8-11-2023 1:40 PM

A Study on the Field of XR Simulation Creation, Leveraging Game Engines to Develop a VR Hospital Framework

Virat K. Tripathi, *Western University*

Supervisor: Eagleson, Roy, *The University of Western Ontario*
Co-Supervisor: De Ribaupierre, Sandrine, *The University of Western Ontario*
A thesis submitted in partial fulfillment of the requirements for the Master of Engineering Science degree in Electrical and Computer Engineering

Abstract

This thesis introduces an adaptable and extensible VR framework designed for clinicians and patients using pre-existing game development software like Blender and Unreal Engine. The framework aids patients in familiarizing themselves with hospital scenarios and environments, reducing anxiety, and improving navigation. Clinicians can use the tool to educate patients and collaboratively design new aspects of the environment. A prototype implementation demonstrates the system's effectiveness, with usability studies indicating that teleport movement is preferred over sliding for locomotion and that navigation speed can improve with subsequent trials in the VR simulator. The framework's potential for enhancing patient experience and facilitating informed consent is also discussed. The research findings provide valuable insights for future VR healthcare applications while affirming the valuable future applications of the hospital framework and development workflow.

Keywords

Virtual Reality (VR), Augmented Reality (AR), Mixed Reality (MR), Extended Reality (XR), Game Engine, Unreal Engine, Blender, Healthcare, Digital Twin, Hand Tracking, Simulation

Summary for Lay Audience

This research is centered on creating an immersive and interactive Virtual Reality (VR) framework for a hospital environment designed to benefit both healthcare professionals and patients. The focus is on making this hospital as realistic and functional as possible, using the VR tool to replicate the complexities of a real-world hospital setting. For patients and their families, this VR hospital aims to familiarize them with the hospital's layout and facilities while offering patients a way to experience simulations of potential medical procedures. This helps to reduce anxiety and provides them with knowledge about the hospital environment, improving their ability to navigate within it. The unfamiliarity of hospital settings can be stressful, so this virtual tool serves as a practice run, making real visits less daunting. For clinicians, the VR tool offers a unique platform for patient education and facility design.

Healthcare professionals can use the tool to explain medical procedures, teach patients about the hospital environment, and even design new aspects of the hospital in a modular fashion. This tool makes clinicians part of the design team, empowering them to create new layouts and specialized rooms tailored to their needs. The VR framework was tested through a prototype demonstration, inviting participants to explore the tool and provide feedback. These insights were invaluable in identifying areas for improvement and assessing the tool's effectiveness. The goal is to make the system user-centric, prioritizing comfort, intuitive navigation, and interactivity. This innovative VR framework holds significant potential for enhancing patient experience and facilitating informed consent. By easing navigation difficulties, reducing patient anxiety, and providing an immersive educational platform for clinicians, it may reshape interactions within healthcare environments. Beyond immediate hospital applications, the insights and methods from this research could have broad implications for future VR developments in various sectors.

Acknowledgments

It is only through the guidance and support of many family members, colleagues, friends, and mentors that enabled the construction of this thesis, all of whom could not be listed here. However, the author would like to express his deepest gratitude and appreciation to several people that influenced this work and helped support them during the time of research and writing.

Dr. Roy Eagleson and Dr. Sandrine de Ribaupierre for their guidance, vision, encouragement and advice through the program development and thesis writing process. Their recommendations and ideas for test scenarios were crucial during this Masters.

My colleagues in the Brain3dviz lab group who helped in creating portions of the environments in Blender and assisted in layout capture at Victoria Hospital when I was not allowed to enter due to COVID – 19 guest restrictions.

On a more personal note, I'd like to thank my parents Dr. Chandrashekhar Tripathi and Abhilasha Tripathi along with my siblings Varsha Tripathi and Varun Tripathi for their selfless support, inspiration, and encouragement. Special thanks to my brother Varun for the hours he put into helping with editing the language used in this thesis and ensuring the content correctly conveyed the thoughts in my head. I would also like to thank my girlfriend Aarushi Vasudev and her family for the support and encouragement they gave throughout my time spent developing and writing in London, Ontario.

Without the unyielding support of all these people and others not mentioned here, this thesis would not have been possible.

Table of Contents

List of Tables

List of Figures

List of Appendices

Chapter 1

1 « Introduction »

1.1 « Background and Technological Motivation »

VR refers to an immersive technology that simulates a three-dimensional environment, allowing users to interact with and manipulate objects in the virtual world (Khor et al., 2016). Though earlier versions of the technology have been available for over three decades, only recently has VR experienced a surge in adoption across various industries, some of which include architecture, manufacturing and gaming (Ferche et al., 2015). However, despite its proven usefulness in a few sectors, its adoption has been relatively slow in others.

An industry which could benefit greatly from its integration of VR technology but has been slow to adopt, is healthcare. Hospitals can be difficult to navigate, causing anxiety for younger patients, the elderly and those with mental health conditions (Jiang & Verderber, 2017). VR can not only help lower anxiety for these patient groups by digitally simulating the experience ahead of time, but its capacity for 3D representation means VR is a great tool for clinicians to use in educating all patients on everything from surgeries to pregnancies, reducing anxiety and also ensuring informed consent. Additionally, they can use the tool to author new aspects of the environment, making the clinicians part of the design team and collaboratively configuring new layouts and specialized rooms. These advantages can be extended to other groups as well. Hospital medical and administrative staff can use these tools for training, on-boarding, orientation and even remote collaboration. The importance of remote work was illustrated by the recent COVID-19 pandemic, as the need for an alternative method of joint work and training without the risk of disease transmission became urgent (Banerjee, 2023).

Despite the potential benefits, there are a number of barriers which have historically contributed to the slow adoption of VR in the healthcare sector. These include technical limitations of the devices, significant development costs, and lack of standardization across devices and platforms (Eagleson et al., 2014). Another major challenge has always been the unnatural implementation of human-computer interfaces (HCI), which result in motion sickness and a diminished sense of immersion (Ferche et al., 2015).

However, a number of recent advances in the space can help overcome these barriers. For example, availability of low-cost headsets, and free development tools like Blender and Unreal Engine can be used to greatly lower development overhead (Staples et al., 2021). Additionally, there have been major improvements in Human-computer interfaces as a result of novel interaction techniques which focus on natural, hand-based gestures and motions (Talbot & Chinara, 2022).

1.2 « Research Objectives »

The main goal of this thesis is to design and analyze a VR hospital framework of the London Health Sciences Centre (LHSC) Victoria Campus that caters to two primary user groups: clinicians and patients (along with their families), by using free tools like Blender and Unreal Engine in a development workflow. This framework aims to address the need for a low-cost workflow for creating natural, educational, and interactive VR hospital environments that individuals or small teams can easily develop, adopt, and modify while reducing the risk of motion sickness.

In addition to the primary objective, supporting goals include:

- Explore the potential of the framework to improve ease of navigation, alleviate patient anxiety, and provide a platform for remote professional training at the LHSC Victoria Hospital.
- Ensure the framework is scalable, modular, and easily deployable, enabling both individuals with limited technological training and clinicians to actively participate in the design and authoring process. This will enable them to create new aspects of the environment, collaboratively configure new layouts, and design specialized rooms tailored to their needs.

To evaluate the success of these objectives, the framework's performance will be assessed through an objective-driven, proof-of-concept demo. This demo will utilize ready-to-use elements from the framework, focusing on intuitive system design elements such as navigation assistance, interactivity, and clinical scenario recreation. In order to be considered a success, the developed demo must:

- Faithfully represent the layout of the Victoria Hospital.

- Demonstrate the framework's built-in functionality that can be easily adapted for trial scenarios.
- Demonstrate the framework's built-in data capture capabilities, and the ability to estimate user performance in navigation and manipulation tasks.
- Exhibit easy expandability for additional environments and digital assets.
- Prove user-friendly for patients or trial subjects through natural, user-centric design.
- Provide a comfortable experience, with special attention given to minimizing motion sickness.

To gauge these aspects, the demo will leverage built-in data collection functions to gather user performance metrics on gesture learning, movement methods, and navigation skills. A post-demo survey will collect feedback on user experience and the effectiveness of the simulation. This data will be used to evaluate, iterate, and optimize the framework, guiding future use and expansion.

1.3 « Thesis Structure »

This thesis paper provides a comprehensive overview of both the workflow and the options available for an individual or small team as they move from concept to creation of a VR hospital framework. This is done in the context of a proof-of-concept demo. The research includes technical decisions which were made to ensure its modularity as a development tool. Built-in interaction modalities allow for easy adaptation, while modularity allows experienced developers to customize the system. The thesis includes not only a feasibility study based on the demo but also ideas for iterative improvement and a next generation method for future work. The three terms above: The thesis paper, the framework, and the demo refer to 3 different, but overlapping components of the study which are illustrated along with the included components in Appendix A.

The study will serve as a guide for those considering XR simulation development, along with a snapshot of the 2021-2022 technological landscape and potential application of technologies set to arrive in 2023. Given the major advancements in artificial intelligence since 2021 this research also includes the possibilities it presents towards improving workflow efficiency. In summary, this thesis captures the development cycle, resources, and distribution mediums available, as well as the proposed next-generation VR development cycle which further streamlines environment creation.

Chapter 1: Introduction: In this chapter, we introduce the background of the problem, the motivation for the study, the research objectives, and the structure of the thesis. We also provide an overview of the VR hospital framework, its development, and the demo created for testing.

Chapter 2: Literature Review: This chapter presents a comprehensive review of the relevant literature. We discuss the current state of VR in healthcare and other industries, the different types of XR including VR, and various research into the applications of VR. We also delve into the use of game engines in the field along with challenges and limitations of VR accounting for its slow adoption, including the issue of motion sickness.

Chapter 3: Design Methodology: In this chapter, we detail the development approach, beginning with the selection rationale for simulation genre, environment, medium of delivery, head-mounted display (HMD), interaction modalities, and software tools like game engines. In later portions of the chapter, we discuss the development timeline for design of the VR hospital framework. This includes the implementation of hand tracking, locomotion methods, construction of the environments, and steps to address motion sickness in VR.

Chapter 4: Experiment Methodology: This chapter outlines the methodology used to test the VR hospital framework. It discusses the creation of the feedback survey and proof-of-concept demo, which includes user evaluation and data collection methods used in the study, along with the formatting of the collected data for analysis and testing demographic selection.

Chapter 5: Results and Discussion: This chapter presents the findings from demo testing and the post-demo survey. It discusses the implications of these objective and subjective results for the VR hospital framework and the field of VR in healthcare more broadly.

The chapter also delves into the challenges and limitations encountered during the study and highlights the opportunities these present for future research and iterations of the VR framework. The chapter concludes by discussing the prospects and impact of the study, emphasizing the value of user feedback as part of the iterative design process, the economic impact of the study, and the study's contributions to the field of VR in healthcare.

Chapter 6: Conclusions and Future Work: This chapter combines the analysis of research outcomes with a look towards the future. It summarizes the key findings, contributions, and insights gained from the study, highlighting the research's significance in healthcare and within the broader context of VR simulation development. In addition, the chapter reflects on the extent to which research and testing objectives have been met, drawing connections between the various chapters and the overall narrative of the thesis. Furthermore, this section discusses how emerging technologies could lead to higher fidelity environments, streamlined development, and easier expansion. Possible enhancements, such as incorporating new interaction techniques, leveraging artificial intelligence (AI), and expanding the system's scope, are also addressed, offering a roadmap for ongoing and future research efforts.

Chapter 2

2 « Literature Review (VR Application and Development Landscape »

This chapter presents a comprehensive review of the literature surrounding the application and development landscape of Virtual Reality (VR). It begins by exploring the various applications of VR in different industries, with a particular focus on healthcare. The review highlights the potential of VR in staff training, patient education, rehabilitation, pain management, and mental health treatment. It also discusses the use of modern game engines in the development of realistic VR environments and the challenges associated with their use. The chapter concludes by addressing the significant issue of motion sickness in VR, its causes, effects, and potential mitigation strategies.

Before delving into the real-world applications of VR, it is important to understand the forms of immersive reality that lay the foundation for these interactions. Extended reality (XR) is an umbrella term that includes virtual reality (VR), augmented reality (AR), and mixed reality (MR) along with other immersive technologies, each offering unique capabilities for enhancing human experiences beyond the physical world.

VR creates entirely digital environments, offering enhanced immersion and engagement, but requires high-performance hardware and can potentially isolate users from the real world. AR overlays digital content on the user's actual environment, enhancing interactivity and accessibility, but requires precise tracking and risks information overload. MR combines the physical and digital worlds in real-time, allowing users to interact with both simultaneously.

For the Victoria Hospital Simulation, VR was chosen for its immersive capabilities. It allows users to explore and interact with the digital representation of the hospital without real-world distractions, making it ideal for educational purposes and practicing medical procedures in a controlled environment. These 3 mediums of simulation delivery and the rationale behind selecting VR are further elaborated upon in section 3.3 of this thesis.

2.1 « VR applications in other industries »

Virtual reality (VR) and its related forms such as augmented reality (AR) are increasingly being utilized in various industries, including manufacturing, construction, retail, education, urban planning, and graphic design. The applications of this technology extend beyond the scope of this study, yet they provide valuable insights into the potential benefits and effectiveness of VR in different contexts.

In the retail industry, for instance, Walmart has been using VR for employee training. While this is not an academic study and thus cannot be officially referenced, it provides a practical example of VR's potential. Walmart's VR training modules cover a range of scenarios that employees might encounter, from the everyday to the extraordinary. This includes managing the holiday rush, dealing with difficult customers, and responding to emergencies. The immersive nature of VR allows employees to learn by doing, which can lead to better retention and understanding.

In the realm of academic research, several studies have explored the use of VR for training, education, and orientation purposes. For instance, a study by Noble et al., (2022) investigated the use of VR for learning in a discretionary context, where students could choose between VR and video for learning a novel task (suturing). The study found that students' acceptance of VR was mainly driven by their performance expectancy, or the belief that VR would help them achieve their learning goals. The study also examined the role of effort expectancy and social influence as predictors of VR acceptance. The paper contributes to the literature on VR acceptance by using relative weight analysis to compare the importance of these factors and by measuring acceptance in two ways: behavioral intention and preference.

In addition to education, VR has also been used for personnel assessment and selection in various industries. For example, a study by Simons et al., (2023) investigated the use of VR for measuring intelligence and cognitive abilities, using the commercial VR game Job Simulator and the intelligence test BIS-4. The study found that participants who completed the VR game faster demonstrated higher levels of general intelligence and processing capacity than those who completed it slower. The study also examined the role of other intellectual abilities, such as memory, verbal ability, and figural ability, in playing the VR game. The paper contributes to the literature on VR assessment by using a casual simulation game that is intuitive and attractive to different groups of applicants and by comparing the importance of different predictors of VR acceptance.

Another domain where VR has been applied is disaster preparedness and response training. A study by Hsu et al., (2013) reviewed the state of the art of VR-based disaster training and identified the challenges and opportunities for future research and practice. The study found that VR can provide realistic, immersive, and safe environments for training various skills and competencies related to disaster management, such as situational awareness, decision making, communication, and teamwork. The study also discussed the technical, ethical, and pedagogical issues that need to be addressed for developing effective and engaging VR-based disaster training. The paper contributes to the literature on VR training by providing a comprehensive overview of the current trends, gaps, and directions for VR-based disaster training.

In similar ways to the studies mentioned in this subsection, the developed framework can be tailored to custom scenarios and environments. The default environment of Victoria Hospital may not be of use but thanks to the ease of expandability, future developers can use the Unreal Engine hospital levels as an outline for how to structure their own environments while still using the built in mechanics and data gathering functions.

2.2 « VR usage in healthcare »

In addition to the previously mentioned industries, Virtual reality and Augmented reality technologies have also gained attention in the healthcare sector, showing potential for major breakthroughs in staff training, patient education, rehabilitation, pain management, and mental health treatment.

Some primary avenues where VR has been leveraged are medical training, education, and telesurgery. Specific applications of VR simulation span surgery, emergency scenario training, staff onboarding, and orientation. The immersive nature of VR allows for risk-free, cost-effective, and repeatable training sessions. For instance, de Ribaupierre &

Eagleson, (2017) discuss the use of ventricular drain insertion trainers, procedure training simulators, and the Dextroscope. The Dextroscope is a VR training and planning platform for which Research by Kockro et al., (2016) concluded that planning in the VR environment improved the spatial understanding of the vascular anatomy and resulted in excellent clinical outcomes. This illustrates the effectiveness of VR in enhancing surgical training. The thesis further extends these principles, exploring ways to increase the efficacy of training methods. The training methods for complex procedures not only help practitioners perform better through low-risk repetition but also come at almost one tenth of the cost of an immersive physical simulation (Pottle, 2019).

Beyond just simulating surgeries, VR is used to actually perform Minimal Invasive Surgery (MIS) as it requires a surgeon looking at a monitor which can be enhanced with the immersion that VR enables (Aziz, 2018). Although performing surgeries falls outside the scope of the framework developed in this study due to proprietary robot connections, the various uses listed above illustrate the effectiveness of VR in enhancing surgery and surgical training/planning. The thesis further extends these principles, exploring ways to increase the use of VR training methods through the development of an adaptable, low-cost framework on which the simulator can be built.

Some other forms of medical training that can be improved through their integration with VR and the use of the thesis framework are employee orientation and patient assessment training. Studies have shown that nursing students are able to improve their clinical reasoning skills and confidence in assessing patient healthcare needs within home settings by practicing through VR scenario trials (Yoshioka-Maeda et al., 2022). Additionally, for employee orientation, one study explores replacing hospital tours in residency interviews with a virtual experience (Zertuche et al., 2020). The study showed that the applicants preferred VR simulations over in person tours due to "interview fatigue" from physically touring every hospital during the matching process. The study found that Rutgers Med-Peds department saved 2345 USD on transportation in a round of interviews by making this switch to VR. With the developed adaptable framework, rather than conduct repetitive on-site tours of every hospital taking up time and resources, each hospital could make a scripted tour simulation once and deliver the software to each resident, making it a one-time cost. And on the resident's end, they can freely tour and explore every hospital they are matched with without having to travel in person.

In the context of patient care, interactive VR simulations can provide patients with a comprehensive understanding of their medical procedures or conditions (van der Kruk et al., 2022), reducing associated anxiety and improving their overall healthcare experience. In a study by Pandrangi et al., (2019), patients who were shown a three-dimensional model of an abdominal aortic aneurysm (AAA) felt better informed and more engaged using VR than when a description was verbally relayed. In addition, another study by Eagleson et al., (2014) explored serious games for patient education, further expanding the utility of serious games in healthcare, they can be classified into three categories: Distraction, Exergaming, and Education. Distraction-oriented games aim to alleviate anxiety and discomfort during painful treatments. Exergaming promotes physical activity for fitness and rehabilitation, especially beneficial for patients recovering from trauma or stroke. Educational games focus on teaching patients or their families about upcoming procedures or disease management effectively, fostering better understanding and self-care. Through these categories, serious games revolutionize patient education and engagement, promoting better treatment adherence and health outcomes.

This familiarization and relief in anxiety has also been shown in studies involving children. A randomized clinical trial by Stunden et al., (2021) compared three methods for familiarizing patients with the pediatric MRI procedure and found both a VR-based simulation app (VR-MRI) and physical hospital-based Child Life Program (CLP) to be more effective in reducing anxiety than the standard preparatory manual. However, the VR experience developed and investigated by the researchers used a passive 360 video with overlaid graphics. Although this 3DOF approach is more engaging than traditional preparatory methods, the stationary nature of the experience limits immersion as it prevents patients from freely moving around and interacting with the environment. The framework developed in this thesis addresses this limitation by facilitating self-directed exploration with the potential for direct environmental interaction. The fully immersive, 6DOF and game-like interactive experience is particularly appealing to the pediatric population, making it even more effective.

Moreover, the use of VR extends to the field of rehabilitation and pain management. VR-based rehabilitation programs have been developed for patients recovering from various conditions, including strokes and cognitive injuries, offering them engaging, personalized therapy sessions, with incorporated gamification (Kim et al., 2020; Peláez-Vélez et al., 2023). The developed framework takes full advantage of the Oculus Quest 2 and its native hand tracking capabilities, these hand tracking functions can be leveraged within the unified platform to further develop applications for this rehabilitation therapy. Hand tracking would be very good for hand and finger rehabilitation while an outside-in system or the addition of body trackers could be used to perform arm rehabilitation. Programs built on the framework would be able to draw functionality and assets from one another or be expanded upon by other practitioners or researchers. Additionally, as mentioned earlier, VR has been found effective in managing pain, where the immersive, distracting nature of VR experiences can help patients cope with acute or chronic pain. A comprehensive literature review by Indovina et al., (2018) summarized clinical trials using VR distraction during different medical procedures, such as burn injury treatments, chemotherapy, surgery, dental treatment, and other diagnostic and therapeutic procedures. The review showed that VR distraction can reduce procedural pain and distress, as well as cancer-related symptoms, with minimal side effects. These distraction techniques can take the form of any immersive VR experience. The framework is designed to be easily expanded and can easily allow clinicians to form distractive simulations for pain management based on the position of the patient during the treatment as done in the example MRI in the feasibility demo.

In the realm of mental health, VR has been leveraged for exposure therapy in treating conditions such as phobias and post-traumatic stress disorder. According to Boeldt et al., (2019), "the controlled environments provided by VR allow for gradual and safe exposure to fear-inducing situations, which can be therapeutic" (p. 2). However, they also acknowledge that further research and collaboration is needed to advance the development and dissemination of VR applications for mental health. As a targeted example, a study by Barnett et al., (2022) investigated the potential use of VR for agoraphobia exposure therapy by conducting a survey with patients diagnosed with agoraphobia. They found that VR could help expose the patients to different environments while being in a safe space and assist them in learning coping mechanisms for anxiety and panic attacks. They also identified key areas of focus for VR scenarios, such as transport, crowded areas, and work environments which can all be modelled according to the current or future workflows discussed in the thesis and added for access from the VR hospital to provide gradual exposure. The study suggested that VR could be a beneficial aid to assist in the treatment of agoraphobia alongside other medical treatments.

Yet, despite these promising applications and their demonstrated benefits, the adoption of VR in healthcare faces several challenges. According to Ferche et al. (2015), these include cost-related issues, such as the high price of VR equipment and software development, which may limit the availability and affordability of VR solutions for both providers and patients. Another challenge is the issues generated by the human-computer interfaces that VR uses, which may seem unnatural or uncomfortable for first-time users, or may require a longer time to accommodate to. These factors may affect the performance, reliability, and safety of VR applications in healthcare settings. Moreover, the adoption of VR in healthcare may encounter immersion issues that may influence the quality of the rehabilitation process, such as the transfer of performance from the virtual to the real world, the maintenance of motivation and attention, and the possible side effects or aftereffects of VR exposure such as motion sickness. These issues may pose barriers to the integration of VR into clinical practice and research. Therefore, these hurdles need to be addressed for VR to become a mainstream tool in healthcare. The framework developed in the thesis aims to address each of the issues presented in this section, using pre-existing development tools made for video games is one of the ways the framework brings down development costs. In addition to the pre-existing development tools, the framework itself gives developers and clinicians a unified starting point, reducing the amount of effort as well as the development time poured into building each new program from scratch.

2.3 « Use of Game Engines for Simulation Development »

Modern game engines have evolved to meet the growing demand from developers to include animation, quality graphics, artificial intelligence, physics and 3D rendering capabilities (Vohera et al., 2021). This capacity to use a single program to develop realistic game environments can be leveraged to create VR based systems. These strengths and the capacity for broad platform deployment led to the choice of using a game engine to develop and modify the VR hospital framework developed in this thesis. A number of game engines currently exist with these built in functions and have been studied for their application in developing non-game software. One such engine - Evolution engine – was evaluated by Carnegie, (2015) for its potential use in the development of their Endoscopic Third Ventriculostomy simulator. In the end, given that it is optimized for the creation of first-person shooter games and would be difficult to adapt for the purpose of their study, a decision was made against its use.

Unreal Engine (UE) and Unity engine are much more open and are not limited by the narrow scope of engines optimized for single genre games. In fact, these game engines have recently seen an increase in usage for VR simulation development. For instance, a study by Zikas et al., (2020) outlines Unity's capabilities for VR development focusing on developing an easier method of coding which mimics the native visual scripting of UE to then create the C# code which Unity runs on. While Unity's features are robust, the UE has native visual scripting and, as detailed by Vohera et al., (2021), offers advantages such as built in VR support and high graphical fidelity with powerful built in lighting and texture tools. The study acknowledges UE's steep learning curve due to its reliance on visual scripting but also states that, compared to learning C# from scratch, this proprietary system is easier for beginners entering the field. Since the VR hospital framework was designed to be adapted and extended by researchers or medical staff who may not have prior coding experience, the visual scripting blueprint system's ability to easily route new functionality and visualize the code were found to align more closely with the objectives of this thesis.

The studies mentioned above eventually informed the decision to select UE for the thesis framework. By leveraging Unreal Engine's robust capabilities and strategically addressing the challenges associated with game engine use, the developed framework aims to create immersive, effective VR simulations for healthcare applications. Therefore, this literature review not only serves as a survey of the existing landscape of game engine use in VR simulation development but also provides direct evidence supporting the choices and strategies implemented in this thesis.

2.4 « Motion Sickness in VR »

Immersive VR has shown impressive capabilities for gaming, training, and education but one of the biggest drawbacks has always been the potential to induce motion sickness in users. Sometimes referred to as cybersickness (McCauley & Sharkey, 1992), motion sickness in VR is a discomfort in the form of nausea, eye fatigue, dizziness, and/or disorientation that is experienced by some users when using VR (Chang et al., 2020; Grassini et al., 2021). As the focus of this thesis is developing a framework for use in healthcare settings where potential users may be suffering from pre-existing health conditions, mitigating motion sickness during regular use is important. Regular use refers to all simulation situations where the user has the ability to select their locomotion method. This includes all experiences in the demo aside from trial 1 where users were directed to use both teleport locomotion and slide locomotion for purposes of data collection and movement comparison. Although other barriers to VR adoption can be removed with the choice of hardware and development approach, unfortunately no methods have been found to completely eliminate the risk of motion sickness in VR.

The cause of motion sickness in VR is attributed primarily to a sensory mismatch between visual and vestibular systems. When a user's visual input suggests movement that is incongruent with the physical input received by the vestibular system, it can result in discomfort and the symptoms of motion sickness (Conner et al., 2022). This is particularly common in VR applications that involve artificial locomotion or rapid, abrupt movements.

The implications of motion sickness are substantial as the user discomfort directly reduces immersion. Pöhlmann et al., (2023) demonstrated that the onset of motion sickness symptoms can negatively impact task performance in VR, reducing the effectiveness of the simulation. This is especially relevant in the context of healthcare, where the effectiveness of training or therapeutic interventions could be compromised by the user's disorientation when using VR.

Various strategies have been explored to mitigate motion sickness in VR. Technological improvements such as reducing latency, increasing frame rate, and optimizing field of view have been shown to minimize symptoms (Kawamura & Kijima, 2016). Design strategies have also been employed, such as the use of teleportation for movement or incorporating a static visual reference point within the VR environment (Monteiro et al., 2021). However, these solutions are not universally effective, as individual susceptibility to motion sickness can vary widely.

In the development of the VR hospital framework presented in this thesis, careful attention was paid to controlling motion sickness. The framework employs a user-centered design, favoring a choice of movement method (teleportation or sliding) and full head turning to limit artificial locomotion and reduce sensory mismatch during navigation trials (demo trial 2). Additionally, the design encourages self-paced interaction, allowing users to adjust their experience pace to their comfort level. Future studies involving the framework should consider user feedback related to motion sickness, since this is shown to provide valuable insights into further refinement of the design as shown in the current thesis.

Motion sickness in VR remains a significant challenge, affecting the usability and effectiveness of VR applications. By understanding its causes and effects, and by exploring and implementing mitigation strategies, it is possible to improve the user experience and increase the effectiveness of VR simulations. This is particularly crucial in the context of healthcare, where the aim is to enhance learning and therapeutic outcomes, and to ensure the safety and comfort of users.

Chapter 3

3 « Development & Evolution of VR Hospital System Simulation »

This section summarizes the development process for the VR Hospital Simulation Framework, which includes the development decisions made and the rationale behind each. It begins with the process of simulation subject selection and ends with the full development timeline for the framework, including mitigation strategies to address issues inherent to VR as a medium. The thought process behind the development is outlined in the flowchart below (Figure 3-1), where the red path represents choices made from subsections 3.1 to 3.6 and the black path traces the development timeline explored in subsection 3.7. The split paths found at both the subject and engine selection gateways represent independent parallel processes which need to be completed before they merge at the next major task.

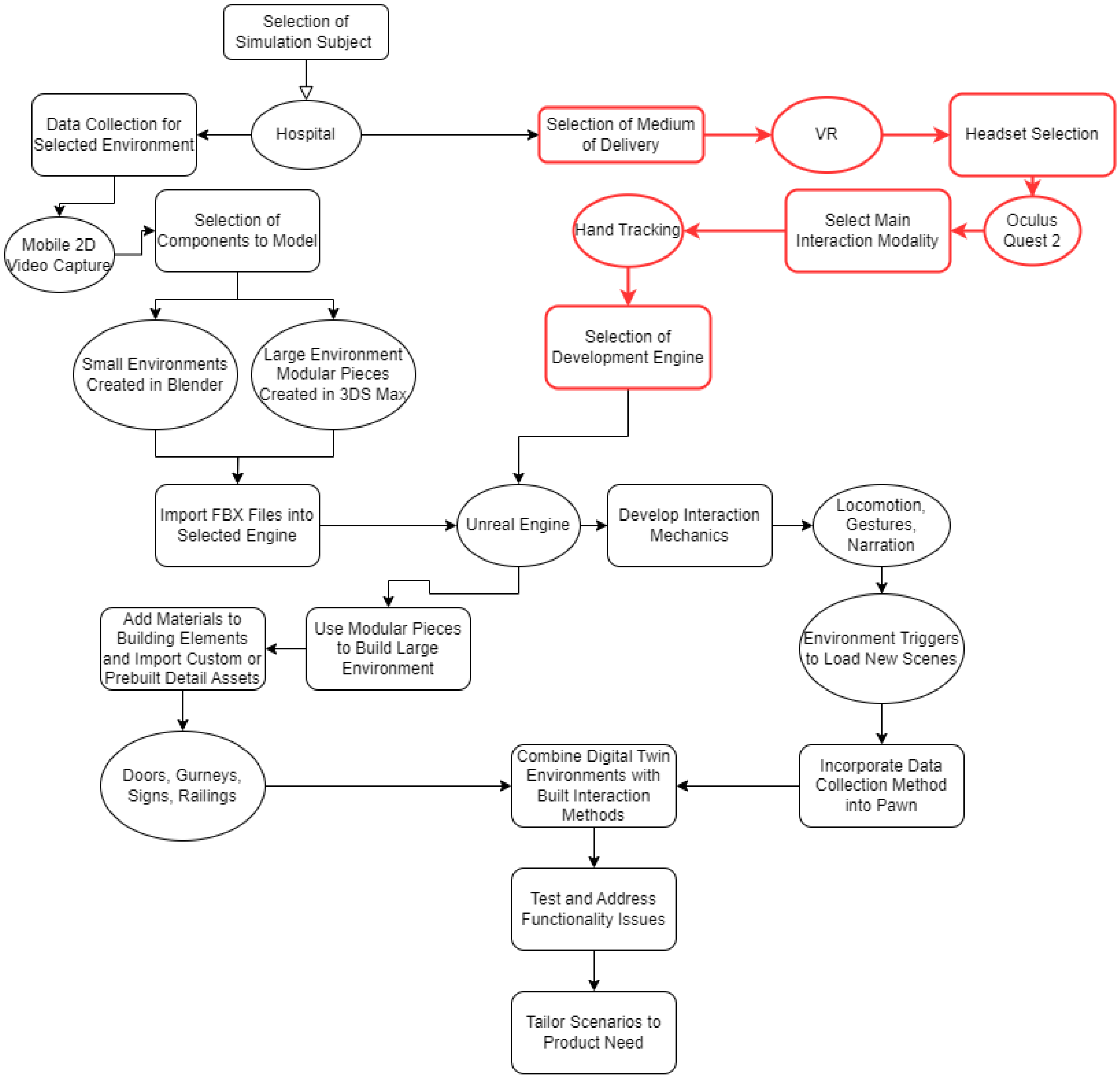


Figure 3-1 Process of Developing the VR Framework

The purpose of the simulation should be established, and its successful characteristics defined, prior to visiting the first node. The goal is to create a framework system which is expandable and from which new functionality can be adapted, regardless of developer experience level. Furthermore, it is critical for the simulation to be immersive, interactive, and collect data for future framework improvement.

The users should feel a level of physical and perceptual comfort in the environment as they explore and learn from the experience. Also, the data collection functions must be built in and remain modular to allow for adaptability. This data and adaptability will enable others to modify or extend the software, customizing it to meet their specific objectives.

3.1 « Simulation Subject Selection »

Simulations of physical environments allow for the possibility of a tactile experience, affordable iterations and interactive mechanics. The first step is to narrow down what kind of physical environment to simulate, the primary decision criteria of which was value. It was important to select an environment best suited to showcase the value of the workflow explored in this thesis. Although there is no objective measure for which field is most effective in demonstrating the usefulness of new technologies, healthcare and military are commonly among the sectors where many of them see broad-scale commercial application. Given their role as pillars of a nation's stability, these fields receive the spotlight regardless of a nation’s ever-changing circumstances. This was on display with the COVID-19 pandemic, during which national security and public health received non-stop attention, criticism and funding. With these factors in mind, the decision was made to create a simulation of a healthcare center, specifically, Victoria Hospital in London Ontario to showcase the value of the work. Once the subject of the simulation has been selected, the programmer is tasked with identifying its components.

3.2 « Components of the Simulation »

A successful Victoria Hospital simulation requires that a user feel deeply immersed in the digital environment and a high level of comfort when using interactions and movement mechanics. The software should include tools for data collection; the insights from which can be used to make modifications and improve the system. A simulation of a physical environment consists of 2 main components: a digital twin of the environment, and a method of interaction for the purpose of exploring and experiencing particular scenarios.

3.2.1 Digital Twins

A digital twin is a virtual counterpart of a physical entity, which leverages elements of both the virtual representation and the physical environment to the benefit of the entire system (Jones et al., 2020). At its core, the purpose is to allow users to visualize the environment in digital space by producing a duplicate, or “twin”, of the subject (Grieves, 2016).

Components of a digital twin:

1. Physical location in real space
2. Virtual environment in virtual space
3. Connections of data and information that tie the two together (Grieves, 2015).

These twins afford professionals a compact, sometimes manipulable virtual representation of physical entities which are otherwise inaccessible, dangerous, or time-consuming to access. As such, they can reduce costs, improve safety, and reduce wasted effort. Additionally, some projects or tasks can benefit from increased accuracy by prototyping entities as digital twins prior to physical testing. For example, digital twins can be used in architecture to create 3D floor plans, in manufacturing to create digital prototypes, and in healthcare to recreate clinical environments. In the past, developing a digital twin has required significant expertise, resources, and time. In this study, the developer leverages resources available at little to no cost, to create a digital twin of various sections of Victoria Hospital, showcasing a novel development workflow. This interactive digital twin program can easily be adapted, expanded, or modified based on project requirements.

3.2.2 Interaction

Interactivity is an important component of simulations as it enables users to engage with the environment and manipulate objects in a virtual space. This learner-content interaction is essential for ensuring the intended learning outcome and makes learning more engaging and effective, providing users with a sense of control and agency (J. Xiao, 2017).

A study conducted by SenthilKumar, (2019) found the use of interactive simulations in physics education increased academic achievement in students. In the study, the experimental group using interactive simulations had a 21.14% increase in closed book test performance compared to the control group, suggesting that active learning simulations promote information retention more effectively than passive observation or auditory instruction. The researchers conclude that improved retention, motivation and engagement is a result of the sense of agency and action which the interaction provides.

In a healthcare setting, haptic feedback can be used to simulate patient interactions like taking vitals, administering medication, or performing difficult procedures, allowing trainees to practice tasks without causing potential discomfort to patients. This was the basis for a haptic-feedback enabled VR simulation framework for neonatal endotracheal intubation which was recently developed by Xiao et al., (2020) so that medical professionals could practice this difficult procedure in a safe and controlled environment.

In an industrial setting, manipulable environments can be utilized to simulate processes, test prototypes, improve efficiency and reduce the need for physical trials. In a recent study by the CEA (French Atomic and Alternative Energies Commission) virtual reality technologies were successfully applied in a nuclear decommissioning context. This included the use of VR technologies to simulate remote handling systems and optimal dismantling scenarios to help avoid foreseeable problems during real operations (Chabal & Soulabaille, 2016). In these situations, where radiation exposure poses a threat, practicing in manipulable virtual environments improves response time, increases efficiency and reduces the number of physical trials needed.

However, designing effective interactive interfaces requires careful consideration of user needs, accessibility, and usability. Interactivity can also be computationally expensive and requires significant resources to develop and implement. Therefore, it is important to balance the benefits of interactivity with the many costs, all while ensuring the simulation design adheres to evidence-based practices.

Furthermore, the ethical implications of using simulations for training and education should be carefully considered to ensure that they do not cause harm or reinforce stereotypes. Although each developer will attempt to avoid it, decisions made around system interface optimization will inevitably reflect their biases and assumptions. For this reason, user testing is vital in the early stages of development. Though the number of participants considered sufficient would depend on the individual study, having a diverse range is important. The covid pandemic made it challenging to secure participants for early user testing. As a result, early feedback was limited. The future work portion of the thesis reflects recommended changes which arose from final user testing feedback.

3.3 « Medium of Simulation Delivery »

The delivery medium is the main transmission point between the user and the digital world and lays a foundation for the interactions to be built on. Compatibility with interaction modalities and effective hospital environment delivery are key considerations for medium selection. Simple 2D displays such as computer monitors or tablets remain the current industry standard for viewing digital twins in use cases like CAD (Computer Assisted Design), virtual building tools, or part inspection. These involve manipulation via keyboard and mouse for a monitor, or touch screens for a tablet. This approach can be used for specific cases like architecture, construction, or manufacturing, where the goal is to view and maybe manipulate the environment at a high level. However, it is not optimal for situations where the goal of the digital world is to improve layout familiarity or environment acclimatization. The more appropriate delivery medium in these cases is XR, which is a cutting-edge display platform that stimulates deep human-digital interaction (Xiong et al., 2021). XR or extended reality is an umbrella term that refers to immersive technologies, like virtual reality (VR), augmented reality (AR), or mixed reality (MR), which extends human experiences beyond the physical world. In the case of the Victoria Hospital Simulation, the immersive and interactive nature of the VR environment increases patient comfort levels with clinical procedures and improves efficiency by familiarizing them with the hospital layout ahead of time.

