Plymouth University

School of Computing, Electronics and Mathematics

PRCO304

Final Stage Computing Project

2016/2017

BSc (Hons) <*Computer Science*>

<*Jonathan Holmes*>

<*10426286*>

<*A Virtual Reality game made in Unity3D for the HTC Vive*>

Supervisor: <Nigel Barlow>

Second marker: <Pushpa Subramaniam>

Contents

[Abstract 2](#_Toc482003484)

[Introduction 5](#_Toc482003485)

[Background and Project Objectives 5](#_Toc482003486)

[1.1 Initial project idea 5](#_Toc482003487)

[1.2 Choosing a platform 6](#_Toc482003488)

[2. Personal Objectives 7](#_Toc482003489)

[3. Deliverables 7](#_Toc482003490)

[4. Project Management / Method of approach 7](#_Toc482003491)

[4.1 Unity 7](#_Toc482003492)

[4.2 C# 8](#_Toc482003493)

[4.3 Blender 9](#_Toc482003494)

[4.4 Otomata 9](#_Toc482003495)

[4.5 Version control 10](#_Toc482003496)

[4.6 Development Methodologies 10](#_Toc482003497)

[4.7 User feedback 11](#_Toc482003498)

[4.8 User Stories 12](#_Toc482003499)

[5. Legal, social, ethical, and professional issues 12](#_Toc482003500)

[5.1 Legal 12](#_Toc482003501)

[5.2 Professional 13](#_Toc482003502)

[5.3 Ethical 13](#_Toc482003503)

[6. Design 13](#_Toc482003504)

[7. Implementation 14](#_Toc482003505)

[7.1 Modelling and animation 14](#_Toc482003506)

[7.2 Audio 15](#_Toc482003507)

[7.3 UI 16](#_Toc482003508)

[7.4 AI 16](#_Toc482003509)

[7.5 Unity 16](#_Toc482003510)

[8. Stage Breakdown 17](#_Toc482003511)

[8.1 Stage 1 17](#_Toc482003512)

[8.2 Stage 2 17](#_Toc482003513)

[8.3 Stage 3 18](#_Toc482003514)

[8.4 Stage 4 19](#_Toc482003515)

[9. Project post-mortem 20](#_Toc482003516)

[9.1 Personal Objectives 20](#_Toc482003517)

[9.2 Deliverables 21](#_Toc482003518)

[10. Method of Approach 21](#_Toc482003519)

[10.1 Methodology 21](#_Toc482003520)

[10.2 Technologies / Implementation 21](#_Toc482003521)

[10.3 Initial planning (PID, user stories, designs) 21](#_Toc482003522)

[10.4 Developer performance 22](#_Toc482003523)

[11. Conclusions 22](#_Toc482003524)

[12. Statement of word count 22](#_Toc482003525)

[13. Reference List 22](#_Toc482003526)

[14. Appendices 23](#_Toc482003527)

[14.1 User Guide 23](#_Toc482003528)

[14.2 Project Management Artefacts 23](#_Toc482003529)

[14.3 Other materials (UMLs, designs, test results, sounds clip URLs) 23](#_Toc482003530)

## 

## Abstract

Virtual reality has been developing for many years. However, it has seen dramatic growth particularly in the last decade. Platforms such as the HTC Vive allow for a high framerate and FOV experience playing even the latest intensive games.

This report outlines a game development project using Unity3D to produce a VR game made for the HTC Vive. It outlines the project idea as a single player puzzle game and justifies why the HTC Vive was chosen over other devices. It explores the developer’s desire to improve his skills in Unity3D, Blender and C# and familiarise himself with VR technology. The legal, social, and ethical issues are explored, some of which are unique to the field of virtual reality. The design and implementation stages are considered, showing how a drawn design finds its way into the finished project. The project is then broken down into its development stages giving a chronological viewpoint of the whole project.

A final project post-mortem is then carried out, where the above stages are subjected to critical review by weighing up their pros, cons, and viable alternatives.

Details of interim project management artefacts can be found in the document appendices.

## Introduction

The purpose of this project was to create a Virtual Reality game made using Unity3D, intended for use with the HTC Vive HMD (Head mounted display).

Although there was never a definitive intended client, there are many potential ones. Small innovative indie games can often end up on digital distribution platforms such as Steam after being noticed by online communities such as Steam Greenlight or itch.io. In this case the intended client is any consumer interested in pc gaming, particularly those who are interested in small-scale, innovative games.

## Background and Project Objectives

### Initial project idea

The initial plan for the project was a single-player VR puzzle game. After exploring the best potential puzzle-like applications for VR, a plan was drawn up for a “Factory Worker Simulation game” . The player would stand in front of a conveyor belt with different items moving past them on the conveyor. The player would have to do something with the items e.g. sort them, throw specific ones away etc. However, after about a week this idea was scrapped. (An image of the prototype can be found in the appendices). The reason for this was because the idea could not easily be expanded on to produce a full-bodied game. This was not a wasted experience though, as it highlighted that more planning was necessary to ensure that an idea be fully fleshed out before any development starts.

More planning let to a new idea – a game consisting of a set of “mini-games”. Inspiration was taken from Valve’s “The Lab”, described as a “compilation… of room-scale VR experiments” ("The Lab (Video Game)"). With high-end VR being (relatively speaking) in its infancy, these sorts of games give the best introductory experience to virtual-reality due to their varied nature and intuitive mechanics. The idea was that the player would start in a hub room and could travel to whichever mini-game he chose to play/experience, and then travel back to the hub room to select a new game.

To solidify the project’s objectives a Project Initiation Document (PID) was drafted (available in the appendices) which gave an overview of the project details. The aims for the project were specified in terms of its scope. This was split into things that should definitely be achieved, things likely to be achieved, and things that may be achieved if there is enough time. The PID proposed a “puzzle based game”, with the ability to “pick up and interact with objects”. It also specified at least “20” short puzzle based levels, 15 3D modelled assets and at least 10 hand recorded sounds. In hindsight, it appears that most of these aims were met, whilst some that ended up becoming no longer entirely relevant were re-adjusted.

### Choosing a platform

One of the big initial steps in planning the project was to choose what platform it would be built for. I had to decide between a mobile or non-mobile platform. In the last decade there has been a “rise of smartphones with high-density displays and 3D graphics capabilities” ("History Of Virtual Reality - Virtual Reality"). This makes them extremely practical as virtual reality devices. Google have already taken advantage of this by launching the ‘Google Cardboard’, bringing “immersive experiences to everyone in a simple and affordable way” ("Get Cardboard – Google VR"). Samsung followed suit soon after by releasing the ‘Gear VR’ ("Samsung Gear VR With Controller"). Both these products are simply devices to hold your phone comfortably in front of your eyes. However, they simply do not have the processing capabilities to provide a smooth experience in most cases.



Figure - Google's Cardboard headset and Samsungs Gear VR

In addition to the surge in cheap, mobile VR devices and applications, there has also been huge growth in VR head-mounted displays (HMDs). Three main competitors corner this market at present: Sony with the PlayStation VR, Oculus (now owned by Facebook) with the Oculus Rift, and HTC with the HTC Vive. These devices offer unparalleled performance for VR experiences due mainly to the quality of the displays. For example, both the Rift and Vive offer “two OLED panels boasting a combined 2,160x1200.” This means that “each eye gets its own 1080 x 1200 display” (Swider). As the goal of VR is to convince the brain that you really are in the virtual space being simulated, it is essential to have displays that offer a resolution as close to that of the human eye as possible. At present this is the main drawback of mobile VR; it simply does not offer a good enough resolution to make the experience completely convincing or immersive. After conducting this research, I settled on a non-mobile HMD as my target platform – specifically the HTC Vive because at the time only the HTC Vive had support for Unity3D, via an asset store plug-in.

## 2. Personal Objectives

The personal objectives whilst undertaking this project were to develop skills relating to Game development. These include but are not limited to Game Design, asset creation/3D modelling and proficiency in specific software such as Unity3D and Blender. The developer also aimed to improve his knowledge and skills in developing for VR. In addition, carrying out a project of this scale as a solo developer from start to finish provided invaluable experience in development methodologies, effective versioning using GIT, bug tracking and time management.

A further objective was to gain experience in marketing a game, through producing a trailer to listing it on a seller’s platform such as Steam or itch.io.

## 3. Deliverables

The deliverables were a series of mini-games built in Unity3D for the HTC Vive. The games should offer a good diversity of experience in terms of VR interaction. They should also be intuitive and appropriate for all ages.

The main deliverable for this project was a Unity executable. However, I also gathered requirements information, produced a Gantt Chart to plan my work, took screenshots and video footage throughout development and drew designs at the start of the process. Together these provide a clear story of how the project progressed.

## 4. Project Management / Method of approach

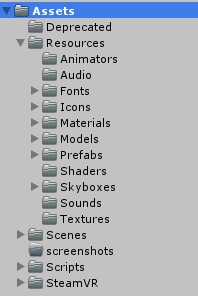
During this project, several software tools were utilised. The main three were Unity (for game development), Blender (for asset creation/modelling), and Git (for versioning).

A good, well-thought out approach can save a huge amount of time during a large project. Unity, Blender and Git all have ways to speed up and optimise the development process, especially when iterating between all three. For example, the developer set up Blender to save all assets straight into the Unity asset directory (under a subdirectory named “models”). This meant that when a blender file was saved, it would automatically update the Unity asset, keeping everything in-sync.

### 4.1 Unity

In terms of Software, Unity was chosen for the game development. This was down to several reasons. Firstly, the developer already had experience creating other, smaller, games with it. Secondly, at the time of the start of the project, the only other viable alternative (Unreal Engine), did not yet have any VR support.

Unity was for the most part a pleasure to work with. It provides organisational and structural tools which ensure as the project grows the structure does not become overwhelming or confusing, provided the effort is put in to keep things organised. During the project it was ensured that all assets were organised into a hierarchical format to keep things streamlined and easy to navigate.



Another benefit of Unity is access to the asset store, where you can download third party assets. As the developer wanted to keep the project as independent as possible he refrained from using this, but it did provide the SteamVR Unity plug-in and the Skyboxes that were used in the project.

The SteamVR plug-in for Unity was a compulsory asset if the project was to be targeted at the HTC Vive. The plug-in allows developers to “target a single interface that will work with all major virtual reality headsets from seated to room scale experiences”. It also provided access to the tracked controllers and a useful 2D debug mode so that the developer could move around the project scene using the keyboard if needed instead of having to put on the headset which ended up saving a lot of time.

Unity also makes it very easy to deploy to multiple platforms, which means that if the intended platform changed at any point in the development process it would make this transition much more manageable than other development tools.

Within Unity I set my default external scripting tool as MonoDevelop. MonoDevelop is a lightweight IDE used mainly for scripting. It includes all the essential features such as automatic code completion, source control and a GUI. However, the developer had problems with MonoDevelop crashing halfway through development and eventually switched to Visual Studio, which soon proved to be a more robust and stable IDE. However it did suffer from longer loading times.

