Imperial College London

MSc Individual Project

Imperial College London

Department of Computing

Virtual Reality Visualization of Data using HTC Vive and Manus VR gloves to define queries with gestures

Author:
Benjamin Mai

Supervisor:
Dr. Thomas Heinis

September 6, 2018

Abstract

Mind Explorer Enhanced VR is a virtual reality application built for the HTC Vive, based on the virtual reality application Mind Explorer VR created by Oliver Robinson in 2017. Mind Explorer Enhanced VR uses a new technology of gloves for virtual reality called the Manus VR gloves to perform different queries with the aim of improving the virtual reality visualization experience of the neocortex of a rat.

The application enables users to execute queries using the gloves on a high level three-dimensional model of the network directly or on cubes to interact with this network indirectly. Those queries can be executed within distance from where the user stands or by grabbing objects directly with the hands. Either ways, those functionalities allow users to move objects in a virtual dark room, scale them, rotate them and trigger messages in the neocortex, all of this by executing specific gestures with the hands.

Acknowledgements

I would first like to thank my thesis advisor Dr. Thomas Heinis of the departement of Computing Science at Imperial College of London. The door to Prof. Thomas office was always open whenever I ran into a trouble spot or had a question about my research or writing. He consistently allowed this paper to be my own work, but steered me in the right direction whenever he thought I needed it.

I would also like to thank all my professors at Imperial for this very challenging and enriching year during which I discovered the beauty and complexity of Computer Science.

Finally I would like to thanks my parents and my brother who have always been extremely supportive in my studies, my flat mates Simon and Nathan without who my years spent in London would not have been the same and my girlfriend Ilana who is always a source of motivation and inspiration.

Contents

Chapter 1

Introduction

1.1 Motivation

Virtual reality is a computer generated graphical environment that offers opportunities for users to view and interact with the virtual environment in stereoscope (i.e., three-dimensional visuals) [1]. As VR is becoming more affordable to public, new discoveries and applications have emerged from this area. Although VR was initially mainly applied to entertainment and gaming ends, this technology is now also applied to advance fields of medicine, engineering, education, design and training.

In particular, Virtual reality can be used to better apprehend all sort of complex physical data to improve its visualization. For example, Mind Explorer VR, developed by Oliver Robinson in 2017, enables users to visualize an example of a neural network forming the neocortex of a rat as a three-dimensional model. However, very little has yet been done in using virtual reality gloves to provide more friendly users environment to improve the visualization of complex data.

Indeed, so far, most of current virtual reality applications, including Mind Explorer VR, have been making use of remote controllers. For example, those devices have been extensively used to simulate human hands or weapons in all sorts of virtual environments. However, those controllers present a number of limits. Due to virtual reality still being a new field, users may feel uncomfortable using them, as they are lacking traditional buttons and controls. Some may find that it alters the virtual reality experience as they do not allow the reproducing of natural movements from hands that could be performed in the real world. Besides, one of the main issues that comes with VR environments can be a sensation of sickness felt by users, just like if the body assumes it's been poisoned [2]. Although this feeling of sickness mainly comes from rotation of the head in the virtual reality [3], the non natural experience of using the controllers further worsen this problem.

Therefore to address these issues related to remote controllers, a better and more promising alternative involves the use of virtual reality gloves. They give users the possibility to reproduce movements with their hands in a virtual reality environment in the same way as they would do in their daily lives. Furthermore, the glove's low latency improves and contributes to that feeling of being inside a virtual reality world more fully and therefore decreases images' disturbances that are known to cause nausea. To this regard, this project will be concerned with the aim of improving users' experience when visualizing the neural network of a rat's neocortex in a virtual reality environment by using the Manus VR gloves.

1.2 Initial Main objectives

The main goal of this project is to incorporate the Manus VR gloves into the Mind Explorer VR application to improve the virtual reality's user experience even more. The result of this work is a new application called Mind Explorer Enhanced VR. This new application keeps the same functionalities as Mind Explorer VR but makes no use of the Vive controllers anymore. In addition, it adds the possibility to touch and grab objects with the hands in the Virtual Reality. The front-end environment of the Mind Explorer VR application has been kept and in particular the properties of the neural network including the query, message and connectivity modes (more of this will be explained in subsection 2.3.1). Therefore, almost the entirety of this project work was concerned with the back-end's changes of controls that the application allows from Vive controllers to Manus VR gloves. The main objectives defined at the beginning of this project are mentioned below.

1. **Object interactions within distance:** From the beginning, it was planned that Mind Explorer Enhanced VR should give the possibility for users to use the gloves to manipulate objects within distance, just like if the user had telekinesis power. Those objects that could be manipulated include the neocortex of the rat directly or cubes that enable to interact with the neocortex indirectly. For both of these objects, the initial objectives of Mind Explorer Enhanced VR was to use the Manus VR gloves to perform queries on them within distance so that users could:

    - Move the chosen object with the right or left hand by making a fist with either the right of left hand. Then while holding the fist, move the object in a direction by moving the hand in that desired direction.

    - Scale the object by a transform proportional to the distance between the right and left hands. This query is performed by making a fist with both hands and then change the dimensions of an object by increasing or decreasing the distance between the two hands while keeping the hands making a fist.

    - Launch a query, message or connectivity signal in the cortex by clapping hands. For example, once a square is located in a desired area of the cortex,

execute the mode chosen by executing the hands clapping. Such action is currently undertaken in Mind Explorer VR by pressing the backward button of the controllers.

- Rotate the neocortex of the rat by executing a gesture that yet had to be chosen at the early stage of this project.
- Move around the room by executing a specific gesture with the hands that yet had to be defined. The current methods for displacement that is being used in Mind Explorer VR is a teleportation system, which allows users to move around the room by displaying an arc that roots from a controller to somewhere else in the room. Such a query is performed in Mind Explorer VR by pressing the central button of the controllers to display the arc and by loosen the button to enable the displacement. However, since the Vive controllers will be replaced by the gloves, a solution that will involve use of the gloves needed to be found to enable users smooth displacement around the room.

2. **Direct object interaction:** Beyond controlling objects within distance, it was expected that Mind Explorer Enhanced VR should give the possibility for users to use the gloves to grab and throw objects in the 3D room in the same way they would naturally do in real life. As part of the initial objectives, it was expected that users would be able to:
   - Grab objects in the scene (the different types of cubes or the neocortex) the way they would do it in reality. The grabbed object should rotate with the movement of the hands once grabbed.
   - Create an object velocity when the object is touched or thrown away. The object should follow a behavior similar to an object being thrown in reality.
   - Activate the Menu as it is currently displayed in Mind Explorer VR, by touching the Menu button located on the right or left wrist with respectively the left or right index. Then being able to select a mode, again by touching the one desired with the right or left index.
   - Pushing the reset button with the right or left index to reset the scene.

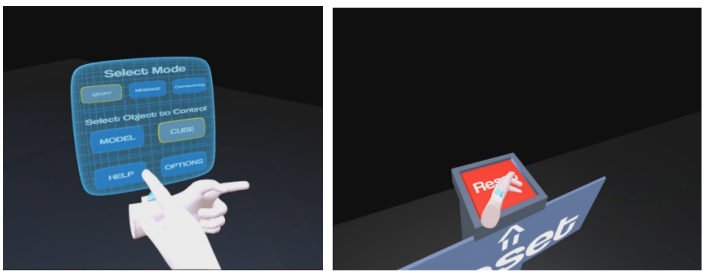


Figure 1.1: Left: The main menu display, allowing users to change what mode they are in or what they are controlling. Right: A user hitting the button using their virtual hand in VR in order to reset the scene

3. **Haptic Feedback:** Each glove contains a fully programmable vibration motor for haptic feedback such that when a user performs an action, this should give him the possibility to feel a vibration at the back of the hand. Therefore, every time an object is grabbed with the hands, it was expected as one of the initial objectives that a haptic feedback should be generated.

Chapter 2

Background

This chapter will go through main aspects that involved research at the early stage of this project including: A presentation of the hardware and software resources used, already existing research involving the use of virtual reality gloves for gestures' recognition. Finally, this chapter will present the preliminary work undertaken early on necessary to pursue this project.

2.1 Hardware Resources

This section will introduce the pieces of hardware that were used as part of this project.

2.1.1 HTC Vive Kit

The virtual reality package that was used is the HTC Vive kit. It includes the pieces of hardware necessary for most of current virtual reality applications: A headset, two Vive controllers and 2 base stations. The headset allows for the immersion into the virtual reality environment. The Vive controllers enable users to execute actions and the base stations, which need to face each other within a certain distance, enable to recognize users' movements. As of now, HTC Vive's main competitor in the virtual reality industry is the Oculus Rift headset. Other headsets exist, especially for mobile phone applications, however they lack the positional tracking components of the last two introduced and do not present necessary computational power required for an application like Mind Explorer Enhanced VR.

Both the HTC Vive and Oculus Rift use room scale tracking systems, enabling users to move in a 3D space and use motion-tracked controllers to interact with the environment. From a practical point of view the HTC Vive's setup is slightly more tedious than the Oculus Rift one. However, the Vive headset allows for a larger tracked position area in the virtual environment. This aspect is highly desirable in the context of

Figure 2.1: The htc Vive headset along with its controllers and base stations

Mind Explorer Enhanced VR to enable users to walk around the room over a larger area. This results in better visualization and easier interaction with the neocortex. For example when a user grabs a cube with his hands, he can then walk towards the neocortex to locate it in a desired location, to then trigger one of the possible signals in the neocortex (more of this is discussed in chapter 3).

2.1.2 Manus VR Gloves

The Manus VR Gloves represent the primary aspect and challenges of this project. They are developed by a start up called Manus VR that was set up in 2014. This technology is therefore still relatively new in terms of research and marketability. The gloves can be used for this project thanks to their compatibility with Unity. They also have native support for Unreal Engine as well as Autodesk Motion Builder.

The Manus VR gloves work by calculating the relative position of the hand. They rely on a technology that uses the tracking solution of other systems, such as Xsens and Vive tracking to get the positional tracking of the hands in the virtual reality (see section 2.1.3 on Vive trackers). However, any tracking solution that gives such location can be used to provide positional tracking. The Manus VR gloves are currently compatible with technologies such as OptiTrack, Vicon, Xsens, PhaseSpace and Vive Tracking. The orientation is measured through an accelerometer, gyroscope and a magnetometer and each finger is tracked by two sensors designed by Manus VR. Through these sensors the company collects reliable, easy-to-use data on what the hands are doing at all time [4].

The gloves are also comfortable to wear so that users forget that they are using gloves. They are made of a stretchy material that wraps around the hands and fingers, but leaves the tips of the fingers free. Another very interesting feature is that each glove contains a fully programmable vibration motor for haptic feedback [5]. For example, It can be activated when the user touches an object to replicate the sensation of touching it.

Figure 2.2: The data glove is machine washable and includes two sensors per finger, a vibration motor for haptic feedback, and a rechargeable battery

Using Manus VR gloves present many advantages compared to using the Vive controllers. For example, users usually feel unfamiliar with Vive controllers when first using them due to this technology being still relatively new to the market and to its non traditional buttons. Instead, the Manus VR gloves are much more intuitive as they are used using real life's movements from the hands and fingers. Also, when using Vive controllers, users tend to still be aware that they are holding controllers, which may have a negative effect on the virtual reality experience. On the other hand, the gloves give an even greater sense of reality in the virtual reality environment. Those gloves are in line with the ultimate virtual reality aim of replicating movements and sensations of the real world in virtual reality.

Due to their still currently high price, applications for public customers are relatively limited. However, the company has created partnerships with companies to introduce the use of the gloves in different sectors. For example, the company is working with the NASA to provide virtual reality experiments and simulations that help to astronauts learn how to use tools and how to maneuver through the International Space Station in a zero-gravity environment. This helps to train and prepare astronauts and those helping them in the control room. Besides, it enables astronauts to have a better understanding of how to use on-board equipment, what to do when tools float, or what to do if there's disorientation or space sickness [6].

