

Exploring Gesture Recognition in the Virtual Reality Space



UNIVERSITY OF
LINCOLN

Marlon Gilliam

School of Computer Science

University of Lincoln

Submitted in partial satisfaction of the requirements for the
Degree of Master of Science by Research
in Computer Science

Supervisor Dr. Chris Headleand

January, 2019

Acknowledgements

First and foremost I would like to thank my supervisor Dr. Chris Headleand. Chris has been a mentor throughout much of my academic learning and his support through out this degree was integral to its completion.

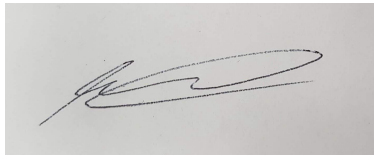
I would like to thank all my family and friends for their support throughout my academic studies. They may not understand it but they sure listened to it well without too many complaints! A special mention here must be given to the member of the Int-Lab group to whom I say “Sooo, Dooo”.

A special thank you goes to one small chicken, without whom I would certainly not have been able to keep up motivation and focus through this degree. Her love and support were endless and she was always supportive of my late nights working (provided I stopped at a certain fast food restraunt on the way home).

Statement of Originality

The work presented in this thesis/dissertation is entirely from the studies of the individual student, except where otherwise stated. Where derivations are presented and the origin of the work is either wholly or in part from other sources, then full reference is given to the original author. This work has not been presented previously for any degree, nor is it at present under consideration by any other degree awarding body.

Student:

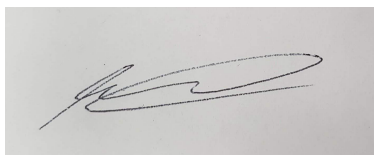


Marlon Gilliam

Statement of Availability

I hereby acknowledge the availability of any part of this thesis/dissertation for viewing, photocopying or incorporation into future studies, providing that full reference is given to the origins of any information contained herein. I further give permission for a copy of this work to be deposited to any repository authorised for use by the University of Lincoln. I acknowledge that the University of Lincoln may make the title and a summary of this thesis/dissertation freely available.

Student:



Marlon Gilliam

Abstract

This thesis presents two novel modifications to a gesture recognition system for virtual reality devices and applications. In doing this it evaluates users movements in VR when presented with gestures and uses this information to develop a continuous tracking system that can detect the start and end of gestures. It also expands on previous work with gestures in games with an implementation of an adaptive database system that has been seen to improve accuracy rates. The database allows users to immediately start using the system with no prior training and will improve accuracy rates as they spend more time in the game.

Furthermore it evaluates both the explicit and continuous recognition systems through user based studies. The results from these studies show promise for the usability of gesture based interaction systems for VR devices in the future. They also provide findings that suggest that for the use case of games continuous system could be too cumbersome for users.

Contents

1	Introduction	1
1.1	Motivation	2
1.2	Research Questions	4
1.3	Objectives	5
1.4	Thesis Overview	5
1.5	Contributions	6
1.6	Conclusion	8
2	Related Work	9
2.1	Video Games	9
2.1.1	Gestures in Games	10
2.1.2	Virtual Reality Games	10
2.2	Gesture Recognition	12
2.2.1	Gestures as Human interaction	12
2.2.2	Recognition Algorithms	13
2.2.2.1	Dynamic Time Warping	14
2.2.2.2	Support Vector Machines	14
2.2.2.3	Neural Networks	15
2.2.2.4	Linear Classifiers	16
2.2.2.5	Hidden-Markov-Models	16
2.2.3	Continuous Recognition	18
2.3	Validation	20
2.3.1	Gesture Validation Methods	20
2.3.2	Coordinated Multiple Views	21
2.4	Conclusion	22
3	Gesture Recognition System	24
3.1	Motion Tracking	24
3.2	Gesture Classification	25
3.3	HMM Topology	26
3.4	Gesture Storage	27
3.4.1	Database Adaption	28
3.4.2	Implementing the Adaptive System	28
3.5	System Testing and Results	29
3.6	The Commands	31

3.6.1	Testing Military Gestures	32
3.7	Continuous Recognition	33
3.8	Summary	34
3.8.1	The Four Versions	35
3.8.2	Unity Implementation	36
3.9	Conclusion	36
4	User Interaction with VR Gestures	38
4.1	Design and Implementation	38
4.1.1	Game Testing	40
4.1.2	Gestures	41
4.1.3	Shooting	42
4.1.4	Targets	43
4.1.5	Scoring	44
4.1.6	Logging	45
4.1.7	Analysis	46
4.2	Evaluation	49
4.2.1	Data Collected	49
4.2.2	Methodology	50
4.2.3	Results	51
4.2.4	Discussion	54
4.3	Conclusion	56
5	Gesture Recognition Game	57
5.1	Study Design	57
5.2	User Testing	59
5.3	The Game	59
5.3.1	Level Design	59
5.3.2	Movement	61
5.3.3	Shooting	62
5.3.4	Artificial Intelligence for Non-Player Characters	63
5.3.4.1	Animations	64
5.3.4.2	Enemy AI	65
5.3.4.3	Way-point System	66
5.3.4.4	Target Selection	67
5.3.4.5	State Machine	67
5.3.5	Friendly AI	71
5.3.5.1	Cover Point System	72
5.3.5.2	State Machine	72
5.3.5.3	Receiving Commands	73
5.3.6	Score	74
5.4	Tutorial	74

5.5	Conclusion	75
6	Evaluation of Gesture Recognition Game	76
6.1	Data Collected	76
6.2	Methodology	77
6.3	Results	79
6.3.1	Observed Behaviours	82
6.3.2	Thematic Analysis	83
6.3.2.1	Immersion and Presence	84
6.3.2.2	Control Differences	85
6.3.2.3	Improved Tutorial	86
6.3.2.4	Practice Methods	87
6.3.2.5	Customising Gestures	87
6.3.2.6	More Unique Gestures	88
6.4	Discussion	89
6.5	Conclusion	91
7	Conclusion	92
7.1	Thesis Summary	92
7.1.1	Introduction	92
7.1.2	Literature Review	93
7.1.3	Gesture Recognition System	93
7.1.4	User Interaction With Gestures in VR	94
7.1.5	Design and Implementation of a Gesture Recognition Game	94
7.1.6	Evaluation of Gesture Recognition Game	95
7.2	Discussion	96
7.3	Limitations and Future work	98
	References	101
8	Appendices	107

List of Figures

2.1	An example of symbolic hand gestures [25].	12
2.2	An example of hand gestures with a trajectory [26].	13
3.1	The gestures used in the game visualised by their representations from the military gestures documentation[84] (follow, rush, sneak, cease firing, start firing).	31
4.1	An overview of the level for the game.	39
4.2	40figure.caption.10	
4.3	The number 5 being drawn in the game. Shown from the players view perspective (player has a wider view in VR).	42
4.4	One gun fully visible, one half way through dissolve in animation.	42
4.5	A target in the game being shot. The score reward can be seen in the top right of the image (+300).	44
4.6	The class map for the data model that we are logging.	45
4.7	A view of our csv before the implementation of markers for gesture start and end (markers manually added with yellow dashes).	47
4.8	An example of the heatmap view with the game background shown behind it to give context to the hand positions.	48
4.9	An example of the CSV displaying the trace of the players head movement. The high peaks in the line chart are from controller connection spikes which occur occasionally but have no impact in game.	48
4.10	An image depicting a participant starting their gesture without pre-positioning their hand.	52
4.11	An example of the CSV displaying a user moving their hand to a starting position before beginning their gesture.	52
4.12	An example of the CSV displaying a smaller gesture.	53
4.13	An example of the CSV displaying a gesture that ends with trailing motion.	53
5.1	An overview of the level for the game.	60
5.2	An example of a room from the game, populated with enemies and objects for decoration and cover.	61
5.3	An example of teleport points in game. The blue one is being targeted by the player.	62

5.4	Visual for the guns when they are disabled from shooting.	63
5.5	A series of animation steps from the dying animation.	65
5.6	The enemy in the idle position.	68
5.7	The enemy in the crouched (cover/sneak) position.	69
5.8	The enemy in the shooting position.	70
5.9	The enemy in the crouched shooting position.	70
5.10	An overview of the tutorial level. The player moves from right to left.	74
6.1	Correlation between accuracy and number of gestures	80

List of Tables

3.1	Accuracy rates for user dependent database. Top axis defines number of states in the HMM and left axis defines the size of the training database.	30
3.2	Accuracy rates for initial single user database.	32
3.3	Variations of gesture recognition system.	35
4.1	Number of hours spent playing games per week for each participant.	51
4.2	Average gesture times in frames and seconds	54
6.1	Average hours of games played per week. Gender left and hours top.	79
6.2	Accuracy results for all systems	79
6.3	Confusion matrix for explicit static system	80
6.4	Confusion matrix for explicit adaptive system	81
6.5	Confusion matrix for continuous static system	81
6.6	Confusion matrix for continuous adaptive system	81

Chapter 1

Introduction

This chapter covers the general subject area of the thesis and introduces the main concepts and paradigms that will be discussed and researched throughout the thesis. It briefly covers the current state of Virtual Reality and gesture recognition before covering our motivations as to why we are exploring the use of them in tangent. It outlines our research questions and the underlying objectives which we will use to answer them. It concludes with an overview of the thesis and the contributions that it has to the field of research.

Virtual Reality (VR) broke into consumer markets with the release of the *Google Cardboard*¹ and the *Gear VR*² both being consumer grade devices that were mostly inexpensive and could be used with a variety of mobile devices. Later higher performance headsets made their way to the market with the *Oculus Rift*³ and the *HTC Vive*⁴ both being capable of high definition VR when powered by high end computers. With this their capabilities are increasing dramatically and in tandem their interaction methods are also advancing. Presently both the Oculus and the Vive have motion controllers as their main input device. Alongside this, the newer versions *Gear VR* and the *Google Daydream*⁵ are also equipped with accelerometer based motion controllers similar to the *Nintendo Wiimotes*⁶ that have previously been used in gesture based research [1], [2].

¹Google, <https://vr.google.com/cardboard>, 2014

²Samsung, <https://www.samsung.com/global/galaxy/gear-vr>, 2015

³Oculus VR, <https://www.oculus.com>, 2016

⁴Valve, <https://www.vive.com/uk>, 2016

⁵Google, <https://vr.google.com/daydream>, 2016

⁶Nintendo, <https://www.nintendo.co.uk>, 2006

Games are at the forefront of the current consumer adoption of VR and with this has come a push into research for best practices. Presently many games are taking advantage of the physicality that is allowed by the world space tracking of the motion controllers. Products are also being developed that can be used in tandem with VR devices as external input devices. The *Leap Motion*⁷ is a device that can be used to track hands using cameras. The Leap has been used as a control input for VR games and can allow users to have the freedom of movement that the motion controllers do but without the burden of holding a controller.

Motion gestures are used in many everyday applications within a wide range of devices including mobile phones and some game controllers. They have been researched extensively and have been applied to many areas of Human-Computer-Interaction. Many modern devices make use of motion gestures to include either novel or useful additional features to devices, ie. smart-phones turning screens on. Within the space of games these inputs have been used as both the complete control method as well as additional or optional control inputs like the *Playstation Dualshock 3 and 4*⁸ which both featured motion control.

1.1 Motivation

Research is constantly exploring new ways in which VR interaction can be improved. These approaches range from complex custom devices for use in serious applications [3] to novel approaches at creating interesting game experiences [4]. With the current industry standard being motion controllers there is the possibility that these controllers like the Wiimote can be used within gesture recognition systems.

Previously games that have used gesture systems as their control inputs have had them as the only way in which users can interact with the game [4]–[6]. Since VR devices already use motion controllers as their input devices it is possible to use gestures as an input method without removing or converting existing controls.

⁷Leap Motion, <https://www.leapmotion.com>, 2013

⁸Sony, <https://www.playstation.com/en-gb>, 2006

This leaves them as a feature that could theoretically be added to any existing VR game as an additional method of control and interaction.

Gesture recognition is a widely researched field in the area of Human-Computer-Interaction and it has previously been explored in games in a number of cases [1], [4], [7]. Gesture recognition is also present in commercial games such as the *Just Dance*⁹ franchise where users poses are used to recognise elements of the dance. We believe that VR games have the potential to benefit from gesture interaction systems and that the motion controllers the headsets ship with are suited to recording gestures.

A significant area of consideration in gesture recognition is the challenge of user independent recognition [8]–[10]. There exists much literature that considers both user dependent data and user independent data and generally results show a significant decrease in accuracy for independent recognition. More recent work [8], [11] show independent accuracies comfortably over 90%. However, the methodologies for these studies have participants in an isolated environment when performing gestures. We aim to show that within the context of a VR game user's movements can be confused by other cognitive demands from the game leading to lower recognition accuracy.

Continuous recognition is another field that is often considered within gesture recognition. The area has seen many novel approaches but no concrete method has been presented. Presently approaches aim to solve the problem for a specific use case or platform in some more generic cases. Since the nature of gestures is versatile it makes sense that problems in the nature of continuous recognition use case specific solutions to attempt to produce stronger results.

It has been considered previously that gesture recognition systems should be "flexible and expandable in order to maximise efficiency, accuracy and understandability." [12]. With this in mind we propose a method discussed by [1] that adapts an independent database as users interact with the system.

⁹Ubisoft, <https://www.ubisoft.com/en-gb/game/just-dance-2019/>, 2009

Our system achieves high levels of recognition accuracy competitive with the best performing existing system whilst requiring minimal user independent training.

Our motivation for this thesis is to design an Adaptive Gesture Recognition system that can be used with a variety of consumer virtual reality devices. We explore and evaluate the use of the system in a game specific environment and determine a base level for the accuracy of the system. Our evaluation methods look to previous research and their solutions to evaluate the effectiveness and viability of our system. We produce a system that can be used to further explore the area of gesture interaction in VR. As well as this we establish that gesture recognition is a feature that should be considered moving forward in the virtual reality space. Furthermore we explore the space of continuous recognition by developing a continuous version of our system along side it. We will use the same tests on both and evaluate how the continuous system matches up to the explicit in a game environment. We evaluate these systems in real world game circumstances as opposed to isolated environments which has allowed us to determine their viability as systems for commercial products.

In doing this we explore the way in which users interact with VR games. From this we provide an insight into the ways in which they do this in such a manner that the knowledge can be used in the design of these systems. We also gather user opinions on both the usability of the systems and their experience in the games. We find that gesture interaction is a feature that users see potential in as a control method for VR.

1.2 Research Questions

This document aims to address the following research questions:

- Are there patterns in the motions of users when interacting with a gesture system in a VR game?
- How do users engage with an optional gesture system within a larger VR game?

- How does a continuous recognition system perform within a VR game?
- Can an adaptive database system improve the accuracy of a gesture interaction system?

1.3 Objectives

To complete the requirements described above we will:

- Review a body of related works and identify limitations and areas for expansion
- Build and test an artefact that can provide insight into user interaction with gestures in VR
- Design and build a gesture recognition system that can also feature an adaptive database system
- Design and build a continuous gesture recognition system on top of the original system
- Validate results of all above systems within the context of a VR game
- Perform analysis upon these results and make suggestions as to the feasibility of these systems

1.4 Thesis Overview

Chapter 2 reviews related work on the topics of VR interaction and Gestures in games before moving on to gestures as a means of human computer interaction. It then discusses gesture recognition algorithms and methods including continuous recognition.

Chapter 3 presents the design and technical implementation of our gesture recognition system. It covers the technical layout of the recognition model and its data types, followed by our initial testing methods.

Chapter 4 introduces our exploration into player movements in VR shooting games. It presents our design and implementation of the system; followed by an analysis and discussion of the results on this study.

Chapter 5 documents the design and implementation of our full VR gesture interaction game. It uses the base implementations of our model and game from chapters 3 & 4 and discusses how these were furthered to better suit the system.

Chapter 6 discusses the results and evaluation of this game comparing accuracy rates and user experience from a number of sources.

Chapter 7 concludes our findings and discusses what effects our results have on the areas of gesture interaction and game design.

1.5 Contributions

The contributions from this thesis are as follows:

- The design for a gesture recognition system that has been proven to work with VR devices
- The design for an adaptive database solution for building gesture databases
- Data on the movement patterns of users performing gestures in a VR game
- A design for implementing a continuous recognition system into a gesture system
- User experience and usability results for a VR game featuring gesture recognition
- User experience and usability for a VR game featuring continuous gesture recognition

This thesis contributes to the areas of gesture recognition and virtual reality interaction, in the field of computer science. The first and most obvious contribution is the design of a gesture recognition system that can be used with multiple VR devices, described in 3.3. The system is designed and presented

in a way that allows for it to be replicated for future works. There is potential for the system to be used as a basis for VR games that want to use gesture recognition in both research and commercial fields.

In tandem with the gesture recognition system an adaptive database solution is presented which we have shown in this work to provide significant accuracy increases to the system. The design of the database solution is described in detail in 3.4.1 where we explain the different ways in which the system can be implemented in order to best match it to a specific system. The effect of the database was one of our research questions that came about from reviewing works in games that had used similar technologies [1].

After deciding to design a continuous recognition system it was decided that research should be carried out into the movement data around the gestures. This was so it could be analysed and searched for patterns that could be used in the development of the system. The basis for the continuous system was based on an method that was used in a system found during the literature review [13]. This system used patterns in the motion data of the gestures to segment and identify them for recognition. The reason for collecting the motion data from our users was the lack of research into the area in VR; spurring us to confirm that similar patterns existed in the medium. The data we contribute provides a general view as to motion patterns that emerge as users perform gesture in VR. We do make note that this data is subject to context and that, as it recommended with all gesture systems, specific use cases are considered and may require their own analysis.

Using the results of the investigation into user movement patterns when performing VR gestures a continuous recognition system was developed. The system was built on top of the previously described gesture recognition system and was developed to work in a similar manner to the “Zero Codeword” system which we evaluated in our literature review [13]. As with the explicit system the continuous systems development is described in detail that allows for the system to be replicated for use in future research or commercial applications.

With both our gesture recognition systems we perform a large user study in which participants played a complete virtual reality game featuring each of the systems. From this we gathered both quantitative and qualitative data on their experience within the game and their perceptions on the usability of the systems. Our findings show that the explicit gesture recognition system was generally perceived to be usable in our VR game however the continuous system was not received quite as well. Our accuracy results for the two systems show them both to have potential, especially with additional tweaking for their specific use cases. Through the thematic analysis of interview data recorded after the studies we build a selection of discussion topics with regards to potential improvements and points of reflection in both the gesture recognition systems and their integration into games.

1.6 Conclusion

In this introductory chapter we have presented an outline for the research area that we cover in this document. We outline gesture recognition and virtual reality games as our main focal points and go on to explain our motivation in undertaking this research. We outline the research questions which the document aims to address and the research objectives which we have outlined as methods for answering these questions. We then go on to briefly summarise the content of the thesis. Finally we outline the contributions that the work makes to the greater research area and give a brief summary of how and where these contributions are made.

Chapter 2

Related Work

In this section we undergo a review of a variety of literature that is related to the topics within this thesis. We cover a wide variety of topics as multiple research areas were covered but the focus of the review will be on the challenge of gesture recognition and its use in video games. We cover several gesture recognition methods in detail and evaluate them in regards to their suitability for our desired solution. The current methods of interaction with virtual reality are explored and the position that our system would fit into is established. We also cover Coordinated-Multiple-Views (CMV's) and how they can be used to analyse data and help transform data in such a way that patterns are human recognisable. This chapter outlines the current state of the research areas which we will be exploring.

2.1 Video Games

With the resurgence of virtual reality for consumer markets a number of different application types have found a home on the devices. Video games are a common use for VR in commercial context and as such there is much ongoing research aimed at finding the best practices for creating VR games. In this section we cover a brief outline of some of the current research relating to VR games that is most relevant for our project.

2.1.1 Gestures in Games

The *Nintendo Wii*¹ brought motion controllers to mainstream gaming and with this the possibility of accelerometer based gestures games. Wiizards [1] is a game that allows users to cast spells by performing gestures using the Wii remotes (Wiimotes) accelerometer to track the movement and a Hidden Markov Model (HMM) to classify the gesture. Their work builds on that of Payne et al. [14] who, prior to the Wii's release, developed a similar game with a simpler gesture set. Both works conclude that the use of gestures in games works well and should be further explored. Payne et al. interestingly conclude that different games could benefit from different levels of teaching and feedback for the gestures; suggesting that for their game a cheat sheet allowing users to constantly see how to perform gestures could have improved their experience.

Cheema et al. [5] developed a game that used 25 different gestures to control all in game actions. Similarly to the earlier work from Kratz et al. [1] they use a Wiimote to track the users hand motion. They achieve very impressive accuracy results for a larger database but conclude that for commercial use approaches that feature multiple classifiers would likely be necessary. The Invoker [6] is an Role Playing Game (RPG) that uses the *Microsoft Kinect*² for its control input. It combines regular movement with gestures in order to allow for combinations of moves to be done in the game.

2.1.2 Virtual Reality Games

Virtual Reality has throughout its life been exposed to a series of novel and expansive range of control methods. Theoretical use of gestures with VR devices was shown from Weissmann and Salomon with their system capable of gesture recognition using a data glove with a VR headset [15]. Their work achieved good recognition accuracy using a neural network to connect the data points on the glove and recognise several static hand poses. Early works from Maki et

¹Nintendo, <https://www.nintendo.co.uk>, 2006

²Microsoft, <https://www.microsoft.com/en-gb/>, 2010

al. [16] explore the use of several virtual instruments using a variety of gloves and magnets as controllers for interaction. Cabral et al. [17] use a rudimentary image processing gesture system to allow users to interact with a "Powerwall environment" room. Experimental Virtual reality interfaces are often seen in the medical research fields using novel methods like pneumatic gloves to improve immersion [3].

Liu et al. [18] developed a rudimentary system that was capable of tracking and recognising 3 different hand gestures using image processing. They explored the use of this technology within a VR environment in a similar way to how the modern *Leap Motion* device is often used with VR [19]. Khundam [19] explored the how the leap motion could be used to track the user's hands and then recognise gestures that are used to move the player in VR space. Duke is an experimental game in development that combines the Oculus Rift with both the leap motion and Kinect [4]. It leverages the Kinect for movement by tracking the users legs and the leap motion for combat controls by tracking the users hands. A work in progress game from Lee et al. [20] is developing a data glove that can be used in the place of the motions controllers for VR. They propose the use of gestures with the glove instead of button presses with a controller; the given example is clutching a sword by making a fist with the glove compared to pressing and holding a button on the controller.

It has been shown to be possible for full human models to track users movement in VR using inverse kinematics and animation blending [21]. Considering the positive effects that multi-player play in VR has been seen to have [22] and the commercial success of the game *VR Chat*³, there is a desire for users to be able to see and interact with each other in VR. Considering gestures as a tool for interaction within the game they would also be easily visible to other players as they can see and recognise the players movements.

³VRChat Inc, <https://www.vrchat.net/>, 2017

2.2 Gesture Recognition

Gesture recognition is a commonly used tool across a many areas in human-computer interaction. It is presently making changes to the way that we use many of our devices including smartphones, wearables and gaming consoles [23]. Due to its widespread use, a large number of techniques have been developed that can be used in the classification and recognition of gestures. In the following section we provide a condensed review of these techniques focused around those we find most prominent and relevant to our research. Prior to this we consider gestures from a more human perspective, reviewing works that look at how gestures are/ can be used within the field of human computer interaction.

2.2.1 Gestures as Human interaction

Gestures can be defined as "expressive, meaningful body motions involving physical movements of the fingers, hands, arms, head, face or body with the intent of: 1) conveying meaningful information or 2) interacting with the environment" [24]. Mitra et al. [24] continue by segmenting gestures into three broad categories: hand and arm, head and face, and body gestures. Much research is focused on the area of hand and arm gestures as these are the ones that can be used more prominently within Human Computer Interaction (HCI).

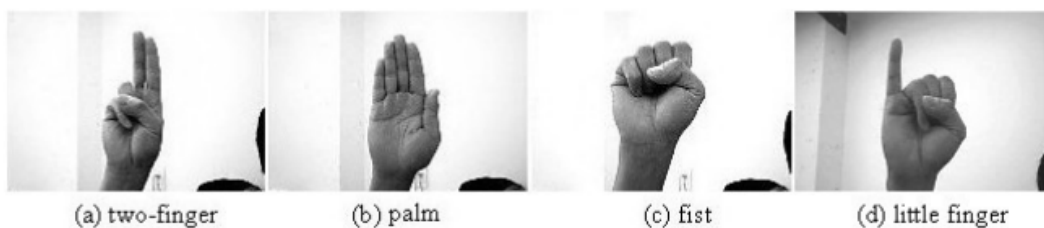


Figure 2.1: An example of symbolic hand gestures [25].

