Dissertation Type: research

DEPARTMENT OF COMPUTER SCIENCE

Motion in Mobile Virtual Reality

Erich-Robert Reinholtz

A dissertation submitted to the University of Bristol in accordance with the requirements of the degree of Master of Engineering in the Faculty of Engineering.

Friday 21st May, 2021

Declaration

This dissertation is submitted to the University of Bristol in accordance with the requirements of the degree of MEng in the Faculty of Engineering. It has not been submitted for any other degree or diploma of any examining body. Except where specifically acknowledged, it is all the work of the Author.

Erich-Robert Reinholtz, Friday 21st May, 2021

Ethics Declaration

The project experiment was approved by the Chair of the Faculty of Engineering Research Ethics Committee, OREMS Ref = 0016

Contents

List of Figures

Executive Summary

My project of choice involves researching motion improvements for mobile Virtual Reality. The purpose of the investigation is finding methods which expand immersion and aid in locomotion while, at the same time, don't cause a remarkable increase of cost. Essentially, the purpose of my analysis is trying to enhance affordable VR while still maintaining its accessibility.

The main area of interest tackled is the reduction of motion sickness, subsequently the increase of interactivity with the environment using real body motion; all these while taking into account the task of maintaining the technology on an accessible level.

> My research hypothesis is that, through affordable hardware, we can achieve a decent mobile VR experience, which lessens motion sickness and increases user immersion.

My contributions are:

- I studied the context and needs of mobile VR technology and its users, see Chapter 1
- I learnt how to use `Unity Engine`, with close to no previous experience.
- I tested different hardware and respective methods of inter-connecting them.
- I investigated previous relevant work, see 2.2
- I researched and developed individual potential methods for mobile VR improvement and then selected successful ones for peer testing, see Chapter 3
- I implemented a working, affordable hand tracker device for VR locomotion and immersion, see 3.1.2
- I implemented a FOV restrictor for dynamic peripheral view manipulation, see 3.1.4
- I created simulations for the user experiment, see 3.2
- The final version of the simulation script contains roughly 500 lines of code.
- I ran the simulations on volunteers and collected data about their experience, see 3.2.3
- I analysed the collected data and drew conclusions, see Chapter 4 and Chapter 5
- I planned out a potential follow-up, based on my findings, see 5.2

Supporting Technologies

- Unity Engine, version 2019.4.21f1 to create 3D mobile VR environments and do post-processing (including the Android Tools package for mobile app generation) https://unity.com/. All applications were built to have a framerate of 60FPS.
- The C# programming language to script all events
- A Bluetooth device from Esperanza (EMV 101) as a button-based hand controller
- An Esperanza pair of phone-compatible VR glasses (EGV300); even cheaper variants such as Google Cardboard can also be used instead
- A Samsung Galaxy S9 as a main VR device (can be replaced by any Android device featuring a gyroscope).
- A Samsung Galaxy A5 as a real-hand-movement controller (can be replaced by any Android device featuring an accelerometer or, with certain tweaks to the algorithm, by other, cheaper accelerometer-bearing devices, i.e. fitness bracelets).
- The Photon Networking Framework to facilitate device-to-device communication https://www.photonengine.com/pun
- The Physics Toolbox app to monitor phone sensor detections and generate graphs https://play.google.com/store/apps/details?hl=en&id=com.chrystianvieyra.physicstoolboxsuite

Notation and Acronyms

VR	:	Virtual Reality
FOV	:	Field of View
FPS	:	Frames per Second
PV	:	Peripheral View
DoF	:	Degrees of Freedom
g	:	Gravitational Acceleration
Lerp	:	Linear Interpolation
SSQ	:	Simulator Sickness Questionnaire

Acknowledgements

The most consistent third-party advice I received is related to experiment running guidelines, which were provided by my supervisor, Anne Roudaut.

For one particular subsection, I also received support from a friend who studies physics, who provided a useful formula and demonstration. The friend chose to remain anonymous.

I should also mention the input that my volunteers have provided by taking part in the experiment.

Chapter 1

Contextual Background

The topic of my project is motion in mobile virtual reality. My investigations relate to the quality of such experiences and finding ways to provide them in accessible, non-costly manners.

Virtual reality has popularly existed as a practical experience for more than 40 years. The first wave of virtual reality as we know it happened in the 1980s [3] and the first consumer headsets started to get sold by video game companies in the early 1990s.

The earliest of VR sets, despite being a positive addition to the global set of interactive technologies, did, however, bring along many downsides. Especially in their early stages, these experiences would cause direct negative effects towards the users.

Since, back in the day, the technology was limited, the image quality was poor and unrealistic. [4] The lack of any body tracking, especially head tracking, was also a big impediment, since this would often prove to be a cause of motion sickness. Other issues would often even result in eye strain and damage.

Since then, the virtual reality industry has experienced unprecedented progress through innovative trackers and sensors which are included within most of the modern devices. Currently, some of the best rated VR headsets [23] are `Oculus Quest`, `Valve Index`, `PlayStation VR`, `HTC Vive` etc. One thing that all of these devices have in common is actually the fairly substantial price.

It is important to note that, despite the progresses which have been done with regards to the quality of VR headsets, the side effects have not yet been eliminated. Many users still experience some form of sickness, either during the experience or after the undergoing has been terminated.

A study conducted by VRHeaven [13] shows exactly that: more than half (57.8%) of all VR users have experienced motion sickness at least once (figure 1.1). Other studies, such as this one [18] show similar outcomes, 64% of participants having had some motion sickness symptoms subsequent to VR exposure (figure 1.2).

This percentage obviously varies depending on the experience's individual characteristics, however, precaution should always be taken, in order to diminish possible side effects. Generally, when developers work on VR-targeted software, they begin with the initial assumption [24] that at least 25% of the users will definitely endure motion sickness to an extent.

There could be many cases to these side effects, however a common one is motion in VR. If, in a virtual environment, looking around and watching doesn't feature much difference from the real world, motion does. This often happens due to the environment scale or the lack of good tracking.

Environment scale is an issue because the user will often "walk" through worlds that are much bigger than the available physical space, hence requiring alternative locomotion methods than just real life physical walking. Such methods include teleportation, touchpad/joystick navigation, head-based movement, etc.

However, these methods can become a necessity for the experience even if the physical movement space is sufficient. That is, when the technology in use does not feature body trackers and, hence, cannot translate real world motion into the virtual environment.

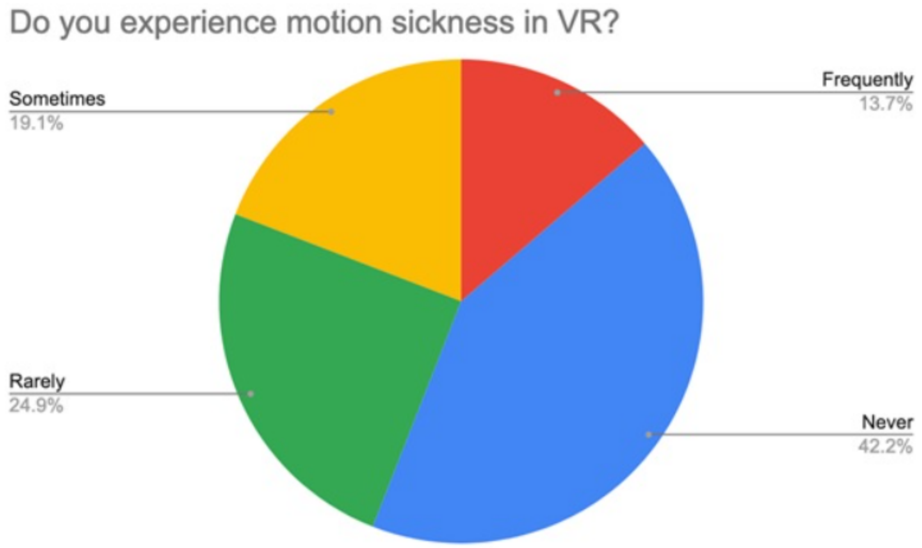


Figure 1.1: Piechart showcasing the percentage of users who experienced motion sickness at least once, courtesy of the VRHeaven study [13]

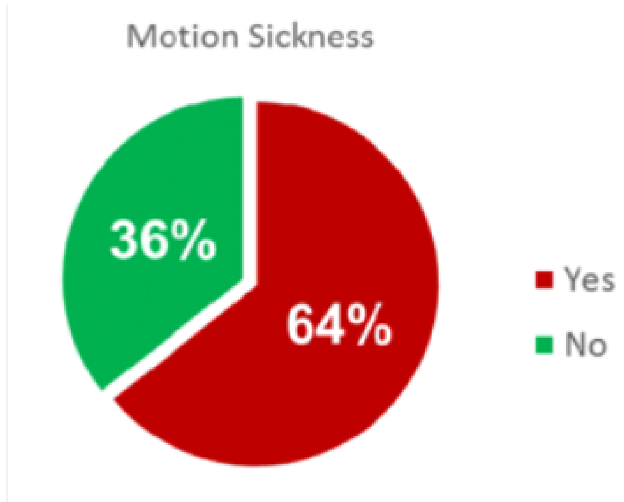


Figure 1.2: Piechart depicting the percentage of users affected by motion sickness, courtesy of this study[18]

A study[7] has made an analysis of Head-Mounted display Virtual Reality Sickness. In order to measure the variation of effects onto the users, the experiences were split into 4 categories: "Gaming Content", "360 degree videos", "Minimalist Content", "Scenic Content".

The quantification of sickness reactions was done through the `Simulator Sickness Questionnaire (SSQ)` [19]. In my experiment, which I described in Chapter 3, I made use of the same method in order to track the participants' symptoms.

The weightings of the results show that the experiences which recorded the highest level of sickness were the "Gaming Content" and the "360 degree videos", the former of which was the most underperforming. This is detailed in figure 1.3.

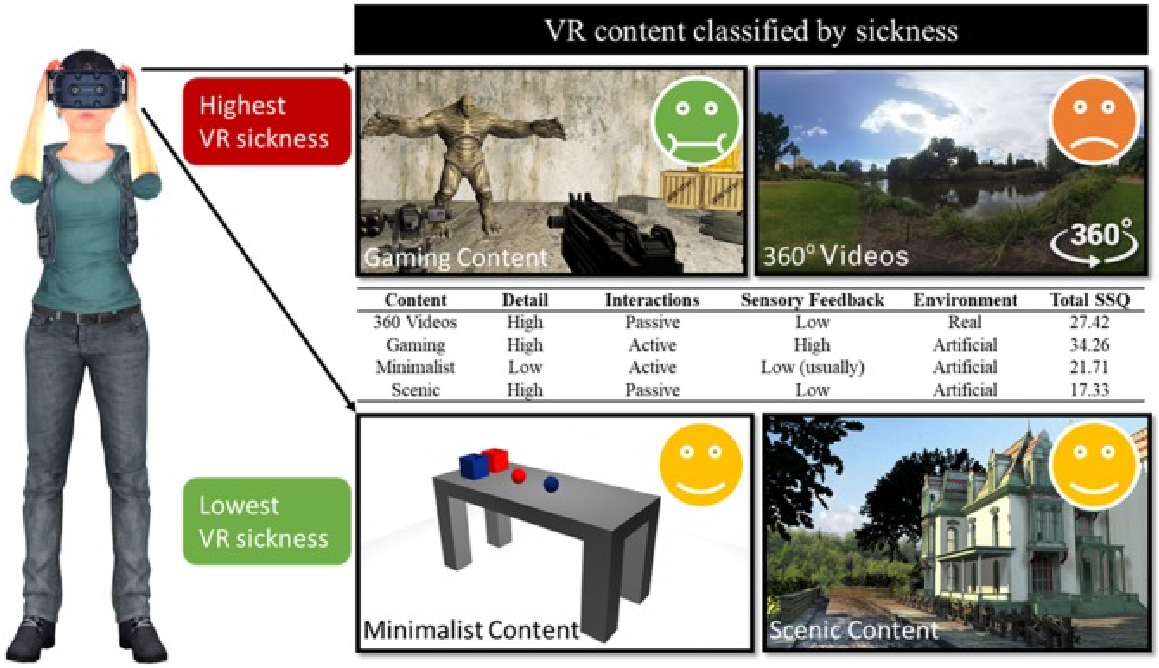


Figure 1.3: Image courtesy of this study[7]

The reason behind this outcome is a combination of many factors, for example the high level of detail, which adds to the spatial confusion of the brain.

However, the truly major deciding factor, which causes sickness in gaming content, seems to be the level of user-application interaction. The integration of real life movements is a key constituent in how true the virtual display is to a real world equivalent.

This factor is further reinforced by the fact that, most of the times, the said real life movements are combined with additional artificial input. This is often done through separate controller hardware (keyboards, joysticks, touchpads) or even means such as actions caused by staring at one particular display point for a certain length. These methods of artificial input tend to compensate for in-game gestures that are unable to be replicated by real live movements. The most popular example is the continuous walk/run motion, but other examples could be jumping maneuvers, object grabbing, collisions and so forth.

This aspect is highly relevant to the particular aims of my project, because of the assertive interaction factor. My attempts are targeting this area specifically; investigations are done regarding character control that is fully under the user's command. This leads to attempted techniques which deliver highly interactive continuous movement.

In many cases, with the passing of time and the accumulation of experience, many users can get accustomed to the VR experience and its differences from real life. Therefore, with practice, the users can adapt to the virtual world and the method of locomotion, hence creating the so-called "VR legs".

Their principle is similar to that of "sea legs" which are often mentioned when it comes to sea sailors' adaptation to months on end spent on a floating boat. Just as they become accustomed and often immune to the side effects of the boat wiggling while at sea, many VR users can follow a similar pattern.

Despite this fact, this is not always the case, as this study[13] suggests.

According to these statistics, even with proper, lengthy VR training, motion sickness will still be experienced by a plentiful amount of users: more than a 5th of the total sample (figure 1.4).

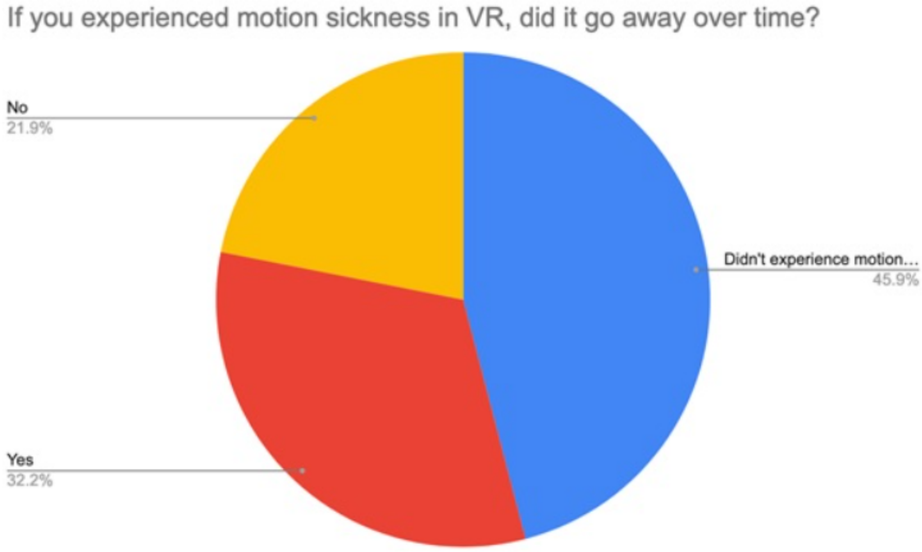


Figure 1.4: Piechart showcasing the percentage of users whose motion sickness diminished over time, courtesy of the VRHeaven study [13]

My paper tackles a very particular area of VR, one that's risen fairly recently and that often gets overlooked in favour of other, more sophisticated alternatives. The area I'm interested in is the very affordable, low-end, widely accessible mobile VR.

It is by far not the most efficient variant of its kind, because of the sheer amount of high performing sensors and trackers present in the other, more costly VR headsets.[23]

According to this study[13], the VR headset type which causes the highest amounts of motion sickness is, in fact, the Cardboard/Low-End mobile version (figure 1.5).

