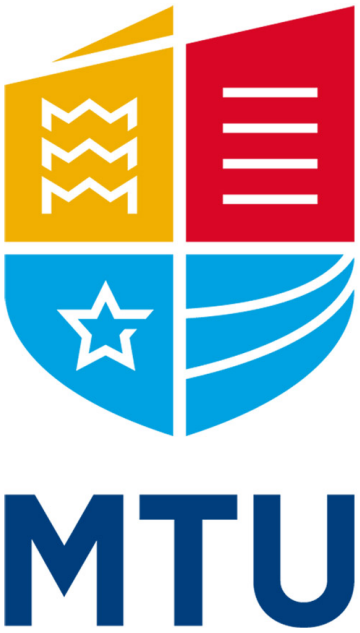


Virtual Reality Presence and its Effect on Interpretation of Simplified Visualization of Statistics and Data

by

Eoghan Spillane

This thesis has been submitted in partial fulfillment for the degree of Bachelor of Science in Software Development

in the
Faculty of Engineering and Science
Department of Computer Science

May 2022

Declaration of Authorship

I, Eoghan Spillane , declare that this thesis titled, 'Virtual Reality Presence and its Effect on Interpretation of Simplified Visualization of Statistics and Data' and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for an undergraduate degree at Munster Technological University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at Munster Technological University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this project report is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: Eoghan Spillane

Date: 4th of May, 2022

Munster Technological University

Abstract

Faculty of Engineering and Science
Department of Computer Science

Bachelor of Science

by Eoghan Spillane

Immersion in Virtual Reality and its effect on the understanding of simplified visualization of statistics and data. In this paper we will explore the dynamics of statistics in an immersive VR environment and how it affects the Approximate Number System and if it helps users gain an enhanced ability to comprehend statistics after their experience.

Acknowledgements

I thank my supervisors over both semester for their feedback and input into my project as it is now.

Another thank you to my lecturers, the department and the University who taught me the skills to learn and adapt that enabled me to complete this research along with the count-less creators of VR specific guides and learning material.

Oh... Also can't forget all the coffee I've consumed along the way...

Contents

List of Figures

List of Tables

Abbreviations

VR	**V**irtual **R**eality
HMD	**H**ead **M**ounted **D**isplay
AR	**A**rtificial **R**eality
DOF	**D**egree **O**f **F**reedom
PC	**P**ersonnel **C**omputer
SDK	**S**oftware **D**evelopment **K**it
CPU	**C**entral **P**rocessing **U**nit
GPU	**G**raphical **P**rocessing **U**nit
ANS	**A**proximate **N**umber **S**ystem

Thank you for reading this paper, I hope you find development in VR as exciting as I do...

Chapter 1

Introduction

This research project will focus on the use of virtual reality to statistics like percentages in a manner that is easier to digest and understand. The aim of this paper is to develop a virtual reality application that will create simple visualizations in a 3d immersive and interact-able environment in order to investigate its effect compared to contemporary and standard means of graphing when taking advantage of immersion and innate Approximate Number System (ANS) ability.

1.1 Motivation

Over the last few years, in a world where we are seeing more and more statistics in the public eye. There is more and more misunderstanding of said statistics over the last 2 years than ever before. Numbers like 7 in 1,000,000 (One Million) often get overblown and numbers like 3 in 100 get underestimated.

This general misunderstanding has led to misinformation as nonfactual opinions about the statistics are taken at face value rather than a person looking into comprehending the statistics further. This trend can cause actual harm and fuel the spread of sometimes purposely incorrect and misleading information.

With this project, we wanted to see if we could develop a virtual reality application that could inspire curiosity towards the statistics a person can be presented with and allow them to engage and gain a new understanding of it through a simplified form of visualization in an immersive environment to improve a person's ability to comprehend not yet encountered future statistics.

1.2 Contribution

This research project has been enabled by multiple modules completed over the course of my degree along with skills and experience gained from work placement.

- Designing a monitoring platform for Company wide Power BI usage.
- Spotting shortfalls in common statistical visualization techniques.
- Creating contemporary normalized data to visualize.
- Breaking down data sets into simple to represent data.
- Creating a clear design language.
- Representing data on traditional graphs.
- Project task planning along with agile methods of development.

1.3 Research Question

"How does a human presence in a large virtual space affect an individual's understanding of large statistics and enormous data outside of the human physical experience and does being presented with different, more relate-able and procedurally generated methods of visual data representation increase a person's ability to comprehend and understand these larger statistics in a safe environment through the use of Artificial Intelligence and virtual reality?"

Over the course of research it became clear that originally proposed research question needed to be retooled to become the following:

"How does presence in a large virtual space affect an individual's understanding, comprehension and engagement of large statistics and data when being presented with simpler, more relate-able and procedurally generated methods of visual data representation in a safe environment through the use of an immersive virtual reality application"

Changes were cutting the machine learning elements, re-defining the question in a more directed manner and focusing in on the virtual reality data visualization and it's more inherent traits that can be capitalized on while more considerate of potentially over-ambitious elements.

1.4 Structure of This Document

In Chapter 2 we will create a basis of understanding to build from as we move into Section 3, where we will be discussing the idea, theories and potential solutions to our posed research question. Section 4, where we will further develop on the solution proposed and Section 5 where we will implement the application. Furthermore we will then review, evaluate and finally draw conclusions in regards to the research question in Sections 6 & 7.

Chapter 2

Background

2.1 Thematic Area within Computer Science

Virtual reality provides a new and interesting way to explore data visualization. With an estimated 10 million units sold, devices like the Facebook/Meta Quest 2 are now an affordable and readily available new form of technology. [10]

This new availability allows for previously enterprise grade and business technologies to be available to the common user where simpler data visualization will be most effective.

This paper will mainly be focused on Human-Centered Computing. This is due to the core areas involving Information Visualisation and Human Computer Interaction.

2.2 Virtual Reality

Virtual Reality is a term that has been used and redefined multiple times over the last few decades, from its original inception in science fiction to its development as a real, useful and affordable technology that is available today.

Virtual reality technology has been slow to develop over the last 200 years, starting out as a simple device like the stereoscope which pioneered the use of lenses to view images from a handheld box, imaginations have been inspired to create more and more advanced forms as the general technology improved and got smaller.

While VR has been available in the enterprise industry for many years, the technology only began to get small enough and affordable. Modern consumer VR has mainly been spearheaded by two very known individuals in the tech and gaming industries,

John Carmack and Gabe Newell. John Carmack was the co-founder of Id Software and ushered in the era of 3d games with Wolfenstein 3D, Doom and Quake. Whereas Gabe Newell worked on the first iterations of windows and pushed for and developed a port of Doom that worked on windows, proving windows as a suitable environment to not only run games, but that games didn't need to be confined to consoles after which Gabe Newell founded Valve and was instrumental in creating boundary pushing games such as Half-Life and Steam

In 2013 John Carmack joined Oculus as CTO and pushed the technology and methods devised by Palmer Lucky to release development kits of the DK1 and DK2, devices that sparked the VR industry into life despite them only tracking head movement on release. This culminated with the release of the Oculus Rift in 2016 which used a "Constellation" tracker to gain sub millimeter accuracy in all degrees of movement. This allowed for a player to stand up and walk around a virtual room for the first time.

Similarly in 2016, the HTC Vive released, being co-developed between HTC and Valve [11] with Valve working on most of the original prototypes and testing different methods of tracking. This headset also allowed for movement in all directions and worked directly with SteamVr, a platform developed independently of Oculus at the time.

These two companies ushered in the age of consumer available VR with 6 degrees of movement. While Alphabet did work on a concept of the Google Cardboard this technology was soon dropped by consumers due to its limited usability and lack of 6 degrees of freedom tracking. It did beat all previous headsets such as the original Oculus DK1 and multiple enterprise and research headsets while just using a regular smartphone, some cardboard and some mass produced plastic lenses. This allowed nearly all owners of a smartphone to at least experience simple things like roller-coasters or viewing 360 degree pictures and video.

2.2.1 Head Mounted Displays

A HMD or head mounted display is the umbrella term for all forms of technology involving image creation or projection that moves with a person's head. These devices are light and started off as simple devices like google glass. These had no tracking but allowed users to read a screen with information at all times. While limited in use, they saw small consumer and DIY ventures over the years.

Commonly now a HMD refers to a device such as the Valve Index or the Oculus Rift. These devices feature high resolution displays with a high refresh rate. The display type also significantly changes the experience produced by a HMD. Factors such pixel layout and PPI (Pixels Per Inch) also prove to be very large factors in the overall fidelity of the display. With the advent of micro OLED (Organic Light Emitting Diodes), HMD resolution is set to take another jump forward in vibrancy and readability along with the reduction of the size, weight and a large increase in comfort for the headset.

2.2.2 Degrees of Freedom

Degrees of Freedom are an important part of VR, it's the technology and feature set that distances HMD technology from just placing a screen in front of a user's eyes.

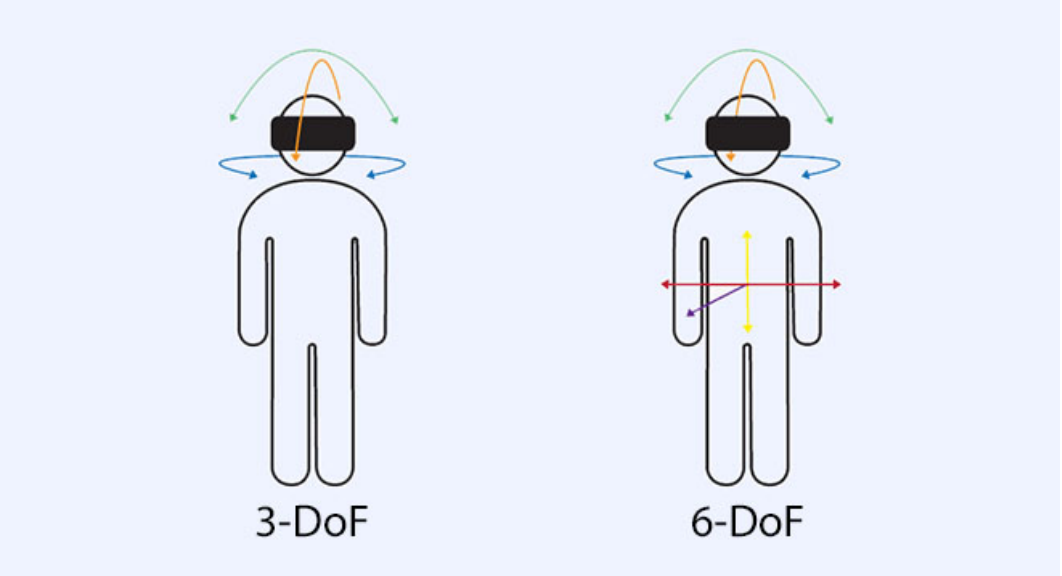


FIGURE 2.1: Degrees of Freedom[1]

2.2.2.1 3 Degrees of Freedom:

This refers to being able to look around from a stationary position. The headset is aware of the three axis angles your head can turn, It tracks these details and passes it back to the computer running the program allowing a user to look around. This includes being able to look up, down, left and right along with tracking the angle of the head.

This type of tracking is great for seated experiences. It works by using a set of gyroscopic sensors. Headsets like Google Cardboard[12] used a smartphone as both the display and it made use of the built in gyroscopic sensors to track movement across these axes.

2.2.2.2 6 Degrees of Freedom:

This refers to the ability to also move in 3d space. It does however require many additional sensors. On early HTC Vive prototypes by Valve this meant the use of a large room covered in qr codes. These were then read by on board cameras to compute an XYZ axis position within the room. This was later replaced with Lighthouse tracking which makes use of specialized sensors to detect the laser scans from the lighthouse and use information such as angle and time to provide sub millimetric XYZ axis positioning. Early Oculus 6 Dof headsets used cameras and IR (Infrared Light) emitters to give the headset positional knowledge but 360 tracking only came with lighthouse tracked headsets at first. Later Oculus headsets and other companies have moved towards camera based inside-out tracking. This means the headset has to find its own points of reference in the room. On headsets such as the Oculus Rift S or the Oculus/Meta Quest 1 and Quest 2, the device used cameras and machine vision to understand its environment and gather enough data to track its position relative to the room on the XYZ axis. This method of track is also used by Windows Mixed Reality (WMR) headsets like the Lenovo Explorer, Samsung Odyssey or HP Reverb G2. Inside out track is liked in the consumer space due to the ease of just putting on the headset, nothing needs to be set up in the room unlike Lighthouse tracked headsets which require base stations to be mounted to the walls.

2.2.3 Spatially Tracked Controllers

While 3 Dof headsets were often controller-less they sometimes used a basic 1 clicker style hand controller or an Xbox controller to control the game or program. When 6 Dof headsets were introduced they were introduced along with controllers that were aware of the positioning in 3D space, this allowed someone wearing a 6 Dof headset to bend over and interact with objects in front of them, either picking them up or moving them about as if they were after grabbing a physical object in the room.

This type of interaction allows for more immersion and can allow a user to explore their curiosities about the virtual world they are in.[13] This added a much needed layer of interact-ability and allowed games within VR to begin to take off. Having controllers tracked in 3d space also allowed for a boom in scientific and business research into the use of headsets for productivity, training and research.

2.2.4 Hand Tracking

While they are great for interaction with virtual objects, enterprise grade technology has shifted to developing hand tracking technologies.

This can be using a specialized high accuracy device like an Ultra Leap Leap Motion. A device designed to keep track of hands accurately as they open, close, mesh fingers together and cross over each other. This allows for controls to get mapped to gestures, for example palm up for a menu button.

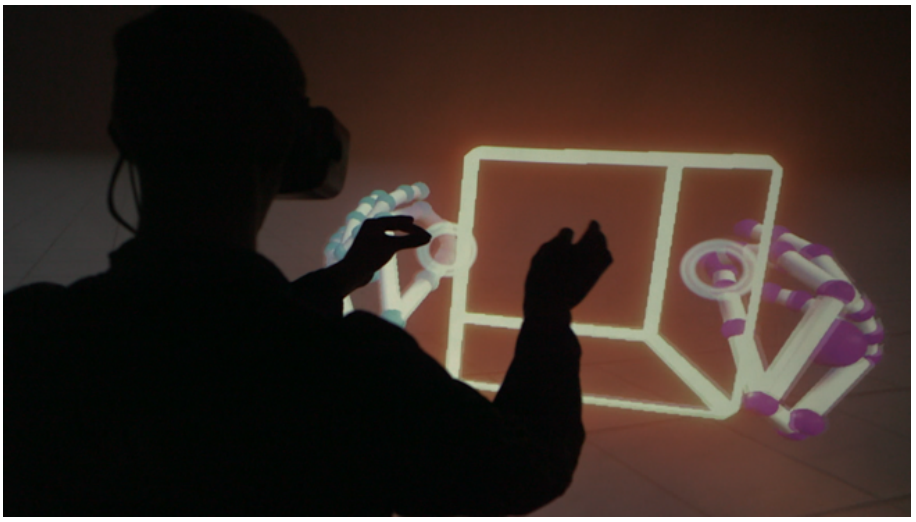


FIGURE 2.2: Using Leap Motion hand tracking in VR[2]

Another implementation can be found on the Meta/Facebook Quest 2, this uses the in-fared cameras normally just used for positional tracking to follow the hands, while flawed as implementation is using less than ideal hardware it still works really well for menu interactions when lighting conditions are favorable and works as a proof of concept that having both options available to the user and a secondary way to interact with the headset and the virtual world. A good use of this is being able to watch a film in VR without having to use the controllers to play and pause.

Hand tracking also removes the initial complexity of VR for many users as they don't need to use a controller to interact with the virtual world. This makes it very useful for running studies across multiple age groups in which controller usage could prove to be an issue.

2.2.5 Eye and Facial Tracking

Eye and Facial Tracking, while less relevant to the current generation of HMD technology, they are promising ways to gain more information from a user. This includes using eye tracking to check where users are looking when in a virtual space. It can track every eye flick allowing for a researcher to see exactly what attracts a users eyes first, even on a subconscious level, this type of data collection is invaluable in judging new virtual environments, especially ones designed to challenge users in some way.[14]

Facial tracking while relatively new, can be combined with eye tracking to collect emotion data and expressions/reactions to stimuli. It can also be used for both business and social VR platforms in order to make the user's avatar more lifelike and expressive.