Early studies categorised hand gestures into either static or dynamic based off their spatial movement over time [27], [28]. For hand and arm, static gestures are comprised of specific hand shapes or poses such as the symbols depicted in figure

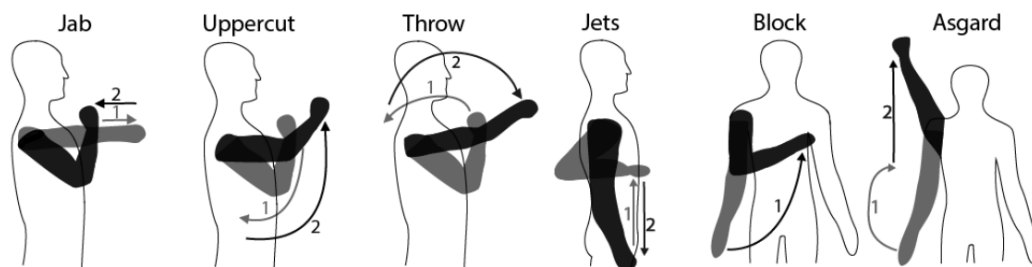


Figure 2.2: An example of hand gestures with a trajectory [26].

2.1 whilst dynamic gestures are depicted as hand motion patterns as depicted in figure 2.2.

Considering the traditional definition of a semaphore; Quek et al. [27] defines semaphoric gestures: “By extension, we define semaphoric gestures to be any gesturing system that employs a stylised dictionary of static or dynamic hand or arm gestures”. They go on to suggest that “Semaphoric approaches may be referred to as “communicative” in that gestures serve as a universe of symbols to be communicated to the machine.”

Semaphoric gestures make up only a very small part of human gesture interaction [27], and have also been said to provide very little functionality and not natural [29]. Despite this they are commonly seen as the gesture method of choice in many studies along with more definite symbols such as letters and numbers.

2.2.2 Recognition Algorithms

As it stands presently there are several algorithms that are currently in use for the recognition of gestures across various systems. Below we provide an explanation and brief review of several of the most commonly used recognition algorithms. We find from our research that a Hidden Markov Model (HMM) is the most appropriate algorithm for our recognition due to its reliability and accuracy with accelerometer system as well as its proven use in games.

2.2.2.1 Dynamic Time Warping

The DTW algorithm works by calculating the distance between each possible pair of points between two signals and uses the distances to calculate a cumulative distance matrix [30]. It then finds the least expensive path through this matrix. It usually features normalised signals and only one-dimensional series [12]. DTW techniques can achieve high accuracy results [10], [31] with minimal training sets but their accuracies are affected by amplitude variation [8] which is often an issue when dealing with accelerometer data due to tilt [31].

Improvements have been made to the accuracies of DTW in recognition on Kinect data by using feature weighting [32] which was further explored and improved by [33]. Lu et al. [34] show a DTW algorithm achieving working with a set of 19 gestures. The system does have lower accuracies than others in the field but its fast computation time on a low power device is impressive considering the larger gesture set.

2.2.2.2 Support Vector Machines

A Support Vector Machine (SVM) works by separating the data classes into hyperplanes in a high dimensional space. It then find classifications from unseen instances in the hyperplanes [35]. SVM's are known to provide great classification performance [36] with low run time computational costs [8]. However SVMs are usually used for static gesture recognition and also require large amounts of training time [12] with large datasets containing high class feature counts to gain greater accuracy [8].

Rashid et al. [37] used both a HMM and SVM together to build a recognition model where the SVM classifies the static poses and the HMM is used to recognise the trajectory of the hand. More recently [38] used both Kinect and Leap Motion sensors in tandem to achieve good accuracy on static hand gestures. Wu considers the performance of an SVM using temporal data with an accelerometer and shows it achieving high accuracy rates in user dependent and independent cases [36],

however the system uses data sets in the thousands for training. He uses *feature fusion* to achieve high accuracy results with mobile accelerometers [39]. Porzi et al. use an SVM on mobile devices to detect gestures using an accelerometer from a smart watch [40]. They achieve high accuracy rates but at times that limit recognition times to that of 1/8th of a second.

2.2.2.3 Neural Networks

Artificial neural networks (ANNs) are statistical machine learning systems that are based on biological neural networks. They consist of networks of neurons across several layers. They are structured in such a way that the first layer is an input layer which sends signals through to subsequent layers. This continues until the last layer which will output a value. Before the system is trained the topology of its input and output must be decided. Typically the input is dependent on the dimension of the feature set and the output is determined by the number of classes to be recognised [41]. There exist many different variations of ANNs of which several can be seen over various areas of gesture recognition.

Time delay neural networks (TDNNs) have been used for the detection of 2D motion trajectories [42]–[44] as well as the segmentation of video data based on skin colour recognition [45]. TDNNs are multi-layer networks that slide over small windows of data in a temporal domain making local decisions; this allows for them to be used in continuous recognition. [12], [46].

Multi-Layer Perceptron (MLP) [47] models can be used to distinguish between non-linearly separable data; such as in [48] where they are used in the real time recognition of static Brazilian sign language. They are also used in the recognition of accelerometer based gestures for culturally specific interactions [49] where they achieve comparable accuracy results in user dependent cases.

2.2.2.4 Linear Classifiers

The Adaboost algorithm is a machine learning meta-algorithm [50] that is used as the base recognition system in the Kinect [51]. The Adaboost algorithm can be used to combine multiple weak classifiers in order to create a strong overall classifier that can achieve impressive results [25]. However in comparison to other recognition methods it has been seen to be less accurate [52]. Hoffman et al. go on to discuss how this result was unexpected due to the sophistication of the classifier.

Hoffman et al. [52] show a linear classifier based on Rubine's method [53] outperforming the Adaboost classifier. Lu et al. [34] show how a Bayes linear classifier can be used in the calculation of smaller gestures on a mobile accelerometer device.

2.2.2.5 Hidden-Markov-Models

Hidden Markov models are doubly stochastic processes that are governed by an underlying Markov chain with a finite number of states; where each state is associated with a set of random functions. The model represents a number of states that move through series of time steps. Each state can be described by its transition probability and its output probability. At each time step the sequence can either output the value from its alphabet associated with the current state or transition to the next state. The model can be formally described as follows [54]:

- A set of N states, denoted as $S = \{S_1, S_2, \dots, S_N\}$.
- A set of M possible output symbols, denoted as $C = \{C_1, C_2, \dots, C_M\}$.
- An $N \times N$ matrix of the transition properties for states, denoted as $A = \{a_{ij}\}$.
Where a_{ij} is the probability of state S_i transitioning to state S_j .
- An $N \times M$ matrix of observation probabilities for each state, denoted as $B = \{b_{ij}\}$. Where b_{ij} is the probability of state S_i outputting symbol C_j .
- An initial state distribution vector, denoted as $\pi = \{\pi_1, \pi_2, \dots, \pi_N\}$.

Previous to its role in gesture recognition the HMM was used in speech recognition [54], [55]. It was then introduced as a method for gesture recognition [56] where it was used to recognise the numbers 0 to 9, drawn in 2D with mouse input, with remarkable accuracies of up to 99.78%. Following this, vision based gesture recognition was achieved on the American Sign Language [57] with impressive accuracies. They noted some important findings in that often with continuous detection, gestures would frequently double up if the path was of a certain manner. The possibility of power point presentation control was explored [58]. They achieve strong accuracies in line with previous research using vision techniques for tracking whilst managing to recognise gestures continuously using a method of end point detection and the Viterbi algorithm [59] for backtracking. Numbers and letters from the English sign language were recognised in [37] using a HMM to recognise trajectory data and an SVM for the recognition of hand postures. Their high accuracies highlight the strengths of both recognition methods and give a good example of how they can be used in tandem. Premaratne et al. [60] recognise 16 letters from the English Language. They implement an initial segmentation using direction from origin to categorise gestures into 5 groups based off the stroke directions of: up, down, left, right, and general diagonal. Their system achieves good accuracy and provides the benefit of lowering computation times. Truong and Zaharia [61] present works that consider the possibility of a HMM recognition system to recognise full body gestures from a Kinect system consisting of poses defined by Laban Movement Analysis. Their system is capable of competitive accuracies on complex gestures with a good real time recognition performance. However, the system requires a significant training time (23m 45s) on a set of 12 gestures; while this may not be an issue considering the possibility of offline training it does not coincide with dynamic adjustment of the training data.

In 2004 the XWand [62] introduced the idea of physical UI devices being used for gesture recognition using data from accelerometers. The XWand was tested using: a Linear Time warping recognition method; a Dynamic Time Warping method and a HMM system which outperformed the other two systems by a significant factor. Following this further proof of HMM working with accelerometer data was presented in both [63] and [64]. They introduced a method of training using noise models to produce variations of a single gesture sequence that could then build a

user dependent database. Mantyjarvi et al. [63] prove that when using vector quantization on the accelerometer data with the k-means algorithm a value of k being 8 is best for 2D data. Kela et al. [64] showed that there was user interest in gesture control for home electronic devices and established a competent 3D accelerometer system to recognise them.

HMM's were proven to work successfully with the user dependent accelerometer data recorded from the Nintendo Wiimote in [2] with user trained data sets. Schlomer et al. [2] follows Mantyjarvi's work on quantization finding k=14 to be suitable for 3D accelerometer data. Furthering this [11] achieves near perfect accuracy results for explicit recognition of gestures in user dependent and independent testing, once again using a Wiimote. Their work consists of a larger degree of gestures. Criticisms have however been made to their findings as they only use gestures "confined to a 2D planar motion" [26]. Where Arsenault's [26] findings show strong accuracy results for his self defined complex gestures, the set size is much lower than both the gesture sets used by Chen et al. [11]. In recent work Whitehead [65] demonstrates significant accuracy on a wearable accelerometer for a game application. They translate the 3D accelerometer data into a single data string using a cubic bisection method to outline positions. They consider how users have a desire to improve their use of the system therefore wanting to learn the gestures and how this matches a game environment where users generally strive for improvement. Interestingly for the context of game use they notice that often the accuracy rates are highly user dependent depending on the skill and memory of the user.

2.2.3 Continuous Recognition

The majority of previously discussed gesture recognition systems have the user dictate when a gesture starts and sometimes when it ends as well. This allows for easy segmentation and extraction of the gesture data but has been said to be less user friendly [66]. In contrast to this some systems attempt to make the detection of gestures completely devoid of user input aside from the actual pose or motion

of the gesture. Since this is quite a “fuzzy” problem in terms of logic many of the works feature novel and often very case specific methods for achieving this.

In early works Jung-Bae et al. present a system that is capable of continuously recognising 18 words in Korean sign language sentences making use of a HMM for classification [67]. They use changes in angle of trajectory to separate the data into different HMMs allowing for complex symbols to be recognised using simple individual gestures. They then use a difference in the speed of the hand to detect the start and end of words. The *Zero-Codeword Detection System* uses a similar system to separate the recognition of different numbers [13]. The system uses a minimum number of pixels per frame to calculate the static velocity and uses a minimum threshold on this value to determine the start and end of gestures. The *Zero-Codeword Detection System* shows improved accuracy rates using a simplified system but on a smaller data set and in a static environment. In their discussion Jung-Bae et al. consider how due to the fuzzy logic of the problem systems that attempt to recognise more complete data sets might struggle.

Raffa et al. [66] show a system that is capable of achieving continuous accuracy results that are equal to explicit ones. They achieve this on low budget hardware using accelerometers and discuss how it has the ability to save battery life on devices using their HMM garbage model; which works by discarding many non valid gestures before they reach the intensive computational stage of recognition within the HMM.

Evangelidis et al. [68] manage to achieve strong accuracy results using multiple SVM outputs to continuously recognise pose gestures using skeletal data. In a similar fashion Yin and Davis implement a continuous gesture recognition system to detect poses from skeletal data from the Kinect [69]. They use this system in combination with a discrete program that detects trajectory based gestures from the same information. Their findings show the continuous system being much less reliable in terms of accuracy, however this could be due to the difference in data that they are attempting to classify. Several recent works explore the possibility of using neural networks to tackle continuous recognition of the *ChaLearn* [70] data set which contains 47933 examples of 249 different gestures [71]–[73]. While

the accuracies of these works are not in the same range as many smaller scale recognition systems they provide strong grounding for how large scale continuous recognition could be solved in the future.

2.3 Validation

2.3.1 Gesture Validation Methods

There are many techniques used to assess gesture recognition systems. The most straight forward being simply taking averages for accuracy results across all gestures to get a big picture evaluation of the system. This can then be broken down into accuracy results per gesture to see if the system has a bias in its success with different gesture types. This information is then elaborated on through the use of confusion matrices which allow a more detailed look into which gesture the system is mistaking for other gestures. Examples of all these techniques can be seen in work from Chen et al. [11]. More recently a new method for evaluation gesture system has been presented by Vatavu et al. in which heatmaps are used to evaluate the gestures [74]. The system can be used in the same way as a confusion matrix but allows for more insight to be drawn into the patterns that caused the confusions in recognition.

For our validation we are using the classic methods of percentage accuracies and confusion matrices as due to our small gesture set size we do not find it necessary to employ the heatmapping technique. However we do use a similar heatmapping technique inspired by the system from Vatavu et al. [74] in the analysis of our data, in the CMV program, when looking for patterns in gestures.

When considering how to validate our gesture recognition system in terms of its usability we employ the use of both questionnaires and interviews in order to gather user feedback on the system. The System Usability Scale (SUS) is a commonly used scale in evaluating the usability of a hardware or software system or product [75]. The SUS is a good tool for evaluating an initial feel for how

viable a system is; several different methods of evaluating the results of the SUS exist of which we employ the “School Grading” and “Adjective Scale” systems [76], [77]. We use the Game Experience Questionnaire (GEQ) to evaluate our game as it provides a broad look at user experience over many elements of the game [78], as well as being free of charge. The main questionnaire gives us a overall view of users experience in the game. The social presence module allows us to evaluate the users behavioural involvement with the AI in the game, while the post game module aims to asses how the user feels after completing the game. Limitations and criticisms have been made of the GEQ which we discuss further in 7.3.

We conduct a semi structured interview with users about the system in which we ask questions that are related to both the game and the gesture system. The questions about the gesture system were influenced by those asked by Yin and Davis as we found the results of their questions to be very useful in the design of our system [69].

2.3.2 Coordinated Multiple Views

A Coordinated Multiple View (CMV) is a visualisation technique for computer systems that allows users to view data across multiple windows where each offers a unique perspective. They are commonly seen in 3D modelling and engineering softwares such as *Autodesk*⁴ and *MatLab*⁵ due to the need for the softwares to display multiple informations about a singular piece of data simultaneously. Roberts explains how they are fundamentally designed to allow for users to have a “dialogue with the data” [79]. They allow the user to understand trends and locate anomalies. Users can interact with human readable, often visual, data and then see the precise complexities of this data in the partnered views. Wang et al. discuss how the use of multiple views has been used in medical care where one stream of data is not sufficient for the monitoring and detection of certain medical issues [80]. In this case is it not necessarily that any view contains more pertinent

⁴AutoDesk, <https://www.autodesk.co.uk>, 2018

⁵MathWorks, <https://www.mathworks.com/products/matlab.html>, 2018

or readable data but that the data is equally valuable and works better in tandem or that the data are only valuable or understandable when viewed simultaneously.

CMV systems can be complex to design and implement [80]. This is especially prominent in systems such as the *Hybrid Virtual Environment* [81] where views are synced across devices including tablets and head mounted displays. Considering the fact that we are looking for trends in how users interact with gestures in three different ways a CMV is a suitable system for this analysis. From observing the values used in previous works [11] we discern that we need to be considering position, orientation and acceleration as possible usable data streams for our recognition.

2.4 Conclusion

This chapter has covered several topics with the focus being on gesture recognition methods (with the extension of continuous recognition), gesture recognition in games and control methods for VR. We have identified that within the space of VR there are presently no concrete rules for control and input methods and present gesture recognition as a possible solution for this problem. We also find that while gesture recognition has been prevalent in games research it has predominantly been at the forefront of the game as the only control mechanic. Our research aims to address this by considering a game in which the gesture recognition system is an optional feature within the game, while still remaining a core feature. After considering existing works in the area of continuous recognition we aim to explore this area for virtual reality games to evaluate whether it works as a control method when controlled to more explicit gesture recognition that is seen in the majority of games gesture research.

This chapter completes our first research objective and in turn also contributes greatly to all of the remaining objectives as it provides us with greater knowledge and understanding of how to tackle the objectives. It provides an overview of the systems that exists and has allowed us to identify limitations in existing

systems and knowledge gaps in the research areas; which we use as influences when designing, building and conducting our research.

Chapter 3

Gesture Recognition System

In this chapter we cover the program design and implementation of our gesture recognition system. We cover how we track and record the motion, the classification of the gestures using a HMM and the values which we find to provide the system with the best results. We also describe the structure and implementation of our database systems, including the adaptive database. The chapter concludes with the initial testing and optimisation of the system including initial base results and values for the topology of the system.

3.1 Motion Tracking

Since we are using the touch controllers paired with an Oculus Rift headset we have access to a large amount of motion data. Unlike many present systems that use accelerometers or image processing which in turn only have access to positional and acceleration data our system is closer to a depth camera system like the Kinect in the amount of data that can be collected from the devices. With this in mind the Kinect has this data for everything in frame where we are limited to the three points of motion being the head and hands of our users.

Due to the direct mapping of movements into 3D space we can extrapolate position and orientation from which we can derive acceleration and velocity for both. Work from Chen et al. suggests positional and orientation data combined is the most accurate [9]. However, since we investigate this for our specific use case we design our gesture recognition system generically; allowing it to work with any of the

mentioned data sets. This also has the advantage of the system being possible for use with multiple projects in the future.

To track motion we attach a listener to both touch controllers, which mimic all their transformations. These listeners are attached to the players headset as a child object meaning that their positions are relative to the headsets current position and orientation. This is done to eliminate the problem that occurs when the player moves in real world space and the position of the controllers is naturally offset. To extrapolate the data from the listeners we poll their transform data during the fixed update loop within unity. The position and rotation are both extracted as vectors using their Euler values for rotation. These values are passed through to the recogniser and stored in lists. When being parsed for either storage or recognition these values are separated into arrays of double values so that they can be passed into the HMM.

Since the Vive has controllers that feature the same motion capabilities of the Touch controllers we are able to use the system across both devices. Although all future testing of the system is completed with the Rift these initial stages were all tested across both systems. This is handled with the motion tracking script receiving its information from a separate input script that can be swapped in and out depending on the platform. With this in mind it is also easy to modify the system in the future for use with less complex motion inputs such as accelerometer devices. Systems that feature accelerometer input often use additional steps in the classification process such as smoothing but again these are something that can be handled in external scripts and then the final data passed into the system. This means that the only real modifications to the system that need to be made are those related to the topology of the HMM.

3.2 Gesture Classification

The motion tracking system records motions from both controllers. Although there is the possibility for the use of gestures that make use of both hands for

now we will only consider gestures that use one hand. This allows the system to be simpler in that it has less data to process but more importantly it allows us to use gestures that are simpler. We hope that in only using one hand to record the gestures it prevents users from being overwhelmed by the gestures.

We do consider the use for two handed gestures in the future. The system is designed in such a way that the number of inputs could simply be increased to include another hand. However, this is a naive approach and could cause high levels of confusion in the system. An alternative method could be the use of another system in parallel that performs individual classification on each hand. There is much room for exploration here with varying thresholds and system configurations. As for teaching users to use the system the application could be structured such that users are first introduced to single handed gestures and then later shown two handed gestures. The two handed gestures could also be designed in a fashion where their uses are related to the single handed gestures that they incorporate so as to keep the motions related for the user.

3.3 HMM Topology

The topology of a HMM consists of several variables that are used to determine the way in which the HMM interpolates data, along with the algorithms that the HMM uses in the training and classification of said data. For our system we use the HMM from the *Accord.NET*¹ machine learning library as it is written primarily in C# and therefore can be easily integrated into Unity projects.

Motion gestures are “order-constrained time-evolving” signals therefore the left-right HMM topology is most often considered to be the most appropriate [11], [63]. Although there have been publications that argue that for there is no difference between ergodic and left-right topologies [2], [82] we will be using left-right as it stands as the most commonly used in modern HMM gesture recognition systems. The Baum-Welch algorithm is used to train model parameters and the forward algorithm is used in the probabilistic evaluations [83]. Schlomer et al. find that

¹César Roberto de Souza and other authors, <http://accord-framework.net>, 2018

the number of states in the HMM has very little impact on the accuracy of the model after a certain threshold [2]. We confirm this in our testing and settle on 7 states. No minimum matching threshold is applied to the classification of gestures. For the explicit system this should not have a noticeable impact as users know when gestures are checked but it is possible that it is a detriment to the continuous system. However, we do not add a threshold for the continuous system as we wish to compare its base results to the explicit system. The optimisation of the threshold for the continuous system is something that should be explored in future works.

3.4 Gesture Storage

Since our HMM system takes arrays of double values we store our data in this format to prevent any additional processing after loading. We store gestures in a custom class consisting of:

- an array of double arrays
- a string for the gesture name
- an integer for the gesture index identifier

Locally we store gestures as a list of these classes to allow for easy modification during game run time. When saving the data we map this list to a Binding List which allows us to use the C# XMLSerializer to store the data in an efficient manner. This is also useful for loading as the data can be directly extracted into a Binding List and then mapped back to our List for use in run time. We also use a dictionary with a key of string and a value of integer so that we can decipher the result of the HMM into a string for parsing into the game. Using this data structure assures our program is extensible and flexible in the sense that the structure and data types can be used in different languages and engines. The system also makes use of elements that can be multi-threaded to increase performance in later iterations of the system.

3.4.1 Database Adaption

One of the challenges that we aim to address in this work is that of improving user independent recognition rates. Our proposed method makes use of database adaption where the original user independent data is subsidised by user data throughout the game. This solution was found after considering proposed methods from [1]. When doing this there are several approaches that can be taken to the method. These include expanding the database, overwriting the database, overwriting and then shrinking the database, and expanding and shrinking the database. The implementation of these systems are relatively simple. When a new gesture is recorded we either add it to the database or we find an element with the same key as it and then remove that element before adding the new one.

One decision that must be made here is when to update the database. Simply adding a gesture when it comes out of the HMM runs the risk of adding a false entry to the database. This can be countered by having a form of user validation where they must confirm that the gesture recognised was correct. A variation of this can be used in a tutorial where the player is asked to perform a specific gesture and the system only proceeds when that gesture is matched. This prevents the user from having to deal with the decision of what gestures are correct manually.

3.4.2 Implementing the Adaptive System

The adaptive system is used throughout the tutorial and first play-through of the game. It is not used during the second play-through as it is not needed as user will not be using the system after this. As previously mentioned when first discussing the system having it run freely could have negative effect on the database if mistakes are made during the recording. This will not happen during the tutorial as it only allows correct gestures through. From this the expected effect is *rolling* in the sense that if the adaption from the tutorial has a strong positive effect over to the first game then this will be strengthened for the

continuous game. Simultaneously if the tutorial does not have a positive effect then things could get increasingly worse by the continuous game.

In the adaptive system when a new gesture is recorded the database is searched for the first record of that gesture and then it is replaced with the new one. In doing this after our tutorial each gesture in the database will contain 7 gestures from the original database and 3 gestures from the user. The HMM must be retrained with this data before it can have an effect on accuracy therefore this is the setup for all users using the adaptive system for the first game. During the first game users can perform any number of each gesture. Some of them could completely populate the database with their own gestures and others may only adjust a few values. This means that we are expecting further variance in the continuous game using the adaptive system.

3.5 System Testing and Results

The system was regularly tested for its accuracy performance throughout all stages of its development as it was the easiest way to gather whether a certain method would work. The most prominent and first change from this was the decision to make all the positional data related to the players headset. This came about from the data originally being recorded using its global positions. This then meant that if the user was stood in a different space next time they repeated the gestures then they would achieve very low accuracy results as the positions were wildly different from the database. During this phase a debugging tool was developed that would allow users to view the paths of all gestures from the database and the testing so that visual debugging was a simpler problem. This tool was the key in the realisation that global data was not suitable to the systems as even with the little freedom of motion that the players had they were able to move to positions that would miss align the data enough to completely throw off recognition.

Following the completion of the core of the system we begin batch testing with a number of independent variables. The variables we consider are the number of points trained along each gesture and the number of instances of each gesture in the database. For this initial testing an individual who was comfortable with motion in VR recorded a set of 50 gestures for each number from 0 to 9. These gestures were then split between training the gesture recognition system and testing the system. The results of this data are seen in 3.1. For each number 25 gestures were tested against the HMM leaving a possible total of 250 gestures to recognise.

Table 3.1: Accuracy rates for user dependent database. Top axis defines number of states in the HMM and left axis defines the size of the training database.