3.3.1 Virtual Reality (VR)

Virtual reality (VR) is a disruptive technology that creates entirely digital environments with which users can interact. Users wear a head-mounted display (HMD) that projects a

3D environment directly in front of their eyes. The headset tracks its movements with varying degrees of freedom and adjusts the view of the 3D environment accordingly. Degrees of freedom refer to the number of directions a headset can track in virtual space. Every headset explored in this study is a 6 degrees of freedom device, meaning it tracks both rotational (pitch, yaw, roll) and translational (forward/backward, left/right, up/down) motion. There are 2 types of VR headsets: 1) Tethered (sometimes referred to as PCVR only), which require a physical connection to a computer. 2) Standalone, which have all of the necessary components built in. VR offers enhanced immersion, increased engagement, and improved learning outcomes. Some applications which leverage these capabilities include gaming, training, education, and collaborative group work. For instance, a study by (Mao et al., 2021) found that medical students using a VR simulator for training showed significant improvements in their surgical skills, particularly in procedural time to completion.

In psychology, VR has also been used for exposure therapy, as a tool for helping patients overcome phobias or anxiety disorders by simulating environments or situations that trigger their symptoms in a controlled setting.

However, VR also presents some challenges, such as the need for high-performance hardware, and the possibility of users feeling isolated from the real world after extended sessions. Furthermore, a major technical issue which has limited adoption is the potential to induce motion sickness, which is triggered when movements in the physical world do not correspond with the virtual.

3.3.2 Augmented Reality (AR)

Augmented reality (AR), it is a semi-immersive technology that displays digital content over the actual environment surrounding the user (Azuma et al., 2001). Commonly used in mobile applications, AR leverages a smartphone's camera and motion sensors to recognize real-world objects, using them as anchors and augmenting the surrounding area with digitally generated assets. Navigational apps like Google maps are beginning to use this technology to scan physical environments for anchor points on which to overlay guidance elements (arrows and paths) that help users orient themselves. The 2016 global phenomenon “Pokemon GO” is another example of a mobile application using augmented reality technology. It encouraged users to travel to real-world locations on a map and use their smartphone cameras to find and catch digitally rendered Pokemon. Aside from AR apps on mobile devices, wearables like smart glasses have also been developed to allow for a more immersive experience by overlaying digital assets directly in the user’s field of view. In addition to its use in gaming and navigation, AR is also used in marketing, collaborative work, and education (Bacca-Acosta et al., 2014).

Though AR offers enhanced interactivity, improved engagement, and increased accessibility, it has drawbacks, including the need for precise tracking, a limited field of view, and the risk of information overload.

There are two subcategories of AR: Video see-through and Optical see-through.

- The most common is video see-through AR, which can be implemented using almost any smartphone. This includes any AR experience in which the mobile cameras embedded in a device stream a live feed and render digital assets overtop of the video. Using the same rendering pipeline as the live feed without the need to calculate the eye’s perspective makes this form easier to implement. Drawbacks include the need for a high frame rate and level of fidelity (for capture and display), both of which lead to a satisfactory user experience and if implemented with a headset can lead to similar issues to VR. It is the AR used in the navigational programs and games mentioned above.
- The other option, Optical see-through, necessitates combining the digitally generated assets with a see-through lens where the user directly sees the physical world around them. This is usually implemented using an angled semi-reflective lens mirroring an image generated by a screen, allowing the user to see the elements overtop what they see in the real world. The advantage of this method is that if the device malfunctions or ceases to function, the user can still see the world around themselves through semi-transparent lenses. The development challenge of optical see-through AR is that since humans see the world in 3 dimensions with a level of depth,

the graphics need to go through a manual translation layer to line up and provide some semblance of depth correctly and the programmers must account for the perspective of the eye.

3.3.3 Mixed Reality (MR)

Exploring the third sub category, Mixed reality (MR) is an immersive technology that mixes the real world with digitally generated content, allowing users to engage with and manipulate both (Kishino & Milgram, 1994). MR combines the physical and digital worlds to generate a hybrid environment in real-time, paving the way for communication and collaboration in a similar way to the others but in a way that allows collaborators to influence the environment around themselves and each other. MR is often used in industrial applications like designing and testing products and training and education (Maas, 2017). MR shares some of the same advantages and challenges as AR, but the major difference lies in that MR allows users to interact with the natural world simultaneously with the digital content.

Much like augmented reality, MR allows for greater spatial awareness and improved productivity. Also similar to augmented reality, the problems lie in the requirement for precise tracking, complexity, and the potential for digital distractions. Although AR and MR may seem similar, their implementation and uses vary. While AR has a digital content anchor to real-world objects and locations, allowing the user to interact with digital artifacts within that environment, MR, as previously mentioned, blends the two into a hybrid in which the user can interact with the natural world simultaneously with the digital content. While implementations like Snapchat use AR to overlay filters on the natural world or faces, implementations like Ford’s automotive design and prototyping process use mixed reality to allow engineers to interact with the virtual models and the physical space.

3.3.4 Selection

Virtual reality (VR) is the most suitable choice for the hospital simulation, as it addresses both user interaction and full immersion within the modeled environment. While augmented reality (AR) and mixed reality (MR) headsets enable users to view and interact with digital twins of objects, they lack the ability to fully immerse the user in the digital environment. VR, on the other hand, provides a completely immersive experience, allowing users to explore and interact with the digital representation of the hospital without distractions from the real world. This level of immersion is advantageous for educational purposes and practicing medical procedures in a controlled environment (Napocensis et al., 2021). Moreover, VR has been successfully used in various medical applications, further supporting its selection for this study (Bacca-Acosta et al., 2014). As such, a head-mounted VR display is the ideal device choice, as it provides the necessary immersion while eliminating distractions from the surrounding non digital environment, giving the developer complete control over the user's experience. VR also promotes a high degree of visual information retention, as demonstrated in the study by Huang et al., (2019). Once the medium of delivery is specified, the most suitable headset for the project is selected based on its capabilities and limitations.

3.4 « Headset Selection »

The most popular headsets on the market are produced by Meta (formerly Oculus), Valve, HTC, Pimax, and Pico (Table 1). These headsets fall into two categories: PC VR-only headsets (also called tethered) and standalone android-based headsets. Some standalone headsets have the ability to offer full PC VR functionalities when tethered, though with noticeable drawbacks. Similar to monitors, PC VR-only headsets display content rendered on an external PC and transmit the images to the headset through high bandwidth proprietary cables. Standalone headsets, on the other hand, are powered by mobile chips capable of storing, rendering, and displaying virtual content without external connections. Each type has its own advantages and disadvantages.

Tethered headsets provide application developers more freedom and allow for higher-quality graphics due to their ability to leverage a PC's processing power. They also offer more advanced and precise tracking capabilities with their lighthouse-based outside-in systems. However, these headsets require a powerful computer and a physical tether, making the system expensive and limiting its flexibility and portability. The setup, teardown, and calibration of each lighthouse also add to the complexity.

Standalone VR headsets offer a more portable and flexible option, as they can be used anywhere without needing a separate computer for processing. Despite including an onboard processor, Wi-Fi card, and battery, these headsets are often cheaper than their tethered counterparts. With no need for a tethered computer or lighthouses, these headsets have a shorter and simpler setup process, making them more accessible for non-technical users, medical staff or research students. The downside is the reduced processing power of the mobile chip, which results in lower graphics quality. Additionally, inside-out tracking, while innovative, has limitations due to its reliance on cameras mounted on the user's face, which can only capture a hemisphere-shaped field of view in front of the user. This can lead to imperfect motion predictions when subjects hands/controllers move out of view, resulting in tracking errors and a less immersive experience. To make an informed decision on headset selection, it is crucial to explore the most popular devices available in the market.

3.4.1 3.4.1 Manufacturer Research

This subsection provides an overview of the headsets considered for this project, each manufactured by one of the previously mentioned companies. The specifications discussed in this subsection are summarized in Table 3-1.

Table 3-1 List of VR capable headsets under consideration

Manufacturer	Headset	Launch Year	Type	Resolution (per eye)	Price	Refresh	FOV
Valve	Index	2019	PCVR	1440x1600	1300	120	130
HTC	Vive Cosmos Elite	2020	PCVR	1440x1700	1250	90	97-110
HTC	Vive XR Elite	2023	Standalone	1920x1920	1500	90	110 max
HTC	Vive Focus 3	2021	Standalone	2448x2448	1750	90	120

Pimax	Vision 8k X	2019	PCVR	3840x2160	1750	60 - 114	170
Pimax	12K QLED	2022 (Delayed 2024)	PCVR/ Standalone	5760x3240	3275	75 - 200	150-200
Pimax	Crystal	2023	Standalone	2880x2880	2200	160	120-140
Pico	Neo 3 Pro	2021	Standalone	1832x1920	1000	72-90	98
Pico	Neo 4	2022	Standalone	2160x2160	600	90	105
Meta	Quest 2	2020	Standalone	1832x1920	460	90-120	85-97

3.4.1.1 Valve

In 2019, Valve, a company primarily focusing on software development, released their cutting-edge and much-anticipated Index headset (Figure 3-2), which was out of stock for nearly two years afterwards due to demand. The Index falls within the PC VR only category and was known for the high-quality optics and comfort of the headset itself. It has a 1440x1600 resolution per eye, a native 120Hz refresh rate, and a 130-degree FOV, making it highly immersive. The headset has fully adjustable lenses and a well-designed factory head strap, making it comfortable for extended sessions. The Index was also well known for the included, highly innovative "Knuckles" controllers (Figure 3-3). These controllers were designed to be more ergonomic and intuitive to hold than the competitors at the time of release and are still highly regarded in 2023. The controllers moved away from the touchpad design of its predecessors (HTC controllers) and introduced thumb sticks to VR controllers and brand-new force sensors. The force sensors were embedded into the controller's handle along with a full array of capacitive sensors, with the controllers being strapped to your palm rather than your wrist. This allowed software to recognize full finger articulation accurately and the ability for the user to fully release the controllers without leaving the optimal grip position. They also use these(force) sensors to detect the force of the grip strength the user exerted. The controllers also had the unique ability to track the movement of user's hands, even if they were outside the lighthouse's field of view. The downside of the Valve Index is not limited to the high PC cost of tethered headsets in general but also its CAD 1300+ MSRP launch price and the inability to purchase due to stock shortage. The points of consideration with this headset are the robust tracking capabilities with the new base stations and the innovative and ergonomic controllers. Out of all controllers studied for this thesis the knuckles controllers provide the most immersive and comfortable interaction experience.

Figure 3-2 Valve Index Headset

Figure 3-3 Valve Knuckles Controllers

3.4.1.2 HTC

HTC, a veteran in the field of VR headset development, released their flagship Vive Cosmos Elite (Figure 3-4) HMD in 2020. The Cosmos Elite fell under the tethered PC VR category and came with a 1440x1700 resolution per eye at a 90 Hz native refresh rate and FOV of 97 – 110 degrees. The Cosmos and Cosmos Elite headsets were a unique line using a swappable faceplate; choosing between inside-out tracking or base station implementing outside tracking was possible. Although this is a unique feature among VR headsets, on a tethered PC VR headset where portability is limited, including the option to use inside-out tracking only helped lower the setup time. The base stations used for the headset out of the box are also dated as they are the first-generation stations for SteamVR 1.0 rather than the upgraded ones included with the year older Valve Index. The headset's design was based on a flip-up function which allowed the user to switch between VR and the natural world by tilting it up, like a welding mask instead of removing it entirely. The Cosmos Elite launched at a similar $1300 price point as the Valve Index and shared the high cost of entry with the requirement of a high-end PC. At the entry price, the Index is superior in most aspects and would be considered for the study before the Cosmos Elite. HTC also recently released a standalone-only headset called the Vive XR Elite (Figure 3-5) which featured a resolution identical to the Vive Cosmos Elite with the same 90 Hz refresh rate. The Vive XR Elite was also touted as a VR/AR compatible headset although for the purpose of this study only VR capabilities are considered. This headset was out of consideration as it retailed for double the other standalone headsets reviewed for the study and released in 2023. The third researched HTC headset, the Vive focus 3 (Figure 3-6) was designed for use in professional environments in the business sector and boasted a 2448x2448 per eye resolution at 90Hz and a high FOV. Unfortunately, coming in at a high price of 1750 it was also eventually ruled out in favour of a cheaper device.

Figure 3-4 Vive Cosmos Elite w/ Controllers and Lighthouses

Figure 3-5 Vive XR Elite w/ Controllers

Figure 3-6 Vive Focus 3 w/ Controllers

3.4.1.3 Pimax

Slightly different from the rest of the companies was Pimax. Pimax's priority in manufacturing is to develop the highest fidelity headset at the time of release and has announced a new headset each year, pushing the boundaries of VR visual fidelity. Although these headsets were researched for this study, they were ruled out almost immediately as the high fidelity did not justify the difficulty of acquisition and high price points. In 2020, their flagship headset was the Pimax 8K X (Figure 3-7). Although this headset came in at a lower refresh rate of 75 Hz, the resolution of the displays was 3840x2160 per eye with a FOV of 200 degrees. Unfortunately, this impressive PC VR headset was released at USD 1849 removing it from consideration for the simulation. Since then, Pimax has unveiled two new flagship devices, the Pimax 12K QLED in 2021 and the Pimax Crystal (Figure 3-8) in 2022, which can be used both as tethered PC VR devices or as standalone devices with their native visual fidelity being decreased. Both headsets outstrip all competitors in price and visual fidelity, with the Crystal boasting a resolution of 2880x2880 per eye at 160 Hz and a 140-degree FOV but, unfortunately, retailing at USD 1600 when it launches. The 12k QLED is a headset announced in 2021 but has yet to be released. This headset boasts an unbelievable 6K per eye resolution in PC VR mode with a 200 Hz refresh and 240-degree FOV and a 4K per eye resolution in standalone mode with a 120 Hz refresh and a 200-degree FOV. These impressive specs come for $2400 USD, making it the highest-price headset researched for this study. Although these headsets are inaccessible for a practical application in this study, the technology they use can help drive further VR development and better paint the picture of the hardware landscape at the time of writing. The workflow created in this study can be used to develop simulations for any headset meaning they could be considered in a future case where fidelity is a higher priority than accessibility.

Figure 3-7 Pimax 8K X

Figure 3-8 Pimax Crystal w/ Controllers

3.4.1.4 Pico

Moving into the brands focusing on standalone headsets, the leading overseas competitor is Pico. In 2021 and 2022, this company released the Pico Neo 3 Pro (Figure 3-9) and the Pico Neo 4 (Figure 3-10), respectively. The Neo 3 Pro brought a resolution of 1832x1920 pixels per eye, while the Pico 4 upgraded this to 2160 x 2160 per eye.

As the Pico 4 is a recent headset, it has access to innovations that were not available at the time the previous headsets were released. One of these innovations was the adoption of pancake lenses to make the headset more compact, comfortable, and greatly reduce the visual artifacting present on every headset using the older Fresnel lens design. The artifacting took the form of the screen door effect and the god ray effect. The screen door effect, which arises from the uneven distribution of pixels on a display, causes an apparent grid-like pattern that can be observed when viewing content. This gives the impression of peering through a screen door. The god ray effect, commonly known as lens flare or light scatter, produces streaks or halos of light around bright objects in photographs and videos. With the size reduced, the main body of the Neo 4 is almost half the size of its main competitor the Meta Quest 2, and more comfortable to wear for longer periods of time.

Although the Pico headsets are great standalone headsets, they are difficult to procure in North America. These headsets also sell at a relatively high price of 1000 CAD (approximation) for the Neo 3 Pro and 600 CAD (approximation) for the Neo 4. Among Pico's headsets, the Neo 4 was the first to introduce native hand tracking, which would allow for a greater level of customization in software development with the option to program for controller use or hand tracking.

Figure 3-9 Pico Neo 3 Pro w/ Controllers

Figure 3-10 Pico Neo 4 w/ Controllers

3.4.1.5 Meta (Oculus)

Oculus (also referred to as Meta) has been one of the most valuable VR headset manufacturers since the boom of VR development. The latest consumer-level headset in their arsenal, the Quest 2 (Figure 3-11), has become one of the market's most influential and popular headsets. The Meta Quest 2 is a standalone headset that can tether to a PC for more advanced graphical performance, albeit not at the same level as a dedicated PC VR headset. Rather than using a myriad of cables like many tethered headsets, the Quest 2 can physically tether to a computer using a single USB C cable (which inherently has lower throughput than dedicated display connections of PC VR headsets). The Quest 2 also has the hardware to tether to an external PC wirelessly, as long as both are on the same network with Wi-Fi 6 capabilities. This functionality works with similar limitations to the USB C tether on the Quest 2, where it is limited by bandwidth and unable to quite match up with tethered headsets. This is most visible if the user turns their head very quickly as they will see the edges of the eye box, meaning the field of view is pre-rendered, compressed and sent to the headset, similar to how a video feed works. This ability was often used in development to remove the interference of the cable. A mobile setup using this technology could allow developers to leverage the power of PCVR with the comfort and portability of standalone headsets. The headset is equipped with a resolution of 1832x1920 per eye with the help of a single panel split across both eyes with a native refresh rate of 90hz, an experimental refresh rate of 120hz and an approximate FOV of 91 degrees. As mentioned when describing this headset category, the Meta Quest 2 uses inside-out tracking to allow for impressive portability and ease of setup. Setup for this headset requires the user to stand near their chosen VR space, touch the ground to calibrate ground height, then, using the cameras and a monochromatic video see-through AR experience, draw a guardian border around their play space. If they approach it, to avoid collisions, this border appears in the user's vision. A bonus for the Quest 2 is that similar to the Pico 4, it also supports native hand tracking which opens up new possibilities for interacting with the environment without a controller. The Quest 2 was initially priced on release at CAD 460 for the 64 Gb version but had since increased due to inflation then lowered to offer the 128 GB at that same price since the release of the Meta Quest Pro (priced at $2000+). This low price point compared to other headsets coupled with the flexible processing method (standalone and PCVR) and the portability brought upon by inside-out tracking made it a solid contender for the study demo. Another downside to Quest 2 along with those of inside out tracking, was the mandatory Facebook sign-in, which raised many privacy concerns. This also left users without an account with a non-functioning device, this mandate was later removed by Meta, allowing free use of the headset. Including a processor, battery, and wireless connectivity hardware, many users initially found the device to be front-heavy on their faces. This problem is only further exacerbated with the stock headband; the headset sits on your face with minimal support leading to many users complaining about nose, neck and head soreness. This problem can be easily remedied by purchasing a third-party Halo-style headstrap or the official elite strap (Figure 3-12) made by Meta as well as by keeping simulation sessions short. Short sessions are a valid solution to this issue as standing Virtual Hospital experiences would last no longer than the procedure itself with no waiting for queues.

Figure 3-11 Oculus (Meta) Quest 2 w/ Controllers

Figure 3-12 Quest 2 Elite Strap

3.4.2 Headset Selection

Following the Engineering design method, a decision matrix (Table 3-2 Decision Matrix for Headset Selection was generated to compare the headsets under consideration. Categories included:

1. Price

2. Ease of Purchase
3. Portability
4. Flexibility (Standalone and PC capable)
5. Interaction options
6. Reported stock comfort level

Each category was assigned a weight based on its priority level, with the order of the list reflecting their importance. Price was the top priority since the workflow in this study aims to utilize readily available resources to minimize the high costs associated with current digital twin and environment interaction development. The software is also intended to be an accessible starting point for those wanting to implement new hospital training simulations and evaluations. To showcase its widespread capabilities, the system is optimized for low power hardware.

Ease of purchase was the next priority, as certain headsets were frequently out of stock or unavailable for purchase in North America. The goal was to develop software that is easy to deploy and cost-effective. While PC VR headsets offer higher visual fidelity, mobility and adaptability were of greater importance.

Interaction options were a lower priority since at the very least all VR devices researched could be controlled using controllers, which are acceptable for developers and users. However, they could cause issues for those unfamiliar with VR, as controller buttons are not visible from VR at a glance. The lowest priority was the reported stock headset comfort level; as a personal trait that varies among users, it was challenging to assign a score without side-by-side comparisons and most headsets also allow for modification to enhance fit and comfort. Visual fidelity was not included as a factor, as each headset on the list reaches a minimum level of fidelity acceptable for most use cases.

After scoring each headset based on these categories (Table 3-2), the Meta Quest 2 emerged as the clear winner. With its lower price point, flexible deployment style (standalone and PC VR, including wireless PC VR), portability, dual interaction options, and ease of purchase, it outperformed all other headsets. The developed workflow supports both standalone and PC VR applications, making the Quest 2 the perfect match for the study. Its use facilitates easier scaling and requires a lower initial investment for small organizations or individuals.

Table 3-2 Decision Matrix for Headset Selection

High score is better	(6) Price	(5) Ease of Purchase	(4) Portability	(3) Flexibility	(2) Interaction options	(1) Stock Reported Comfort	TOTAL
Valve Index	0	1	0	0	0	1	6
HTC Vive Cosmos Elite	0	1	0	0	0	1	6
HTC Vive XR Elite	0	1	1	1	0	1	13
HTC Vive Focus 3	0	0	1	0	0	1	5
Pimax Vision 8k X	0	1	0	0	0	0	5
Pimax 12K QLED	0	0	1	0	0	0	4
Pimax Crystal	0	0	1	1	0	0	7
Pico Neo 3 Pro	0	0	1	1	0	0	7

Pico Neo 4	1	0	1	1	1	1	16
Meta Quest 2	1	1	1	1	1	0	**<u>20</u>**

3.5 « Development Engines and Tools »

Selecting development tools and a dedicated engine is essential to the study as they form the basis of the workflow being explored for small team simulation creation. From the study's inception, there was always a plan to leverage a pre-existing game engine. Game engines now provide a ready-to-develop environment with fundamental functionalities desired by programmers, saving time in initial development and subsequent iterations. Popular engines handle memory management and asset loading, while more capable ones also manage lighting and physics. They offer out-of-the-box integration with artist and CAD tools, compatibility with most 3D asset types and textures, and allow for cross-platform development.

Using pre-built libraries and modules, developers can bypass low-level technical details, as they are handled natively and popular engines like Unreal Engine (UE) and Unity natively support most commonly used headsets. Using a pre-existing game engine saves unnecessary costs and time. However, some disadvantages include the need for Software Development Kit (SDK) and codebase knowledge to modify the engine (needed for UE 4.25 for hand tracking), potential engine development bugs, and possible inefficiency in smaller projects due to overhead requirements. For this study, a VR simulation is too large a project to be coded by an individual without an engine.

3.5.1 Past Game Development Technologies

Before the dominating popularity of the two game engines above, VR training simulators were still developed. However, they required the resources of a large organization, making it almost impossible for an individual or small team to compete. In this thesis, the most popular options before the widespread use of Unreal Engine and Unity have been listed below, along with some early implementations of XR simulations:

- One of the most prevalent was **Vizard**, created by the company **WorldViz**, which found application across diverse industries like aerospace, automotive engineering, and military training. It still exists today and is based on the Python scripting language. Vizard provides a versatile range of tools for constructing immersive virtual settings that feel authentic and engaging, and it is adaptable with a wide array of VR equipment, including head-mounted displays, motion tracking systems, eye trackers, biofeedback monitors and haptic feedback devices.
- Another popular development engine, **Virtools**, a product of **Dassault Systèmes**(now known as **3DVIA**), was another VR development platform employed in professional simulations. It was prevalent in the automotive sector, where it was used for generating driving simulators that featured realistic driving environments and precise vehicle dynamics and physics models.
- With the advancement of virtual reality (VR), **Virtual Reality Modeling Language** (VRML) was introduced as an interactive tool for users to create and view VR environments through web browsers. This technique was succeeded by **X3D**, which is still widely used today. The ability to create and experience simulations in an accessible manner brought significant progress toward furthering VR's potential. Although the **Web3D Consortium** owns X3D, it is essential not to confuse it with the unifying framework, web XR which is a JavaScript API.
- Furthermore, the **OpenSceneGraph** library played a significant role in the creation of bespoke VR development platforms within the professional simulation domain. OpenSceneGraph is a potent graphics engine that offers a plethora of tools for designing vivid virtual environments, and it could be customized to accommodate specific simulation requirements.
- In the late 1980s, **NASA** developed the **Virtual Environment Workstation Project** (**VIEW**), a VR training simulator for astronauts. VIEW utilized a head-mounted display developed by **VPL research**, a data glove developed by **MIT**, and a 3D audio system created by **Crystal River Engineering** to create an incredibly realistic simulation of space activities. The software which held all these aspects together was written directly in C and used a custom graphics

library called **IRIS GL**. You can see how, 30 years ago, VR simulation development took several large organizations with expertise in the required sector. This pioneering approach laid the groundwork for VR technology and enabled advancements in VR training and simulation systems.

- Another noteworthy example is the **Cave Automatic Virtual Environment** (**CAVE**). It was developed by the **University of Illinois** in Chicago in 1992 and functioned similarly to the headsets used today, but instead projected images onto a room-sized cube's walls and floor. With head-tracking devices designed by **Polhemus Inc**. and stereoscopic glasses at its disposal, VR technology took another step forward as it can now create immersive simulations for several users. This development generated collective immersive experiences, which are now an integral part of current VR applications. The software for this experience was written in C++, used a graphics library called **CAVElib** and leveraged **OpenGL** for rendering.
- In addition, the **Distributed Interactive Simulation** (**DIS**) protocol emerged from the **Department of Defense** in 1989, allowing multiple simulators to connect and operate within a shared virtual space. The network protocol was developed by the **Institute for Simulation and Training** at the **University of Central Florida**, while the simulation engine itself was developed in C by the **Defense Advanced Research Projects Agency (DARPA)** using the graphics library called **Performer**. The way this project linked multiple VR simulations through this technique has paved the way for collaborative VR training and simulations, providing practitioners with novel means to improve their skills.

Vizard, Virtools, VRML, and OpenSceneGraph were among the most favoured VR development platforms for professional simulations before Unity and Unreal Engine gained prominence. Furthermore, early VR Simulation systems like the DIS, CAVE, and VIEW demonstrate how a large team or several teams were required to work together to develop simulations. With the system explored in this research, all it takes is a single programmer to develop a base environment and an interaction suite which can then be expanded upon by any number of users. The system developed in this work to test the workflow could be added to by anyone with 3D modelling skills with little knowledge of the game engine used.

3.5.2 Comparison of Unreal Engine and Unity 3D Engine

Selecting the right game engine is crucial for developing a VR simulation. Unreal Engine and Unity Engine are the two primary options for development, as both are freely available to the public and have strong support communities. These engines increase development efficiency which is strong support for this thesis, proving that a small team or even an individual can transition from a real-world simulation idea to a functional prototype within the study's timeframe. By leveraging these tools, the Victoria Hospital VR simulation framework was designed to be easily expanded and adapted to suit various design needs. A next-generation method for expanding the framework easily is outlined in chapter 6. Comparing these powerful engines requires examining their highlights, general development cycles, and VR development features up to UE 4.27. A discussion on UE 5 is reserved for the future work portion of the thesis.

Unreal Engine is known for its physics engine and exceptional graphics quality, thanks to advanced lighting techniques and shading systems that produce visually stunning effects (Ciekanowska et al., 2021).

Unity users benefit from seamless access to numerous assets and plugins through the Asset Store platform, including ready-to-use VR integrations. Although Unreal Engine's Marketplace offers similar resources, Unity's offerings are nearly 3 times more extensive.

If ease of scripting languages is a priority, Unity stands out due to its more manageable learning curve and simplicity. Unity uses C# (Figure 3-13) for game development compared the UE’s blueprint visual scripting, helping developers transition to a familiar coding environment and focus on creating digital experiences.

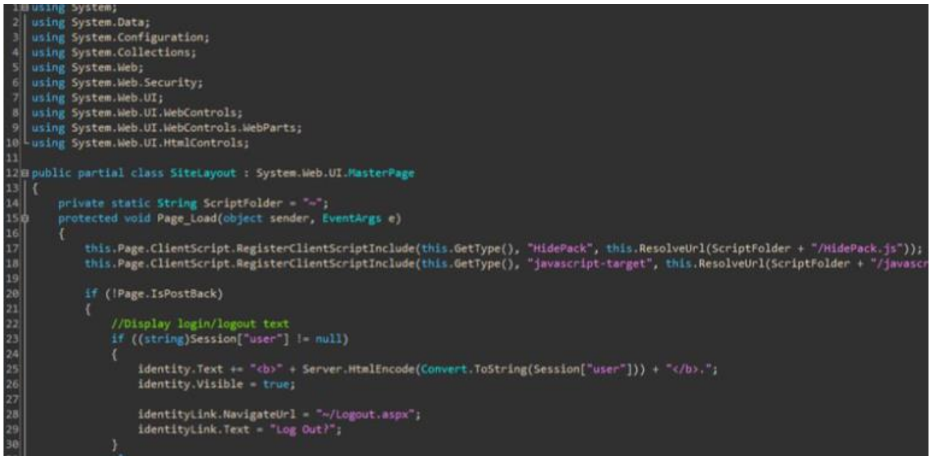


Figure 3-13 Example of C# Code

Conversely, Unreal Engine's Blueprint Visual Scripting (Figure 3-14) system provides a node-based, visual interface for creating game logic, making it more accessible for non-programmers or designers. This system enables developers to create complex game mechanics and interactions without writing code, speeding up development and fostering collaboration among team members with diverse skill sets. Unreal Engine also supports C++ as a scripting language but heavily promotes the use of Blueprints.

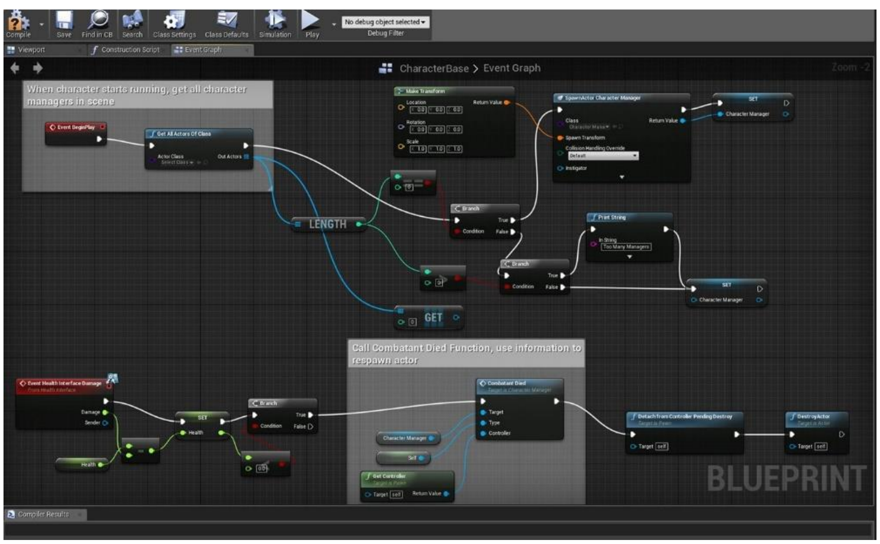


Figure 3-14 Example of Unreal Engine Visual Scripting

Beyond programming languages, virtual reality development is another significant factor when comparing these engines. Both platforms offer excellent support for motion tracking and VR hardware; however, Unreal Engine focuses more on VR development optimization and ease of expansion for beginners. This engine excels due to its optimization explicitly tailored to virtual reality requirements, achieved through the visual blueprint scripting language. Although it initially presents a learning curve for those with a traditional coding background, the visual connections and functions become intuitive when developing interactions in virtual space.

After careful consideration, Unreal Engine holds a slight advantage in graphics, VR capabilities and optimization, and customization options for advanced VR simulations like the hospital simulation project. Choosing Unreal Engine allows non-programmers to learn simple steps to adapt the framework to their needs, and its excellent tools for integrating with other game development pipeline software make it a popular choice for many developers.

3.5.3 Development Tools

As game engines have limited capabilities when it comes to designing assets and textures, certain software tools were integral in development of the environment. The specific uses in the project is expanded upon in the development timeline.

- Blender - A free and open-source program for 3D modeling, animation and rendering. Its reliable suite of tools that range from texturing and rigging to visual effects and simulations have made it unbeatable in terms of capability. Furthermore, thanks to work done by its growing community and its open-source nature, the software has consistently delivered improvements keeping up with demands across several industries.
- Autodesk 3DS Max - A commercial 3D modeling, animation, and rendering software developed by Autodesk, renowned for its robust toolset, versatility, and compatibility with other industry-standard applications. It is commonly used in various fields such as architectural visualization, game development, and film production. With a wide range of modeling tools, advanced modifiers, and powerful rendering engines, its also great for creating high-quality 3D assets.

- Photopea - A web-based image editor that supports many popular file formats such as PSD, PNG, JPG, SVG and GIF. It offers a similar interface and functionality to Adobe Photoshop, but runs entirely in the browser. As such, it can also be used to refine textures and edit images. It is a convenient and accessible tool for editing images and creating graphics without installing any software.

Both Blender and 3DS Max were used in this study to demonstrate the different programs, Blender's interface is known to have a steeper learning curve but is highly customizable. Meanwhile, 3DS Max has a more familiar and user-friendly interface, is easier for newcomers to learn to use, and also has a much more widespread adoption than Blender in the professional industries. Each has its distinct strengths, 3DS Max is known for its robust modeling tools, advanced modifiers, and compatibility with other Autodesk products. Blender, on the other hand, has a strong emphasis on animation and visual effects, and has a very active and passionate user community that contributes to its development and compatibility.

Some of the modern tools used to create simulations or digital environments, which were studied along with this work and important references for new developers but not used in the demonstrated workflow, include:

- 3D Slicer – A free and open-source medical image analysis and visualization platform. It enables the creation of 3D models of anatomical structures from MRI scans, as well as perform various tasks such as segmentation, registration, measurement, and simulation. It would have been used in this product had the demo been for practitioner training and is an incredibly useful tool for the studied workflow.
- Autodesk Maya - A 3D modelling software which allows the user to create 3d assets. It also has an addition called "Create VR," which allows the modelling to be done within VR.
- A-Frame - A web framework which allows developers to create VR experiences using HTML, CSS, and JavaScript, which can be hosted in a web browser on any device. A-Frame makes use of Web XR.

- WebXR - A JavaScript API enabling web developers to access supported VR and AR devices universally.
- OpenXR - A standard for access to VR and AR platforms and devices. Developers can use OpenXR, which provides a standard set of unified APIs across supported platforms. OpenXR aims to unify XR development by enabling interoperability across different vendors and platforms. It differs from the above 2 in that it is not meant for browser applications and allows the developer to access the XR devices directly using any programming language which interfaces with C. Some web XR implementations use OpenXR as a base
- VRdirect - An online platform that allows inexperienced coders to drag and drop interfaces and templates to create simple VR simulations.
- Adobe Photoshop – An image editing and manipulation software with a comprehensive set of tools to work with. It is commonly used to refine textures and edit images for use in countless digital content creation pipelines.

3.6 « Interaction Modalities »

As established earlier in this thesis, an environmental simulation consists of both a digital twin and a form of interactivity. Now that it has been determined that the goal is creating a VR-based immersive simulation of the hospital and discussed the tools being used to design the environments, it is essential to decide how users will interact with the simulation and what kinds of actions the interactions will enable the user to do. The focus of interaction throughout the framework centers around two primary types of interactions: navigation and manipulation.

Navigation involves movement through hallways and rooms, while manipulation refers to interactions that change the state of objects within the environment, such as opening doors, turning on lights, or moving an MRI scanner table. These interaction types are essential as they lay the foundation for the experiments conducted in this study. Furthermore, the framework also considers the concepts of selection, quantification, position, and text. The users must be allowed to make selections such as the way and where they move about the environment, there must be a quantifiable method of understanding the user actions which is handled via data collection, the environments are designed to give subjects a sense of position in virtual space to then translate to physical, and finally, text interaction is done using the medical scenario recreations and also as the users read written content and are tasked with using stationary navigational tools emulating the hospital to navigate through. Not all functionality is showcased in the demo used for trials, but these concepts were kept in mind throughout the development process. A few specific interaction decisions have been set prior to development.

3.6.1 Controllers vs Hand Tracking

The Meta Quest 2 was selected as the chosen medium partially due to its dual interaction options, leading to two primary methodologies to analyze: using the Quest 2 controllers as a direct physical form of manipulating the environment, or employing hand tracking, which relies on gesture triggers and hand positioning. Controllers offer several benefits, such as intuitive joystick movement, allowing for seamless and relatively intuitive navigation in the virtual environment. Holding a physical device also provides users with a sense of familiarity, making the VR experience feel more grounded. Button presses are reliable, ensuring the system accurately interprets users' actions. However, controllers also have drawbacks, including potential confusion for inexperienced users who may struggle to learn and remember button functions and locations without being able to see them with the headset on. Additionally, there is a risk of dropping controllers, leading to disorientation or difficulty in retrieving them, even with wrist straps (Figure 3-15). Limitations in button availability may also hinder the interaction experience, and finding an appropriate place for controllers during seated or lying-down experiences can break immersion.

Figure 3-15 Right Handed Oculus Quest 2 Touch Controller

In contrast, hand tracking offers an excellent alternative for interaction with the virtual world and numerous advantages. Hand tracking allows for a more comfortable, hands-free experience, enabling users to interact with the virtual environment using natural gestures (Talbot & Chinara, 2022). This approach is often more comfortable for users of varying ages and can accommodate a wide range of hand sizes since they do not need to reach any buttons. With virtually unlimited gesture options, the system can be tailored to individual users' preferences and abilities, making it more accessible to a broader range of users. Hand tracking eliminates the need for holding a controller, reducing the learning curve for users unfamiliar with gaming peripherals, and potentially increasing immersion, as users feel more connected to their virtual surroundings without their hands full (Figure 3-16). Additionally, hand tracking allows for a more intuitive interaction method for those without gaming experience. Nonetheless, hand tracking presents challenges, such as inconsistent performance due to the technology's infancy and reliance on camera visibility, leading to occasional inaccuracies or missed gestures. Difficulty executing some gestures due to limited hand or finger flexibility, as discovered during human trials, can also frustrate users and limit the system's effectiveness, although a solution is discussed in later sections. The Quest 2 is more easily able to track controllers than hands; however, advancements in hand-tracking technology are continually being made with each update.

Figure 3-16 Hands as Seen Within the VR Hospital Framework

The Quest has improved tracking with an official Hands 2.0 upgrade, which developers can easily enable for standalone programs. Despite the drawbacks, hand tracking is chosen for the simulation (Table 3-3), as the benefits of immersion and the ability to cater to the experience for all ages are an integral part of a hospital simulation.