2.2.6 Immersion

In general when using a virtual reality headset, immersion or virtual presence varies based on the technology used, the program and then the user themselves. Even if all conditions are correct some users will inherently never be immersed in VR whereas other users will be overly immersed depending on the stimuli they are provided[15]. Each user can react differently and behave differently, in some case enough to trigger fight or fight response in users.[16]

A key point in many studies is that graphics in VR are less important, many users will find themselves immersed in worlds with lower quality textures and visuals if all elements of the scene are immersive with elements like lighting, sound design and object physics playing a big part in this.

2.2.7 Game Engines

Most VR applications are now developed in one of two engines. They are either built using Unity, or they're built using Unreal Engine. Both of these engines are fully featured and have numerous plugins to enable easier development. Both have many tutorials available and the engines are designed to run on many devices and are well proven. Both Unity and Unreal are both performant and they allow developers to build nearly anything. While not standard for research projects, both Unity and Unreal engine are well documented and there are a great many tutorials online to aid in the development.

Unreal Engine 5 is extremely interesting lately due to the development of Nanite[3], a system in which millions of polygons can be used onscreen for minimal performance impact, allowing for higher detail models and even stones on the ground to be highly detailed.

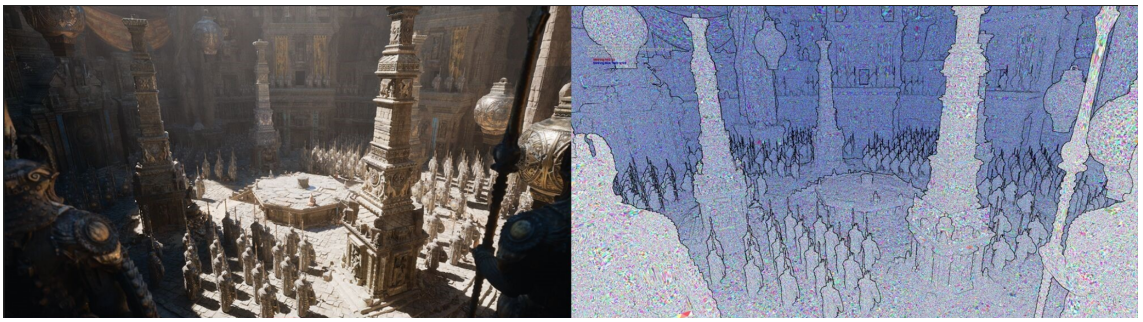


FIGURE 2.3: Unreal Engine using Nanite for high poly environments[3]

2.2.8 Open XR

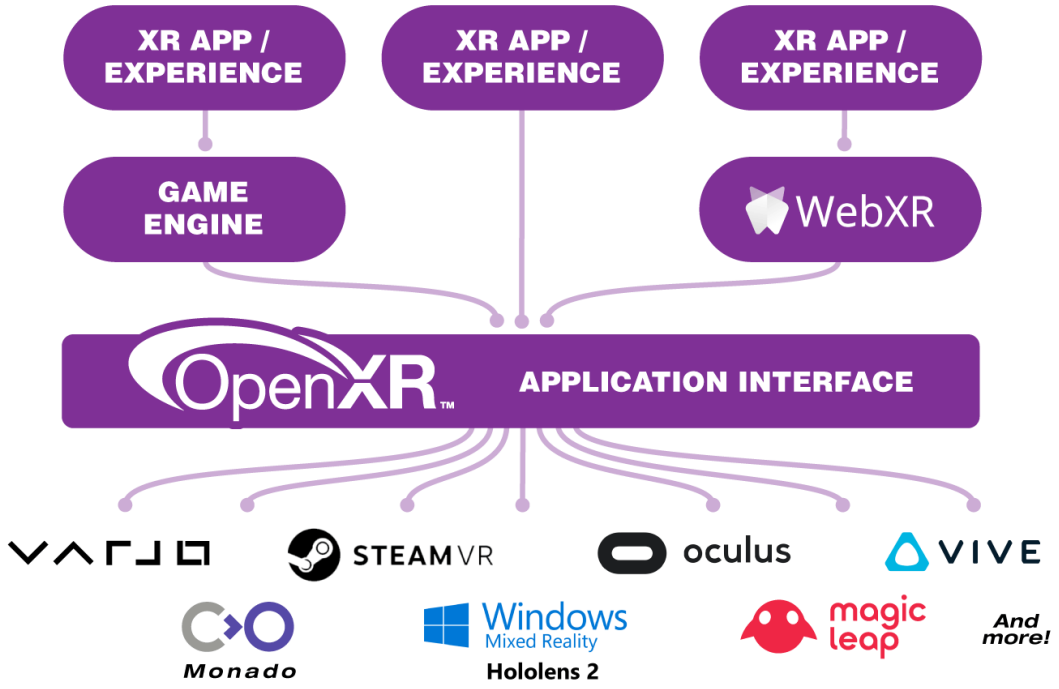


FIGURE 2.4: With OpenXR[4]

This is a relatively new change but it's a framework programs can be built around to ensure compatibility with a wide range of headsets. In previous games, each headset and controller type that could be used needed to be accounted for. However if your game implements the Open XR layer then your program will work by default with all current and any headsets released in the future. It provides a common way to communicate with the headset and does away with proprietary software from the headset manufacturer. OpenXR was endorsed by both Valve and Oculus for all of their headsets going forward , encouraging all developers to switch over to using it.

2.3 Artificial Reality

Artificial Reality, Augmented Reality and what's sometimes referred to as Mixed Reality is a branch of Virtual Reality and it uses very similar technologies, the key differences are found in the implementation of the applications.

The key difference between AR and VR is that with AR the physical environment around you is part of the program. Mobile base examples of AR technologies are Pokemon Go, Google Measure. Both programs introduce virtual elements to the real world that can be viewed using a mobile device. They use the camera to pass the images through to the application where machine learning and AI based depth and visual processing is used to convert this to a Mixed Reality experience by smartly adding digital elements to the screen that are mapped to the physical world allowing a user to walk around the virtual object with their device.

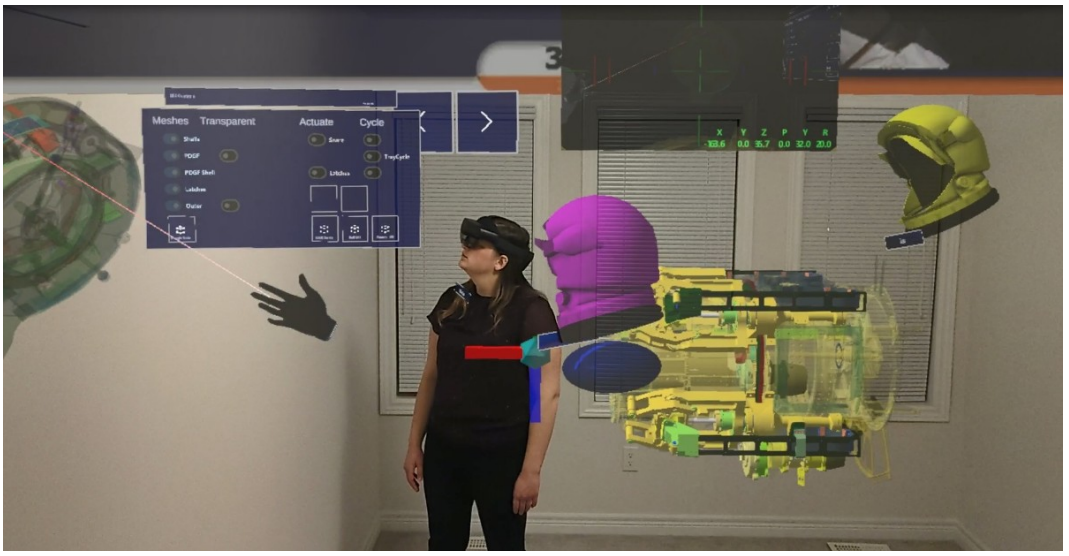


FIGURE 2.5: Augmented Reality in use with Microsoft's Hololens[5]

Mixed Reality devices such as Microsoft's Hololens[17] have been used in many applications alongside devices such as Leap Motion, a technology used to use a user's hands accurately as controllers. This allows for a lightweight HMD that can have accurate and intelligently designed interactions without dedicated controllers needing to be carried or for a more standard touch input device like a screen or single button clicker being needed.

2.4 Data Visualization

Data Visualization can be broken down into three key aspects for this paper,

- Standard 2d Graphics
- Standard 3d Graphics
- 2d vs 3d Traditional Graphics currently in VR.

2.4.1 2d Graphics

2D graphs are very common and have had many interactions over the years with the most common types of 2d graphs being Bar and Line charts. They are usually used for representation of changes to a value over a set variable

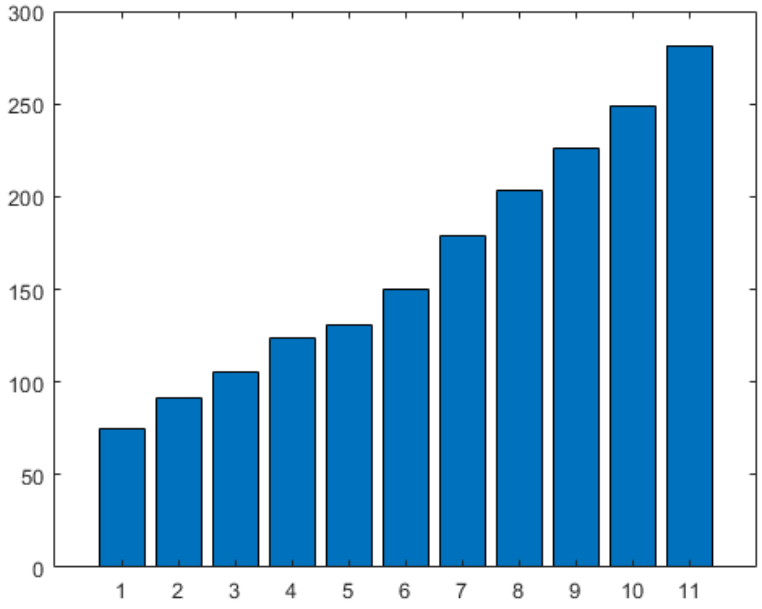


FIGURE 2.6: Basic Bar Chart[6]

Bar and their direct sibling, line charts are staples of the data visualization space and are probably one of the most common statistical visual aides in use today. They can be used to represent 2 axes of data, the X and Y.

2.4.2 3d Graphics

These bar charts can then be used to represent a third axis by adding a 3rd dimension. This can both improve the quality of the data and allow it to represent more complex data, however these types of graphs can become cluttered and confusing fast. This along with a 2 dimensional mouse input, viewing these types of graphs can be difficult lead to misunderstandings, especially if the graph isn't constructed right to begin with.

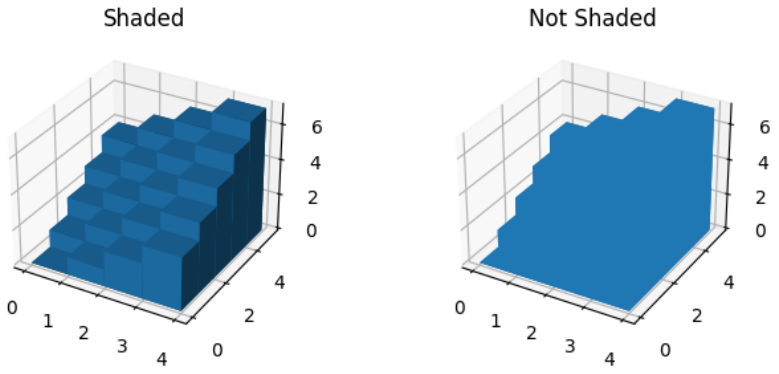


FIGURE 2.7: Basic 3 dimensional Bar Chart[7]

While 3d graphs are limited to one angle in 2d visualizations, in VR however they can be viewed in full 360 degrees, allowing a user to walk around them and observe them from multiple angles, lean in and get a better look[18]. This type of interaction increases a user's level of curiosity and the added intractability makes it easy for a user to manipulate the chart and view the data making them better and more effective visual aids.

2.4.3 2d vs 3d Traditional Graphics currently in VR

While 2d Graphs may be useful in VR, it's the place where 3d Graphs truly shine with many programs, plugins and tool-kits being built for it including ImAxes(Immersive Axes) a system designed for "Exploring multivariate data using fluid, mode-less interaction"[19]. IATK (Immersive Analytic Tool-Kit)[20] a plugin for Unity that allows users to create their own 3d graphics through a virtual GUI.

Telerik is a Unity3D VR based data visualizer[8]. It focuses on 3d bar charts and scatter plots along with setting up multiple 2d versions of these tables in special arrangements to help make understanding the data easier for the user.

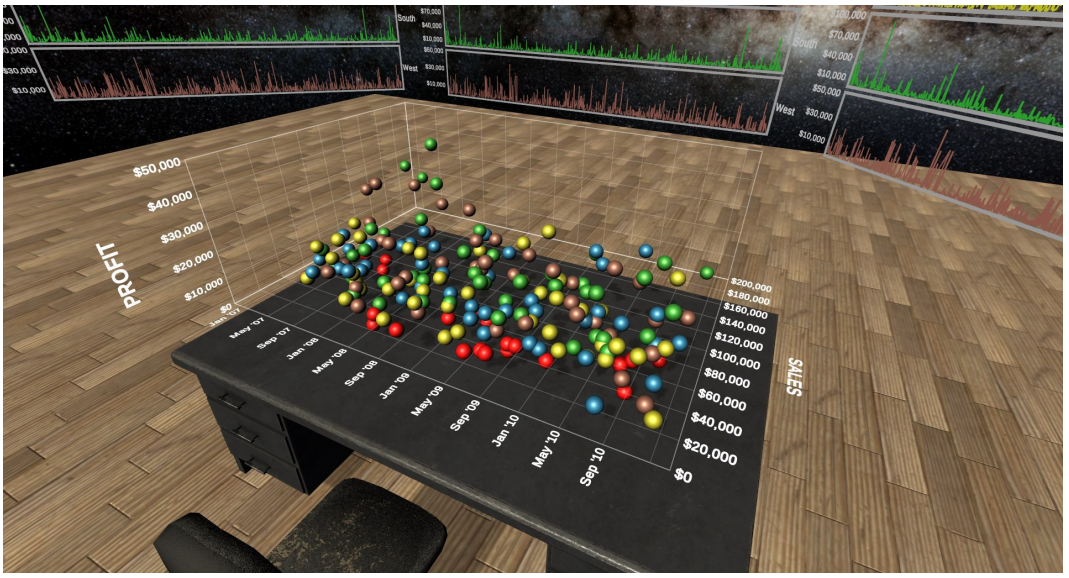


FIGURE 2.8: Telerik Data Vizualizer[8]

2.4.4 Approximate Number System

The Approximate Number System or ANS is a persons ability to look at a group of objects and immediately approximate how many objects are there before counting can begin[21]. This gives the person the ability to make assumptions based on visual evidence as to how many of a thing is in front of them. This usually falls in the region of assumptions that lands with common language.

For example, common phrases like, "A few", "Several" and "loads" often correspond to immediate approximations about the amount of the thing being counted and further enforce the estimate by not typing a number to it.

Chapter 3

Virtual Reality Presence and simplified visualization of Statistics

3.1 Problem Definition

In current times statistics have made their way more and more into daily life. They can be found everywhere, either on the radio, on the news, online and on social media.

A problem we have discovered over the past year is that when people are shown statistics, the statistics are often accompanied by an opinion, or a charged series of events when used to back up potentially political or social issues. Unfortunately in these scenarios and more the actual meaning behind the statistic is lost and comprehension of the numeracy involved is often a secondary factor.

We feel this behavior may be brought about due to the quick question, quick answers era we are now in, in which the answer to any question, no matter how complicated, is a single web search away relying on the person searching to use their own judgment when finding the correct answer. This leads to a person finding potential answers that have confusing messaging surrounding them. In regards to statistics, a person can often include the comment section as a potential answer in how to comprehend and interpret the statistic used.

This need for an quick answer has been seen more and more often across the internet and media as many may not try to comprehend the data and potentially not process their own opinions on the matter before them, leading them to instead find another persons

interpretations of the data, regardless of the quality or reputation of the source of said solution/opinion.

Using virtual reality to provide a stimulating, immersive yet also isolating experience is it possible to encourage a person to interact with a simplified data visualization tool and improve their ability to draw their own conclusions rather than out-sourcing their statistical comprehension and interpretation to someone else, and further gain the ability to more accurately and quickly process statistics using their ANS abilities.

3.2 Objectives

The planned objectives of this research paper are to create an application that can address the above problem definition in a way that is

- Easy to pick up.
- Immersive and Informative.
- Provides simplified visualizations.
- Statistics can be customized within virtual reality.