	1	2	3	4	5	6	7
1	36%	46%	54%	56%	54%	60%	58%
2	44%	50%	64%	66%	70%	68%	70%
3	48%	54%	64.8%	68%	74%	76.4%	76%
4	50%	54.8%	66%	72%	76.4%	80%	80.8%
5	58%	64%	72%	78.4%	82.8%	86.4%	88%
6	64%	68.4%	74.8%	80.4%	83.6%	88.4%	90%
7	66%	69.2%	75.6%	82.8%	88.8%	89.6%	92%
8	66.4%	71.2%	76.4%	83.6%	88.8%	91.2%	94.4%
9	66.4%	72.4%	77.2%	85.6%	89.6%	92.4%	95.6%
10	66.8%	73.6%	80.8%	88%	90.4%	94.4%	96.4%
11	67.2%	75.2%	81.6%	89.2%	90.8%	94.8%	96.8%
12	66.8%	76.4%	83.2%	92%	93.2%	95.2%	97.2%
13	66.4%	77.2%	83.6%	93.6%	92.4%	95.6%	97.6%
14	67.2%	77.6%	83.6%	93.6%	93.6%	95.6%	97.6%
15	67.6%	77.6%	85.6%	93.6%	93.6%	96.4%	98%
16	66.8%	78%	86.8%	94.4%	94.4%	98%	98.4%
17	68%	77.6%	88%	95.2%	95.6%	98.4%	98.4%
18	67.6%	78.4%	88.4%	96.4%	97.2%	98.4%	98.4%
19	67.2%	78.4%	88.8%	96.8%	98%	99.2%	98.8%
20	67.6%	78.8%	88.8%	96.8%	98.4%	99.2%	99.2%

The results of the initial testing show that the recognition system performs very well with 7 states. The states show to have a significant impact at the lower numbers which we believe is due to the similarity in pathic data of several of the gestures. Since all the numbers were drawn in roughly the same z plane with a low number of state it is likely that many number shared the same location for some states. This matches the findings from our later results when testing the military gestures in 3.2.

We also find from these results that after a database size of 10 the increase in accuracy becomes very steady with small improvements coming from every few extra gestures in the set. This allows us to consider a database size of around 10 for use in the studies which allows for very fast training times and means that modifications made using the adaptive database could have a larger impact.

3.6 The Commands

Since our game has the player shooting enemies and commanding teammates that will also be doing the same we want the commands to resemble real military commands. Another benefit of this is that the by using real military gestures we assure a level of consideration has already been given to the usability and clarity of the gestures. We adapt the meaning of the gestures where appropriate to give them better context within the game.

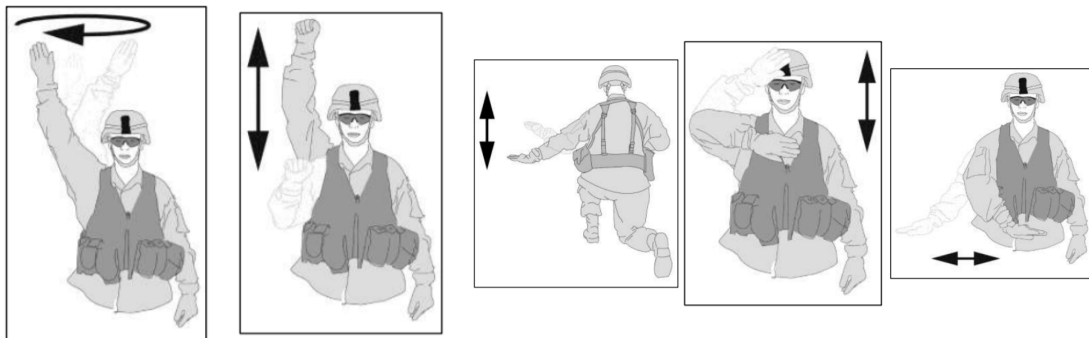


Figure 3.1: The gestures used in the game visualised by their representations from the military gestures documentation[84] (follow, rush, sneak, cease firing, start firing).

The selected gestures from the military database are: “on me”, “rush”, “get down”, “start firing” and “cease firing”. We adapt “on me” to “follow” and “get down” to “sneak” to give them better context in game; the other commands convert without modification. Refer to 5.3.5.3 for more information on the uses for these commands.

Table 3.2: Accuracy rates for initial single user database.

	1	2	3	4	5	6	7
1	82	72	54	54	54	52	48
2	80	84	86	84	80	80	80
3	86	84	84	84	86	86	86
4	82	82	84	84	88	90	90
5	88	90	92	92	92	92	94
6	92	92	92	94	94	94	94
7	93.6	94	94	94	94	94	94
8	94	94	94.4	94.4	94	94.4	94.4
9	94.4	94.4	94.4	95.6	95.6	95.6	95.6
10	96	96.4	96	96	96.4	96.4	96.4
11	96.4	96.4	96.8	96.8	96.8	96.8	96.8
12	97.2	97.2	97.2	97.2	97.2	97.2	97.2
13	97.2	97.2	97.6	97.6	97.6	97.6	97.6
14	97.2	97.2	97.6	97.6	97.6	97.6	97.6
15	97.6	97.6	97.6	97.6	97.6	97.6	98
16	98	98	98	98	98	98	98.4
17	98	98	98	98	98	98.4	98.4
18	98.4	98.4	98.4	98.4	98.4	98.4	98.4
19	98.4	98.4	98.8	98.8	98.8	99.2	98.8
20	98.8	98.8	98.8	98.8	99.2	99.2	99.2

3.6.1 Testing Military Gestures

One notable thing that can be seen from the initial testing is that accuracy results in 3.2 with very few states can manage to achieve relatively strong accuracy. We believe this to be due to the fact that our gestures all take place in fairly distinct positions therefore meaning that segmentation based on simply one step gives the average position and therefore produces moderately distinct results; especially within only a single user test environment. This is consistent through all database sizes and even sees usable accuracy results with only one state, we do not consider this the best for practical use as once user independent use is introduced this may begin to vary. We can see that after size 6 the accuracy results seem to stabilise over 90% and after size 10 they begin to incrementally increase at a generally stable rate of 0.4% (or one gesture) for every subsequent size increase.

The main difference in these accuracies when compared with previous tests using the numbers is that they achieve much higher recognition rates at the lower level. This demonstrates how systems can be optimised by using gestures that are well

designed to be used together. The key here is that each gesture should aim to be different from the others in the metric that is being used for motion detection. For ours it is having them in differing positions but using an accelerometer based system this could be related to both speed and motion of the gesture.

We are happy with our accuracies here for our chosen gestures and so do not make any changes to the set before the testing of the game. We are also confident in the topology of the system for the new set of gestures and so leave this the same as well.

3.7 Continuous Recognition

We design our continuous recognition system to be similar to that of the “Zero Codeword” system used by Elmezain et. al [13]. In our discussion of the previous study we consider several elements from our findings that could influence the way in which we implement a continuous system. Many of these values and considerations are given lenience in their implementation due to the fact that the testing was run on a different gesture set. The main difference between the sets is that the numbers all exist in a similar general position (all in front of the player), while the military gestures take place in varying positions around the body. In terms of length and complexity we consider them to be similar.

Our continuous system works as follows and is based on data discussed in 4.2.3 and 4.2.4. The movement threshold is set to 0.05 units matching the number that was found to be most suited from the results in 4.2.3. Gesture recording starts when the player hold their hand still (within the movement threshold) for 5 frames. It then begins recording the motion until any of three triggers ends it. The triggers include: the players manual cancellation of the gesture by holding down the trigger on the controller; the time limit of 2 seconds ending; the player keeps their hand still (within the movement threshold) for 5 frames.

The start threshold is calculated from the average wait time minus the standard deviation as this is our only trigger for the start of gestures so we want it to have

higher detection rates. For the ending threshold we do not alter the average by the standard deviation as it is not the only method of ending a gesture; therefore making the system less dependant upon it. We choose to structure it like this as we do not want to have gestures being cut off because the ending threshold is too low. Two seconds is used as the cutoff time as this nice rounded number slightly greater than the average gesture time plus the standard deviation. As mentioned previously we do not set the HMM likelihood threshold as we want the system to be identical in this manner to the explicit system. It is noted that this is a likely improvement that can be made to the system in order to improve accuracy.

Although we considered the idea that we could cull gestures that do not make up a minimum length. We decided to exclude this feature as it had potential to work against the adaptive system which could allow gestures to become shorter. There is the potential here for future systems to consider this feature in such a way that it dynamically adjusts the threshold as the system changes but this was out of the scope of our initial implementation.

We do not modify the topology of our HMM in any way for the continuous system as we want to obtain a baseline for the effectiveness of the system that is more comparable to the explicit system. There is potential for future works to consider the ways in which the HMM system could be better designed to accommodate the continuous gestures. For this it would be good to record the motion of continuous and explicit gestures and then evaluate differences in the motion paths. There is also the possibility of optimisation that can be made to the system by the improvement of the adaptive system for continuous use.

3.8 Summary

In this section we briefly outline the final state of the system ready for its use in the final study. We outline the four different systems that we have for testing as well as how the system is implemented within the Unity game engine.

3.8.1 The Four Versions

Table 3.3: Variations of gesture recognition system.

	Explicit	Continuous
Static	Explicit Static	Continuous Static
Adaptive	Explicit Adaptive	Continuous Adaptive

Our gesture system now has 4 distinct variants with the continuous and adaptive database versions that can be seen in 3.3. The explicit system is the base recognition using a predefined database and gestures identified with user defined starts and ends. This system is the closest to the majority of systems found in literature. The continuous versions of the system differ in the way the user starts and ends gestures. As described in the previous section the gestures are started using a low velocity threshold recognition. They are then ended with either a user defined cutoff, the end of a timer or a repeat of the low velocity threshold. The static system uses a predefined set of gestures to train the HMM. This gesture set consists of one gesture each from a number of different people (all of whom were excluded from taking part in studies). The adaptive system uses the same gesture set to start with but modifies it with gestures completed by users as they use them in the game.

The systems are set up so that in testing users will either user both adaptive or both static systems. Therefore all users will interact with both the explicit system and the continuous system. In testing users will either interact with the static or adaptive system but not both. The explicit system acts as a baseline for testing the differences between the static and adaptive databases as well a baseline to compare the results of he continuous system to. The continuous system is tested using both the adaptive and the static database to garner if there is any difference in accuracy when the continuous system features user specific gestures.

3.8.2 Unity Implementation

The Implementation of the full gesture recognition system follows this flow, for specific details on elements of the implementation refer to the previous sections. First a gesture is signalled as being started using either the continuous recognition system or an explicit command from a button press from the player. The continuous system identifies the signal by tracking the positional data from the players right hand and waiting for a period of low motion. Once the gesture has been started a script will record the position of the players right hand once per fixed update frame (60 times a second), stored in an array of Vector3's. The gesture is ended either by the player releasing the button in the explicit system; or from any of the methods in the continuous system (button interrupt, low motion period, timeout). Once ended the gesture is passed into the HMM and the HMM outputs the most likely matching gesture from its database. This gesture is then passed back to the game.

3.9 Conclusion

This chapter completes our third and fourth research objectives and provides the backbone for our research as the systems developed in it are core to the research. We have designed and implemented a gesture recognition system that works with modern VR devices whilst also being flexible in its design to allow for easy improvement in the future. The design and technical aspects of the implementation help to provide the research area with a concrete method for building systems of this nature. Furthermore the chapter describes the implementation of a continuous recognition version of this system that addresses an entirely new method of control input for VR devices. The continuous recognition system is designed based off user data in a manner that allows for future research to consider how user motion data can be used to control systems of this nature. As well as the implementation and design of these systems the chapter also performs light testing on the systems showing their capability as well as methodologies that can be used in the testing of systems in their category.

The chapter also covers the theory behind adaptive databases in gesture recognition. It describes the implementation of the database adaptation system in Unity and c#, whilst also considering a variety of other methods for how the system could be implemented.

Chapter 4

User Interaction with VR Gestures

This chapter covers the design, implementation, results and a brief discussion about the first study completed as part of this thesis. The first study takes a preliminary look at how users interact with VR in a shooting game scenario. More specifically it looks at their movements when they are asked to perform gestures whilst inside the game.

The aim of this study is to identify if there are any distinct patterns in the motion data of gestures performed by users in VR. Furthermore it looks to consider whether this data is capable of being used in the identification of gestures; specifically in recognition of the start and end of gestures. The study aims to provide us with this data so that we can develop an informed version of a continuous gesture recognition system that is suitable for use in VR. The study completes our second research objective and contributes towards the completion of the fourth objective.

4.1 Design and Implementation

The first study was designed from the start to be a lightweight application that did not require lengthy implementation time. The study was designed so that it could be run and completed over the course of a week to allow for ample development of the second study. The main goal of the game was to give users an

environment in which they could move and interact with the game in a similar manner to how they would in the full game. To complete this task a shooting gallery style game was devised as an appropriate starting block as the full game was originally scoped to have users facing down a single direction for the majority of the time. The users periodically perform gestures in the form of the number 0-9. The numbers were used so that users did not have to be explained what the gestures were allowing them more natural interaction with them. Since the data we aim to collect from this study is motion data logged by the system we do not require any qualitative data from the participants and only minimal quantitative data covering the basic information about their VR and game experience as well as generic personal information such as age and gender.



Figure 4.1: An overview of the level for the game.

As we want users to complete gestures in amongst normal actions in the game, the shooting gallery is designed as a wave based shooter. The user faces 10 waves of targets and after each wave is asked to draw a number between 0-9. The numbers are randomised each time so as to avoid any order bias between users.

The aesthetic of the game is a sci-fi setting using low polygon models so as to keep the visuals efficient for high frame rates in virtual reality. We use baked lighting to give the game an immersive and atmospheric feeling without losing any run-time performance. An overview of the level and its visual style can be seen in figure 4.1.

We implement the game in the Unity game engine as it stands as a popular engine for the development of Virtual Reality games. The game is written in the C# language as our main game is also written in this language to allow for its easy integration with the HMM model we have created.

The *Oculus Sample Framework*¹ was used as the baseline for our character interaction in the game. This provides us with all of the required camera and controller management for our VR device. The character height was set to 1.70m as default and we tested heights from 1m to 2m to make sure that the camera and controls worked at all heights between them. A visual example of some of what is provided by the Oculus Virtual Reality (OVR) plugin can be seen in figure 4.2.

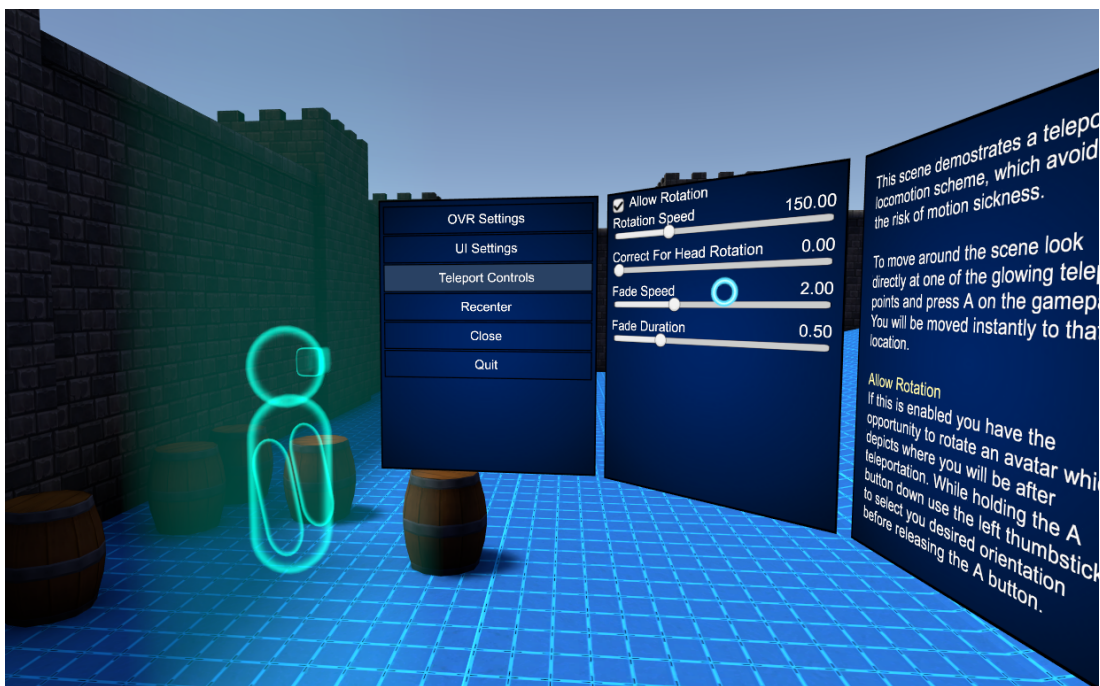


Figure 4.2: The Oculus Sample Framework². Showing the capsule view of the player avatar and the inspector

4.1.1 Game Testing

Throughout the development of the game informal user testing was completed with several different users. Research students and lecturers from the lab in which this project was developed were periodically asked to play the game and give their thoughts on the balance and difficulty of the games design. Along with this

¹Oculus, <https://developer.oculus.com/documentation/unity/latest/concepts/unity-sample-framework>, 2016

2 people with very little VR and general game experience were asked to do the same. These short testing sessions were completed to ensure that the balance of the game was correct for as many different user types as possible to avoid people being unable to interact with the game during the studies. The specific changes that were made as a result of these informal testing sessions are discussed throughout the individual mechanics design sections.

4.1.2 Gestures

Between the waves of targets to shoot, the player is asked to perform a gesture in the form of a number between 0-9. The numbers are given in a random order for each participant and all participants see the full range of numbers, meaning 10 waves in total for each user. The participants are told to draw the number in the way that felt most natural to them. The number that they should draw is shown to them in text in front of the short wall where the targets come from. The users also receive an audio cue that instructs them which number to draw.

Numbers were selected as the gesture patterns as it gave all players a level playing field for understanding. It prevented any users from not understanding the patterns or how to go about drawing them allowing them to instead focus on completing the gestures in whichever way was most comfortable to them.

In order to perform the gesture the participants must hold down a button on their right controller and draw the gesture with their right hand. They are allowed freedom of time when drawing the gesture. While they are holding the button a blue trail will follow their hand showing the user the path that they have recorded. Once they release the button the trail stops and the gesture time ends; the trail will remain visible for 2 seconds before fading out along the path that it was drawn. In tandem with the trail dissolving the game waits 2 seconds before the next round of targets being launching. A visual example of what a gesture looks like from the players perspective can be seen in figure 4.3.



Figure 4.3: The number 5 being drawn in the game. Shown from the players view perspective (player has a wider view in VR).

4.1.3 Shooting



Figure 4.4: One gun fully visible, one half way through dissolve in animation.

The game gives the player two guns which follow the motion of the Oculus touch motion controllers. The guns are only visible when the player holds down a button on the controllers and use a dissolve effect to transition between visible and invisible; the guns can only be fired when they are visible. Figure 4.4 shows the guns with an example of the visual transition they go through. The guns shoot lasers which raycast directly from the tip of the guns barrel. This allows

users to use the guns to aim accurately without too much training time learning how to account for things such as bullet drop. When the lasers make contact with a surface a small particle effect plays at the contact location to give the players visual feedback for the hit location. The lasers play a sound effect on firing to give audio feedback for the action. The guns have a cool down of 0.25 seconds between shots to prevent the player from being able to constantly fire. The fire button may be held to fire the gun automatically. Since the game takes place in a closed environment we allow the guns to fire at an infinite range.

The guns are bound to the transform of the players hand in the game. They replace the hand models as the only visual aid the player has of their hands. It has been shown that frame of reference points decrease the effects of VR sickness [85], [86]; for which visible player hands are commonly used. This was an oversight in the design however considering that the players have the gun visible for the majority of the game we do not consider it to be a serious issue; especially when considering that the players are not subject to any non physical movement modalities in this game.

4.1.4 Targets

The targets are moved up from behind a low wall in front of the user, when they are launched they make a sound effect to indicate the player of their launch. They rise to a random height between 0.5-0.9 times the height of the room before slowing to a stop and then falling back down behind the wall. Once the target falls back behind the wall the player will be punished by deducting 100 points from their total score. If the player successfully manages to hit a target then they will be rewarded with a score between 0-500 depending on how long the target has been visible for. When hit the target will dissolve out using the same effect as the guns to transition. A visual example of the target being hit and dissolving can be seen in figure 4.5. The targets are released on a timer such that after the last target is released the next one will release between 1-2.5 seconds later. This means that if the user does not shoot a target until it has almost left the screen

then there will likely be a new target either on or entering the screen. The targets come in waves of 10 and do not fire when the player is asked to perform a gesture.



Figure 4.5: A target in the game being shot. The score reward can be seen in the top right of the image (+300).

During testing of this mechanic we had both an experienced and a non experienced VR user play the game and report their feeling on the challenge and balance of the game. Originally the targets would move too slow but at the same time the time between targets was too low. This was reported to frustrating as it made the targets too easy to hit but also overloaded the player as too many things would appear at once. The final balance numbers were found to be better as they provided sufficient challenge for both users without being too mentally over-encumbering.

4.1.5 Scoring

The players score is displayed on the top left of the wall behind the targets. The score will start at 0. Whenever the player gains or loses score a visual indication of the amount is shown as a number just below the total score indicator. This number is red in colour if the amount is negative and green is positive, the text fades out and rises slowly to transition out. Originally when the targets were hit the score indicator would be positioned at the location that the target was hit. This originally seemed intuitive but since the targets kept moving for the

duration of their dissolve animation after being hit it was decided that having the score in this location was distracting visually for the player. Instead having both score appear in a reliable location was more consistent for the player and was less distracting.

4.1.6 Logging

This study is designed to keep track of users actions throughout the game. We do this by logging their actions in such a way that we are able take an in depth look at their behaviour during analysis. We record both the orientation and position of both the controllers as well as the headset. We also keep track of when the player starts and stop gesturing. From this data we can calculate other variable such as acceleration, velocity and angular velocity.

```
public sealed class TransformDataMap : ClassMap<TransformData>
{
    public TransformDataMap()
    {
        Map(m => m.HeadX).Index(0);
        Map(m => m.HeadY).Index(1);
        Map(m => m.HeadZ).Index(2);
        Map(m => m.HeadRotX).Index(3);
        Map(m => m.HeadRotY).Index(4);
        Map(m => m.HeadRotZ).Index(5);
        Map(m => m.RightX).Index(6);
        Map(m => m.RightY).Index(7);
        Map(m => m.RightZ).Index(8);
        Map(m => m.RightRotX).Index(9);
        Map(m => m.RightRotY).Index(10);
        Map(m => m.RightRotZ).Index(11);
        Map(m => m.LeftX).Index(12);
        Map(m => m.LeftY).Index(13);
        Map(m => m.LeftZ).Index(14);
        Map(m => m.LeftRotX).Index(15);
        Map(m => m.LeftRotY).Index(16);
        Map(m => m.LeftRotZ).Index(17);
        Map(m => m.Gesturing).Index(18);
        Map(m => m.GestureTime).Index(19);
        Map(m => m.GestureNumber).Index(20);
    }
}
```

Figure 4.6: The class map for the data model that we are logging.

We import the open source *CsvHelper*³ library to make our logging more manageable. This plugin allows for us to create a class that can be output directly to the log using a class map as well as being used again for direct importing of the data when parsing without the need for any sorting. We create a

³Josh Close, <https://joshclose.github.io/CsvHelper/>, 2018

class for logging and attach it to an empty object in the scene. We bind references of all the values including booleans and transforms to this class. For transforms we take the positional vectors and the Euler coordinates and extract each individual value as a float for storage. In Unity's *FixedUpdate* loop (running at 60fps) we tell the logger to write one CSV line containing all the bound information. We do this by creating an instance of our *dataLog* class and populating it with that frames information. This class uses a class map (which can be seen figure 4.6) in order to dictate which order the variables are added to the CSV as well as to write a heading for each column. By logging the full transform information of both the hands and the HMD on the player we are able to later recreate a complete visual model of their movements during the game.

4.1.7 Analysis

Our system for analysing the data stems from our desire to evaluate whether the gestures that users performed are in line with the patterns that Elmezain et al. used in their continuous recognition system [13]. Our use of heatmaps to look for patterns in motion data is inspired by the work from Vatavu et al.[74].

We use a Coordinated Multiple View (CMV) for analysis of our data from this study. Our CMV is able to display: a 3D representation of the users avatar in 3D space; a 2D heatmap of the users hand positions; and a time dependent line chart that is able to show any form of velocity or acceleration. The CMV can be scrubbed between any 2 time points during the data and it recalculates all of the views accordingly. The 3D representation updates itself in real time as timeline is scrubbed through.

