**The Lab 2**

Final Report for CS39440 Major Project

*Author*: Shankly Richard Cragg ([shc27@aber.ac.uk](mailto:shc27@aber.ac.uk))

*Supervisor*: Dr David Hunter (dah56@aber.ac.uk)

26th April 2017

Version 1.0 (Draft)

This report is submitted as partial fulfilment of a BSc degree in  
Computer Science (G401)

Department of Computer Science

Aberystwyth University

Aberystwyth

Ceredigion

SY23 3DB

Wales, UK

**Declaration of originality**

I confirm that:

* This submission is my own work, except where clearly indicated.
* I understand that there are severe penalties for Unacceptable Academic Practice, which can lead to loss of marks or even the withholding of a degree.
* I have read the regulations on Unacceptable Academic Practice from the University’s Academic Quality and Records Office (AQRO) and the relevant sections of the current Student Handbook of the Department of Computer Science.
* In submitting this work I understand and agree to abide by the University’s regulations governing these issues.

Name Shankly Richard Cragg

Date 26th April 2018

**Consent to share this work**

By including my name below, I hereby agree to this dissertation being made available to other students and academic staff of the Aberystwyth Computer Science Department.

Name Shankly Richard Cragg

Date 26th April 2018

**Acknowledgements**

I am grateful to…

I’d like to thank…

I’d like to thank my supervisor, David Hunter, for being a valuable source of knowledge, and great sounding board for ideas.

Jon Shire for guidance on writing fantastic reports.

My mother, Michelle Cragg, and father, Philip Cragg, for always encouraging me to be my best.

**Abstract**

With the ever-increasing presence and development of virtual reality in the modern world [1], exploring virtual reality as a medium is of paramount importance as mainstream appeal grows.

This project takes inspiration from Valves “The Lab” [2], and Owlchemy Labs “Job Simulator” [3], and looks to create an experience which is difficult or dangerous to recreate in physical reality, taking full advantage of the strengths inherent in virtual reality as a medium.

“The Lab 2” is therefore an introductory tool to the possible experiences virtual reality can give which cannot be taught better any other way. Specifically, this project looks to give the unusual experience of being a Steam Engine Fireman [4], who must keep his train running under the Drivers orders.

With a coherent play space, and audio/visual feedback, this experience is functional as a user’s first virtual reality experience, while offering something to those who are familiar with the platform.

This report details the process of a developing with virtual reality for the first time, and the process used to ensure development went as smoothly as possible. With analysis of the challenges a 1-man team faces when trying to develop a feature-rich and complete experience.

**Contents**

1. Background 6

1.1. Background Research 6

1.2. Ideation 6

1.3. Motivation 6

2. Analysis 7

2.1. Existing Projects 7

2.1.1. The Lab 7

2.1.2. Job Simulator 8

2.2. Hardware 9

2.3. Game Engine 9

3. Process 10

3.1. Scrum for 1 10

3.1.1. Sprints 10

3.1.2. Task list 11

3.2. Documentation 12

4. Objectives 13

4.1. The Task 13

4.2. What this entails 13

5. Restrictions of VR 14

5.1. Effective Resolution of Head-mounted Displays 14

5.2. Maintaining suitable frame rate 14

5.3. Play space size 15

6. Designing the Game 16

6.1. Environment and motion 16

6.1.1. Static environment 16

6.1.2. Static train 16

7. Implementation 17

8. Testing 18

9. Critical Evaluation 19

9.1. Were the requirements correctly identified? 19

9.2. How correct were the design decisions? 19

9.3. Could a more suitable set of tools have been chosen? 19

9.4. How well did the software meet the needs of those who were expecting to use it? 19

9.5. How well were any other project aims achieved? 20

9.6. What wasn’t achieved 20

9.7. What would change if starting again? 20

10. Appendices 21

10.1. Third-Party Code and Libraries 21

10.1.1. Unity [14] 21

10.1.2. VRTK - Virtual Reality Toolkit [17] 21

10.1.3. SteamVR [20] 22

10.2. Ethics Submission 23

10.3. Scrum User Story Examples 25

10.3.1. Story 1 25

10.3.2. Story 2 25

10.4. Bibliography 26

# Background

## Background Research

Valve Corporation [5] is an American video game developer and digital distribution company. They are the developers of a video game released in April 5th, 2016 called “The Lab”. This was Valve attempting to “Understand existing genres though the lens of VR” [6]. The Lab features 8 short and varied experiences, ranging from Archery to a visual representation of the Solar System.