Since this project is aimed at a wide audience, ease of use, along with a minimal learning curve is an important objective to aim for while creating an immersive environment for a novice VR user to experience and interact with.

3.3 Solution Requirements

3.3.1 Functional Requirements

The application has 5 functional requirements that it needs to meet in order to fulfill the objectives outlined above.

1. 3D virtual sandbox environment.
2. Host vast quantities of 3d objects.
3. Dynamically arrange 3d objects based on user input
4. User movement, interaction and immersion.
5. Support Open XR in order to work with all current and future VR HMDs

A 3D virtual sandbox is a virtual space that will host every object needed to display the statistics required by the user. This space must feature large open areas along with a view platform for a birds eye view of the data visualization.

User movement, interaction and 6 Degrees of Freedom will allow a user to walk around and inspect the objects that are representing the data. They will be able to look around different rows of objects that represent the data. This will allow them to be up close with the data and gain a greater sense of scale.

3.3.2 Non-Functional Requirements

Immersion - The solution should be immersive. It should be able to hold the attention of the user and provide enough interaction and visual stimuli so that the application doesn't become a chore to the user

Visualization - The solution should be able to generate visualizations controlled by the user. These should be customized and be performant enough that the application can be used to its fullest potential and display massive statistics.

Simplicity - The solutions should be easy to use, catering to users of varying ages and skill levels. It should include a basic tutorial to ensure users are aware of basic controls.

Chapter 4

Implementation Approach

In this chapter, we're going to discuss what needs to happen in order to get this solution off the ground.

4.1 Solution Architecture

This project will be built using the Unity Engine. It has a large abundance of resources available including tutorials, developer guides and large troubleshooting forums.

Unity is primarily a game engine but has seen many uses in research due to its highly versatile nature, it has the ability to run on many platforms which will allow for potential porting to the ever growing Android based stand alone headset market. Unity is based on C Sharp which is a very well documented language, used across industry.

The OpenXR API and Unity will allow the project to be portable to current standalone headsets such as the Facebook/Meta Quest 2, Vive Focus 3 and hopefully any upcoming standalone headsets based on the Qualcomm Snapdragon XR2 platform.

The technology being used in this project will be the newest and most inter-operable systems and technology currently available in order to keep the solution compatible with the fast paced and exploding VR development scene.

4.1.1 Architecture Diagram

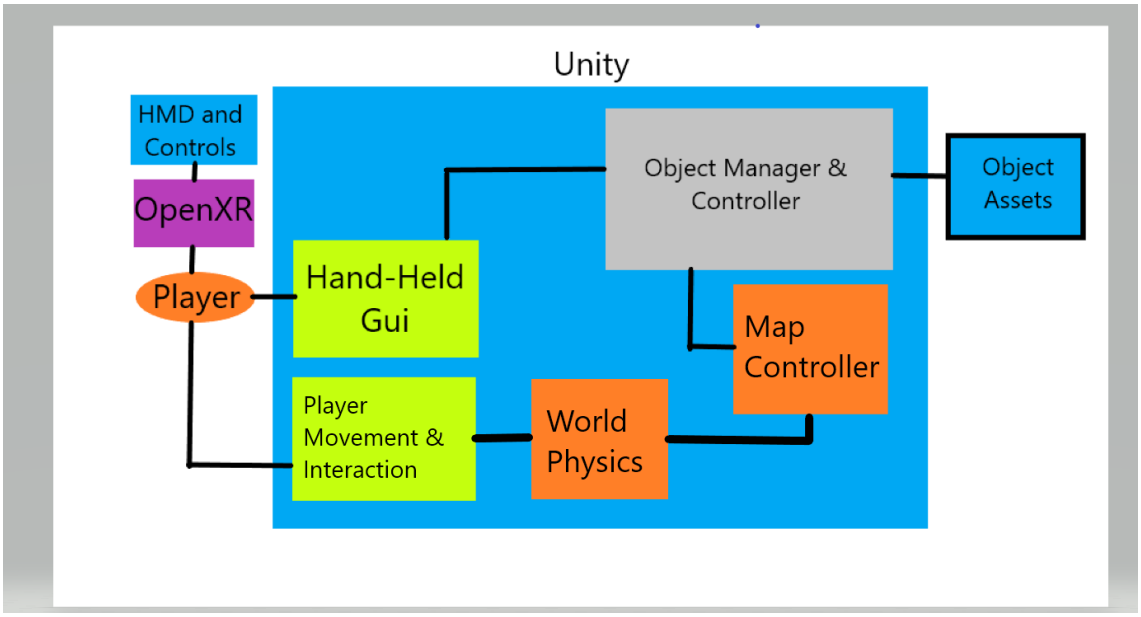


FIGURE 4.1: Solution Architecture

While development has yet to begin, apart from using OpenXR to interface with the HMD hardware, nearly all functionality of the program will be retained inside of Unity itself.

4.2 Use Cases

In order to meet the functional requirements a user must be able to:

- **Spawn Object:** The player should be able to spawn different objects in different ratios.
- **Interaction:** The player should be able to interact with each type of object in a rewarding manner.
- **Player Movement:** The player should be able to loco-mote in various manners suitable for different levels of VR experience in order to prevent motion sickness.

Use Case Name	Spawn Objects
Scenario:	The player wants to visualize a statistic from a giants perspective
Actors:	Player, Object Controller, GUI, Player Controller
Flow of Events	Player: Opens Gui Gui: Opens Player: Selects the statistic ratio and object type to spawn Gui: Passes information to Object Controller Object Controller: Spawns Objects Player: Selects large player size Gui: Passes information to player controller Player Controller: Changes player scale
Failure States	• GUI fails to open. • Objects don't spawn. • Player scale is unchanged.

4.3 Risk Assessment

Potential Risk involved in this project.

1. **C-Sharp Language Knowledge:** C-Sharp is a very well documented industry used programming language. However apart from C, Java and Python we have had limited exposure to this language which may slow down initial development.
2. **Unity Knowledge:** Current basic experience for 2d should help with the initial development of this solution but due to the complexity involved with developing a VR application this may also slow down implementation.
3. **Virtual Reality - New and changing Technologies:** VR technologies are constantly changing at the whims of larger companies, API's are being introduced and deprecated on a yearly basis as the industry finds it's legs and develops standards

4.4 Methodology

In order to complete this project on time it will be important to approach the project carefully using well established techniques.

4.4.1 Kan-Ban Boards

Kan-ban boards are key to managing time and project tasks and goals. By using a clearly and accurately laid out backlog it should be possible to implement the project in a step by step manner until the project is complete.

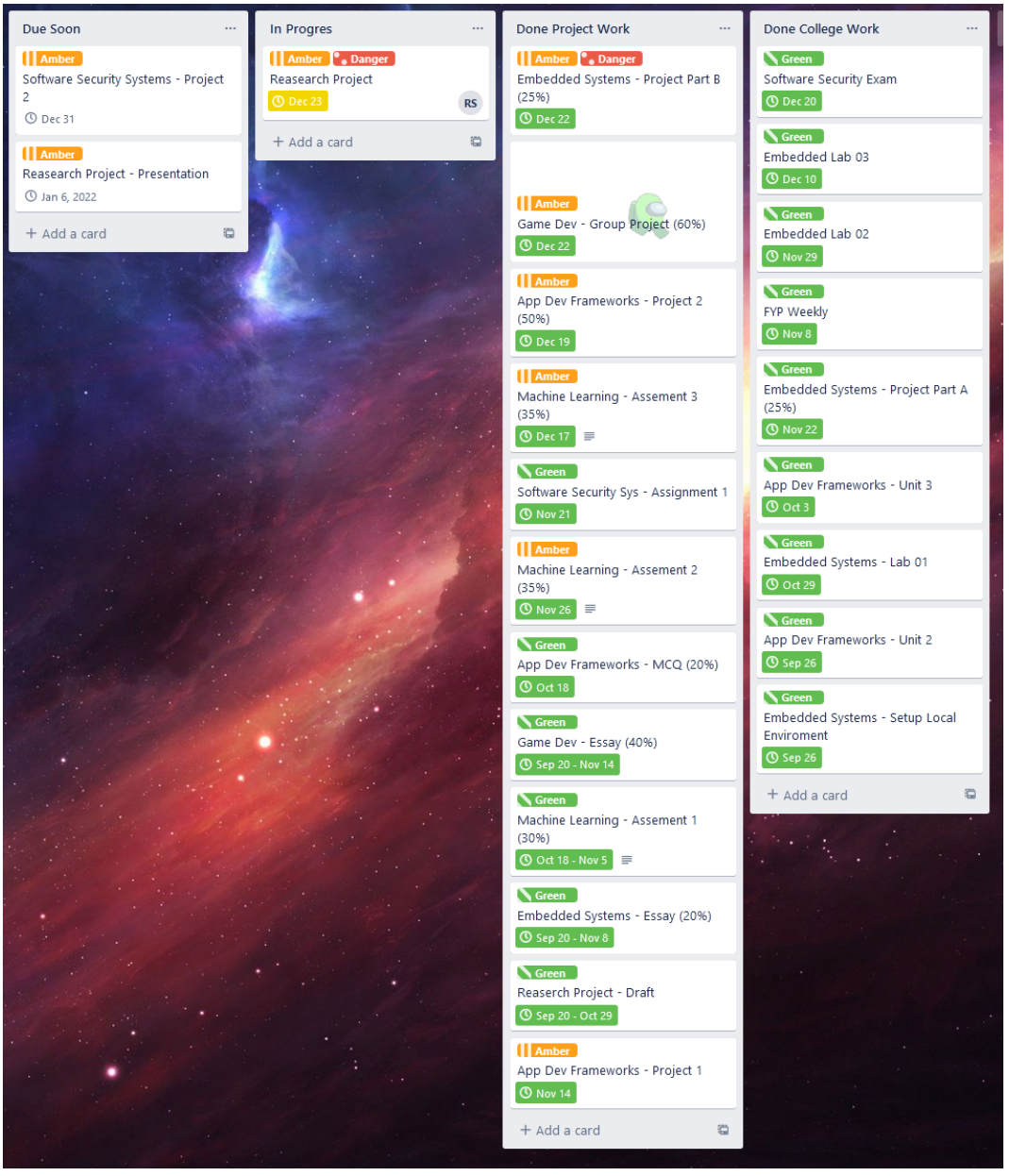


FIGURE 4.2: Kan-ban Board Example

4.4.2 Naming and Code Convention

Due to how Unity works, if files or functions names need to change later in the project this can cause massive issues and setbacks setting a clear and defined naming convention from the start will prevent any unnecessary headaches. Using camelCase will increase readability along with clear commenting should make the project easier to understand as it expands to full functionality.

4.5 Implementation Plan Schedule

The current plan to develop the application.

1. **January:** Learn the in's and out's of developing a basic VR application in Unity along with C-Sharp and basic blender modeling skills.
2. **February:** Begin Prototyping Object Spawner & Controller along with basic GUI Elements.
3. **March:** Add Extra Features and Polish, adding technical details to the Research Paper.
4. **April:** Trail Run program to try and answer the research question.
5. **May:** Finish Touches and Submit Project

4.6 Evaluation

In order to evaluate the project it will have to fulfill all of the requirements below while still remaining perform-ant.

- Easy for non-VR users to pick up.
- Feature a simple and clear user interface.
- Have multiple ways to interact with spawned objects.
- Adaptable spawning system to meet multiple statistics.
- Run on different hardware for future and backwards compatibility.

4.7 Prototype and Mock up

While the solution is yet to be implemented we have a clear vision of how the solution will function at a core level.

4.7.1 Menus

Menus in the solution will be primarily wrist and hand mounted. Using an offhand pointer system. It will feature one menu on each hand that will control most of the in-solution interactions. Wrist and controller mounted interfaced and tried and tested in nearly all VR applications. It will also add potential for hand tracking support in later Headsets.

The solution will also feature a menu on launch that will allow the player to set their height and start with some preset statistics and visualizations. The menu itself will feature multiple options for assets.

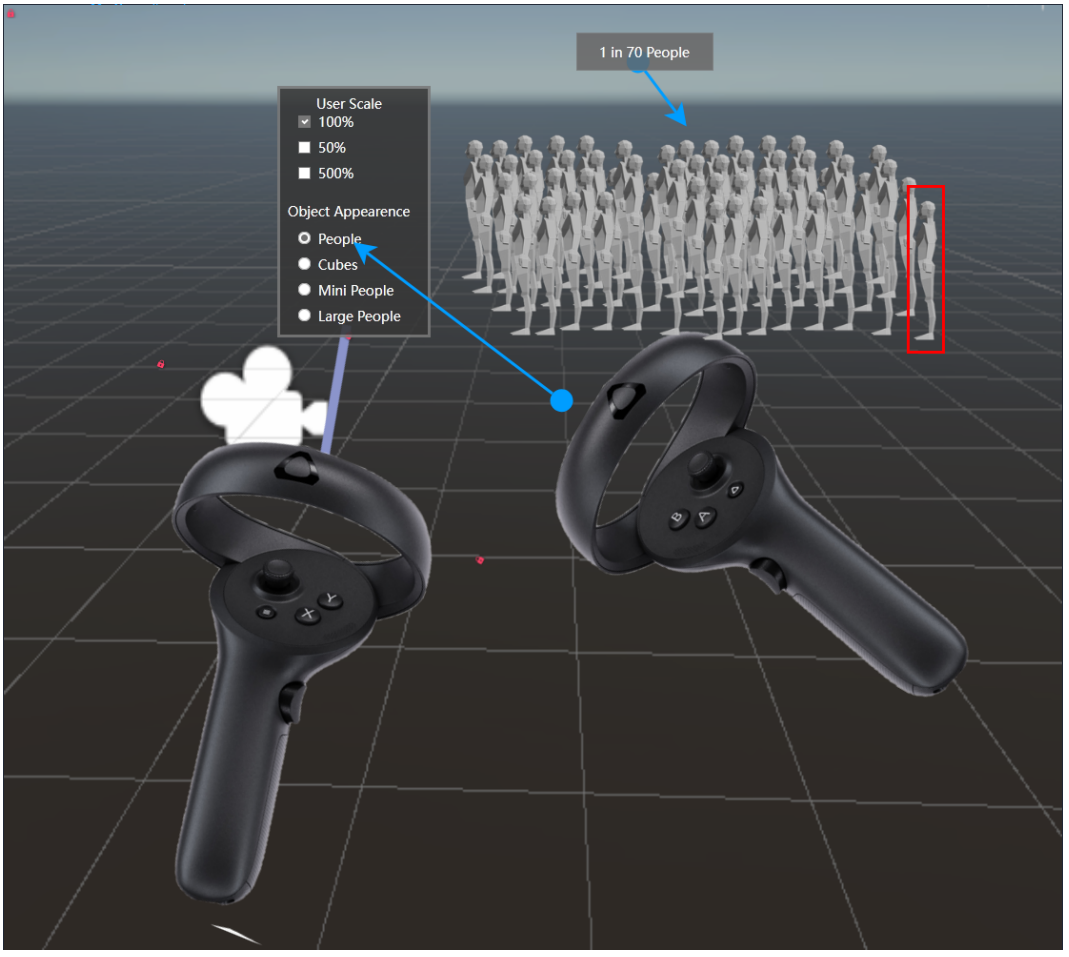


FIGURE 4.3: Application Basic Mock-up of menus and player view

4.7.2 Interaction

The player will be able to scale themselves to view the in-solution object and interact with them. They should also be able to interact with objects to set up small, fun simulations and dynamic moments. This will include having colors spread between close objects and letting objects interact with each other in explosive ways.