Figure 2.3: Manus VR has created a partnership with the NASA to help astronauts train in a virtual environment

2.1.3 Vive Trackers

A Vive tracker is a motion tracking accessory. It is designed to be attached to physical accessories, so that it can be tracked via the Lighthouse system. Vive Trackers feature a connector that can be used to communicate with the accessory it is attached to. Currently, Vive Trackers are coupled with accessories in virtual reality games, such as a Hyper Blaster (a light gun-style controller), and a racket used for sports games. When using the Manus VR gloves, the two vive trackers each need to be fixed on a band that can be attached to a user's wrists. Such position of the vive trackers enables to track the position of the forearms and the hands in the virtual reality. Other applications may also make use of the bands to attach trackers to a user's legs for example.

Figure 2.4: A pair of HTC Vive trackers

2.2 Software Resources

2.2.1 Unity

Unity is a cross-platform game engine developed by Unity Technologies, which is primarily used to develop both three-dimensional and two-dimensional video games and simulations for computers, consoles, and mobile devices [7]. Unity is an ideal tool to develop virtual reality applications because of its compatibility with most virtual reality's headsets among which the HTC Vive and the Oculus Rift. Unity VR enables to target virtual reality devices directly from Unity, without any external plug-ins in projects. It also provides a base API and feature set with compatibility for multiple devices and has been designed to provide forward compatibility for future devices and software [8].

Games in Unity are created by manipulating objects in 3D and attaching various components to them. Even 2D games must be manipulated in 2D/3D. Scripts can be written in C#, Boo or JavaScript and attached to 3D objects as components. These scripts have the useful property of containing public fields that can be changed either from scripts directly or from the inspector. For example, if a script contains a

public float value, such variable can be changed from script or from from the inspector directly. These values can even be changed in the inspector while the game is playing. This becomes extremely useful when it comes to test different values for a variable until finding a suitable one. For example, it can be helpful to test different floating values to find the right speed of a given object in a video game. Functions in a script can also be public. This allows to call the function of a script attached to an object from another script to have control on every possible object in the scene from all scripts.

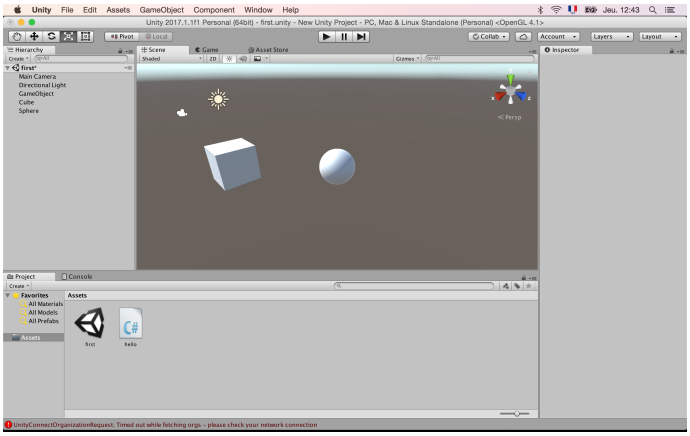


Figure 2.5: An example scene with 3D objects and a C# script attached to one of them

It was decided that Unity would be chosen for this project as Mind Explorer VR was created using this tool to display the rat's neocortex in Virtual Reality. In addition, Unity provides good support for all virtual reality headsets available and is compatible with the Manus VR gloves.

2.2.2 Steam VR

SteamVR is a Software Development Kit that provides key virtual reality functionalities. It is provided as a free package on the Unity asset store and includes a single interface that will work with all major virtual reality headsets from seated to room scale experiences [9]. The Steam VR package contains a prefab of the camera rig that just needs to be dropped into the scene. It will automatically align the camera to the user's head movement when he has the headset on, and allow him to move controllers in the virtual reality environment. Users can then easily customize these functionalities by adding or removing scripts to give the desired behavior to their virtual reality application. Since Mind Explorer Enhanced VR does not make use of the Vive controllers, only the camera's alignment to the headset was used.

2.3 Existing Work

This section will review some of the existing research and works related to this project and to the use of virtual reality gloves and their applications.

2.3.1 Mind Explorer VR

The most important piece of work and the one that served as a foundation for my project is Mind Explorer VR, realized by Oliver Robinson as part of his MSc project and under the supervision of Dr Thomas Heinis. Mind Explorer VR is a virtual reality application that allows for the visualization of a neural network dataset from a rat's neocortex. This application uses the HTC Vive technology and in particular the Vive controllers to interact with the neural network using different functionalities:

1. **Model Interaction:** The user can easily manipulate objects in the virtual three-dimensional room. This enables to perform queries on objects present in the scene using the Vive controllers. The application provides the options to either manipulate the cortex directly or to manipulate a 3D cube that enables user to interact with the cortex indirectly through the query, message or connectivity modes.
2. **Query Mode:** This enables the user to display a smaller specific area of the cortex rather that the model in its entirety. Such system is especially useful to better observe details in zone of the cortex that are highly dense in neurons. The area is chosen using a 3D cube that can be easily manipulated in the virtual environment using the Vive controllers. Then, the user presses the back button of one of the controllers to output an extremely detailed subsection of the neocortex model that was located inside the cube. This subsection of the model can then be freely manipulated in the scene (see figure 2.6).

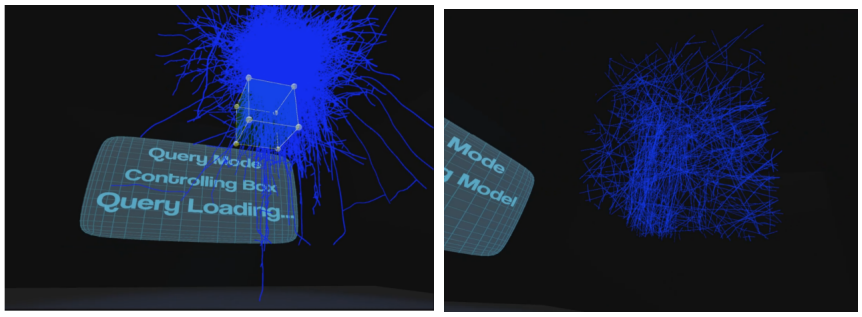


Figure 2.6: Left: A user performing a spatial query in VR. The cube with yellow outlines is the 'query cube' to be positioned by the user. Right: The result from the query being performed in the figure on the left displayed in VR

3. **Message mode:** The message mode can be used to visually send a stimulus in the model. Similarly to the Query Mode, the user can place a cube in a desired area of the neocortex. Then, by pressing the back button of one of the

controllers, a message that departs from the cube propagates through neurons of the cortex. Such message is visualized by thousands of short red segments. As the message propagates towards the center, it starts branching rapidly and splits towards various branches of the cortex. Then it dies out when it reaches the end of edges (see figure 2.7).

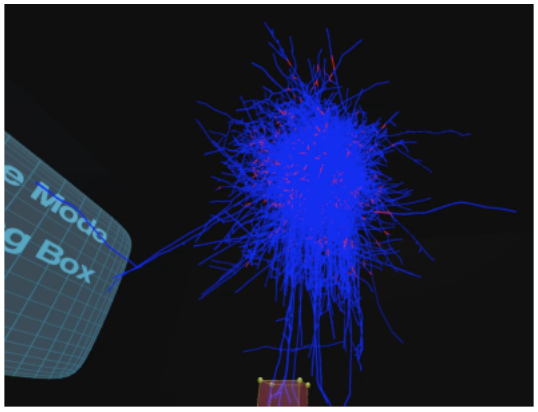


Figure 2.7: An example of a message signal propagating through the neocortex

4. **Connectivity mode:** This allows the user to highlight all neurons included in the cube within a range than can be chosen in the menu interface, displaying some of the inner connectivity of the network (see figure 2.8).

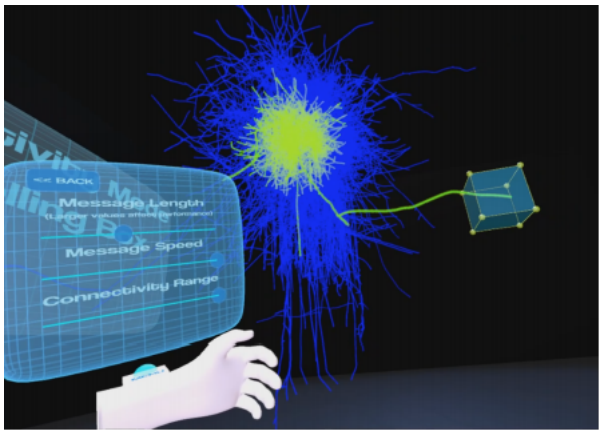


Figure 2.8: An example of a connectivity message departing from a 'connectivity cube' and highlighting branches of the cortex

2.3.2 Gesture Recognition using Naïve Bayes Classifier

Gestures' recognition was a key component of my project. Different methods have been designed to give the best gestures' recognition results. Researchers from the National Tsing Hua University have developed a wireless virtual reality glove with a gestures' recognition algorithm based on machine learning. In particular, they classified gestures using Naïve Bayes Classifier. To reach this goal, they specified two layers.

One for the state mode the user desires to be in (each state mode is associated with a specific gesture), followed by another layer for the motion mode, representing a continuous movement (similarly, each motion mode is associated with a specific gesture) [10]. Since there are three state modes, three different classifiers were trained. They were trained with feature data of positions and rotations of component of the hands, such as the finger bending angle and the palm's orientation from the glove. Such data were collected from around 10,000 hand movements. Figure 2.9 shows the range of possible gestures recognized resulting from the training of the classifiers.

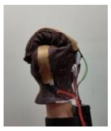
Hand Gesture				
Feature	Right click in mouse state.	Entering mouse state / FPS state from idle state.	Left click in mouse state.	Screen scrolling up or down in mouse state.
Hand Gesture				
Feature	APP shifting.	Back to idle state from both mouse and FPS state.	Fire / Walk forward in FPS state.	Reset reference orientation in all state.

Figure 2.9: Gestures trained with Naïve Bayes Classifiers

Results were assessed by computing the average inside-test recognition rate (RAA). The more features that were used the higher the RAA got (see figure 2.10). The final methods yield an overall RAA over the three State modes of 85.2%, suggesting promising results.

# of features	Idle	Mouse Control	FPS
1	0.7474	0.6335	0.5266
2	0.7805	0.8062	0.7018
3	0.7746	0.8522	0.8168
4	0.7905	0.8706	0.8860
5	0.8023	0.8816	0.9242
6	**0.8057**	**0.8886**	**0.9402**

Figure 2.10: The htc Vive headset along with its controllers and base stations

However, such a machine learning approach was not undertaken in the context of Mind

Explorer Enhanced VR because of the difficulties that computing a sufficient database of hands' movements would have involved. Besides, the application is already very computationally demanding, affecting the graphical performance. Having added a machine learning implementation in the back-end would have most likely increased the latency even more.

2.3.3 Gesture Recognition using 3D positions from components of the hands

Some researchers have implemented a system of gestures' recognition including three hand gestures. An index pointing gesture to select virtual objects but also to translate and rotate them based on the position of the left or right index fingertip and the hand orientation. A hand-open gesture for the release of a selected object and the gesture of two moving hands with the ring and small fingers closed to scale an object. To enable the object selection and rotation, the index pointing gesture had to be recognized. Such gesture required a movement from the index, meaning that the right and left index fingertips had to be determined and tracked. With the distal phalanx length known through the measurement of the user's index finger, the index fingertip position could be determined based on the distal interphalangeal joint position and the distal phalanx bending angle with respect to the middle phalanx provided by the data glove [11] (see figure 2.11).

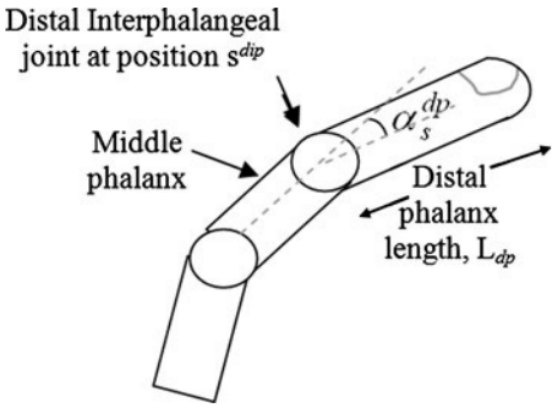


Figure 2.11: Variables that enable to determine the index tip position

This research gave me the idea to apply a similar methodology, but instead of measuring the positions of the fingertips manually, use unity's functionalities on objects' transform to obtain the position of each fingertips relative to the hand when performing a gesture. This approach is discussed later in subsection 3.2.2.

