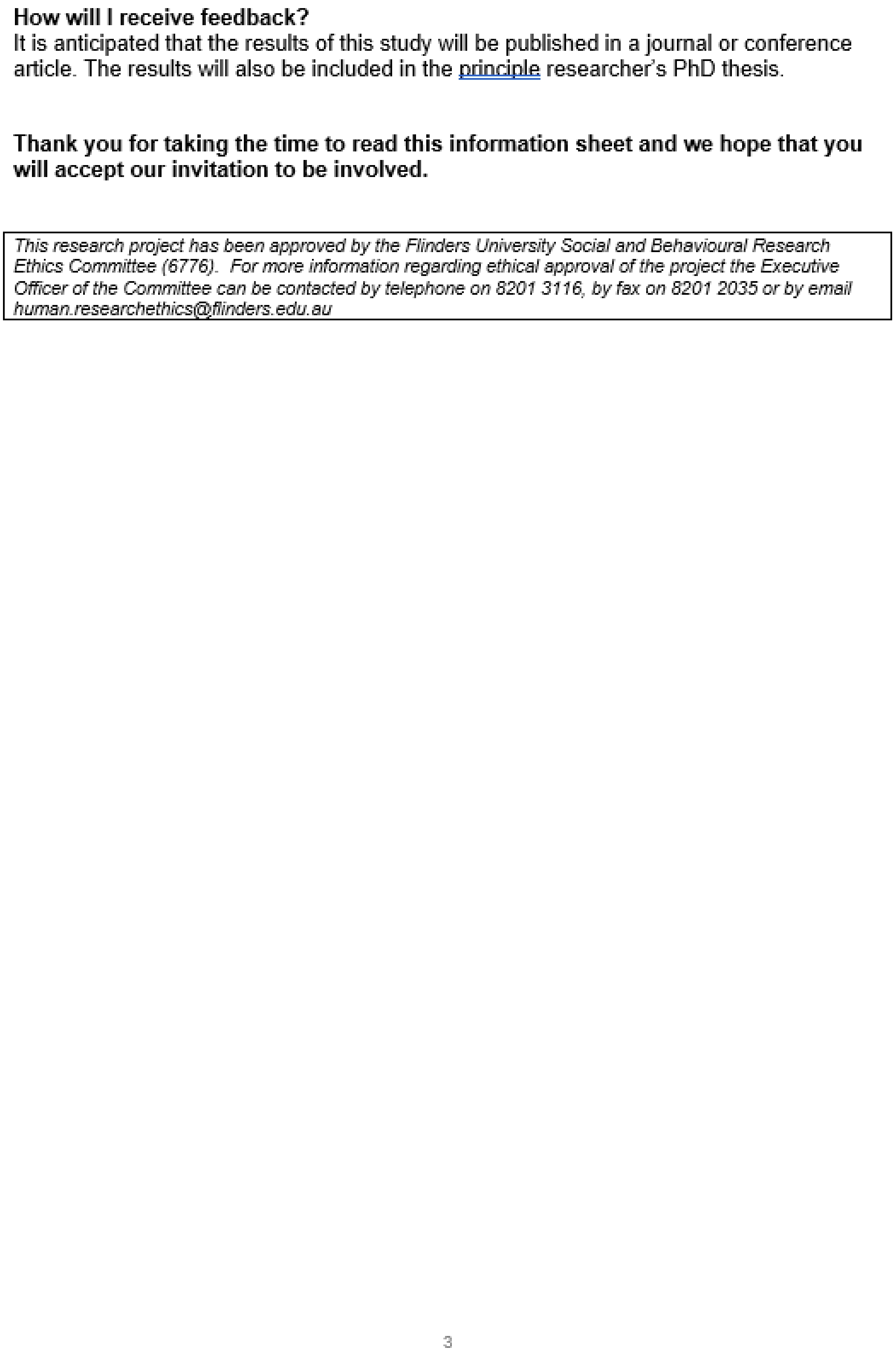
Are there any risks or discomforts if I am involved?
It is not expected that any risks or discomforts will arise from participation in the study.

Where will the study take place?
The study will be conducted in the IST building at Flinders University. A specific room number will be disseminated to participants closer to the commencement of the study and when an appropriate room booking has been confirmed.

Participation requirements:
You must be at least 18 years of age to participate. No prior experience with h devices or augmented reality is necessary.

How do I agree to participate?
You can agree to participate by responding to the email with your preferred time slots as well as completing the attached consent form. You will receive a confirmation email with a selected time from those indicated as well as the location of the experiment. You will need to bring the completed consent for with you when you come along to participate.

Participation in this study is voluntary. You may withdraw at any time without consequence.

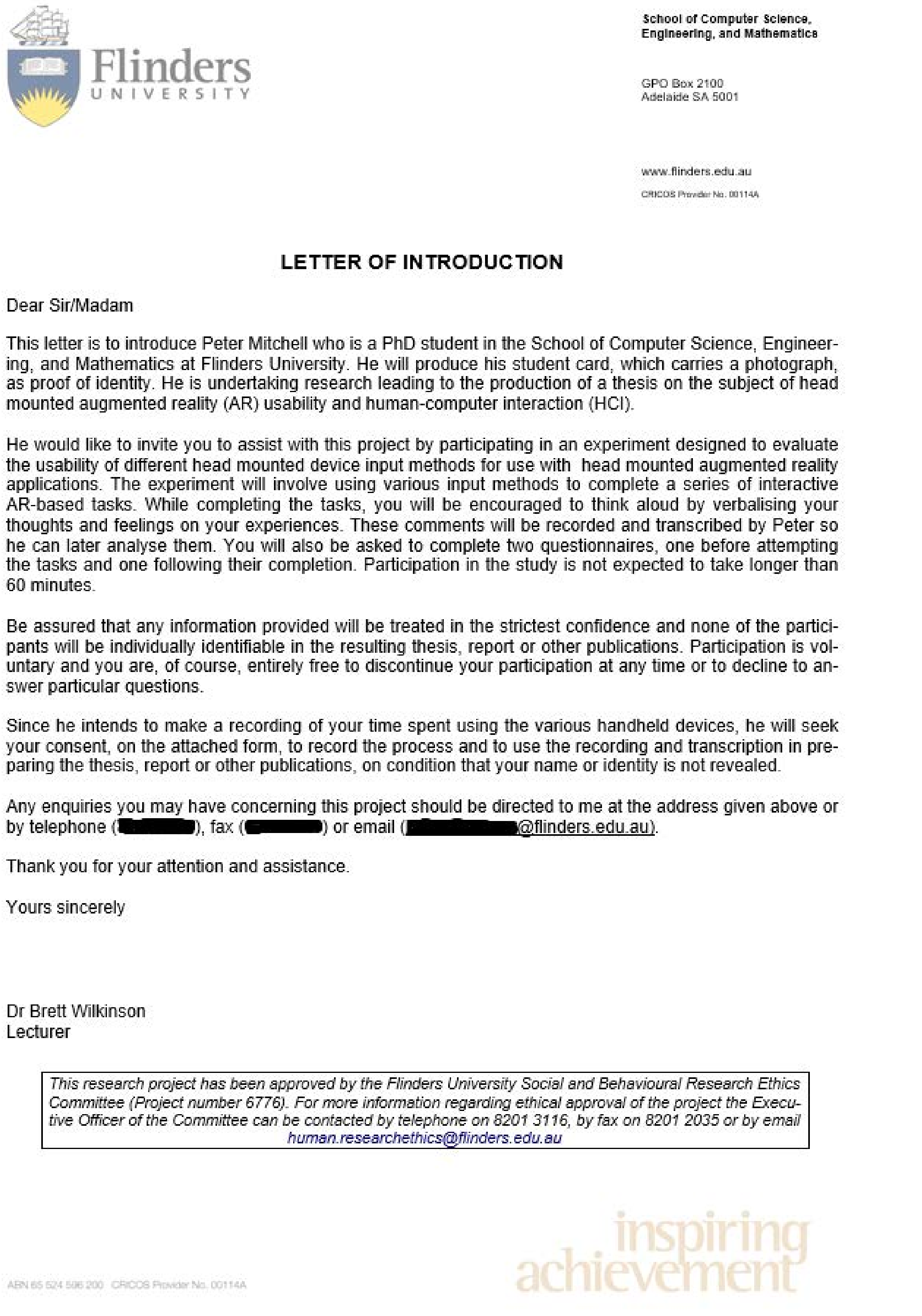


How will I receive feedback?
It is anticipated that the results of this study will be published in a journal or conference article. The results will also be included in the principle researcher's PhD thesis.

Thank you for taking the time to read this information sheet and we hope that you will accept our invitation to be involved.

This research project has been approved by the Flinders University Social and Behavioural Research Ethics Committee (6776). For more information regarding ethical approval of the project the Executive Officer of the Committee can be contacted by telephone on 8201 3116, by fax on 8201 2035 or by email human.researchethics@flinders.edu.au

A.2.5 First Experiment Letter of Introduction



Flinders
UNIVERSITY

School of Computer Science,
Engineering, and Mathematics

GPO Box 2100
Adelaide SA 5001

www.flinders.edu.au
CRICOS Provider No. 00114A

LETTER OF INTRODUCTION

Dear Sir/Madam

This letter is to introduce Peter Mitchell who is a PhD student in the School of Computer Science, Engineering, and Mathematics at Flinders University. He will produce his student card, which carries a photograph, as proof of identity. He is undertaking research leading to the production of a thesis on the subject of head mounted augmented reality (AR) usability and human-computer interaction (HCI).

He would like to invite you to assist with this project by participating in an experiment designed to evaluate the usability of different head mounted device input methods for use with head mounted augmented reality applications. The experiment will involve using various input methods to complete a series of interactive AR-based tasks. While completing the tasks, you will be encouraged to think aloud by verbalising your thoughts and feelings on your experiences. These comments will be recorded and transcribed by Peter so he can later analyse them. You will also be asked to complete two questionnaires, one before attempting the tasks and one following their completion. Participation in the study is not expected to take longer than 60 minutes.

Be assured that any information provided will be treated in the strictest confidence and none of the participants will be individually identifiable in the resulting thesis, report or other publications. Participation is voluntary and you are, of course, entirely free to discontinue your participation at any time or to decline to answer particular questions.

Since he intends to make a recording of your time spent using the various handheld devices, he will seek your consent, on the attached form, to record the process and to use the recording and transcription in preparing the thesis, report or other publications, on condition that your name or identity is not revealed.

Any enquiries you may have concerning this project should be directed to me at the address given above or by telephone ([redacted]), fax ([redacted]) or email ([redacted]@flinders.edu.au).

Thank you for your attention and assistance.

Yours sincerely

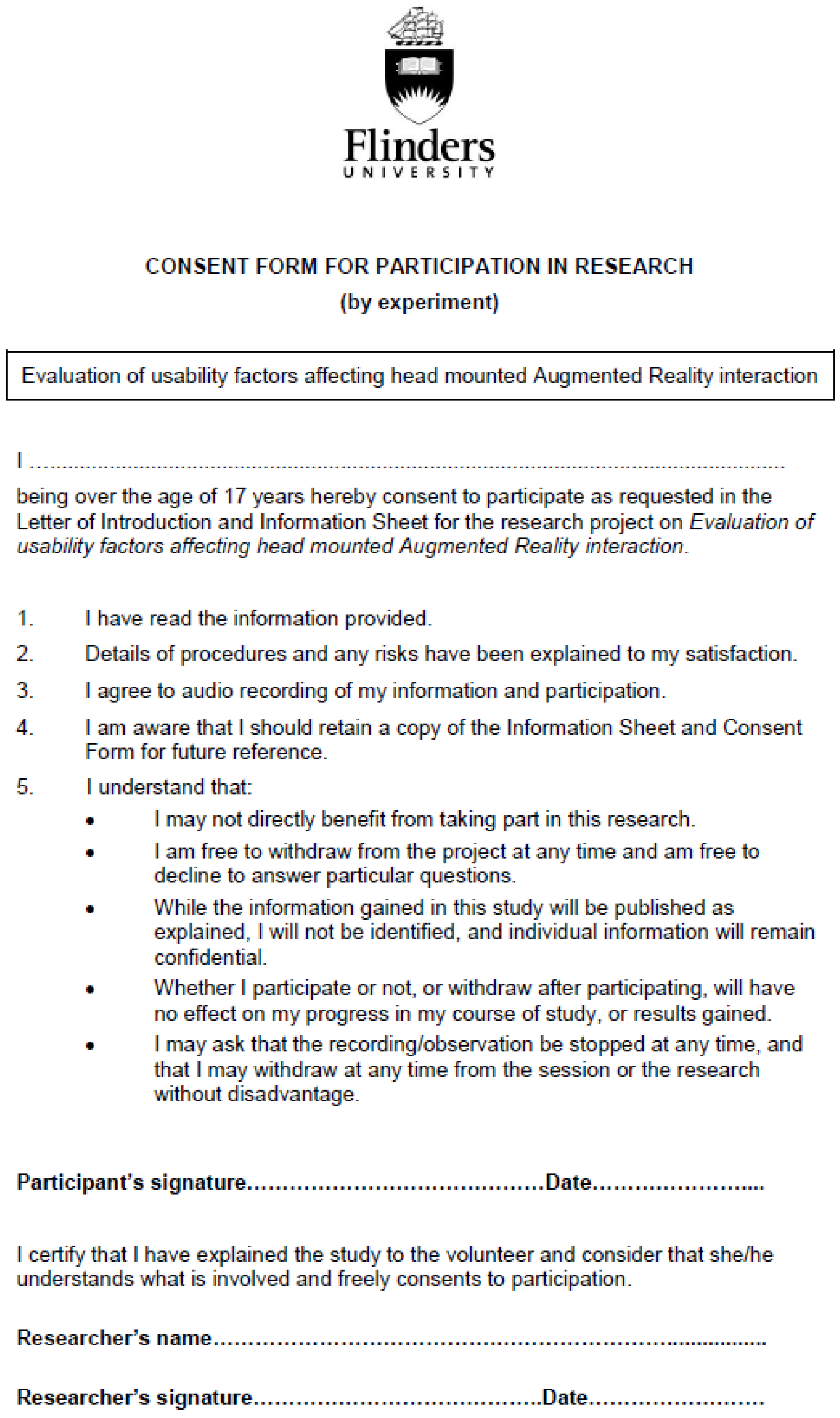
Dr Brett Wilkinson
Lecturer

This research project has been approved by the Flinders University Social and Behavioural Research Ethics Committee (Project number 6776). For more information regarding ethical approval of the project the Executive Officer of the Committee can be contacted by telephone on 8201 3116, by fax on 8201 2035 or by email human.researchethics@flinders.edu.au

ABN 65 524 596 200 CRICOS Provider No. 00114A

inspiring
achievement

A.2.6 First Experiment Consent Form



Flinders
UNIVERSITY

CONSENT FORM FOR PARTICIPATION IN RESEARCH
(by experiment)

Evaluation of usability factors affecting head mounted Augmented Reality interaction

I ....

being over the age of 17 years hereby consent to participate as requested in the Letter of Introduction and Information Sheet for the research project on *Evaluation of usability factors affecting head mounted Augmented Reality interaction*.

1. I have read the information provided.
2. Details of procedures and any risks have been explained to my satisfaction.
3. I agree to audio recording of my information and participation.
4. I am aware that I should retain a copy of the Information Sheet and Consent Form for future reference.
5. I understand that:
   - I may not directly benefit from taking part in this research.
   - I am free to withdraw from the project at any time and am free to decline to answer particular questions.
   - While the information gained in this study will be published as explained, I will not be identified, and individual information will remain confidential.
   - Whether I participate or not, or withdraw after participating, will have no effect on my progress in my course of study, or results gained.
   - I may ask that the recording/observation be stopped at any time, and that I may withdraw at any time from the session or the research without disadvantage.

Participant's signature.......................................Date.......................

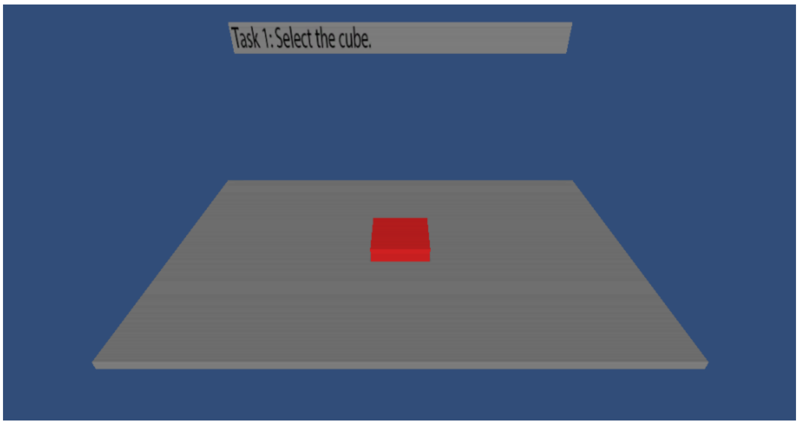
I certify that I have explained the study to the volunteer and consider that she/he understands what is involved and freely consents to participation.

Researcher's name...

Researcher's signature.......................................Date........................

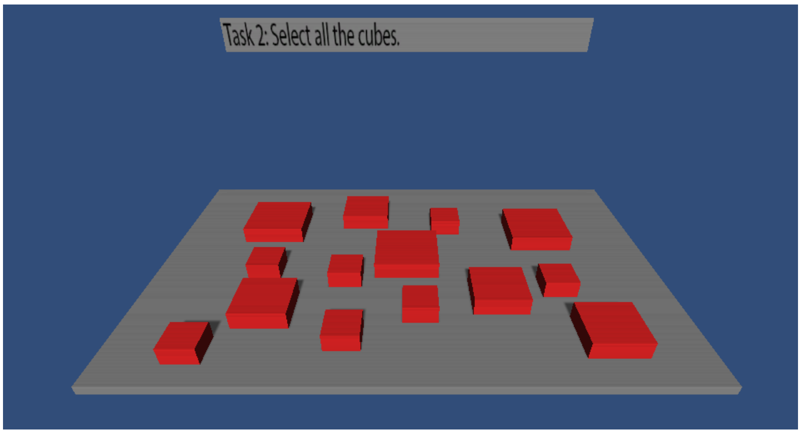
A.2.7 First Experiment Screenshots of Tasks

Task 1



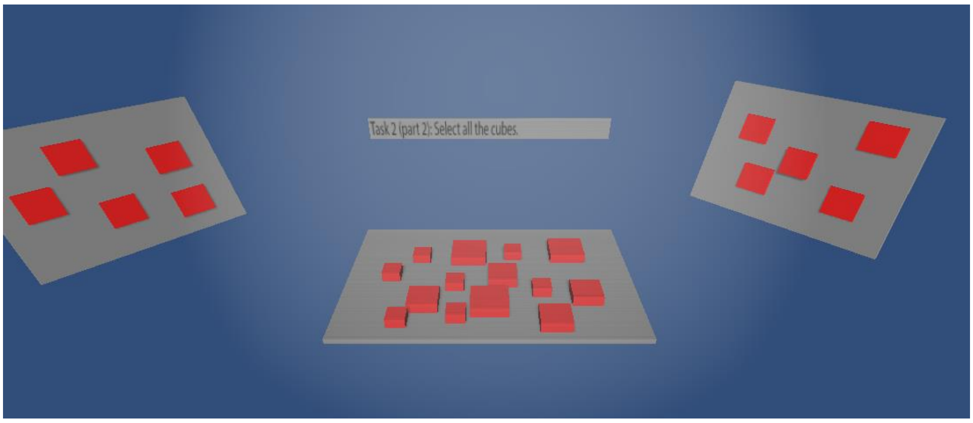
Task 1: Select the cube.

Task 2



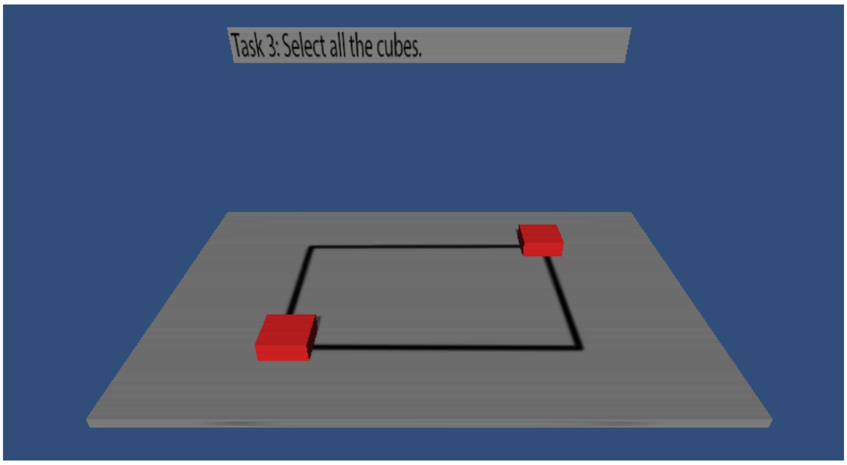
Task 2: Select all the cubes.

Task 2 (part 2)

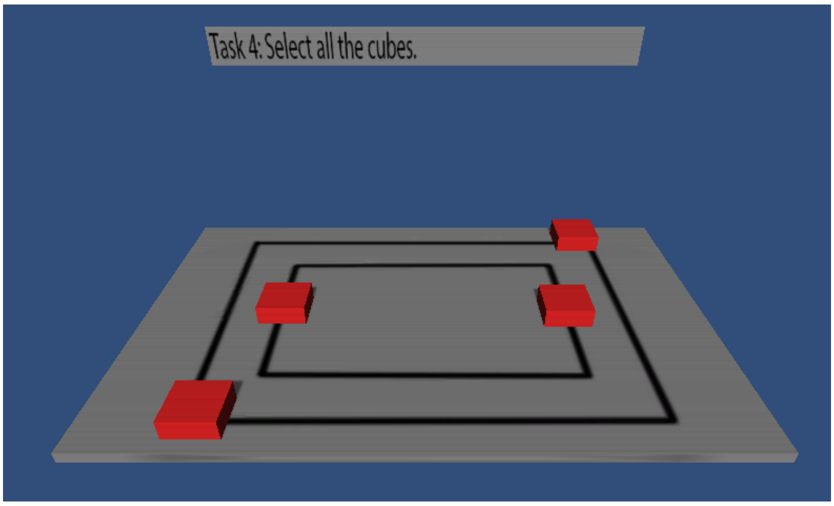


Task 2 (part 2): Select all the cubes.

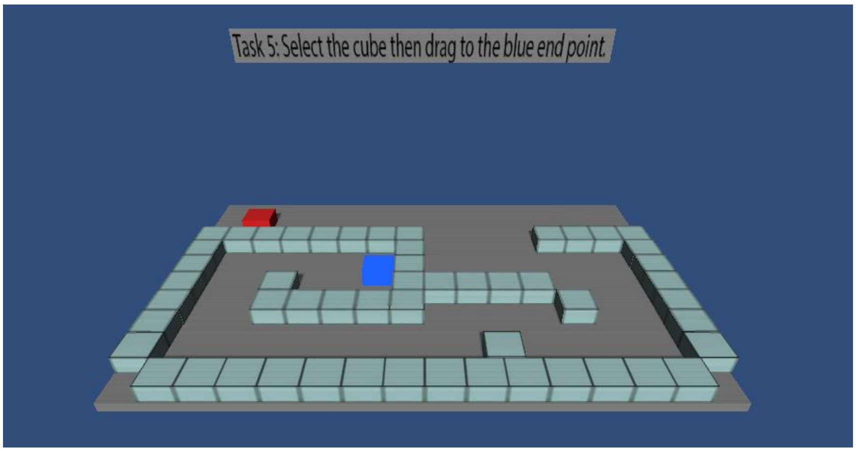
Task 3


Task 3: Select all the cubes.