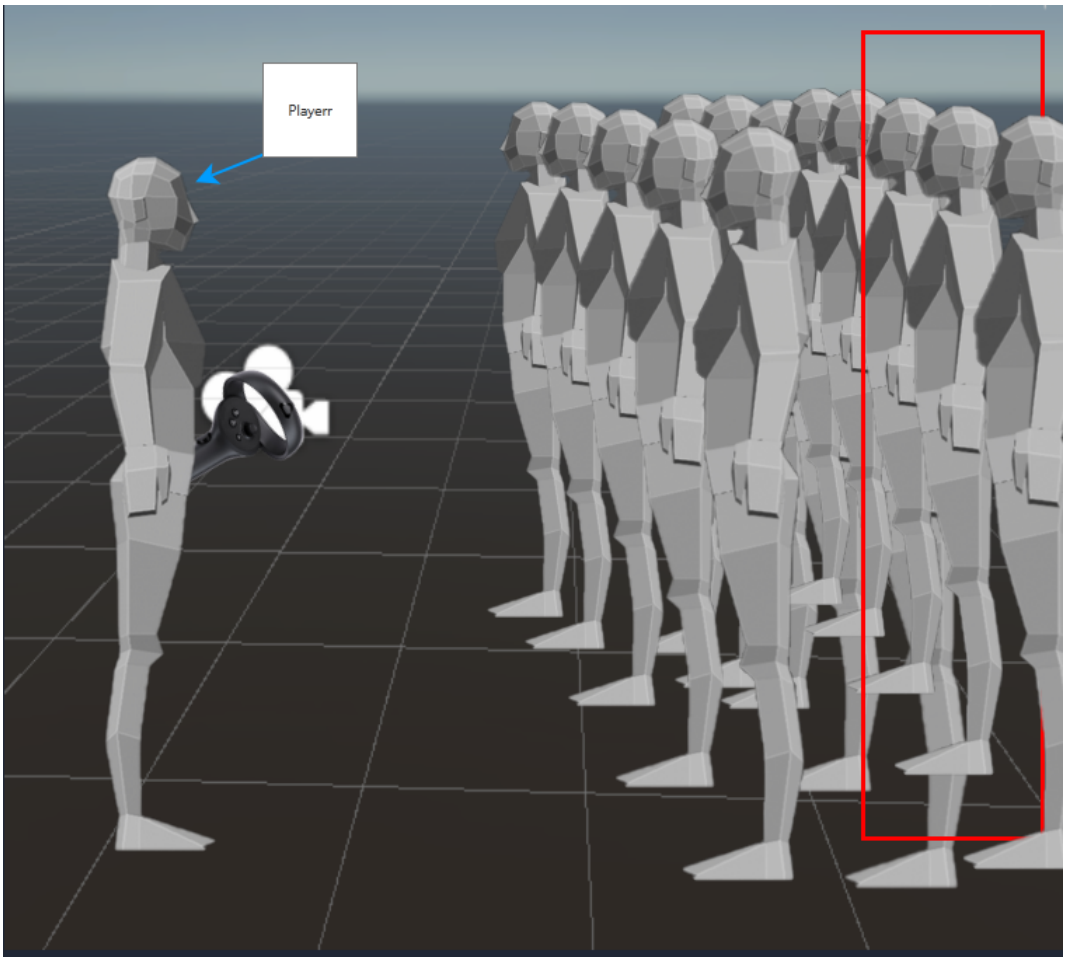


FIGURE 4.4: Application Basic Mock-up When Scaled to same size as objects

Chapter 5

Implementation

In this chapter we will discuss the approach to implementation planning, difficulties encountered, if the implementation varied from the implementation approach in Chapter 4 and then the actual development process.

5.1 Project Development

During development the Kan-Ban board became very structured. Due to the reactive nature of development, the project became an example of Empirical Process Control as not all tasks could be planned out at the beginning with many being added after certain milestones of functionality were reached, this allowed us to scale up and scale down the remaining features depending on the time remaining to complete the project and how many difficulties were encountered along the development timeline.

5.1.1 Project Schedule

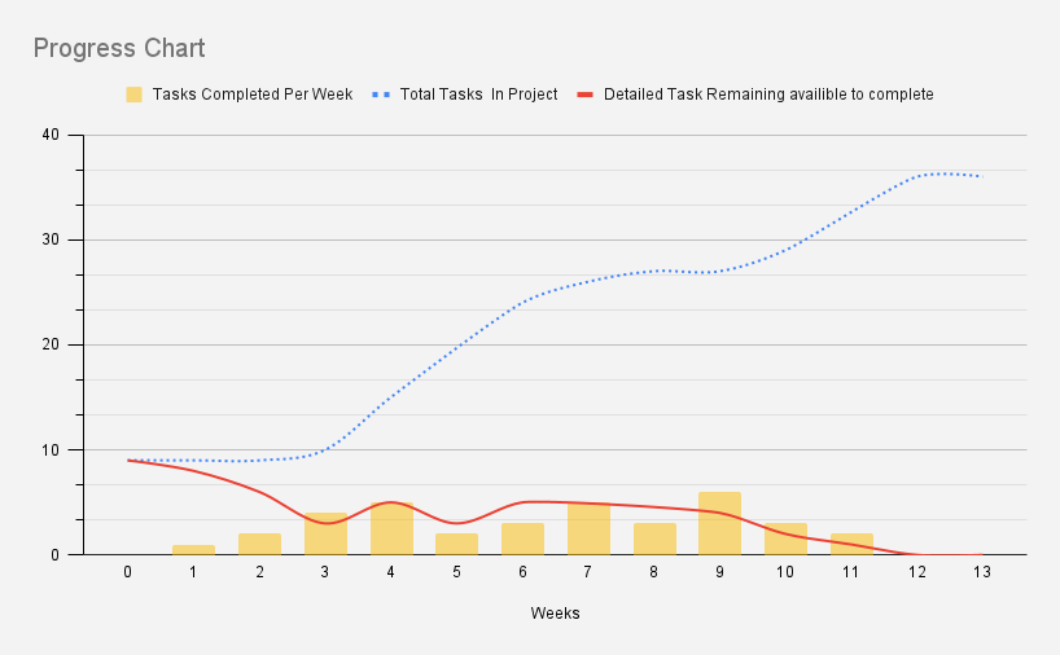


FIGURE 5.1: Rate of work done each week

Over the course of the 12 weeks, tasks were completed on a regular basis across the semester allowing for steady progress towards and an achievable end goal and deadline. Certain elements were scaled down to make more time to polish the core 20% of functionality and ensure that the applications would be in a polished functional and mostly bug free state by the end of the semester.

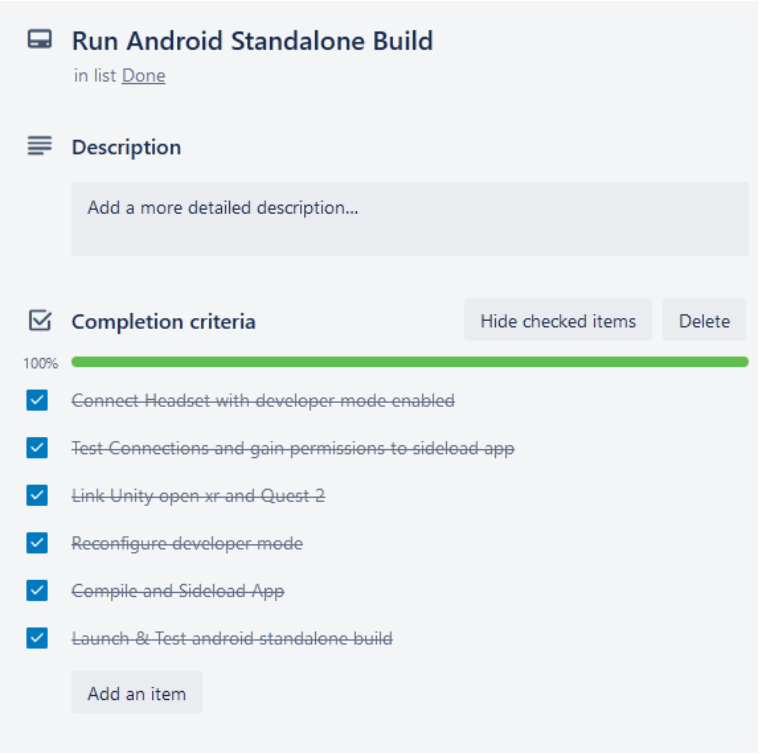


FIGURE 5.2: Completion Criteria Example

Each task on the Kan-Ban board had a list of completion criteria allowing them to define how each task would be worked on. All of the tasks were ordered into 4 columns depending on whether they could be started, if they were in progress or they were blocked until a current task was completed.

Trello Logs are detailed in A

This meant that when we had time to work on the application, the task was structured and would progress the project in a meaningful way.

5.2 Difficulties Encountered

Over the course of development there were three difficulties that altered the original development plan.

5.2.1 Future Proofing

While much of the research was conducted into current standards due to the rapid pace of the VR industry, more time was taken up during development to adapt to ongoing changes within the industry. Careful planning was needed to make sure to use plugins and systems that would be relevant and open to future development and not conflict with future standards and technologies. With the release of numerous SDKS over the last year by Valve along with many updates to future and backwards compatibility to SteamVr and with Oculus moving away from their proprietary run-times[22] along with new VR hardware providers entering the market. Many of the systems previously used for development have been updated so much that developers will struggle to adopt the new standard with older projects, however with these new standards in place and our development with these standards in mind our application should function untouched for many years.

Much of this future proofing could be done by the use of a preview version of a new framework for Unity 2021.2.10f1 at the beginning of development. Called the XR Interaction Toolkit Version 2.0.0 preview, it rewrote much of the way that controller inputs, movement vectors, grab physics and menu interactions are used within Unity XR applications including VR and AR. This allows for multiple types of controllers and is open to new controllers and headsets if they're being managed by either the Oculus, SteamVr or OpenVR/OpenXR standards. This toolkit is now released to version 2.0.1 as of 03/04/22[23].

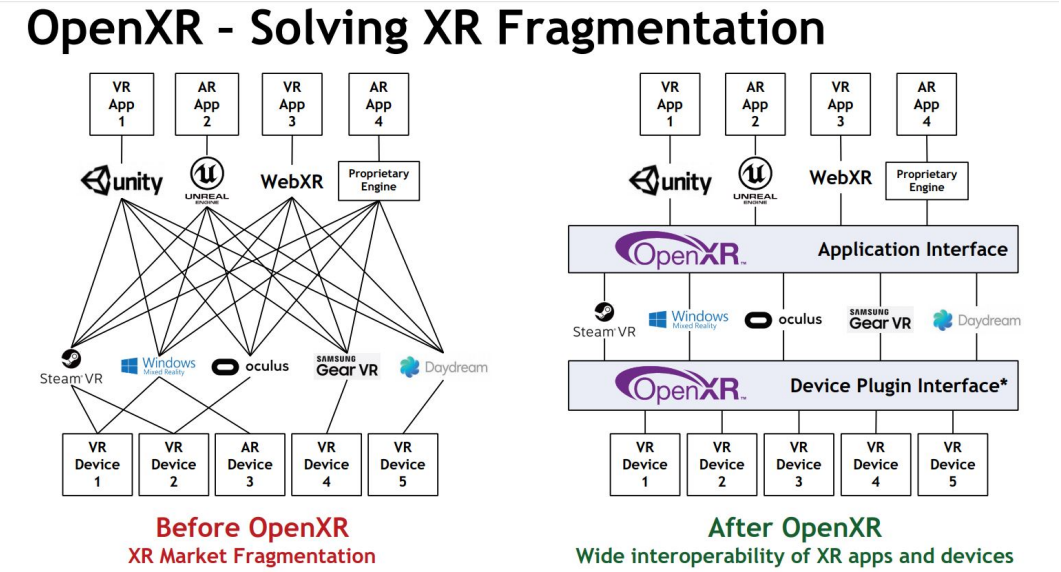


FIGURE 5.3: Open XR's Approach and the importance of it[9]

By designing our application with this framework and OpenXR compatibility in mind, our application is hardware agnostic for the foreseeable future as the main run-time other than OpenVR used in industry has been deprecated after Meta/Oculus declared in July 2021 that new projects, yet to begin development would be designed without using the original proprietary run-times created by Oculus[22] and instead use the new open-source OpenXR standard.

These frameworks allowed us to make the application compatible with all currently known HMDs, controller layouts and designs including the ability to function on yet unreleased Android based platforms not using the Oculus Quest 2 Android API. This meant that our application is compatible with the upcoming Quest 2 competitor, the "Pico Neo 3" HMD designed and released by Tik-tok's parent company Bytedance. These headsets along with business grade HMDs from HTC and others use many of the new systems that all new headsets will be using going forward. Using the new toolkit should enable full future compatibility with any upcoming headsets across many operating systems, system architectures and controller designs.

5.2.2 Unity Object Limits

Over development and testing we ran into issues with object registry handling within Unity. Initially we believe it was due to the textures and polygons in use by the objects but we soon discovered after extensive optimization and analysis of the applications threads, internal process queue and the rendering pipeline that performance was heavily CPU bottlenecked.

We found that each new object tracked within Unity added significantly less than a millisecond onto the CPU queue on a per-frame basis. This meant that in comparison to normal game development where GPU performance and resolution is the main performance limiter due to large amounts of polygons, textures and visual/post-processing effects. We encountered an issue that was harder to tackle. As more objects are added to the application, the larger this bottleneck becomes and adds more delay on a per frame.

While multiple steps were taken to reduce the bottleneck by reducing the graphical render thread delay this couldn't compete with the delay in frame-times created by excessive distinct objects. The graphic render pipeline was drastically sped up by using a shared mesh between all of the objects and three textures and by using static and dynamic batching along with removing physics, shadows and combining the meshes of all objects after they are spawned in, this means that the GPU and CPU is aware that all of the objects use the same data and can reuse the data in memory extremely fast. By going without shadows, we no longer need to render shadows dynamically on thousands of objects at a unique angle from the light source, in this case the sun.

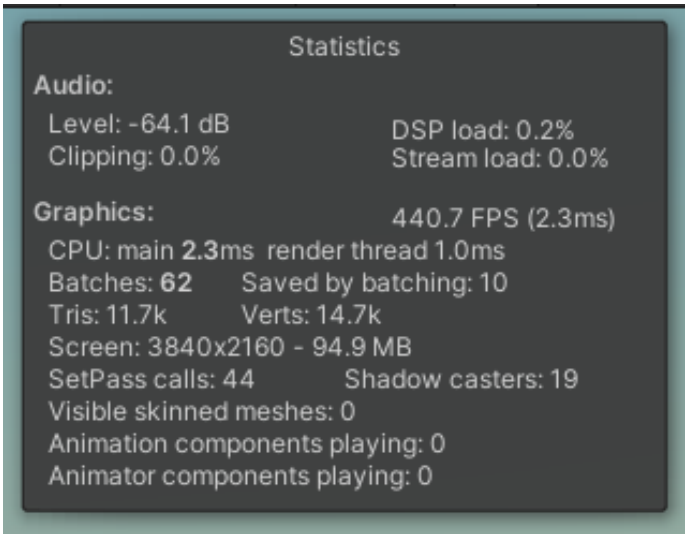


FIGURE 5.4: Performance with 0 Objects

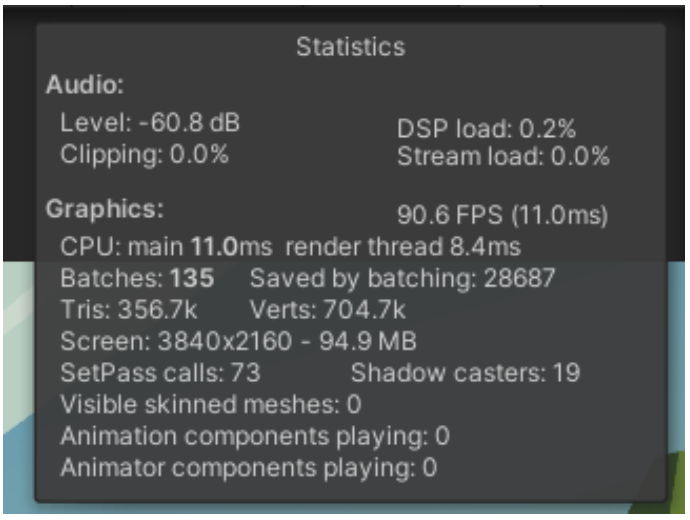


FIGURE 5.5: Acceptable Performance Metrics with 20,000 Objects

We had to settle on a limit of 20,000 objects as larger amounts of objects increased the time needed to generate each frame. While performance would be fine on a desktop screen, going below 45fps on a HMD can be quite unpleasant especially to the more motion sickness susceptible first-time VR users. This optimisation also reduces the interact-ability of the objects

5.2.3 Performance and Frame-rate within VR

Due to the nature of VR, HMDs run at a high resolution and a higher than normal screen refresh rate. Due to also being worn on the head any lag, or stuttering can cause the user motion sickness and is generally quite unpleasant. Due to the discovery of the performance bottleneck, measures had to be taken in order to make the application less nauseating at high object counts, this is mainly apparent from the object limit of 20,000.