### 4.2 C#

Unity has only ever supported three languages – UnityScript (AKA JavaScript for Unity), Boo, and C#. In 2014 only 0.44% of Unity developers used Boo, whilst 80.4% used C#. Since the Unity5.0 release Unity dropped support for Boo documentation to focus their resources in a more constructive way.

Due to user feedback they received in 2014, since then they have been aiming to provide in their documentation “C# examples across the board” whilst moving internally to provide the “best support for C#” ("Documentation, Unity Scripting Languages And You – Unity Blog") that they can. From this research it was decided that C# was the best supported and most popular language to use in Unity 5.0, and should be the language of choice for this project.

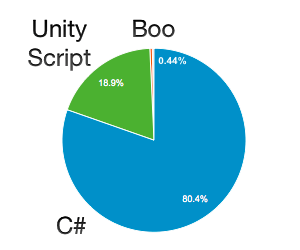


Figure - Unity's official blog showing the proportion of each language used in 2014

C# played its part to add ease to development. It is type safe unlike C or C++ which reduces bugs and crashes. It also provides access to the .NET framework class libraries, which are extensive.

### 4.3 Blender

Blender is a free and open source 3D creation suite. It has a huge range of capabilities, but during the project it was used primarily for modelling and asset creation. Although Blender can be intimidating or seem chaotic to some developers, when focusing on a particular set of features the workflow becomes quicker and easier over time. It provided the developer with a diverse set of tools to model, rig and animate as he needed.

### 4.4 Otomata

‘Otomata’ was used to create music for the game. It is a “Generative Musical Sequencer”. It “employs a cellular automaton type logic… to produce sound events” ("Otomata - Generative Musical Sequencer"). Otomata was used to randomly generate sound patterns which were then used as music tracks for the game levels. It proved an intuitive and easy-to-use tool which produced an unlimited number of note sequences. As the synthesiser used for each note is the same it also provided a consistent sound across all the levels of the game. The licensing rights and requirements for this tool are discussed in the ‘Legal, social, ethical and professional issues’ section of this report.

### 4.5 Version control

Git was used throughout the project as the version control software of choice. Instead of using the standalone GitHub software, GitBash was used to track, commit and push files from the local to the remote repository.

A commit was made almost daily during work periods, however when the project work required balancing with other commitments this number dropped to once every few days. This frequent backup ensured that if necessary the project could be rolled back to a previous commit, reducing risk of data loss through hardware failure or other means. Roll-backs were done on multiple occasions and potentially saved the project days of work. For example, when updating the version of the SteamVR plug-in for Unity, some bugs emerged, probably because the update was new and not completely stable. Even after rolling-back the plug-in version the bugs remained. It seemed the only solution was to revert back to a previous commit and wait for the bugs to be fixed before attempting the update again. Sure enough, after updating a week or so later no bugs emerged.

A few problems arose with Git throughout the project duration, which were documented in highlight reviews. One problem occurred because GitHub would not allow free users to upload any file over 100MB to a repository. Some of the project’s Blender (.blend) files were above the limit. This problem was solved in two ways. Firstly, by compressing the Blender files by default to reduce their size, and secondly by adding the Unity project metadata folder to the repository’s ‘gitignore’ list. This meant this folder was not uploaded to the repository. This did not reduce the integrity of the backup as this meta-data is generated every time the project is launched in Unity and so can be excluded from versioning.

### 4.6 Development Methodologies

As software development processes work much better with a team rather than an individual developer, a slightly laxer approach was adopted during the project. A toned-down version of XP (‘Extreme Programming’) was followed. The PID laid down a group of processes that should be followed, but soon after project start only the most effective ones were followed strictly. The XP processes followed were –

1. Keep a spreadsheet of required features, prioritized.
2. Define specific engineering tasks to get done (This ended up being done on paper as there were many small tasks that could be dealt with quickly) A notepad was always kept at hand.
3. Time-box each session. (This was done for most (but not all) sessions). It did become clear that on average more work was done during time-boxed sessions, and it made it easier to quantify exactly how much had been done in a specific amount of time.
4. Utilise effective versioning (Using git as discussed).
5. Frequently reprioritize Gantt spreadsheet appropriately.

The most helpful of these processes was keeping a Gantt chart to organise the project over time. The chart made it easy to reprioritise tasks and their expected work days of effort, and shift deadlines along if priorities changed. It also provided a way to know if the project was broadly on track at any point. This allowed the developer to spend more time on extra features for certain areas of the game if he was ahead of schedule or vice versa if he was behind schedule. The chart is split into rows denoting each task, sub-task, their start and end date and the number of work days predicted to complete them. This also encouraged the develop to think about each task before he undertook it, ensuring he started the task with a good approximation of how much work it entailed.

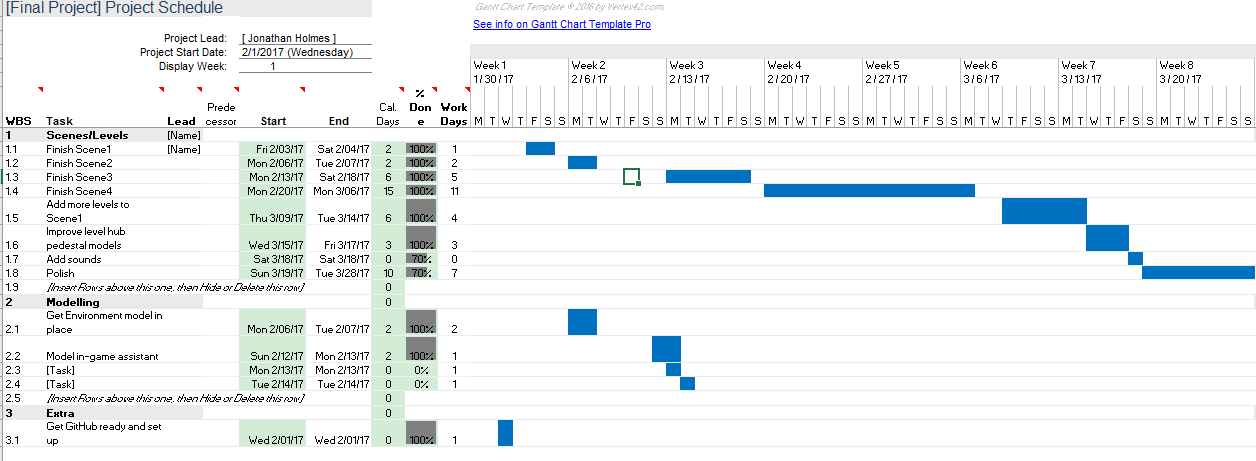


Figure - Gantt chart used to organise the project over time

### 4.7 User feedback

Feedback was obtained using two main methods. Firstly, throughout the project the game was tested by 2-3 different people and feedback was received. A greater breadth of feedback could have been obtained by testing with even more people, preferably those of different ages. However there was no feasible way to carry this out. Many improvements were made to the project based on this user feedback. For example, one user became confused when asked to press the menu button on the Vive controller. He made it clear that having too many controls to remember can become confusing and frustrating in VR as you cannot see your hands directly with the headset on. Instead he suggested the user interact e.g. grab an object in the game level to return to the hub (the same method used to get from the level hub to a game). This way the user never has to fumble with the controller and intuitively knows what to do. This was later implemented and made level transition much more intuitive especially for first-time / newer VR users.

The second, more diverse way of obtaining user feedback was through the internet forum 'Reddit'. The developer made a submission to the HTC Vive subreddit showing a clip of some gameplay and a request for any thoughts / feedback. A good amount of feedback was obtained from a total of 24 comments.

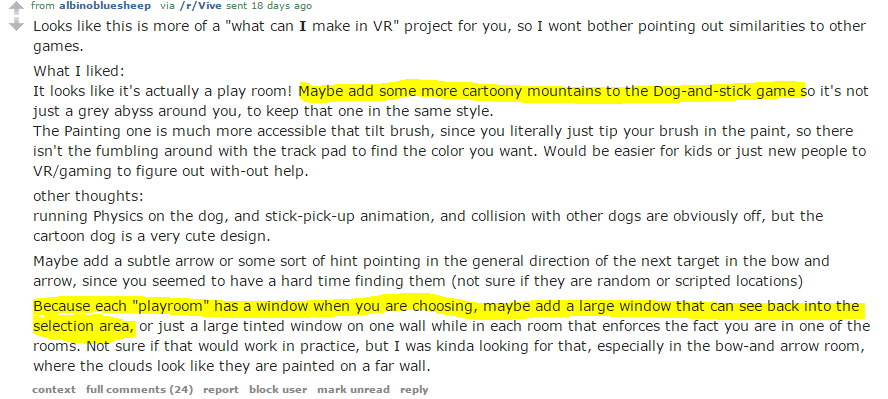
****

Figure - A reply to the developer's post on the HTC Vive sub-reddit

The above figure shows an example feedback comment, in which the user says what they liked and didn’t like about the game, and suggests some improvements which have been highlighted in yellow. The first suggestion is to add ‘cartoony mountains to the Dog-and-stick game’. This was a valid suggestion as at the time the environment just faded into fog. The next suggestion was to be able to see back into the ‘selection area’ or level hub from each game area. This was another good suggestion as it provided some consistency to the game and at the same time made it seem more like a series of play rooms. Both suggestions were implemented in the next sprint session.

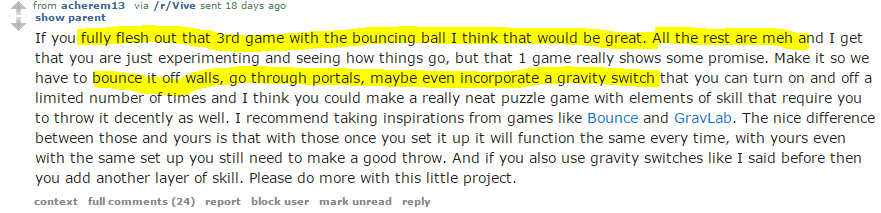


Figure - A reply to the developer's post on the HTC Vive sub-reddit

Another comment suggested that whilst one of the 4 mini-games showed promise, the others were lacking. The writer also recommended fleshing out further the ‘ball-in-pipe’ game to make it more interesting. Due to time constraints, this feedback was not implemented. However, these features could well be implemented in future as I plan to continue developing the project after the deadline and so they still have good value.

This method allowed the developer to gain feedback on a scale not previously thought possible for the project. It provided feedback from others who also owned VR equipment and played VR games, which at present is still a relatively small portion of the gaming community.

### User Stories

As part of the project’s agile approach, user stories were drawn up summarising the desired gameplay functionality. To draw these up the developer played VR games already on the market and took note of what he deemed essential for a comfortable, immersive VR experience. These mainly encourage keeping the user comfortable and providing fun VR object interaction. These user stories also offer good basic criteria to conclude whether the final product met it’s aims.

## Legal, social, ethical, and professional issues

### 5.1 Legal

The main 3rd party asset used (the SteamVR sdk / plug-in ) was free to use for commercial use.

All music was generated using Otomata, the generative music sequencer, is open for commercial use, however the creator states that he would appreciate some attribution, but it is not compulsory.