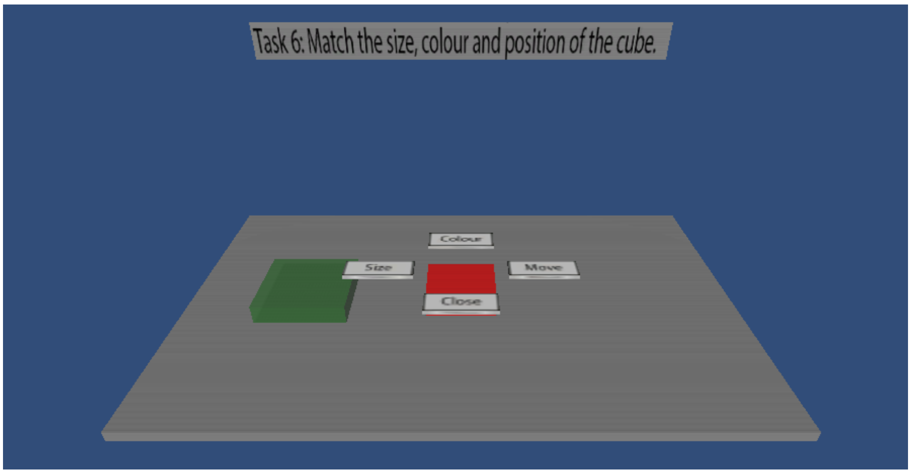
Task 4


Task 4: Select all the cubes.

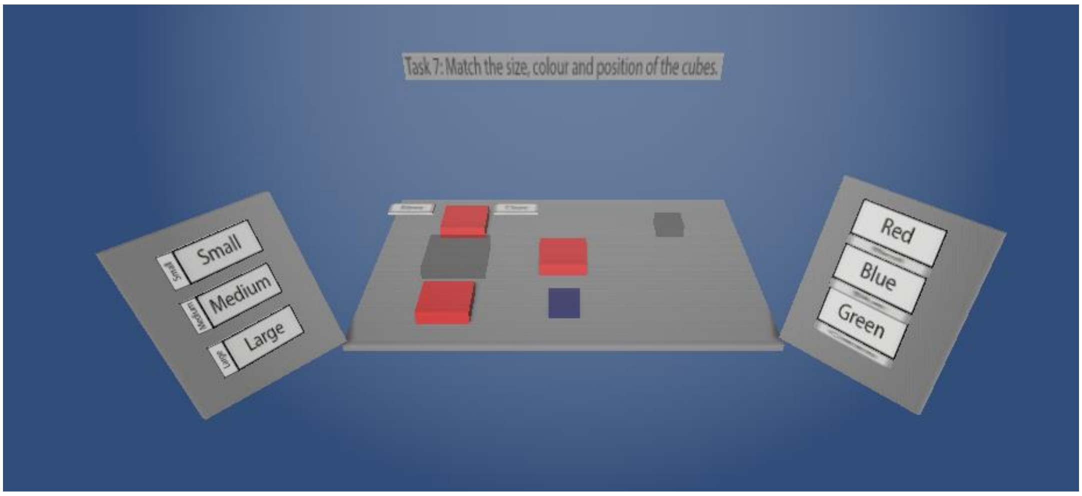
Task 5


Task 5: Select the cube then drag to the blue end point.

Task 6


Task 6: Match the size, colour and position of the cube.
Colour
Size
Move
Close

Task 7


Task 7: Match the size, colour and position of the cubes.
Small
Small
Medium
Medium
Large
Large
Red
Blue
Green

A.2.8 First Experiment Pre-Experiment Questionnaire

1. Which of the following age ranges do you fall into?

   [ ] Under 21

   [ ] 21 to 30

   [ ] 31 to 40

   [ ] 41 to 50

   [ ] 51 to 60

   [ ] 61 and over

2. What is your gender?

   [ ] Male

   [ ] Female

3. Which of the following apply to your current situation?

   [ ] Student

   [ ] Academic Researcher

   [ ] Teaching Staff Member

   [ ] Other

4. If in question 3 you indicated you were currently a student what is your area of study?

   ____

5. If in question 3 you indicated you were currently an Academic Researcher or Teaching Staff Member, what area of knowledge is your research or teaching in?

____

6. Have you previously participated in any research as a volunteer for a project involving Augmented Reality, Head Mounted Displays or Mobile/Tablet based computing? (tick as many as apply to you):

   [ ] Augmented Reality

   [ ] Head Mounted Displays

   [ ] Mobile/Tablet Computing

7. If you selected any of the options in question 6, did you have a generally positive experience with the previous involvement in research?

   [ ] Yes

   [ ] No

8. Have you ever used an Augmented Reality application?

   [ ] Yes

   [ ] No

9. If you answered Yes to question 8, what types of Augmented Reality application/s have you used and where did you use them?

____

____

10. Have you ever used any Head Mounted display type devices?

    [ ] Yes

    [ ] No

11. If you answered Yes to question 10, what types of Head Mounted displays have you used and where did you use them?

____

____

12. How many hours a week would you use a computer on average?

    [ ] Less than 10 hours

    [ ] 10 to 20 hours

    [ ] 20 to 30 hours

    [ ] 30 to 40 hours

    [ ] More than 40 hours

13. How many hours a week would you spend playing video games on average?

    [ ] Less than 10 hours

    [ ] 10 to 20 hours

    [ ] 20 to 30 hours

    [ ] 30 to 40 hours

    [ ] More than 40 hours

14. If you play computer games, what are a few examples of games you play and the devices you play those games on?

____

____

15. On a scale of 1 to 10 (1 being not interested and 10 being very interested), how interested would you be in using a head mounted display for activities other than gaming?

Not Interested									Very Interested
1	2	3	4	5	6	7	8	9	10

16. On a scale of 1 to 10 (1 being not important and 10 being very important), how important is the fashionability of the device to you as a prospective wearer in a public setting?

Not Important									Very Important
1	2	3	4	5	6	7	8	9	10

17. What factors do you feel most influence whether you would want to wear a head mounted display around in public? (number items from 1 to 5, where 1 is the most influential and 5 is the least influential):

    a. ___ The size of the device.
    b. ___ The weight of the device.
    c. ___ The comfort of the device.
    d. ___ The outward visual appeal of the device.
    e. ___ The usefulness of the device.

18. On a scale of 1 to 10 (1 being useless and 10 being very useful), how useful do you believe a head mounted display using augmented reality would be for elderly or those with other assisted living needs?

Useless									Very Useful
1	2	3	4	5	6	7	8	9	10

19. If you could only select one of the following three input combinations for completing the tasks you are about to perform, which would you select?

    [ ] Oculus Rift only.

    [ ] Oculus Rift and a computer mouse.

    [ ] Oculus Rift and a mobile device.

    Why?

____

20. Please rank from 1 to 3 in order how effective you think the following input combinations will be for the tasks you are about to perform:

    a. ___ Oculus Rift only.

    b. ___ Oculus Rift and a computer mouse.

    c. ___ Oculus Rift and a mobile device.

A.2.9 First Experiment Post-Experiment Questionnaire

Please answer the following questions in relation to the tests you have just completed.

1. Please rate how effective you thought each input method was for completing the tasks where 1 represents extremely ineffective and 5 represents extremely effective:
   Circle the number that best represents your answer.

Oculus Rift only:

	Extremely ineffective				Extremely effective
Task 1	1	2	3	4	5
Task 2	1	2	3	4	5
Task 3	1	2	3	4	5
Task 4	1	2	3	4	5
Task 5	1	2	3	4	5
Task 6	1	2	3	4	5
Task 7	1	2	3	4	5

Oculus Rift with a computer mouse:

	Extremely ineffective				Extremely effective
Task 1	1	2	3	4	5
Task 2	1	2	3	4	5
Task 3	1	2	3	4	5
Task 4	1	2	3	4	5
Task 5	1	2	3	4	5
Task 6	1	2	3	4	5
Task 7	1	2	3	4	5

Oculus Rift with a mobile device:

	Extremely ineffective				Extremely effective
Task 1	1	2	3	4	5
Task 2	1	2	3	4	5
Task 3	1	2	3	4	5
Task 4	1	2	3	4	5
Task 5	1	2	3	4	5
Task 6	1	2	3	4	5
Task 7	1	2	3	4	5

2. Please answer the following question for each task:

Circle the dot that best represents your answer.

Overall, I found completing this task using the Oculus Rift only:

	Very difficult				Very easy
Task 1	•	•	•	•	•
Task 2	•	•	•	•	•
Task 3	•	•	•	•	•
Task 4	•	•	•	•	•
Task 5	•	•	•	•	•
Task 6	•	•	•	•	•
Task 7	•	•	•	•	•

Overall, I found completing this task using the Oculus Rift with a computer mouse:

	Very difficult				Very easy
Task 1	•	•	•	•	•
Task 2	•	•	•	•	•
Task 3	•	•	•	•	•
Task 4	•	•	•	•	•
Task 5	•	•	•	•	•
Task 6	•	•	•	•	•
Task 7	•	•	•	•	•

Overall, I found completing this task using the Oculus Rift with a mobile device:

	Very difficult				Very easy
Task 1	•	•	•	•	•
Task 2	•	•	•	•	•
Task 3	•	•	•	•	•
Task 4	•	•	•	•	•
Task 5	•	•	•	•	•
Task 6	•	•	•	•	•
Task 7	•	•	•	•	•

3. What three aspects did you find most **enjoyable** about using the **Oculus Rift only** input to complete the tasks?

    1. ____
    2. ____
    3. ____

4. What three aspects did you find most **difficult** about using the **Oculus Rift only** input to complete the tasks?

    1. ____

    2. ____

    3. ____

5. What three aspects did you find most **enjoyable** about using the **Oculus Rift and computer mouse** input to complete the tasks?

    1. ____

    2. ____

    3. ____

6. What three aspects did you find most **difficult** about using the **Oculus Rift and computer mouse** input to complete the tasks?

    1. ____

    2. ____

    3. ____

7. What three aspects did you find most **enjoyable** about using the **Oculus Rift and mobile device** input to complete the tasks?

    1. ____

    2. ____

    3. ____

8. What three aspects did you find most **difficult** about using the **Oculus Rift and mobile device** input to complete the tasks?

    1. ____

    2. ____

    3. ____

9. Overall, if you had to pick just <u>one</u> of the input methods to complete the tasks again, which would you choose and why?

   [ ] Oculus Rift Only ____

   [ ] Oculus Rift and Mouse ____

   [ ] Oculus Rift and Mobile ____

   [ ] Other

   What type of input would you prefer to use instead?

   ____

   Please explain why you would prefer an alternate input method?

   ____

10. Would you use an augmented reality (AR) application on a head mounted display in the future?

    [ ] Yes

    [ ] No. Why? ____

11. For each form-factor, please indicate the level of fatigue you experienced while completing the tasks.

    Please circle the number that best represents your answer on a scale of 1 (no fatigue) to 5 (extreme fatigue).

Oculus Rift Only:

	No fatigue				Extreme fatigue
Task 1	1	2	3	4	5
Task 2	1	2	3	4	5
Task 3	1	2	3	4	5
Task 4	1	2	3	4	5
Task 5	1	2	3	4	5
Task 6	1	2	3	4	5
Task 7	1	2	3	4	5

Oculus Rift and Computer Mouse:

	No fatigue				Extreme fatigue
Task 1	1	2	3	4	5
Task 2	1	2	3	4	5
Task 3	1	2	3	4	5
Task 4	1	2	3	4	5
Task 5	1	2	3	4	5
Task 6	1	2	3	4	5
Task 7	1	2	3	4	5

Oculus Rift and Mobile Device:

	No fatigue				Extreme fatigue
Task 1	1	2	3	4	5
Task 2	1	2	3	4	5
Task 3	1	2	3	4	5
Task 4	1	2	3	4	5
Task 5	1	2	3	4	5
Task 6	1	2	3	4	5
Task 7	1	2	3	4	5

12. For each input method, please circle the number that best describes your feelings towards that method in relation to the tasks you have just completed:

    a. I think that I would like to use this input method frequently

	Strongly disagree				Strongly agree
Oculus Rift Only	1	2	3	4	5
Oculus Rift and Mouse	1	2	3	4	5
Oculus Rift and Mobile	1	2	3	4	5

b. I found the input method unnecessarily complex

	Strongly disagree				Strongly agree
Oculus Rift Only	1	2	3	4	5
Oculus Rift and Mouse	1	2	3	4	5
Oculus Rift and Mobile	1	2	3	4	5

c. I thought the input method was easy to use

	Strongly disagree				Strongly agree
Oculus Rift Only	1	2	3	4	5
Oculus Rift and Mouse	1	2	3	4	5
Oculus Rift and Mobile	1	2	3	4	5

d. I found the various functions in this input method were well integrated

	Strongly disagree				Strongly agree
Oculus Rift Only	1	2	3	4	5
Oculus Rift and Mouse	1	2	3	4	5
Oculus Rift and Mobile	1	2	3	4	5

e. I thought there was too much inconsistency with this input method

	Strongly disagree				Strongly agree
Oculus Rift Only	1	2	3	4	5
Oculus Rift and Mouse	1	2	3	4	5
Oculus Rift and Mobile	1	2	3	4	5

f. I would imagine that most people would learn how to use this input method very quickly

	Strongly disagree				Strongly agree
Oculus Rift Only	1	2	3	4	5
Oculus Rift and Mouse	1	2	3	4	5
Oculus Rift and Mobile	1	2	3	4	5

g. I found the input method very cumbersome to use

	Strongly disagree				Strongly agree
Oculus Rift Only	1	2	3	4	5
Oculus Rift and Mouse	1	2	3	4	5
Oculus Rift and Mobile	1	2	3	4	5

h. I felt very confident using the input method

	Strongly disagree				Strongly agree
Oculus Rift Only	1	2	3	4	5
Oculus Rift and Mouse	1	2	3	4	5
Oculus Rift and Mobile	1	2	3	4	5

i. I needed to learn a lot of things before I could get going with this input method

	Strongly disagree				Strongly agree
Oculus Rift Only	1	2	3	4	5
Oculus Rift and Mouse	1	2	3	4	5
Oculus Rift and Mobile	1	2	3	4	5

13. Do you have any other comments or thoughts regarding head mounted display and AR?

B Appendix B: Additional Second Experiment Details

This appendix is similar to the other two appendices for the first and third experiments. It provides both additional details about the experiment implementation and documentation related to interaction with participants.

B.1 Second Experiment Additional Information

This section covers important additional implementation details for the second experiment excluded from the primary dissertation because they were not necessary for understanding the content. The additional information provided here may help replicate the experiment. The style of presentation is similar to the first experiment's appendix.

The first section on level management in B.1.1 leads into the level change interactions in B.1.2, describing how level swaps occurred during the experiment. This discussion is followed by the summary of important class files in B.1.3 and the unity game object hierarchy in B.1.4. The experiment used some purchased and freely available assets to make it visually appealing. These are listed, including costs in the unity store assets section in B.1.5.

The last two sections provide some additional information about the implementation of tasks. The section on the configuration of the tower construction task shows the unity hierarchy for that task in B.1.6. And the last section covers an expanded discussion on the implementation of waves of enemies used for the tower defence in B.1.7.

B.1.1 Level Management

There were three different aspects related to the level management within the Unity project for the experiment. Mostly this is technical concerning what files and structure were used for the project while keeping the information brief.

B.1.2 Level Change Interactions

Figure B.1 (over) shows a top-down view of the whole area used by the experiment. The mountains in the middle of the game world divide the area into two distinct parts. The tower defence game is seen on the right, and the area used for pre-and post-game interaction is seen on the left. There are no camera angles available to the participant where they could see through the mountains to the content on the other side. Changing levels (experiment tasks) was automatically handled by a state machine. This state machine included states for introductory information and questions after each task. The states specific to each task will be identified in their respective sections. The experiment could at any time be paused by pressing "P" to halt the update loop till it was pressed again. Due to the nature of tasks in this experiment relying on data from previous tasks, mostly experiment tasks 2 and 3, no skip functionality was built to jump between experiment states manually. The AutoSave feature was created to enable returning to the same game state if the game had crashed or been closed for technical reasons. Therefore, not losing any of the related data. This feature was not needed during the running of this experiment but was a useful debugging feature while developing.

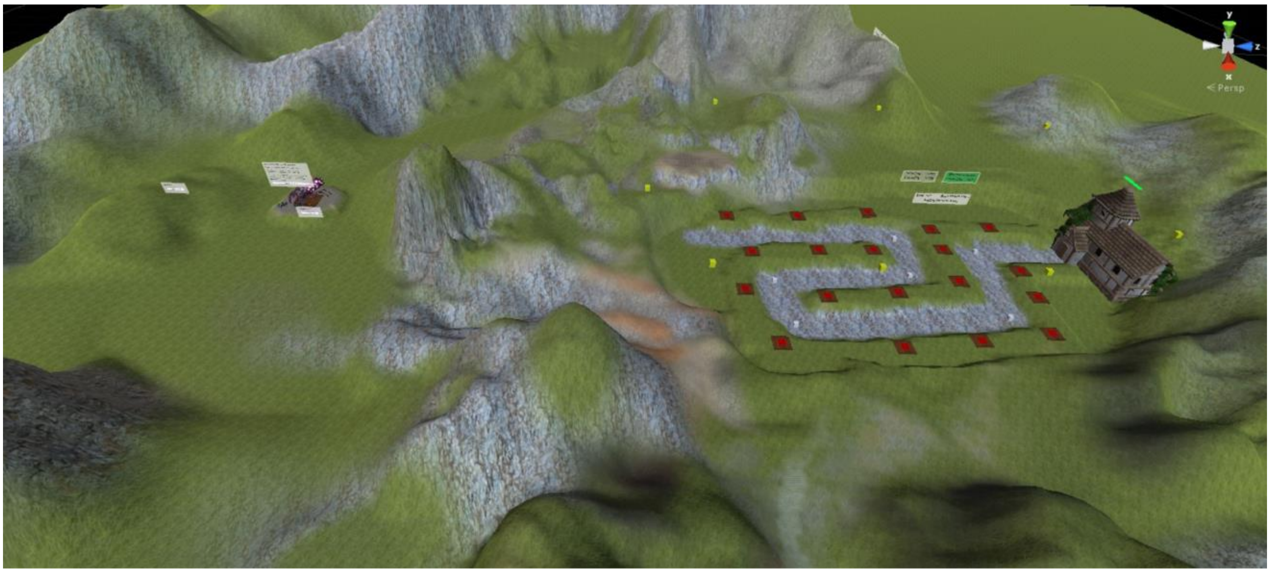


Figure B.1: Full View of Experiment Areas

B.1.3 Class Files

The following list provides brief definitions for the developed C# scripts used to provide the functionality to game objects within the Unity project. They have been listed to provide a clear definition of the scope of content developed within this experiment. Some of the classes will be defined more specifically in later sections to provide specific information for tasks. As can be seen, some classes were improved upon while moving from the first experiment to this second experiment. The third experiment reused many of these classes with some improvements.

- **CameraBehaviour.cs**: Used to cycle the primary view between objects, focusing on managing the camera's position and interactions with input methods. Including shortcut keys, debug interactions and management of tower interactions.
- **ExperimentState.cs**: Used to manage the entire state of the experiment.
- **PersistantStateBehaviour.cs**: Used to maintain the **ReplayDatabase** and **ExtraData** log with support for data retention between scene transitions (this experiment did not need to use this as it only used the one input method).
- **SharedStrings.cs**: Some String definitions of tower data for use in interface elements.
- **ConstructionTaskScripts/ConstructionSnapBehaviour.cs**: Represents the stack of tower components.
- **ConstructionTaskScripts/TowerComponentBehaviour.cs**: Keeps track of the tower component type for a single tower component.
- **ConstructionTaskScripts/TowerConstructionManager.cs**: Calculates the modifiers for towers based on the data generated using construction snap behaviours. It keeps track of all different towers, updates the text dialog for the tower construction task, and provides the data for new towers created during the tower defence.
- **MenuSensitivityScripts/MenuSensitivityTest.cs**: Coordinates the calibration task by randomising a sequence of actions. Then acting as a state machine using the data from the **SensitivityMenuBehaviour**.
- **MenuSensitivityScripts/SensitivityMenuBehaviour.cs**: Simple dialog button interaction tracking with one instance for each of the left, right, and middle dialogs during the calibration task.
- **PeripheryMenuScripts/InformationMenuBehaviour.cs**: Defines a simple dialog with space for text information and a continue button.
- **PeripheryMenuScripts/MenuBehaviour.cs**: Defines the menu displayed by a PVMS triggered event.
- **PeripheryMenuScripts/PeripheryBehaviour.cs**: Defines the primary code for the Periphery Menu System.

- **PeripheryMenuScripts/PeripheryHoverTask.cs**: Used to toggle on the additional features for the periphery context preview task.
- **PeripheryMenuScripts/QuestionMenuBehaviour.cs**: Defines interactions for a menu with variable numbers of options to select one. This behaviour was used for the 1 to 5 scale questions after each task.
- **ReplayDataStorageScripts/AutoSave.cs**: Automatically stores information related to the experiment state so that, if necessary, the experiment could be closed and reopened to continue from the same place. They are defined by a list of String pairs with a property name and property value.
- **ReplayDataStorageScripts/ErrorLog.cs**: Defines a quick and easy way to log errors out to a file and spot any significant problems quickly. Primarily for debugging.
- **ReplayDataStorageScripts/ExtraDataRecorder.cs**: Defines a singleton with a list of **ExtraDataCollection**s. Allowing multiple separate smaller lists (of type String) to be generated simultaneously for easier data processing.
- **ReplayDataStorageScripts/ReplayDatabase.cs**: Controls a list of **ReplayEvent**s to either store new ones or iterate through a replay.
- **ReplayDataStorageScripts/ReplayEngine.cs**: Used by the **CameraBehaviour** to initiate a Replay or Record approach while managing a **ReplayDatabase**.
- **ReplayDataStorageScripts/ReplayEvent.cs**: Defines the information required for a single frame of a replay.
- **ReplayDataStorageScripts/Settings.cs**: Provides a singleton of itself with a list of **SettingProperty** type objects to be autoloaded and saved to a Settings.txt file. If the file does not exist, this is auto-generated with defaults.
- **StartMenu/StartBossAnimator.cs**: Automatically managed random animations for the enemy boss at the start menu.
- **TowerDefenceScripts/LevelManager.cs**: Defines the state management for the tower defence. Controlling the waves and delays to use for the **AISpawner** and automatically handling the inputs from the PVMS.

