Raquel Duarte Macedo

Bachelor in Computer Science and Informatics Engineering

Paralympic VR Game

Immersive Game using Virtual Reality Technology

Dissertation submitted in partial fulfillment
of the requirements for the degree of

Master of Science in
Computer Science and Informatics Engineering

Adviser: Nuno Manuel Robalo Correia, Full Professor, NOVA University of Lisbon

Co-adviser: Teresa Isabel Lopes Romão, Assistant Professor, NOVA University of Lisbon

Examination Committee

Chairperson: Professor José Augusto Legatheaux Martins, NOVA University of Lisbon
Raporteur: Professor Rui Filipe Fernandes Prada, Instituto Superior Técnico
Member: Professor Nuno Manuel Robalo Correia, NOVA University of Lisbon

FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

November, 2018

This document was created using the (pdf)LaTeX processor, based in the "novathesis" template[1], developed at the Dep. Informática of FCT-NOVA [2].
[1] https://github.com/joaomlourenco/novathesis [2] http://www.di.fct.unl.pt

Acknowledgements

First and foremost I wish to thank the *Faculdade de Ciências and Tecnologias*, NOVA University of Lisbon, that helped me grow professionally. I would like to thank all my professors, specially my advisers Nuno Correia and Teresa Romão for their continuous guidance, patience, availability and support throughout the realization of this work.

Furthermore, i would like to express my gratitude to Caroline Delmazo of *Faculdade de Ciências Sociais e Humanas*, NOVA University of Lisbon, and the *Associação Portuguesa de Deficientes de Lisboa* for their collaboration in this work.

I'm also incredibly thankful to all my friends and class mates, in particular the ones who helped me by giving feedback and suggestions relating to the developed system. I would also like to thank all volunteers who participated in the user tests.

Finally, I must thank my family, especially my parents, who have always encouraged and supported me.

Abstract

Throughout the years virtual reality has been used for a wide range of applications, and several types of research have been made in order to improve its techniques and technology. In the last few years, the interest in virtual reality has been increasing partially due to the emergence of cheaper and more accessible hardware, and the increase in content available. One of the possible applications for virtual reality is to lead people into seeing situations from a different perspective, which can help change opinions.

This thesis uses virtual reality to help people better understand paralympic sports by allowing them to experience the sports' world from the athletes' perspective. For the creation of the virtual environment, both computer-generated elements and 360 video are used. The integration of these two components presented a challenge to explore.

This thesis focused on wheelchair basketball, and a simulator of this sport was created resorting to the use of a game engine (Unity 3D). For the development of this simulator, computer-generated elements were built, and the interaction with them implemented. Besides allowing the users to play the sport as if they are in the athlete's shoes, users can also watch 360 videos in which explanations of the modality (rules and classification) are presented. They are also capable of interacting with some of these videos through virtual elements that are placed over the videos.

User studies were conducted to evaluate the sense of presence, motion sickness and usability of the system developed. The results were positive although there are still some aspects that should be improved.

Keywords: Virtual Reality; Simulator; 360 Video; Computer Graphics

Resumo

Ao longo dos anos a realidade virtual tem sido usada para uma grande variedade de aplicações e têm sido feitas várias investigações no sentido de melhorar as técnicas e tecnologias associadas a esta. Nos últimos anos, o interesse pela realidade virtual tem vindo a aumentar, em parte devido ao aumento do conteúdo disponível e ao surgimento de hardware mais barato e acessível. Uma das possíveis aplicações para a realidade virtual é levar as pessoas a experienciar situações de uma perspectiva diferente, o que pode ajudar a mudar as opiniões.

Nesta tese faz-se uso da realidade virtual para levar as pessoas a ter uma maior compreensão relativa aos desportos paralímpicos, sendo que permitirá experienciar o universo do desporto da perspectiva dos atletas. Para a criação do ambiente virtual, são usados elementos gerados por computador e vídeo 360, sendo que a integração destes dois componentes foi um dos desafios a explorar.

Esta tese focou-se no basquetebol em cadeira de rodas e envolveu a criação de um simulador deste desporto, implementado em plataforma de videojogo (Unity 3D). Para o desenvolvimento deste simulador, foram criados elementos gerados por computador e implementada a interação com estes. Para além dos utilizadores terem a oportunidade de experimentarem praticar o desporto na perspectiva do atleta, também podem ver vídeos 360 nos quais estão presentes explicações sobre a modalidade (regras e classificação). Podem ainda interagir com alguns destes vídeos através de elementos virtuais que são colocados sobre os vídeos.

Foram realizados testes com utilizadores para avaliar a sensação de presença, a *motion sickness* e a usabilidade do sistema desenvolvido. Os resultados foram positivos apesar de ainda existirem alguns aspectos a melhorar.

Palavras-chave: Realidade Virtual; Simulador; Video 360; Computação Gráfica

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This chapter presents the context of the project in which this thesis is integrated, the motivation behind it, the proposed solution, and the resulting contributions.

1.1 Context

This thesis is a collaboration with the project Paralympic VR: an immersive experience, by Caroline Delmazo, developed at iNOVA Media Lab of *Faculdade de Ciências Sociais e Humanas*, NOVA University of Lisbon. The project has the purpose of exploring virtual reality techniques and its main goal is the creation of an immersive experience in the paralympic sports world. This experience intends to present a narrative in a different and more engaging way, leading the people to better understand paralympic sports, and to be able to see it from an athlete's perspective. It is expected that this project may also help promote and value paralympic sports.

This work had the collaboration of *Associação Portuguesa de Deficientes de Lisboa*, who provided help with insight on the sport and in the evaluation of the system developed. In the future there is also the possibility of a cooperation with the International Paralympic Committee.

1.2 Motivation and Problem Definition

Virtual reality has a wide range of possible applications. It can even be applied for social purposes. This project intends to allow people to gain a better and fairer understanding regarding the paralympic sports.

Through the use of virtual reality techniques, this project creates an immersive experience that allows people to see from a new perspective, experiencing things differently and potentially even changing opinions. Concretely the experience allows someone to witness and/or take the place of an athlete, granting the chance to perceive the sport from his/her perspective. This way people are expected to see beyond the athlete's disability and see him/her as any other sports competitor.

Besides the opportunity to play the sports, the system also provides didactic information regarding the sports, such as rules, history and curiosities.

In this thesis, the sport that was focused in is the wheelchair basketball.

1.3 Solution

In this thesis, to help achieve the goal of the project, a virtual reality system was created. This system is composed by two parts. The first is a 360-degree access to trainings, through the use of video, with explanations and interviews. The second allows the user to try out the sport through a virtual simulation of it, developed with a game engine.

The system includes both 360-degree video and computer-generated elements. Therefore, part of the challenge was how to integrate both together as seamlessly as possible. Hence, experiences were performed to attempt to conclude how to best integrate the two aspects over time, with one following the other in sequence, and even displaying both simultaneously. Three different techniques were applied to achieve this integration: fading from video to virtual, use of video as a texture, and the placement of computer-generated elements onto tracked 360 video. The achievement of a seamless combination of 360 video and 3D computer-generated models presents one of the challenges of this work.

Another challenge is how to make a user feel like he/she is the athlete, making him/her perceive the virtual world from a first-person perspective. For this, there is the need to make the environment surrounding the user, and the interactions with it believable. Therefore, care was taken in the creation of the 3D elements' models. In order to simulate the elements that exist in the real world, as well as their behaviour, there was the need to study how to translate the physics of the reality to the simulation.

In the scope of this thesis some of the models of the 3D elements were built. In particular, a virtual model of a wheelchair and its movement had to be developed. When a user interacts with an element, its physical model should lead it to behave as similar to what would happen in reality as consequence of the interaction as possible.

The interaction with the elements of the virtual environment must be faithful enough for the user to perceive it as being genuine, so that he/she can feel immersed in the environment and thus truly feel like he/she is in the athlete's place.

Finally, considering that the project is in collaboration with *Faculdade de Ciências Sociais e Humanas*, there is a journalistic component that must also be taken into consideration. Hence, there was also the need to contemplate how to best transpose a narrative to the simulation. Therefore, it was necessary to consider possible ways to tell a story in this new medium that is virtual reality, which is reflected in the succession of scenes available for the user to access at each time.

1.4 Contributions

As a result from the work executed in the scope of this thesis, there are the following contributions:

- **Wheelchair basketball simulator** - Development of a paralympic sports simulator that simulates wheelchair basketball. This includes the creation of 3D computer-generated elements, the implementation of the interaction with those elements, and the assembling of the virtual environment.
- **Integration of 360 video with a computer-generated environment** - Development and testing of techniques for the integration of 360 video and computer-generated elements, both with one following the other as seamlessly as possible, and incorporating the two simultaneously.
- **System evaluation** - Evaluation of the developed system through user tests, in terms of sense of presence, motion sickness, and usability. User tests were also performed with actual athletes to verify the fidelity of the simulation.

1.5 Document Structure

This document is divided in the five following chapters:

- **Introduction** - This first chapter describes the context and motivation of the thesis. It also presents a brief description of the proposed solution, and the main contributions that resulted from it.
- **Related Work** - The second chapter presents the result of the study of techniques and case studies that helped in the development of the solution. In this chapter are introduced useful concepts related to virtual reality, in particular related to the use of virtual reality in sports and simulations. Knowledge related to the recording, storage, and reproduction of 360 videos is also described. At the end of the chapter is presented an overview of ways to evaluate a virtual reality system.
- **Design and Implementation** - The third chapter delineates the solution, describing some of the techniques, hardware, and software that were used. It presents the main functionalities of the system developed and explains its implementation.
- **Evaluation and Results** - This chapter explains how the system was evaluated, analyses the results of those evaluations, and describes the improvements that resulted from them.
- **Conclusions and Future Work** - The fifth and last chapter presents the conclusions drawn from the creation and evaluation of the system developed, and also some improvements and further work that should be done in the future.

Chapter 2

Related work

This chapter is divided in four sections: Virtual Reality, Virtual Reality Simulators, 360 Video, and Evaluation. The first section gives a sense of what is virtual reality, the technologies associated with it, and aspects that must be taken in consideration when creating a sports virtual environment. The second describes virtual reality simulators, particularly sports and wheelchair simulators. The third exposes knowledge related to the creation, storage and reproduction of 360 video. Finally, the last section presents an overview of ways to evaluate a virtual reality system.

2.1 Virtual Reality

Virtual reality (VR) can be described as a three-dimensional computer generated environment which can be explored and interacted with. Since VR is a ”reality” that is ”virtual”, in principle, anything that can happen in reality can be programmed to happen but virtually. However, the real power of VR is not necessarily to produce a faithful reproduction of ”reality” but rather that it offers the possibility to step outside of the normal bounds of reality and realize goals in a totally new and unexpected way. In the virtual world, there is an infinity of possibility that does not exist in the physical world, because in the virtual world people are able to do anything and be anything, without any limitations other than the imagination.

An important goal of VR is to replace real sense perceptions by computer-generated ones. Perception is an active process that combines the processing of sensory inputs with a person's previously existing model of the world. Our perceptual system is capable of inferring a full model of, for example, an object or place, just from a small amount of sensory information. Thanks to this VR works relatively well even with simplistic rendering of the surroundings as long as it provides enough cues for the perceptual system to, using as base an internal model, infer a full model.

If the sensory perceptions are substituted with the computer-generated ones, then the brain will infer its model from that sensory data rather than from reality, resulting in immersion into the virtual world. The most targeted sensory system is vision and often also the auditory. Sometimes touch and force feedback (haptics) is also made use of. On the other hand smell and taste are rare. The more senses are targeted in a system, the more easily that system will manage to be immersive, as long as the sensory stimulus complement each other [39].

Haptic devices can be real physical objects corresponding to the virtual ones so that when users ”touch” the virtual object they feel the tactile feedback from actually touching the physical object. Another form of haptic device are the vibrotactile feedback devices, which make use of many small vibrating transducers to provide users with feedback [25].

2.1.1 Technology

In order to be able to experience and interact with a VE, the appropriate technology is needed. Several types of technologies can be considered, such as: display technologies; tracking technologies; and haptic technologies. An essential part of VR systems is to provide feedback to the user. The feedback must occur in real time. Considering that the sense people rely on the most is the sight, the visual display technologies need particular attention. The generally used ones are head-mounted displays (HMDs) and cave automatic virtual environments (CAVEs). All of these can be used to present stereoscopic images, with a variety of methods for separating the left and right eye's image [25].

CAVE The CAVE is a projection-based VR display. It generally is a cubed room in which images are back-projected to three screen walls (front, left and right) and down-projected to the floor screen. This way participants will not cast shadows on the walls but will be able to do so on the floor. As the viewer moves around in the environment, his/her position is tracked and an off-axis perspective projection is calculated according to that position. The tracker is mounted on top of the stereo glasses the viewer uses. Using the tracked position, the position of each eye can be determined and used to obtain the correct projections. Stereo vision is achieved by having in each screen a projection for each eye using frame sequential stereo[1]. The glasses the viewer uses are lightweight shutter glasses which alternatively have one eye lens opaque and the other transparent, thus achieving a 3D stereo effect [9]. Using the CAVE a participant is able to see his/her own body which may be an advantage in some cases, but may also be a problem if the VE created is one in which the participant is not supposed to have a human body for example. Each Cave typically has to be tailor-made and occupies a lot of space, reason why it has not became a mass product [39].

[1] Sending the different left and right eye images sequence one after another in time

HMD As the name suggests, a HMD is a display device used on the user's head. Recent HMDs deliver a computer-generated image to each eye. Both images are computed and rendered respecting the position of the corresponding eye in the three-dimensional virtual scene, therefore forming a stereo pair. The images are presented to the user via two displays, one in front of each eye, mounted in a frame that also includes a mechanism to capture the position and orientation of the user's head, and therefore gaze direction. As the head moves, this information is transmitted to the computer, which recomputes the image and sends it to the displays [39]. Hence, the HMD is not just a output device but also an input one.

2.1.1.1 HMDs

Through the years many HMDs have been created. In some the displays are provided by a smartphone that is placed in the head mount and in others the display is included in it. In the ones that use a smartphone, its sensors such as the gyroscope are used to provide the input, and the output images are displayed on the phone screen in VR format [59]. Google Cardboard is a famous VR platform that allows the use of both Android and iOS with a simple, cheap and easy-to-use headset [10]. The users can even build their own headset. Therefore, the creation of this platform helped VR reach the general public. Other examples of HMDs are:

Oculus Rift and Samsung Gear VR The Oculus Rift HMD [16][28] offers light ergonomics, 360 degree tracking, and asymmetric lenses to maximize field of view and image quality. Rift also has integrated headphones that provide 3D spatial audio[2] and comes with a pair of touch controllers that enable the manipulation of the VEs. Recently the feature of room-scale motion tracking was added. The Samsung Gear VR is an offshoot of the Oculus Rift that uses a smartphone GALAXY. It also comes with a controller.

Playstation VR Playstation VR [12] is the official HMD for the Sony Playstation 4. It has integrated 3D spatial audio and a microphone. The Playstation Camera tracks the head movement thanks to LEDs positioned on the surface of the HMD. The camera has double lenses and depth sensors so that it can track the device's position.

