

# Final Report:

## Investigating Factors Affecting User Immersion in Virtual Reality Applications

Module CM2303 –

One Semester Individual Project

40 Credits

Module Leader – Dr Frank Langbein

*Author - Alexander Scott (Student Number 1738162)*

*Supervisor - Charith Perera*

*Moderator – Nervo Verdezoto Dias*

*Word Count – 17975*

*Word Count (Body Only) – 16931*

## Contents

Table of Figures.....	3
Abstract.....	5
Acknowledgements.....	5
Section 1 – Introduction .....	6
Section 2 – Background and Related Work .....	7
Section 3 – Specification and Designs.....	13
Technical Requirements.....	13
User Requirements.....	14
Concepts and Designs .....	15
Architecture and Development.....	27
Section 4 – Implementation.....	30
XR Toolkit.....	30
Rigging and Animating Models .....	37
Custom Scripts.....	45
Questionnaire Content.....	50
Application Distribution .....	50
Section 5 – Results and Evaluation .....	55
Questionnaire Results .....	55
Requirements Testing .....	67
Results Evaluation .....	70
Section 6 – Future Work .....	71
Section 7 – Conclusions .....	72
Section 8 – Reflection on Learning .....	72
Bibliography .....	74

## Table of Figures

Figure 1 - Development Flowchart .....	16
Figure 2 - User Representation Concepts .....	17
Figure 3 - State 1 Final Model .....	18
Figure 4 - State 2 Final Model .....	19
Figure 5 - State 3 Final Model .....	19
Figure 6 - Interaction Concept 1 .....	20
Figure 7 - Level 1 - Third Person .....	21
Figure 8 - Level 1 - First Person .....	21
Figure 9 - Interaction Concept 2 .....	22
Figure 10 - Level 2 - Third Person .....	23
Figure 11 - Level 2 - Third Person .....	23
Figure 12 - Level 2 - First Person .....	23
Figure 13 - Interaction Concept 3 .....	24
Figure 14 - Interaction Concept 3 Alternate View .....	24
Figure 15 - Level 3 - First Person .....	25
Figure 16 - Level 3 - Third Person .....	25
Figure 17 - Questionnaire Concept.....	26
Figure 18 - Questionnaire Instructions .....	27
Figure 19 - Questionnaire Implementation .....	27
Figure 20 - Scene Structure in Unity .....	28
Figure 21 - Level Architecture in Unity .....	29
Figure 22 - Toolkit Plugin .....	31
Figure 23 - Depreciated VR Support .....	31
Figure 24 - XR Rig Default Setup .....	31
Figure 25 - XR Context Menu .....	31
Figure 26 - XR Ray Interactor .....	32
Figure 27 - XR Device Based Controller .....	32
Figure 28 - Socket Interactor .....	33
Figure 29 - Direct Interactor .....	33
Figure 30 - Rigidbody Default .....	34
Figure 31 - Grab Interactor .....	34
Figure 32 - Scene Teleport Area.....	35
Figure 33 - Teleport Area .....	35
Figure 34 - Teleport Provider .....	35
Figure 35 - Commented LocomotionController.....	35
Figure 36 - Movement Script Contents.....	36
Figure 37 - Movement Script Contents.....	36
Figure 38 - Movement Script Editor Window .....	36
Figure 39 - Smooth Provider .....	37
Figure 40 - Snap Provider.....	37
Figure 41 - Movement Script Contents.....	37
Figure 42 - Rigify MetaHuman Rig .....	38

Figure 43 - Level 3 Model Rigged with MetaHuman Rig .....	38
Figure 44 – Overview of Animating in Unity .....	39
Figure 45 - Animation Keyframes .....	40
Figure 46 - Blend Tree Overview.....	40
Figure 47 - HandAnimator Script in Editor.....	41
Figure 48 - HandAnimator Script .....	42
Figure 49 - HandAnimator Script .....	42
Figure 50 - Blend Tree Pinch Demonstration.....	43
Figure 51 - Blend Tree Grab Demonstration.....	44
Figure 52 - Joint Listener Script.....	46
Figure 53 - Joint Listener Script.....	46
Figure 54 - Screen Fade Method Call.....	47
Figure 55 - Screen Fade Implementation.....	48
Figure 56 - Screen Fade Co-Routine.....	48
Figure 57 - Scene Fade Alpha Calls .....	48
Figure 58 - Handedness Checks .....	49
Figure 59 - Handedness Checks .....	49
Figure 60 - Questionnaire Content 1 .....	51
Figure 61 - Questionnaire Content 2 .....	52
Figure 62 - Optional Content 1 .....	53
Figure 63 - Optional Content 2 .....	54



## Abstract

Virtual Reality (VR) in its current form has been rising in popularity since 2013, due to technological advances in the field and increased consumer accessibility in terms of price and compatibility. As VR expands further, a strong focus is emerging on the ideas of presence and immersion. These ideas and what they represent in respect to VR experiences establish a set of ever-moving goalposts for VR developers as consumer equipment evolves and grows.

This report details the development process of a short puzzle-based experience in VR, with the goal of determining how changing a user's level of interactivity with their environment can alter the feeling of presence in the created environment. The data gathered from the users of the application is then used to determine how users feel about the interaction methods and the immersion they feel when engaging with them.

The final application presented to users was ineffective in providing concrete correlations between the developed interaction methods and their sense of immersion, but the resultant data and overall limitations of the application provide grounds for determining improvements in the methodology used to investigate, as well as presenting additional data regarding factors of immersion not primarily investigated by the application itself.

## Acknowledgements

I would like to thank my supervisor for this project, Charith Perera, for providing me with guidance across this project as well as allowing for me to pursue my initial project plan, giving me the creative freedom to explore this project at my own pace whilst ensuring I remain on track.

I would also like to thank my friends and family for providing me with feedback along the project development process and keeping me driven. In addition, thanks are extended to the anonymous users who took part in the study. Finally, I would like to thank the many tutorials provided by members of the Unity and VR community that enabled me to undertake this project to begin with and have inspired me continue learning about VR and engage with it further.

## Section 1 – Introduction

The primary aim of the project was the creation of a short gameplay experience that offers the user multiple methods of environmental interaction within a set of puzzles designed around core environmental elements. Once the user experiences each of these methods, they are asked to decide which they would prefer to revisit, and this selection is used as the basis of a questionnaire assessing the user's immersion and engagement. The data collected would be used to evaluate the effectiveness of each represented method of interaction within the context of the application and discuss whether this data could be extrapolated to apply to other applications following similar principles.

To explore these ideas, a short game named "Warehouse" was designed and created, where the user is placed in an environment in which tasks centre around manipulating shipping containers in a warehouse. This game is a single-player experience and presents the user with simple colour coordination puzzles where they must move a container in to its respective area to continue. There are 3 different methods of performing this action, all of which intend to offer more direct interaction with the environment itself than the previous method. The overall design and aesthetic of the game is simplistic and industrial to provide a clear design language to the user while they engage with the application.

The intended audience for this specific iteration of the game is firmly rooted in PC VR headset users. Targeting a broader spectrum of headset users was intended to allow for a greater range of data points as opposed to allowing only one subset of PC VR users. Unfortunately, due to development constraints, this application is not targeting the emerging standalone VR headset market, and as such the limitation to PC VR only may cause a trend towards more experienced or financially invested users as opposed to more casual viewpoints. This requirement may also result in a trend in age groups experiencing the game. Whilst there is no target age set for the application itself, participating in the study requires users to be over the age of 18 due to the submitted ethical application not accommodating children.

As a result of the high requirements in equipment for users to be able to experience this game, the assumption being made that all users engaging with the game are both capable and aware of how to operate and safely use a VR headset. While the design of the game allows for extremely low performance requirements, the second assumption is being made that each user has at minimum a VR ready computer to run the game on.

This project involved first planning and designing the creation of an application for use with VR headsets. This was achieved by drawing inspiration from TETRIS and its focus on organisation and pattern recognition, and this subsequently evolved into the idea of utilising a warehouse and warehouse props to emulate organisational puzzles and provide a setting where the idea of sorting is not abstract. The idea was then further fleshed out utilising a VR-based 3D modelling tool, for previsualisation of the stages of the game's puzzles. Finally, the project moved to prototyping, development and final implementation using the Unity engine and C# scripting language. The application was then pushed to GitHub and

distributed to the public to gather anonymous user data regarding their experiences with the application.

This project also involved researching and determining available methods for assessing a user's immersion in a VR application, and then implementing this questionnaire into the VR application itself to make user participation smoother and easier. Once data was collected, it was collated to see if trends emerged between chosen interaction methods and levels of immersion in users.

No significant trends emerged except the presentation of heavily mixed responses to engagement with the environments presented, as well as the revelation of an extremely strong sense of presence users reported, even when actively disengaged from the virtual environment presented to them. These results provide grounds for speculation on other methods that could be undertaken to investigate factors of immersion in a more thorough manner.

## Section 2 – Background and Related Work

This section of the report presents an overview of the two main topics relevant to the project, Virtual Reality, and Presence and Immersion, then provide a further context for the reasons why this project was devised.

This section also evaluates existing or similar approaches to evaluating user presence in VR, and how these informed or influenced the direction taken with this project as well as listing constraints on the approach taken over the project timeline.

### Virtual Reality

The operational principles behind the way that VR hardware is currently designed mechanically, and as a result, aesthetically, can be traced back to NASA's VIEW headset system, developed around 1985, which uses a lens-based system to display images. The concept of using lenses in VR stems from the LEEP (Large Expanse, Extra Perspective) system, which used side-by-side pre-distorted images in conjunction with lenses to give users a sense of depth and a large field of view (FoV). This concept was then combined with digital displays to create the NASA VIEW system, giving the same sense of depth that the LEEP concept could with still images, and then apply the concept to a virtual environment rendered through a computer.

Describing these environments and interactions using the term 'Virtual Reality' was popularised back in 1987, by Computer Scientist Jaron Lanier, founder of VPL Research. This was one of the first companies to sell VR products, and as such some of the products sold by VPL Research have become synonymous with the idea of VR itself, such as the DataGlove, which was used as an input device or a method for tracking a user's finger positioning in basic poses, and the EyePhone, an early commercially available VR headset that builds upon the principles of NASA's own VIEW headset designs (n.a, 1990). The core idea of a virtual world that can be interacted with in an immersive and engrossing way to experience dates back as far as 1935, in which short story by Stanley Weinbaum describes a pair of 'goggles' allowing you to experience a movie in which "You are in the story, you speak to the

(characters) and they reply” (Norman, Unknown). The history of this technology essentially beginning as science fiction and growing over the course of almost a century to become a viable and accessible technology for many is a testament to the drive and desire of the pioneers of this technology to see it come to fruition.

Commercially, however, the history of VR has been troubled. Whilst maintaining itself as a point of interest in laboratory-based work for a great many years, the public’s interest in virtual reality had a surge and subsequent fall, which has been followed by a steady resurgence in the past few decades. Public interest was high in and around the years of 1991-1995, when Sega first announced that they would be making a VR headset compatible with large-scale arcade cabinet games and the Sega Mega Drive – a product extremely popular as a home entertainment system at the time. Unfortunately, the system only launched for arcade use due to the technical limitations of the Mega Drive, and all the fervour surrounding it dissipated. In 1995, VR experienced an overwhelming surge in coverage when Nintendo released the Virtual Boy, a headset styled device that used two displays to create an illusion of depth. Due to the way the perception of depth relied on the physical distance between displays, it depended on users to focus the device’s screens to a parallax level that was comfortable for them. Due to this design decision, it seems that many users and critics could not achieve the correct level of comfort, and as a result suffered from intense dizziness and nausea after using the device. The Virtual Boy stands as one of Nintendo’s biggest failures due to the design choices and motion sickness leaving a bad taste in the mouths of reviewers, and the failure of the device likely serving as one of the causes of public indifference towards the concept of VR for a long period of time. After it was discontinued, only enthusiasts and laboratories remained invested in attempting to further the technology between 1995 to around 2010.

The advances that have been made within VR are firmly placed within the last 10 years of development. This began in 2010 with the emergence of a prototype of the first consumer targeted device called the Oculus Rift. The Rift began as a modification to an existing headset by Oculus founder Palmer Luckey that allowed for a previously unachieved FoV of around 90 degrees. This came with its own set of issues regarding distortion of the image seen by the user due to the nature of the lenses, and id Software’s John Carmack took interest in helping solve this issue with Luckey to further the development of the headset and moved to join the company shortly after. The Kickstarter crowd-funded method of financing development of the Oculus Rift was successful when it launched in 2012, pushing the idea of virtual reality back into the public spotlight. Over the course of the Oculus Rift’s iterations and eventual release in 2016, other companies saw the opportunity to invest in the space due to the popularity of the crowdfunding campaign. To expand interest in production of VR hardware, Valve Corporation pursued low persistence display technologies; in 2013 making their findings publicly available for other companies to use in their headsets and subsequently developed OpenVR, a software development kit compatible with the headset that they launched in 2016 in partnership with HTC called the HTC Vive. Valve decided to take a very open approach to the industry and their findings to push the hardware further, whereas Oculus decided to make their own Software development kit based around their own hardware instead.

The release of the HTC Vive introduced 'Room-Scale' experiences, in which the user's hands and head were tracked with 6 degrees of freedom (DoF) within a predefined area. Allowing for freedom of physical movement, albeit within the length of the headset's tethered cable, this was a step forward for immersion within virtual environments, as previously most consumer VR experiences were sitting or standing experiences involving no movement. Oculus shortly followed suit in bringing 'Room-Scale' to their device with the release of their Touch controller system and improved tracking system to bring their system up to par with the Vive's tracking system. These two companies have ended up setting the standard in which most users will experience high-end VR going forward; a tracked area in which they can freely move, within the limits of the connection between the headset and the computer running it. Both headsets also shipped with internal displays boasting refresh rates of 90Hz and resolutions of 1080\*1200 pixels per eye, further enforcing the similarity in experience that consumer VR could provide until the introduction of different headsets to the space later. The original specifications are an acceptable baseline that have been improved upon continuously as time has progressed and as a result greatly enhanced the experience users can have with the medium.

### Presence and Immersion

The meaning of these terms when relating to VR is essentially the sense of physically being present in a non-physical environment, but both terms cover different areas of achieving the same goal of making the user feel like they are truly in the depicted environment.

Exactly which area each term each represents seems to be unclear, but the central idea is that there is a technological element and a psychological element present.

For example, in a presentation from Michael Abrash, from the VR team at Valve, a list of requirements needed to "create a sense of presence" contains the following: a FoV of 80 degrees or more, a screen resolution of at least 1080p, a screen refresh rate of around 95Hz, and precise tracking (Abrash, 2014). These factors all listed are not sole methods to improve a user's experience in VR but must instead be combined to be able to give the user the sense of being physically involved in their environment. These could all be considered barriers to presence in the sense that improving all these qualities would in time make it easier for the user's brain to believe it is somewhere else with only the sensory data that the hardware provides to it.

On the other hand, in a paper titled "Immersion and Presence" (Metre, 2005), the author posits that immersion is defined in technical terms when it comes to virtual environments, and that presence is "The sensation of being there", with a focus on the fact that the sensation is in fact only the user's belief that they are in a different place. This order of the terms is also stated in the personal blog of a VR developer (SkarredGhost, 2016) in that immersion is the technical aspect and is the hardware's ability at "tricking you" in to believing you are elsewhere through only sensory information. Presence is similarly defined here as being how you "feel yourself inside the virtual world".

In a different developer's own personal post (Adams, 2004) focusing on immersion in traditional games, they put forward the idea of there being "Three Types of Immersion": Tactical, Strategic, and Narrative, respectively.

Tactical immersion defined here as the point in which the user is absorbed in the gameplay and its elements themselves, sometimes referred to as "in the zone" by users. This style of immersion is often a result of allowing the user to make split second decisions that have a consistent flow and predictable result to them. This flow is critical, as disruption or abrupt changes to patterns can easily shake a user out of this state.

Strategic immersion is set out as more of a broad sense of engagement in terms of mental stimulation, driving the user to complete tasks to optimise the situations presented and achieve a win condition overall. This immersion often deeply engrosses the user in such a way that mechanical actions and their overall results are the primary focus of the user and other aspects of the experience are likely to be uninteresting or not engaged with at all.

Narrative immersion is described as the same sense of "drawn in" that a film or book may elicit from a user, where the driving force is the characters and overall story arc presented in the game. Users show a tendency to overlook small and sometimes even large flaws in gameplay design if they are engaged enough with the narrative in place.

With the conflict in definitions for this term, I will be using the term presence to refer to the use of technical and physical means to enhance the feeling that the user is present in the virtual space, and the term immersion will be used to refer to the psychological aspects that convince a user they are present. The two core aspects of immersion I will be referring to when using the term are Narrative and Tactical immersion.

### Project Context

As mentioned, VR has grown extensively as an emerging technology since the initial version of the Oculus Development Kit appeared in the demonstration circuit in 2013, and consumer interest in the technology surged after the full commercial releases of modern headsets in 2016. In the relatively short time since launch, it has become a rapidly expanding area of interest within multiple industries, such as games, films, and more.

Drawing the focus to the market surrounding games, there are several prime examples that demonstrate how best to transfer traditional gameplay loops and mechanics to the medium of VR to provide engaging and rewarding experiences. One such title is 'Pavlov', a game that sets out to translate the core mechanics of 'Counter Strike', a popular traditional First-Person Shooter (FPS), and create a fun and engaging VR experience that as a result sees many recommendations to new users (Reddit, 2020), (Reddit, 2021).

As we see more titles emerging and accessibility of hardware growing, more approaches are being taken to strive for different and alternative styles of gameplay that would not be possible without VR. As this section of titles begins to grow, there is a greater number of complaints as well as praise with respect to the level of immersion a user may experience in these applications.

For example, the software title 'Boneworks' is often touted by more experienced users as a must-have VR game due to the full physical representation of the player and the way it attempts to obey the regular laws of physics that affect our real bodies (Reddit, 2021). On the other side of the argument, there is the software title Half-Life: Alyx, which users have described as being one of the most immersive applications released since VR headsets hit the public market (Reddit, 2021) but represents the user as only a pair of hands, and instead relies on narrative and environment to immerse the user. Both titles have users striving both for and against the chosen gameplay mechanics, and these two titles specifically are pitted against each other regularly in terms of recommending one over the other in enthusiast communities (Reddit, 2021), (Reddit, 2020).

The arguments on both sides regarding the medium and its immersion factors vary, but recurring factors are often how the player is physically represented, and how the environment reacts to the player and their representation, as well as narrative strength. This divide naturally raises the question of whether there is a universally immersive method of representing a user in a virtual environment, and in conjunction with how the user is represented, what additional factors could be layered to achieve a deeper sense of involvement for the user.

To consider one aspect of the questions raised by this argument, the application created and presented to participants aims to provide three key points: different levels of interaction with corresponding environment reactions, differing physical representations to reflect the changes in interaction, and subsequently questions about the user's feelings about their preferred method of interacting with the environment.

### [Related Works](#)

The most notably similar study undertaken found when researching aspects of this project is the study "Using Presence Questionnaires in Virtual Reality" (Schwind, et al., 2019). This study took the approach of presenting its participants with both an abstract and realistic environment, and subsequently either a digital questionnaire in VR, or a physical pen and paper questionnaire, to evaluate whether changing the method of questionnaire completion would influence a user's questionnaire result.

The overall outcome of the study determined that the difference in questionnaire format did not seem to have a direct effect on the user's immersion but did appear to increase the consistency in variance when completing the questionnaire in VR. One of the key points in the conclusion is that the VR approach allowed for the studies to be conducted more seamlessly for both user and researcher. The report also supported the use of the IPQ survey for the undertaken study, which directly influenced the decision to evaluate the IPQ survey and its subsequent use for evaluating user's feelings about their environment in this project.

### [Methods or Tools Used in the Solution](#)

The application made during this project relied upon the Unity engine for its development from beginning to end. Unity is a game engine that has a user-friendly environment to develop applications, with the ability to target multiple platforms for release built into the



engine itself. Unity Technologies also offers Unity Learn courses to provide workable tutorials that help users navigate and understand the environment while learning basic development principles, and as an engine is perfectly capable of allowing users to create games with no major uses of programming themselves. As well as supporting and encouraging new users, Unity is an extremely accessible engine which provides licencing to students and independent users free of charge if they are not using it commercially or earn under a threshold amount from applications developed in the engine. As a result of this accessibility, there is also a large community built around teaching methods of development through blogs, video tutorials and community run courses to teach things in different ways to Unity Learn or things that Unity Learn does not yet cover.

Unity was adopted for this project because of this accessibility and the community resources, and the forward compatibility with VR through Unity's own XR Toolkit plugin and native SteamVR compatibility, which gives users quick and easy ways to prototype and access VR development tools in the engine. As with all emerging technologies, requirements or standards often change before settling, and Unity often ends up depreciating many of its own tools and implementing newer versions in its latest unstable releases. The engine's old method of built in VR support has been depreciated in the current builds of Unity (2020.3.X-LTS), to be replaced with different SDKs that correspond to different headsets, which are Oculus, SteamVR, and Windows Mixed Reality, respectively. Due to the disparity in compatibility between versions, development of the application used Unity 2019.4.19f1-LTS, as this was also the newest LTS version of the Unity engine at the beginning of development, and as such was the version the final application was developed in.

As well as providing a robust and beginner friendly engine and tools, Unity also provides the Asset Store, which is a way of distributing premade "Assets" – anything from 3D models to sound effects or interaction mechanics packaged up for free distribution or sale. The environments in the final application were built utilising a royalty-free asset pack distributed by Unity to encourage fast level design prototyping, and the puzzle object models are also free assets distributed on the store (Originals, 2020), (Weibel, 2016).

Finally, Unity allows users to script their own events that can trigger in the engine using the C# programming language. This language shares very similar syntax and operations to Java, and as such was used to add customised interactions into the application. The overlap in C# and Java styling meant that it was much simpler to adapt to and work with across the duration of the project as opposed to learning a language with no familiarity.

### Constraints on Project Approach

There are numerous constraints affecting the approach taken for this project, the first and most crucial being the continuing disruption globally due to COVID-19 and the ensuing measures to keep the spread of the virus in check, namely lockdowns and social distancing measures. Sharing VR headsets between users is often a sanitary issue in places like demonstration booths and VR arcades, and as such this survey is reliant on being a remote study, as well as the remote participants having access to a VR ready PC and VR headset themselves. This is a limiting factor in the number of users able to participate in the study but is likely to also skew the data towards the preferences of enthusiast VR users as



opposed to the general population. This could have been solved by taking a more traditional approach to the study, but in the continuing global climate it is unlikely to be safe to conduct a study this way for a considerable amount of time.

The second biggest constraint on this project was the level of personal knowledge of the Unity engine. Unity was earmarked as a potential method of developing this project before the project officially began as VR was already where I knew I wanted to base my project at the start of the academic year, because I find it engaging as a consumer. I had completed some of the beginner Unity Learn tutorials to gain familiarity with the editor in the term before the project timeline began because of this interest but did not further develop my familiarity with the more advanced functions and VR support provided by the engine. Fortunately, as previously discussed, the community tutorials and the similarity between Java and C# meant that whilst a learning curve was present across the project, I was able to deliver an adequate experience that was able to gather user feedback and implement, at minimum, a functional representation of my idea.

## Section 3 – Specification and Designs

This section sets out the requirements decided at the beginning of the project to shape the final application to be delivered to users, and the design choices made around them. These requirements evolved from the initial plan put forward and were fleshed out with more ideas regarding the direction of the application. The requirements themselves are separated into Technical and User requirements.

This section also details the planning and design stages of the applications to demonstrate the approaches taken to fulfil the requirements laid out.

### Technical Requirements

The criteria listed below are separated by order of importance for the overall application. Each point will be given acceptance criteria as subpoints.

Must:

- Use Unity to provide a SteamVR compatible application
  - o Compatibility with SteamVR native headsets and Oculus Headsets
- Present users with Participant Information and Consent forms before they progress, and be playable without consent
  - o Filling out Consent form does not allow users to progress without accepting all points before allowing them to begin the game
  - o Present an “opt out” button that begins the game immediately
- Illustrate the interaction methods users will be presented with during the course of the application
  - o Visual or written explanation of progress made in the application
- Provide 3 methods of interacting with the environment for users to solve puzzles with in a linear fashion
  - o First – basic grab interaction for manipulation of objects directly in to designated areas, demonstrated in a room-like environment with scaled down objects

- Second – More advanced grab interaction based around the manipulation of levers to control a crane that manipulates objects, demonstrated with scaled version and full size
- Third – allow the user to interact with the objects through manipulating them at a small distance through pointer interaction and movement at full scale.
- Allow users to choose one of the 3 methods experienced for the 4<sup>th</sup> and final puzzle
  - The 4<sup>th</sup> puzzle must be identical in progression across all 3 types of solution
- Allow users to take part in the questionnaire inside of the application at the end of the 4<sup>th</sup> puzzle and provide instructions for remote participation once completed
  - Questionnaire generates “.txt” file upon completion with codified results for User upload

#### Should:

- Allow for some level of physics interactions between objects the user interacts with and the environment
  - Collisions between physics objects in at least one puzzle
- Provide the user with a visual representation to determine their position in the environment
  - Demonstrate where the user’s hands are with a model or prefabs

#### Could:

- Add higher customisation to visual quality in regarding performance settings
  - Low, Medium, High visual settings declared before launch
- Implement Pause/Resume states
  - Pause button bound to controller button and immediately suspends application alongside visual interruption
- Integrate modularity in design to allocate for adding or removing methods of interaction or additional development
  - Clear consideration for modularity in implementation

### User Requirements

The user requirements listed are divided into sections of similarity and given acceptance criteria as subpoints.

#### User Interaction

- Visual feedback and animations
  - Animate hand gestures corresponding to user gestures
- Add active sounds to interactions
  - Add menu sounds and grab sounds to interactions

#### Usability and Accessibility

- Text for user instruction must be clear and visible at a comfortable eyeline to be read
  - Use of large font at an acceptable distance from the user’s position
- Provide options at the beginning of the application for users to set preferences
  - Dominant Hand selection
  - Locomotion preferences
    - Snap turning or Smooth turning
    - Smooth movement or Teleport movement

## Environment Design

### Basic –

- Demonstrate core concepts for environments and player models
  - o Derived from engine primitives and placeholders

### Intermediate –

- Have environments and player models be more developed and visually interesting
  - o Use Assets and Primitives

### Advanced –

- Have a consistent and engaging environmental and player design
  - o Provide a mix of Asset Store and custom modelled assets for environment and player model
- Add ambient sound to environments
  - o Background track plays when running application

## Security

- Application needs to preserve user anonymity
  - o .txt files generated are wiped of metadata when uploaded
- User's information security maintained
  - o User can only upload data manually instead of it being automatic

## User Engagement

- Provide well designed and engaging puzzles for the user to interact with
  - o Users engage well with puzzles and find them enjoyable
- Construct a short or compelling narrative to tie the puzzle sections together
  - o Narrative told through environment or exposition to the user

## Concepts and Designs

Once the requirements were laid out, the backbone for the progression in the application was determined. As a result of the initial idea of linear progression through puzzles, the development route was laid out in a flowchart, to set out a clear path for both development milestones and the application's layout, shown in Figure 1. This structural plan was adhered to across the development timeline without any major deviations.

With the development route laid out, the environment and puzzle design needed pre-visualisation and concepting. This began simply by creating notes about ideas and directions for the art style and environment design. This progressed to creating rudimentary models and environments using a VR Modelling tool, Microsoft Maquette. Using this tool allowed for quick and simple realisation of basic 3D environments from the listed design notes and allowed for further development of these ideas in a 3D space. The limitations of the tool only allow for the use of 3D primitives and premade models, which in hindsight was beneficial for keeping polygon counts low on the models and the environments in the final version, to maintain performance on lower end systems.