- **TowerDefenceScripts/LevelStatusDialogBehaviour.cs**: Defines the information dialog showing player's gold, base health, and current enemy wave status.
- **TowerDefenceScripts/TowerStatusDialog.cs**: Defines the information dialog showing the currently selected towers and those in the process of being selected.
- **TowerDefenceScripts/LevelObjectScripts/AISpawner.cs**: Receives wave commands from the **LevelManager** and will spawn prefabs of enemy units with delays.
- **TowerDefenceScripts/LevelObjectScripts/AIWayfinder.cs**: Defines the AI logic, health, health bar, movement properties, animation management, and other properties related to each enemy unit.
- **TowerDefenceScripts/LevelObjectScripts/CameraSnapBehaviour.cs**: Defines a location the **CameraBehaviour** can be snapped to within the game world.
- **TowerDefenceScripts/LevelObjectScripts/EndPointBehaviour.cs**: Defines the player's base representing the last node waypoint for the **AIWayfinder** to reach. Tracking the health of the player's base with a health bar.
- **TowerDefenceScripts/LevelObjectScripts/EnemyProjectileBehaviour.cs**: Defines a specific projectile for enemy units to fire at towers.
- **TowerDefenceScripts/LevelObjectScripts/HealthBarBehaviour.cs**: Used to represent the health above **AIWayfinder** objects and the **EndPointBehaviour**.
- **TowerDefenceScripts/LevelObjectScripts/ProjectileBehaviour.cs**: Defines a projectile fired by towers with logic for the different types of projectiles and damage application upon reaching targets.
- **TowerDefenceScripts/LevelObjectScripts/TowerBehaviour.cs**: Defines the towers' properties. Controlling the firing of prefabs for the **ProjectileBehaviour**s.
- **TowerDefenceScripts/LevelObjectScripts/TowerRangeBehaviour.cs**: Shows a visual pulsing representation of the tower range.

- **TowerDefenceScripts/LevelObjectScripts/TowerSnapBehaviour.cs**: Represents information needed for a single tower to snap to a predefined location.
- **TowerDefenceScripts/LevelObjectScripts/WaveCommand.cs**: Defines wave commands with a time to wait and unit id to spawn. The class primarily provided functionality to convert between a float time with int unit id and a String. It also handled and error management automatic during conversion.
- **TowerDefenceScripts/LevelObjectScripts/WayPointBehaviour.cs**: Hides the WayPoints to make them visible in the editor but not visible while the game is running. Used to represent the nodes each **AIWayfinder** will move between to reach the **EndPointBehaviour** object.

B.1.4 Unity Game Object Hierarchy

Some of the generalised game object hierarchy is shown in the figures below to give insight into the object structure used for this experiment's Unity project. Hierarchy specific to the tasks will be shown in the relevant task sections. Figure B.2 shows the full hierarchy of game objects shrunk down using the categories based on the related task or shared object types. The **ParentCamera** is expanded in Figure B.3, showing the camera and nodes used for the Periphery Menu System. For specific discussion about how this works, refer to section 4.3.3.

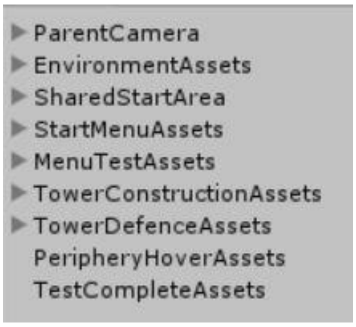


Figure B.2: Unity Hierarchy All Objects

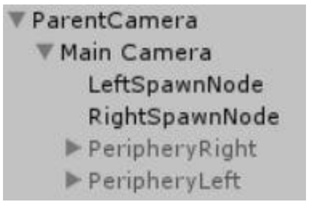


Figure B.3: Camera Object Hierarchy:

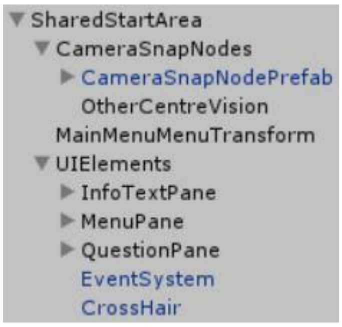


Figure B.4: Shared Objects Hierarchy

Additionally, the two **PeripheryRight** and **PeripheryLeft** objects are related to the Periphery Context Preview Task. Figure B.4 shows a collection of the shared assets used by all tasks. Including all the positions for the camera to be set to, interface elements, the interface Unity event system, and crosshair.

B.1.5 Unity Store Assets

Developing custom art and audio assets were considered outside the scope of this experiment. For this reason, a variety of free options were considered before purchasing some assets to improve the visual appeal of the experiment. A total of $86.49 AUD was spent purchasing assets used for this experiment and were all reused for the third experiment.

Enemy Monsters

- Basic enemy (front left): ($10)
  https://www.assetstore.unity3d.com/en/#!/content/7694
- Fast enemy (front middle): ($10)
  https://www.assetstore.unity3d.com/en/#!/content/35184
- Dangerous enemy (front right): ($17)
  https://www.assetstore.unity3d.com/en/#!/content/34949
- Boss enemy (back middle): ($15)
  https://www.assetstore.unity3d.com/en/#!/content/28304

The chosen monster assets each had a variety of colours included except for the boss enemy. The colours and models were chosen to make the different enemy types distinct.

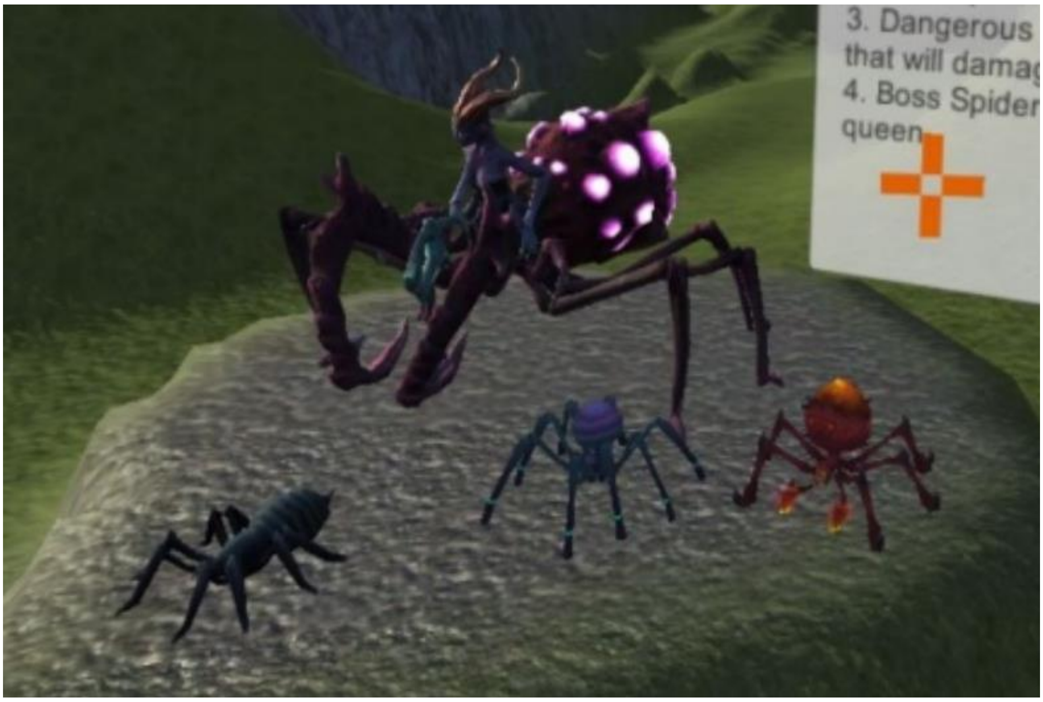


Figure B.5: Enemy Creature Models

Other Purchased Assets

- Explosive Tower Projectile Explosion: ($5 – no longer available) https://www.assetstore.unity3d.com/en/#!/content/19658
- Terrain pack: ($19.50) https://www.assetstore.unity3d.com/en/#!/content/30701
- Tower construction completion effect: ($9.99) https://assetstore.unity.com/packages/vfx/particles/spells/quest-and-rpg-fx2-29983

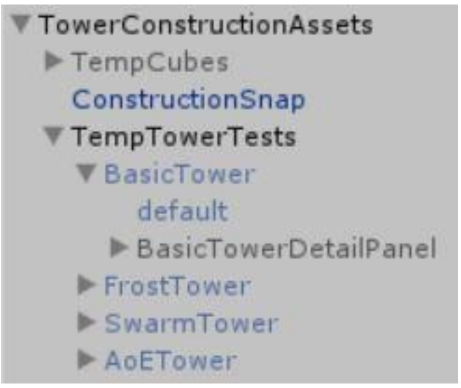
Free explosion effects did not suit the desired visual output, but one explosion effect in the pack above was the right type. The terrain pack was not necessary for the prototype. The terrain pack's assets were stripped, and the heightmap significantly altered. The changes left some remnants of the terrain heightmap and the ground textures partially intact. Originally additional features were to be included to provide trees, rocks, and other visual assets surrounding the game area. These were excluded to leave the focus on the gameplay.

Free Assets

- Free Medieval House: https://www.assetstore.unity3d.com/en/#!/content/31856
- Free Medieval Tower: https://www.assetstore.unity3d.com/en/#!/content/51230
- Trailer background music: http://www.freesound.org/people/Setuniman/sounds/155407/
- Tower Needing Repair Fire Effect: https://www.assetstore.unity3d.com/en/#!/content/50735

The free assets selected all fit well into the desired theme. The medieval house was used as a base for the participant to protect. The towers were modified with different colour hues to differentiate the four different towers used in the game and the tower texture was reused for the construction task blocks.

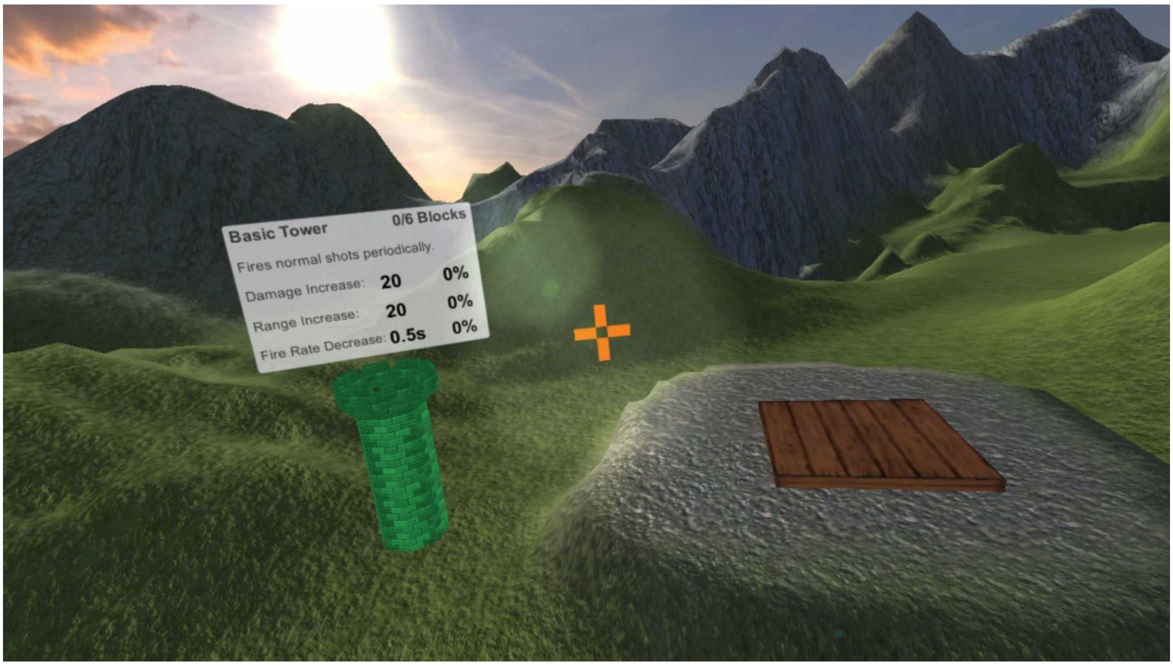
B.1.6 Configuration of Tower Construction Task



```
▼ TowerConstructionAssets
   ▶ TempCubes
     ConstructionSnap
   ▼ TempTowerTests
      ▼ BasicTower
          default
        ▶ BasicTowerDetailPanel
      ▶ FrostTower
      ▶ SwarmTower
      ▶ AoETower
```

Figure B.6: Tower Construction Task Unity Object Hierarchy

The Unity hierarchy of the tower construction task can be seen in Figure B.6. Each different tower had its own model and an interface panel to show the stats related to the specific tower. The **`ConstructionSnap`** is represented by the brown wooden block seen in Figure B.7. This block indicated where modifiers would be stacked through the drag-and-drop interactions. The dialog shown above the tower indicates a few specific details. At the top right is the number of assigned blocks to show how many modifiers have been applied. The other two values for each property type represent the current value on the left and the percentage modifier on the right.



Basic Tower 0/6 Blocks

Fires normal shots periodically.

Damage Increase:	20	0%
Range Increase:	20	0%
Fire Rate Decrease:	0.5s	0%

Figure B.7: Tower Construction Task First Tower Starting View

B.1.7 Configuration of Tower Defence Task Wave Definitions

The waves of enemies were predefined using formatted strings of numbers. Each **`WaveCommand`** was separated by a semi-colon (;). Each **`WaveCommand`** consisted of a **`timeToWait`** and a **`unitIDToSpawn`** separated by a colon (:) between them. For example, **`timeToWait:untiIDToSpawn`** could be 1:0 to spawn unit type 0 after 1 second. -1:-1 was used to represent the end of a wave. The data has been separated for clarity.

- Stage One Definition (3 waves)
  - Wave 1: (5 basic): 1:0;1:0;1:0;1:0;1:0;-1:-1;
  - Wave 2: ((2 basic, 2 fast) twice): 1:0;1:0;1:2;1:2;1:0;1:0;1:2;1:2;-1:-1;
  - Wave 3: ((4 basic, 4 fast) twice):
    1:0;1:0;1:0;1:0;1:2;1:2;1:2;1:2;1:0;1:0;1:0;1:0;1:2;1:2;1:2;1:2;-1:-1
- Stage Two Definition (3 waves)
  - Wave 1: (5 dangerous): 1:1;1:1;1:1;1:1;1:1;-1:-1;
  - Wave 2: ((2 fast, 1 dangerous) three times):
    1:2;1:2;1:1;1:2;1:2;1:1;1:2;1:2;1:1;-1:-1;
  - Wave 3: ((1 basic, 1 dangerous, 1 fast) three times):
    1:0;1:1;1:2;1:0;1:1;1:2;1:0;1:1;1:2;-1:-1
- Stage Three Definition (3 waves)
  - Wave 1: (12 fast): 1:2;1:2;1:2;1:2;1:2;1:2;1:2;1:2;1:2;1:2;1:2;1:2;-1:-1;
  - Wave 2: ((2 basic, 4 fast) twice, then 5 dangerous):
    1:0;1:0;1:2;1:2;1:2;1:2;1:0;1:0;1:2;1:2;1:2;1:2;1:1;1:1;1:1;1:1;-1:-1;
  - Wave 3: ((2 basic, 2 dangerous, 2 fast) three times, then 1 boss)
    1:0;1:0;1:1;1:1;1:2;1:2;1:0;1:0;1:1;1:1;1:2;1:2;1:0;1:0;1:1;1:1;1:2;1:2;5:3;-1:-1
- Boss Stage Definition (1 wave)
  - (boss, dangerous, boss, dangerous, boss, boss, dangerous, boss, dangerous, boss): 1.5:3;1.5:1;1.5:3;1.5:1;1.5:3;1.5:3;1.5:1;1.5:3;1.5:1;1.5:3;-1:-1

Listing B.1 (over page) shows how the above was represented in code. The data was all combined into the **`stateData`** variable to merge **`WaveCommand`** data with delays used for time between waves.

```
private const string FIRSTWAVEDEFINITION =
"1:0;1:0;1:0;1:0;1:0;"  // 5 basic
+ "1:0;1:0;1:2;1:2;1:0;1:0;1:2;1:2;"  // (2 basic, 2 fast) * 2
+ "1:0;1:0;1:0;1:0;1:2;1:2;1:2;1:2;1:0;1:0;1:0;1:0;1:2;1:2;1:2;1:2;-1:-1";  // (4
basic, 4 fast) * 2

private const string SECONDWAVEDEFINTION =
"1:1;1:1;1:1;1:1;1:1;" // 5 dangerous
+ "1:2;1:2;1:1;1:2;1:2;1:1;1:2;1:2;1:1;"  // (2 fast, 1 dangerous) * 3
+ "1:0;1:1;1:2;1:0;1:1;1:2;1:0;1:1;1:2;-1:-1"; // (1 basic, 1 dangerous, 1 fast) *
3

private const string THIRDWAVEDEFINTION =
"1:2;1:2;1:2;1:2;1:2;1:2;1:2;1:2;1:2;1:2;1:2;1:2;" // 12 fast
+ "1:0;1:0;1:2;1:2;1:2;1:2;1:0;1:0;1:2;1:2;1:2;1:2;1:1;1:1;1:1;1:1;"  // (2 basic,
4 fast) * 2, 5 dangerous
+ "1:0;1:0;1:1;1:1;1:2;1:2;1:0;1:0;1:1;1:1;1:2;1:2;1:0;1:0;1:1;1:1;1:2;1:2;5:3;-1:-
1;";  // (2 basic, 2 dangerous, 2 fast) * 3, 1 boss

// Boss, Dangerous, Boss, Dangerous, Boss, Boss, Dangerous, Boss, Dangerous, Boss
private const string BOSSWAVEDEFINTION =
"1.5:3;1.5:1;1.5:3;1.5:1;1.5:3;1.5:3;1.5:1;1.5:3;1.5:1;1.5:3;-1:-1";

private string[] stateData = new string[]{ "", // Wait for begin

                                "60", // Initial Construction (60 seconds)

                                FIRSTWAVEDEFINITION, // First Wave Set

                                "60", // First Down Time (60 seconds)

                                SECONDWAVEDEFINTION, // Second Wave Set

                                "60", // Second Down Time (60 seconds)

                                THIRDWAVEDEFINTION, // Third Wave Set

                                "60", // Third Down Time (60 seconds)

                                BOSSWAVEDEFINTION, // Boss Wave

                                ""}; // Complete
```

Listing B.1: Experiment 2 Wave Definitions

B.2 Second Experiment Materials Provided to Participants

The materials that will appear in this section follow a very similar structure to that of the first experiment. They have mostly iterated with improvements and are used in the same way. The individual materials do not explain their use, so each section is briefly discussed and linked in the following text as was done for the first experiment materials.

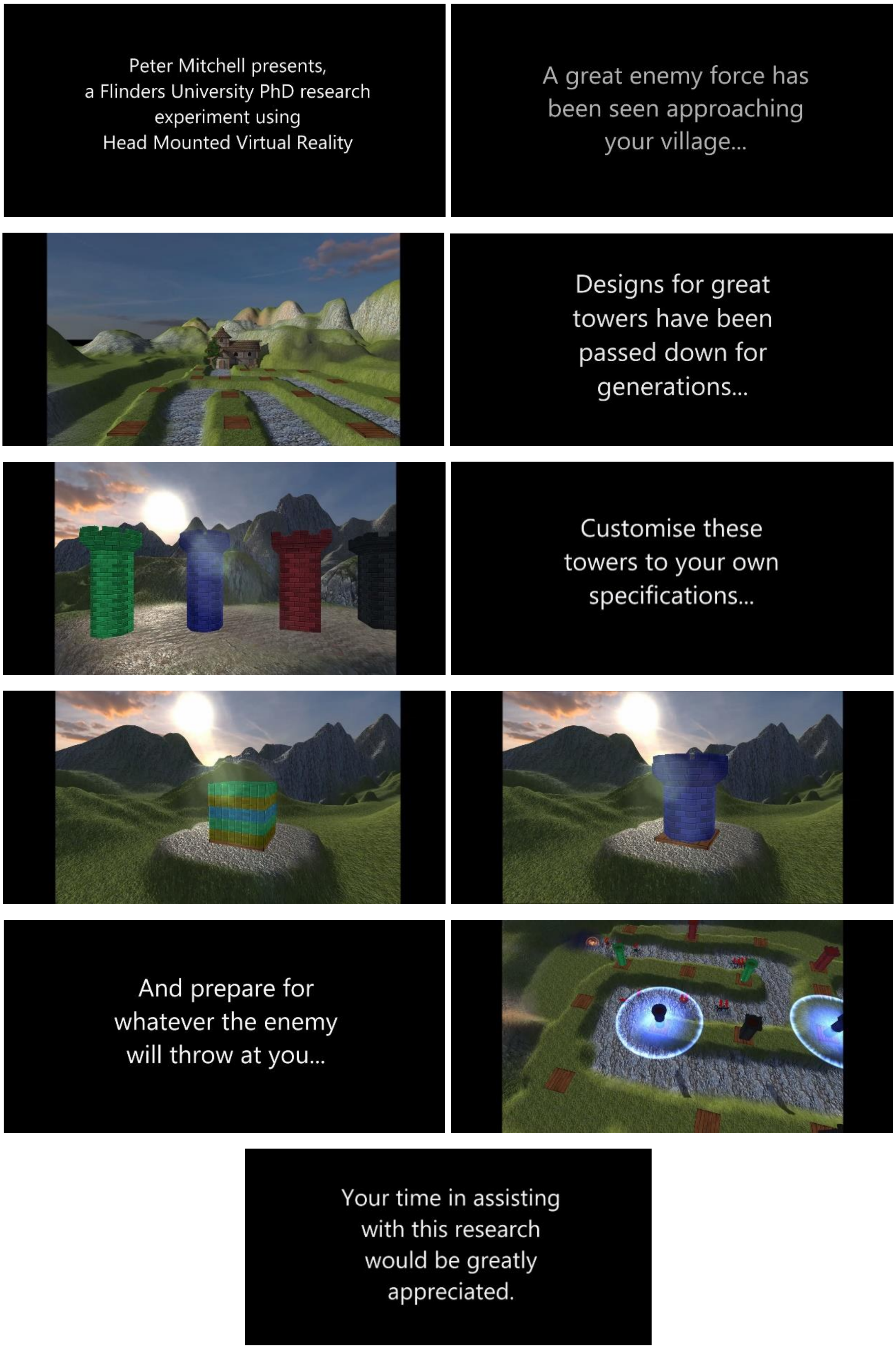
A storyboard of the trailer used to recruit participants can be found in B.2.1. The ethics approval email in B.2.2 was used to identify the project number and show it was approved in any material or communication with potential participants. The first line of communication with potential participants was with the recruitment email seen in B.2.3.

Individuals interested in seeking to participate would respond to the recruitment email and be sent back an information pack email as seen in B.2.4. Along with the email, participants were provided with the information sheet B.2.5, letter of introduction B.2.6, and consent form B.2.7. The email included a link to a google spreadsheet where available times were shown. Sending with the link had the advantage of not having to send a finalised list of times, and it would update based on updates by the researcher. Participants could be reasonably confident the times available on the spreadsheet were still available.