All sounds that were downloaded from ‘freesound.org’ were under the Creative Commons 0 license, which means any user can “copy, modify, distribute and perform the work, even for commercial purposes, all without asking permission”("Creative Commons — CC0 1.0 Universal").

For the trailer the developer produced to showcase the game, an audio track was used from ‘Bensound.com’ ("Royalty Free Music By Bensound | Creative Commons Music"), and appropriate attribution was given.

### 5.2 Professional

**Coding standards eg. DRY coding**

A matter of high importance during this project was to keep the code neat, readable and consistent. This was essential when such a large number of scripts were involved. Principles such as DRY (Don’t Repeat Yourself) were enforced from the start. It stayed an important goal that every script and indeed every function had ‘a single, unambiguous, authoritative representation within a system’ ("Don't Repeat Yourself"). This meant that even after weeks or months of not working on an area of code, it was easy to come back and continue working on it without becoming confused. In Unity3D’s object based system, this was achieved through attaching multiple scripts to one object, each with its own clear behaviour. E.g. for the game mascot (a cube character who flies around exploring the levels), one script attached handles the movement whilst another script is attached to the thrusters whose sole purpose is to keep the thrusters pointing towards the ground. This was in an effort to favour composition over inheritance. Too much inheritance can lead to long hierarchies which can pose problems when moving classes around. Composition avoids this problem. This modular approach also keeps the scripts organised and intuitive based on their name and the object they are attached to.

It was also essential to stick to consistent coding conventions in terms of line-spacing and commenting. The developer adhered to the Unify Community C# Coding Guidelines ("Csharp Coding Guidelines - Unify Community Wiki"), which specified correct indentation for loops, correct use of bracing, comment style and spacing. For example, the document specifies that comment tags should be placed “wherever possible… above the code instead of beside it”, and that “braces should never be considered optional”. Having solid conventions such as these saves time not only when writing the code but also when the developer of other reads it back and attempts to understand it.

### Ethical

There are many ethical considerations for video games in general, and even more so with Virtual Reality. This is mainly due to increased realism. The gaming industry is providing ‘more realistic and believable looking animation’ using motion capture. In addition, highly ‘immersive virtual environments have been further developed’ ("Ethics Of Virtual Reality Applications In Computer Game Production"). Placing a person in such a virtual environment with such a high level of presence or immersion could, if not in the short-term then in the long-term, have unseen repercussions.

The first is the problem of desensitisation. In the past, it has been claimed that violent non-VR games could be subliminally encouraging users to replicate that behaviour in the real world, as they ‘actively seek out this type of scenario for the adrenaline rush and sense of power’ that they get from playing the game. This proved relatively uncommon, but it could become more likely with VR games due to their increased immersion.

Addiction could also prove to be a higher risk with VR games that non-VR. Some gamers play games to escape real world worries and problems, and this is only going to become more alluring with increased immersion. Many could spend ‘increasing amounts of time in the virtual environment which would have a detrimental effect on their real world life’("Virtual Reality And Ethical Issues - Virtual Reality").

Virtual criminality was never taken truly seriously in non-VR games, but may find itself a prevalent issue in VR gaming. Virtual criminality occurs when ‘several people are immersed within a virtual environment but one of these participants becomes injured or traumatised due to the actions of another’. This raises many questions. Is it possible for someone to suffer such an injury in a virtual environment? Would the perpetrator be punished ‘in a similar way to someone who commits this action in the real world?’ ("Virtual Reality And Ethical Issues - Virtual Reality"). This is an area which will likely see much discussion in the near future as VR grows.

Due to the high levels of immersion, many users report suffering motion sickness which using VR equipment. This has even been dubbed ‘Virtual reality sickness’. This can include ‘headaches, stomach awareness, nausea, vomiting, fatigue, sweating’ and more **(**"Virtual Reality Sickness"). VR sickness is different to motion sickness as it only requires the ‘visually-induced perception of self-motion; real self-motion is not needed’. VR sickness has greater consequences than this however, as it could discourage a significant portion of the user base to stop using VR. It could also be a barrier to effective use of training tools and rehabilitation tools.

As the game that was made during this project was meant for any ages, there was no violence included. This should limit or even stop any desensitisation while playing the game. In addition, due to the small -scale nature and structure of the game (a disconnected set of mini-games), there is a very low chance of addiction.

Many games collect data from their players. This data could be the device information that the player is using, player habits, behaviour trends or bug/crash reports. Collecting this data gives the collector an ethical responsibility. They should ‘make participants aware’ that the data is being collected and allow the user to give their consent. They should also unless stated otherwise ‘maintain confidentiality’ of the data. The game that was made during this project does not ask for the user to sign up with credentials and there is no user data collected, so these considerations do not need to actively be considered.

As motion sickness is caused by the current technology in the HTC Vive, the game that was made during this project is just as susceptible to the problem as any other VR game. However, to limit the effect of this a 1:1 motion system was ensured. This means the user cannot teleport or jump around the levels, they can only move in the virtual space the same amount they move in the real world. This, of course, limits the user in terms of how far they can move in the virtual space. However, the game is set in a series of small rooms, so the user doesn’t really need to move out of these.

## Design

**Level design**

During the project, sketches and designs were continuously drawn up, along with the desired basic functionality of the game. This allowed initial ideas to develop and change early on, ensuring a solid idea before any modelling or programming began. From here an iterative approach was adopted to achieve the desired level design in Unity.

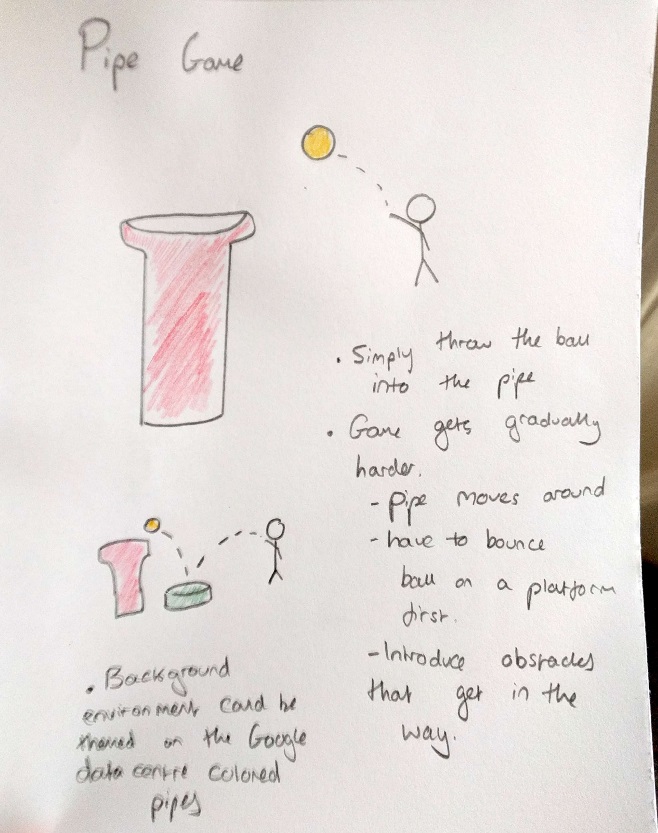


Figure - Level design sketch made for the 'Ball-In-Pipe' mini-game

Once the idea for the game had been solidified, the developer would then experiment in Blender to scope out how the assets could best be modelled. For the ‘Ball-In-Pipe’ mini game, for example, it was planned that the player would be able to see a series of interconnected pipes outside the play area, to give the feel of some sort of factory. In this example, instead of drawing up a sketch, a reference image was used. The developer wanted it to look like Google’s famous datacentres and so used a relevant reference image to model against.

Once the required assets had been modelled, the level design process was then taken to Unity3d. As the blender files were always saved to the Unity assets folder, this meant that they could be iteratively edited in Blender and any changes would be automatically updated in Unity.

## Implementation

### 7.1 Modelling and animation

The process of producing the models needed from the initial designs was a gradual one. First a basic pipe model was created. This was then expanded on to produce more complex combinations of pipes. These were then duplicated and rotated to produce a seemingly random array of pipes.

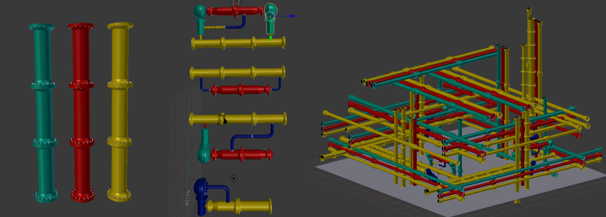


Figure - The process of developing a model in Blender

This approach combined initial planning with experimentation until a result was achieved that fitted what the developer envisioned.

Blender’s armature system was utilised to allow for easy animation. A model can be fitted with ‘bones’ which then join to a particular part of the mesh. When the bone is moved, the mesh will move. Once a model was rigged with an armature, it was animated using Blender’s dope sheet window via keyframes.