HTC VIVE HTC VIVE (fig. 2.1) [16][54][48] was developed in cooperation with the game studio Valve. The headset covers a field of view of about 110º and has a refresh rate of 90Hz. It provides room-scale tracking (fig. 2.2) and comes with two controllers that support haptic feedback. HTC VIVE includes built-in microphone, but unlike the previously mentioned HMDs, does not include built-in 3D spatial audio, having only the

[2] Audio that replicates the physics of how humans hear in real life, but delivered over a normal pair of headphones. As people look around them the sound updates accordingly, mimicking what happens in real life.

means to plug headphones into it. However, the new HTC VIVE Pro already has built-in 3D spatial audio.

Figure 2.1: HTC VIVE's headset, controllers and base stations [54]

HTC VIVE's tracking measurements have high precision and its system latency is low. The headset comes together with two base stations, which are infrared laser emitter units. These emitters alternatingly send out horizontal and vertical infrared laser sweeps. The difference of time at which the various photodiodes, that can be found on the surface of the headset and controllers, are hit by the laser helps tracking the position and orientation of this devices. The base stations can be used together, in which case they must be synchronized, or individually, using only one of them. VIVE also includes inertial measurement units (IMUs), a system that measures motion using gyroscopes, accelerometers and others. The information obtained by these is also used for the tracking. In order to achieve higher update rates, the information from the IMUs is employed together with those obtained from the photodiodes to estimate the headset most likely current position through dead reckoning [26].

Figure 2.2: HTC VIVE's room-scale tracking [30]

2.1.2 Virtual Body

In order for a user to truly feel like he/she is inside a VE, the existence of a body which the user can control and use to interact with the VE is advantageous. Unlike in CAVE systems, where the user can still see his own body, when using a HMD the body will have to be a virtual body. The advantage of the virtual body is that it does not have to be like a person's real body. Therefore it can be used to produce illusions of body morphing. The user can see himself/herself has having the body of a completely different person or even of a different creature.

When using a virtual body, however, there is the correlated concern of making the user feel a sense of ownership over that body. That is, of making the user unconsciously act like the virtual body is his/her own. This will assist the user in accomplishing a sense of being embed in the VE.

Studies have shown that synchronous visuotactile stimulation tends to lead to an illusion of ownership, leading to the impression that an object is part of the body or even that a completely different body is the person's real body. This synchronous visuotactile stimulation is when a person feels a sense of touch corresponding to the one seen to be happening to the fake body. The visual and tactile information together generate the illusion of the fake body being the real one.

The illusion has also been demonstrated to occur through the use of visuomotor correlation between the movements of a hidden real hand and a seen fake hand.

When a virtual body exists and the user sees the VE from the viewpoint of that body's eyes, it is said that it is being viewed from a first-person perspective (1PP). On the other hand, when the virtual body is visible but is not spatially coincident with the real body, it is being viewed from a third-person perspective (3PP) [39].

The 3PP tends to be common in video games for actions such as moving around the environment since it provides a more global view of the environment to the user. The 1PP is usually used for the actions that require more precision. In VR, using a HMD, with the 3PP it can be disturbing to turn the head and always looking at the same place with an avatar turning its head. On the other hand, distance estimation tends to be worst when using 1PP then with the 3PP. Thus, both perspectives have their disadvantages. Salamin et al. [33] believe that it might be useful to be able to switch between the two perspectives depending on the action to perform, like in the video games. They argue that both perspectives are needed during the simulations composed of varying actions.

Slater et al. [41] reported an experiment that shows that ownership can be transferred to an entirely virtual body. They demonstrate that a 1PP of a life-sized virtual human female body appearing to replace the male subjects' real bodys is sufficient to generate a body transfer illusion even though the virtual body did not resemble the subjects' real one. Slater et al. verified that the most important factors leading to the temporary illusion of ownership of the virtual body are the participants' perspective, touch (visuotactile synchrony) and movement (synchrony of the virtual body's head movement with the real one). They refer that when considered together, perspective, particularly the 1PP, dominates as an explanatory factor for subjective and physiological measures of ownership. When a virtual body is perceived to be in the same place the real one is expected to be in, the brain appears to be tricked into believing that the body must be the person's own.

Therefore, the existence of a virtual body and the use of the 1PP is highly beneficial for the feeling of immersion in a VE.

2.1.3 Sports Virtual Reality

As previously mentioned, VR has a wide range of applications. It is, for example, used in sports. In this area it can be used for entertainment reasons, for learning and training, to help plan strategies and tactics, for rehabilitation after an injury, and to allow spectators to watch matches they cannot physically attend [39].

VR can be useful for sports training since it allows to practice in a safe and cheap environment, in a large set of specific situations, and provide additional information and feedback about the user performance. However, when designing a system for sports training one must pay special attention to how well skills can be transferred from the virtual to the real world since many factors specific to VEs lead to differences in the way users execute motor tasks [7]. Therefore, in sport-themed VEs is very important to have good perceptual fidelity, and even more important to have high functional fidelity.

Motor control skills, especially accuracy, tend to be trained through VEs by practicing tasks such as aiming and throwing balls [25]. In VE users tend to underestimate distances, which leads to motor adaptations in the VE that are not the correct ones for success in the real environment. Covaci, A. et al. [7] made an experiment and tested the effects of different visual conditions in a training simulator. As expected, the 1PP was shown to be the worst for distance estimation. The results supported the theory that 3PP with guidance feedback is better to reduce the error of the perceived distance and to guide the user towards a better throwing technique.

Guidance feedback is a type of feedback that helps guiding the users on how to perform the next action. The informative feedback, on the other hand, is a type that supplies the users with information and statistics regarding their performance. Feedback essentially guides the performer to the correct movement pattern and increases skill acquisition. Nevertheless, if the learner receives feedback too frequently they may develop a dependency on it, which leads to the performance suffering when the feedback becomes unavailable [25].

Gradl, S. et al. [16] conducted a survey among athletes in order to uncover their acceptance of virtual reality headsets for training in different kinds of sports. The results showed that despite most people not knowing about VR beforehand, the majority is still inclined to use it after being presented with possible usage scenarios.

Since many sports revolve around a physical object, the use of real, physical haptic items could be beneficial to maintain links to reality and, therefore, increase the feeling of immersion. Audio feedback can also have a big impact since it can help, for example, to judge how hard a ball has been hit [39].

Sport VEs often benefit from the existence of a crowd watching and supporting the athletes. Most sports-related video games exclusively use sprite animations to simulate that crowd since these are capable of effectively conveying crowd cheering such as applause and celebrations. Sprites are 2D graphic objects that are essentially standard textures for which exist special techniques for combining and managing them [49]. A possible alternative to sprites would be the use of 3D crow avatars but these tend to affect framerates, leading to loss of performance [21].

An aspect to have in consideration when creating a VR sports system is that the sport may involve the need for participants to make rapid motions. This can lead to problems related to latency since even small delays could have a big impact, making the VE seem unnatural and possibly leading to a sense of motion sickness. The use of HMDs can also cause problems due to the weight and possibility of causing unwanted constraints to the user's movements.

Another potential problem for VR in sports where the athlete can move in a large play area is that the effective space in which a user will be able to move in the VR system will be much smaller than the real space. Different techniques can be used to face this issue. For example, the user may move around by using a joystick, by point-and-click techniques, by walking in place, etc. The sense of presence tends to be better with techniques that involve walking rather than point-and-click techniques [39].

Interrante, V. et al. [19] explored a metaphor for walking-based locomotion - the Seven League Boots. The key characteristic of this method is that it involves determining the user's intended direction of travel and then augmenting only the component of his or her motion that is aligned with that direction. The intention is to let each step that the user takes in the real world appear to have the same consequence as that of taking, for example, seven steps in the virtual world. The boots can be activated through a wand that has a button, which when pressed activates the boots. An alternative approach, that is harder to use is having the boots always active and having the speed smoothly increase and decrease as the user starts to walk and stops respectively.

The Iowa State University (ISU) athletic department and the Virtual Reality Applications Center (VRAC), in order to help attracting athletes in the highly competitive college football recruiting process, created a VE that highlights the atmosphere of the campus by simulating a football game-day [21]. The experience was designed and developed for display in both a high resolution six-sided VR environment - the C6 [3], and a portable HMD system. Results indicate that both are an improvement over the standard practice of showing videos to convey the atmosphere, and that the two performed similarly in terms of immersion.

For the development of the VE, the game engine Unity was employed due to its

[3] http://www.vrac.iastate.edu/facilities/c6/

simplicity in scripting a VE, including avatar animations, as well as its support for Oculus Rift HMD. For custom animations and polygon count reduction Autodesk Maya was put to use, and Mixamo [4] was used for applying animations of avatars. In order to improve the framerates and, therefore, improve performance of the system, the objects that had significant polygon counts such as railings were replaced by textures wrapped on two triangles, interior geometry that did not add direct value to the application were deleted from the stadium geometry, and different levels of detail were assigned for various avatars depending upon their importance. To create the sensation of a full stadium a crowd was created by wrapping textures of actual crowd instead of bleachers textures and by using sprites.

Wear the Rose was a campaign which had the goal of attracting support for the England rugby team. As part of this campaign an immersive experience was created [52]. Considered to be the first immersive 360 degree live action gaming experience created using Oculus Rift technology, it was designed by UNIT9 Films [5], and allows fans to feel like they are training with the England team. The experience places the user at the center of the national rugby team's training session. The video was recorded during a genuine training session using nine GoPro Hero 3 cameras, using a custom designed camera rig that allowed to capture a 360-degree video. Animated graphics that display player stats, the distance the ball has traveled and other trivia overlay the video, adding to the experience.

2.2 Virtual Reality Simulators

VR can be used to simulate real situations and environments. Nowadays simulations keep getting more realistic. VEs are becoming increasingly immersive and similar to reality. This is in great part due to the continuous advances regarding the previously referred technologies, methods to stimulate the sensory systems, and interaction strategies.

Simulators typically use physics models to achieve a more realistic simulation of an event. The real physical parameters needed for the mathematical models are not always easy to obtain. Furthermore, despite the fact that highly complex mathematical models produce more accurate results, they generally require great computational power which is sometimes too much for a real-time solution. In consequence simplifications are often used [25].

Nonny de la Peña, known as the "Godmother of virtual reality" and recognized for her ground-breaking work in immersive journalism, created in 2007 the project "Gone Gitmo" [20][39] together with the digital artist Peggy Weil. Gone Gitmo is a fact-based simulation of the Guantanamo Bay prison in which the participant possesses a virtual body and sees the VE from a 1PP. This simulation allows the user to experience what it

[4]https://www.mixamo.com/
[5]https://www.unit9.com/

feels like to be a prisoner in the camp and results from combining data from actual events with computer-based reconstruction.

Simulations can have a great variety of applications, such as for journalism, like the Gone Gitmo project, allowing the viewers to actually experience the news and therefore causing a bigger impact; sports practicing; learning to drive vehicles; scientific experiments; among many others.

2.2.1 Sports Simulators

In the sports context, simulators can be used for different applications. Simulators are utilized especially for training which, as previously mentioned, requires high fidelity to guarantee a better knowledge transfer to the reality.

Ribeiro, J. et al. [32] created a realistic boccia [6] game simulator adapted for people with disabilities or motor disorder. The simulator's main focus is the rehabilitation of patients with impairments and disabilities and is aimed both at casual users and athletes. The later can use it for training from home without needing their coach.

An important aspect of this boccia simulator is the attempt to implement a user interface that is adaptable to various disabilities. For this, a multimodal interface was chosen since it allows multiple input methods, granting the possibility to make the interface adapt automatically to the users easily and to create specific and customizable profiles for each.

Ribeiro, J. et al. tried to make a realistic simulator, and for that a realistic field, balls, and ball throwing platform were created. The balls were made available with different toughness, and their physical model was adapted using values obtained from real throws so that the results of the simulation would be as close to the reality as possible. The ramp of the ball throwing platform was created using Maya, and it was placed in a robotic model created in a robotic simulator. The game engine used in this project was the Unreal Engine.

In early 2018, IMG Studio [7] launched a virtual reality wheelchair basketball game [3][45][15]. The game was created to help recruit players, serve as a training tool for athletes, and increase awareness of paralympic sports.

IMG Studio tried to make the VR game as accurate as possible by studying the physics of the game and making the wheelchair behave like it would in real life. The VR world created features a stadium with a scoreboard, a wheelchair and even cheering fans. When playing, the user uses touch controllers to shoot the basketball from the perspective of an actual player, seated in a sports wheelchair.

Sports simulators are also being created for entertainment purposes. Some VR games that simulate sports are already available. An example is the "VR Shoot Around" [8], a

[6] Indoor paralympic sport
[7] https://theimgstudio.com/
[8] http://store.steampowered.com/app/671740/

realistic basketball simulator that uses physical calculations to reproduce the movement of the rim-net, and includes power and angle assist to help shooting the ball correctly.

2.2.2 Wheelchair Simulators

Several wheelchair simulators have been created with different goals in mind. These simulators offer a range of possibilities. They can be used to help in the development and testing of new wheelchair concepts and devices, to assess patient capability, to train users to use powered wheelchairs and/or perform maneuvers, to evaluate the accessibility of built environments, and offers the possibility to experience what it is like to be a wheelchair user [44][8].

In a wheelchair simulator, sensorimotor interfaces allow the user to move in the virtual environment. There are two main types of wheelchairs: powered and manual. The first is the most simple to simulate since its interface can consist of a standard computer peripheral joystick used in a manner directly analogous to a powered wheelchair's control interface. The simulation of the manual one is more complex as the interface is through the rotation of the driving wheels and is more directly linked to the users own physical effort [17].

As already mentioned, people perceive their environment through their senses and it can be divided according to which sense is stimulated. The vestibular sensory system provides information about angular and linear movement and balance. Wheelchairs stimulate this system when the user accelerates, decelerates, among other situations. Therefore, to be effective, a wheelchair simulator should stimulate the user's vestibular system which can be done through a motion platform.

For manual wheelchair simulators, force feedback is also of great importance since it can be used to give the perception of the effort required to push the handrims. It can inclusively be used to simulate conditions such as the physical features of the ground and collisions.

Simulation of the virtual wheelchair in the virtual environment requires graphics design and a precise behavioral simulation of the wheelchair to create a link between the behaviors of the real and virtual wheelchairs. This means that the virtual wheelchair must behave the same way as it would be expected from a real one present in the same conditions [29].

Sørensen, L., and Hansen, J. [44] developed a prototype of a low-cost manual wheelchair simulator that consists of a stationary platform with 4 rollers and 2 encoders that provide input to a VR-model. A real wheelchair is placed on top of the platform and as the user moves the real wheelchair's wheels, the rollers allow the wheelchair to stay in the same place and pick up the movements. The encoders are connected to the rollers and send the information to a computer. This simulator does not provide physical feedback or vestibular sensations which led some subjects to experience motion sickness.

Grant, M. et al. [17] have also developed a project that aimed to design and build a motion platform capable of simulating wheelchair navigation in virtual buildings. Their goal was to create a virtual reality facility that could be used to generate guidelines which address the issue of wheelchair access to, and within, the built environment. Unlike the low-cost platform, this one attempts to provide haptic feedback that informs of collisions with virtual objects and matches the altered sense of effort needed to propel a wheelchair over varying surfaces and slope conditions.

Likewise, Challenging Environment Assessment Laboratory (CEAL) of Toronto Rehabilitation Institute also developed a manual wheelchair simulator [8] with the intention of encouraging research that can meaningfully address the challenges faced by wheelchair users. It has a motion platform designed to facilitate the simulation of non-planar motions. It also provides the subject with force-feedback at the handrims.