When attending and participating in the experiment, participants would be provided with a physical copy of the information sheet B.2.5 to review again if they wanted to and a consent form B.2.7 to sign if they wanted to participate. After signing the consent form in B.2.7, the pre-experiment questionnaire B.2.8 was provided. After conducting the experiment, the participants were provided with the post-experiment questionnaire in B.2.9.

B.2.1 Second Experiment YouTube Trailer Storyboard

The sequence of frames is left to right, then top to bottom.



Peter Mitchell presents,
a Flinders University PhD research
experiment using
Head Mounted Virtual Reality

A great enemy force has
been seen approaching
your village...

Designs for great
towers have been
passed down for
generations...

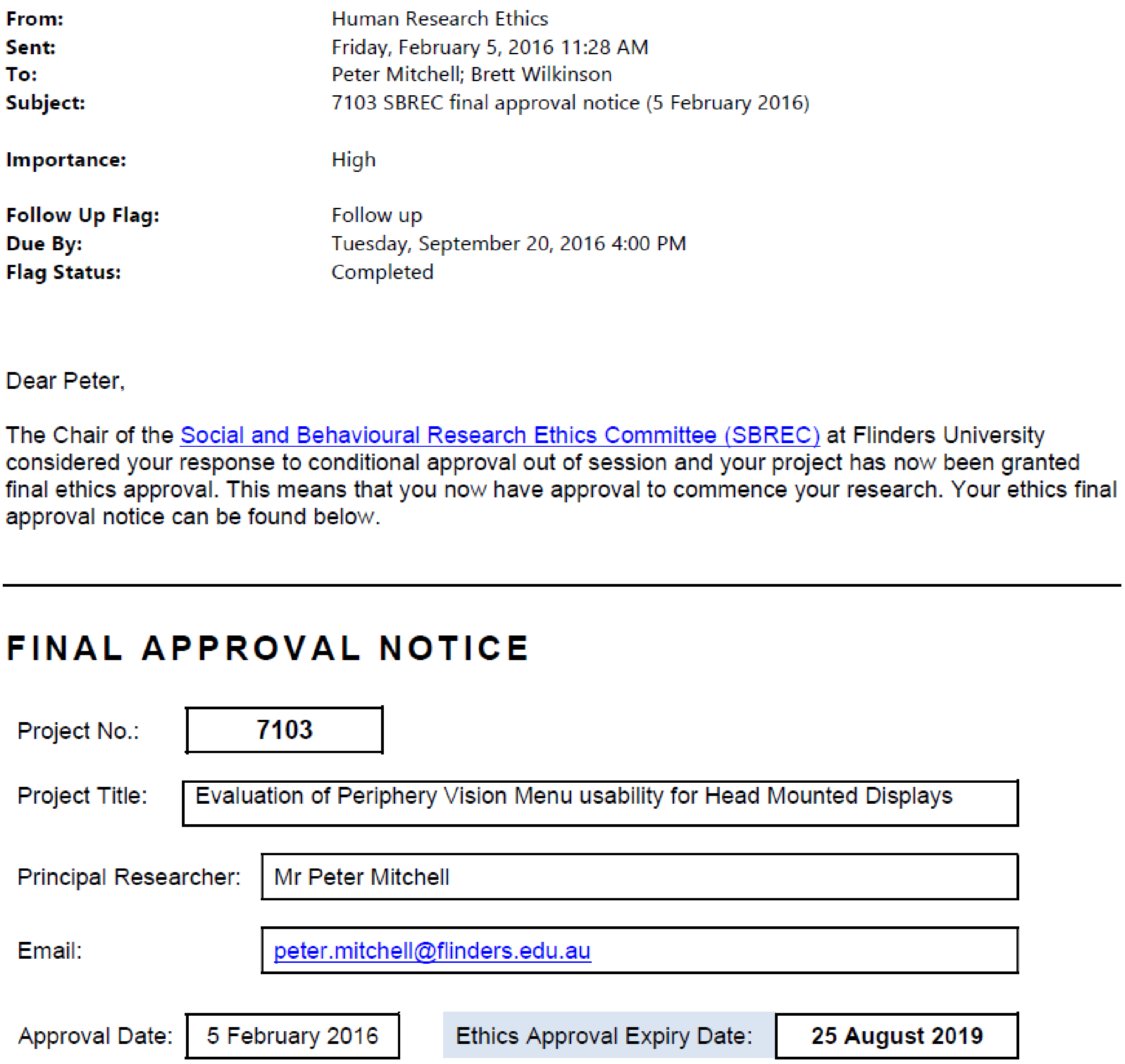
Customise these
towers to your own
specifications...

And prepare for
whatever the enemy
will throw at you...

Your time in assisting
with this research
would be greatly
appreciated.

Figure B.8: Second Experiment Trailer Storyboard

B.2.2 Second Experiment Ethics Approval Email



From:	Human Research Ethics
Sent:	Friday, February 5, 2016 11:28 AM
To:	Peter Mitchell; Brett Wilkinson
Subject:	7103 SBREC final approval notice (5 February 2016)
Importance:	High
Follow Up Flag:	Follow up
Due By:	Tuesday, September 20, 2016 4:00 PM
Flag Status:	Completed

Dear Peter,

The Chair of the Social and Behavioural Research Ethics Committee (SBREC) at Flinders University considered your response to conditional approval out of session and your project has now been granted final ethics approval. This means that you now have approval to commence your research. Your ethics final approval notice can be found below.

FINAL APPROVAL NOTICE

Project No.:	**7103**		
Project Title:	Evaluation of Periphery Vision Menu usability for Head Mounted Displays		
Principal Researcher:	Mr Peter Mitchell		
Email:	peter.mitchell@flinders.edu.au		
Approval Date:	5 February 2016	Ethics Approval Expiry Date:	**25 August 2019**

B.2.3 Second Experiment Recruitment Email

Hello

This email is to introduce Peter Mitchell who is a PhD student in the School of Computer Science, Engineering, and Mathematics at Flinders University. He will produce his student card, which carries a photograph, as proof of identity. He is undertaking research leading to the production of a thesis on the subjects of head mounted user interaction, virtual reality (VR), augmented reality (AR) usability, and human-computer interaction (HCI).

He would like to invite you to assist with this project by participating in an experiment designed to evaluate the usability of different head mounted device input methods for use with head mounted device applications. This research will provide evaluation of the effectiveness of the proposed interface system in this study. This will help aid future development of novel interfaces for use with head mounted displays in both virtual and augmented realities.

To assist with this research, he is seeking volunteers to participate in a brief experiment. The experiment will involve participants completing interactions with a couple of different input methods while playing a Tower Defence game. As part of this game you will be completing a number of tasks, in addition to two questionnaires, one before attempting the tasks and one following their completion.

The following trailer will introduce you to the game that you will be playing as part of this experiment:

https://www.youtube.com/watch?v=bVtB0wj8ehI

The study will be conducted in the Tonsley building at Flinders University. The time commitment required is expected to be around 60 minutes. Participation in this study is completely voluntary and no penalties will be incurred by choosing not to participate. Your participation will be treated anonymously and you will be free to withdraw at any time without consequence. Supervisors and your lecturers will not know who has agreed to participate and who has not.

If you are interested in participating or would like further information, please email Peter Mitchell at peter.mitchell@flinders.edu.au. Additional information has also been attached to this email to read through if you wish. You do not need to print a consent form yourself, these will be provided when completing the experiment. A list of available time-slots for participation will be provided to you to provide a list of flexible options. If for whatever reason you do not wish to participate, simply ignore this email. Any additional enquiries or concerns regarding this project that can't be directed to Peter Mitchell may be directed by telephone (REMOVED) or email (REMOVED).

Thank you for your time, attention, and assistance.

Yours sincerely

Dr Brett Wilkinson

B.2.4 Second Experiment Information Pack Email

Hello {name},

Thank you for requesting further information. Please find attached an information pack containing an information sheet, letter of introduction, and consent form.

If after reading the documents in the information pack, you would like to participate in the experiment, you can view available times for participating at the following Google Docs sheet. (You won’t be able to edit the sheet yourself)

https://docs.google.com/spreadsheets/d/1vNyJBGkNP877NNx0ziGvAlvzoIh0ZKV6S5g7awv7WP0/edit?usp=sharing

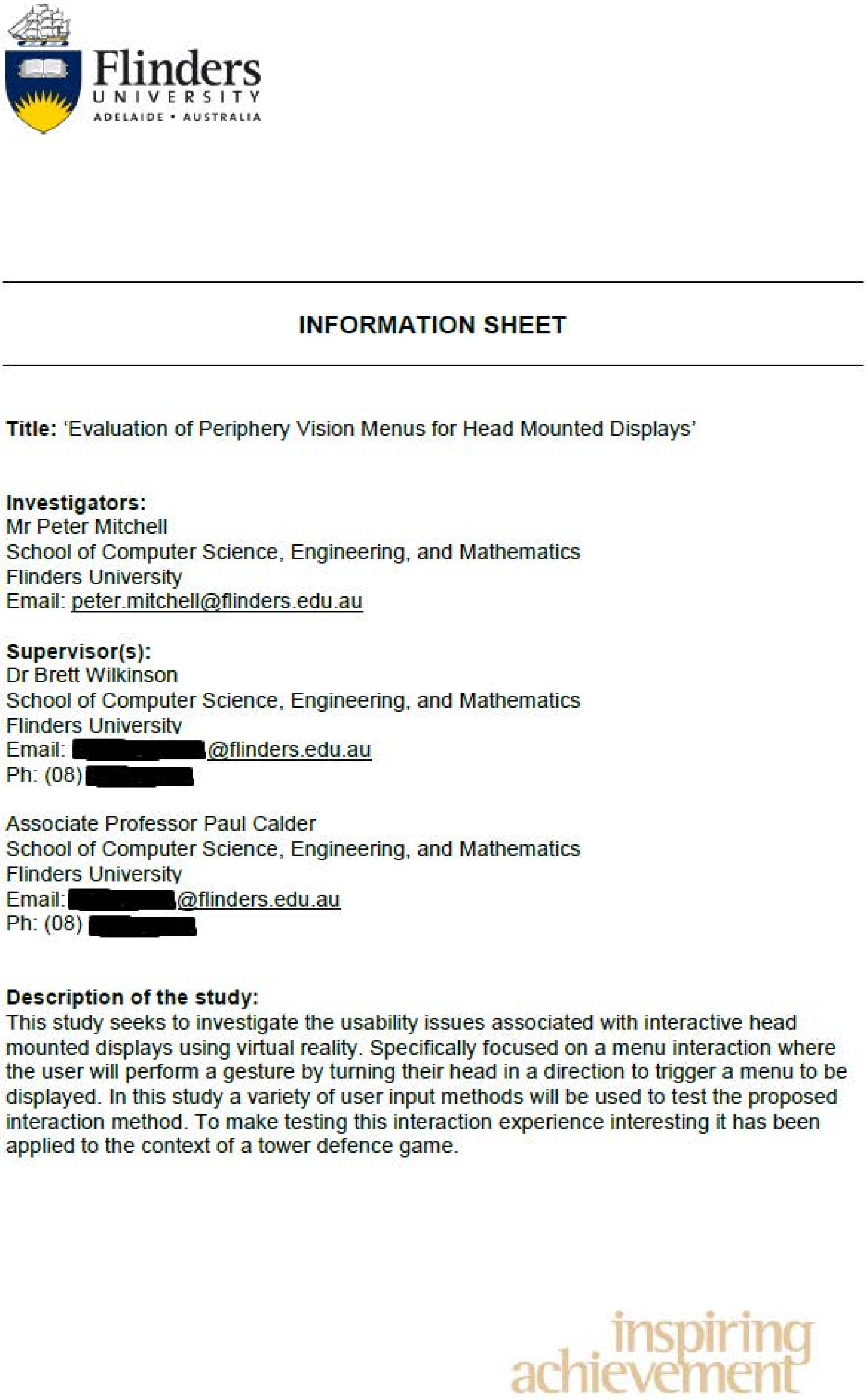
You can reply to indicate any times you wish that have not been taken yet on the link above. If none of the times available work for you and you are still interested let me know and we can work something out. The location of the experiment will be in the Interactive Research Lab on 4th floor at Tonsley. Consent forms will be provided when you come, or you may choose to complete the form in advance and bring it with you.

If for whatever reason you do not wish to participate, you do not need to respond. This email in no way commits you to participate.

If you have any questions regarding any of the material in the information pack or the experiment, please don’t hesitate to contact me.

Thanks,
Peter

B.2.5 Second Experiment Information Sheet



Flinders
UNIVERSITY
ADELAIDE • AUSTRALIA

INFORMATION SHEET

Title: 'Evaluation of Periphery Vision Menus for Head Mounted Displays'

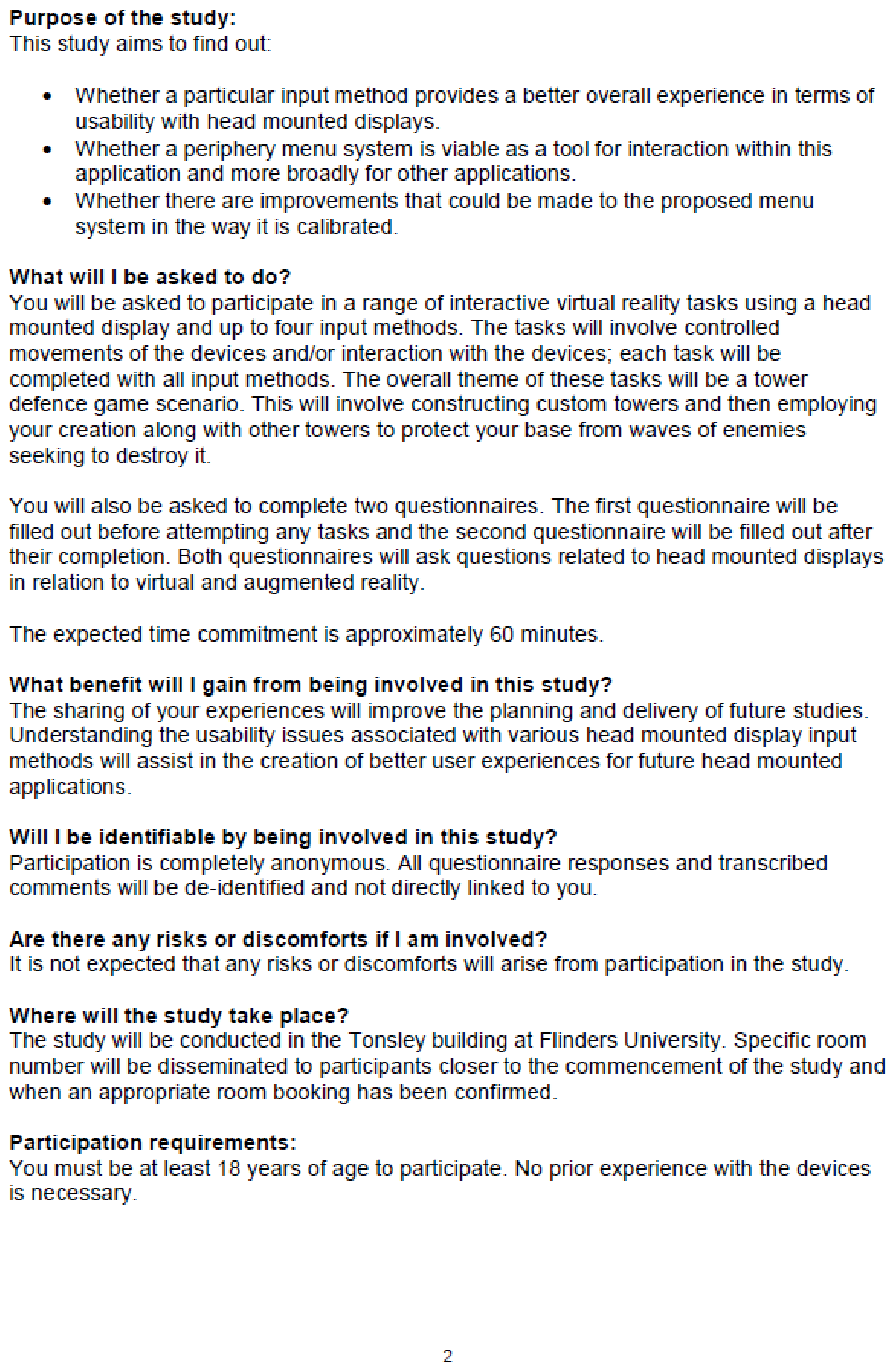
Investigators:
Mr Peter Mitchell
School of Computer Science, Engineering, and Mathematics
Flinders University
Email: peter.mitchell@flinders.edu.au

Supervisor(s):
Dr Brett Wilkinson
School of Computer Science, Engineering, and Mathematics
Flinders University
Email: [redacted]@flinders.edu.au
Ph: (08) [redacted]

Associate Professor Paul Calder
School of Computer Science, Engineering, and Mathematics
Flinders University
Email: [redacted]@flinders.edu.au
Ph: (08) [redacted]

Description of the study:
This study seeks to investigate the usability issues associated with interactive head mounted displays using virtual reality. Specifically focused on a menu interaction where the user will perform a gesture by turning their head in a direction to trigger a menu to be displayed. In this study a variety of user input methods will be used to test the proposed interaction method. To make testing this interaction experience interesting it has been applied to the context of a tower defence game.

inspiring achievement



Purpose of the study:
This study aims to find out:

- Whether a particular input method provides a better overall experience in terms of usability with head mounted displays.
- Whether a periphery menu system is viable as a tool for interaction within this application and more broadly for other applications.
- Whether there are improvements that could be made to the proposed menu system in the way it is calibrated.

What will I be asked to do?
You will be asked to participate in a range of interactive virtual reality tasks using a head mounted display and up to four input methods. The tasks will involve controlled movements of the devices and/or interaction with the devices; each task will be completed with all input methods. The overall theme of these tasks will be a tower defence game scenario. This will involve constructing custom towers and then employing your creation along with other towers to protect your base from waves of enemies seeking to destroy it.

You will also be asked to complete two questionnaires. The first questionnaire will be filled out before attempting any tasks and the second questionnaire will be filled out after their completion. Both questionnaires will ask questions related to head mounted displays in relation to virtual and augmented reality.

The expected time commitment is approximately 60 minutes.

What benefit will I gain from being involved in this study?
The sharing of your experiences will improve the planning and delivery of future studies. Understanding the usability issues associated with various head mounted display input methods will assist in the creation of better user experiences for future head mounted applications.

Will I be identifiable by being involved in this study?
Participation is completely anonymous. All questionnaire responses and transcribed comments will be de-identified and not directly linked to you.

Are there any risks or discomforts if I am involved?
It is not expected that any risks or discomforts will arise from participation in the study.

Where will the study take place?
The study will be conducted in the Tonsley building at Flinders University. Specific room number will be disseminated to participants closer to the commencement of the study and when an appropriate room booking has been confirmed.

Participation requirements:
You must be at least 18 years of age to participate. No prior experience with the devices is necessary.

How do I agree to participate?

You can agree to participate by responding to the email with your preferred time slots as well as completing the attached consent form. You will receive a confirmation email with a selected time from those indicated as well as the location of the experiment. Consent forms will be provided when attending to participate, but you may bring a pre-completed form if desired.

Participation in this study is voluntary. You may withdraw at any time without consequence.

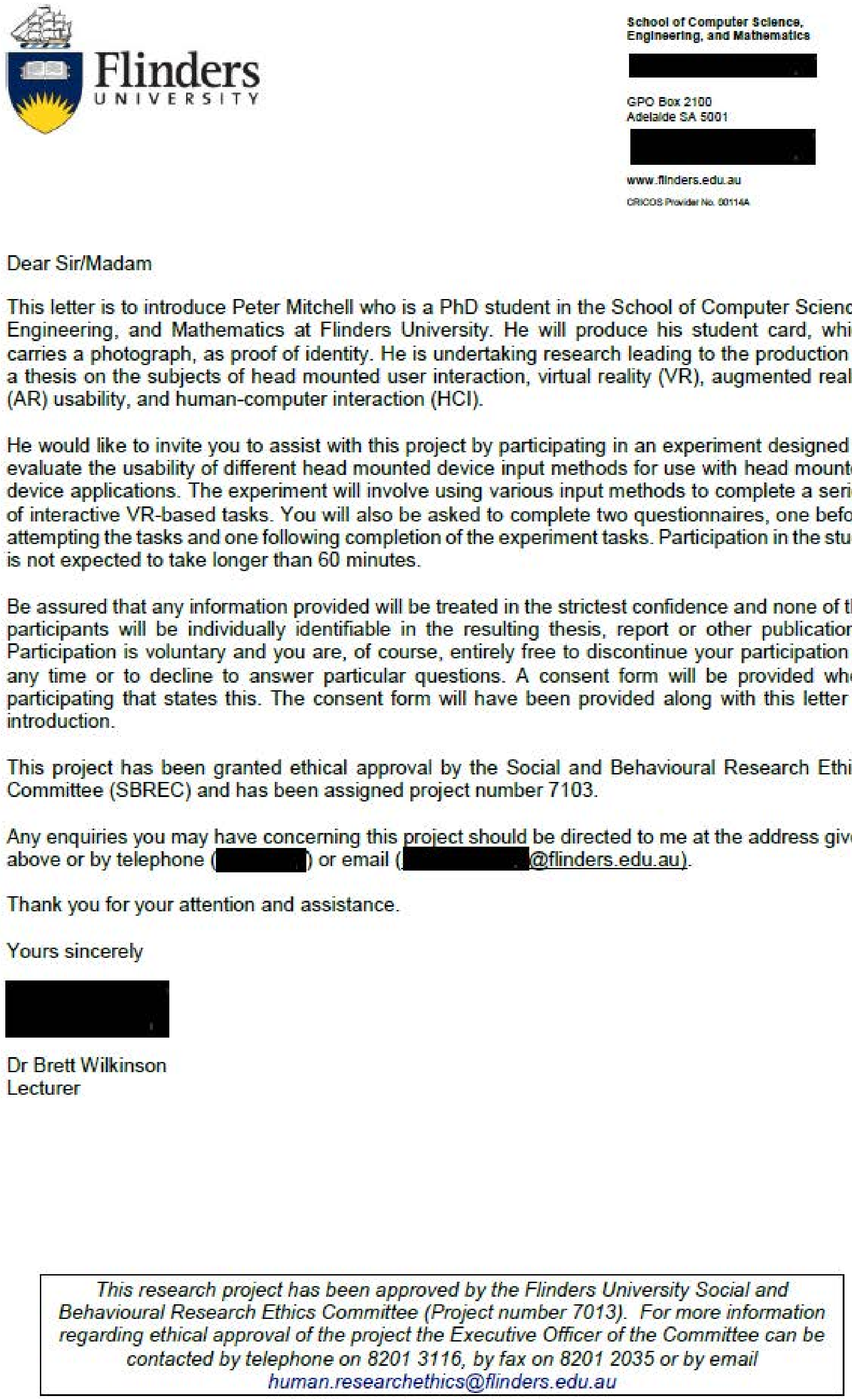
How will I receive feedback?