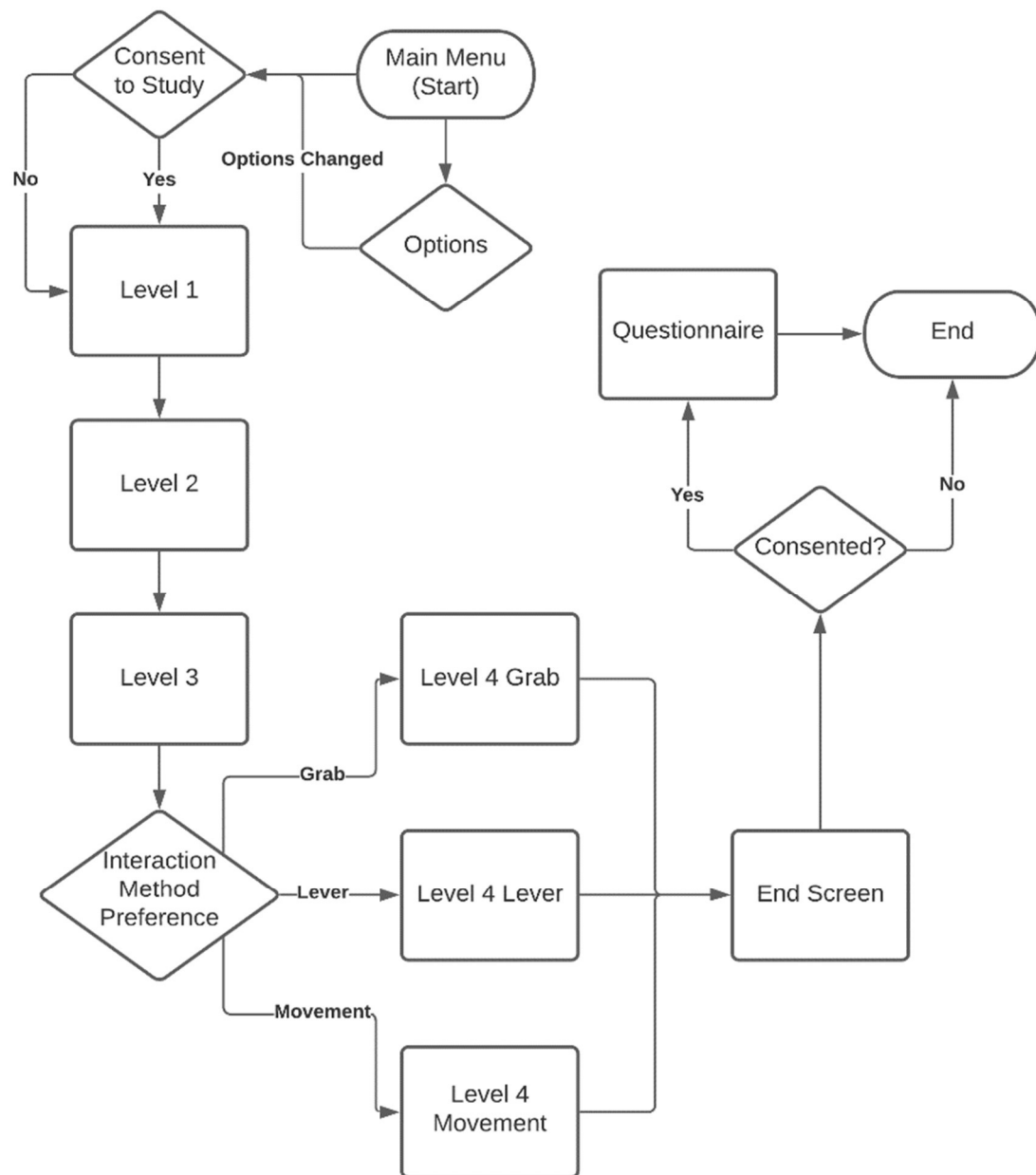


Figure 1 - Development Flowchart

### User Representation Concept

The concepts shown in Figure 2 demonstrates the idea of the user gaining physical elements to their player model across the course of the application, first starting at the least possibly accurate representation and progressing further into a visibly more capable version of themselves upon completing puzzles. With more time, the aspects of the user's physicality could be investigated separately as an immersive factor, but the core idea here was simply that the player was able to see aspects of themselves had changed and as such would determine that it enabled different interaction methods.

From left to right, the distinctions between each proposed state are as follows –

- State 1 – The user's hands are non-descript and are only capable of pointing
- State 2 – The user is given full hands to properly allow them to manipulate aspects of the environment within reach
- State 3 – The user is given a body in conjunction with hands to allow for their free movement in the environment

The humanoid figure at the far right is a placeholder for dimensions used to properly size the models. The height of this humanoid was 1.8 meters for reference.

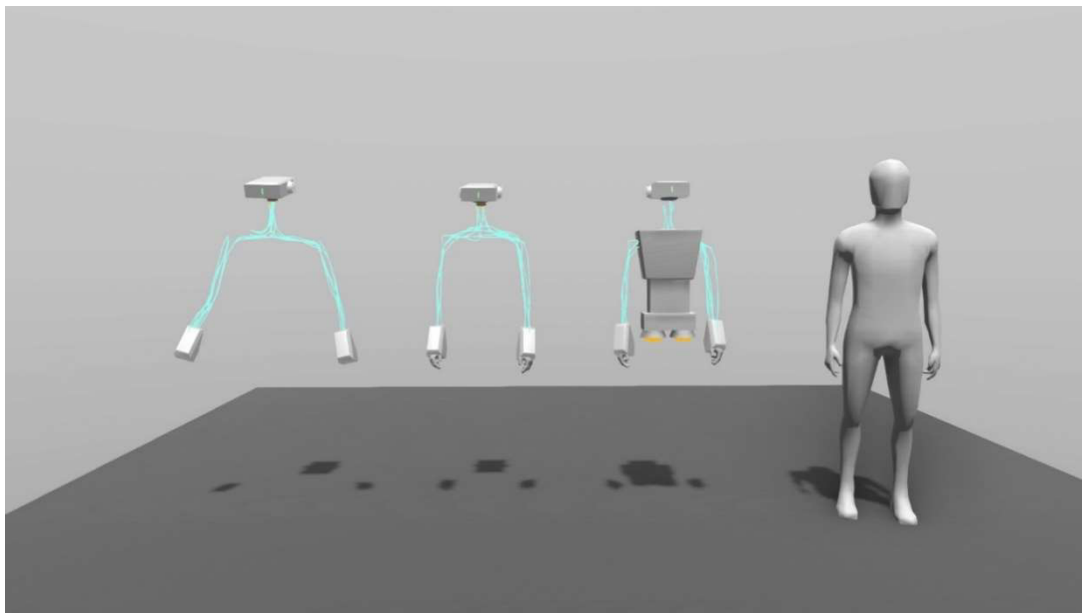


Figure 2 - User Representation Concepts

### User Representation Design

The realisation process of the user representations shown in Figure 2 breaks down in to two sections – Modelling and Rigging. To realise these designs in a way that allows for their use in Unity projects, the models had to be rebuilt in Blender, an open-source 3D modelling and animation tool. Modelling will be discussed in this section, and rigging will be explained in the implementation section of this report. Notably, the method of VR compatibility implemented in the application combined with the lack of opportunities for the player to see the representation of themselves meant that whilst the head was modelled for all the states, it was not added to the tracked components of the user's setup in any of the states in the application.

Below in Figures 3, 4 and 5 are the models that were created in Blender to be used to for each of the representation states in the concepts. Figure 3 was the simplest model to create due to the fact that simple geometry and symmetry made modelling quick and easy, and the lack of rigging requirements for the hands made importing and using the assets shown seamless.

Figure 4 shows the iterative development of Figure 3 with the addition of hands to the player model. This added complexity to the process in both the method of modelling, due to the design having to be somewhat more organic than the already angular geometry, as well as modelling whilst trying to regard the intention of allowing the hands to be posable due as a response to certain user inputs.

Figure 5 shows the model for state 3 clearly having a more developed body in combination with the hands from the previous state. This was the second easiest design to create, with the utilisation of simple shapes, angular geometry, and symmetry making this section of the modelling process almost as easy as the first.

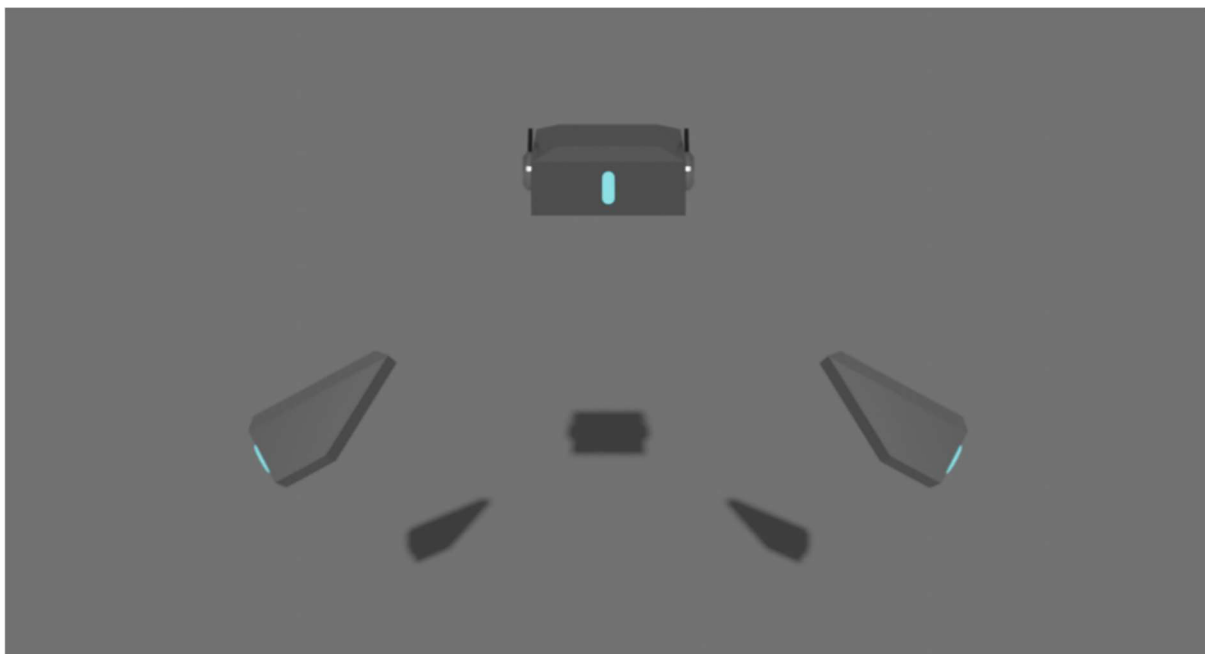
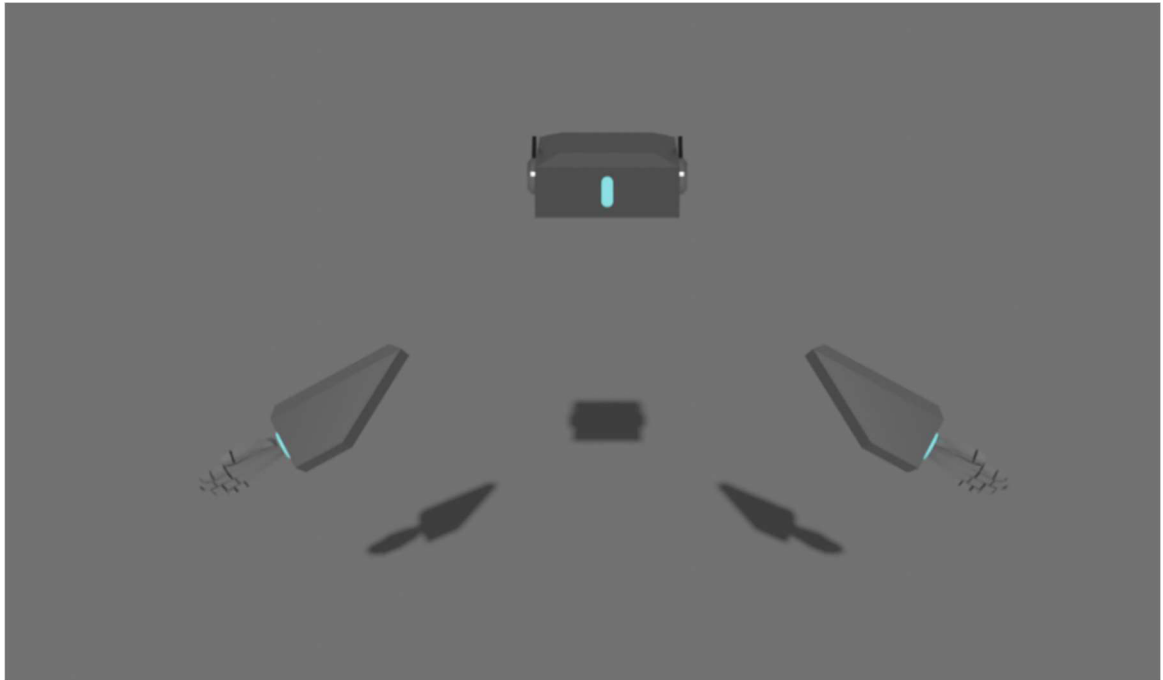
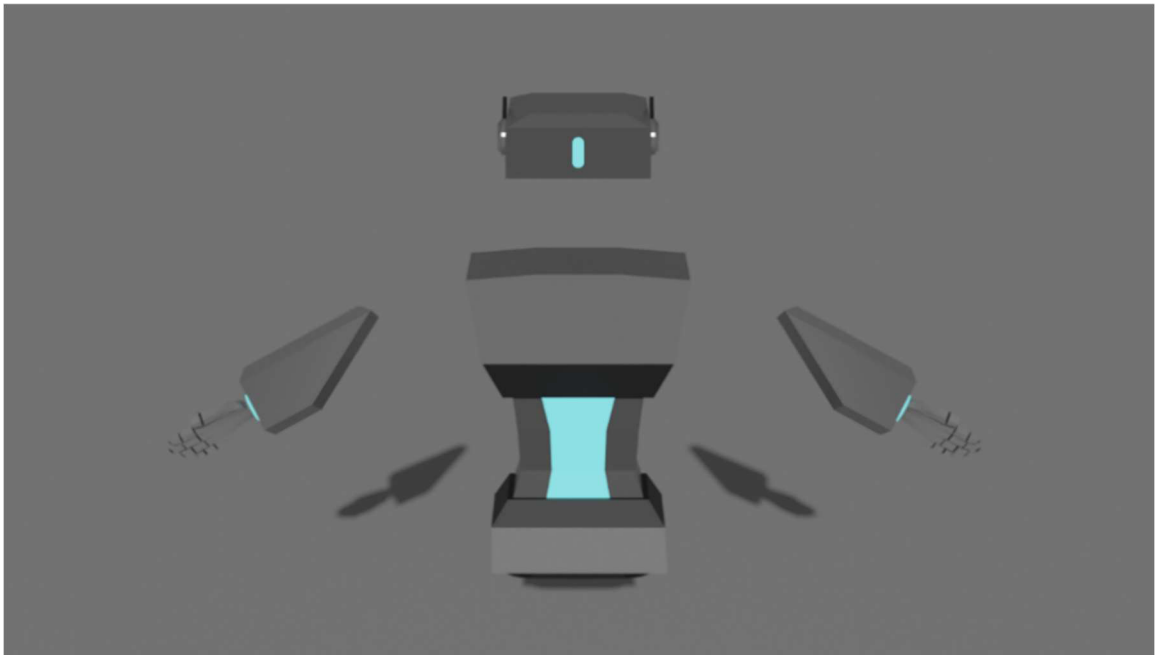


Figure 3 - State 1 Final Model



*Figure 4 - State 2 Final Model*

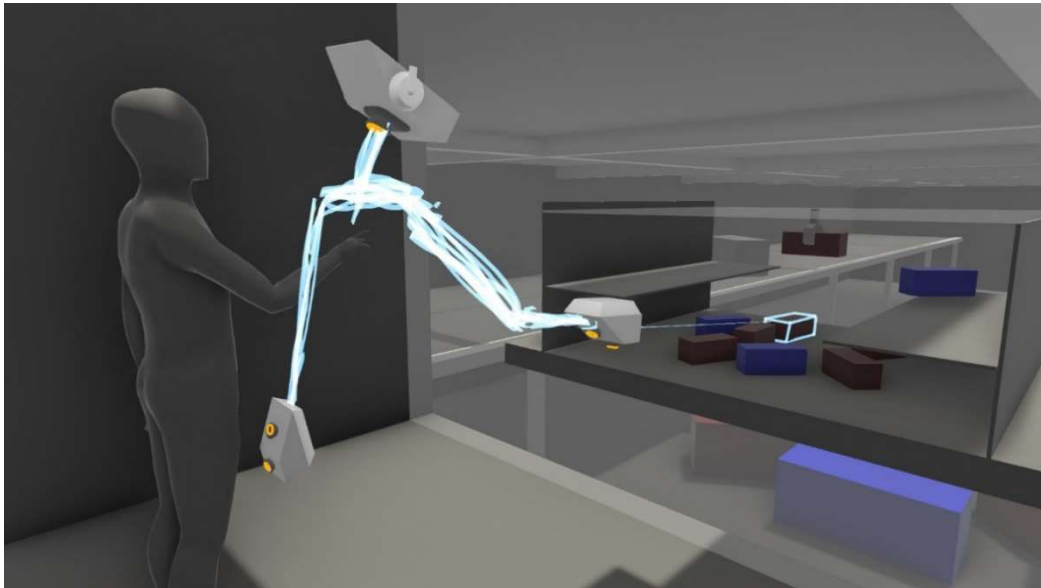


*Figure 5 - State 3 Final Model*

### Interaction Method 1 Concept

In Figure 6, the conceptual idea for the first puzzle is shown, as well as the larger scale surrounding environment. The idea was that users were presented with a miniature version of the warehouse that was in fact around them and would use simple point and select methods shown below to organise coloured containers into areas indicated to the player.

The method devised was that the user could simply select an object to move, and then a location to move it to, and both the small and large environments would reflect the change ordered by the player, with orders to be carried out by a loading crane in the environment.



*Figure 6 - Interaction Concept 1*

### Interaction Method 1 Design

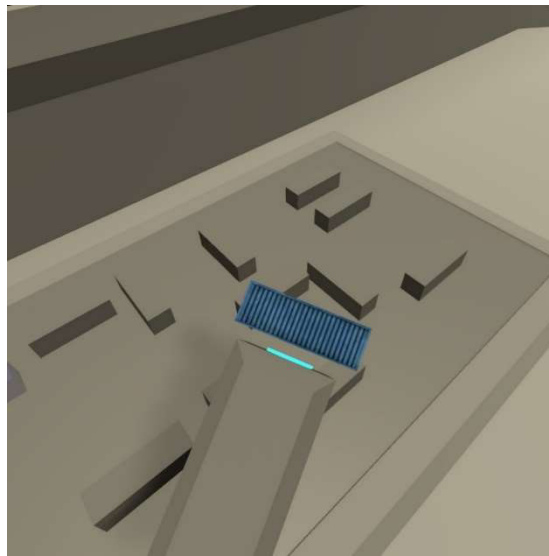
The realised version of the first interaction method underwent a series of heavy changes and evaluations to determine what was technically feasible within the project's timeframe.

The milestones laid out at the beginning of the project meant that this was the second area to be focused upon in the development timeline, after the implementation of the main menu. Due to the simplicity of the development of the main menu, a core understanding of the constraints and limitations of both technical knowledge and Unity itself had not been properly explored, and as such this proved to be the most challenging idea to implement.

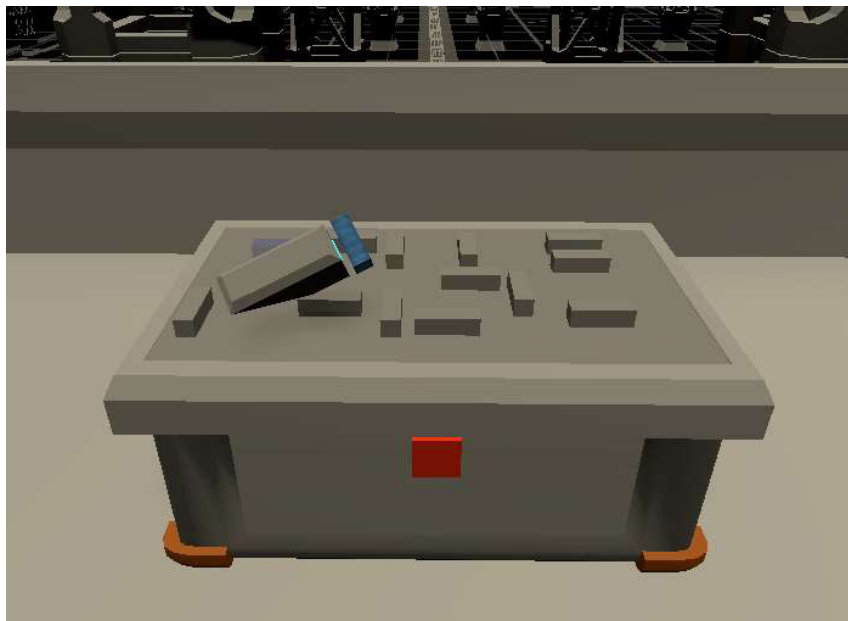
The first revision of the idea decided to pivot to the idea of using the main menu's control scheme of pointer interactions to give users a large screen-like interface in which they interacted with 2D representations of the puzzle objects and environments and had to sort through the objects this way. This second idea was also a victim of my inexperience with the engine however, as it would have required knowledge of more advanced methods of interaction as well as attempting to adapt methods from traditional 2D and 3D game implementations to VR. After numerous failed attempts to bring these methods over to VR in a way that was satisfactory or suitable for the idea, the idea was scrapped.



The second revision scaled back even further, resorting in creating the scaled down version of the puzzle and having the user interact with it in a visually and physically basic method. This meant removing the idea of the miniature crane moving the objects for the user entirely as well as the small scale and large-scale changes, in favour of a basic interaction that was easier to implement but left this section much sparser in terms of things that were environmentally and mentally engaging. This final interaction method can be seen from the user's perspective in Figure 8 and the overall perspective from a 3<sup>rd</sup> person view in Figure 7. The core idea of the environment being at least visible to the user whilst they complete the puzzle remained in this implementation but could likely have been removed considering the removal of the double manipulation, as the context of the larger warehouse floor becomes irrelevant unless explained to the user.



*Figure 8 - Level 1 - First Person*

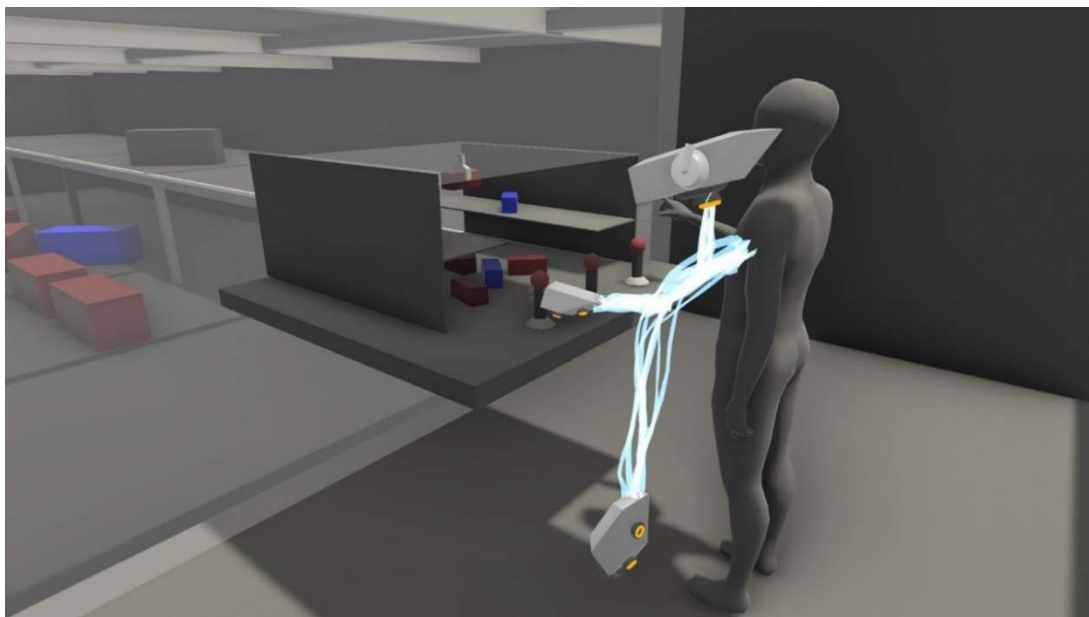


*Figure 7 - Level 1 - Third Person*

### Interaction Method 2 Concept

The planned evolution of the environment presented in method 1 is shown in Figure 9, in which both the environment and the user's model changes to reflect that there are now more ways for them to interact with the environment.

The most distinct change here is the addition of the levers in front of the player's scaled down environment. These levers would allow the user direct control over the loading crane as opposed to the action and reaction interaction from the crane in the first method. As with the first environment, the concept was to have the cranes both directly influencing the smaller environment as well as the larger exterior environment to give the user a better sense of the results of their interactions.



*Figure 9 - Interaction Concept 2*

### Interaction Method 2 Design

This interaction method was able to have more success in implementing the idea presented in the concept but still ended up with some major deviations in how it was implemented.

The biggest difference was that the small scale to large scale manipulation was separated in to its two sections. A brief introduction to the method of crane manipulation using the levers by interacting with a small-scale crate and crane inside the room environment, and then followed with the actual puzzle itself taking place in the full-scale environment with full scale objects. The same lever design is used for both puzzles to provide consistency across the level.

This was done to compensate for the complete lack of environmental interaction implemented in the first puzzle to engage the user more thoroughly. In the figures provided there is both a user perspective, Figure 12, and 3<sup>rd</sup> person perspective from the editor to show the small-scale interaction in the room environment, Figure 11, followed by a third person editor view of the large-scale environment to give an indication of the change in scale to the user, Figure 10.

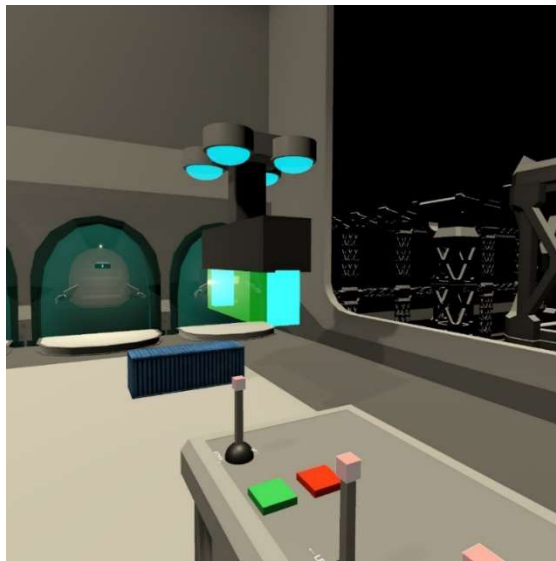


Figure 12 - Level 2 - First Person

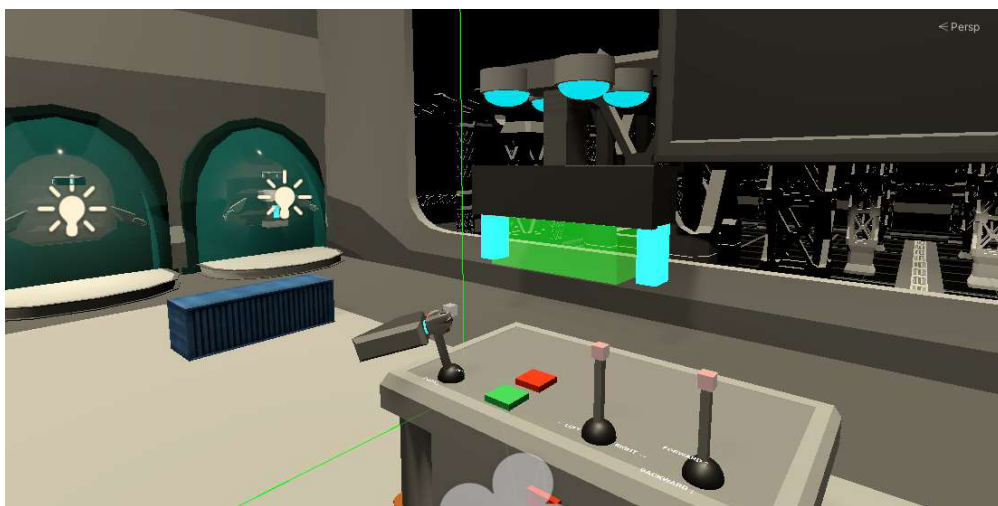


Figure 11 - Level 2 - Third Person



Figure 10 - Level 2 - Third Person

### Interaction Method 3 Concept

The final method of interaction is demonstrated in the two figures below. This method was designed to be a clearer step in a different direction for the user, by allowing them to bypass the scaled down environments they had been interacting with up to this point and instead directly allow the user free roam in the environment to directly manipulate any of the puzzle objects themselves.