With our data types decided we design a layout that allows the user to view the players movements on a 3D model whilst looking for interesting frames of data. As seen in figure 4.7 our CMV design shows 3 windows with one that can be alternated to a different view. In the inspector there is a data value for both start and end frame as well as a list of start frames for all of the gestures. Scrubbing the start frame will manipulate our 3D representation in real time according to

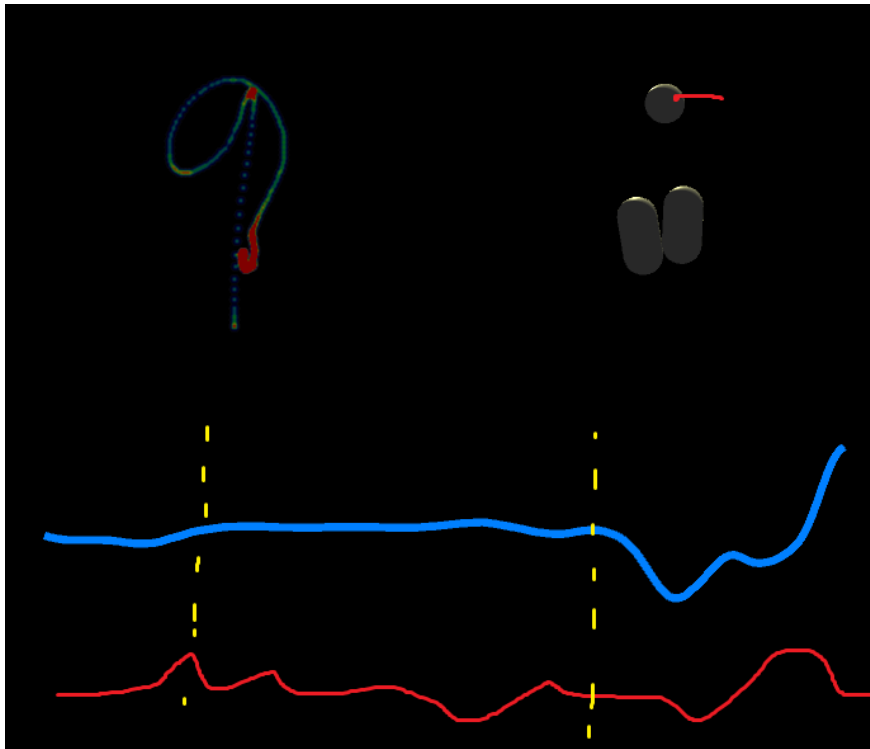


Figure 4.7: A view of our csv before the implementation of markers for gesture start and end (markers manually added with yellow dashes).

the speed of the frame scrub. Our 3D representation of the player consists of two models of the guns used in the game to represent the hand position and orientation of the player. The player's head is represented by an oval with a cylinder position in the middle front of the shape. This cylinder leaves a trail that lasts for 2 seconds before dissipating as it moves, allowing the user to more easily identify the player's head movements when scrubbing the view in real time. The head movement can be seen clearly in figure 4.7 where the red line next to the sphere in the top right represents the movement of the player's head.

The upper left view contains a heatmap for the positional data of our player's right hand; since all gestures are performed with the right hand there is no need to display the left hand data. The heatmap is implemented using a shader that dynamically adjusts the blending of the colours depending on the density and count of the data. The heatmap displays data from the start frame till the end frame inclusively. Since the heatmap requires a small computational time it does not automatically adjust instead it requires a button in the inspector to be pressed to trigger an update. An example of the heatmap can be seen in figure 4.8, in



Figure 4.8: An example of the heatmap view with the game background shown behind it to give context to the hand positions.

this image the heatmap is displayed scaled onto an background of the game level to give context of the hand motion for the game.

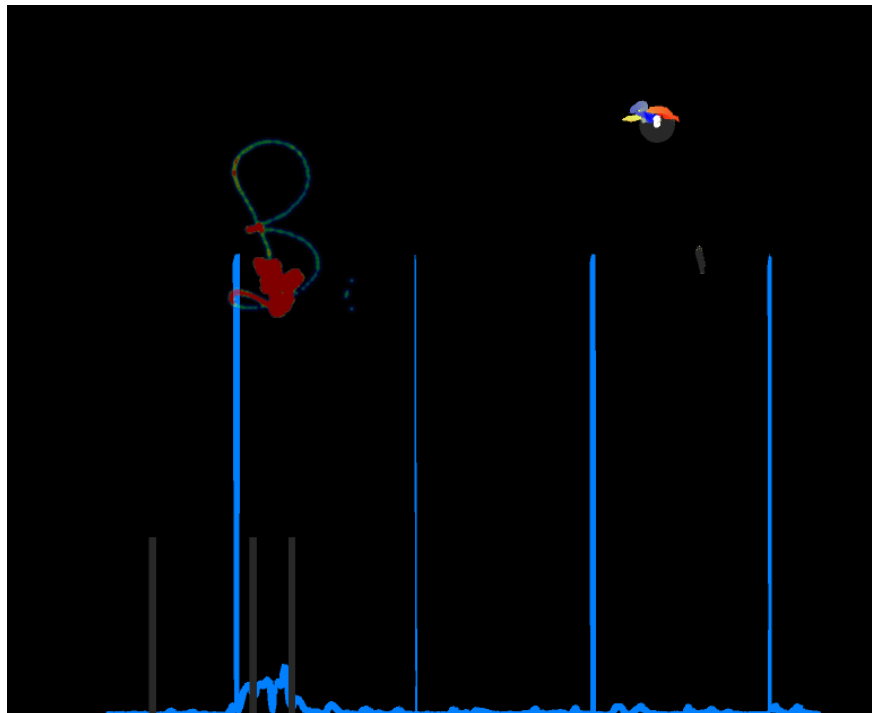


Figure 4.9: An example of the CSV displaying the trace of the players head movement. The high peaks in the line chart are from controller connection spikes which occur occasionally but have no impact in game.

The bottom view of the model contains a line chart for either the velocity or the orientation. It displays data from the start frame till the end frame; it also displays a grey line vertically at the point of the start of any gesture that occurs during its time frame and another grey line for the end of a gesture. This was designed not only to help analyse the data around gestures in this view but also

help to position data for the heatmap. The data in the line chart is all normalised according to the maximum value found within the current data range to allow for closer inspection of relative changes. Similar to the heatmap both the charts can be generated using the two buttons located in the inspector. Figure 4.9 shows an example of the line chart displaying velocity. Looking at the heatmap you can see that the area of higher velocity on the left that occurs during the gesture matches the heatmap where it is clear that the player is moving their hand fast without stopping.

As mentioned we are predominantly looking for motion patterns that match those found in the work of Elmezain et al. on their “Zero Codeword” system [13]. These patterns consist of low velocity and movement sections before and after gestures. We extend our analysis to consider more broad areas such as general speed and size metrics. To identify motion patterns we traversed all the data of an individual user manually to look for consistencies in their motion before, after and during the gesture. In doing this for all participants we are able to build a profile of sorts for each user with regards to the commonalities between their motion records for gestures. The commonalities found in these profiles make up the consistencies in the motion that form the patterns we extract.

4.2 Evaluation

In this section the methodology for the study is described, followed by the results and finally a we undertake a discussion of the findings gathered and their possible applications.

4.2.1 Data Collected

From each participant their: age, gender, self determined VR experience and number of hours of games played per week were recorded. Along with this data we record a full log of the players motion in the game as well as several bits of

information about the game state. For thorough detailing on the implementation of the logging refer back to 4.1.6. The data we record contains information about the position and angle of the players head and hands. We also record when they start and stop pressing the button to initiate gesture recording as well as when the game tells them to draw the gesture. All of this data is recorded 60 times per second in time with Unity's physics update loop.

We record all of this data so that we are able to fully recreate and visualise the participants movements in our CMV program. This allows us to perform detailed analysis on the movements in post. As well as visual manual analysis we are able to pass the data into various tests to obtain statistical data such as averages for movement thresholds and times. See Appendix item 8.1 for an example of a set of the logged data and its layout.

4.2.2 Methodology

As described in the previous section the study is focused around gathering data from participants in the forms of logs of their gameplay. Aside from the log the only other data recorded is basic information about the participant. These included age, VR experience, amount of hours spent playing games and gender.

At the start of the study the participants read a document covering the: details of the study, what they will have to do, and any risks it might contain. They are then offered time to ask the invigilator any questions they might have and are given a consent form to sign. Once they have given consent they are given the personal information sheet. After this they are introduced to the system. They are given the headset and the invigilator explains what will happen and how they interact with the game. The users are given as much time as they need to equip the Oculus Rift headset and touch controllers; making themselves comfortable in VR. Before beginning the game the invigilator checks that the participant is comfortable in VR and informs them again that they can remove the headset at any time should they need to do so. Once the participants are comfortable the game is loaded and started. The participants then play the game which involves

them shooting the waves of targets and performing gestures between waves. The playtime for the game averaged around 7 minutes and halfway through the game the users are asked whether they still feel comfortable in VR.

38 participants took part in the study (20 male, 18 female) with an age range of 18 - 28 and an average age of 20.6. The average self defined VR experience was 4.9 for males and 4.2 for females on a scale from 1-7. Refer to 4.1 for the distribution of number of hour per week spent playing games.

Table 4.1: Number of hours spent playing games per week for each participant.

	>1	1-5	5-10	10-20	20+
Male	2	4	6	5	3
Female	4	3	6	3	2

When analysing the data we adopted a system where we manually look through all of the data sets looking for patterns in any of the sets. Once patterns are identified we perform a second pass on the data noting down all the areas in which this pattern is seen and finally once we have identified all of the visual patterns we run numerical analysis on the data to gather any collated averages in the data that appeared worth investigating from the pattern analysis.

4.2.3 Results

Through the manual analysis of the data several patterns were observed; the most notable of these findings are listed here in chronological order. First is the two approaches to beginning a gesture. The vast majority of participants fall into two groups that approach the start of a gesture in a different manner. Upon being instructed to perform a gesture the first group (a) move their hand to the position in which they wish to start the gesture and then stop (this can be seen in 4.11). They then hold the gesture button down and perform the path. The second group (b) hold the button down as soon as they are instructed to perform the gesture and then move their hand to their desired starting position, pause, and then continue with the gesture (an example of this can be seen in 4.10). There is a common denominator here in that both groups pause at their desired start point before beginning the gesture. There were a small number of participants

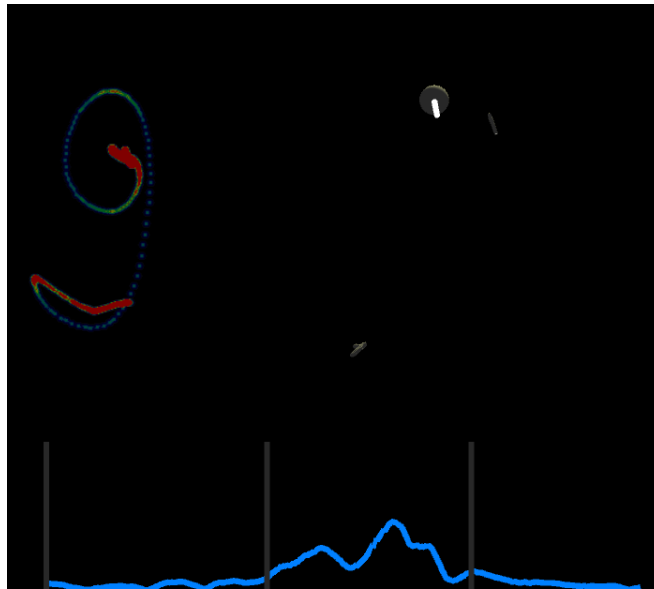


Figure 4.10: An image depicting a participant starting their gesture without pre-positioning their hand.

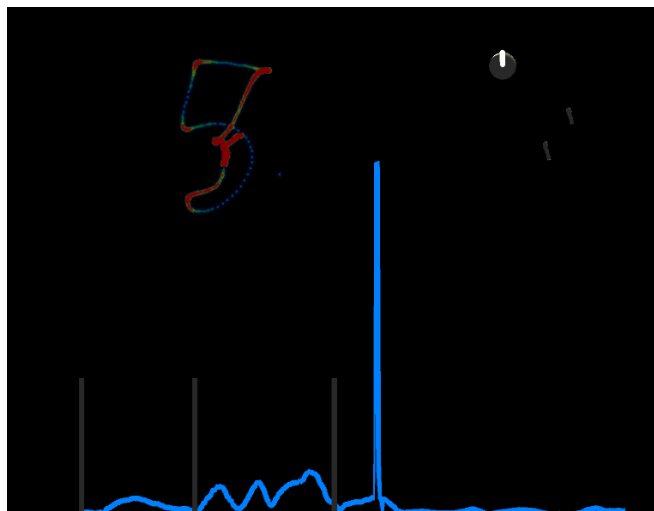


Figure 4.11: An example of the CSV displaying a user moving their hand to a starting position before beginning their gesture.

that did not follow either of these patterns. Generally these participants would just begin the gesture from wherever their hand was upon the instruction being given.

The second observed pattern was the size difference that could be seen between the drawn gestures. Generally participants were consistent with the sizes of their gestures choosing to either draw them using large arm movements or concise wrist movements. The main difference between these two types was the average time taken to complete the gesture as on average the large ones were slower. On the other hand, some participants simply moved faster and some moved more

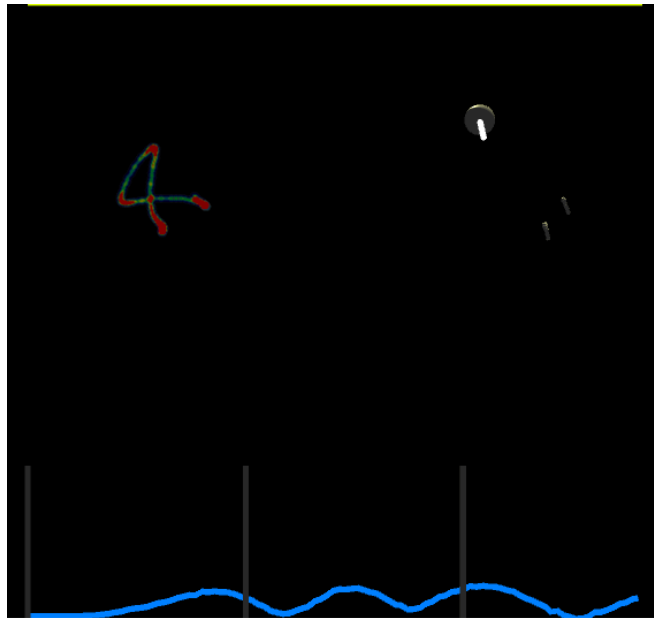


Figure 4.12: An example of the CSV displaying a smaller gesture.

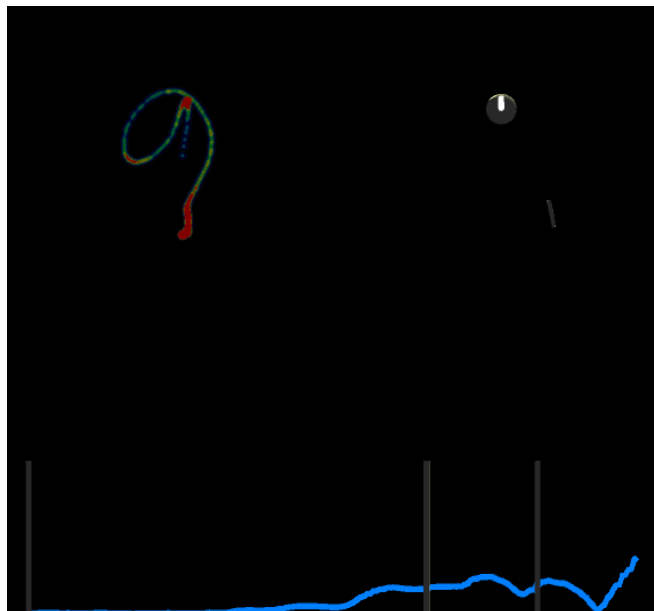


Figure 4.13: An example of the CSV displaying a gesture that ends with trailing motion.

carefully. Examples of smaller and larger gestures can be seen in 4.12 and 4.11 respectively.

The third observed pattern considered the way in which people ended their gestures. The two groups from the first pattern also for the most part had differing behaviours here. Group (a) tended to pause at the end of their gesture before letting go of the button, as seen in 4.11. While group (b) would just trail off releasing towards the end of the gesture, seen in 4.13. There was

another observed behaviour here which showed some users stopping at the end of the gesture even if they had released already.

There were no real observable consistencies in how people started or ended gestures shown in the angular velocity data or the orientation data. As for velocity there were observed stills at the start and end of gestures that line up with the movement patterns mentioned previously. It was noted that generally during gestures participants kept their head movements to a minimum. There were some that would follow the gesture path as they drew it but the majority of participants kept still head movements.

Table 4.2: Average gesture times in frames and seconds

Number	0	1	2	3	4	5	6	7	8	9
Frames	93	55	82	74	95	99	72	67	75	71
Seconds	1.55	0.92	1.37	1.23	1.58	1.65	1.2	1.12	1.25	1.18

The overall average frame time for the gesture was 78.3 frames which is equivalent to 1.305 seconds with a standard deviation of 0.365 seconds. Using a threshold distance of 0.05 units in Unity and starting the detection as soon as a gesture instruction is given we find the following timings. The average for rest frames before a gesture was 8 frames with a standard deviation of 3.1 frames; after a gesture was 6 frames and this had a higher standard deviation of 4.4 frames. The size of drawn gestures varied greatly with the smallest being only 0.35 units in radius and the largest being 1.82 units. The average was 1.25 units with a standard deviation of 0.4.

4.2.4 Discussion

From our findings we observe that on average users rest in a low velocity static position for an average of 8 frames before beginning a gesture and an average of 6 frames after a gesture. We can reliably trigger an event to start a gesture with this using a threshold listener. Looking at the standard deviations of these results it is possible that lowering the threshold could produce better results. This is something that needs to be measured carefully on a system by system basis as

lower thresholds could lead to a highly misleading system with events that trigger often at wrong times.

Considering the disparity in lengths of different gestures there are two major points which we take forward into consideration in our following works. The first of these is wherever possible gestures should be designed to be as close as possible to a singular length. This is not always possible and users can perform gestures in unique ways which contradict the desired motion. An example of this is when a gesture is designed to consist of two repetitions of a single motion but users only repeat the motion once.

The second consideration is with the ways in which we decide the end of a gesture. The methods we consider are: static motion (low velocity/ position change); time since gesture began and external input that explicitly states gesture time is over. Using these multiple approaches we aim to maintain accurate end times prevent miss classification by having paths that are too long and confuse the model.

With the length of the gesture in consideration it would be good to be able to run a preliminary study in which we can decipher the run times of our final gestures for the game. However, this is out of scope due to the time constrictions in gathering users. We can locally test the gestures with a small group of people and determine an average run time that encompasses the highest average. In cutting gestures to this length it is possible that we increase the bias towards the recognition of this longest gesture but considering that we only intend to track position in our model it should not have too heavy of an effect. The more dangerous possibility would be that if gestures can continue on for too long then they can add large amounts of extra data points to the model causing higher levels of confusion. Whilst these gestures could be filtered out in the threshold it is possible that many slip through as the general movements of users can line up with some gestures.

One design decision that will need to be made is whether gestures are processed or culled when they are interrupted by other inputs. For example if a user activates their gun and begins shooting straight after performing a gesture should the

gesture count or should it be discarded. This has the potential to have negative impacts on users to react quickly in the game and do these things intentionally in order to do better. On the other hand it could have the opposite effect, where gestures are triggered accidentally by pauses in player motion and then can potentially interpret the wrong input from the user. There could be a balance here where in order to be processed the gesture must be over the minimum gesture average threshold (or some slightly lower value to account for user variance) so that the gestures are filtered more accurately earlier on.

These findings are used in the design and development of the continuous recognition system described in Chapter 3. The main takeaways from the study that influenced the system are: the static motion that users have at the start of the gesture and the length and threshold of the motion; the same static motion with different values for the end of a gesture; the average lengths of gestures for cutoff points when designing the system. These three findings make up the majority of logic behind the continuous recognition elements of the gesture recognition system. The findings confirm that the “Zero Codeword” system used by Elmezain et. al could also be used in our setting for gesture recognition in VR [13].

4.3 Conclusion

This chapter completes our second research objective and in doing so also answers our first research question. Through the analysis of user movement data in a VR game featuring gestures we have identified several patterns of movement. These patterns enable the development of the continuous recognition system from Chapter 3. The three main patterns that we identify also line up with those used in the “Zero Codeword” system used in [13]. The game mechanics developed for the player controls also make up the foundation of the mechanics for the game in Chapter 5.

Chapter 5

Gesture Recognition Game

This chapter covers the design and implementation of a VR shooting game which we use to evaluate our gesture recognition systems. It starts with the study design, discussing how and why the game is built the way it is and what that allows us to achieve. It then covers how user testing was used throughout the building of the game to make sure it fits the needs of the study. The rest of the chapter goes into detail covering how the various elements of the game are put together. The main sections of implementation are the friendly and enemy AI along with the movement systems. In 5.3.5.3 the friendly AI's responses to gestures are described. The chapter concludes with the implementation of the games tutorial.

While this chapter does not directly address a specific research objective it does build the system that is required for the completion of the 5th research objective. This also allows us to directly work towards our second research objective. The artefact is used in the following chapter to achieve many of the other research objectives as it is used in a study.

5.1 Study Design

This study is responsible for the testing of both the explicit and the continuous recognition system, as well as the adaptive database system. We also want to test how gestures work in a VR game, with the extension that we want the gestures to be an optional feature that are not required for the completion of the game.

The first step which we take to achieve these objectives is to build a set of core player mechanics that allow the player to move shoot and complete the game without the need for any interaction with the gesture system. We then implement the gesture recognition system on top of this game and design them to interact with NPCs. This system allows us to have the gestures perform a core role in the game encouraging interaction with them, without forcing it so if players do not like the system they do not have to interact with it. The core systems in the game are designed to be largely simplistic so as to avoid overwhelming the player with information. We expect experienced game playing and VR users to become familiar with the game mechanics easily and have no trouble playing. With the simplistic design decisions we allow for less experienced users to learn and play the game with only small amounts of training.

In our literature review we find that there is a gap in the research area of gesture recognition in games where the gestures are not the main input mechanic. Having the gestures be a part of the game instead of the only input mechanic we are able to gather information about the system's accuracy when users have other points of focus excluding the gestures. We theorised that this would have an impact on the accuracy of users' gestures as they would not be as focused on giving consistent inputs, which we believe is more akin to how gesture systems would be used outside of test environments.

As defined in 3.8.1 we have four different systems which we want to test with this study. We divide the testing of these systems so that half the participants will use the adaptive system and half will use the static database system. All participants will use both the explicit and then continuous system in that order. This could have potentially caused an order bias as we do not counterbalance the conditions. However, we maintain this order as during the user testing, described in 5.2, we found that less experienced users were not able to grasp the continuous system before they had already interacted with the explicit system. Further consideration to this is discussed in 7.3.

5.2 User Testing

Throughout the development of the game a small subset of users were repeatedly asked to test and provide feedback on mechanics in the game. The group which conducted the testing was the same as for the first game except from the inexperienced VR users which were different as it was thought that the original two users had become experienced with VR systems since the original game was built. Predominantly the testing was confined to a single mechanic in an isolated environment; for example when testing the traversal mechanics we had users in a mock level with two rooms and a corridor. However as the game neared to completion users would test the game in a more complete state so they could give feedback on the mechanics having used them in a more natural environment; this was particularly necessary when testing how overwhelmed the users could become at certain stages of the game. More specific details around how the user testing impacted the design is seen throughout the individual subsections. The users testing was crucial in enabling us to build a game that could feature several mechanics whilst still being accessible to new VR users with minimal training.

5.3 The Game

We wanted the game to be between 7-15 minutes so that users had time to experience the gesture system fully without subjecting them to an extended VR experience.

5.3.1 Level Design

The levels in the game are designed as primarily linear experiences that are segmented into different rooms. Since we are designing the game for VR we are limited in terms of the scale of the visuals that we can offer. In order to keep the framerate above a minimum of 90fps we design each room as an encapsulated

environment where we can cull most of the rest of the level out using occlusion culling. This meant that when adding windows to a room we needed to be careful that if they revealed other sections of the level that it was not a particularly large section.

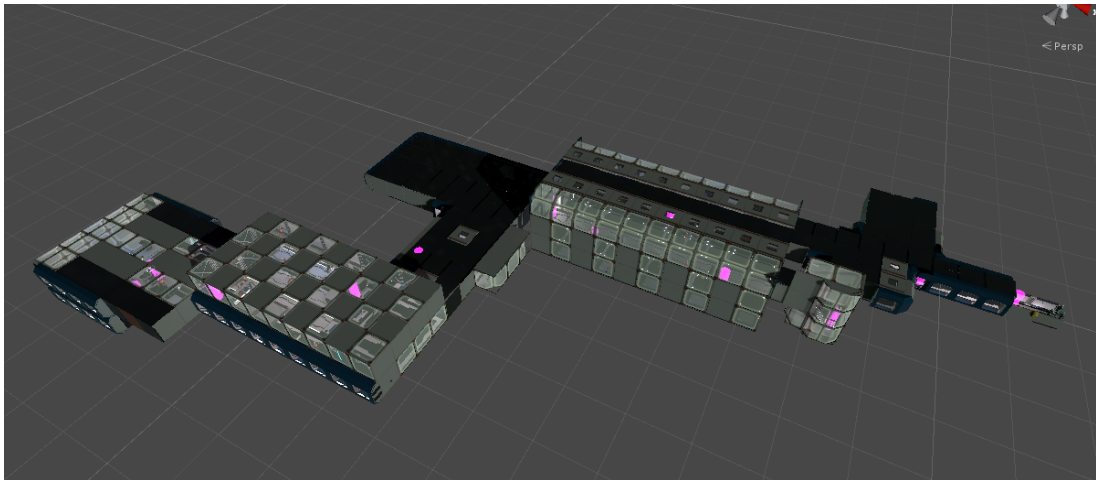


Figure 5.1: An overview of the level for the game.