“The Lab” has the difficult task of being the first virtual reality experience for a huge number of consumers, not only in 2016 when the game was released, but even now in 2018 and beyond. Valve accomplish this with comical slapstick humour and simple intuitive controls and tutorials aplenty. The Lab is easy to pick up and play with no prior experience with the virtual realm to this day, with the game’s polish and wide appeal a testament to the developers.

This project aims at acting as a spiritual successor to the fun fuelled mini-game collection by Valve, with a similar focus on a unique scenario and fun intuitive interaction with Objects and the Environment.

## Ideation

The inspiration for the theme and feel of “The Lab 2” and the Steam Engine Fireman idea came from Owlchemy Labs “Job Simulator”. This game features a range of real world jobs turned comical such as “Auto Mechanic” and “Gourmet Chef”. These exaggerated experiences take place in an environment many users will be intimately familiar with. For the project a decision was made to take the dynamic of hectic interactive gameplay in a workplace and look for a dangerous or difficult to recreate scenario in the real world to apply these criteria to.

After some brainstorming, I came upon the role of a “stoker”, or “fireman”, whose job it is to tend to the fire of a steam engine, by shovelling coal into the engines firebox.

## Motivation

My motivation for this project stems from the exciting range of experiences afforded to me during my Industrial Year. I had the chance to work with Augmented Reality (AR) using the Microsoft HoloLens [7], which similarly to virtual reality development can make use of cross-platform game engine Unity [8]. This experience with AR development was a joyous one and having been introduced to virtual reality for the first time with their internal HTC Vive kit gave me great curiosity into immersing myself into the world of virtual reality development. This project is a great opportunity to further develop this skillset.

# Analysis

## Existing Projects

### The Lab

The project is to create a virtual reality experience, and as such my research involved the playtesting of existing games found on the Steam platform.

The research done showed that the best VR experiences focus on interaction as main motivation of play. For example, in figure 1 below, Valves “The Lab” allows users to travel to exotic and beautiful spaces to explore, including a Venice town square, or the top of a mountain in the Icelandic wilderness. However, the captivating thing to do in these areas is play with some sticks present in all locations. Throwing them, juggling them, watching them bounce off things. This focus on physical interaction and gameplay came as the standout experience even in the face of such beautiful vistas.

Figure "Postcards" from The Lab



Source: Gameplay from “The Lab”

This ensured that the very core of my gameplay loop was going to encourage constant interaction with the elements in the world.

### Job Simulator

Something Owlchemy Labs “Job Simulator” had to offer was the vibrant and expansive experience that can be offered in small spaces.

Figure Job Simulator Office Level



Source: Steam page for Job Simulator

The variety of interaction in such a small space was compelling gameplay. As a 1-man project, it made a huge amount of sense to limit the design space to as small a space as possible in order to reduce the amount of environment design necessary, an area I feel particularly uncomfortable with going into the project.

## Hardware

Regarding hardware there are 2 main competitors when it comes to virtual reality Systems and Head Mounted Displays (HMDs). These are the HTV Vive, and the Oculus Rift [9]. These direct competitors have a lot in common, and the choice of platform for most users comes down to price point and experiences which are exclusive to each system.

For this project, I am developing with the HTC Vive, as the University Campus features a room complete with a HTC Vive system. Because of this, I will also acquire a HTC Vive to aid development off-site and keep playtesting as convenient and available as possible.

## Game Engine

Similarly, there are 2 options concerning game engine to develop on as well. These are Epic Games “Unreal Engine 4” [10], and Unity Technologies own “Unity”. Both are cross-platform game engines free to use for development by individuals or for educational purposes. Once again this comes down to personal preference. I have some prior experiences working within Unity, and when researching development kits, the most popular one found named “VRTK” (Virtual Reality Toolkit) is exclusive to Unity.

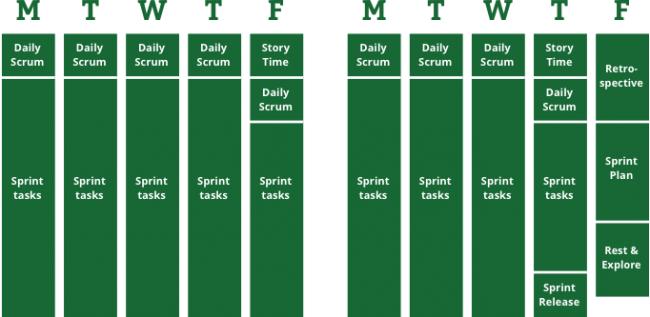
# Process

## Scrum for 1

### Sprints

For this project, I decided to use a modified version of the agile methodology Scrum. When researching single team development practices, I came across an article from Alex Andrews titled “Scrum Of One: How to Bring Scrum into your One-Person Operation” [11]. Here he discussed a 2-week sprint cycle, shown in figure 3 to visualize his process.