The major factor to indicate this change was to be the addition of a modelled body to the user's avatar that they would be able to perceive themselves, as well as the introduction of an extra hand-based device shown in Figure 14 that would allow the user to manipulate the crates as physics objects, being able to rotate, push or pull them around the environment to solve the presented puzzle.

This stage of manipulating was planned to be a way to give users a sense of weight and scale to the tasks they had previously been performing as small scale operations in a separated environment, as well as be engaging and interesting in allowing the user total freedom to manipulate puzzle objects as they wish, shown in Figure 13, as opposed to within the constraints of the previous controlled methods.

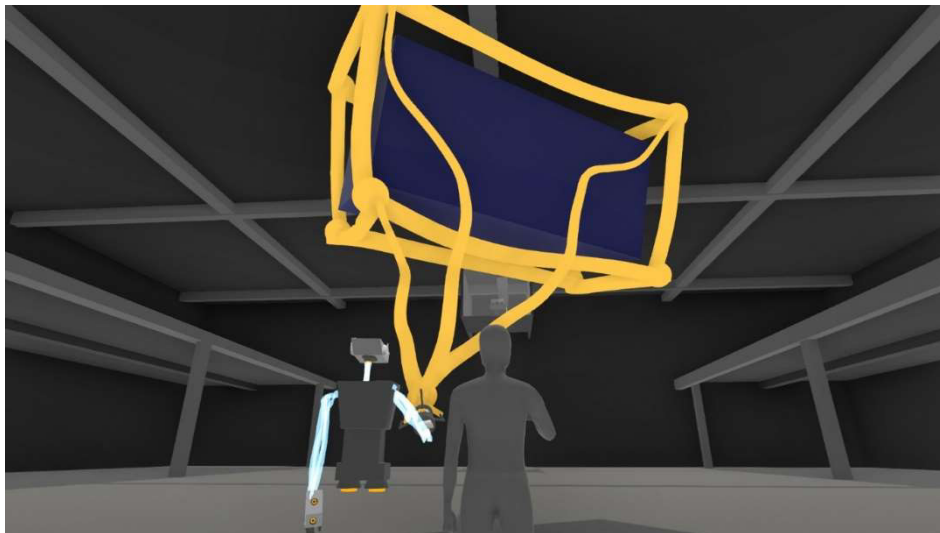


Figure 13 - Interaction Concept 3

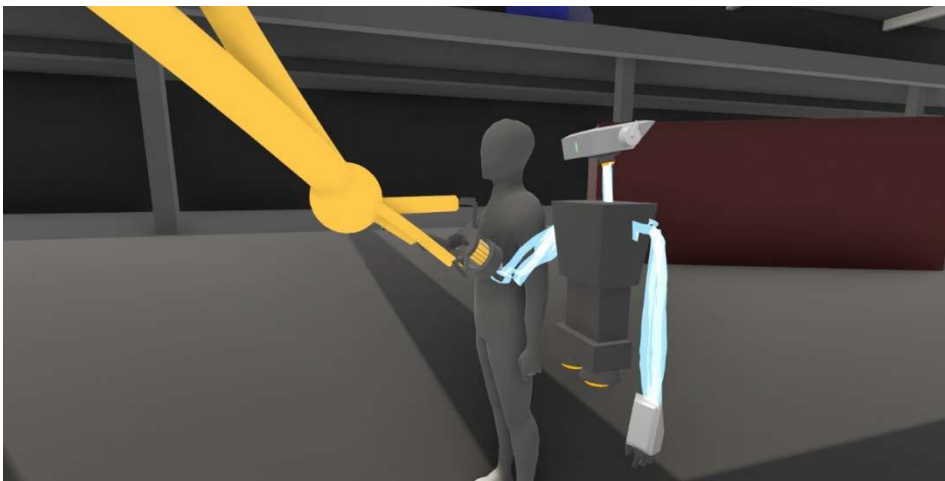


Figure 14 - Interaction Concept 3 Alternate View

### Interaction Method 3 Design

This method of interaction is the one that had to change the least in terms of the central ideas presented in the original concept. All the core functionality intended was in fact able to be implemented, minus the intended flair illustrated in Figure 13.

The main loss here was the implementation of the user's body, where the concept attempts to illustrate that the body be fully visible to them if they had looked down at their torso. Due to technical limitations, This model was not in fact able to be implemented in the game to represent the player. The model was instead used to indicate to the user through the environment that this would be a progression step for them, but the limiting factor ended up being the time constraints on the project when regarding how a full body avatar is implemented in Unity.

This final stage of interaction was instead implemented with the addition of a small model similar to the device shown on the user's arm in Figure 14, which was then attached to the user's hand models from the previous stage to give a visual indicator in the change of interactivity. This was offered in conjunction with an explanation to the user that they were now able to move freely and manipulate puzzle objects differently before they began the puzzle.

A first person perspective, Figure 16, and a 3<sup>rd</sup> person editor view, Figure 15, are provided of the final interaction method in use to demonstrate the changes made between concept and implementation.

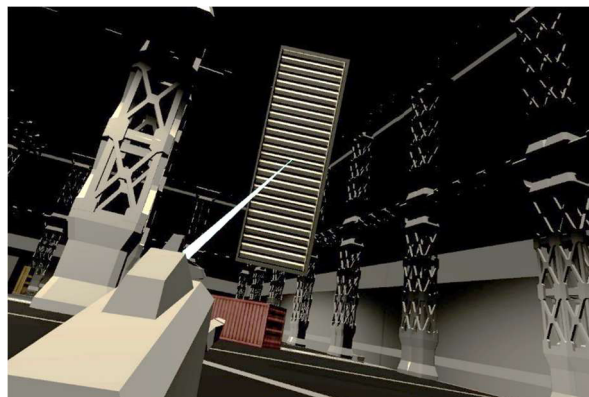


Figure 15 - Level 3 - First Person

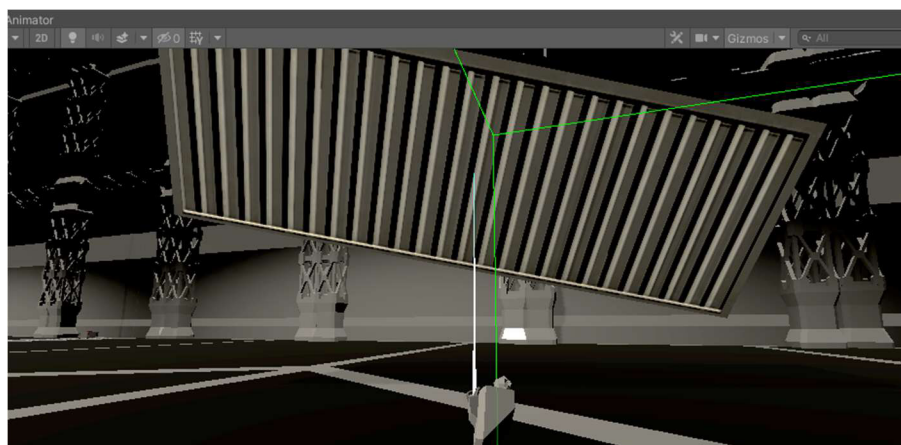
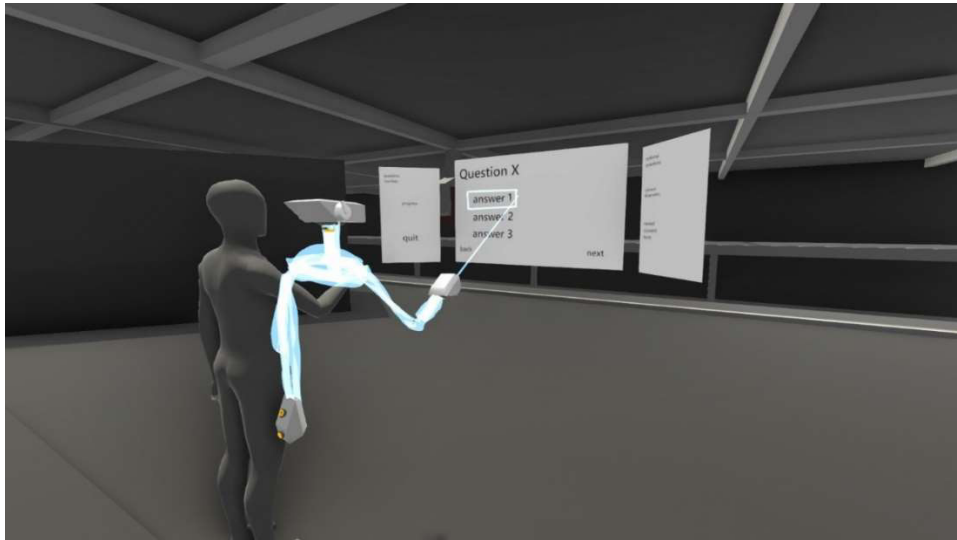


Figure 16 - Level 3 - Third Person

### Questionnaire Concept

The questionnaire section of the application was laid out to present the user with an interface that utilised the first concept interaction method to have users answer questions on a virtual screen in the same VR environment they had been completing puzzles in, shown in Figure 17. The intention here was to present users with a seamless, in-app way to leave their feedback if they had opted into participation. This was intended to be a one click submission once the questionnaire was finished. The actual questionnaire content was not outlined in the concepting stage.



*Figure 17 - Questionnaire Concept*

### Questionnaire Design

The design of the questionnaire moved from simple page by page presentation questions to a scrolling window where users could parse through questions like a webpage. The difficulty here was not the implementation of the questions themselves but the method of getting the results from the users to remote storage. This step of the questionnaire underwent multiple revisions in strategy before arriving at the current implementation. The first idea was devised in the concept stage and was planned to implement a single button upload by taking a stored file and implementing an API call for a remote upload solution.

This was investigated further in the development stages, and due to the limitation of using university provided OneDrive access for secure storage of user data, this implementation had to be scrapped. This was because using the OneDrive API only allowed for files to be uploaded once a user logged in before uploading, which would have prevented users from being anonymous. The only way to allow users to remain truly anonymous was to ask users to open a OneDrive submission link in an incognito window to ensure that the user was not logged in to their personal Microsoft accounts. Below in Figure 18 is the questionnaire that the users undertake in the application, and Figure 19 contains the instructions shown to users when they exit the application to indicate the steps needed to finalise participation.



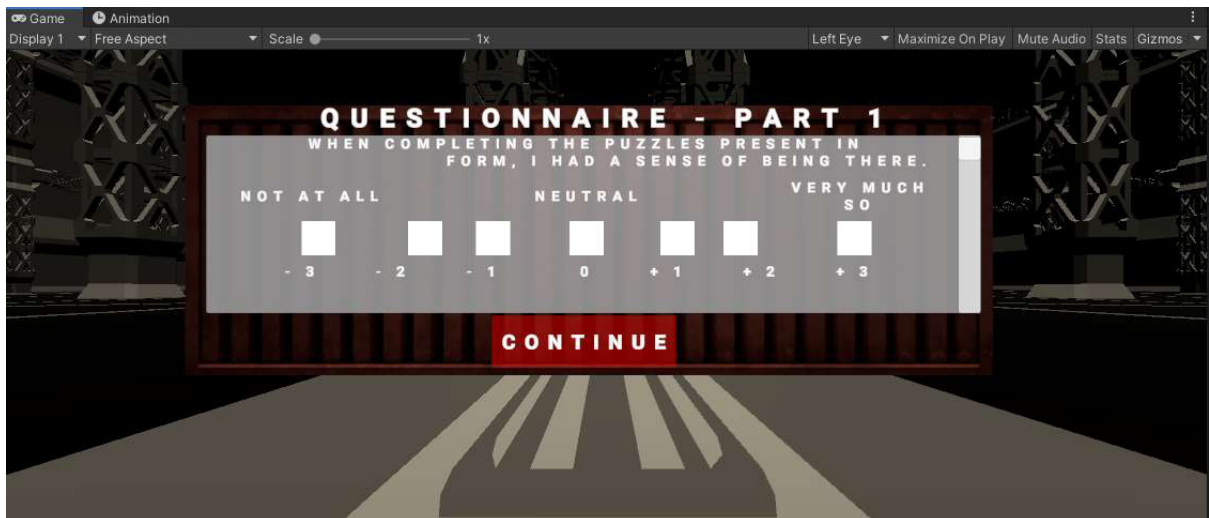


Figure 19 - Questionnaire Implementation

```

1 -----
2 Hello! There are a few steps to this process which will all be detailed here at the top of the file.
3 Step 1: Please open the PDF located alongside this .txt named 'Consent Form' and tick all the boxes. do not add any identifying information!
4 (You already did this in game, but I need this document to make it official)
5 Step 2: Open an INCOGNITO/PRIVATE WINDOW in a browser of your choice.
6 Step 3: In your INCOGNITO/PRIVATE WINDOW, copy and paste the following URL to access my OneDrive Upload Link
7 URL --> https://cf-my.sharepoint.com/:f/g/personal/scottaj4\_cardiff\_ac\_uk/EgcJfKE0MtNFnFLvmFRUTckB051nh0DS9pxh6dd3t8sdIg
8 Step 4: When prompted for your name, please enter First Name: 'Anonymous', Second Name: 'User'
9 Step 5: Upload and relax! your part is done! Get in touch if you have any feedback of any kind. Thanks again for your time!
10 -----

```

Figure 18 - Questionnaire Instructions

## Architecture and Development

Overall, a significant portion of the original concepts did come to be realised in the final application, but as mentioned previously, not without core changes or alterations to methodology or design. Unlike the concepts presented however, the progression flowchart did stay the same throughout, and the structure presented in Figure 1 ended up being separated across Scenes, or levels, in Unity.

This separation into scenes is illustrated in Figure 20 and details the attempt to maintain the progression across separately loaded sections, to try to keep each section of the game as clean as possible without containing any irrelevant data. Scene 5 is notably different as it contains the mechanics of all three previous levels but uses the same environmental puzzle objects for two of the implemented methods, as well as the same progression routes, so as a result all 3 methods of interaction were integrated into the one scene. In this scene the user's choice in interaction method is made, and so only the required section is activated in response to this choice.

The division of scenes illustrates all the necessary sections of the application, as well as the order they were worked on. Sectioning the progression flowchart into scenes provided a good way to allow for separation and eventual combination of the developed interaction methods. To better present the internal architecture of each of these scenes, in this final flowchart shown in Figure 21 each level section has been broken down further into its mechanics and overall object architecture involved in the scene to give a better idea of what each level implementation consists of.

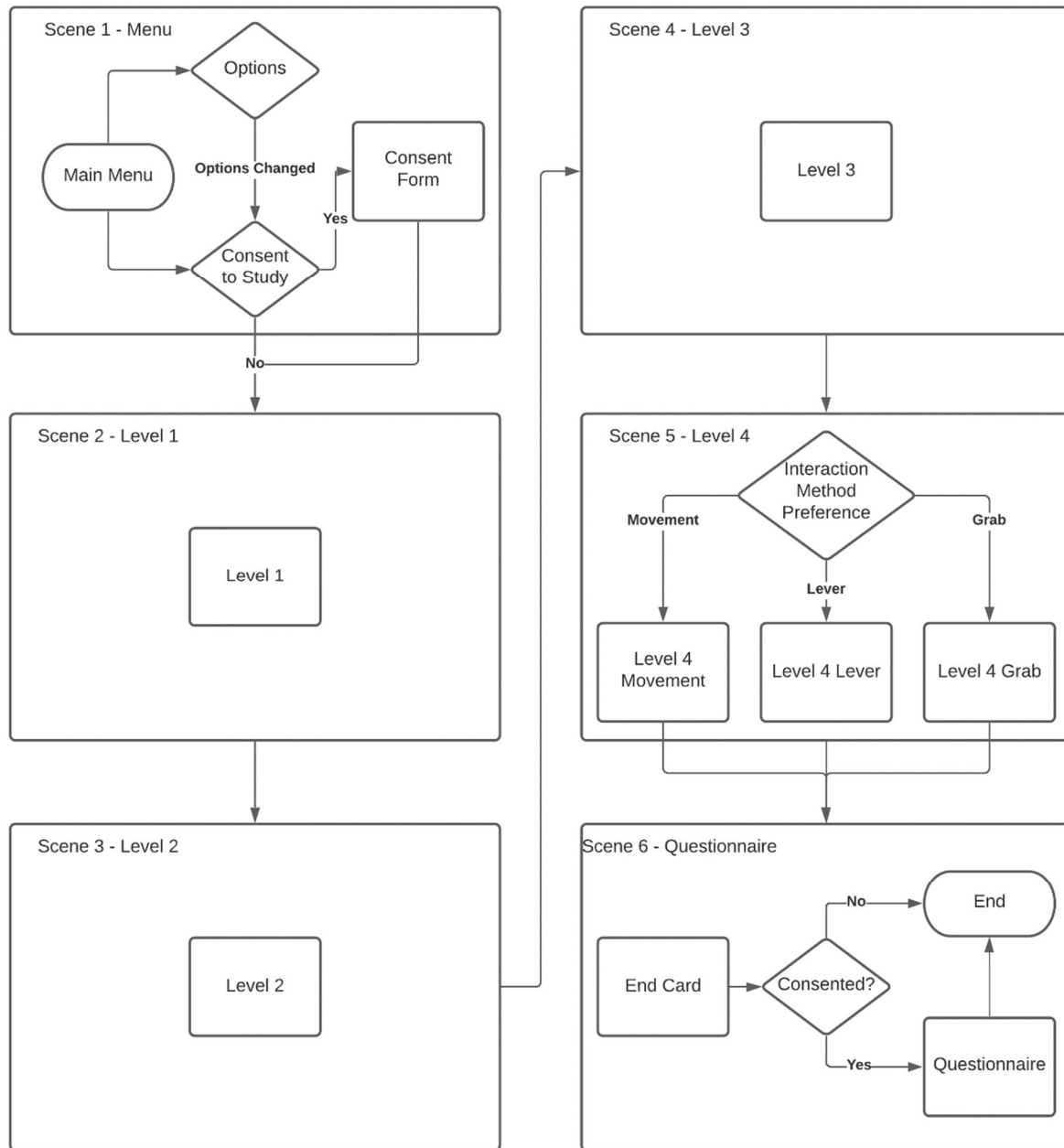


Figure 20 - Scene Structure in Unity



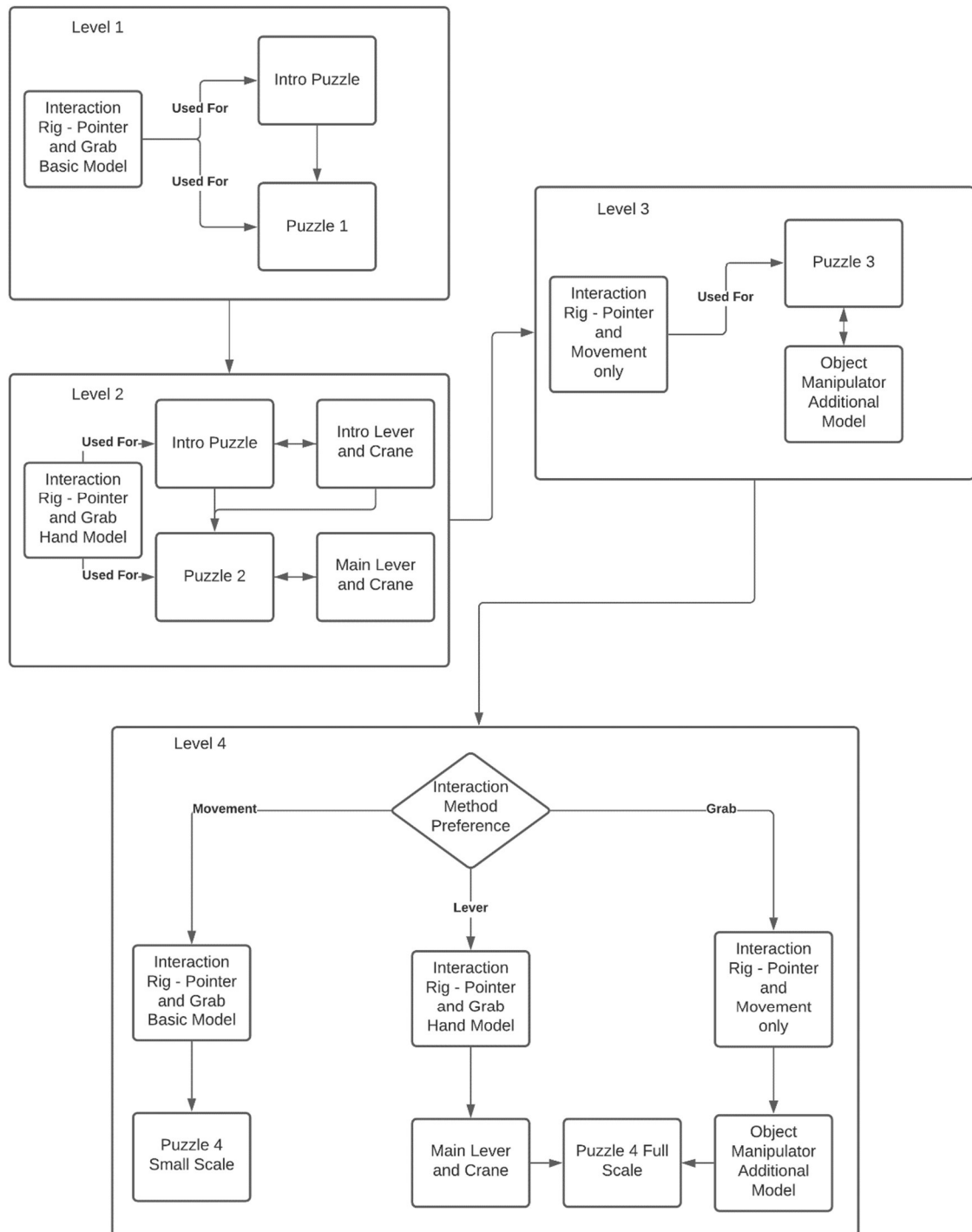


Figure 21 - Level Architecture in Unity

## Section 4 – Implementation

This section will give descriptions of code considered critical to the final application. Code snippets and editor screenshots will be provided to give context to the concepts explained, as well as the provision of the source code for further examination.

The most critical aspects of the application can be narrowed down to the following sections:

- XR Toolkit implementations for VR within Unity
  - o XR Rig
  - o XR Interactors
  - o XR Grab Interactors
  - o Movement Providers
- Rigging and Animating models
  - o Hand Rigging
  - o Full Body Rigging
- Custom Scripts for Additional Features
  - o Lever Joints
  - o Screen Fader
  - o Movement Checks
- Questionnaire Content
- Environmental Assets

### XR Toolkit

To begin utilising XR Toolkit, the preview package must first be downloaded and enabled using the Package Manager in the editor, by selecting “Advanced” in the top left and enabling the use of preview packages, shown in Figure 22. Once this is done, VR support must be enabled in the Project Settings by adding VR SDKs to the list through the plus button at the bottom of the SDK list. Here it illustrated that the build has compatibility with the two most common PC based VR SDKs. Once installed, XR Toolkit allows the use of the scripts explained in the following sections to develop VR compatible applications quickly and easily.

In Figure 23 is how VR support was enabled in Unity. The method used here is depreciated and subsequently removed in later Unity versions, but it was necessary to employ the use of this depreciated method to add OpenVR support. OpenVR is the old SDK that was used to allow compatibility with native SteamVR headsets and is being actively being replaced by OpenXR. At the time of development however a stable release of OpenXR was not available, and due to the headset used in development being SteamVR native, this method had to be used. If developed using an Oculus or Windows Mixed Reality headset it would have allowed use of the XR Toolkit preview implementation to enable support. Whilst not using the Toolkit method of enabling VR support for Unity, the application is still able to make use of all the Toolkit additions in the development process.

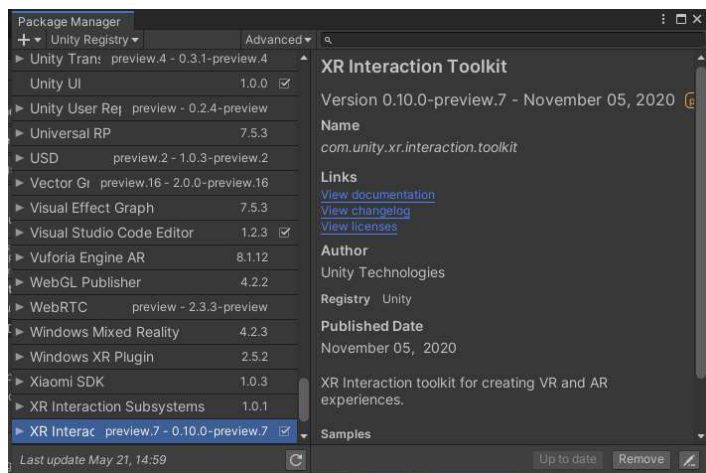


Figure 22 - Toolkit Plugin

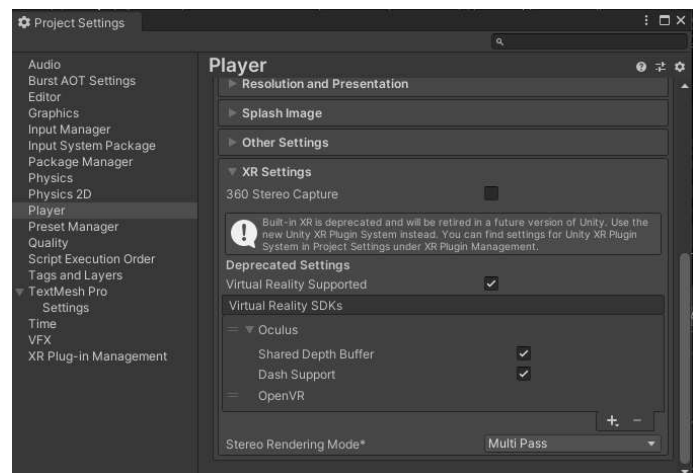


Figure 23 - Deprecated VR Support

## XR Rig

Adding an XR rig to a scene is simple, and only requires the deletion of the main camera in a default empty scene to begin using it. In the context menu shown in Figure 25, there are two kinds of rig, with two subclasses of rig inside of them, these being Action Based and Device based, and Room Scale and Stationary respectively. The distinction between the types is that the Action based device uses Input Actions to determine user input in a contextual method, whereas Device Based uses the InputDevice methods to directly take input actions from the user's controller. The final application uses a combination of Stationary and Room Scale rigs but only uses Device Based, as that was the most common method used when looking for documentation and tutorials.