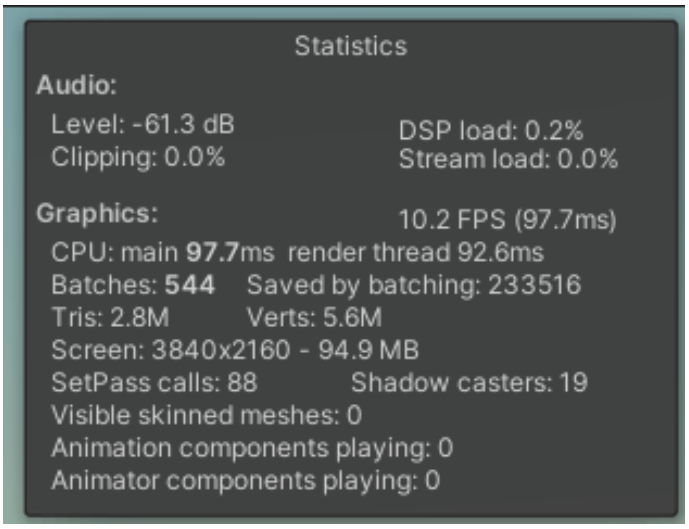


FIGURE 5.6: Unacceptable Performance Metrics with 100,000 objects

While this is less than we had hoped for, combating and increasing the object could be done, however this would require a rework of how object spawning is handled too near to the deadline as the simplest solution would be to have single objects that mimic groups of objects. Further research would need to be done in order to find if this could be done dynamically or if variations of mimicked objects would need to be made and the spawning system altered to allow for example, 9 red objects to be replaced with 1 object that looks like 9 objects.

5.2.4 Difficulty Classification

Classification of Difficulties

Difficulty	Difficulty Level	Changes to approach
Future Proofing	Easy	None
Unity Object Limits	Hard	Limited amount of objects
Performance and Frame-rate	Medium	Required time to optimize

5.3 Actual Solution Approach

During this project there were 5 major stages to the development of the features found in the final product.

5.3.1 Unity Setup & Testing

Initial setup of unity took extra time due to the intricacies of setting up Unity for VR. This differs from normal development in that VR depends on connecting to external hardware that Unity cannot interface with out of the box.

In order to use Unity to develop VR games an extension called OpenXR is needed to to interface with the HMD and it's controllers. It allows Unity to connect to the default OpenXR run-time on the users PC, in our case this was SteamVR which opens and handles the computers connection to the headset and OpenXR allows Unity to receive and interpret data from SteamVR in a hardware agnostic way. The data it receives is information such as headset position, rotation, resolution along with similar information for the controllers. SteamVR handles the device, sends video to it and interprets the positional data from the headset in regards to a play-space set within SteamVR.

With the OpenXR plugin installed along with the Oculus Plugin for PC's only using Meta/Oculus Software & Hardware this allows Unity to render the application in a specific way. Due to how HMDs lenses work to bend, stretch and focus the image we can see this is a large high resolution image that will then be manipulated and stretched by the image to make a screen an inch away from the users eye imitate a world around them, Unity sends two images to the headset that are square in shape, this is due to the headset have to lenses. Each image for each eye is different as the left eye has to be offset from the right-eye in order to simulate depth perception and properly immerse the user.

Another key component was the use of the XR Interaction Framework V2.0.0 Preview which is how a users viewpoint is bound in game and is their presence within the virtual space, it handles controller movement in the 3D space and allow the user of the program to interact with things using their VR Controllers. It also provides changes to menus, and provides systems for a users vr controller to pick up and manipulate objects in the world.

This required a lot of tweaking as found in the next phase of development.

5.3.2 Environment & VR Fundamentals Table

After initial setup and testing of the connection between Unity and our development HMD, we found that the default environment, while functional, could be intimidating to new users. We also needed an area to test out functionality of object physics.

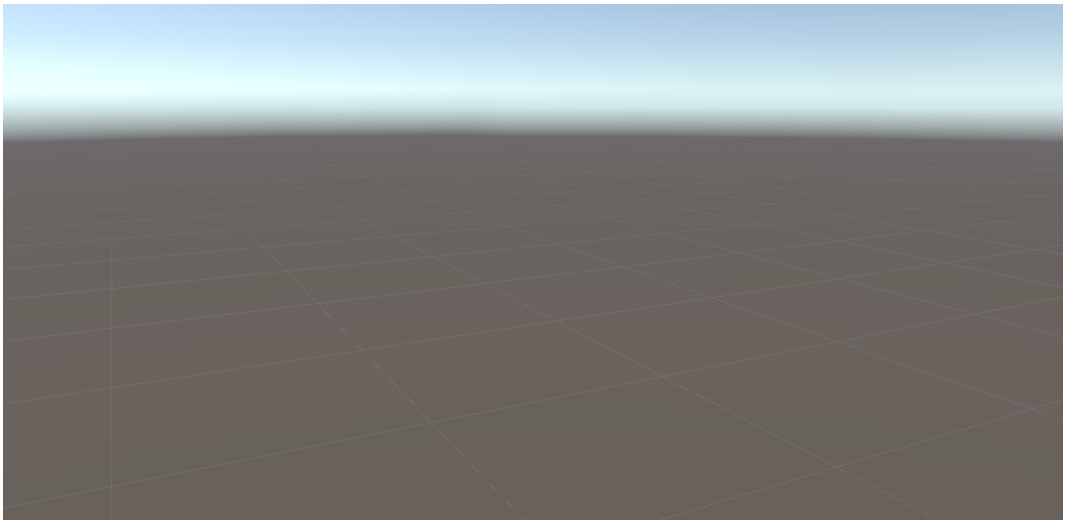


FIGURE 5.7: Default Unity Environment

Due to how new VR is, many people may be unfamiliar with how it works or have never used it in the past. By creating a friendlier environment the application should be less intimidating for a first-time user. We implemented this by using a simple performant cartoon-like art style to replace the bleak grays with a nice green floor and sky-box along with some simple royalty-free assets from the unity store to decorate the area. We found however that something was still not quite right and we found that also adding some basic environmental ambient sound made the world less intimidating.

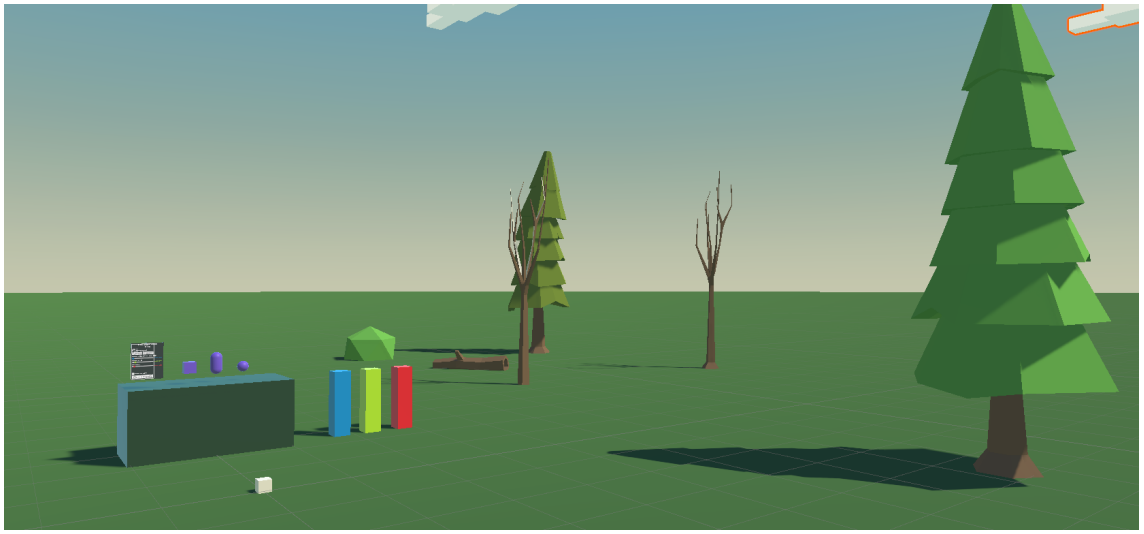


FIGURE 5.8: Simple, friendly environment

Another key element here was the "interaction table", that allows new users to pick up and manipulate virtual objects and get to grips with how the 3d space in VR coincides with their movements in their play-space. I.e. Leaning, crouching and stepping in different directions and how it varies from the movement used by the application when the left stick on the user's controller is used to move. This initial area makes it easy to explain basic principles to a user that are hard to get across with words alone and makes it easier to progress to the data visualization portion of the application with less confusion.

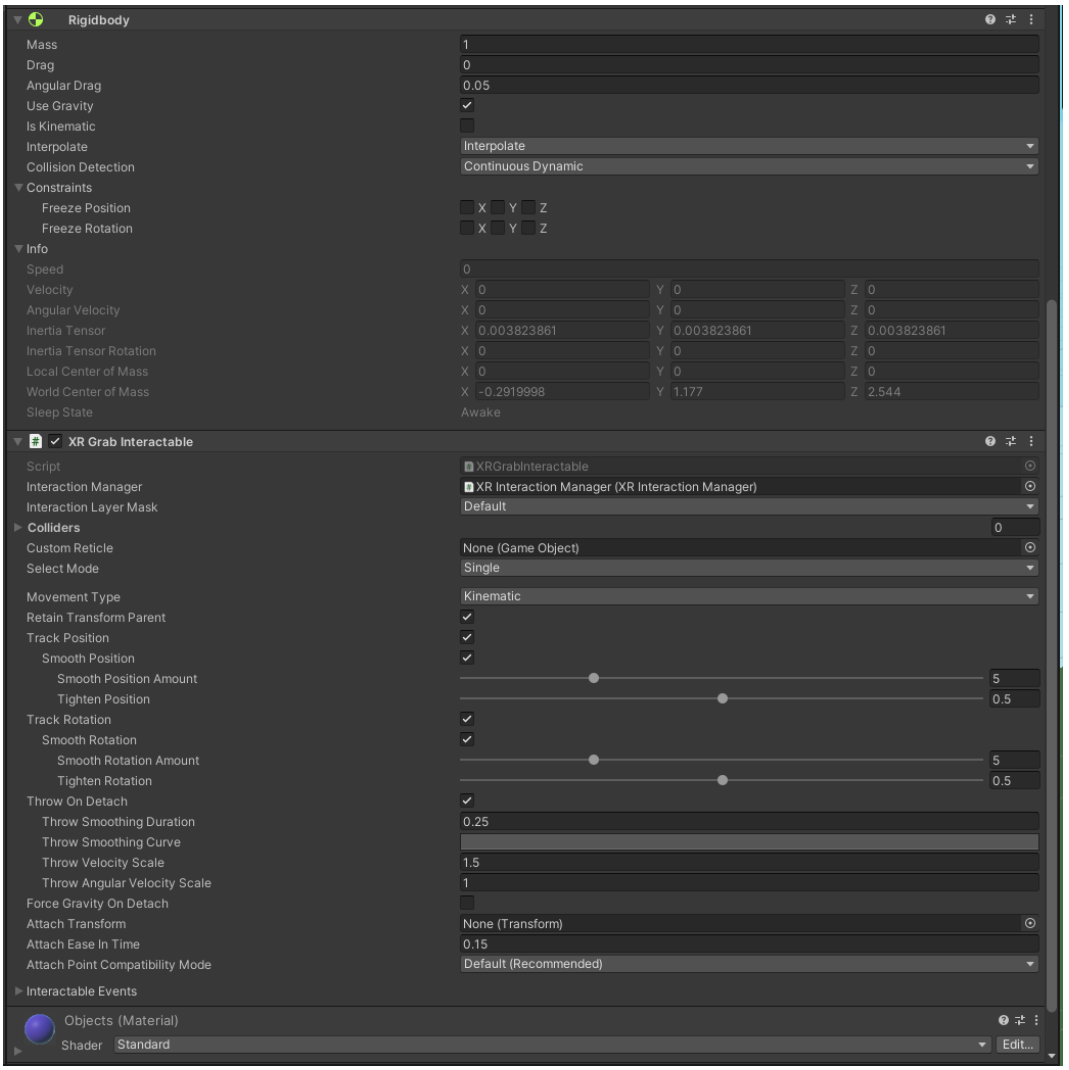


FIGURE 5.9: Final object configuration

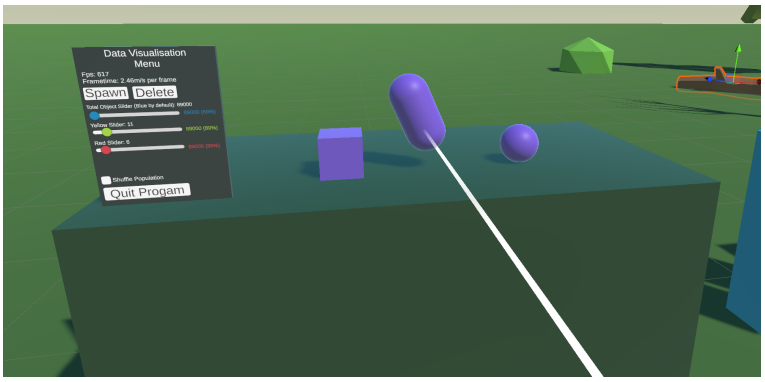


FIGURE 5.10: User Interaction Table

A key component with this interaction table was to test physics within Unity and evaluate different setups. However this was not as simple as enabling gravity, What we found when testing the components is how they reacted to being held and moved within the world wherein we settled with the above configuration to allow for smooth movement in the hand.

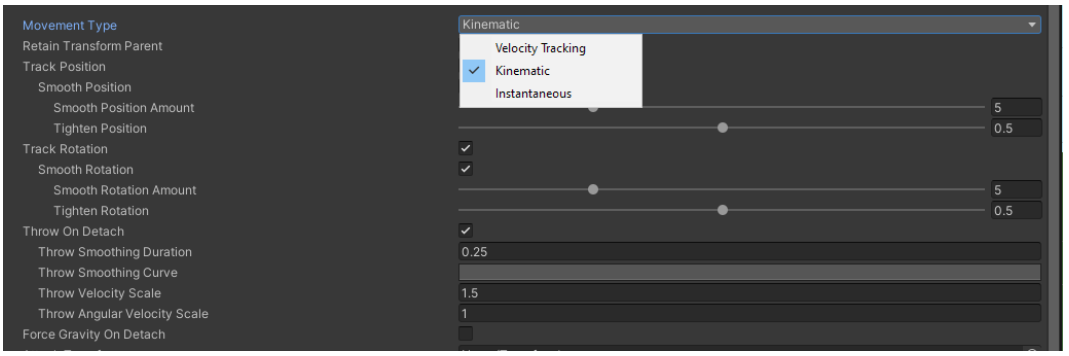


FIGURE 5.11: Object movement in hand configuration

The key changes to default found above is the use of Kinematic movement on the object. This setting moves the object through the world space smoothly based on the coordinates of the hand. It is stable as it moves through the air and does not experience flickering like similar configurations. This when combined with interpolation of positions further smoothed the movement of the object. 1 downside to this is that while the object is "held" it doesn't have collision with objects as it moves when compared to velocity tracked which tries to move the object in relation to the controllers movements and can then collide with the table as it moves due to velocity and if it touches the table for instance the velocity is still applied but the movement is blocked by the object. This provided stuttered and sometimes lagging movement wherein the object followed the controller but was often slower than the controller allowing for a disconnect between controller movement and the object.

5.3.3 Testing Data Visualization Limits

During initial testing a random amount of objects are spawned in the area in-front of the table in the spawn/staring area. This began as a grid of 50 x 50 equalling 2500 objects. On initial testing and after a demo the initial direction of the visualizations received favorable feedback informing our direction with the next phase of the project.

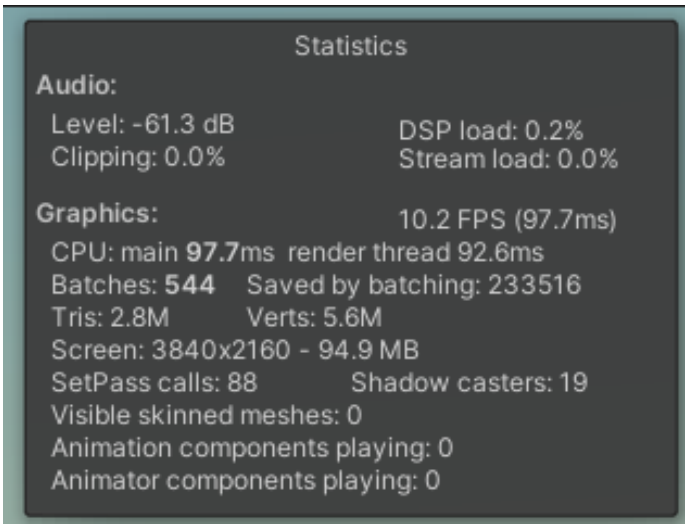


FIGURE 5.12: Unacceptable Performance Metrics with 100,000 objects

This is when we encountered our major difficulty in this project. Every object added to unity added less than a millisecond of processing to unity's render pipeline. Initial steps to optimize the application focused on reducing the graphical elements being rendered. Simplifying the models, removing dynamic shadows, removing physics and collider meshes from each object helped reduce the amount of time each object added to the render queue, then using a shared texture and mesh reduced the draw calls needed allowing static and dynamic batching of objects. This reduced the amount of I/O operations that had to pass the CPU as each object was drawn.