Figure Alex Andrew's 2-week sprint

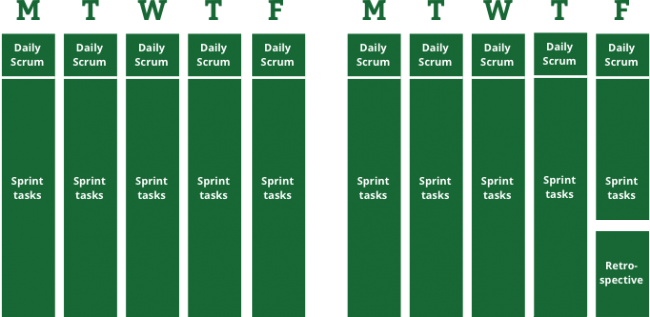


There were a few changes made to this schedule. The major made change to the schedule was the removal of “Story time” as a time dedicated to the direction of the project. As the project was working in unfamiliar space, sizings and the direction of the project was going to be incredibly fluid. This was instead done dynamically during the Daily Scrum at the start of each day, and reflectively at the end of the day as I wrote my daily diary.

The “Big picture” would instead be thought about in the bi-weekly retrospective. At the end of every 2 weeks the previous week would be reviewed, with concern regarding about which tasks were easier, and which tasks were harder than expected. This allows maximum focus on development time and gaining experience with the technology and give more leeway to on the fly decisions being made as new knowledge is gained.

This agile approach suits the project due to the open-ended nature of the project, and the lack of experience working with the technology. An agile process allows more flexibility over the day to day activities compared to a plan driven approach. The streamlined process used can be seen in figure 4.

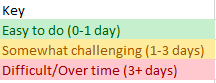
Figure Adapted variation on the 2-week sprint



To keep track of tasks to be done, were in the process of doing, and had been completed, a task list was kept in an excel document. This had a key for the time it would take to accomplish a task seen in figure 5.

### Task list

Figure Key for the task list



These tasks were considered at the start and end of every working day as I wrote my daily diary. As iterations continued, new tasks could be added, or their expected sizing changed based on previous experiences. The main goal of this was to ensure that what amounts to a sprints worth of work was being completed every sprint. Naturally this wasn’t always the case due to inaccurate sizings, but this made sure progress was consistent, and kept track of while multiple tasks were being developed.

At the start of the project, a few user stories were created to give some concrete end goals for the project. Some of these are epics which struggled to be narrowed down to smaller concrete user stories, such as;

*“As a player, I want a high FPS so that I avoid feelings of motion sickness and lack of immersion.”*

This is not a goal which can be completed within a single sprint, and it cannot be compartmentalised into smaller stories, as optimisation of individual parts of the system can only come about after each in game element is added. The stories mentioned can be seen in Appendix [APPENDIX WITH STORIES HERE].

## Documentation

As part of using scrum, both a daily diary and updated task list were used to keep track of progress on various aspects of the project.

The code is documented with XML documentation for all classes and functions. XML documentation is generated at compile time, and can be distributed with the .NET assembly such that IDEs such as Visual Studio can show information about the classes and functions within.

The codes progress was also documented via frequent commits to GitHub, whose version tracking software keeps a record of all changes made to the program alongside my comments on what was done.

# Restrictions of VR

## Effective Resolution of Head-mounted Displays

We mention effective resolution when discussing the reality of using Head-mounted Displays, because looking purely at the number of pixels on each screen doesn’t give enough information to properly convey the quality of the image.

When discussing HMD resolution, what matters is called “*pixels per degree*” or ppd. The reason this is a better metric than others such as total pixels, or pixels per inch, is that it takes into account magnification.

The Vive has two displays, one per eye. Each individual display has a resolution of 1080x1200 for a combined resolution of 2160x1200.

Put simply, despite the relatively good resolution compared to your average 1920x1080 desktop monitor, the 2160x1200 pixels of the Vive are spread over a much larger area due to how close they sit to your face. This very large field of view that only a nominally larger number of pixels must cover leads to individual pixels appearing larger to the viewer, and therefore when using the Vive, users may report it looking “pixelated”.

What this means for the project is that small, intricate details are harder to see clearly in Virtual Reality. This includes effects such as small text being near impossible to read, and complicated/intricate objects and textures being hard to properly understand.