It is anticipated that the results of this study will be published in a journal or conference article. The results will also be included in the principle researcher's PhD thesis.

Thank you for taking the time to read this information sheet and we hope that you will accept our invitation to be involved.

This research project has been approved by the Flinders University Social and Behavioural Research Ethics Committee (7103). For more information regarding ethical approval of the project the Executive Officer of the Committee can be contacted by telephone on 8201 3116, by fax on 8201 2035 or by email human.researchethics@flinders.edu.au

B.2.6 Second Experiment Letter of Introduction



Flinders
UNIVERSITY

School of Computer Science,
Engineering, and Mathematics
[redacted]

GPO Box 2100
Adelaide SA 5001
[redacted]

www.flinders.edu.au
CRICOS Provider No. 00114A

Dear Sir/Madam

This letter is to introduce Peter Mitchell who is a PhD student in the School of Computer Science, Engineering, and Mathematics at Flinders University. He will produce his student card, which carries a photograph, as proof of identity. He is undertaking research leading to the production of a thesis on the subjects of head mounted user interaction, virtual reality (VR), augmented reality (AR) usability, and human-computer interaction (HCI).

He would like to invite you to assist with this project by participating in an experiment designed to evaluate the usability of different head mounted device input methods for use with head mounted device applications. The experiment will involve using various input methods to complete a series of interactive VR-based tasks. You will also be asked to complete two questionnaires, one before attempting the tasks and one following completion of the experiment tasks. Participation in the study is not expected to take longer than 60 minutes.

Be assured that any information provided will be treated in the strictest confidence and none of the participants will be individually identifiable in the resulting thesis, report or other publications. Participation is voluntary and you are, of course, entirely free to discontinue your participation at any time or to decline to answer particular questions. A consent form will be provided when participating that states this. The consent form will have been provided along with this letter of introduction.

This project has been granted ethical approval by the Social and Behavioural Research Ethics Committee (SBREC) and has been assigned project number 7103.

Any enquiries you may have concerning this project should be directed to me at the address given above or by telephone ([redacted]) or email ([redacted]@flinders.edu.au).

Thank you for your attention and assistance.

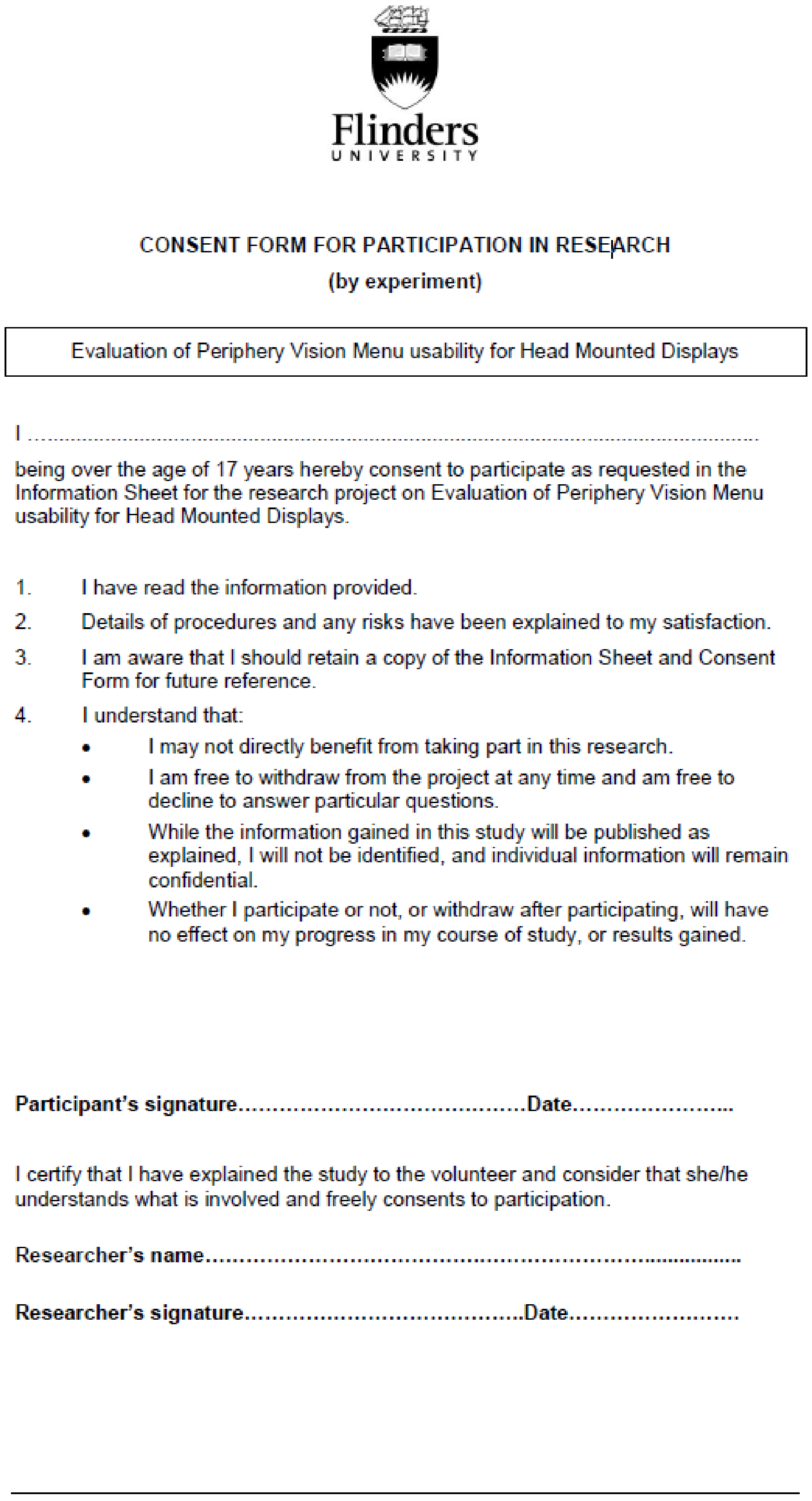
Yours sincerely

[redacted]

Dr Brett Wilkinson
Lecturer

This research project has been approved by the Flinders University Social and Behavioural Research Ethics Committee (Project number 7013). For more information regarding ethical approval of the project the Executive Officer of the Committee can be contacted by telephone on 8201 3116, by fax on 8201 2035 or by email human.researchethics@flinders.edu.au

B.2.7 Second Experiment Consent Form



Flinders
UNIVERSITY

CONSENT FORM FOR PARTICIPATION IN RESEARCH

(by experiment)

Evaluation of Periphery Vision Menu usability for Head Mounted Displays

I ....

being over the age of 17 years hereby consent to participate as requested in the Information Sheet for the research project on Evaluation of Periphery Vision Menu usability for Head Mounted Displays.

1. I have read the information provided.
2. Details of procedures and any risks have been explained to my satisfaction.
3. I am aware that I should retain a copy of the Information Sheet and Consent Form for future reference.
4. I understand that:
   - I may not directly benefit from taking part in this research.
   - I am free to withdraw from the project at any time and am free to decline to answer particular questions.
   - While the information gained in this study will be published as explained, I will not be identified, and individual information will remain confidential.
   - Whether I participate or not, or withdraw after participating, will have no effect on my progress in my course of study, or results gained.

Participant's signature...Date.......................

I certify that I have explained the study to the volunteer and consider that she/he understands what is involved and freely consents to participation.

Researcher's name...

Researcher's signature.......................................Date.........................

B.2.8 Second Experiment Pre-Experiment Questionnaire

1. Which of the following age ranges do you fall into?

Under 21	21 to 30	31 to 40	41 to 50	51 and Over

2. What is your gender?

Male	Female

3. Which of the following apply to your current situation?

Student	Academic Researcher	Teaching Staff Member	Other

4. As a student or academic researcher: what is your area of study/research? ____

5. Have you previously **participated in any research** as a volunteer (or researcher) for a project involving one of the following? (tick as many as apply to you, and if possible suggest what projects they were):

   [ ] Virtual Reality : ____

   [ ] Augmented Reality: ____

   [ ] Head Mounted Displays :____

   [ ] Mobile/Tablet Computing :____

6. Have you previously used any of the following for **personal use** outside of research? (tick as many as apply to you, and if possible suggest what applications or contexts they were used in):

   [ ] Virtual Reality : ____

   [ ] Augmented Reality:____

   [ ] Head Mounted Displays: ____

   [ ] Mobile/Tablet Computing:____

7. How many hours a week would you use a computer on average?

Less than 10 hours	10 to 20 hours	20 to 30 hours	30 to 40 hours	More than 40 hours

8. How many hours a week would you spend playing video games on average?

Less than 10 hours	10 to 20 hours	20 to 30 hours	30 to 40 hours	More than 40 hours

9. If you play computer games, what are a few examples of games you play and the devices you play those games on?

____

10. On a scale of 1 to 10 (1 being not interested and 10 being very interested), how interested would you be in using a head mounted display for activities other than gaming?

Not Interested									Very Interested
1	2	3	4	5	6	7	8	9	10

11. On a scale of 1 to 10 (1 being never having played a tower defence game and 10 being very often playing) how often do you play tower defence type games?

Never									Very Often
1	2	3	4	5	6	7	8	9	10

12. When considering the usability of a user interface for head mounted displays with augmented reality or virtual reality what features do you believe are the most important? (Rank features from 1 to 4, where 1 is most important and 4 is least important).
    a. ___ The speed of accessing features.
    b. ___ The accuracy of accessing features.
    c. ___ The simplicity of physical interaction required.
    d. ___ The visual appeal of the interface.

    Any other features you feel are important:

    ____

13. In regard to devices for interaction how would you rank the following for personal preference? (1 is most desired, and 4 is least desired).
    a. ___ **Oculus Rift by itself**. A standalone experience with no other peripherals required. Using an approach of looking at objects to interact and the rotation of head to make menus appear.
    b. ___ **Oculus Rift with a computer mouse**. The same form of interaction as using the Oculus Rift by itself, but with the tactile interaction of being able to click to provide direct menu interactions.
    c. ___ **Oculus Rift with a mobile device**. The same as the mouse, but using the touch surface of a mobile to provide tactile interactive feedback.
    d. ___ **Oculus Rift with the LEAP or Microsoft Kinect Sensors**. An approach using hands free where the hands are detected and used as a form of gesture input themselves to signify actions for selection and manipulation.

14. Are there any other devices or interaction methods you are particularly fond of that you would like to see used for user interface interaction within head mounted displays for augmented or virtual reality?

    ____

    ____

15. On a scale of 1 to 10 (1 being no influence and 10 being high influence), how much did the game trailer influence your desire to participate in this research?

No Influence									High Influence
1	2	3	4	5	6	7	8	9	10

16. On a scale of 1 to 10 (1 being no influence and 10 being high influence), how do you feel visual presentations such as the game trailer would influence you to participate in future research?

No Influence									High Influence
1	2	3	4	5	6	7	8	9	10

B.2.9 Second Experiment Post-Experiment Questionnaire

Please answer the following questions in relation to the tasks you have just completed.

The Periphery Vision Menu Interaction

1. Did you find the periphery menu system to be useful in the way it appeared? (Y/N)

Why? ____

2. On a scale of 1 to 10 (1 is not accurate, 10 is very accurate), how accurately did the periphery menu system respond when you wanted to make it display?

Not Accurate									Very Accurate
1	2	3	4	5	6	7	8	9	10

3. On a scale of 1 to 10 (1 is not often, 10 is very often), how often did the periphery menu system display at the **wrong times** or when you didn't mean to display it?

Not Often									Very Often
1	2	3	4	5	6	7	8	9	10

4. On a scale of 1 to 10 (1 is not often, 10 is very often), how often did the periphery menu system display at the **correct times**?

Not Often									Very Often
1	2	3	4	5	6	7	8	9	10

5. On a scale of 1 to 10 (1 is not useful, 10 is very useful), how useful do you feel the gesture of rotating your head to make a menu appear?

Not Useful									Very Useful
1	2	3	4	5	6	7	8	9	10

6. Is there anything you would change about the periphery menu system in respect to the way the system is interacted with? Eg, either anything to do with the way the gesture works, or the hardware being used.

____

____

7. On a scale of 1 to 10 (1 is not likely, 10 is very likely), how likely would it be for you to want to use this interaction in the future?

Not Likely									Very Likely
1	2	3	4	5	6	7	8	9	10

8. What three aspects did you find most **enjoyable** about Periphery Vision Menu?

1. ____
2. ____
3. ____

9. What three aspects did you find most **difficult** about using the Periphery Vision Menu?

1. ____
2. ____
3. ____

10. Please circle the number that best describes your feelings toward each of the following in regard to the periphery vision menus.

a. I think that I would like to use this menu system frequently.

Strongly Disagree				Strongly Agree
1	2	3	4	5

b. I found the menu unnecessarily complex.

Strongly Disagree				Strongly Agree
1	2	3	4	5

c. I thought the menu was easy to use.

Strongly Disagree				Strongly Agree
1	2	3	4	5

d. I thought that I would need the support of a technical person to be able to use this menu.

Strongly Disagree				Strongly Agree
1	2	3	4	5

e. I felt that the options presented by the menu were well integrated.

Strongly Disagree				Strongly Agree
1	2	3	4	5

f. I felt that there was too much inconsistency with the menu.

Strongly Disagree				Strongly Agree
1	2	3	4	5

g. I would imagine that most people would learn how to use these menus very quickly.

Strongly Disagree				Strongly Agree
1	2	3	4	5

h. I found the menus very cumbersome to use.

Strongly Disagree				Strongly Agree
1	2	3	4	5

i. I felt very confident using the menus.

Strongly Disagree				Strongly Agree
1	2	3	4	5

j. I needed to learn a lot of things before I could use the menus.

Strongly Disagree				Strongly Agree
1	2	3	4	5

11. Please rank the following from 1 to 5 (1 is most desired, 5 is least desired). Of the following what would you desire to use the menu system for most?

a. ___ As a method of interacting with **Games**.

b. ___ As controls for **Viewing a Movie** (either through the headset or as an augmented reality interface through glasses while watching a TV).

c. ___ As a tool for selecting components for **Constructing Models** (similar to the tower construction task or more complex).

d. ___ As a method for accessing **instant messenger services or voice chat** at any time.

e. ___ As a tool for navigating common **operating system controls and menus**.

12. Are there any other scenarios that you would see yourself using this menu system?

____

____

13. (If you are unsure of what augmented reality is, please ask the researcher to define it) On a scale of 1 to 10, how useful do you feel this menu interaction would be well suited for an augmented reality scenario?

Not Suited									Very Suited
1	2	3	4	5	6	7	8	9	10

14. Would you prefer to use this interaction for virtual reality, augmented reality, or do you not have a preference either way? Why?

[ ] Virtual Reality

[ ] Augmented Reality

[ ] No preference either way

Why?

____

15. When considering the usability of a user interface for head mounted displays with augmented reality or virtual reality what features do you believe are the most important? (Rank features from 1 to 4, where 1 is most important and 4 is least important).
    a. ___ The speed of accessing features.
    b. ___ The accuracy of accessing features.
    c. ___ The simplicity of physical interaction required.
    d. ___ The visual appeal of the interface.

Any other features you feel are important:

____

Interaction Devices

16. Now that you have completed the tasks using the Oculus by Itself; in regard to devices for interaction how would you rank the following for personal preference? (1 is most desired, and 4 is least desired).
    a. ___ **Oculus Rift by itself**. A standalone experience with no other peripherals required. Using an approach of looking at objects to interact and the rotation of head to make menus appear.
    b. ___ **Oculus Rift with a computer mouse**. The same form of interaction as using the Oculus Rift by itself, but with the tactile interaction of being able to click to provide direct menu interactions.
    c. ___ **Oculus Rift with a mobile device**. The same as the mouse, but using the touch surface of a mobile to provide tactile interactive feedback.
    d. ___ **Oculus Rift with the LEAP or Microsoft Kinect Sensors**. An approach using hands free where the hands are detected and used as a form of gesture input themselves to signify actions for selection and manipulation.

17. Are there any other devices or interaction methods you are particularly fond of that you would like to see used for user interface interaction within head mounted displays for augmented or virtual reality?

    ____

    ____

18. On a scale of 1 to 10, how much fatigue did you feel while using the Oculus Rift by Itself for performing the menu interactions?

Low Fatigue									High Fatigue
1	2	3	4	5	6	7	8	9	10

19. On a scale of 1 to 10 (1 is Strongly Disagree, 10 is Strongly Agree), do you feel the Oculus Rift by itself with the provided functionality provides enough functionality to stand alone?

Strongly Disagree									Strongly Agree
1	2	3	4	5	6	7	8	9	10

20. Do you have any other thoughts that might be useful in regard to anything you experienced during this experiment?

____

____

____

____

Thank you for taking the time to participate in this experiment.

C Appendix C: Additional Third Experiment Details

This appendix provides additional materials related to the third experiment, including additional implementation details and the materials provided to participants.

C.1 Third Experiment Additional Information

The first part covers similar sections to those in the first and second experiments. Starting with the level management discussed collectively in C.1.1, supporting information on the level change interactions in C.1.2, elaborating on the class files used in C.1.3, and the Unity game object hierarchy in C.1.4. Many assets were reused from the second experiment detailed in B.1.5, and additional assets for the third experiment are listed in C.1.7.

Unity scene hierarchy for the experiment is discussed in C.1.5 related to Task 1 and 2 and in C.1.6 for Task 3 and 4. In the final appendix section (C.1.8), the waves data used for experiment three's tower defence game can be found. More details beyond the definition can be seen in the wave data information for the second experiment in B.1.7.

C.1.1 Level Management

This section will cover three different aspects related to the level management within the Unity project for the experiment. Mostly this is technical concerning what files and structures were used for the project while keeping the information brief.

C.1.2 Level Change Interactions

Figure C.1 shows the top-down view of the full experiment's testing area. As with the second experiment, everything was included in a single scene to remove the need for loading screens. Most of the structure was sculpted out of the terrain with various rocks and different terrain textures used to create the ambience. The theme was used to create a canyon appearance leading to the cave where the creatures came from as they spawned.

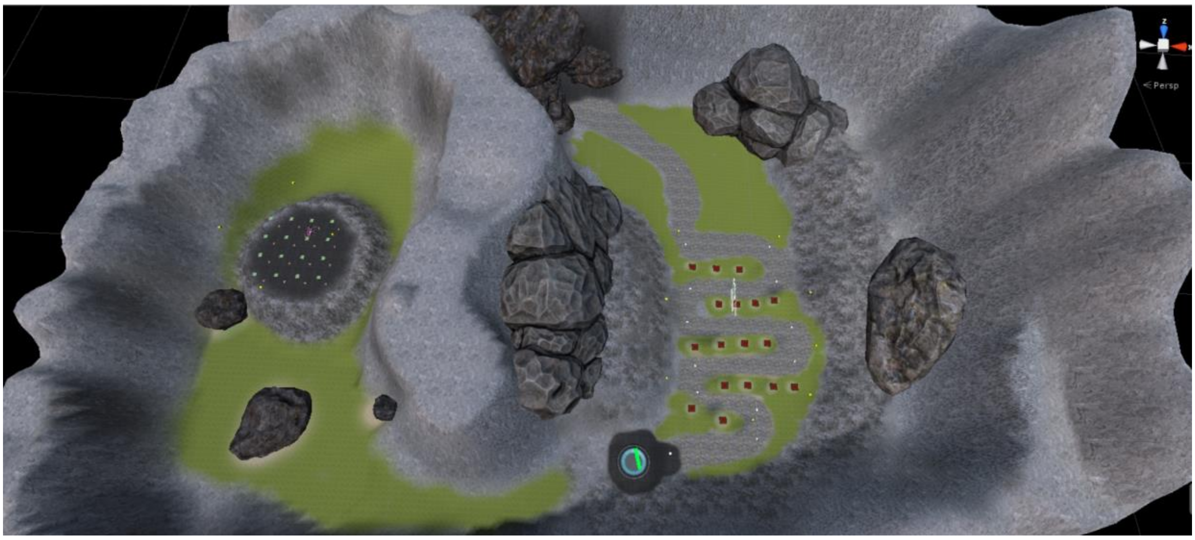


Figure C.1: Top view showing full Experiment Three.

C.1.3 Class Files

Many of the class files used in the third experiment mirrored those of the second experiment. The new classes added as part of new features and the object matching task are listed below. In addition to these classes, the script files presented in section B.1.3 were included for this experiment. The files included from the second experiment excluded those related to calibration and tower construction.

- **`ObjMatchTaskScripts/DeseletButtonBehaviour.cs`**: Provides functionality to a single deselect button related to a **`MatchSnapBehaviour`**.
- **`ObjMatchTaskScripts/MatchObjectBehaviour.cs`**: Controls the size, position, shape, and colour of an object modified by the participant.
- **`ObjMatchTaskScripts/MatchObjectManager.cs`**: Maintains the game state with a collection of **`MatchObjectBehaviour`** and **`MatchObjectSolBehaviour`** objects. Responsible for spawning the **`MatchObjectSolBehaviour`**s to provide the solution and comparison operation to determine a win state.
- **`ObjMatchTaskScripts/MatchObjectSolBehaviour.cs`**: Like the **`MatchObjectBehaviour`** except that it is spawned with defined properties remaining the same for a single puzzle.
- **`ObjMatchTaskScripts/MatchSnapBehaviour.cs`**: Represents a location in the grid with references to a possible **`MatchObjectBehaviour`** (or

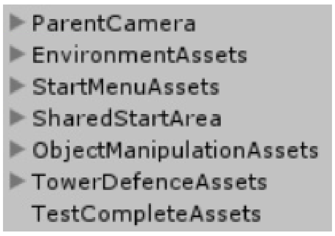
MatchObjectSolBehaviour if there is anything present at that location), storing the individual cell if both the behaviours match in properties. If the solution has been met at this **MatchSnap,** then the **TickBoxBehaviour** is enabled.

- **ObjMatchTaskScripts/TickBoxBehaviour.cs**: forces the tick box to face the camera when a **MatchSnapBehaviour** recognises a correct solution for a single object.
- **PeripheryMenuScripts/TwoStepBehaviour.cs**: Provides the two-step behaviour as described in the Technology Overview chapter.
- **TowerDefenceScripts/TowerStatData.cs**: With the removal of the tower construction task for configuring the towers, this class was added to serve the same data where it would not be modified.
- **TowerDefenceScripts/LevelObjectScripts/SpawnAndDestroyEffect.cs**: One of the additional features added was a visual effect when each tower spawns. This script was attached to the separate effect object to spawn the tower then destroy the effect object after a specified time.
- **TrailerScripts/ExplosionSimulationManager.cs**: Used to manage the collection of rocks for the explosion animation. It was used for the trailer and again for when the final boss spawns with the same effect.
- **TrailerScripts/RockMoveBehaviour.cs**: Updates an individual rock's position from a start point approaching an endpoint with a Sine function for height and Lerp function for the position. Also performed a rotation on the rocks to make them look more natural as they flew to the target locations.
- **TrailerScripts/SpawnObjectSequencer.cs**: Used during the trailer for the towers spawning as a sequence. With 0.3 second pauses between making predefined objects visible from a list of arbitrary game objects.
- **TrailerScripts/SpawnSomeStuff.cs**: Used to simulate the action sequence of the trailer by spawning a wave of enemies and updating dummy towers where the tower damage was set to 0.
- **TrailerScripts/TrailerCamera.cs**: Acts as a multi-sequence camera. Sequences were defined with a start point and endpoint with variable rotations,

rotation and panning speeds and connected to the other trailer scripts to spawn content needed for sequences.