2.3 360 Video

Unlike the most common type of video, in which is used a standard camera that captures only in the direction it is pointed at, 360 videos are recorded in all directions, allowing for a complete 360-degree view [13].

Cinematic virtual reality is a type of virtual reality that adapts filmmaking to VR. It uses 360-degree video which allows the viewer to look all around the scene as it unfolds. The video is filmed using a panoramic video camera system and played back as an equirectangular video file (fig. 2.3). This video is seen as if projected in a sphere surrounding the viewer. This type of VR has the advantage of scenes looking completely real and not computer generated. However, the viewer cannot move around the scene freely. There is only movement if the camera is moved during filming.

Figure 2.3: Equirectangular format

Depending on the camera system and stitching process the scenes can be either monoscopic or stereoscopic. Mono footage is flat and has no depth. Everything is projected back to the same depth of the 360-degree viewing sphere. On the other hand, stereo footage gives a 3D effect in every direction, so objects in the 360-degree sphere can appear to get closer to the viewer. This may lead to a more naturalistic and immersive feeling since it is more similar to how things are experienced in real life. Jaunt Studios[9] defends the importance of stereoscopic video in VR, referring that to truly get the sense of being present one must shoot in stereoscopic 3D wherever possible [46].

However Bessa et al. [2] made a study comparing visualization using HMD of monoscopic 360 video with 360 3D video and the results showed no significant differences between the 2D and 3D video when considering the sense of presence or cybersickness, two variables often used in the evaluation of VEs (as explained further ahead in the evaluation section). Therefore they concluded that the use of 3D 360 video does not improve the VR experience. Baños, Rosa M., et al. [1] also made a similar study and the results likewise showed no difference between stereoscopic and monoscopic video neither in terms of presence nor emotional reactions.

Besides the 360 images, the audio is also a very important part of the cinematic virtual reality. The proper audio can help with immersion and spatial 3D sound increases a person's sense of place.

2.3.1 Cameras

There are many types of camera systems for shooting 360 video [46].

Panoptic This is the most popular type of VR camera rig. These camera systems, generally inspired by the visual system of flying insects, consists of many discrete camera modules arranged on a sphere, dome or another shape. The images of the different cameras overlap so that they can the stitched together to form the equirectangular video. The camera rig must have enough cameras to provide sufficient overlap between images to properly stitch the adjacent frames together. To be able to provide stereo stitch even more are needed. Due to being small, lightweight, and relatively inexpensive, the GoPro is a common camera in the VR camera rigs. However, the use of this cameras has several problems, the most important one being the lack of sync. For the overlapping images to match precisely and be easily stitched together it is crucial for all cameras to be in sync. The GoPro has the problem of not having built-in syncing capability and that even if synced after recording based on audio/visual cues they can still drift over time.

Mirror rigs This is also a common type of panoramic 360-degree camera and typically has a number of cameras in a circular configuration shooting up into a collection of mirrors that are facing out into the scene at a certain angle. Due to the mirrors, these rigs tend to be bigger and heavier. However, they have the advantage of making stitching very easy and relatively artifact free. Many allow for real-time stitching and transmission of live 360-degree imagery, which brings the inconvenience of needing to be attached to a

[9] https://www.jauntvr.com/

relatively powerful computer. The mirror rigs can produce stereo video if they have two cameras shooting into each mirror.

Fisheye Since they are relatively cheap, small, lightweight, and easily stitched, many consumer camera rigs are of this variety. However, the quality of the video tends to be relatively low. None of these types of cameras produces stereoscopic 3D images. Some of these use only one lens, capturing only 180 degrees, while the ones with two lenses capture 360 degree.

Light-field With this type, instead of focusing light through a lens and onto a sensor, there are hundreds of tiny micro-lenses that capture light rays from every conceivable direction, which in essence captures a hologram of the environment as it would be viewed from a certain volume of 3D space. Thereupon it becomes possible for the viewer to lean into the shot and change his perspective. Most video based light field cameras use numerous camera modules with wide lenses arranged in a grid or sphere configurations, and these multiple video streams can then, through processing, be packed into a compressed light field format. Light field capture allows for more flexibility in post-production, allowing to, for example, change camera position, frame rates, and even completely remove an object from the scene. These cameras still do not tend to be used in production since they require a large array of computers attached to the camera, and the storage, bandwidth, and processing requirements are quite high [36].

2.3.2 Stitching

Once the 360 video has been recorded, generally the diverse video streams captured by different cameras will have some overlapped areas that need to be stitched together to form a single 360-degree video. Several video stitching software exist to accomplish this, such as the VideoStitch Studio[10], the Autopano Video[11], and the Vahana VR[12].

There are various approaches to perform the stitching of the videos, though the techniques are usually classified into two: Direct and Feature-based techniques. In direct techniques pixels of an image are matched with those of another image by comparing their intensities. On the other hand, the feature based techniques extract distinct features from the processed images and correlate them. This has the advantage of being more robust against scene rotation and other variations [31].

An example of a direct technique is optical flow stitching. Optical flow algorithms calculate the movement of every pixel in a scene usually across a series of frames [46]. Checking the movement across neighboring frames leads to better matching of the pixels than by only comparing the pixels of an isolated frame.

[10]https://www.orah.co/software/videostitch-studio/
[11]http://www.kolor.com/autopano-video/
[12]https://www.orah.co/software/vahana-vr/

A crucial part of the feature based techniques is the feature extraction. There are several feature detector methods, such as the Scale-Invariant Feature Transform (SIFT) and the Speeded Up Robust Features (SURF). The extracted features are then matched between images [31].

The stitching process is performed in three main steps: registration, calibration, and blending. The registration is the step in which the images are compared and aligned. This is the step where the direct or feature-based techniques are used to match the images. Once the global correspondence between the images has been established, the geometric transformation between them can be estimated.

After registration comes the calibration step in which adjustments are made to attempt to minimize the differences between an ideal lens model and the camera lens used to take the video.

Finally, the blending step involves stitching the individual feeds together as well as executing the adjustments found in the calibration. In this step, the pixels are mapped to a surface and then are blended, for example, by using a weighted average [51].

2.3.3 File Format

360 videos do not have specifically developed video formats. Their file format is the same as conventional videos since 360 video files can be thought of as common videos that are processed by new technological means [43]. As previously mentioned 360 video after recorded is generally stitched to transform it into an equirectangular format that can be mapped onto a 3D object, predominantly a sphere.

The most commonly used file format is the MP4, encoded with MPEG4 or H.264. Other common formats are the Webm, FLV, MPEG, MKV, MOV, among others [42].

In video players and browsers, for the 360 video to be spherically displayed, rather than shown in its equirectangular format, the information that the video is 360 degrees needs to be available. This information is typically kept in metadata that must be associated with the video [55][18]. The metadata will be able to specify how to project the video, the viewer's initial perspective, and other information useful for displaying the video. The perspective can then be altered by the viewer through, for example, mouse control or by a mobile device's gyroscope and accelerometer.

The 360 video's metadata can be simply a part of the video file. For example, Google has a metadata scheme[13] by which MP4 (ISOBMFF) and WebM (Matroska) multimedia containers may accommodate spherical videos.

Relatively to the audio, as referred above, 360 video tend to be better perceived when accompanied by spatial audio. Ambisonics is a method for recording, mixing and playing back three-dimensional 360-degree audio. Its basic approach is to treat an audio scene as a full 360-degree sphere of sound coming from different directions around a center point where the microphone is placed while recording. Ambisonics B-format is the most

[13] https://github.com/google/spatial-media/blob/master/docs/spherical-video-v2-rfc.md

popular Ambisonics format used in 360 video. It uses a 4-channel format to reproduce a complete sphere of sound. A big advantage of Ambisonics is that sound is not represented just on a horizontal dimension, it is also represented as coming from above or below [56].

2.4 Evaluation

When creating interactive systems there is the need for assessing the designs and testing the systems to ensure that these actually behave as expected and meet user requirements. This is the role of evaluation. Evaluation has three main goals: to assess the extent and accessibility of the system's functionality, to assess users' experience of the interaction, and to identify any specific problems with the system [11].

When studying the usability and user experience of VE systems, the study of the experience of presence is essential [24]. A problem often associated with VEs is the existence of motion sickness. Thus, in the evaluation of this type of systems it is common to analyse this specific problem.

2.4.1 Presence

There are several definitions and theories on presence. Research has been made to determine what are the factors that contribute to presence and to develop methods for measuring it.

Presence has been considered, for example, as: realism (the extent to which a medium can seem perceptual and/or socially realistic); transportation (sensations of ”being there”); immersion (extent to which the senses are engaged by the mediated environment); social actor within medium (Extent to which the user responds socially to a representation of a person through a medium); and others. When related to immersive VR, presence is most often characterized by concept of presence as transportation [35].

The phenomenon of feeling the sensation of inhabiting a simulated space as been described by the terms presence and immersion. Both terms have been widely used, yet there seems to be a lack of consensus as to what either refers to. Sometimes the terms are used interchangeably, others are given complementary meanings, and others are even given conflicting meanings [5].

A possible definition defended by M. Slater [38] is that immersion is simply ”what the technology delivers from an objective point of view” and presence is the ”human reaction to immersion”. In 1997, together with S. Wilbur, Slater had defined immersion as ”an objective and quantifiable description of what any particular system does provide” and presence as ”a state of consciousness, the (psychological) sense of being in the virtual environment” [40]. This idea conflicts with B. G. Witmer and M. J. Singer's, whose use of immersion is very similar to Slater's use of presence.

Witmer and Singer [57] consider that presence ”is defined as the subjective experience of being in one place or environment, even when one is physically situated in another” and believe that when applied to a VE ”presence refers to experiencing the computer-generated environment rather than the actual physical locale”. They also mention that both involvement and immersion are necessary for experiencing presence, considering that involvement in a VE depends on focusing one’s attention and energy on a coherent set of VE stimuli and immersion depends on perceiving oneself as apart of the VE stimulus flow. Though the factors underlying involvement and immersion may differ, the levels of immersion and involvement experienced in a VE are interdependent. That is, increased levels of involvement may lead users to experience more immersion in an immersive environment and vice versa.

Presence is subjective, therefore it is not easy to objectively measure it. B. G. Witmer and M. J. Singer created a presence questionnaire (PQ) in order to measure the degree to which individuals experience presence in VEs and the influence of possible contributing factors. They also developed an immersive tendencies questionnaire (ITQ) to measure differences in the tendencies of individuals to experience presence. The PQ and ITQ use a seven-point scale format that is based on the semantic differential principle.

For the creation of the PQ, B. G. Witmer and M. J. Singer used factors believed to underlie presence. These factors are grouped in four categories: Control Factors, Sensory Factors, Distraction Factors, and Realism Factors. The factors may exert their influence on presence by affecting either involvement, immersion, or both. The control factors relate to how much and how intuitively a person can control the VE. After performing cluster analysis the following data-driven subscales were identified: involved/control; natural; interface quality. The subscale labels were chosen based on the factors initially associated with the items from each subscale.

In posterior work in 2005 B. G. Witmer and M. J. Singer [58] developed an updated version of the PQ which includes auditory items and haptic items. A principal-components factor analysis of the core items of this updated version resulted in an update of the previously considered factors. The adaptation/immersion subscale was an important addiction that proved the existence of a immersion factor. After further analysis the factors and subscales got updated to: involvement; adaptation/immersion; sensory fidelity; and interface quality [58].

PQ has been widely used in the study of presence in immersive technologies. It has been adapted to test different systems and translated to other languages such as the adaptation to brazilian portuguese by Silva, G.R. et al [37].

Aila Kronqvist et al. [24] believed PQ to be excessively long and arduous to complete. They used the B. G. Witmer and M. J. Singer’s PQ as basis for the creation of a faster and easier to fill questionnaire which is used to evaluate the authenticity of VE experience. The authenticity index consists of a questionnaire designed to measure immersion, control, and the side effect of simulator sickness. The immersion factors are the feeling of presence and anticipated affordances compared to fulfilled affordances. The control ones are feeling of control, discovery ratio and amount of technical problems experienced. For measuring simulator sickness, a modified simulator sickness questionnaire was constructed. The three sum variables (immersion, control, and simulator sickness) are combined into an index of authenticity by using a principal components analysis (PCA).

M. Slater and colleagues developed a questionnaire - the SUS - over a number of studies. It is based on several questions, all variations on one of three themes: the subject's sense of "being there"; the extent to which the VE becomes the dominant reality; and the extent to which the VE is thought of as a "place". The presence is measured as the number of questions that receive a high score [53].

Schubert, T. et al [34] have also created a scale for measuring the sense of presence experienced in a virtual environment (VE) - the Igroup Presence Questionnaire (IPQ). The IPQ has three main factors: spatial presence; involvement; and experienced realism. It also has an additional general item that assesses the general "sense of being there". Some of the IPQ items are from previously published questionnaires such as B. G. Witmer and M. J. Singer's work and Slater et al.'s. IPQ was originally created in german but has been translated to different languages such as english, french, japanese and portuguese.

Other presence questionnaires have been created, some of which are based on the already referred ones. Presence questionnaires can be designed specifically for immersive VR or for a wide range of media.

Although measuring presence is done almost exclusively via questionnaires, some objective measures can also be used. A mean of obtaining a behavioral objective measure is examining a person's reaction to mediated stimuli. Reflex responses can be measured, for example. Measuring subjects' responses to virtual cues when they are also presented with conflicting real cues is another form of obtaining a behavioural objective measure. Physiological measures are also a form of objective measures. Skin conductance, for example, has demonstrated to have a correlation with presence [35].

2.4.2 Motion Sickness

"Motion sickness is an aversive behavioral state that affects several psychophysiological response systems" [14]. Several symptoms can be experienced due to motion sickness.

Peter J. Gianaros et al. [14] mentioned that motion sickness (MS) may be more appropriately quantified as a multidimensional syndrome rather than a univariate symptom, and more appropriately analysed via a questionnaire that provides a score for each of its dimensions. For that reason they developed the motion sickness assessment questionnaire (MSAQ) which can be used both to assess the overall experience of MS (using total scores) and the distinct dimensions of it (using subscale scores). This dimensions, obtained through factor analyses, are: gastrointestinal, central, peripheral, and sopite-related.

Robert S. Kennedy et al. [22] refer that the symptoms associated with simulator sickness (SS) tend to be less severe and to originate from elements of visual display and visuo-vestibular interaction divergent from those that induce the typical MS. Before the creation of questionnaires specific for SS, most studies indexed SS severity with some variant of the Pensacola Motion Sickness Questionnaire (MSQ), which consists of a list of symptoms related to the MS. However that was not the best way for measuring it since the patterns of symptom presence and severity associated with SS are sufficiently different from those of motion sickness to justify the use of separate measuring systems. Hence, Robert S. Kennedy et al. developed a simulator sickness questionnaire (SSQ).

The SSQ was derived from the MSQ using a series of factor analyses. After analysis, the three factors obtained and used as the basis for the SSQ subscales were: oculomotor; disorientation; and nausea. Each of these factors can be seen as a dimension that operates on a different target system of the human organism. A simple index of total severity (TS) can be computed from the sum of the three subscale scores and used to assess the overall extent of symptom severity. The subscales can help discover what target system is experiencing problems and what is the nature of a possible solution.