2.3.4 Soft Robotic Glove

The currently type of feedback used by the Manus VR gloves is defined as a tactile stimuli, meaning that a vibration is felt when an object is touched. There is another type of haptic feeback called a kinesthetic stimuli, which refers to the sense of force and position of our fingers and hands [12]. Such stimulus is currently not implemented in the Manus VR Gloves.

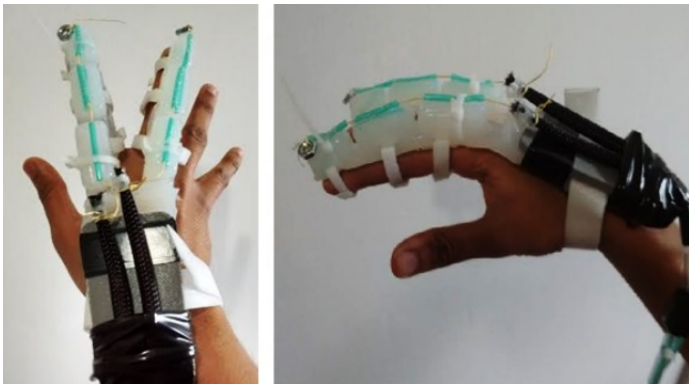


Figure 2.12: Final assembly for the soft robotic glove

However, some researchers from the University of California San Diego developed a robotic glove that generates kinesthetic haptic feedback. They made use of a soft exoskeleton with soft actuation for the haptic glove which makes the glove compliant and compact. The gloves were then tested in a virtual environment enabling users to press piano tiles. The average force applied on a button was computed to measure the force that the actuator should apply to mimic the pressing of the tiles in the virtual reality environment. To enable actuators to applicate a quick and high pressure on the hands when touching a tile, the actuators were built using long latex balloon. Results obtained were extremely promising with all testers arguing that the nature of the kinesthetic stimulus was improving the virtual reality experience. However, this glove is still at a very early prototype level and cannot be marketable yet, nor could have been used for a project like Mind Explorer Enhanced VR.

2.4 Preparatory Work

I began this project without any experiences in virtual reality nor in using Vive. Besides, I had never used neither Unity neither coded in C# before. Therefore, at the early stage of this project, I spent a great amount of time familiarizing myself with those tools. To first get my hands on unity and C#, I undertook a few short projects involving making 3D video games. In particular I made a rail shooter game from scratch, whose design and functionalities are shown in figure 2.13 below.

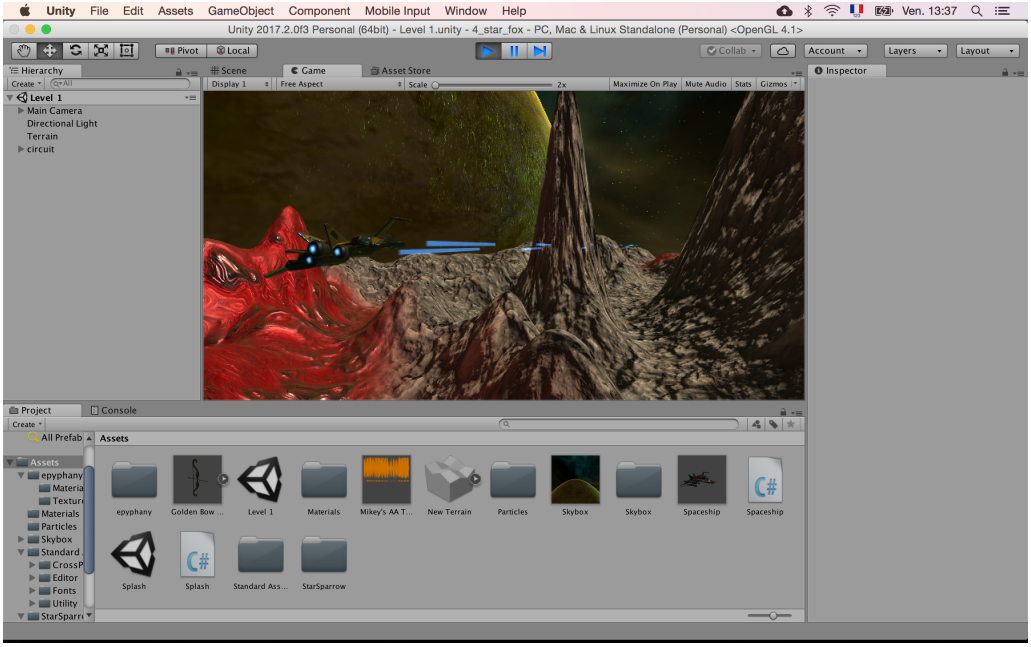


Figure 2.13: Making of a game rail shooter to learn functionalities of unity and C#

The realization of this small game helped me understand the fundamentals of Unity necessary to undertake my project. Some of the most important elements that this game enabled me to learn before hand among others are:

1. **Scenes:** Every project in Unity must have at least one scene. A scene is the place in which a game is created. For example many games have different levels. For each different level will be associated a different scene that will have its own specificities. For example the Rail Shooter game that I made has three scenes: One for the welcome and option Menu and two others that specify two levels of difficulties. In Mind Explorer Enhanced VR, only one scene was created that immerses the user into a large dark room with a giant neocortex of a rat in the middle of it.

2. **Design and architecture of a game:** The design of a game is realized from the unity inspector. On this regards, it offers countless number of possibilities in the creation of objects, their shapes, materials associated with them, soundtrack, particles system and many others. Although the architecture and design aspect in the creation of the rail shooter was very interesting and enjoyable, I did not spend too much time on it as just a basic understanding of it was sufficient for this project.

3. **Prefabs:** Different objects in unity can be combined to make a new one. Such new object will present unique specificities than can be kept by saving the object as a prefab. Making an object a prefab becomes extremely useful when many entities of the same object must be created. Whenever any of the instances of a prefab is changed in the scene, all other instances of the same prefab present in the scene are changed accordingly. In Rail Shooter, some ships' enemies were

included in the scenes from one prefab created. This means that operating a change of design on the enemy's prefab for example would apply the changes to all enemies' objects included in the scene from this prefab. This aspect was useful in Mind Explorer Enhanced VR with the gloves, represented by two hands in the virtual reality, being saved as a prefab. This prefab can be included in any projects that would make use of the Manus VR Gloves.

4. **Use of the Unity interface:** The Unity interface is where all controls can be accessed when a game is created. Its main components are the project, hierarchy, inspector, scene, game and console windows. The project window (see figure 2.14) enable to sort and organize all components created in a game including scenes, all prefabs, materials, particles etc... The hierarchy window (see figure 2.15) shows all objects present in the scene and enables to create objects directly from it. The inspector shows (see figure 2.15) all components attached to an object (such as scripts, colliders, materials, transform, rigidbody etc...). The scene window is the actual area in which the game is built. The game window shows the player's view when the game is played. Finally the console window is the equivalent of the terminal where all coding executions are shown (for example a print method will be shown in the console).

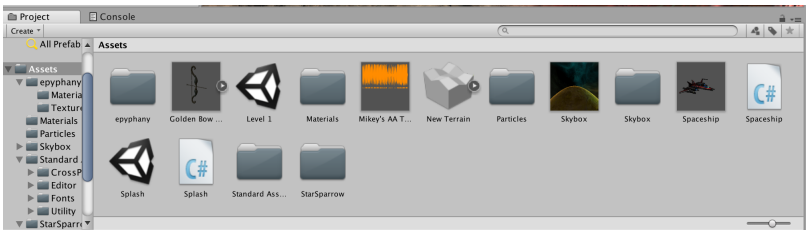


Figure 2.14: Display of the project window organizing files, prefabs, audio track and other components

5. **Object Transform:** Each object in unity is associated with a transform that specifies its position, rotation and scale. The transform of an object can be accessed from script or from the unity inspector directly. As this project is focused on being able to perform queries with gestures, understanding the object transform aspect of Unity was fundamental. Regarding the Rail shooter game, movements and rotations of the ship were allowed using the left, right, up and bottom arrows on the keyboard. Unity works by displaying a large number of frames per second just like in a movie. Therefore, when one of the key arrows is pressed, a displacement and rotation offsets to the ship current transform position are executed in the direction of the arrow key pressed. With a new current position and rotation at each time frame, and with many time frames per second, this technique gives the impression of the space ship moving and rotating in the direction of the key arrow pressed.

6. **Physics of objects:** Physics of objects in Unity is managed using a rigidbody [13]. As for the transform, the rigidbody of an object can be changed from scripts or directly from the unity inspector. For example, this component determines the

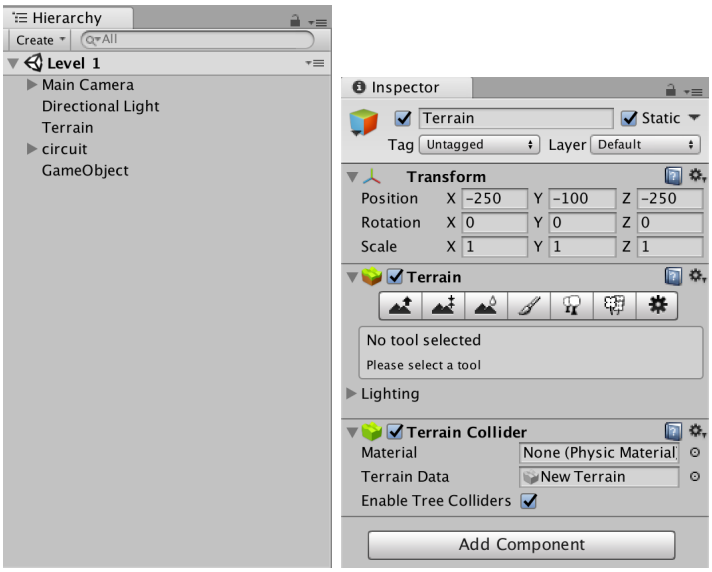


Figure 2.15: Left: Hierarchy window that shows all objects present in the scene, here a camera, a light, a terrain, circuit and a ship. Right: display of the inspector, here showing the transform of the circuit in rail shooter, its terrain and terrain collider

velocity, gravity or the force that should be applied on an object. Understanding the functioning of a rigidbody was another critical point in this project. Indeed, in Mind Explorer Enhanced VR, all object interactions that involve grabbing or touching an object with the hands make use of the rigidbody. Besides, in order to improve the visualization experience, all objects in the scene are statically gravitating in the air. This aspect is also regulated by using a rigidbody attached to each object and by removing gravity.

7. **Object collider:** A collider component defines the shape of an object for the purposes of physical collisions [14]. It enables to signal when an object enters in contact with another one and how both objects should behave when they touch each others. For example, whether they should be treated as transparent components or whether physics should be applied. Collider is at the heart of this project as it is used for hand's gestures recognition (see subsection 3.2.2) and the direct grabbing of objects with hands.

There are obviously many more aspects of Unity, however those mentioned above were the main ones used as part of this project.

Chapter 3

Development

This chapter will go into the details of the making of Mind Explorer Enhanced VR, divided into different sections that each explain features of the application and details of their development.

3.1 Integration of the Manus VR Gloves

As the aim of this project was to use the Manus VR Gloves to improve the use of Mind Explorer VR's functionalities, a consistent first step was to integrate the Manus VR gloves into this application. The purpose of this first step was simply to make the gloves work in Mind Explorer VR by being able to move fingers from both hands. As no code had yet been written, no queries on objects could yet be performed. The intent at this stage was not to be able to interact with those objects but rather a simple integration.

Such integration of the gloves involved removing all appearances of codes related to the Vive Controllers in Mind Explorer VR as well as the right and left hand prefabs. As a result of these removals, a few changes had to be done in the code to keep the program compiling as all functions involving use of the Vive controllers were not relevant anymore. This stage also involved understanding the code architecture built in Mind Explorer VR as most of the scripts in it had some references to the Vive controllers. Therefore, it was necessary to be careful with what had to be removed to keep a level of consistency in the code. Manus VR company created a package that easily enables to integrate the gloves into any project as shown in figure 3.1.