C.1.4 Unity Game Object Hierarchy

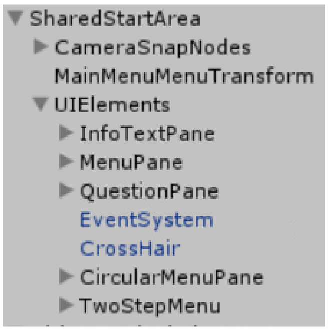
The various categories of game objects within the Unity Hierarchy provided a way to view how the overall project was constructed. The full compressed hierarchy is seen in Figure C.2. The categories matched similar themes to how they were in the second experiment. The camera was kept separate from other objects so that it could move independently. The expanded object was unchanged from the second experiment.



```
▶ ParentCamera
▶ EnvironmentAssets
▶ StartMenuAssets
▶ SharedStartArea
▶ ObjectManipulationAssets
▶ TowerDefenceAssets
  TestCompleteAssets
```

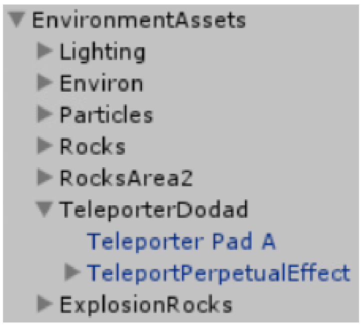
Figure C.2: Unity Hierarchy Scene Overview

The three objects after the **ParentCamera** each provided non-task specific features. **StartMenuAssets** included some graphical objects to show while the player was viewing the menu. The **SharedStartArea** and **EnvironmentalAssets** shared multiple parts between tasks, as are seen in Figure C.3 and Figure C.4. The two new additions under **SharedStartArea** were the **CircularMenuPane** and **TwoStepMenu**. The **CircularMenuPane** reused the menu code from **MenuBehaviour**. The **TwoStepMenu** represented the visual elements used for the **TwoStepBehaviour**. **EnvironmentalAssets** shows the kinds of elements used within the overall scene for aesthetic effect.



```
▼ SharedStartArea
   ▶ CameraSnapNodes
     MainMenuMenuTransform
   ▼ UIElements
      ▶ InfoTextPane
      ▶ MenuPane
      ▶ QuestionPane
        EventSystem
        CrossHair
      ▶ CircularMenuPane
      ▶ TwoStepMenu
```

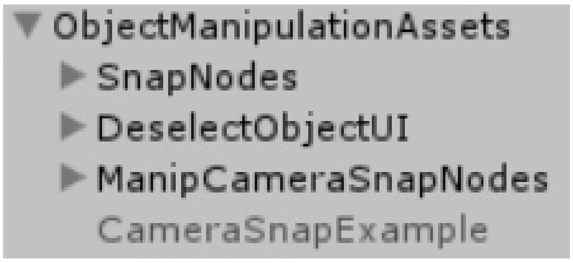
Figure C.3: Unity Hierarchy Shared Assets



```
▼ EnvironmentAssets
   ▶ Lighting
   ▶ Environ
   ▶ Particles
   ▶ Rocks
   ▶ RocksArea2
   ▼ TeleporterDodad
        Teleporter Pad A
      ▶ TeleportPerpetualEffect
   ▶ ExplosionRocks
```

Figure C.4: Unity Hierarchy Environment Assets

C.1.5 Unity Scene Configuration for Task 1 and Task 2



```
▼ ObjectManipulationAssets
    ▶ SnapNodes
    ▶ DeselectObjectUI
    ▶ ManipCameraSnapNodes
      CameraSnapExample
```

Figure C.5: Object Matching Unity Hierarchy

Figure C.5 shows the Unity Hierarchy for both Task 1 and Task 2. Most of these elements are visible in Figure C.6. The **`SnapNodes`** are shown as the small brown cubes in Figure C.6. Each of these provides the element for a single node in the grid used for possible puzzle locations. This grid's nodes were used to keep track of the solution, so each node has its own **`TickBoxBehaviour`** seen as the ticks in Figure C.6. They remain hidden until a puzzle element has been solved on that node. The **`DeselectObjectUI`** was used for Task 1 to provide a single button for deselecting the current object. **`ManipCameraSnapNodes`** are all seen in Figure C.6 as the yellow cubes. The bottom right node was the default for this puzzle. The final **`CameraSnapExample`** element was seen in Figure C.6 as the example yellow cube.

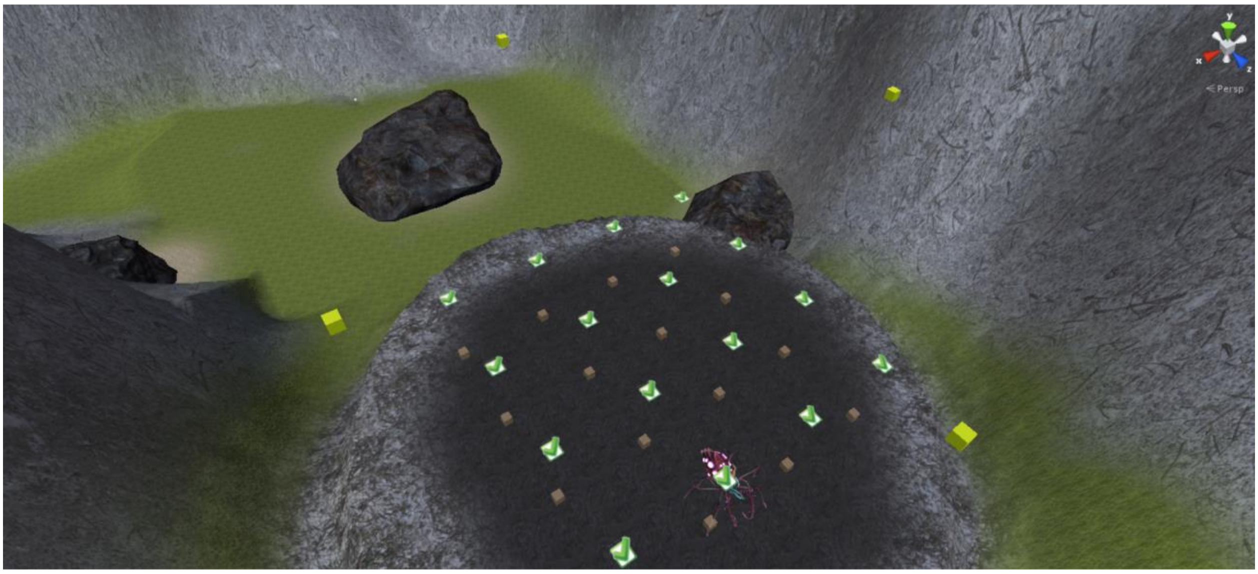
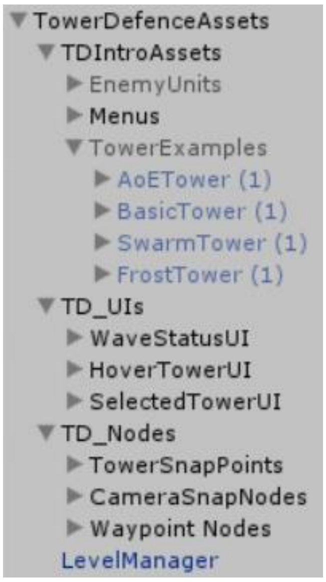


Figure C.6: Object Matching Top View in Editor

C.1.6 Unity Scene Configuration for Task 3 and Task 4

Figure C.7 **(over)** shows an expanded view of the Unity Hierarchy for this task. This hierarchy is almost identical to the categories of game objects in the second experiment. The only visible difference is the addition of **`TowerExamples`** to the **`TDIntroAssets`**. The hierarchy has been included here to show the same overall structure was used as in the second experiment (section 5.2.8 and Figure 5.38) for this task. Figure C.8 shows the full view of the new level built for this experiment. The player's base and health bar are shown as the teleporter pad on the left. Enemies spawn from the cave on the right. The small yellow cubes are the points for camera snapping. The small white cubes represent the waypoint nodes for enemy pathfinding. The red cubes mark all the places to create towers. The path is much wider in this environment compared to the second experiment, specifically to give more room for the final boss.



```
▼ TowerDefenceAssets
  ▼ TDIntroAssets
    ▶ EnemyUnits
    ▶ Menus
    ▼ TowerExamples
      ▶ AoETower (1)
      ▶ BasicTower (1)
      ▶ SwarmTower (1)
      ▶ FrostTower (1)
  ▼ TD_UIs
    ▶ WaveStatusUI
    ▶ HoverTowerUI
    ▶ SelectedTowerUI
  ▼ TD_Nodes
    ▶ TowerSnapPoints
    ▶ CameraSnapNodes
    ▶ Waypoint Nodes
    LevelManager
```

Figure C.7: Tower Defence Task Unity Hierarchy

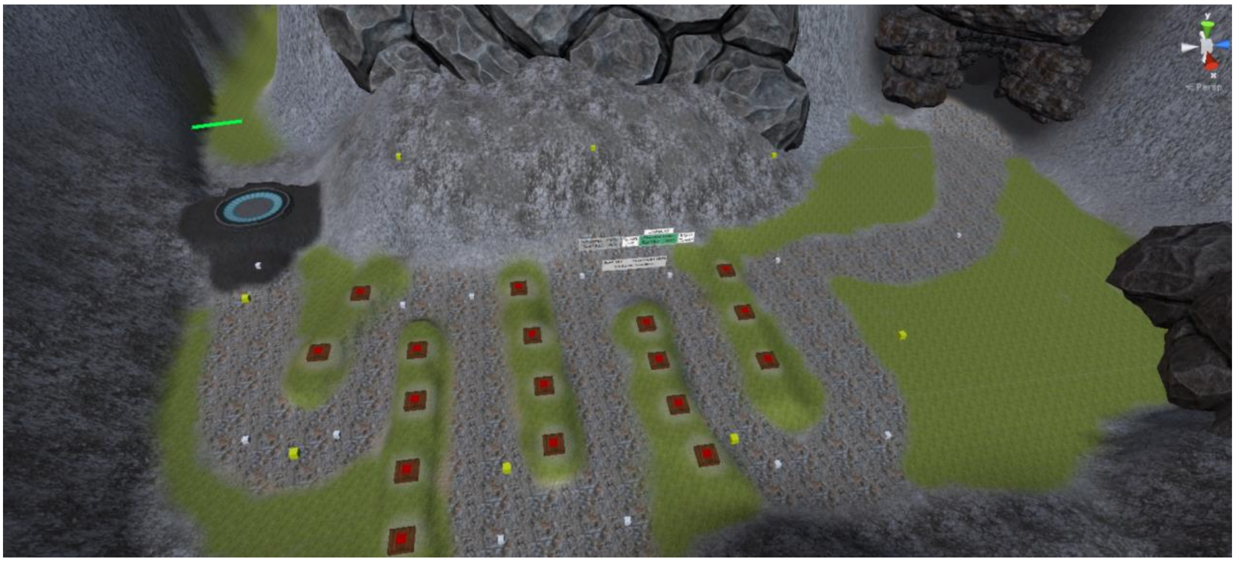


Figure C.8: Tower Defence Task with Wide Perspective

C.1.7 Unity Store Assets

Art and audio assets were utilised from 3rd parties to reduce development time. Particularly as the development of assets was determined to be out of scope when it came to research and collecting data. The art assets from the second experiment were all reused in some capacity. Those assets will not be relisted here and can be referenced in section B.1.5.

Only one new asset was purchased to provide the new enemy threat. The new boss was deliberately not shown in the trailer for an air of mystery. The new creature was mostly selected for the way it fitted into the existing theme. And how the creature worked well as a very large creature to show the vastly more dangerous kind of enemy. The other free assets were mostly audio assets to improve the quality of the trailer, in addition to an explosion effect for the cave explosion.

New Enemy

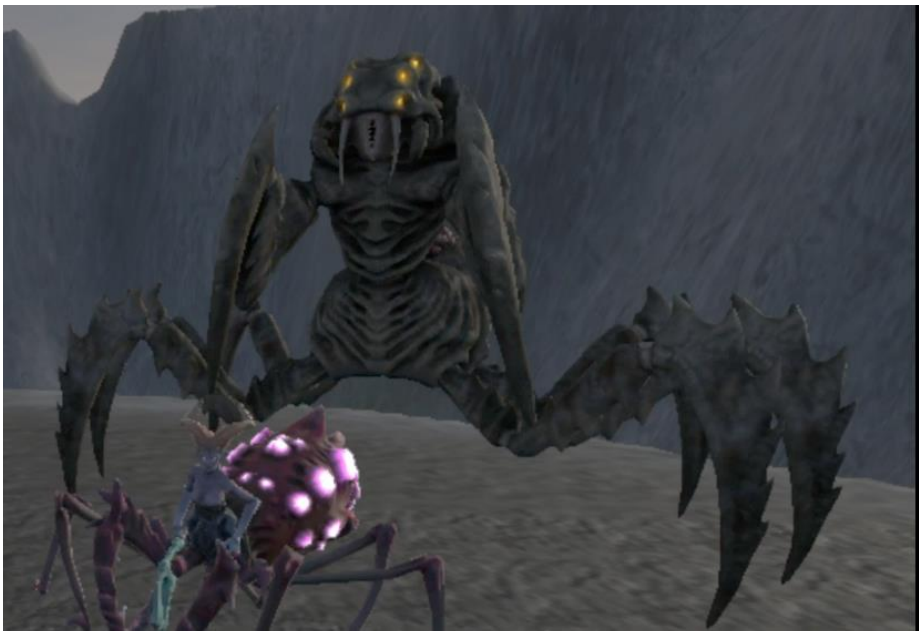


Figure C.9: Final Boss Creature Model

- Final Boss (Figure C.9): ($24.90) https://assetstore.unity.com/packages/3d/characters/creatures/cavecrawler-54650

New Free Assets

- Cave Explosion Effect: https://assetstore.unity.com/packages/vfx/particles/detonator-explosion-framework-1
- Trailer Audio:
  - Creature Noise: https://freesound.org/people/noahpardo/sounds/345735/
  - Drum Noise: https://freesound.org/people/limetoe/sounds/342465/
  - Background music (Ice Of Phoenix by Audiomachine): https://www.youtube.com/watch?v=XUeQ0Ew_Wh0

C.1.8 Configuration of Tower Defence Task Wave Definitions

This section presents full form versions of the wave definitions showing the **`WaveCommand`** structured data. Additional details can be found in section B.1.7. The only difference from the second experiment's data structures is the addition of a final boss as part of the boss stage definition.

- Stage One Definition (3 waves)
  - Wave 1: (5 basic): 1:0;1:0;1:0;1:0;1:0;-1:-1;
  - Wave 2: ((2 basic, 2 fast) twice): 1:0;1:0;1:2;1:2;1:0;1:0;1:2;1:2;-1:-1;
  - Wave 3: ((4 basic, 4 fast) twice): 1:0;1:0;1:0;1:0;1:2;1:2;1:2;1:2;1:0;1:0;1:0;1:0;1:2;1:2;1:2;1:2;-1:-1
- Stage Two Definition (3 waves)
  - Wave 1: (5 dangerous): 1:1;1:1;1:1;1:1;1:1;-1:-1;
  - Wave 2: ((2 fast, 1 dangerous) three times): 1:2;1:2;1:1;1:2;1:2;1:1;1:2;1:2;1:1;-1:-1;
  - Wave 3: ((1 basic, 1 dangerous, 1 fast) three times): 1:0;1:1;1:2;1:0;1:1;1:2;1:0;1:1;1:2;-1:-1
- Stage Three Definition (3 waves)
  - Wave 1: (12 fast): 1:2;1:2;1:2;1:2;1:2;1:2;1:2;1:2;1:2;1:2;1:2;1:2;-1:-1;
  - Wave 2: ((2 basic, 4 fast) twice, then 5 dangerous): 1:0;1:0;1:2;1:2;1:2;1:2;1:0;1:0;1:2;1:2;1:2;1:2;1:1;1:1;1:1;1:1;-1:-1;

  - Wave 3: ((2 basic, 2 dangerous, 2 fast) three times, then 1 boss)
    1:0;1:0;1:1;1:1;1:2;1:2;1:0;1:0;1:1;1:1;1:2;1:2;1:0;1:0;1:1;1:1;1:2;1:2;5:3;-1:-1
- Boss Stage Definition (1 wave)
  - (boss, dangerous, boss, dangerous, boss, boss, dangerous, boss, dangerous, boss, final boss):
    1.5:3;1.5:1;1.5:3;1.5:1;1.5:3;1.5:3;1.5:1;1.5:3;1.5:1;1.5:3;5:4;-1:-1

Listing C.1 (over page) shows the code used for representing the above information along with the different delay timers between states compared to experiment two.

```
private const string FIRSTWAVEDEFINITION =
"1:0;1:0;1:0;1:0;1:0;"  // 5 basic
+ "1:0;1:0;1:2;1:2;1:0;1:0;1:2;1:2;"  // (2 basic, 2 fast) * 2
+ "1:0;1:0;1:0;1:0;1:2;1:2;1:2;1:2;1:0;1:0;1:0;1:0;1:2;1:2;1:2;1:2;-1:-1";  // (4
basic, 4 fast) * 2

private const string SECONDWAVEDEFINTION =
"1:1;1:1;1:1;1:1;1:1;" // 5 dangerous
+ "1:2;1:2;1:1;1:2;1:2;1:1;1:2;1:2;1:1;"  // (2 fast, 1 dangerous) * 3
+ "1:0;1:1;1:2;1:0;1:1;1:2;1:0;1:1;1:2;-1:-1"; // (1 basic, 1 dangerous, 1 fast) *
3

private const string THIRDWAVEDEFINTION =
"1:2;1:2;1:2;1:2;1:2;1:2;1:2;1:2;1:2;1:2;1:2;1:2;" // 12 fast
+ "1:0;1:0;1:2;1:2;1:2;1:2;1:0;1:0;1:2;1:2;1:2;1:2;1:1;1:1;1:1;1:1;"  // (2 basic,
4 fast) * 2, 5 dangerous
+ "1:0;1:0;1:1;1:1;1:2;1:2;1:0;1:0;1:1;1:1;1:2;1:2;1:0;1:0;1:1;1:1;1:2;1:2;5:3;-1:-
1;";  // (2 basic, 2 dangerous, 2 fast) * 3, 1 boss

// Boss, Dangerous, Boss, Dangerous, Boss, Boss, Dangerous, Boss, Dangerous, Boss,
Mega Boss
private const string BOSSWAVEDEFINTION =
"1.5:3;1.5:1;1.5:3;1.5:1;1.5:3;1.5:3;1.5:1;1.5:3;1.5:1;1.5:3;5:4;-1:-1";

private string[] stateData = new string[]{ "", // Wait for begin
                                "40", // Initial Construction (40 seconds)
                                FIRSTWAVEDEFINITION, // First Wave Set
                                "30", // First Down Time (30 seconds)
                                SECONDWAVEDEFINTION, // Second Wave Set
                                "30", // Second Down Time (30 seconds)
                                THIRDWAVEDEFINTION, // Third Wave Set
                                "30", // Third Down Time (30 seconds)
                                BOSSWAVEDEFINTION, // Boss Wave
                                ""}; // Complete
```

Listing C.1: Experiment 3 Wave Definitions

C.2 Third Experiment Materials Provided to Participants

Materials relevant to the participants for the third experiment are provided in the following sections. The individual sections follow a similar format to those of the first and second experiments, where there is no further elaboration on the content provided in the sections. The following paragraphs briefly summarise the content included.

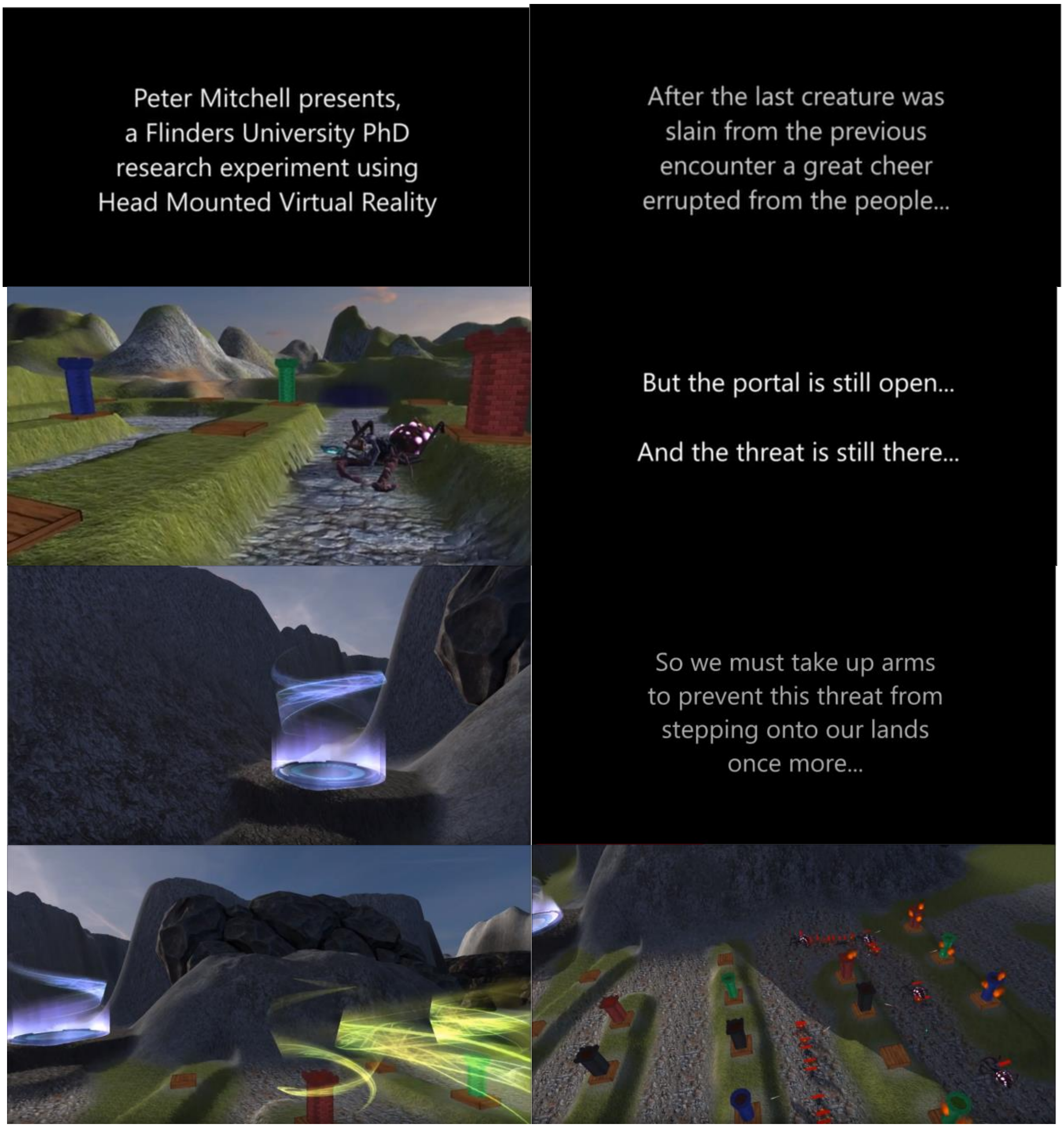
A storyboard showing the trailer used for recruiting is found in C.2.1. The ethics approval email in C.2.1 provides the approval number for communicating with participants when conducting the experiment. The recruitment email in C.2.3 was sent to participants with links to additional information. The additional information included the information sheet in C.2.6, the letter of introduction in C.2.7, and the consent form in C.2.8. In addition to sending out a general email, there was a follow up targeted email in C.2.4 that was sent to people who had participated in the previous experiments to give them the opportunity to participate if interested.