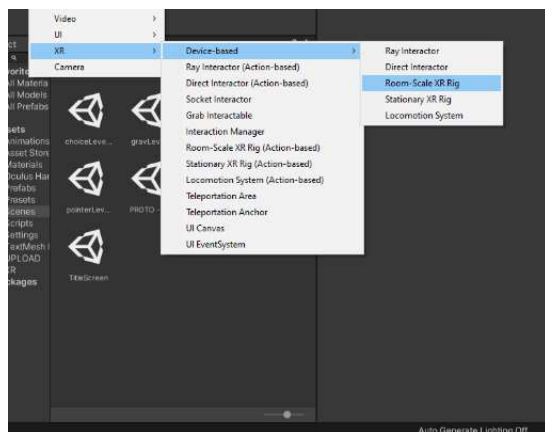


Figure 25 - XR Context Menu

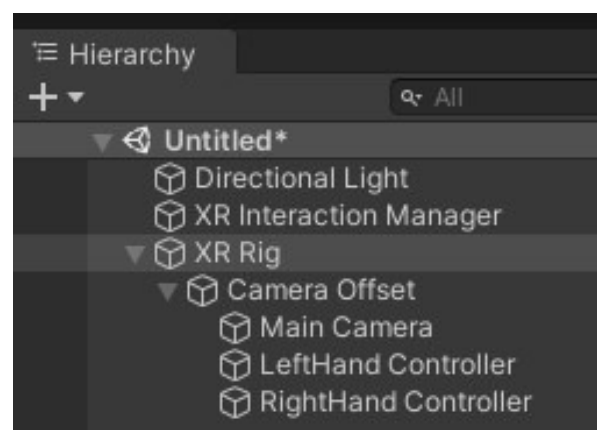


Figure 24 - XR Rig Default Setup

Demonstrated in Figure 24, the creation of an XR rig automatically provides a tracked element for all the key elements, in the user's head and hands. The camera offset is used to determine how far from the floor the user's perspective is. By default, each of the player's hands is initialised as an XR Controller to allow for tracking, and as such is provided with a suite of customisability options shown in Figure 27. The default interactivity option provided is a Ray Interactor, shown in Figure 26.

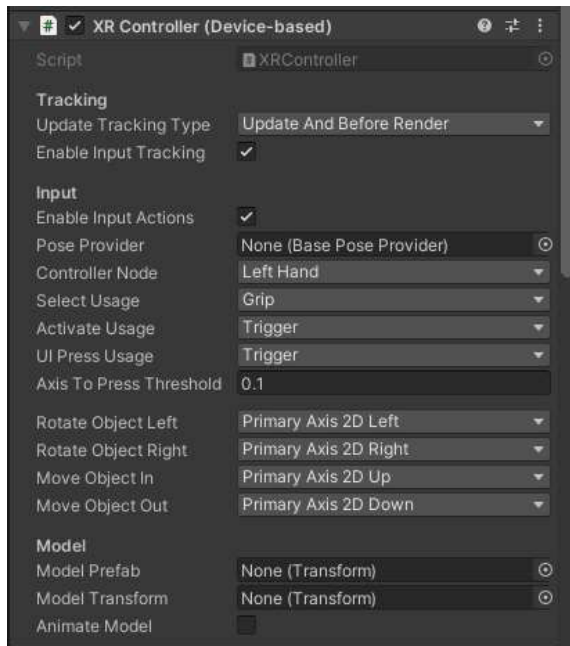


Figure 27 - XR Device Based Controller

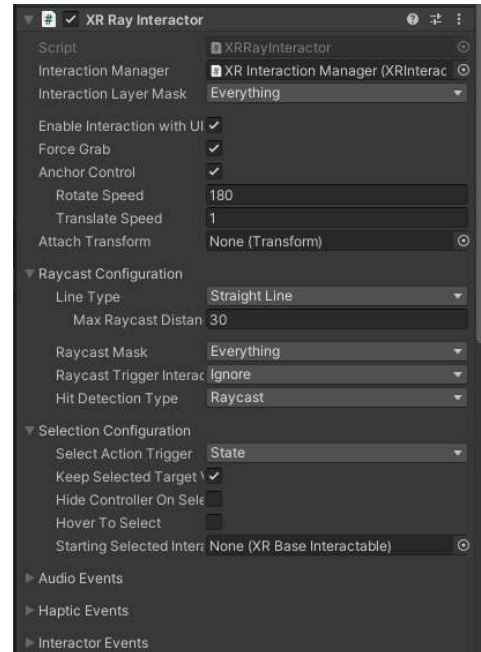


Figure 26 - XR Ray Interactor

## XR Interactors

XR interactors are the methods in which objects can interact with other certain objects in the environment. There are three types of interactors: Ray interactors, Direct interactors, and Socket interactors. All three of these provide unique methods of object interaction, but only two of these are directly useful for having the user interact with their environment, which are Ray and Direct.

Ray interactors, Figure 26, are a way to implement a method of distance-based interaction and are used in the application throughout all the scenes to enable the user to interact with the guiding text presented to them. These interactors operate off unity's Physics.Raycast method, which casts a ray from an origin point in a specified direction against colliders in the scene. This interactor is combined with a Line Renderer to show the user where their raycast will hit, as well as a visual demonstration whether what the user is pointing at can or cannot be interacted with. Whilst ray interaction can be used for any number of cases and objects, the Toolkit provided version is intended for use with the user's tracked hand objects within the editor.

Direct interactors, Figure 29, are used when objects are touching or interacting with a collider as well as the object targeted being a valid target for the interactor. Direct interactors are used in the first and second interaction methods implemented to allow for the user to pick up and manipulate designated objects in front of them. Colliders assigned to direct interactors can be easily adjusted and are highly versatile when it comes to how the object attaches to the interactor itself.

Socket interactors, Figure 28, are the third type of interactor provided in the XR Toolkit but serve a separate purpose to the first two described. These interactors work best when used in the environment as opposed to the user's interaction method. These work in a similar

way to the direct interactors, but provide a collision area for valid objects, and when an object is released in this collision area, it is snapped into place and held static within the socket. This provides a quick and easy way for users to store objects in an inventory or bind items in areas on or around the user's person. Socket interactors are used in every puzzle presented in the final application as the locations in which the user must place their respective puzzle object and this method was also used as the main technique that allowed the user-controlled crane to pick up and manoeuvre the puzzle objects independently of the user having to put the objects in the crane themselves.

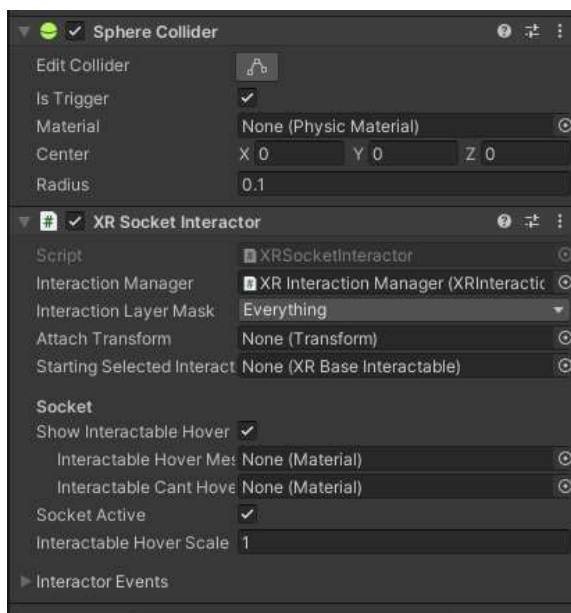


Figure 28 - Socket Interactor

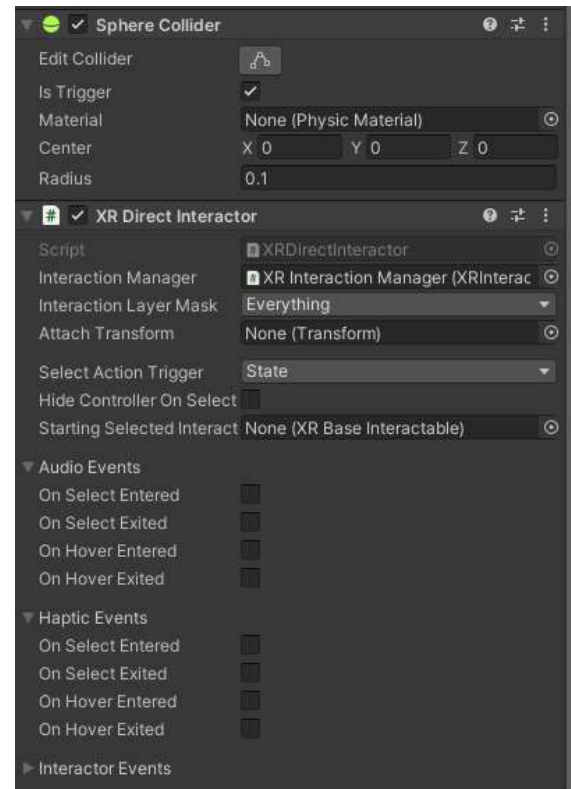


Figure 29 - Direct Interactor

## XR Grab Interactors

Grab interactors, Figure 31, are what marks an object as interactable with the previously listed interactors. This allows for the object to be “grabbed” by the interactor and attach to it while following its position. This interactor also implements a rigidbody, shown in Figure 30, which is a component to indicate to the engine that the object is now a physics object and must obey the physics parameters set for the application. This is what allows for puzzle objects to be manipulated by the interactors bound to the user's hands, and allows for users to drop, throw, and place objects around the environment. Finally, implementing this script on an object attaches a collider to allow for collision targets to be used as the method of determining if the objects can interact.

The use of Grab interactors is the core of how most of the final application was delivered, as it provided a simple and customisable way for the user to be able to manipulate objects in mechanically different ways whilst still maintaining ease of use through XR Toolkit provided scripts.

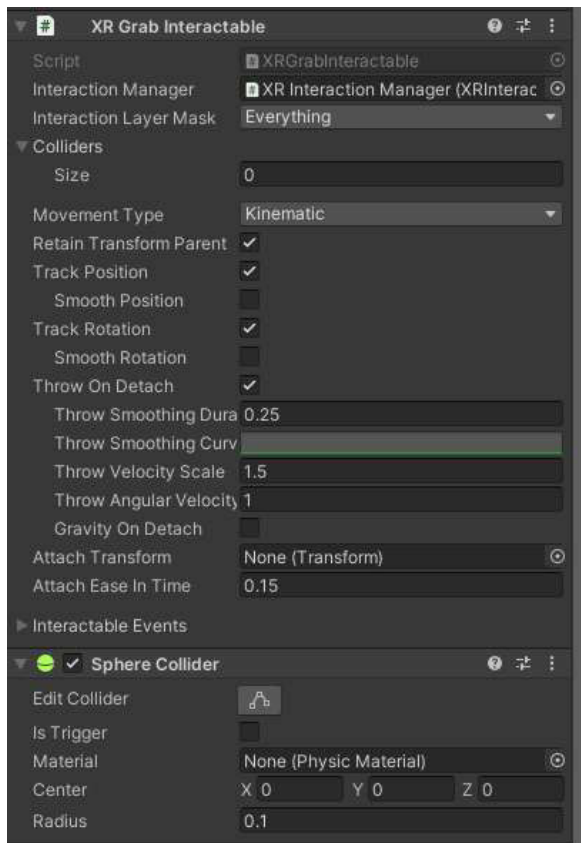


Figure 31 - Grab Interactor

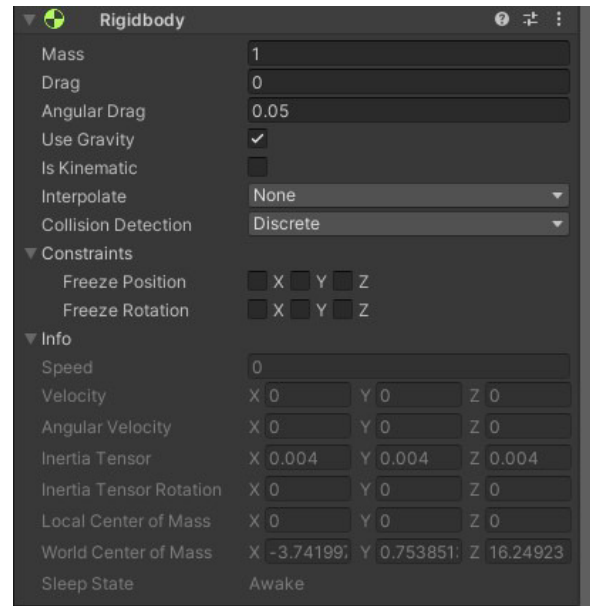


Figure 30 - Rigidbody Default

## XR Movement Providers

The XR Toolkit provides all basic tools that would be necessary for creating simple movement for a user in VR. The implemented methods are what has evolved over time to be the default methods of user locomotion in VR titles, Teleportation and Continuous movement, of which the final application implements both, but only the method determined by the user.

Teleportation movement systems require the additions of either Teleportation areas or Teleportation Anchors, which either allow the user to freely teleport around a specified area or allow the user to teleport to a single fixed point. When the user selects a valid area to teleport to, the XR rig is instantly moved to the destination point. In Figure 34 is the implementation in the editor that applies to the XR rig, and in Figure 33 is the respective script attached to the teleportation area demonstrating the additional parameters that can be set. Teleportation areas also need a collider attached so that they can be targeted by raycast. In Figure 32 there is a view of the teleportation area that was implemented in the virtual environment in the final application. The implementation in the application also includes a separate custom script, “locomotionController.cs”, provided by a community Unity tutorial, which allows for the setting of custom activation thresholds for teleportation, as well as allowing for it to be disabled or enabled on specific controllers and only activate the teleportation targets and teleport controllers when necessary. The code is shown in Figure 35 and commented to give context.



```

// Update is called once per frame
@ Unity Message | 0 references
void Update()
{
    //if left teleport boolean returns true
    if (leftTeleport)
    {
        //activate the teleport reticle, but only if button depressed to threshold value
        leftTeleport.gameObject.SetActive(enableLeftTeleport && CheckActivated(leftTeleport));
    }
    //if right teleport boolean returns true
    if (rightTeleport)
    {
        //activate the teleport reticle, but only if button depressed to threshold value
        rightTeleport.gameObject.SetActive(enableRightTeleport && CheckActivated(rightTeleport));
    }
}
2 references
public bool CheckActivated(XRController controller)
{
    //check the teleport button is pressed to the minimum threshold value and return true if so
    InputHelpers.IsPressed(controller.inputDevice, teleportButton, out bool isActivated, thresholdActivation);
    return isActivated;
}

```

Figure 35 - Commented LocomotionController

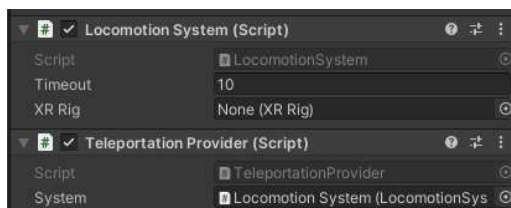


Figure 34 - Teleport Provider

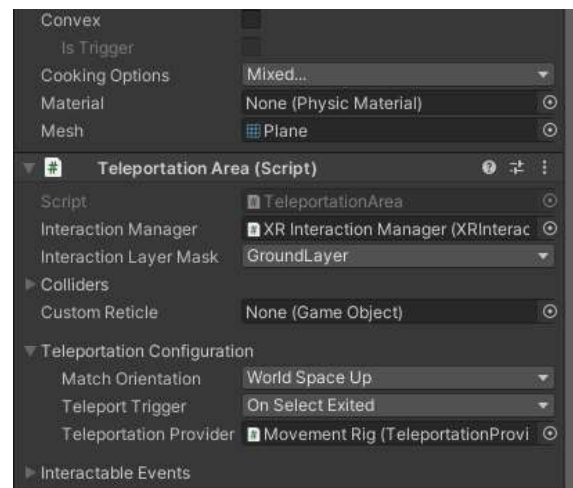


Figure 33 - Teleport Area

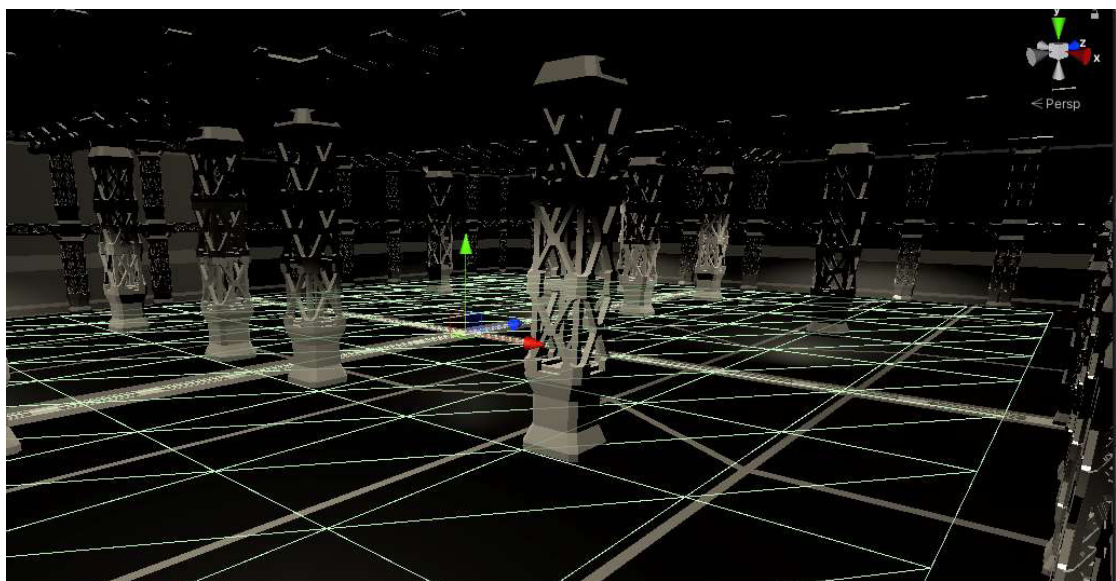


Figure 32 - Scene Teleport Area

Smooth movement systems simply move the XR rig based on the analogue value provided to the script from a determined input source, in this case the primary input from the controller, either joystick or touchpad. XR Toolkit provides a highly customisable continuous movement provider, but the method implemented in the application itself was a custom script called “ContinuousMovement.cs” as part of a tutorial on Unity VR development, shown in editor view in Figure 38. This code has been commented to best provide context for how operations unfold and their purpose in Figure 37, Figure 36 and Figure 39, as this is not the default movement provider, but as stated, this script is provided in a player movement tutorial for Unity. This could have been substituted out with the XR Toolkit version at any point for more customisability, such as allowing the user to determine whether the movement was based on their look direction or their hand direction, but due to following the tutorial it was not apparent that there was an XR Toolkit alternative until late enough in the development it did not seem worth changing considering the current implementation worked well.

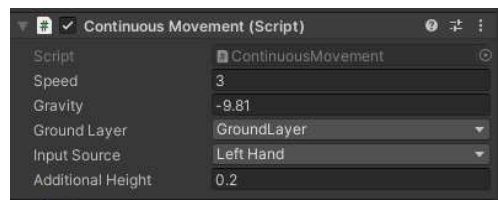


Figure 38 - Movement Script Editor Window

```

@ Unity Message | 0 references
private void FixedUpdate()
{
    //movement uses fixed update so that it is consistent as opposed to every frame update which can lead to inconsistency
    //call function
    controllerFollowHeadset();
    //get the yaw of the player's head as a quaternion
    Quaternion headYaw = Quaternion.Euler(0, rig.cameraGameObject.transform.eulerAngles.y, 0);
    //manipulate vector3 directions to get the head's direction of movement
    Vector3 direction = headYaw * new Vector3(inputAxis.x, 0, inputAxis.y);
    //move the character controller object from the direction devised
    character.Move(direction * Time.fixedDeltaTime * speed);
    //gravity and falling functions
    //check if touching ground
    bool grounded = checkOnGround();

    if (grounded)
    {
        //don't need to fall as on ground
        fallingSpeed = 0;
    } else
    {
        //multiply speed by gravity over time until touching ground
        fallingSpeed += gravity * Time.fixedDeltaTime;
    }
    //manipulate the controller's direction based on fallingspeed
    character.Move(Vector3.up * fallingSpeed * Time.fixedDeltaTime);
}

```

Figure 37 - Movement Script Contents

```

bool checkOnGround()
{
    //check player controller is on the ground
    //raycast from middle of controller to floor
    Vector3 rayOrigin = transform.TransformPoint(character.center);
    //shoot out ray for .01 distance from bottom of controller object (feet)
    float rayLength = character.center.y + 0.01f;
    //raycast for boolean
    bool hasHit = Physics.SphereCast(rayOrigin, character.radius, Vector3.down, out RaycastHit hitOutput, rayLength, groundLayer);
    //output bool value based on whether touching floor or not
    return hasHit;
}

```

Figure 36 - Movement Script Contents



```

void controllerFollowHeadset()
{
    //get the player's height plus offset
    character.height = rig.cameraInRigSpaceHeight + additionalHeight;
    //determine center of player controller object from the player's head
    Vector3 capsuleCenter = transform.InverseTransformPoint(rig.cameraGameObject.transform.position);
    //set the center of the controller object as the new vector3
    character.center = new Vector3(capsuleCenter.x, character.height / 2 + character.skinWidth, capsuleCenter.z);
}

```

Figure 41 - Movement Script Contents

The Toolkit also provides scripts that facilitate the manipulation of the player's camera view without the need for the player to turn their heads themselves in the two most common methods, Snap turning and "Smooth" or Continuous turning. Snap turning immediately moves the facing direction according to user input by a set angle, commonly 30, 45 or 90 degrees. Smooth turning achieves the same thing but over time, creating a slow pan effect of the user's field of vision. Both methods were implemented, again at the user's preference as to which should be used, as well as taking in to account the user's primary hand and assigning this action to their offhand, as well as allowing for a custom angle between 30 and 90 degrees. Snap Turning is shown in editor view in Figure 40, and Smooth is shown in Figure 41.

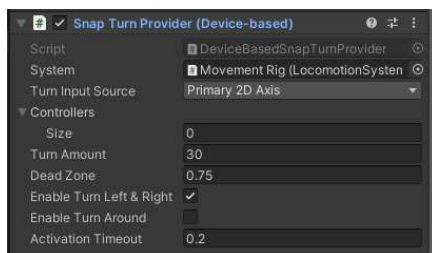


Figure 40 - Snap Provider

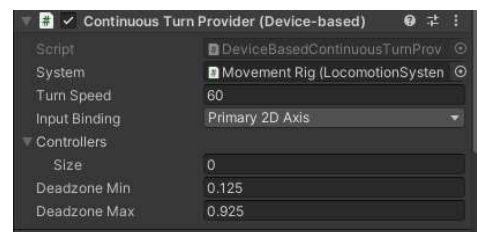


Figure 39 - Smooth Provider

## Rigging and Animating Models

As previously mentioned in this report, the process of creating custom models is divisible in to two sections, Modelling and Rigging. Rigging is the more complex and directly affects whether animations can be used in applications, but animations for user's hands in VR applications is often the default, and as a result was decided to be implemented. All rigging was done in Blender, and the animations were created and managed in Unity.

### Rigging

Rigging is the process of adding the ability to control objects, typically to allow for animation (Team, n.a). There are many different tutorials on how to rig objects for different programs, but Blender offers Rigify, which is a tool that helps to speed up the process of rigging by providing fully generated rigs for you to attach your model to. This is the method that was used for both the user's hands and the full body model. To begin, you enable the Rigify plugin, and then using the "Add" context menu, you add a Human (Meta Rig) from the Armature section to your blender environment, as shown in Figure 42. Whilst this rig works perfectly fine in Blender or other bespoke animation software, the rigs created by Rigify have a lot of issues when importing them to Unity. To bypass this issue, the custom script RigifyToUnity (AlexLemminG, 2020) was used to properly convert the rigged models for use in Unity.

This pre-made rig is an advanced rig intended to allow for facial expression animation as well as full body animation, but the facial bones were removed for the full body model, shown in Figure 43, and only the hand bones were kept in the case of the hand model shown in Figure 44. Once the necessary bones are left, rigging becomes a process of careful and consistent adjustments of all the bones in the rig to properly fit where to where they are needed. Primarily, due to the quick and easy way the hands were modelled, arranging the bones for the hands was likely the most time-consuming task due to the geometric disparities between the default Rig pose and the pose that the model was made to have. Once these were aligned, the bones can be attached to the model, and then the Rigify generator button is used to properly generate a full Rigify rig for animation. Once the main rig is created, the custom script RigifyToUnity is used to convert the rigs to a format that is better compatible with Unity.

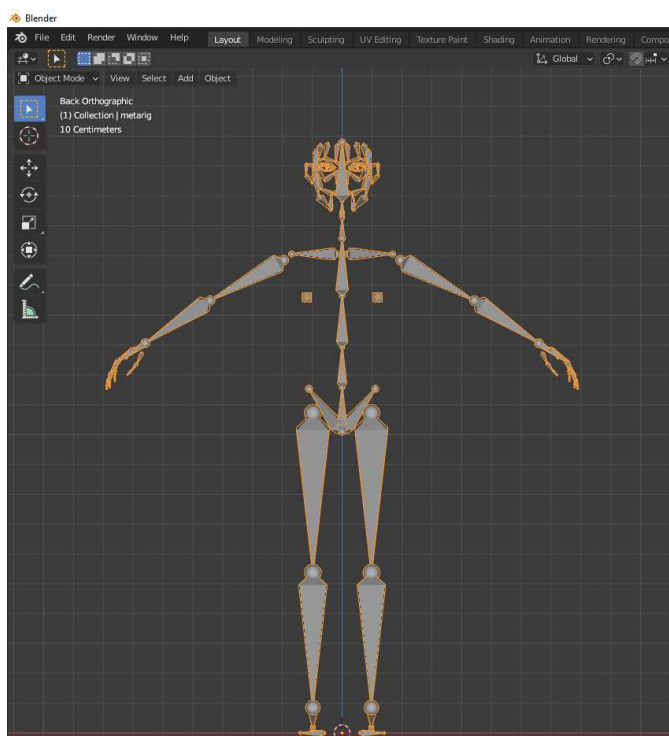


Figure 42 - Rigify MetaHuman Rig

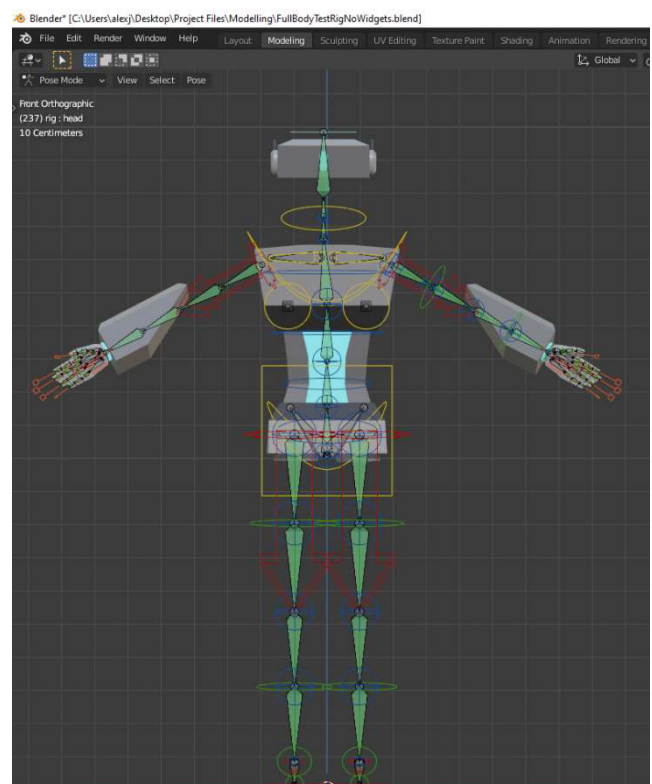


Figure 43 - Level 3 Model Rigged with MetaHuman Rig

## Animating

The rig shown in Figure 43 has been mentioned previously as being unable to be implemented due to technical limitations. The issue here was that the creation of the rig was necessary to create to attempt to use it with Unity's built in Inverse Kinematics model that would apply rotations of the user's head and hands to try to properly reflect the position of the rig skeleton. Unfortunately, all initial implementations of this method led to finding issues with the rigs generated and where the bones had attached to, as well as the Inverse Kinematics not behaving smoothly and generating a lot of computational overhead for no gain. The rig was subsequently not implemented, but the decision was made to take the existing hand model and its rigged bones and create a separate rig for just the single hand and import that instead, shown in the Unity Editor in Figure 44.