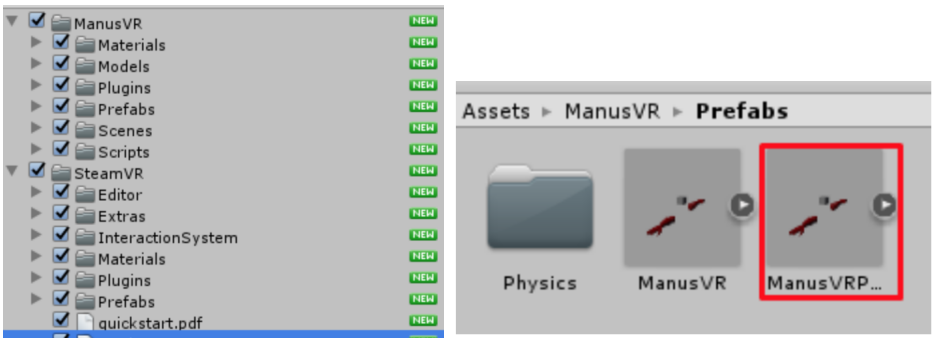


Figure 3.1: Left: Manus VR package that includes the Manus VR gloves setting and the Steam VR library. Right: Manus VR gloves prefab that was dropped into the scene

The package provides a prefab of the gloves that just had to be placed into the scene. This first step was therefore smooth and simple.

3.2 Distance Mode

This section will go through details of the implementation of the Distance Mode. It will first explain why this mode was particularly suitable in the context of Mind Explorer Enhanced VR, then it will present its possibilities and implementations.

3.2.1 Motivations behind the Distance Mode

The main objective of this project is to interact with an example of complex data, here the neocortex of a rat. This task can be done by performing queries on the model directly to visualize it better or on cubes to trigger different sort of signals in the model.

To this purpose, the cortex of the rat and the cubes appear like fixed objects gravitating in a dark room (see figure 3.2). This creates a pleasant virtual reality visualization experience and provides great conditions to observe the cortex nicely and easily. Besides, because of the neocortex's shape made of thousands of neurons, the model cannot be statically placed on the ground, thereby also justifying why objects need to gravitate in the scene. However such position of objects, gravitating in the room, poses some issues concerning their manipulations and controls. Indeed, because of this, they are hardly within a direct grasp of the hands. It becomes apparent why a Distance Mode that enables users to interact with these objects within distance becomes necessary.

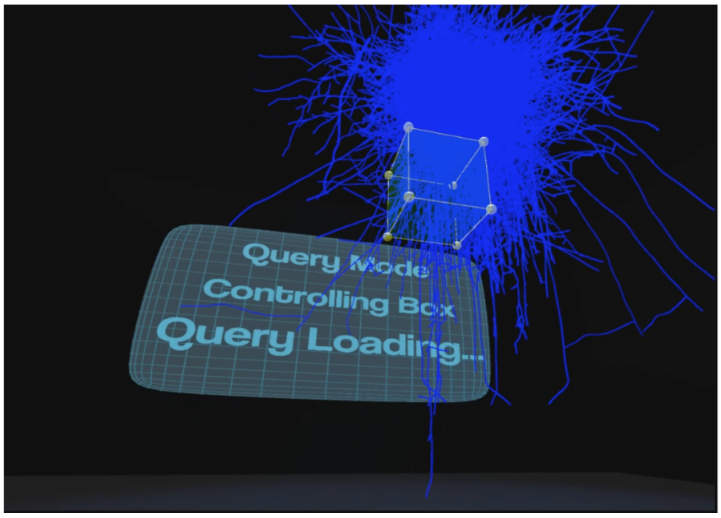


Figure 3.2: Neocortex of the rat and cubes are statically gravitating in the scene.

In addition, for the visual experience to be the best, the virtual reality room in which all the queries can be performed as well as the cortex of the rat need to be big quite in size. Yet the area of displacement in virtual reality is still limited by the size of the room in which the user is using the HTC vive. In the unfortunate case that an object is located outside of the VR walking distance, it again becomes very useful to be able to perform queries on objects within distance.

All in all, a Distance Mode would allow users to perform queries on objects present in the scene in a precise and accurate manner, directly from where they stand. For all the reasons just mentioned, its implementation became a top priority.

3.2.2 Distance Move Mode

The Distance Move Mode aims at moving any objects present in the scene within distance by performing a specific gesture with the hands. Those objects that can be moved include the neocortex of the rat, the query cube, the message cube and the connectivity cube. The initial idea that I had in mind was to enable the moving of any of those objects by forming a fist with the right or left hand. The intuitive aspect of this gesture, its easiness to perform as well as positive feedbacks received from test users comforted me in this decision. The Distance Move Mode can be used as follow: Making a fist with either hands gives the feeling of grabbing the object at distance. Then, still while maintaining the hand forming a fist, the object can be moved in the scene in the same direction as the user's hand's movement direction. Then, opening the hand again leaves the object at its exact position at the time of the fist release. This operation enables a smooth and precise way to move any objects in the scene.

Figure 3.3: Left: Hand is open and no queries are being performed apart from moving fingers. Right: One hand makes a fist triggering the Distance Move Mode.

The move mode was implemented in two steps: First by creating a fist signal so that the program recognizes when the fist gesture is executed by one of the two hands. Second by writing the code that enables the desired displacement of the object to be moved in the scene.

1. To build the fist signal, the objective was to find a way for the program to detect when a hand makes a fist. This goal was at the heart of this project which is focused on gestures' recognition. The recognition of the fist gesture involved different possible approaches. These approaches as well as their limitations are described below. Details of the method that was ultimately chosen are also explained.

   My first idea was to divide some components of the hands into single different game objects. For example, the tip of each finger, each knuckle of each hand, the palm of each hand, the wrists and so on. In Unity, each of these objects have a default vector position relative to the hand (such position is defined as the local position) that corresponds to the three-dimensional vector (0,0,0). As those objects are moving in the virtual environment, their position vectors relative to the hand are changing. This means that when a user is making a fist with a hand, each of these objects is associated with a new specific position vector resulting from the action of making a fist. Therefore, I collected a sample of position vectors of these game objects from the hand, with each sample corresponding to the execution of a different fist. For each game object, and for each sample, the new vector position was associated with a new x, y and z position. Then, I computed an x, y and z confidence intervals for each game object to obtain different ranges. These ranges were then manually entered into the code to specify the positions that each game object should belong to in order for a fist gesture to be recognized. For example, for the object "left hand's tip of the index", I first computed a sample of its vectors' position relative to the left hand when making different fists. Then, x, y and z positional ranges were computed with a

95% confidence interval using the x, y and z variables collected from the sample of vectors' position. A 95% confidence interval was used such that the range of recognition is large enough to get a high recognition rate result. This process is then repeated for other game objects on the hand to get the final algorithm.

This method gave promising fist's recognition rate results. It was noticed that as more game objects from each hand were chosen, the chances of inadvertently triggering the Distance Move Mode were reduced. In other words, the chances of getting a false-positive (the fist is not made, yet the gesture is recognized) went down, which is here desirable. However, using more components was also associated with less chances of recognizing a fist gesture when it is actually performed by the user. This means, the chances of getting a false-negative (the fist is made, yet the gesture is not recognized) also increased, which is here non desirable. It was decided that I ultimately would not adopt this method for the final output. Indeed, It involved a lot of hard-codings and data collection by manually collecting the position vectors of each object defined when a hand was making a fist. Besides, it gave decent fist's recognition results, yet this could be improved more.

$$I_c = \left[\bar{x} - t_\alpha \frac{s}{\sqrt{n}} \; ; \; \bar{x} + t_\alpha \frac{s}{\sqrt{n}} \right]$$

Figure 3.4: Formula of a confidence interval when the population standard deviation is not known.

Therefore, a second method that had better recognition rate results was adopted. Such approach makes use of a system of colliders' interaction. In particular, I implemented a sphere collider on the tip of each index and a box collider at the bottom of the palm of each hand (see figure 3.5 and 3.6). The positions of those colliders are such that when a user makes a fist, the two colliders touch each others. Then, using the Unity function *OnTriggerEnter()* which is called when two colliders enter in contact with each others, I was able to trigger the following action written here in pseudo code: "If the sphere collider on the right index collides with the box collider on the right hand, activate Move Mode for right hand and if sphere collider on the left index collides with box collider on the left hand, activate Move Mode for left hand ". This way, the only possibility for the right and left indexes to collide with respectively the right and left box colliders and therefore to trigger the Move Mode is to form a fist with the right or left hand. In addition, this code prevents from the non-desirable action of the sphere collider of one index to collide with the box collider placed on the other hand. To stop the signal of the fist, the function *OnTriggerExit()* is used. This function recognizes when a collider exits the contact zone of another collider. Therefore, opening the hands again enables to stop the fist gesture's recognition.

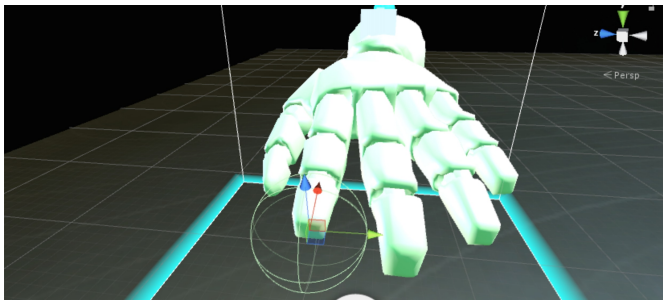


Figure 3.5: A sphere collider at the tip of the index.

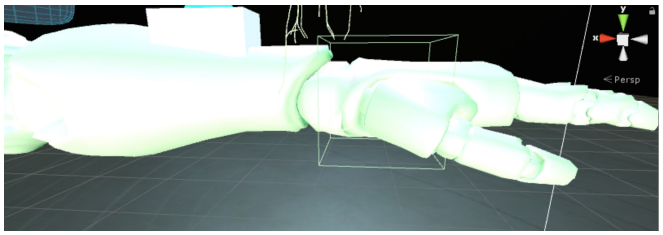


Figure 3.6: A box collider at the bottom of the hand. When the sphere collider of the index enters in contact with the box collider, by the user forming a fist with the hand, this triggers the Distance Move Mode.

Similarly to the first method, in which I used different game objects from the hand, I here tried this approach using different colliders on each hand. For example, placing sphere colliders on the tip of each finger. As expected, It was observed that just like in the first approach, the more colliders' interaction were used between different objects of the hand, the lower were the chances of getting false-positives but the greater the chances were of having false-negatives. In Mind Explorer Enhanced VR, false negatives are more damageable that false positives. For this reason, it was decided that only the sphere colliders at the tip of each index and the box colliders in the palm of each hand would be used for purpose of fist's recognition. With this final approach, recognition rate was extremely high. For this reason, this method was used in the final version of Mind Explorer Enhanced VR.

2. Once a fist gesture is recognized, a signal is sent to a script called "Model Manipulator" attached to each object that can be moved in the scene. This signal triggers the code in Model Manipulator to allow the moving of the desired object. Then the moving of the object in itself is achieved by first saving as a variable the position vector of the hand that makes a fist (initialHandPos) and the position vector of the object to be moved (initialObjectPosition) when the fist gesture is executed. Then, at every time frame, the difference between the new position vector of the hand (newHandPos) and the initial position vector of the hand is computed. This difference is then multiplied by a multiplier proportional to the distance between the hand and the object (distanceMultiplier) to obtain an

offset (see equation 3.1). Finally this offset is added to the initial position vector of the object to obtain the new position of the object (newObjectPos) at a given time frame (see equation 3.2). Thanks to this method, the object to be displaced moves in the same direction and at the same pace as the hand that makes a fist. Besides, thanks to the distance multiplier, when the object to be moved is close to the hand, a small movement of the hand results in a small displacement of the object. When instead the object is located further away from the hand, the same small movement of the hand will result in a larger displacement of the object. The pseudo code that corresponds to this query is given by:

$$\textbf{offset} = \textbf{(newHandPos - initialHandPos)*distanceMultiplier} \tag{3.1}$$

$$\textbf{newObjectPos} = \textbf{initialObjectPosition} + \textbf{offset} \tag{3.2}$$

3.2.3 Distance Scale Mode

The Distance Scale mode enables users to change dimensions of objects in the scene. As for the Distance Move Mode, those objects still include the rat cortex and the different interacting cubes. This new query is in line with the objective of using virtual reality for visualization purpose. For example, being able to change the dimensions of the cortex of the rat provides users with different perspectives to observe the model with the best experience possible. Increasing the dimensions of the cubes enables to increase the number of neurons that will be included in those cubes and therefore the number of neurons affected by a chosen signal (query, message or connectivity).