Table 3-3 Hand Tracking VS Controller

Metric	Controller	Hand Tracking
Intuitive pre-learning curve	1	0
Immersion	0	1
Interaction variety	0	1
Lower Fatigue (comfort)	1	0
Cross Platform Compatibility	1	0
Input reliability	1	0

Comfort when not in use	0	1
Portability	0	1
Intuitive post-learning curve	0	1

3.6.2 Roomscale VR

Room-scale VR provides another layer of interaction, allowing users to physically move within their environment, walking from place to place within the headset's guardian boundaries, thus enhancing immersion and promoting a more intuitive sense of space (Pirker et al., 2018). However, this can sometimes lead to disorientation regarding the user's starting position, causing confusion and detracting from the overall experience. To address this issue, during specific experiences like the MRI, users are guided to a specific location (such as a bed or couch) within the simulation, providing a fixed point of reference and ensuring a more coherent experience. In the system demo designed for this study, the user has two main options for locomotion within the VR space while always being able to make small adjustments by moving at room scale: teleportation-based movement and slide-based movement (elaborated on in the section below). Although these are the main methods conveyed to the user, each teleport and trigger to begin sliding sets the origin point concerning the user, removing any offset that may have occurred by the user's real-life movement. This functionality has been built into each pawn with movement capabilities, so every experience expanded from the base system will inherently allow the user to leverage room-scale VR in moving around the levels. This feature enables users to turn around freely and lean in to read smaller text, similar to the physical world.

3.6.3 Locomotion Options

Aside from roomscale, two main options for locomotion or movement were studied, teleport locomotion and sliding locomotion. Teleport locomotion allows users to target, and instantly move to a selected point within the virtual environment. This method has the advantage of reducing motion sickness for some users, as it eliminates the sensation of continuous motion in VR while the physical body remains stationary. It also enables users to navigate through the environment quickly and efficiently. However, teleportation can be disorienting for some users and may break the immersion, as it does not closely resemble real-world movement. Sliding locomotion, on the other hand, involves the user smoothly moving through the virtual environment at a constant or accelerating pace. This method more closely mimics real-world walking and can provide a more immersive experience. However, sliding locomotion may cause motion sickness for many users, particularly those who are sensitive to artificial movement. Ultimately, providing users with both locomotion options allows them to choose the method that best suits their preferences and comfort levels, making the simulation more accessible and enjoyable for a wider range of users.

3.6.4 Dropdowns and Voice activation

Two more options of interaction were studied: dropdown menus and voice controls. Dropdown menus provide users with an accessible way to navigate and control variables during the simulation, delivering a consistent interface that is easy to learn and use. On the other hand, voice controls offer a more natural, hands-free interaction method that could be particularly useful for users with physical limitations or those who prefer verbal communication over gestures. Voice commands can also simplify processes such as repeating instructions or requesting additional information during the simulation, reducing the cognitive load on users and enhancing the overall experience. However, voice controls may not be as reliable or accurate as dropdown menus, as they can be affected by external noise, accents, or pronunciation variations.

Ultimately, combining these two interaction methods could provide the most versatile and user-friendly solution for the hospital simulation, allowing for both structured navigation and flexible, personalized communication options. Although the creation of a quickly accessible dropdown menu was successful and passed preliminary tests, incorporating voice recognition technology into this version of the system was abandoned. The adoption of this technology requires external software integration, as Unreal Engine currently does not natively support it. Developers are therefore required to spend significant amounts of time familiarizing themselves with third-party libraries or plugins that might present complex functionality. The main reason voice commands were not used in the system is that most approaches couldn't be used in standalone implementations and were unable to access the microphones on the Quest 2 headset.

3.7 « Development Timeline for System »

This section fully explores the discovery, development, and iteration process used in creating the framework software which is the focal point of the study. As Unreal Engine was selected as the development engine, a few software specific terms commonly used in the section are defined below for reader context:

- Blueprints: A visual scripting system within Unreal Engine that allows developers to create game logic using nodes and connections without the need for traditional programming languages.
- Component (WITHIN Blueprint context): A modular element within Blueprints that represents a reusable part of an Actor, allowing developers to add functionality or properties to the Actor by attaching predefined components.
- Pawns: Basic actor types in Unreal Engine that represent characters or objects with movement capabilities, commonly used for player-controlled characters and non-player characters (NPCs).
- Actor: A general object type in Unreal Engine, representing any interactive object that can be placed into a level, such as characters, items, or environmental elements.
- Class-based system: A programming approach in Unreal Engine that organizes game elements into classes and leverages inheritance and polymorphism for efficient code management and reusability.

- Teleport locomotion: A movement method in VR applications where the player's avatar is instantly transported to a new location, minimizing motion sickness by reducing the discrepancy between visual and vestibular perception.
- Slide Locomotion: A continuous movement method in virtual environments where the user's avatar moves smoothly along the surface of the environment based on the input from controllers or motion tracking, providing a natural sense of motion but potentially increasing the risk of motion sickness.
- Roomscale VR: A virtual reality experience that allows users to physically move around and interact with the environment within the confines of a pre-defined play area.
- Mesh: A 3D model composed of vertices, edges, and faces that define its shape and structure, commonly used to represent game objects and characters in Unreal Engine.
- NavMesh: A navigation mesh that represents the walkable areas of a level in Unreal Engine, enabling AI-controlled characters to navigate and avoid obstacles intelligently.
- Asset: A reusable piece of content in Unreal Engine, such as 3D models, textures, materials, or sounds, which can be imported, created, and managed within the engine's content browser.
- Materials: A set of properties and shaders in Unreal Engine that define the appearance and surface properties of a 3D object, controlling its color, reflectivity, transparency, and other visual attributes.
- Textures: 2D images or patterns applied to 3D models in Unreal Engine, providing details like color, roughness, or normal information to enhance the visual appearance of objects and characters in the virtual environment.

3.7.1 Experimental Apparatus

In the realm of high-fidelity game creation, there is often a preference towards high-powered gaming desktops. This research project was conducted using a combination of both desktop and mobile platforms - initially an Nvidia RTX 2060 mobile AMD Ryzen 4000 laptop, and later transitioning to the more powerful RTX 3070 mobile Ryzen 5800H laptop. However, a desktop machine, powered by a Ryzen 5900X and 3080Ti GPU, outperformed the mobile systems in terms of render times, and framework performance, emphasizing the advantage of increased hardware power.

The planning and assembly of the desktop system was influenced by both component availability and specifications. The system was designed with the demands of VR, AR, and XR applications in mind which mandate a system that can run the programs at a high, consistent framerate with minimal latency an develop UE software without long build times.

The central processing unit (CPU), an AMD Ryzen 5900X, was selected for its 12 core, 24 thread capabilities as a high core count drastically influences Unreal Engine's light rendering, shader compilation, project packaging, and source code compilation times. For use in UE VR development, a 5900X performs around 25% better than its competitor, the Intel i9 10900K. An Arctic Freezer 2 CPU cooler was used to avoid any chances of CPU throttling due to high temperatures, the cooler also has an onboard VRM fan to ensure stable power delivery, even with an overclock.

The graphics processing unit (GPU), an RTX 3080Ti, was carefully sourced amidst the global GPU shortage, owing to its critical role in delivering the high performance necessary for VR development. As mentioned before, rendering high-resolution visuals at a consistent, high framerate (90 to 120 frames per second) is a non-negotiable requirement for VR applications to minimize the chance of motion sickness, and Unreal Engine is particularly adept at leveraging the power of high-end GPUs to meet this demand. For reference, in UE 4.26, the 3080Ti touts a 16% frame increase compared to the second consideration, the 3080 and a 50% increase over the competing 6900 XT.

Figure 3-17 Motherboard with RAM Installed for Desktop Build

The motherboard, an MSI X570 gaming Pro carbon (Figure 3-17), with its compatibility with the Ryzen 5900X and its sufficient number of RAM slots, was an important part of the system. It also provided the necessary USB 3.0 ports for connecting the to the Quest 2 while including a built in Wi-Fi card with Wi-Fi 6 capabilities.

32GB of fast RAM was installed to handle the memory-intensive operations involved in creating complex environments like building lighting and working with high-resolution textures in VR development. To further enhance the system's responsiveness and load times, a Solid State Drive (SSD), a WD Black was used for storage. Finally, a 1200w power supply unit (PSU) was chosen to ensure reliable operation of all these components with high power draws.

The process of system assembly was followed by the installation of an operating system (OS). Although linux is a more efficient operating system that would have provided slightly better performance, the wide compatibility and easy usability of Windows made it the OS of choice.

Finally, with the build complete, Unreal Engine and other necessary VR development software such as Oculus Developer Hub (for Quest 2 compatibility) were installed. This entire process exemplified the journey of building such a system, from sourcing the components to the software setup. The mobile systems allowed for remote development and were still capable of rendering and displaying the simulation while leveraging the desktop's power whenever possible always sped up the process. This shows the frameworks applicability for those of all budget levels.

3.7.2 First Iteration Drop-in VR Hand

This subsection highlights the first attempt at implementing hand functionality in Unreal Engine, focusing on the differences in the older engine version used at this stage. Detailed instructions are minimized as much of the VR setup is explained in 3.7.2.

The primary objective at this stage was to use Unreal Engine 4.25 to recognize users' hands and enable proper finger articulation. UE 4.25 predates the VR functionality overhaul and starter stage present in subsequent versions. At this time, Epic Games had not yet released proper support for the built-in hand-tracking capabilities of the Quest line in Unreal Engine which required the developer to either download the Oculus fork (modified source code) of Unreal Engine 4 or use a third-party plugin. While this task may seem daunting to a beginner, a basic understanding of source code made the process manageable.

1. Download the Oculus fork of Unreal Engine 4.25 from the Oculus Developer center (fork deprecated an no longer available on the site).
2. Extract the files from the zip folder and ensure the necessary build tools are present (Visual Studio, the Windows 10 SDK, and the DirectX SDK)
3. Generate project files for the UE source using the GenerateProjectFiles.bat script, creating a Visual Studio solution file for building the engine from the source code.
4. Configure the build settings for the Oculus platform according to the Oculus documentation.
5. Add the "OculusHandTracking" plugin to your Unreal Engine project, either through the Epic Games Launcher or by manually adding the plugin to your project directory.
6. In your project settings, enable the "Oculus VR" plugin and the "Oculus Input" plugin.

With the fork installed, initial testing began using the default VR starter level and the base VR tracking components in UE, assessing whether a plug-and-play method could detect the hands. Although the Quest's hand-tracking features do not function within the Oculus Link environment, Unreal Engine can surprisingly stream the data over the connection used for display. This enables testing and iterating on the program without constantly repackaging the project for Android. After enabling the plugin, Unreal Engine detected the hands but treated them like controllers with no orientation detection. The hands would point in different directions (Figure 3-18), but the location of the hands was detected, and the movements were tracked in the X, Y, and Z planes. However, the virtual representations of the hands were flat and rigid, with no articulation.

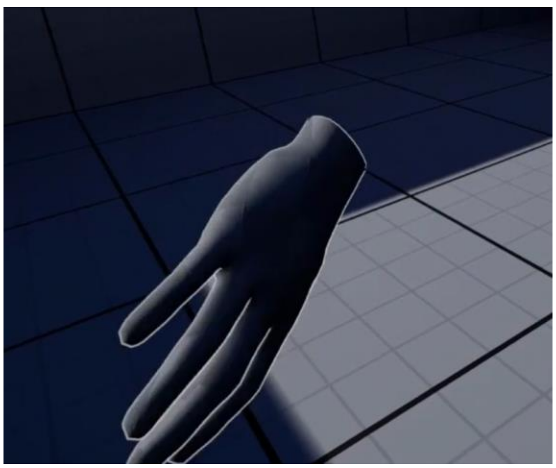


Figure 3-18 Hand with Incorrect Orientation

To enable proper hand orientation and articulation, additional steps were required:

1. Add the "OVRCameraRig" and "OVRHand" components to your VR pawn or camera rig blueprint.
2. Utilize the "OVRHand" component to retrieve and manipulate hand data in your project.
3. Generate or download a hand asset with the correct bone and socket mappings for full articulation.

Simultaneously, a shift from a single pawn-based system to a class-based system was decided upon for further development.

3.7.3 Second Iteration Class-based UE 4.26 Method

3.7.3.1 Importance of Classes

Adopting a class-based system is a widely accepted and advantageous approach to creating in-game characters. This organizational method provides developers with enhanced structure and increased ease of use in their workflow. One of the main benefits of using a class-based system for pawns is its inherent organizational structure. By creating different classes for pawns with varying capabilities, designers can focus on innovating gameplay experiences based on pre-built objects, streamlining their development process instead of rewriting code for each type of pawn functionality.

Leveraging the power of inheritance in classes, developers can simplify the management and readability of game code through a structured approach to various pawn types. By separating each pawn type into its own class, standard functionalities shared between different pawns can be defined in a base class and then inherited by derived classes. This avoids code duplication within the actor blueprint and promotes distinct responsibilities, leading to improved ease of system maintenance. The essentiality of a class-based system guarantees the simulation's long-term operability and ease of upgrade path. An inexperienced user can inherit from a base pawn class with specific movement controls and attach additional functionality as needed. Changes made to one type of child pawn do not affect the others, making it a more viable option. If developers desire to modify base functionalities for all pawns, they can simply alter the base pawn class, and the changes will propagate throughout the system. The advantages of adopting this system for VR development include the ability to quickly create new customized pawns with specific and unique functionalities optimized for individual VR experiences, such as advanced interaction systems, movement and navigation, or sensory input handling. With inheritance and polymorphism, various VR pawns created in Unreal Engine can share identical essential functionalities such as head tracking, hand tracking, locomotion, or collision detection. Unreal Engine already uses a form of this system; any time a developer makes a new blueprint, the engine will ask which class to base it off. The class-based system enables developers to encapsulate data and methods in a single unit, which also aids in debugging. This programming approach offers significant advantages for VR applications.

3.7.3.2 UE 4.26 Development

The second iteration of the VR hand-tracking pawn was still coded manually. However, development had shifted to Unreal Engine version 4.26, which introduced numerous quality-of-life improvements for developers working with the platform. In comparison to Unreal Engine 4.25, which required the use of the Oculus SDK and manual updates to the engine's source code for hand tracking on the Quest platform, UE 4.26 made significant strides in simplifying the development process.

The integration of the Unreal HeadMountedDisplay plugin in UE 4.26 natively supported Oculus hand-tracking technology, streamlining the implementation process and minimizing the need for engine reconfigurations. Developers could now easily enable hand tracking in their applications by adding a "Hand Tracking" component to the player controller blueprint and configuring the plugin settings within the Unreal Editor.

The inclusion of this plugin greatly reduced the complexities associated with integrating Oculus hand tracking and eliminated compatibility issues arising from clashes between different platforms or software versions. Furthermore, the streamlined process meant that developers no longer needed to modify the source code, making application development more accessible than ever before.

To utilize the new hand-tracking components, developers first needed to modify some project settings to enable the project to run on the Quest and to configure the hands as controllers within the engine. Key settings to be changed included disabling mobile HDR and configuring the project to work on Android and the Quest platform (Figure 3-19). Additionally, developers needed to adjust the OculusVR plugin settings to set hand tracking support to "hands only” (Figure 3-20).

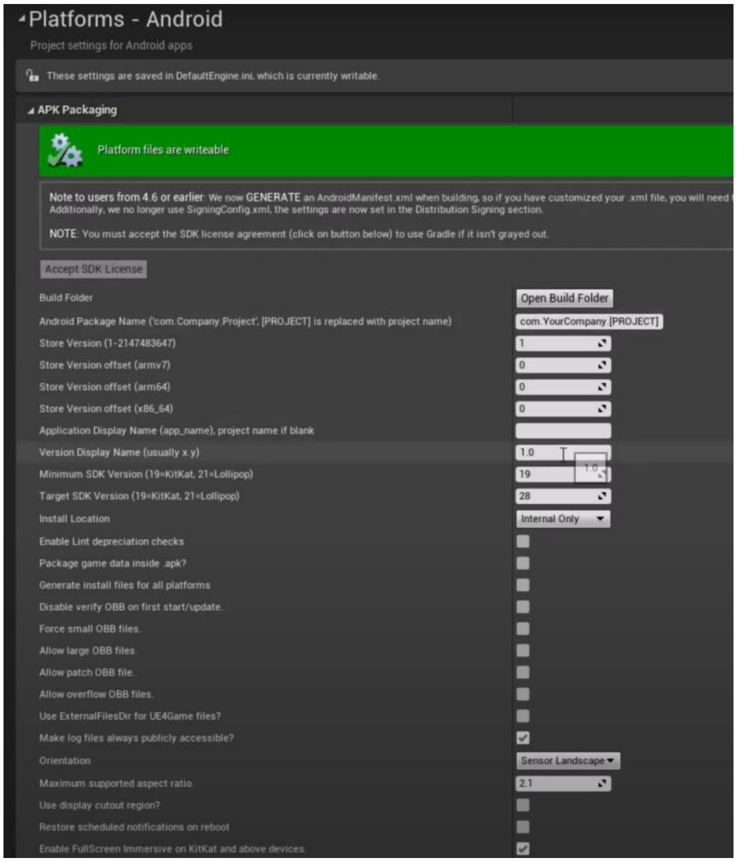


Figure 3-19 Platform Settings to Modify for Android Compatibility

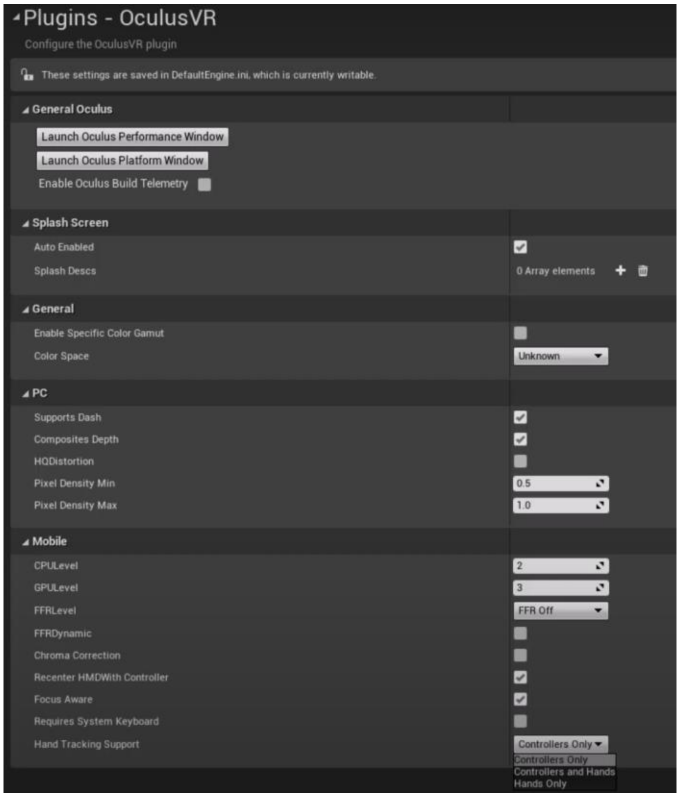


Figure 3-20 Settings to Modify in the OculusVR Plugin

After configuring these settings, developers could create a new blueprint class, selecting the pawn class category from the popup (Figure 3-21), and including a camera component for a user viewport and motion controller components for each hand to treat them as controllers. The process involved adding an Oculus hand component to each motion controller component to drive the location and functionality of each hand (Figure 3-22). Next, add a “set the tracking origin” node to fire on begin play as in Figure 3-23, this function sets the height and location parameters for the VRPawn. Place the new VRPawn in the map and delete the default pawn. At this point, it’s possible to test the hand-tracking fidelity. This method resulted in an immediate improvement over the previous one. The hands were placed overtop of the developer’s physical hands and enabled finger articulation as the Oculus hand component already has a default hand asset with the appropriate bones mapped out (Figure 3-24). There is the option to create a custom hand model, but the default asset already has the bones mapped, saving a large amount of unnecessary effort. It is also possible to buy ready to use hand assets through the marketplace or design using the tools discussed in the previous subsection.

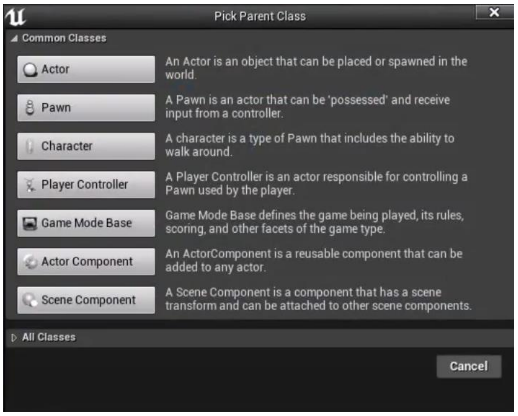


Figure 3-21 Parent Class Selection for Blueprint Creation

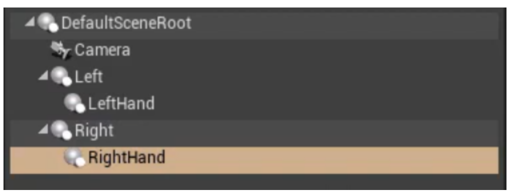


Figure 3-22 Component Tree for Pawn Blueprint

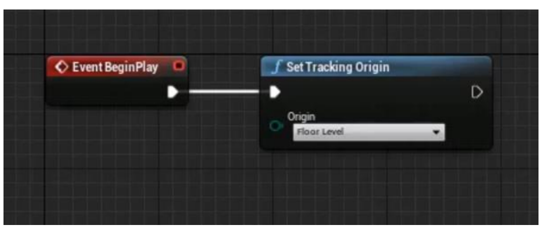


Figure 3-23 Set Tracking Origin as Fist Action in Pawn

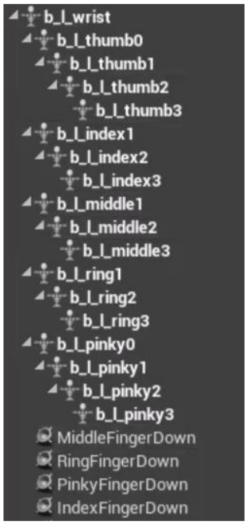


Figure 3-24 Oculus Hand Asset Bone Default Mappings

With the base detection function working, the developer could now implement gesture recognition. Create a class blueprint, use the search to find the "OculusHandComponent" and use it as the parent instead of pawn. In the main event graph, and following Figure 3-25, create variables for each finger's socket, each finger's bone (only 4), a value for threshold and a Boolean for each finger to hold its status as closed or open. To avoid repetition, create a new function with a memorable name to check if a finger is in the up or down position. The function should take the inputs of bone name and socket name; within the function, connect the built-in "get bone location by name" node and "get socket location" nodes to the inputs and follow the structure in Figure 3-26 below.

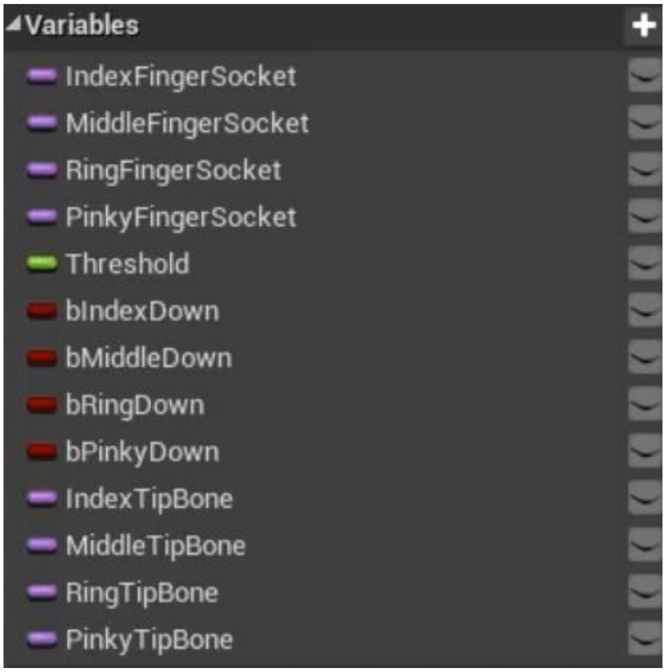


Figure 3-25 Variables Needed for Custom Gesture Detection

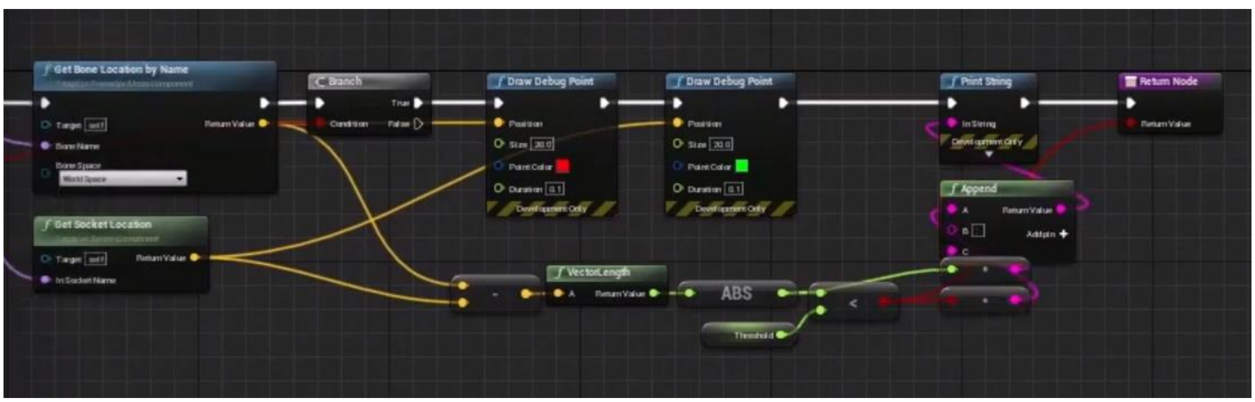


Figure 3-26 "Is Finger Down?" Function Event Graph for Custom Hand Component

The next step involved connecting the new function to a “sequence four” block linked to the tick event (Figure 3-27). The function checked if the finger bone and socket passed the activation threshold. If so, the status was set to close; otherwise, it was set to open. Repeating these steps for each finger (Figure 3-28) provided a rudimentary gesture detection system that continuously updated each finger's status as up or down. Developers then created a variable to view the hand gesture's previous frame status and an event dispatcher as a flag for detecting a teleport gesture.

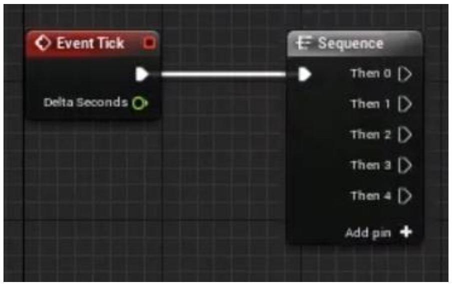


Figure 3-27 Sequence Block in the Event Graph

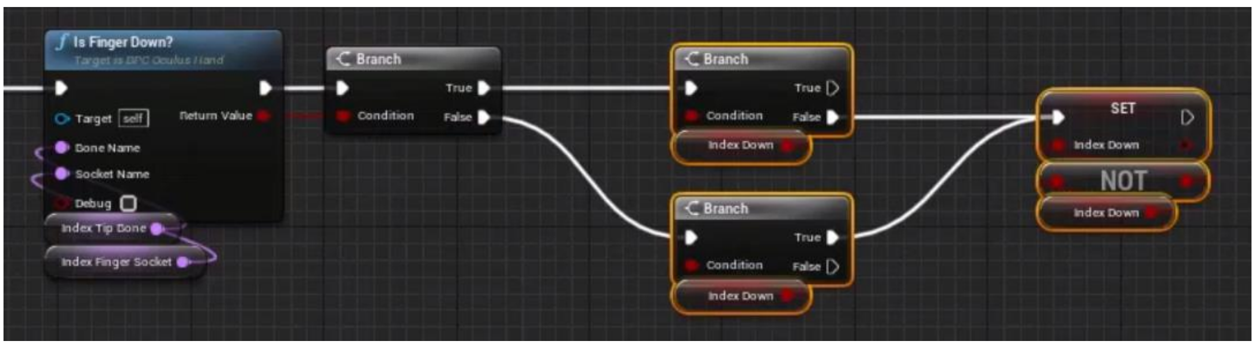


Figure 3-28 Sequence for Index Finger Status

With finger status recorded, the developer added functionality to detect the teleport gesture. The gesture has 2 stages, aiming and confirmation. The program first checked if all fingers were closed except for the open index finger, as shown in Figure 3-29. If confirmed, the status was set to pointing. Next, the program checked if all fingers were down as the confirmation gesture and compared it to the variable monitoring the previous gesture (detecting a pointing gesture). If the pointing gesture was undetected, the program continued as the user was just making a fist. If the previous gesture was detected as pointing, the user was intentionally attempting to teleport. The program then called the event dispatcher made to trigger a teleport before setting the pointing variable to false, as seen in Figure 3-30. If neither of these combinations of finger patterns is detected, the pointing variable remains false anyways.

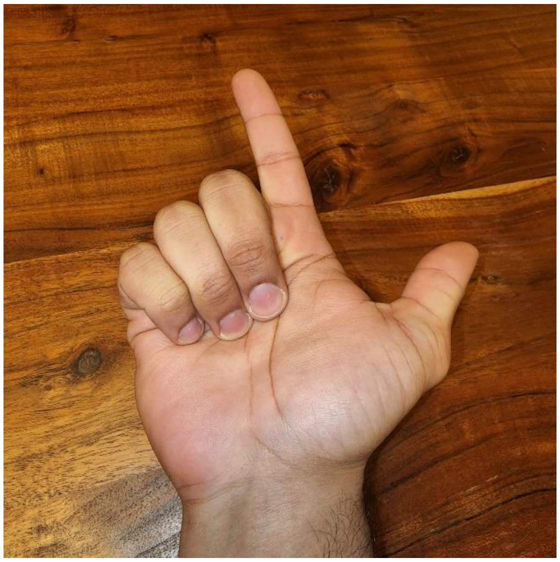


Figure 3-29 First Gesture for Detection

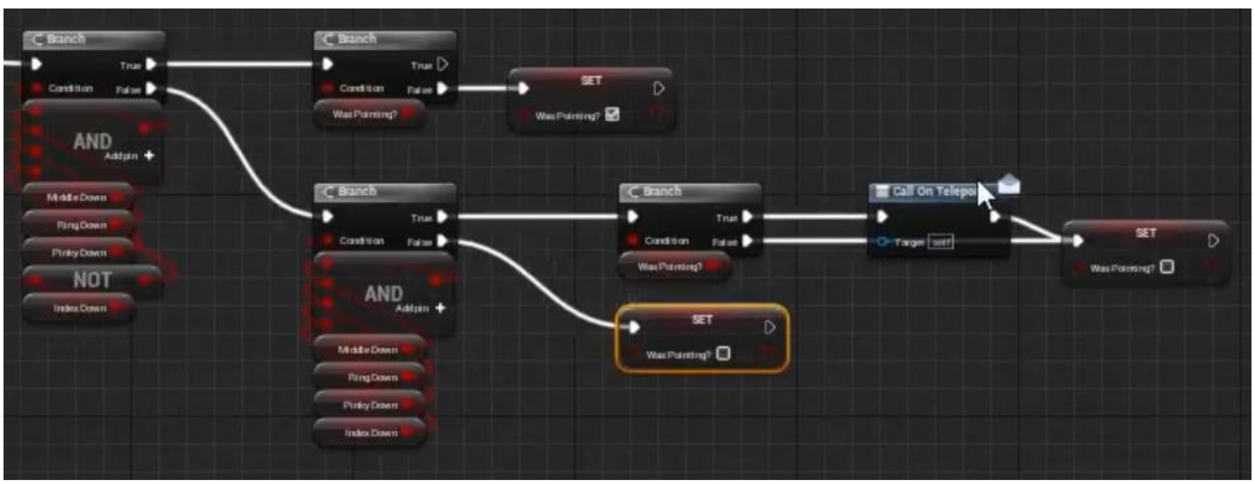


Figure 3-30 Teleport Gesture Detection Event Graph

Returning to the pawn blueprint, the developer replaced the built-in Oculus hand components with the newly created one and assigned the mesh, bone mappings, and properties from the default one. To properly implement the teleport function, the mechanic used for the stock motion controller for VR is used as a reference. Immediately after setting the origin, a bind event block bound to the teleport event is placed with the target input being the newly created hand component (Figure 3-31). Next, they created a

Teleport custom event on the system graph, which the binding block called upon. After the Teleport event was triggered, it was compared to a new variable checking if the user selected a valid teleportation "landing" point. If the event was found to be true, the teleport function was called upon. A new vector variable holding the "teleportPoint" location and a "GetActorRotation" block served as inputs to the teleport block.

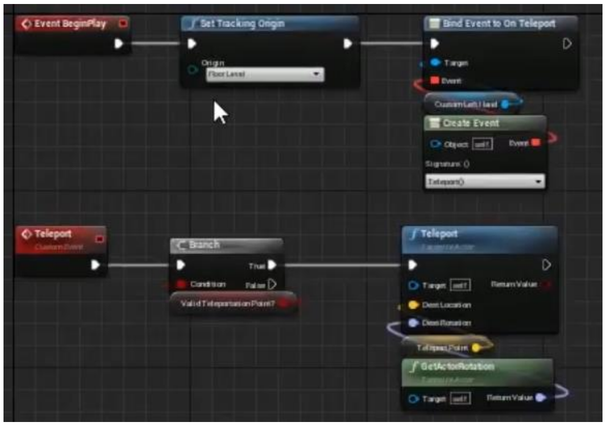


Figure 3-31 Event Graph to Trigger Teleport Within Pawn

The developer then added an arc spline and arc direction component as children to the custom hand component within the constructor script to visualize the teleport points. Within the constructor, they placed a "get socket Location" block, which obtained the hand direction from the "laserBeam" socket in the skeletal mesh provided with the hand asset (Figure 3-32).

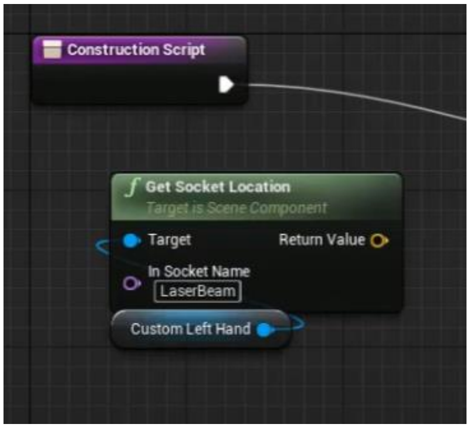


Figure 3-32 Get Socket Location from Hand Component Within Constructor Script

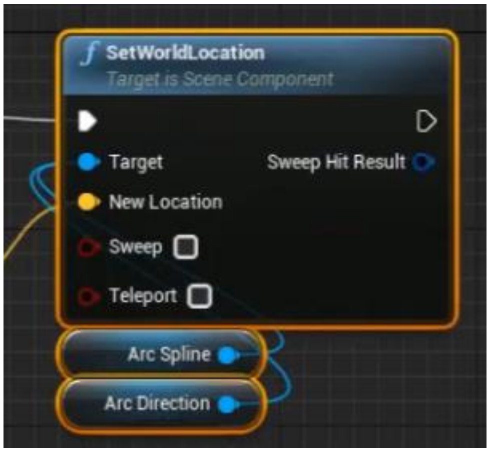


Figure 3-33 Setting the World Location Based on Arc Spline from Socket

A "setworldlocation" block with the arc spline and arc direction components as target and the "TeleportLocation" variable as New Location was added (Figure 3-33). To check user intent to teleport, a branch was added to the tick event, which compared the "get up vector" block against the palm vector of the hand and checked its orientation via dot product (Figure 3-34). The dot product was checked against a >0.7 threshold to allow for some forgiveness in detection. If the branch returned false, the boolean for a valid location for teleport was set to false; if true, it connected to a predicted projectile block to measure the arc (Figure 3-35).

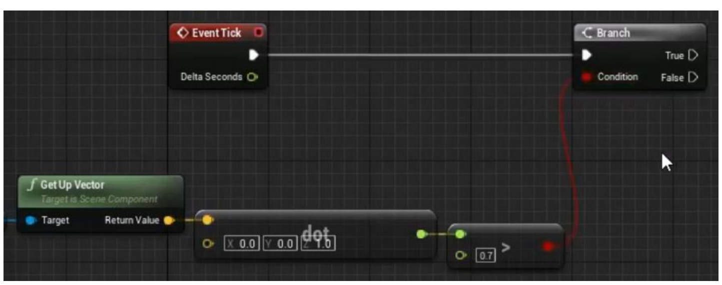


Figure 3-34 Check for Hand Orientation

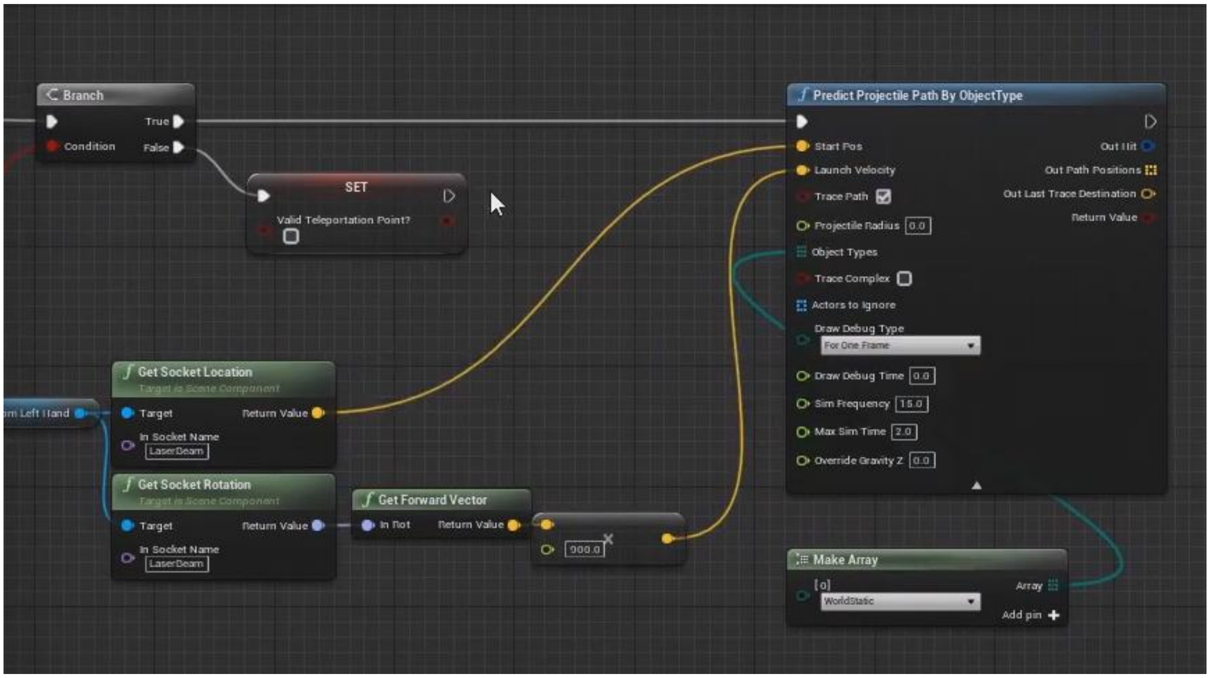


Figure 3-35 Connections to Predict Projectile Block

A make array block fed a WorldStatic array to the new projection block, while the start position and launch velocity were derived from the laserbeam socket of the hand component. To check if the projectile hit something, a "break hit result" node was introduced to obtain the hit location. Next, the location output from the break is output to a "projectpointtonavigation" node with the "Query Extent" set to 500. It also took the navigation system as an input using the "Get Navigation System" node. If the point was on the navmesh, the "valid teleportlocation" boolean was set to true; otherwise, it was set to false. Immediately after the boolean was set to true, a "set teleportation point" node was used, grabbing the location from the projection landing location for the spline (Figure 3-36).