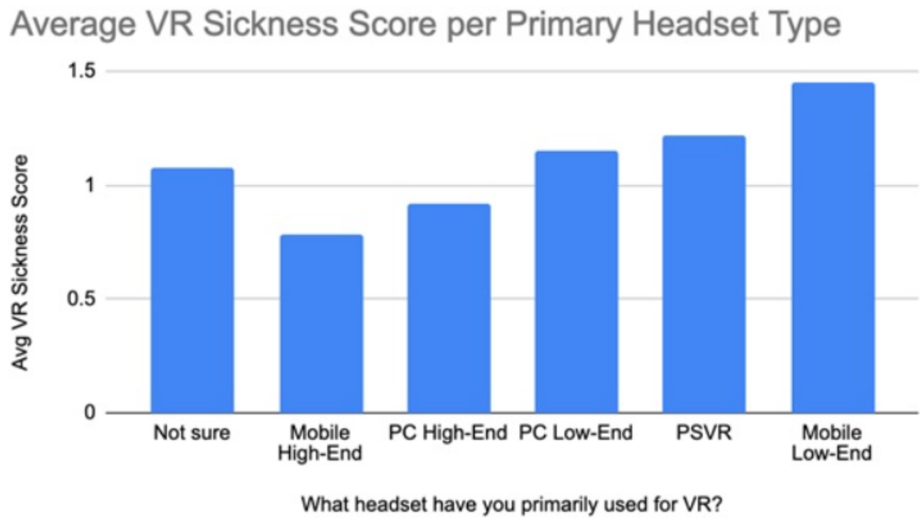


Figure 1.5: Graph showcasing VR types and their respective Sickness Score, courtesy of the VRHeaven study [13]

It is understandable why this may happen because, despite having gyroscopes and accelerators, mobile devices lack many other features which expensive headsets provide. The most notable such feature is the presence of positional tracking systems. Not just the head placement, but also an active in-world visual of arms, fingers and even legs, can really make a difference when it comes to the immersion of a VR experience.

However, the crucial aspect about mobile VR lies in its accessibility and spread. Smartphones capable of running VR applications are a frequent sight nowadays. Extending them to turn them into VR sets can be done as easily as through purchasing a cheap head mount. The price of such a piece of equipment can be as low as 3 pounds. *[example]*

This means that this particular variant of VR can become really common amongst broad audiences at a very quick rate. The equipment is already there so there is almost no need for extra purchases.

This also might mediate those people who maybe one day would want to experience expensive headsets but are not sure if the investment is worth it. Hence, using improved mobile VR would be a perfect introductory experience, after which the user can choose to opt in or out of a more expensive headset acquisition.

On a more general level, Virtual Reality is becoming increasingly popular amongst broader audiences. The worldwide shipment of VR devices in 2021 (11 million) is projected to double the value recorded in 2020 (5.5 million).[11]

These VR technologies are multi-purpose. Not only are they utilized by many gamers, but they are becoming extremely important in other areas as well, such as education, medicine, tourism, mental health, even military training or astronaut simulations. [12][25]

Since this technology is becoming a crucial tool in more and more areas of interest, it only makes sense that a larger number of people will need access to it in some form.

Due to their high availability and accessibility, mobile head-mounted VR display devices have the most potential for wide-spreading.

Considering these two factors, the growth of VR at a general level, paralleled with the easiness of expansion of the cheap mobile variant, we can clearly depict that this latter type of VR will become even more heavily relied on by a sheer amount of users.

Therefore, I think it is crucial that steps are taken so that the experience gets improved. The aim here is trying to find ways to reduce the side effects of such experiences, like motion sickness, and attempting to increase the overall immersion and interaction with the environment.

The key element about this is that this needs to be done in a non-costly manner. The whole idea behind choosing to focus on this variant originates in its importance as a cheap alternative to expensive VR sets.

Therefore, the purpose of my paper is researching potential improvements to motion in VR; in order to conserve the accessibility factor, they either shouldn't use any additional hardware tools, or should use extremely low-budget ones, potentially making use of devices which can be found lying around unused in the user's home.

Here is the bulletpoint list of concrete aims:

1. Researching improvements for motion in mobile VR
2. Filtering the methods so that they are done in an affordable manner, to preserve accessibility
3. Creating simulations featuring the successfully implemented techniques
4. Having the simulations ran through by volunteers and collecting data about their experiences
5. Using the data to draw conclusions and identify what works and where there is room for future improvement

Chapter 2

Technical Background

Previous research about the particularities of my project's aims is actually quite limited, when compared with that of peak-level high-performing headsets.

This makes sense considering that the former is generally much more acclaimed by experts and enthusiasts of VR who, understandably, tend to steer away from the variant that I'm attempting to work on.

Hence, all resources and technological background used for my project is more or less picked out from various distinct sources, many with different original intents.

2.1 Technical resources

Low-level mobile-hosted Virtual Reality generally is used in some contexts more than others. Because of the limitations of the DoF to only three dimensions, the mobile VR experiences are often restricted to passive interactions. Examples of this are virtual observations of real or fictional places, educational simulations, virtual rides or even simply 360°videos. [15]

What I'm looking for, however, is creating mechanisms for experiences where the user takes full control over their virtual maneuvers. I'm exploring methods for highly interactive continuous movement, which will require a combination between various different pieces of hardware and software, as well as integrated utilization of extensive relevant documentation.

The software basis on which my project is built consists mostly of basic functions and components native to the `Unity Engine`. In order to achieve this, I had to acquire basic Unity knowledge from almost ground-up.

I have used Unity's 3D environment editor, with tools such as scale, rotate, etc. in order to build testing grounds for the various techniques I've explored. In order to obtain body-to-body dynamics and in-game forces such as gravity, I have utilized Unity's colliders and rigid bodies. The language used for all scripting was Unity's native `C#`.

I have also utilized Unity's own `Android` plugin, which compiles the experience into an Android-supported app. The app is ran on the VR host smartphone, which is placed in a cheap phone-compatible pair of VR glasses (figure 2.1). The said pair can be replaced by even cheaper variants, such as `Google Cardboard`.

No pre-implemented first-person controller was used; everything related to the perspective controller was coded by myself from scratch.

In order to achieve basic VR locomotion, I have made use of a cheap `Esperanza Bluetooth` controller (figure 2.2), which acts as a simplistic remote control for `Android/iOS`. It features a few simple buttons, two of which are placed on the front. Their positioning helps in giving the user the impression of a grabbing gesture, which aids later in some of the implemented techniques. It also features a joystick which, in certain cases, I've utilized to generate movement.

All device-to-unity button mappings and functionalities were done by myself from ground up; no pre-implemented plugin or script was borrowed.

As mentioned previously, in certain cases, my project involves making use of a secondary device, besides the one responsible for hosting the actual VR simulation. The secondary device, which I call controller device (not to be confused with the Bluetooth joystick controller) can, in theory, be any piece

Figure 2.1: The *VR headset* used

Figure 2.2: A depiction of the Bluetooth *remote controller* for Android that was used

of equipment that features an accelerometer. An example would be another, possibly older phone or a cheap fitness bracelet; in my case, I've utilized the former.

The communication between the controller device and the VR host device is facilitated through the `Photon Realtime Network`. Hence, I made use of its own methods in order to pack and send data from one end, respectively receive and unpack it on the other. To achieve this, I made extensive use of Photon's own official *documentation*.

Another plugin I should mention is Unity's own Post-processing , which was utilized for one of the investigated techniques.

In order to incorporate real body motion, I have attempted several techniques, which target different body parts and use different mechanisms. The common ground between all of these techniques is the shared idea of using the device's own incorporated sensors in order to detect motion.

The most frequently present sensor in any device, which is used by most of the attempted techniques, is the accelerometer. The main function of any accelerometer is detecting the device's acceleration in 3-dimensional space.

One important thing to note here is that the identified acceleration values are not, by default, independent from the detected value of the G-force. This means that, depending on the position of the device, the accelerometer will take the Earth's gravitational pull into account and interpret it as an addition to the acceleration in the direction of a certain axis. When the phone tilt is situated in between certain positions, the respective gravitational acceleration value will be spread amongst at least two different axes, as seen in figure 2.3.

This is really useful because it means that, if the phone is in a stationary state in a particular position, the ratio of these values can reflect the vertical rotation state of the device. Hence, the most common sensor in devices, found even in some of the cheapest pieces of technology, can even, to a certain extent, give rotation information.

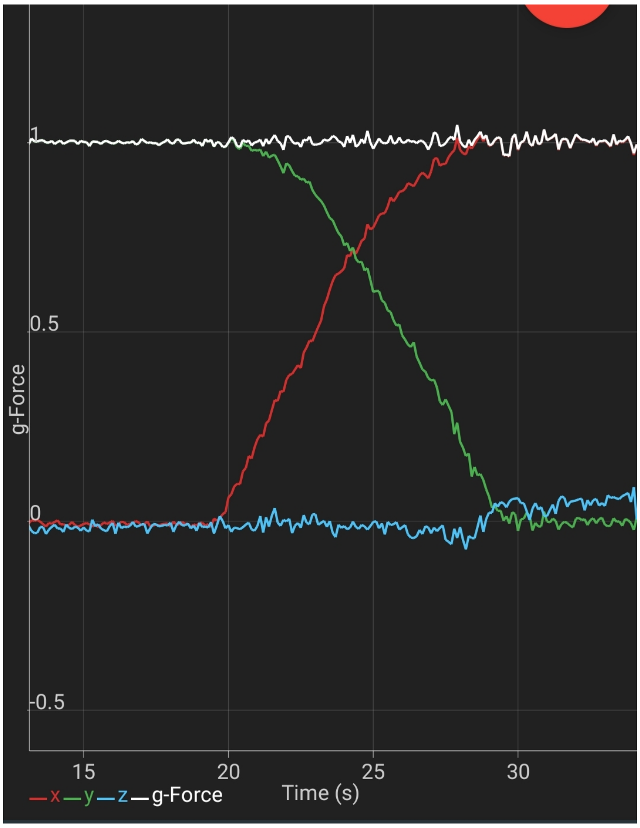


Figure 2.3: This is a graphical depiction of the gravitational acceleration spread amongst the axes. The data was collected by the Android device's accelerometer sensor and the application used for the illustration is `Physics Toolbox`.

The figure begins with the phone in a state where it's vertical, aka "portrait", and stationary. All axes show zero acceleration except for y, which encompasses the entire value of g (9,8N m/square second). After roughly time point 19, a slow horizontal tilt movement is applied, which ends around time point 29, with the phone being stationary in the horizontal, "landscape" position.

During the movement, a gradual transfer of the g value from the y axis to the x axis can be noticed. This is a demonstration of how, during the procedure, the g detection gets spread amongst the axes and can be used on a three-dimensional level to determine the device's inclination at any given point. Since the sensor in work here is just the accelerometer, in order to suffice this process, there is no need for a gyroscope.

Since the former sensor is more common and inexpensive than the latter, this aids in the overall objective of the project, which involves finding techniques for mobile VR which are as accessible as possible.

The other sensor that is crucial for the technology to work is the gyroscope (figure 2.4). It is mandatory that the gyroscope be present in the main device which is inserted into the VR glasses, however any other piece of technology utilised in this project will not be in need of one.

The gyroscope is important because of its ability to not only show rotation on one axis, like the accelerometer, but rather, on all 3. The precisions of the gyroscopes of most phones are decently accurate and can be translated into rotation values in Unity. This means that we can get very natural, immediate representations of real-world head movement into the virtual environment.

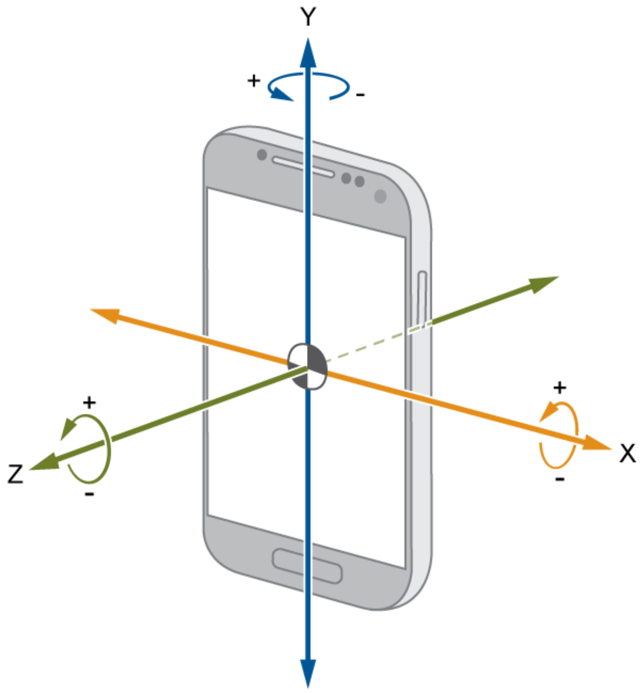


Figure 2.4: An image borrowed from the MathWorks documentation. It features mobile-specific gyroscope mechanics, which offer 3DoF.

It would also be important to note that the built-in gyroscope functionalities in Unity allow for immediate calculations of rotation rates. This can be extremely beneficial in determining when rotation is occurring and when not, hence helping in filtering out what each motion detected by the device actually corresponds to in real life.

When working with the values recorded by these two sensors, I made slight use of the help of a friend who studies physics. They provided me with a formula and demonstration aiding in distance calculation, based on acceleration (figure 2.5).

I later constructed a variation on this, in order to attempt at getting virtual distance equivalents from device sensor acceleration data.

$$a = \frac{dv}{dt} \Rightarrow a\,dt = dv \Rightarrow a\int_{t_0}^{t_1} dt = \int_{v_0}^{v_1} dv$$

$$\Rightarrow a(t - t_0) = v - v_0 \Rightarrow at = v - v_0$$

$$\Rightarrow v(t) = a \cdot t + v_0$$

$$v = \frac{dx}{dt} \Rightarrow v\,dt = dx \Rightarrow (a \cdot t + v_0)\,dt = dx$$

$$\Rightarrow \int_{t_0}^{t_1} a \cdot t \cdot dt + \int_{t_0}^{t} v_0\,dt = \int_{x_0}^{x} dx$$

$$\Rightarrow a\left(\frac{t^2}{2} - \frac{t_0^2}{2}\right) + v_0(t - t_0) = x - x_0$$

$$\Rightarrow \frac{at^2}{2} + v_0 t = x - x_0 \Rightarrow x(t) = x_0 + v_0 t + \frac{at^2}{2}$$

$$\xRightarrow[v_0=0]{x_0=0} x(t) = \frac{at^2}{2}$$

Figure 2.5: The background and formula for acceleration-to-distance calculation, based on the physics student's indications.
- `a` represents acceleration
- `t` represents time
- `v` represents velocity
- `x` represents distance

2.2 Previous related work

I would like to begin with a piece of previous related work which I drew inspiration from; I am going to briefly present it.

The method itself is not implemented within a mobile VR context, but actually integrated in an application which I've personally trialed on the headset `HTC Vive`.

The technique is only a small ingredient in the overall functionality of the application, however is related to an important topic that I'm exploring in Subsection 3.1.4. This is an important factor [8] in the occurrence of VR sickness, regardless of the type of headset.

The asset in question is a feature on Google Earth VR, which can be toggled to activate during motion. It enacts at the moment when the user utilizes the hand controllers to move around the map using drags. What the method does is, when this motion is initiated, it removes the outer visual margins completely. Therefore, the Peripheral View content is suddenly dynamically diminished, which leads to a lessening of user motion sickness.

A concrete example of this technique can be watched here. In Subsection 3.1.4, I am showcasing my investigations for mobile VR sickness issues with a dynamic Peripheral View handler, for mobile headsets. Furthermore, I will also be detailing my attempts at balancing this out with one of the aims of my project, immersion. My idea is investigating a mobile VR method that contributes to the post-VR effects but which doesn't take away from the experience itself.

What I'm also interested in is how this particular asset would perform when combined with different locomotion methods, and whether it would contribute to significant changes.

Next, I am going to talk about interaction and locomotion.

Throughout previous studies, several attempts have been made at upgrading the interaction element of mobile VR experiences.

Due to the mobile VR variant's restrictive nature, caused by device limitations, many of these approaches led to utilization of extra hardware. Although useful, this can often be counterproductive when it comes to the principles that I'm trying to adhere to. Some such devices add unnatural motion and also increase the budget of the hardware, which goes against my goal of accessibility.

One such initiative is FaceTouch [14], where the interaction with the virtual world is done through a headmounted touchpad. This touchpad is situated on the oppposite side of the VR host device, whose screen is unaccessible because it is functioning as a display.

This display method has great advantages because, when scaled properly, the back mount controller can successfully mirror the front-side host device's display, hence allowing the user to pick on certain elements such as menu options or certain points in the VR environment (figure 2.6).