Similarly to the Distance Move Mode, it was important to find an intuitive way for users to be able to change dimensions of objects with a gesture from the hands. The most natural solution that I thought of was to activate the Distance Scale Mode by forming a fist with both hands. This method allows users to switch very easily from the Distance Move Mode to the Distance Scale Mode. Then while keeping both hands making a fist, users can change the dimension of the chosen objects proportionally to the distance between the hands (see figure 3.7). When the Scale Mode is activated, all other functionalities from the Distance Mode are being deactivated. In other words, it becomes impossible to move an object around the room while changing its scale. The fist recognition in the Distance Scale Mode makes use of the same method described in the previous subsection. However, it can only be activated when both hands have been detected as making a fist at the same time.

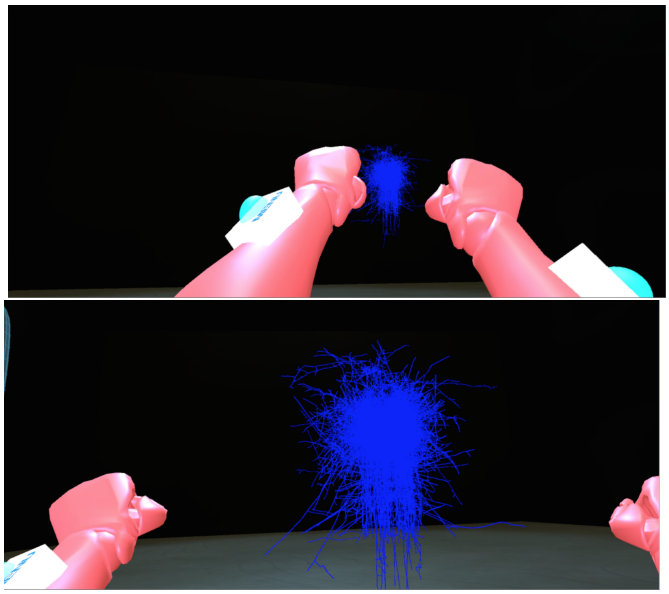


Figure 3.7: Up: Initial scale of the cortex. Down: New scale of the cortex after stretching the arms while making a fist with both hands.

The actual scaling of the object is coded by multiplying by a scale factor, at every time frame, the initial dimension of the cube (startingObjectScale) when the Distance Scale Mode is triggered. At each time frame, this scale factor is equal to the ratio of the distance between the hands (currentHandDist) at the corresponding time frame divided by the initial distance between the hands (startingHandDist) when the Distance Scale Mode was triggered, multiplied by a scalar multiplier (3.3). Then, the new dimensions of the object is obtained by multiplying the starting scale of the object by ths scale factor just computed. To impose a limit on the maximum scale that an object can take, I used the *min()* function such that the new scale of the object is the minimum between the dimensions chosen by the user and a maximum object scale variable that I chose to be suitable (3.4). The pseudo code for this implementation is given below:

$$\textbf{scaleFactor} = \textbf{(currentHandDist/startingHandDist)*scale} \tag{3.3}$$

$$\textbf{newObjectScale} = \textbf{min(startingObjectScale * scaleFactor, maxObjectScale)} \tag{3.4}$$

3.2.4 Distance Query Triggering Mode

To keep the same functionalities as Mind Explorer VR, one of the priorities was to find a solution to trigger the query, message and connectivity modes. In Mind Explorer VR, this is achieved by pressing the button at the back of one of the vive controllers once a cube is positioned over an area of the neocortex of the rat. Then depending on the mode, a query, message or connectivity signal is sent through the network,

departing from the area of the cortex contained within the cube. It was decided that the effect of pressing the back button with the Vive controllers would be replaced by a clapping of the hand with the Manus VR gloves as shown in figure 3.8.

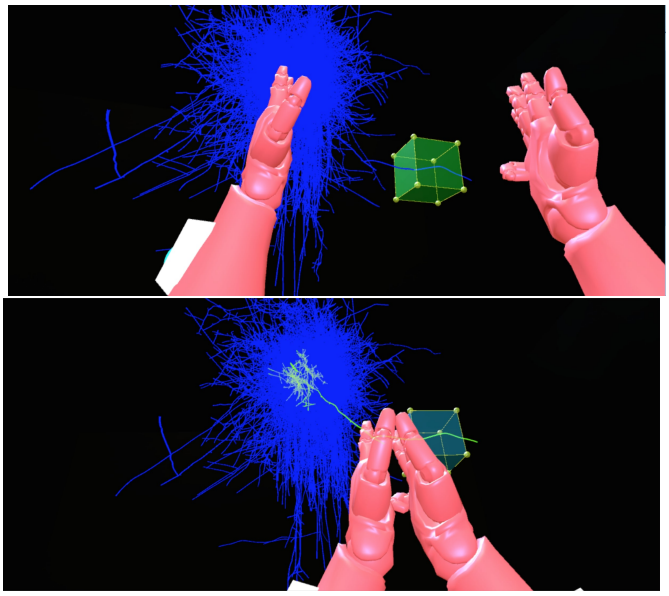


Figure 3.8: Up: Connectivity cube is in position and hands ready to clap. Down: After clapping of the hand, a connectivity message that departs from the cube is sent into the neocortex.

In order to recognize the clapping of the hands, I used a similar system of object collision as the fist gesture's recognition described in subsection 3.2.2. In pseudo code the query coded translates to the following action: "if right hand collides with left hand, trigger signal for the current mode chosen".

3.2.5 Distance Rotation Mode

The rotation mode allows users to rotate the neocortex model of the rat. As part of a project whose aim is ultimately to improve the visualization of complex data, being able to rotate the neocortex to observe it from different angles was important. One of the issues encountered was that I first fell short of ideas for potential gestures to perform the rotation. With the making of fists already used for moving and scaling actions, I had troubles finding a fluid move that would enable users to perform accurate rotation. This problem emphasizes one of the issues with using virtual reality gloves, that movements that can be executed with hands and fingers to perform queries are somewhat limited compared to controllers which have many buttons.

It appeared to me that the most suitable way to rotate the neocortex within distance would be to use all possible rotations from the wrist. This way, users can rotate the

neocortex along the y-axis by rotating the forearm, thereby rotating the wrist. They can rotate the model along the z-axis by rotating the wrist up and down. Finally, rotations along the x-axis are performed by rotating the wrist right and left.

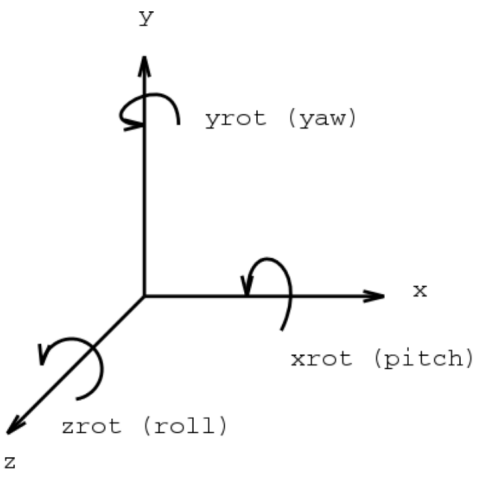


Figure 3.9: Possible rotations along the x,y and z axis.

In coding, this type of rotation is obtained by computing the initial rotation of the hand chosen to make the rotation (handInitRot) and the initial rotation of the cortex (cortexInitRot). Then, at each time frame, the new rotation of the hand is collected (handCurRot) to compute the rotation difference between the initial rotation in the hand with the new rotation of the hand (see equation 3.5). This rotation difference (rotationDiff) is then multiplied by the initial rotation of the cortex to finally obtain the new rotation of the cortex at each time frame (see equation 3.6). For example, if the rotation difference gets bigger, this means that the rotation performed by the hand is getting bigger, reflecting in a more significant rotation of the model in the direction of the rotation. The pseudo code for this implementation can be found bellow:

$$\textbf{rotationDiff} = \textbf{quaternion.inverse(handInitRot*handCurRot)} \tag{3.5}$$

$$\textbf{cortexNewRot} = \textbf{cortexInitRot*rotationDiff} \tag{3.6}$$

However, it yet remained to find a solution for users to easily trigger the Rotation Mode. Since rotation is enabled using rotations of the wrist, It made sense to trigger the Distance Rotation Mode by making a fist and then executing the desired rotation by rotating the wrist in the way described above. However, since the making of a fist was already used for the Distance Move Mode, I added the constraint that the bottom part of the forearm jointed to the hand making the rotation should be touched with the other hand (therefore the one not making the rotation). Therefore, when user is in Distance Move Mode, Rotation Mode can easily be started by performing the gesture just described above.

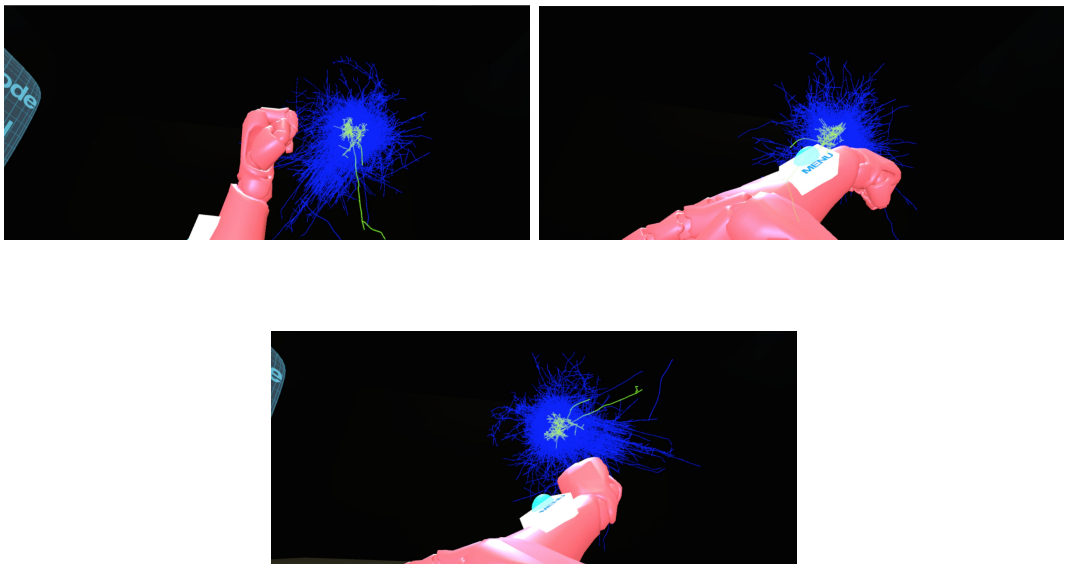


Figure 3.10: Rotations of the wrist along the x, y and z axis to rotate the neocortex accordingly.

I again used a transparent box collider that I placed at the bottom of each forearm such that for example when the right hand enters in contact with the collider located in the left forearm, a boolean expression is set to true. When this boolean is triggered and the left hand makes a fist, the Distance Rotation Mode is started.

3.3 Object Grabbing Mode

Controlling an object within distance using different gestures with the hands enables users to perform queries they cannot do in reality. Although the Distance Mode is very useful in Mind Explorer Enhanced VR to perform smooth and accurate queries on objects within distance, one of the main purposes of using the Manus VR Gloves is for users to be able to grab objects directly. Therefore, It naturally came to me to implement an Object Grabbing Mode that would allow users to control objects by directly being able to catch them, similarly to the way they would do in real life.