Any individual who either asked for more information or indicated an interest in participating would be sent the information email in C.2.5. This email included a link to a google spreadsheet similar to that from the previous experiment. The spreadsheet included links to the information sheet in C.2.6, the letter of introduction in C.2.7, and the consent form in C.2.8. They could email preferred times for when to participate.

When participating in the experiment, they would be given the information sheet in C.2.6 and the consent form in C.2.8 to sign. After the consent form was signed, they would be given the pre-experiment questionnaire in C.2.9. Then after the experiment was completed, it would be wrapped up with the post-experiment questionnaire in C.2.10.

C.2.1 Third Experiment YouTube Trailer Storyboard

The sequence of frames is left to right, then top to bottom. The first parts are shown on this page, and the last few frames are shown over on the following page.



Peter Mitchell presents,
a Flinders University PhD
research experiment using
Head Mounted Virtual Reality

After the last creature was
slain from the previous
encounter a great cheer
errupted from the people...

But the portal is still open...

And the threat is still there...

So we must take up arms
to prevent this threat from
stepping onto our lands
once more...

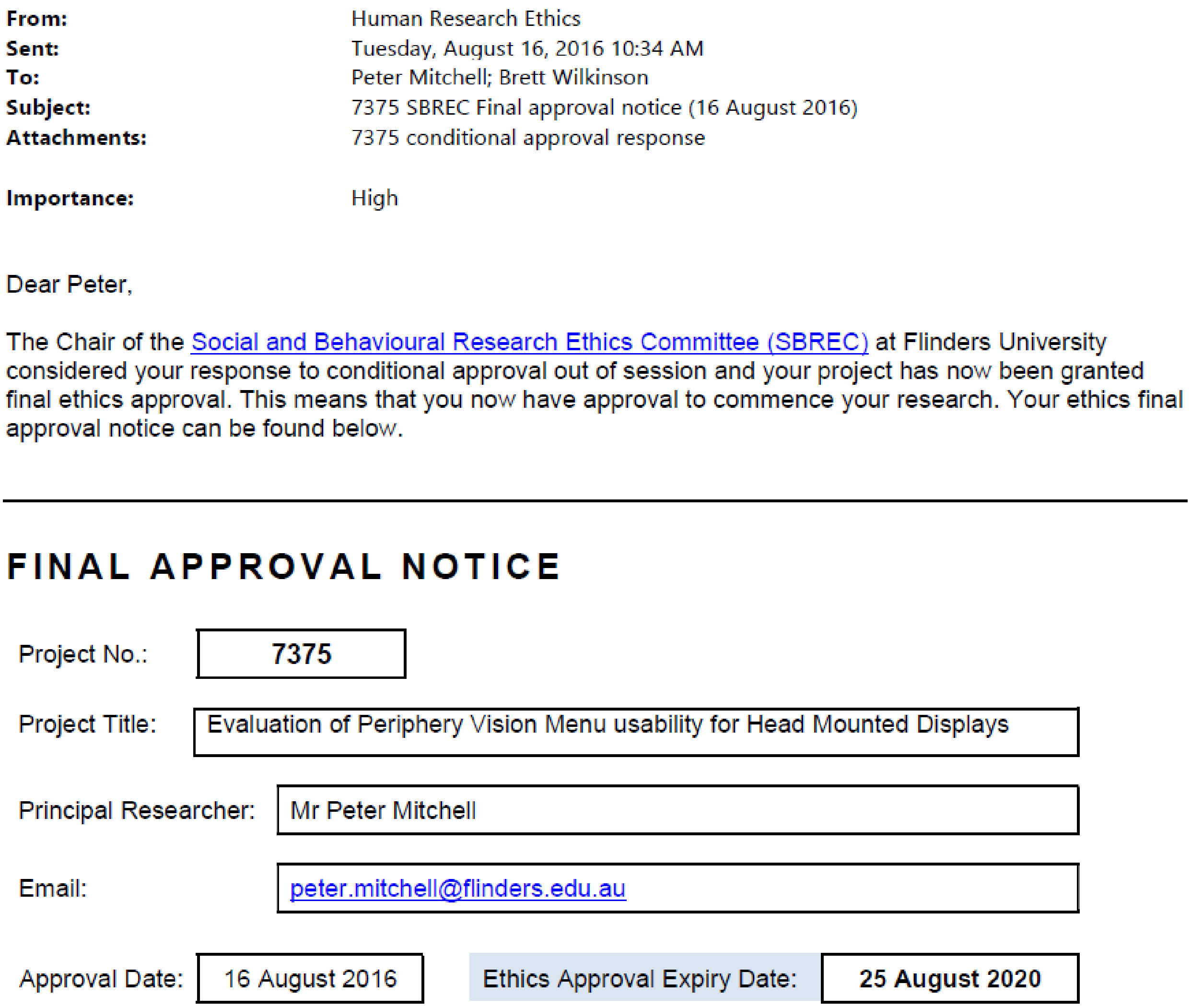
Figure C.10: Third Experiment Trailer Storyboard (part 1)

Your time in assisting with this research would be greatly appreciated.

For additional information or interest in participation please contact me at: peter.mitchell@flinders.edu.au

Figure C.11: Third Experiment Trailer Storyboard (part 2)

C.2.2 Third Experiment Ethics Approval Email



From:	Human Research Ethics
Sent:	Tuesday, August 16, 2016 10:34 AM
To:	Peter Mitchell; Brett Wilkinson
Subject:	7375 SBREC Final approval notice (16 August 2016)
Attachments:	7375 conditional approval response
Importance:	High

Dear Peter,

The Chair of the Social and Behavioural Research Ethics Committee (SBREC) at Flinders University considered your response to conditional approval out of session and your project has now been granted final ethics approval. This means that you now have approval to commence your research. Your ethics final approval notice can be found below.

FINAL APPROVAL NOTICE

Project No.: **7375**

Project Title: Evaluation of Periphery Vision Menu usability for Head Mounted Displays

Principal Researcher: Mr Peter Mitchell

Email: peter.mitchell@flinders.edu.au

Approval Date: 16 August 2016   Ethics Approval Expiry Date: **25 August 2020**

C.2.3 Third Experiment Recruitment Email

Hello,

This email is to introduce Peter Mitchell (peter.mitchell@flinders.edu.au) who is a PhD student in the School of Computer Science, Engineering, and Mathematics at Flinders University. He will produce his student card, which carries a photograph, as proof of identity. He is undertaking research leading to the production of a thesis on the subjects of head mounted user interaction, virtual reality (VR), augmented reality (AR) usability, and human-computer interaction (HCI).
As part of this final interactive experiment you will have the opportunity to use an Oculus Rift (DK2).

The experiment will involve participants completing interactions with a couple of different input methods while playing a Tower Defence game. As part of this game you will be completing a number of tasks, in addition to two questionnaires, one before attempting the tasks and one following their completion.

The following trailer will introduce you to the game that you will be playing as part of this experiment:

https://youtu.be/bVtB0wj8ehI

The study will be conducted in the Tonsley building at Flinders University. The time commitment required is expected to be around 25 to 30 minutes. Participation in this study is completely voluntary and no penalties will be incurred by choosing not to participate. Your participation will be treated anonymously and you will be free to withdraw at any time without consequence. Supervisors and your lecturers will not know who has agreed to participate and who has not.

Please Note: You will be unable to wear glasses with the Oculus Rift headset in the configuration for the experiment. If your eyesight is very poor without glasses it may be difficult to participate.

The following links may be viewed for additional information related to this project:

- Information Sheet: https://drive.google.com/open?id=0B4PJ5TA7ht1JMEJvMXNJNGJuSDQ
- Letter of Introduction: https://drive.google.com/open?id=0B4PJ5TA7ht1JeE9CNHpvY2Y5N28
- Sample Consent Form: https://drive.google.com/open?id=0B4PJ5TA7ht1JVjBLU1lwNktHbms

If you are interested in participating or would like further information, please email Peter Mitchell at peter.mitchell@flinders.edu.au. Additional information has also been attached to this email to read through if you wish. You do not need to print a consent form yourself, these will be provided when completing the experiment. A list of available time-slots for participation will be provided to you to provide a list of flexible options. If for whatever reason you do not wish to participate, simply ignore this email. Any additional enquiries or concerns regarding this project that can't be directed to Peter Mitchell may be directed by telephone (REMOVED) or email (REMOVED@flinders.edu.au).

Thank you for your time, attention, and assistance.

Yours sincerely

Dr Brett Wilkinson

C.2.4 Third Experiment Targeted Recruitment Email

Hello,

You are receiving this email as you previously participated in one of my experiments. I am emailing previous participants to let them know I am running a final experiment and would like to invite you to participate. This experiment will be similar to the previous experiment with improvements, but importantly will require a lower time commitment. The following is the email text sent to all students to inform you of the details of this experiment.

The experiment will involve participants completing interactions with a couple of different input methods while playing a Tower Defence game. As part of this game you will be completing a number of tasks, in addition to two questionnaires, one before attempting the tasks and one following their completion.

The following trailer will introduce you to the game that you will be playing as part of this experiment:

https://youtu.be/bVtB0wj8ehI

The study will be conducted in the Tonsley building at Flinders University. The time commitment required is expected to be around 25 to 30 minutes. Participation in this study is completely voluntary and no penalties will be incurred by choosing not to participate. Your participation will be treated anonymously and you will be free to withdraw at any time without consequence. Supervisors and your lecturers will not know who has agreed to participate and who has not.

Please Note: You will be unable to wear glasses with the Oculus Rift headset in the configuration for the experiment. If your eyesight is very poor without glasses it may be difficult to participate.

The following links may be viewed for additional information related to this project:

- Information Sheet: https://drive.google.com/open?id=0B4PJ5TA7ht1JMEJvMXNJNGJuSDQ
- Letter of Introduction: https://drive.google.com/open?id=0B4PJ5TA7ht1JeE9CNHpvY2Y5N28
- Sample Consent Form: https://drive.google.com/open?id=0B4PJ5TA7ht1JVjBLU1lwNktHbms

If you are interested in participating or would like further information, please email Peter Mitchell at peter.mitchell@flinders.edu.au. Additional information has also been attached to this email to read through if you wish. You do not need to print a consent form yourself, these will be provided when completing the experiment. A list of available time-slots for participation will be provided to you to provide a list of flexible options. If for whatever reason you do not wish to participate, simply ignore this email. Any additional enquiries or concerns regarding this project that can't be directed to Peter Mitchell may be directed by telephone (REMOVED) or email (REMOVED@flinders.edu.au).

Thank you for your time, attention, and assistance.

Yours sincerely

Peter Mitchell

C.2.5 Third Experiment Information Pack Email

Hello,

Thank you for your interest in participation.

You can view available times for participating at the following Google Docs sheet. (You won’t be able to edit the sheet yourself)
https://docs.google.com/spreadsheets/d/1vNyJBGkNP877NNx0ziGvAlvzoIh0ZKV6S5g7awv7WP0/edit?usp=sharing

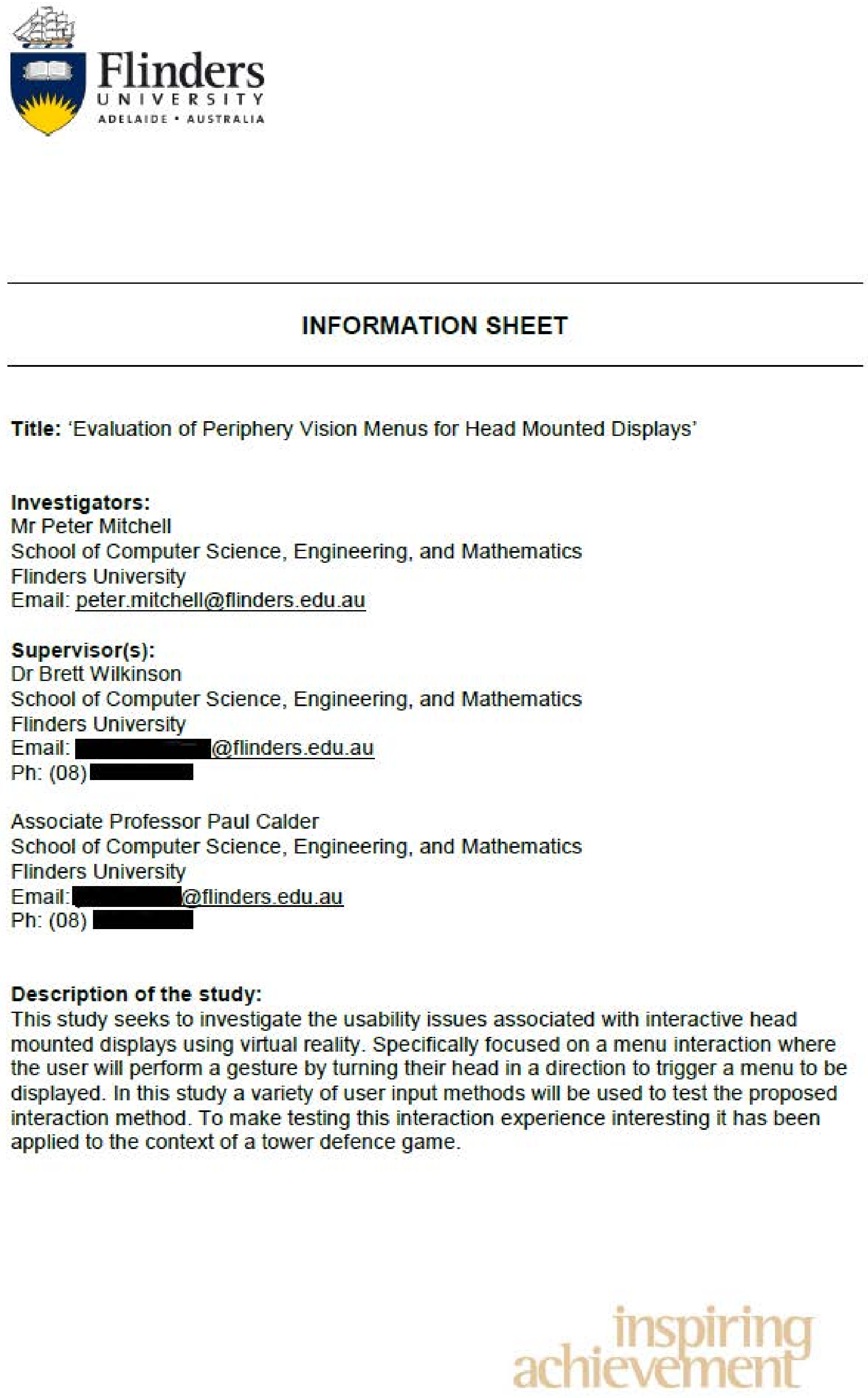
You can reply to indicate any times you wish that have not been taken yet on the link above. If none of the times available work for you and you are still interested let me know and we can work something out. The location of the experiment will be in the Interactive Research Lab on 4th floor at Tonsley. Consent forms will be provided when you come, there is no need to print one off.

If for whatever reason you do not wish to participate, you do not need to respond. This email in no way commits you to participate.

If you have any questions regarding anything to do with the experiment, please don’t hesitate to contact me.

Thanks,
Peter

C.2.6 Third Experiment Information Sheet



Flinders
UNIVERSITY
ADELAIDE • AUSTRALIA

INFORMATION SHEET

Title: 'Evaluation of Periphery Vision Menus for Head Mounted Displays'

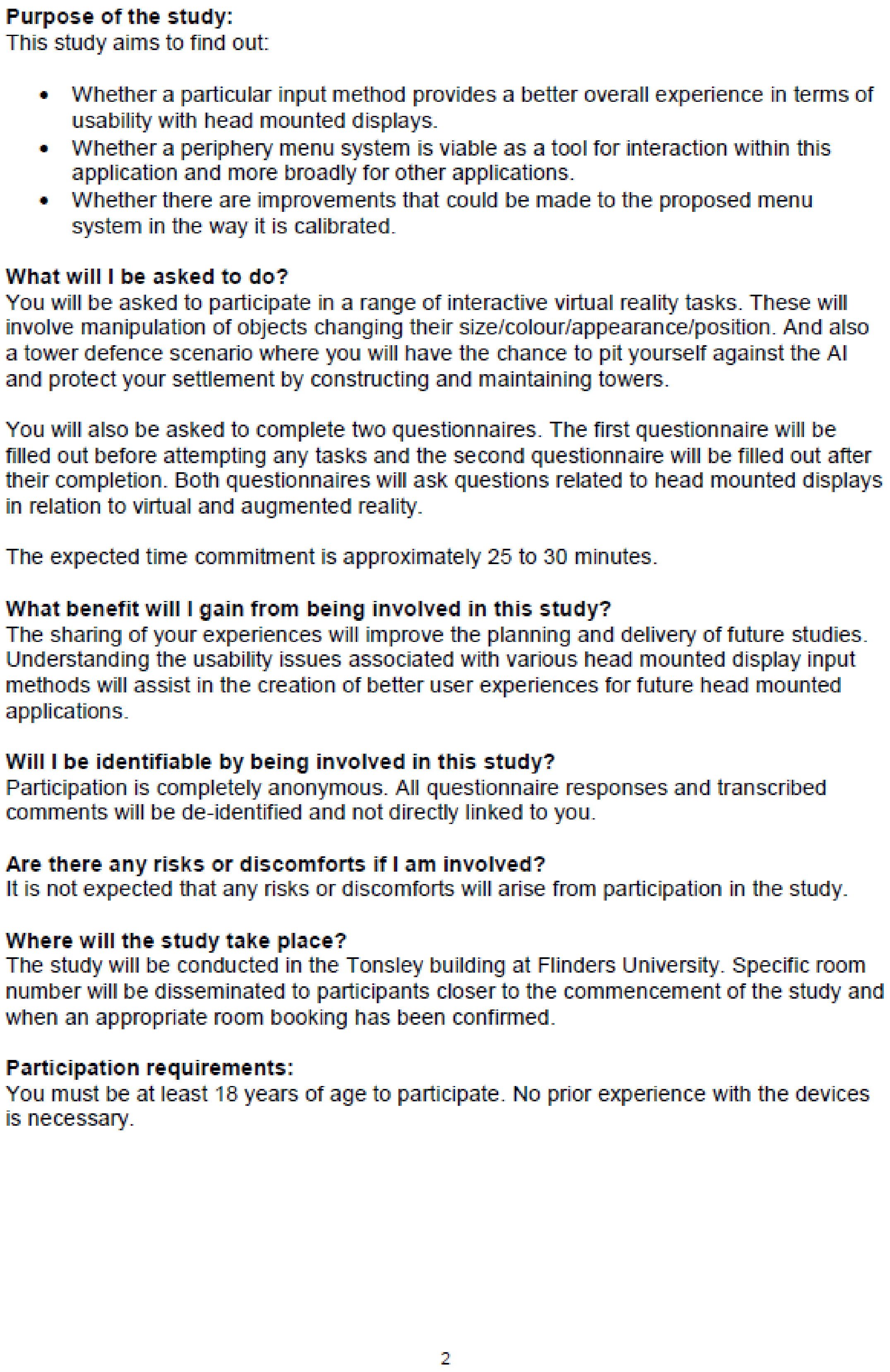
Investigators:
Mr Peter Mitchell
School of Computer Science, Engineering, and Mathematics
Flinders University
Email: peter.mitchell@flinders.edu.au

Supervisor(s):
Dr Brett Wilkinson
School of Computer Science, Engineering, and Mathematics
Flinders University
Email: [redacted]@flinders.edu.au
Ph: (08) [redacted]

Associate Professor Paul Calder
School of Computer Science, Engineering, and Mathematics
Flinders University
Email: [redacted]@flinders.edu.au
Ph: (08) [redacted]

Description of the study:
This study seeks to investigate the usability issues associated with interactive head mounted displays using virtual reality. Specifically focused on a menu interaction where the user will perform a gesture by turning their head in a direction to trigger a menu to be displayed. In this study a variety of user input methods will be used to test the proposed interaction method. To make testing this interaction experience interesting it has been applied to the context of a tower defence game.

inspiring achievement



Purpose of the study:
This study aims to find out:

- Whether a particular input method provides a better overall experience in terms of usability with head mounted displays.
- Whether a periphery menu system is viable as a tool for interaction within this application and more broadly for other applications.
- Whether there are improvements that could be made to the proposed menu system in the way it is calibrated.

What will I be asked to do?
You will be asked to participate in a range of interactive virtual reality tasks. These will involve manipulation of objects changing their size/colour/appearance/position. And also a tower defence scenario where you will have the chance to pit yourself against the AI and protect your settlement by constructing and maintaining towers.

You will also be asked to complete two questionnaires. The first questionnaire will be filled out before attempting any tasks and the second questionnaire will be filled out after their completion. Both questionnaires will ask questions related to head mounted displays in relation to virtual and augmented reality.

The expected time commitment is approximately 25 to 30 minutes.