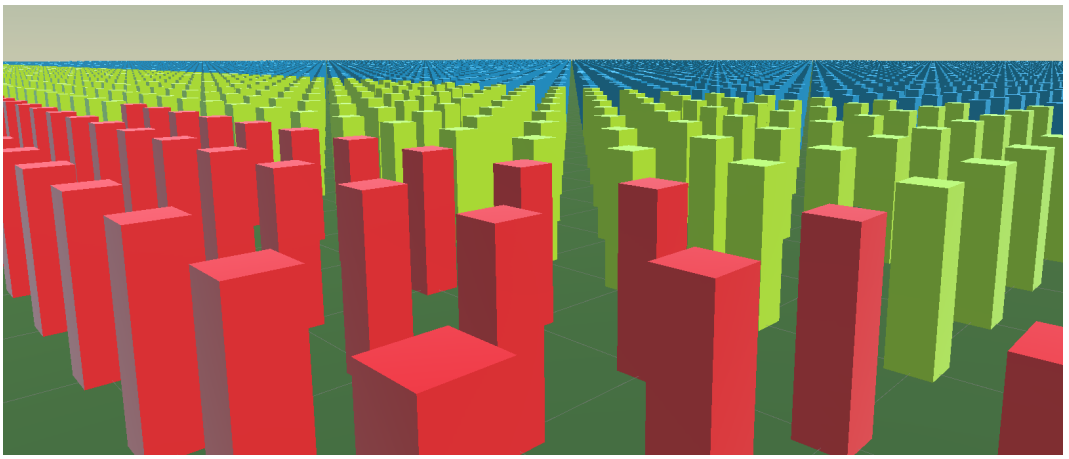


FIGURE 5.13: 20,000 Objects stretching to the horizon

After optimizations the new limit to objects before the performance deterioration rendered the application unpleasant to use is currently 20,000. While not as high as hoped. It does provide the horizon stretching visualizations planned in the research phase. The removal of shadows, colliders and physics does also remove our ability to interact with these objects. A system could be implemented in the future to choose to have fewer objects but retain the ability to interact with the objects using the controllers.

We also made sure to optimize our use of dynamic occlusion culling, this system within unity allows objects behind you to be removed from the render queue, increasing performance massively as only what is in front of the user affects the performance especially if the player is located in the center of the objects instead of viewing them all from on side.

5.3.4 Basic Menu System

The next key element of the application was to allow the user to change how many objects were spawned, and the amount of each type of object that was spawned. To do this we implemented a tablet style interface that can be found alongside the test objects. This tablet system allowed the controls for the application to be portable and move with the user in a simple yet intuitive manner.

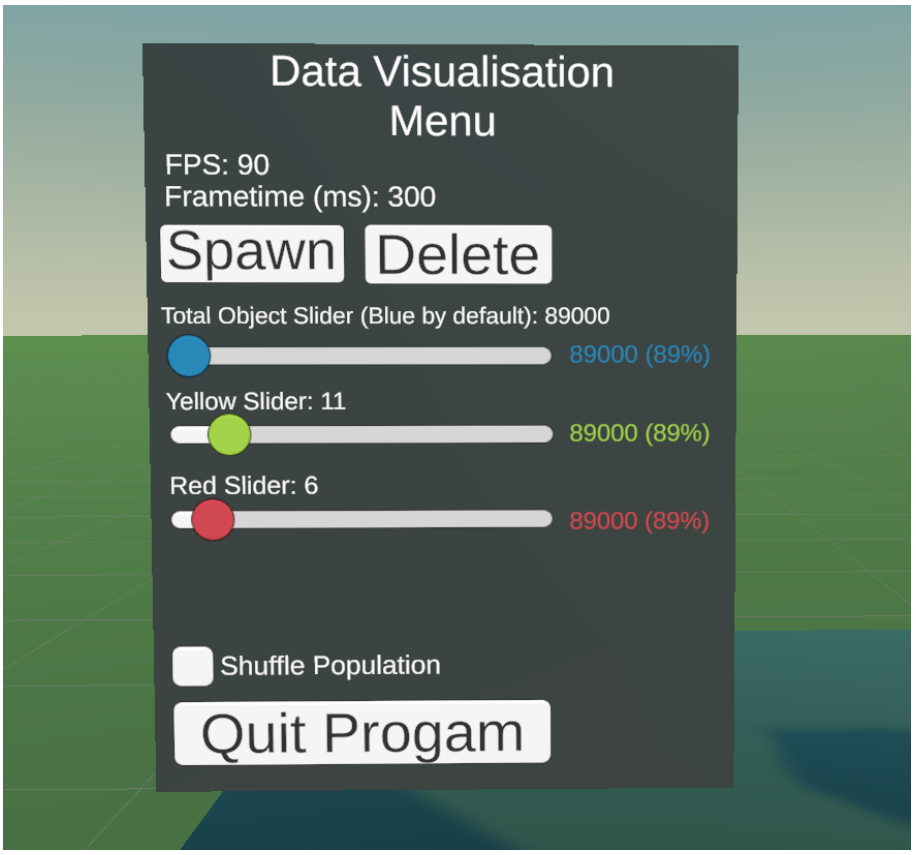


FIGURE 5.14: Finalized tablet

To make this tablet easier to use while not interfering with performance, the GUI only updates when the slider on the tablet is used. This ensures fast and responsive sliders. The slider's maximum value also changes in a descending manner. This means that if 50% of the objects are coloured one color it's only possible to set the other colors within the remainder of this percentage. The sliders are also ratcheted and round to different values within certain ranges to allow accurate steps. Above 2500 objects it switches to rounding to every 500, then 1000 and finally after 10,000 it switches to steps of total objects. This makes it easier to select exact numbers along with color coding the sliders and the text feedback helps to intuitively explain the usage of the menu. The simple slider control system with coloured sliders and feedback on how many of each type of object there was also helps convey extra information to the user.

While this changed from the solution approach to have a controller bound menu, this was a quicker system to implement that is more inline with the approachable nature of the application. It is also expanded with the options to have seperate menus on the same table or to have a second table perhaps with intermediate controls with added complexity.

5.3.5 Data Visualization Customisation

The menu currently gives the user 2 ways to visualize data and challenge their ANS. You can color code each object three specific ways using either blue, red, or yellow or all three at once. This allows the program to be adapted to a wider variety of statistics and improve its functionality and represent more complicated statistics. By being able to adapt how many of each object is on screen accurate while also receiving a coloured and easy to perceive percentage it gives the user yet another angle to view the data along with the visualization from.

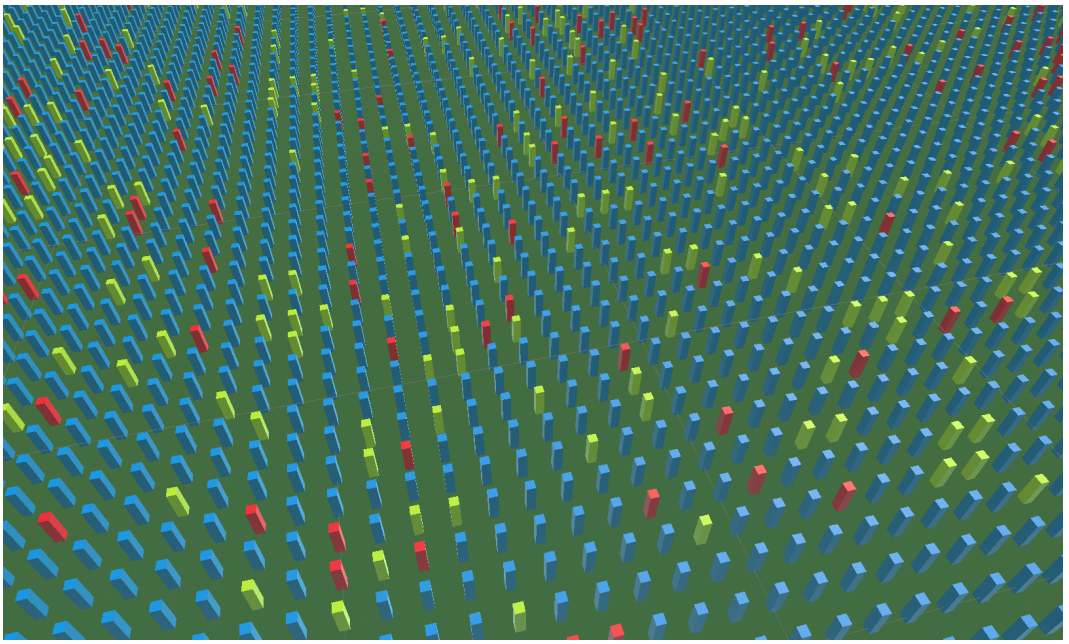


FIGURE 5.15: Shuffled Population

The next piece of functionality is the "Shuffle Population" button. This allows the user to see the percentages they have chosen but spread out instead of grouped. This is an interesting way to view both sides of the statistics and helps give a different outlook on the data similar to looking at a city street with different types of people mixing when compared to a football stadium split by ticket type etc. This piece of functionality made it more difficult to optimize the program due to each object needing to be an individual

object in order to be shuffled and placed within the world which ruled some methods of optimization.

This type of additional functionality is an example of the modularity of the application. To add a new way to the view or arrange the data inputted with the slider, it's as simple as adding a new placement method into the gridspawner.cs file. Each new feature can then be added to the tablet menu.

Chapter 6

Evaluation and Testing

In order to evaluate the usefulness, functionality and use of the application we will use the objectives outlined in chapter 3 and the criteria proposed during the Implementation Approach stage.

6.1 Evaluation

- Easy for non-VR users to pick up and learn.
- Feature a simple and clear user interface.
- Have multiple ways to interact with spawned objects.
- Adaptable spawning system to meet multiple statistics.
- Run on different hardware for future and backwards compatibility.

FIGURE 6.1: Evaluation Criteria proposed in chapter 4

The above criteria, while proposed before development of the solution remain relevant in order to evaluate the prototype application after some slight rephrasing. Furthermore, these qualities will then be used as the main focus of a limited user testing run.

6.1.1 Easy to pick up

Throughout development an intuitive application suitable for all ages was important. This led us to the development of the tablet-styled interface. This system is familiar for people of all ages and has multiple benefits. A tablet style interface has an inherent use. It features a simple user interface with familiar elements. A slider is a common convention in computers, phones and has numerous uses and can be found in multiple physical interfaces found in multiple industries. Buttons as well are a commonly used method of input that users of all ages should have a basic familiarity with and since the menu is used on a tablet this is immediately reminiscent of digital tablets such as an Apple IPad.

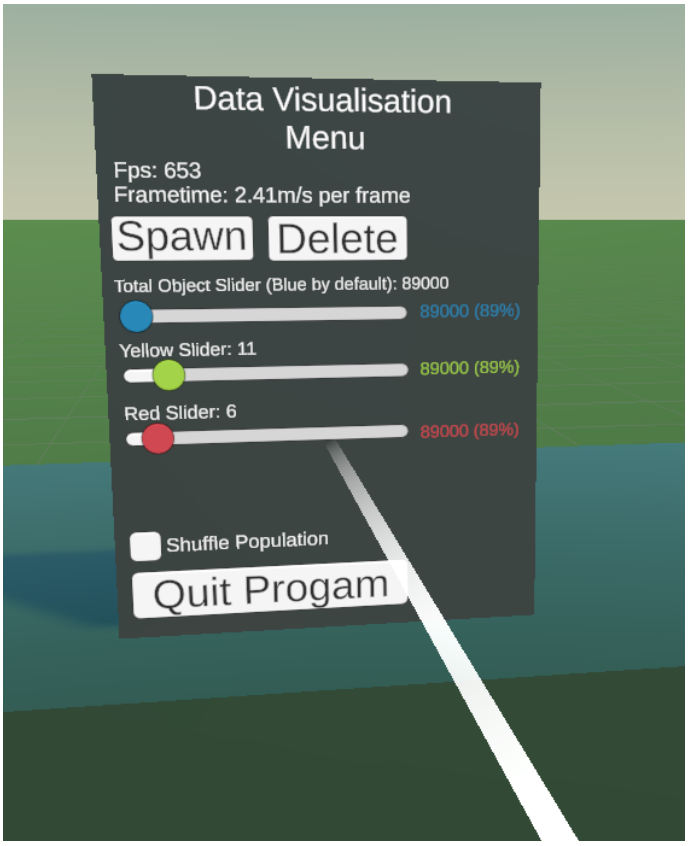


FIGURE 6.2: The Tablet Interface

This familiar concept allows our interface to be more actionable and open to input than a holographic menu attached to the wrist or a wall mounted menu often found in Unity applications which new users would be less familiar with. Wall based, floating and wrist-based menus are an alien and unknown style of user interface to users new to VR and Unity applications and by removing this and using a more common tablet interface the application is easier to explain to a wider audience.

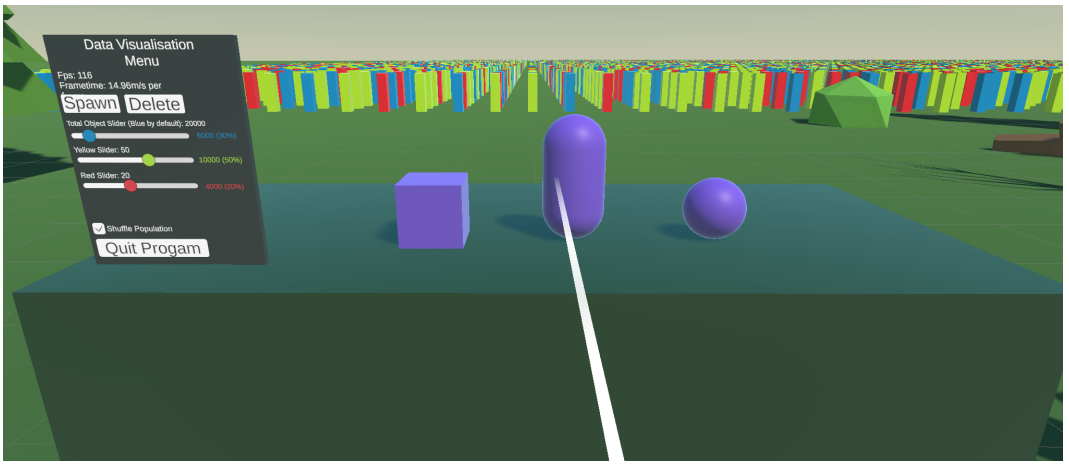


FIGURE 6.3: User Interaction Table

The rest of the application is straightforward and requires little explanation. Once the application is started the user can begin using the controls available to them to explore the main area, the table nearby acts as a two part system. It gives users their first small goal which is to traverse to the table. This involves using the thumb-stick of the left controller. Movement is set to slow in order to minimize motion sickness. After a few brief moments the user is at the table where they can be prompted to reach out and grab an object of their choice from the table where they can find they can pick it up, and upon putting it down it reacts realistically by rolling on the table. This does two things, it intrigues the user of the program with the concept of interacting with virtual objects while also grounding the fact that the virtual object behaved realistically in relation to the virtual table allowing the lines to be blurred between physical and virtual objects.

It's not uncommon when a user is immersed in a VR application for the line to be blurred between what is real and what is virtual[24]. This leads to users leaning, climbing or sitting on objects only found in the virtual world. This type of immersion is important as it will allow the statistics and objects used to represent them more impact-full if the user perceives them as objects with volume that surround them rather than just boxes on a screen.

6.1.2 Simple User Interface

The user interface of the tablet as previously discussed in section 6.1.1 allows the applications controls to be picked up and moved by the user of the application in a familiar manner for people not versed with the GUI systems in-place within Games and VR. This simplistic implementation is recognizable to all ages and after a minute or two at the object table. The jump from picking up objects to using the menu is a small change for the new user.