The code that allows the grabbing and touching of an object with the gloves, developed by Manus VR company, was already provided to me in a script. This script has to be attached to the object to be grabbed together with a Rigidbody and a collider. The Rigidbody in Unity enables to perform all queries that relate to the physics of the object. For example, whether gravity should be applied to the object or not, whether the object should collide with other objects or be treated as a transparent entity etc... Because the script that enables the grabbing of an object was already done, my work focused on the integration of the possibilities it gives in the context of Mind Explorer Enhanced VR.

3.3.1 Activation of the Grabbing Mode

Once the script that enables the grabbing of an object was attached to one of the query, message or connectivity cubes, the first attempt to grab one of those directly with the hands did not work as expected. As I was trying to grab the cube, like it would be done in real life by closing my hand on it, the Distance Move Mode, which I recall is triggered by making a fist, was also being activated. This undesired event resulted in the cube starting vibrating and moving extremely quickly around the hands, making it impossible to grab it and creating some bugs in the application. To solve this issue, the code was changed such that whenever a hand touches a cube (to grab it or throw it away for example), all functionalities from the Distance Mode would be deactivated. This way, it was expected that the making of a fist with the hand when grabbing an object would activate the Grabbing Mode rather than the Distance Move Mode. However, this attempt again did not work as planned. The grabbing of an object with the gloves works with a collider system. When the collider that covers the hand enters in contact with an object, the program recognizes than an object is touched or needs to be grabbed in the virtual reality. But whenever a hand was closing on the cube to grab it, although the Distance Move Mode was deactivated as desired, as soon as the hand was touching the cube, as small impulse was sent on to the cube. This had for effect to immediately push the cube out of the colliding zone between the hand and the cube, therefore reactivating the Distance Move Mode. With this process happening at many frames per second, this resulted in some bugs and confusion in the program on whether the cube should be grabbed on controlled within distance.

To fix this problem, I decided to clearly separate the timing of Grabbing Mode and Distance Move Mode. To reach this goal, I changed the code such that the Distance Move Mode would not be deactivated when the hands touch the cube, but instead when the cube is located within a short distance of the hands. Let's call this distance between the cube and the hands the *Grabbing Object Zone*. This distance is large enough to clearly separate the Distance Mode from the Grabbing Mode such that all bugs are removed but short enough to enable users to easily grab the object within reach of their arms. Therefore, when an object enters this zone, the Distance Move Mode is being deactivated so that the object gets fixed right in front of the hands and ready to be grabbed. Thanks to this solution, the Grabbing Mode and Distance Move Mode are not mixed up anymore enabling to switch smoothly from one mode to another and therefore grab an object more conveniently. In figure 3.11, the hands have got close enough to the cube such that the cube is located in the Object Grabbing Zone. As a result, any functionalities from the Distance Mode are being deactivated as soon as the hands remain in that zone, enabling to easily grab the cube.

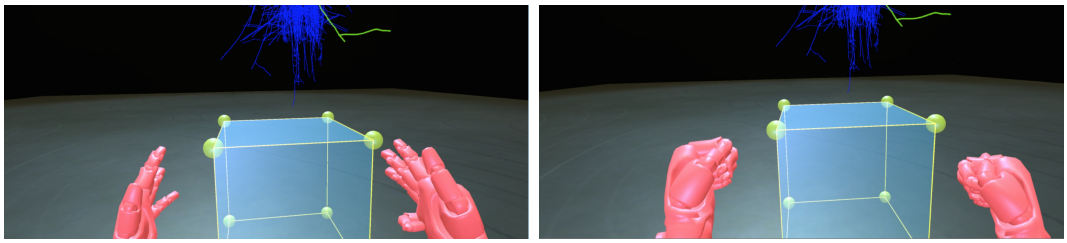


Figure 3.11: Cube is located in the Grabbing Object Zone, such that all gestures from the Distance Mode (for example Distance Move Mode and Distance Scale Mode) are being deactivated.

3.3.2 Catching of the objects

Thanks to the implementation of the Grabbing Object Zone, the object to be caught becomes static when the hands get close to it. This process enables to catch the object much more easily. Once grabbed, the object can be rotated and moved with the hand within the reach of the forearm (see figure 3.12). This gives the feeling of manipulating an object just like it would be done in real life. Besides, it is possible to walk around the room while holding the object in the hands. Therefore, one application of the object grabbing mode is to enable users to grab a cube, walk towards the cortex and place it where they want query, connectivity or message signals to depart from. Users finally just have to clap their hands to trigger the desired signal.

To improve even more the sensation of catching an object, I made use of the haptic feedback functionality of the Manus VR gloves. The Manus VR package provides access to a function that enables to trigger a vibration in the gloves. I added this function such that it is called in the right and left hands when their respective attached colliders enter in contact with any of the cubes in the scene. The function takes in arguments a variable of type "Hand" that refers to the hand concerned by the vibration (right or left) and a variable of type float associated with the strength of the haptic feedback. I chose an intensity of vibration that is strong enough to replicate the feeling of touching an object but weak enough such that it does not get annoying to feel this vibration many times.

It is also possible for users to simply push an object with any parts of the hands. For example, the upper part of the hand, the palm, the tip of a finger. The object will then behave with a velocity and rotation proportional to the strength of the pushing just like it would happen when an object is pushed with the hands in real life. In the context of Mind Explorer Enhanced VR, with objects gravitating in a room, the trajectory of an object resulting from its pushing seems to behave like it would do in space.

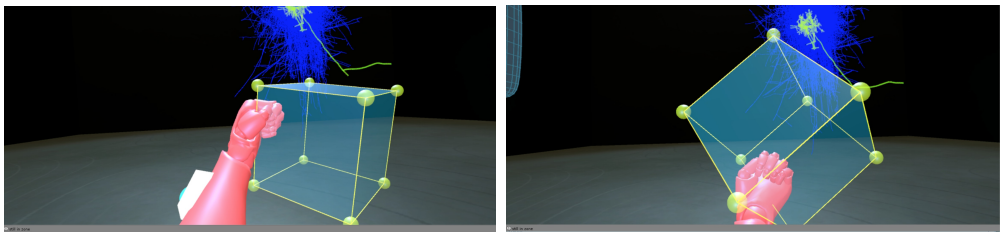


Figure 3.12: Left: Connectivity cube is grabbed with one hand. Right: Connectivity Cube is rotated using rotation from the wrist.

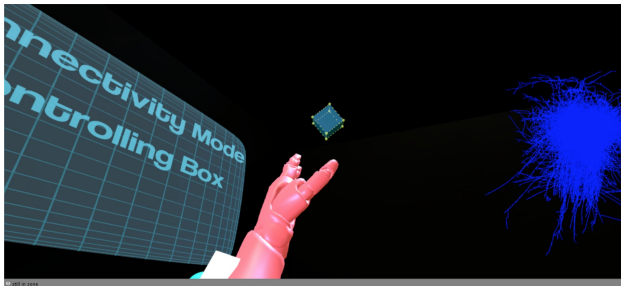


Figure 3.13: Cube is thrown away and moves with a velocity and strength proportional to the throwing.

One issue that I faced was to stop the velocity of the object once it was thrown away. For example, when I was touching a cube to throw it away, I could easily come back to the Distance Move Mode by just making a fist to control the object again within distance. However, once I was opening my hand again to stop the Distance Move Mode with the intent of fixing the cube at a given position in the scene, the cube was starting to move and rotate again with the same velocity and strength as the ones given by the initial throwing of the object. Such behavior can be be explained as followed: When a cube is thrown away, a function that manages the velocity of the cube is called and this function runs while the velocity of the object is positive. Then, when a fist is made to activate the Distance Move Mode's query, a function that manages this query is also being called. However, although the cube behaves according to the Distance Move Mode's function being called, the velocity function still runs in the back-end. Therefore, as soon as the Distance Move Mode's function stops being called, which should normally result in the object getting fixed, the cube instead retakes its initial velocity given by the velocity's function still running. This means that I had to find a solution to stop the velocity of the cube when no queries were being performed (in other words when the hands stay open).

To solve this issue, I created a function *FixedObjectPosition()* that applies to the object a constant position at every time frame. This function is only called when no queries are being executed on the object. Therefore, even if the velocity function still runs in the back-end, the cube gets fixed in the scene by the FixedObjectPosition() function. The implementation of such function was a followed: First, the position

of the object being controlled was saved as a variable when a hand was opening to stop the Distance Move Mode. Let's here call this variable *FixedObjectPosition*. In code, this means that such position was saved whenever the collider located at the tip of the index was stopping colliding with the collider at the bottom of the hand, in other words when the hand was opening (see subsection 3.2.2 for recalls on this). This implementation is undertaken with the function OnTriggerExit() that is called when two colliders stop colliding with each others. Then, the FixedObjectPosition is applied to the transform position of the object at every time frame when no queries are being executed. All in all, thanks to this implementation, when an object is being thrown away with the hands in the Grabbing Mode, if a user then switches to the Distance Move mode to control the object within distance (by making a fist) and then opens his fist again to stop the Distance Move Mode, the object's position when the hand opens remains fixed until the next query is executed.

3.3.3 Grabbing Scaling Mode

To enhance the range of possible functionalities when a cube is grabbed, I developed the Grabbing Scaling Mode that enables users to change the scale of a cube when this last one is grabbed with both hands (see figure 3.14). This mode can only be activated under the constraint that both hands directly touch the cube to be scaled and both hands make a fist. This gives the perception of a flexible cube whose dimensions can be increased or decreased by grabbing its edges.

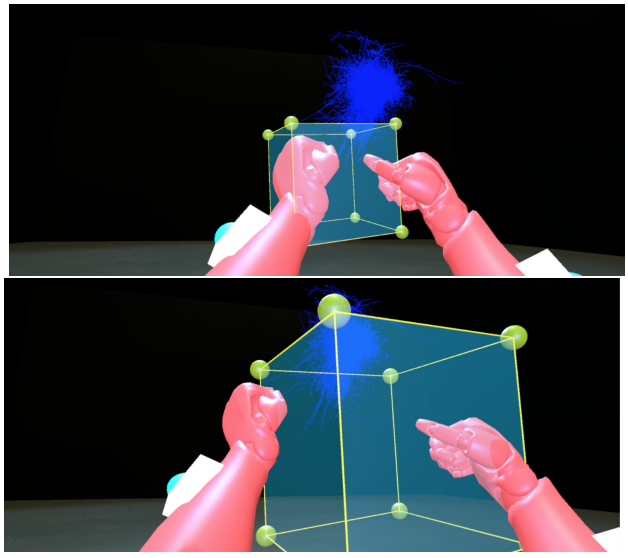


Figure 3.14: Left: Small cube is grabbed. Right: Cube gets bigger as user stretches its hands.

Such implementation was fairly easy as It mostly used already built functionalities. The recognition of both hands making a fist was done by re-using two booleans already

used as part of the Distance Move Mode. Then, two booleans were created to signal when the right and left hands touch the cube. Once all these booleans are activated, the cube can be scaled using the same method as the one described in the Distance Scaling Mode (see subsection 3.2.3) but only this time by grabbing the cube with both hands.

3.3.4 Limitation the Object Grabbing Mode

Being able to grab an object directly with the hands, rotate it, touch it or throw it away along with haptic feedback improve the visualization's virtual reality experience. However, It presented a few limitations in the context of Mind Explorer Enhanced VR. These limitations are specific to this application and may not apply to other ones making use of the gloves.

1. One of the disappointments is to not having managed to make it possible to grab the neocortex directly. The main issue with the neocortex is the complexity of its shapes made of thousands of neurons. Therefore using simple shapes like a box collider or sphere collider to grab it was obviously not suitable for such complex model. For objects with complex shapes, Unity has designed a mesh collider component which reads the properties of the transform of the object to set its position and scale correctly [15]. It is therefore much more accurate for collision detection. However, It seems that the shape of the neocortex was overly complex as this method to be able to grab the cortex by its neurons' branches did not work . They might be a way to solve this issue but I could not find the resources in Unity.