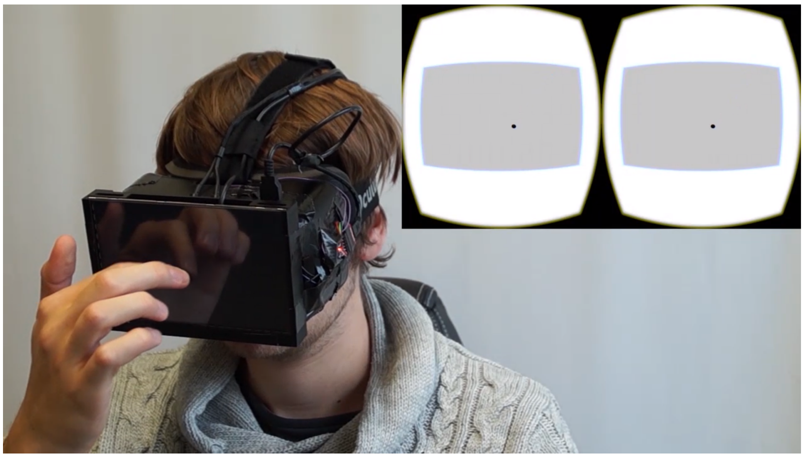


Figure 2.6: Screenshot from the FaceTouch[14] study's presentation video.

However, this technique has two aspects which are unsuitable for the aims of my project.

The first issue is that, by default, a customised head-mount which also features the touch controller would be weakening the affordability aspect. Workarounds could be found, such as using an older phone's screen as a touch controller, however this would still require a custom headmounted piece which can fit both that and the VR host device. This, once again, is an impediment to the accesibility goal that I'm trying to achieve.

Another issue is related to the hand motion itself. In real life, people rarely do that particular act, especially when it comes to locomotion. I anticipate that incorporating such a gesture in VR locomotion would, at least in case of my particular project, not contribute to the naturalness of interactions.

Another great attempt at mobile VR interaction, to which I've come across, was done by real hand gesture immersion [21].

The cited article shows the development of a technique which immerses real hand movements into the virtual experience. Single finger, as well as full hand gestures are transmitted from the real world to the application (figure 2.7).

However, despite its interactive nature, this method also has unsuiting aspects.

Firstly, the affordability aspect is, once again, not maintained, because this method requires several other pieces of hardware: two infrared cameras and one infrared ray.

Secondly, despite the method bringing about an increased level of interaction, this does not really tackle movement itself, but rather stationary interaction. In my paper, what I'm particularly interested in is immersion that is not only achieved through interaction at rest but also through user-controlled continuous movement.

There have also been attempts at creating the impression of real body motion. One such example, known as SIP (Swing-In-Place) [26] proposes a walking-in-place method for VR.

Figure 2.7: Illustration courtesy of the hand gesture interaction study[21].

Basically, the application relies itself on the body gestures done through a walking process, but not the actual motion itself. In real life, the user literally swings their body left to right while not actually changing place.

This way, body posture is preserved and the continuous movement doesn't depend on the user's gaze direction. This means that the user can look around the environment whilst maintaining a constant virtual forward direction.

There are, however, several inconveniences with this.

One of them is simply the fact that the motion proposed is not actual real motion, but rather an attempt at deceiving the human mind. The fact that the user is not effectively moving forward, respectively that the body posture corresponds to a walking motion, may contradict themselves and could cause inner ear balance perception issues.[17]

The other inconvenience is that each person walks differently and also swings differently. This means that it is very difficult finding uniform standard parameters for the degree to which the swing moves should be done, in order to get correct algorithmic interpretations.

It is quite clear that, in order to achieve the intended goals of my project, I had to take a different approach to that in previous investigations.

I aimed to explore methods which create tangible control of continuous locomotion, whilst maintaining the accessibility factor. I also looked at other potential assets to combine these with in order to reduce the quantity and intensity of VR sickness.

I will conclude this section by referring to the `Simulator Sickness Questionnaire (SSQ)` [19]. Other than interview-style questions and user ratings, this form has been the primary source of data collection from my experiment participants.

The form is an extensive method of comparing simulations and quantifying the side effects which they cause. It is a highly popular tool, frequently utilized when evaluating Virtual Reality experiences.

On a general level, it features a list of various symptoms; the experiment participants assign the level of severeness to which they experience each of them. The options are usually "None", "Slight", "Moderate" or "Severe".

Within Subsection 3.2.3, I have provided a detailed explanation of my application of the SSQ within the experiment.

Chapter 3

Project Execution

I carried out my project through 3 major parts:

- The research and implementation of mobile VR motion methods 3.1
- Compiling and conducting an experiment which tests the aforementioned techniques 3.2
- Critically evaluating the data obtained from the experiment 4

Within this chapter, I will be discussing the first two bullet points. I will be going over each method attempted, the outcome obtained and how I put the experiment together. In the next chapter (4), I will be evaluating what worked as expected, which results were less anticipated, as well as how the methods and experiments could be improved for future use.

3.1 Method Research and Implementation

I have investigated 4 different approaches regarding continuous movement in mobile Virtual Reality. I have developed interactive experiences with them and filtered them out for the following stage, the experiment.

3.1.1 Technique 1: Real-life movement

The first attempt that I did, which was the riskiest, was finding a way to translate real-life linear movement into the VR experience.

Initially, I attempted to achieve this solely through using the accelerometer sensor, while allowing the gyroscope to only be used with regards to the player head rotation and vision.

The idea behind this experiment was attempting to do this mechanism without the use of any other external equipment. An example of such a piece of equipment would have been a full body tracker or, at the very least, a head tracker. However, for the purpose of this experiment, I tried to implement this behaviour independently, by only using the device and its sensors at hand.

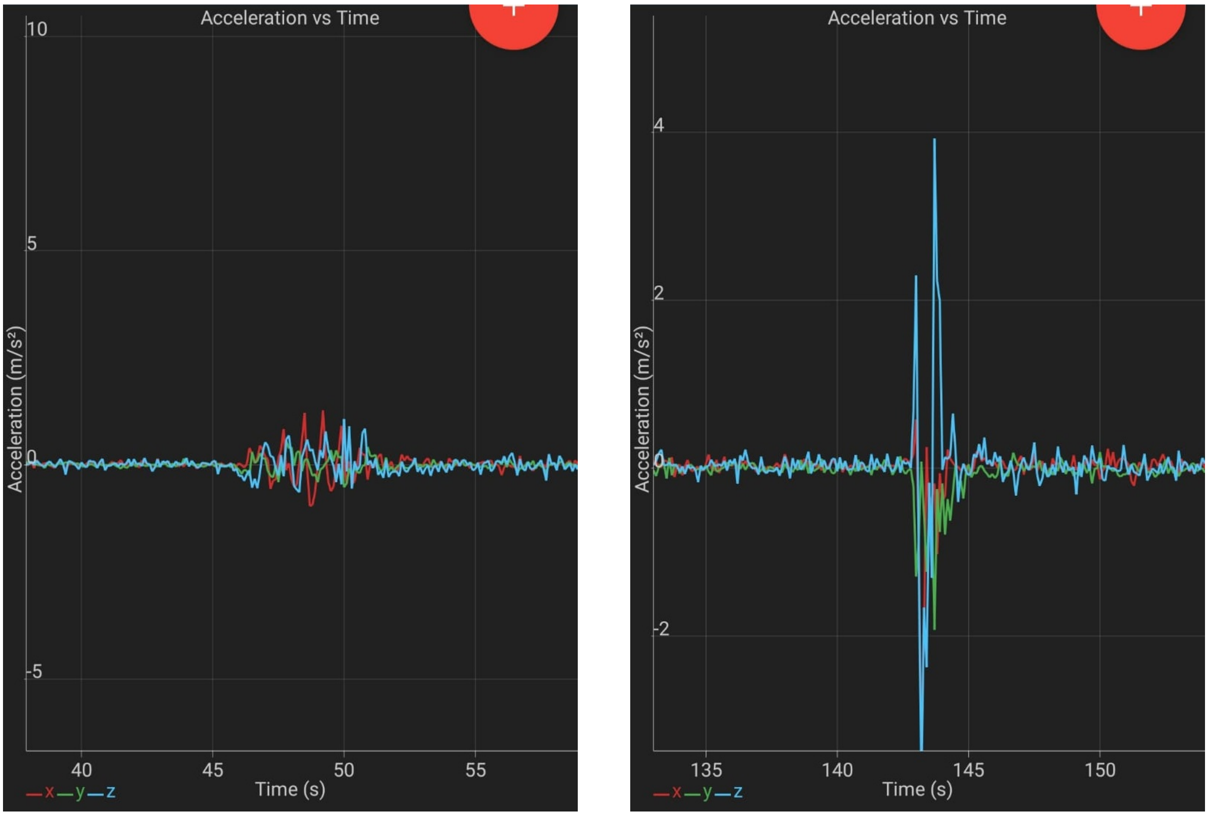
Essentially, the aim of this challenge was increasing the mobile VR degrees of freedom [2], from 3DoF to 6DoF, without additional gear or increased cost.

What I was actually interested in is the linear acceleration detected by the accelerometer, in order to find the value of the motion and its direction. Once detected, the increase in linear acceleration would notify the application to generate similar in-simulation movement, scaled to the virtual environment's unit of distance.

Below is a potential equation for determining the virtual position change on a certain axis, in this case the example axis being X. The acceleration detected by the device sensor on the respective axis is represented by `accX`, `t` is time and `RVS` represents a flexible scaler of real world distance to virtual world distance. The equation assumes `zero` initial velocity. This is an adaptation of the formula in figure 2.5.

$$PosX = PosX + \frac{1}{2} \cdot accX \cdot t^2 \cdot RVS \tag{3.1}$$

This seemed like a good idea, however, there is a big problem in this case. The mobile device's acceleration sensor in itself is only exactly what it is: an acceleration sensor. It detects any instance of



(a) Acceleration values when walking in a straight line.

(b) Acceleration values when doing a tilt.

Figure 3.1: The above graphs were generated using the application "Physics Toolbox". They show the 3-axis linear acceleration detected by the device's sensor.
The first graph depicts rest, followed by movement in a straight line, followed by another rest.
The second one features rest, followed by a phone tilt, followed by another rest.
Both graphs were generated using the "Linear acceleration" feature of the app, which removes the dependency of the values on Earth's gravitational acceleration.

an acceleration/force which is applied to itself; it doesn't assess where that particular value originates or the purpose of it. In other words, an acceleration on a certain axis can easily be provoked by many other manoeuvres than just the user linearly walking.

The most common cause for such confusions with regards to the sensor's perception is actually phone tilting itself. More particularly, quick, sudden tilts may generate spikes on axis-based acceleration monitors. The values at hand are often way larger than the ones created by, for example, linear forward walking motion. This is exemplified in figure 3.1.

Phone tilting is not a removable element in this technology, because it is actually the reason for the 360 degree perspective of the VR experience.

The potential solution I found which partially diminishes the issue is finding a way to segregate the spikes generated by rotation from the ones generated by linear movement.

The way I went about doing this was through evaluating the outcomes of `gyro`'s rotation rates. I would try to assume that, on a millisecond scale, the more the rates approach zero, the more it is likely that the user's head is at rest. This would mean that any detection of acceleration at that time would not be attributed to device rotation, but rather, to device linear movement, hence the user is walking.

However, the downside of this "fix" is that it is based on an assumption. Hence, it is impossible to detect if the stationary state is real or just apparent. Some value offsets used for this solution definitely outperformed others but, expectedly, none were able to completely erase the problem. This means that the device would still detect false positives during tilts and the virtual entity would be propelled forward, despite the user's stationary state.

One possible way to improve this method in future studies is by collaborating with machine learning researchers. An ML algorithm might be able to train to classify the sensor data depending on the source of movement. Hence, the spikes caused by linear walking and those generated by rotation could be very clearly distinguished and, hence, aid in creating more accurate real-to-virtual translations. I think this idea would be considerably useful to explore in future studies.

After careful consideration, I concluded that adding this particular technique to my series of volunteer trials was impractical. The experience, if it were to be done under these conditions, would be, for now, too unrealistic, and would annoy the users more than it would help them.

After proper refinement is done (for example, through the aforementioned collaboration with machine learning researchers), I think it would be really interesting to explore adding this method to a user-targeted experience.

3.1.2 Technique 2: Hand tilt

The next approach I took was actually inspired by the astronauts aboard the International Space Station (ISS). In order to do locomotion through the zero-G environment, conventional "walking" is unattainable. The method they often use is grabbing onto particular elements of their inner compartment and giving themselves a push or a pull.

I thought this technique could actually be applied to Mobile VR as well. One of the problems which cause motion sickness is the lack of correspondence between real life stationary legs and the virtual moving state of the character. A way to tackle this is to find a manner to incorporate some form of real body movement into the VR world in order to facilitate locomotion.

Similarly to the way astronauts drag themselves around the ISS, I decided to implement a similar method, by finding a way to track the hand gestures of the user. That way, I would get an in-world live representation of the hand orientation and allow the users to use that in order to grab themselves onto a particular world point that they choose. They would then push or pull themselves around the environment, the same way astronauts do, hence creating locomotion.

The first step in this process was finding a way to track the hand gestures and translate them into VR. One important thing to note here is the importance of accessibility: a quick solution would have been acquiring a more expensive controller. However, that is against the purpose of my project, which aims at bringing improvements to mobile VR without losing its widely cost-wise availability.

Hence, I decided to try implement this particular piece of technology with a component which has a high level of spread and/or is really easy to obtain. Therefore, to track the arm movement, all my method generally requires is virtually any device which has an accelerometer.

This particular sensor is inexpensive and can often be found in tiny gadgets sold for even under 4 pounds, for example a fitness bracelet. Another example that does the desired task would be an old phone that's just lying around one's room, gathering dust. As long as the device features an accelerometer, the method that I'm proposing will work.

For my particular demo, I use a secondary phone which I attach to the lower arm of the user. I use two elastic hairbands to fix it in place. Obviously, there are better, more elegant ways to do this, however this is to show the accessibility and flexibility of the tech requirements.

I have included an image of the custom arm controller in figure 3.2.

Figure 3.2: The custom arm controller, attached to the user.

The method implies fetching the values from the controller sensor and shipping them to the host, alias the phone that is running the VR application. This could be done in several ways: via turning the controller into a Bluetooth transmitter, or by establishing a local connection through a Wi-Fi router, etc.

The particular working method I opted for when building my simulations was connecting the two devices via a Photon Network. This network works through a server which mediates the transmission from the controller (which in our case is the primary client), to the VR host device (which in our case is the secondary client).

I have depicted the workflow of the network in the graph in figure 3.3.

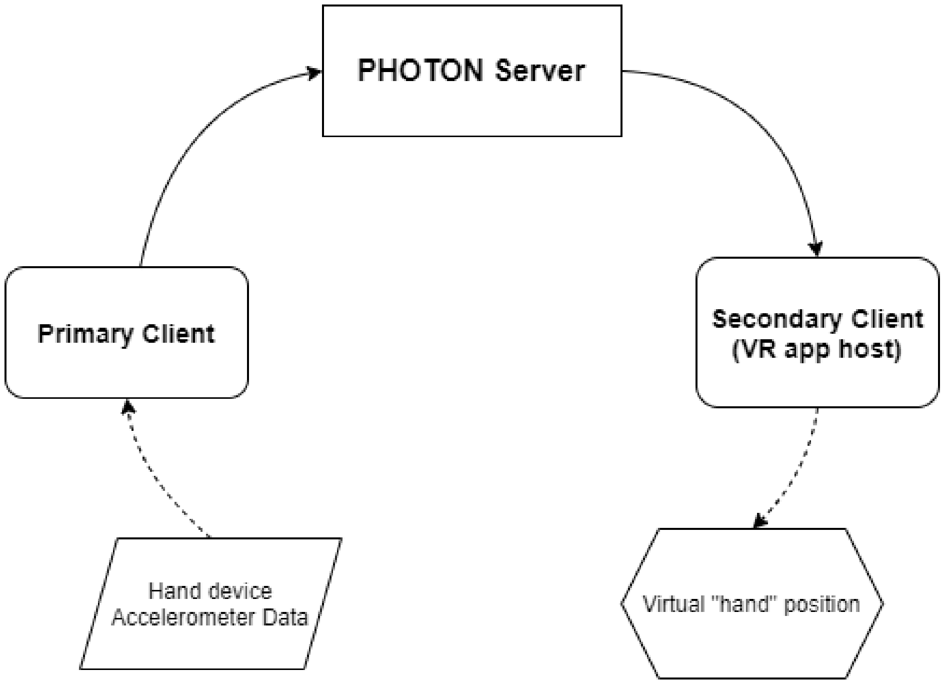


Figure 3.3: A chart featuring the workflow of my Photon Network. The dotted arrows signify local processes while the continuous arrows represent client-server transfers.