Each room contains a number of enemies and enemy way-points usually positioned in the back 2 thirds of the room. This is to attempt to avoid overwhelming the player with too many directions of interaction at the start of each room. Originally in a few select spots we had enemies that would be positioned to the sides or behind teleport points to ambush players. This was removed after user testing as we found that it was too overwhelming for less experienced players. The player teleport points are positioned throughout the length of the rooms generally in places where the player is able to take cover from enemy fire in one direction. They are positioned such that they alternate sides of the room giving better lines of sight to both the next point and the rest of the room.

The rooms are populated with various *clutter* items in order to achieve a number of tasks. The first and most important is to provide a more interesting and challenging field for the player to navigate and combat in. On top of providing cover the objects also make the game feel more like a traditional game as they help to build a world narrative. This is furthered by rooms being themed around different sections of the ship like engine room and sleeping quarters.



Figure 5.2: An example of a room from the game, populated with enemies and objects for decoration and cover.

5.3.2 Movement

We want to be able to strictly curate the players journey through the level so that all participants are presented with as similar scenarios as possible. One of the ways in which we can achieve this is by limiting their movement through the level. A common movement modality in VR games is to have the player move between fixed points in the world; at each of which freedom of movement is given to them in the physical space they have. By doing this we can provide the player with a number of different movement and combat points at which they can take as much time as they like to explore the area. With this in mind we design the movement through the game as a series of teleport locations; each of which is in vision when the player is at the next and previous teleport points. All positions for teleport points were tested and iterated upon based upon user feedback for their positions. This allowed us to make sure that all points gave a fair line of sight and were not positioned confusingly.

Upon holding down a button on the left controller the player can aim a laser out of their left hand (from the gun tip if the gun is visible). When the laser is hovered over a teleport location the colour will change and if the button is released the player will be teleported to the new location. Through testing it was found that some players find this disorienting and forget which way is forward

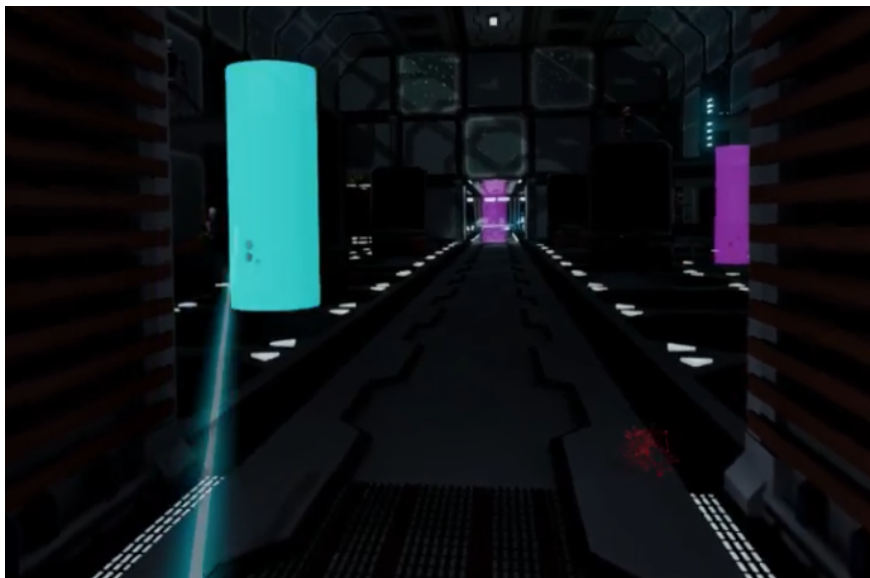


Figure 5.3: An example of teleport points in game. The blue one is being targeted by the player.

in the level. This is countered by having previously travelled to points turn to a different colour and they will not change colour when hovered over. We found this left players with much better sense of direction.

5.3.3 Shooting

The base mechanics for the shooting are taken from the previous game. The player still interacts with the guns in the same way and the visual effects the guns produce are the same. The predominant difference in the shooting is that when hit, the players guns will be disabled preventing them from shooting them for 3 seconds. During this time the guns will be coloured with a matt grey colour and text will appear above them saying "Guns Disabled". When the player is hit a voice over also states "Guns Disabled" along with a flash of the screen to white to signify that they have been hit; this was added after feedback suggesting that it was not always obvious when you had been hit. While the guns are disabled the player is still able to teleport and perform gestures meaning that this mechanic does encourage the use of gestures.

The hit-boxes on all AI NPC's consist of a capsule that covers the whole of the characters body from their feet to their head. The capsule is roughly the same

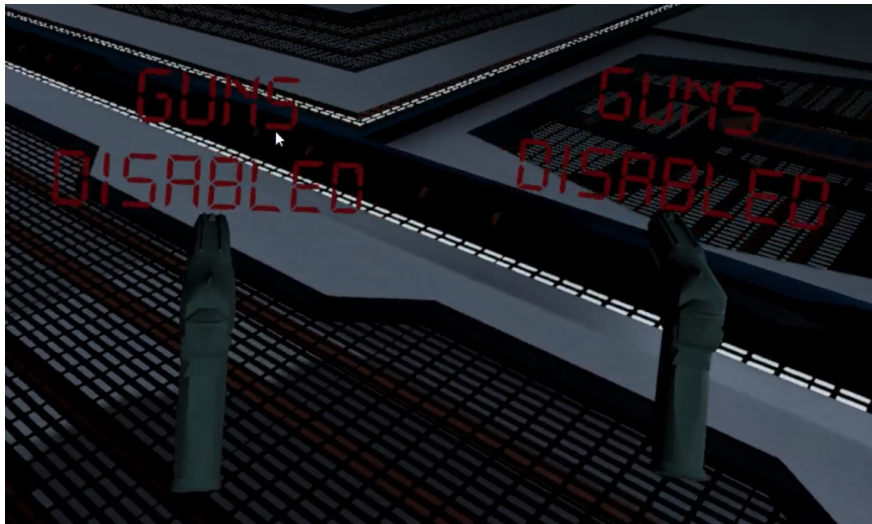


Figure 5.4: Visual for the guns when they are disabled from shooting.

radius as their body with obvious lenience around the head and leg areas. Their arms do no contain hit-boxes. The players hit-box is a small sphere attached to their head. We do not have a hit-box on the other areas of the players body since they are not able to see it and therefore it would be visually confusing for them to be hit there.

5.3.4 Artificial Intelligence for Non-Player Characters

Since we require both friendly and enemy AIs with many similar movements and behaviours a generic AI class was designed and implemented.

This class started with a shooting behaviour that simply took a target and then fired a projectile from the start point of a gun barrel towards the target. This was then made more dynamic by adding an element of randomness to how the bullet would travel by adding a slight random offset in both the X and Y axis of the bullets trajectory. This gives the effect of the entity dynamically aiming at the target. The projectile consists of a simple sphere with a bright emissive colour and a trail renderer to emphasise the movement of the object. The accuracy of the enemy was altered throughout testing in accordance with users stating that enemies felt unfair or not strong enough.

The majority of the AI behaviours are dictated by a state machine. It exists as a switch statement populated by enums that define all the different actions that the AIs can undertake. While both enemies and friendlies have similar behaviours in their state machines the actions these causes and the way that they are triggered differ largely so will be discussed individually.

All movements that the NPC's undertake are handled using a Unity Navmesh. This allows us to simply dictate walkable terrain and obstacles and then let the Navmesh handle the path-finding for the agents.

5.3.4.1 Animations

As the game takes place in VR it was necessary to have the NPCs move around with relatively polished animations. Since our AI characters use generic humanoid rigs we are able to use assets from the *Mixamo*¹ collection. Mixamo exists as a free site containing a vast library of high quality animations for humanoid rigs. Thanks to this we are able to bind animations to our characters that are close to the quality of some commercial games.

We use several animations for our characters: crouched, walking, running, crouch shooting, standing shooting, idle. Thanks to Unity's animation blending feature we are able to combine these animations and have a fairly comprehensive animation set. The only oversight in our animation system is that of rotating the characters when they are stationary. Turning while moving looks natural without any additional animation but static rotation would have required additional animations to be added and blended with the system. Considering the development scope of the game we decided that this was not a necessary addition.

Along with the animation the NPCs are all equipped with a rifle. This is attached to their right hand and in firing stances is positioned so that it sits comfortably in both of their hands. The gun is not always visible depending on the current

¹Adobe Inc, <https://www.mixamo.com/>, 2008

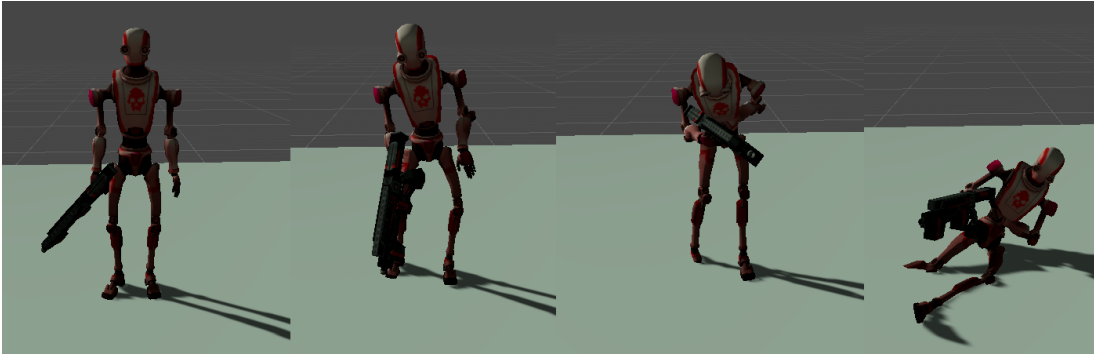


Figure 5.5: A series of animation steps from the dying animation.

behaviour of the enemy and it uses the same effect as the player guns when it fades into and out of visibility.

Where appropriate animations are loosely synced up to audio events. These include the footsteps of all agents; using pitch and time dependant on distance and speed to vary the sounds to seem more appropriate. The firing of the guns trigger a single sound effect that is pitch modulated between shots in order to make it seem more dynamic.

5.3.4.2 Enemy AI

The enemy AI in the game feature basic movement and shooting mechanics. They are able to shoot at and take cover from both the player and the friendly AI units. The enemies have a range of base behaviours that they will complete when the players have not presently engaged them but are able to see them. When they notice the player or a friendly AI they choose between a range of actions that can have them either move to cover or begin shooting from their current location. A 50f unit bounding radius between the enemy and the player is used to determine whether the enemy should begin their behaviours. This could pose a problem in situations where the player has a long line of sight but since our game is designed in such a way that the player generally only has a view of an encapsulated room this is not an issue.

5.3.4.3 Way-point System

In order to keep the consistency with our desired gameplay path and also to minimise the number of issues that can occur, the enemy movement is constricted to movement between predefined way points. With the navmesh system that we are using it is a simple task to implement movement of AI that can navigate the whole level. However, since the system is not perfect and our navmesh is cluttered with obstacles it is probable that in doing this agents will move to position in which they render themselves stuck, or they will go a direction before changing their decision and approaching from a different way if the route seems blocked. Our way-point system tackles these issues by having predefined points that we know the agents can reach reliably.

The system is user planned, where a number of way-points are scattered throughout the level in positions that the designer believes appropriate. Each enemy can then be assigned any number of these points which will act as their personal list of possible locations which they can move to. This allows us to prevent enemies from room 3 trying to move to cover in room 1; it also allows us to give certain enemies specific way-points giving them unique behaviours. This can be seen in our game on some of the enemies on raised platforms. There are two different types of way-point (default, cover) to allow the designer to specify points that are well covered for more interesting behaviour implementation. A secondary list containing way-points that make up an enemies patrol positions exists as its own entity so that specific patrol paths can be placed.

The way-points contain a boolean that dictates their current state (empty, assigned, populated), this allows us to prevent multiple enemies from moving to the same location. The search for positions uses distance to isolate the highest priority way-points for the enemy. If none of the enemy's way-points are empty then it can consider assigned points. These points will check to see if their currently bound enemy can move to a different point. If this is possible then the enemy moves to the previously assigned point and the other enemy moves to their new point. This case is not always run and its alternative, that is chosen based on a random chance, gives the enemy a command that will keep them stationary.

5.3.4.4 Target Selection

Originally the enemies were designed using a raycast based vision approach where they would not locate units unless they were moving within their direct line of sight at a close distance. This method was deemed inappropriate as in testing it was shown to encourage players to focus more on their own movements and not engage with their team mates at all.

The enemies select their target from all the friendly NPCs along with the player using a distance weighted search. The search has two cutoffs for distance: a cutoff of half vision range for if the target is crouched; a cutoff for full vision range when the target is stood. Following this they will attempt a number of shots over which time if they are unable to perform a direct raycast that hits their chosen target then they will attempt to either move to a new location or choose another target to fire on. They run the distance based search for targets once in every second to make sure that they are constantly aware of any nearby enemies.

The enemies have a bias that means that during their search the players position can be buffered to make it a large target. We use a buffer of 3 units in Unity to make the player a more likely target. The enemies also have a secondary method for target selection that will force them to shoot at the player if they are visible from a raycast. This method fires at a rate of 1 in 5 in our game. These methods were implemented after it was mentioned by multiple testers that the player should be under more pressure to encourage them to use their team mates for help. Our balance numbers were selected to be in a range that allowed for some increased challenge to experienced users without making the experience too challenging for beginners.

5.3.4.5 State Machine

The enemy AI bases the core of its behavioural decisions on a state machine that contains behaviours: idle, cover, taking cover, patrol, sneak, shooting, cover shooting and dead. These states are decided using a enumerator variable that

is stored in the class and can be changed from a number of specified locations. Since the enemy checks every frame whether the variable has been changed it will update its behaviour according to state regularly. The flow of state behaviour is designed in such a way that any state that all behaviours contain their own encapsulated timers and therefore do not require any global time setting for tick events such as delay between shots.



Figure 5.6: The enemy in the idle position.

The idle state triggers the gun to fade out of view if it is not already hidden. It stops the agents movement and removes any assignments to way-points that it currently has. It will also stop the agent from shooting and move it to standing if it was crouched.

The cover behaviour implements the same behaviour as the idle state except that the enemy adopts the crouched position.

The taking cover behaviour hides the gun if it is not already hidden and stops the agent making them move to standing and stopping them from shooting. It then checks to see if they have a way-point selected and if not searches for a cover point. If it does not find one then it polls a random decision check, with equal weighting, and makes a decision between: shooting from its current position either crouched or standing; searching for a way-point designated as a regular point. If a



Figure 5.7: The enemy in the crouched (cover/sneak) position.

cover point is assigned then the agent moves towards the point at a running pace. Once the agent reaches this point it will crouch and start the cover behaviour.

The patrol state will fade the gun out and move the player to standing. If no current way-point is active then it will search for a way-point from the enemies specified patrol point list. If a point is had then it will move towards this point at a walking pace. Once the point is reached the next point in the list is searched for, skipping over populated points, and moved towards after a delay between 1-2 seconds. If all points are populated then the state switches to idle.

The sneak behaviour operates in the same fashion as the patrol state except the agent is crouched when moving and resting.

The shooting state will fade the gun into view if it is hidden and stop the agent moving them to standing. Once the gun is fully in view the agent will begin to fire on their selected target. This behaviour has its own tick rate set to go between 0.4 and 0.8 of a second. The gun firing is synced to trigger an animation of knock back for the agent to give more visual indication of the shot. All target selection methods apply here.

The cover shooting behaviour acts in the same way as the shooting behaviour except that the agent is in the crouched position. If the agent assumed this

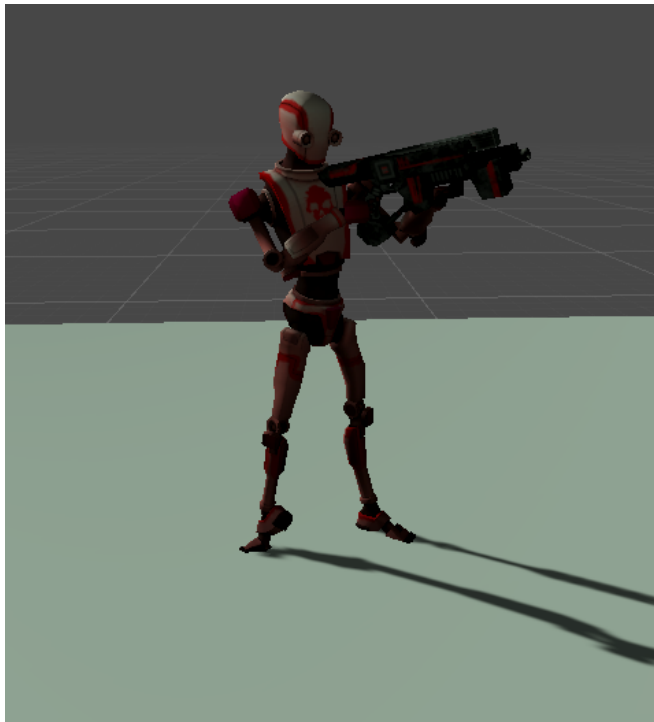


Figure 5.8: The enemy in the shooting position.



Figure 5.9: The enemy in the crouched shooting position.

position from the taking cover state then they will occasionally stop firing and assume the cover state with a timer running to trigger their resumed firing.

The dead state removes any currently bound way-points on the agent. It dissolves the gun out and starts an animation of the agent falling to the ground. The agent

is set to dead and no other state calls will run or can change the state. After the falling animation has complete the enemy dissolves out using the same effect as the guns.

5.3.5 Friendly AI

Originally the friendly AI would make its own decisions as to what to do on random timers if it did not receive any commands from the player. However it became apparent that as the AI's actions were limited there was little need for the player to interact with the AI using commands as it generally did what they wanted it to do. When we stopped the AI from any independent decision making it vastly increased the amount that the players would interact with the AI.

The target selection for friendly agents works using the same base mechanics as the enemy one except there is no bias or external command to reshuffle the targets except when their target dies.

The original implementation of the friendly agents movement system had them free to move to any position on the navmesh. This meant that when selecting where to move, several moderately expensive calls would have to be made in order to find an appropriate position for them to navigate to. This was an issue for player commands as it meant that the player could continuously give commands to the agents which they would seemingly not be responding to. Even though they are receiving the calls find they would not be able to move either because they are blocked or because the location that they are selecting is blocked. While there are several approaches that can be taken to attempt to optimise this system and where many edge cases can be located and accounted for it is still likely that errors will occur and due to the scope of the project this was seen as inappropriate.

5.3.5.1 Cover Point System

In a similar vein to the way-point system we implement a cover point system for friendly AI agents to keep their movements more consistent through different play sessions. This also has the advantage of less ambiguous command behaviours which helps to prevent apparent false gestures.

The cover points are linked to player teleport points where each team mate has one cover point for each teleport point. The points are positioned strategically around the teleport points so as to give the agents cover where possible. They are also positioned such that the agents' placement should not obstruct the player's view. When searching for and moving to a cover point they will each perform a search to find the closest available point.

5.3.5.2 State Machine

The friendly AI bases the core of its behavioural decisions on a state machine that contains behaviours: idle, moving, sneaking, shooting, cover shooting and dead. These states are handled in the same way as the states for the enemy AI; only with different methods for how they are updated.

- The idle state stops the agent's movement and removes any assignments to cover-points that it currently has. It stops the agent and moves them into the standing idle animation.
- The moving behaviour causes the agent to move to their given cover-point at a rushing pace. When at the location they will resume any previous behaviours like shooting or crouching.
- The sneak behaviour causes the agent to move to its given cover-point at a walking pace and with the crouched moving position, upon arrival it resumes previous behaviours.
- Both the shooting and cover shooting states mimic the enemy ones.

- The dead state removes any currently bound cover-points on the agent. They will fall to the ground with the death animation and stay there for a period of 2 seconds before getting back up and moving to the idle state.

5.3.5.3 Receiving Commands

The commands given by the player provide the agents with specific instructions that dictate their use and movement through behaviour states. As we previously mentioned the agents are designed to be “dumb” as it increases the player interaction with them. The commands are absolute in the sense that performing a new one will overwrite any previous commands.

The *Cease Fire* command will put the agents into the idle state, which they will not leave until instructed otherwise. This includes the case where the agent is downed and revives itself.

The *Start Firing* command will put the agent into either the shooting or cover shooting state depending on their current state. It will stop all current movement they are doing and reset their target selection.

The *Follow* command will set the agents cover points to be the ones associated with the player’s current teleport point. The agents will move to the point at a running pace and upon arrival either begin shooting or idling. The decision as to whether they start shooting is dependant upon their last instruction regarding shooting.

The *Rush* command will set the agents cover points to the ones associated with the teleport point after the one the player is currently at and move them there at a running pace. It will override any previous shooting commands and set the agents to search for a new target and start firing when they reach the cover point. It will also move the agents to standing.

The *Sneak* Command will move at random 2 agents to the cover points one ahead of the player and 2 agents to the ones at the players current point. The agents

will move to these at walking pace and crouched; on arrival they will remain crouched and resume and previous firing commands.

5.3.6 Score

The score will not play a major role in the game but will give visible rewards when an enemy is defeated or a bonus is shot. When either an enemy or bonus is shot the score will emerge from the model as a 3D text model that is rotated to face the player. 500 points are rewarded for the defeating an enemy and 200 for hitting a bonus.

5.4 Tutorial



Figure 5.10: An overview of the tutorial level. The player moves from right to left.

The game features a tutorial which aims to introduce the player to the controls for shooting, movement and using gestures. It also stands as a way to get the player familiar with the VR game space before they play the game. The tutorial level consists of 2 teleport points, a single enemy, a bonus object and a screen on which the gestures are played. Since the player will be guided through this tutorial there is no real in game text or prompts to signify to them what they need to do. Firstly the player will defeat an enemy and then they will teleport to

the next location. There they will shoot the bonus object before teleporting again to the last point. At this point they will be facing a large screen on which videos will play demonstrating the motion for the gestures. To move to the next gesture the player must complete the motion of the gesture. Each gesture is shown three times in a random order each time to remove any order bias from the training.

5.5 Conclusion

This chapter covered the design and implementation of the game that is used in the following chapter to evaluate the four systems developed in chapter 3. It gives a thorough description of all of the systems in the game so as to allow for considerations about specific elements of its design. This level of detail allows the chapter to contribute to the research area in a greater way as it allows for any future works to consider and replicate areas of the design in detail. It also explains how the gestures are integrated into the game and how they interact with the AI.

Chapter 6

Evaluation of Gesture Recognition Game

This chapter covers the study that was done using the artefacts described in the previous chapter. It outlines how the study was performed and what data was collected from the game and participants. It then lays out all out all of the results from the study and they are discussed. This chapter completes our 5th and 6th research objectives and in doing so answers our 2nd, 3rd and 4th research questions. The discussion at the end of the chapter covers our thoughts and findings about specific areas of the research. The chapter also contains a thematic analysis which was done on the participant interviews.

6.1 Data Collected

Matching the first study, from each participant we recorded their: age, gender, self determined VR experience and number of hours of games played per week. Along with this we record every time the system thinks the user has input a gesture. The invigilator also records every time the user performs a gesture motion. We record both of these so that we are able to determine which gestures the system got correct and which it confused with others. We record the user information so we can look for any significant relationships between any of the results and a particular user demographic. The invigilator also recorded any notes on the users behaviour and play style that they found. This was done so that any consistencies in behaviour could be found and noted for future design considerations.

We collected a SUS, GEQ, GEQ post game and GEQ social presence questionnaire from each participant after each game. The SUS allows us to evaluate the gesture recognition system with regards to its usability as an interaction system. The GEQ and post game GEQ provide us with an insight into the players experience with the game. The GEQ social presence questionnaire provides us with insight as to whether the NPCs in the game had any social impact on the player.

We also conduct a semi structured interview with the following core questions:

- What did you think of the experience?
- What did you think of the gesture system?
- Would you rather use the gesture or a more traditional button input?
- Do you think VR is a good platform for this system?
- How many times would you be willing to repeat the gestures during the tutorial?
- Would you like to be able to record your own gestures?
- Would you have liked to have seen more unique gestures in the game?
- Any other comments?

The interview allows us to gather qualitative results on the participants experiences with the game. The questions are designed to gather information on whether the user like the system and if they had any ideas for ways which they would like to see it improved. It also aims to gather whether they think that the system is a good fit in VR games.