## Maintaining suitable frame rate

A common complaint from VR users is what the virtual reality industry refers to as “simulation sickness”. This results from participating in virtual reality on a machine which cannot run the software at a high enough frame rate to avoid jitter or stuttering. The frame rate needed to avoid this with consistency, and what has become the standard for virtual reality hardware developer, is 90 frames per second. The HTC Vive comes as standard with 90Hz displays.

The HTC Vive for its minimum requirements asks for at least an Nvidia GeForce GTX970 graphics card, which alone costs upwards of £300 at the time of writing.

The reason for such high-end components being required is that the HTC Vive houses 2 displays, one for each eye. This means anything needing to be rendered must be rendered twice from slightly different positions because of the distance between the eyes, which is what gives the feeling of perspective. This is much more costly resource wise than simply having 1 screen doubled twice. Both screens must maintain the same frame rate of 90fps ideally.

The project was developed on a Nvidia GTX980 graphics card, which is slightly better than the minimum requirements for the HTC Vive, and is listed under the complete list of recommended graphics cards HTC provide [12].

This makes a key area of the project performance optimisation. Keeping the scene size in Unity as small as possible, with as few tricky things to render or collisions to calculate as possible to ensure this standard of 90fps is met, avoiding simulation sickness.

## Play space size

According to Valve themselves, 3 in 4 VR users have a play area large enough to support roomscale VR [13]. This leaves a sizeable number of users, about a quarter, whose play area does not support roomscale VR. These users have access “Standing Only” play areas, which is roughly 1 metre by 1 metre on average.

The play space the project was developed in is classified as a “Standing Only” play area. With such a percentage of players being in this situation, and the project ideally being available to everything, the project should at least be functional in a standing only play space.

# Objectives

* Consistent frame rate greater than 90 frames per second.
* Dangerous or difficult to recreate scenario.
  + Steam Train fireman
* Consistent interaction enforced through game design.
* The program must run on the HTC Vive.

## The Task

Development of a dangerous or difficult to re-create situation with a spin to the educational side, has led us to the player taking the role of a gamified steam engine fireman/stoker.

The most well-known role of this job is to shovel coal to fuel the engine. A natural cause of this is that temperatures will increase. This intuitively implies that the system will need to be cooled in some way. This implication is a logical step to the second task that will need to be completed, which is keeping the temperature to a minimum.

Interweaving these gameplay elements, with a score system in place encouraging interaction while running at a high and smooth frame-rate are the goals to be completed.

# Designing the Game

With the base elements of shovelling coal, and pouring water decided on, finding ways to make these elements interactive as possible, with an intuitive method of feedback needed to be found.

## Shoveling coal

There are 3 components to think about with regards to this activity. These are;

* Spawning of coal
* Interaction of moving coal into furnace
* Deletion of coal

Each of these components need to make sense in universe, progress the goal of interactive gameplay, and feel fun.

### Spawning of coal

With the spawning of individual coal rocks, there are few ways to make this inherently interesting to a player. This is a means to an end to allow the player to interact with the objects being spawned.

Something inherently enjoyable in the VR space is motion, due to physically having to move your eyes and head to follow objects. The way we will spawn coal to take advantage of this will be from a pipe placed on the ceiling of the room, meaning all coal spawned falls and crashes to the floor.

The spawning will start slowly. This allows the player to see where coal is coming from, and have time to mentally take note of this. However, as time progresses, coal will spawn faster and faster to a maximum amount, increasing the chaos and intensity in gameplay.

### Moving coal into furnace

The furnace the coal is to be moved into should be raised slightly so there is some effort required from the user, but the task isn’t too finicky or tricky. The shovel should be good relative size with the coal to allow finesse from the user, but not make interaction harder than necessary.

### Deletion of Coal

Once the coal is in the furnace, it needs to disappear somehow. A couple of methods are possible for this. One method is to despawn the coal after x seconds inside the furnace, however this is very interactive. Another option is to have the coal stay in the furnace but lose its physics properties and colliders so the user can see it build up in a satisfactory way. But this eventually would make it hard to see new coal entering the furnace, and has the same problems as the first with regards to lack of interactivity.

The method used takes some creative liberty with the real-world function of coal automatically being used, but adds that interactive element while still making sense in play. This is to add a crusher mechanism to the furnace, which the user must pull down to crush and destroy the goal, at this time the coal is then added to the system and destroyed.

This means coal is added in bulk now, which is a bit tricky for the user to control when they wish to add a certain amount. This can be seen as a skill within the game gained through play however, where the user eventually intuitively figures out how much coal is needed to add X amount to the coal counter before pulling the crusher, and potentially over-fuelling the engine.