The main point of interest when it comes to controller data is the presence of the gravitational acceleration. As long as the data is **not** filtered to be independent of Earth's gravity (similarly to the "Linear acceleration" option of the "Physics Toolbox" app), the value of g will be present amongst the axes. It will be a constant force which acts upon the device and gets spread between the axes depending on the phone orientation. This occurrence has been described in Chapter 2, in figure 2.3.

Hence, when looking at, for example, the vertical axis, this value can be used to determine whether the device is being held upward or downward.

If the controller is attached to the arm of the user, its orientation will be directly dependent on the arm movement, therefore the value recorded by the accelerometer will reflect that. By only shipping one float variable through the Photon network, the controller device can feed the host exact information about the arm movement, which is then interpreted and transposed into the virtual world.

Hence, I obtain an inexpensive and consistent method of adding immersion in mobile VR, by tracking the real life hand movement.

In order to show how this would work in practice, I've added an entity to the experience, a non-stationary crosshair. Its position changes depending on the hand movement, hence allowing the user to point towards a certain element of the environment (figure 3.4). We can regard this action as similar to being in a classroom and lighting up a certain place on the blackboard using a laser pointer.

One thing to mention is that, depending on the connection quality or the crowdedness of the server, tiny latency interruptions do occur between physical movements of the hand and their VR crosshair position interpretations. However, I'm handling such eventualities through frame-by-frame interpolation.

With interpolation, I obtain a fluid crosshair movement, which otherwise would have had occasional moments of frozenness or sudden movement.

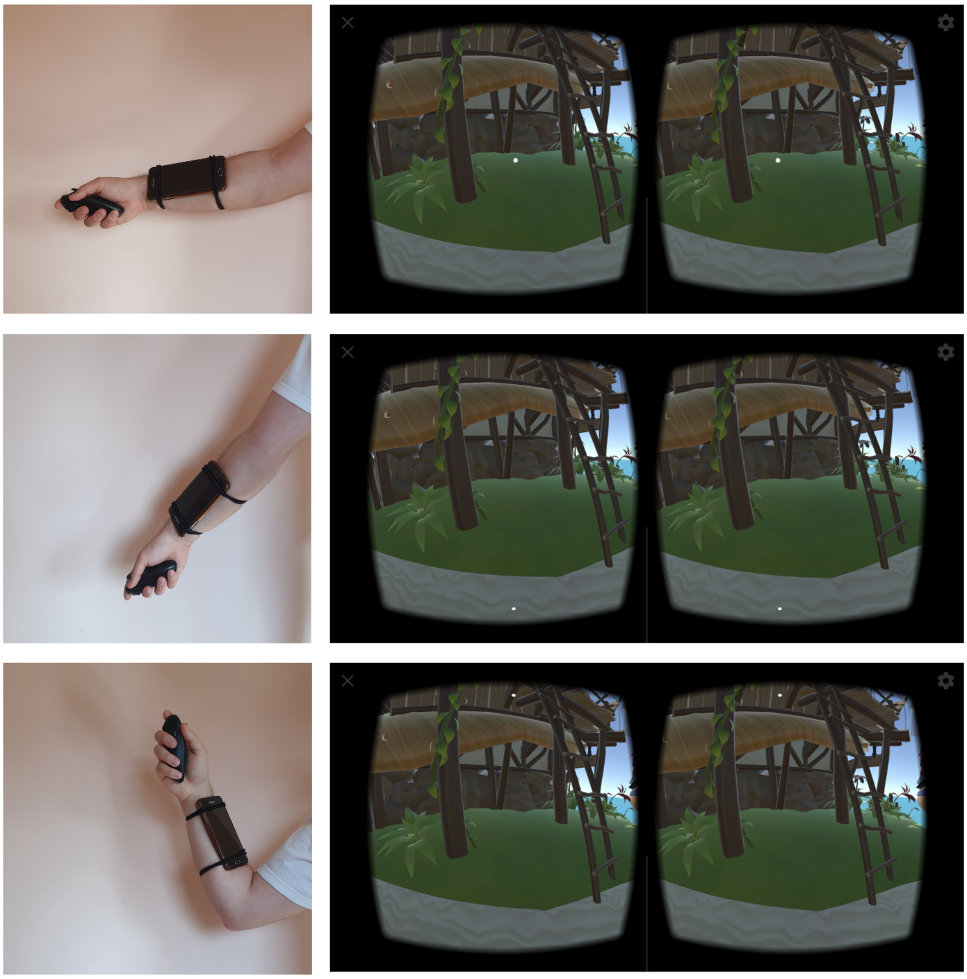


Figure 3.4: The dynamic change of the crosshair position, depending on user arm movement.

$$p_0 = Lerp(p_0, p_1, s) = p_0 + s \cdot (p_1 - p_0) \tag{3.2}$$

Linear interpolation equation, used for smoothened crosshair transitions.
`p0` = last known position
`p1` = last received target position
`s` = smoothing step

In order to facilitate the grab action, I made use of the cheap Bluetooth joystick device mentioned in the previous chapter. The presence of the front buttons allows for a natural hand gesture; when pressed, the hand shapes itself somewhat similarly to how it would when grabbing an actual physical object in real life.

Hence, the grab action sequence works like this:

- step 1: the user points towards a particular point in the environment which they choose as destination (this is done using the information received from the hand tracker attached to their arm);
- step 2: they then trigger the grab action with the Bluetooth device;
- step 3: they push and/or pull themselves around the environment;

- step 4: once the destination is reached, they release the grab;
- step 5: if they wish to continue their movement, they start the process again from step 1 onward

The subsequent explanation describes the actual process behind the grab/push/pull motion within step 3:

This process, at its core, involves one main element: **raycasting**.

Raycasting is used in order to emit a ray between the crosshair pointer and the attempted destination. It then sends back details about the location, such as if the distance is sufficiently short to do the motion.

Once the grab trigger is initiated, new rays are consistently casted from the crosshair position. Hence, when arm movement is done, the aforementioned position changes, hence the virtual world point that the user is targeting changes. In order to compensate for this, the virtual character position counteracts through movement. I have illustrated the mechanics of this in figure 3.6.

I have also included a code snippet from the `C#` movement script in figure 3.5. The segment specifically showcases the act of raycast-based drag/push/pull movement.

```
// checking if any drag has been initiated
if(isDragging == false){
    if(Input.GetKey(KeyCode.JoystickButton4)){
        // initiate move
        dotCoords = getDot();
        ray = Cam.ScreenPointToRay(dotCoords);

        // RaycastHit hit;
        if(Physics.Raycast(ray, out hit)) {
            isDragging = true;
            currentDestination = hit.point;
        }
    }
}
// doing movement
if(isDragging == true){
    // key still needs to be pressed
    if(Input.GetKey(KeyCode.JoystickButton4)){
        // movement code
        dotCoords = getDot();
        ray = Cam.ScreenPointToRay(dotCoords);

        // RaycastHit hit;
        Vector3 currentHit;
        if(Physics.Raycast(ray, out hit)) {

            currentHit = hit.point;

            Vector3 diff = currentDestination - currentHit;
            diff.y = 0.0f;

            if(red.activeSelf == false)
                transform.position += diff/5;
        }
    }
    else {
        isDragging = false;
    }
}
```

Figure 3.5: This snippet of code features the drag/raycast workflow.
The manual drag trigger is identified, which emits a ray towards the direction to which the user points. The ray sends back the targeted point in VR environment coordinates.
While the trigger is maintained, the motion status is retained. Continuous rays are emanated, which signal any change in the targeted location.
When the user repositions the arm, the pointed at coordinates shift and the difference vector is calculated.
The y factor is manually removed, in order to avoid an incorrect jumping motion.
The user position is then shifted, to simulate the push/pull maneuver.
At the moment of trigger release, movement stops and the flow of casted rays terminates.

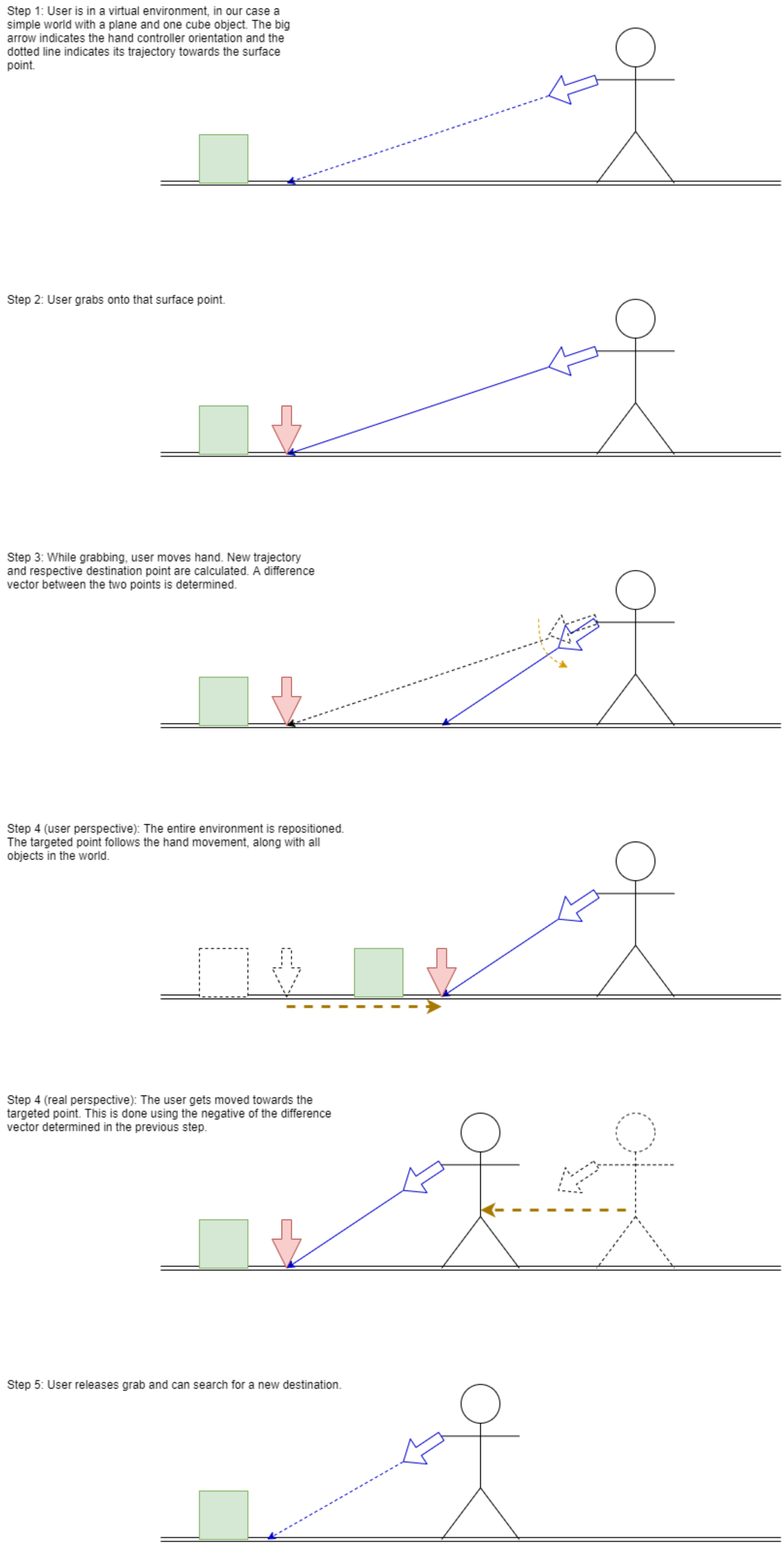


Figure 3.6: The workflow of the grab/push/pull technique.

3.1.3 Technique 3: Head tilt

The third method I attempted was meant to be a simplification of Method 2. The intended purpose of it was to have a similar grab + push/pull locomotion mechanism, but without the need for any additional accelerometer-based controllers.

This method still adheres to the principle of importing real body movements to the VR experience, only this time, the possibility of having the hand do that is denied. Hence, the control of the motion is done exclusively through head tilts.

The head tilts act correspondingly to what the hand controller would have done. This time, the crosshair is fixed to the centre of the screen and the user points towards the desired destination by looking at it directly.

The grab action and pushing/pulling through the environment is then effectuated exactly as described in the 4 steps of Method 2 (3.1.2), the only difference being that, this time, the motion is not done through hand tilts but rather with head tilts.

The grab action is also done through the cheap Bluetooth device, however it can be adapted so that even that particular piece of equipment is not needed. The grab action can be transferred to a screen tap, which can be performed using the physical button present in many phone VR headsets, including cheap cardboard ones.

Hence, this method removes any need for additional gear: all it takes is the VR headset and the mobile device.

Despite the accessibility of this method, it has downsides that can make the experience harsh. Excessive head movement, caused by using the locomotion technique, can result in heavy secondary effects. By self-trials, this has shown to generate symptoms like dizziness at an exponentially higher rate than all other locomotion methods I've implemented.

Hence, the resulting experience tends to go against the purpose of my project, which is that of reducing secondary effects as much as possible and increasing the quality of mobile VR experiences. Therefore, I decided not to include this particular method in any volunteer-oriented simulations.

3.1.4 Technique 4: Field-of-View Reduction

The following technique is not as much of a means of locomotion itself, but rather, an addition to movement that is applied in parallel.

It has been shown [8] that, during a head-mounted simulation such as the one I'm proposing, the peripheral view degree impacts both the user immersion as well as sickness symptoms during and after the experience.

There is a sensitive balance which often endorses one of the aspects in favour of the other. On one side, the more the peripheral vision is restricted, the less likely the user is to feature symptoms related to motion sickness. However, the downside to this is that, if the field of view gets diminished, this significantly takes away from the overall experience and immersion. The overall VR quality is at risk of getting devaluated.

What I thought about doing in order to tackle this issue was finding a way to create some form of perspective which satisfies both requirements: motion sturdiness and good user immersion.

I decided to implement a Field-of-View restrictor which blurs out the user's peripheral view whenever motion occurs, but which is inactive when the user stands still.

This would, in turn, bring up another question, which is that of the suddenness of the transition. My purpose is to try make the VR experience as natural as possible. Under these circumstances, suddenly taking away a whole part of the user's sight while they move seems counter-intuitive.

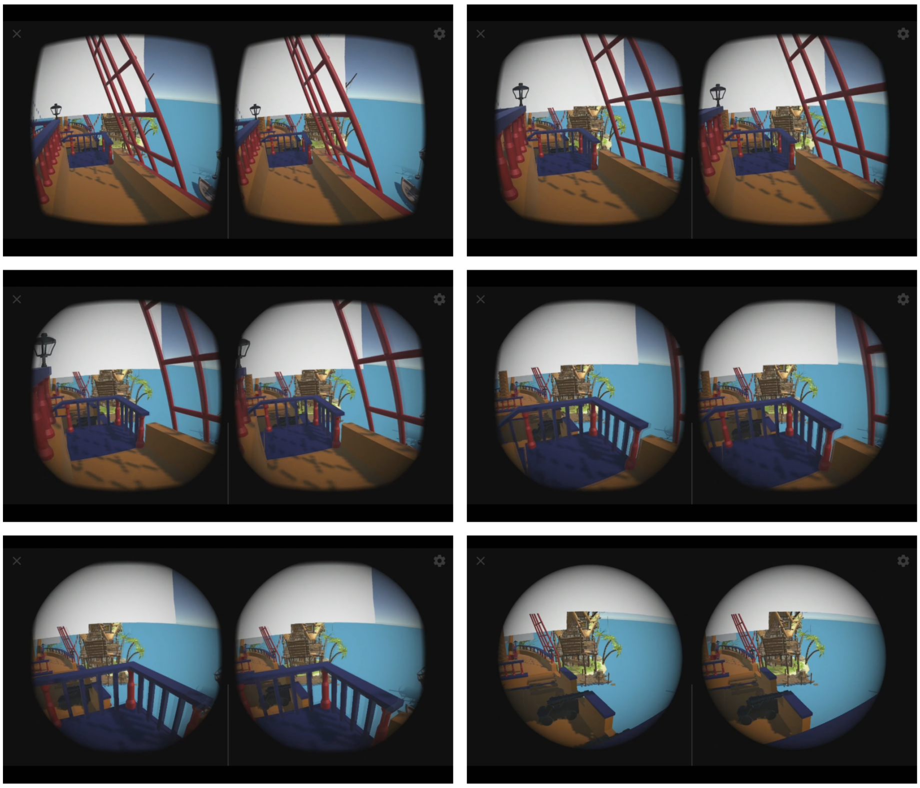
The way I eventually tackled this was by making the FOV restrictor not blur the side vision through a sudden spawn, but rather gradually. This means that, in a fully stationary state, the user experiences natural, full vision. Looking around is as fluid and non-restricted as normally. However, the change occurs when motion is initiated, regardless of the form in which that happens. This means that, at the moment of motion detection, the FOV restrictor mechanism begins to apply itself gradually. This way, the more a user moves, the less peripheral view they are left with. This means that the PV is shrunk exactly when it's needed the least – during motion, because of the risk of inflicting more motion sickness onto the user. The reverse process happens when the user halts the motion and the character becomes stationary again: the FOV restrictor begins to decrease, also gradually, eventually leading to a full peripheral vision again.