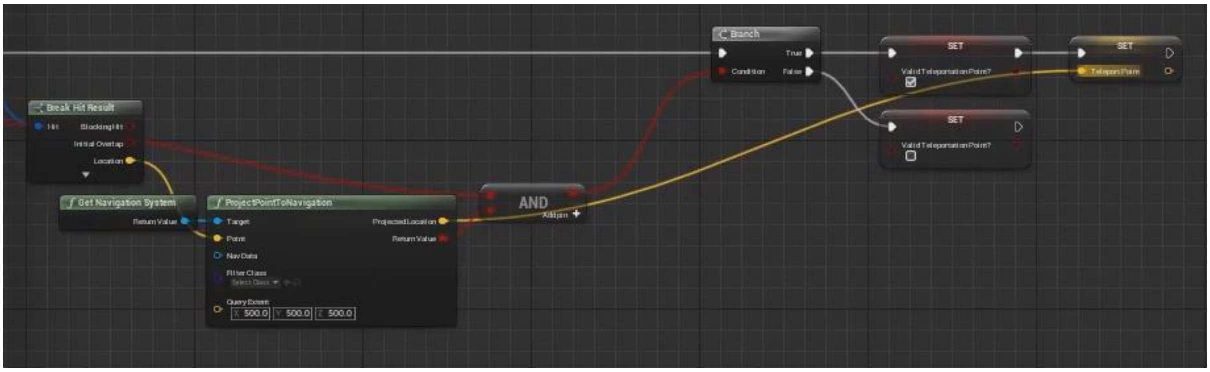


Figure 3-36 Event Chain from Break hit to Setting Teleport Point

Lastly, a constructor was created to draw the arc and clear the arc after a teleport was performed. With this in place, developers could test the teleport system. If the palm was facing up and the gesture requirements were met, the VR pawn would teleport to the location indicated by the projected spline. This code layout was not developed specifically for this project; instead, it utilized a modified version of how the base motion controllers handle teleport based on a button press.

This implementation of hand tracking and teleportation worked well and functioned adequately in the demo level. However, it was abandoned for the final flow once the project was upgraded to UE version 4.27, as it required extensive setup and would be challenging for an inexperienced programmer to modify with new gestures.

Unreal Engine 4.27 brought significant improvements and functionality changes to VR development and Oculus hand-tracking technology. Notably, built-in hand-tracking components eliminated the need for cumbersome manual configuration by developers. The "Hand Actor" component simplified the process of enabling Oculus hand tracking in projects, allowing developers to drag and drop it into their scene and adjust settings such as hand size, pose, and gesture recognition for better calibration and support. Version 4.27 also added native finger articulation support, enabling individual fingers of virtual hands to move and bend more naturally, enhancing immersion and realism. Although functional in 4.26, the articulation was less precise and required more background work. Unreal Engine 4.27 updated the built-in APIs and introduced a new VR template with a redesigned VR navigation and starter level, which included new features such as a VR movement system. Changes in the update broke the previously developed teleport function, necessitating structural modifications and library updates to function in the new environment.

3.7.4 Third Iteration Plugin Class -based Method

Considering the impact of engine updates on the framework, the developer decided to use a plugin called VR Hand Tracking from the Unreal marketplace for handling interaction functionality. This approach eliminated the need for constant repairs after updates, as the plugin developer managed potential engine conflicts. This would also make the system easier to maintain for researchers, clinicians and hospital staff. Although this entailed starting the interaction engine from scratch, this step was taken to optimize workflow efficiency since the developer no longer needs to worry about debugging hand detection. While the new VR resources and the plugin help hasten the setup process, some settings must still be changed to bring everything together. In the plugins menu, as the VR hand tracking is built off the OculusVR system, it is essential to go and disable all OpenXR plugins to avoid conflicts while enabling the OculusVR plugin and the new VR Hand Tracking plugin.

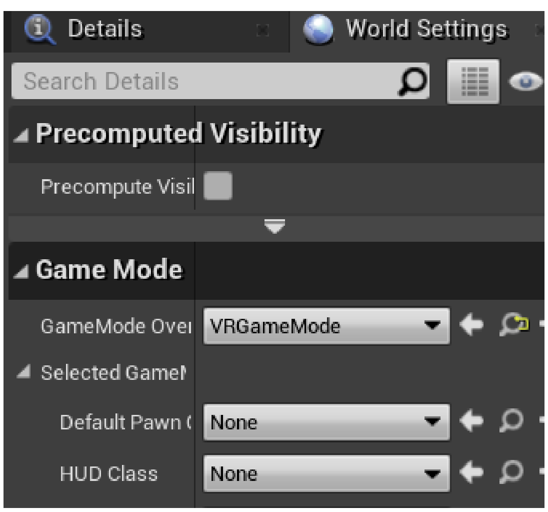


Figure 3-37 GameMode Override in World Settings

The class-based approach began with creating a new class blueprint pawn to hold the base functionality. The next step, introduced with update 4.27, involved creating a new game mode in the world settings panel (Figure 3-37) and setting the default pawn to the custom VRPawn. The new pawn event graph included setting the origin, and adding a Camera component, and two motion controller components for the left and right hands. The HT Hand Tracking component, derived from the new plugin, was then added to the motion controllers (LeftHand and RightHand in the Figure 3-38).

This new component built off the original Oculus Hands component inherits all its original settings and has some new functionality built in. Implementing teleportation in this new system was simpler and only required adding the HTPointLocomotion component to the actor and selecting the hand-tracking component for the desired hand (left or right).

Upon testing, users saw an arc from their fingertip and a circle on the nav mesh floor when performing the appropriate gesture (Figure 3-39). When the user made the beckoning gesture, the pawn then teleported with the same fade animation.

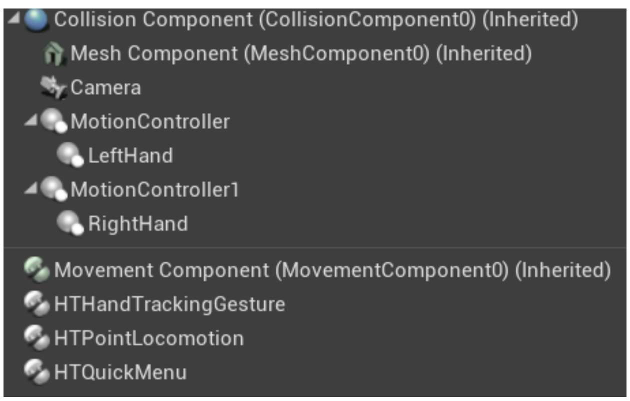


Figure 3-38 Component Tree for New Pawn Blueprint

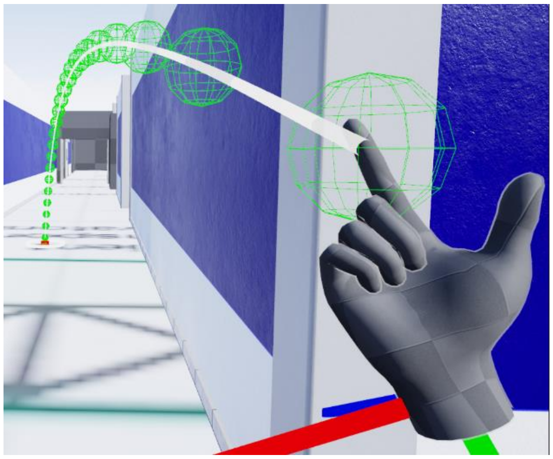


Figure 3-39 Example of the Developer View for Teleporting with New System

Although the plugin streamlined hand tracking implementation, an update temporarily broke the teleport functionality. The plugin developer (who is very active with support) provided support and quickly resolved the issue in the next update. The update allowed developers to view and adjust finger up and down values and thresholds, enabling better control over teleportation gesture sensitivity. To accommodate users unable to keep their hands/arms completely still during gestures, the beam's speed was decreased to minimize jitter. For room-scale VR, resetting the actor's position on each "teleport" event ensured the virtual camera remained centered.

The “HTHandTrackingGesture” (Figure 3-38) eliminated the need for manual coding of gestures (in section 3.7.2) based on variables by providing functionality for recording and saving gestures to use as function triggers. The gesture component also has a gesture threshold like the manual version to quantify how close the hand shape must be to the recorded shape; the higher the value, the more forgiving the detection. This functionality streamlined gesture building and the recording function now built into the framework lends to more simple future expansion and increased adaptability.

Adding the component to the pawn class and setting the parent motion controller began the gesture implementation process. Blank "hand-tracking gesture" assets were created in the project content window to save the recorded gestures. A keyboard trigger (Figure 3-40) triggered the save gesture function. It was crucial to select the target hand gesture asset for saving to or risk overwriting your previous gesture. Also, a good practice is to check the "Mirror hand gesture" box so the gesture can be used for either hand. Once the program has run and a gesture has been recorded, enter the pawn blueprint once again, select the gesture component and navigate to the gesture settings tab. It is now possible to link recorded gesture start and finishes with events so the events now fire each phase change.

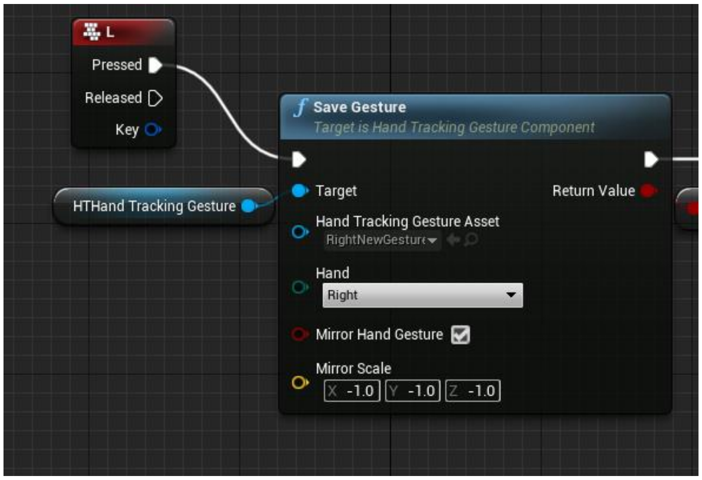


Figure 3-40 Scripting to save Hand Gestures

In the demo program developed using this framework, four distinct gestures were utilized:

- Teleport – A two part gesture where the user sticks out their right hand with the palm side up with 3 of 4 fingers folder inwards to select the location. And the second where the user makes a becoming gesture to move to the location of the target (Figure 3-41 and Figure 3-42).
- Spiderman – This gesture is used to trigger the slide movement. This gesture resulted in some trouble and will be changed or simplified in the future. With the palm facing upwards fold in your middle two fingers (Figure 3-43).
- Hang ten – this gesture is used in the MRI simulation to signal the user has sat down and is ready to resync their virtual location to their physical. Facing upwards, this time fold in the index, middle, and ring fingers (Figure 3-44).

- Peace – the final gesture in the MRI simulation, this gesture I used to signal the user has laid down and is ready for the MRI to begin the process (Figure 3-45).

Figure 3-41 Teleport Gesture Phase 1

Figure 3-42 Teleport Gesture Phase 2

Figure 3-43 Spiderman (Sliding) Gesture

Figure 3-44 Hang Ten Gesture

Figure 3-45 Peace Gesture

To ensure maximum visibility from the Quest 2's inside-out cameras, it was recommended to perform gestures approximately 30 degrees above the horizon.

Additional functionality developed with plugin integration but unused in the pilot demo included:

- Grab and Pickup item Manipulation System
  - Designed for user-object interaction and door handle manipulation, it was removed for performance improvement and demo simplification. To enable, the developer adds the component ability to the pawn while using the "pickup" actor blueprints for physics enabled objects.
- Slider functionality
  - Originally designed for the quick menu to control slide movement speed, it was removed for simplification with an automatic acceleration curve instead. Programmed in by adding a slider element to the quick menu and linking it to the speed value in a slide enabled pawn.
- Floating widget

    - These interactable widgets were developed for survey questions regarding user preferences on locomotion methods, it was removed to streamline and shorten the demo. Survey questions were moved to a post-trial survey.
- Quick menu.
    - Created for offhand use to access toggleable settings (like the speed slider) and a dev menu for teleporting between environments, this menu required creation of an actor (asset) with the desired buttons. Menu spawning could be achieved through the gesture system or by adding the Quick Menu component to the pawn actor.

3.7.5 Environment V1: BSP Prisms and Layout Data Collection

The development procedure for creating the environment underwent several iterations, but the starting point remained the same. Due to the COVID-19 pandemic, the developer had no access to the hospital for data collection, mapping, scale measurement, or image capture, which greatly compounded the difficulty of the task at hand. Using basic smartphone footage provided by the project supervisor, the first iteration of the environment took shape.

This version was designed as a scale test for interaction and movement mechanics but was never completed with textures or detail items, such as doors in doorways or objects found on site. To create this basic environment, the developer, new to Unreal, used the Binary Space Partitioning (BSP) brush tool to create hallways with long rectangles. The height of the hallways was estimated based on standard hospital door measurements found online, compared to the door-to-ceiling distance in the captured footage. The rectangles were given the estimated height, then stretched to the length estimation made by audibly counting the supervisor's steps in the video. A second, subtractive rectangle was placed inside each hallway to hollow out the prism (Figure 3-46). This iteration was not very intuitive but was helpful for comparing the length of the virtual halls to the recorded footage in real-time by flying through in VR.

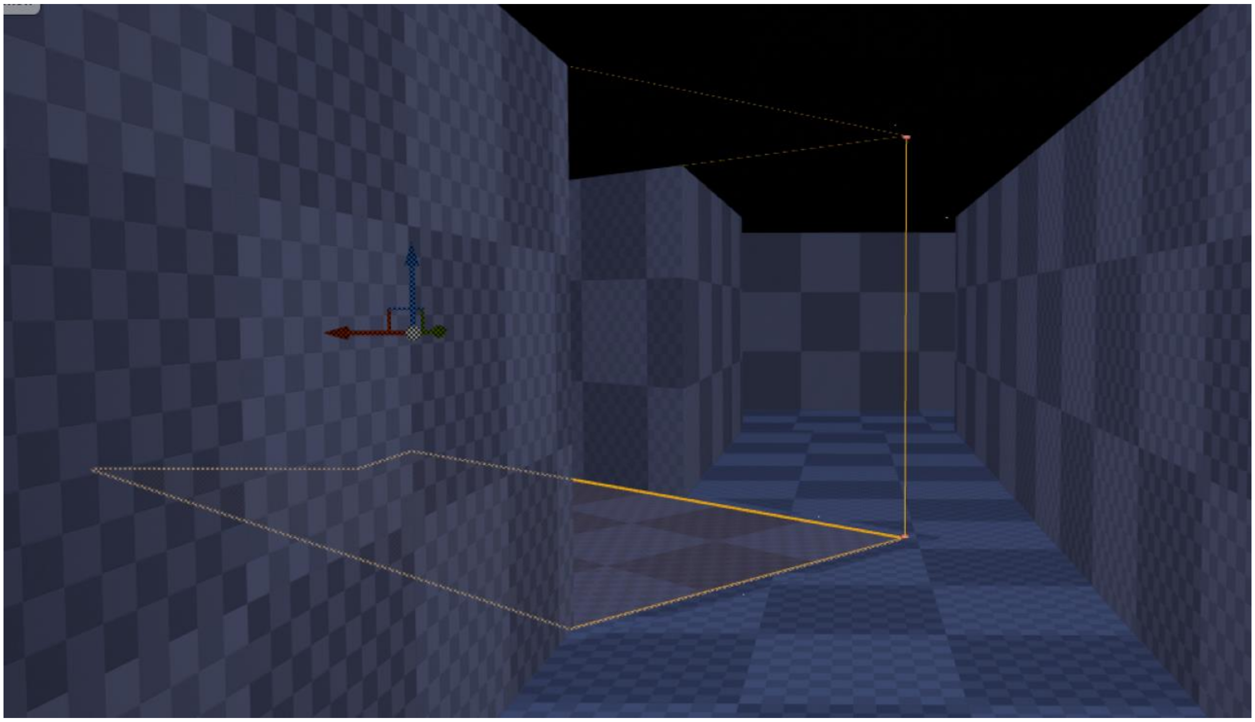


Figure 3-46 BSP Blockout Using Prisms

3.7.6 Environment Version Two BSP Walls (Scale test)

The second iteration of the environment was more accurately scaled, thanks to additional environment footage and a relative map sketch based on the step count of an in-hospital colleague. Although more accurate than audibly counting steps from the video, this method still left some room for inaccuracies, especially when certain portions of hospitals had unique designs (recessed doorways, different hall widths, and shapes) that could be difficult to capture in a freehanded map. Nevertheless, having a map allowed for a completely different development approach.

To provide building scale, a flat plane was placed in the engine, and the colleague's map was set as a texture overtop, creating a marked-out floorplan when viewed in VR. BSP blocks were used again to create components such as walls, ceilings, doorways, and space-consuming items in the environment, rather than full hallways (Figure 3-47 and Figure 3-48). Subtractive rectangles were still used to hollow out certain sections, but these were for cut-outs of doorways and windows rather than the full halls. These walls were created over the map plane and, even without textures, provided a more realistic experience than the previous iteration, as it had some hall shapes from the actual buildings. Common practice dictates sticking to simple shapes when building with BSP block-out brushes. However, to match the recorded video and get a proper sense of scale, walls were shifted off the lines, and details such as door arches, elevator recesses, and strips of BSP rectangles along the walls to emulate the railings at the hospital were added. This was done to provide a better sense of scale, as handrails are a similar height to door handles.

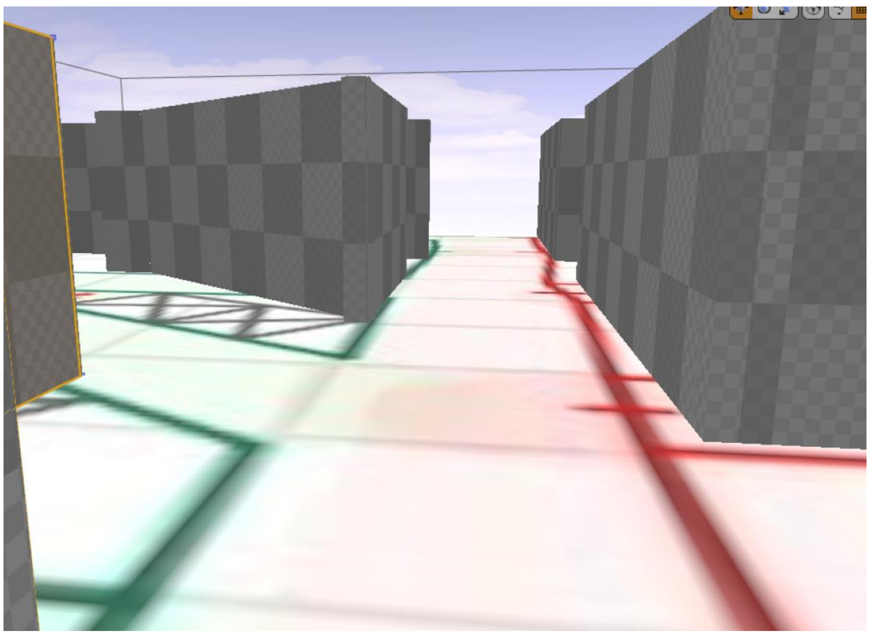


Figure 3-47 BSP Blockout Walls

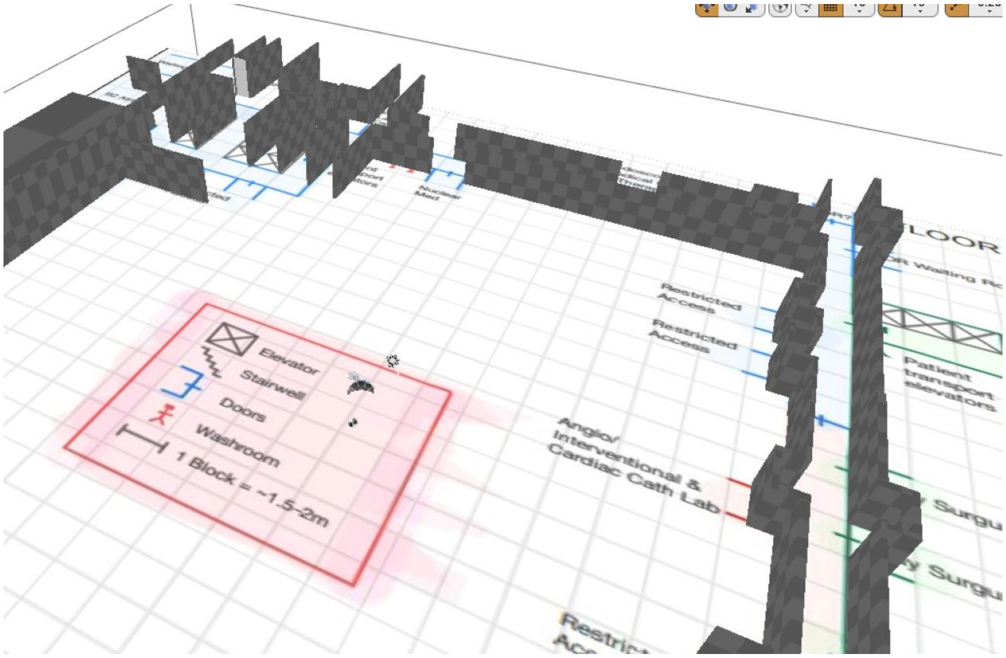


Figure 3-48 BSP Blockout Walls on Floorplan

3.7.7 Full Environment Development Approach (addition of several smaller rooms)

Following the second iteration of the environment, the next logical step was to develop realistic building elements and more detailed assets for the hospital simulation. Blender was crucial in creating small rooms complete with necessary objects such as cabinets, doors, tables, and everyday hospital items like the one shown in Figure 3-49 and Figure 3-50.

Figure 3-49 Virtual (left) and Physical (Right) PCCU

Figure 3-50 Virtual (left) and Physical (Right) PCCU Wash Station

When creating a room and objects in Blender, most graphic designers use several photos or videos as reference points. It's crucial to start by defining the shapes, structures, and details seen in the reference material accurately. For detailed work, Blender has a wide variety of sculpting and modeling tools that aren't present in the game engine. This means that assets or environments must leave Blender in a nearly finished state aside from lighting. Once this is done, UV unwrapping comes next before applying textures and materials to give them the intended visual appearance. However, optimizing these assets for real-time usage is critical; this requires measures like reducing polygon count and consolidating materials to minimize draw calls. Following the optimization process, assets are exported as Filmbox (FBX) files. The newly formatted assets can then be conveniently imported into the engine where they undergo further modifications to create a cohesive environment that integrates various game mechanics, interactivity elements, and other features crucial for delivering an immersive virtual experience. In this case, the environments often had some materials replaced with UE-specific ones to increase realism. The lab group was incredibly helpful in creating some of the Blender pieces and rooms. However, this approach was slow and time-consuming for developing large-scale environments.

3.7.8 Modular Development Approach (large env creation)

To address the issue of larger environments, a modular approach using 3DS Max was implemented which entailed designing modular wall pieces, floor tiles, doorways, and ceiling tiles of standard sizes. These pieces, in conjunction with seamless textures allow the developer to construct and modify the virtual environment within Unreal Engine. It's worth noting that this approach can also be accomplished using Blender.

The initial step involves determining the specific dimensions and fundamental shapes of the various building elements. To ensure modularity, a set of standardized measurements are established to permit all components to fit together seamlessly in a grid-like form. In this use case the modular pieces were built to a standard size of 1.25 meters by 2.5 meters (Figure 3-51). This grid size meant no wall pieces needed to be stacked on each other and square tiles were simply modelled as half height. Subsequently, individual components are modeled, adding details and refining geometry while also keeping modularity in mind, aiming towards perfect unity upon assembly. To ensure seamless snapping and alignment in Unreal Engine, it is imperative to establish consistent pivot points for all elements or risk texture scaling problems. Same as in Blender, the modeling process should be followed by optimization procedures suited for real-time usage, which include the reduction of polygon counts and the maintenance of clean geometry. Once optimization is achieved, the individual pieces are exported as FBX files as well. By incorporating these modular assets into the Unreal Engine, developers can create large-scale environments with ease, while ensuring a cohesive and immersive experience. This method streamlines the development process and offers the flexibility to adapt and modify the environment as needed. Although these modular pieces can also be made in Blender, the developer used 3DS Max to show the usage of two among the most popular 3D modelling tools.

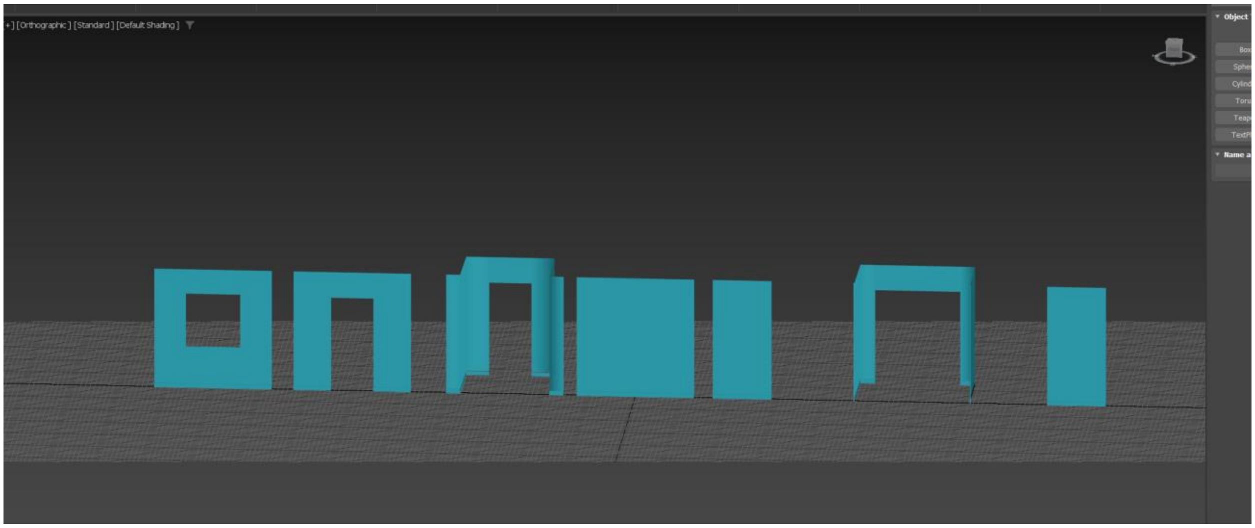


Figure 3-51 Modular Components Made in 3DS Max

3.7.9 Importing assets to Unreal Engine and optimization

Upon importing assets to Unreal Engine (UE), developers can refine them further by adjusting or utilizing them in various configurations to meet the requirements of interactivity and game mechanics. To achieve a cohesive visual appearance and consistency among modular components, UE is employed for texturing, material assignment, and proper lighting of the scene. This process necessitates careful planning during the design phase to ensure materials align with the desired aesthetic standards and that lighting conditions are realistic and conducive to user immersion. To achieve this, a combination of point light and spotlight elements illuminated the environment effectively, striking a balance between ambient and direct lighting while adding depth and emphasis to specific areas. Although the modular approach simplified design, it remained time-consuming due to the lack of an accurate sense of scale for the real environment. As a result, hallway lengths and other architectural elements often required resizing upon VR review, necessitating corresponding adjustments for other components to maintain an immersive environment.

Instead of relying on pre-existing BSP block-out components, assets were designed and imported from 3DS Max to optimize the construction process and facilitate asset reuse. The UE marketplace was instrumental when unique or unusual assets were needed, such as gurneys, wet floor signs, water fountains, fire emergency elements, elevator doors, and irregularly shaped walls. These items were purchased on the marketplace to expedite the design process. Online resources and pictures were invaluable for sourcing textures to create new materials, and this stage saw significant use of Photopea. To ensure proper scene illumination and maintain a realistic environment, careful attention was given to lighting placement, intensity, and color temperature. Varying levels of detail (LODs) were implemented to ensure seamless VR experiences across different hardware tiers.

Figure 3-52 First Floor Transition to Main Area (Physical)

Figure 3-53 First Floor Transition to Main Area (Digital)

Figure 3-54 Second Floor Zone D Elevator Intersection (Physical)

Figure 3-55 Second Floor Zone D Elevator Intersection (Digital)

The modular development process was employed to construct the first and second floor levels of Victoria Hospital, as shown in the accompanying figures (Figure 3-52, Figure 3-53, Figure 3-54, Figure 3-55). Although several other hospital rooms were modeled in Blender for potential inclusion in the system, they remained unused in the feasibility demo. These environments, such as CT scanner rooms, consultation rooms, waiting rooms, operation studios, checkup rooms, and PCCUs, are available for future expansion and trials for healthcare purposes.

The initial modeling of the MRI room (Figure 3-56) was performed in Blender before being segmented for exportation to UE. This approach granted greater flexibility in customizing object interactions (e.g., MRI tray interactions) and facilitated the replacement of materials for elements with conflicting UV Unwraps in UE. The Event Dispatcher system streamlined communication between the pawn and world blueprint, enabling world events to be triggered based on flags raised by the VR Pawn actors. Event dispatchers were employed in general trials to switch between stages, initiate recording events, and, in the MRI simulation, to allow a world sequence interaction to be triggered by a hand gesture.

Figure 3-56 MRI Demo Room

Ultimately, the environment evolved from an initial, rudimentary BSP block-out architecture into a sophisticated one, utilizing a combination of Blender and 3DS Max techniques and some pre-fabricated assets to enhance realism. Despite limitations imposed by the COVID-19 pandemic, the developer successfully created an immersive and accurate representation of the hospital environment without ever visiting the physical location.

3.7.10 Integration of hand tracking and user interactions

It was crucial to verify the functionality of both the generated environment and the interaction/movement functions together, ensuring natural movement and immersive interactions. This subsection summarizes development features implemented after completing the environment. The developer conducted preliminary testing with themselves and close colleagues to gather initial feedback on the integration of both systems. Positive feedback allowed for further expansion of the mechanics and levels mentioned earlier.

The teleport function mechanics development were described above, but only the gesture triggers and a high level overview of the slide motion was mentioned. The “Spiderman” gesture served as a trigger for the Move HMDDirection function (Figure 3-57).

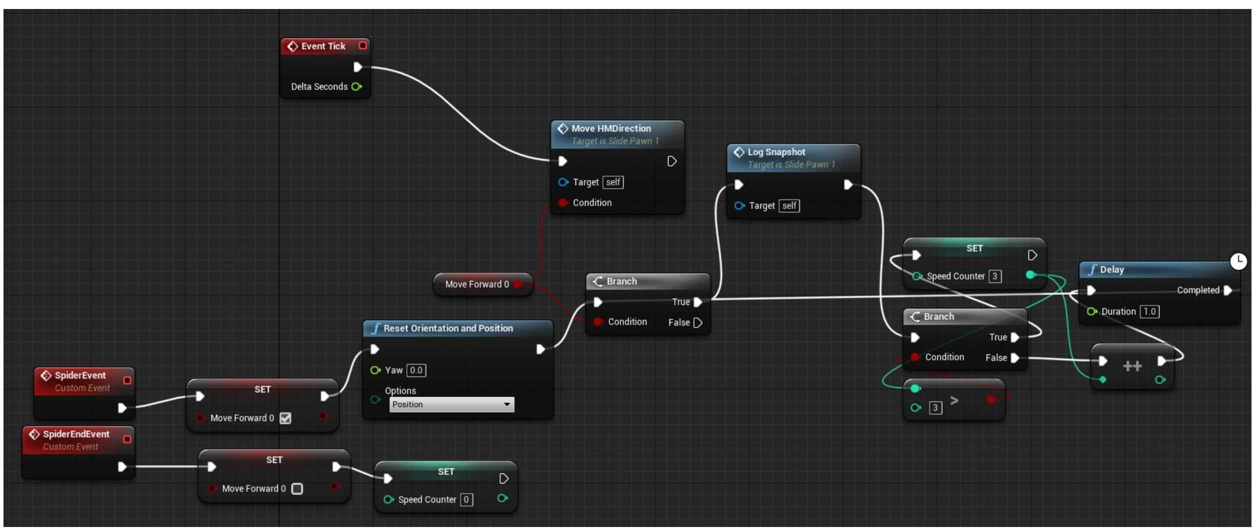


Figure 3-57 Event Graph Leading to Move HMDirection

To synchronize movement with the program tick, the gesture served as an indirect trigger. Forming the gesture set the "Move Forward" Boolean to true, while releasing the gesture set it to false. This activated the movement function (Figure 3-58), the data logging function, and the counter representing acceleration increase.

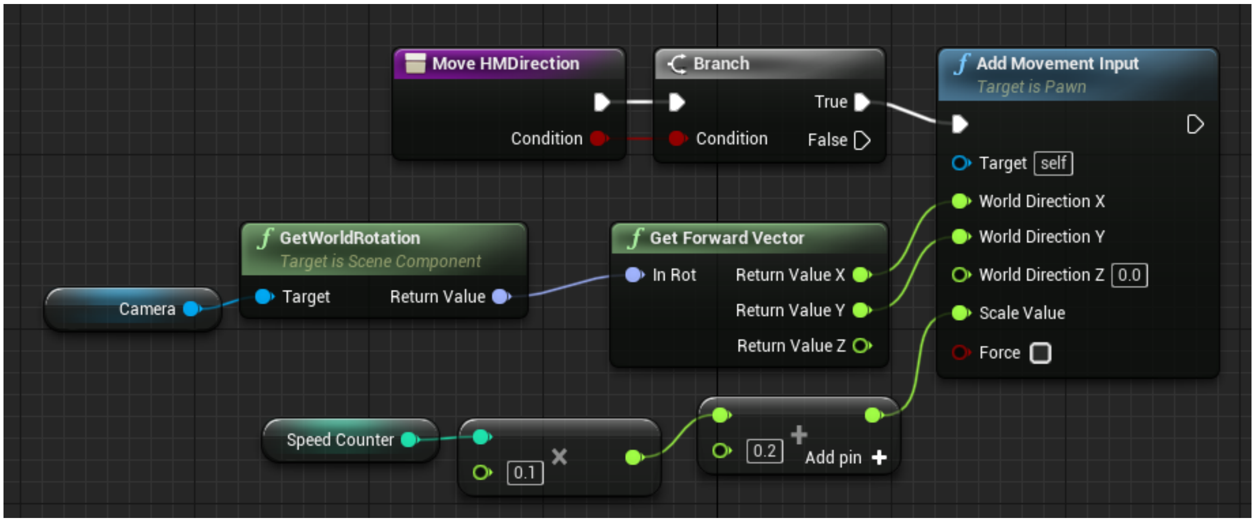


Figure 3-58 Move HMDirection Function

A primary focus during the pilot demo development was the MRI simulation, aiming to provide users with an experience as close as possible to an actual MRI without additional haptic equipment. In line with this objective, the demo required a location for the subject to lie down. As there is no way to synchronize real-world objects with virtual ones in UE during setup or runtime, a creative solution was needed to incorporate the physical bed while maintaining immersion. Using unique gestures to trigger world movement, the developer designed a function that re-adjusted the virtual world view and re-aligned the user's virtual body to its place on the MRI tray (Figure 3-59) once the patient was seated at the edge of their surface (or bed). This function was triggered using the hang ten gesture once the user had sat down. As the user lay down on their bed, their virtual body lay down on the MRI tray, and they used another gesture (the peace gesture) to indicate they were ready for the MRI experience to begin. The virtual body then slid up on the tray as the lights dimmed and the patient's head entered the MRI machine. After a few seconds, the user heard accurate MRI machine noises circling their head, simulating the real-life experience.

Figure 3-59 MRI Machine

A key objective of the simulation was the ability to design individual hospital experiences or create programs with user tasks and data collection, as demonstrated in the testing methodologies section. To vocally relay tasks to the user, the software utilized a disembodied voice resembling a PA system. Task instructions were written out and read using a Text-to-Speech (TTS) engine, which recorded them as sound clips. These clips were inserted into the program and triggered by hand gestures, task completion (through event dispatchers), or on stage load. The TTS engine was chosen over manual recordings to maintain consistency, ensuring uniformity regardless of when recordings were made. The chosen TTS engine for this program was Natural Reader on Android.

3.7.11 Data Collection and Modularity

To record performance metrics for healthcare or environment testing purposes, the system needed a way to collect and export player data. The "DirectExcel" plugin addressed this requirement, offering several useful functions to simplify and enhance the project's modularity.

Initially, the program's I/O capabilities were coded from scratch using a custom C++ class with String to File and File to String functions. The C++ code for string I/O functionality is shown in Figure 3-60 below.

```
#include "ExtRWFuncLibrary.h"


FString ExtRWFuncLibrary::LoadFileToString(FString Filename)
{
    FString directory = FPaths: : GameSourceDir();
    FString result;
    IPlatformFile& file = FPlatformFileManager::Get().GetPlatformFile();
    //IFileManager& file = IFileManager::Get();


    if (file.CreateDirectory (*directory))
        //if (file.DirectoryExists(*directory))
    {
        FString myFile = directory + "/" + Filename;
        FFileHelper::LoadFileToString(result, *myFile);
    }
    return result;
}


FString ExtRWFuncLibrary::SaveStringToFile(FString Filename, FString Data)
```

Figure 3-60 ExtFuncLibrary Code

However, the final iteration of the system used the plugin instead, as it better handled Excel output. A function (Figure 3-61, Figure 3-62, Figure 3-63, Figure 3-64) called "save player data" was “written” within the base pawn class, taking the player pawn as an input along with the test subject's name. This function first checked for an existing workbook with the subject's name; if none was found, the program created and saved one. If a current file existed, the function logged several details, including a time and date stamp, the last method of locomotion used, the XY location of the player pawn, and the XYZ gaze direction. An iteration of this function also saved other data, such as interaction and distance covered since the last log, but these were omitted for simplicity. This function was called every time a teleport was triggered and, for slide and joystick movement, every time the trigger was detected, each second it was held, and once it was no longer detected. As base functionality, the framework logged a timestamp every time a pawn spawned into a level, a user teleported, or the pawn moved using the slide method for a second. This can be built upon as demonstrated in the demo program.