## Cooling the engine

Like shoveling coal, this gameplay element has 3 components;

* Spawning of water
* Interaction of moving water to coolant pipe
* Deletion of water

## Environment and motion

Being on a moving train, it is important to give the player the feeling of motion and speed.

Since the user is situated inside the train, the train becomes his frame of reference.

Importantly, all that matters are that the user appears to be moving, not that they are actually moving. This means there are 2 possible options for giving this experience.

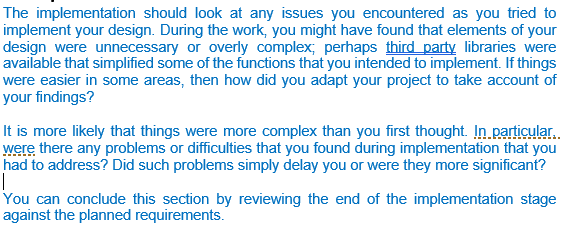
### Static environment

The first of these options is to have a static environment. This would entail having the train itself gain and lose speed as it moves through the x or z coordinate space.

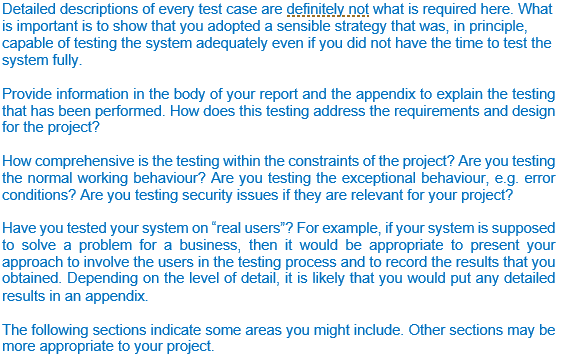
### Static train

The second option, and the one chosen for the project, was to have the train remain stationary at the centre of the scene (Vector 0,0,0). Instead we have the environment move backwards past the train, to give the illusion of movement.

# Implementation



# Testing



VR experiences, and Unity games in general are incredibly dynamic in their nature. This project does not feature input boxes which require string validation, or a server-side login system with security concerns. Instead the tests will take a black-box approach. This ensures the system functions for the user.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Case ID | Test Step | Expected Result | Observed Result | Pass/ Fail |
| 1 | Double click .exe | The game loads | The game loads | Pass |
| 2 | Turn on Vive controls, look at them with HMD | The controllers are visible in the game world | The controllers are visible in the game world | Pass |
| 3 | Press right grip button on right controller | Shovel appears in front of controller. | Shovel appears in front of controller. | Pass |
| 4 | Place controller inside shovel | Shovel is highlighted | Shovel is highlighted | Pass |

# Critical Evaluation

## Were the requirements correctly identified?

I found writing requirements for this project an incredibly difficult part of the process. My previous work prototyping with augmented reality in Unity during my Industrial Year was a nice introduction to the use of Unity, but not a lot more when it came to create a work such as this as a 1-man unit.

This left the requirements somewhat vague as after some experimentation, I was not sure exactly what the process or expected outcome of the project would be until I was already many weeks into development and had a better sense of the scale and size of individual tasks.

The requirements I did set however were a good outline for a higher-level view into what needed to be done, and kept an end goal in mind.

## How correct were the design decisions?

It’s fair to say the design from the start was vague when it came to the nitty gritty details of what would be achieved. This is due to the emergent nature of game design along with a lack of experience in the field. In the end I believe this to have been the correct decision, as meeting high-level goals such as “maintain high frames per second” and “give intuitive visual feedback” allowed the freedom to explore and experiment with what may be the best way to solve these tasks. With the result being a product that is a direct outcome of the agile process used in this project, I took full advantage of the task list to help update my estimates and gain a valuable insight into the scope of tasks as I was forced to size more and more.

## Could a more suitable set of tools have been chosen?

As mentioned in the Analysis section, there were other options for both hardware and software. Having now worked with these for a number of months, I have to give praise to Unity as a platform for development in particular. The engine makes many tasks that could be extremely tricky such as physics and rendering which need to be in place before you even begin creating a product. Building an executable file to run is made as easy as a single digit number of clicks and works perfectly. Its user interface is customizable and intuitive once you learn the way everything meshes together. While I have no doubt Unreal Engine 4 could have also been a good choice, I don’t believe it would have been substantially better, or even equal to that of Unity.