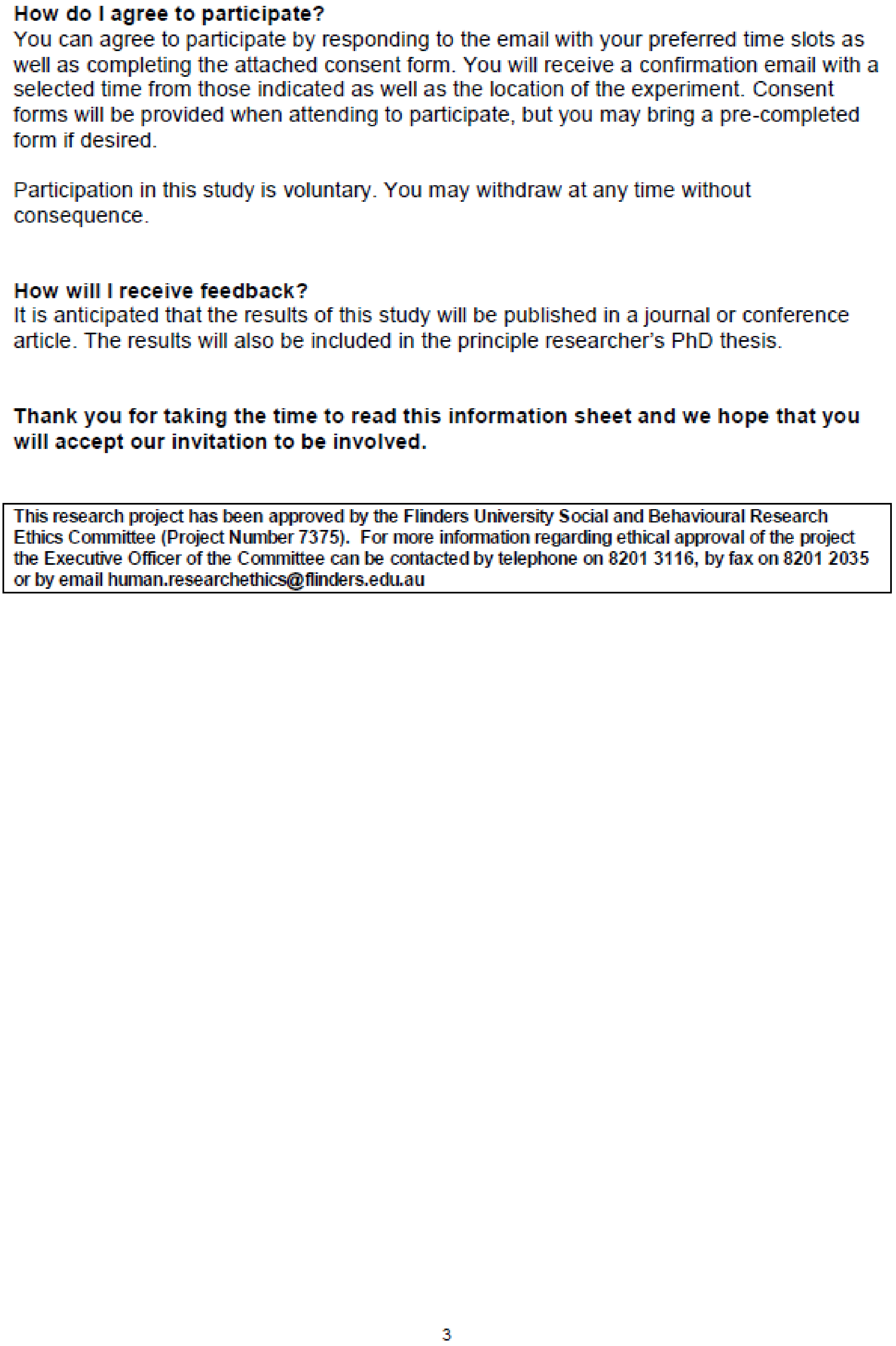
What benefit will I gain from being involved in this study?
The sharing of your experiences will improve the planning and delivery of future studies. Understanding the usability issues associated with various head mounted display input methods will assist in the creation of better user experiences for future head mounted applications.

Will I be identifiable by being involved in this study?
Participation is completely anonymous. All questionnaire responses and transcribed comments will be de-identified and not directly linked to you.

Are there any risks or discomforts if I am involved?
It is not expected that any risks or discomforts will arise from participation in the study.

Where will the study take place?
The study will be conducted in the Tonsley building at Flinders University. Specific room number will be disseminated to participants closer to the commencement of the study and when an appropriate room booking has been confirmed.

Participation requirements:
You must be at least 18 years of age to participate. No prior experience with the devices is necessary.



How do I agree to participate?
You can agree to participate by responding to the email with your preferred time slots as well as completing the attached consent form. You will receive a confirmation email with a selected time from those indicated as well as the location of the experiment. Consent forms will be provided when attending to participate, but you may bring a pre-completed form if desired.

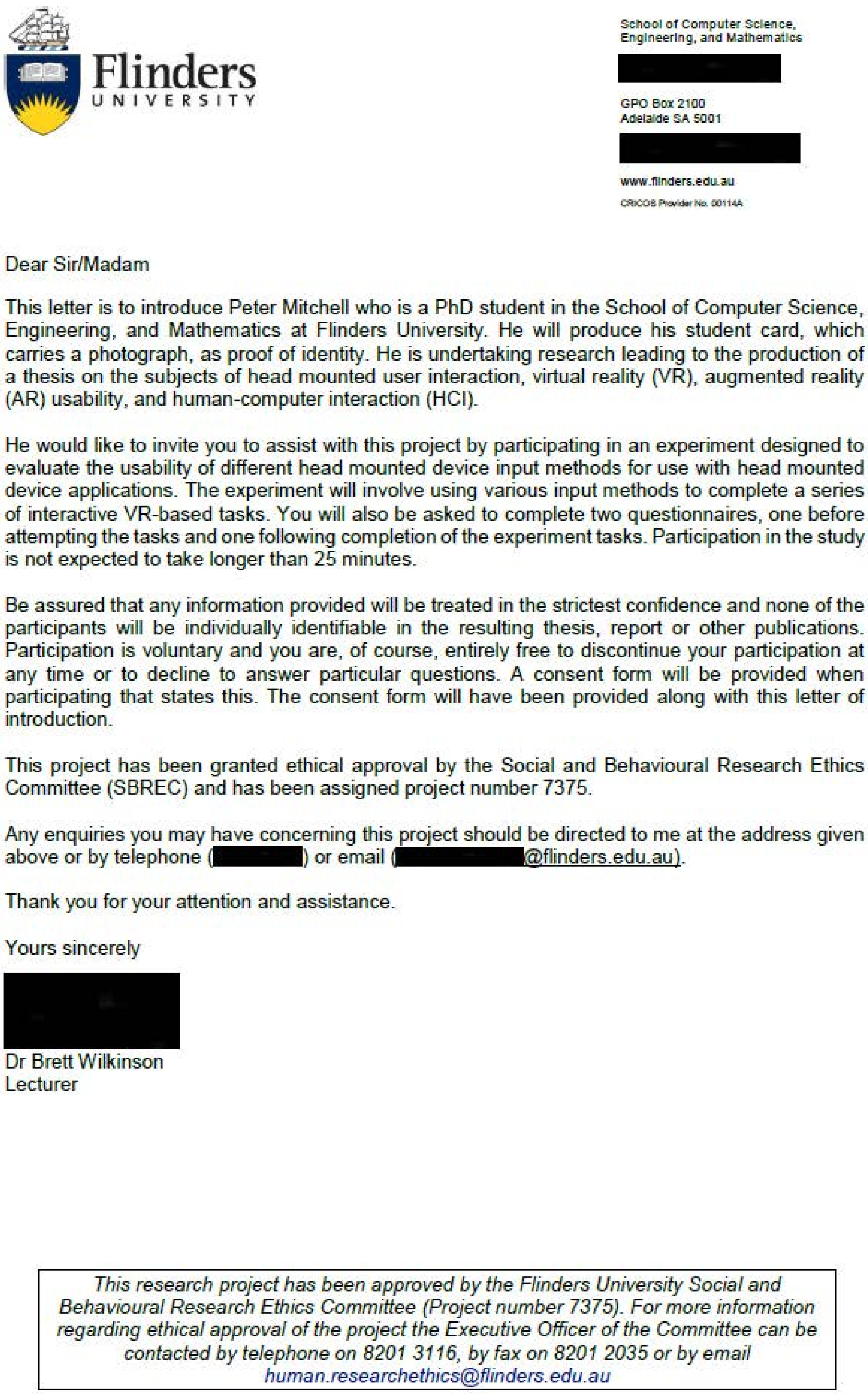
Participation in this study is voluntary. You may withdraw at any time without consequence.

How will I receive feedback?
It is anticipated that the results of this study will be published in a journal or conference article. The results will also be included in the principle researcher's PhD thesis.

Thank you for taking the time to read this information sheet and we hope that you will accept our invitation to be involved.

This research project has been approved by the Flinders University Social and Behavioural Research Ethics Committee (Project Number 7375). For more information regarding ethical approval of the project the Executive Officer of the Committee can be contacted by telephone on 8201 3116, by fax on 8201 2035 or by email human.researchethics@flinders.edu.au

C.2.7 Third Experiment Letter of Introduction



Flinders UNIVERSITY

School of Computer Science,
Engineering, and Mathematics
[redacted]

GPO Box 2100
Adelaide SA 5001
[redacted]

www.flinders.edu.au
CRICOS Provider No. 00114A

Dear Sir/Madam

This letter is to introduce Peter Mitchell who is a PhD student in the School of Computer Science, Engineering, and Mathematics at Flinders University. He will produce his student card, which carries a photograph, as proof of identity. He is undertaking research leading to the production of a thesis on the subjects of head mounted user interaction, virtual reality (VR), augmented reality (AR) usability, and human-computer interaction (HCI).

He would like to invite you to assist with this project by participating in an experiment designed to evaluate the usability of different head mounted device input methods for use with head mounted device applications. The experiment will involve using various input methods to complete a series of interactive VR-based tasks. You will also be asked to complete two questionnaires, one before attempting the tasks and one following completion of the experiment tasks. Participation in the study is not expected to take longer than 25 minutes.

Be assured that any information provided will be treated in the strictest confidence and none of the participants will be individually identifiable in the resulting thesis, report or other publications. Participation is voluntary and you are, of course, entirely free to discontinue your participation at any time or to decline to answer particular questions. A consent form will be provided when participating that states this. The consent form will have been provided along with this letter of introduction.

This project has been granted ethical approval by the Social and Behavioural Research Ethics Committee (SBREC) and has been assigned project number 7375.

Any enquiries you may have concerning this project should be directed to me at the address given above or by telephone ([redacted]) or email ([redacted]@flinders.edu.au).

Thank you for your attention and assistance.

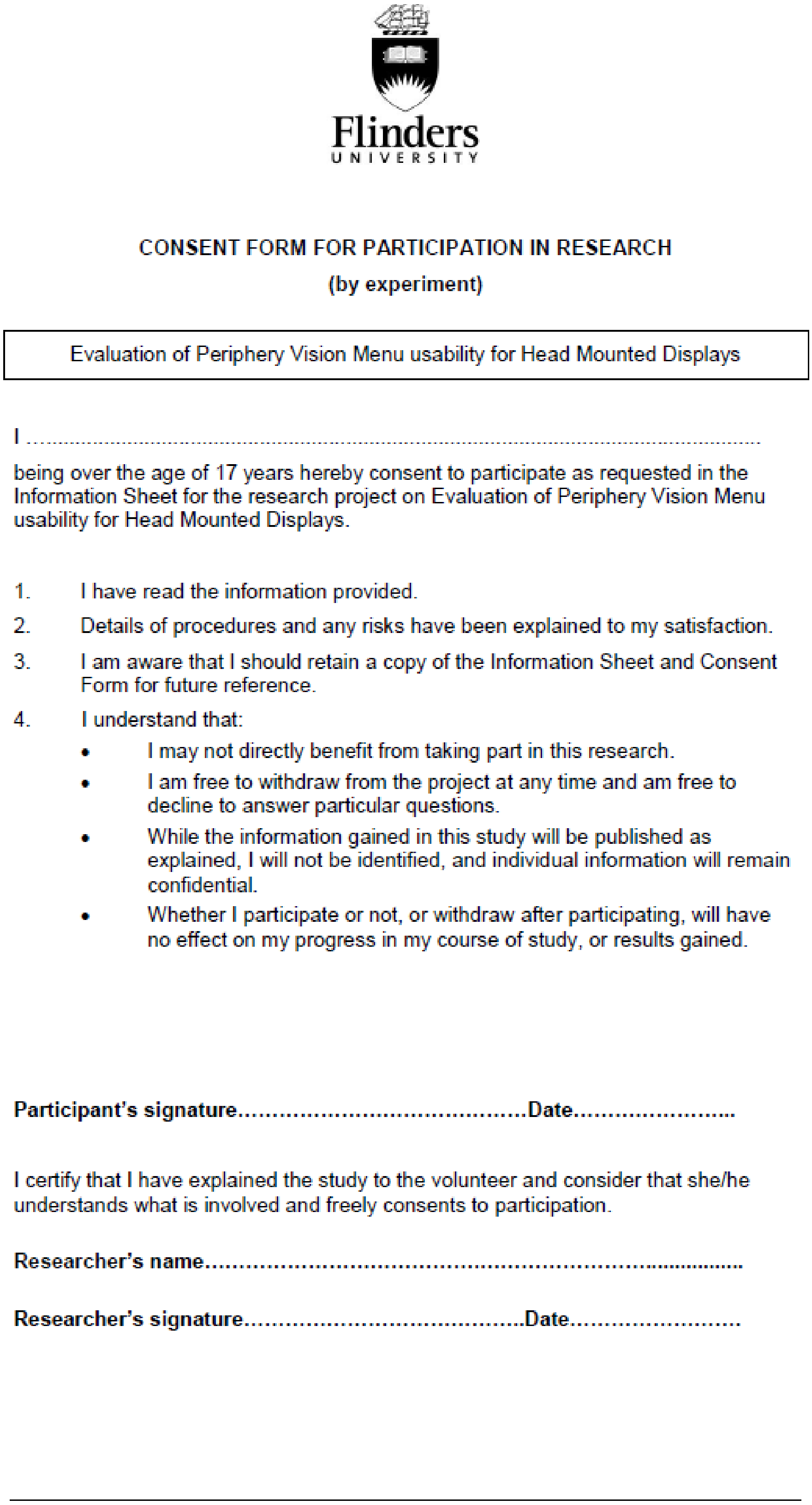
Yours sincerely

[redacted]

Dr Brett Wilkinson
Lecturer

This research project has been approved by the Flinders University Social and Behavioural Research Ethics Committee (Project number 7375). For more information regarding ethical approval of the project the Executive Officer of the Committee can be contacted by telephone on 8201 3116, by fax on 8201 2035 or by email human.researchethics@flinders.edu.au

C.2.8 Third Experiment Consent Form



Flinders UNIVERSITY

CONSENT FORM FOR PARTICIPATION IN RESEARCH

(by experiment)

Evaluation of Periphery Vision Menu usability for Head Mounted Displays

I ………………………………………………………………………………………………

being over the age of 17 years hereby consent to participate as requested in the Information Sheet for the research project on Evaluation of Periphery Vision Menu usability for Head Mounted Displays.

1. I have read the information provided.
2. Details of procedures and any risks have been explained to my satisfaction.
3. I am aware that I should retain a copy of the Information Sheet and Consent Form for future reference.
4. I understand that:
   - I may not directly benefit from taking part in this research.
   - I am free to withdraw from the project at any time and am free to decline to answer particular questions.
   - While the information gained in this study will be published as explained, I will not be identified, and individual information will remain confidential.
   - Whether I participate or not, or withdraw after participating, will have no effect on my progress in my course of study, or results gained.

Participant's signature……………………………………Date…………………..

I certify that I have explained the study to the volunteer and consider that she/he understands what is involved and freely consents to participation.

Researcher's name…………………………………………………………..............

Researcher's signature…………………………………..Date…………………….

C.2.9 Third Experiment Pre-Experiment Questionnaire

If you have not viewed the game trailer, please ask to view it before answering any questions.

1. Which of the following age ranges do you fall into?

Under 21	21 to 30	31 to 40	41 to 50	51 and Over

2. What is your gender

F	M

(F/M)?

3. Which of the following apply to your current situation?

Student	Other (please specify): ______________________

4. As a student, what is your area of study/research (please specify level of study as well, eg, 1st year or PhD, etc.)?

____

5. Have you previously participated in any of the following activities (mark all that apply)?

[ ] First Experiment of this Research (Conducted between May and June 2015)

[ ] Second Experiment of this Research (Conducted between March and May 2016)

[ ] Other VR or AR research (please specify):

____

[ ] Other HMD research (please specify):

____

6. How many hours a week would you use a computer on average?

Under 10	10 to 20	20 to 30	30 to 40	Over 40

7. How many hours a week would you spend playing video games on average?

Under 10	10 to 20	20 to 30	30 to 40	Over 40

8. If you play computer games, what are a few examples of games you play and the devices you play those games on?

____

____

9. On a scale of 1 to 7 (1 being not interested and 7 being very interested), how interested would you be in using a head mounted display **<u>for gaming</u>**?

Not Interested						Very Interested
1	2	3	4	5	6	7

10. On a scale of 1 to 7 (1 being not interested and 7 being very interested), how interested would you be in using a head mounted display **<u>for activities other than gaming</u>**?

Not Interested						Very Interested
1	2	3	4	5	6	7

11. On a scale of 1 to 7 (1 being never having played a tower defence game and 7 being very often playing) how often do you play tower defence type games?

Not Often						Very Often
1	2	3	4	5	6	7

12. Have you at any point used your head as an interaction tool? Eg, Microsoft Kinect interactive games, the Oculus Rift or other Head Mounted Displays etc. If Yes, Please specify any examples you have used in the past:

____

13. Select the option that applies most to you:

[ ] I am planning on buying a Head Mounted Display. Specify model/s:

____

[ ] I already own a Head Mounted Display. Specify model/s:

____

[ ] I am undecided and waiting to see more of VR/AR before making a decision.

[ ] I am not planning on buying a Head Mounted Display

Space for comment if desired:

____

14. On a scale of 1 to 10 (1 being no influence and 10 being high influence), how much did the game trailer influence your desire to participate in this research?

No Influence						High Influence
1	2	3	4	5	6	7

15. On a scale of 1 to 10 (1 being no influence and 10 being high influence), how do you feel visual presentations such as the game trailer would influence you to participate in future research?

No Influence						High Influence
1	2	3	4	5	6	7

C.2.10Third Experiment Post-Experiment Questionnaire

Please answer the following questions in relation to the tasks you have just completed.

The Periphery Vision Menu Interaction

1. On a scale of 1 to 7 (1 is not often, 7 is very often), how often did the periphery menu system display at the **wrong times** or when you didn’t mean to display it?

Not Often						Very Often
1	2	3	4	5	6	7

2. On a scale of 1 to 7 (1 is not often, 7 is very often), how often did the periphery menu system display at the **correct times**?

Not Often						Very Often
1	2	3	4	5	6	7

3. On a scale of 1 to 7 (1 is not useful, 7 is very useful), how useful do you feel the gesture of rotating your head to make a menu appear?

Not Useful						Very Useful
1	2	3	4	5	6	7

4. On a scale of 1 to 7 (1 is not likely, 7 is very likely), how likely would it be for you to want to use this interaction in the future?

Not Likely						Very Likely
1	2	3	4	5	6	7

5. What two aspects did you find most **enjoyable** about Periphery Vision Menu?

   a. ____

   b. ____

6. What two aspects did you find most **difficult** about using the Periphery Vision Menu?

a. ____

b. ____

7. Please circle the number that best describes your feelings toward each of the following in regard to the periphery vision menus.

a. I think that I would like to use this menu system frequently.

Strongly Disagree				Strongly Agree
1	2	3	4	5

b. I found the menu unnecessarily complex.

Strongly Disagree				Strongly Agree
1	2	3	4	5

c. I thought the menu was easy to use.

Strongly Disagree				Strongly Agree
1	2	3	4	5

d. I thought that I would need the support of a technical person to be able to use this menu.

Strongly Disagree				Strongly Agree
1	2	3	4	5

e. I felt that the options presented by the menu were well integrated.

Strongly Disagree				Strongly Agree
1	2	3	4	5

f. I felt that there was too much inconsistency with the menu.

Strongly Disagree				Strongly Agree
1	2	3	4	5

g. I would imagine that most people would learn how to use these menus very quickly.

Strongly Disagree				Strongly Agree
1	2	3	4	5

h. I found the menus very cumbersome to use.

Strongly Disagree				Strongly Agree
1	2	3	4	5

i. I felt very confident using the menus.

Strongly Disagree				Strongly Agree
1	2	3	4	5

j. I needed to learn a lot of things before I could use the menus.

Strongly Disagree				Strongly Agree
1	2	3	4	5

8. Please rank the following from 1 to 7 (1 is most desired, 7 is least desired). Of the following what would you desire to use the menu system for most?

a. ___ As a method of interacting with **games**.
b. ___ As controls for **viewing a movie or TV shows** in a virtual cinema.
c. ___ As a tool for selecting components for **constructing models or building maps.** (Eg, like placing objects as in the tasks completed during this experiment)
d. ___ As a method for accessing **instant messenger services or voice chat** at any time.
e. ___ As a tool for navigating common **operating system controls and menus**.
f. ___ As a method of interaction with a **virtual tour guide**. (this could be a fully virtual location such as going for a tour through a virtual replica of a city or even as an extension though augmented reality of a museum)
g. ___ As a method of **browsing the internet**.

Any other situations you feel you would use it in:

____

9. When considering the usability of a user interface for head mounted displays with augmented reality or virtual reality what features do you believe are the most important? (Rank features from 1 to 4, where 1 is most important and 4 is least important).

a. ___ The speed of accessing features.
b. ___ The accuracy of accessing features.
c. ___ The simplicity of physical interaction required.
d. ___ The visual appeal of the interface.

Any other features you feel are important:

____

10. On a scale of 1 to 7 (1 is not useful, 7 is very useful), how useful did you find the **circular menu fixed to an object** as compared to completing actions with the periphery menu?

Not Useful						Very Useful
1	2	3	4	5	6	7

11. On a scale of 1 to 7 (1 is not useful, 7 is very useful), how useful did you find the **periphery menu** as compared to completing actions with the circular menu?

Not Useful						Very Useful
1	2	3	4	5	6	7

12. On a scale of 1 to 7 (1 is worse experience, 7 better experience), how did you find the experience of interacting with the **addition of the Xbox controller as a selection tool,** as compared to using the head alone as a selection tool?

Worse Experience			Similar			Better Experience
1	2	3	4	5	6	7

13. Do you have any other thoughts that might be useful in regard to anything you experienced during this experiment? Is there anything you would change about the periphery menu system in respect to the way the system is interacted with? Eg, either anything to do with the way the gesture works, or the hardware being used.

____

Thank you for taking the time to participate in this experiment.

D Appendix D: How to Access GitHub Code for All Experiments

The code for all three experiments is available on GitHub. The following briefly describes the content included as part of each repository.

First Experiment

GitHub URL: https://github.com/Squirrelbear/PhD-First-Experiment

The repository for the first experiment includes:

- First Experiment folder: Contains the Unity project with all assets and is designed to run with Unity version 5.0.1f1. You will need an Oculus Dev Kit 1 and the associated drivers to run the application in VR mode with the provided project. As part of the folder, there are pre-built versions of both the experiment executable and the replay executable.
- TouchNetworkApp folder: Contains a Java Android application used for the mobile input that functions by taking control of the screen, sending messages over the network, and taking input from screen taps.
- ExtraDataGatherer folder: Contains a C# Visual Studio project with code used to generate data by iterating over every frame of the replay data file and generating a log of all events that occurred.

Second Experiment

GitHub URL: https://github.com/Squirrelbear/SecondExperimentPhD

The repository for the second experiment contains only the Unity project, with all assets used for the experiment included. The Unity version of the project is 5.3.2f1 and may be required to open in its original form. The project includes all the assets used to generate the trailer, view replays and run the original version of the application used for experimentation.

Third Experiment

GitHub URL: https://github.com/Squirrelbear/ThirdExperimentPhD

The repository for the third experiment contains all the same setup as the second experiment. It includes a project built with Unity version 5.3.2f1 with assets used to generate the trailer, view replays and run the original experiment.