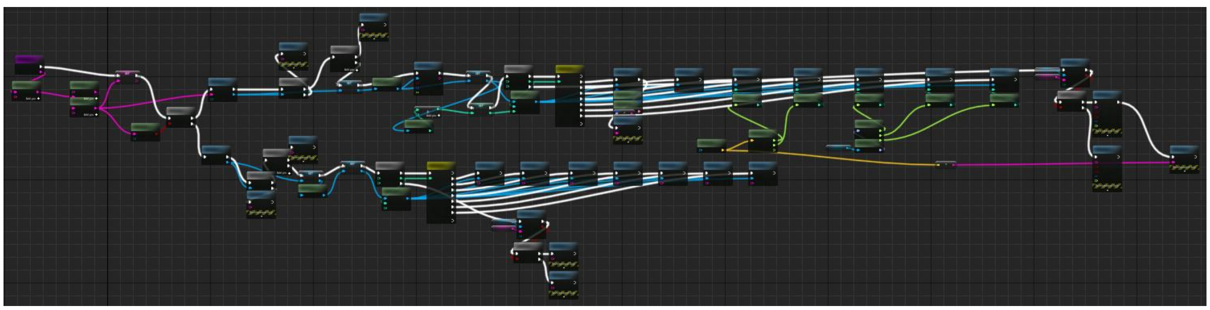


Figure 3-61 Save Player Data Function

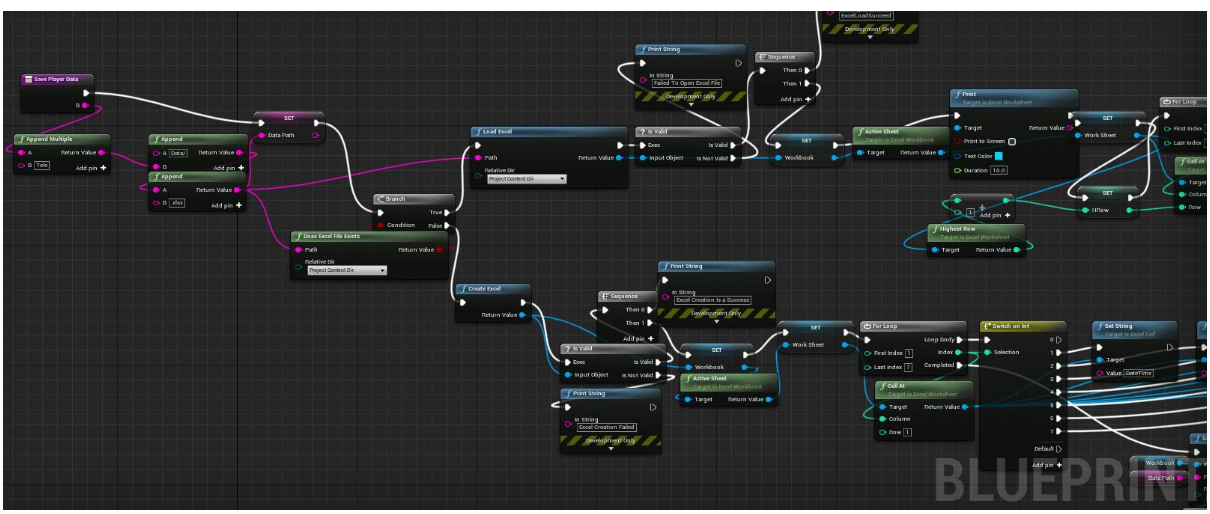


Figure 3-62 Save Player Data Function (Part 1/3)

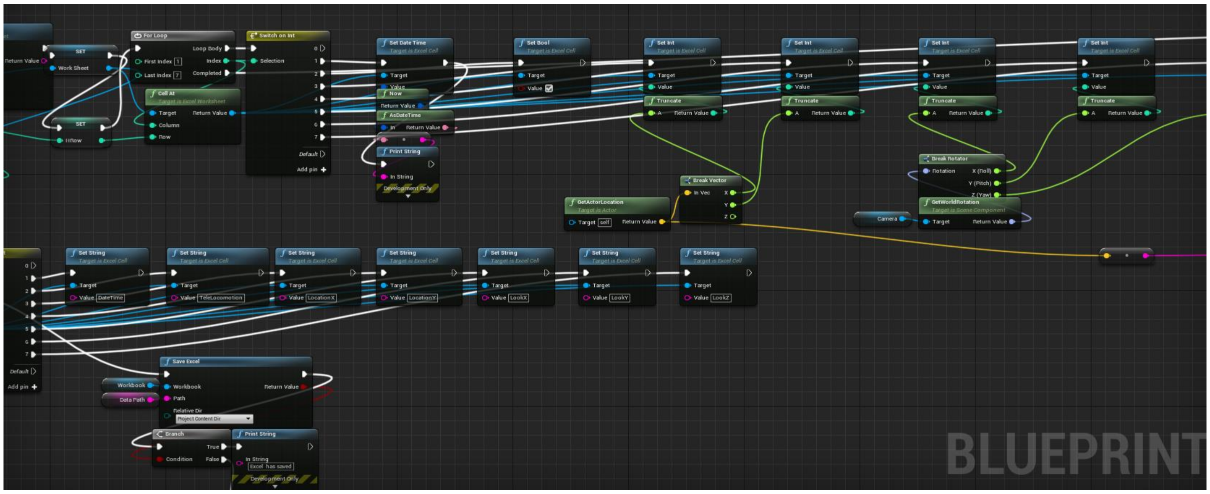


Figure 3-63 Save Player Data Function (Part 2/3)

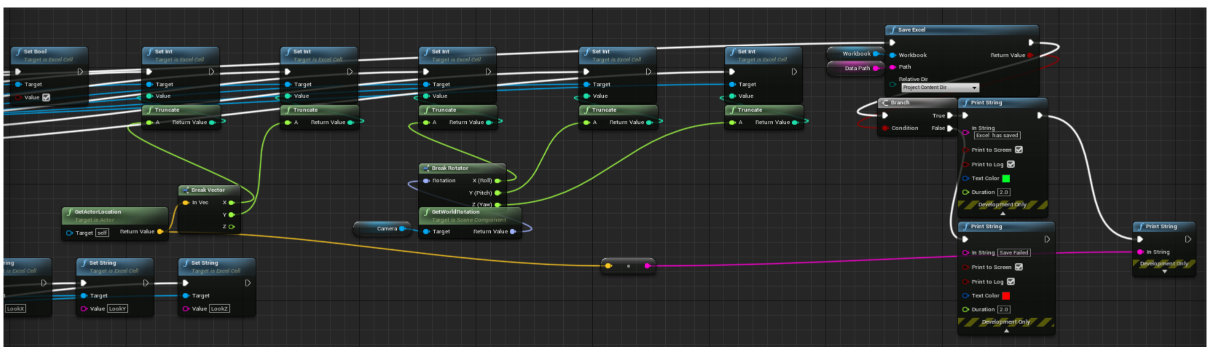


Figure 3-64 Save Player Data Function (Part 3/3)

The current framework simplifies the creation of an objective-driven VR experience using it as a starting point. A new developer can make a new level and select certain parts of Victoria Hospital to use or build their own environment using the modular pieces or the workflow shown in Figure 3-65 below. The workflow also demonstrates the next step for new interaction functionality. With a level created, the user can choose which pawn to use for VR hands functionality or create their own pawn class that inherits the base functionality of one of the pre-built ones. If the user has their own model, the framework's modularity allows for easy integration with minor adjustments, such as collisions, lighting, and navigation mesh.

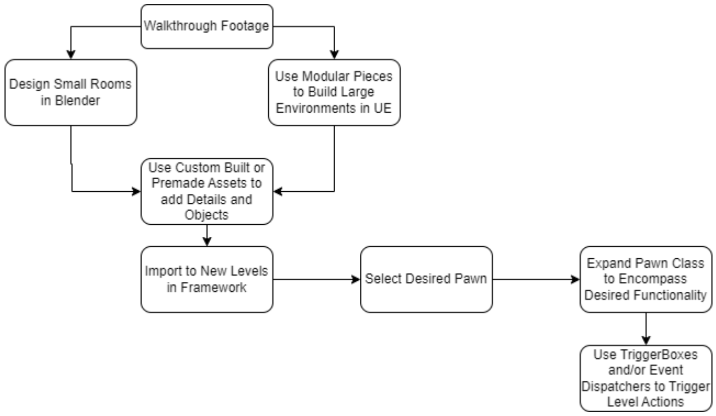


Figure 3-65 Workflow for Framework Environment Expansion

3.8 « Addressing Motion Sickness Concerns »

A significant concern when developing VR simulations is the potential for inducing motion sickness. Motion sickness is a common user concern and can considerably impact the overall experience and, in extreme cases, render the simulation unusable (Conner et al., 2022). Motion sickness arises from discrepancies between the visual and vestibular systems and can be intensified by factors such as virtual environment movement while the user is stationary, inconsistent frame rates, and latency spikes (Kawamura & Kijima, 2016). Frame rate, latency, and lag are critical factors for user experience; a high, consistent frame rate ensures smooth visuals, while low latency guarantees that users' actions are quickly and accurately reflected in the virtual environment, reducing differences in virtual and physical head pose (Palmisano et al., 2020). Inconsistencies in frame rate and latency or lag spikes introduce a mismatch between user motion and display output, leading to motion sickness.

To address this issue, optimizing locomotion methods and ensuring a smooth, responsive virtual environment is essential. By doing so, developers can minimize motion sickness risks and provide a more comfortable, enjoyable user experience. Teleportation is the hypothesized optimal locomotion method to prevent motion sickness, as it moves the character pawn to a predetermined location with a fade-in/fade-out transition, giving the sensation that the virtual body does not move. Slide and joystick movements differ, as their nature conflicts with the virtual body remaining stationary.

Another step in reducing motion sickness (simulator sickness) involves ensuring the simulation environment does not become too resource-intensive for the host device and keeping the subject focused on specific tasks (Jasper et al., 2020). If the visual fidelity exceeds the device's capabilities, latency will significantly increase, causing freezing and skipping.

Chapter 4

4 « Experimental Methodology »

Evaluating the feasibility of the simulation system necessitates understanding what constitutes a satisfactory virtual simulation experience. To determine if the created system meets the standards of a satisfactory experience, three main objectives were devised, guiding the development of the feasibility demo. The demo incorporated some, but not all features, mechanics, and environments created throughout the project. Each objective was linked to evaluating the demo's ability to meet it through a combination of both objective and subjective metrics. Test subjects were then asked to experience the demo, which taught them the necessary gestures, gave them two separate tasks with multiple steps, followed by an optional experience and a mandatory feedback survey.

4.1 « Demo Objectives »

The primary objective of the feasibility demo was to provide users with a natural, immersive environment that facilitates learning, testing, and comparing distinct movement methods, gestures, or interactions with virtual objects.

The secondary objective is to effectively promote users' ability to familiarise themselves with and navigate through environments that may be entirely foreign to them. An extension of this objective for testing in future studies is that once users are comfortable with the environment layout in virtual space, they can visit the location the simulation emulates (in this demo, a portion of Victoria Hospital's first floor, the Atrium, connected hallways, and certain rooms on the second floor) and navigate the physical location in the real world.

As the drive behind selecting this specific environment as a simulation was to alleviate preoperative or pre-visit anxiety, the tertiary objective is to reduce apprehension or unease by providing the user with a comfortable simulation of the environment and procedures they may be experiencing for the first time. The idea is that reducing these factors(unease) would result in fewer adverse clinical, behavioural, or psychological outcomes such as separation anxiety or psychologically triggered post-operative pain.

Although the results of anxiety reduction in preoperative patients after experiencing the VR simulation fall outside the scope of this study, it remains essential to evaluate the pilot demo to assess the framework's potential to provide a comfortable experience that positively impact users and does not heighten stress or cause further discomfort such as motion sickness.

4.2 « Procedure and Task Description »

A few iterations of the task list were created for the feasibility demo, balancing comprehensiveness and brevity to match the average system user's comfortable VR session duration. The first proposed iteration was rejected by the supervisors due to an excessive number of tasks and experiences where the original intent was for users to test every feature and environment made for the simulation and experience every clinical scenario before completing the post-demo survey in VR. The second iteration reduced the number of tasks as the goal of the study was no longer to specifically evaluate user capabilities or every individual simulated experience, but to test the ability to apply the framework in a clinical trial-like situation from a researcher's perspective. In this scenario, the demo serves as the researcher's tool for user data capture. The third iteration refined the existing demo structure by removing the need for users to enter their information or state their movement style preference in VR, adding an external survey at the end to obtain users' relevant past experiences and impressions of the feasibility demo.

The final iteration of the demo began with the test giver providing a quick explanation of the two gestures used in the demo. The system does explain the gestures at the beginning, however, it was concluded that this step was needed as some users needed live feedback on their specific hand pose to correct their finger or gesture position. This subject is further discussed in the "Results and Discussion" section. After the gestures were explained, the user stepped into the guardian area of the quest headset and placed the headset on their head. At this stage, some users required assistance with headband and IPD adjustment. If the subject chose to experience the optional MRI simulation, the test giver must remember to include the surface on which the subject will lie within the guardian area. For perspective, the final demo lasted approximately 15 minutes, depending on user performance and their decision to do the optional demo component.

When the demo begins in earnest, a message plays, introducing the user to the demonstration, followed by a quick rundown of what they can expect throughout their experience. Before proceeding to the first trial, the user is taught and shown the movement gestures in VR and required to try them out. First the slide gesture, then the teleport gesture with each needing to be successfully detected before moving on. The demo then describes the first trial, the movement method test, where the user is spawned on the first floor of Victoria Hospital and is told to follow the green arrowed path to the box at the goal. They are informed the test will be done twice, once with the slide locomotion system, then once with the teleport system, with each run being timed. Whenever the user is ready to proceed, they touch the countertop on which the images of the path and gestures are displayed. On contact, they are respawned inside a box on the first floor. In both tests, additional timestamps (to the built in pawn triggers) are recorded when the users leave the boundary of the start zone. In the first run, when the user reaches the box, a message congratulates them on completing the task and informs them that the teleport trial is next before respawning them. Then on the second completion, the user is moved back to the room in which they started for the next set of instructions.

The program then informs the patient that the next test will take place on the 2nd floor of the hospital and their goal will be to navigate to the locations the program audibly indicates. They are notified that the test will be done thrice and that maps are placed around any location where they see a colour-coordinated compass. The order of the goals was initially randomized, but to get more stable measures on improvement, the order was set as the PCCU, then the OR, and finally to the EEG/EKG. In this trial, the users can use any locomotion method they wish, and the program will log whichever is used. Between each test, they are sent back to the starting area to receive the next location. This was done so the user could begin the timer when ready and provide a break if needed. Rather than goal boxes like the first test, the users grab the respective rooms' doorhandle to establish that they have reached their goal. This was done to provide the user with a more natural form of HCI.

When the user has returned to the starting area for the last time, they are asked by the voice if they would like to experience the MRI simulation. They are informed that they can exit by grabbing one of the oranges on the table nearby or continue to the MRI experience by touching the counter again. The orange as an exit method was used as a novel method that the user has a lower chance of touching by accident. In an earlier revision, the bed in the room was used, but some preliminary testers touched it during the locomotion gesture training portion and accidentally closed the program. The oranges are out of the way and will now only provide an exit if both trials have been completed.

The optional MRI experience is a quick demo that showcases the developed system's ability to simulate clinical scenarios and the laydown mechanic discussed earlier. The user is spawned into the MRI room and is informed they are free to explore the room using either movement method introduced in the other portions of the demo. When finished looking around, the user is told to navigate to the couch, bed, or table (any raised surface they are willing to lay down on), sit on the edge, and show the program the designated gesture to sync the environment (in this case the orientation and location of the MRI patient table) with the selected physical surface. For this, the user was asked to use the hang ten gesture (make a fist and stick out both the thumb and the pinky). Once the virtual environment matches the direction of the physical one, the patient is instructed to lie down and show the camera a peace symbol when ready for the MRI to begin. At that point, the room's lights dim simultaneously as the table slides into the MRI machine. Once the table has slid into place, the demo simulates the audio from an MRI machine around the patient's head. After the demo is done, the tray slides back out of the machine, the lights brighten, and the user can walk to either door, which will take them to the starting area, at which point all parts of the demo are complete, and the user can explore the room or leave the simulation to move on to the post demo survey.

4.3 « Metrics for Evaluation »

To effectively evaluate the usability and effectiveness of a VR system, a comprehensive approach to testing and data collection is necessary. In the VR hospital simulation demo, volunteers were recruited to perform navigational tasks within the virtual environment. By collecting timestamped location and gaze data and user interactions, such as button presses and doorknob touches, developers can assess different mobility methods' performance and identify areas for improvement. This data collection helps measure the demo's success in fulfilling the first two objectives. The feasibility demo first employs timestamping to compare two examples of different locomotion methods, providing a quantitative metric for faster user navigation. When combined with subjective comfort questions from user feedback, it is possible to tailor VR experiences to individual users and for researchers to draw conclusions on which form of locomotion provides a better experience. The third objective is measured using a survey that users complete after the demo. Although surveys provide only subjective feedback, there is no objective way to measure user comfort or motion sickness except if a user is too uncomfortable to complete the entire demo in one sitting. Even in these extreme cases, and experience which triggers motion sickness for some may not in others. To maximize valuable data gathered from the survey, it includes various question types, such as multiple choice, semantic differential scales, Likert scales, and open-answer questions.

{LISTED QUESTIONS WITH RESPONSE TYPES}

Personal Information

1. First Name? *
2. Last Name *
3. Email Address
4. Which age category do you fall into? -> Multiple choice: <12, 13-17, 18-21, 22-23, 30-55, >55
5. How much past experience do you have with VR? -> Semantic differential scale: 1-5
6. How much past experience do you have with any Video Games? -> Semantic differential scale: 1-5
7. How familiar are you with Victoria Hospital? -> Semantic differential scale: 1-5
8. I felt comfortable using the VR system. -> Likert Scale: 1-7
9. I found the tasks to be too difficult because I am unfamiliar with that hospital. -> Likert Scale: 1-7
10. The VR environment allowed me to explore new parts of the hospital. -> Likert Scale: 1-7

11. I felt some motion sickness or dizziness using the system. -> Likert Scale: 1-7
12. I found the views and progression through the demo to be confusing. -> Likert Scale: 1-7
13. I found the gestures needed in the demo to be confusing or difficult to use. -> Likert Scale: 1-7
14. I have the following comment for what is bad about the system: -> Open response
15. I have the following comment for what is good about the system: -> Open response
16. I have an idea for improving the system: -> Open response
17. I have an idea for what can be removed from the system as unnecessary: -> Open response
18. If you felt any motion sickness during the demo, at what point did you feel sick? -> Open response

After presenting the first iteration of the survey to research supervisors, they suggested revisions to shorten the survey, eliminate redundant questions, and improve user experience. One such recommended change involved modifying a question from asking the user to enter their age to instead providing multiple choices for age ranges, addressing concerns about users' willingness to share their exact age. Each survey question is designed to collect data for measuring the system's success in fulfilling each testing objective. Combinations of answers between questions allow for drawing specific conclusions, which are further discussed in the result and discussion sections.

Question Rationale:

- Question 4 – to observe how differing age groups' performances vary or how different ranges have different learning rates for the gestures and environments.
- Questions 5, 6, 7 – to observe how prior experience with VR, video games, and the real Victoria Hospital translate to simulation performance.
- Questions 8 – 13 – to receive quantitative feedback on the user experience during the demo.

- Questions 14, 15 - to provide specific feedback for anything users did or did not enjoy during the demo. These two questions are essential for future development as the feedback form this can carry over to framework components not included in the demo.
- Questions 16, 17 – to provide specific feedback on what a user would add to or remove from the system. Although these seem similar to the last two, the subtle difference of asking them to make a change provides further insights for future improvements.
- Question 18 – to provide direct feedback on which part of the demo, if any, caused motion sickness.

As the workflow/system was developed from an engineering mindset, it is imperative to acknowledge that this can result in implementing some less user-friendly mechanics. Programmers can sometimes become biased towards their developed system or immune to any flaws causing physical discomfort because they have experienced it several times. However, in the end, success is dictated by the user and the data collected. Therefore, the open-ended questions are important, especially the final one as it questions the user on one of the critical issues with VR experiences, motion sickness. More than the weight of the headset, cable tethering, or poor control methods, motion sickness can immediately end one's experience, resulting in the user falling over or being too sick to continue. The question pairs with number 11 of the survey to allow the users to more precisely indicate what part of the demo made them feel the most ill and to what extent.

4.4 « Testing Demographics »

When selecting the testing demographic, it was essential to include people with varying experience levels in VR, video games, and familiarity with the real-world site modeled (Victoria Hospital). Accounting for a limited sample size, the testing criteria included 6 test subjects under 30, 6 test subjects over 30 (Question 4), ideally with at least 4 of the test subjects having experience at the physical location (Question 7). Beyond these minimum requirements, the testing aimed to include users with varied prior VR and video game experience levels (Questions 5 and 6). Given the goal of demonstrating framework potential and the difficulty obtaining subjects at the time of testing, this criteria was discussed with project supervisors and was sufficient for the proof of concept demo. In future work, it would be beneficial to test the demo on a larger sample size and to test if the navigation translates to the real world by running equivalent navigation trials in the physical hospital for more detailed conclusions.

4.5 « Data Collection and Analysis Methods »

The collected objective and subjective feedback helps refine the VR experience and ensure it meets users' needs across various applications. The data collected includes users' positions, time, interactions within the environment, and movement method selections. Additionally, gaze position and timestamped user X and Y data are recorded to determine the user's location on the map and the object they are looking at.

The objective and subjective feedback collected by the overall demo is crucial for refining the VR experience, to meet users' needs and further adapt to various applications. As stated, using the built in framework functions, data was amassed including position, time, movement method, and gaze direction within the environment.

Gaze position and timestamped user X and Y data were recorded identify the users' location within the environment as well as their focus point.

By evaluating this data and combining it with the movement method used and the subjective feedback, several performance and experience metrics about the trial can be inferred.

1. By calculating the time difference between the first and last logged location coordinates, the researcher can calculate completion time for the tasks in both trials.
2. The time of completion is used to weigh teleportation movement against slide movement in the first trial and observe users' navigation ability and average traversal speed across 3 tasks in the second trial. If user speed and therefore confidence increases with subsequent trials, it would indicate they are becoming familiar with the hospital environment.

3. Location and gaze data can be combined to observe which maps users looked at the most in trial 2 and also highlight problem areas where they may have lost their way.
4. As each data point records the last used locomotion method, researchers can observe the user's preferred mechanic. Furthermore, by observing what method users preferred at certain points in the hospital, it is possible to test if certain movement methods are more efficient for certain tasks or in certain locations of the hospital.
5. When paired with the survey mentioned earlier, the objective metrics can be used to investigate the correlation between learning curves for the VR experiences and users experience levels.

To analyze the data, the first step involves separating the logged spreadsheets for each task and checking for inconsistencies or outlying data, which would indicate the system lost tracking. This would occur in earlier iterations of the program which featured continuous logging and saving, the problem was remedied by only logging on customizable triggers like at each "onteleport" event. Although the change seemed like a loss in data resolution, it made the log data easier to read and removed the burden of frequent IO on the hardware.

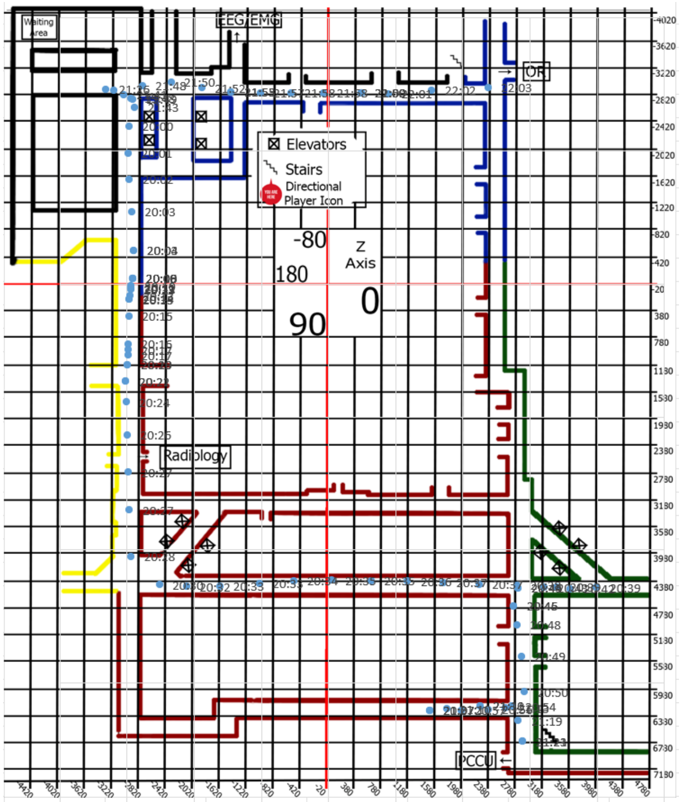


Figure 4-1 Timestamped Locations Plotted on Final Map

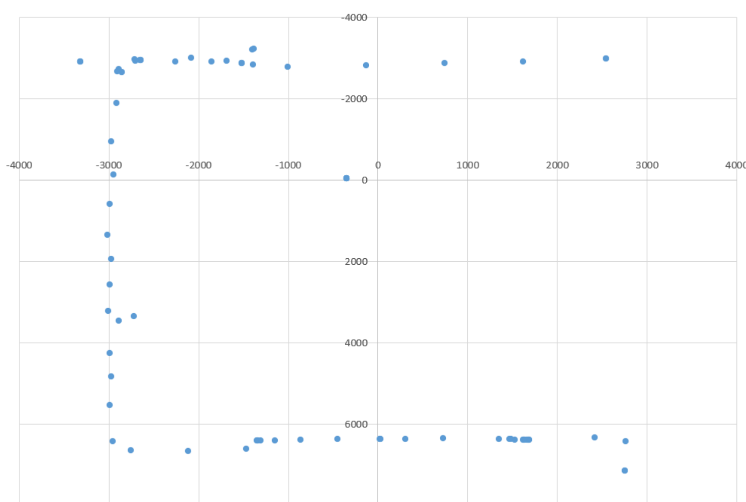


Figure 4-2 Locations Before Being Overlayed on Map

The location points are then plotted (Figure 4-2) on a coordinate grid overlaid on a map of the floor to visualize the user's path and identify any commonly used maps (Figure

4-1), areas of confusion, or areas where users felt most confident in their sense of direction. This was beneficial in the second trial when users had to determine their own route. The areas of confusion can be viewed as an accuracy measure as they indicate when the user got disoriented and needed to reference a ap to find their way again.

Finally, user survey results are reviewed to gather subjective feedback on the user experience. Although the statistical significance of subjective ratings may be lower than quantitative performance ratings, this feedback is crucial for planning future improvements to the pipeline, general interaction, and environment experience. Common complaints or positive comments from open-ended responses are grouped anonymously, with the number of similar responses indicating the significance of certain feedback.

Chapter 5

5 « Results & Discussion »

This chapter presents the results and discusses the findings of the study, focusing on the performance of the Victoria Hospital Demo built using the framework, the demo's objective measurements correlated with the survey's subjective ones, the impact of the simulation demo on users' spatial orientation and navigation skills, and finally, the affordability and resource efficiency of the developed framework. The discussion will also highlight the insights gained during the development and testing of the simulation, address the limitations of the study, and explore the implications of the results and user feedback for future work. Furthermore, the chapter will emphasize the success of the thesis in fulfilling its objectives through the framework and resulting proof of concept demo.

One of the main goals of this research was to showcase how a small team or an individual could, (A) leverage low cost, pre-existing resources and tools to transition from an idea to a VR simulation framework. In order for the framework to be considered a good base that others could use, it first needed to, (B) represent the real-world Victoria Hospital since that would allow it to be used for purposes like site orientation, and navigation trials. Next, to allow users that may not have the technological background of a software developer to utilize the framework it needs to have, (C) built in functionalities and components they can adapt for trial scenarios or for patient experiences. If they require additional specialized assets or environments, the system needs to remain flexible and, (D) expandible for easy addition and incorporation of foreign virtual assets and levels. This makes the system easy to use for the design team but its important to note that the system must also be, (E) easy to use for the subjects or patients, meaning it needs to be intuitive to navigate through the hospital, to perceive objects and layouts, and to interact with entities in the virtual world. Finally, the framework can not be detrimental to the users, so it needs to, (F) provide a comfortable experience where special considerations are made to minimize motion sickness.

To measure the success of meeting the study goals, it was important to evaluate if the final product meets the desired criteria of a good simulation. To summarize, for the framework to be considered a success, it needed to be:

-A) Developed at a low price with the use of pre-existing game development tools

-B) A faithful layout representation of Victoria Hospital

-C) Have built in functionality the design team can easily adapt for trial scenarios

-D) Easily expandable for additional environments and digital assets

-E) Easy for the patients or trial subjects to use through natural, user-centric design

-F) Be a comfortable experience with consideration to minimizing motion sickness

The success of meeting these study goals was gauged by two distinct sets of results, (quantitative and qualitative) which in turn evaluated the achievement of the study. The first measured the performance of the Demo through its representation of Victoria Hospital, adaptability, comfort, and interactivity. The second assessed the overall cost-effectiveness and utilization of pre-existing tools within the framework. The adaptability of the framework was evidenced in the creation of the trial demo, designed using the built-in environments and functionalities, and thus serving as an emulation of potential users' experience. Comfort and educational effectiveness were assessed through post-task user questionnaires and a comparative analysis of subject speeds in navigating different rooms during subsequent tasks of the second trial. This approach provided a comprehensive view of the framework’s effectiveness, from its fidelity to the real-world environment, to its user-centric design, adaptability, and comfort for users. The following subsection presents the objective user performance from the demo.

5.1 « Evaluation and Analysis of User Performance »

5.1.1 Subject Demographics

It is important with the small sample size to observe the demographic data presented by each subject. As mentioned in testing methodology, after each test subject completed all trials and exited the simulation, the next form of data collected for evaluation was the subjective results of the post-experience survey. The survey's first portion asked the user to select the age range which best suited them. The second portion was used to gauge their prior experience with VR technologies, video games, and their experience in the real Victoria Hospital. The third section specifically questioned the users on their experience in the VR Hospital Simulation Demo while the final portion consisted of open-ended questions for direct feedback about their experience.

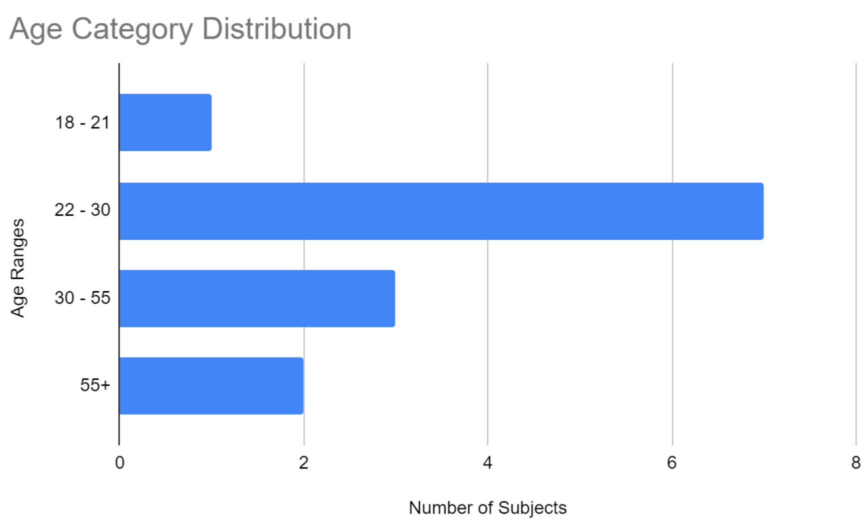


Figure 5-1 Number of Subjects in Each Age Category (Survey Categories)

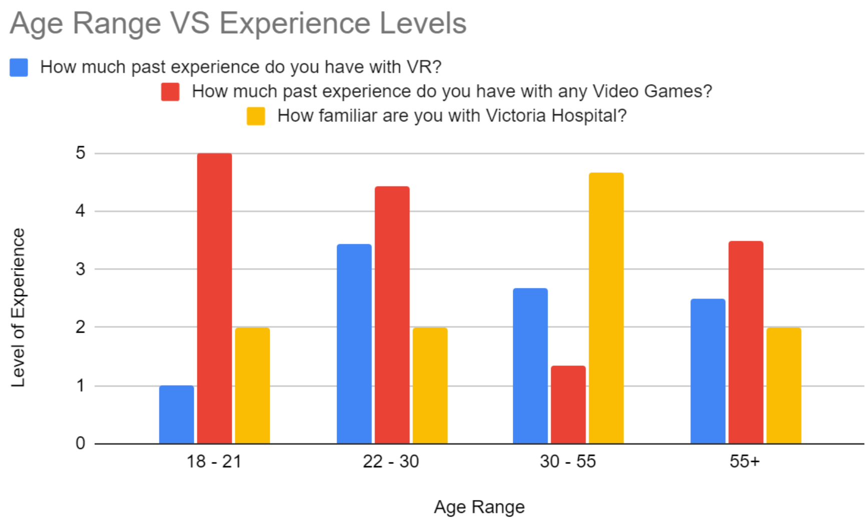


Figure 5-2 Average Experience Levels by Age Range (Survey Categories)

Figure 5-1 above shows the number of subjects tested from each age range. Figure 5-2 shows the distribution of age compared to the average levels of experience stated by each subject. The size of the study group negatively impacts the ability to draw distinct conclusions from the trials if the group is split into too many categories as illustrated by the youngest age group. The youngest bracket tested consisted of a single individual making them the singular data point representing that demographic's experience and performance. It is likely a larger sample set would account for a better representation of the population and a better spread of subjects over the age ranges. A larger sample size would also likely mitigate noise created from outlier performance data, providing more trends with which to draw conclusions. For this reason, although the feedback form has 5 divisions, this analysis will split the subjects into 2 groups. The two groups consist of trial subjects below the age of 30 and those above the age of 30 resulting in the following charts:

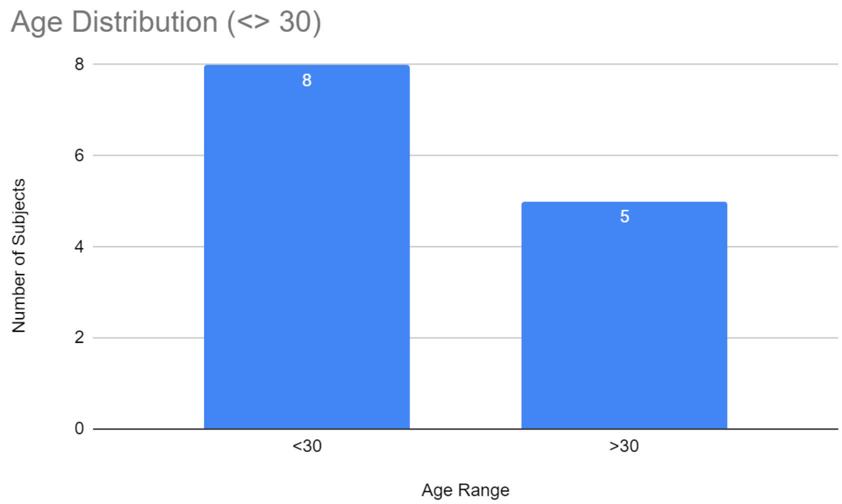


Figure 5-3 Number of Subjects in Each Age Category (Simplified Categories)

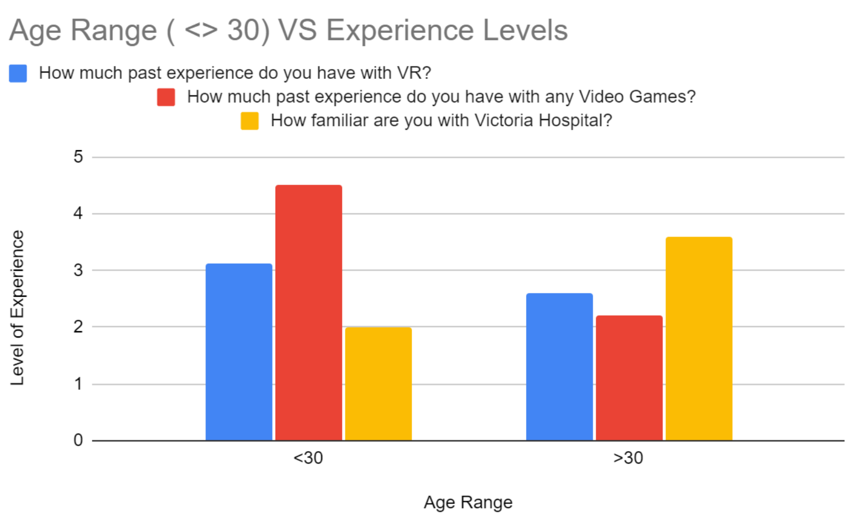


Figure 5-4 Average Experience Levels by Age Range (Simplified Categories)

From this simplified data, some observations on the demographics can be made. First, the trials consisted of 8 test subjects under the age of 30 and 5 subjects over the age of 30 (Figure 5-3). This distribution removes problems of a single subject representing an age range as stated above, resulting in a greater likelihood of drawing valuable conclusions based on age. Figure 5-4 shows the levels of experience split buy the new age range distinction. Of the subjects tested, the older age range had a higher average experience level at the physical site while the younger group had significantly higher experience levels in video games and a little higher average VR experience level.