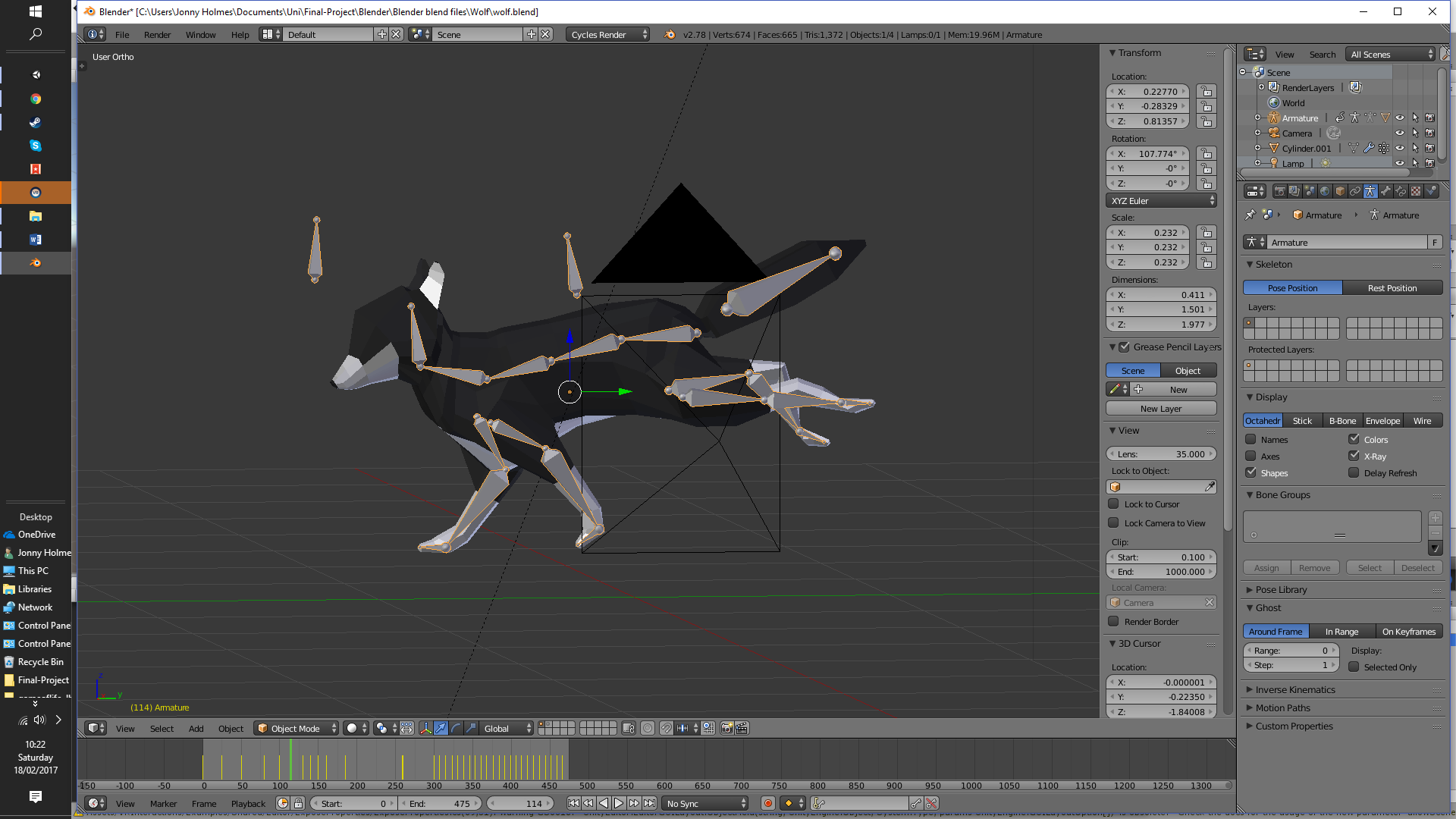


Figure - Model rigged with armature, showing keyframes

The area of a mesh that a bone influences, along with the strength of the influence, can be changed in Blender’s weight paint mode. This enabled the developer to create various animations for a model with limited effort. These animations would then be automatically imported into Unity along with the model. The animations can then be used in Unity using the Animator window.

### 7.2 Audio

The required audio for the game was acquired using three methods. Some audio was created by the developer using a microphone and Foley methods, some was acquired from a website called ‘freesounds.org’ and some was created using ‘Otomata’, the generative music sequencer. Once the audio was ready it was edited using Audacity. The purpose of the editing ranged from reducing/removing background noise, cropping unneeded sections or looping the audio. Once the audio has been edited it was then stored within the Unity assets directory under an ‘Audio’ subfolder. From there it was then used as needed by Unity’s ‘Audio source’ components, which can be attached to Unity game objects to play music or sounds. The Audio Source components allowed the developer to easily set up and manage spatial audio in all the scenes, which is very important for immersion in VR.

### 7.3 UI

For VR, UI is a controversial issue. Some developers will overlay UI elements directly onto the user’s camera view. However, this reduces presence and immersion and so most VR games adopt a system where the user interacts with virtual objects in the game world to replace this. The ‘menu’ system in PlaySpace VR simply consists of grabbing spheres with the trigger button on the controller. This makes for a much smoother experience than having to look at UI elements on the camera view or with a separate menu button on the controller.

### 7.4 AI

Several agents within the game have AI driven behaviour. The wolves featured in the ‘fetch game’, for example, utilise Unity’s navigation mesh system. This system allows the developer to specify game objects that are ‘navigation static’. This are objects which will never move and will form the baked navigation area for any agents that want to traverse it. Some wolves in the game will randomly wander by choosing random points on the navigation mesh and calculating a path towards it. The wolf which fetches the stick thrown by the player will calculate a path first to the stick, then back to the player, avoiding any obstacles in its way.

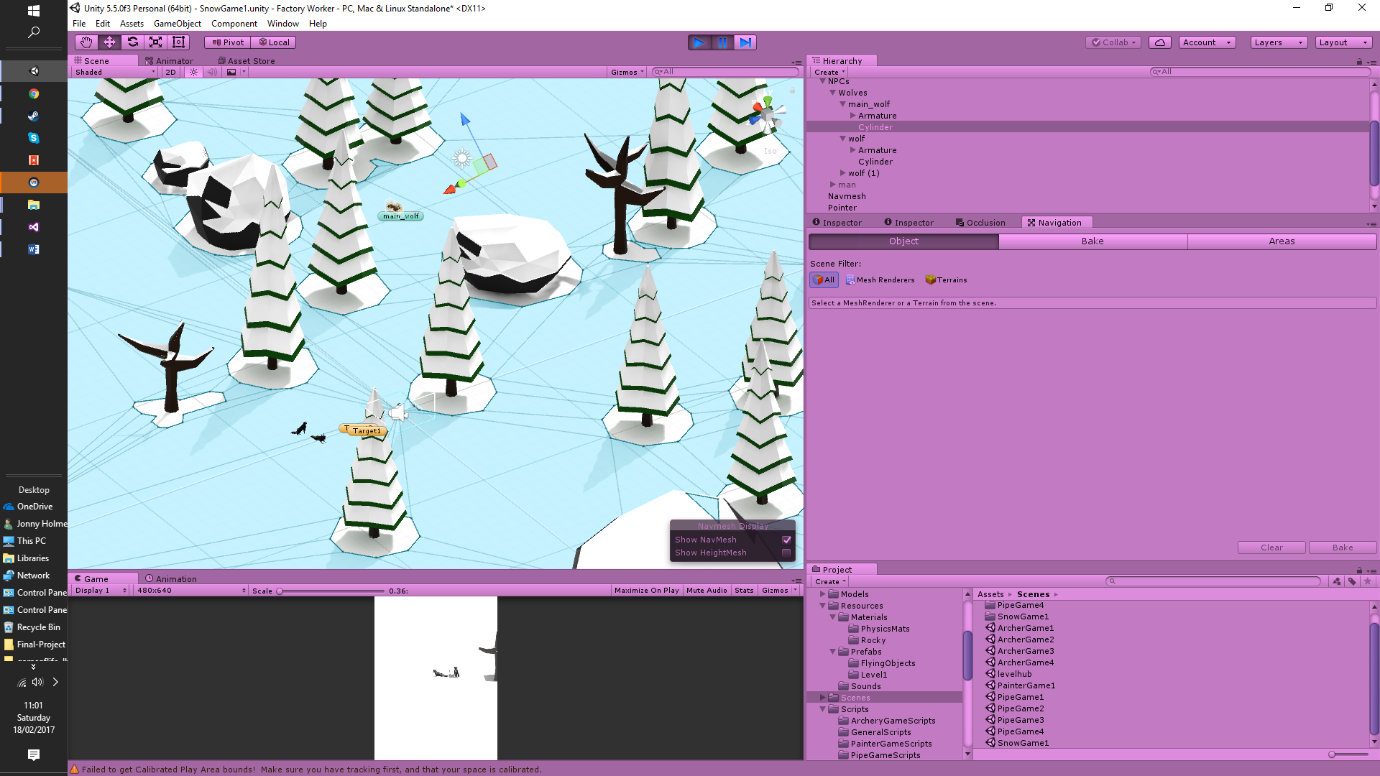


Figure - Navigation mesh (in blue) in Unity

Another example of AI is shown by the cats that float around in the ‘painter game’. They follow an artificial flocking algorithm to move around. The algorithm was originally defined by Craig Reynolds and was known as the ‘Boids’ algorithm ("Boids (Flocks, Herds, And Schools: A Distributed Behavioural model)"). Each cat follows three rules: alignment, separation and cohesion. Alignment steers the heading of each Boid to the average heading of neighbouring Boids. Separation steers each Boid away from any other Boid that gets to close. Cohesion steers each Boid to the average position of neighbouring boids. Together they attempt to accurately emulate flocking behaviours of animals.

### 7.5 Unity

Every Unity script inherits from the ‘MonoBehaviour’ class (Technologies). This class provides the functionality to start co-routines (effectively threads), which is a very useful feature. ‘MonoBehaviour’ itself inherits from the ‘Behaviour’ class, which provides the methods to enable and disable the script. Behaviour inherits from Component, which provides the base class for everything attached to ‘gameObjects’. This means not only user written scripts but also components provided by Unity. At the top is the Object class, which acts as the base class for all objects in Unity. It offers instantiation and destruction functions. Every script you create in Unity will by default inherit from ‘MonoBehaviour’.

Creating and using
Object
Component
Behaviour
(enabled)
MonoBehaviour
(coroutines)
Script
 

Figure - Unity's Class hierarchy (Studio)

## Stage Breakdown

### 8.1 Stage 1

Most of Stage 1 was spent solidifying the high-level structure of the scenes. It was important to develop a stable prototype as soon as possible, as this would make it clear early-on if any part was not feasible or needed to be changed. The focus of this stage was on modelling the large environment scenes that were to be used in the various mini-games.

The wolf asset for the snow game was initially modelled, rigged and animated and a basic prototype of the scene was implemented where the player could play fetch with the wolf. However, at this stage the wolf would not look at the player and the model and texture was basic and rugged.

Basic implementations of the archery and painting scenes were also implemented at this stage, helping to further solidify the structure of the game.

The task of designing and implementing an in-game assistant was considered, as this was included in the PID scope but had not yet been looked at. Designs were drawn up but nothing was implemented at this stage due to time constraints.

The version control that was to be used throughout the project was set up, and the Excel Gantt chart to be used as a plan was populated as far ahead as was feasible.

### 8.2 Stage 2

Stage 2 was spent adding to the prototype created in Stage 1 to create a more substantial game. A level hub was constructed for the user to traverse the levels, and each mini-game was substantially fleshed out.

From this stage onwards these sections will be split based on the respective mini-games that were worked on during the stage.

#### 8.2.1 Painter Game

A brush and an eraser were modelled and added to the painter scene. The logic of the line renderer code for the painter scene was implemented, allowing the user to draw lines with the controller. More wolves were added to the snow game which would roam around, making the world feel more alive.

#### 8.2.2 Ball-In-Pipe Game

At this stage, this game was something completely different to what is in the final product. The first implementation was a game involving a conveyor belt and various shapes. The player had to move the shapes arriving on the conveyor belt into the right pipes. This was later replaced with the ‘Ball-in-pipe’ game visible in the final product.