SSQ helps determine not only if a system has problems and what SS symptoms the system causes, but also what type of characteristic of a system might be the cause of the symptoms. Susan Bruck and Paul A. Watters [4] used the SSQ to pin out which symptoms experienced while being on a VE result from simulated motion rather than simply from being inside the VE. The SSQ was used to compare the responses in case of immersion in a VE with low simulated motion with the responses from high simulated motion condition. The study showed that most symptoms have significant increase as result from the simulated motion, only those associated with gastric activity and vestibular activation do not.

On the evaluation study to determine whether 3D 360 video enhances the user's VR experience by Bessa, Maximino, et al. [2], previously mentioned in the 360 video section, both a portuguese version of the IPQ and the SSQ were used for measuring the presence and SS respectively. These measures enable the comparison of the use of 2D and 3D 360 videos.

2.4.3 Heuristic Evaluation

A heuristic is a "rule of thumb that can guide a decision" and heuristic evaluation "is a method for structuring the critique of a system using a set of relatively simple and general heuristics" [11]. Nilsen and Molich [27] refer that heuristic evaluation is "an informal method of usability analysis where a number of evaluators are presented with an interface design and asked to comment on it". The evaluators should test the interface and evaluate it according to certain rules. Nielsen has created a set of ten usability heuristics [14] to be used when evaluating usability.

Studies have shown that the interaction styles of VE systems are radically different from standard graphical user interfaces (GUIs). A. Sutcliffe et al. [47] developed a heuristic method based on Nielsen's but specific for evaluating VEs. They present twelve

[14] Heuristics available at: https://www.nngroup.com/articles/ten-usability-heuristics/

heuristics, some derived from Nielsen's and some motivated by questionnaire-based techniques, that address usability and presence issues. The heuristics were motivated by the different nature of VEs, considering in particular characteristics such as the need for intuitive interaction and the sense of immersion.

The evaluation method proposed by Sutcliffe et al. follows Nielsen's approach but adding a technology audit, executed during the familiarization period. In this phase the evaluator explores the VE and notices which features are present in it. These results are used to calibrate the judgement for different techniques and to adjust the heuristics to the different styles of the VE, removing the heuristics that are not relevant to the VE being tested. After completing the technology audit, the evaluator executes a set of user tasks noticing any problems. These problems are then associated to the heuristics and to design features responsible for them. Finally the evaluator judges the severity of each error using the following scale:

- **severe** - would make completing the task successfully impossible;
- **annoying** - would disrupt the user's task but most users would learn how to cure the error given an explanation;
- **distracting** - would disrupt the user's task but most users would discover the fix relatively quickly given a hint;
- **inconvenient** - could disrupt the user's task but most users would discover the fix unaided.

The severity of the errors associated with each heuristic is also rated according to the amount and severity of the errors. The rankings resulting from the heuristic evaluations besides providing a summative evaluation of the VE, can also be used to uncover what parts of the design need improvement more urgently and may help identify how to accomplish that improvement.

Chapter 3

Design and Implementation

As previously referred in chapter 1, throughout this thesis was developed a virtual reality system that allows the immersion of a person in the world of paralympic sports, particularly, of wheelchair basketball. This chapter describes the decisions made regarding the system and the development platforms used, as well as the environment and interactions implemented.

The chapter ends with the description of the techniques used in the attempt to integrate 360 video with the computer-generated elements.

3.1 Design

The system is composed by both 360-degree videos and computer-generated virtual elements with which a user can interact. It can be thought of as being composed by two parts: one in which users are presented with information through 360-degree videos of, for example, trainings or interviews; and another in which the users can interact with the virtual environment around them and try practicing the sport in the role of an athlete.

As previously referred in section 2.3 of the related work, 360 video can be either mono or stereoscopic. Since studies have shown no significant differences between the two in terms of presence, and stereoscopic is harder to achieve, the videos used for the system are monoscopic.

Regarding the interactive part, in which the user can try playing the sport, there are two possible perspectives that can be used for the user's visualization: the first person perspective (1PP) and the third person perspective (3PP). As explained in 2.1.2, 1PP is better to achieve immersion and give a sense of being in the athlete's shoes, since it allows to experience the environment from the point of view of the athlete's eyes. However, this perspective does not give a good sense of the whole environment surrounding the user since it is hard to correctly estimate distances, and the field of view is limited. On the other hand, 3PP helps provide a more global view of the environment around the player which can be advantageous, for example, for distance estimation. This perspective is better for sports training since it leads to the development of correct techniques that are more easily transferred to the real world. However, it is not as good as 1PP in terms of the feeling of immersion.

Given the main goal of this project, the priority is to make the user feel like he/she is the athlete, putting him/her seeing through the athlete's eyes. Therefore, the sense of immersion is of great importance. Furthermore, since the system is not meant to be used for the training of the athletes, there is no need for the skills obtained in the virtual environment to be transferred to the real world. This means that even if the user can not correctly estimate distances and the behaviour of elements as a result of a user's action is not exactly the same as in reality, it is not an issue. Thus, the interactive part was created to be experienced from a 1PP point of view.

Since the user will experience the virtual world from a 1PP, and to increase the feeling of immersion and help the user feel like he/she truly is an athlete, it was decided that the user would have a virtual body.

Regarding the hardware for the display of the virtual reality, the one employed was the head-mounted display (HMD) HTC VIVE. With this the user has a headset and two controllers that are used to track his/her movements.

As mentioned in 2.1.3, HMDs can cause problems in sports virtual environments due to the weight and possibility of causing constraints to the user's movements. However, since the sport simulated in this thesis is wheelchair basketball, the user can play while sitting down considering that being in the same position as the athlete would make the user's situation more similar to the real one and hence increase the sense of immersion. Therefore, there probably will not be many wide movements, and the constrain will not be significant.

A possible alternative to the HMD would be a CAVE, however, this type of display technology needs to be tailor-made, occupies a lot of space, and is expensive. HMDs are cheaper and more compact, which makes them convenient for transportation. This is useful, especially for the testing stage.

It was decided that the system would present the following functionalities:

- Allow the user to play wheelchair basketball from an athlete's perspective
- Allow the user to control the athlete's virtual body
- Allow the user to move through the basketball court
- Allow the user to interact with a virtual wheelchair, pushing it around the court
- Allow the user to interact with a ball, grabbing it and throwing it so that he/she may attempt to score

3.2 Implementation

The system's environment, elements, and functionalities had to be implemented. Therefore, rose the need to obtain resources and select the technologies required for the development of the system.

The techniques of integration of 360 video with computer-generated elements were also implemented, and are described in 3.2.5.

3.2.1 Technologies

The software chosen to implement the system was Unity[50], which is a powerful cross-platform game engine and development environment, primarily used to develop video games and simulations. It provides a base API and feature set with compatibility for multiple devices, including virtual reality devices.

In Unity, a system can be composed by one or more scenes. Each scene can be viewed as being a piece of the system. For example, when creating a game, each level of that game will probably be a different scene, containing its own environment and menus.

Within each scene, there can exist several GameObjects. These are the fundamental objects in Unity that represent characters, props, and scenery. Each GameObject contains components which control its behavior. Besides Unity's built-in components, such as, for example, the Transform Component which controls the GameObject's position, rotation and scale, new components can be created using scripts. A script is a specific type of component that is created by the developer and adds functionality. Unity's scripts can be written in either C# or JavaScript. All scripts created in this thesis are in C#.

The SteamVR plugin for Unity, developed by Valve, permits the development of VR applications for HTC VIVE. It provides prefabs, scripts and other assets that allow to see the scene and the tracked controllers through the headset, as well as make available several other functionalities. This plugin is essential to be able to experience the created system using the head-mounted display.

For the creation of some of the 3D elements needed for the system, the open-source 3D computer graphics software Blender was used.

The stitching of the 360 videos was made using Samsung: Gear 360 ActionDirector, a 360 video editing software designed specifically for the Samsung Gear 360 camera. The reason for using this software is because the videos available were recorded using this camera, and because the stitching is performed automatically.

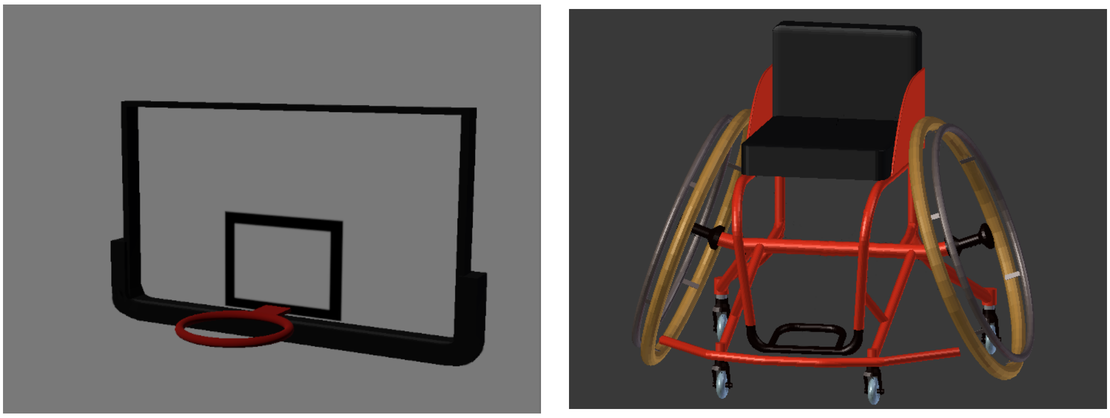
Adobe After Effects, an Adobe software capable of creating motion graphics and visual effects, was used in the manipulation and editing of some of the 360 videos. The video editing software Sony Vegas Pro was also used, but mostly for the video's audio editing.

Adobe Photoshop, an image editing software, was used to help covering the tripod that appeared in some of the 360 videos.

Finally, OpenCV (Open Source Computer Vision Library), a software library designed for computational efficiency that provides several efficient tools for image processing, was applied in processing videos, as further explained in 3.2.5.3.

3.2.2 Resources

For the development of the system, some resources were required, such as virtual 3D elements, 360 videos, and audio. Since the system is a game that simulates wheelchair basketball, the virtual 3D elements needed were a basketball ball, a backboard, a wheelchair, and also a traffic cone [1]. The backboard (fig. 3.1a) and wheelchair (fig. 3.1b) were created using the Blender software. Besides the mentioned objects, virtual 3D characters were also needed to represent the bodies of the athletes. These characters were obtained from *https://www.mixamo.com*, a website that provides both rigged characters and animations for those characters.



(a) Backboard

(b) Wheelchair

Figure 3.1: 3D computer-generated elements created with Blender

Regarding the 360 videos, several recordings of a practice of the Portugal national team at *Centro de Alto Rendimento* in *Vila Nova de Gaia* were provided by Caroline Delmazo of *Faculdade de Ciências Sociais e Humanas*, NOVA University of Lisbon. The video "Paralympic VR 360 video" [2], edited by Caroline as part of her master's project, was also provided and is present in the system developed.

As mentioned in 2.1 of the related work, audio is an important sense to target in order to increase immersion. The audio used for background sound in the system's practice scenes was made using the audio of the 360 videos. Pieces of audio of different videos were put together using Sony Vegas Pro. The sound of a basketball bouncing was obtained from *https://freesound.org/*, and is used for when the virtual ball collides with an object of the virtual environment.

[1] The basketball ball and traffic cone's 3D models were obtained from *https://www.turbosquid.com/*

[2] Available at: *https://www.youtube.com/watch?v=eW_e02pSaG4*

3.2.3 Navigation

The created system is composed of several scenes and the user can use menus to change the one he/she is in. When the system starts, the first scene the user is presented with is the "Moving Video Tracking" scene. In it, the user will see a small 360 video clip of a basketball practice, recorded by a camera placed in a wheelchair moving through the basketball court. This scene has an interactable computer-generated basketball ball that follows the video's motion. The user can interact with this ball by pointing at it with the controller and pressing the trigger button (fig. 3.5) to select it, thus activating the action associated with it. Once selected, the user goes to the "Main Menu" scene.

From the "Main Menu" the user can go to any of the scenes in which he/she can play the game or view a video by using the menu present in this scene (fig. 3.2). There are five scenes the user can select to go to from the main menu: two video scenes and three practice scenes. The practice scenes are the ones in which the user is in an athlete's shoes and can practice playing basketball or controlling the player's wheelchair.

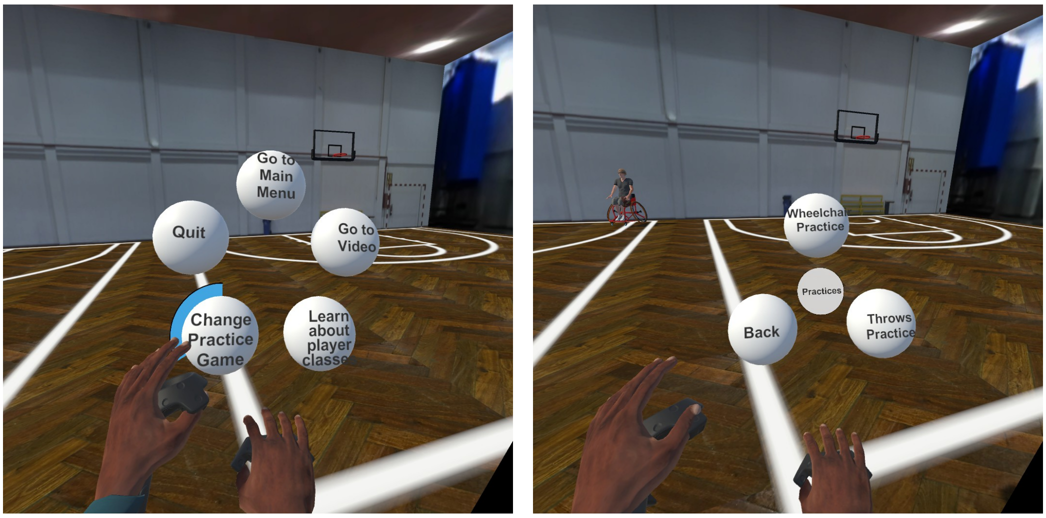
Figure 3.2: Full Practice scene being selected in the main menu

The practice scenes are: the "Full Practice" scene, the "Throws Practice" scene, and the "Wheelchair Practice" scene. In the "Full Practice" the user can move around the court, interact with the ball and the wheelchair, and interact with a non-player character that is helping the user practice basketball. After the initial user tests that are mentioned in section 4.1, it was concluded that there should be the possibility of practicing just the ball throws and just the wheelchair control individually. Therefore, the ”Throws Practice” and the ”Wheelchair Practice” scenes were created. In the first, the user is only capable of being in certain positions of the court, from which he/she can practice throwing balls to the basket. In the second, the user can only move through the court by interacting with a virtual wheelchair.

Whenever the user chooses to go to the ”Full Practice” scene, he/she is first presented with another scene in which a small 360 video clip of the empty court is played. This is further explained in section 3.2.5.1.

The video scenes are: the ”Paralympic VR Video” scene, and the ”Classification Video” scene. In the first one, the user can watch the already mentioned video made by Caroline Delmazo. In the second one, the user will watch and be able to interact with a video in which the wheelchair basketball classification, based on the players’ limitations in functional skills, is explained.

When in one of the video or practice scenes, the user can choose to change to another of these scenes or go back to the main menu one, by using a menu that is available by pressing the application menu button of the controller (fig. 3.5). Every time this button is pressed in one of the controllers, the menu appears above that controller (fig. 3.3a). If it is pressed in a controller that already has the menu attached, then the menu disappears.