Whenever I add an element to a simulation, I tend to have regard to finding real world equivalents to those experiences, similarly to the astronauts in ISS example in Technique 2 (3.1.2). In this case, the real life example that I thought of is hiking on a mountain. For example, while traversing a more difficult path, the climber tends to focus on doing that particular track successfully. It is only when they stop and rest - become stationary - that they actually look around, to admire the beautiful view around themselves. This is the exact principle behind my FOV restrictor mechanism.

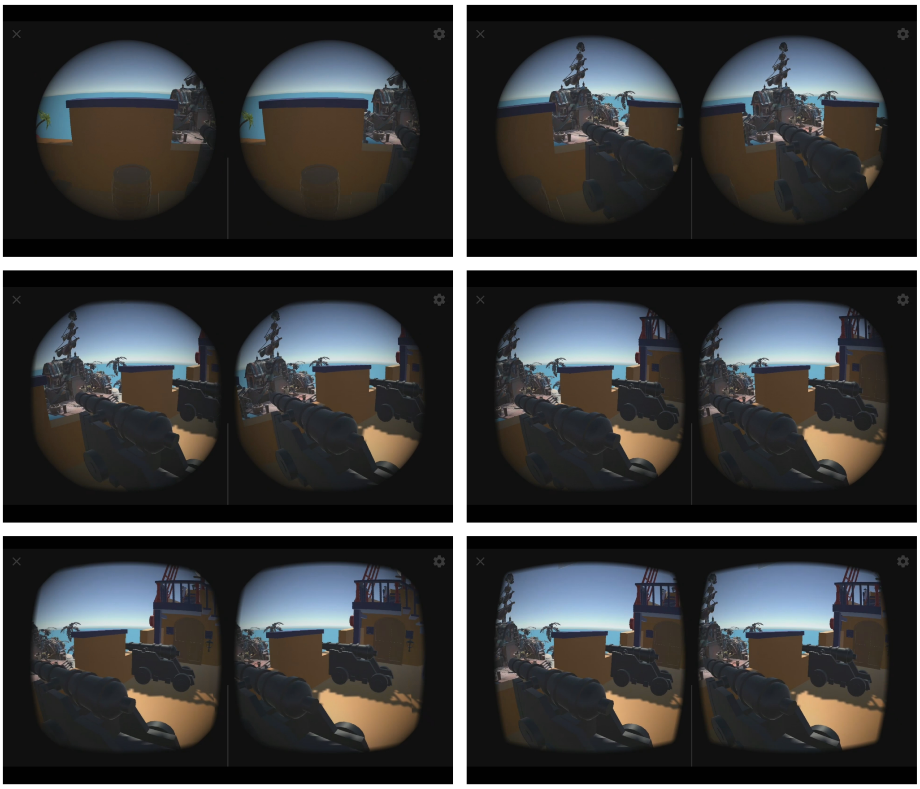
In order to implement this restrictor, my first attempt was using a partially transparent image element on the main canvas, in front of the camera. The element would shrink in size gradually, narrowing the user's vision.

However, I found a method that allows for better flexibility and more reliance. The resource I used is a downloadable plugin for Unity Graphics, the `Post-Processing` effects. More specifically, the effect that I made use of was the `Vignette`. The ability to automatize altering its characteristics, such as shape, margin blur, intensity, etc. was crucial in developing a suitable FOV restrictor.

I have added these images in figure 3.7, which depict the FOV restrictor extending and contracting during motion, respectively immobilization.



(a) FOV restrictor extending during movement



(b) FOV restrictor contracting during stillness

Figure 3.7: The dynamic extension and contraction of the FOV restrictor, caused by the presence, respectively absence of motion.

3.2 The Experiment

One big part of this project consists of exploration of aspects of the issue, researching possibilities of tackling it and developing a real functioning app which implements them.

However, the other, equally important section of my study is actually determining the degree of efficiency and improvement which my findings actually bring.

In my project's case, in order to draw conclusions about my techniques, I had to integrate them all into simulations. These would then be tested sequentially by real users and data about them would be collected.

3.2.1 What is being tested for

In the last sections, I ran through all explored techniques and picked those which were to be included in peer-intended experimental evaluations. I was looking at the following aspects:

- The real motion grab/push/pull movement.

  This one had to be put into balance with another, more conventional means of locomotion. The more popular methods, commonly found in VR apps, are: teleportation, head tilt movement and touchpad/joystick based movement. Out of all of these, the latter is the more similar one to my own method. On one hand, it features some form of interaction using the hand. Secondly, it provides continuous movement to a certain place, unlike teleportation, which would simply spawn the user in the desired location. Teleportation would also be undesirable in the next aspect, due to a lack of continuous movement. Therefore, a comparison experience had to be made between my grab/push/pull locomotor and joystick-based movement.
- The Field-of-View Restrictor

  When it comes to this particular element, the key is that it is not entirely dependent on the locomotion type. It would not really work with methods such as teleportation, however, because the entire functionality behind it requires for the movement to be continuous. This means that, as long as the locomotion type is continuous, the FOV restrictor can be attached to the user perspective.

This means that I needed to build simulations which feature comparisons of the following:

- two separate means of locomotion, one conventional (joystick) and one custom (grab/push/pull)
- the presence or absence of the custom FOV restrictor

A 2x2 combination of these features results in 4 distinct simulations, which, for the purpose of this paper, will now be referred to as A, B, C and D, as seen in figure 3.8.

	Joystick Locomotion	Grab/Push/Pull Locomotion
FOV restrictor = OFF	A	C
FOV restrictor = ON	B	D

Figure 3.8: The simulations

3.2.2 Who the participants are

Selecting volunteer participants was by no means easy, not only because of the ongoing sanitary risks, but also because of the willingness to run as much accurate of an experiment as possible.

In order to obtain valid results, the number of variables when it comes to partakers had to be reduced as much as possible. The addition of any extra variable (for example, age or VR expertise) would imply the need for increasing the number of participants.

When it comes to the age variable, I opted for selecting familiar people belonging to the age group of which I have the most acquaintances. I am currently 22 and I allowed an offset of +/- 2 or 3 years, hence all participants who have taken part in the experiment have ages which range between 20 and 25.

When it comes to VR expertise, it was challenging because it is tricky to define what expertise actually means. Is it measured in time spent in VR? Is it measured in the type of VR utilised? Or maybe in what kind of games/simulations the user took part in.

For the purpose of this project, I initially intended to consider any little form of VR expertise as labelling the respective candidate as experienced. Therefore, I would simply have to label everyone as either experienced or non-experienced and only pick one group to work with.

However, out of all enthusiasts whom I had at my disposal, there were essentially two minorities and one majority:

- one minority of people with zero VR experience
- one minority of people with considerable VR experience and home-owned professional VR headsets
- a majority of people with some very faint VR experience, ranging from a few minutes to around maximum 2 hours in their lifetime

Because of the ongoing circumstances, in order to reach a decent number of participants, I decided to work with those who either had zero experience or very faint experience with VR. It is highly unlikely that a two-hour-long lifetime VR experience is enough for the “VR legs” to form. Certain forums and articles[1] have shown people accounting a need of several weeks to even months of VR experience in order to obtain the “VR legs”.

This means that, despite some exposure to this type of simulations, respective users would much more likely be similar in attitude, response and side-effects to the non-experienced, rather than a trained VR user, who uses this technology on a regular basis.

After considering this criteria, the number of participants who fit my selection and who actually turned up to the experiment is 13. Due to availability issues, despite initially inclining to do all of the testing in one day, I had to split the participants into two groups: one of 5 and the other of 8.

3.2.3 The experiment itself

For each individual simulation, there were two things that I was interested in: the user ability to adapt to the method of locomotion and the side effects that the experience causes. In order to solve the latter, I recorded data using the Simulator `Sickness Questionnaire (SSQ).` [19]

The questionnaire features a list of symptoms which can each be rated as: ”None”, ”Slight”, ”Moderate” or ”Severe”. After each individual simulation, the volunteer would complete a form by matching each symptom to the degree to which it has affected them. The symptoms I’ve enlisted on the form represent a union of options found in two distinct variants of the SSQ: Variant 1[5], Variant 2[10].

I added an ”other” section, in case a user experienced a condition not present in the table. The final list of symptoms in my form is represented in figure 3.9.

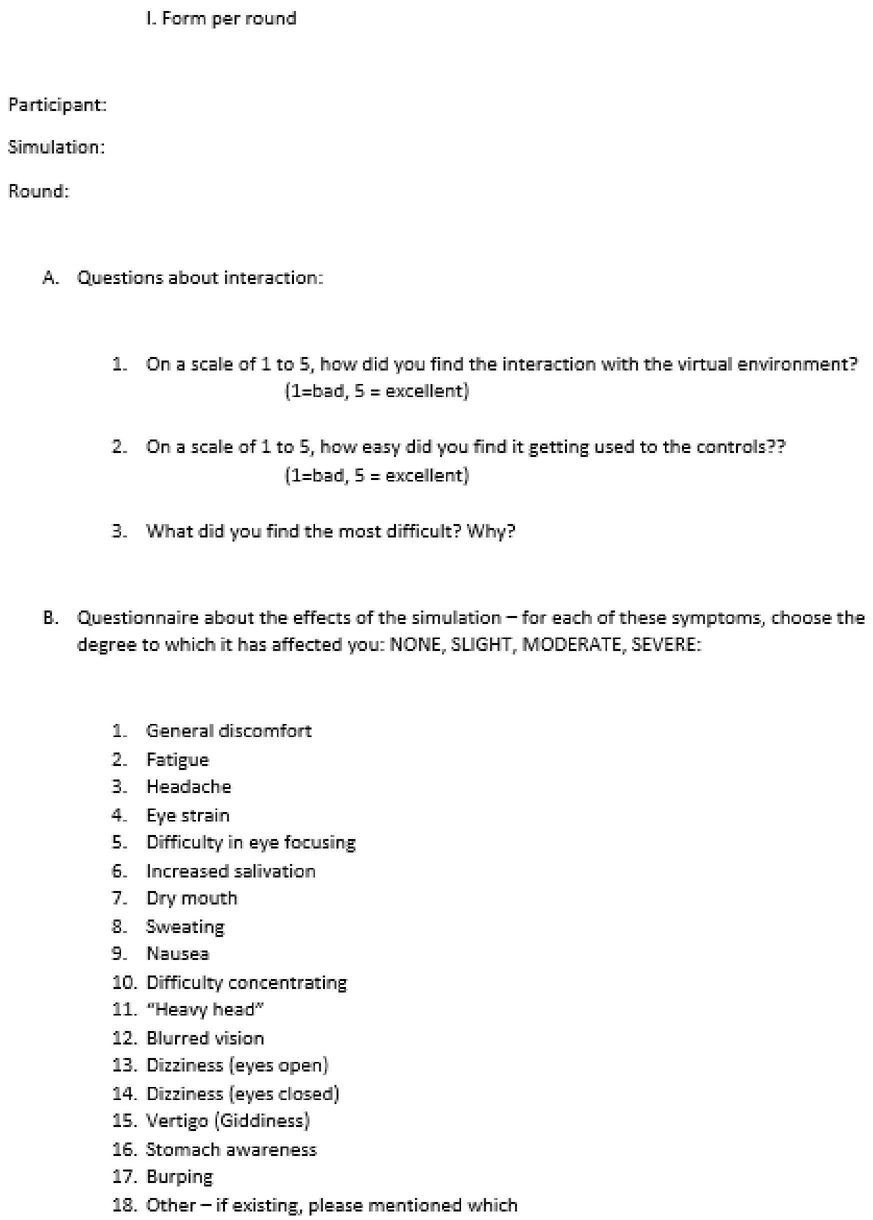
General discomfort	None	Slight	Moderate	Severe
Fatigue	None	Slight	Moderate	Severe
Headache	None	Slight	Moderate	Severe
Eye strain	None	Slight	Moderate	Severe
Difficulty in eye focusing	None	Slight	Moderate	Severe
Increased salivation	None	Slight	Moderate	Severe
Dry mouth	None	Slight	Moderate	Severe
Sweating	None	Slight	Moderate	Severe
Nausea	None	Slight	Moderate	Severe
Difficulty concentrating	None	Slight	Moderate	Severe
"Heavy head"	None	Slight	Moderate	Severe
Blurred vision	None	Slight	Moderate	Severe
Dizziness (eyes open)	None	Slight	Moderate	Severe
Dizziness (eyes closed)	None	Slight	Moderate	Severe
Vertigo (Giddiness)	None	Slight	Moderate	Severe
Stomach awareness	None	Slight	Moderate	Severe
Burping	None	Slight	Moderate	Severe
Other	None	Slight	Moderate	Severe

Figure 3.9: The list of symptoms in my SSQ form

The SSQ was only a section of the forms, because I also included interview-style questions and ratings. The forms are shown in figure 3.10, respectively figure 3.11.

On the first form (figure 3.10), I've asked the users to rate, from 1 to 5, the interaction with the virtual environment, respectively how much easiness they had with getting used to the controls. I also asked them to write about which part they found to be the most challenging and why.

After a volunteer finished all 4 simulations, I asked them to complete a more conversational form (figure 3.11) about their overall experience. I ask them which their favourite and least favourite experience was overall, as well as which the most/least immersive one it was. I also queried about each of the two assets (the grab/push/pull controller and the FOV restrictor): I asked how big of a factor they felt it was for their experience and if they think it would be a good idea to encounter the respective asset in future VR experiences. I also allowed them to add additional comments and suggestions.



I. Form per round

Participant:

Simulation:

Round:

A. Questions about interaction:

1. On a scale of 1 to 5, how did you find the interaction with the virtual environment?
(1=bad, 5 = excellent)

2. On a scale of 1 to 5, how easy did you find it getting used to the controls??
(1=bad, 5 = excellent)

3. What did you find the most difficult? Why?

B. Questionnaire about the effects of the simulation – for each of these symptoms, choose the degree to which it has affected you: NONE, SLIGHT, MODERATE, SEVERE:

1. General discomfort
2. Fatigue
3. Headache
4. Eye strain
5. Difficulty in eye focusing
6. Increased salivation
7. Dry mouth
8. Sweating
9. Nausea
10. Difficulty concentrating
11. "Heavy head"
12. Blurred vision
13. Dizziness (eyes open)
14. Dizziness (eyes closed)
15. Vertigo (Giddiness)
16. Stomach awareness
17. Burping
18. Other – if existing, please mentioned which

Figure 3.10: Simulation form to be completed per round. In total, 4 of these were filled per user.

II. Overall questionnaire

Participant:

1. Which of the 4 experiences did you find the most comfortable? Why?
2. Which of the 4 experiences did you find the lease comfortable? Why?
3. Which of the 4 experiences did you find the most immersive? Why?
4. Which of the 4 experiences did you find the least immersive? Why?
5. Do you think that the Field-of-View restrictor was an important factor in experience immersion?
6. In the future, do you think it would be a good idea for experiences to feature a Field-of-View restrictor?
7. Do you think that control through natural hand movement was an important factor in experience immersion?
8. In the future, do you think it would be a good idea for experiences to feature control through natural hand movement?
9. Would you add/change anything to the mechanisms presented in the simulations?
10. Are there any other observations you would like to add?