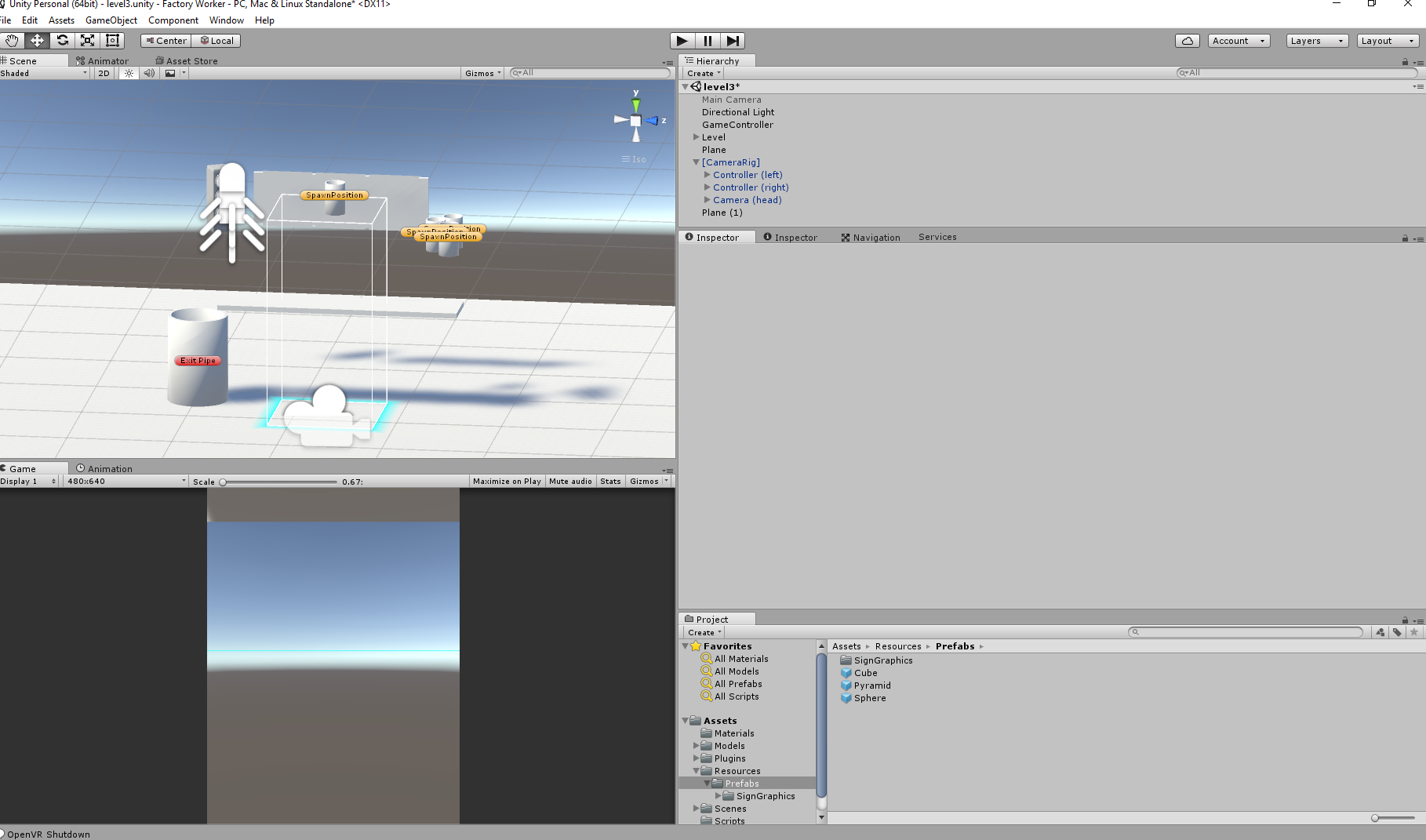


Figure - Initial game idea which was scrapped and replaced

#### 8.2.3 Archery Game

At this stage, the archery game was also unrecognisable to the final version. The initial idea was for an in-door archery range where the player could see all the targets at once and had to shoot all of them to pass the level.

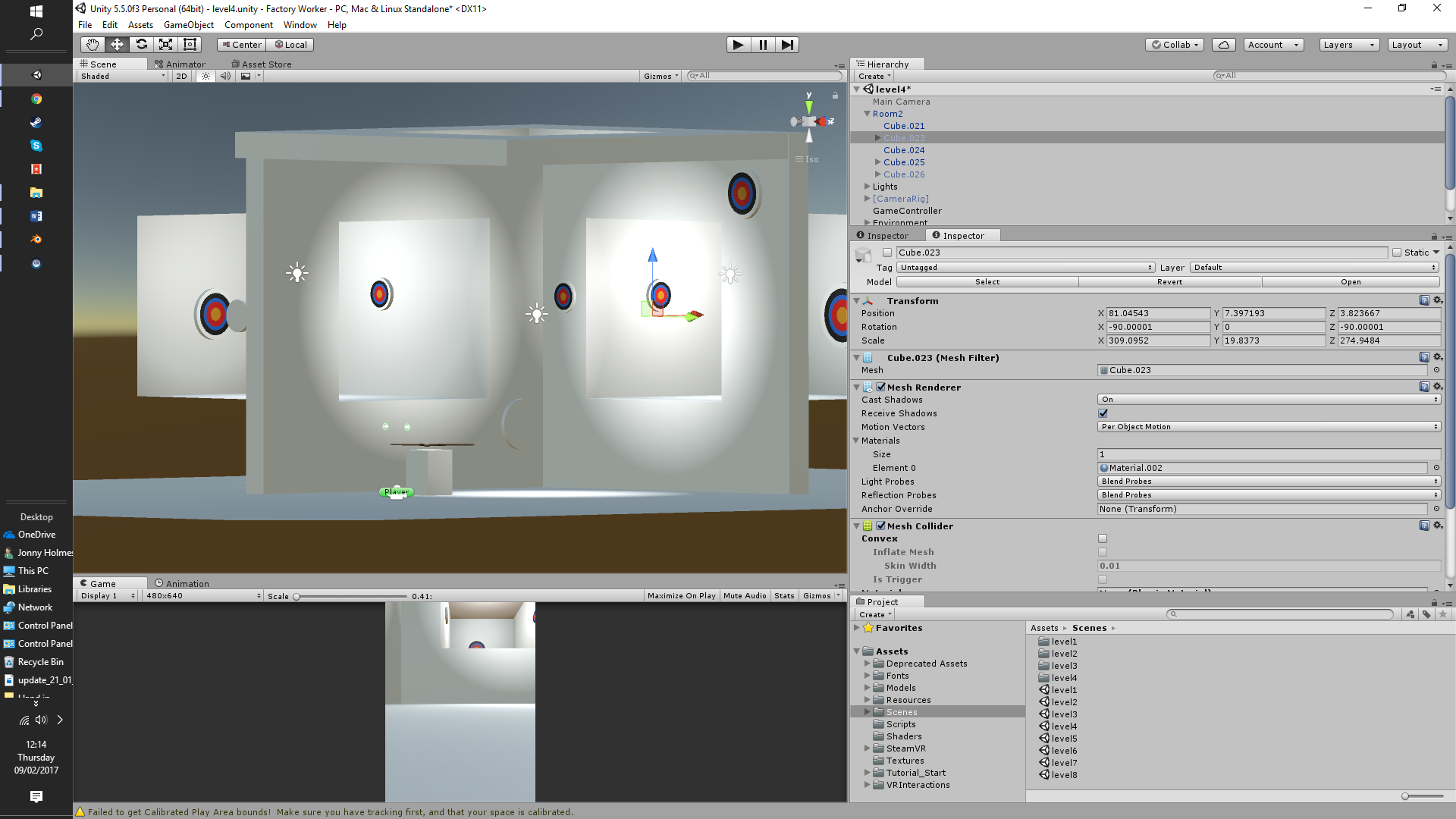


Figure - Initial implementation of the archery game

Deciding to go back to the drawing board with 2 out of 4 of the planned mini-games highlights the importance of prior planning. Time could have been saved in the modelling and implementation phase simply by recognising early that there was a more effective solution available.

### Stage 3

At this point a more consistent and solid identity for the project was drawn up. The idea of the game being in the theme of a child’s ‘play space’ was drawn up. Each game could be a separate play area but should share a consistent theme so that they all feel like they are part of the same world. The game was at this point titled ‘PlaySpace VR’.

#### 8.3.1 Archery Game

The archery game was changed to an outdoor environment and its core game play mechanic changed from “hit all the targets” to “see how many targets you can hit in a specific amount of time”. This change was made after some user testing and the mini-game being described as “lacking a game mechanic that actually holds your attention”. Now the game felt more intense and provided a sense of purpose and urgency.

#### 8.3.2 Painter Game

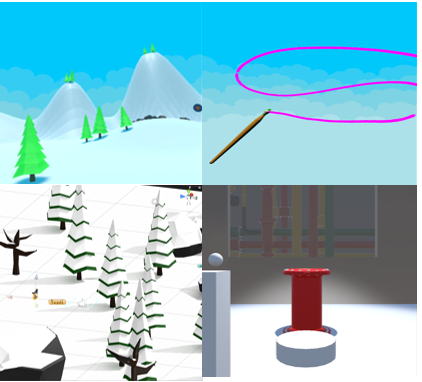
An important design choice was made at this stage for the painter game. At this stage, the user would select the colour that they wanted to paint with by pointing at a colour pallet with one controller and painting with the other, similar to a painter holding their pallet with one hand and the brush with the other. After some user testing it was clear that this was “fiddly” to use and detracted from the intuitiveness of the rest of the game. The interaction system was soon changed so that one hand would hold an eraser and the other a brush. A decision was made to replace the colour pallet with a system of multiple paint pots filled with different colours of paints. The user would dip the paint brush in the paint colour that they wanted to use. The upside of this system was that it was much more intuitive and user-friendly, the downside was that the number of colours available was limited to the number of paint pots, whereas the colour pallet offered an infinite colour range.

#### 8.3.3 Ball-In-Pipe Game

The ball-in-pipe game by this point had been completely revamped, and involved a new mechanic where the user had to bounce a ball off a platform with the correct trajectory such that it landed in a red pipe. This was to be the foundation that the rest of the levels would build upon. Feedback was received that this was a simple but fun and intuitive mechanic.

#### 8.3.4 Snow Game

Some good work was done to the Snow Game by this stage. The user could now play fetch with an animated wolf.



### 8.4 Stage 4

The first half of the final stage was spent adding more features to the mini-games. The second half was spent polishing and bug-fixing. By this point the game had come together well and there was time to add a feature that, under the scope list defined in the PID, was only to be added if the time was available. This was an in-game mascot or assistant.

He was an animated mascot that would fly around each scene being curious and looking at various objects. He was modelled and animated in Blender and was the most complex entity so far modelled. This was because he had multiple layers of animation. His eyes would move around, whilst his thrusters would move smoothly to face towards the ground. He provided a much-needed element of consistency to make all the mini-games feel as though they were connected and part of the same whole. He provided the feeling that he was experiencing the game at the same time as the user. Feedback was received that users liked being able to feel that they were making eye contact with another being, as this felt like a personal interaction.



## Project post-mortem

### 9.1 Personal Objectives

The main personal objectives for this project were to improve the developer’s skills relating to Game development, and to gain experience with VR as the target platform. The former objective has been met as the developer feels much more confident in using Unity3D, C# and Blender. However, he is by no means proficient and there is clearly much more to learn from each of these utilities. The latter has had partial success in that experience has only been gained targeting the HTC Vive. If possible the developer would have liked to make the game cross platform to support the Oculus Rift, but this was impractical in terms of resources and time.

The developer has also become more skilled at using versioning software such as GIT. However, this objective has had limited success as the developer didn’t fully utilise the capabilities of the technology. He did not gain experience in managing and merging different branches and throughout the whole project stuck to one branch. This limited the full array of benefits that versioning offered and the skills learned. In future projects, he will be sure to utilities these features.

Another personal objective was to improve time management skills. This objective has been met through practicing time-boxed sprints where feasible, however it could have been more beneficial if these sprints were more strictly timed and organised, as they ended up being loose guidelines.