(a) Selecting the option to go to a different practice scene

(b) Sub-menu to choose the practice scene to go to

Figure 3.3: Menu and Sub-menu

Four scripts were created to manage the menu: the *MenuManager.cs*, the *MenuNavigation.cs*, the *RadialMenu.cs*, and the *MenuButton.cs* script. The *MenuManager.cs* is responsible for enabling and disabling the menu, and attaching it to the right controller.

The menu may have sub-menus (fig. 3.3b). Therefore, there is the need to control which menu options are displayed. This control is of the responsibility of the *MenuNavigation.cs* script. Whenever the user selects a menu option that opens a different sub-menu, this script enables that sub-menu and disables the previous one so that only the right one is visible.

Both the menu and sub-menus are radial menus, in which the options are displayed in a circular arrangement. The *RadialMenu.cs* script calculates the position each menu option must be in and places it there.

Each of the GameObjects that represent a menu option has, as a component, the *MenuButton.cs* script. One of the properties of this component is which action will be performed when that menu option is selected. The actions available are:

- **Change Scene** - Loads a new scene whose name is also a property of the component.
- **Change Menu** - Changes which sub-menu is displayed.
- **Quit** - Exits the system.
- **None** - No action.

To select a menu option the user just needs to touch that option with the controller the menu is not attached to. After a period of time touching an option, during which a radial progress bar gets filled, that option is selected.

The main menu is similar to the menu of the other scenes, but is not displayed in a circular arrangement, is not attached to a controller, and instead of an option being selected by having the user touch it, it is selected when the user points at it with the controller and presses the trigger button. Raycasting is used to create this interaction with the GameObjets of the menu options by allowing to know which option is being pointed to.

Whenever the user goes to a different scene, the *LoadLevel.cs* script is the one responsible for the loading of that scene and for showing either a loading image or just pitch black, while changing between scenes. The option of displaying the loading image is used for when the scene being loaded takes a long time to load. This script is further explained in 3.2.5.1.

3.2.4 Virtual Environment and Interaction

For the development of the system, there was the need to construct the virtual environment, including the elements present in it, and to implement the interaction with those elements, as well as other functionalities.

3.2.4.1 Virtual Body

As previously mentioned, when the user is playing the game he/she will be in an athlete's shoes, seeing the world from his eyes. To make this happen the user will have a virtual body. This body will be a character, sitting in a virtual wheelchair, that the user will be able to control.

In wheelchair basketball, athletes have different functional capabilities and are classified according to their limitations in functional skills. The virtual body in this system is that of an athlete with few limitations, who has a wide range of motion. Therefore, this virtual body will be able to mimic most movements performed by the user, which will help increase the user's sense of ownership over the body.

As already referred in section 2.1.2 of the related work, in order to make the user have a sense of ownership over the virtual body, and therefore increase the feeling of immersion in the environment, it is essential that this virtual body moves synchronously with the user's real body. Another factor that helps set the illusion of ownership of the virtual body is perspective. If the virtual body is perceived to be in the same position as the real one is expected to be in, the user's brain will be tricked into believing that the body must be the user's own.

Since the virtual game is meant to be played while sitting down, the user's real body will be in the same posture as the virtual body of the athlete who is sitting in the wheelchair. Therefore, to make the body appear to be in the same perspective as the user's own, it is sufficient to make the virtual body appear in the position of the virtual world that corresponds to the one the user will be sitting at in the real world.

Making the virtual body move synchronously with the real body is a bit trickier considering that, due to the hardware used, the only input available regarding the real body's condition is the position of the head and of the two controllers which are expected to be in the user's hands. So, to try to make the movements as similar as possible, it was necessary to make the virtual head move with the real head, and the virtual hands move with the controllers the user holds.

To make the virtual hands and head follow the positions provided by the input, the use of inverse kinematics (IK) is required to have each joint of the skeleton of the game character be oriented in a valid way that allows the positions to match the ones provided.

Unity supports the use of IK in characters. However, the IK built into Unity can only be used to control the arms and legs, affecting only the limb in question and, therefore, having no effect in the body's torso or other limbs. Thus, to be able to apply IK to the head and also be able to move the torso according to the motion of the hands and head, a full body IK was needed. Hence, the asset SAFullBodyIK [3], which is a full body IK component, was used. The script *IKVirtualBodyControl* was also created to attach the virtual body parts' positions to the ones provided by the input.

Regarding the control of the virtual hands, since the controllers have few buttons, the hand control available is limited. Hence, the only motion of the virtual hands it was decided to implement was the open and closing of the hands, which is controlled by the pressing of the trigger button (fig. 3.5). For this, the *handsClosing.cs* script was created.

[3] Available at: https://github.com/Stereoarts/SAFullBodyIK

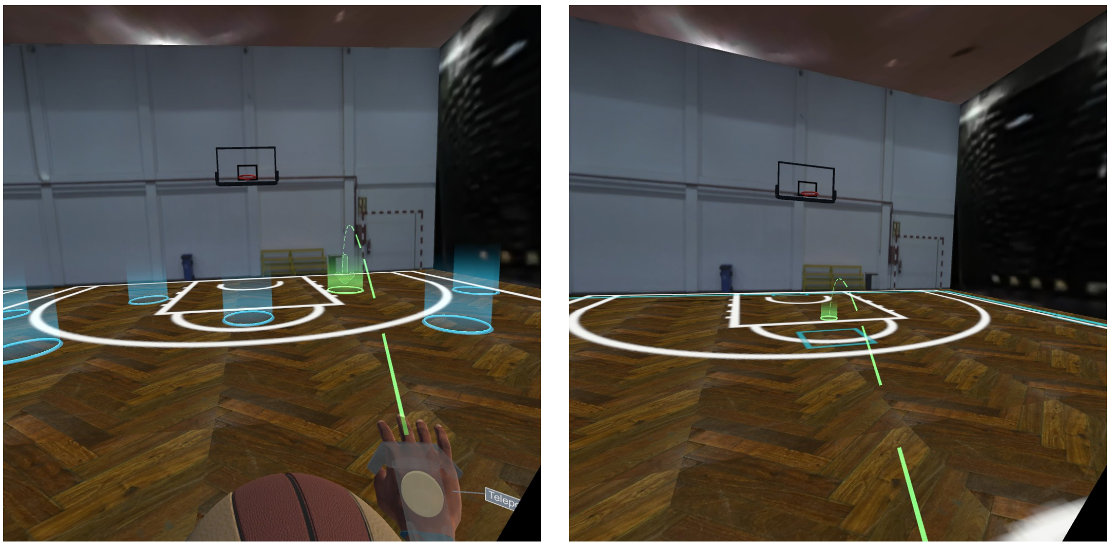
This script reads how far down the trigger button of each controller is pressed, and uses the corresponding value to control the animation of the matching hand. The value is passed to the animator through an animation parameter, which controls the blending between two animations: one of the hand open and one of the hand closed. Both animations have just one frame that simply determines the position of the hand when it is, respectively, open or closed.

3.2.4.2 Locomotion

Once the user is capable of controlling the virtual body of the athlete, next it becomes necessary for him/her to be able to move through the virtual environment. Otherwise, he/she would be stuck to a position and the game would have very limited functionality.

Two ways of locomotion were implemented to move through the virtual court: virtual wheelchair maneuvering and teleport.

The type of locomotion available varies with the practice game the user is playing. In the wheelchair practice scene, since the goal is for the user to train maneuvering the wheelchair, that is the only locomotion type he/she will be able to use. In the ball throwing practice scene, since the idea is to only practice the throws and, therefore, the user does not need to move around the court, only the teleport is available, and only for specific locations (fig. 3.4a). Finally, in the full practice scene, in which the user is expected to freely experience practicing the basketball game, both methods of locomotion are available and, when teleporting, he/she can do it to any location within a certain teleport area which covers the court (fig. 3.4b).



(a) Throws practice　　(b) Full practice

Figure 3.4: Teleport

The SteamVR plugin provides a teleport system which supports teleporting to specific teleport points or within a general teleport area. This system was adapted and used in the game created in this thesis.

By using the teleport system provided by SteamVR, when a user presses the touchpad of one of the controllers a teleport pointer shows up in the scene. If when the touchpad is released the pointer is pointing at a valid spot, then the player teleports. This allows the user to change position, but not to rotate. The player's body will still remain turned to the same side. Therefore, the system was adapted to also permit the user to turn while physically remaining sitting still in the same place. Changes were made to the *Teleport.cs* script to allow the user to do this.

When the touchpad is pressed it is possible to obtain the coordinates of the click. The coordinates have an x and a y value that are both between -1 and 1. The changes to the teleporting script made it so that when the point of the touchpad pressed by the user has an x value equal or under -0.6 the player is rotated 30 degrees to his/her left, and when the value is equal or above 0.6 the player is rotated 30 degrees to his/her right (fig. 3.5).

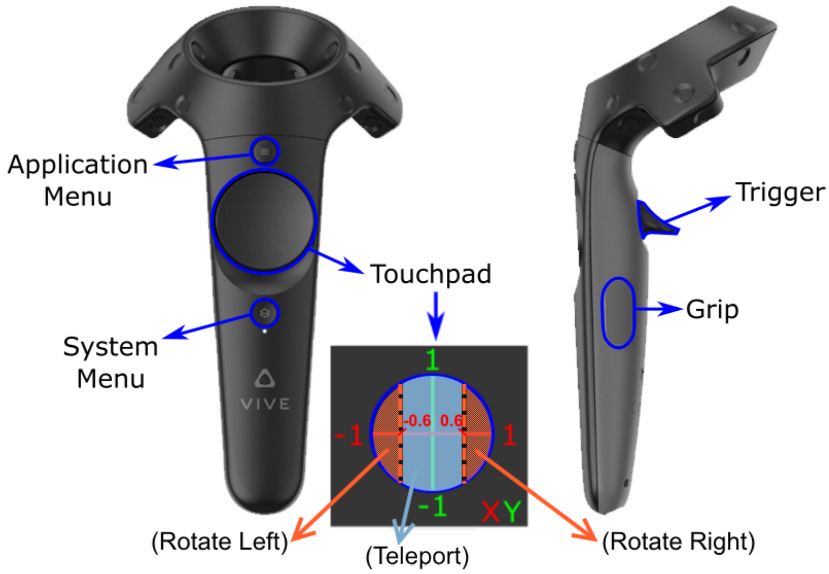


Figure 3.5: Controllers' buttons. Adapted from: *https://www.raywenderlich.com/792-htc-vive-tutorial-for-unity*

Another adaptation made to the *Teleport.cs* script was to make the player's wheelchair be teleported together with him, instead of being left behind.

SteamVR's *CircularDrive.cs* script was used as base for the wheelchair maneuvering locomotion. This script allows to move an object in a circular motion when the user uses his/her hand to interact with it. The script was adapted to permit a wheel to rotate around itself, since in the original script an object could not rotate around itself when it had an inclination and the wheelchair's wheels are bent 20 degrees to the ground. It was also adapted to calculate and apply a force to the wheelchair to make it move according to the movement of the wheels. The adapted script was named *CircularDriveAdapt.cs*, and was placed as a component in each of the wheelchair's driving wheels.

The adaptation also permits the wheel to keep correctly rotating around itself as the wheelchair moves through the court, changing the wheel's rotation and position in the world space. In the original *CircularDrive.cs* script, the position of the hand in the wheel and the movement it makes while holding it is obtained in the world referential. Therefore, since the wheel moves through the world, its rotation would get incorrectly calculated. Hence, in the adapted script, a referential local to the wheel is used to determine the position the hand is in in relation to the wheel and, consequently, the movement that the hand makes while holding the wheel. Since this way the values are all obtained relatively to the wheel and not the world, the calculus are correct even when the wheel moves through the world. This local referential is a referential that remains the same, relatively to the wheelchair, as the wheel's actual local referential rotates with the wheel. If the wheel's actual referential was used the movement of the hand would get incorrectly calculated since the referential would be changing as the hand moved.

As a wheel rotates, a force is applied to the wheelchair in the position where that wheel connects with it. The calculus of the force has in consideration the velocity and length of the wheel's displacement, and results from empirical observations. It is calculated by the following formulas:

$$x = \frac{diameter \times \pi \times \theta}{360}$$

$$force = \frac{x \times 0.75}{t^2}$$

In which θ is the angle of the wheel's rotation, x is the displacement of the wheel due to its rotation, and t is the time that occurred since the force was last calculated (time since last frame update).

In order to homogenize, smooth the transition between values and thus avoid big spikes of speed, the force actually applied to the wheelchair results from a weighted average that considers not only the already mentioned *force*, but also the last few calculated forces and a default force. That weighted average is obtained by:

$$appliedForce = 0.3 \times \frac{\sum_{i=1}^{Count} forces[i]}{Count} + 0.2 \times defaultForce + 0.5 \times force$$

In which *Count* is the amount of previous forces used in the weighted average, *forces[i]* are those previous forces, the *defaultForce* is a fixed force value, and *force* is the already mentioned calculated force.

As explained, when a user moves the wheels, the impulse given to it is transformed into a force that is applied to the wheelchair, thus making it move. For the wheelchair to roll through the court, and not stop moving too soon, the driving wheels and smaller wheels' friction had to be reduced. This way the wheelchair moves for longer, making the movement seem more natural, as if it is actually rolling.

After the preliminary user tests presented in section 4.1, haptic feedback was added to the wheelchair's wheels so that when a controller is over a wheel that controller vibrates and, thus, the user will know that he/she has a hand over that wheel without having to constantly look at it. This way, the user can focus on the environment around him/her instead of the wheels.

The virtual wheelchair maneuvering method was created with the goal of allowing the user to experience a more realistic way of moving through the environment, attempting to simulate a real athlete's way of locomotion. However this method is expected to possibly lead to motion sickness due to the lack of vestibular sensations or physical feedback provided. On the other hand, the teleport method, which is a point-and-click technique, despite not being a natural way of locomotion in the real world, has the advantage of not being likely to lead to motion sickness. It also makes it easier for the user to take big leaps through the environment, which the users might see as advantageous since they'll be able to move across the court field faster, even though that departs from the real experience.

3.2.4.3 Ball Interaction

Since the system allows the player to practice playing basketball, the possibility to interact with a ball is essential. The user must be able to hold and throw the ball. For this, some of the SteamVR's scripts were applied, such as the *Interactable.cs*, the *VelocityEstimator.cs* and the *Throwable.cs*. The *Interactable.cs* is used to identify the basketball ball as an object the player's hand can interact with. The *Throwable.cs* allows the basketball ball to the picked up by the player, attaching the ball to the player's hand, and when the player releases the ball this script applies a velocity to it. This throwing velocity may be obtained by using the *VelocityEstimator.cs*, which estimates the ball's velocity through its change in position, or by using the velocity of the controller used to hold the ball.

Originally, the velocity applied to the ball by the *Throwable.cs* script included the cross product of the angular velocity applied to the ball for the vector that gives the position of the ball's center of mass in relation to the controller. However, the script was adapted, removing this cross product and leaving only the estimated velocity already mentioned. This change was made in order to improve the trajectory of the ball when thrown, and resulted from empirical observations. The adapted script was named *ThrowableAdap.cs*.

The full practice scene has only one basketball ball, which the player can freely interact with. In the ball throwing practice scene, however, the player has a static ball always in front of him/her from which new balls are spawned. Whenever the user tries to grab the static basketball a new ball is spawned and that new ball is the one that will be attached to the player's hand. The spawned balls, after dropped or thrown, are eventually destroyed from the scene.