Figure 3.11: Final, "conversational" form to be completed after all rounds were finalised. One of these was filled per user.

One important issue when running an experiment with multiple simulations per user is the learning factor. The fact that, after each round, the user acquires more adaptation to the environment, means that a way had to be found to even this out. Hence, I set the participants do the simulations in different orders, using a variant of the Latin square method [20]. This way, I achieved counterbalancing by each participant having the 4 simulations in a unique order.

Hence, I distributed the different simulations amongst the participants according to a table, depicted in figure 3.12.

Participant/Simulation number	1	2	3	4
1	A	B	C	D
2	B	C	D	A
3	C	D	A	B
4	D	A	B	C
5	A	C	B	D
6	C	B	D	A
7	B	D	A	C
8	D	A	C	B
9	C	B	A	D
10	A	D	B	C
11	D	B	C	A
12	B	C	A	D
13	C	A	D	B

Figure 3.12: The simulation order sequences

Any participant spends 6 minutes in one simulation. After the simulation ends, the next user comes and so forth, until an entire column in the table above is traversed. This gives each participant time between their own rounds so that the previous symptoms fade away. Popular VR headset makers claim [9] that 10-15 minute breaks every 30 minutes of VR usage should be enough. Scaled to my experiment's simulation duration, that would mean 2-3 minutes of break time per round. Considering that, in each day of the experiment, no less than 4 other people would do one simulation each until a particular user's turn comes again, this leaves an inter-turn gap of at least 24 minutes. According to the claim [9], this number is more than enough to thin out previous symptoms and have the user ready for the next simulation.

The simulations themselves were similar overall, with the exception of the method of locomotion and/or the peripheral view handler.

On a general level, a simulation would begin in a friendly pirate-themed environment (figure 3.13), where the user gets the chance to get used to the controls and interacting to the world. The objects in this particular set were **not** sketched by myself, but were adapted from models by Yano Claeys[6], Alena Shek[22] and Lazaroe[16].

Once the user familiarises themselves with the technology at hand, they can go through a portal which sends them to a different environment. This time, they have to make use of the acquired movements in order to traverse a parkour-style obstacle course (figure 3.14). Successfully completing the course finalises in another portal which sends them back to the starting point in the pirate world. The simulation then ends.

I have analysed the data acquired from the experiment in the following chapter.

Figure 3.13: The pirate-themed world, where users get to familiarize themselves with the environment and the controls.

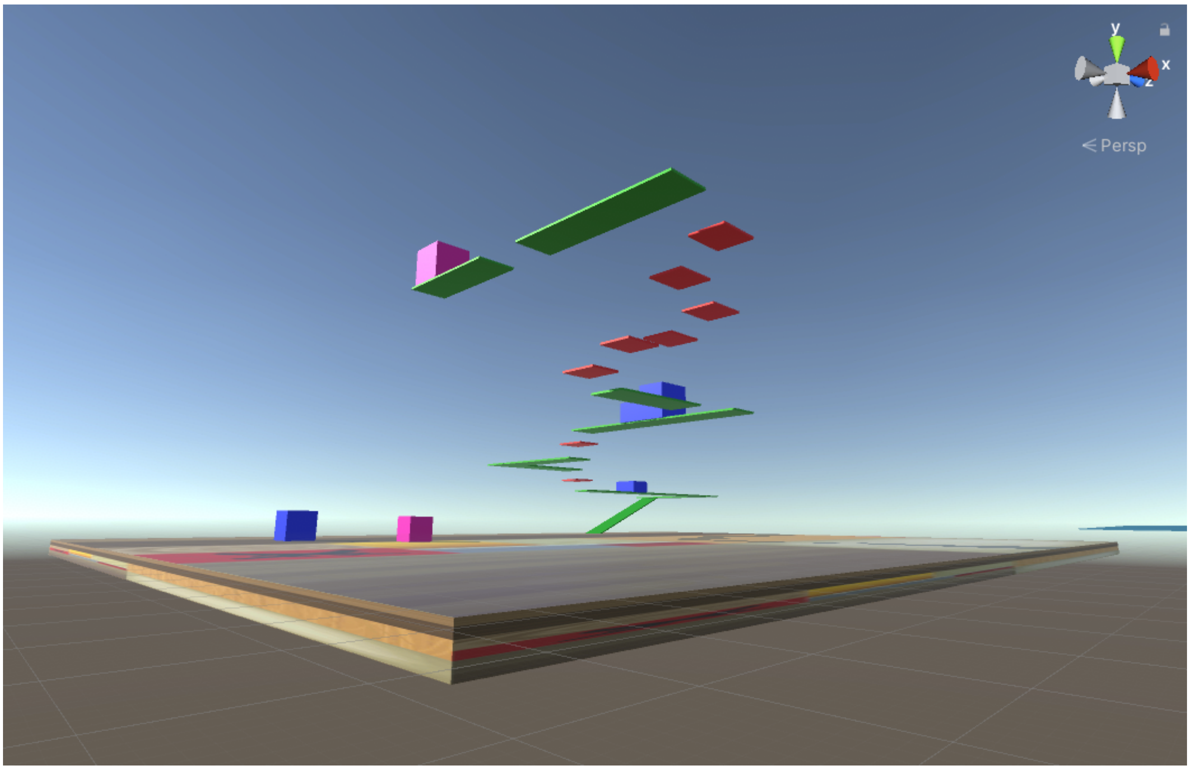


Figure 3.14: The parkour-style obstacle course.

Chapter 4

Critical Evaluation

In this chapter, I would like to analyse the outcome of my experiment. I will be determining which assumptions about the efficiency of my techniques were correct and which not. I will also be establishing, based on user feedback, which aspects require additional improvement and why.

4.1 Overview

- METHOD A: Common joystick movement, no FOV restrictor

  The interactive experience: On a scale of 1 to 5, the participants rated it, on average, to 4.07

  The controls: On a scale of 1 to 5 the participants rated it, on average, to 4.69

  After the 6 minute simulation, 6 out of 13 participants experienced at least some form of secondary effect.

- METHOD B: Common joystick movement, with FOV restrictor

  The interactive experience: On a scale of 1 to 5, the participants rated it, on average, to 4.46

  The controls: On a scale of 1 to 5 the participants rated it, on average, to 4.76

  After the 6 minute simulation, 5 out of 13 participants experienced at least some form of secondary effect.

- METHOD C: Grab/push/pull locomotion, no FOV restrictor

  The interactive experience: On a scale of 1 to 5, the participants rated it, on average, to 3.46

  The controls: On a scale of 1 to 5 the participants rated it, on average, to 3.3

  After the 6 minute simulation, 4 out of 13 participants experienced at least some form of secondary effect.

- METHOD D: Grab/push/pull locomotion, with FOV restrictor

  The interactive experience: On a scale of 1 to 5, the participants rated it, on average, to 3.92

  The controls: On a scale of 1 to 5 the participants rated it, on average, to 3.69

  After the 6 minute simulation, 5 out of 13 participants experienced at least some form of secondary effect.

In figure 4.1, have compiled a chart using the aforementioned data, for better visualisation.
The average number of symptoms, regardless of type, over all 4 methods, is:
22.25 for "slight", 3.5 for "moderate" and 0.75 for "severe", as can be seen in the pie chart in figure 4.2.

This is not yet an in-depth analysis of the particularities of, for example, the individual symptoms, but more of a general overview.
The first thing which can be detected here is that, in this 13-user sample, the presence of a FOV restrictor tended to even out the percentage of people who experienced side effects.
Another thing that can be noticed is that, when it comes to the different locomotion mechanisms, the users were way happier about the functionalities of the joystick-based simulations rather than the

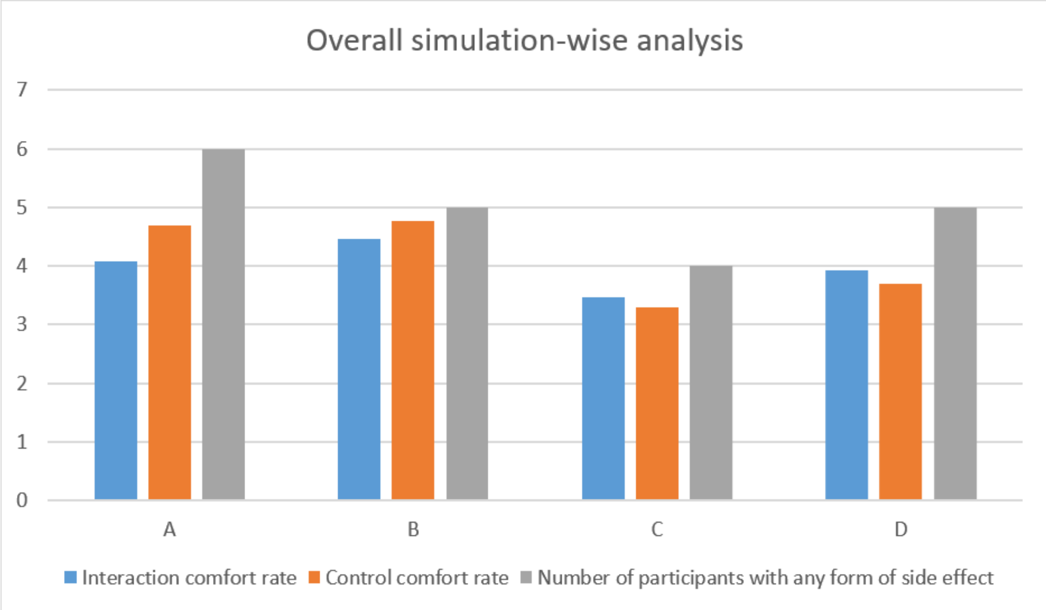


Figure 4.1: Overall simulation-wise analysis

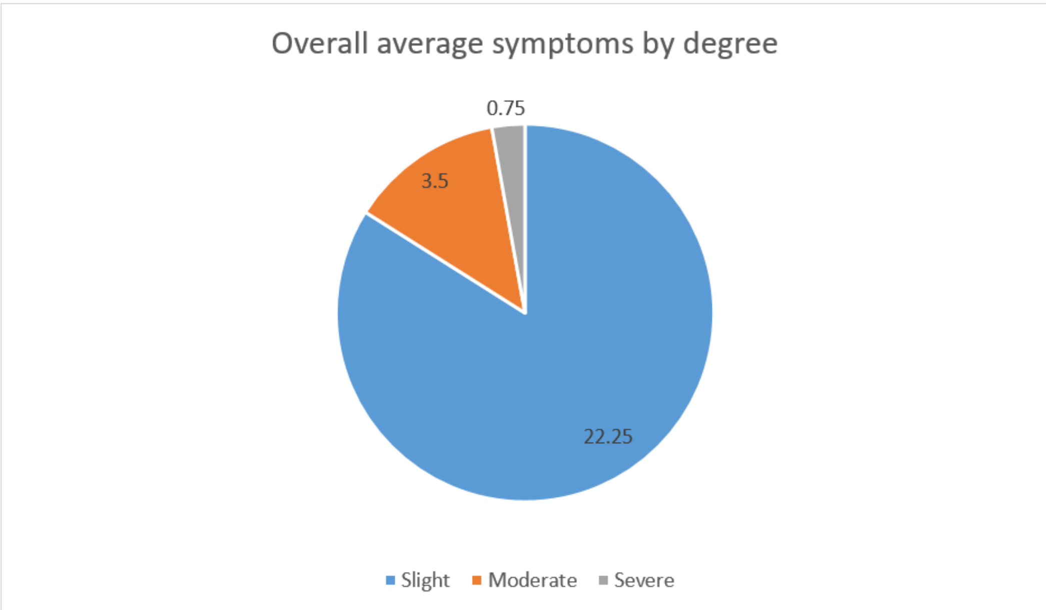


Figure 4.2: Overall average number of symptoms

grab/push/pull locomotion. This was an expected outcome as, during the experiment, I had noticed the users struggle to understand how the controller actually works in the first place. Regular joystick movement, on the other hand, which is considerably more commonly found, did not pose such an impactful challenge on the users.

4.2 User interviews

Before analysing the individual symptoms in depth, I would like to take note of some of the highlights the users have written in the final forms. That is, the more conversational forms which the users filled after having completed all 4 simulations:

> *"The grab-based control should rather be used for movements which simulate, for example, climbing, or throwing a lasso."*

> *"A combination between the two types of movement would help."*
>
> *"The coordination in grab-based movement should be smoother."*
>
> *"I could make use of left-right grab-based movement too."*
>
> *"The grab-control should be optimised for a more real time movement."*
>
> *"I would love to use the joystick to move and the grab-based control to grab and drop objects instead."*
>
> *"The-grab based control could be a good idea, depending on the experience."*

Even just throughout the few quotes I mentioned, patterns can already be depicted. There is preference over certain ideas, which are commonly found when the entire scope of information provided is regarded.

A thematic analysis of this can lead to clusters of ideas, which can help assemble and consolidate the information.

I have done this filtering and compiled user feedback and suggestions into an Affinity Diagram, which is shown in figure 4.3.

The user suggestions were initially laid down at random, after which I grouped the similar ones into clusters, eventually leading to 5 distinct categories: "FOV restrictor", "Purpose of grab controller", "Functionality of grab controller", "Adapting to grab controller" and "Other". The latter only contains one item, which is not related to my additions to the mobile VR experience specifically, but refers to user real-life full-body rotation. (figure 4.3)

The initial highlighted statements, combined with the outcome of the Affinity Diagram, show a confirmation of my assumption during the experiment: the participants were not as happy with the grab/push/pull locomotion as with the joystick.

Despite the fact that, on average, less people felt side effects whilst using the former, the latter was simply easier to assimilate in a short amount of time.

This, however, shows that there is potential for better results, provided more training with the new locomotion technique is done.

I should also note that finding an inexpensive way to add 2-axis grab-control would help. So far, this can be done provided the controller device has a gyroscope. However, compared to an accelerometer, a gyroscope is much less commonly found and also pricier.

Even if these two factors get evened out, it would still be challenging to add horizontal movement because of a problem mentioned in the previous chapter. The problem is related to the fact that the device's sensors would only know the force values applied to them, not their origin or purpose.

Hence, to the device, a stationary user spinning their hand would be indistinguishable from a spinning user with a stationary hand. In the absence of a movement tracker, many tweaks would then be required to segregate the two individual cases. A good machine learning classifier algorithm could potentially help with this issue.

Another important observation concluded from the statements, which confirms an initial assumption of mine, is related to the coordination between real hand movement and virtual movement. Since the connection between the controller device and the host device is done via a server, there is always a factor of risk when it comes to latency.

I think this can be fixed by researching a method to turn the controller device into a Bluetooth emitter. An alternative to this would be a direct router link between the two devices. This way, the information could be picked up by the host in more real time, improving the VR environment response to the real life hand gestures.