Between the start and end dates of the project, I got a chance to try the Oculus Rift, and while I was impressed with the ease of setup when compared to the HTC Vive, the use of the kit itself was very similar, with nothing apparent in my short playtime separating the experiences of the Vive and Rift besides the controllers used. The project would have handled near identically no matter the hardware choice, and so no more suitable choice could have been chosen.

## How well did the software meet the needs of those who were expecting to use it?

Upon demonstrating the project during the mid-project demonstration, feedback was very positive as to the fun and interactivity elements. I greatly enjoy playtesting the game as it reached the conclusion of the project, and feel it gives the experience in a manner that many can find value in. When I compare it to the inspirations of Valves “The Lab” and Owlchemy Labs “Job Simulator” I think it would fit into those packages well with some refinement and polish to the rough edges of its aesthetic design.

## How well were any other project aims achieved?

Coding in C# for the first time, I was worried there might be a learning curve to using the language, but very quickly I felt extremely comfortable writing in C#. It felt like I had been coding in it for a while. This innate familiarity due to many similarities with Java meant that code quality, specifically with regards to following Microsoft’s C# standards was no extra effort. The code even featured comprehensive XML documentation on top of this in an attempt to document as thoroughly as possible.

## What wasn’t achieved

The visual and audio side of the project is a mix on unfinished and untouched. While I’m very pleased with the environment design and sense of motion it gives, and the gauges are a neat intuitive feedback tool, there is more that needs to be done. There is some audio attacked to the coal spawner, but this is very unrefined, and it disabled in the final release. It can be turned on it unity by enabling the audio files attached to the objects as components, but for now they remain disabled due to detracting from the experience as the noises are garish and distracting. Also, the internal design of the train is functional but clearly not complete, with prototyping textures covering the ins and outs of the main play area.

## What would change if starting again?

Brainstorming a project which required less on the aesthetic side of game design, and more on an interesting coding challenge would play to my strengths more. While I enjoyed seeing that side of development, with the short span of time we have in the grand scheme of things, the experience is missing a lot of polish, and even glaring visual elements of the steam train such as the actual room you play in are completely unfurnished and still in an alpha phase of development.

A more code focused project would avoid this issue, while allowing more exploration into what’s possible in Unity, with the extra time not spent on environment / aesthetic design, a save/load system would have been implemented, or local storage of high scores.

# Appendices

## Third-Party Code and Libraries

### Unity [14]

Developer: Unity Technlologies   
Version: 2017.3.1f1 Personal(64bit)

Unity Personal is available for use for free to students in education. The following quote is taken from Unity’s education page [15].

*“Students who are using Unity outside the classroom to build or refine their skills are welcome to download the free and fully-featured Unity Personal.”*

Unity Personal also has a financial threshold after which use of Personal is no longer permitted. This would require an excess of $100,000 in the last 12 months [16].

### VRTK - Virtual Reality Toolkit [17]

Developer: TheStoneFox

VRTK is self-described as “A productive VR Toolkit for rapidly building VR solutions in Unity3d”. It includes a lot of basic models and interactions which are necessary to even begin developing an VR experience. Including but not limited to:

* Locomotion within virtual space
* Interaction
  + touching
  + grabbing
* Body physics
* 3D models for the HMD and controls

This allows us to focus more on creating an immersive experience, rather than being bogged down in the complexities of modelling interaction between the real-world controls and the digital one.

In this project, 2 gameobjects from VRTK example scenes are used to detect a HTC Vive is in use and load the correct associated prefabs and models with help from SteamVR. SteamVR is required to be installed for VRTK to work with the HTC Vive.

There are 2 sources for VRTK, the Unity Asset store, and the VRTK GitHub repository. The developer StoneFox, when asked which version to use, mentions in the FAQ for VRTK the following statement.

*“The*[*GitHub master version*](https://github.com/thestonefox/VRTK)*of VRTK is always the most up to date version with more features and bug fixes, however it is not as stable as the*[*GitHub releases*](https://github.com/thestonefox/VRTK/releases)*or the*[*Unity Asset Store version*](https://www.assetstore.unity3d.com/en/#!/content/64131)*. It is recommended that to keep up to date with the latest features, the*[*GitHub master version*](https://github.com/thestonefox/VRTK)*is used.”* [18]*.*

Due to the expert opinion of the developer, we decided to use the latest GitHub branch of VRTK available at the time.

VRTK is released under the MIT license [19].

The source code for VRTK was unchanged for this project. There is a piece of code which hooks into the VRTK event list to listen for Vive controller touches. Specifically, the “FixCoalSpawner” script for a custom listener event. Note this did not require directly modifying VRTK code.