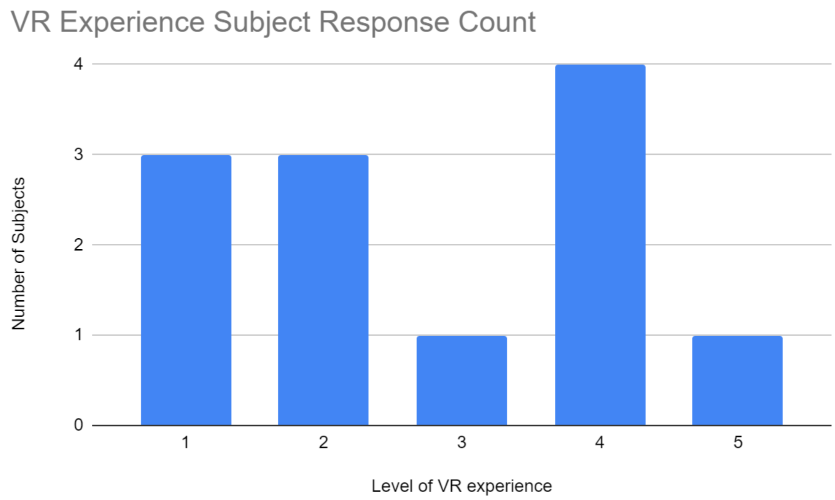


Figure 5-5 Number of Subjects for Each VR Experience Level

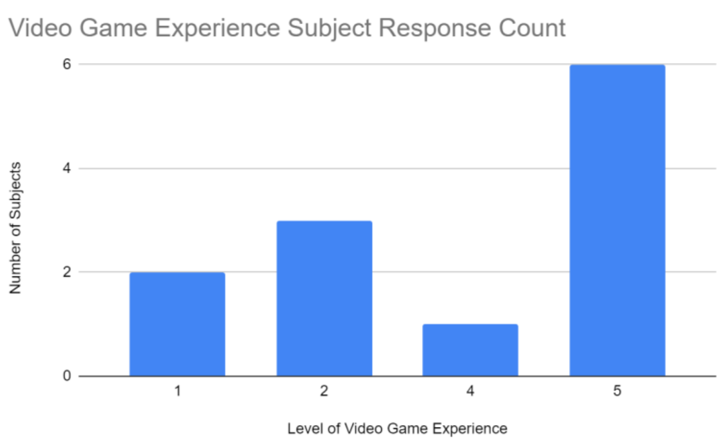


Figure 5-6 Number of Subjects for Each Video Game Experience Level

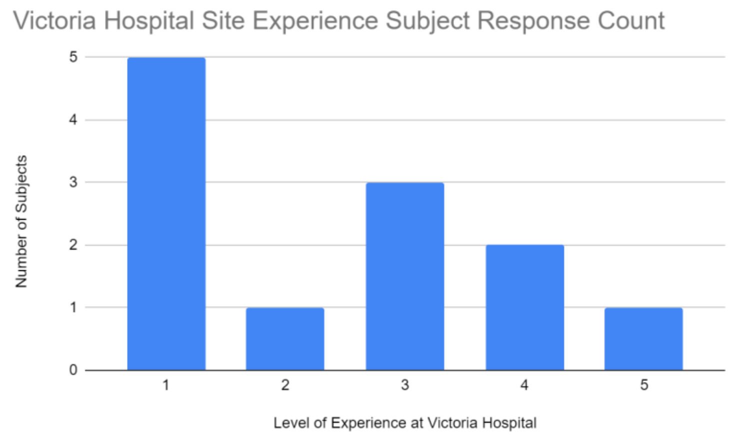


Figure 5-7 Number of Subjects for Each On-Site Experience Level

Continuing the analysis on the general subject demographics, the above charts (Figure 5-5, Figure 5-6, Figure 5-7) show the number of subjects rated at each of the experience levels. Of the subjects tested there was a close distribution in those with at least some level of prior VR experience (levels 4 and 5) and those with little to none (levels 1 and 2). Additionally, many of the users tested had a high level of experience with video games (levels 4 and 5) and a large number of users reported themselves as having little to no experience with Victoria Hospital (levels 1 and 2).

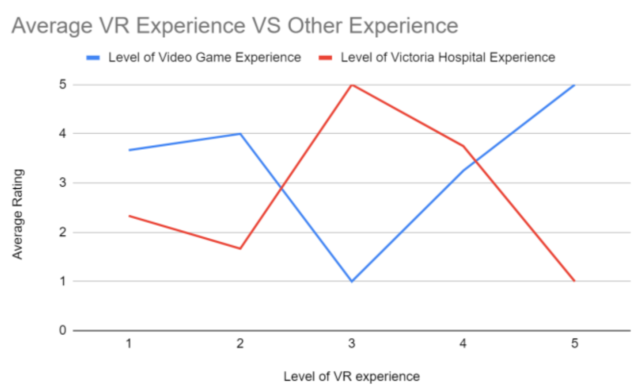


Figure 5-8 Level of VR Experience Correspondence to Other Levels

Figure 5-9 Level of Video Game Experience Correspondence to Other Levels

Figure 5-10 Level of Victoria Hospital Experience Correspondence to Other Levels

Interesting observation can be made about the subject demographics based on the charts shown above which show correspondence between the different experience types from different perspectives (Figure 5-8, Figure 5-9, Figure 5-10). Of the subjects with experience at the location, none had an above-3 degree of video game experience and only two rated above a 3 in VR experience. On average, subjects that rated themselves a 5 in terms of prior VR experience tended to have a lower level of experience on-site at Victoria hospital but a high level of experience in Video Games. This can be attributed to the age distribution mentioned above but also suggests that most VR users tend to use VR for gaming. This observation is reflected in the other two charts as they show the same data from the perspectives of video game experience and hospital experience respectively.

Within the simulation demo, the program shows images for each gesture as it re-explains them. The timestamp function records this, and also records the moment a successful gesture is detected. The time difference between the two represents the gesture learning time for each user. Figure 5-11 below illustrates the time taken for subjects to form gestures once instructions have been relayed.

5.1.2 Gesture Learning

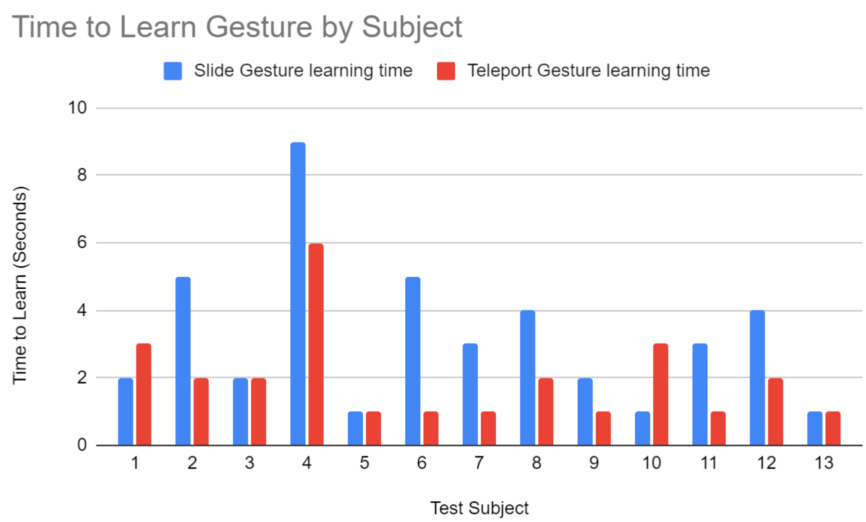


Figure 5-11 Time Each Test Subject Took to Learn Both Gestures

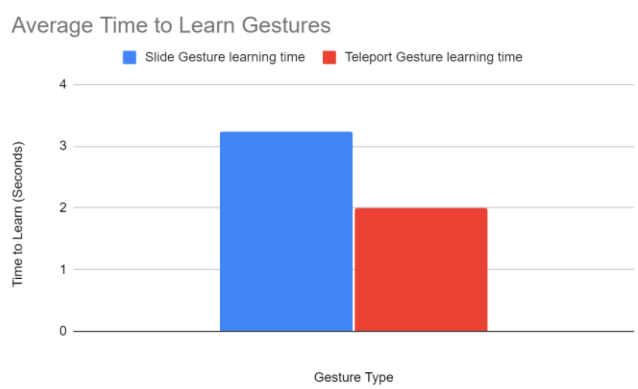


Figure 5-12 Comparison of Average Time Taken to Learn Each Gesture

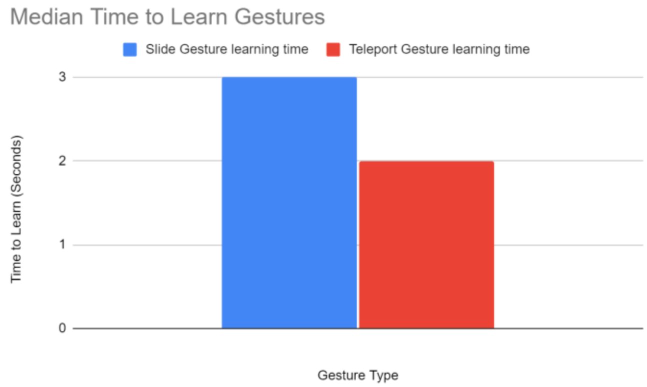


Figure 5-13 Comparison of Median Time Taken to Learn Each Gesture

A majority of test subjects were able to form both the sliding gesture and the multistage teleportation gesture in under 5 seconds respectively, though it took one user up to 9 seconds. This indicates the gestures were easy to learn for the majority of test subjects.

As a group, the mean and median times to learn the sliding gesture were 3.2 seconds and 3 seconds, compared to 2 seconds and 2 seconds for teleportation (Figure 5-12, Figure 5-13). Subjects managed to form the teleport gesture an average of 1.2 seconds faster and a median of 1 second faster.

It is important to note that there was a high degree of variability between users in learning the gestures, as indicated by the standard deviations. The standard deviation for the slide gesture was 2.24 seconds while that of the teleport gesture was only 1.4. The higher standard deviation can be explained partially by the time it took the system to detect subject 4's gestures.

The data shown in both figures suggest that although both gestures were learned quickly, on average, the teleport gesture was easier to form than the sliding gesture. It was observed that subjects with the longest sliding gesture times struggled specifically with the mechanics of keeping the pinky finger parallel with the index finger the sliding gesture. All users could bring their ring and middle fingers down to their palm while keeping their index fingers straight. This indicates the threshold value for the slide gesture detection was set to be too high a value.

Slide gesture difficulties were not noticed during preliminary testing as indicated in previous sections, as a result, the gesture remained unmodified for final trials. Once this difficulty was observed, the gesture remained the same as not to skew later results with a sharp performance increase as a result of simpler gesture implementation. Potential improvements to the detection system and gestures used are reflected in the Open Feedback and Implications for Future Development section.

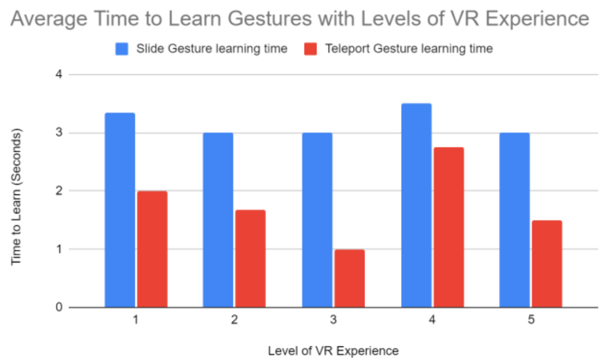


Figure 5-14 Average Time Taken to Learn Each Gesture By VR Experience Level

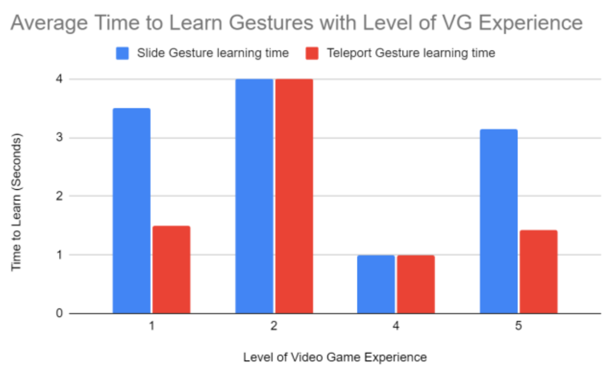


Figure 5-15 Average Time Taken to Learn Each Gesture By Video Game Experience Level

Figure 5-14 and Figure 5-15 above show there was no direct correlation between time to form gestures and the 2 experience level stats measuring:

- Prior level of experience with VR
- Prior level of experience with VG

As reflected in the overall results, no group performed better at forming the slide gesture when compared to the teleport gesture regardless of the experience level they reported in the categories. It is suspected that with a larger sample size resulting in a lower impact of outlier data, past VR experience would have a larger impact on gesture learning. This hypothesis is formed on the basis that subjects with large amounts of prior VR experience would have more practice learning distinct hand gestures for different VR games/applications.

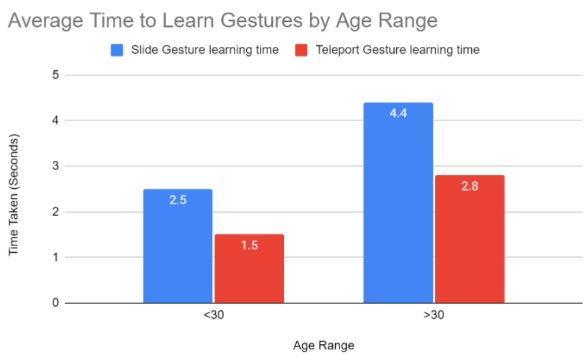


Figure 5-16 Comparison of Average Time Taken to Learn Each Gesture By Age

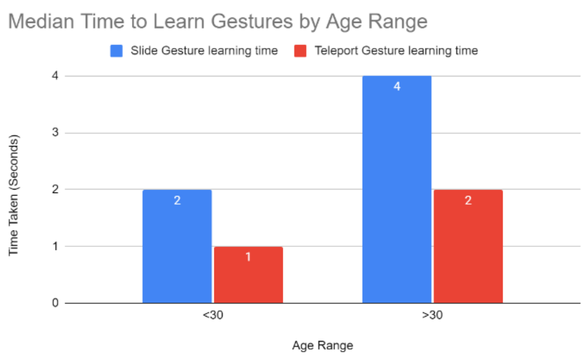


Figure 5-17 Comparison of Median Time Taken to Learn Each Gesture By Age

Although there was no correlation between experience levels and gesture performance, the results in the charts above (Figure 5-16 and Figure 5-17) indicate there is a correlation between age and the average time taken to learn gestures. Both the average and median results reflect that the subjects below the age of 30 had an easier time performing both gestures than those above the age of 30. The difference in both cases shows that the younger age bracket took half the time to learn, then successfully perform the gestures when compared to the older age group. This result can be attributed to the possibility of hand dexterity decreasing with age. A study by (Bowden & McNulty, 2013) explores the possibility of decreased dexterity with age and concludes that although there is a small correlation till the later years, it does have an impact.

5.1.3 Performance in Locomotion

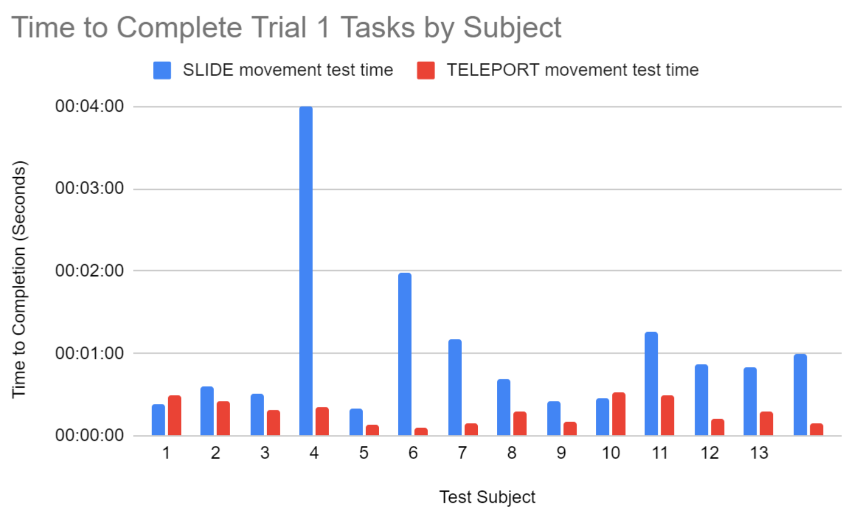


Figure 5-18 Time Each Test Subject Took to Complete Trial 1

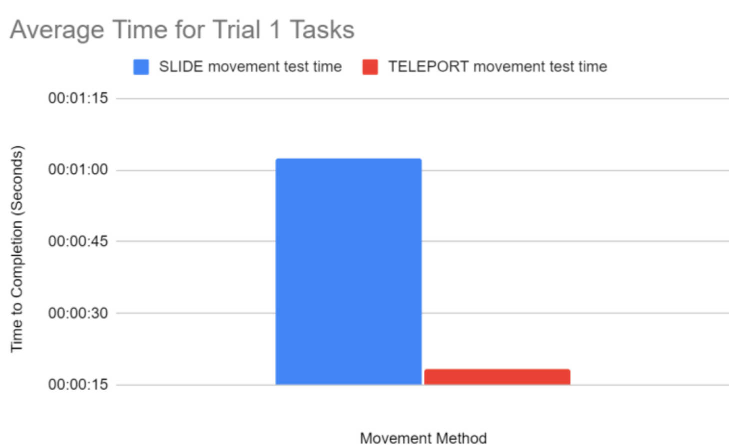


Figure 5-19 Comparison of Average Time Taken to Complete each Task from Trial 1

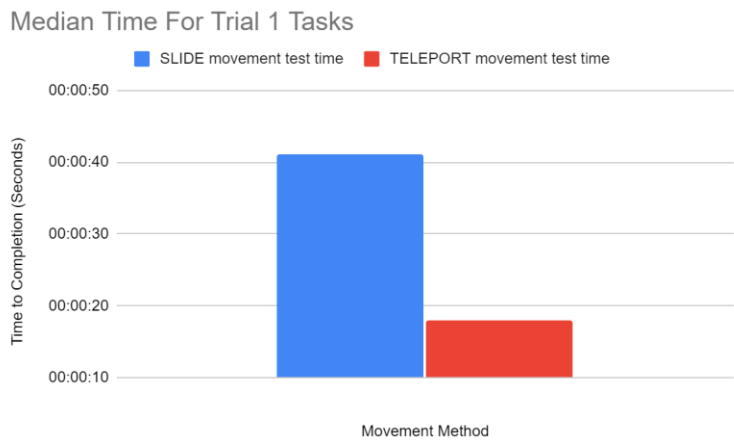


Figure 5-20 Comparison of Median Time Taken to Complete each Task from Trial 1

In the first task where movement methods are being compared, the user was given the goal of following an indicated path to the target area of the first floor of Victoria Hospital. They were asked to preform this task twice; once with the sliding method of locomotion and the other using the teleport method. The path to follow was kept simple and required the user to travel down a hallway and make one turn to reach the goal. Overall recorded times ranged from a min-max of 20-240 seconds with slide locomotion and 6-32 seconds with teleport locomotion (Figure 5-18). The mean, median, and standard deviation times for completing the Slide movement test were 62 seconds, 41 seconds, and 60 seconds, respectively. For the Teleport movement test, the mean, median, and standard deviation times were 18 seconds, 18 seconds, and 9 seconds, respectively (Figure 5-19 and Figure 5-20). These figures indicate that the average time to complete the Slide movement test was longer than that for the Teleport movement test. However, the high degree of variability in the data, as illustrated by the large standard deviation figures, suggests that some participants were able to complete the tests much faster than others. The use of median times alongside the mean provides a more accurate representation of the central tendency of the data and helps to minimize the effects of outliers for the small sample size.

It is important to note that with the removal of the outlier, the standard deviation for the Slide movement test decreases significantly from 60 seconds to 29 seconds. Additionally, the mean time to complete the Slide movement test also decreases from 62 seconds to 48 seconds, bringing it closer to the original median time of 41 seconds. This suggests that the outlier had a significant impact on the average time to complete the Slide movement test. This result cannot be taken to conclude that navigating via teleport is more efficient than sliding as the hand gesture likely had an impact on user performance. When the measure of success was a single successful gesture, subjects had no need to consistently hold the gesture but when using it to slide the gesture needed to be continuously held. If users were unable to hold the gesture, then their movement and acceleration (since slide movement accelerates as it is held) would come to halt, taking more time.

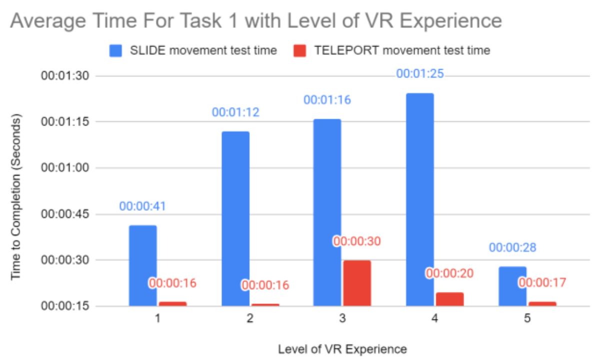


Figure 5-21 Average Time Taken to Complete Trial 1 Tasks By VR Experience Level

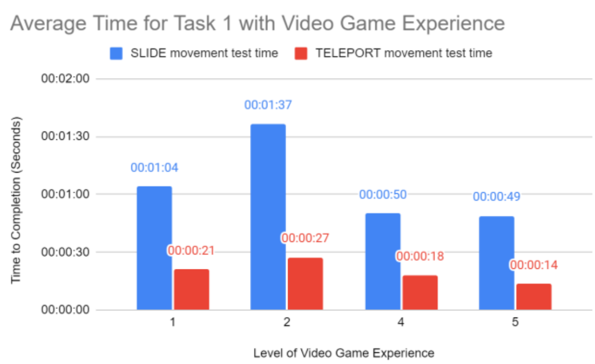


Figure 5-22 Average Time Taken to Complete Trial 1 Tasks By Video Game Experience Level

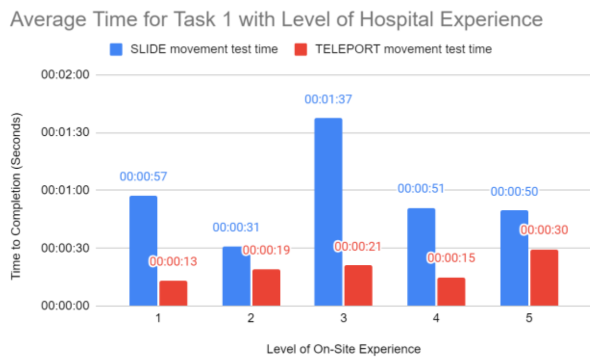


Figure 5-23 Average Time Taken to Complete Trial 1 Tasks By Experience at Victoria Hospital

As the charts above (Figure 5-21, Figure 5-22, and Figure 5-23) show, there was no significant correlation between time to complete task 1 and the 3 experience level stats measuring:

- Prior level of experience with VR
- Prior level of experience with VG
- Prior levels of experience with the real Victoria Hospital

It is interesting to note that on average, the subjects who rated themselves a 5 in VR experience performed among the best in the trials but those who rated themselves a 4 performed among the worst. This idiosyncratic result may be due to outlier data as the median slide test time for that group is 38 seconds. On average the subjects which rated themselves at a 4 and 5 in terms of video game experience had shorter times to completion than those who rated themselves as having low experience. The slower results of the inexperienced users could be a result of induced motion sickness since those who felt sick were unable to slide continuously from point A to point B and required breaks thereby reducing performance. This aligns with the author’s expectation as a study performed by Theresa Pöhlmann et al., (2021) showed that video game experience has a direct impact on a target’s susceptibility to motion sickness. The lack of correlation in Victoria Hospital experience and time to completion was expected since, as mentioned before, all other doorways were closed, and users were shown a path to follow to stop navigation difficulty from being a factor.

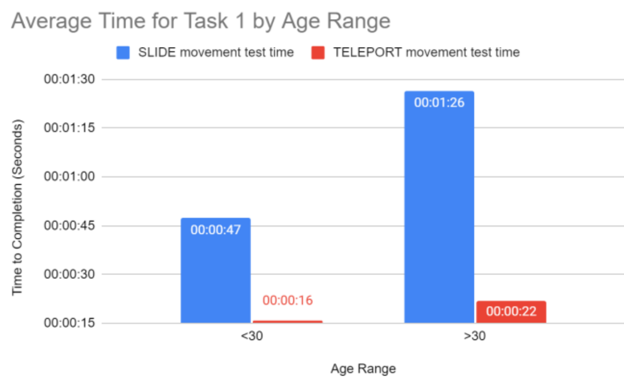


Figure 5-24 Comparison of Average Time Taken to Complete Trial 1 Tasks By Age

Figure 5-24, showing the times for task 1 completion against age range illustrates that the subjects below the age of 30 performed the task significantly faster than the subjects above the age of 30 using slide locomotion and a little faster while using teleportation locomotion. This result is also reflected when comparing median values meaning the discrepancy is not a result of the outlying data. The larger difference in completion time for the sliding test may be compounded by difficulties forming the activation gestures. As seen in the previously mentioned evaluation, the older group had a more difficult time forming the gesture for the sliding movement which would slow down locomotion speed each time they need to reform the gesture.

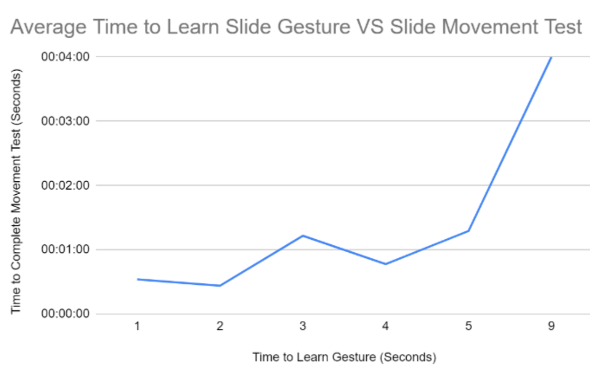


Figure 5-25 Correlation Between Slide Gesture Learning Time and Slide Movement Task

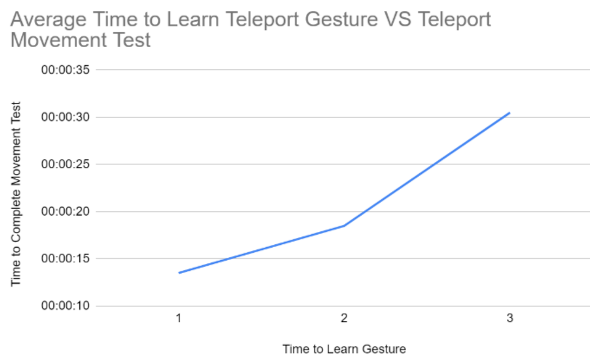


Figure 5-26 Correlation Between Teleport Gesture Learning Time and Teleport Movement Task

As mentioned above, it is important to note that the results of this movement test are correlated with any difficulties in forming the gestures. Since the metric which measured time to learn the gestures waited for the first successful detection to trigger the timestamp function, this meant that users who strained their hand to straighten their pinky fingers were still able to form it under 10 seconds. The movement tests require the users to hold the slide gesture and to repetitively use the teleport gesture. Figure 5-25 above illustrates that if a subject had difficulty forming the spiderman (slide) gesture during the tutorial it usually has a direct correlation with their time to completion of the sliding movement test. The correlation on gesture performance's effect on the movement test performance is further strengthened with observation of Figure 5-26. Users that took longer to learn the teleport gesture also took longer to complete the movement test.

The developed system being sensitive to user performance yielded the innovation of also being used to measure the difficulty in certain gestures. The dual interaction styles when compared, showed a significant difference in the user performance when forming a gesture they found easier. This carried over to each movement test showing that the more difficult gesture played a strong part in the performance of the slide movement. In the later trial when navigation ability is tested, the user is free to use their preferred method as the default as the system offers the option of both movement techniques. In future iterations, interactions and capture techniques will become more refined (such as iterating the slide gesture to be easier) but the testing methodology can remain the same, that way the changes can be compared.

Aside from the result that many users would have difficulties forming the slide gesture, prior to testing, it was suspected that that many users would have difficulty sliding continuously from the start to the goal.

There were two reasons for the hypothesis:

1. The user would turn their head in the direction of the turn without bringing their gesture-holding hand with it. This results in the cameras losing track of the hand, thereby bringing movement to a halt and resetting the acceleration counter.
2. Some users had to use the slide movement in small bursts to avoid experiencing motion sickness. The slide locomotion was designed to gradually accelerate, the longer the gesture is continuously held. These users were observed to release the gesture when the movement speed reached near- maximum levels before reengaging.

This would inherently result in poorer time to completion than those able to travel at max speed continuously and turn without losing hand tracking. Potential solutions to these problems are explored later in discussion.

It is evident that in future work, more testing should be done comparing the two methods using a modified gesture so the activation method is not a factor in people performing poorly using slide locomotion.

5.1.4 Performance in Navigation

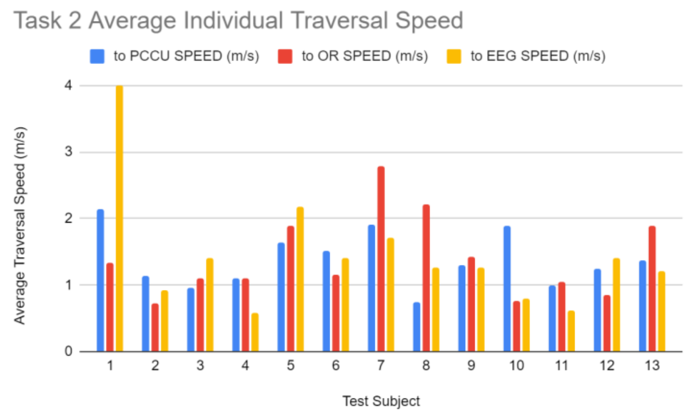


Figure 5-27 Average Traversal Speed (m/s) for Each Test Subject During Tasks of Trial 2

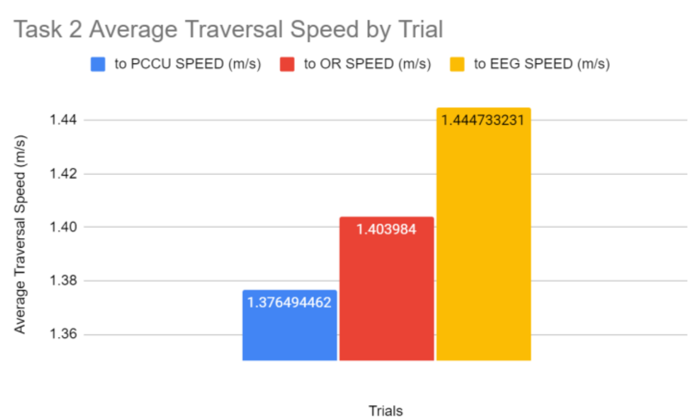


Figure 5-28 Average Overall Traversal Speed (m/s) During Tasks of Trial 2

The next and final timed test required the subjects to navigate to specific locations on the second floor of Victoria Hospital. The trial subjects were able to use whichever traversal method they preferred and the recreation had additional maps placed along the colour coordinated walls with a compass near each one.

The above charts (Figure 5-27 and Figure 5-28) show the average speed at which individuals traversed one the second floor of Victoria hospital to complete the navigational tasks. In this comparison it wasn't feasible to compare time to completion as the goals were set at different distances (using shortest route). The PCCU was located

120 meters from the start point, the OR was 64 meters, and the EEG was 28 meters. To calculate time the simple distance/time equation was used.

Illustrated in the second chart, the average speed for reaching the PCCU was 1.38 meters per second(m/s), for the OR it was 1.40 m/s, and for the EEG it was 1.44 m/s. These results show that the speed of traversal increases consistently across subsequent tasks on the second floor of the hospital.

This increase in speed suggests users gained confidence in navigating the environment and orienting themselves in the space through subsequent tasks. Although this does not necessarily mean their knowledge of the virtual environment would translate to confidence in navigating the real-world location of Victoria Hospital, it indicates that the software does help users grow more accustomed to navigating the hallways and turns. The argument is only strengthened when noting that some users, after inspecting the map several times on the way to the PCCU, no longer needed to reference it when navigating to the other two goals.

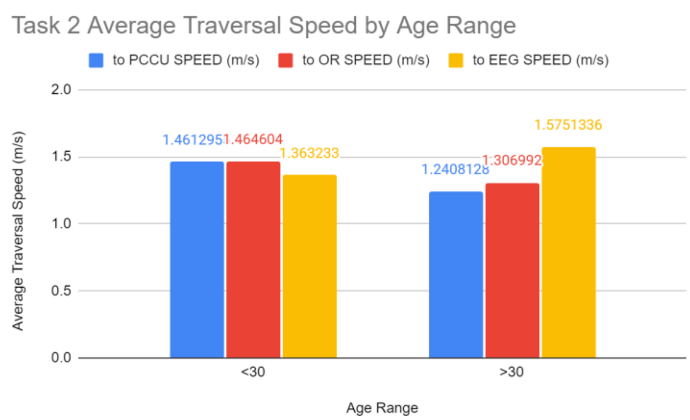


Figure 5-29 Comparison of Average Traversal Speeds During Trial 2 Tasks By Age

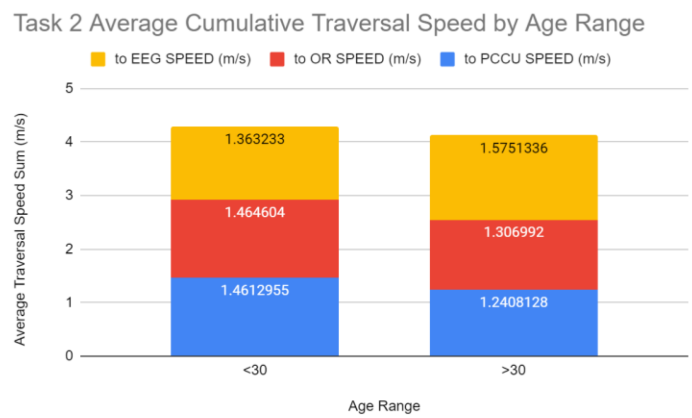


Figure 5-30 Comparison of Average Cumulative Traversal Speed During Trial 2 By Age

Figure 5-29 compares the average traversal speed through trial 2 to the age range of the test subject. In this test, the subjects over 30 were slower on the way to the PCCU and the OR but moved faster on their way to the EEG, this outcome is mostly attributed to subject 1's high traversal speed for the final task of the second trial. They managed to move through the hospital at 4 m/s which was higher than all others in every task. The older age group also showed more signs of improvement from task to task over the younger who performed the slowest in the last task. While they may have improved more, the age group under 30 still moved a little faster on average across all three trails (Figure 5-30).

There is the possibility that this result could be attributed to the formation of hand gestures but as this trial allowed the user to move using their preferred method, it is unlikely. The two charts below (Figure 5-31 and Figure 5-32) show at most a very weak correlation between gesture learning time and trial 2 traversal speed which further disproves the theory.

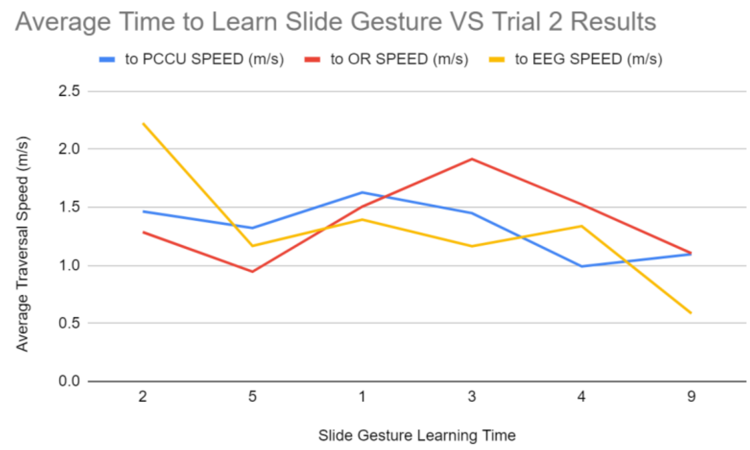


Figure 5-31 Average Traversal Speeds During Trial 2 Tasks Compared with Time Taken to Learn Slide Gesture

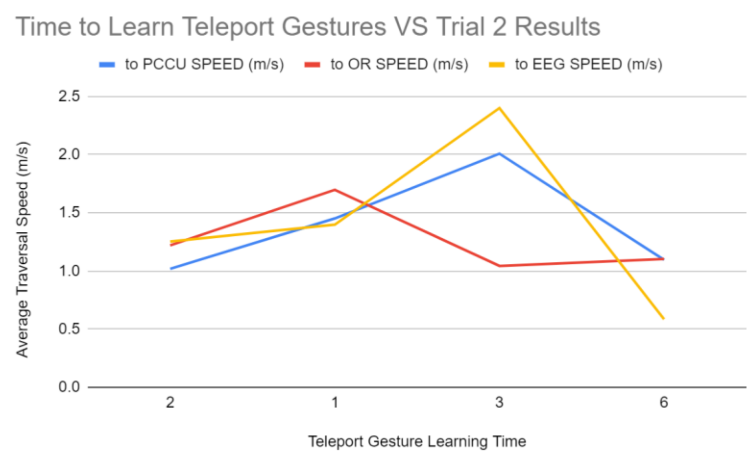


Figure 5-32 Average Traversal Speeds During Trial 2 Tasks Compared with Time Taken to Learn Teleport Gesture

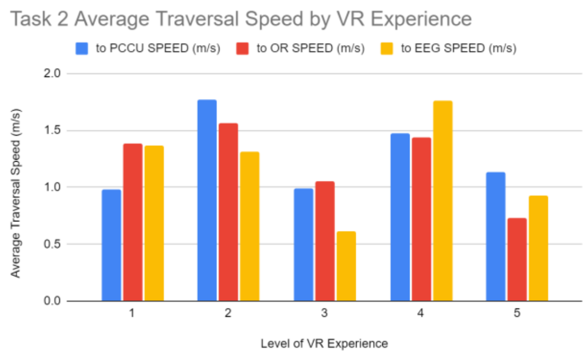


Figure 5-33 Comparison of Average Traversal Speeds During Trial 2 Tasks By VR Experience Level

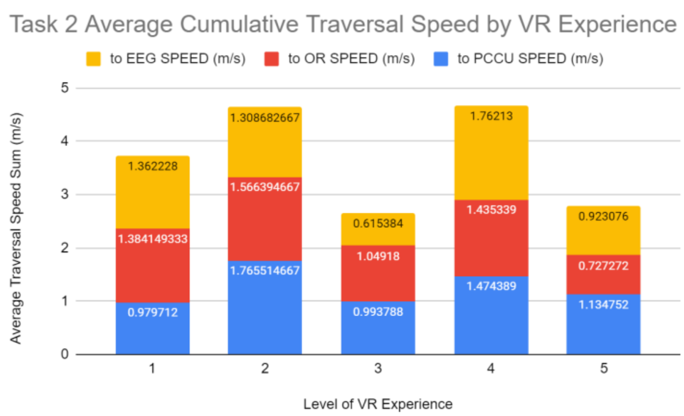


Figure 5-34 Comparison of Average Cumulative Traversal Speeds During Trial 2 By VR Experience Level

The next set of charts (Figure 5-33 and Figure 5-34) show the traversal speed per task and altogether when compared to the level of prior VR experience. These results do not illustrate a correlation between the prior VR experience and the traversal speed with the subjects who rated their experience as level 1 and 2 outperformed those who rated themselves higher. This is a result that deserves further research with a larger study but it is important to note the importance of the system still being able to collect the data to test the relation.