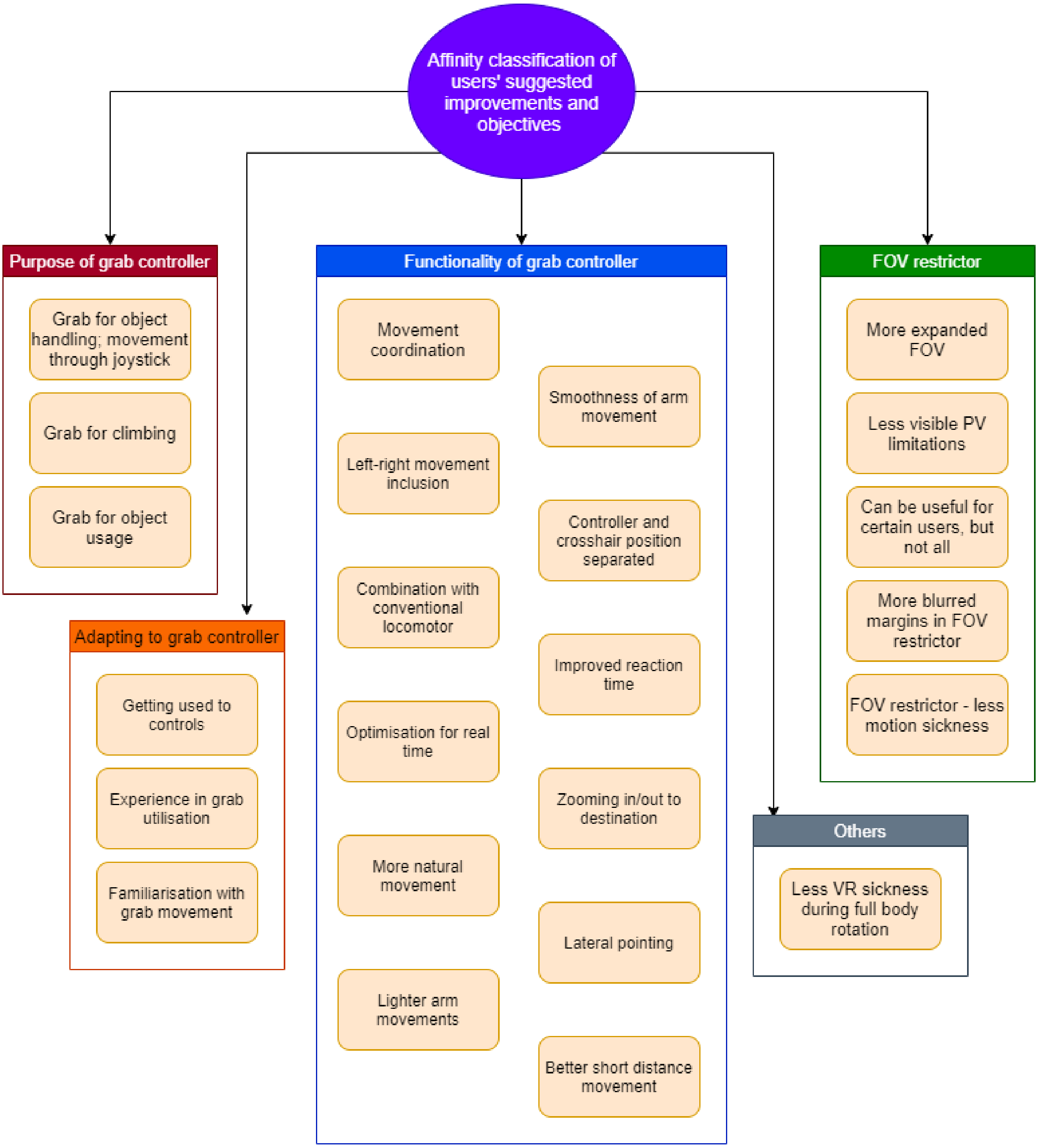


Figure 4.3: Affinity diagram, compiled based on participants' suggestions, noted in the forms.

4.3 In-Depth Symptom Analysis

I will now make a more in-depth analysis of the actual symptoms experienced by the volunteers.

The first thing that I should note is that, out of the list of conditions in the Simulator Sickness Questionnaire, there were 3 manifestations which were not felt by any of the participants at all: increased salivation, sweating and burping. The latter did come as a surprise to me, as I myself did actually feel it slightly when I was testing the application.

I have created distinct tables for each of the 4 simulations, featuring the accumulated results of the SSQ, as well a comparison between them:

- Simulation A: figure 4.4
- Simulation B: figure 4.5
- Simulation C: figure 4.6
- Simulation D: figure 4.7
- Comparison: figure 4.8

	Slight	Moderate	Severe
General discomfort	1	0	0
Fatigue	1	1	0
Headache	1	0	0
Eye strain	1	1	1
Difficulty in eye focusing	1	1	1
Increased salivation	0	0	0
Dry mouth	1	0	0
Sweating	0	0	0
Nausea	3	0	0
Difficulty concentrating	1	1	0
"Heavy head"	2	1	0
Blurred vision	2	0	0
Dizziness (eyes open)	3	0	0
Dizziness (eyes closed)	3	0	0
Vertigo (Giddiness)	2	1	0
Stomach awareness	1	0	0
Burping	0	0	0
Other	0	0	0

Figure 4.4: Simulation A:
This is the simulation which, overall, caused the highest number of "severe" symptoms (2 in total) as well as the highest number of "moderate" symptoms (6 in total).
The number of "slight" symptoms is 23, so it sits roughly above the average of 22.25.

	Slight	Moderate	Severe
General discomfort	3	0	0
Fatigue	1	0	0
Headache	1	0	0
Eye strain	2	1	0
Difficulty in eye focusing	2	1	0
Increased salivation	0	0	0
Dry mouth	1	0	0
Sweating	0	0	0
Nausea	3	0	0
Difficulty concentrating	1	0	0
"Heavy head"	2	1	0
Blurred vision	2	0	0
Dizziness (eyes open)	0	0	0
Dizziness (eyes closed)	3	0	0
Vertigo (Giddiness)	1	0	0
Stomach awareness	1	0	0
Burping	0	0	0
Other	disorientation	0	0

Figure 4.5: Simulation B:
No "severe" symptoms were reported.
When it comes to "moderate" symptoms, this simulation shows improvement from the previous one. Only 3 such occurrences were recorded, which is half of the number in A.
The number of predefined "slight" effects remains the same as in the previous simulation. However, this time, an additional symptom was noted, which was not present on the list: disorientation.

	Slight	Moderate	Severe
General discomfort	3	0	0
Fatigue	3	0	0
Headache	1	0	0
Eye strain	2	2	0
Difficulty in eye focusing	2	1	0
Increased salivation	0	0	0
Dry mouth	1	0	0
Sweating	0	0	0
Nausea	1	0	0
Difficulty concentrating	1	0	0
"Heavy head"	3	1	0
Blurred vision	1	0	0
Dizziness (eyes open)	1	0	0
Dizziness (eyes closed)	4	0	0
Vertigo (Giddiness)	1	0	0
Stomach awareness	1	0	0
Burping	0	0	0
Other	0	eye stinging	0

Figure 4.6: Simulation C:
No "severe" symptoms were reported.
Surprisingly, this particular simulation resulted in the highest number of "slight" symptoms (25) and the second highest in "moderate" symptoms (4).
The initially considered explanation for this was that the method of locomotion itself, despite being intended to increase immersion and interaction, actually impedes them, hence leading to worse symptoms. However, this reasoning is not valid because it is inconsistent with simulation D, which features the exact same method of locomotion, yet very low levels of side effects.
This simulation also led to a symptom not present on the initial list, which is eye stinging.

	Slight	Moderate	Severe
General discomfort	2	0	1
Fatigue	1	0	0
Headache	0	0	0
Eye strain	3	0	0
Difficulty in eye focusing	2	1	0
Increased salivation	0	0	0
Dry mouth	0	0	0
Sweating	0	0	0
Nausea	1	0	0
Difficulty concentrating	1	0	0
"Heavy head"	1	0	0
Blurred vision	2	0	0
Dizziness (eyes open)	0	0	0
Dizziness (eyes closed)	2	0	0
Vertigo (Giddiness)	1	0	0
Stomach awareness	1	0	0
Burping	0	0	0
Other	0	0	0

Figure 4.7: Simulation D:
In contrary to the other grab/push/pull based simulation, C, this one shows the smallest number of symptoms overall: 17 "slight", 1 "moderate" and 1 "severe". I should note here that the person who added the "severe" symptom later mentioned that the cause for it was frustration.

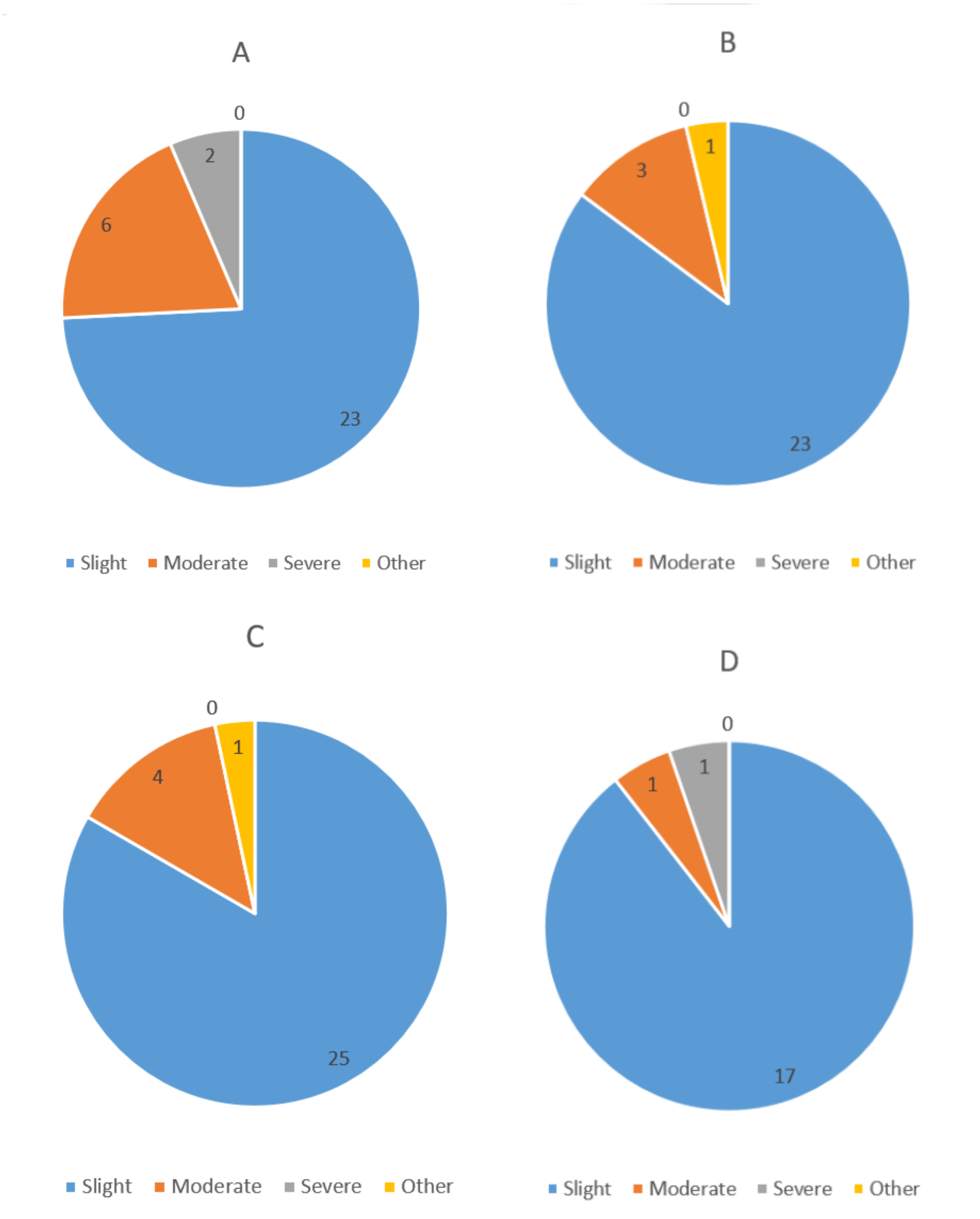


Figure 4.8: Comparison between the four simulations, based on the quantities and the proportions of accumulated symptoms.

The first visible conclusion which can be drawn from this data is that the FOV restrictor does appear to improve the experiences. Both simulations featuring this mechanism lead to decreases in side effects compared to their non-FOV-restrictor-bearing equivalents.

Despite this improvement which the FOV restrictor brings, there is still the issue of how much of the overall experience it takes away. Two people noted, in the conversational forms, that they found the FOV restrictor to be really annoying when combined with the grab/push/pull locomotion (simulation D). However, the other users did not share the same opinion, and 4 of them said they had not even noticed the FOV come into place.

The second thing that can be noted from the data is that there is a huge discrepancy between methods C and D, which are based on the same mechanism for locomotion.

In order to find an answer to this dilemma, a careful analysis needs to be made.

Firstly, the method which performed better overall was actually D, the combination between the grab/push/pull locomotion and the FOV restrictor. It is not unfeasible to deduce that the FOV restrictor did have a positive impact on the experience, helping it perform better than C.

Another alongside factor that actually could have played a part in this discrepancy was only discovered much later, and could be related to the distribution of simulation order amongst users.

Out of 13 participants, 6 experienced at least one of the methods A and B before C (or D). This means that the volunteers might had adjusted themselves to the conventional joystick-based movement before trying the grab-movement. This could have led to an increased level of frustration of going to this novel means of motion all the way from a method which, despite being less immersive, is more commonly encountered in everyday life. This means that almost half of all participants might have been exposed to this risk.

Another crucial factor is the order in which C and D occurred. If we are to focus solely on these two simulations, we can depict that 8 out of 13 users did them in the order C-D and the others viceversa. This means that almost two thirds of participants got exposed to C first, so the levels of frustration felt by certain users could have concentrated onto this particular first encounter of the grab/push/pull. The levels would then alleviate in the second encounter which, in their case, was D. This ascending process of adaptation, combined with the aid from the FOV restrictor, could have contributed to the high level of improvement that method D showcases.

I should also mention that this particular discrepancy is not present in the other 2 simulations: A and B. In their case, I later determined that their particular table orders are roughly evenly distributed. (7 times A-B, 6 times viceversa).

It is also important to note that, despite the simulation C having the highest number of "slight" symptoms, it is also the simulation with the least amount of people who overall experienced any form of secondary effect at all. This means that the volume of negative reactions was concentrated within a smaller group of people, as shown in figure 4.9.

This opens up the possibility of an offsetting alignment of concomitant factors:

- Few people experiencing more symptoms, the majority experiencing no symptoms at all
- Indirect imbalance in simulation order sequence
- User adaptability issue to a new mechanism, which leads to frustration and more intense concentration

These results have pushed me to return to the original forms of the 4 symptomatic participants in C, in order to discover any other potential data. Out of them, only one user was part of the batch whose simulation arrangements contained the subsequence of D-C. The other 3 experienced the opposite.

In percentages, this means 20% of the volunteers from the former succession (5 users), respectively 37.5% from the latter (8 users), experienced side effects. What should also be noted is that the one symptomatic volunteer in the former batch also showed post-VR manifestations when doing the other simulation, which, in his case, was the initial one.

I think this pattern would be really interesting to further explore. It appears that, when taking into account the percentage-wise majority of the sample, careful combinations and successions of training, user-locomotor adaptation and peripheral view manipulation, can produce effective results.

I think that, hereafter, in a follow-up experiment, this aspect should be tackled through 2 simultaneous aims:

1. Focusing solely on the locomotion method at hand, ensuring a bigger sample of participants and levelled out orders of simulation occurrences
2. Training the volunteers to utilise the method **before** the experiment, possibly in a non-VR environment.
   For example, a user could be asked to play a short video game using the grab/push/pull controller; however, rather than the game be part of a VR experience, it would be displayed on a regular computer screen.
   This way, the user would get accustomed to the method itself and gain the reflexes needed, without any exposure to VR's secondary effects.
   The participant would only afterwards be included in the actual VR simulation, with a more consolidated basis for interaction.