Finally, the developer wanted to gain experience in marketing an app. This objective was fully met via the production of a trailer[[1]](#footnote-1) and the listing of the game on the site itch.io.[[2]](#footnote-2) The game has sold one copy as of writing this report.

### 9.2 Deliverables

The main deliverables met the desired expectations well. User feedback gained through Reddit indicated that they did offer a good diversity of experience in terms of VR experience and was appropriate for all ages. However, it also showed that several users weren’t content with some of the mini-games not having an active purpose and just being a relaxing environment (e.g. playing fetch with a wolf). If the project were to be done again it would be ensured that all mini-games had an active game mechanic like the ball-in-pipe game.

## Method of Approach

### 10.1 Methodology

The form of ‘XP’ programming that was followed was a success as it ensured the project was kept organised and on-track. It also allowed the developer to keep track of project priorities. Without this form of approach the project would likely have become disorganised and fallen behind schedule.

### 10.2 Technologies / Implementation

Unity proved a pleasure to work with. It posed very few problems throughout the project and its organisational tools kept the project neat and professional. It also coupled well with the use of Blender and saving Blender files directly into the Unity asset directory, saving time and effort. Blender was very useful for asset creation and animation. Other technologies such as Maya and 3ds Max would have been viable alternatives and posed no difficulties, but the developer was already familiar with Blender, and so time was saved by choosing it. C# posed no problems that couldn’t be solved via quick internet research or the Unity documentation, as the documentation now provides C# examples for almost everything. Choosing Boo or UnityScript would have been possible but would not have been a logical choice as the documentation for these is almost nil.

In terms of music creation for the game. Otomata proved a simple and easy way of procedurally generating music. However, it was limited by having only one sample sound. This gave all music generated a consistent feel to it, but ideally the music would have been better had it been more varied. If the project was done again a music creation tool such as FL Studio would be used to create some more interesting and professional music.

Git proved a good version control software, but did pose some problems during the project. For projects which can include slightly large audio or fbx files, it’s 100MB per file limit proved frustrating. This limit can be upgraded for a fee, however. This was the only frustration with it, and so Git would still be used in projects going forward.

### 10.3 Initial planning (PID, user stories, designs)

The requirements and aims laid out by the PID were mostly met. Of those that were not, this was usually due to the project changing direction over time, making these objectives not entirely relevant. For example, one “definite” aim was that the game should be “puzzle-based”. This only ended up being true in one of the mini-games (the ball-in-pipe game). Despite this, the scope laid out by the PID was a useful tool to allow effective prioritisation by specifying fixed numbers e.g. “At least 15 3D modelled assets”. This gave the project more solid requirements to aim for.

Obtaining user feedback through internet forums such as Reddit proved extremely useful. Targeting a forum that is based around the HTC Vive meant that most people who viewed the post would have a HTC Vive themselves and be interested in or even have experience playing VR games, making their suggestions even more important. Many changes were made to the project based on the feedback received. If the project was done again this would only be done more.

User stories were a small but extremely beneficial part of the planning process. They enabled the project to always keep in mind it’s most important aims in a clear way. If the project was done again, more user stories would have been written to cover a wider variety of user needs.

One thing that the planning phase would have benefited from is use-case diagrams. These were not drawn up to save time but in hindsight would have made the requirements for the project much clearer.

### 10.4 Developer performance

The developer found his performance throughout the project to be good all round but with room for improvement. He lacked the discipline to strictly follow a time-boxed time management methodology but at the same time put in the number of hours required to get the project done to the level that was desired. The developer is very pleased with the final product, especially in terms of its completeness, and feels that he has met almost all the goals that he set out with.

## Conclusions

The project met most of the objectives set out at the start of the project. The developer succeeded in developing the relevant game development, 3D modelling and technical skills. He gained experience in development methodologies, version control, bug tracking and time management. The developer feels the game did offer a good diversity of experience in terms of VR interaction. Even the final goal of publishing the game on a 3rd party seller’s website was achieved. However, there were elements of all development phases which could have been improved. Use-case diagrams and more user stories would have improved the planning phase, whilst stricter time-boxing and scrum management would have improved the workflow and time taken to build the game. More incremental testing would have highlighted beneficial improvements to the game earlier on and saved time. Despite this, the project met most of its objectives and produced a deliverable very close to what was initially planned for.

## Statement of word count

## Reference List

"Get Cardboard – Google VR". *Vr.google.com*. N.p., 2017. Web. 16 Apr. 2017.

"Google Cardboard – Official VR Headsets – Google Store". *Store.google.com*. N.p., 2017. Web. 3

"History Of Virtual Reality - Virtual Reality". *Virtual Reality*. N.p., 2017. Web. 5 May 2017.Apr.

"Samsung Gear VR With Controller". *The Official Samsung Galaxy Site*. N.p., 2017. Web. 11 Apr. 2017.

Swider, Matt. "HTC Vive Vs Oculus Rift: Which VR Headset Is Better?". *TechRadar*. N.p., 2017. Web. 2 May 2017.

"The Lab (Video Game)". *En.wikipedia.org*. N.p., 2017. Web. 2 May 2017.

"Creative Commons — CC0 1.0 Universal". *Creativecommons.org*. N.p., 2017. Web. 5 May 2017.

"Documentation, Unity Scripting Languages And You – Unity Blog". *Unity Technologies Blog*. N.p., 2017. Web. 1 Apr. 2017.

"Otomata - Generative Musical Sequencer". *Earslap*. N.p., 2017. Web. 11 Apr. 2017.

"Royalty Free Music By Bensound | Creative Commons Music". *Bensound.com*. N.p., 2017. Web. 5 May 2017.

"Boids (Flocks, Herds, And Schools: A Distributed Behavioralmodel)". *Red3d.com*. N.p., 2017. Web. 2 May 2017.

"Csharp Coding Guidelines - Unify Community Wiki". *Wiki.unity3d.com*. N.p., 2017. Web. 29 Apr. 2017.

"Don't Repeat Yourself". *En.wikipedia.org*. N.p., 2017. Web. 5 Apr. 2017.

Technologies, Unity. "Unity - Scripting API: Monobehaviour". *Docs.unity3d.com*. N.p., 2017. Web. 4 May 2017.

"Ethics Of Virtual Reality Applications In Computer Game Production". N.p., 2017. Web. 10 May 2017.

Studio. "Academy PRO: Unity 3D. Scripting". *Slideshare.net*. N.p., 2017. Web. 5 May 2017.

"Virtual Reality And Ethical Issues - Virtual Reality". *Virtual Reality*. N.p., 2017. Web. 10 May 2017.

"Virtual Reality Sickness". *En.wikipedia.org*. N.p., 2017. Web. 10 May 2017.

## Appendices

### 14.1 User Guide

Player Guide:

1. Launch the game executable.
2. You will find yourself in the game lobby. Choose a level by grabbing one of the yellow spheres which show the text ‘Grab Me’.
3. In the painter level, one controller acts as a paintbrush and the other an eraser. With the hand holding the paintbrush, pull the trigger on the controller to paint. To change the colour of the paint, dip the brush in one of the paint pots on the table.
4. In the snow level, grab the stick out of the wolf’s mouth with the trigger, and throw it by releasing the trigger.
5. In the Ball-In-Pipe game, pick up the yellow ball by holding your controller over it and pulling the trigger. Release the trigger to throw the ball. Try to bounce the ball on the platform and into the pipe.
6. In the archery game – pick up the bow by hovering over it with one of the controllers and pressing trigger. Use your other controller to grab the bow string, at which point an arrow should appear. Pull the bow string back, and release. Try and hit as many targets as possible in the time given.
7. To return to the level hub, grab the yellow sphere in each level showing the text ‘Menu’, or just press the menu button on the controller at any time.
8. To reset the high scores, grab the red sphere in the level hub showing the text ‘Reset High scores’

### 14.2 Project Management Artefacts

#### 14.2.1 PID

**Introduction to the project**

The intended project involves creating a Virtual Reality game made using Unity3D, intended for use with the HTC Vive HMD (Head mounted display).

Although I do not have an intended client at present, I believe there are many potential clients for my product. Small innovative indie games can often end up on digital distribution platforms such as Steam after being noticed by online communities such as Steam Greenlight. In this case the intended client any person interested in pc gaming, particularly those who are interested in small-scale, innovative games.

**Background / Motivation + Business Case**

The primary reason I am undertaking this project is to further my skills in Game Design, asset creation/3D modelling and C# programming. In addition, carrying out a project of this scale from start to finish will give me invaluable experience in utilizing software development processes/methodologies.

In terms of a Business case, consumers might purchase my product because they are interested in small indie games with a low-price tag. Another potential reason is that currently the VR games industry is not yet saturated, and small games have more of a chance of being noticed and bought.

**Project Objectives**

1. To complete a requirements plan specifying the core features of the game, and any additional features that will be included if time allows.
2. To complete the Unity3D game in line with the specified requirements.

**Initial Scope**

**What will definitely do -**

1. The proposed game / product will feature
   1. A puzzle based game
   2. Ability to pick up and interact with objects using the VR controllers/wands
   3. At least 20 short puzzle based levels
   4. A variety of 3D modelled assets (at least 15) to give a consistent art style whilst providing an engaging environment.
   5. A variety of hand recorded sounds (at least 10), to make the environment more immersive

**What I think I am likely to achieve –**

I feel confident that I can achieve all the above.

**What I would really like if I had the time –**

I would very much like to include some polished looking animation and particle effects to give it that professional feel. I would also like a well built ‘game assistant’. A scripted character or voice which helps the player along and is triggered into saying certain lines etc.

**What it won’t do –**

It won’t feature any multiplayer functionality at all. Multiplayer VR is currently just not an easy task to get right.

**Platform**

The project will be for PC Windows only, as it using the HTC Vive.

**Resources and dependencies**

The project is critically dependent upon:

* My HTC Vive VR headset remaining in a working state.
* My VR-ready PC components e.g. gpu remaining in a working state.

**Method of approach**

The software development process I will adopt will be a scaled down version of XP (‘Extreme Programming’). This will involve -

1. Keeping a spreadsheet of required features, prioritized.

2. Define acceptance criteria (what done looks like).

3. Define specific engineering tasks to get done.

4. Time-box each session

5. Utilise effective versioning

6. Frequently reprioritize my spreadsheet appropriately.

The technologies I will be using include -

1. Unity 3D

2. Mono-develop (simple scripting software provided with Unity 3D. Using C#.

3. Blender (3D modelling software)

**Why Unity3D?**

I am using Unity3D as it is a high level tool with built in support (currently in the form of a plug-in), for the HTC Vive. The only other viable alternative for my project would have been Unreal Engine, so I made a call and chose Unity3D. I chose this because it feels more intuitive and can do everything I need to do for the project.