For this spawning and destroying of basketball balls two new scripts were created: the *BallSpawning.cs* and the *BallDestroy.cs*. This first script, placed as a component in the static basketball ball GameObject, creates and attaches to the player's hand a new ball when that hand attempts to grab the static ball by pressing the trigger button while over it. Furthermore, the *BallDestroy.cs* script, which is a component of every basketball ball spawned by the *BallSpawning.cs* script, checks the ball's velocity and if it is being held by the player. If the ball is not attached to one of the player's hands and its velocity is under a certain value, it is suspected of needing to be destroyed. If after 2 seconds the ball is still not attached to a hand and with low velocity, then it is considered that the ball is no longer necessary and is, therefore, destroyed.

The player can grab a basketball ball by pressing the trigger button of a controller which is over that ball. The ball becomes attached to the player's hand and will remain so until the trigger stops being pressed. Therefore, to release the ball, detaching it from the player's hand, the user just needs to release the trigger button. If he/she does so while giving impulse to the ball, the velocity applied to it as a result will lead to it being thrown.

A basketball ball can only be attached to one hand at a time, which means that the user won't be able to catch or hold it with both hands. This was pointed out as a problem in the initial user tests mentioned in section 4.1, since most people will instinctively try to catch a ball thrown their way with both hands, and attempt to throw it also by resorting to the use of both hands. Another problem revealed at said user tests is that if a user can interact with a ball only by seizing it, then it will be difficult to dribble it. Due to the risen issues, it was decided that the player's hands should be able to, not only grab, but also collide with the balls. That way the player can dribble the ball by hitting it with his/her virtual hand and use the hand that is not attached to the ball to help support it when performing a throw. However, this raises the problem of how to control if the ball is supposed to collide or not with the hand at the moment they come into contact. If the user wishes to catch the ball but it collides with his/her hand than he/she won't be able to grab it. Hence, the *HandCollision.cs* script was created to both allow and manage the collisions.

When the trigger button of a controller is being pressed, even if just slightly, the *HandCollision.cs* script disables the possibility of a collision. Thus, if a ball touches the hand it won't collide and, therefore, the user will be able to grab it. On the contrary, if the trigger button is not being pressed then the ball will collide with the hand.

For it to be realistic, when the player uses a hand to which the ball is not attached to help throw the ball by giving an impulse to it, this hand should apply a force to the ball. Therefore, the *HandCollision.cs* script uses a *VelocityEstimator.cs* component to estimate the velocity granted to the ball as a result of the movement of the hand while it is in contact with it. This velocity value is used to apply a force to the ball at the moment it stops being in contact with the hand, thus applying to it the impulse resulting from the hand pushing the ball.

In order to increase the feeling of immersion and sense of realism of the environment, an audio is played every time the ball collides with something. The *BallAudioPlay.cs* script is responsible for activating the sound every time there is a collision, making the volume of it depend of the relative linear velocity of the two colliding objects according to the function:

$$volume = \frac{velocity}{velocity + 1}$$

The audio's pitch depends on whether the ball collided with the player or another object, since the sound a ball makes when hitting the floor or a wall is different from the one it makes when hitting a human body. The original pitch is used for all objects other than the player.

3.2.4.4 Non-Player Character

In the full practice scene, in addition to the character controlled by the user, there is also another character, a non-player character (NPC). This NPC's behaviour is controlled by the *OtherPlayerController.cs* script.

Depending on the situation of the game at a certain point, the NPC can be in one of four different states: "Defend", "Pass Ball", "Fetch Ball", and "Move To Pass Ball" (fig. 3.6). The *OtherPlayerController.cs* script implements these states and manages the transition between them. In the "Defend" state the NPC tries to protect his backboard so that the player can not score. The "Fetch Ball" state is when the basketball ball is not in the possession of any of the basketball players and, therefore, the NPC moves towards it and tries to grab it. The "Move To Pass Ball" state is when the NPC has the ball in his possession, and is positioning and orienting himself in a way from which he can successfully pass the ball to the user. Finally, the "Pass Ball" state is the one the NPC is in while he is passing the ball to the user.

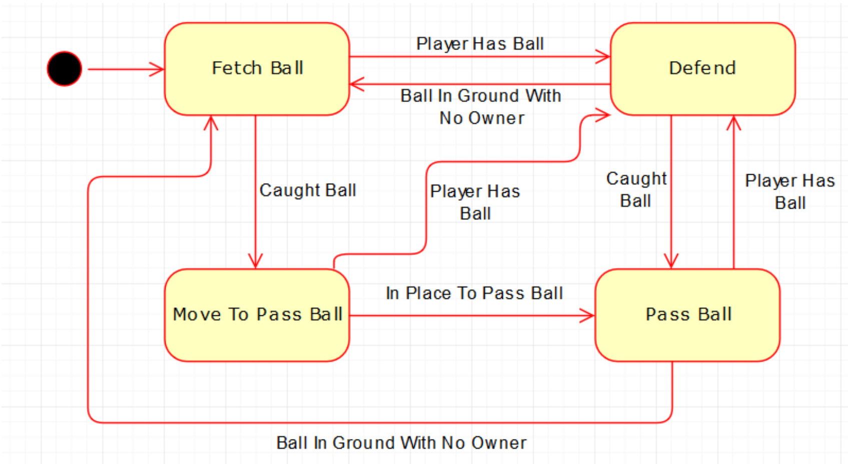


Figure 3.6: NPC's behaviour states.

Currently, the NPC is programmed to defend his backboard from the user, but every time he catches or grabs the ball he will pass it to the user, giving him/her, once again, a chance to score.

The movements of the NPC's body result from animations. The transitions between this animations are controlled by the *OtherPlayerController.cs* script. When the NPC is moving through the court, pushing the wheelchair, the animation of his body's movements is controlled by the *CharacterAnimation.cs* script. This script calculates the NPC's speed percentage, dividing his current speed by the maximum movement speed he can achieve. This speed percentage is passed to the animator of the NPC (fig. 3.7), which uses it in the blending between two animations: one of the character sitting idle, in which he moves his hands from the wheelchair's wheels to his lap, and one of him pushing the wheels forward.

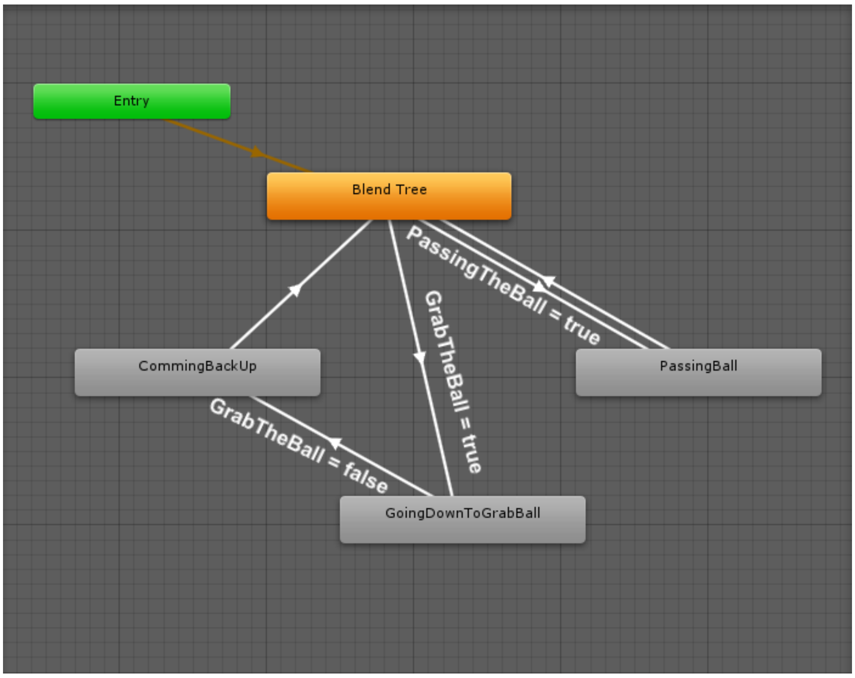


Figure 3.7: NPC's animator's state machine. *Blend Tree* is the state in which there is blending between the sitting idle and the moving forward animations.

In order to know in which state the NPC must be in, one information that is needed is if the basketball ball is in anyone's possession and, if so, in whose. Hence, the script *BallPossession.cs*, which is placed as a component of the ball, was created. Every time the ball is attached to a character's hand that character is considered to be in possession of it. Once the ball gets detached from the hand, and its distance from the one in possession of it surpasses a certain value, it is considered that the ball is no longer in anyone's possession.

When the ball is in the user's possession the NPC goes into the "Defend" state. In this state, the NPC keeps himself between the user and the backboard he is defending in an attempt to block the user's throws. The position he places himself at is obtained by the following expressions:

$$\vec{u} = backboar\vec{d}.position - player.\vec{p}osition$$

$$targ\vec{e}tPos = player.\vec{p}osition + \vec{u} \times 0.25$$

In which $\vec{u}$ is the vector that gives the position of the backboard being defended in relation to the user's position, and $\overrightarrow{targetPos}$ is the vector that gives the position the NPC must move to. Once he reaches that position, he turns to face the user's character.

Subsequently, the NPC will raise one hand attempting to block potential throws made by the user. The position the hand must be at to do this is calculated, and depending on the distance of that position to the right and left arms of the NPC, the closest one will be the one to be raised. The corresponding hand is put in the calculated position, using the IK built into unity.

The way the position of the hand is calculated will depend on the distance between the ball and the NPC. While the ball is still far, the position is one that attempts to block the path to the backboard (fig. 3.8), and is calculated by:

$$\overrightarrow{up} = (0,1,0)$$

$$\vec{v} = \overrightarrow{ball.position} - \overrightarrow{backboard.position}$$

$$\vec{n} = \vec{v} - Proj_{\overrightarrow{up}}\vec{v}$$

$$\overrightarrow{v2} = \overrightarrow{ball.position} - \overrightarrow{npc.position}$$

$$\overrightarrow{v3} = \overrightarrow{v2} - Proj_{\vec{n}}\overrightarrow{v2}$$

$$\overrightarrow{v4} = (\overrightarrow{v3}_x, defendingHandHeight, \overrightarrow{v3}_z)$$

$$\overrightarrow{handPos} = \overrightarrow{npc.position} + \overrightarrow{v4}$$

In which $\overrightarrow{up}$ is the vector normal to the Y=0 plane, $\vec{n}$ is the projection of the $\vec{v}$ vector onto the Y=0 plane, the $\overrightarrow{v3}$ vector is the projection of $\overrightarrow{v2}$ onto the plane whose normal is $\vec{n}$, *defendingHandHeight* is a defined value, and $\overrightarrow{handPos}$ is the position the hand is placed in. Also, $\vec{v}$ is the vector that gives the position of of the ball in relation to the defended backboard's position and $\overrightarrow{v2}$ is the vector that gives the position of the ball in relation to the NPC's position.

What this calculations do is obtain the normal of the plane orthogonal to the line that connects the (X, Z) positions of the ball and the backboard ($\vec{n}$), which is a vector that gives the direction of the ball in relation to the backboard. Then, the vector of the ball's position in relation to the NPC ($\overrightarrow{v2}$) is projected in the obtained plane, and the result of this projection gives the (X, Z) coordinates of the NPC's hand's position when defending the backboard. This is done so that the positions the hand can be placed in will be restricted to the obtained plane, placed over the NPC. The Y coordinate of the hand position while defending is a defined value so that said hand is always risen above the NPC's head at a height that seems natural for it to be in while trying to block the path to the backboard.

When the ball is within a certain radius from the NPC it is considered to be close enough for him to start trying to catch it. Therefore, the hand will move closer to the ball's actual position. At this point, the hand's position is obtained by:

Figure 3.8: NPC defending its backboard.

$$\overrightarrow{handPos} = \overrightarrow{ball.position} + \overrightarrow{ball.velocity} \times \vec{t}$$

In which $\vec{t}$ is calculated based on the distance between the ball and the NPC. The farther they are from each other, the higher the value of $\vec{t}$, and the farther ahead to where the ball is predicted to be in the future the hand will go. The closer they are, the lower is the value of $\vec{t}$, and the closer the hand will get to the ball.

Each of the NPC's hands has, as a component, the *OtherPlayersHand.cs* script. When a hand touches the ball this script is responsible for making the character grab the ball. It attaches the ball to an attach point of that hand, and informs the *OtherPlayerController.cs* script that the ball has been caught.

When the NPC is in the "Defend" state and catches the ball he transitions to the "Pass Ball" state. In this state, the *OtherPlayerController.cs* script changes the NPC's animator's parameter *PassingTheBall* to true, thus activating the animation of the character passing the ball (fig. 3.7). In this animation the character makes a throwing motion, appearing to push the basketball with both hands. Once the moment in which a real player would release the ball is reached, the animation calls an animation event which calls the function *ThrowBall* of the *OtherPlayerController.cs* script. This function makes the *OtherPlayersHand.cs* component of both of the NPC's hands throw the ball if it is attached to it. It does this resorting to the use of the adapted *ThrowableAdapt.cs* script, previously mentioned in 3.2.4.3. In this script was created a function responsible for calculating the velocity with which the ball must be thrown in order to reach the user[4]. The formula for calculating this initial velocity was derived from the projectile motion formulas.

Since the only information the ball initially has about its throwing is the target position, which is a point in front of the user's character's chest, there are two unknowns: the launch angle and the initial velocity. Therefore, and since it is not possible to calculate both, the launch angle is defined as being of 40 degrees, and the initial velocity needed to reach the target position given that angle is what is calculated.

The projectile motion formulas only consider the trajectory to have two dimensions, the horizontal and the vertical one. Thus rises the need to rotate the ball to face the target position, changing its referential so that the forward vector (the Z coordinate) points towards the target. This way, the velocity can be obtained in the ball's referential, with the Z coordinate being the horizontal dimension and the Y being the vertical one (fig. 3.9), and then converted to the world coordinates.

Summing up, the ball's initial velocity is obtained through the following steps:

1. Rotation of the ball's referential so that it faces the target point.

2. Calculation of the initial velocity using the formula:

   $$V_i = \sqrt{\frac{dist \times g}{\sin 2\theta}}$$

   In which *dist* is the distance between the ball's initial position and the target's, g is the gravity, and θ is the launch angle.

3. Decomposition of the initial velocity in its horizontal (V_z) and vertical (V_y) components, in order to obtain the velocity vector.

4. Conversion of the velocity vector to world coordinates.

If at some point the ball is on the ground and is not considered to be in any of the character's possession, the NPC will go into the "Fetch Ball" state. In this state, he will go after the ball and try to grab it. The NPC will push his wheelchair, moving towards the ball and then turn to face it. When he is within a certain distance from the ball, it is considered to be in range to be grabbed and, therefore, the character will stop and bend down towards the ball. This bending motion is made through an animation which the animator activates when its *GrabTheBall* parameter becomes true (fig. 3.7). If while the NPC is reaching for the ball it rolls out of grasping distance, then the *OtherPlayerController.cs* script puts the parameter's value back to false, activating in the animator the transition to an animation in which the character gets back up.