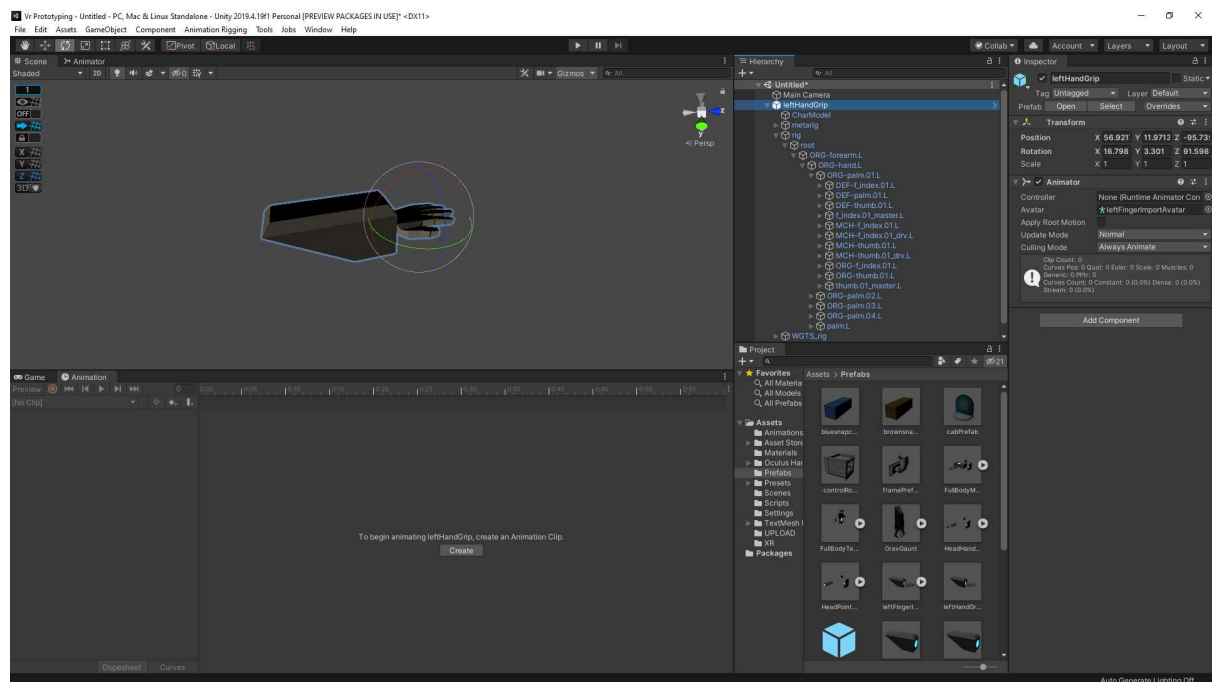


Figure 44 – Overview of Animating in Unity

In Figure 44, the structure for this object is shown in the hierarchy section on the right-hand side. All the objects named 'DEF' are deformation bones that allow for the individual bones of the hand to be posed in the editor. This object also has an animator attached that provides it with an avatar, which is generated when importing the model. The avatar is generated by Unity to set up bone structure in a way that makes it usable with the animation methods provided.

Once correctly imported, the animation tab at the bottom allows for the creation of animations directly by manipulating the object in the editor window. This timeline is the centre of animation creation and management within the Unity engine and is very simple to use. The lines across the dope sheet, Figure 45, are to track the transformations on the object it is alongside in the hierarchy on the left-hand side. The points indicate a transformation to that object, and these points can be set at any point across the timeline shown at the top of the sheet. If all these transforms were set at the 1.0 second mark for example, playing the animation would show a loop of the hand going from its default to the

closed position in a single second. However, the method of animation used for the player hands does not require bespoke timing of animations in this case, and instead has the animations set up as states that the hand can be in, shown in Figure 45.

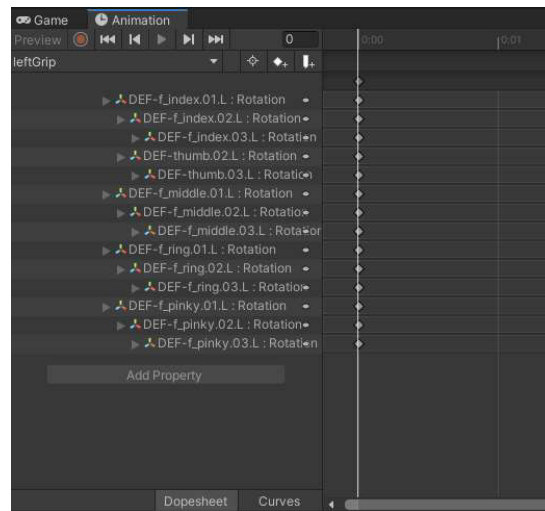


Figure 45 - Animation Keyframes

This shows the transformations recorded on each deformation bone to create the pose for gripping, shown in the bottom of Figure 51 - Blend Tree Grab Demonstration, and marks the position they are in at 0.0 seconds. When played this animation would immediately transition the hand to the closed position with no delay, but due to the different approach taken, this is the ideal outcome.

The method used to transition between the states made for the hand is the use of blend trees. Blend trees are an animation method provided by Unity specifically to allow for smooth transitions between multiple animations, but in this case, it is being used as the primary animator to blend the hand model between states. In Figure 46, the animator tab is shown with the parameters used in the animator which are the float values of the trigger input and the grip input buttons, and the blend tree itself on the right.

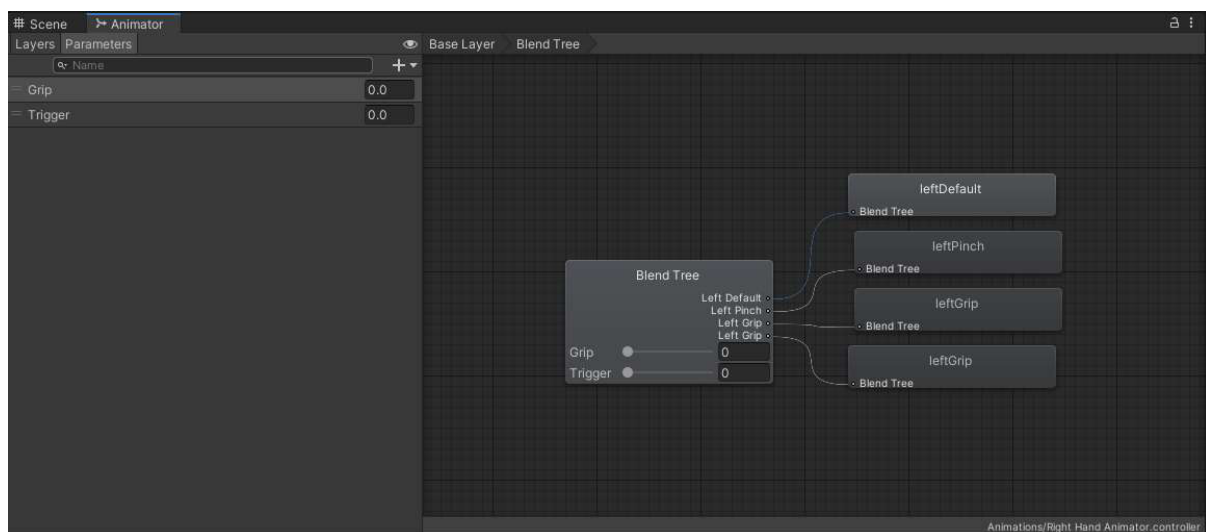


Figure 46 - Blend Tree Overview

The blend tree is being used in a way that takes the granular values of the two inputs, the grip and trigger values, and blending between the animator states based on where the values are between 0 and 1. Based on these values, a 2D freeform directional blend is used to create 4 points – Grip and Trigger not Pressed, Grip fully pressed, Trigger fully pressed, Grip and Trigger fully pressed, shown at the top of Figure 51 - Blend Tree Grab Demonstration

In Figure 50 - Blend Tree Pinch Demonstration Figure 50 the preview animation is showing the animation state and the position of the blend tree when the Trigger value is 1.0, and the Grip value is 0.0, and in Figure 50, the Grip value is 1.0 and the Trigger value is 0.0 as well as the blend tree position. When both values are 1.0, the state used for the grip value continues to be used, as when the player actuates the grip and trigger button their hand is likely curled into a fist with the most common controller layouts.

To retrieve the Grip and Trigger values, a script named “HandAnimator.cs” was created following a tutorial to allow for the Grip and Trigger values to be passed to the animator for either the left or right hand from an assigned controller, Figure 47. This tutorial was adapted to make use of the Blend Tree functions with the custom hand assets.

The script performs two core actions, one in the initialisation of the application, and the other on every frame of the application, shown in Figure 49 and Figure 48. This code is commented to best explain its functions.

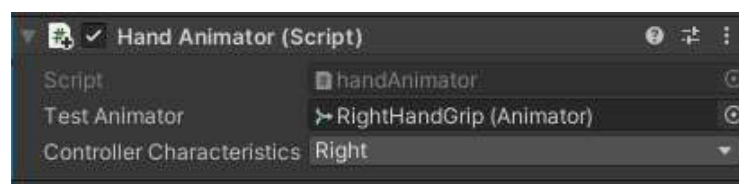


Figure 47 - HandAnimator Script in Editor

```

void Start()
{
    //initialisation check
    tryInit();
}

1 reference
void tryInit()
{
    //generate a new list of all VR input devices
    List<InputDevice> devices = new List<InputDevice>();
    //retrieve input device that matches the determined controller(L/R)
    InputDevices.GetDevicesWithCharacteristics(controllerCharacteristics, devices);
    foreach (var item in devices)
    {
        //outputting listed controllers for debugging
        Debug.Log(item.name + item.characteristics);
    }
    //if matching controller found
    if (devices.Count > 0)
    {
        //set the inputted target to the first value in the device list
        targetDevice = devices[0];
    }
}

// Update is called once per frame

```

Figure 49 - HandAnimator Script

```

Unity Message | 0 references
void Update()
{
    //if able to get value of trigger
    if (targetDevice.TryGetFeatureValue(CommonUsages.trigger, out float triggerValue))
    {
        //pass in the trigger value retrieved to the animator
        testAnimator.SetFloat("Trigger", triggerValue);
    }
    else
    {
        //if can't get value, it's not being pressed
        testAnimator.SetFloat("Trigger", 0);
    }
    //if able to get a value of grip
    if (targetDevice.TryGetFeatureValue(CommonUsages.grip, out float gripValue))
    {
        //pass in the grip value in to the animator
        testAnimator.SetFloat("Grip", gripValue);
    }
    else
    {
        //if can't get value, it's not being pressed
        testAnimator.SetFloat("Grip", 0);
    }
}

```

Figure 48 - HandAnimator Script



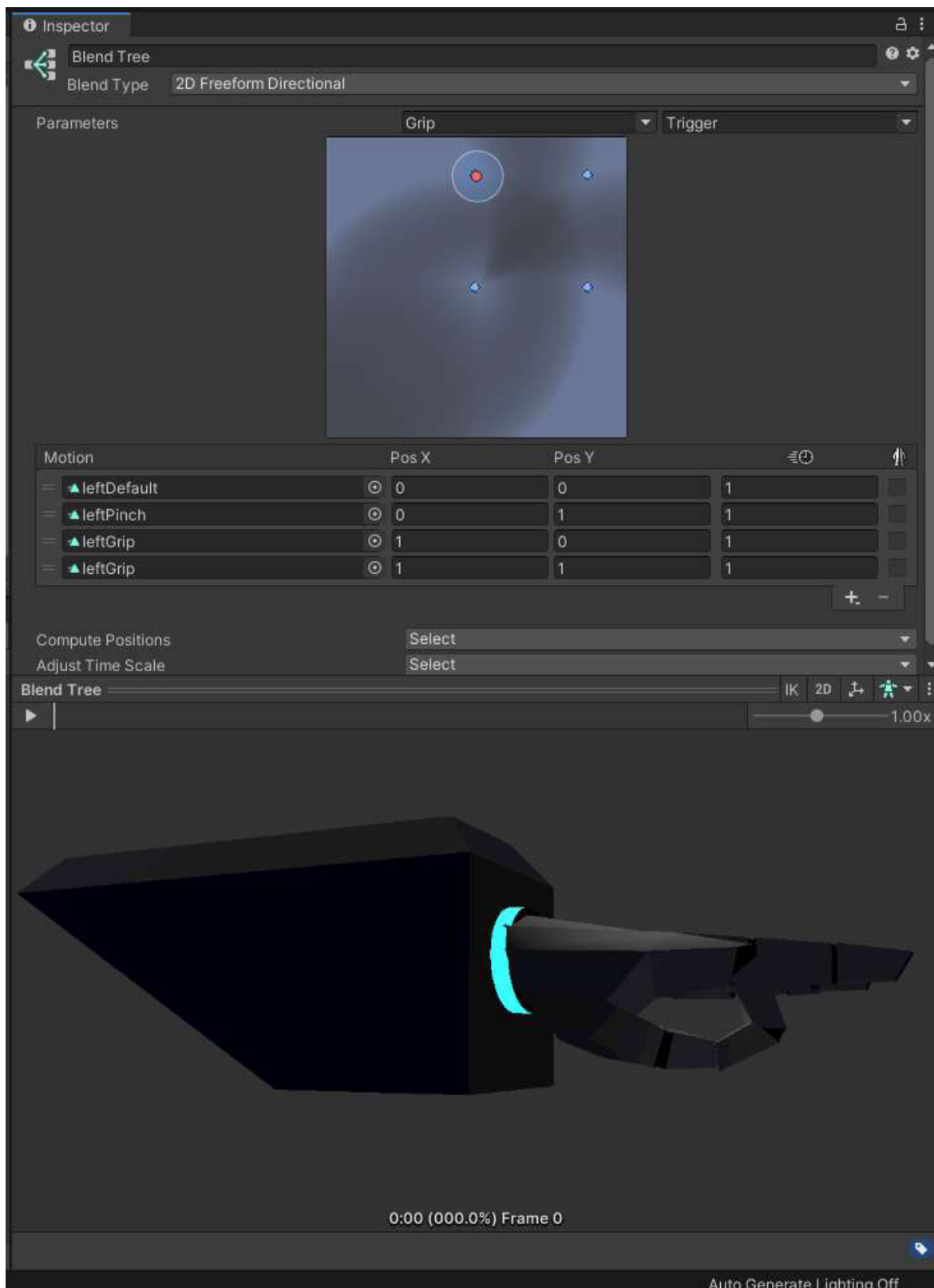


Figure 50 - Blend Tree Pinch Demonstration

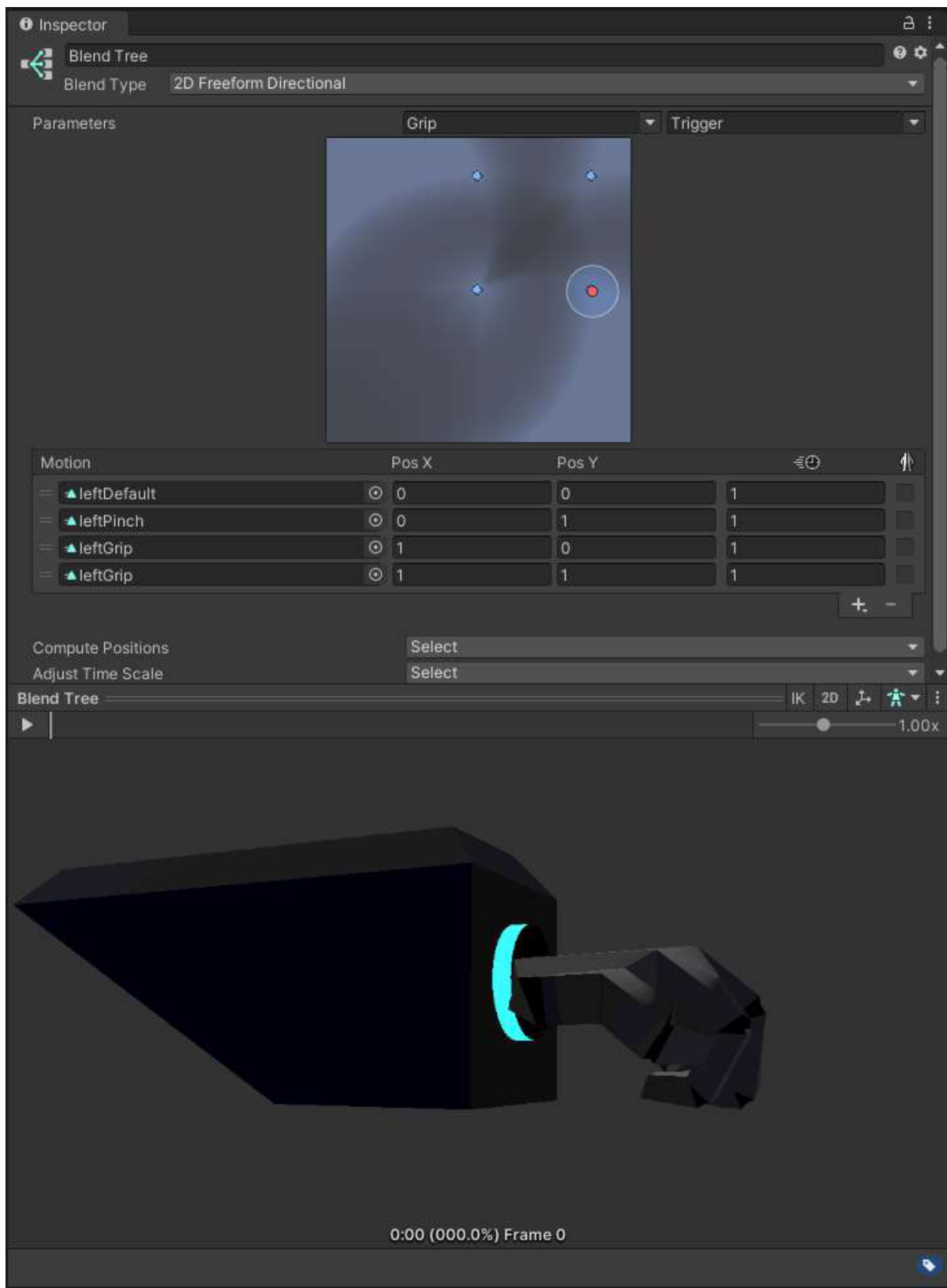


Figure 51 - Blend Tree Grab Demonstration



## Custom Scripts

This section details the noteworthy custom C# scripts created to perform specific functions in the application that were not available in any predetermined scripts or plugins nor in any community provisioned tutorials that were found at the time of development.

### Lever Joints

Premade physics levers and buttons do already exist as a set of assets downloadable for free from the asset store (UtilityFunction, 2016) but this was not downloaded in favour of using simpler methods and the predetermined interactions applicable to XR Grab Interactors configurable within the editor as opposed to scripts when regarding buttons. This set of assets was also the only free set of assets, and last updated in 2016, so it is a safe assumption that there is little to no compatibility with the current XR Toolkit implementation of VR interactions.

The script created for the application is named “hingeJointListener.cs” and has an identical variant named “noSoundListener.cs” that is also used. The listener is assigned to the lever objects directly and can accommodate which lever is being used, as all 3 levers serve separate purposes listed in separate methods in Figure 52. “noSoundListener.cs” is the commented version of the code, originally made the main version of the script as the implementation of sounds was intended but scrapped due to its tendency to break the builds as well as lack of quality implementation. “hingeJointListener.cs” is identical in function with the sound actions commented out and no overall comments for function. The scripts remained separated to allow for different speeds of crane control between the introduction crane and the main lever crane implemented. The code is commented to provide context, and the three “check” functions are all functionally identical, but manipulate different aspects of the manipulated objects transform, so only one is provided in Figure 53.

```

@ Unity Message | 0 references
private void FixedUpdate()
{
    //fixedupdate to solve issues with framerate affecting physics of manipulations
    {
        //if parent object name is left
        if (direct.name.ToString() == "LeftLever")
        {
            //only perform left
            checkLeft();
        }
        //if parent object name is middle
        if (direct.name.ToString() == "MidLever")
        {
            //only check middle
            checkMid();
        }
        //if parent object name is right
        if (direct.name.ToString() == "RightLever")
        {
            //only check right
            checkRight();
        }
    }
}

```

Figure 52 - Joint Listener Script

```

private void checkLeft()
{
    //check whether past 20 degrees of center forward
    if (hinge.angle < (centerAngle + 20))
    {
        //if forward, multiply the up vector by the hinge value
        manipulate.transform.position += Vector3.up * ((hinge.angle - centerAngle) / 800);
    }
    //if backwards
    else if (hinge.angle > (centerAngle - 20))
    {
        //multiply the up vector by the new value
        manipulate.transform.position += Vector3.up * ((hinge.angle - centerAngle) / 800);
    }
    //else do nothing
    else
    {
        //do nothing
    }
}

```

Figure 53 - Joint Listener Script

## Screen Fader

To allow for a smooth and visually appealing way to transition between scenes, it was determined that the implementation of fading the screen to black and back whilst scenes loaded in the background was the most suitable way of doing so. Whilst looking for tutorials on this subject, it transpired that one of the template methods provided (Andrew, 2021) was incompatible with the Unity version used in development without major debugging, and there was little to no explanations as to what may have caused compatibility issues that were easily solved.

In the comments for the tutorial, one user suggested that the method they would choose to implement would be to implement an image that obscured the main camera's vision and change its transparency value over time. The author states that this is apparently an older method recommended by Oculus within their development tutorials but considering the author's tutorial was not easily compatible with my application, this was the approach I decided to use. It is worth noting that Oculus provide a screen fader in their SDK, and Valve also provide one with their older plugin for Unity (Facebook Technologies, LLC, n.a), (Valve Corporation, n.a) but neither of these were viable to implement without creating another layer of complexity that would have likely caused more development issues than solved.

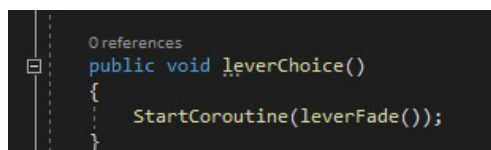
To create the obscuring image, a canvas was attached as a child of the main camera on all XR rigs used in the application, and within the canvas, a large rectangle was placed with a transparency, or alpha, value of 1 out of a maximum of 255. When set to 255, the effect is as shown Figure 55 in the camera preview in the bottom right, as the image is positioned so close to the user's headset, they cannot see past it.

In combination with this image in place, a custom script called "URPScreenFade.cs" was created to be the facilitating method of changing the alpha of the image over time. This script has become quite cluttered over time and would greatly benefit from refactoring, but as it stands it allows for the program to function with some level of intended polish and is quite delicate in its operation methods.

The core of this script is the use of public methods that apply to specific states, like fading from one puzzle to the next, as they can be assigned to interactions in the editor. In these public methods, all that is done is the call of a specific co-routine that performs the specific functions required whilst being able to implement time delays. In Figure 54 is an example method and coroutine, and the actions it is performing, commented for context, and the implemented methods the co-routine calls in Figure 56 and Figure 57.