**Project Plan**

|  |  |  |  |
| --- | --- | --- | --- |
| **Stage** | **Expected Start Date** | **Expected Completion Date** | **Products/Deliverables** |
| 1. Initiation |  | 30th Dec | PID |
| 1. Investigation and outline requirements | 25th Jan | 1st Feb | Outline requirement / features of the game. Core + additional. |
| 1. Initial high level design | 2nd Feb | 10th Feb | Get high level features in place. Solidify the base structure of the game. |
| 1. Increment1 | 11th Feb | 25th Feb | Increment requirements and design.  Choose features from requirements to complete this increment, complete them. |
| 1. Increment2 | 26th Feb | 12th March | Increment requirements and design.  Choose features from requirements to complete this increment, complete them. |
| 1. Increment3 | 13th March | 27th March | Increment requirements and design.  Choose features from requirements to complete this increment, complete them. |
| 1. Increment4 (Easter Vacation) | 3rd April | 21 April | Increment requirements and design.  Choose features from requirements to complete this increment, complete them. |
| 1. Assemble and complete final report | 22nd April | 5 May | PRC0304 Report |

**Control Plan**

The following PRINCE2 control techniques will be employed:

1. Highlight reports as dictated by the PRCO304 module – brief review of the stage - whether the stage’s objectives, deliverables and timescales were met , and to make sure the appropriate action is taken if not. This will ensure regular feedback on the project is obtained and the project stays on track.

2. Review meetings with project supervisor as dictated by the PRCO304 module; additional ad-hoc meetings as are necessary

**Communication Plan**

Review meetings will be held with the supervisor in line with the Control plan. Further ad-hoc communications will take place as needed. There are no real stake holders involved. This guarantees constant feedback on the project’s progression. Meeting records will be documented to make sure everything discussed is recorded. If the project goes off track then meetings may increase, and vice versa.

**Initial Risk List**

|  |  |
| --- | --- |
| Risk | Management strategy |
| VR Headset breaks. | I would either attempt to get my hands on another headset, or attempt to change the game into a non-vr interaction game. |
| Schedule Overrun | An exception plan will be developed and approved by the project supervisor. |
| Difficulty learning/using the development technologies | I have already reached the point of a basic prototype, so this is a very low risk |
| Technology failure | Backups using Git will be taken frequently. |
| Overy optimistic deliverables | Again, unlikely due to the fact that I already have a prototype working, but nevertheless possible. In this case I would have to re-evaluate the requirements and deliverables and update the project as necessary. |

**Initial quality plan**

|  |  |
| --- | --- |
| Quality Check | Strategy |
| Requirements | Requirements will be iteratively checked to ensure they are still feasible, correct and up-to-date. |
| Design validation | The design will be checked against requirements as the project goes forward. |
| Project Deliverables | Validate the deliverables throughout the project to ensure all deliverables are still realistic. |

**Legal, ethical, social and/or professional issues**

As I am creating a game (with no mature content), this will not be a huge topic, but nevertheless must be considered. I will consider this in my final report.

### 14.2.2 Highlight Reports

|  |
| --- |
| **PRCO304: Highlight Report** |
| **Name: Jonathan Holmes** |
| **Date**: 09/02/17 |
| **Review of work undertaken**First highlight (so I will also include details of tasks more than a week ago. I have finished the high level structure of the scene e.g. the environment models etc. I have a basic prototype of the first scene. No issues since the last highlight. I have uploaded the first of my highlight videos onto my YouTube account (private settings), for my project supervisor to review, and have also prepared a executable where you can move the camera around in a basic form. (The real project of course will be using a VR camera).  I also set up GitHub for my project and committed my changes so far, and an Excel sheet Gantt Chart to serve as a plan. I have completed the tasks for this week according to my plan, which was to   * Set up GitHub * Model the environment assets in Blender * Complete Scene1 ( consists of 4 sub-scenes )   The only issues I have identified and solved were issues with GIT. GIT has a maximum file size of 100MB and if you commit a file to your repo above 100MB it will throw an error. Not only this, but you must carefully remove the file from your repo, github and from your entire commit history (the tedious bit), before you can commit again. Unity generates MetaData files which can exceed these sizes, and some of my more complex Blender models also exceeded these sizes. To fix this, I added the Unity metadata file to my ‘.gitIgnore’ file, and in Blender changed the default setting to compress the files. I have spent the necessary 30 hours on the project this week. |
| **Plan of work for the next week** *(derived from the current stage plan).*  Next I will set up Scene2, in accordance with my Gantt Chart plan. I will continue to backup to github and revise my Gantt Chart plan where necessary. |
| **Date(s) of supervisory meeting(s) since last Highlight**  First Highlight. Had a brief check-up last week to check everything was on track and ready to dive in. |
| **Notes from supervisory meeting(s) held since last Highlight** |
| **Stage review** *In the case when a stage has completed since the last Highlight, a brief review of whether the stage’s objectives, deliverables and timescales were met (or not).* |

|  |
| --- |
| **PRCO304: Highlight Report** |
| **Name: Jonathan Holmes** |
| **Date**: 16/02/17 |
| This week I finished 2 deliverables, both of which were scheduled for this week on my Gantt chart. I added a new scene called the Forest Scene, and modelled and animated an in-game assistant.  The Forest scene – I wanted a scene where the player could relax in VR and just look around and be in a nice world. So I modelled a snowy forest scene in blender with trees, rocks etc. I also modelled a fully animated wolf. In the scene the player can play fetch with the wolf.  The in-game assistant was a feature that I was only going to implement if I had enough time. I felt however that I was ahead of schedule so decided to do it now.  I have been pleased to see the GIT problem that I was having for weeks previously ( which I mentioned in last weeks report) has not re-surfaced at all since the fix last week. I have spent between 20-25 hours on the project this week. This is under target but I feel as though this is simply because I did a lot of preparation before the project started, and I needed no more than this to complete the deliverables specified by my plan. |
| **Plan of work for the next week** *(derived from the current stage plan).*  Next week I will be starting the next scene (scene 4), which will be another mini game. The details of which are still being solidified. The nature of my game (series of unrelated mini-games / VR experiences ). Means that I can comfortably change around the details of each game at quite short notice before implementation, as they do not affect any of the other games/scenes. |
| **Date(s) of supervisory meeting(s) since last Highlight**  No meeting since last highlight. I will schedule a meeting between now and the next highlight. |
| **Notes from supervisory meeting(s) held since last Highlight** |
| **Stage review** *In the case when a stage has completed since the last Highlight, a brief review of whether the stage’s objectives, deliverables and timescales were met (or not).* |

|  |
| --- |
| **PRCO304: Highlight Report** |
| **Name: Jonathan Holmes** |
| **23/02/2017** |
| This week I have revamped the archery level. I modelled a new terrain environment in Blender and exported to Unity. This wasn’t explicitly on my plan but I decided I was dissatisfied with the current level. I have also started on the next game which will be a VR painting game similar to Tilt Brush. Progress this week has been hindered by a deadline for my other module. I may have to extend the deadline for the next scene on my Gantt chart if need be. I feel comfortable to do this as I have left enough time spare for these sort of delays. |
| **Plan of work for the next week** *(derived from the current stage plan).*  Next week I will continue on the next painter scene. I will also start looking into implementing some sound effects into the game. |
| **Date(s) of supervisory meeting(s) since last Highlight - 20/02/17** |
| **Notes from supervisory meeting(s) held since last Highlight**  We looked at my progress video on YouTube and discussed whether I felt I was on track or not. We decided the project is currently on track. |
| **Stage review** *In the case when a stage has completed since the last Highlight, a brief review of whether the stage’s objectives, deliverables and timescales were met (or not).*  As of this review I have completed 3 of the stages laid out in my PID on time. I have completed the ‘Initiation’ stage, which involved writing my PID. I have completed stage 2 – ‘Investigation and outline requirements’. I did this in my PID and in a separate document that I have uploaded under ‘Miscellaneous deliverables’ on SPMS. I have also completed Stage 3 – Which involved putting in place the high level features and solidifying the base structure of the game. |

|  |
| --- |
| **PRCO304: Highlight Report** |
| **Name: Jonathan Holmes** |
| **Date**: 04/03/17 |
| **Review of work undertaken**  This week I focused on QoL improvements in small bursts as I was limited with the amount of time I had (due to interviews and other module etc). I did also make some new features for the painter game. After a while in Unity your assets can become quite untidy which can decrease productivity, so I spent some time organising it all. The more projects I undertake the better I get at tidying on the go, and will take this as a lesson going forward. I also added some features and bugfixes throughout the project, and made a video showing where I am in the project. - https://www.youtube.com/watch?v=pgHPsqti-A4 |
| **Plan of work for the next week** *(derived from the current stage plan).*  *Looking at my gantt chart, I can see I was behind schedule for the next scene (painter scene). Although I have made good progress in it I have increased the number of work days by a large amount to keep myself on track. I will continue to work on this next week.* |
| **Date(s) of supervisory meeting(s) since last Highlight**  20/02/17 |
| **Notes from supervisory meeting(s) held since last Highlight**  We looked at my progress video on YouTube and discussed whether I felt I was on track or not. We decided the project is currently on track. In-line with my PID, I am currently in increment 3. Despite my initial head-start, I am starting to feel the pressure ramping up. |
| **Stage review** *In the case when a stage has completed since the last Highlight, a brief review of whether the stage’s objectives, deliverables and timescales were met (or not).* |

|  |
| --- |
| **PRCO304: Highlight Report** |
| **Name: Jonathan Holmes** |
| **09/03/17** |
| This week I have added many more levels to one of the scenes. The player can now go to the next or previous level. There are 15 levels to the first pipe game scene now. I have also completed the Painter game scene. It will be decided later whether there is anything else I would like to add to it. |
| **Plan of work for the next week** *(derived from the current stage plan).*  *For the next week I will be improving the level hub, which models made in blender. I also need to add more levels to the archery game scene (as I have done for the pipe game). I also need to plan how I will add sound to the game. As I recognise this is an important feature in a VR game that should not be overlooked.* |
| **Date(s) of supervisory meeting(s) since last Highlight - 20/02/17** |
| **Notes from supervisory meeting(s) held since last Highlight** |
| **Stage review** *In the case when a stage has completed since the last Highlight, a brief review of whether the stage’s objectives, deliverables and timescales were met (or not).*  None since last report. |

|  |
| --- |
| **PRCO304: Highlight Report** |
| **Name: Jonathan Holmes** |
| **18/03/17** |
| This week I did more general improvements along with some features specified on my Gantt chart. I modelled some more detailed level pedestals. I also added in more levels for scene 1 (pipe game). I added 15 new levels. I also added sounds in accordance with my gantt chart (this was much overdue). I added sounds to the archer game and the pipe game. I also redesigned the level hub. A very productive week. |
| **Plan of work for the next week** *(derived from the current stage plan).*  *Next week I will add more features to the archer game. Such as changing the difficulty. This is a reasonable task in terms of effort days.* |
| **Date(s) of supervisory meeting(s) since last Highlight - 20/02/17** |
| **Notes from supervisory meeting(s) held since last Highlight** |
| **Stage review** *I am now in increment 3. Nearing the final increment (over Easter vacation). I feel as though I am on track.* |

|  |
| --- |
| **PRCO304: Highlight Report** |
| **Name: Jonathan Holmes** |
| **23/03/17** |
| This week I have been polishing and bug fixing. I have also re-vamped the painter scene so that instead of using a colour wheel and a laser pointer to select the colour you are going to paint with, you instead dip your brush into a paint pot. This was a design choice I had been debating for a while. The pros of the colour wheel/laser pointer is that you can choose literally any colour you want, but the drawback is that it is fiddly and not intuitive. The pros of the paint pot approach is that it is more intuitive and immersive, however the con is that you are restricted to a certain number of colours. I have also added the other hand to be used as an eraser, to erase lines that you paint. |
| **Plan of work for the next week** *(derived from the current stage plan).*  *In my last highlight I said I would be adding more features to the archer game. I didn’t end up doing this so I will try and do this this week.* |
| **Date(s) of supervisory meeting(s) since last Highlight - 20/03/17** |
| **Notes from supervisory meeting(s) held since last Highlight**  Video of working VR portal and several mini-games (ball into the hat, archery, paint shapes, dog and stick). |
| **Stage review** |

|  |
| --- |
| **PRCO304: Highlight Report** |
| **Name: Jonathan Holmes** |
| **01/04/17** |
| This week I have mainly been focusing on other modules, but I had time to gather some user feedback via an online forum. This feedback was based on a video demo. I have recorded the useful feedback and plan to act on quite a bit of it. I also did general bug fixing and improvements. Made a new video. Started my final report. |
| **Plan of work for the next week** *(derived from the current stage plan).*  *Action feedback from online forum.* |
| **Date(s) of supervisory meeting(s) since last Highlight - 20/03/17** |
| **Notes from supervisory meeting(s) held since last Highlight**  Video of working VR portal and several mini-games (ball into the hat, archery, paint shapes, dog and stick). |
| **Stage review**. |