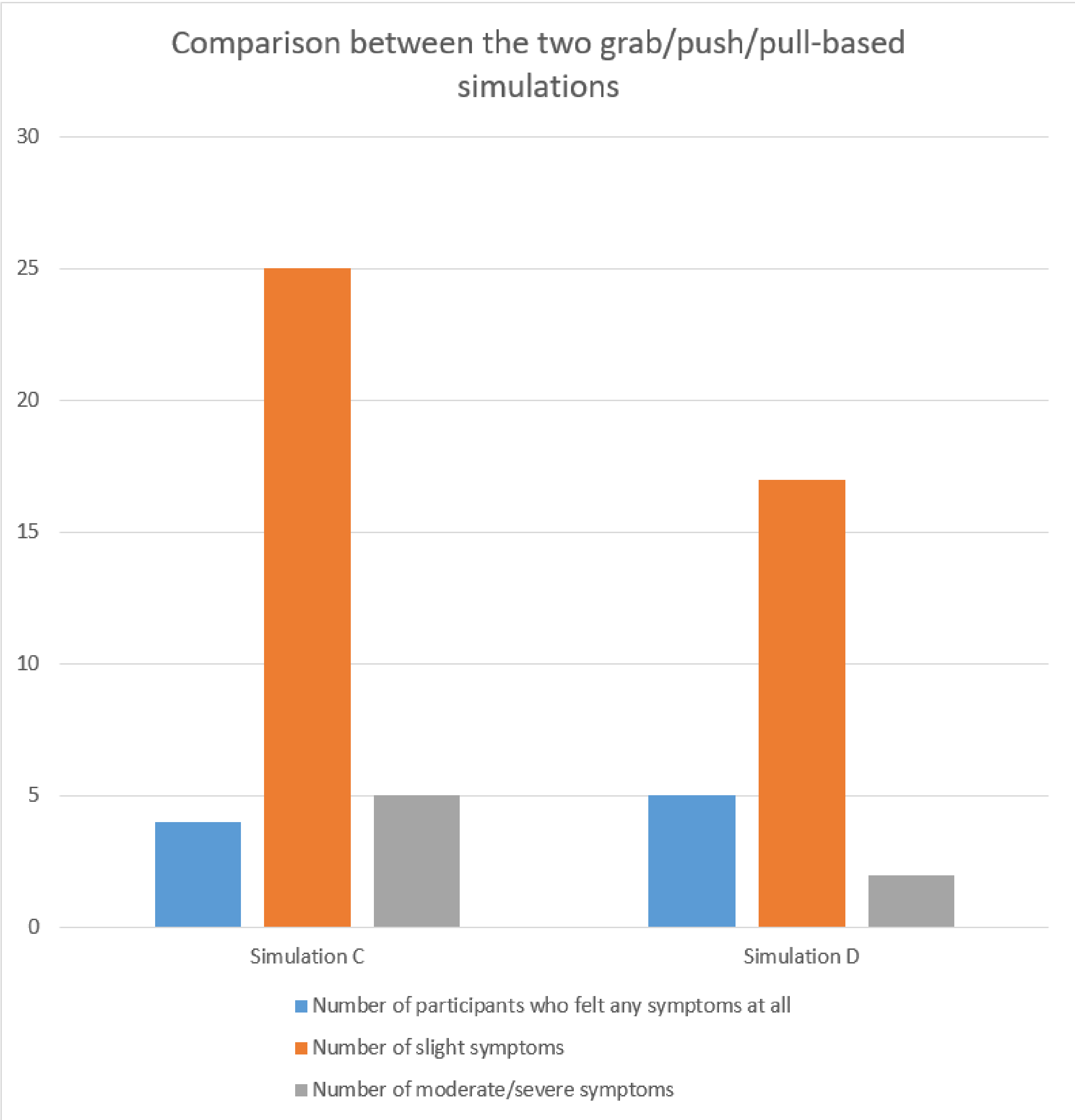


Figure 4.9: Comparison between the two grab/push/pull-based simulations

Chapter 5

Conclusion

5.1 Project Outcome

Throughout the whole duration of my project, there were 2 aims that I always had concomitantly: finding improvements in immersion and sickness for uncostly VR (mobile VR); respectively making sure that the said improvements don't eradicate the accessibility and availability aspect.

The latter aspect was conserved until the finalisation of the project, since I used hardware which is either non-costly or commonly found and reusable.

I have familiarised myself with all the hardware and software needed to create mobile VR applications (Unity, Android, force sensors, networking, etc).

I have made attempts in four different techniques for motion in mobile VR:

1. Real life body movements

   I attempted to incorporate real world motion and translate it into the virtual environment, without the use of any external technologies, such as body trackers, but rather through solely using the device's sensors.

   Because of the device's incapability to distinguish the source movement applied to the sensors (e.g. tilting inducing sensor illusion of linear movement), the resulting motion was not realistically accurate.

   Before passing this technique on to human trials, there is a need for fixes about classifying sensor detections based on their causing movements. This could be done through a collaboration with Machine Learning researchers, with a consistent recorded data set.

2. Movement through real life hand swings

   I created a mechanism which allows the VR user to grab onto points in the VR environment and then push or pull themselves around the world. I did this using two external devices: one Bluetooth remote and an accelerometer-bearing controller device, attached to the user's arm.

   The device functioned as expected and hence was passed on to be part of simulations within my experiment.

3. Movement through head tilts

   This was meant to be an adaptation of method 2, only this time not requiring the use of any external devices.

   The hand movement tracker would be replaced by head tilts, while the rest of the functionalities would mirror those of method 2.

   The technique, however, after medium-term use, would defeat the first purpose of the project itself, which is reducing the factors which cause the symptoms of VR sickness. Extensive use of the head, especially in virtual places requiring quick, narrow moves, actually leads to counter-productive outcomes. Hence, the method was not passed on to be used in the experiment.

4. Continuous movement featuring peripheral view alterations

I have created a mechanism which manipulates the peripheral view, which is a common factor in VR sickness. I have implemented a dynamic Field-of-View restrictor, which acts accordingly to continuous motion.

The method attempts at maintaining the quality and immersion of the virtual experience, through inactivity during stationary states, while simultaneously contributing to the lessening of VR side effects originated during motion. The transitions are done in a gradual manner, in order to reduce the level to which they are noticed by the user.

The technique was passed on to be part of the simulations within my experiment.

I then conducted an experiment with 20-25 year-old participants, in order to measure the effects of my implemented techniques. I forged 4 simulations, which represent 2x2 combinations between: common continuous movement (finger joystick) and custom grab/push/pull movement; respectively the presence or absence of the custom FOV restrictor.

I recorded data about the volunteers' post-VR symptoms using the Simulator Sickness Questionnaire.

I also recorded data about users' easiness of adapting to the techniques and environments, as well as comparisons between the 4 simulations.

When it comes to the FOV restrictor, the majority of users did not feel bothered by it and think it would be a good idea to feature it in future developments and experiments.

When it comes to the grab/push/pull locomotion, the majority of users had difficulties adapting to the mechanism.

The main reason for this is the fact that the method is much less common than, for example, the alternative joystick. Therefore, it requires an additional level of concentration and time to incorporate into their reflexes.

Another factor that could have aided in the assimilation of this method is a more direct connection between the controller device and the host device, hence leading to more accurate real-time representations of users' movements into the virtual world.

When it comes to side effects of VR sickness, both simulations which showcase an active FOV restrictor performed better than their full-peripheral-view counterparts.

There was however a possible anomaly regarding one of the simulations, namely C (grab-movement, non-FOV-restricting). Despite not issuing as many moderate or severe reactions, the number of slight symptoms featured an increase compared to other simulations. I speculate that the cause for this is an imbalance regarding the distribution of user frustration between the two grab/push/pull methods. I think that this opens possibilities of doing future experiments which focus specifically on this particular aspect. The fix would consist of focusing solely on this method whilst removing certain possible factors, such as the order of simulations, or the user frustration. The latter could be solved through training the participants to use the hand grab technique in non-VR conditions, and only afterwards moving them on to the virtual reality experiment.

The method which performed best was the one which combined both the grab/push/pull hand movement as well as the Field-of-View restrictor.

This shows that, with a good peripheral view handler, combined with enough user training and adaptation to the new locomotion method, the resulting mechanism can prove to have a great amount of potential. The method has the ability to surpass the other methods, including ones based on more conventional means of locomotion.

I should also reiterate that all of this was executed in a manner which requires as inexpensive of hardware as possible. This is done in order to maintain the technology's accessibility to larger amounts of people.

I think there could be good use of future exploration about the particularities of these methods, to which I think my study is a useful starting point.

5.2 Follow-up

I would like to talk about the approach I would take to this project, if I were to redo it provided the knowledge gained. I will be showcasing the steps I would take regarding the two phases: Research and Development, respectively Evaluation.

1. Research and Development:

- Method 1:
  The first goal I would forge for myself is making a new attempt at the real linear continuous motion technique.
  This would be a substantial phase, since, as mentioned previously, collaboration with Machine Learning researchers would be of upmost importance.
  The first thing needed to be done is data collection. Since different people move around in different ways, a substantially large sample would need to be collected.
  After forging the training data, the Supervised Learning algorithm would need an evaluation of its own, in order to prove accuracy in movement classification.
  Once a convenient performance is reached, the process could then be incorporated into the mobile VR experience.
  Careful attention would need to be directed towards a correct translation and scaling of real life distances and velocities into the virtual world.
- Method 2:
  When it comes to the grab/push/pull motion technique, there are several adjustments that are in need.
  The first one is one that was heavily frequent in the Affinity Diagram: better coordination between the controller device and the VR host. I think the best approach to this would be allowing the devices direct communication by turning the sender into a Bluetooth emitter. Another previously mentioned alternative would be direct communication through a Wi-Fi router. By researching the means to achieve this challenge, the experience should feature more natural movements.
  Another aspect to consider is adding two dimensions, possibly even three dimensions, to the controller input itself. This means that the hand device would be removed of any dependency on the Earth's gravitational pull.
  This fix would require sufficient results obtained in the previous method. The same classifier which detects the origin moves of sensor readings could be altered to work for the controller as well.
  This would, however, require a slight compromise when it comes to the affordability aspect, since the arm device would be required to possess a functioning gyroscope.
- Method 3: I do not intend to explore this head-tilt movement method again for now. I do not think that it is a beneficial alternative, compared to the other existing techniques.
- Method 4:
  Peripheral View manipulation has shown to aid in the diminishing of VR sickness symptoms. A careful broadening of the detail spectrum could be done, by experimenting various areas, such as:
  - even further reducing the degree to which the FOV restrictor is noticed by the user;
  - doing a more detailed manipulation of the parameters of the restrictor itself. Examples include margins, shape, blurriness.
- Other:
  Extra methods could be added, in order to improve the palette of available potential sickness reducing factors.
  One such example that came to my mind is experimenting with different ways affordable trackers can be produced.
  Examples would be a simple head-only or head+controller tracker or even full body tracking. These would still have to be done in affordable manners, through cheap or common items.
  An example would be using cheap webcams or older phone cameras and adapting them to provide some form of basic tracking.
  This could aid in the performance of the mobile VR experience and yet maintain its accessibility factor.

2. Evaluation:

   - The first step, before testing even begins, should be selecting a larger batch of participants. The larger the sample, the more experimental flexibility there is. Therefore, more sickness factors could be sought and taken into account.

A key note to add here is that this would allow for a multilaterally-balanced experiment, the method sequence orders being equally distributed amongst the volunteers.

- The next step was proved to be crucial by my own experiment: very careful consideration of user adaptation to a new, uncommon technique. Speed in accommodating to new translations of gestures in interactive experiences is a very important variable, which could shift in value from user to user.
  As mentioned previously, in order to counter-balance for this, preliminary training should be done, in order to accustom the volunteers to the gestures themselves, before commencing the actual simulation.
  Participants would be asked to play a game with a regular 2D display, yet which features any new locomotion technique used.
  After overcoming the barriers of the new methods, the users would move on to run through the actual VR simulation.

5.3 Final word

This project has been a perspective-widening experience for me. It has opened me up to the vast possibilities offered by mobile virtual reality, but also to the challenges that it gives rise to.

I think this area is of great importance and, quite possibly, will increase in significance even more in the future.

I am looking forward to continuing contributing to this area and, hopefully, stimulating others to do the same.

Bibliography

[1] Andreas Aronsson. *Simulator Sickness and VR Legs*. A. Aronsson's WeBOLLog, 2014. http://andreasaronsson.com/2014/08/18/simulator-sickness-and-vr-legs/.

[2] Dom Barnard. *Degrees of Freedom (DoF): 3-DoF vs 6-DoF for VR Headset Selection*. VirtualSpeech, 2019. https://virtualspeech.com/blog/degrees-of-freedom-vr.

[3] Dom Barnard. *History of VR - Timeline of Events and Tech Development*. VirtualSpeech, 2019. https://virtualspeech.com/blog/history-of-vr.

[4] Sophie Charara. *Virtual reality: Then and now - why it won't fail this time*. Wareable, 2015. https://www.wareable.com/vr/virtual-reality-then-now-why-it-wont-fail-this-time.

[5] Mark Billinghurst Christina Dicke, Viljakaisa Aaltonen. *Simulator Sickness in Mobile Spatial Sound Spaces*. ResearchGate, 2009. https://www.researchgate.net/publication/221494074_Simulator_Sickness_in_Mobile_Spatial_Sound_Spaces.

[6] Yano Claeys. *Pirate Island - Cargo Bay*. Sketchfab, 2019. https://sketchfab.com/3d-models/pirate-island-cargo-bay-a45eb7d3f0c9475cb696b2a8fdfe4c1e.

[7] Brandon Birckhead Hannah A. D. Keage Albert Rizzo Tobias Loetscher Dimitrios Saredakis, Ancret Szpak. *Factors Associated With Virtual Reality Sickness in Head-Mounted Displays: A Systematic Review and Meta-Analysis*. Frontiers in Human Neuroscience, 2020. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7145389/.

[8] Byounghyun Yoo Eunhee Chang, Hyun Taek Kim. *Virtual Reality Sickness: A Review of Causes and Measurements*. International Journal of Human–Computer Interaction, 2020. https://www.tandfonline.com/doi/full/10.1080/10447318.2020.1778351.

[9] Kaylee Fagan. *Here's what happens to your body when you've been in virtual reality for too long*. Business Insider, 2018. https://www.businessinsider.com/virtual-reality-vr-side-effects-2018-3.

[10] Artem Kovalev Galina Menshikova. *Vection in virtual environments: psychological and psychophysiological mechanisms*. National Psychological Journal, 2015. https://www.researchgate.net/publication/306006181_Vection_in_virtual_environments_psychological_and_psychophysiological_mechanisms.

[11] Nestor Gilbert. *74 Virtual Reality Statistics You Must Know in 2021/2022: Adoption, Usage Market Share*. FinancesOnline, 2021. https://financesonline.com/virtual-reality-statistics/.

[12] FDM Group. *5 Exciting Uses for Virtual Reality*. FDM Group, 2020. https://www.fdmgroup.com/5-exciting-uses-for-virtual-reality/.

[13] VR Heaven. *VR Motion Sickness Statistics — Age, Gender, Experience and More*. VRHEAVEN, 2020. https://vrheaven.io/vr-motion-sickness-statistics/.

[14] Christian Winkler Gabriel Haas Enrico Rukzio Jan Gugenheimer, David Dobbelstein. *FaceTouch: Touch Interaction for Mobile Virtual Reality*. Association for Computing Machinery, 2016. https://dl.acm.org/doi/abs/10.1145/2851581.2890242.

[15] Valeriia Kyselova. *44 Best VR Apps for Google Cardboard*. Jasoren, 2019. https://jasoren.com/44-best-vr-apps-for-google-cardboard/.

[16] Lazaroe. *Pirate Ship Destroyer.* Sketchfab, 2018. https://sketchfab.com/3d-models/pirate-ship-destroyer-7179904a059340adbaf9d7a5adc3549e.

[17] Stacy Sampson Noreen Iftikhar. *You Inner Ear Explained.* Healthline, 2019. https://www.healthline.com/health/inner-ear.

[18] John Oyekan Chris Turner Ashutosh Tiwari Petti A., Windo Hutabarat. *Impact of model fidelity in factory layout assessment using immersive Discrete Event Simulation.* ResearchGate, 2016. https://www.researchgate.net/publication/306537486_Impact_of_model_fidelity_in_factory_layout_assessment_using_immersive_Discrete_Event_Simulation.

[19] Kevin S. Berbaum Michael G. Lilienthal Robert S. Kennedy, Norman E. Lane. *Simulator Sickness Questionnaire: An Enhanced Method for Quantifying Simulator Sickness.* The International Journal of Aviation Psychology, 2009. https://www.tandfonline.com/doi/abs/10.1207/s15327108ijap0303_3.

[20] Anne Roudaut. *Controlled experiment/Hypothesis testing - Lecture.* University of Bristol, 2021. https://web.microsoftstream.com/video/10774390-781f-41c6-9631-722d290c41f7.

[21] Jinmo Kim Seunghun Han. *A Study on Immersion of Hand Interaction for Mobile Platform Virtual Reality Contents.* Symmetry, 2017. https://www.mdpi.com/2073-8994/9/2/22/htm.

[22] Alena Shek. *Sea House.* Sketchfab, 2020. https://sketchfab.com/3d-models/sea-house-bc4782005e9646fb9e6e18df61bfd28d.

[23] Scott Stein. *Best VR headsets for 2021.* CNET, 2021. https://www.cnet.com/news/best-vr-headsets-for-2021/.

[24] Sophie Thompson. *Motion Sickness in VR: Why it happens and how to minimise it.* VirtualSpeech, 2020. https://virtualspeech.com/blog/motion-sickness-vr.

[25] Tracy Watson. *Virtual Reality for Space Exploration and Astronaut Training.* Skywell Software, 2020. https://skywell.software/blog/virtual-reality-for-space-exploration-astronaut-training/.

[26] Rahmita Wirza O. K. Rahmat Noris Mohd Norowi Yangyi Ang, Puteri Suhaiza Sulaiman. *Swing-In-Place (SIP): A Less Fatigue Walking-in-Place Method With Side-Viewing Functionality for Mobile Virtual Reality.* IEEE Access (Volume: 7), 2019. https://ieeexplore.ieee.org/abstract/document/8935248.