### Movement Checks

The script "movementChecks.cs" was created to accommodate for the options menu implemented in the starting scene. This script systematically sorts through all aspects related to the provided options and enables and disables the related scripts on the movement-based rigs in the required puzzles. It is divided in to two sections, the start method, and the handedness methods, Figure 58 and 59. The code shown is commented to provide context, though only the right-hand section of the handedness method is shown as the left is functionally identical, but the operations are reversed.



```
Oreferences
public void leverChoice()
{
    StartCoroutine(leverFade());
}
```

Figure 54 - Screen Fade Method Call

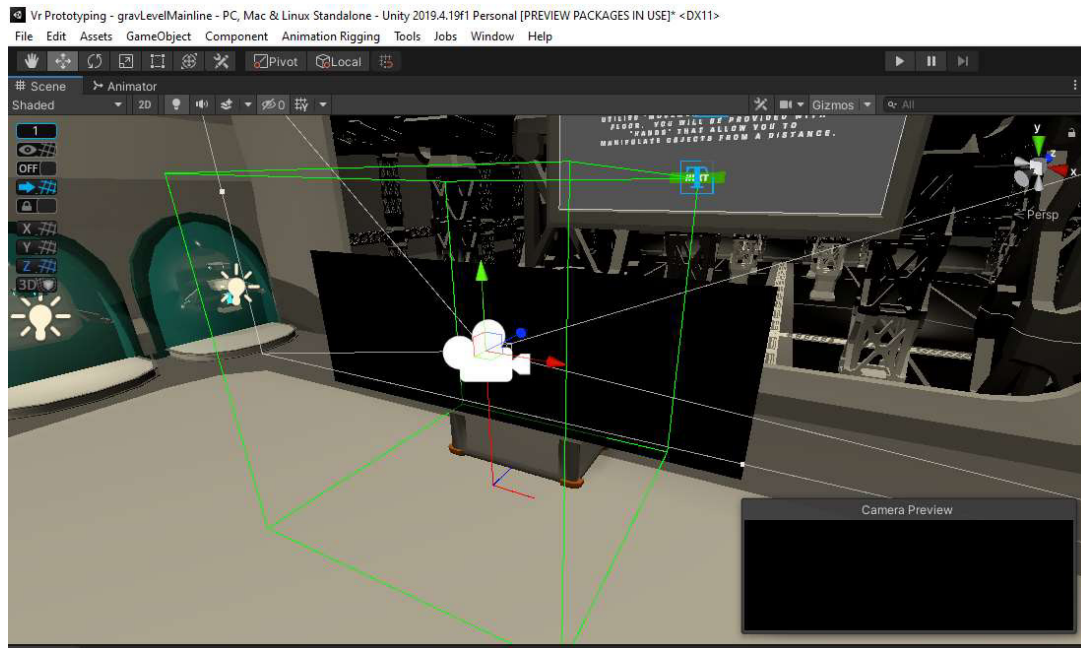


Figure 55 - Screen Fade Implementation

```

1 reference
private IEnumerator leverFade()
{
    //run fade in to set screen to black over 1 second
    scenefadeIn();
    //wait 2 realtime seconds
    yield return new WaitForSecondsRealtime(2);
    //disable the pointer rig
    pointerRig.SetActive(false);
    //disable the pointer puzzle if it is somehow active
    pointerPuzzle.SetActive(false);
    //enable the lever level rig
    leverRig.SetActive(true);
    //enable the lever puzzle
    leverPuzzle.SetActive(true);
    //set the screen to fade out over 1 second
    scenefadeOut();
}
1 reference

```

Figure 56 - Screen Fade Co-Routine

```

13 references
private void scenefadeIn()
{
    //this is the the way the image is faded from clear to black across 1 second
    fadeimage.CrossFadeAlpha(255f, 1.0f, false);
}

8 references
private void scenefadeOut()
{
    //this is the way the image is faded from black to clear across 2 second
    fadeimage.CrossFadeAlpha(1f, 1.0f, false);
}

```

Figure 57 - Scene Fade Alpha Calls



```

public void handedness()
{
    //if player hand is selected as right in options, disable and enable sections of scripts to reflect control
    if (PlayerPrefs.GetString("primaryHand") == "RightPrimary")
    {
        //set main source to left for cont movement
        rig.GetComponent<ContinuousMovement>().inputSource = leftMovement;
        //set right controller as the snap turning control
        rig.GetComponent<DeviceBasedSnapTurnProvider>().controllers.Add(rightContMovement);
        //set right controller as smooth turning control
        rig.GetComponent<DeviceBasedContinuousTurnProvider>().controllers.Add(rightContMovement);
        //set the left hand as the teleport controller
        rig.GetComponent<LocomotionController>().enableLeftTeleport = true;
        //disable the right hand teleport controller
        rig.GetComponent<LocomotionController>().enableRightTeleport = false;
        //enable the right hand gravity model
        rightGrav.SetActive(true);
        //disable the left hand gravity model
        leftGrav.SetActive(false);

        //disable ray interactor and line rendered on left hand for grabbing puzzle objects
        leftContMovement.GetComponent<XRRayInteractor>().enabled = false;
        leftContMovement.GetComponent<LineRenderer>().enabled = false;
        leftContMovement.GetComponent<XRInteractorLineVisual>().enabled = false;
        //double check right hand line renderer and ray interactor are enabled
        rightContMovement.GetComponent<XRRayInteractor>().enabled = true;
        rightContMovement.GetComponent<LineRenderer>().enabled = true;
        rightContMovement.GetComponent<XRInteractorLineVisual>().enabled = true;
    }
}

```

Figure 58 - Handedness Checks

```

Unity Message | 0 references
void Start()
{
    //multiply the user's speed by the specified multiplier from the options menu
    rig.GetComponent<ContinuousMovement>().speed = (rig.GetComponent<ContinuousMovement>().speed) * PlayerPrefs.GetFloat("moveSpeed");
    Debug.Log("SpeedSet");
    //set the snap turning angle based on the user inputted value
    rig.GetComponent<DeviceBasedSnapTurnProvider>().turnAmount = PlayerPrefs.GetFloat("snapAngle");
    //if teleport is selected by user, disable all irrelevant scripts to use teleport only
    if (PlayerPrefs.GetString("moveType") == "Teleport")
    {
        Debug.Log("TeleEnabled");
        //enable teleprovider
        rig.GetComponent<TeleportationProvider>().enabled = true;
        //enable locomotion controller (for teleport)
        rig.GetComponent<LocomotionController>().enabled = true;
        //disable continuous
        rig.GetComponent<ContinuousMovement>().enabled = false;
    }
    //if continuous is selected by user, disable all irrelevant scripts to use cont. only
    if (PlayerPrefs.GetString("moveType") == "Smooth")
    {
        Debug.Log("ContEnabled");
        //disable tele
        rig.GetComponent<TeleportationProvider>().enabled = false;
        //disable locomotion controller for teleport
        rig.GetComponent<LocomotionController>().enabled = false;
        //enable continuous movement
        rig.GetComponent<ContinuousMovement>().enabled = true;
    }
    //if snap turn is selected by user, disable all irrelevant scripts to use snap only
    if (PlayerPrefs.GetString("turnType") == "Snap")
    {
        Debug.Log("SnapEnabled");
        //disable smooth turn
        rig.GetComponent<ContinuousTurnProviderBase>().enabled = false;
        //enable snap turn
        rig.GetComponent<SnapTurnProviderBase>().enabled = true;
    }
    //if smooth turn is selected by user, disable all irrelevant scripts to use smooth only
    if (PlayerPrefs.GetString("turnType") == "Smooth")
    {
        Debug.Log("SmoothEnabled");
        //enable smooth turn
        rig.GetComponent<ContinuousTurnProviderBase>().enabled = true;
        //disable snap turn
        rig.GetComponent<SnapTurnProviderBase>().enabled = false;
    }
    //run the handedness check to then set which inputs need to be used for the remaining enabled scripts
    handedness();
}

```

Figure 59 - Handedness Checks

## Questionnaire Content

The content of the questionnaire realised during the development window while researching other VR studies had been undertaken. After searching, the previously mentioned study about immersion (Schwind, et al., 2019) suggested the use of the IPQ (iGroup Presence Questionnaire). This questionnaire has been specifically tailored to determining the sense of presence a user feels in a virtual environment, and as such provided an excellent structure to help determine if any trends are present between interaction methods presented in the application (igroup.org, n.a).

In Figure 60 and 61 are the questions in the IPQ are listed that were presented to the user in the application. An additional set of questions were devised, shown in Figure 62 and Figure 63, to be presented as an optional extra section of the questionnaire and ask the user about their experience level with virtual reality as well as how they feel about the medium overall, to try to gain demographic information without getting the user to reveal too much about themselves to maintain anonymity. The second set of questions was devised to see if trends emerged between the experience level of the user and the immersion that they might feel in an application.

The results from the questionnaire will be discussed in the following section.

## Application Distribution

To allow users to take part and engage with the application and the questionnaire, it was decided early in the project's lifespan that the application would be distributed publicly as a personal GitHub project, to allow for users to inspect and investigate the files provided before downloading the application to their own personal computers. This approach was taken to provide a layer of transparency for any prospective users allowing them to both read the participant information leaflet and investigate scripts before engaging with the project in any capacity. Whilst taking part in the questionnaire is not mandatory, it was made clear that the application's primary focus is to gather data though the questionnaire, so it is unlikely that anyone interacted with the project without engaging in the application as well.

The GitHub repo was then posted to two reddit forums, one focused on virtual reality (Reddit, 2009), and the other focused on the Valve Index device (Reddit, 2019) that the application was developed on. These were chosen because they seemed to have a good balance of giving the application a chance to get in front of a lot of users that would likely meet the requirements of having a VR ready PC as well as being over the age of 18. The posts unfortunately did not draw enough attention to draw large numbers of participants, but enough to gather results to analyse.

**How aware were you of the real world surrounding while navigating in the virtual world?  
(i.e. sounds, room temperature, other people, etc.)?**

extremely aware	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	not aware at all
	-3	-2	-1	0	+1	+2	+3	
moderately aware								

64/inv1/0

**How real did the virtual world seem to you?**

completely real	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	not real at all
	-3	-2	-1	0	+1	+2	+3	

49/real1/1

**I had a sense of acting in the virtual space, rather than operating something from outside.**

fully disagree	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	fully agree
	-3	-2	-1	0	+1	+2	+3	

31/sp4/2

**How much did your experience in the virtual environment seem consistent with your real world experience ?**

not consistent	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	very consistent
	-3	-2	-1	0	+1	+2	+3	
moderately consistent								

7/real2/3

**How real did the virtual world seem to you?**

about as real as an imagined world	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	indistinguishable from the real world
	-3	-2	-1	0	+1	+2	+3	

59/real3/4

**I did not feel present in the virtual space.**

did not feel	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	felt present
	-3	-2	-1	0	+1	+2	+3	

28/sp3/5

Figure 60 - Questionnaire Content 1

**I was not aware of my real environment.**

fully disagree

☐
☐
☐
☐
☐
☐
☐

-3
-2
-1
0
+1
+2
+3

fully agree

37/inv2/6

---

**In the computer generated world I had a sense of "being there"**

not at all

☐
☐
☐
☐
☐
☐
☐

-3
-2
-1
0
+1
+2
+3

very much

62/g1/7

---

**Somehow I felt that the virtual world surrounded me.**

fully disagree

☐
☐
☐
☐
☐
☐
☐

-3
-2
-1
0
+1
+2
+3

fully agree

44/sp1/8

---

**I felt present in the virtual space.**

fully disagree

☐
☐
☐
☐
☐
☐
☐

-3
-2
-1
0
+1
+2
+3

fully agree

33/sp5/9

---

**I still paid attention to the real environment.**

fully disagree

☐
☐
☐
☐
☐
☐
☐

-3
-2
-1
0
+1
+2
+3

fully agree

40/inv3/10

---

**The virtual world seemed more realistic than the real world.**

fully disagree

☐
☐
☐
☐
☐
☐
☐

-3
-2
-1
0
+1
+2
+3

fully agree

47/real4/11

---

**I felt like I was just perceiving pictures.**

fully disagree

☐
☐
☐
☐
☐
☐
☐

-3
-2
-1
0
+1
+2
+3

fully agree

30/sp2/12

---

**I was completely captivated by the virtual world.**

fully disagree

☐
☐
☐
☐
☐
☐
☐

-3
-2
-1
0
+1
+2
+3

fully agree

38/inv4/13

---

Figure 61 - Questionnaire Content 2



Question 1 –

+ where do you consider yourself when playing virtual reality games?

New	Beginner	Intermediate	Advanced
I have little to no experience with any VR titles or experiences	I have played a few casual titles and experiences	I have played some casual titles and tried some intense titles and experiences	I have played a lot of the more intense titles and experiences
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Question 2 –

On average, how much time do you spend using VR weekly?

Minutes or less						Tens of hours or more
-3	-2	-1	0	+1	+2	+3
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Question 3 –

When it comes to VR, I am invested in the space and excited to see it grow.

Strongly Disagree			Neither Agree nor Disagree			Strongly Agree
-3	-2	-1	0	+1	+2	+3
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Question 4 –

I prefer Single-player experiences as opposed to Multiplayer experiences in VR games.

Strongly Disagree			Neither Agree nor Disagree			Strongly Agree
-3	-2	-1	0	+1	+2	+3
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Figure 62 - Optional Content 1

Question 5 –

Mechanics are more important to me than Narrative in VR games.

Strongly Disagree			Neither Agree nor Disagree			Strongly Agree
-3	-2	-1	0	+1	+2	+3
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Question 6 –

VR is the future of games as a medium.

Strongly Disagree			Neither Agree nor Disagree			Strongly Agree
-3	-2	-1	0	+1	+2	+3
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Question 7 –

I play VR games more than playing games traditionally.

Strongly Disagree			Neither Agree nor Disagree			Strongly Agree
-3	-2	-1	0	+1	+2	+3
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Figure 63 - Optional Content 2

## Section 5 – Results and Evaluation

This section of the report provides an overview of the questionnaire results as well as comments on trends that appear in the results, in addition to an overview of the application's requirements testing for the final build pushed to users.

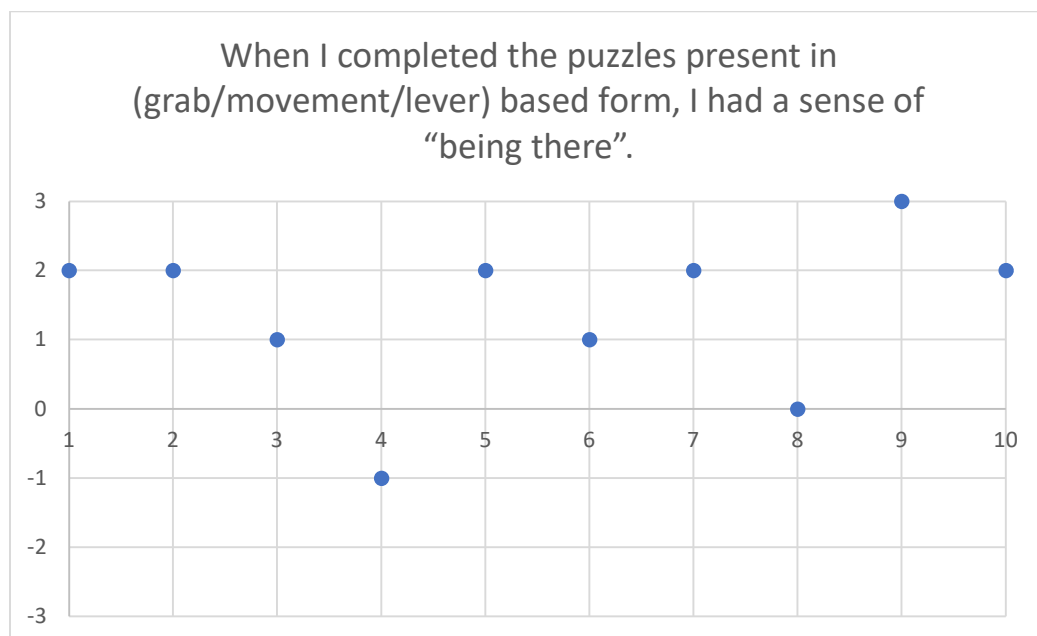
As indicated in Figures 60-63, each of these questions except for the first optional question is graded with a response that varies on a scale from positive to negative, indicating positive as either agreeing or a higher scale value, and negative as disagreeing or a low scale value, except for questions where this is deliberately inverse. The first optional question is graded on a scale of 1 to 4, 4 indicating the user is highly experienced in the VR consumer market and 1 indicating little to no experience.

For the first section of questions, the X axis values represent each anonymous user who took part in the questionnaire, and the Y axis represents the scale value they gave in response. In the optional section, only the users who completed the section will have a value, and users who did not are represented as blank spaces in the graph. Data was taken from the OneDrive link provided to users and entered in a random order in an excel spreadsheet.

### Questionnaire Results

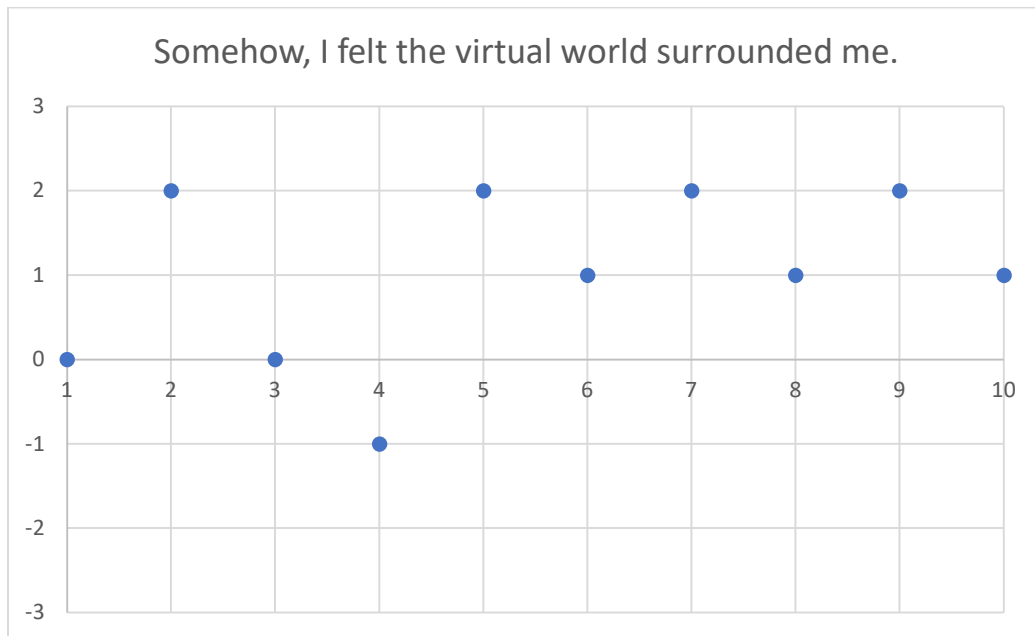
#### Question 1

This question appears to users to show the final choice they selected after completing all 3 methods of interaction. The most common response in this question was +2, and only 1 negative response with an averaged response of +1.4, indicating at the very least that there was some sense of physical presence, but not necessarily indicating immersion.



### Question 2

As with question 1, this question's modal response is +2, with only 1 negative result again coinciding with an average response of +1, with an overall trend towards the positive on the scale. The same principle likely applies here as well however, in that there is at least an indication that there is a sense of presence, but not necessarily immersion.



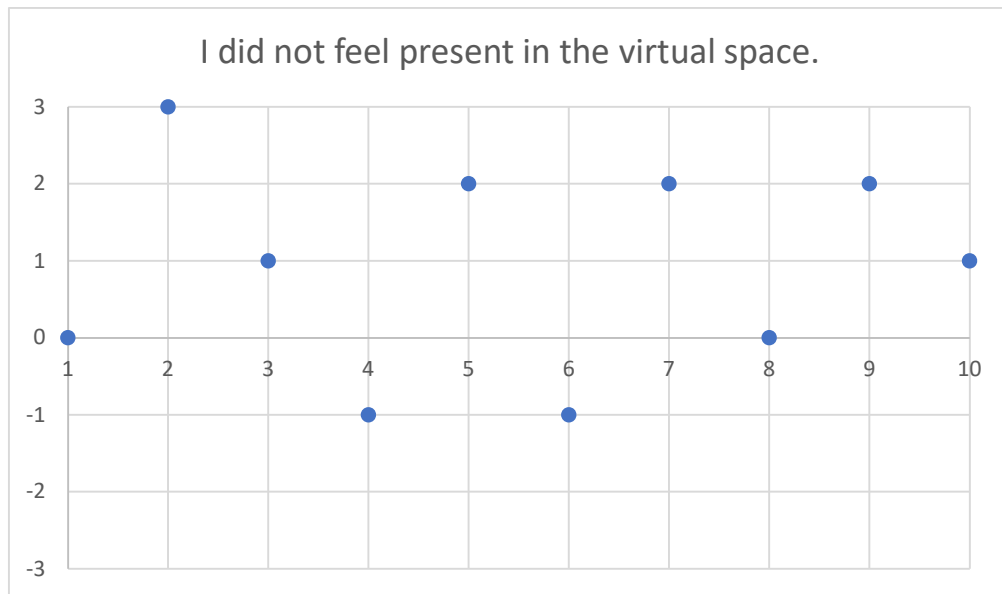
### Question 3

This question deviates slightly from the emerging trend of the previous two, with the modal response being a neutral 0, but only 3 of the 10 participants responded neutrally. The overall trend here is less clear, but at the very least has more negative responses than positives. This question as with the previous two appears to be determining to what extent the hardware the user is equipped with can give a sense of being in the space, as none of them reflect particularly on engagement or immersion.



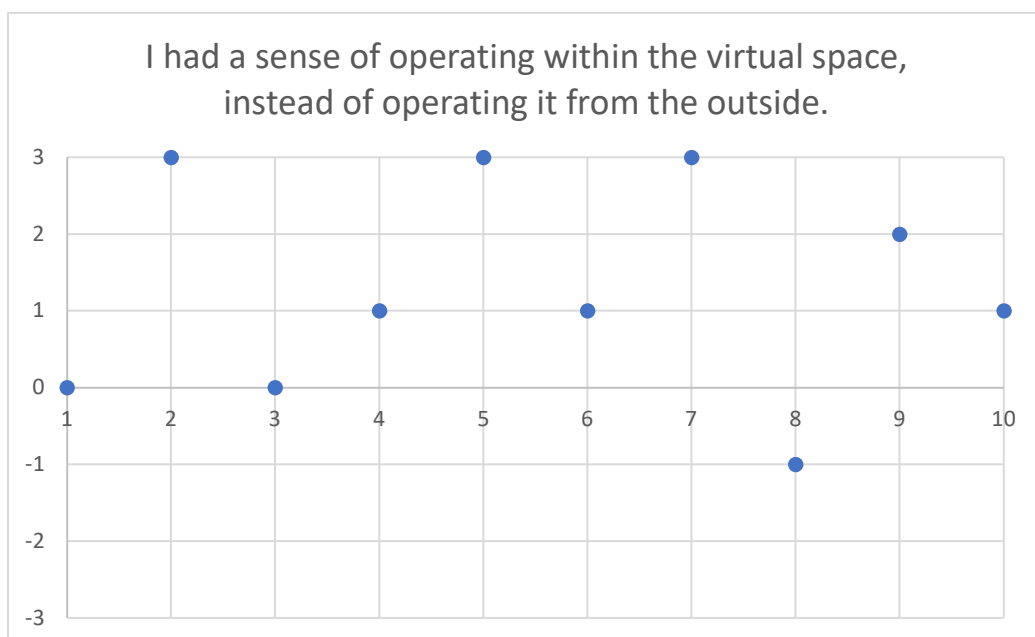
#### Question 4

In this question the user is directly addressed with the term “present”, which as explained in the Presence and Immersion section appears it could have different meanings when referring to VR. Regardless of interpretation, the responses show an overall positive affirmation, with a modal value of +2, and average response of +0.9. There was an increase of 1 negative response in this question as well as maintaining neutral response, which could be down to either understanding of the term as to whether they felt “immersed” or “present”.



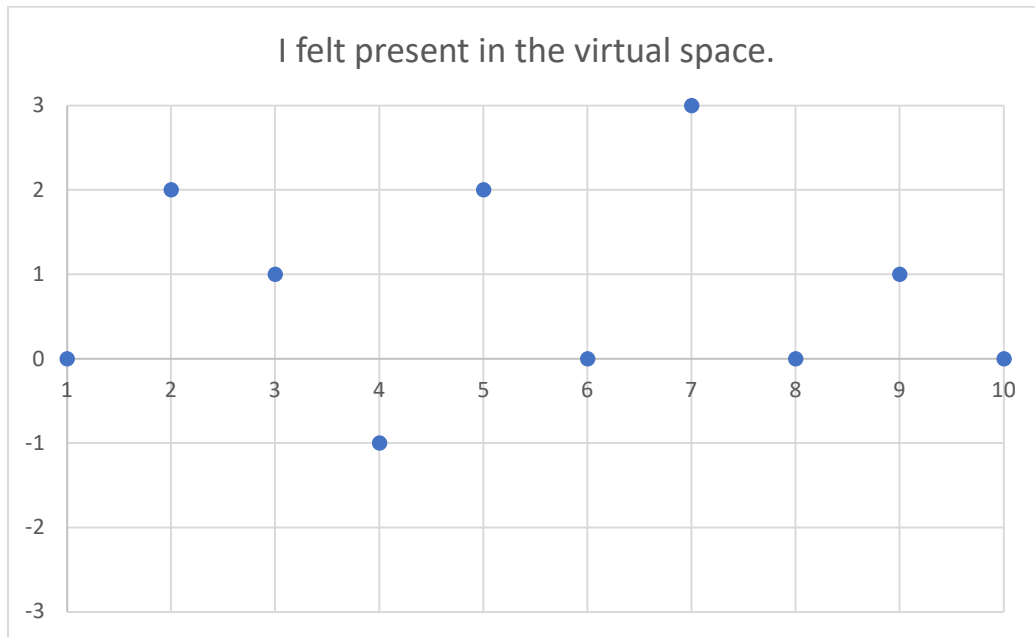
#### Question 5

Here we see another question following the trend set by the first two. This question has a modal value of +3, with 3 responses making up that value, and an average value of +1.3. This question still maintained one negative response, but it is likely user’s feelings about their setups trends towards providing them with a proper sense of presence.



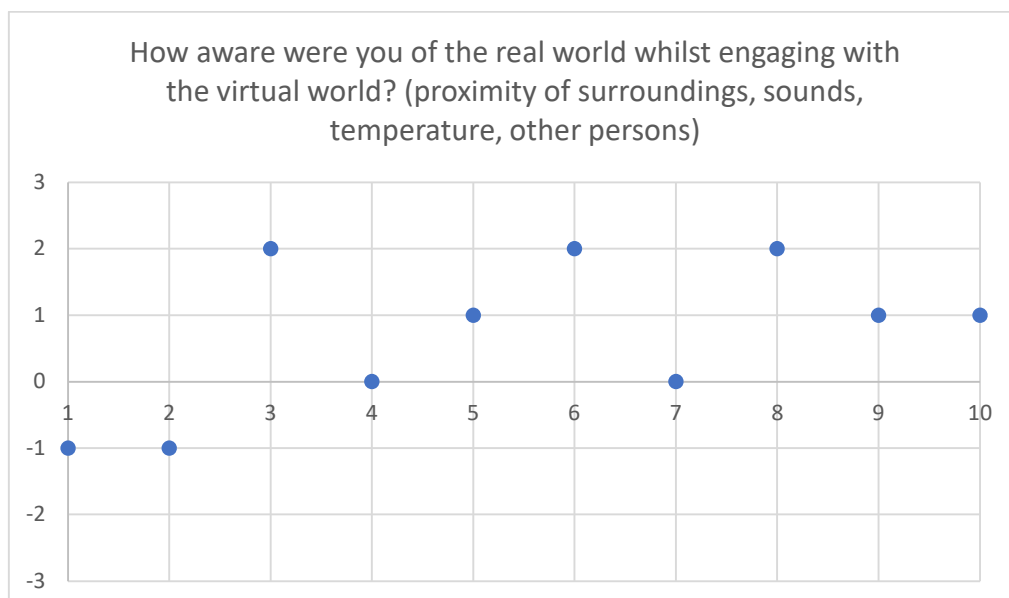
### Question 6

This question is a direct reversal of question 4, instead providing agree or disagree as opposed to present or not present as the scale indicators. There is a modal response of 0 at 4 responses, and an averaged response of +0.8. Notably, the modal value has dropped 2 places with this change in format from question 4, but the averaged response has only decreased by 0.1. This could be due to the scale change provisioning a standpoint from the user as opposed to more binary scale indicators offered in present or not present.



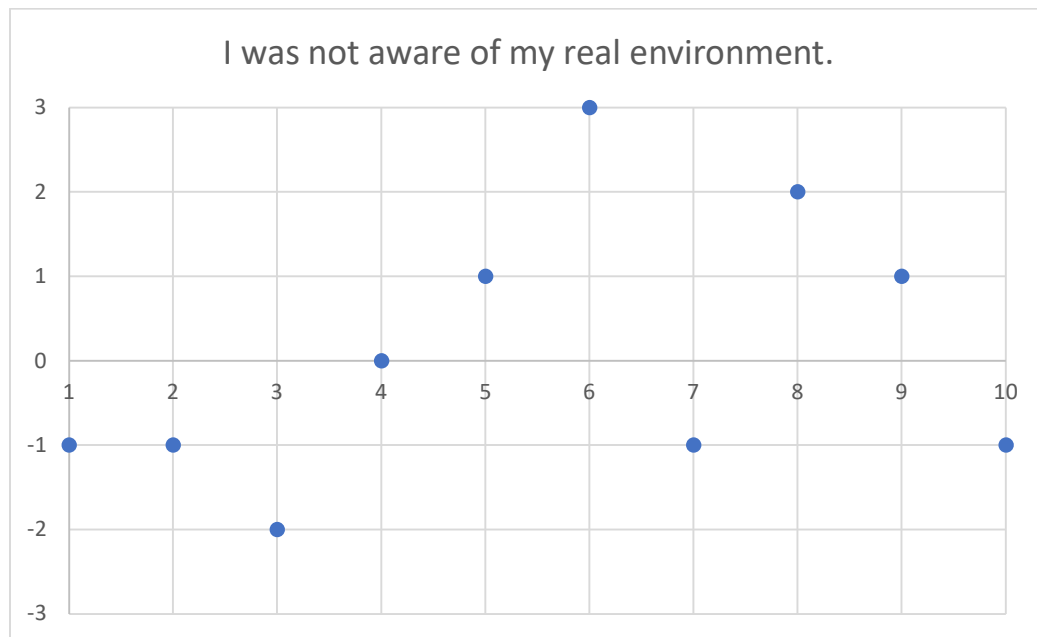
### Question 7

The modal response for this question has both +1 and +2 at three responses each, and an averaged response value of +0.7. The overall positive trend here is likely a result of current VR devices and their design philosophies inherited by older systems, as explained in the section Virtual Reality, to expand and fill the user's periphery and block out surrounding elements to increase the sense of presence.