[4]Based in the code available at: https://vilbeyli.github.io/Simple-Trajectory-Motion-Example-Unity3D/ (last access: 27 Nov. 2018)

Figure 3.9: Ball's trajectory when thrown by the NPC

While the NPC is bent down, reaching for the ball, unity's built-in IK is used to make both hands move towards the ball. Once one of the hands touches it, the ball is caught the same way it is in the "Defend" state. However, instead of going immediately into the "Pass Ball' state, the animation that makes the character rise back up is activated and, afterwards, the NPC goes into the "Move To Pass Ball" state.

In the "Move To Pass Ball", the NPC tries to place himself so that he can successfully pass the ball to the user, thus avoiding passes, for example, from behind the user. For this, two parameters are had in consideration: the distance between the user's player and the NPC, and the angle between the player's orientation and the one he would need to have to be facing the NPC. The NPC is considered as being in a good position when he is at a distance higher than a certain value, and the angle is no higher than 40 degrees. The NPC will move towards the position $player.\vec{position} + player.\vec{forward} \times 10$, in which $player.\vec{forward}$ is a vector with the direction the user is facing, until he reaches a proper position. Once there, he will stop, face the user, and finally go into the "Pass Ball" state. If the NPC is not capable of reaching a proper position because, for example, the user is close to a wall and facing it, once he is no longer capable of getting closer to his goal he will also turn to face the user and go into the "Pass Ball" state.

3.2.5 360 Video Integration

As previously mentioned, in this thesis 360 video was used together with the computer-generated elements in an attempt to create a more realistic environment that combines the virtual with the reality. This brought the challenge of how to integrate the video and virtual elements together as seamlessly as possible. Three different techniques were applied to achieve this goal: fade from video to virtual; video as textures; and integration of 3D objects onto tracked 360 video.

3.2.5.1 Fade from Video to Virtual

The first technique used to try to increase the realism of the virtual environment was to make an association between the reality and the environment, starting by showing a 360 video of the real environment and then fading to the virtual one with which the user can interact, and which is made to look as similar to the real one as possible.

In the system, this technique is present when the user chooses to play in the full practice. When choosing to do so, the user is first presented with a scene in which he/she sees a 360 video that shows the inside of a court that is empty (fig. 3.10a), recorded from the middle of that court. After a few seconds a new scene, the one in which the user can actually interact with the environment and play the game, is loaded. This new scene, the full practice scene, has a virtual court made to replicate the one in the footage (fig. 3.10b), and starts with the user sitting in the middle of that court, in the same position the video was recorded from. From one scene to the other, while the interactable one is being loaded, there is a fade to and from black.

As already mentioned in 3.2.3, the loading of a new scene is made by the *LoadLevel.cs* script. This script was based in the *SteamVR_LoadLevel.cs* script of SteamVR, and was created in an attempt to make the loading occur faster and allow the transition between the scenes to be a fade to and from black. The created script loads the scene asynchronously. It fades to the Compositor and, after finishing loading the scene, fades back from it to that scene. The script is responsible for making the Compositor be fully black, with no grids, and gives the possibility of displaying a loading image in it.

Before the *LoadLevel.cs* script as created, some other attempts at making the fade between scenes were made. These did not work since while loading a new scene the user would always be transported to the Compositor due to the drop in FPS. Therefore, it became clear that for the fade to work what was needed was for the Compositor to be edited.

The court present in the recording, and which the interactable scene attempts to replicate, is from *Centro de Alto Rendimento* in *Vila Nova de Gaia* where, as already mentioned in 3.2.2, the 360 video recordings were made.

Since the video of the empty court used shows the tripod used while recording, there was the need to remove it. For that Photoshop was used to create an image to cover the tripod in the videos, that is similar to what the real floor under the tripod would look like.

3.2.5.2 Video as Textures

In order to help increase the realism of the virtual environment, real pictures and videos of the court were used as textures that are applied over the floor, walls, and ceiling of the virtual court. These textures are present in all the practice scenes, including the full practice. Therefore, when there is the fade from the video to the full practice scene, this textures further help in making the environments seem like the same.

(a) 360 video of empty court (before fade)

(b) Virtual court (after fade)

Figure 3.10: Transition from the video scene to the computer-generated one.

To create the floor a picture of a piece of the real floor was used as a tile. Over the tiling was placed an image of basketball court white lines created with Paint.

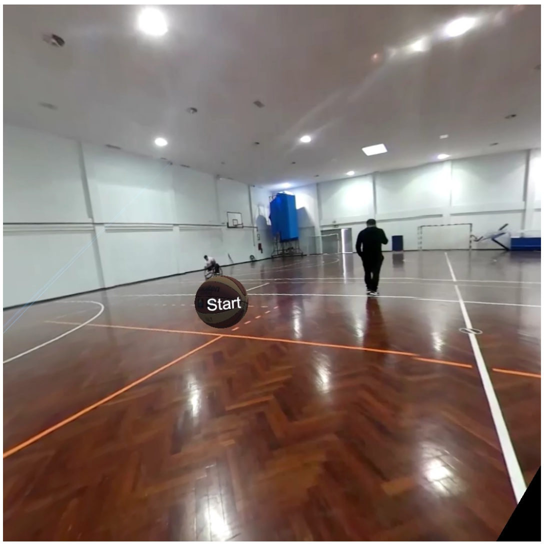
For the ceiling, a piece of the previously mentioned video of the empty court was used. The video was manipulated, using Adobe After Effects, to get a rectilinear view of just a particular part of the court, in this case the ceiling, from the equirectangular 360 video. To do this the video was converted from equirectangular to a cube-map and had its camera view orientation changed, and then the desired piece of the court was cropped. This was used to obtain some short clips of the ceiling of the court, and also of the two walls that do not have backboards. Each video was then used in the video player placed in the object of the scene corresponding to the part of the court represented in it. An image of a frame of the video was used as the texture of that same object, so that it will already look like that part of the court before the video manages to start playing and replace the texture with said video.

Regarding the two remaining walls, the ones with the backboards, an edited picture of one of them, without the backboards visible, was used. This allowed to place the computer-generated backboards over the walls, leading to a proper integration of the virtual elements with the video.

3.2.5.3 Integration of 3D Objects onto Tracked 360 Video

The third technique applied has the purpose of allowing the user to interact with the video by having interactable computer-generated elements overlaying the video. These elements move with the video, appearing to be part of it.

The most noticeable scene in which the technique was applied is the ”Classification Video” one, but it is also present in the ”Moving Video Tracking” scene, the initial scene that appears before the main menu one. In the "Moving Video Tracking" scene (fig. 3.11a), a computer-generated basketball ball, which the user must click in order to go to the main menu, appears as if the ball was within the court shown in the video. In the "Classification Video" scene (fig. 3.11b), a computer-generated element is placed above the head of each of the two players in the video whose classification is being explained. These classifications are based on their limitations in functional skills. When a user points at the element above one of the player's head, a canvas with the explanation about his class appears by the player's side. When the user stops pointing, that canvas disappears. If the user clicks in the element, the corresponding canvas becomes fixed and does not disappear when he/she stops pointing at the element.



(a) "Moving Video Tracking" scene

(b) "Classification Video" scene

Figure 3.11: Scenes with the integration of 3D objects onto tracked 360 video present

Interacting directly with a video would be difficult since when playing the video what happens is that the frames of the video are continuously inserted as the texture of the surface the video is displayed in. Therefore, it would be extremely difficult to know anything about the content of the video and thus hard to, for example, know what a user is pointing towards in the video when he/she is trying to interact with it. Due to this, it was decided to place computer-generated elements, with which is easy to interact with, overlaying the video. To be able to place these elements in the correct position, and with the right scale, it was necessary to process the video.

In the video processing there is the need of tracking, in the video, the position the computer-generated element is supposed to be placed at. To make this simpler it was decided that the videos should be edited so as to insert easy to track markers in them which mark the place to overlay. These markers are balls, each of an easily identifiable color.

In order to insert the markers, the videos are pre-processed. The Adobe After Effects software was used for this, performing motion tracking on the video to track, for example, a player's head. After tracking the motion, a marker that follows that motion is inserted.

The marked videos are played in Unity. However, for Unity to be able to place the elements over the marks, it needs to track those marks. For this, C++ code created outside of Unity was included in the form of a plugin. This allowed to use the OpenCV library to process the video, tracking the markers, and giving the resulting information to Unity.

The C++ code developed is inserted in the Unity project as a *.dll* library file. The file's functions are exported from the library and can then be called from a C# script.

In order to track a mark, it is necessary to detect it throughout the frames. Thus, within the C++ code there is the *TrackInFrame* function, which is responsible for detecting a circle of a specific color in a frame. The detection algorithm implemented (fig. 3.12) uses color detection[5] to find the pixels with the color of the mark to track. This is done by creating a binary image in which a pixel is assigned to '1', if its color is within a threshold that identifies the color being tracked, and '0' otherwise. To eliminate noise from the resulting binary image a morphological opening (erosion followed by dilation) is applied. This is then followed by a morphological closing (dilation followed by erosion) that eliminates potential small holes from the tracked mark.

Once the color detection is concluded, it becomes necessary to find the coordinates of the circular mark and its radius. This is accomplished by computing the contours of the objects[6] that appear in the binary image that resulted from the color detection. After finding the contours, the largest contour, which is expected to be the one of the circular mark being tracked, is found by comparing the contour areas. The minimum enclosing circle of this larger contour is then computed. This circle is considered to be the one being tracked, and its center's coordinates and radius are the information regarding the mark that is eventually passed to Unity.

The script created in Unity that is responsible for using the developed plugin and placing the computer-generated interactable elements in the right positions, overlaying the tracking marks, is the *VideoTrackingOpenCV.cs* script.

In order to detect a mark in a frame, there is the need to obtain that frame so that it can be passed to the *TrackInFrame* function. The algorithm initially created to accomplish this, does it by obtaining each frame as it is rendered in Unity and calling the *Track* function from the plugin created, sending it that frame to process. Every time a new frame is ready, the *OnNewFrame* function is called. In this function, the frame is obtained as a texture, converted to an array of colors (representation of RGBA colors in 32 bit format), and sent to the plugin's *Track* function, in the form of a pointer to the beginning of the array, to process. In the *Track* function, the data in the colors' array is used to create a *Mat* object that contains the frame. The frame is then converted from RGBA to BGR before being passed to the *TrackInFrame* function, since that function is expecting that format, which

[5] Based in the code available at: *https://www.opencv-srf.com/2010/09/object-detection-using-color-seperation.html* (last access: 3 Oct. 2018)

[6] Based in the code available at: *https://www.pyimagesearch.com/2015/09/14/ball-tracking-with-opencv/* (last access: 3 Oct. 2018)

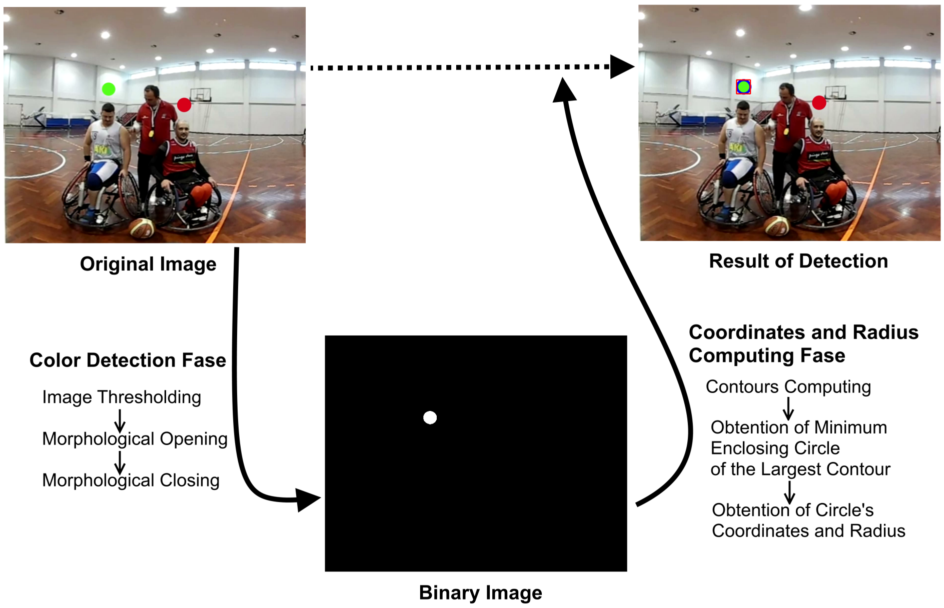


Figure 3.12: Scheme of the detection of a mark in a frame algorithm

it then converts to HSV. The HSV color space is used so that the image thresholding done for the color detection can be performed using an interval of hue, saturation and value that identifies a color.

It was observed that this strategy of obtaining each frame, as it is about to be displayed as a texture, and processing it leads the system into skipping frames. This happens because it takes too long to obtain and process each frame, so in order to catch up with the current play time of the video, some frames are skipped. The reason it takes so long is in part due to the time taken to process the frame, but mostly due to the use of the *ReadPixels* function to obtain the frame. The use of this function is needed to be able to get the frame in Unity and, hence, to be able to process it. The issue is that this function passes the frame image from the GPU to the CPU and is very performance consuming.

It became necessary to find a way to reduce the skipped frames. One strategy applied to this effect was that the information obtained from processing the frames started being saved so that, when a video is played in a loop, a frame that was already previously processed does not need to be processed again and, therefore, does not need to be obtained to pass to the processing function, which increases performance. This way, each time the video is played, more of the frames that are not skipped, and thus will get displayed, will already have been processed in previous times, since they already got displayed before. Therefore, the skipped frame rate will be reduced each time the video is played (fig. 3.13).

However, this improvement is not enough since the first few times the video is played it will still be too slow and lead to too much frames being skipped, and not all videos are displayed in a loop. Hence came the need to reduce the time it takes to know the

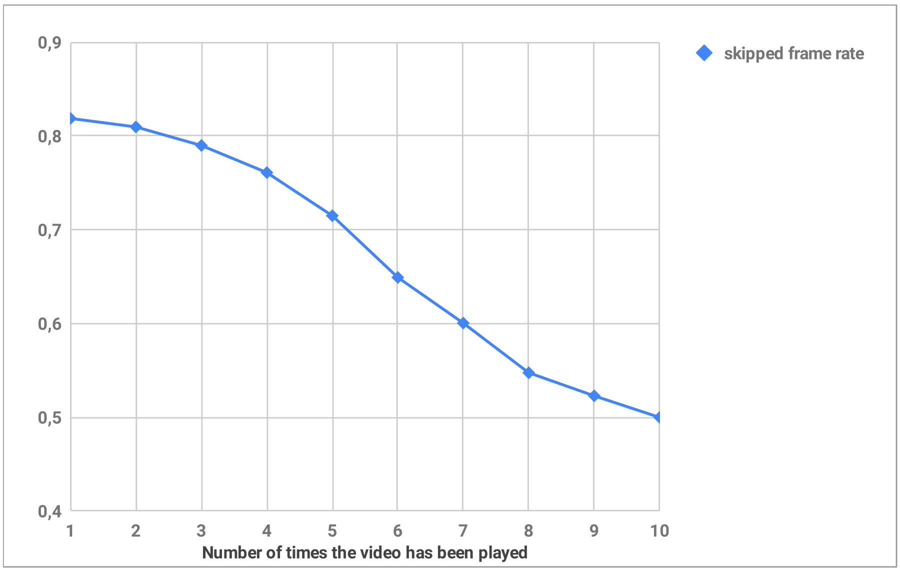


Figure 3.13: Skipped frame rate evolution as the video is played in loop (Data obtained using the video displayed in the "Classification Video" scene)

position and radius of the mark in the frame that is going to be displayed. Having this in consideration, the second algorithm to be implemented, instead of obtaining and processing each frame just before displaying it, processes all frames before the video starts to be played and saves that information. This way, whenever a frame is to be displayed, it is only necessary to consult the saved information relative to that frame.