   Another different attempt to manipulate the cortex was to place a few objects with the simple shape of a button around the neocortex and child the cortex to those objects. This means that the buttons and the cortex became one single object. Therefore, moving or scaling any of the buttons objects was leading to the moving and scaling of the neocortex in the same manner. Because of the simple and spherical aspect of those buttons, the inclusion of a simple sphere collider was sufficient to fit their shapes. All in all, this method was allowing users to move the neocortex by grabbing the buttons around it. This strategy aimed at replicating the sensation of grabbing and manipulating the cortex directly. However, the result led to poor visual aspect, with the buttons hiding some parts of the neocortex. Besides, the expectation of replicating the feeling of grabbing the neocortex directly was not met. For all these reasons, this strategy was removed from the final output.

   However, I believe that having the possibility to grab the cortex directly with hands would not have brought much for the visualization purpose of this project. Due to the extreme density of the neurons towards the center, the only viable solution to manipulate the model with the hands would have been to grab it

by branches that emerge towards the external part of the model (where the density of neurons is much lower). Because each neuron's ramification is so thin, it would have felt odd to move the entire highly dense model by just grabbing one of them. Besides, grabbing the model directly with the hands would have involved being extremely close to it (within a forearm reach). At this distance, because of the large dimensions of the neocortex, visualization would have been extremely reduced. Instead, the Distance Move Mode appears mores suitable in this case to observe the cortex from a greater distance but in its entirety.

2. As mentioned earlier, in the context of Mind Explorer Enhanced VR, the cubes and the neocortex are statically gravitating in the scene for some necessary reasons explained before. This aspect yet poses some issues concerning the direct grabbing of the cubes with the hands. Because of this absence of gravity in the cubes, objects behave accordingly. As a result of this, attempts to grab the cube with the hands are sometimes interpreted by the program as a desire to simply push it. This results in the object being sometimes thrown away when users simply want to catch it. For example, if the object would be placed on a table, its grabbing would become much easier because of the physical resistance exerted by the table when a hand closes on it to grab it. To facilitate the grabbing of an object in Mind Explorer Enhanced VR, users can use both of their hands to create a physical resistance that reduces the effect of the absence of gravity on the object.

Chapter 4

Evaluation and Performance

This chapter will provide both qualitative and personal evaluations of Mind Explorer Enhanced VR. The goal of the application is ultimately focused on user experience. Therefore there was not much room for quantitative results.

4.1 Qualitative Evaluation

Qualitative results are extremely important when developing an application like Mind Enhanced Explorer VR. This is especially the case for an application that is concerned with queries' executions. Throughout the development of the project, feedbacks were constantly asked mainly to relatives. Those feedbacks were taken into account to find suitable gestures to perform queries or consider which features of the application could be useful to implement.

However towards the end, more proper qualitative results were collected. This involved finding users to experience Mind Explorer Enhanced VR as well as the previous application that makes use of the Vive controllers. Then, testers were invited to fill in a short questionnaire of 11 questions (see figure 4.1). The aim of this questionnaire was mainly to assess the ease by which testers can perform queries available in Mind Explorer Enhanced VR, compare their virtual reality experience in the new application compared to the previous one and estimate users' perceptions of the usefulness of each query available. Whenever a question asked users to give a grade, they were always required to mark between 0 and 10 with 0 being the worst grade and 10 the best. Full results can be found in appendix B.

To make sure that results would be the most meaningful possible, the application was tested by people with and without experiences in virtual reality. Most testers had scientific background but other sectors like economics and business were also represented by a few testers. Age range varied from 17 to 59 with an average age of 33 and a median age of 27. In total, 11 people tested the application and completed the

The distance Move Mode refers to all queries that enable to execute actions on objects within distance.	
How easy was it to move objects within distance? (rate from 0 to 10)	
How easy was it to scale objects within distance? (rate from 0 to 10)	
How easy was it to rotate objects within distance? (rate from 0 to 10)	
How easy was it to perform queries in general in the application? (rate from 0 to 10)	
How useful did you think the Distance Mode was in general? (rate from 0 to 10)	
The Grabbing Mode refers to all queries that enable to execute actions by directly grabbing objects.	
How easy was it to grab an object with the hands?	
How useful did you think the Grabbing Mode was in the context of this application?	
You tried 2 applications, a first one that uses gloves and a second one that uses controllers.	
10) How would you compare your VR experience in the first application compared to the second one (answer of those: much better, better, same, worst, much worst)	
11) Did you think the first application was more intuitive to use? (answer yes or no)	

Figure 4.1: The feedback questionnaire given to users after having tested the new and previous application

questionnaire. This number is quite low as testers could only try the application from my place due to the heavy setup involved.

Testers were first asked to assess the ease by which they could use each of the queries from the Distance Mode. The Distance Move Mode, Distance Scale Mode and Trigger Mode were all considered very easy to use with a respective average of 8.00, 7.45 and 8.55. This suggests that the fluidity of these queries was well appreciated and the gestures were intuitive and easy enough to perform. The ease of performing the Distance Rotation Mode was graded slightly lower with an average of 6.73. Most testers mentioned that the rotation of the neocortex allowed by rotations from the wrist was quite nice but having to put the hand that is not performing the rotation on the forearm of the hand doing the rotation was quite unintuitive. Some testers also added that holding this position could get quite uncomfortable. Overall, testers seemed to find the Distance Move Mode rather easy to use with an average of 7.18. More importantly, they judged the Distance Move Mode extremely useful in the context of Mind Explorer Enhanced VR by granting an average grade of 8.18.

Concerning the Object Grabbing Mode, testers were very enthusiastic about the possibility to grab an object directly with their hands in the virtual reality environment. All of them, even the ones with experiences in virtual reality, had never experienced using VR gloves. The average grade given to whether they found the possibility to grab objects pleasing was of 8.64 supporting the statement that this functionality was enjoyed by all. However, as expected, users sometimes had issues grabbing an object. They were naturally trying to grab it with one hand sometimes without success (object was sometimes being thrown away). This issue was confirmed in the average grade of 6.18 in the ease of use of the Grabbing Mode's question. Besides, although most testers liked being able to grab objects with hands in the virtual environment, they did not grant it much usefulness in the context of the application by giving an average grade of 5.64. When asked about it, they considered that it was easier to interact with the cortex by using cubes in the Distance Mode rather than the Object Grabbing Mode.

Finally, users were invited to answer questions that compare the new application Mind Explorer Enhanced VR with the previous one that uses the Vive controllers. 8 people found that the gloves' controls in the new application resulted in a better VR experience, 1 person answered "much better" and 2 people thought that the VR experience in both applications was similar. This suggests that the integration of the gloves truly improve the virtual reality immersion. Besides, all the testers answered yes to the question asking whether they found Mind Explorer Enhanced VR more intuitive to use than Mind Explorer VR. This result confirms the ease that people have to use gestures that they are familiar with and are used to execute in real life rather than using the still unfamiliar virtual reality controllers.

4.2 Personal Evaluation

Looking at the overall development of the application and qualitative results, It seems that most of the initial objectives have been met. The Manus VR gloves have been integrated into the previous application Mind Explorer VR to produce a new application that enhances the virtual reality experience even further. Different queries can be performed using the gloves from within a distance of the objects or by grabbing them directly.

One thing that definitely needs to be improved on is the ease by which objects can be grabbed in Mind Explorer Enhanced VR. As of now, it can sometimes be hard to grab an object with a hand because the program sometimes interprets the intent of grabbing this object as a willingness to simply push it. This results in the object being thrown away when user simply wants to catch it.

One other source of disappointment is that, I believe, the initial application was not the most suitable to make full benefits of the potential of the Manus VR gloves. In-

deed, in Mind Explorer Enhanced VR, although it is very enjoyable to be able to grab an object directly with the hands, the most convenient way to perform queries on objects is yet to use the Distance Mode. For example, the application enables users to catch a cube with the hands, walk around the room with that object, place it in a desired area of the cortex and then triggers a query, message or connectivity signals in the cortex with that cube. However, the reach of the cube's position is limited to the length of the forearm. Since dimensions of the cortex are much greater than this reach, this makes it impossible to place the cube in higher area of the cortex. It is still possible to reduce the size of the cortex to much smaller dimensions but this considerably reduces the visualization possibilities. With this regards, it appears more convenient to use the Distance Move Mode to move the cube in any areas of the cortex or the room.

Besides due to the shape of the neocortex, made of thousands of very thin neurons, I was unsuccessful in developing a viable solution to grab the model with the hands (details of this were developed in chapter 3). However, due to specificities of the application, grabbing the neocortex directly with the hands by one of its branches would not have brought much regarding the virtual aim of visualization.

Working with such a new technology was extremely enriching but not without challenges. Due to the gloves still being at an early stage and barely on the market, almost no information could be found on the Internet about how to apply this technology in Unity. This made the solving of even the most trivial issue very puzzling. All in all, this project enabled me to develop some skills in unity, c#, vive and more broadly in the area of virtual reality with a steep learning curve. I would now be extremely interested to work in the gaming industry in the future and get my hands on Unity again.

Chapter 5

Conclusion and Further Work

5.1 Conclusion

The final output of Mind Explorer Enhanced VR is a new virtual reality application that provides different queries that enable to interact with the neocortex of a rat to facilitate its visualization.

While Mind Explorer Enhanced VR still requires more work to be considered marketable, it gives a first good foundation into the use of Virtual reality gloves for visualization purposes. Its final features include:

1. **Fully integrated gloves:** Fingers, wrists and forearms can be moved and rotated as one would do in real life.
2. **Distance Move Mode:** It allows users to interact with objects in the scene within distance. Possible queries include moving, scaling, rotation of objects and triggering of query, message or connectivity signals into the neocortex.
3. **Object Grabbing Mode:** It allows users to directly grab cubes in the scene with the hands, throw them away, place them to a specific location (in particular the neocortex to then trigger a signal) and scale them. It also enables to have access to a menu on the wrists and its functionalities by using indexes from hands.

5.2 Further Work

Mind Enhanced Explorer VR provides a first step in the direction of the use of VR gloves for virtual reality visualization's purposes. However, there are rooms for improvements in many aspects. Below are some points that could be improved on and implemented for future projects.

- **VR Arc Teleporter:** The previous application was making use of the VR teleporter package to allow user to easily move around the room without having to walk. It was done by pressing the track pad at the front of the Vive controller to create an arc whose endpoint was pointing to the location to move to. The track pad was then released to execute the moving action. This option was not included in Mind Explorer Enhanced VR due to lack of time. The VR Arc Teleporter package is coded specifically to be used with the Vive controllers, however it could be integrated with the gloves so that the arc roots from the index rather than from the controllers for example. Then, a specific movement could be chosen to execute the displacement in the direction of the arc. From a practical point of view, finding a suitable gesture with the gloves that would allow such query might be an issue.
- **Improved grabbing precision:** As mentioned earlier, the program is sometimes creating confusion on whether an object is grabbed or pushed. This can make it harder for users to grab an object, which negatively affects the virtual reality experience. Therefore precision to better distinguish between user's willingness to grab an object with hands or simply touch it or push it could be improved further.
- **Improved haptic feeback:** As of now, the haptic feedback simply involves a slight vibration at the top of the hand when an object is grabbed for example. This improves the feeling of really touching an object but does not quiet yet realistically replicate a real contact. therefore Another interesting aspect would be to replicate the touch of object by sending haptic feedbacks in specific areas of the hand. For example, if only the finger touches an object, only apply the haptic feedback to that area. Such implementation would improve the degree of immersion into the virtual environment even more but would involve working on the hardware of the gloves as well. Alternatively, other virtual reality gloves that provide this improved haptic feedback could be used for future projects.
- **Other Complex Data:** Further work could involved using the gloves and Vive headset to enable the visualization of other complex data, especially for medical applications. Such data could include other highly complex parts of the body such as a heart.

Chapter 6

Ethical Requirements

As part of the new regulations in place, this section will list some of the ethical concerns related to my project.

1. **Desensitization:** It is well known that being fully and regularly immersed in a virtual reality environment can result in some users to become desensitized in the real world. For example, virtual reality is already used as a tool to emotionally desensitize individuals against phobias [16]. Nevertheless, outside of this supervised use, such desensitization could become harmful, with for example users becoming less affected by acts of violence found in some virtual reality games. A research on 36,965 participants showed that violent video games increase the probability of an individual committing an aggression or demonstrating aggression-related variables [17]. Due to higher levels of immersion, It is reasonable to assume that these effects could be even more significant with virtual reality.