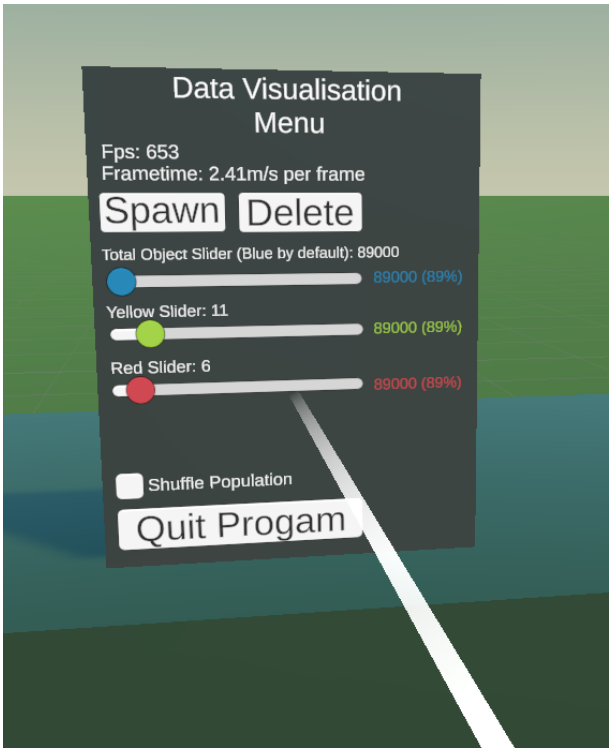


FIGURE 6.4: The Tablet Interface

Each button is clearly marked and the user of the sliders to choose, how many objects and what percentage of each color is color-coded. If the yellow slider and red slider are set to 0 then 100% of the objects will be blue. When moving the slider it says what position its and and if the yellow or red slider is set to 50% and the other slider is still at 0%, the maximum that the unmoved slider can be moved to is 50% and all will yellow and red with 0% blue. Each piece of text at the end of the slider says how many objects will be coloured that way in both text form and percentage to allow the input of statistics in an easier manner.

Some more work may need to be done with the "Spawn" and "Delete" buttons to make their use more clear as without a prompt from the program or more information given by the supervisor of the user. "Spawn" is an uncommon word and it also does not refer

to what will be spawned when out of the context of this paper. There is also a large "Quit Program" button that.... quits the program.

The "Shuffle Population" checkbox changes how objects will be arranged when spawned and its effects are obvious when tried. It does however only work before the objects are spawned so changes to make it seamlessly shuffle the objects would cut down on the number of menu interactions a user has to do.

The testing environment of the application features some simplistic assets, sky and clouds along with wind and bird sounds to make the application more approachable to a beginner VR user or those who have never worked with statistics. This added environment, while simple, is a feature not found in many research papers surrounding VR and statistics[25] that in-turn makes this program more approachable to a novice user.

6.1.3 Multiple Ways to interact with objects

After extensive work on optimisation the application does not feature any grab or physics interaction with the statistical data. This was cut to reduce the time each object took to render in the CPU pipeline allowing an increase in the final amount of objects on display at an acceptable performance level.

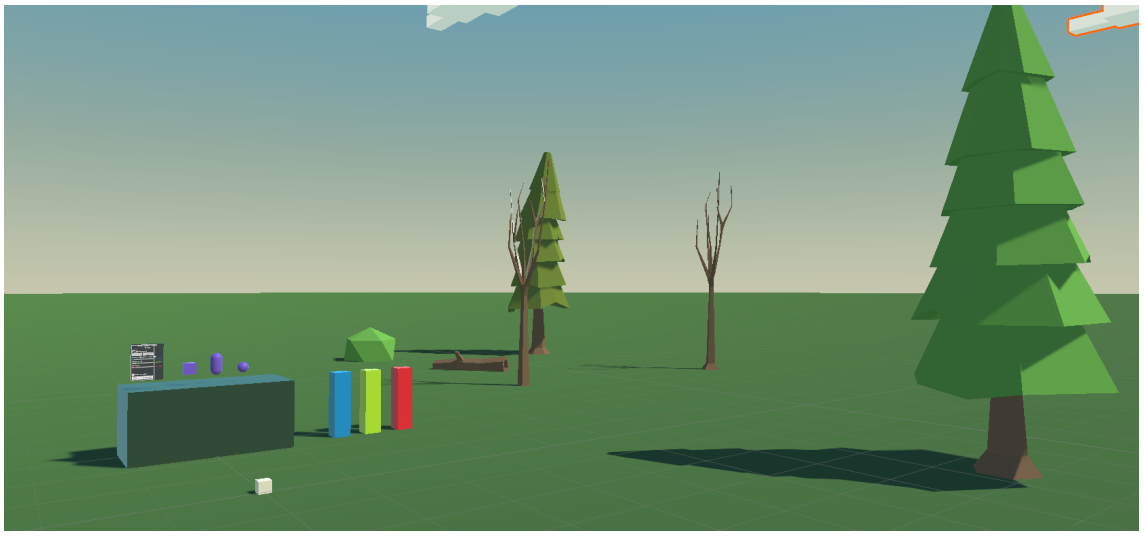


FIGURE 6.5: Simple, friendly and welcoming environment

Objects on the table can be grabbed, moved and pushed off each other and thrown. This feature could be re-added to statistical objects in future if using low amounts of objects allowing for domino style effects and more etc.

6.1.4 Adaptable Spawning System

The total number of objects can be increased and decreased with color-coded feedback on exactly how many objects of each color are present. There are three color options to choose from this adds the ability to add in extra percentages or split percentages in use dynamically as colors are limited by percentages having 70% of the objects yellow limits the user to at most either having an additional 30% of the objects blue or having a combination by setting the red slider to 15% leading there to be also 15% of the objects coloured blue.

The objects can be viewed in two ways. Each gives a slightly different interpretation of the data selected. This aerial view displays the difference between shuffled and grouped. Each is useful for viewing different types of statistics regarding populations.

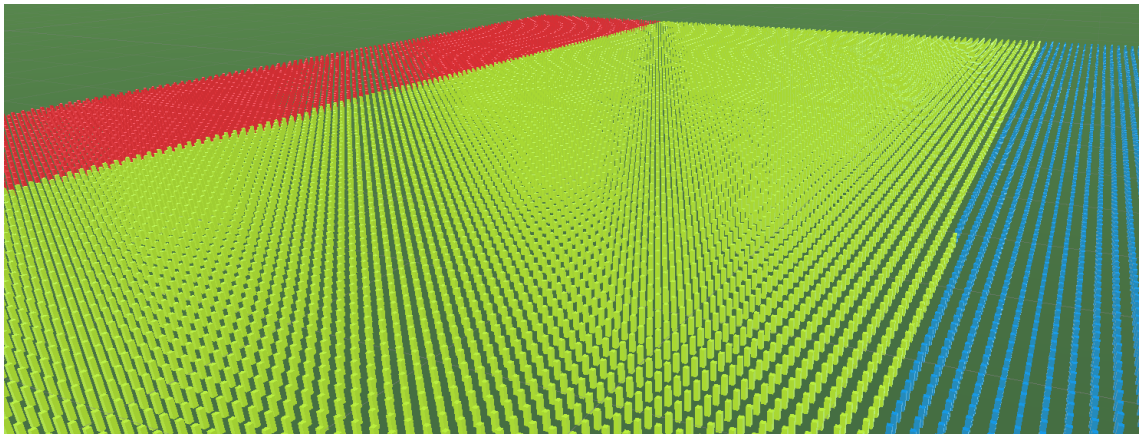


FIGURE 6.6: Grouped objects of 20,000. Set to 50% Yellow, 20% Red and therefore 30% Blue

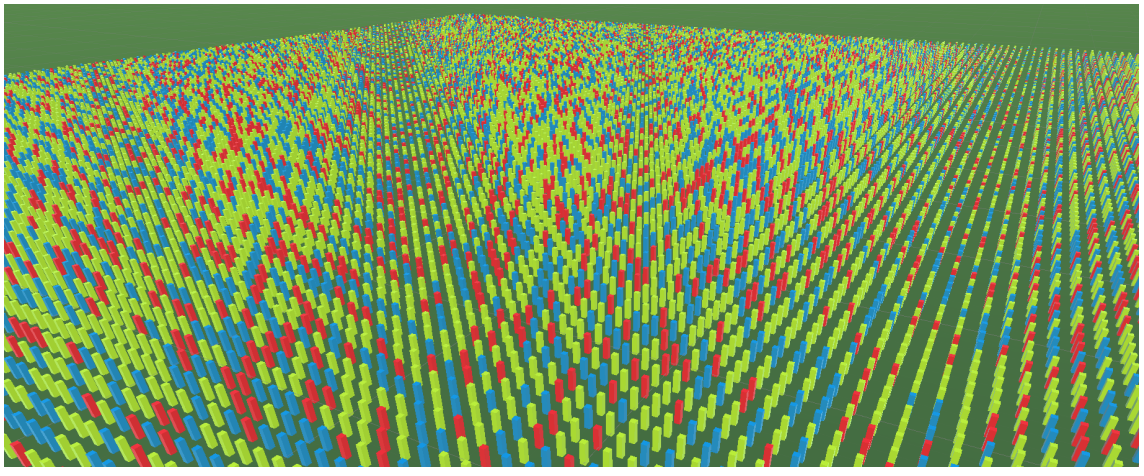


FIGURE 6.7: Shuffled objects of 20,000. Set to 50% Yellow, 20% Red and therefore 30% Blue

However this data will be viewed in application from a human height, allowing the user to crouch and lean. The thumb-stick dictates the location of the user's play-space within the virtual space. From eye-level 20,000 objects stretch to the horizon and past the usable resolution of the HMD.

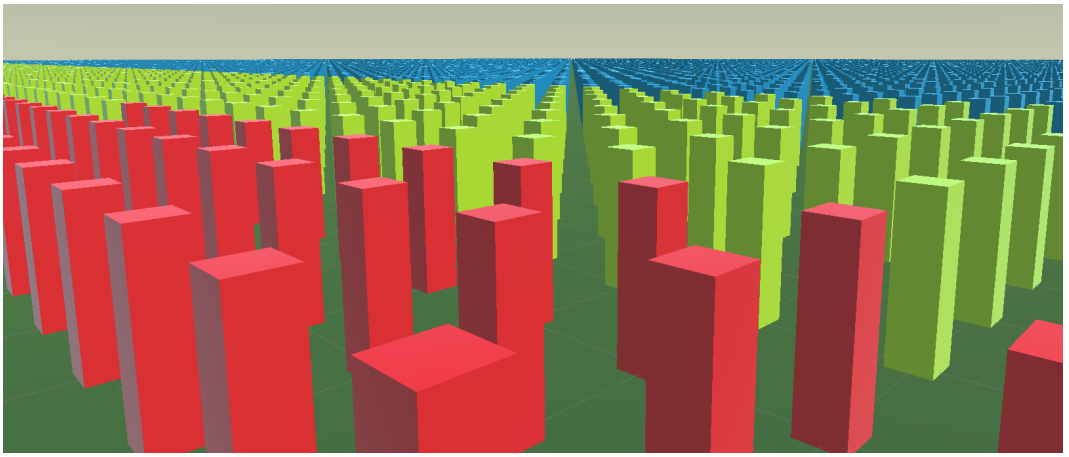


FIGURE 6.8: 20,000 Objects stretching to the horizon

When viewed from eye-level, while not as numerically high as originally intended, does have the intended effect of putting a user within the statistic or population. From this perspective it would be possible With further tweaking to have an infinite grid, without exceeding the object limit.

6.1.5 Forward/Backwards Compatibility With Hardware

With a multitude of standards, technologies plugins and versions available it was important to gather, review and rate candidate technologies based on their background documentation maturity, or lack thereof in the case of the "XR Interaction Toolkit 2.0" that was in preview at the beginning of development. The trade-offs involved between future and wide compatibility vs the available documentation and material useful to the development of the solution

The forward and backwards compatibility of the application is ensured with the implementation of OpenXR. This standard allows the application to work in-dependant of HMD type or VR run-time that is used assuming the standard is still in use even after potential radical changes to how PC's run VR applications in the future.

While optimization is untested, the application can be compiled and run on an android based headset such as the Quest 2 and potentially upcoming headsets that are entering the standalone space.

6.2 Objectives, Functional & Non-Functional Requirements Review and Exploration

At the beginning of the project, in sections 3.3.1 - "Functional Requirements" and 3.3.2 - "Non-Functional Requirements" we proposed the intended requirements to complete the objectives laid out in Section 3 - "Problem Statement". This occurred before development of the "Solution Approach" and the "Implementation Approach" had begun.

These objectives directly influenced the evaluation criteria previously reviewed in this section and we evaluated and featured much of the objectives, functional and nonfunctional requirements in some parts.

Table 6.1: Objectives Addressed by Evaluation

Objectives	Addressed?
Easy to pick up	Yes
Immersive and Informative.	Yes
Provides simplified visualizations.	Yes
Statistics can be customized within virtual reality.	Yes

Table 6.2: Functional Requirements Addressed by Evaluation

Functional Requirements	Addressed?
3D, Virtual Sandbox Environment	Yes
Vast Quantities of 3d objects	Yes
Dynamically arrange 3d objects based on user input	Yes
User movement, interaction and immersion.	Yes
Support OpenXR	Yes

Table 6.3: Non-Functional Requirements Addressed by Evaluation

Non-Functional Requirements	Addressed?
Immersion	Yes
Visualization	Yes
Simplicity	Yes

6.3 User Testing & Feed-back

During development, the application was often tested on the limited number of potential testers found in close-proximity. The feedback received was useful in making changes and decisions when working with the difficulties encountered over the course of development.

During the semester an early version of the application was also demonstrated to the project's supervisor. The result of which ratified the prototype as the correct direction going forward even with a reduced amount of objects to be found within the application.

A consistent remark from testers was that they always thought there were a lot more objects surrounding them than there were numerically by an order of magnitude further proving the hypothesis and solution approach in regards to trying to train the user to more accurately approximate these numbers. This further reinforced the direction of the project and led to refinement of the menu system into a tablet system that was color coded as it was easier to understand. This feedback also added to the stepping mechanism of the total object slider, which allowed the slider to ratchet to even intervals such as 20,000 rather than 19,789 for example. These intervals also scale up with the slider and past 10,000 it switches to intervals of 2,500, an element that requires further refinement.

6.4 Results of User Testing

When tested by changing the amount of objects spawned, users remarked that they got better at approximating that amount surrounding them, especially after they were able to use the menu tablet to test out specific amounts of objects first and then get tested with a random amount again afterwards.

Another finding was that while users could approximate the percentage of each color when they were grouped. Even when the percentage of colors remained the same but the groupings we shuffled users were unable to say with any certainty if it was the same, more or less. Similarly when tested with shuffled populations first, users either underestimated or overestimated the percentages involved and found that revealing the correct percentages swayed some opinions immediately whereas with other

6.5 Potential Future User Testing

If further testing was to occur, we would suggest a larger scale test with multiple users with a set amount of approximate number system challenges along with the use of different statistics and percentages represented in both shuffled and un-shuffled form before and after the approximate number system challenges. This should gauge accurately if a user's ability to comprehend such statistics improves for the user after their VR experience with the application.

Chapter 7

Discussion and Conclusions

In this section we will discuss the solution we implemented to answer our research question, the project's progress, difficulties and skills learned over the course of this research paper and implementation.

7.1 Solution Review

After a thorough quantitative evaluation in Section 6 in line with our proposed evaluation criteria while also revisiting the previously stated objectives of the solution. We also evaluated if the functional and non-functional requirements had been met over the course of the solution implementation and what changes were made after user testing.

At the beginning of the project our intentions were as follows:

- To discuss the research question.
- Develop and implement an approach to the research question.
- Validate our approach after evaluation and review.

Our solution addresses the problem outlined in Section 3 by following the objectives initially laid out after formulating the research question. The applications follow through on the following Objectives, Functional & Non-Functional Requirements outlined below.

TABLE 7.1: Objectives Addressed by Evaluation

Objectives	Addressed?
Easy to pick up	Yes
Immersive and Informative.	Yes
Provides simplified visualizations.	Yes
Statistics can be customized within virtual reality.	Yes

TABLE 7.2: Functional Requirements Addressed by Evaluation

Functional Requirements	Addressed?
3D, Virtual Sandbox Environment	Yes
Vast Quantities of 3d objects	Yes
Dynamically arrange 3d objects based on user input	Yes
User movement, interaction and immersion.	Yes
Support OpenXR	Yes

TABLE 7.3: Non-Functional Requirements Addressed by Evaluation

Non-Functional Requirements	Addressed?
Immersion	Yes
Visualization	Yes
Simplicity	Yes

7.2 Project Management Review

Over the course of the project I encountered new difficulties, constraints and the skills to overcome them and develop a solution that was aimed to help answer the question initially put forth at the beginning of my final year project.

7.2.1 Planning & Task Bandwidth

After the initial research and planning stage in semester 1, whereupon receiving feedback that the project could be potentially over-optimistic I kept this in mind as I planned out and developed the features and functionality of my solution.