### SteamVR [20]

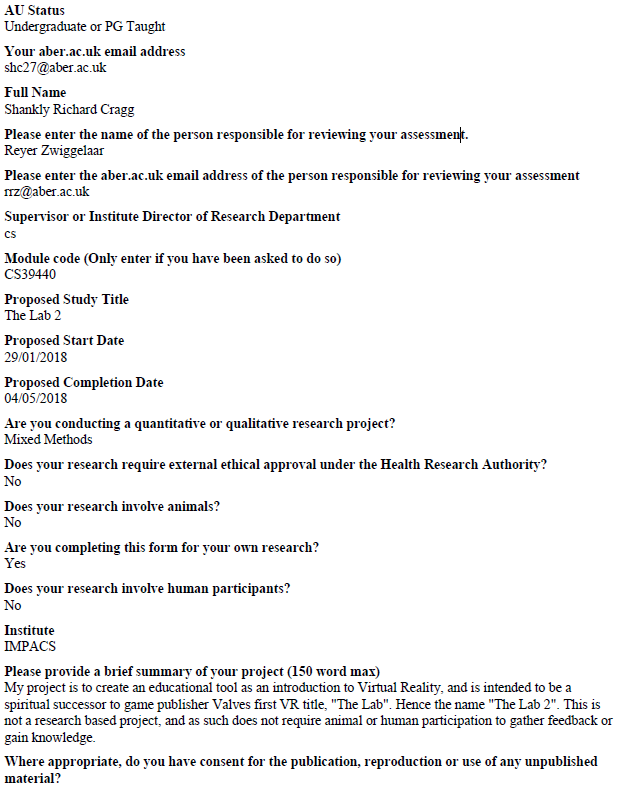
Developer: Valve Corporation

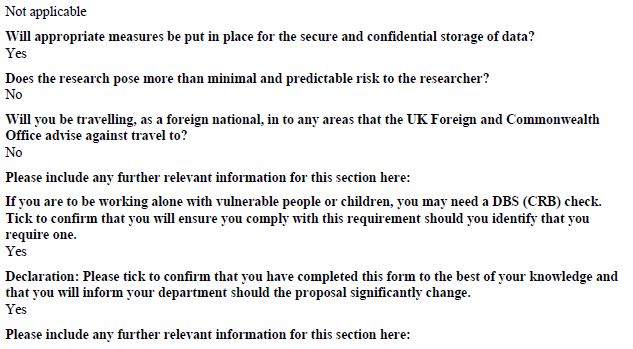
Unity does not natively support HTC Vive use. A plugin called SteamVR when imported into Unity will automatically configure Unity environment configurations specifically for HTC Vive development.

SteamVR is released under the BSD 3-Clause “New” or “Revised” License [21].

This plugin was used completed unedited. All use of SteamVR is done via VRTK.