6.2 Methodology

This study takes a more long form approach than the first. Participants each play two games, answer 2 sets of 4 questionnaires and partake in one interview at the end. Along with this, metrics about their gameplay and interaction are recorded.

To begin the experiment the participants are given a description of the study and are asked to sign the relevant medical and consent forms. They are given time

to ask the invigilator any questions they have about the study. Following this they are given an explanation of the basics of how the game will work and are introduced to the Oculus Rift headset. To account for new users participants are given freedom of time to become comfortable with the headset and the invigilator checks that they are comfortable in VR before they begin the study.

First the participant is taken through a tutorial that covers the basic mechanics of the game including how to perform the gestures. Once this is completed they are allowed to discuss with the invigilator any further questions they have about the game and the invigilator explains the exact effects that each of the gestures have. On completing this the player is put into the explicit game. In the game the participant moves through a level by teleporting to points throughout the level. Along the way they defeat enemies by either shooting them or commanding their teammates to shoot them. They command their teammates using the gesture system. The invigilator watches the players movement and a screen showing their in view in the game. When the player starts a gesture the screen notifies the invigilator and they make note of what gesture the player performs and the time that they did such.

Upon finishing the game the participant is given the SUS questionnaire and the three versions of the game experience questionnaire (main questionnaire, social presence questionnaire, post game questionnaire). They are left to answer these with any required assistance in understanding the wording of the questions being provided by the invigilator. After completing the questionnaires the change in controls is explained by the invigilator and the participant is put back into the Oculus Rift. They are put into the continuous game and the invigilator repeats the recording process. Upon finishing the game the participant is given the same set of questionnaires.

Finally a semi-structured interview is conducted with the questions mentioned in the previous section.

During the interview the participant was encouraged to explore any design feedback that they had.

40 participants took part in the study (22 male, 18 female) with an age range of 18 - 28 and an average age of 21.9. The average self defined VR experience was 5.3 for males and 5.0 for females on a scale from 1-7. Refer to 6.1 for the distribution of number of hour per week spent playing games.

Table 6.1: Average hours of games played per week. Gender left and hours top.

	>1	1-5	5-10	10-20	20+
Male	3	2	7	6	4
Female	5	3	6	3	1

6.3 Results

Figure 6.1 depicts a slight negative correlation between the amount of gestures performed by the user and the accuracy they achieved. This is likely linked to the fact that users would repeat gestures if they performed one incorrectly. This idea is also backed by the fact that on average the users using the static system performed more gestures than those with the adaptive and their accuracy was a significant margin lower. This is shown by the gesture numbers shown in tables 6.3 to 6.6 as well as figure 6.1.

Table 6.2: Accuracy results for all systems

	Correct	Wrong	Missed
Static Explicit	74.4%	25.6%	25.8%
Static Continuous	73.1%	26.9%	57.3%
Adaptive Explicit	88.8%	11.2%	15.4%
Adaptive Continuous	84.1%	15.9%	57.0%

For the explicit recognition method we observe from our work that the adaptive system (M = 88.8%, SD = 8.5%) performs better than the static system (M = 74.4%, SD = 18.6%). This difference, 14.4% was statistically significant, $t(38) = -3.150$, $p < .003$. For the continuous recognition method we find that the adaptive system (M = 84.0%, SD = 10.9%) once again outperforms the static system (M = 73.1%, SD = 18.2%). This difference was also found to be statistically significant, $t(38) = -2.299$, $p < .027$. There was found to be no statistically significant differences or correlations between any other of our participant factors and their accuracy (gender, age, VR experience, game hours played per week).

Figure 6.1: Correlation between accuracy and number of gestures

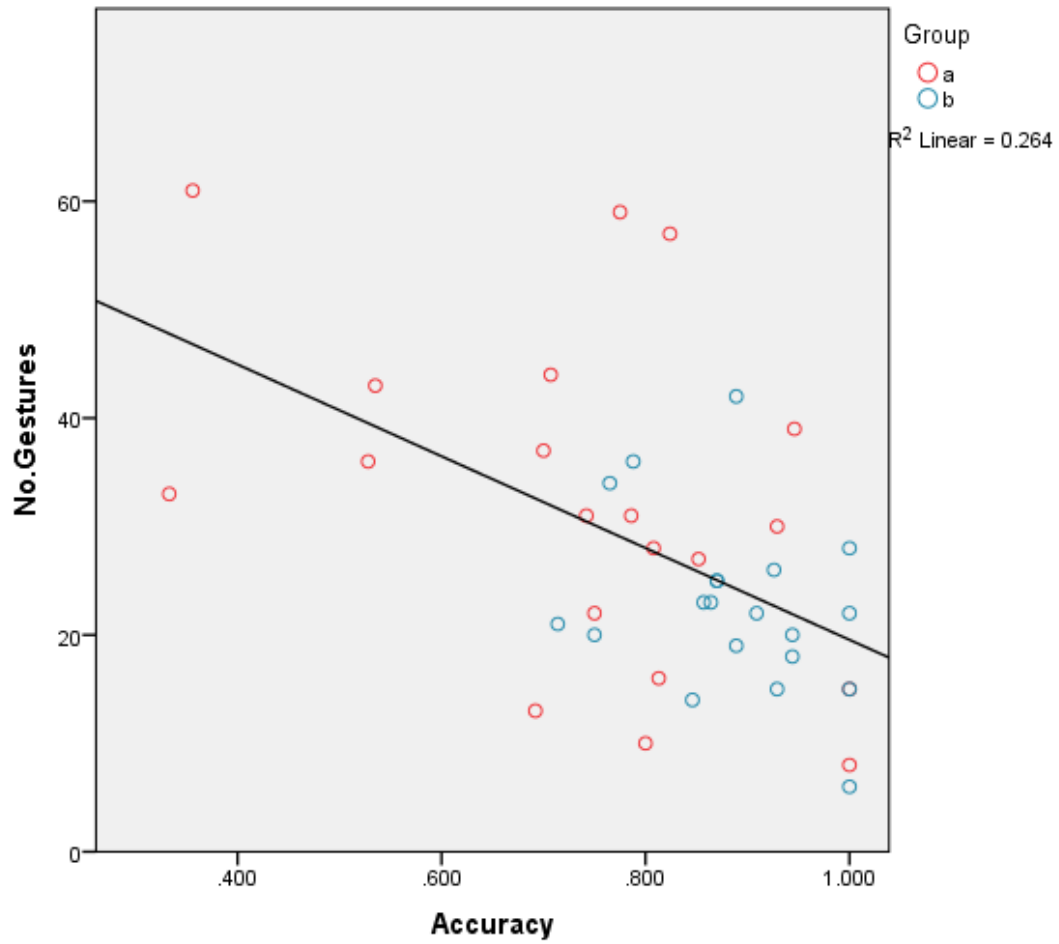


Table 6.3: Confusion matrix for explicit static system

	cease	start	follow	rush	sneak	total
cease	84%	2.6%	11.8%	0.8%	0.8%	119
start	8.9%	77.4%	9.5%	2.4%	1.8%	168
follow	2.9%	1.8%	91.2%	2.9%	1.2%	171
rush	6.4%	2.3%	41.9%	48.8%	0.6%	172
sneak	30%	0%	10%	0%	60%	10

For the static system the *Follow* gesture was by far the most accurate. This could be due to it being the most distinct from the other gestures in terms of movement pattern.

Rush achieved very low accuracy results, its confusion with *Follow* is almost as high as its recognition rate. This makes sense as both gestures take place in the same spatial area and only differ in motion pattern. It was observed that often users would relax the motion they made for *Rush* over time and it is probable that this is the cause behind the high confusion levels.

Table 6.4: Confusion matrix for explicit adaptive system

	cease	start	follow	rush	sneak	total
cease	92.4%	6.1%	1.5%	0%	0%	66
start	4.1%	90.4%	5.5%	0%	0%	73
follow	2.4%	0%	96.6%	1%	0%	207
rush	3.2%	0%	30.1%	66%	0%	94
sneak	0%	0%	0%	0%	100%	14

Matching the explicit system *Rush* once again has a large confusion rate with *Follow*, although it is lower in this system. In our later discussion we consider how different user groups trained the system and how this could have affected these results. For all other gestures we see a consistent increase in accuracy results where confusion rates remain proportional to those in the explicit system.

Table 6.5: Confusion matrix for continuous static system

	cease	start	follow	rush	sneak	total
cease	80.5%	9.2%	3.5%	5.8%	1.1%	87
start	11.7%	76%	9.9%	1.8%	0.6%	171
follow	5.7%	2.3%	92.2%	0%	0.8%	128
rush	7.2%	3.3%	40.5%	48.4%	0.6%	153
sneak	0%	1.1%	5.6%	0%	83.3%	18

Our confusion rates in the static continuous system follow the same pattern as the explicit system with a few outliers such as *Cease* having a higher confusion rate with both *Start* and *Rush*; comparably less with *Follow*.

Table 6.6: Confusion matrix for continuous adaptive system

	cease	start	follow	rush	sneak	total
cease	81.3%	12.5%	3.1%	0%	3.1%	32
start	3.3%	88.4%	7.4%	0%	0.9%	215
follow	1.5%	3.1%	89.7%	5.2%	0.5%	194
rush	8.4%	6.7%	16.8%	68.1%	0%	119
sneak	0%	4%	4%	8%	84%	25

Once again the adaptive version of the system improves rates in line with the static system. *Cease* is an outlier in this circumstance as it only very slightly improves overall accuracy but removes all confusion with *Rush* and increases confusion with *Start* and *Sneak*.

Game Experience results were compared between the two versions. On average the explicit game had higher annoyance/tension scores ($M = 0.48$, $SD = 0.52$)

than the adaptive game ($M = 0.17$, $SD = 0.30$). The difference of 0.31 was found to be statistically significant, $t(38) = 2.346$, $p < .024$. We believe this could be related to the difference in accuracy between the systems.

The mean score for positive effect on the GEQ was 3.31 and the average for flow was 2.94 suggesting that the explicit game has potential as a enjoyable player experience. The continuous game consistently featured lower scores than the explicit game but only by a small margin. None of the differences were found to be statistically significant.

We analyse the SUS results using the "School Grading" and "Adjective Scale" methods discussed in [76] and [77]. Using the Adjective scale we we had the following results for the explicit system: Excellent 1, Good 17, Ok 17, Poor 5. Using the Grading scale we had: 9 B, 14 C, 12 D, 5 F. The average score was ($M = 69.63$, $SD = 11.53$) which is above the average score of 68 for the SUS.

For the continuous system the adjective system gave these results: 1 Best Imaginable, Excellent 2, Good 6, Ok 19, Poor 9, 3 Awful, 1 Worst Imaginable. For the grading scale the results were: 1 A, 5 B, 3 C, 11 D, 20 F. The average score of ($M = 57.13$, $SD = 18.45$) is below the average for SUS scores and is lower than the explicit system. The difference between them was found to be statistically significant $t(38) = 4.70$, $p = 0.004$.

A full table with all the means and standard deviations for all results from the SUS, GEQs and gesture accuracy can be seen in 8.9.

6.3.1 Observed Behaviours

Throughout the studies several behavioural patterns were observed across participants. The first of these was the tendency to group the use of gestures and do them rapidly one after another. An instance where this was commonly seen was when users would enter a new room filled with enemies. Initially they would perform a movement gesture once or twice and then follow it with start firing. Due

to the way the gestures are implemented this meant that the movement gestures were overridden and users would be confused for a moment before realising their teammates were not moving up to them. At which point they would repeat a gesture. It was very common for users perform a gesture twice instead of just once. Considering this behaviour it is possible that future systems could be designed with more direct feedback that prevents users wanting to repeat gestures. Alternatively some systems could stack gesture inputs and move through their behaviours in a queue.

Another common observation was that there were two distinct player types in the way that they interacted with their teammates. The first group were highly proactive in their instruction giving, always making sure that they sent their teammates first or just after them; they would be patient and wait for their movements. The second group were less proactive and would only think to move their teammates when they were in distress. They would move into a room and then attempt to defeat as many enemies as possible and would only gesture to move their teammates in after they became overwhelmed. They would often perform their gestures in a more rushed or panicked way and thus would make more mistakes or confuse the behaviours of the gestures.

It was noticed that some users in training would be strict and rigid with their movements; mimicking the gestures with high degrees of accuracy. However, when they proceeded into the main games they would relax their movements when doing the gestures. On the other hand there were some users that would maintain “looser” gestures throughout their training and the main games. This would have had a significant effect on the results of the adaptive system as it would have been trained using gestures that do not in fact match the users active gestures accurately.

6.3.2 Thematic Analysis

Thematic analysis is a widely used method of qualitative analysis in social science and other research areas. Braun and Clarke [87] detail how thematic analysis is

often used but not documented well and the methodology of its use is also left out which they consider to be a flaw in its use. With this in mind we complete our analysis based on the guide laid out in their work, with the consideration in mind that as they state analysis is not meant to be linear and should be adapted where appropriate for the data.

The interview data was all transcribed directly in groups of about 4-6 interviews at a time. The interviews varied in length largely with some lasting over 7 minutes and some that would only just cover a minute and a half. Throughout the process some overarching areas were noticed to be mentioned frequently. Many participants gave longer and more in depth answers to questions surrounding the tutorial area of the game. Although all participants were encouraged in the interviews to expand on any short answers many of them did not want to give any further details or would get stuck and be unable to explain their thoughts in any greater depth. There was no apparent linking between depth of interview and enjoyment or distaste of the game and again no link to previous game experience. The varied input seemed to be purely based on interest in discussion of the game, with some participants being keen to suggest ideas for improvement.

Since our interviews are semi-structured there is an initial baseline for codes that we expect to extract from the data. However, these are not comprehensive as we encouraged users to expand on any individual information they were willing to disclose. Whenever users would mention an interesting point additional questions would be asked along the lines of “Would you like to explain that in more detail?” and “Why did you think that about it?”. If they were particularly positive or negative around a system without giving any specific details as to why.

6.3.2.1 Immersion and Presence

VR is often accredited as being an immersive experience. The research provided no formal method for evaluating player’s immersion within the game. However, a substantial proportion of our interviewees mentioned that the game felt immersive directly. Several participants specifically labelled this as being improved or related

to the gestures and being able to communicate physically with their teammates; “Being able to turn my head and watch my teammates moving after I did a gesture really adds to the immersion” (participant 64a7). Other participants suggested that the gesture control system is a natural fit for VR games; “In VR, when you have gestures you get really immersed, it’s a much better control method.” (participant 95a3).

Other participants mention how the gestures felt “realistic” and that they “really made them feel like the player”. We take from this that using gestures may also have an effect on the players presence. This is further backed by a number of participants mentioning how it made them more focused on the game and less aware of their surroundings; with one player stating that it “Took you away from the world” (participant 37b8).

Considering the prevalence of research into the areas of Immersion and Presence in VR we suggest that future research furthers this area of exploration. This should be done with the consideration of whether it is simply gestures; or gestures that can be used to interact with *Characters* that have impacts on presence and immersion.

6.3.2.2 Control Differences

One of the most commonly discussed themes was that of the control differences between the explicit and the explicit system. Nearly every participant stated that either the explicit was easier to control or that the continuous was more difficult to control. Along with this there was a substantial number of participants that described the continuous system as being frustrating or annoying.

“I felt more confident in the first one... when I had to use the second one it felt like it kept recognising stuff when I didn’t want it to. I think the first one is just a better idea for the system, you don’t really need to improve upon it” (participant 47b3)

There was a group of participants that discussed the idea that the second system felt more natural and another smaller subset of users explained how they thought the continuous system could be the better system if it was improved.

6.3.2.3 Improved Tutorial

A common theme that almost every participant mentioned in their interview was the desire for an improved tutorial of some form. The most common request was simply for a greater degree of feedback during the tutorial. Some participants expressed the simple desire for a confirmation when they performed a correct action whereas other participants mentioned the desire for a more involved system that would “coach” you and after performing a gesture “tell you how you can improve it, with visual examples” (participant 13a9). It was suggested by the majority of these participants that more feedback would improve their learning of the gestures.

A prevalent theme around the tutorial was the desire for a more involved tutorial in which you are actively taking part in learning that involves interaction closer to the full game. There were several participants that directly mentioned a “more involved tutorial” where you can learn in a “proper level of the game”. Along with this there were other participants that, while not directly suggesting this, described the tutorial as “boring” and said that they would get frustrated with it in a commercial game. We suggest that both improved feedback and a more involved tutorial would prevent the players from getting bored. As well as improving the experience, many participants suggested that using the gestures in an environment that showed off their use cases would help them not only to remember the gestures better but also encourage them to use them more in the game.

While talking about learning the gestures “having it be part of a more involved tutorial where you can see what happens after the gestures would help them to make more sense and for you to remember them better.” (participant 75b3)

There were also several participants that suggested that the gestures should be taught in order instead of randomly. It was thought that having the gestures appear in random order prevented them from remembering any of them specifically as they did not spend significant time with any of them. The random order was only done to remove any order bias in the training when it came to the use of gestures so this is a simple suggestion for any commercial uses of the system.

6.3.2.4 Practice Methods

A prevailing theme among many participants was the desire to be able to practice the gestures with more free will. There were a smaller number of participants that simply wanted to be able to determine their own length for the tutorial but the vast majority of participants displayed interest in more freedom in how they choose to practice. Participants were also keen to suggest a form of reminder similar to the *Cheat Sheet* seen in work by Payne et al. [14].

Considering these suggestions we believe that users would respond well to an optional on screen reminder for how to perform gestures that they are able to remove after they have mastered them. We also propose that users should be able to revisit a variety of training sessions at their own will which they can use to better their knowledge of gestures. It becomes apparent that these ideas are heavily related to improvements in the tutorial and it is possible that having one of them might reduce the need for the other. However, it could also be the case that the opposite is true and a solution that incorporates an amalgamation of the ideas provides the best solution.

6.3.2.5 Customising Gestures

Since there was a direct question on the topic of users recording their own gestures it is not surprising that a theme has emerged from it. However, while the question was focused on the area of players being able to record their own gestures the responses were split between participants wanting to either record completely new

gestures or make small adjustments to existing gestures. Whilst the majority of participants were positive on one of the above ideas there is a substantial minority that did not want the ability and record their own gestures. The responses in this category were not directly opposed to the idea but more so tended to focus around the opinion that it was unnecessary and could over complicate the system: “No, I feel that would over-complicate it. You might forget what you’ve done in the past” (participant 64a7). The idea that users would struggle with memory when it came to their own gestures was a prevailing response in this section with another participant stating: “I think I would forget them pretty quickly, the ones in the game have already been designed and checked to be good.” (participant 57a1). This response also highlights a smaller code that appeared which was around the idea of designed gestures being superior. A number of users explained that with systems like this they trust the designers to have chosen the best possible elements for the system and doubt that they could improve on them. This was non-ubiquitous though as there were a good deal of participants who regardless of their confidence in design felt that being able to personalise the system allows for its possible improvement.

The sub theme of users wanting to be able to adjust the gestures focus on the idea that while it would not always be necessary it is a good fail safe for the system: “Unless they weren’t working out then I might like to be able to change or personalise them if I don’t like them.” (participant 55b4).

6.3.2.6 More Unique Gestures

The majority of participants responded to the idea of unique gestures with a positive response. Of these users around half gave simple examples and responses explaining how they thought it could make the game more interesting or fun. The other half mentioned that they would like the extra or unique gestures to be introduced later in the game. The overarching reasoning for this was to keep the game simple to learn at the start and then build up more interesting gameplay over time to keep players engaged. There was an emerging sub-theme within this of users who wanted to be able to have more direct control over individual units.

This was brought up as being something that could be addressed with gestures for specific units or a system in which you are capable of selecting which unit(s) you wish to address with commands.

There was a smaller number of users who directly discussed how they would like to be able to combine or link pre-existing gestures to form their own more complex commands. Of these participants most mentioned the idea of being able to target commands at specific units or having commands that are only received by specific units. Participant 64a7 mentions both of these ideas, “Maybe giving specific units a series of commands that they then follow through in sequence”.

There were a small number of users that thought that adding in any gestures was unnecessary and could over complicate the system. The majority of these participants also made note that as long as the extra gesture were not necessary then they would not detract from the game.

6.4 Discussion

Of all the gestures *Sneak* was used the least by a large margin. This is possibly due to the fact that arguably it has the most ambiguous behaviour; in the sense that it has a set of 2 actions rather than just one as the rest of the gestures do. It is also possible that it is related to the fact that it is the only gesture that requires a downward motion. It is not unique in being the only lower body gesture as *Start Firing* is also lower and is used frequently. However, it is possible that *Start Firing* is an anomaly as it is an almost necessary gesture; as to make sure that the AI are shooting the user must perform this gesture.

Our adaptive accuracy mean of 88.8% is lower than accuracy results from popular recent works. However, we believe this is due to the nature of our results coming from a more dynamic environment. Our participants were actively engaged in activities that distract them from doing gestures, as well as require them to move their body actively. We believe that this had a significant effect on the accuracy of

gestures. This is a promising and under researched area, especially with systems that are used in VR.

The continuous system's similar accuracy results provide confidence that there is potential for a system that can work using continuous recognition. However, the false detection rate on the system was very high rendering it unusable in its current state, as confirmed by the systems significantly lower SUS scores. Future work should aim to consider methods that can be used to reduce this false positive rate. A starting point for this would be an evaluation of the system using differing thresholds for recognition likelihood. We believe that our threshold was much too low and this made up for a large degree of our incorrect detection. Other future works could consider different methods for the segmentation of how gestures are broken up.

We have expanded on the efforts of [1] in showing that user training at the start of an application can be used to increase accuracies. In this area we have also shown that it is possible to lower the training time using adaptive databases. There are many other ways in which this system can be explored and future work will look to explore all the different possibilities and their use cases.

Our research has demonstrated that there is user interest in a game system that uses this technology in the field of VR. Outlines for how these systems can be designed and improved have also been suggested, such as methods for training, and perceptions on complexity and count of gestures. In particular it became apparent from the thematic analysis that having a well thought out tutorial could be key in improving user experience for gesture based VR games. We suggest that future systems should look to perform extensive testing on tutorials and consider ways in which they allow users to practice throughout experiences. One design suggestion for tutorials that we consider to be practical is that of designing them to be flexible in terms of training time. This allows users to change how long they spend in specific areas of the tutorial dependent on their comprehension time.

Our SUS results suggest that while the system is usable in its current state it is not as good as it should be for commercial use. We believe that with the

implementation of some of the suggested improvements, that we have gathered from both user feedback and system performance, the system could be made to be a highly usable system. We expect that many of the lower SUS scores were related to frustrations with the gesture recognition system misinterpreting gestures.

Future work should consider a longitudinal study where a game is released into the “wild” and we track user metrics over a long period of time. We also consider doing further lab studies that extensively cover the different aspect of adapting the system, as well the introduction of a larger number of gestures over time, and allow participants more learning time.

6.5 Conclusion

This chapter completes our fifth and sixth research questions and in doing so answers our second, third and fourth research questions. It covers the methodology of our study which evaluates all of our gestures recognition systems and outlines what data we collect during the studies. It then gives an in depth explanation of all of the results we obtain from our study. The results cover each gesture recognition system and their respective accuracies and usability results. They also cover the performance of the adaptive database solution. Following these results a thematic analysis is performed on the participant interviews which provides a more qualitative insight into the user experience with the system.

The results are then discussed in detail with thoughts and suggestions for why certain patterns were observed. The findings and results in this chapter make up the main body of contributions that this work offers the wider research area, along with of course the system that was developed and the methods that were used along the way.

Chapter 7

Conclusion

In this final chapter we summarise the entire thesis and present our final discoveries and thoughts. The chapter begins with a summary that covers what was done in each chapter and the results and discoveries it produced. We then perform a final discussion in which we discuss our research objectives and questions in relation to our findings from the studies we ran. The chapter then concludes with a limitation section in which we discuss the overall limitations that we ran into during the completion of the work as well as possible ideas for future work that expands on the work achieved in this document.

7.1 Thesis Summary

7.1.1 Introduction

The introduction briefly introduced the areas of gesture recognition and virtual reality. It covers the current state of both areas and begins to consider how they can be used together, looking at a few of the existing examples. Following this it lays out our motivations for completing this research establishing our reasoning for considering our specific areas. With the motivations in place it outlines our research questions the objectives we set in order to answer these. It concludes with a summary of contributions to our research area.

7.1.2 Literature Review

The review begins discussing gesture recognition in games and virtual reality. It covers how gestures have previously been seen and implemented in research games, where we noticed that there was a lack of games that do not feature gestures as the only input element in the game. It then discusses the current state of control and input methods in virtual reality. A few of the newer pieces of research establish some ground work in the areas of gesture recognition but in more novel ways that take a specialised focus with new technologies for VR. Considering these we outline that there is a gap in the research area for a more traditional gesture recognition system in the field of VR.