2. **Virtual Criminality:** Philosopher Thomas Metzinger is one of the first to have expressed concerns that Virtual Reality applications could be used in the future by the military as a presupposed "ethical" alternative to known interrogatory torture. Such possibility emphasizes the necessity to understand the psychological and physical status of pain, damage, violence, and trauma inflicted to users in a virtual environment.

3. **Privacy:** Privacy is and will remain a major concern with current and future technologies of information. The emergence of virtual reality is no exception. For example, private agencies might find in virtual reality solutions to acquire details about users' preferences and interests [18]. They might do so by tracking and collecting details of users' movements in the virtual reality. This could potentially include eye movements, involuntary facial gestures, and other measures of what researchers call low-level intentions or "motor intentions". To go even further, if in the future, avatars themselves were to be be used as "humanoid interfaces," consumers could be influenced and manipulated by real-time feedback of the avatar's own facial and eye movements. Commercials in virtual reality could even feature images of users themselves using a marketable product.

4. **Sensory vulnerability:** Virtual Reality aims at fully immersing users in a virtual reality environment. Therefore, while being in the virtual reality, consumers are fully disconnected from the world, in terms of what they see or hear. Although the resulting sensation is extremely realistic, people using virtual reality and especially those using it alone at home should become a concern. Having limited access to sense data leaves users vulnerable to accidents, home invasions, and any other unfortunate events that result from being totally disconnected and distracted from the real world.

Bibliography

[1] K. B. C. Tyler Rose, Chang S. Nam, "Immersion of virtual reality for rehabilitation - review," *Applied Ergonomics*, vol. 69, pp. 153–161, 2018.

[2] "collider." https://en.wikipedia.org/wiki/Illusions_of_self-motion.

[3] K. C. Iwan Kartiko, Manolya Kavakli, "The impacts of animated-virtual actors' visual complexity and simulator sickness in virtual reality applications," *2009 Sixth International Conference on Computer Graphics, Imaging and Visualization*, vol. 16, pp. 147–152, 2009.

[4] "Manus website." https://manus-vr.com/faq.

[5] "Manus website." https://manus-vr.com/hardware.

[6] "Nasa is using manus vr gloves to train astronauts in mixed reality." https://uploadvr.com/nasa-using-manus-vr-gloves-train-astronauts-mixed-reality/.

[7] "Unity platform." https://en.wikipedia.org/wiki/Unity_(game_engine).

[8] "Unity platform." https://docs.unity3d.com/Manual/VROverview.html.

[9] "Steam." https://assetstore.unity.com/packages/templates/systems/steamvr-plugin-32647.

[10] P. T.-Y. C. H.-R. C. J.-Y. Hsiao Jen-Hsuan, Deng Yu-Heng, "Design of a wireless 3d hand motion tracking and gesture recognition glove for virtual reality applications," *26th ASME Annual Conference on Information Storage and Processing Systems*, pp. 1–3, 2017.

[11] L.-K. S. H. Z. Gan Lu, "Immersive manipulation of virtual objects through glove-based hand gesture interaction," *Virtual Reality*, vol. 16, pp. 243–252, 2011.

[12] B. K. M. T. T. J. P. S. Saurabh Jadhav, Vikas Kannanda, "Soft robotic glove for kinesthetic haptic feedback in virtual reality environments," *Society for Imaging Science and Technology*, pp. 19–24, 2017.

[13] "Unity platform." https://docs.unity3d.com/ScriptReference/Rigidbody.html.

[14] "collider." https://docs.unity3d.com/Manual/CollidersOverview.html.

[15] "Manus website." https://docs.unity3d.com/Manual/class-MeshCollider.html.

[16] A. A. R. Thomas D. Parsons, "Affective outcomes of virtual reality exposure therapy for anxiety and specific phobias: A meta-analysis," *Behavior Therapy and Experimental Psychiatry*, vol. 39, pp. 250–261, 2005.

[17] D. O. M. Tobias Greitemeyer, "Video games do affect social outcomes," *Sage Journals*, 2014.

[18] T. K. M. Michael Madary, "Real virtuality: A code of ethical conduct. recommendations for good scientific practice and the consumers of vr-technology," *frontiers in Robotics AI*, 2016.

Appendix A

User Guide

This section will provide a guide on how to use Mind Explorer Enhanced VR.

1. The program requires a Windows operating system (64 bit). It also requires an HTC Vive kit, two vive trackers sold with their respective dongles, a pair of Manus VR Gloves sold with a dongle and two wrist bands on which the trackers can be griped.

Figure A.1: Virtual reality material required for Mind Explorer Enhanced VR

2. Install SteamVR via Steam on the machine.
3. Plug the dongle for the gloves and the two dongles for the vive trackers into the machine and turn on the gloves and Vive trackers.
4. The user needs to equip himself by placing each band with the trackers attached on his wrist and put on the gloves. With all these things in place, the program can be run by opening the 'MindExplorerEnhancedVR' executable.
5. Once the program loads, the headset can be put on. The hands should appear misaligned. Setting the rotation of the hands can be done automatically by pressing space-bar repetitively until the hands are aligned .

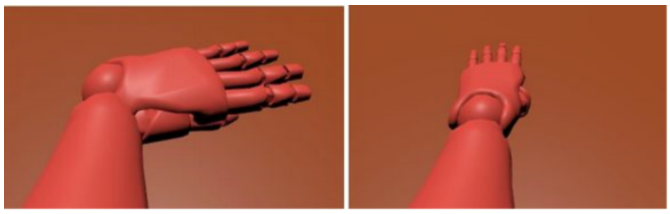


Figure A.2: Left: Hand misaligned as it first appears. Right: Hand aligned by pressing the space key

> The user should be able to look around and see himself in a dark room with black walls. There should be a grey column with a red button on top, along with a sign that says 'reset'. The user should also see a large screen attached to one of the walls, and the full neocortex line model in the centre of the room. This may first appear to be behind the user, depending on VR calibration. The user can move around the room by walking, if the set up room scale VR in SteamVR has been set up.

It is possible to perform queries on objects in the scene within distance or grab them directly as followed:
Accessing the Menu: Use the right or left index to respectively press the left and right menus located on each forearm. The menu can be closed in the same way as opening it.
Distance Move Mode: Make a fist with the right or left hand to move the neocortex directly when this last one is selected in the menu, or query, message and connectivity cubes when they are selected in the menu.
Distance Scale Mode: Make a fist with the right and left hands, then stretch the arms to increase the scale of the selected object or decrease the distance between the hands to reduce the scale of the selected object.
Distance Rotation Mode: Select "Model" in the Menu. Make a fist with left or right hand then place the hand that is not making a fist on the elbow of the hand that makes the fist. Then while holding this position, rotate the model along, the x, y and z axis using rotations from the wrist.
Distance triggering Mode: Select "Cube" in the Menu and the desired mode (query, message or connectivity). Place the cube in a desired area of the cortex from the signal will depart from. Then clap with the hands to trigger the signal.
Object Grabbing Mode: Select "Cube" in the Menu. Grab the cube with the hands when it is within grabbing distance. If some difficulties are experiences to grab the cube, use both hands to facilitate the grabbing. **Grabbing Scaling Mode:** Grab the cube with two hands by making a fist with each hand and scale the cube by changing the distance between the hands.

Appendix B

Test Results

The distance Move Mode refers to all queries that enable to execute actions on objects within distance.												Average
How easy was it to move objects within distance? (rate from 0 to 10)	8	9	8	8	7	7	8	10	7	8	8	8,00
How easy was it to scale objects within distance? (rate from 0 to 10)	8	7	7	8	7	6	8	8	7	8	8	7,45
How easy was it to rotate objects within distance? (rate from 0 to 10)	6	6	7	6	7	6	7	7	7	8	7	6,73
How easy was it to trigger a signal in the neocortex? (rate from 0 to 10)	9	9	8	9	9	8	8	9	9	8	8	8,55
How easy was it to perform queries in general in the application? (rate from 0 to 10)	7	7	7	7	7	6	8	8	7	8	7	7,18
How useful did you think the Distance Mode was in general? (rate from 0 to 10)	8	9	9	8	8	7	8	8	8	9	8	8,18
The Grabbing Mode refers to all queries that enable to execute actions by directly grabbing objects.												
How enjoyable was it to be able to grab objects directly with hands? (rate from 0 to 10)	9	8	9	9	9	10	9	8	7	9	8	8,64
How easy was it to grab an object with the hands? (rate from 0 to 10)	6	7	7	6	5	5	6	7	6	7	6	6,18
How useful did you think the Grabbing Mode was in the context of this application? (rate from 0 to 10)	6	6	6	5	4	5	6	7	5	6	6	5,64
You tried 2 applications, a first one that uses gloves and a second one that uses controllers.												
How would you compare your VR experience in the first application compared to the second one (answer one of those: much better, better, same, worst, much worst)	better	better	better	better	same	same	better	better	better	much better	better	
Did you think the first application was more intuitive to use in terms of controlability? (answer yes, same or no)	yes	yes	yes	yes	same	yes	yes	yes	yes	yes	yes	

Figure B.1: Questionnaire results from 11 people who tried both Mind Explored Enhanced VR and Mind Explorer VR

Appendix C

Ethical Requirements

	Yes	No
Section 1: HUMAN EMBRYOS/FOETUSES		
Does your project involve Human Embryonic Stem Cells?		✔
Does your project involve the use of human embryos?		✔
Does your project involve the use of human foetal tissues / cells?		✔
Section 2: HUMANS		
Does your project involve human participants?	✔	
Section 3: HUMAN CELLS / TISSUES		
Does your project involve human cells or tissues? (Other than from "Human Embryos/Foetuses" i.e. Section 1)?		✔
Section 4: PROTECTION OF PERSONAL DATA		
Does your project involve personal data collection and/or processing?	✔	
Does it involve the collection and/or processing of sensitive personal data (e.g. health, sexual lifestyle, ethnicity, political opinion, religious or philosophical conviction)?		✔
Does it involve processing of genetic information?		✔
Does it involve tracking or observation of participants? It should be noted that this issue is not limited to surveillance or localization data. It also applies to Wan data such as IP address, MACs, cookies etc.	✔	
Does your project involve further processing of previously collected personal data (secondary use)? For example Does your project involve merging existing data sets?		✔
Section 5: ANIMALS		
Does your project involve animals?		✔
Section 6: DEVELOPING COUNTRIES		
Does your project involve developing countries?		✔
If your project involves low and/or lower-middle income countries, are any benefit-sharing actions planned?		✔
Could the situation in the country put the individuals taking part in the project at risk?		✔
Section 7: ENVIRONMENTAL PROTECTION AND SAFETY		
Does your project involve the use of elements that may cause harm to the environment, animals or plants?		✔
Does your project deal with endangered fauna and/or flora /protected areas?		✔

Does your project involve the use of elements that may cause harm to humans, including project staff?		✔
Does your project involve other harmful materials or equipment, e.g. high-powered laser systems?		✔
Section 8: DUAL USE		
Does your project have the potential for military applications?		✔
Does your project have an exclusive civilian application focus?		✔
Will your project use or produce goods or information that will require export licenses in accordance with legislation on dual use items?		✔
Does your project affect current standards in military ethics – e.g., global ban on weapons of mass destruction, issues of proportionality, discrimination of combatants and accountability in drone and autonomous robotics developments, incendiary or laser weapons?		✔
Section 9: MISUSE		
Does your project have the potential for malevolent/criminal/terrorist abuse?		✔
Does your project involve information on/or the use of biological-, chemical-, nuclear/radiological-security sensitive materials and explosives, and means of their delivery?		✔
Does your project involve the development of technologies or the creation of information that could have severe negative impacts on human rights standards (e.g. privacy, stigmatization, discrimination), if misapplied?		✔
Does your project have the potential for terrorist or criminal abuse e.g. infrastructural vulnerability studies, cybersecurity related project?		✔
SECTION 10: LEGAL ISSUES		
Will your project use or produce software for which there are copyright licensing implications?	✔	
Will your project use or produce goods or information for which there are data protection, or other legal implications?		✔
SECTION 11: OTHER ETHICS ISSUES		
Are there any other ethics issues that should be taken into consideration?	✔	

Figure C.1: Ethical Requirements Questionnaire