As suggested by my FYP project implementation supervisor, the "Pareto Principle", otherwise known as the 80/20 rule was followed due to the agile and reactive nature of VR development. In the context of software development it outlines that often 80%

of the consequences or reception by users, good or bad is caused by 20% of either the features or issues in an agile software development context and that it is often wise to identify the most important 20% of a project, and devote 80% of the task bandwidth towards completion as the 20% is the most important part for a user and in this case developing a solution to the research question.

With this rule and past feedback in mind, the final feature-set of the solution was based on time remaining and the bandwidth available around other projects and responsibilities encountered within my 4th year of university. This meant that a solid core functionality was imperative to the final solution presented in this paper.

7.2.2 Difficulties Encountered

While the major difficulties and the approach to solving each potential issue is covered in Section 5.2 - Difficulties. A key issue I had that did not appear previously is the difficulties in finding time to develop new features. With a VR application I often found that multiple steps in development had to pause, or untested features had to pile up until I could test it using my VR headset. This was important during the early and initial stages of development as immersion was an important element to be addressed and could not be tested or worked on until returning home.

While initially this was, with in-person classes this meant my time to work on the project was limited to weekends. With meetings with my supervisor set for a Friday which did fit my schedule better this posed a difficulty in that discussions focused on the FYP revolved around work completed the previous weekend. As the semester progressed, my potential task bandwidth while at home with my VR headset reduced drastically.

When combined with the massive success and time management disaster that was winning "Best Academic Society" in Ireland for "Programming Society", essential time and bandwidth for other modules had to compete with preparation for BICs, meaning projects due near, or just after the reading week had to be completed as a high priority and bandwidth to develop the FYP faced a critical point.

Due to much of the prototype feature-set being developed previous to week 8, a decision had to be made to reevaluate the prospective future feature set and focus on the finalization and polish of the most important 20% required in order to meet the objectives outlined in the solution approach and develop results and a conclusion to the research question.

7.2.3 Skills Learned for future developments

- **Deeper Knowledge of VR development:**
  Knowing what I do now, I would approach a future project differently with a focus on getting 1 core element correct as the focus at the beginning of development.
- **Good hardware doesn't beat optimization:**
  During testing of the prototype with my implementation supervisor, running the project on my laptop, while still a very powerful laptop, netted reduced performance. The solution was to reduce resolution and frame-rate targets. While this did work, reducing the resolution has no major effect on the CPU requirements of an application. This also meant that testing of the android based version of the project was put on hold. In future projects a key focus will be on what performance target needs to be hit from the get go, rather than running into performance and optimization issues too late in the development schedule to redevelop the solution.
- **Time Management:**
  This was incredibly important as outlined above in detail. In a future project like this, even if the length of development time was reduced, having the project be my sole focus for 2 weeks would net similar results than only being able to spend a 6 uninterrupted hours at most a week on a project over 13 weeks, while balancing other commitments and priorities that were shifting unpredictably.

7.3 Future Work

In the future, when revisiting this project there are four main tasks we'd like to further.

7.3.1 Adding range based culling & central platform

By implementing range based culling focused around a central platform, it would be possible to have a larger number of objects on "display" at any time. This is due to the finite resolution of the HMD and vision. In the current version this effect can be seen as the further off objects begin to blur together as the details become smaller than the pixels available to display them with current HMD technology available at a consumer level. Once this distance is determined the objects would no longer need to be rendered.

When combined with a central view platform in the middle of the statistics and the un-queuing of objects from the CPU render queue outside of the users field of view, the amount of objects rendered would be lower but there would be more objects up to an untested but potentially ludicrous amount.

7.3.2 Perspective switching & new locomotion

While not compatible with larger numbers of objects than currently implemented, the ability to view statistics from an aerial perspective either by scaling down or moving to a higher platform in the sky would be a great addition to the solution. A similar perspective switch could also be implemented by adding the ability to fly for instance however this approach would not be friendly to new VR users due to its nauseating abilities.

7.3.3 Large scale testing of Research question

Testing this application with a larger audience in a repeatable way would be highly informative but a testing structure would need to be setup with placebo tests and double blind testing would need to be performed to ensure the quality of the data. A key element is that the less a user knows before being tested in regards to statistics the more accurate the result of the testing would be.

7.3.4 Better name along with branding in order to distribute the application

While not an addition to function, a good and unique name along with custom unity assets to create the world and a redesign of the world along with a proper tutorial and color scheme to make it a cohesive experience that could be distributed via SideQuest, Itch.io or Oculus AppLab to potential new users who might find this application useful. Ideally AppLab and itch.io as without an Oculus developer account it is the only way to install a third-party application. However there is a long approval process that ends in your app only being located if the exact name is entered into the Oculus Store search menu.

7.4 Conclusion

- **Research Question:**
  "How does presence in a large virtual space affect an individual's understanding, comprehension and engagement of large statistics and data when being presented with simpler, more relate-able and procedurally generated methods of visual data representation in a safe environment through the use of an immersive virtual reality application"

From limited user testing and extensive background research we found that the application did invoke the intended reaction. When tested and shown statistics before and after their VR experience their comprehension ability was better and they were able to more approximately guess the amount of objects after their VR experience and they genuinely expressed more interest afterwards in investigating statistics they encountered further.

They found the application simple to use when receiving additional instruction while using it. The tablet menu system was easily grasped and due to the short length required by the experience minimal motion sickness was experienced.

From this we can conclude that the solution application works as intended. A user's presence and immersion within the application did affect their comprehension and engagement with large statistics in a marked way. Furthermore users were engaged with the data and lost track of time. While more engaged users spent time leaning, crouching and observing the data from as many angles as possible before stating their approximation.

However a flaw in this conclusion is that there is no direct comparison against testing if a user would show similar statistical comprehension improvement when simply shown images or tested in a city street, guessing how many users are present at any one time. This comparison would be a key factor going forward in developing new features and then testing the application with a larger audience.

A separate conclusion that can be drawn from the application is that having separate short stages or short term goals, reduced motion sickness. Using the interaction table before using the main feature of the application made users feel more comfortable as they had a better understanding of how the VR environment reacted regardless of the user's age. Time with this table also has to be controlled as we noted that some user's were more interested in spending time with the virtual objects than others.

Bibliography

[1] D. Barnard. (2019) Degrees of freedom. [Online]. Available: https://virtualspeech.com/blog/degrees-of-freedom-vr

[2] Ultraleap, "World-leading hand tracking: Small. fast. accurate." [Online]. Available: https://www.ultraleap.com/tracking/

[3] E. Games. Nanite virtualized geometry. [Online]. Available: https://docs.unrealengine.com/5.0/en-US/RenderingFeatures/Nanite/

[4] K. Group. (2020) Open xr. [Online]. Available: https://www.khronos.org/openxr/

[5] Mda takes mixed reality into orbit with hololens 2. [Online]. Available: https://customers.microsoft.com/en-us/story/1377057233739728271-mda-manufacturing-hololens-canada

[6] Bar chart. [Online]. Available: https://uk.mathworks.com/help/matlab/ref/bar.html

[7] Demo of 3d bar charts. [Online]. Available: https://matplotlib.org/stable/gallery/mplot3d/3d_bars.html

[8] G. Atanasov, "Head-mounted ar/vr for human realistic 3-d data visualization," Dec 2018. [Online]. Available: https://medium.com/telerik-ar-vr/head-mounted-ar-vr-for-human-realistic-3-d-data-visualization-40f570a8a363

[9] Ars-Technica. (2019) How openxr could glue virtual reality's fragmenting market together. [Online]. Available: https://arstechnica.com/gaming/2019/03/how-openxr-could-glue-together-virtual-realitys-fragmenting-market/

[10] S. Hayden. (2021) Oculus quest 2 sales. [Online]. Available: https://www.roadtovr.com/qualcomm-meta-oculus-quest-2-sales/

[11] A. Souppouris. (2016) Htc vive - an oral history. [Online]. Available: https://www.engadget.com/2016-03-18-htc-vive-an-oral-history.html

[12] Degrees of freedom. [Online]. Available: https://developers.google.com/vr/discover/degrees-of-freedom

[13] V. Nanjappan, H. N. Liang, F. Lu, K. Papangelis, Y. Yue, and K. L. Man, "User-elicited dual-hand interactions for manipulating 3d objects in virtual reality environments," *Human-centric Computing and Information Sciences*, vol. 8, pp. 1–16, 12 2018. [Online]. Available: https://hcis-journal.springeropen.com/articles/10.1186/s13673-018-0154-5

[14] A. Grimley. (2018, Jul) Eye tracking, a new era for vr or just another gimmick? [Online]. Available: https://medium.com/kainos-applied-innovation/eye-tracking-a-new-era-for-vr-or-just-another-gimmick-115f1ad7e441

[15] C. Silva, S. Bouchard, and C. Bélanger, "Journal pre-proof children's perception of phobogenic stimuli in virtual reality," *International Journal of Child-Computer Interaction*, 2021. [Online]. Available: https://doi.org/10.1016/j.ijcci.2021.100417

[16] G. L. Gillespie, S. Farra, S. L. Regan, and S. V. Brammer, "Impact of immersive virtual reality simulations for changing knowledge, attitudes, and behaviors," *Nurse Education Today*, vol. 105, p. 105025, 2021. [Online]. Available: https://doi.org/10.1016/j.nedt.2021.105025

[17] Microsoft hololens: Mixed reality technology for business. [Online]. Available: https://www.microsoft.com/en-us/hololens

[18] L. Merino, A. Bergel, and O. Nierstrasz, "Overcoming issues of 3d software visualization through immersive augmented reality," *Proceedings - 6th IEEE Working Conference on Software Visualization, VISSOFT 2018*, pp. 54–64, 11 2018.

[19] M. Cordeil, A. Cunningham, T. Dwyer, B. H. Thomas, and K. Marriott, "Imaxes: Immersive axes as embodied affordances for interactive multivariate data visualisation," *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*, 2017. [Online]. Available: https://doi.org/10.1145/3126594.3126613

[20] ——, "Iatk: An immersive analytics toolkit," *26th IEEE Conference on Virtual Reality and 3D User Interfaces, VR 2019 - Proceedings*, pp. 200–209, 3 2019.

[21] S. J. Cheyette and S. T. Piantadosi, "A primarily serial, foveal accumulator underlies approximate numerical estimation," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 116, pp. 17 729–17 734, 9 2019.

[22] Oculus, "Oculus all in on openxr: Deprecates proprietary apis." [Online]. Available: https://developer.oculus.com/blog/oculus-all-in-on-openxr-deprecates-proprietary-apis/

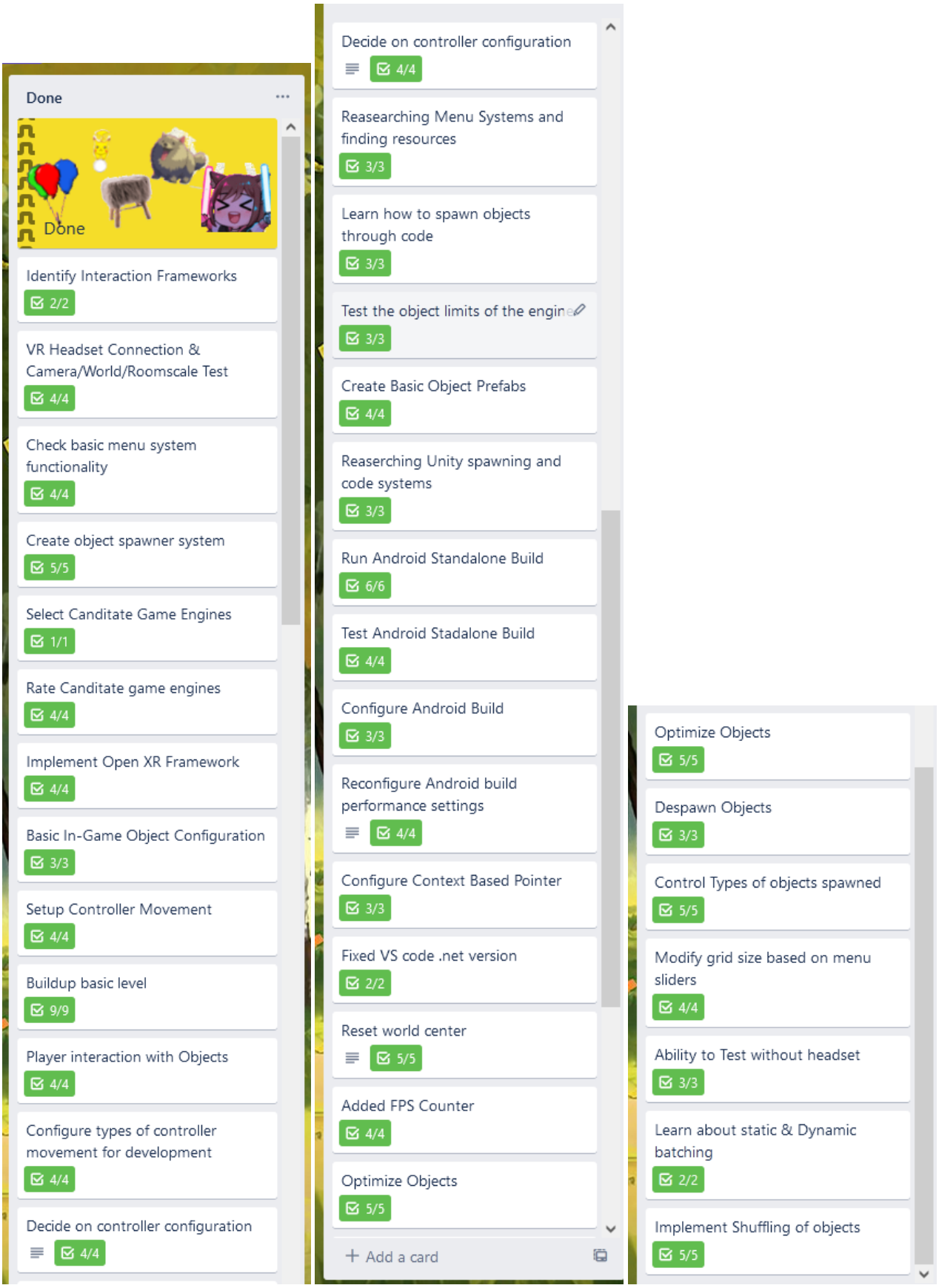
[23] U. Corporation, "Changelog: Xr interaction toolkit: 2.0.1." [Online]. Available: https://docs.unity3d.com/Packages/com.unity.xr.interaction.toolkit@2.0/changelog/CHANGELOG.html

[24] D. D. Georgiev, I. Georgieva, Z. Gong, V. Nanjappan, and G. V. Georgiev, "Virtual reality for neurorehabilitation and cognitive enhancement," *Brain Sciences*, vol. 11, pp. 1–20, 2 2021. [Online]. Available: /pmc/articles/PMC7918687//pmc/articles/PMC7918687/?report=abstracthttps://www.ncbi.nlm.nih.gov/pmc/articles/PMC7918687/

[25] A. Fonnet and Y. Prie, "Survey of immersive analytics," *IEEE Transactions on Visualization and Computer Graphics*, vol. 27, pp. 2101–2122, 3 2021.

Appendix A

Completed Trello Tasks


Done
Done
Identify Interaction Frameworks
2/2
VR Headset Connection & Camera/World/Roomscale Test
4/4
Check basic menu system functionality
4/4
Create object spawner system
5/5
Select Canditate Game Engines
1/1
Rate Canditate game engines
4/4
Implement Open XR Framework
4/4
Basic In-Game Object Configuration
3/3
Setup Controller Movement
4/4
Buildup basic level
9/9
Player interaction with Objects
4/4
Configure types of controller movement for development
4/4
Decide on controller configuration
4/4
Decide on controller configuration
4/4
Reasearching Menu Systems and finding resources
3/3
Learn how to spawn objects through code
3/3
Test the object limits of the engine
3/3
Create Basic Object Prefabs
4/4
Reaserching Unity spawning and code systems
3/3
Run Android Standalone Build
6/6
Test Android Stadalone Build
4/4
Configure Android Build
3/3
Reconfigure Android build performance settings
4/4
Configure Context Based Pointer
3/3
Fixed VS code .net version
2/2
Reset world center
5/5
Added FPS Counter
4/4
Optimize Objects
5/5
Add a card
Optimize Objects
5/5
Despawn Objects
3/3
Control Types of objects spawned
5/5
Modify grid size based on menu sliders
4/4
Ability to Test without headset
3/3
Learn about static & Dynamic batching
2/2
Implement Shuffling of objects
5/5