Following this the review moves onto gestures with a focus on methods for achieving gesture recognition. It briefly covers what gestures are and how they fit in with human computer interaction. The review covers several gesture recognition system in depth. It evaluates each system with consideration for how well it would work within the context of a game system. Being the commonly used in previous games research as well as being the most robust in smaller system a HMM system is decided upon for our research. We then consider the area of continuous recognition and identify a system that has been used previously that could act as a strong baseline for the implementation of a continuous recognition system in a game.

The review also covers methods for validating gesture systems and establishes an outline for how we evaluate our system. Included in this is a review of CMVs in which we establish their use cases.

7.1.3 Gesture Recognition System

This chapter covers the design, implementation and initial testing of our gesture recognition systems. It takes into account all our findings and groundwork in the area from the review and uses it to build an accurate HMM based recognition system that works with multiple VR devices. This system is expanded on to

be able to work continuously using our findings from the review as a baseline for where to begin with the system. The system is built using the data that was obtained from the results of chapter 4 in which we analyse user movement patterns when performing gestures in VR.

The chapter also covers the development of an adaptive database solution that allows for the gesture database to be modified on a per user basis. It covers several possible methods for implementing such a system as well as how it will be used with our systems.

7.1.4 User Interaction With Gestures in VR

This chapter covers the development and evaluation of a game that was built in order to gather user movement data in VR when performing gestures. The game in this chapter is developed so as to act as a baseline for some of the players mechanics in the larger game used in the gesture recognition study. The results of this chapter inform the implementation of the continuous recognition system in chapter 3. These results include information about patterns in movement in users hand when performing the gestures, as well as information about the length and timings of recording gestures.

7.1.5 Design and Implementation of a Gesture Recognition Game

In order to evaluate our gesture recognition system the development of a game that can support gestures as a core mechanic whilst also being playable with out the use of them in undergone. The development of the game is described in detail so as to allow for replication based of the explanations in this chapter is possible. After explaining how the base mechanics of the friendly AI units works a summary of how the gestures interact with the game is presented. The gestures each relate to a specific command which instructs the AI units to perform a certain action.

The game in this chapter takes in the original game from the first study as a baseline and when complete presents an artefact that is capable of evaluating all four of our gesture recognition systems.

7.1.6 Evaluation of Gesture Recognition Game

With the game completed and described in the previous chapter we now design a study which can be used to evaluate all four of our gesture systems. This chapter firstly covers the methodology for running our study and the data that we collect from it. It then displays all our results from the study covering both game evaluation and evaluation of the gesture systems. The results from the chapter find our gesture recognition system to be successful in VR and that our adaptive database solution has a significant impact on improving the accuracy of the system. It finds the continuous system to perform well with its performance being heavily weighted on the participants' use of it. Considering the results from the SUS the explicit system was found to be a usable system while the continuous system was not. The chapter also covers the specifics for how each gesture fared in accuracy in more depth as well as discussing possible factors that may have affected the system.

The game was found to be enjoyed by the majority of users with a slightly higher degree of positive experience being had in the game featuring the explicit system according to the GEQ results. The chapter outlines any behaviours that were observed in the users whilst playing the game and interacting with the systems.

A thematic analysis was performed on the results of the interviews. The analysis outlined 6 major areas of discussion: immersion and presence, control differences, improved tutorial, practice methods, customising gestures and more unique gestures.

The discussion in the chapter covers the contributions of the study in detail. These being mainly: the explicit system and adaptive database solution as a method that can be used in future VR gesture recognition systems; The possible

reasons for the failing of the continuous system and possible improvements to the system that could be explored; as well as a generally more in depth discussion on specific elements of the system.

7.2 Discussion

Through the completion of our literature review we complete our first research objective and subsequently establish our reasoning for considering our specific research areas.

Through the evaluation of spatial data obtained from users performing gestures inside a VR game this thesis provides evidence that there are patterns in how users both start and end gestures in a VR game. Our findings show that most users will either stop or slow to a pause before beginning and after completing a gesture. We also find that generally users are consistent in the size of gestures they perform, although this can vary per gesture. These findings complete our second research objective and in doing so also answer our first research question.

Our third and fourth research objectives are covered in chapter 3 where we cover the development of our gesture recognition systems and the adaptive database system. These objectives do not directly answer any of our research questions but they do in turn provide the ground work for us to answer the remaining three systems. The fifth and sixth research objectives are completed in chapters 5 and 6 where we design, build, test and evaluate a gesture recognition game.

The results from our GEQs and the participant interviews show that on average users have a positive interaction with our game. Furthermore we show that users respond positively to an optional gesture system being used in the game and that the majority of users actively engage with the system throughout the game. From our thematic analysis of user interviews about this game we outline several areas of importance when designing a game in this context. We believe these can be used in future systems to improve their user appeal. This combined with the respectable accuracy results answer our second research question as overall we

find that users interact positively with a gesture interaction system within a VR game.

While the continuous system we have developed was seen to work on a rudimentary level, there is a lot of room for improvement in most areas of the system. Methods like that from Raffa et, al [66] provide ideas for how systems such as ours could be further improved in order to be integrated into a less powerful device using less demanding computational methods. We believe that a significant failure in the system was the HMM likelihood threshold being too low and that this was the most significant factor in causing the high miss-classification rates. Overall we believe that, despite lower than average SUS scores, users do not have a negative opinion towards the use of continuous systems in game environments. Instead we suggest that the high miss-classification rates in our system were the cause for the lower scores and that a system with less confusion would perform better. However we do find that the need for continuous recognition systems in games might not exist as users are comfortable using explicitly controlled systems. Answering our third research question we find that our continuous recognition system does not perform particularly well within the context of a VR game, but we are hopeful that future improvements to the system could see this change.

Finally, answering our fourth research question, we find that an adaptive database system can have significant positive effects on the accuracy of gesture recognition systems. We show this working in a challenging environment with high levels of user distraction and achieve a substantial increase in accuracy rates. We believe that due to the prevalence of tutorials in games adaptive systems are a perfect fit as they can be integrated with no real cost. From our thematic analysis we also find that for experiences like ours, users are keen to have involved tutorials and options to practice at their own free will. Both of these provide excellent opportunities for the training of the adaptive system.

7.3 Limitations and Future work

One consideration that should be addressed in the future is the way that we link the movement of the player's headset to the gestures. As mentioned previously this was done in order to stop the issue with global positions being different. The issues that this fixed were that of the y rotation of the player being consistent so that no matter which direction they are facing we are able to map the gestures to consistent positions. However, this had a negative effect in the other orientation axis as it was common for players to have their head either moving when they performed the gestures or for them to want to look at the gesture they are performing. A possible solution to this problem could be a method that uses all the data gathered by the VR devices and determines their body orientation using this metric, this is similar to many inverse kinematics systems that are presently being explored in VR.

Like [5] we consider that using multiple classifiers would be a good option for future commercial gesture systems. This could also be used in tandem with pre-segmentation approaches like those used by [60]. We believe that combinations of techniques like this can be used to make systems that are designed for specific use cases and are capable of achieving excellent performance.

One novel idea that could be considered is that of separating sets of gestures using differing buttons or hands. For example one controller with two buttons could be used to dictate two entirely different sets of gestures without over complicating one dictionary. The same can be said for using different hands for different gestures.

Due to the scope of the work one element we did not evaluate was the effect of different dimensions on the classifier. Future works could evaluate the effects of other parameters such as accelerometer data or lengths/times of gestures. There is a possibility here of using time as a modifier for gestures; where the length that a gesture is done for has an impact on what output the gesture has.

As discussed immersion was mentioned several times within our user studies. Considering the salience of immersion improvement within VR research spaces at the moment it would be good to consider evaluating our systems with regards to user immersion.

In 6.1 we state that we use the GEQ, GEQ post-game and GEQ social presence questionnaire as a metric to measure player experience over 7 factors. Since completing the study it has come to light that the core GEQ questionnaire has never been fully validated as a measure and its psychometric properties are yet to be established [87]. Law et al. go on to show that in particular the *Challenge* and *Negative Effect* are particularly problematic and their results should not be considered; in relation to this caution should be taken around results for *Tension* as it is heavily related and influenced by them. Law et al. find the *Positive Affect* and *Immersion* factors to be acceptable so these results can be taken with slightly more confidence. The criticisms of Law et al. are confined to the core GEQ questionnaire; the social presence and post-game questionnaire are excluded from their work. Bearing this in mind we recommend that all results from the core GEQ be interpreted with caution and although they are not directly considered in the criticism we also recommend that caution be taken around the results of the other two GEQ questionnaires.

We previously mentioned in 5.1 that we did not counterbalance the conditions of the gesture recognition study. This is caused by the fact that we had all participants complete the explicit gesture recognition game before the continuous one. Our reasoning for doing this was that from initial user testing with the system less experienced users could not get to grips with the continuous system without previous experience using the explicit system. This would likely have been solved by some form of more involved tutorial or training period that explained the continuous system in more detail; improved training being something that was highlighted as a good improvement from our thematic analysis. Since we did not counterbalance the conditions it is possible that we have an order bias in the results of the systems. We identify two common possible outcomes from this being that users either performed better with the continuous system as they have more experience with the system; or they struggled with the system as they

were used to the explicit system and therefore confused by the change of controls. If we were to repeat the study, having an individual tutorial for each game that participants complete before each game could stand to solve the understanding issue and allow us to counterbalance the game order.

We do not believe that this invalidates the results obtained but it does mean that they should be interpreted with caution considering the possible effects that these limitations provide. In summary, it means that all user experience data from the GEQ may be considered but not should not be used as any form of justification. It also means that both user experience and accuracy results for the continuous system should be considered only in comparison to the explicit system when used after it. While some of the results or themes in the results may still be accurate it is noted that this information should be further validated.

References

- [1] L. Kratz, M. Smith and F. J. Lee, ‘Wiizards: 3D gesture recognition for game play input’, *Proceedings of the 2007 conference on Future Play Future Play 07*, pp. 209–212, 2007.
- [2] ‘Gesture recognition with a Wii controller’, New York, New York, USA: ACM Press, 2008, p. 11.
- [3] L. Connelly, Y. Jia, M. L. Toro, M. E. Stoykov, R. V. Kenyon and D. G. Kamper, ‘A pneumatic glove and immersive virtual reality environment for hand rehabilitative training after stroke’, *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 18, no. 5, pp. 551–559, 2010.
- [4] M. H. Amir, A. Quek, N. R. B. Sulaiman and J. See, ‘DUKE’, in *Proceedings of the 13th International Conference on Advances in Computer Entertainment Technology - ACE2016*, New York, New York, USA: ACM Press, 2016, pp. 1–6.
- [5] S. Cheema, M. Hoffman and J. J. LaViola, ‘3D Gesture classification with linear acceleration and angular velocity sensing devices for video games’, *Entertainment Computing*, vol. 4, no. 1, pp. 11–24, 2013.
- [6] A. Quek and J. See, ‘The invoker: Intuitive gesture mechanics for motion-based shooter RPG’, *2015 Game Physics and Mechanics International Conference, GAMEPEC 2015 - Proceeding*, pp. 6–10, 2015.
- [7] S. S. Rautaray and A. Agrawal, ‘Interaction with virtual game through hand gesture recognition’, *Multimedia, Signal Processing and Communication Technologies (IMPACT), 2011 International Conference on*, pp. 244–247, 2011.
- [8] F. Hong, S. You, M. Wei, Y. Zhang and Z. Guo, ‘MGRA: Motion gesture recognition via accelerometer’, *Sensors (Switzerland)*, vol. 16, no. 4, pp. 1–25, 2016.
- [9] M. Chen, G. Alregib and B. H. Juang, ‘A new 6D motion gesture database and the benchmark results of feature-based statistical recognition’, *2012 IEEE International Conference on Emerging Signal Processing Applications, ESPA 2012 - Proceedings*, pp. 131–134, 2012.
- [10] J. Liu, L. Zhong, J. Wickramasuriya and V. Vasudevan, ‘uWave: Accelerometer-based personalized gesture recognition and its applications’, *Pervasive and Mobile Computing*, vol. 5, no. 6, pp. 657–675, 2009.
- [11] M. Chen, G. AlRegib and B. H. Juang, ‘Feature processing and modeling for 6D motion gesture recognition’, *IEEE Transactions on Multimedia*, vol. 15, no. 3, pp. 561–571, 2013.
- [12] S. S. Rautaray and A. Agrawal, ‘Vision based hand gesture recognition for human computer interaction: a survey’, *Artificial Intelligence Review*, vol. 43, no. 1, pp. 1–54, 2015.

- [13] M. Elmezain, A. Al-Hamadi, J. Appenrodt and B. Michaelis, ‘A Hidden Markov Model-based continuous gesture recognition system for hand motion trajectory’, *2008 19th International Conference on Pattern Recognition*, no. January, pp. 1–4, 2008.
- [14] J. Payne, P. Keir, J. Elgoyhen, M. McLundie, M. Naef, M. Horner and P. Anderson, ‘Gameplay issues in the design of spatial 3D gestures for video games.’, *CHI '06 extended abstracts on Human factors in computing systems - CHI EA '06*, p. 1217, 2006.
- [15] J. Weissmann and R. Salomon, ‘Gesture recognition for virtual reality applications using data gloves and neural networks’, *Neural Networks, 1999. IJCNN'99. . . .*, vol. 3, pp. 2043–2046, 1999.
- [16] T. Mäki-patola, J. Laitinen, A. Kanerva and T. Takala, ‘Experiments with virtual reality instruments’, *Proceedings of the 2005 international Conference on New Interfaces for Musical Expression*, pp. 11–16, 2005.
- [17] M. C. Cabral, C. H. Morimoto and M. K. Zuffo, ‘On the usability of gesture interfaces in virtual reality environments’, *Proceedings of the 2005 Latin American conference on Human-computer interaction - CLIHC '05*, no. OCTOBER 2005, pp. 100–108, 2005.
- [18] Y. Liu, Y. Yin and S. Zhang, ‘Hand Gesture Recognition Based on HU Moments in Interaction of Virtual Reality’, in *2012 4th International Conference on Intelligent Human-Machine Systems and Cybernetics*, IEEE, 2012, pp. 145–148.
- [19] C. Khundam, ‘First person movement control with palm normal and hand gesture interaction in virtual reality’, *2015 12th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, pp. 325–330, 2015.
- [20] S. Lee, K. Park, J. Lee and K. Kim, ‘User Study of VR Basic Controller and Data Glove as Hand Gesture Inputs in VR Games’, *Proceedings - 2017 International Symposium on Ubiquitous Virtual Reality, ISUVR 2017*, pp. 1–3, 2017.
- [21] F. Jiang, X. Yang and L. Feng, ‘Real-time full-body motion reconstruction and recognition for off-the-shelf VR devices’, *Proceedings of the 15th ACM SIGGRAPH Conference on Virtual-Reality Continuum and Its Applications in Industry - VRCAI '16*, pp. 309–318, 2016.
- [22] S. Liszio and M. Masuch, ‘Designing Shared Virtual Reality Gaming Experiences in Local Multi-platform Games’, 2016.
- [23] N. Gillian and J. a. Paradiso, ‘The Gesture Recognition Toolkit’, *Journal of Machine Learning Research*, vol. 15, no. November, pp. 3483–3487, 2014.
- [24] S. Mitra and T. Acharya, ‘Gesture recognition: A survey’, *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, vol. 37, no. 3, pp. 311–324, 2007.
- [25] Q. Chen, N. D. Georganas and E. M. Petriu, ‘Real-time Vision-based Hand Gesture Recognition Using Haar-like Features’, *2007 IEEE Instrumentation & Measurement Technology Conference IMTC 2007*, pp. 1–6, 2007.
- [26] D. Arsenault, ‘A quaternion-based motion tracking and gesture recognition system using wireless inertial sensors’, PhD thesis, 2014.
- [27] F. Quek, D. McNeill, R. Bryll, S. Duncan, X.-F. Ma, C. Kirbas, K. E. McCullough and R. Ansari, ‘Multimodal human discourse: gesture and speech’, *ACM Transactions on Computer-Human Interaction*, vol. 9, no. 3, pp. 171–193, 2002.

- [28] M. Karam and m. c. Schraefel, ‘A Taxonomy of Gestures in Human Computer Interactions’, *Technical Report, Eletronics and Computer Science.*, pp. 1–45, 2005.
- [29] A. Wexelblat, ‘Research challenges in gesture: Open issues and unsolved problems’, in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 1371, 1998, pp. 1–11.
- [30] P. Senin, ‘Dynamic Time Warping Algorithm Review’, *Science*, vol. 2007, no. December, pp. 1–23, 2008.
- [31] A. Akl and S. Valaee, ‘Accelerometer-based gesture recognition via dynamic-time warping, affinity propagation, & compressive sensing’, *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, no. April, pp. 2270–2273, 2010.
- [32] M. Reyes, G. Dominguez and S. Escalera, ‘Featureweighting in dynamic timewarping for gesture recognition in depth data’, in *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, IEEE, 2011, pp. 1182–1188.
- [33] S. Celebi, A. S. Aydin, T. T. Temiz and T. Arici, ‘Gesture Recognition using Skeleton Data with Weighted Dynamic Time Warping’, in *Proceedings of the International Conference on Computer Vision Theory and Applications*, SciTePress - Science, and Technology Publications, 2013, pp. 620–625.
- [34] Z. Lu, X. Chen, Q. Li, X. Zhang and P. Zhou, ‘A hand gesture recognition framework and wearable gesture-based interaction prototype for mobile devices’, *IEEE Transactions on Human-Machine Systems*, vol. 44, no. 2, pp. 293–299, 2014.
- [35] C. J. C. Burges, ‘A Tutorial on Support Vector Machines for Pattern Recognition’, *Data Mining and Knowledge Discovery*, vol. 2, pp. 121–167, 1998.
- [36] J. Wu, G. Pan, D. Zhang and G. Qi, ‘Gesture recognition with a 3-d accelerometer’, *Proceedings of the 6th International Conference on Ubiquitous Intelligence and Computing*, vol. 5585, pp. 25–38, 2009.
- [37] O. Rashid, A. Al-Hamadi and B. Michaelis, ‘A framework for the integration of gesture and posture recognition using HMM and SVM’, *Proceedings - 2009 IEEE International Conference on Intelligent Computing and Intelligent Systems, ICIS 2009*, vol. 4, pp. 572–577, 2009.
- [38] G. Marin, F. Dominio and P. Zanuttigh, ‘Hand gesture recognition with leap motion and kinect devices’, *2014 IEEE International Conference on Image Processing, ICIP 2014*, pp. 1565–1569, 2014.
- [39] Z. He, ‘Accelerometer based gesture recognition using fusion features and SVM’, *Journal of Software*, vol. 6, no. 6, pp. 1042–1049, 2011.
- [40] L. Porzi, S. Messelodi, C. M. Modena and E. Ricci, ‘A smart watch-based gesture recognition system for assisting people with visual impairments’, *Proceedings of the 3rd ACM international workshop on Interactive multimedia on mobile & portable devices - IMMPPD '13*, no. October, pp. 19–24, 2013.
- [41] R. Lun and W. Zhao, *A Survey of Applications and Human Motion Recognition with Microsoft Kinect*, 05. 2015, vol. 29, p. 1 555 008.
- [42] M.-H. Yang and N. Ahuja, ‘Extraction and classification of visual motion patterns for hand gesture recognition’, *Proceedings of the 1998 IEEE*

- Computer Society Conference on Computer Vision and Pattern Recognition*, no. 1998, pp. 892–897, 1998.
- [43] M. H. Yang, N. Ahuja and M. Tabb, ‘Extraction of 2D motion trajectories and its application to hand gesture recognition’, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 8, pp. 1061–1074, 2002.
- [44] C. Wöhler and J. K. Anlauf, ‘An adaptable time-delay neural-network algorithm for image sequence analysis’, *IEEE Transactions on Neural Networks*, vol. 10, no. 6, pp. 1531–1536, 1999.
- [45] L. Sigal, S. Sclaroff and V. Athitsos, ‘Skin Color-Based Video Segmentation under Time-Varying Illumination’, 2003.
- [46] P. K. Pisharady and M. Saerbeck, ‘Recent methods and databases in vision-based hand gesture recognition: A review’, *Computer Vision and Image Understanding*, vol. 141, pp. 152–165, 2015.
- [47] D. RUMELHART, G. HINTON and R. WILLIAMS, ‘Learning Internal Representations by Error Propagation’, in *Readings in Cognitive Science*, Elsevier, 1988, pp. 399–421.
- [48] M. D. S. Anjo, E. B. Pizzolato and S. Feuerstack, ‘A real-time system to recognize static gestures of Brazilian sign language (libras) alphabet using Kinect’, *Proceedings of the 11th Brazilian Symposium on Human Factors in Computing Systems - IHC '12*, vol. 5138, pp. 259–268, 2012.
- [49] M. Rehm, N. Bee and E. André, ‘Wave like an Egyptian: accelerometer based gesture recognition for culture specific interactions’, *Proceedings of the 22nd British HCI Group Annual Conference on People and Computers: Culture, Creativity, Interaction*, pp. 13–22, 2008.
- [50] Y. Freund and R. E. Schapire, ‘A decision-theoretic generalization of on-line learning and an application to boosting’, in, 1995, pp. 23–37.
- [51] Microsoft, *Kinect gesture builder*, 2014. [Online]. Available: [https://docs.microsoft.com/en-us/previous-versions/windows/kinect/dn785529\(v=ieeb.10\)](https://docs.microsoft.com/en-us/previous-versions/windows/kinect/dn785529(v=ieeb.10)).
- [52] M. Hoffman, P. Varcholik and J. J. LaViola, ‘Breaking the status quo: Improving 3D gesture recognition with spatially convenient input devices’, in *2010 IEEE Virtual Reality Conference (VR)*, IEEE, 2010, pp. 59–66.
- [53] D. Rubine, ‘Specifying gestures by example’, *ACM SIGGRAPH Computer Graphics*, vol. 25, no. 4, pp. 329–337, 1991.
- [54] L. Rabiner, ‘A tutorial on hidden Markov models and selected applications in speech recognition’, *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.
- [55] X. D. Huang, Y Arik and M. A. Jack, ‘Briefly Noted Hidden Markov Models for Speech Recognition’, Tech. Rep., 1990.
- [56] J. Yang and Y. Xu, ‘AD-A282 845 Hidden Markov Model for Gesture Recognition’, Tech. Rep., 1994.
- [57] T. Starner and A. S. Pentland, ‘Real-Time American Sign Language Recognition Hidden Markov Models from Video Using’, *AAAI Technical Report FS-96-05*, pp. 109–116, 1996.
- [58] Hyeon-Kyu Lee and J. Kim, ‘An HMM-based threshold model approach for gesture recognition’, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 10, pp. 961–973, 1999.
- [59] T Schlomka and T. C. Mo, ‘The Viterbi Algorithm’, *Technology*, pp. 302–309, 1972.