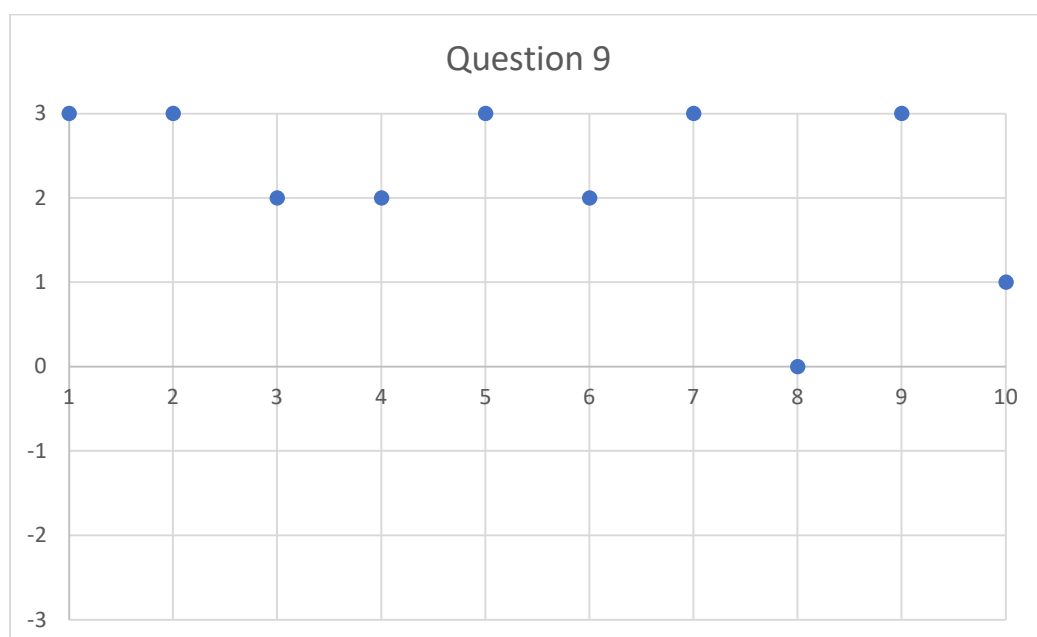
### Question 8

This question similarly follows question 6, in that it is a reversed indicator version of question 7. However, there is a notable change in language as well. The modal response for this question was -1 at 4 responses, and an averaged response of +0.1. This change in responses could be since combining VR with physical space constraints requires trying to maintain some awareness of their space around them, and the change in scale and language could allow this necessary awareness, however minute, to present itself.



### Question 9

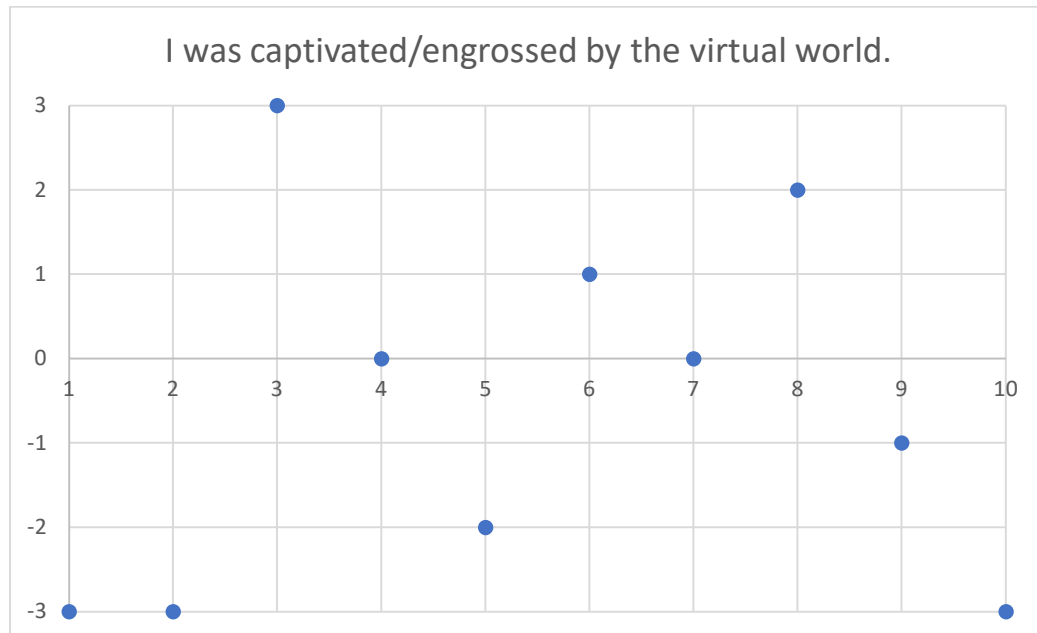
This question could affirm the point made about question 9 with the inclusion of "Still" in the question title. The modal value for this question was +3 at 5 responses with a mean response of +2.2. This again is likely to affirm a sense of presence, and potential subconscious engagement with VR as opposed to with the virtual environment presented.





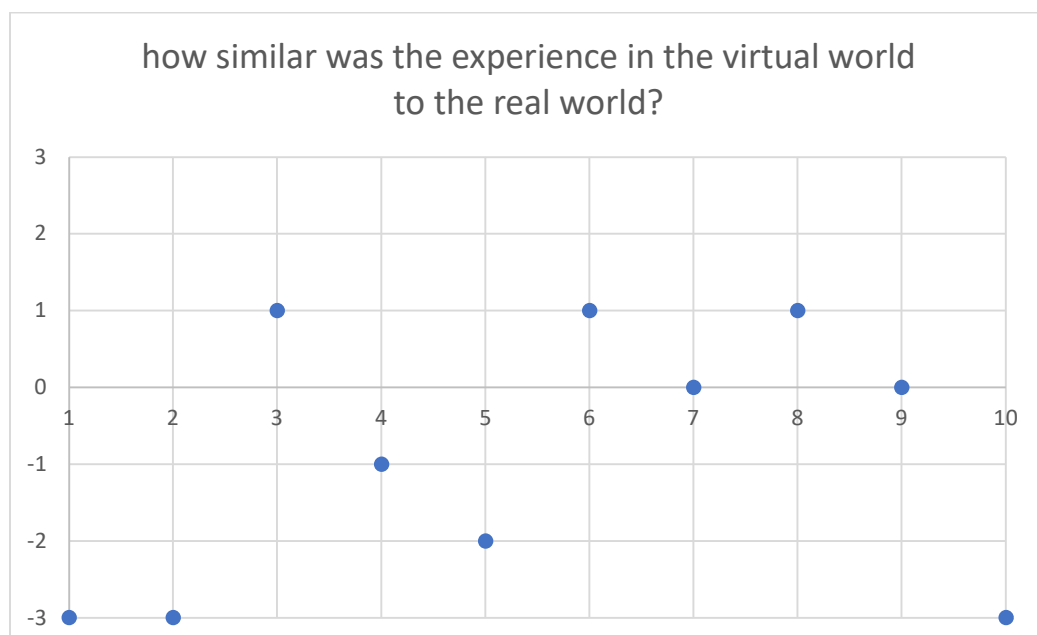
### Question 10

This is the first question that directly references the virtual environment created for the application itself and is where a much more extreme division in results appears. The modal value is -3, with 3 responses, and a mean response of -0.5. This presents a separation from the sense of presence indicated in the previous questions and the environments presented and could indicate that the devised application was less engaging to users than intended.



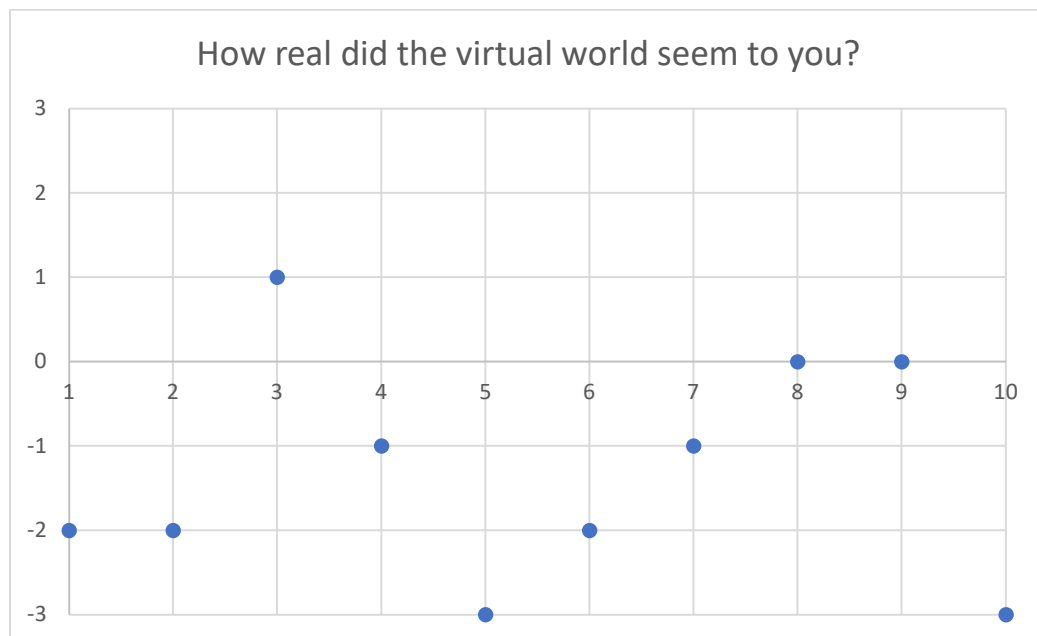
### Question 11

Here we see modal values of +1 and -3, both with 3 responses each, and a mean value result of -0.9. This overall sees a more neutral to negative trend in results but relies on drawing comparison between reality and the virtual environment, which has been dictated by a stylistic choice in the application and presenting the question in this way could lead to this negative trend.



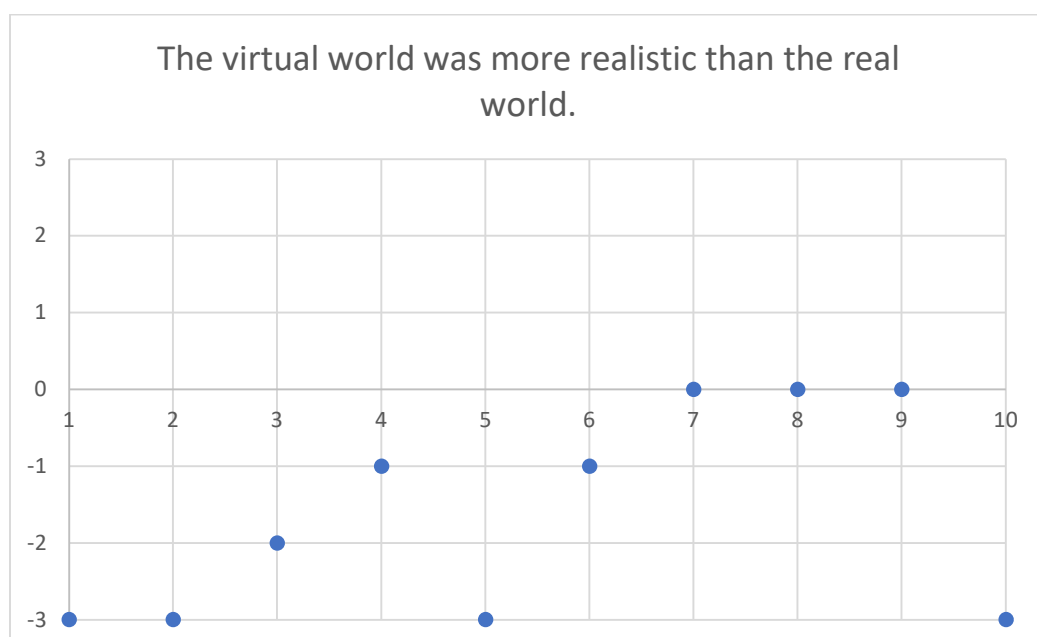
### Question 12

The third question to directly reference the created environment sees a continuing negative trend in responses to the virtual environment. The modal value here is -2 with 3 responses, and a mean response value of -1.3. However, as with the past question, there is a clear comparison between the virtual environment and real environments being made, where this difference was a design choice as opposed to a constraint or accidental.



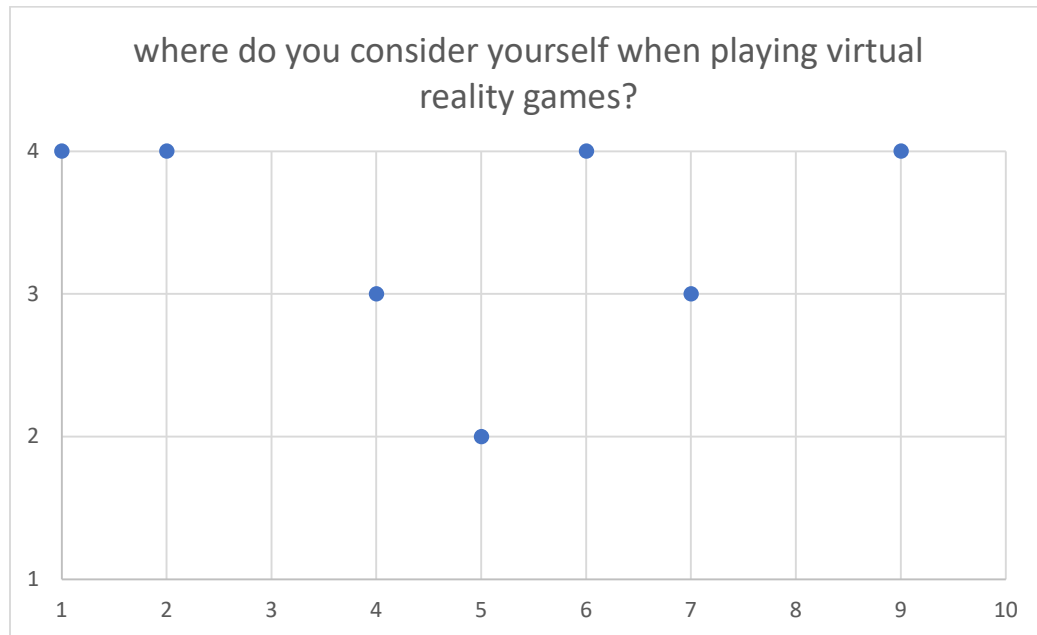
### Question 13

The final question further reinforces the focus on the comparison between reality and the virtual environment and presents another negative trend. The modal value is -3 at 4 responses, and the mean response is -1.4. These questions clearly illustrate that the application is not “realistic”, but this is not a likely indicator for whether the user is immersed or not.



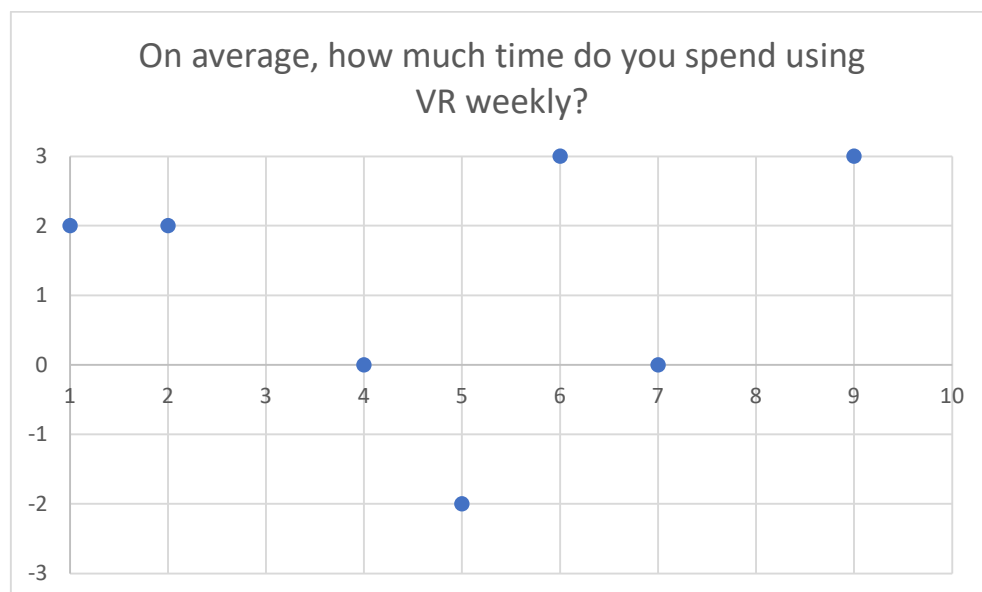
### Optional Question 1

The opening question for the optional questions presents interesting data. Of the 7 users who undertook the optional questions, not one identified themselves as new to the medium. The modal response here was 4 – Advanced User, at 4 responses, and the mean value is 3.43, placing the mean user somewhere between Intermediate and Advanced. This could explain the deviations in responses for how captivated or engaged users were with the application.



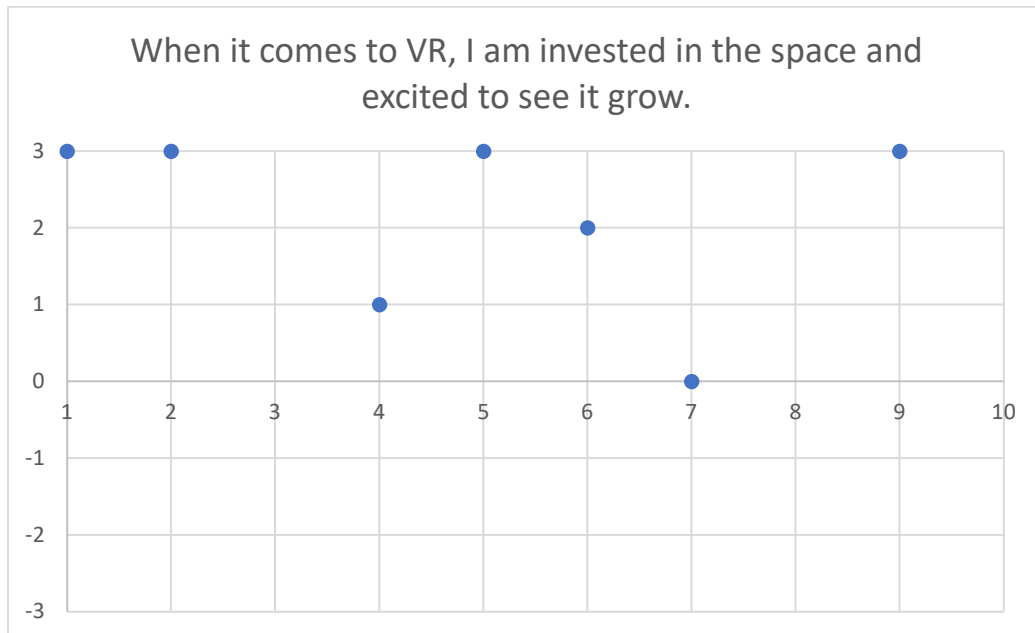
### Optional Question 2

This question was positioned to further determine how much the user engages with VR and as a result whether this could influence a user's opinion on the application. The modal value for this question presents +2, +3 and 0, at 2 responses each. Overall, there does not seem to be a particular trend directly between playtime and engagement in the dataset available but could emerge if conducted with larger and broader participant pools.



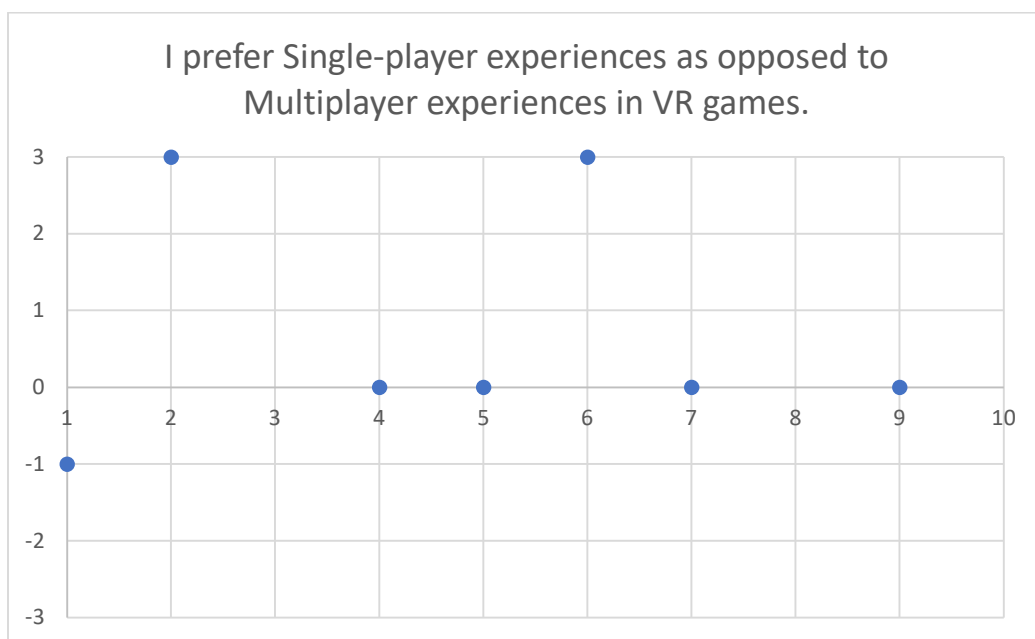
### Optional Question 3

Another question devised to determine the level of intensity that the user participates in VR and its surrounding technologies, this presented a modal value of +3, and a mean value of +2.14. Of note, there is no response lower than a neutral 0, which is likely a result of the platform restrictions in place for the application and the cost requirements to engage with the medium.



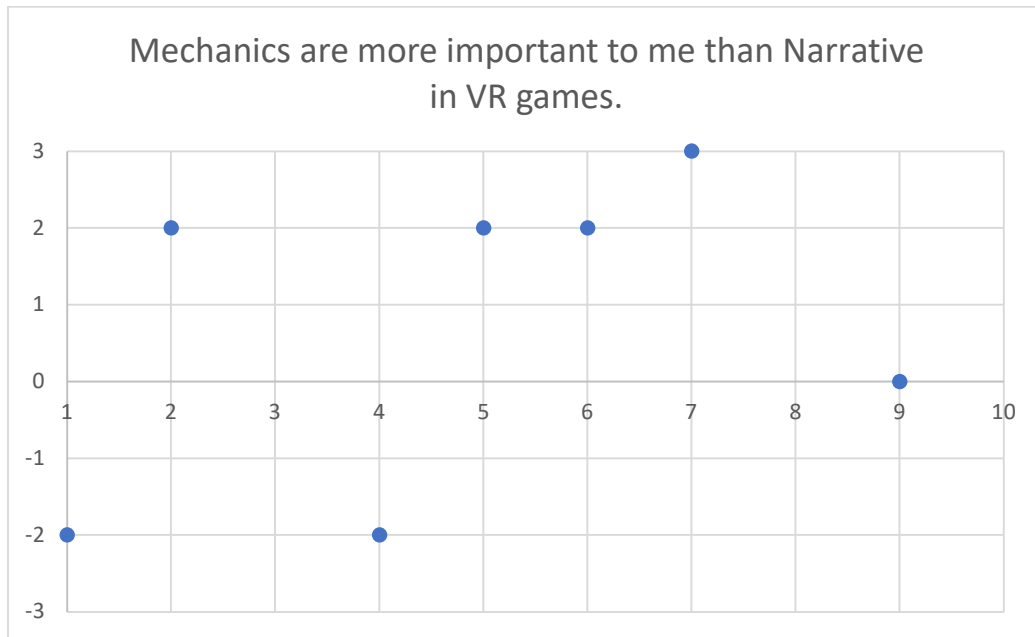
### Optional Question 4

This was a question to see if it could be devised as to whether a user engaged dependent on their preferences in titles. With a modal value of 0 with 4 responses and a mean response of 0.7, this question did not provide any tangible correlations to the engagements presented in question 10, but like question 2, could see a trend emerge with a larger participant pool.



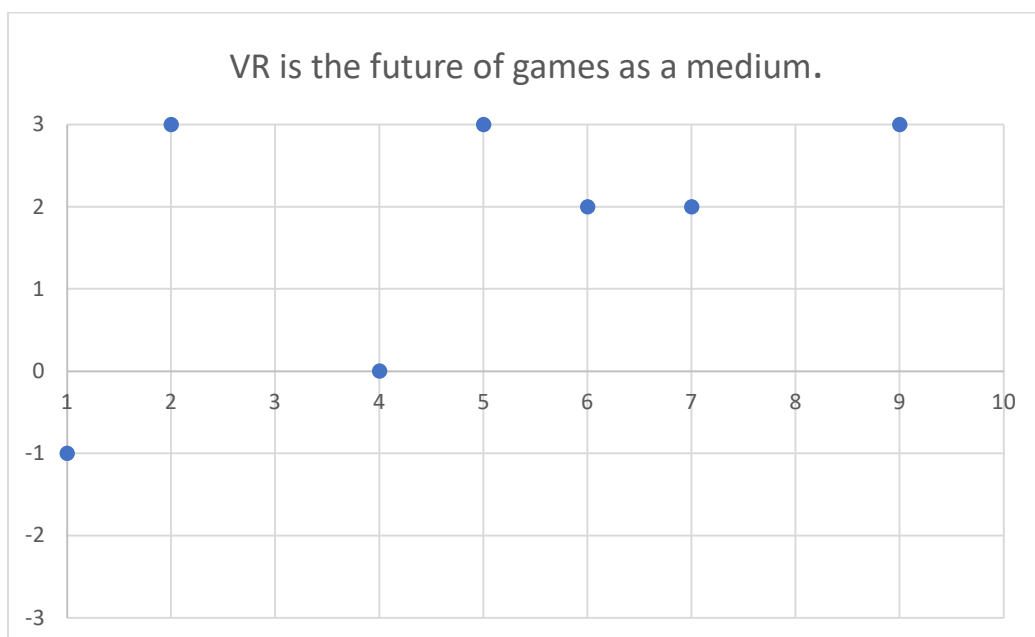
#### Optional Question 5

Like optional question 4, this question was devised to determine if there was a correlation between the data points I engagement in question 10 and whether a user saw a game as more engaging if offered satisfactory mechanical systems to use in the environment. Due to the results of question 10 being so mixed with regards to user engagement, this is another question that is unlikely to be answered with the current dataset. This question had a modal value of +2 with 3 responses, and a mean response of +0.7.



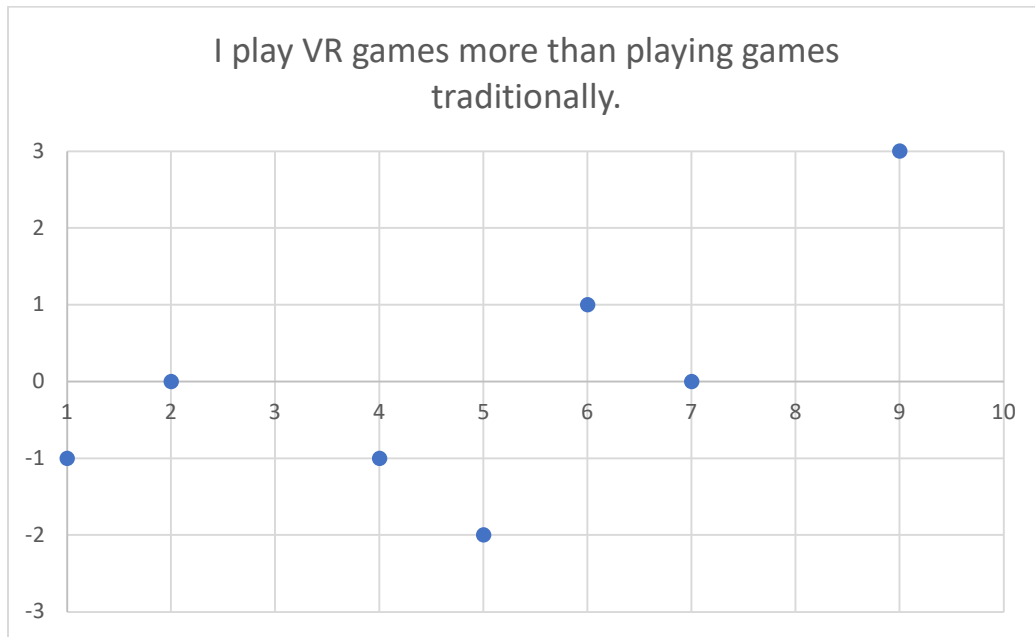
#### Optional Question 6

Like optional question 3, this question was devised to further assess the user's level of investment into VR as an emerging technology. With a modal response of +3 with 3 responses and a mean response of +1.7, there is a definite clear trend that those who have invested into ecosystems for PC based VR believe it is worth pursuing.