### Other materials (designs, test results)

#### User Stories

As a player, I want to be able to interact with objects using the controllers so that I can feel like I’m having a real impact on the environment around me.

As a player, I want to be able to try a variety of experiences so that I can really see the potential of VR.

As a player, I want the experiences to be intuitive, I don’t want to have to ask how to do anything, so that I remain fully immersed.

As a player, I want to be able to hear spatial audio around me so that I feel fully immersed.

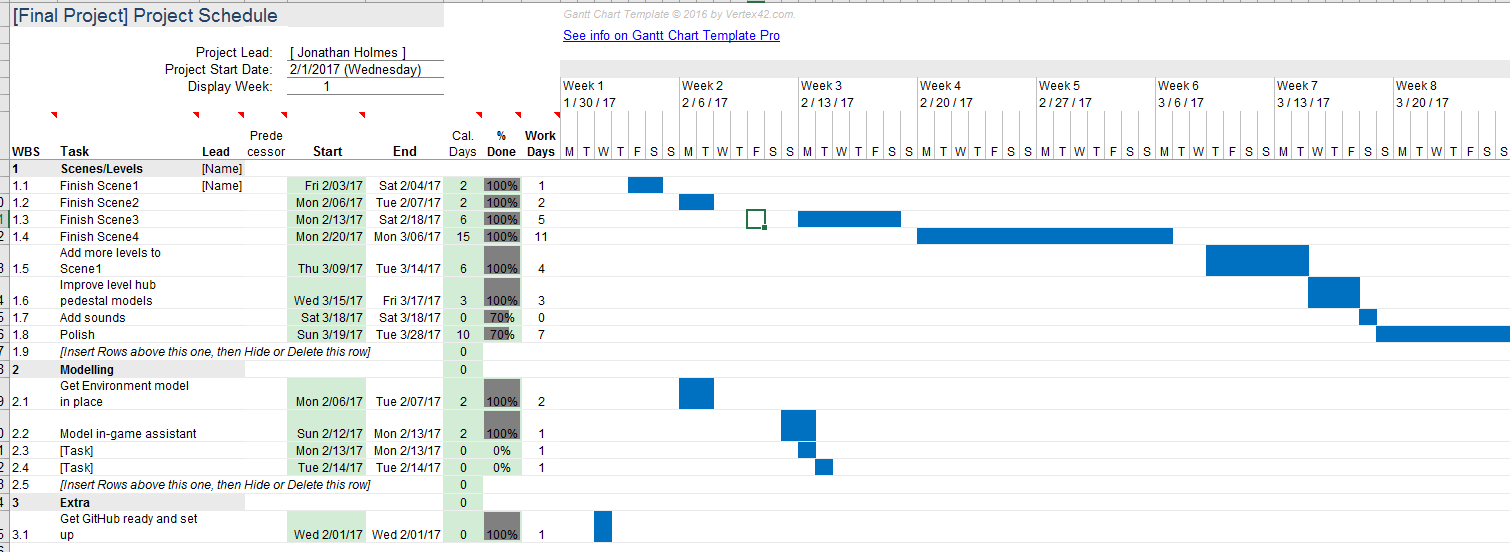
As a player, I want transitions between scenes to be smooth so that I don’t feel motion sickness.

As a player, I want to be able to move around the room-space so that I feel fully immersed.

#### 14.3.2 Designs



#### Gantt Chart



#### Requirements List

**Core**

|  |
| --- |
| Ability to pick up and interact with objects using the VR controllers/wands |
| At least 20 levels |
| A variety of 3D modelled assets (at least 15) to give a consistent art style whilst providing an engaging environment. |
| A variety of hand recorded sounds (at least 10), to make the environment more immersive. |
| Smooth gameplay, with as little motion sickness as possible, where possible. |

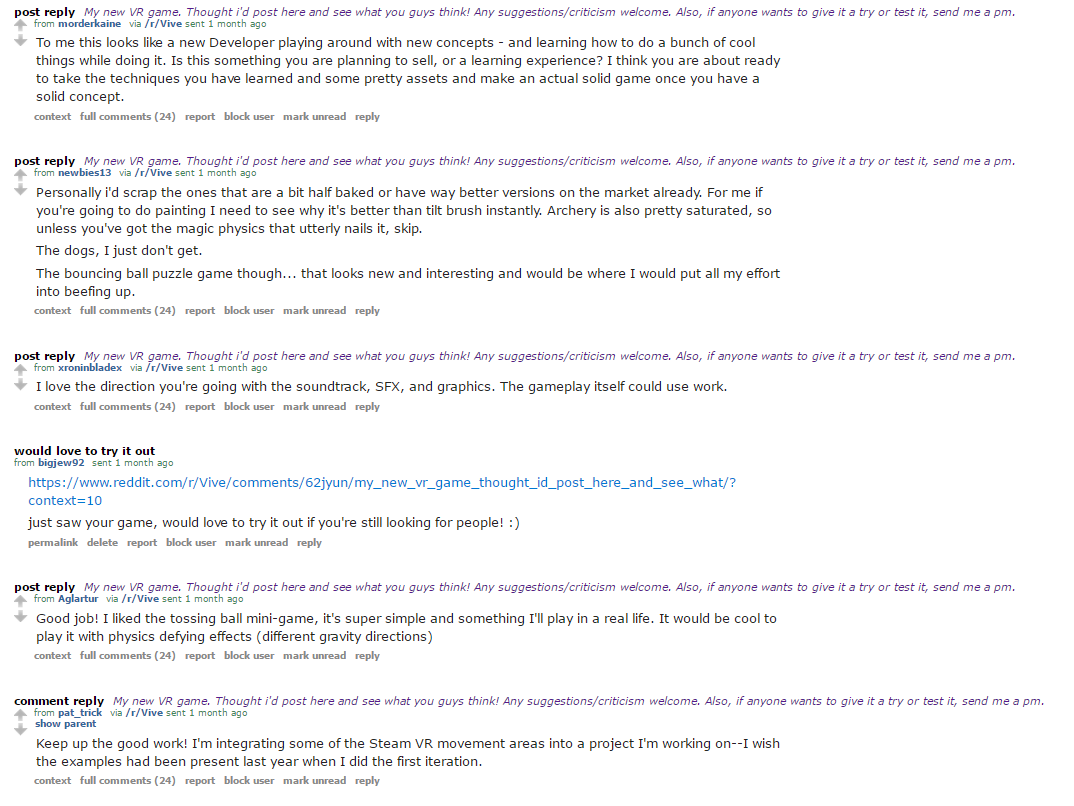
**Likely additions / I should have the time to implement these, but if not I still have the core functionality**

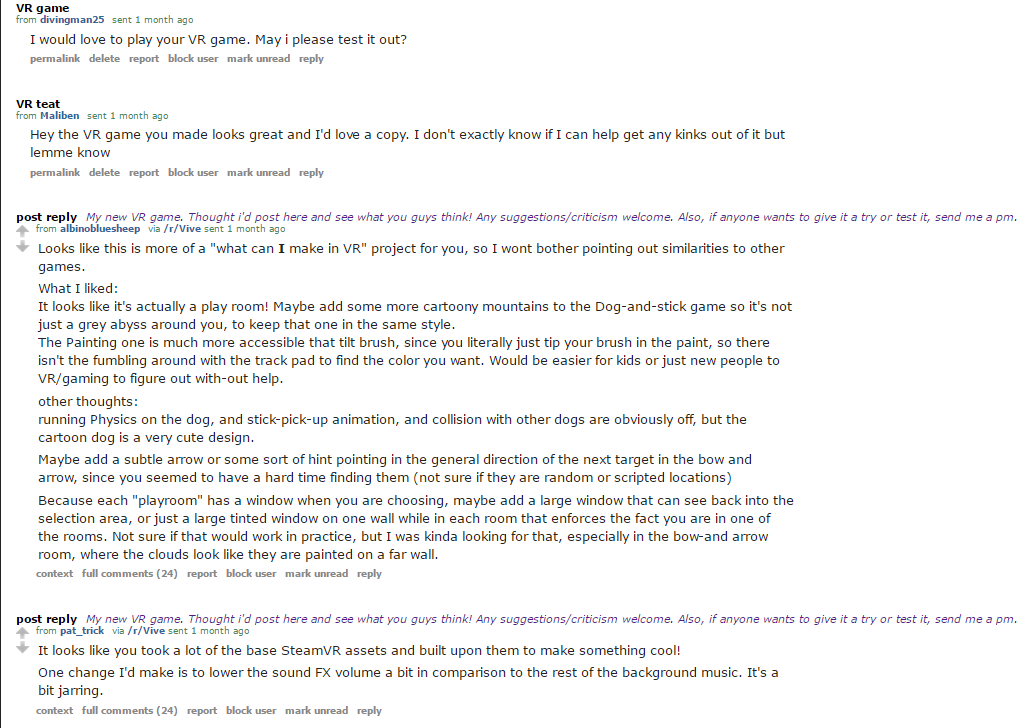
|  |
| --- |
| Animations created in Blender for some assets, to bring the scene alive. |
| A scripted AI character which talks to the player, and gives the player assistance. |

**If I have time**

|  |
| --- |
| More levels, around 30-40. |
| An immersive UI menu to choose the levels |
| A music track playing in the background. |

#### Reddit User Feedback





1. https://www.youtube.com/watch?v=6GpEoGTDQng [↑](#footnote-ref-1)
2. https://happyadjustablespanners.itch.io/playspacevr [↑](#footnote-ref-2)