- [60] P. Premaratne, S. Yang, P. Vial and Z. Ifthikar, ‘Centroid tracking based dynamic hand gesture recognition using discrete Hidden Markov Models’, *Neurocomputing*, vol. 228, no. February 2016, pp. 79–83, 2017.
- [61] A. Truong and T. Zaharia, ‘Laban movement analysis and hidden Markov models for dynamic 3D gesture recognition’, *Eurasip Journal on Image and Video Processing*, vol. 2017, no. 1, 2017.
- [62] D. Wilson and A. Wilson, ‘Gesture Recognition Using The XWand’, 2004.
- [63] J. Mäntyjärvi, J. Kela, P. Korpipää and S. Kallio, ‘Enabling fast and effortless customisation in accelerometer based gesture interaction’, *Proceedings of the 3rd international conference on Mobile and ubiquitous multimedia - MUM '04*, no. November 2014, pp. 25–31, 2004.
- [64] J. Kela, P. Korpipää, J. Mäntyjärvi, S. Kallio, G. Savino, L. Jozzo and S. Di Marca, ‘Accelerometer-based gesture control for a design environment’, *Personal and Ubiquitous Computing*, vol. 10, no. 5, pp. 285–299, 2006.
- [65] Anthony D. Whitehead, ‘Gesture Recognition with Accelerometers for Game Controllers, Phones and Wearables’, *GSTF Journal on Computing (JoC)*, vol. 3, no. 4, 2018.
- [66] G. Raffa, J. Lee, L. Nachman and J. Song, ‘Don’t slow me down: Bringing energy efficiency to continuous gesture recognition’, in *Proceedings - International Symposium on Wearable Computers, ISWC*, 2010.
- [67] Jung-Bae Kim, Kwang-Hyun Park, Won-Chul Bang and Z. Bien, ‘Continuous gesture recognition system for Korean sign language based on fuzzy logic and hidden Markov model’, in *2002 IEEE World Congress on Computational Intelligence. 2002 IEEE International Conference on Fuzzy Systems. FUZZ-IEEE'02. Proceedings (Cat. No.02CH37291)*, vol. 2, IEEE, 2002, pp. 1574–1579.
- [68] G. D. Evangelidis, G. Singh and R. Horaud, ‘Continuous gesture recognition from articulated poses’, in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 8925, 2015, pp. 595–607.
- [69] Y. Yin, ‘Real-time Continuous Gesture Recognition for Natural Multimodal Interaction’, = *Bulletin of Center for Educational Research and Development*, no. 21, pp. 1–8, 2014.
- [70] J. Wan, S. Z. Li, Y. Zhao, S. Zhou, I. Guyon and S. Escalera, ‘ChaLearn Looking at People RGB-D Isolated and Continuous Datasets for Gesture Recognition’, in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 2016, pp. 761–769.
- [71] P. Wang, W. Li, S. Liu, Y. Zhang, Z. Gao and P. Ogunbona, ‘Large-scale Continuous Gesture Recognition Using Convolutional Neural Networks’, Tech. Rep., 2016.
- [72] N. C. Camgoz, S. Hadfield, O. Koller and R. Bowden, ‘Using Convolutional 3D Neural Networks for User-independent continuous gesture recognition’, in *Proceedings - International Conference on Pattern Recognition*, 2017, pp. 49–54.
- [73] X. Chai, Z. Liu, F. Yin, Z. Liu and X. Chen, ‘Two streams Recurrent Neural Networks for Large-Scale Continuous Gesture Recognition’, in *Proceedings - International Conference on Pattern Recognition*, 2017, pp. 31–36.
- [74] R.-D. Vatavu, L. Anthony and J. O. Wobbrock, ‘Gesture heatmaps: Understanding gesture performance with colorful visualizations’, in

- Proceedings of the 16th International Conference on Multimodal Interaction*, ACM, 2014, pp. 172–179.
- [75] A. S.f. P. Affairs, ‘System Usability Scale (SUS)’, 2013.
- [76] A. Bangor, P. Kortum and J. Miller, ‘Determining what individual SUS scores mean: Adding an adjective rating scale’, *Journal of usability studies*, vol. 4, no. 3, pp. 114–123, 2009.
- [77] J. Brooke, ‘SUS : A Retrospective’, *Journal of Usability Studies*, vol. 8, no. 2, pp. 29–40, 2013.
- [78] K. Poels, Y. A. W. de Kort and W. A. IJsselsteijn, ‘The Game Experience Questionnaire’, *IJsselsteijn, W.A. de Kort, Y.A.W. Poels, K.*, 2013.
- [79] Roberts Jonathan C; ‘State of the Art:\nCoordinated & Multiple Views in Exploratory Visualization’, *Fifth International Conference on Coordinated and Multiple Views in Exploratory Visualization*, no. Cmv, pp. 61–71, 2007.
- [80] M. Q. Wang Baldonado, A. Woodruff and A. Kuchinsky, ‘Guidelines for using multiple views in information visualization’, *Proceedings of the working conference on Advanced visual interfaces - AVI '00*, no. February 2013, pp. 110–119, 2000.
- [81] J. Wang and R. Lindeman, ‘Coordinated 3D Interaction in Tablet-and HMD-Based Hybrid Virtual Environments’, 2014.
- [82] F. G. Hofmann, P. Heyer and G. Hommel, ‘Velocity profile based recognition of dynamic gestures with discrete Hidden Markov Models’, in, 1998, pp. 81–95.
- [83] L. E. Baum and T. Petrie, ‘Statistical Inference for Probabilistic Functions of Finite State Markov Chains’, *The Annals of Mathematical Statistics*, vol. 37, no. 6, pp. 1554–1563, 1966.
- [84] ‘VISUAL SIGNALS 2017 Visual Signals Contents’, Tech. Rep.
- [85] E. Chang, I. Hwang, H. Jeon, Y. Chun, H. T. Kim and C. Park, ‘Effects of rest frames on cybersickness and oscillatory brain activity’, *2013 International Winter Workshop on Brain-Computer Interface, BCI 2013*, pp. 62–64, 2013.
- [86] S. Freitag, B. Weyers and T. W. Kuhlen, ‘Examining Rotation Gain in CAVE-like Virtual Environments’, *IEEE Transactions on Visualization and Computer Graphics*, vol. 22, no. 4, pp. 1462–1471, 2016.
- [87] E. L.-C. Law, F. Brühlmann and E. D. Mekler, ‘Systematic review and validation of the game experience questionnaire (geq)-implications for citation and reporting practice’, in *Proceedings of the 2018 Annual Symposium on Computer-Human Interaction in Play*, ACM, 2018, pp. 257–270.
- [88] Braun, V. and Clarke, V., 2006. Using thematic analysis in psychology. *Qualitative research in psychology*, 3(2), pp.77-101.

Chapter 8

Appendices

8.1 Examples of logged data from First Study

HeadX	HeadY	HeadZ	HeadRotX	HeadRotY	HeadRotZ	RightX	RightY	RightZ	RightRotX	RightRotY	RightRotZ	LeftX
-0.4361613	2.204123	-3.306857	0.06156465	0.07096734	0.02167186	-0.3884069	2.150887	-2.216752	0.2279628	-0.03303751	-0.04623203	-0.74446911
-0.4361826	2.204109	-3.308653	0.06080273	0.06914116	0.02279707	-0.3898061	2.148554	-2.214693	0.2254245	-0.03070654	-0.04620202	-0.7444404
-0.4361826	2.204109	-3.308653	0.06080273	0.06914116	0.02279707	0.1574141	2.138012	-2.356748	0.2728234	-0.02453561	-0.05307259	-1.078215
-0.4358071	2.203374	-3.309444	0.05974108	0.06688582	0.02310808	-0.3939819	2.143125	-2.213198	0.2362544	-0.01601393	-0.03569261	-0.744125
-0.4357263	2.202914	-3.310226	0.05941147	0.06566719	0.02279531	-0.3987986	2.136259	-2.210708	0.2439246	-0.002491671	-0.03306606	-0.7452919
-0.4357263	2.202914	-3.310226	0.05941147	0.06566719	0.02279531	0.1596374	2.125594	-2.337251	0.2895351	0.02545229	-0.02786682	-1.080148
-0.4357089	2.202669	-3.310692	0.05913939	0.06441852	0.02227521	-0.4092465	2.120403	-2.204527	0.2372524	-0.004852617	-0.06281794	-0.7535102
-0.435641	2.202591	-3.31109	0.05838288	0.06379086	0.02189174	-0.4084857	2.118745	-2.202858	0.2270353	-0.01237232	-0.05700983	-0.7551407
-0.435641	2.202591	-3.31109	0.05838288	0.06379086	0.02189174	0.1580202	2.109473	-2.352058	0.2795846	-0.0172445	-0.0686435	-1.09083
-0.4356066	2.202346	-3.311248	0.0578825	0.06342402	0.02169622	-0.4072376	2.115945	-2.200431	0.2103013	-0.02014976	-0.04164436	-0.755815
-0.4356591	2.202151	-3.311343	0.05726777	0.06301538	0.02187415	-0.4077852	2.115266	-2.19945	0.1988467	-0.02189761	-0.03102645	-0.7550589
-0.4356679	2.201885	-3.311419	0.05656113	0.06278961	0.02213263	-0.4071693	2.115325	-2.200084	0.1992191	-0.01327902	-0.02715906	-0.7539095
-0.4356529	2.201628	-3.311406	0.0560371	0.06277696	0.02232266	-0.4086065	2.113358	-2.200022	0.1978454	-0.01567324	-0.02671373	-0.7537934
-0.4356665	2.20143	-3.311366	0.05525372	0.06280866	0.02265134	-0.41039	2.111412	-2.200206	0.1982474	-0.01587093	-0.02890565	-0.7537697
-0.435757	2.201346	-3.311271	0.05462085	0.06241519	0.02269771	-0.4124067	2.10935	-2.20064	0.2002036	-0.01251467	-0.03092315	-0.7540733
-0.4358324	2.201319	-3.311145	0.05409834	0.06237059	0.02296551	-0.4140989	2.107289	-2.200623	0.2010397	-0.0106644	-0.03102969	-0.7543266
-0.4357732	2.201232	-3.311045	0.0533839	0.06262268	0.02241678	-0.415075	2.105347	-2.200205	0.1992053	-0.01038953	-0.03020828	-0.7540903
-0.4357769	2.201121	-3.310918	0.05300824	0.06286386	0.02238694	-0.4157199	2.103325	-2.19937	0.1938799	-0.01165379	-0.02924367	-0.753399
-0.4357769	2.201121	-3.310918	0.05300824	0.06286386	0.02238694	0.1625636	2.093573	-2.331536	0.2533247	0.009705162	-0.02614633	-1.094215
-0.4358108	2.200853	-3.31065	0.05172576	0.06272399	0.02209891	-0.4158641	2.10112	-2.198196	0.1840643	-0.0135358	-0.02598342	-0.7523977
-0.4358507	2.200647	-3.310408	0.05119263	0.06256426	0.02173816	-0.4159478	2.098884	-2.196756	0.1710451	-0.01390959	-0.02235427	-0.7510491
-0.4357886	2.200499	-3.310161	0.05050334	0.06272364	0.02135681	-0.4160837	2.096491	-2.195486	0.1610296	-0.01463708	-0.01888288	-0.7498165
-0.4359255	2.200457	-3.309986	0.05023067	0.062603	0.02183796	-0.4171392	2.094594	-2.193872	0.1553298	-0.01914652	-0.01869107	-0.7487563
-0.4360003	2.200417	-3.309749	0.04997513	0.06254482	0.02198664	-0.4173117	2.092396	-2.194521	0.1580527	-0.01697039	-0.01890375	-0.7478257
-0.4360004	2.200413	-3.309438	0.04924383	0.062943	0.02242468	-0.418356	2.090341	-2.194513	0.1604358	-0.01517772	-0.01968733	-0.7471555
-0.43612	2.200466	-3.309201	0.04916632	0.06260452	0.02251485	-0.4203268	2.088476	-2.194072	0.1595544	-0.01608351	-0.02447085	-0.7466713
-0.4361674	2.200506	-3.308905	0.04878284	0.06250113	0.02249699	-0.4221338	2.086351	-2.193851	0.1579955	-0.01381331	-0.0281992	-0.7460641
-0.4362728	2.200531	-3.308479	0.04805314	0.06233351	0.02271655	-0.4237307	2.084247	-2.19344	0.1563032	-0.01219441	-0.03008401	-0.7452817
-0.4363543	2.200587	-3.308276	0.04794808	0.06224658	0.02265138	-0.425005	2.082373	-2.192979	0.1544293	-0.01019907	-0.0303925	-0.7444208

LeftY	LeftZ	LeftRotX	LeftRotY	LeftRotZ	Gesturing	GestureTime	GestureNumber
2.058854	-2.288411	0.07310255	0.1147851	0.1011172	FALSE	FALSE	0
2.056432	-2.286494	0.07619345	0.1162948	0.1075422	FALSE	FALSE	0
2.080513	-2.418716	0.05969897	0.0817555	0.0775654	FALSE	FALSE	0
2.04949	-2.280833	0.08688802	0.1291192	0.1379375	FALSE	FALSE	0
2.042697	-2.27358	0.09449086	0.1473016	0.168719	FALSE	FALSE	0
2.068226	-2.411781	0.1636821	0.1251473	0.1162617	FALSE	FALSE	0
2.033587	-2.259348	0.07949112	0.1658752	0.1653872	FALSE	FALSE	0
2.032888	-2.258352	0.07004605	0.1570701	0.1434871	FALSE	FALSE	0
2.056882	-2.404945	0.1301146	0.1285919	0.1004989	FALSE	FALSE	0
2.030594	-2.25512	0.06707624	0.1607069	0.1454911	FALSE	FALSE	0
2.029548	-2.254735	0.07092988	0.1614031	0.152665	FALSE	FALSE	0
2.029901	-2.254672	0.07071151	0.1630748	0.1589786	FALSE	FALSE	0
2.029171	-2.253608	0.07069703	0.1635076	0.1596148	FALSE	FALSE	0
2.028855	-2.252552	0.0690461	0.1635403	0.1571036	FALSE	FALSE	0
2.029227	-2.251881	0.06669399	0.1618125	0.1473799	FALSE	FALSE	0
2.029637	-2.251343	0.06370129	0.1590911	0.1354357	FALSE	FALSE	0
2.029819	-2.251055	0.06136458	0.1560271	0.1259658	FALSE	FALSE	0
2.029372	-2.250687	0.0586685	0.1531995	0.121019	FALSE	FALSE	0
2.051875	-2.398636	0.1165845	0.1199315	0.08182914	FALSE	FALSE	0
2.028573	-2.250206	0.05596174	0.151482	0.1201748	FALSE	FALSE	0
2.027741	-2.249955	0.05552294	0.1504083	0.1220731	FALSE	FALSE	0
2.027071	-2.249747	0.0560619	0.1495824	0.1222463	FALSE	FALSE	0
2.02628	-2.249308	0.05481156	0.148712	0.1210404	FALSE	FALSE	0
2.025324	-2.248658	0.05230812	0.1479803	0.1185921	FALSE	FALSE	0
2.024376	-2.247803	0.04868918	0.1462035	0.1128791	FALSE	FALSE	0
2.023621	-2.246937	0.0440389	0.1439333	0.1060437	FALSE	FALSE	0
2.023013	-2.24615	0.04000307	0.1404855	0.09751016	FALSE	FALSE	0
2.022608	-2.245446	0.03684072	0.1369881	0.08882655	FALSE	FALSE	0
2.022357	-2.244909	0.03405173	0.1331605	0.08018377	FALSE	FALSE	0

8.2 Example consent form

INFORMED CONSENT FORM

Title of Project: Evaluating the effect of gesture interaction on player experience in virtual reality.

Researcher: Marlon Gilliam

Supervisors: Dr Chris Headleand, Dr Jussi Holopainen

1. I confirm that I have read and understood the Participant Information sheet provided for the above study. I have the opportunity to consider the information, ask questions and have had them answered satisfactorily.

2. I understand that my participation is voluntary and that I am free to withdraw at any time without giving a reason.

3. I agree to have the audio of my interview recorded. I understand that this data will be anonymised and associated only with my participant number.

4. Should I wish to withdraw from the study I understand how I can action this.

5. I agree to take part in the above study.

Participant's Name

Participant's Signature

Researcher's Signature

Date

8.3 Participant information Questionnaire

PARTICIPANT QUESTIONNAIRE

What gender do you identify as?

Male Female Prefer not to say Other (please specify) _____

How old are you?

On average how many hours a week do you spend playing games?

>1 hour 1 – 5 hours 5 – 10 hours 10 – 20 hours 20+ hours

How experienced are you with virtual reality games and applications?

No Experience Highly Experienced

1 2 3 4 5 6 7

8.4 Game experience questionnaire

Game Experience Questionnaire

Please indicate how you felt whilst playing the game for each of the items, on the following scale. Answer by circling your response.

1. I felt content.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

2. I felt skilful.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

3. I was interested in the game's story.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

4. I thought it was fun.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

5. I was fully occupied with the game.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

6. I felt happy.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

7. It gave me a bad mood.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

8. I thought about other things.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

9. I found it tiresome.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

10. I felt competent.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

11. I thought it was hard.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

12. It was aesthetically pleasing.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

13. I forgot everything around me.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

14. I felt good.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

15. I was good at it.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

16. I felt bored.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

17. I felt successful.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

18. I felt imaginative.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

19. I felt that I could explore things.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

20. I enjoyed it.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

21. I was fast at reaching the game's targets.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

22. I felt annoyed.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

23. I felt pressured.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

24. I felt irritable.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

25. I lost track of time.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

26. I felt challenged.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

27. I found it impressive.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

28. I was deeply concentrated in the game.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

29. I felt frustrated.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

30. It felt like a rich experience.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

31. I lost connection with the outside world.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

32. I felt time pressure.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

33. I had to put a lot of effort into it.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

Game Experience Questionnaire

Please indicate how you felt whilst playing the game for each of the items, on the following scale. Answer by circling your response.

1. I empathized with the others.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

2. My actions depended on the others actions.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

3. The other's actions were dependant on my actions.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

4. I felt connected to the others.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

5. The others paid close attention to me.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

6. I paid close attention to the others.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

7. I felt jealous about the others.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

8. I found it enjoyable to be with the other(s).

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

9. When I was happy, the others were happy.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

10. When the others were happy, I was happy.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

11. I influenced the mood of the others.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

12. I was influenced by the mood of the others.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

13. I admired the others.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

14. What the others did affected what I did.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

15. What I did affected what the others did.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

16. I felt revengeful.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

17. I felt schadenfreude (malicious delight).

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

Game Experience Questionnaire

Please indicate how you felt whilst playing the game for each of the items, on the following scale. Answer by circling your response.

1. I felt revived.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

2. I felt bad.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

3. I found it hard to get back to reality.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

4. I felt guilty.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

5. It felt like a victory.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

6. I found it a waste of time.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

7. I felt energised.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

8. I felt satisfied.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

9. I felt disoriented.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

10. I felt exhausted.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

11. I felt that I could have done more useful things.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

12. I felt powerful.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

13. I felt weary.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

14. I felt regret.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

15. I felt ashamed.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

16. I felt proud.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

17. I had a sense that I had returned from a journey.

Not at all	Slightly	Moderately	Fairly	Extremely
0	1	2	3	4

8.5 Participant information sheet

PARTICIPANT INFORMATION SHEET

Title of Project: Evaluating the effect of gesture interaction on player experience in virtual reality.

Researcher: Marlon Gilliam

Supervisors: Dr Chris Headleand, Dr Jussi Holopainen

Project Description: This study aims to investigate the use of gesture interaction within a virtual reality video game.

What will happen: First the researcher will explain to you the details of the hardware you will be using and how to use it. They will explain the controls on the hardware and what interactions they have within the game. Whilst explaining these controls they will also introduce you to the game and how it will work. You will put on an Oculus Rift VR headset and will be put into the game environment. You will be able to spend some time practicing moving around in the virtual world before the game begins. You will first play a tutorial level consisting of about 3-5 minutes of gameplay. You will then play through 2 versions of the game answering questionnaires after both sessions. You will spend between 8-15 minutes in the game for both of these sessions. After the last questionnaires you will be asked some interview questions by the researcher.

Possible Risks: The virtual environment is safe to use, only requiring you to move around a small location. A possible risk is hitting the confines of the physical test area, to mitigate this the researcher will warn you if you approach the test area boundaries.

Time Commitment: The study will typically take between 40 – 60 minutes to complete.

Participant's Rights: (i) You may withdraw from this research study at any time without explanation; (ii) You have the right to omit or refuse to answer or respond to any question asked of you; (iii) Providing they don't interfere with the study's outcome, you have the right to have any questions answered.

Confidentiality/Anonymity: All data collected from this study will be anonymised, and there will be no way to link it to any personal information to which you can be identified (e.g. your name, email address, etc.). After the study is completed, your anonymised data may be made available to other researchers via accessible data repositories and possibly used for publications.

Further Information: If you have any further questions after reading this information sheet, please ask the researcher before the study begins.

8.6 Participant takeaway slip

PARTICIPANT TAKEAWAY SLIP

Title of Project: Evaluating the effect of gesture interaction on player experience in virtual reality.

Researcher: Marlon Gilliam

Supervisors: Dr Chris Headleand, Dr Jussi Holopainen

Project Description: This study aims to investigate the use of gesture interaction within a virtual reality video game.

If have any further questions regarding this study please feel free to contact the lead researcher below.
Marlon Gilliam
mgilliam@lincoln.ac.uk
07898718779

Participant Number: _____

PARTICIPANT TAKEAWAY SLIP

Title of Project: Evaluating the effect of gesture interaction on player experience in virtual reality.

Researcher: Marlon Gilliam

Supervisors: Dr Chris Headleand, Dr Jussi Holopainen

Project Description: This study aims to investigate the use of gesture interaction within a virtual reality video game.

If have any further questions regarding this study please feel free to contact the lead researcher below.
Marlon Gilliam
mgilliam@lincoln.ac.uk
07898718779

Participant Number: _____

8.7 Sus questionnaire

System Usability Scale

Please rate the following questions on the scale by circling the response you agree with the most.

1. I think that I would like to use this system frequently.

Strongly Disagree **Disagree** **Neither agree nor disagree** **Agree** **Strongly Agree**

2. I found the system unnecessarily complex.

Strongly Disagree **Disagree** **Neither agree nor disagree** **Agree** **Strongly Agree**

3. I thought the system was easy to use.

Strongly Disagree **Disagree** **Neither agree nor disagree** **Agree** **Strongly Agree**

4. I think that I would need the support of a technical person to be able to use this system.

Strongly Disagree **Disagree** **Neither agree nor disagree** **Agree** **Strongly Agree**

5. I found the various functions in this system were well integrated.

Strongly Disagree **Disagree** **Neither agree nor disagree** **Agree** **Strongly Agree**

6. I thought there was too much inconsistency in this system.

Strongly Disagree **Disagree** **Neither agree nor disagree** **Agree** **Strongly Agree**

7. I would imagine that most people would learn to use this system.

Strongly Disagree **Disagree** **Neither agree nor disagree** **Agree** **Strongly Agree**

8. I found the system very cumbersome to use.

Strongly Disagree **Disagree** **Neither agree nor disagree** **Agree** **Strongly Agree**

9. I felt very confident using the system.

Strongly Disagree **Disagree** **Neither agree nor disagree** **Agree** **Strongly Agree**

10. I needed to learn a lot of things before I could get going with this system.

Strongly Disagree **Disagree** **Neither agree nor disagree** **Agree** **Strongly Agree**

8.8 VR medical consent form

Medical Screening Form

We operate this study according to the University of Lincoln School of Computer Science Health and Safety Guidelines for Virtual Reality (VR) equipment. However, before you take part, it is important to determine whether you have any conditions which might impair your ability to use the VR equipment safely or otherwise pose harm to your person.

Please circle either 'yes' of 'no' to answer the following questions. If you need any help of wish to ask for further clarification, please ask:

Do you suffer from Epilepsy, or a similar condition which may be triggered by flashing lights of visual stimulus?	YES/NO
Do you suffer from any significant uncorrected problems with your vision, such as tunnel vision? (this excludes the requirement for glasses or contact lenses).	YES/NO
Are you pregnant?	YES/NO
Do you suffer from any conditions (e.g. related to mobility) which could cause you to be unduly injured by bumping into objects, or people, or by falling to the floor?	YES/NO
Do you suffer from Claustrophobia?	YES/NO
Do you suffer from any other condition which you think might affect your ability to use the VR?	YES/NO

Please do ask if you would like to discuss anything relating to these questions.

Participant's Name

Participant's Signature

Researcher's Signature

Date

8.9 Complete results from gesture study

		Gesture Recognition System	Static Database		Adaptive Database	
			Mean	Standard Deviation	Mean	Standard Deviation
Game Experience Questionnaire - Core (Scale 0-4)	Competence	Explicit	2.19	0.78	2.08	0.86
		Continuous	2.19	0.71	1.84	1.09
	Immersion	Explicit	2.7	0.72	2.44	0.77
		Continuous	2.59	0.75	2.35	0.78
	Flow	Explicit	2.91	0.66	2.96	0.8
		Continuous	2.84	0.8	2.68	1.07
	Tension / Annoyance	Explicit	0.48	0.52	0.17	0.3
		Continuous	0.75	0.72	0.98	1.15
	Challenge	Explicit	1.55	0.58	1.52	0.61
		Continuous	1.52	0.46	1.6	0.59
Negative Effect	Explicit	0.19	0.21	0.14	0.27	
	Continuous	0.44	0.5	0.41	0.55	
Positive Effect	Explicit	3.32	0.35	3.29	0.59	
	Continuous	3.02	0.57	2.86	0.93	
Game Experience Questionnaire - Social Presence (Scale 0-4)	Psychological Involvement Empathy	Explicit	1.58	0.71	1.33	0.82
		Continuous	1.34	0.94	1.23	0.97
	Psychological Involvement Negative Feelings	Explicit	0.94	0.87	1.04	0.68
		Continuous	0.91	0.99	0.92	0.63
	Behavioural Involvement	Explicit	2.53	0.75	2.42	0.75
		Continuous	2.28	1.02	2.28	0.95
Game Experience Questionnaire - Post Game (Scale 0-4)	Positive Experience	Explicit	2.43	0.65	2.28	0.83
		Continuous	2.3	0.81	2.05	1
	Negative Experience	Explicit	0.33	0.42	0.49	0.49
		Continuous	0.23	0.32	0.37	0.34
	Tiredness	Explicit	0.38	0.58	0.33	0.57
		Continuous	0.43	0.52	0.55	0.89
Returning to Reality	Explicit	1	0.52	0.93	0.68	
	Continuous	0.87	0.51	0.85	0.6	
System Usability Scale (Scale 0-100)	SUS Score	Explicit	70.88	10.98	68.38	12.2
		Continuous	61	18.73	53.25	17.79
Gesture Recognition Accuracies (Percentage)	Gesture Accuracy	Explicit	74.40.%	18.60.%	88.80%	8.50%
		Continuous	73.10.%	18.20%	84.10%	10.90%
	Gestures Missed	Explicit	25.80%	17.60.%	15.40%	16.00%
		Continuous	57.30%	27.80%	57.00%	21.60%