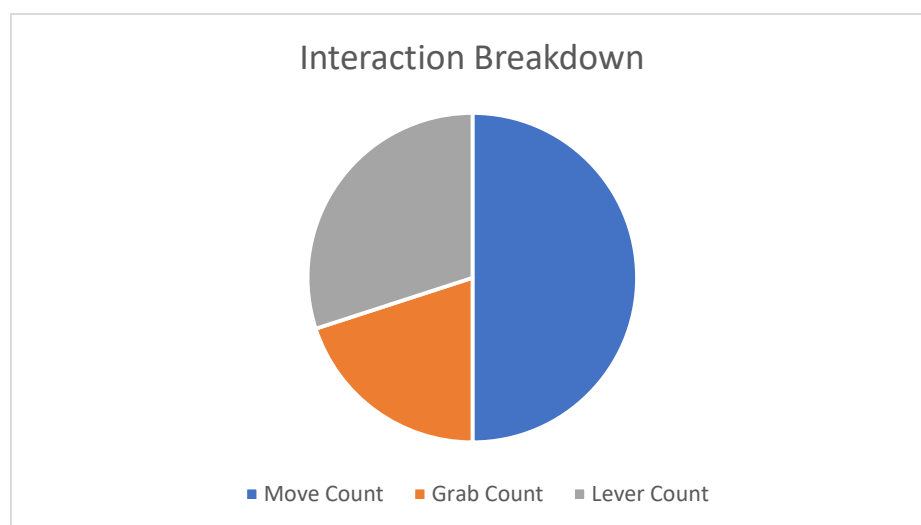
### Optional Question 7

This was the final question to attempt to determine some form of demographic within users and how that may affect how they engage with the application. The modal values were -1 and 0 with 2 responses each, and an average response of neutral 0. This question could present more interesting results with a larger dataset, but it could be assumed that those heavily invested in PC based VR are also playing a lot of regular games to take full advantage of their powerful PCs.

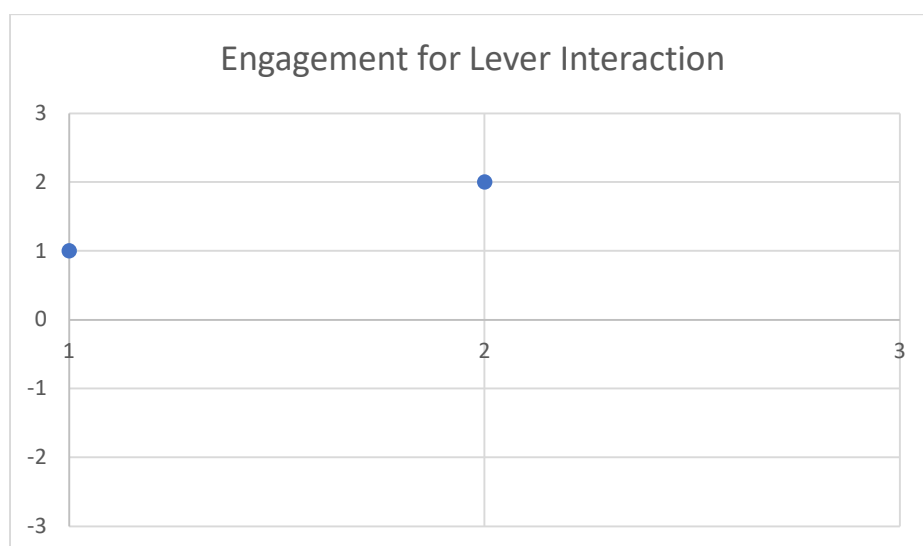
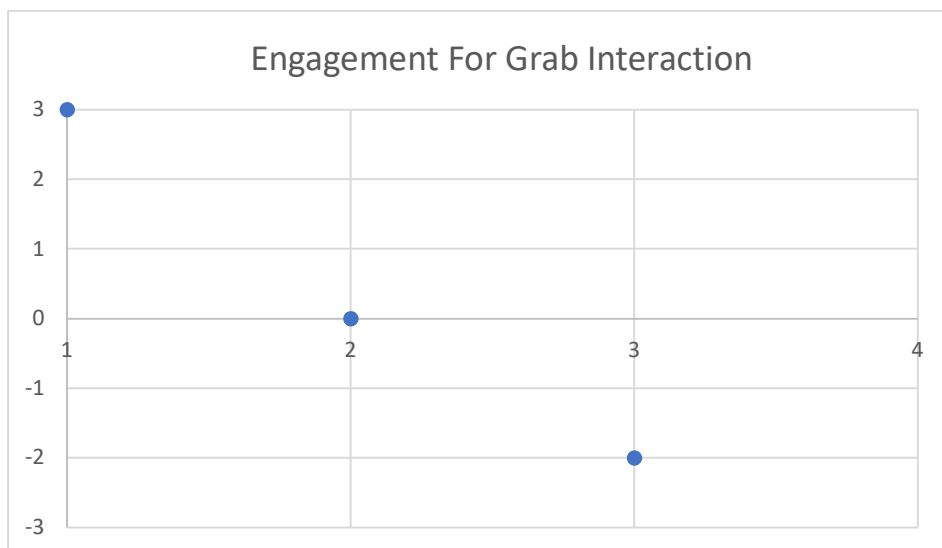
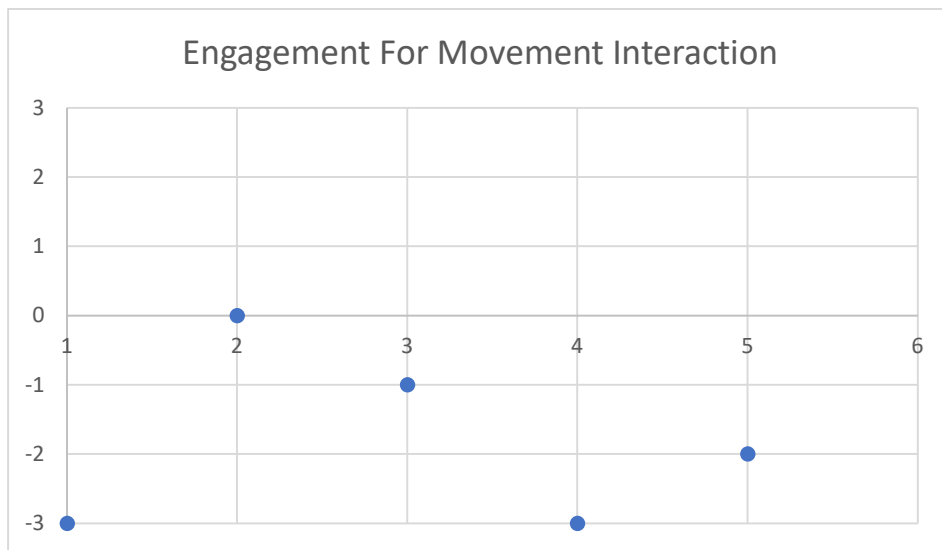


### Methods of Interaction

This pie chart below shows the overall representation of the 10 users and their selection of method. 50% of users chose to complete the final puzzle with movement-based interactions, and 30% chose lever interactions, with the remaining 20% opting for the grab interactions. Considering the high frequency of advanced to intermediate users in optional question 1, the selection of movement could be due to it most closely resembling the default method of environmental navigation in most other applications.



In addition to breaking down the most used method of interaction, the results from question 10 regarding user interaction were grouped by interaction methods, shown in the graphs below.





While the datasets presented in these graphs are clearly too small a sample to provide basis for reliable analysis, the most used interaction method of Movement does begin to present a clearer sense of disengagement when compared to the two users who completed the puzzles with lever interaction. There are several reasons why this could be the case, and this trend could not be present in a larger dataset. It could be speculated that the level of experience the user already has with VR prior to taking part in the study affects whether simpler environments and puzzles engage them. As a direct result of the design of the application, these users may have been disengaged with the application, but still maintained a sense of presence. With the data gathered there appears to be a stronger tendency for experienced and advanced users to pick the Movement interaction method, which could add to the theory that the application not engaging users could be a result of these users having more experience with similarly controlled applications that are more engaging.

### Requirements Testing

This section details the original sets of functional and non-functional requirements to determine if the application was able to meet the initial goals devised for the final application.

#### Functional

##### Must

Requirement	Pass/ Fail	Comments/Justification
Use Unity to provide a SteamVR compatible application	PASS	Application has native SteamVR support and can be launched with SteamVR running
Present users with Participant Information and Consent forms before they progress, and be playable without consent	PASS	Participants must fill out all consent fields present in the presented method before they are allowed to take part in the questionnaire, play without consent does not provide questionnaire to users who opted out
Illustrate the interaction methods users will be presented with during the course of the application	PASS	Tutorial puzzles and written explanations are provided for each puzzle as the user begins that section of the application
Provide 3 methods of interacting with the environment for users to solve puzzles with, in a linear fashion	PASS	Lever interaction, Grab interaction and Movement interaction all successfully implemented
Allow users to choose one of the 3 methods experienced for the 4 <sup>th</sup> and final puzzle	PASS	Choice can be made and appropriate puzzle loads and can be completed
Allow users to take part in the questionnaire inside of the application at the end of the 4 <sup>th</sup> puzzle and provide instructions for remote participation once completed	PASS	Questionnaire is presented and upon application exit, .txt containing instructions and folder containing .txt are opened for the user to see on the desktop.

### *Should*

Requirement	Pass/ Fail	Comments/Justification
Allow for some level of physics interactions between objects the user interacts with and the environment	PASS	Physics implementations between objects present in all 3 methods of interaction
Provide the user with a visual representation to determine their position in the environment	PASS	Hand models provided to demonstrate where user is in the virtual space

### *Could*

Requirement	Pass/ Fail	Comments/Justification
Add higher customisation to visual quality in regarding performance settings	FAIL	No visual settings were implemented in this application that are easily configurable, besides from default Unity implementations in config files
Implement Pause/Resume states	FAIL	No pause resume was implemented, application must be completed in a single sitting
Integrate modularity in design to allocate for adding or removing methods of interaction or additional development	PASS	Design separates functions into scenes and objects and allows for them to be enabled and disabled where needed. Interaction methods modular enough to be able to be implemented in the same scene without issues

### Non-Functional

#### *User Interactions*

Requirement	Pass/ Fail	Comments/Justification
Visual feedback and animations	PASS	User's hands animate upon pressing grip or trigger buttons on VR device controller
Add active sounds to interactions	FAIL	Sound implementations did not make it to final build due to unsatisfactory implementation

### Usability and Accessibility

Requirement	Pass/ Fail	Comments/Justification
Text for user instruction must be clear and visible at a comfortable eyeline to be read	PASS	Text is given a good contrast and is at comfortable eye level in all sections where reading is required
Provide options at the beginning of the application for users to set preferences	PASS	Options section presented at main menu for users wishing to configure settings beyond assumed defaults

### Environment Design

Requirement	Pass/ Fail	Comments/Justification
Demonstrate core concepts for environments and player models	PASS	Environments mapped out clearly with basic shapes and primitives
Have environments and player models be more developed and visually interesting	PASS	Environments further fleshed out using Prefabs and Custom player models as environmental decoration – player hands custom modelled
Have a consistent and engaging environmental and player design development	PASS	Player hands and environment share colour schemes and industrial design
Add ambient sound to environments	FAIL	Sound implementation removed before final build due to unsatisfactory implementation

### Security

Requirement	Pass/ Fail	Comments/Justification
Application needs to preserve user anonymity	PASS	Instructions provided to users to maintain total anonymity
User's information security maintained	PASS	Users must manually upload files themselves if they wish to do so, to a secure Cardiff OneDrive

### User Engagement

Requirement	Pass/ Fail	Comments/Justification
Provide well designed and engaging puzzles for the user to interact with	PASS	4 puzzles with consistent design philosophy presented to users in final build
Construct a short or compelling narrative to tie the puzzle sections together	FAIL	Narrative focus decided as option early on but lost focus as development continued, some loose narrative provided to give context for puzzles

## Results Evaluation

The data gathered from the survey is interesting to parse through to evaluate the demographics of users taking part and how they responded to the application but is unlikely to be of any real-world use for drawing conclusions with regards to the planned method of investigation. In addition to this, it is clear the application delivered had an overall failure to immerse users in the game, when considering whether they “felt” the virtual environment was real. This could be due to stylistic choices, gameplay mechanics not engaging users, or any number of other factors not accounted for. However, with the small number of results gathered, a potential trend can be seen in most users having a sense of presence regardless of their level of engagement with the application developed.

While not able to contact individual users to divine what they would state is “immersive” for them, the data gathered does appear to show most levels had a baseline level of presence, but not necessarily engagement or immersion. The original method that was set out to determine immersive factors is inconclusive when looking at the data gathered and evidently had varying degrees of success engaging users, but in doing so has inadvertently presented data on the fact within users who took part, all but one of them had a semblance of presence that is likely provided by the hardware itself and the stereoscopic 3D effect of VR.

This unintentional result indicated that the headsets available on the market are also in themselves factors that can clearly affect a user’s sense of immersion in the virtual environment as opposed to the virtual environment solely. Whilst different headsets providing different experiences was an oversight of the initial design of the study and could have been factored in if in-person testing had been possible, the overall sense of presence provided by whatever consumer headsets used in the study is have all clearly made significant strides in presence since consumer VR emerged in 2016.

Alongside this, it is evident that more data for different applications should have been gathered to better draw comparisons about the data retrieved from the created application in this project. Timed testing or tailored gameplay sections of highly engaging games such as Boneworks and Half-Life Alyx could have been undertaken with separate groups to retrieve better baseline results in how users compare the real world to the virtual world, before trying to start production of the final application. The result of this would be to better inform the aspects that are key to a user feeling immersed and what makes them feel present and subsequently include them in the testing application. This could also have been focus tested with VR users before setting out the application’s design principles.

## Section 6 – Future Work

By the end of the project timeline, much of the core work set out in the requirements was achieved, but many ideas and tweaks thought to be simple were in fact complex and much more challenging than initially accommodated for.

The first major part of the application that was unintentionally disregarded was the overall sound design. Implementation of sound is simple within Unity, but when testing ambient sounds and music towards the end of development raised issues. The core issue found during development that with the absence of sound completely, users could simply progress in silence, akin to a study, or play music whilst taking part. Music was played throughout all the solo testing stages of the application during development, and as such the absence of sound was not revealed as an issue until the final weeks of the timeline. The testing implementations of looping ambient sounds firstly caused build issues, and once fixed, ended up highlighting the complete lack of any other sounds present in the game. When testing, it was extremely confusing hearing ambient sounds of a warehouse, but not hearing sounds made by objects when picked up, dropped, thrown, or hit with force against another object. The most basic implementation of sound only served to further highlight the complete lack of sound elsewhere in the application, and with the amount of sound to be added, was not something that could feasibly be done in the development timeline set out.

In addition to sound, several ideas initially laid out surrounding narrative integration in the game to attempt to draw a further sense of engagement were left out, solely due to having no experience with narrative construction and using games as a storytelling medium. This is another factor left out that if done right, could probably serve to bypass the simplistic art styles and potentially give the user a better sense of immersion in the virtual environment.

Between the difficulties presented with sound implementation and lack of experience with narrative, it revealed that there are far more factors that can affect a user's immersion, and as mentioned in the results evaluation, this should have been focus tested to draw specific areas that needed to be focused on to determine what factors being added or removed would contribute to the sense of immersion. For example, devising a study that provides realistic environments and physical interactions without narrative, and simplistic interactions and compelling narrative could be one such additional method that could help to determine immersive factors for VR.

In terms of developments to be made to the application in its current state, additions of quality-of-life features such as pause functions and visual fidelity options would have been good to implement, but not essential to the core idea of investigating immersion. The puzzle designs themselves are very basic and could benefit from an overhaul pass either using different mechanics or better scripting to make them feel more dynamic and challenging. Unfortunately, their implementations are a direct result of inexperience with the engine as well as designing and implementing puzzles in a game.

## Section 7 – Conclusions

Reflecting on the project regarding its initial aims, which were the creation of a short puzzle-based experience, and the intention of surveying users regarding their immersion, the objective has been met in the sense that an application was created from scratch and was able to implement a questionnaire to have anonymous users participate in the study. However, the method devised and implemented into the application was clearly indicated to be ineffective at drawing out determining factors for user immersion from the results, and in that regard has failed the aspect of the initial aim that the experience itself was not able to directly reveal immersive factors for users in VR.

Although the intended method failed to provide the anticipated data, the results of the questionnaire did draw out indicators that the modern headsets available are in fact able to provide a very strong sense of presence to a user, almost completely irrespective to how the user engages with the virtual environment presented to them.

The development process for the application was able to be undertaken within the project timeline to an acceptable standard and implemented some of the originally devised concepts without much alteration. In places, it falls short at being able to immerse users to a degree, meaning the approach to determining immersion by simply changing interaction methods was inconclusive.

When considering aspects that were not fully realized, it raises that those aspects themselves could be immersive factors themselves worth investigating and trying to determine which is most crucial is something that could be further expanded upon in the future. The core aim of this project revealed itself to be capable of having multiple avenues to explore, and in time I hope that myself or others can further investigate and properly determine which factors more heavily influence a user's immersion.

Personally, I believe that there is space for compelling narrative and puzzle-based experiences in VR, as the only reference needed to see success of this combination in traditional gaming is the Portal series. Currently there are no spiritual successors to the series available in VR that have managed to captivate people in the same way as the originals, even with the much smaller subset of users with access to VR. By setting out to build an application inspired by the series in this project, I have experienced first-hand how difficult it can be to familiarise yourself with a new medium of development with a target audience, and how crucial it is to consider all aspects of the design of an application. I hope to use this project as valuable learning experience when considering other projects to undertake in VR.

## Section 8 – Reflection on Learning

Overall, the development of the project has been an enjoyable learning experience and has presented the opportunity to evaluate and engage with some of the available tools for the creation of a VR application from start to finish.

Over the course of this project, it has become apparent that pursuing an emerging technology that is of personal interest has the potential to obfuscate the difficulties that are

present in the overall practice of game design and development, as well as the additional complexities introduced by bringing it to the medium of VR. With a deep personal interest as a consumer, with considerable experience since investing in VR in 2017, it has become apparent that my thoughts on the ease of being able to create an engaging and well-designed VR game was vastly different from reality. Including the idea of adding in a questionnaire study, it becomes clear that this project set out with lofty goals that were a struggle to attempt to achieve to their fullest extent.

I am satisfied that I have achieved sections of the aims described, as well as my personal drive to complete them as best as possible. Over the course of the project, I have had to acknowledge that taking a concept and developing it to its fullest potential is difficult to achieve without prior experience as a VR-focused developer.

The learning experience from developing alternative approaches for ideas I originally envisioned, and subsequently adapting to my limits. It has helped me understand that trying to push and develop ideas beyond my means as opposed to utilising straightforward implementations is not a constructive or conducive way to get results. I have found this is an iterative learning experience, and this continued approach of learning my limits will further help me push myself and plan my projects accordingly in the future.

Undertaking this project has reinforced my understanding that manageable scope is one of the most crucial things to focus on when planning a project, and the goals laid out must first be the achievable ones that encompass all the necessary aspects of an application, as opposed to the goals that will consume time attempting to implement and therefore leading to other aspects being omitted, the prime example was the focus in the presented project's interaction mechanics and user representation leading to the exclusion of sounds in the final application.

With the benefit of hindsight when considering the scope of the project and the planned methods to undertake the study proposed, it could have been approached differently and shifting the design to broader factors of immersion. Instead of relying upon my experience as a designer, the design should have relied upon achievable factors that could be easily manipulated to assess a user's sense of immersion, like realism and stylism in environments, or sound design being present or not present for different participants. However, I feel like the unintentional presentation of factors affecting presence provided by the questionnaire's results do provide an interesting foundation to work from in the future, as well as serving as a learning experience in how to re-assess the original approach to better engage users.

Where practicable, I have met the aims of this project, and it has allowed me the opportunity to engross myself in this emerging technology. This project has allowed me to experiment with the medium in a focused manner that I likely would not have been able to do had this been undertaken as a personal project. Whilst the final application has flaws, the development process was something I thoroughly enjoyed exploring, and I hope to continue to be able to explore VR further across my professional development.

## Bibliography

Abrash, M., 2014. *Abrash Dev Days*. [Online]

Available at:

<http://media.steampowered.com/apps/abrashblog/Abrash%20Dev%20Days%202014.pdf>  
[Accessed 5th May 2021].

Adams, E., 2004. *Postmodernism and the Three Types of Immersion*. [Online]

Available at:

[http://designersnotebook.com/Columns/063\\_Postmodernism/063\\_postmodernism.htm](http://designersnotebook.com/Columns/063_Postmodernism/063_postmodernism.htm)  
[Accessed 5th May 2021].

AlexLemminG, 2020. *AlexLemminG/RigifyToUnity: Converts Rigify Rig to Humanoid Compatible*. [Online]

Available at: <https://github.com/AlexLemminG/Rigify-To-Unity/commits/master>  
[Accessed 3rd April 2021].

Andrew, V. W., 2021. *Building a URP Screen Fader for Unity XR*. [Online]

Available at: <https://www.youtube.com/watch?v=OGDOC4ACfSE>  
[Accessed 12th April 2021].

Facebook Technologies, LLC, n.a. *Oculus - Facebook*. [Online]

Available at: <https://developer.oculus.com/reference/unity/v27/class o v r screen fade/>  
[Accessed 12th April 2021].

igroup.org, n.a. *igroup presence questionnaire (IPQ) overview | igroup.org - project consortium*. [Online]

Available at: <http://www.igroup.org/pg/ipq/index.php>  
[Accessed 2nd April 2021].

Metre, D. R., 2005. *Immersion and Presence*, Marseille: CNRS & University of the Mediterranean.

n.a, 1990. *A New Continent of Ideas*. [Online]

Available at: <https://ntrs.nasa.gov/citations/20020086961>  
[Accessed 4th May 2021].

Norman, J., Unknown. *"Pygmalion's Spectacles"*. [Online]

Available at: <https://www.historyofinformation.com/detail.php?id=4083>  
[Accessed 4th May 2021].

Originals, A. S., 2020. *Snaps Prototype | Sci-Fi / Industrial | Unity Asset Store*. [Online]

Available at: <https://assetstore.unity.com/packages/3d/environments/sci-fi/snaps-prototype-sci-fi-industrial-136759>  
[Accessed 16th March 2021].

Reddit, 2009. *Virtual Reality*. [Online]

Available at: <https://www.reddit.com/r/virtualreality/>  
[Accessed 12th May 2021].



Reddit, 2019. *Valve Index*. [Online]  
Available at: <https://www.reddit.com/r/ValveIndex/>  
[Accessed 12th May 2021].

Reddit, A., 2020. *Pavlov Vr : SteamVR*. [Online]  
Available at: [https://www.reddit.com/r/SteamVR/comments/hdn7nb/pavlov\\_vr/](https://www.reddit.com/r/SteamVR/comments/hdn7nb/pavlov_vr/)  
[Accessed 5th May 2021].

Reddit, G., 2021. *Is pavlov the same titan of the vr shooters in 2021? : virtualreality*. [Online]  
Available at:  
[https://www.reddit.com/r/virtualreality/comments/mr9sqa/is\\_pavlov\\_the\\_same\\_titan\\_of\\_the\\_vr\\_shooters\\_in/](https://www.reddit.com/r/virtualreality/comments/mr9sqa/is_pavlov_the_same_titan_of_the_vr_shooters_in/)  
[Accessed 5th May 2021].

Reddit, L., 2021. *Unpopular opinion: I like Boneworks better than Half Life: Alyx : virtualreality*. [Online]  
Available at:  
[https://www.reddit.com/r/virtualreality/comments/kt9g9e/unpopular\\_opinion\\_i\\_like\\_bone\\_works\\_better\\_than/gil8mti/](https://www.reddit.com/r/virtualreality/comments/kt9g9e/unpopular_opinion_i_like_bone_works_better_than/gil8mti/)  
[Accessed 6th May 2021].

Reddit, M., 2021. *Half Life: Alyx killed a lot of VR games for me. : virtualreality*. [Online]  
Available at:  
[https://www.reddit.com/r/virtualreality/comments/mc6gg1/half\\_life\\_alyx\\_killed\\_a\\_lot\\_of\\_vr\\_games\\_for\\_me/](https://www.reddit.com/r/virtualreality/comments/mc6gg1/half_life_alyx_killed_a_lot_of_vr_games_for_me/)  
[Accessed 6th May 2021].

Reddit, R., 2021. *Boneworks worth it in 2021? : virtualreality*. [Online]  
Available at:  
[https://www.reddit.com/r/virtualreality/comments/m4sjot/boneworks\\_worth\\_it\\_in\\_2021/gr1hzyl/](https://www.reddit.com/r/virtualreality/comments/m4sjot/boneworks_worth_it_in_2021/gr1hzyl/)  
[Accessed 5th May 2021].

Reddit, S.-f.-s., 2020. *Boneworks of Half Life: Alyx? : virtualreality*. [Online]  
Available at:  
[https://www.reddit.com/r/virtualreality/comments/klrgac/boneworks\\_or\\_half\\_life\\_alyx/ghahp83/](https://www.reddit.com/r/virtualreality/comments/klrgac/boneworks_or_half_life_alyx/ghahp83/)  
[Accessed 6th May 2021].

Schwind, V., Knierim, P., Haas, N. & Henze, N., 2019. Using Presence Questionnaires in Virtual Reality. *CHI '19: Proceeding of the 2019 CHI Conference on Human Factors in Computing System*, Volume 360, pp. 1-12.

SkarredGhost, 2016. *The Difference Between Presence and Immersion - The Ghost Howls*. [Online]  
Available at: <https://skarredghost.com/2016/11/09/the-difference-between-presence-and-immersion/>  
[Accessed 4th May 2021].

Team, B. D., n.a. *The Blender 2.79 Manual - Introduction*. [Online]  
Available at: <https://docs.blender.org/manual/en/2.79/about/contribute/index.html>  
[Accessed 6th May 2021].

UtilityFunction, 2016. *VR Buttons and Levers | Physics | Unity Asset Store*. [Online]  
Available at: <https://assetstore.unity.com/packages/tools/physics/vr-buttons-and-levers-66520>  
[Accessed 29th March 2021].

Valve Corporation, n.a. *Class SteamVR\_Fade | SteamVR Unity Plugin*. [Online]  
Available at:  
[https://valvesoftware.github.io/steamvr\\_unity\\_plugin/api/Valve.VR.SteamVR\\_Fade.html](https://valvesoftware.github.io/steamvr_unity_plugin/api/Valve.VR.SteamVR_Fade.html)  
[Accessed 12th April 2021].

Weibel, C., 2016. *Free Shipping Containers | 3D Industrial | Unity Asset Store*. [Online]  
Available at: <https://assetstore.unity.com/packages/3d/environments/industrial/free-shipping-containers-18315>  
[Accessed 12th March 2021].