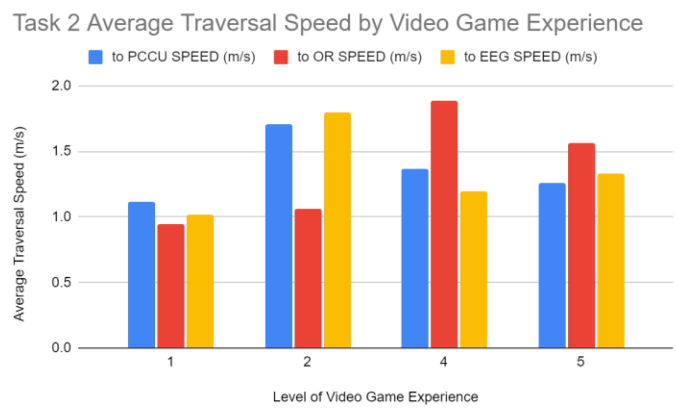


Figure 5-35 Comparison of Average Traversal Speeds During Trial 2 Tasks By Video Game Experience Level

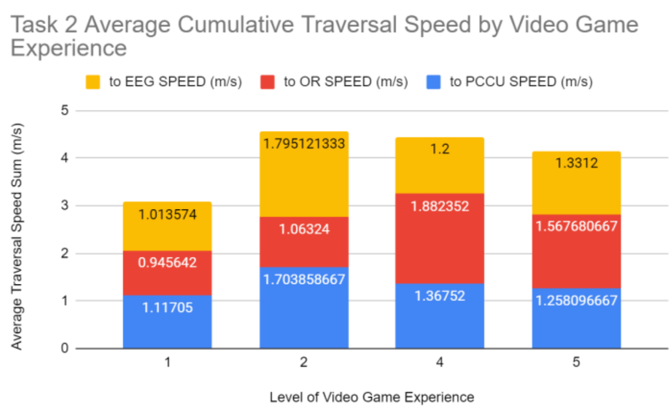


Figure 5-36 Comparison of Average Cumulative Traversal Speeds During Trial 2 By Video Game Experience Level

This set of charts (Figure 5-35 and Figure 5-36) illustrate the level of prior video game experience when compared to the traversal speed for task completion. Although there was no observed correlation from one task to the next since the level 2 experience subject performed the best, the stacked graph shows that the group which rated their experience level at 1 performed the worst which would be the expected result.

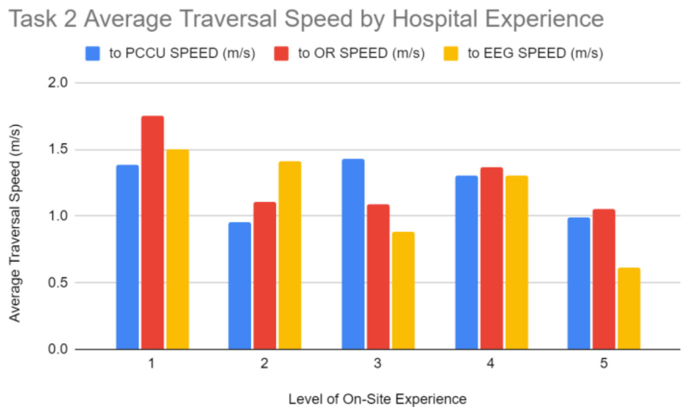


Figure 5-37 Comparison of Average Traversal Speeds During Trial 2 Tasks By Experience at Victoria Hospital

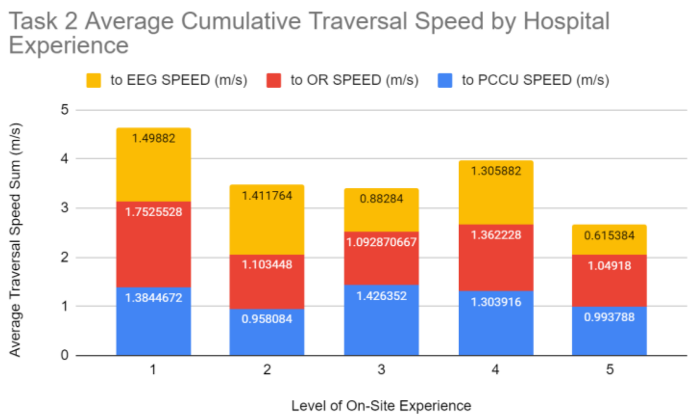


Figure 5-38 Comparison of Average Cumulative Traversal Speeds During Trial 2 By Experience at Victoria Hospital

The final set of charts from the trial (Figure 5-37 and Figure 5-38) exhibit the comparison between the level of experience the test subject had at Victoria hospital to their traversal speed through the virtual environment. Although there isn't a direct correlation between experience and traversal time, a result shows the group having the least amount of on location experience traversed the halls the fastest. However, live observation of the trials showed all subjects with high degrees of location experience were able to complete the second task without the need to stop at the map locations as frequently as any other subjects, 2 of the subjects looked at the map at the very beginning to orient themselves and then did not glance at any on the way to PCCU or any of the other tasked locations.

This would indicate that some navigational familiarity carried over from their on site experience to the trial but they moved slower than other groups using the system mechanics.

5.2 « Success in Meeting Objectives »

As mentioned earlier, the accomplishment of the study and demo's objectives relies on the evaluation of both the performance metrics and subjective responses collected from the feedback survey.

To reiterate, the three objectives planned for evaluating the feasibility demo were as follow:

1. Offer an immersive, lifelike environment conducive to learning, testing, and comparing distinct movement methods, gestures, or interactions with virtual objects.
2. Effectively promote the users' ability to familiarise themselves with and navigate environments that may be utterly foreign to them.
3. Provide the user with a comfortable simulation of the environment and procedures they may be experiencing for the first time.

One of the design objectives was that the framework be a platform from which researchers can conduct empirical research to measure user performance. The demo serves as an example of running trials by using components from the framework. To see if the demo is a successful example, success criteria was made (shown above), to test if the demo meets the first criteria for success (the ability to compare movement methods) we examine the results of the first trial.

The first trial was designed to enable the comparison of user results in a controlled environment with limited variables (movement methods). In this trial, the users were timed as they followed a simple path using each movement method separately. Hallways and rooms diverging from the path were blocked off to avoid confusion. Between the two tasks, the only variables were the movement method the user was required to use, and the gesture set for activating it. The trial revealed that, on average, users performed better using teleport locomotion compared to the slide movement. When paired with survey responses, these results indicate that a majority of users who felt motion sick experienced this discomfort during the slide task, suggesting that the teleport mechanism is the more intuitive movement method. This strong preference for teleport locomotion was further supported by the data logged during task 2 which recorded the movement methods used on the second floor. Furthermore, before the trials began, the software also gathered data on the time taken to learn each gesture which was useful to structure future iterations of the program. Thus, the first objective was achieved, and the gathered data suggest that teleport locomotion is the more user-friendly movement mechanic.

The second objective was evaluated through the second trial where users were given the tasks of navigating to different rooms on the second floor. The gradual increase in subjects' movement rates with each subsequent task indicated their growing familiarity with the hospital layout, which meets the second objective of the demo. This suggests that the virtual hospital simulation can indeed aid in improving spatial awareness and navigational skills. Additionally, users with experience at the physical location were able to apply their real-world knowledge to help themselves navigate the hallways. This suggests that the virtual hospital simulation accurately represents the layout of the real environment.

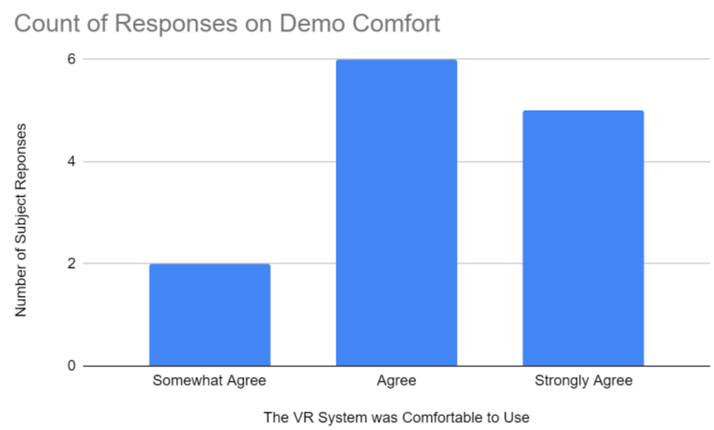


Figure 5-39 Number of User Responses on VR System Comfort

The evaluation of the final objective, user comfort, posed a unique challenge given its inherent subjectivity. This was primarily gauged through post-experiment survey responses, wherein questions 8, 11, 12, 13, and 18 were designed to obtain user feedback regarding their level of comfort and potential confusion while using the system. Question 8 solicited user comfort levels during the demo experience, utilizing a Likert scale to measure responses. Participants were asked to rate their agreement with the statement, "the simulation was comfortable." The response distribution leaned towards agreement, with all participants choosing "somewhat agree" or a more positive response (Figure 5-39). This feedback suggests that the third objective, providing a comfortable simulation, was indeed met.

However, comfort can also encompass the absence of adverse effects like motion sickness, which was explored in questions 11 and 18. Using the Likert scale again, users were prompted to rate their agreement with the statement, "I felt some motion sickness or dizziness when using the system." While some users did acknowledge experiencing some level of discomfort, the majority fell on the disagreement side of the scale, with 4 users strongly opposing the statement (Figure 5-40). An open-ended question (18) provided additional insights into the nature of the motion sickness reported by some users, which generally traced back to a specific movement method. The system's flexible locomotion design, allowing users to select their preferred movement method in task 2 and the MRI, mitigated this issue to a large extent, thereby enhancing overall user comfort. The optional MRI demo, while only experienced by a few users, received positive feedback. Those who opted for it agreed that it offered a satisfactory representation of a real MRI procedure, which bolsters the success of the objective to provide a comfortable and accurate simulation experience.

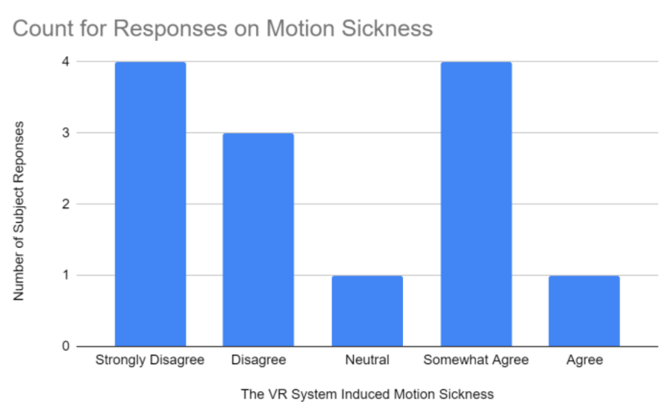


Figure 5-40 Number of User Responses on Motion Sickness in the VR System

With the demo objectives met, the study objectives were as follow:

1. Design and analyze a VR hospital framework of the LHSC Victoria Campus that caters to two primary user groups: clinicians and patients, by demonstrating the potential of using free tools like Blender and Unreal Engine in an affordable development workflow.
2. Explore the potential of the framework to improve ease of navigation, alleviate patient anxiety, and provide a platform for remote professional training at LHSC Victoria Hospital.
3. Ensure the framework is scalable, modular, and easily deployable, enabling both individuals with limited technological training and clinicians to actively participate in the design and authoring process. The purpose being to enable them to create new aspects of the environment, collaboratively configure new layouts, and design specialized rooms tailored to their needs.

To determine the framework's success, the demo needed to exhibit its ability to employ the built-in functions and environments of the framework in developing interactive trial scenarios that collect useful data. The methodology and results of the experiment demonstrate how a clinician could easily extract the relevant hallway section from the full system and assign a pawn with the desired capabilities for the trial. It’s important to note that the framework’s success is measured by its ability to create useful trial software and not by the strength of correlations between performance metrics and user identification variables such as age and experience levels. In this case, the trials aimed to 1) compare two locomotion methods and 2) evaluate users' improvement in navigating different areas of Victoria Hospital's second floor. The demo also included gesture learning time and a survey to provide researchers with additional data to draw conclusions about user performance. This highlights the demo’s success and the developed framework's value in streamlining the trial creation process, making it accessible to novice developers, clinical staff, and other researchers.

Evaluating the affordability of the framework requires a summary of the development cost and a comparison with the average prices of custom VR simulations. The initial feasibility assessment involved using readily available resources to allow an individual or small team to create a successful Hospital Simulation Framework at a lower cost than hiring a full development team for custom simulation creation or purchasing off-the-shelf simulation software. This can be measured by comparing the overall cost of resources required to discover, develop, and use the VR Victoria Hospital Simulation.

Table 5-1 provides a breakdown of the equipment and resources used, along with the approximate cost of each. The table presents three different total prices to differentiate between the software alone and the software with required hardware. The cost of development primarily lies in the price of the assets, textures purchased, and the hardware used for development and testing.

Table 5-1 Cost Breakdown for VR Framework Development

Item	Cost (CAD $)
Blender	Free
Unreal Engine 4	Free
3DS Max (optional)	Free (for 3 years with education license)
PhotoPea	Free
Oculus Quest 2	$460
VR Hand Tracking Plugin	$54.36
DirectExcel Plugin	$27.17
Prop Asset Bundle (optional)	$16.98
Various Textures (optional)	Free

Lenovo Legion 5 (variable)	$1500
Total w/o Laptop and Quest	$98.51
Total w/o Laptop	$558.51
Total	$2058.51

All prices in Canadian dollars, while Blender is a tool freely available for anyone, 3DS Max is a professional tool with a free trial for evaluation but an annual price of $2445 or monthly price of $305. The developer was a student at the University of Western Ontario, therefore with the 3-year free educational licence was able to access the program at no cost. Photopea was used as free web-based alternative to Photoshop through a majority of development as it allowed for easy access to images saved on cloud sources. The final iteration of the Framework upon which the proof-of-concept trial was created utilized two mandatory paid plugins, one optional paid asset pack, and a small number of free to download assets and textures. The VR hand tracking plugin had a price of $54.36, the DirectExcel plugin had a price of $27.17. The Hospital props bundle had a price of $16.98.

The VR headset used for this study, the Oculus Quest 2, had a launch price of $460, and the laptop used for development, a Lenovo Legion 5, was purchased for $1500. With all the development tools and assets included, the total cost of the VR hospital Framework developed for this study was $98.51 in just software, $558.51 with the headset included, and $2058.51 with the development system included. It is important to note that all the costs totalled above do not include any potential wages for developers. The developer of this framework took 2 years to research the field, learn how to use Unreal Engine and the supporting tools, develop the framework, then use the framework to create the demo. If an organization were to create something similar form scratch it would be crucial to factor in the time and cost of a developer to learn how to and to create something similar.

Starting with the framework, and using the tools developed during this study would significantly reduce the time to create new experiences.

Comparatively, existing simulators in the healthcare space are often priced beyond the reach of small research groups or hospitals. For instance, the VIST system, one of the world's most sophisticated VR simulators that models the entire vascular system, carries a hefty price tag of $300,000 USD per unit (Gallagher et al., 2005) making it difficult to acquire. Although this is an extreme example, other custom VR simulations can cost from $2000 to $150000 to procure according to RoundtableLearning.

A study by (Pottle, 2019) explores a comparison between physical simulations and virtual and states, that studies defining the cost of fully immersive medical simulation approximate that for one learner to lead one simulation scenario costs over £200; for example, the paper by (McIntosh et al., 2006) concludes Set up cost was US$876,485 (£758,300) (renovation of existing facility, equipment). Fixed costs per year totalled $361,425 (£275,000). Variable costs totalled $311 (£237) per course hour and a separate study by (Iglesias-Vázquez et al., 2007) states that the ‘cost of ALS (advanced life support) simulation for a 4-day course is €1,320 (£1,140) per passed Participant.

In contrast, as mentioned earlier, this project successfully leveraged the most affordable dedicated (not cell phone powered) VR headset available, the Oculus Quest 2, taking full advantage of its unique traits. Its built-in hand tracking capabilities and the ability to use them with minimal setup due to inside-out tracking proved invaluable. The trials were conducted in several different environments, including test subject homes and the lab in UWO TEB, further demonstrating the framework's adaptability.

This shows the project successfully utilized pre-existing resources, Unreal Engine and Blender, powerful tools for video game creation and 3D development. These tools, industry-standard for creating high fidelity game environments, were proven to be capable of designing full rooms as well as modular components for larger environments within this study.

As mentioned in the Development Apparatus section of this thesis, 3 different systems were used through development while retaining the ability to test the software, 2 laptops and a desktop. The laptops were an Nvidia RTX 2060 mobile AMD Ryzen 4000 laptop, and a more powerful RTX 3070 mobile Ryzen 5800H laptop. The desktop machine used a Ryzen 5900X and 3080Ti GPU where the GPU alone cost more than either laptop but bringing decreased times during compilation, rendering, and packaging.

While the mobile systems used in this study were more than capable of developing the simulations, it was clear that the increased power of a well-specified desktop system allowed for more detailed environments and improved render times. The study thus illustrates that although while workflow can be significantly enhanced with increased hardware power, VR, AR, and XR development is still accessible at lower prices and can be conducted on mobile systems. Although the mobile systems had less power than the desktop, they had the advantage of being portable, the Lenovo laptop was crucial in the testing phase as it made it possible to test users with a mobile PC VR experience allows for higher visual fidelity.

Next, the tools and functionalities of the framework were evaluated for their extensibility to create an objective-driven demo experience. The pilot study demo aimed to demonstrate the capabilities of using the framework to develop a purpose-built trial software for Victoria Hospital easily. The success of this thesis does not depend on the trial demo's ability to provide a scientific conclusion but on the design of the framework's flexibility, scalability, adaptability, and modularity. The trial demo is a proof of concept, demonstrating the systems in place to recreate Victoria Hospital environments and log useful data for comparing different variables.

The framework environments include:

- First floor zone D of the hospital, leading to and including several areas such as CT/MRI Reception, Ultrasound Reception, Victoria Research Lab, Children’s X-ray, and others.
- The second floor zones A, B, C, D, leading to areas such as Registration, Atrium, EEG/EMG Lab Test Center, OR Waiting Room, Day Surgery, and many more.

- Individual rooms like the OR, PCCU, Xray lab, CT, MRI, Operation Theater, and Clinician’s Office.

The results underscored the framework's efficacy, particularly its maps and interactions, to rapidly create a demo used for subject trials.

In conclusion, the study successfully utilized affordable and readily available resources to develop a VR hospital framework, showing its value in streamlining the development process and making it more accessible. The experiment proved that the framework was capable of creating a purpose-built trial software, achieving the study's objectives and demonstrating the significant potential of this approach for future healthcare applications.

5.3 « Open Feedback and Implications for Future Development »

The research project successfully completed all goals laid out for the workflow, the framework, and the demo thereby completing the goals for the study. The workflow showcased how a small team or even an individual can leverage freely available resources and tools to transition from a simulation of an environment to an adaptable interactive digital twin. This is shown by the full extent of the system created with the various hospital rooms, sections, and experiences made with a sole developer with the help of absolute beginners to 3D modelling with limited technical experience with Unreal Engine. The hospital simulation system demonstrated its extreme flexibility in how, using all the resources, blueprints, environments, interactions, and other tools developed, a programmer can design and implement a purpose-built demo with good performance by extracting the desired components in a very short timeframe. The demo can also be modified almost immediately by bringing in desired tools like pawns with active timestamping built in when needed. Finally, as expressed above, the demo also met all goals it was designed for, thereby proving the success of all three elements within the scope of the study. The feasibility demo's final iteration showcases the system's capacity to empower users to directly learn, test, and compare features and locations implemented in this demo. This demonstration is a benchmark for system performance and yields invaluable insights for future enhancements.

This is not to say all three components were without fault; the future plans for expansion are outlined in the future work portion below, as technologies that have developed rapidly within the past six months can be used to change the front end of the approach completely. However, the open-ended questions in the survey led to some direct insights for improvement which would be impossible to obtain using objective measures or the numerical subjective questionnaire. The subjective reporting provides exclusive data on improvements which can be done without a complete overhaul of the framework. A few of the complaints expressed in questions 14, 17, or 18, along with simple-to-implement changes, are listed below:

- The slide speed was too fast and induced nausea.
    - The framework had a speed slider feature on the lefthand menu, allowing speed adjustment on the fly. This was replaced with automatic acceleration for a simpler UI. It can be reverted to the slider in future studies.
- No way to switch the main interaction to the left-hand dominant.
    - In the full dev system, switching the interaction hand is trivial; in future iterations the developer can introduce a button in the starter room to select the dominant interaction hand.
- A bug in the optional MRI demo lead to the instructions doubling over themself (single-user encounter)
    - A simple solution is to use a do-once block on the audio with a reset trigger so that repeating the gesture or flickering gesture recognition does not keep repeating the audio. This functionality was included in the consultation room, which was mitigated from the demo, and was immediately ported to the demo after bug discovery.
- Spiderman gesture recognition is unforgiving, negatively impacting those with lower hand flexibility.
    - Several potential solutions to this issue are already built into the framework and can easily be implemented.
        - The system has a plethora of gesture options available to choose from; changing the trigger to something simpler is trivial. The

workflow also results in a very simple way to generate new gestures.

        - The program can be configured so each user is asked to form the gesture they would like to use to slide. This can be recorded allowing for the functionality to be customized to the test subjects comfort. This way, the slide movement can be triggered easily based on user preference.
        - As mentioned in the development section, the framework (and workflow) has easy access to the gesture detection threshold value. Although it does not need to be adjusted on the fly, the value can be lowered. This may also reduce hand-tracking inconsistency.

- Height bugs are occasionally present, where each time the slide gesture is detected, the player's height increases.
    - This bug deserves further study to uncover the exact cause, but a theorized solution is to set the pawn's height and the tracking origin each time.
- A low-light environment negatively impacts tracking accuracy.
    - It is not something that can be solved through software, but an alternative solution is to allow the user to choose between hand recognition or controllers, as the lights on the controller allow it to be detected in darker environments.
- Sometimes the instructions are forgotten, misheard, or missed altogether.
    - Adding a gesture or button to repeat the last instruction can be implemented. It could be accessed through the offhand dropdown menu present in the system but not the demo.
    - It can also include subtitles with instructions, which is helpful for the audibly impaired.
    - Use animations to show hand gestures instead of still images for clarity.
- Some landmarks missing from the environment, such as Tim Hortons in the Atrium and plants in certain locations, help orient those visiting Victoria Hospital.
    - Although these assets are not present in the system, adding them using the same workflow is simple.

- It feels empty with nobody else.
    - Although it is outside the scope of this project workflow, the solution will be discussed in future work.
- Different locomotion systems affect people in different ways. Most users that felt some degree of motion sickness credited the slide movement, although one subject felt it during a teleport.
    - For the sake of the demo, using a mandatory movement style was only required for the test comparing the two styles, but for future demos that allow it, let the user pick which two movement methods they want access to like in the navigation trial.

Question 15 from the survey asked which elements of the demo they enjoyed and asked for any general positive feedback, which is listed below with developer comments:

- Impressive visual fidelity
    - Although this was universally praised, further improvements are discussed in the future work section.
- Clear tutorials with simple, responsive interactions with door touch and contact with the reception counter.
    - The first portion is subjective, as some users felt some needed to be repeated, so that would be a future addition.
    - The use of trigger boxes for the interactions rather than physics-based handle turns and door openings were done for performance, so it is positive that it was well received.
- Good performance, especially using a mobile setup.
    - Discussed in the development section, but the overall goal would be to shift the demo fully to the quest standalone rather than tether or air link to a laptop.
- Limiting to 2 movement methods is good as it does not overcomplicate the simulation but allows users to choose what they prefer for the specific use case.
    - The two movement methods are present in the demo as they are evaluated. A future iteration can pull another movement method (e.g. virtual joystick)

from the system and allow the user to choose which two they want to use prior to starting.

- Recreation feels faithful to a real hospital with added details like fire alarms.
    - Done by design and to allow for landmarking, but future work discusses how to add more realism.
- The teleportation locomotion is intuitive to learn and comfortable to use.
    - It was hypothesized that teleport movement would be the overwhelming favourite, but a small subset of users did prefer sliding.
    - The gesture is a common one used in the industry which is why it is the gesture of choice in the plugin.
- Enjoy having the maps on the wall, making the user create a mental path. Provides confidence in real-life visitation.
    - This contradicts one user's recommendation to include a live updating mini map.
    - If a different simulation is made where having users learn the environment is not the goal, then I would consider adding it.

In addressing the last user comment, research suggests that when humans rely heavily on GPS and mini maps for navigation, it may impede people's ability to develop a sense of direction and spatial awareness (Berki, 2022). Specifically, step-by-step navigation provided by GPS and mini maps can lead people to become overly reliant on them rather than developing a broader understanding of their surroundings and the spatial relationships between different landmarks and features (Seminati et al., 2022). As a result, individuals may struggle to navigate the hospital effectively when they visit it in real life as there is a lack of maps even on the walls at Victoria Hospital. Using stationary virtual maps instead of live GPS is meant to promote the development of broader spatial knowledge, cognitive maps, and orientation skills in virtual space for when they are needed to navigate independently in a real-world visit (Yamazaki et al., 2020).

Another issue regarding hand tracking was the appropriate hand distance and angle in front of the face for optimal recognition. The Quest 2 has cameras mounted at each front corner of the face of the headset, so if the hand was held too close to the body or at too straight of an angle, the system could not track it properly.

5.4 « Challenges Faced and Limitations of the Study »

The current study, while innovative and insightful in its approach and findings, encountered several challenges and limitations that warrant discussion. Notably, the first four limitations are largely attributed to the global COVID-19 pandemic, which restricted the scope of the study in various ways.

1. **Limited Diversity of Subjects During Development:** Despite the project demo's positive reception from test subjects, who appreciated the offering of two distinct movement methods (or three, including room scale), it became evident that the activation gestures needed refinement. The sliding gesture, for instance, should be simplified or have an easier detection threshold. For the teleport function, one user reported fatigue from the repetitive finger motion. Unfortunately, these issues were not identified during development due to testing on a small, homogeneous group of users who did not experience any difficulties.

2. **Limited Sample Size for Demo Testing:** The study's findings are based on a relatively small group of participants. While their feedback provided valuable insights, the findings might not be fully representative of a larger population. Future studies should strive to include a larger sample size to further validate these findings and ensure broader applicability. The small sample size also limited the range of susceptibility to motion sickness, which could provide a broader perspective on this important issue with a larger group of participants.

3. **Restricted Demographic Range of Demo Test Subjects**: The participants in this study were gathered from the developers contact circle and were not overly diverse in terms of age, physical abilities, and technical expertise. This lack of diversity may have influenced the study's findings. Future research should aim to include a wider range of participants to ensure more comprehensive data analysis and to aid in design improvements. Specifically, a follow-up study should aim to test more members from each age range, as the youngest participant in this study

was 20 years old and it would be beneficial to test with a greater number of teenage participants. Furthermore, a more diverse group of test subjects would help better understand the occurrence and mitigation of motion sickness across different demographic and ability groups.

4. **Lack of On-Site Data:** Due to pandemic-related restrictions, the researcher was unable to gather extensive on-site data from the LHSC Victoria Hospital campus. This has limited the ability to create a fully accurate and detailed VR simulation. Future research would benefit from more thorough fieldwork, capturing subtle yet important details of the hospital environment to enhance the realism and utility of the simulation. Ideally, this would involve the developer visiting the site to take accurate measurements and capture additional images and videos. Additionally, with more on – site data collection opportunities, it would have been possible to model more areas of the hospital and start the simulation from different entrance points rather than the atrium to more accurately simulate a visiting patient.

5. **Inability to Incorporate Voice Commands:** The current version of the VR simulation did not incorporate voice commands as originally planned, due to technical limitations with implementing voice recognition in a VR headset using UE 4. While hand gestures proved effective for navigation, the inclusion of voice commands could have made the simulation more accessible and intuitive, especially for users who might struggle with hand gestures. Additionally, voice commands could be used to trigger additional features such as the repeat last instruciton functionality described in the previous subsection. This remains a potential area for improvement and exploration in future research.

6. **Technical Limitations of the VR Device:** The Quest 2 VR device, while affordable and accessible, had certain limitations. Specifically, hand tracking proved problematic if the user's hand was held too close to the body or at an overly straight angle. This may be improved on the next Quest headset. Additionally, the headset lacks a color external camera that can be accessed through an Unreal Engine program. If the headset supported color passthrough, the AR capabilities could be used provide in simulation real time feedback for users' hand gestures and allow users to navigate to the surface for lying down

without needing to remove the headset or being helped by the researcher. These technical constraints could potentially impact the user experience and should be taken into consideration in future iterations or studies using different VR devices.

Each of these challenges and limitations presents a unique opportunity for future research and development in the field of VR simulations for hospitals. Despite these hurdles, the study has laid a firm groundwork for the development of practical, cost-effective VR solutions for healthcare facilities which can be used and developed by researchers, medical staff, and programmers to the benefit of both clinicians and patients.

5.5 « Prospects/Impact of the Study »

This thesis advances the development and application of virtual reality (VR) in healthcare environments by addressing the lack of standardization across platforms (Eagleson et al., 2014). Pioneering work is presented in leveraging pre-existing tools to develop a user-centric, easily adaptable, low-cost VR hospital framework that makes an effort to minimize the risk of inducing motion sickness. The research outcomes bridge the gap between VR technology and its practical implementation in healthcare, while the proof of concept demo specifically tests using the framework to design software for teaching gestures, comparing movement methods, improving hospital navigation, and providing a comfortable VR simulation for an MRI.

Valuable insights into the dynamics of VR locomotion methods and their impact on user comfort and navigation performance are delivered through this research, contributing to the existing literature in the field. By focusing on reducing artificial locomotion and optimizing environments to reduce latency, the study emphasizes minimizing motion sickness as a priority in VR design. The framework employed strategies such as providing several locomotion options, gaze-based turning, and encouraged self-paced interactions, thereby reducing sensory mismatches and increasing user comfort. This reinforces the critical role of user comfort and intuitive HCI implementations for successful VR adoption, as found in previous studies on motion sickness in VR (Conner et al., 2022; Ferche et al., 2015; Grassini et al., 2021).

One of the study's key contributions is the development of a scalable, modular, and easily deployable VR hospital framework using Blender and Unreal Engine. This cost-effective and user-centered approach to VR development provides a standardized platform that can be adapted and expanded by individuals or small teams. By detailing the workflow for creating accurate virtual environments for framework expansion, this study equips medical professionals and researchers to join the design team, opening avenues for designing VR programs for surgical planning, remote training, site orientation, and pre-op patient education.

The demo serves as an example of adapting the developed VR hospital framework for various scenarios based on specific needs, demonstrating the framework's versatility and applicability across multiple healthcare and research contexts. The simulation trials successfully showed that users can familiarize themselves with hospital environments and procedures, which can significantly reduce anxiety. This opens up opportunities for future research and development, paving the way for a wide range of VR applications in healthcare.

The research emphasizes the value of user feedback as part of the iterative design process through the post-demo survey. Design adjustments based on this user experiences can minimize motion sickness symptoms and maximize user immersion.

As many of the external papers researched in the study mentioned cost being a significant barrier to brad adoption, the study's economic impact is significant. The cost breakdown indicates that the VR hospital framework was developed at a fraction of the cost compared to off-the-shelf or custom-developed VR simulators. By leveraging free tools like Blender and Unreal Engine, and affordable hardware like the Oculus Quest 2, this study provides a roadmap for developing high-quality VR simulations on a budget, further opening access to VR in healthcare, particularly for smaller hospitals or research groups with limited resources.

In conclusion, this study offers valuable contributions to the field of VR in healthcare by providing a concrete tool in the form of the VR hospital framework and a method for building environments to expand it beyond its current capabilities. The learnings from this study provide a strong foundation for future research and development, paving the way for more accessible, efficient, and comfortable VR experiences, ultimately aiming to improve patient care and professional training in healthcare settings.

Chapter 6

6 « Future Work and Conclusion »

This chapter discusses the potential enhancements and future applications of the VR hospital framework established in this thesis. It navigates through the scope of refining the current system based on user feedback and the prospects for further studies focusing on locomotion, navigation, and interaction accuracy. The discussion then shifts to potential improvements in the development process, encompassing advanced technologies and tools that can create more accurate and immersive VR environments. Finally, it presents an ambitious vision for the future, outlining the potential development of an interconnected digital universe that expands upon the work in this thesis. This vision encourages diverse methods of interaction, user-generated content, and thoughtful consideration of the ethical implications that such an environment would entail. The chapter sets the stage for the conclusion, summarizing the key findings and contributions of this thesis.

6.1 « Future Improvements and Applications »

The current VR framework, while effective, presents opportunities for further enhancement, particularly in response to user feedback. Notably, the system could benefit from the development of a self-sufficient feedback loop to teach new users gestures without external intervention. In the current demo, although the software briefly explains them, researchers need to personally guide subjects through the gestures by providing corrections, a process that could be automated for greater efficiency. An advanced system could provide appropriate feedback, instructing users on how to correct their hand position. As mentioned in earlier sections, an example of this situation was the spiderman gesture used for sliding. Several users were unaware that their little and index fingers weren't straight out which resulted in the gesture remaining undetected.

Initial solutions to this could involve recording several partial gestures reflecting common mistakes observed in the testing for this demo. If the system detects one of these incorrect poses for a certain duration, it could then provide pre-programmed feedback to the user. While this approach may not account for all possible gesture mistakes, it is a promising step toward improving user interaction. Another potential solution involves tracking the bone and socket positions, providing pre-recorded feedback based on how the values for the successful gesture differ from the users' current hand pose. Such a system could even provide a virtual translucent hand "mold" to guide users in achieving optimal gesture positioning. This feature, however, would benefit from a headset with color passthrough to allow the user to view their live hand along with the example instead of the systems representation of their hand.

Beyond these functional improvements, there are various future applications for the VR framework. It can be utilized for additional research on locomotion and navigation assistance methods, contributing to the understanding of effective VR movement strategies. Similarly, the mechanics for gaze and location tracking used in this study can be extended to locate the user's hands in virtual space. This system would use the same mechanics to log the X, Y, Z positions of each hand relative to the pawn camera and calculate their distance from interactive objects such as light switches, buttons, or medical instruments to name a few. Additionally, the program can be used to measure navigation accuracy along with time taken and traversal speed during trials like the second. If several paths are recorded for the routes to get to the goal, time spent "off the path" can be used as a measure of accuracy to determine how often, and how long user's got confused while navigating and strayed down incorrect hallways. This data could then be used to assess a user's interaction accuracy based on Fitt's Law and also be used to evaluate the layout of the hospital. The latter test carries implications for how this framework can apply to hospital planning to test layouts prior to building.

6.2 « Future workflow »

Refinements to the current workflow promise exciting advancements in VR healthcare simulations and further streamline the environment creation process while improving photorealism at a lower computational cost than photogrammetry. Environment data capture could be performed using a 360 camera such as the Insta360 one-inch, gathering information in all directions simultaneously, providing developers more footage to reference or using it directly with Instant Neural radiance field (NeRF) technology. NeRF is a technique that generates 3D representations or point clouds of an object or scene from images by using advanced machine learning.

The resultant NeRF scans could be imported into Blender for refining meshes and assets, followed by Unreal Engine to apply mechanics. Such an approach would optimize the environment creation pipeline (Figure 6-1) while retaining high detail, enabling more immersive and realistic simulations. Using this method also delivers more manipulable assets than those created through photogrammetry.

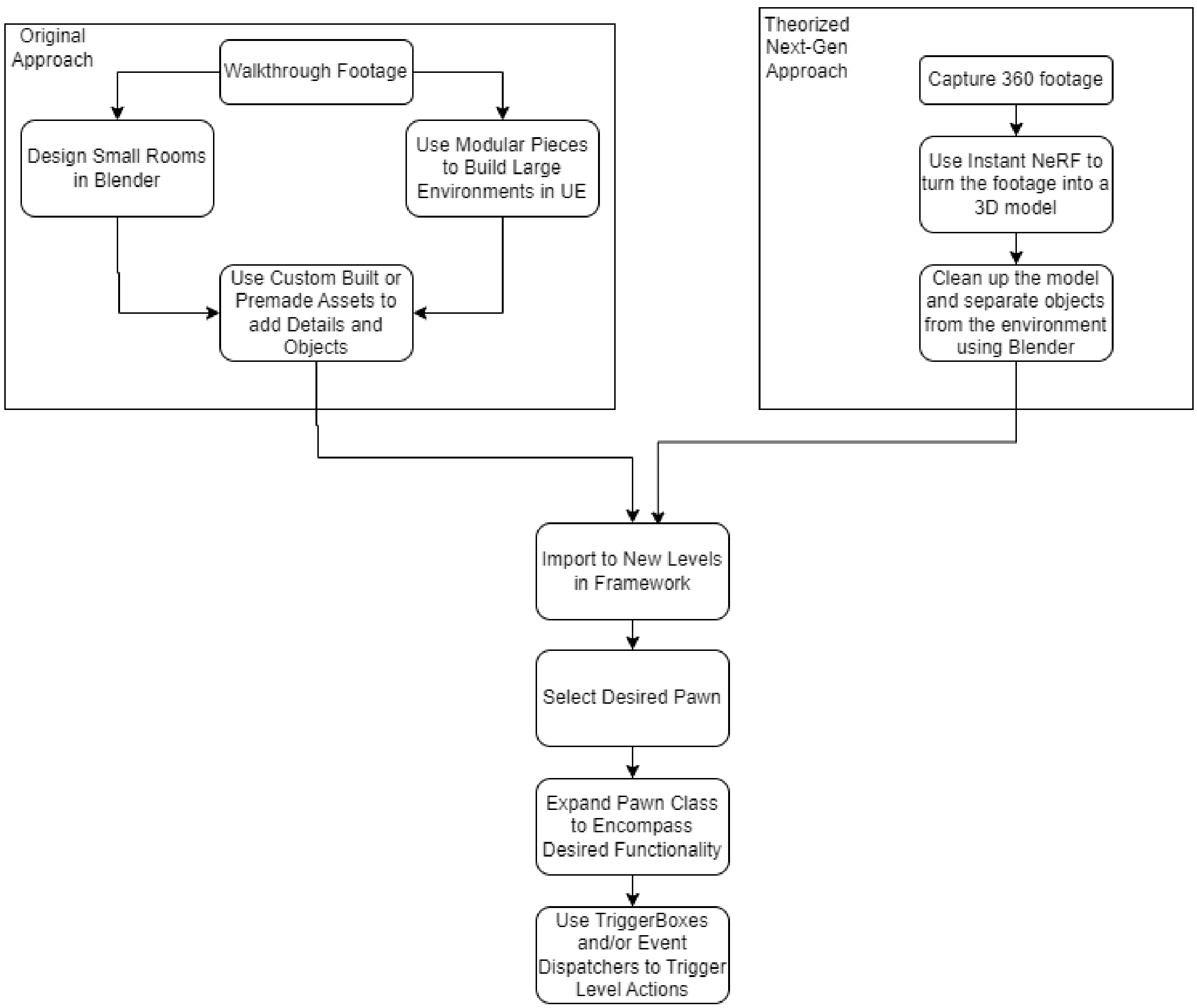


Figure 6-1 Flowchart Showing Framework Expansion with Theorized Method Included

The study also anticipates a shift from UE 4.27 to UE 5.2 for development, which promises significant enhancements. With its improved navigation mesh approach, Nanite and Lumen optimization, and extra development tools, Unreal Engine 5 can optimize the environment effectively while retailing greater detail.