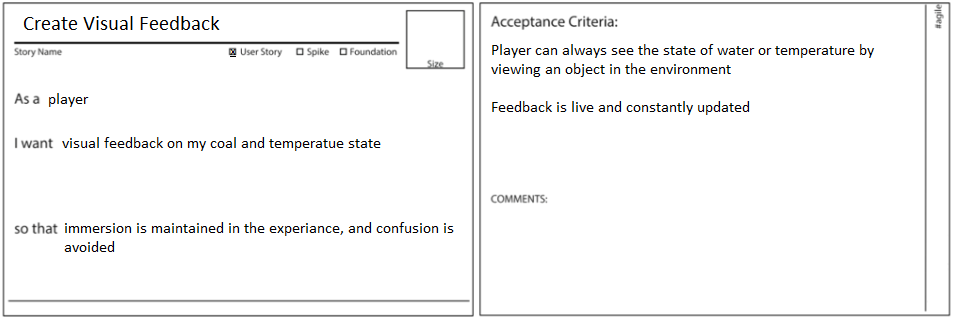
## Ethics Submission



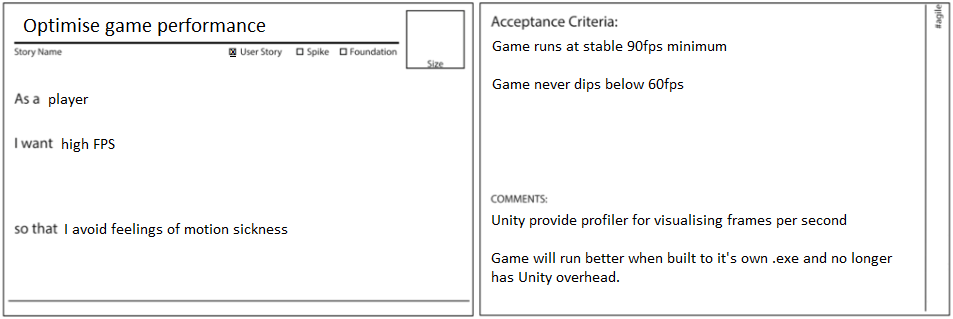


## Scrum User Story Examples

### Story 1



### Story 2



## Bibliography

|  |  |
| --- | --- |
| [1] | Statistica, “• Virtual reality market size worldwide 2016-2020 | Statistic,” 2016. [Online]. Available: https://www.statista.com/statistics/528779/virtual-reality-market-size-worldwide/. |
| [2] | Valve Corporation, “Steam Store,” Valve Corporation, 5 04 2016. [Online]. Available: https://store.steampowered.com/app/450390/The\_Lab/. [Accessed 24 04 2018]. |
| [3] | Owlchemy Labs, “Steam Store,” Valve Corporation, 5 04 2016. [Online]. Available: https://store.steampowered.com/app/448280/Job\_Simulator/. [Accessed 24 04 2018]. |
| [4] | SteamLocomotive.com, “Steam Locomotive Stokers,” [Online]. Available: http://www.steamlocomotive.com/appliances/stoker.php. |
| [5] | Valve Corporation, “Valve,” [Online]. Available: http://www.valvesoftware.com/. |
| [6] | A. Smith, “Hands On: The Lab, Valve’s Portal-Themed VR Games,” Rock Paper Shotgun, 2016. [Online]. Available: www.rockpapershotgun.com/2016/03/21/valve-the-lab-vive/. [Accessed 22 04 2018]. |
| [7] | Microsoft, “Microsoft HoloLens | The leader in mixed reality technology,” [Online]. Available: https://www.microsoft.com/en-gb/hololens. |
| [8] | Unity Technologies, “Unity,” [Online]. Available: https://unity3d.com/. |
| [9] | Oculus VR, “Home,” Oculus VR, 28 03 2016. [Online]. Available: https://www.oculus.com/. [Accessed 25 04 2018]. |
| [10] | Epic Games, “What is unreal engine 4,” [Online]. Available: https://www.unrealengine.com/en-US/what-is-unreal-engine-4. [Accessed 25 04 2018]. |
| [11] | Alex Andrews, “Scrum Of One: How to Bring Scrum into your One-Person Operation,” 2017. [Online]. Available: https://www.raywenderlich.com/162654/scrum-one-bring-scrum-one-person-operation. |
| [12] | HTC Corporation, “Vive Ready Computers,” [Online]. Available: https://www.vive.com/uk/ready/. [Accessed 26 04 2018]. |
| [13] | PersonGuy. [Online]. Available: https://steamcommunity.com/app/358720/discussions/0/350532536103514259/?ctp=2#c133258092253222557. [Accessed 26 04 2018]. |
| [14] | Unity Technologies, “Home,” [Online]. Available: https://unity3d.com/. [Accessed 25 04 2018]. |
| [15] | Unity Technologies, “Education,” [Online]. Available: https://store.unity.com/education. [Accessed 25 04 2018]. |
| [16] | Unity Technologies, “Unity - Unity Pro, Unity Plus and Unity Personal Software Additional Terms,” [Online]. Available: unity3d.com/legal/terms-of-service/software. [Accessed 25 04 2018]. |
| [17] | TheStoneFox, “VRTK Github,” [Online]. Available: https://github.com/thestonefox/VRTK. [Accessed 25 04 2018]. |
| [18] | TheStoneFox, “Frequently Asked Questions,” [Online]. Available: https://vrtoolkit.readme.io/docs/frequently-asked-questions#section-should-i-be-using-the-unity-asset-store-version-https-www-assetstore-unity3d-com-en-content-64131-or-the-github-master-version-https-github-com-thestonefox-vrtk-of-vrtk-. [Accessed 25 04 2018]. |
| [19] | TheStoneFox, “VRTK/LICENSE.md,” 30 08 2017. [Online]. Available: https://github.com/thestonefox/VRTK/blob/master/LICENSE.md. [Accessed 25 04 2018]. |
| [20] | ValveSoftware, “steamvr\_unity\_plugin,” 10 01 2017. [Online]. Available: https://github.com/ValveSoftware/steamvr\_unity\_plugin. [Accessed 25 04 2018]. |
| [21] | ValveSoftware, “LICENSE,” 12 01 2017. [Online]. Available: https://github.com/ValveSoftware/steamvr\_unity\_plugin/blob/master/LICENSE. [Accessed 25 04 2018]. |