To address one of the complaints about the environment feeling empty, the inclusion of Epic Metahumans, can provide realistic avatars, enhancing the sense of presence and social connection in the VR environment. Furthermore, a mix of pre-recorded and generational AI-based conversational avatars could further help combat the feelings of emptiness, enhancing user immersion. The scope of interactions and applications within the VR environment could also be expanded, including the implementation of the Metahuman avatars to replace disembodied voices for instructions, thereby creating a more engaging and intuitive user experience.

6.3 « Developer's Future Plans Based on Study Findings »

Looking ahead, the developer aims to link these environments, initiating an open-source, potentially P2P metaverse powered by Unreal Engine 5 marketed as "The Flip". The Flip is envisioned to transcend XR mediums and be universally accessible, whether through a traditional flat screen device or an XR-powered one, thereby promoting a merged social and interactive experience. Drawing upon the advancements in technology, such as the Unreal Editor for Fortnite (UEFN), The Flip aims to facilitate creation and access across these various platforms. However, in contrast to UEFN, which is designed for use with Fortnite avatars, The Flip will enable those joining via virtual setups to craft EPIC Metahumans as their characters, significantly enhancing the realism of the experience. Despite the promising vision, it also beckons critical ethical considerations that need to be meticulously studied. Nonetheless, the potential for The Flip to be more accessible than current Metaverse attempts, allowing all users to contribute, regardless of their possession of a headset, presents exhilarating prospects for the future of VR in healthcare and other sectors.

The various improvements, enhancements, and future plans highlighted in this section represent the potential trajectory of this field. They underscore the continuing evolution of VR in healthcare, its capacity to improve user experiences, and its ability to transform our understanding and application of these powerful technologies.

6.4 « Conclusion »

Rapid advances in virtual reality technology will continue to drive its adoption in various industries, including architecture, manufacturing and healthcare. Innovations in VR hardware and 3D development software are a major reason for this widespread adoption as the deeper immersion enabled by high fidelity capabilities encourages VR use for training, education, and therapeutic intervention, just to name a few. Currently, the modular VR simulation frameworks required for these functions are costly due to the high level of expertise required for custom development and maintenance. However, the recent explosion in video game popularity has led to powerful development tools becoming available at little to no cost.

This thesis demonstrated the potential for using free development tools like Blender and Unreal Engine to create a modular VR simulation framework that can be used in a variety of settings. A VR hospital simulation was developed to showcase this potential, and to highlight other important characteristics like flexibility and expandability. In order to do this, focus was placed on simulation scalability and key elements of intuitive system design, including navigation, interactivity, and sensation recreation. This will enable future researchers to not only adapt the hospital framework to a wide variety of clinical scenarios but also to use the system design elements for entirely different applications. The development process allows for the creation of realistic environmental representation with accurate spatial dimensions, and familiar hospital scenarios. As new technologies emerge and user requirements evolve, the framework's modularity will ensure its versatility and relevance.

An objective-driven demo was created to help show the adaptability and flexibility of the hospital framework. It exemplifies how the system can be easily customized to gather vital information and run dedicated trials for various use cases within the virtual hospital environment. A series of such trials were executed, and focused on gesture learning, followed by movement method comparison, navigation testing, and concluded with an

MRI simulation. A post-demo survey gathered valuable data on user demographics, prior experiences, and general feedback on the demo. Information like this is essential for refining the framework and ensuring its continued value for research and for medical professionals.

Despite the limited study size of 13 participants due to the COVID – 19 pandemic, the demo successfully shows the hospital framework's effectiveness for data capture and experimentation. The framework was developed using an affordable headset, the Oculus Quest 2, and is designed to allow researchers to easily modify, and expand on the original design to suit their specific needs. The 'Future Works' portion of this thesis also explores a simplified development workflow for room expansion using a combination of next generation technologies such as 360 capture and instant Neural Radiance Fields (NeRFs), which can partially automate the capture and modeling process. This new process should result in even more realistic and high fidelity environments at a much lower computational cost than photogrammetry. The study recognizes motion sickness as a major concern associated with VR, and addresses it in the framework by maintaining a high frame rate, low latency and offering the optimal "blink" animated teleportation locomotion option.

In conclusion, by thoroughly researching the development landscape, making this extensible framework for less than $100 of purchased plugins, and proving its adaptability through the objective-driven virtual Victoria Hospital demo, this thesis demonstrates that VR hospital simulations can now be developed using freely available resources like Blender and Unreal Engine without the need for expensive tools or specialized developers. The VR hospital framework in particular, paves the way for broader adoption of custom VR technology in healthcare, resulting in enhanced medical training, and ultimately improving patient experience.

References or Bibliography

Aziz, H. (2018). Virtual Reality Programs Applications in Healthcare. *Journal of Health & Medical Informatics*, *9*, 305. https://doi.org/10.4172/2157-7420.1000305

Azuma, R., Baillot, Y., Behringer, R., Feiner, S., Julier, S., & MacIntyre, B. (2001). Recent advances in augmented reality. *IEEE Computer Graphics and Applications*, *21*(6), 34–47. https://doi.org/10.1109/38.963459

Bacca-Acosta, J., Baldiris, S., Fabregat, R., Graf, S., & Kinshuk, D. (2014). Augmented Reality Trends in Education: A Systematic Review of Research and Applications. *Educational Technology and Society*, *17*, 133–149.

Banerjee, T. (2023). *Computer Vision-Based Hand Tracking and 3D Reconstruction as a Human-Computer Input Modality with Clinical Application*.

Barnett, S., Mills, I., & Cleary, F. (2022). Investigation of the potential use of Virtual Reality for Agoraphobia exposure therapy. *2022 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*, 898–899. https://doi.org/10.1109/VRW55335.2022.00300

Berki, B. (2022). Overview of the Relationship between Human Spatial Abilities and GPS Usage. *2022 IEEE 1st International Conference on Cognitive Mobility (CogMob)*, 000019–000022. https://doi.org/10.1109/CogMob55547.2022.10117612

Boeldt, D., McMahon, E., McFaul, M., & Greenleaf, W. (2019). Using Virtual Reality Exposure Therapy to Enhance Treatment of Anxiety Disorders: Identifying Areas of Clinical Adoption and Potential Obstacles. *Frontiers in Psychiatry*, *10*, 773. https://doi.org/10.3389/fpsyt.2019.00773

Bowden, J. L., & McNulty, P. A. (2013). The magnitude and rate of reduction in strength, dexterity and sensation in the human hand vary with ageing. *Experimental Gerontology*, *48*(8), 756–765. https://doi.org/10.1016/j.exger.2013.03.011

Carnegie, S. W. (2015). *Endoscopic Targeting Tasks Simulator: An Approach Using Game Engines*.

Chabal, C., & Soulabaille, Y. (2016). *Virtual Reality: A way to prepare and optimize operations in decommissioning projects*.

Chang, E., Kim, H. T., & Yoo, B. (2020). Virtual Reality Sickness: A Review of Causes and Measurements. *International Journal of Human–Computer Interaction*, *36*(17), 1658–1682. https://doi.org/10.1080/10447318.2020.1778351

Ciekanowska, A., Gliński, A. K.-, & Dziedzic, K. (2021). Comparative analysis of Unity and Unreal Engine efficiency in creating virtual exhibitions of 3D scanned models. *Journal of Computer Sciences Institute*, *20*, 247–253. https://doi.org/10.35784/jcsi.2698

Conner, N. O., Freeman, H. R., Jones, J. A., Luczak, T., Carruth, D., Knight, A. C., & Chander, H. (2022). Virtual Reality Induced Symptoms and Effects: Concerns, Causes, Assessment & Mitigation. *Virtual Worlds*, *1*(2), Article 2. https://doi.org/10.3390/virtualworlds1020008

de Ribaupierre, S., & Eagleson, R. (2017). Editorial: Challenges for the usability of AR and VR for clinical neurosurgical procedures. *Healthcare Technology Letters*, *4*(5), 151–151. https://doi.org/10.1049/htl.2017.0077

Eagleson, R., de Ribaupierre, S., Kapralos, B., Haji, F., Stroulia, E., & Dubrowski, A. (2014). Healthcare Training Enhancement Through Virtual Reality and Serious Games. In M. Ma, L. C. Jain, & P. Anderson (Eds.), *Virtual, Augmented Reality and Serious Games for Healthcare 1* (Vol. 68). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-54816-1

Ferche, O., Moldoveanu, A., Moldoveanu, F., Voinea, A., Asavei, V., & Negoi, I. (2015). *CHALLENGES AND ISSUES FOR SUCCESSFULLY APPLYING VIRTUAL REALITY IN MEDICAL REHABILITATION*. 494–501. https://doi.org/10.12753/2066-026X-15-073

Gallagher, A. G., Ritter, E. M., Champion, H., Higgins, G., Fried, M. P., Moses, G., Smith, C. D., & Satava, R. M. (2005). Virtual Reality Simulation for the Operating Room. *Annals of Surgery*, *241*(2), 364–372. https://doi.org/10.1097/01.sla.0000151982.85062.80

Grassini, S., Laumann, K., & Luzi, A. K. (2021). Association of Individual Factors with Simulator Sickness and Sense of Presence in Virtual Reality Mediated by Head-

Mounted Displays (HMDs). *Multimodal Technologies and Interaction*, *5*(3), 7. https://doi.org/10.3390/mti5030007

Grieves, M. (2015). *Digital Twin: Manufacturing Excellence through Virtual Factory Replication*.

Grieves, M. (2016). *Origins of the Digital Twin Concept*. https://doi.org/10.13140/RG.2.2.26367.61609

Hsu, E. B., Li, Y., Bayram, J. D., Levinson, D., Yang, S., & Monahan, C. (2013). State of Virtual Reality Based Disaster Preparedness and Response Training. *PLoS Currents*, *5*, ecurrents.dis.1ea2b2e71237d5337fa53982a38b2aff. https://doi.org/10.1371/currents.dis.1ea2b2e71237d5337fa53982a38b2aff

Huang, K.-T., Ball, C., Francis, J., Ratan, R., Boumis, J., & Fordham, J. (2019). Augmented Versus Virtual Reality in Education: An Exploratory Study Examining Science Knowledge Retention When Using Augmented Reality/Virtual Reality Mobile Applications. *Cyberpsychology, Behavior and Social Networking*, *22*(2), 105–110. https://doi.org/10.1089/cyber.2018.0150

Iglesias-Vázquez, J. A., Rodríguez-Núñez, A., Penas-Penas, M., Sánchez-Santos, L., Cegarra-García, M., & Barreiro-Díaz, M. V. (2007). Cost-efficiency assessment of Advanced Life Support (ALS) courses based on the comparison of advanced simulators with conventional manikins. *BMC Emergency Medicine*, *7*, 18. https://doi.org/10.1186/1471-227X-7-18

Indovina, P., Barone, D., Gallo, L., Chirico, A., De Pietro, G., & Giordano, A. (2018). Virtual Reality as a Distraction Intervention to Relieve Pain and Distress During Medical Procedures: A Comprehensive Literature Review. *The Clinical Journal of Pain*, *34*(9), 858. https://doi.org/10.1097/AJP.0000000000000599

Jasper, A., Cone, N., Meusel, C., Curtis, M., Dorneich, M. C., & Gilbert, S. B. (2020). Visually Induced Motion Sickness Susceptibility and Recovery Based on Four Mitigation Techniques. *Frontiers in Virtual Reality*, *1*, 582108. https://doi.org/10.3389/frvir.2020.582108

Jiang, S., & Verderber, S. (2017). On the Planning and Design of Hospital Circulation Zones: A Review of the Evidence-Based Literature. *HERD: Health Environments Research & Design Journal*, *10*(2), 124–146. https://doi.org/10.1177/1937586716672041

Jones, D., Snider, C., Nassehi, A., Yon, J., & Hicks, B. (2020). Characterising the Digital Twin: A systematic literature review. *CIRP Journal of Manufacturing Science and Technology*, *29*, 36–52. https://doi.org/10.1016/j.cirpj.2020.02.002

Kawamura, S., & Kijima, R. (2016). Effect of head mounted display latency on human stability during quiescent standing on one foot. *2016 IEEE Virtual Reality (VR)*, 199–200. https://doi.org/10.1109/VR.2016.7504722

Khor, W. S., Baker, B., Amin, K., Chan, A., Patel, K., & Wong, J. (2016). Augmented and virtual reality in surgery—the digital surgical environment: Applications, limitations and legal pitfalls. *Annals of Translational Medicine*, *4*(23), 454–454. https://doi.org/10.21037/atm.2016.12.23

Kim, W.-S., Cho, S., Ku, J., Kim, Y., Lee, K., Hwang, H.-J., & Paik, N.-J. (2020). Clinical Application of Virtual Reality for Upper Limb Motor Rehabilitation in Stroke: Review of Technologies and Clinical Evidence. *Journal of Clinical Medicine*, *9*(10), 3369. https://doi.org/10.3390/jcm9103369

Kishino, F., & Milgram, P. (1994). *Http://vered.rose.utoronto.ca/people/paul_dir/IEICE94/ieice.htm*.

Kockro, R. A., Killeen, T., Ayyad, A., Glaser, M., Stadie, A., Reisch, R., Giese, A., & Schwandt, E. (2016). Aneurysm Surgery with Preoperative Three-Dimensional Planning in a Virtual Reality Environment: Technique and Outcome Analysis. *World Neurosurgery*, *96*, 489–499. https://doi.org/10.1016/j.wneu.2016.08.124

Maas, M. J. (2017). *The Reality of Virtual, Augmented and Mixed Reality in K-12 Education: A Review of the Literature*.

Mao, R. Q., Lan, L., Kay, J., Lohre, R., Ayeni, O. R., Goel, D. P., & Sa, D. de. (2021). Immersive Virtual Reality for Surgical Training: A Systematic Review. *Journal of Surgical Research*, *268*, 40–58. https://doi.org/10.1016/j.jss.2021.06.045

McCauley, M. E., & Sharkey, T. J. (1992). Cybersickness: Perception of Self-Motion in Virtual Environments. *Presence: Teleoperators and Virtual Environments*, *1*(3), 311–318. https://doi.org/10.1162/pres.1992.1.3.311

McIntosh, C., Macario, A., Flanagan, B., & Gaba, D. M. (2006). Simulation: What does it really cost? *Simulation in Healthcare*, *1*(2), 109.

Monteiro, D., Chen, H., Liang, H.-N., Tu, H., & Dub, H. (2021). Evaluating Performance and Gameplay of Virtual Reality Sickness Techniques in a First-Person Shooter Game. *2021 IEEE Conference on Games (CoG)*, 1–8. https://doi.org/10.1109/CoG52621.2021.9619145

Napocensis, T., Besnea, F.-L., Ionescu, M., & Hamdan, H. (2021). *INTEGRATION OF A HAPTIC GLOVE IN A VIRTUAL REALITY-BASED ENVIRONMENT FOR MEDICAL TRAINING AND PROCEDURES*. https://www.semanticscholar.org/paper/INTEGRATION-OF-A-HAPTIC-GLOVE-IN-A-VIRTUAL-FOR-AND-Napocensis-Besnea/a70cb72eb92466f619b0a1c621499b5d9801ad82

Noble, S. M., Saville, J. D., & Foster, L. L. (2022). VR as a choice: What drives learners' technology acceptance? *International Journal of Educational Technology in Higher Education*, *19*(1), 6. https://doi.org/10.1186/s41239-021-00310-w

Palmisano, S., Allison, R. S., & Kim, J. (2020). Cybersickness in Head-Mounted Displays Is Caused by Differences in the User's Virtual and Physical Head Pose. *Frontiers in Virtual Reality*, *1*, 587698. https://doi.org/10.3389/frvir.2020.587698

Pandrangi, V. C., Gaston, B., Appelbaum, N. P., Albuquerque, F. C., Levy, M. M., & Larson, R. A. (2019). The Application of Virtual Reality in Patient Education.

Annals of Vascular Surgery, *59*, 184–189. https://doi.org/10.1016/j.avsg.2019.01.015

Peláez-Vélez, F.-J., Eckert, M., Gacto-Sánchez, M., & Martínez-Carrasco, Á. (2023). Use of Virtual Reality and Videogames in the Physiotherapy Treatment of Stroke Patients: A Pilot Randomized Controlled Trial. *International Journal of Environmental Research and Public Health*, *20*(6), 4747. https://doi.org/10.3390/ijerph20064747

Pöhlmann, K. M. T., Li, G., Mcgill, M., Markoff, R., & Brewster, S. A. (2023). You spin me right round, baby, right round: Examining the Impact of Multi-Sensory Self-Motion Cues on Motion Sickness During a VR Reading Task. *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, 1–16. https://doi.org/10.1145/3544548.3580966

Pottle, J. (2019). Virtual reality and the transformation of medical education. *Future Healthcare Journal*, *6*(3), 181–185. https://doi.org/10.7861/fhj.2019-0036

Seminati, L., Hadnett-Hunter, J., Joiner, R., & Petrini, K. (2022). Multisensory GPS impact on spatial representation in an immersive virtual reality driving game. *Scientific Reports*, *12*, 7401. https://doi.org/10.1038/s41598-022-11124-9

SenthilKumar, R. (2019). Work in Progress: Use of Interactive Simulations in the Active Learning Model in Physics Education for Engineering Students at a College in Oman. *2019 IEEE Global Engineering Education Conference (EDUCON)*, 1359–1362. https://doi.org/10.1109/EDUCON.2019.8725215

Simons, A., Wohlgenannt, I., Zelt, S., Weinmann, M., Schneider, J., & vom Brocke, J. (2023). Intelligence at play: Game-based assessment using a virtual-reality application. *Virtual Reality*. https://doi.org/10.1007/s10055-023-00752-9

Staples, J., Edwards, R., Pask, S., Singleton, H., Moran, J., Holley, D., & Rolfe, U. (2021). 64 360 Simulation: Assessing Baby Robin. *International Journal of Healthcare Simulation*, ijohsj_ijaa016.099. https://doi.org/10.54531/MKXW2760

Stunden, C., Stratton, K., Zakani, S., & Jacob, J. (2021). Comparing a Virtual Reality–Based Simulation App (VR-MRI) With a Standard Preparatory Manual and Child Life Program for Improving Success and Reducing Anxiety During Pediatric Medical Imaging: Randomized Clinical Trial. *Journal of Medical Internet Research*, *23*(9), e22942. https://doi.org/10.2196/22942

Talbot, T. B., & Chinara, C. (2022). Open Medical Gesture: An Open-Source Experiment in Naturalistic Physical Interactions for Mixed and Virtual Reality Simulations. *Human Factors in Virtual Environments and Game Design*, *50*(50). https://doi.org/10.54941/ahfe1002054

Theresa Pöhlmann, K. M., O'Hare, L., Föcker, J., Parke, A., & Dickinson, P. (2021). Is Virtual Reality Sickness Elicited by Illusory Motion Affected by Gender and Prior Video Gaming Experience? *2021 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*, 426–427. https://doi.org/10.1109/VRW52623.2021.00095 van der Kruk, S. R., Zielinski, R., MacDougall, H., Hughes-Barton, D., & Gunn, K. M. (2022). Virtual reality as a patient education tool in healthcare: A scoping review. *Patient Education and Counseling*, *105*(7), 1928–1942. https://doi.org/10.1016/j.pec.2022.02.005

Vohera, C., Chheda, H., Chouhan, D., Desai, A., & Jain, V. (2021). Game Engine Architecture and Comparative Study of Different Game Engines. *2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT)*, 1–6. https://doi.org/10.1109/ICCCNT51525.2021.9579618

Xiao, J. (2017). Learner-content interaction in distance education: The weakest link in interaction research. *Distance Education*, *38*(1), 123–135. https://doi.org/10.1080/01587919.2017.1298982

Xiao, X., Zhao, S., Meng, Y., Soghier, L., Zhang, X., & Hahn, J. (2020). *A Physics-based Virtual Reality Simulation Framework for Neonatal Endotracheal Intubation (Best Paper Award)*. *2020*. https://doi.org/10.1109/vr46266.2020.1581028031480

Xiong, J., Hsiang, E.-L., He, Z., Zhan, T., & Wu, S.-T. (2021). Augmented reality and virtual reality displays: Emerging technologies and future perspectives. *Light: Science & Applications*, *10*(1), Article 1. https://doi.org/10.1038/s41377-021-00658-8

Yamazaki, A., Hayashi, Y., & Seta, K. (2020). Model-Based Development of Spatial Movement Skill Training System and Its Evaluation. *IEICE Transactions on*

Information and Systems, *E103.D*(7), 1710–1721. https://doi.org/10.1587/transinf.2019EDP7230

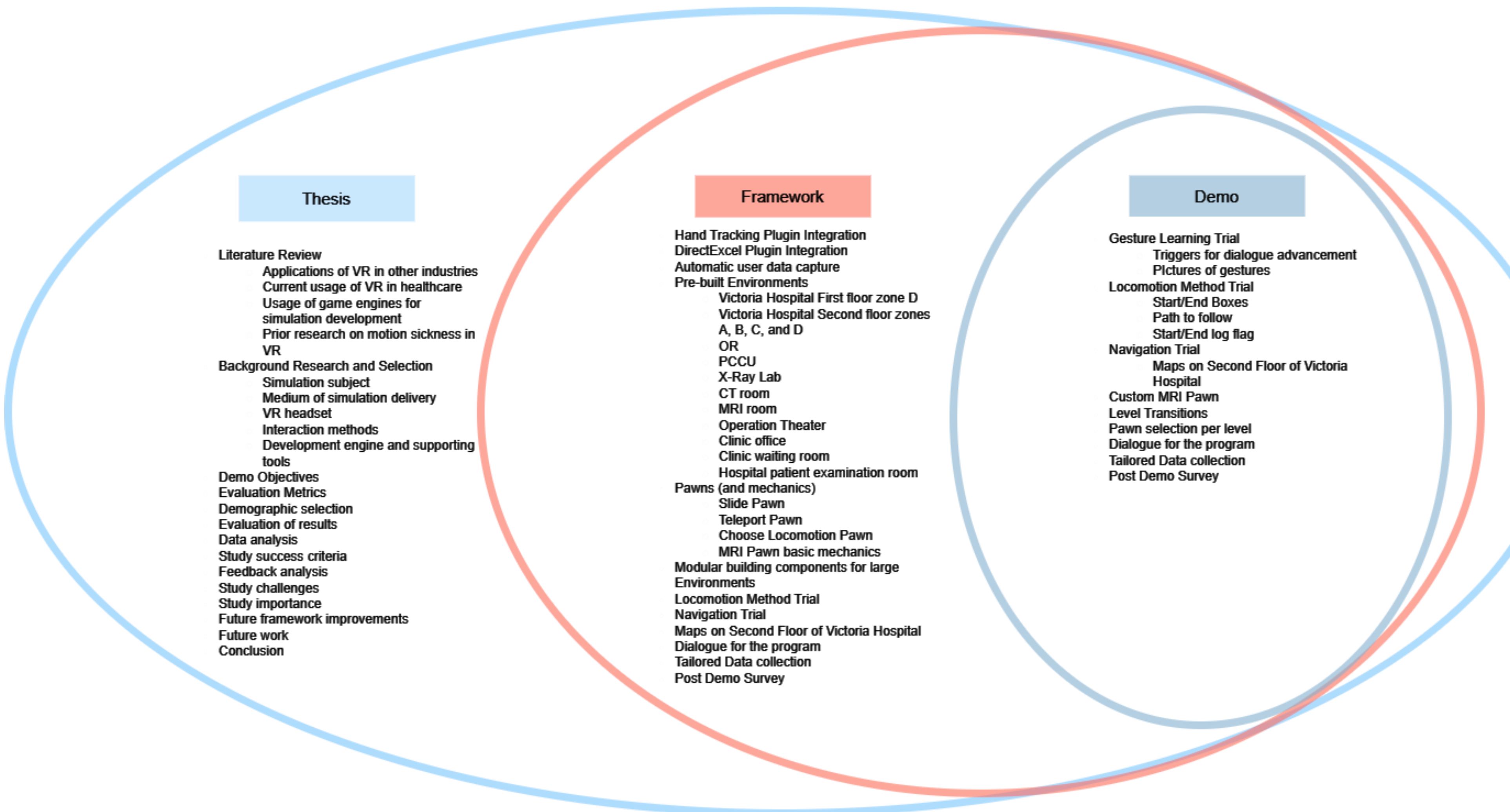
Yoshioka-Maeda, K., Honda, C., Sumikawa, Y., Okamoto, Y., Shimada, M., Fujii, H., Iwasaki-Motegi, R., Miura, T., & Otsuki, M. (2022). Developing a Virtual Reality Simulation Program for Improving Nursing Students' Clinical Reasoning Skills in Home Settings: A Protocol Paper. *Nursing Reports*, *12*(4), 968–979. https://doi.org/10.3390/nursrep12040093

Zertuche, J.-P., Connors, J., Scheinman, A., Kothari, N., & Wong, K. (2020). Using virtual reality as a replacement for hospital tours during residency interviews. *Medical Education Online*, *25*(1), 1777066. https://doi.org/10.1080/10872981.2020.1777066

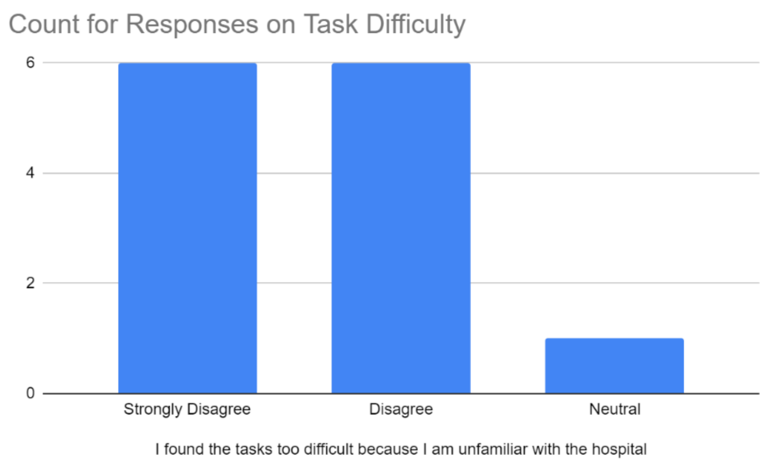
Zikas, P., Papagiannakis, G., Lydatakis, N., Kateros, S., Ntoa, S., Adami, I., & Stephanidis, C. (2020). Scenior: An Immersive Visual Scripting system based on VR Software Design Patterns for Experiential Training. *The Visual Computer*, *36*(10–12), 1965–1977. https://doi.org/10.1007/s00371-020-01919-0

Appendices

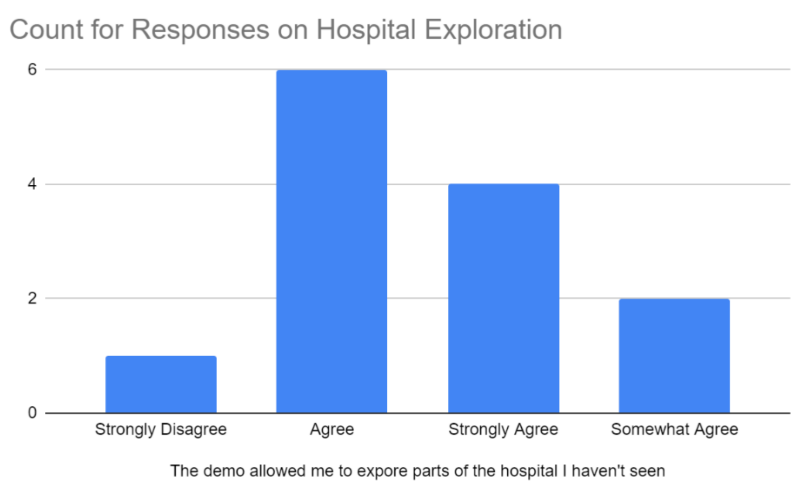
Appendix A Thesis, Framework, and Demo Components


Thesis
Literature Review
Applications of VR in other industries
Current usage of VR in healthcare
Usage of game engines for simulation development
Prior research on motion sickness in VR
Background Research and Selection
Simulation subject
Medium of simulation delivery
VR headset
Interaction methods
Development engine and supporting tools
Demo Objectives
Evaluation Metrics
Demographic selection
Evaluation of results
Data analysis
Study success criteria
Feedback analysis
Study challenges
Study importance
Future framework improvements
Future work
Conclusion
Framework
Hand Tracking Plugin Integration
DirectExcel Plugin Integration
Automatic user data capture
Pre-built Environments
Victoria Hospital First floor zone D
Victoria Hospital Second floor zones A, B, C, and D
OR
PCCU
X-Ray Lab
CT room
MRI room
Operation Theater
Clinic office
Clinic waiting room
Hospital patient examination room
Pawns (and mechanics)
Slide Pawn
Teleport Pawn
Choose Locomotion Pawn
MRI Pawn basic mechanics
Modular building components for large Environments
Locomotion Method Trial
Navigation Trial
Maps on Second Floor of Victoria Hospital
Dialogue for the program
Tailored Data collection
Post Demo Survey
Demo
Gesture Learning Trial
Triggers for dialogue advancement
PIctures of gestures
Locomotion Method Trial
Start/End Boxes
Path to follow
Start/End log flag
Navigation Trial
Maps on Second Floor of Victoria Hospital
Custom MRI Pawn
Level Transitions
Pawn selection per level
Dialogue for the program
Tailored Data collection
Post Demo Survey

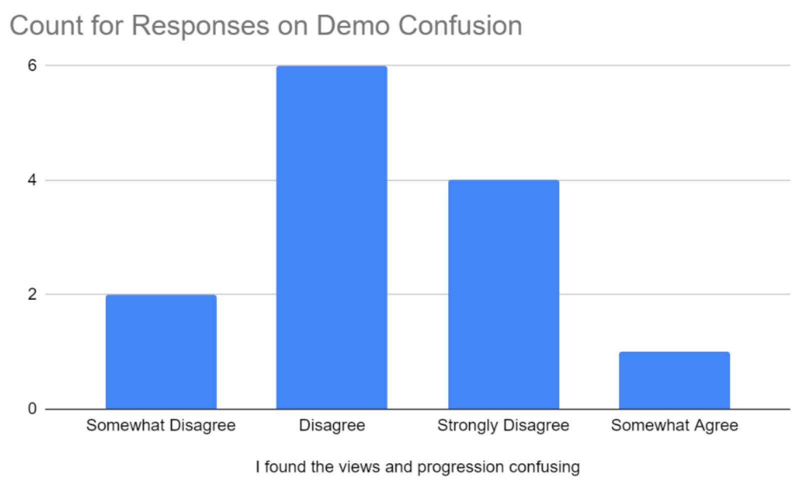
Appendix B Number of Responses for if the Users Found the Tasks too Difficult



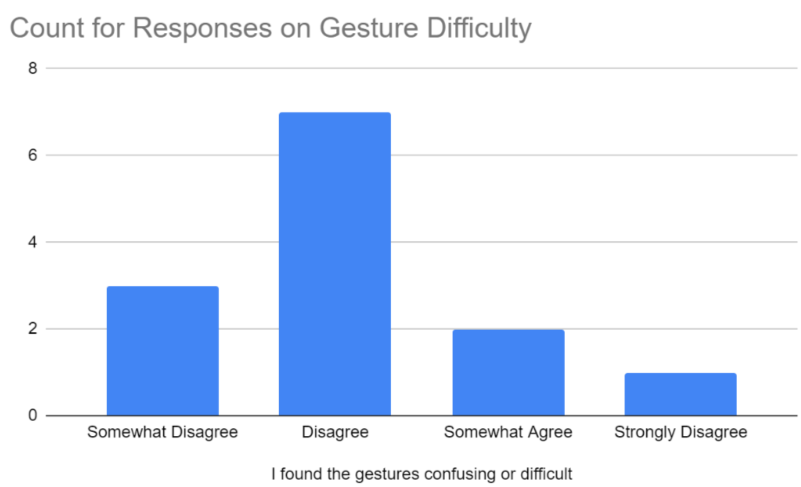
Appendix C Number of Responses for if the Demo Allowed Users to Explore Areas They Have Never Seen



Appendix D Number of Responses for if the Demo was Confusing



Appendix E Number of Responses for if the Users Found the Gestures too Difficult

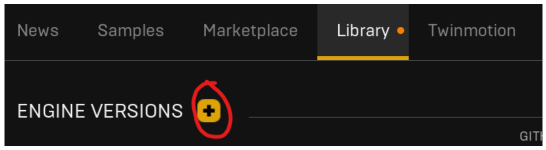


Appendix F Guide for Opening Framework and New Level Creation

Opening the framework file on your system

1. **Copy the base Unreal Engine project folder onto your system**
2. **Open Epic games launcher and ensure an account has been made and that you are signed in**
3. **Install Unreal Engine 4.27 from the library tab in the Unreal Engine section**

a. Click the orange and black '+' beside "Engine Versions



b. Select 4.27 as the install version and set install location

4. **Ensure the correct plugins have been purchased**
    a. From the marketplace tab, search for the "DirectExcel" and the "Hand Tracking" plugins from the following links:
        i. https://www.unrealengine.com/marketplace/en-US/product/directexcel
        ii. https://www.unrealengine.com/marketplace/en-US/product/hand-tracking-plugin
5. **Find the folder and open the .uproject file with Unreal engine**
    a. You may need to associate .uproject files with Unreal Engine if this is the first time opening a project, enabling you to start the editor by simply double-clicking on the file.
        i. Initiate the file association by executing *UnrealVersionSelector.exe /fileassociations* from your engine binaries directory. Please note that the exact name might vary based on your platform.
        ii. Execute *UnrealVersionSelector.exe* without any extensions to register your engine installation.

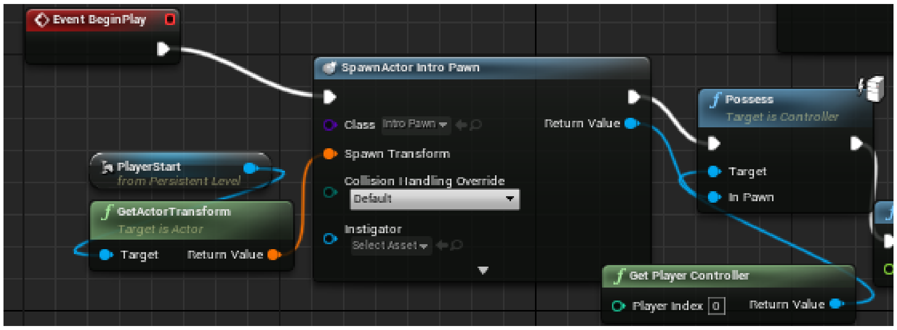
Creating a New Level/Experience in the VR Hospital Simulation

1. **Starting a Level**:
    - **From an Existing Level**: Duplicate an existing level (e.g., First floor, Second Floor, MRI room) and use it as a base.
    - **From Scratch**: Create a new level within Unreal Engine.
2. **Designing the Environment**:
    - **New Experience in an Existing Level**: Remove any unwanted assets.
        - Redraw Nav mesh if necessary.

- **From Scratch**: Place a plane on the "ground" and texture it with your floor plan.
    - Use the modular wall components and other assets provided in the framework to lay out your desired area.
    - Ensure to lay down a nav mesh so the pawns can navigate through the level.
- **For both**: If needed, you can also import new assets into the project to be used in your design.
- Make sure that all new objects have properly set up collisions to avoid any unintended interactions.

3. **Setting Up Player Movement**:
    - Choose one of the prebuilt pawns for movement to posses in the level blueprint:
        - **Slide Locomotion**
        - **Teleport Locomotion**
        - **Both Slide and Teleport Locomotion**

    - If you want to add additional custom capabilities to a pawn, follow these steps:
        1. Inherit from one of the base pawn classes.
        2. (using sample pawns as a reference) Add components for the desired functionality (grab, laser pointers, hand menus).
        3. (referring to instructions in chapter 3) Create any custom gestures to trigger desired functionality and link to the corresponding event.
        4. If emulating a scenario, set up any motion or interaction between the level and pawn blueprints (copy from sample pawns or follow standard UE interaction development)
4. **Location Logging**:
    - Each pawn is set to log its location, gaze, and movement type to an Excel workbook through the DirectExcel plugin. Test to ensure this functionality is active and properly set up. If problems arise, refer to sample pawn structure and modify "Save Player Data" function and function triggers.
5. **Level Blueprint Adjustments**:

- Adjust the level blueprint to handle new pawn spawning and possession. You can copy this functionality from one of the sample levels (like the floors, MRI room, OR, etc.)
- 

6. **Final Checks**:

    - Test the level to ensure that all functionalities are working as intended, especially the movement and location logging.
    - Double-check all collisions and nav meshes to avoid any unexpected issues.

7. **Documentation**:

    - It's always a good practice to document any changes or specific functionalities you've added. This helps future designers understand the specifics of your level.

Curriculum Vitae

Name:	Virat Krishna Tripathi
Post-secondary Education and Degrees:	University of Western Ontario London, Ontario, Canada 2016-2020 B.E.Sc.
	The University of Western Ontario London, Ontario, Canada 2021 - Present M.E.Sc.
Related Work Experience	Teaching Assistant The University of Western Ontario 2020-2022
	Research Scientist Nuclear Innovation Institute 2020 - Present