In this strategy of processing the entire video before it starts to be played, the plugin's *TrackAll* function is called in the beginning and it uses OpenCV's *VideoCapture* to open the video and read each frame, processing them as they are read. This has both the advantage of avoiding the constant need of coping from GPU to CPU, and of not having to spend time processing a frame when it is about to be displayed. By using this strategy, the number of skipped frames while playing the video is reduced to zero.

However, and even though that this implementation was made with relatively small videos in mind, the processing of all the frames of the entire video in the beginning takes a considerable amount of time. Due to this, when loading a scene such as the "Classification Video" one, the scene will take a long time to load because it will have to wait for the processing of the video to finish, which is a problem. Therefore, it was proven necessary to find a way to make the initial processing faster. This lead to the third algorithm implemented which, like the previously explained, processes the video before it begins playing, but has the improvement that instead of processing every single frame in the video, it only processes a frame in every X frames (fig. 3.15).

The *read* function of OpenCV's *VideoCapture* class grabs, decodes and returns the next frame of the video. Whereas when all frames are to be processed there is the need to obtain every frame, when only one in every X frames is processed only the ones to process need to be obtained. Thus, the rest of the frames do not need to be decoded and returned. Therefore, the *TrackEveryXFrames* function of the plugin, created to obtain the data relative to the video in the beginning but only processing every X frames, uses *VideoCapture*'s *grab* function for the frames that do not need to be processed, which only grabs the frame. For the frames that need to be processed, the *read* function is used so that the frame is not only grabbed but also decoded and returned. This way it becomes more efficient, since it will not waste time decoding and returning unnecessary frames.

Due to not all frames being processed, the information for the rest of them has to be obtained by interpolation. The values of the processed frames between which a frame, f, is situated are used to generate its value through a weighted average. The weight depends on the processed frame's proximity to the f frame, the closer the frame is, the higher a weight it has.

It was observed that the best method to process the videos present in the created system is by processing one in every 6 frames, since this is the highest number of frames that can be skipped without any mistake in the overlay being potentially noticed by the user. When the number of skipped frames becomes higher, the deviation between the values estimated by the interpolation and the actual coordinates of a mark become big enough for errors to rise.

In order to further help reduce the time spent in the initial processing of the video, the possibility to define the moment of the video (in seconds) until which it needs to be processed was added. The reason for this addition is that some videos do not need to be processed until the end, because the marks to be overlaid are not present to the end of said video, so it is unnecessary to process the frames that do not have marks in them. For example, the video displayed in the "Classification Video" scene has 80 seconds in total, but the marks are only present until second 78 (fig. 3.14). In this particular case, only 2 seconds of video would be processed unnecessarily, but in others it could be more.

In the "Classification Video" scene's video, there are two marks to track, a green and a red one. Since there is the need to detect two different colors in each frame, the initial processing takes even more time then when there is only one color mark to track. Hence, functions equivalent to the ones previously mentioned (*TrackAll* and *TrackEveryXFrames*) that do the initial processing of a video, but which are prepared to do so for two different color marks, were created in the plugin. These functions have the purpose of attempting to reduce the initial processing of the video time (fig. 3.15). Instead of calling, for example, the *TrackEveryXFrames* function twice, once for each mark, and having to obtain the frames both times in order to process them to detect each color, the equivalent function that processes both color marks can be used. With that function, the frames are only obtained once and are then used to detect each of the colors. This leads to saving time

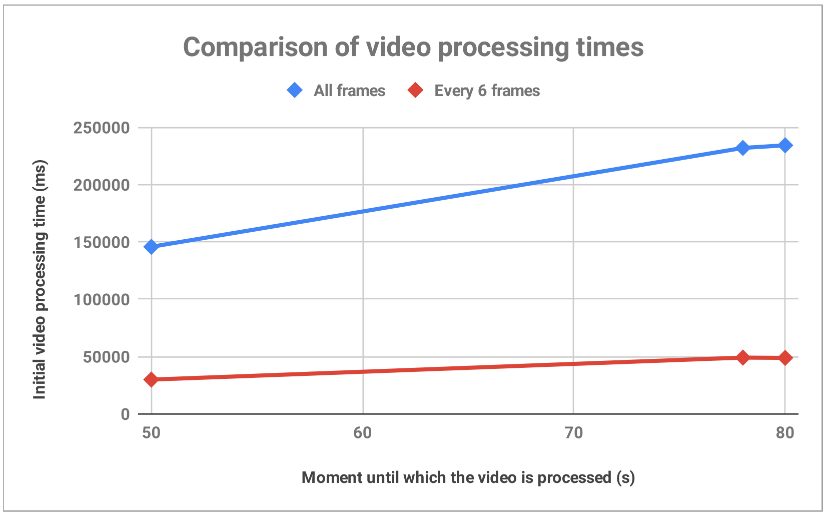


Figure 3.14: Comparison of the video processing time, as the moment in seconds until which the video is to be processed varies. (Data obtained using the video displayed in the "Classification Video" scene. Values from when all frames are initially processed and when one in every 6 frames is processed)

due to not having to obtain the same frames twice.

All 360 videos presented to the user in the developed system, are displayed by projecting the video inside a sphere created to act as a 360-degree video player. The camera through which the user views the environment is placed in the middle of that sphere. This way, as the user looks around himself/herself he/she will be able to watch the video in any direction.

After the information resulting from the processing of a frame (coordinates and radius of the mark) has been obtained, and that frame is about to be displayed, there is the need to know where to place the interactable computer-generated element so that it will appear to the user as overlaying the mark in the video. Therefore, the (X, Y) coordinates of the mark in the video need to be converted to the (X, Y, Z) coordinates of where that part of the video is being displayed, when projected inside the sphere, in the world coordinate space (fig. 3.16).

This conversion is accomplished by starting by converting the (X, Y) cartesian coordinates of the video to the longitude and latitude of the position of those coordinates in the sphere the video will be projected inside of (3.1). This conversion is made having in consideration the width and height of the video. Then, that longitude and latitude together with the radius of the sphere are used as the polar coordinates of the position in the world the mark of the video is being displayed at. Finally, the polar coordinates are

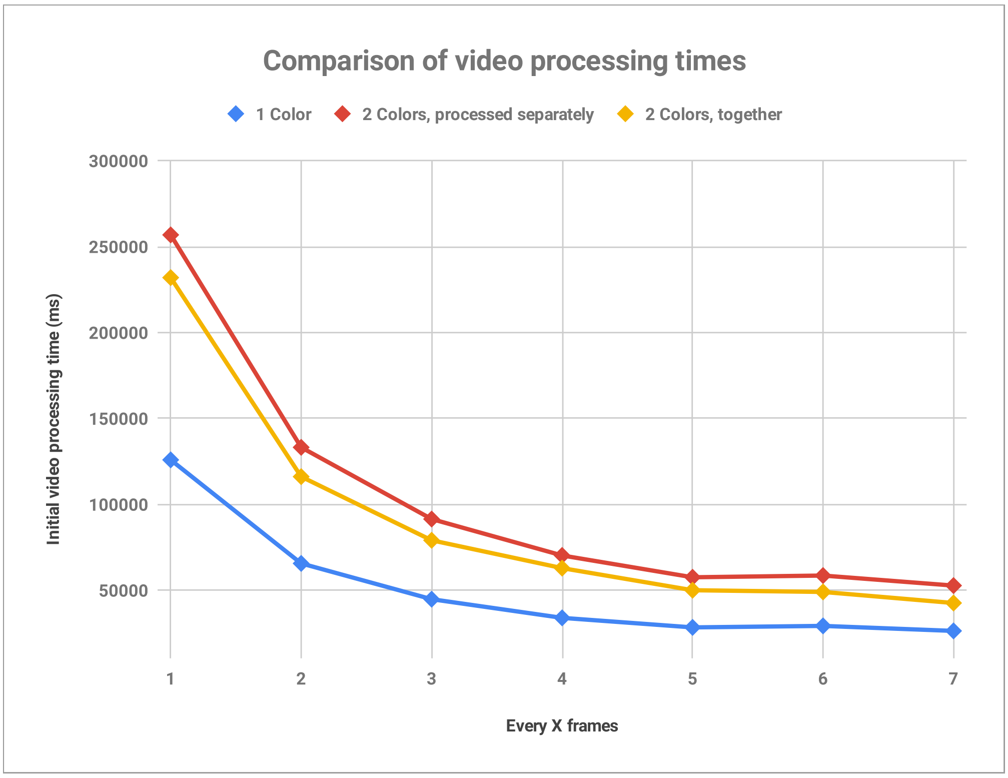


Figure 3.15: Comparison of the video processing time when only 1 color is being tracked, 2 colors are being tracked and the video is processed twice, once for each color, and 2 colors are being tracked and the video is only processed once. (Data obtained using the video displayed in the "Classification Video" scene, processed until second 78)

converted to the (X, Y, Z) cartesian world coordinates (3.2).

$$\phi = longitude = \frac{x \times 2\pi}{width}$$
$$\theta = latitude = -\frac{y \times \pi}{height} \tag{3.1}$$

$$X = radius \times \sin\theta \times \cos\phi$$
$$Y = radius \times \cos\theta \tag{3.2}$$
$$Z = radius \times \sin\theta \times \sin\phi$$

The radius of the mark, obtained as result of the frame processing, is used to compute the scale of the computer-generated element so that the element's radius will vary with the mark's.

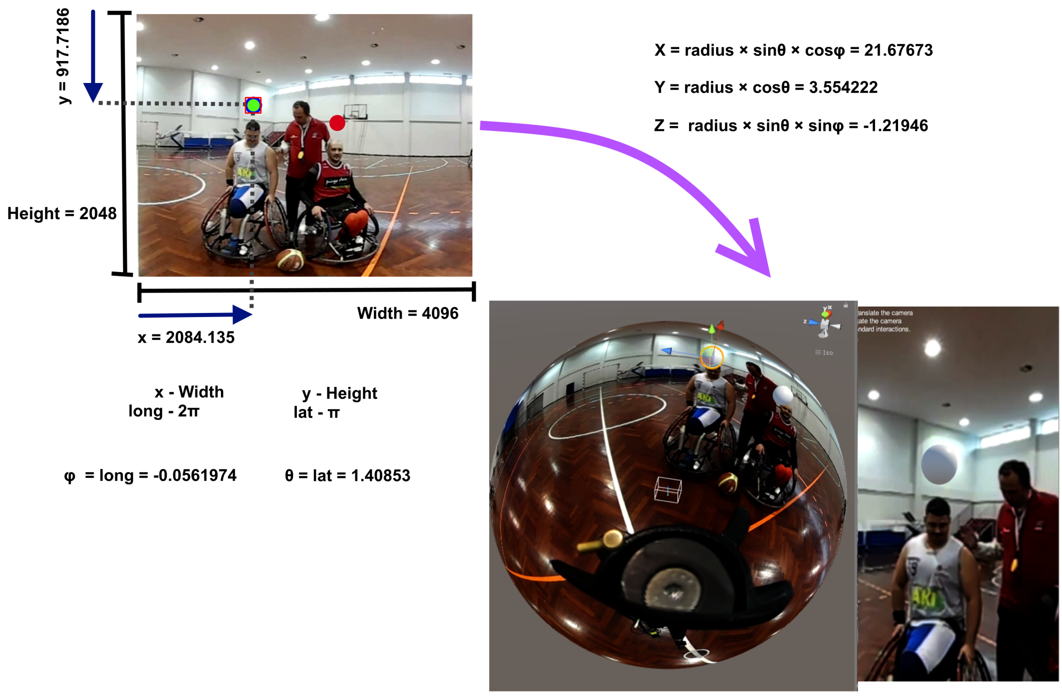


Figure 3.16: Scheme of the coordinates' conversion

Any GameObject can be used as the element placed onto the video. For that GameObject to be interactable, the *VideoInteratabe.cs* script was developed. This script makes it possible for the user to interact with the element by pointing the controller at it. As with the elements in the main menu, previously mentioned in 3.2.3, and which also use this script, raycasting is used to create the interaction with the elements. The *RaycastInput.cs* script, placed in the controller capable of interacting with the elements, was developed to send a ray from said controller forward and detect what element the controller is pointing at. If the element has a *VideoInteractable.cs* script, its funtions will be called when a ray starts, stops or was already and is still pointing at it, and when the controller trigger is pressed while pointing at it.

The *VideoInteractable.cs* script makes it easy to add behaviour as a response to the user's interaction with the element, since it uses *events* to broadcast to any class interested in the event that it has occurred. Different scripts can be easily created to implement the behaviour wished to be activated in response to the user's actions. Those scripts simply have to subscribe to the *events* of the *VideoInteractable.cs* and implement what will happen when those specific actions occur. For example, the *InteractShowCanvas.cs* script, used in the "Classification Video" scene to control whether the information canvas are or not visible, subscribes to the events of one of the interactable elements and, when one of those events is invoked, it changes the state of the canvas it is responsible for. This way, when the user points to the interactable element placed on top of a player's head, this scripts ensures that the canvas with the information regarding the class of players to which that player belongs, becomes visible for the user to see.

The *CanvasPositioning.cs* script was created to keep the canvas properly positioned according to the position of the element it is associated with, and keep it turned towards the user so he/she can read the information written in it easily. This script converts the cartesian (X, Y, Z) coordinates of the interactable element to polar coordinates, adapts those coordinates according to the position the canvas is supposed to be in relation to the element, changing, for example, the longitude. After having the new polar coordinates, they are converted once again to cartesian (X, Y, Z) coordinates, which the canvas is then placed at.

The *InteractLookChange.cs* is another example of a script that implements the behaviour that occurs as response to the *events*. It changes the color of the interactable element depending on whether the element is being pointed at, clicked or is not being interacted with.

Chapter 4

Evaluation and Results

This chapter presents the tests used for evaluating the system created and the obtained results.

An iterative approach was adopted in the design and development of the system. There were two main iterations: a first one in which most functionalities were implemented, and a second one in which the errors detected in the first were corrected and some more functionality added. After each iteration, user tests were performed to evaluate the system and detect aspects to improve.

4.1 Preliminary User Tests

In this first user study, the user was presented with the full practice scene, passing first through the video scene of the empty court that precedes it. Once in the full practice, the user was expected to play the game, interacting with the ball and the NPC, and attempting to score. After a bit of playing in the full practice scene, the user was meant to use the menu to go to the paralympic VR video scene. The rest of the scenes were not tested in this preliminary user tests.

The user was expected to use the system for around 15 to 20 minutes.

4.1.1 Participants and Evaluation Method

For this study, there were two types of participants. Of the total of 10 subjects, 6 were students in *Faculdade de Ciências e Tecnologia*, NOVA University of Lisbon, and 4 were wheelchair basketball athletes of *Associação Portuguesa de Deficientes de Lisboa*.

The students, aged from 22 to 30 with an average age of 23,833 and standard deviation of 3,271, are 4 males and 2 females, and 2 of them had experience with VR, 2 had tried it once before, and 2 had never tried it previously.

The athletes, aged from 17 to 49 with an average age of 29,5 and standard deviation of 15,524, are all male and half of them had never tried VR previously. The other half had tried it once before.

Figure 4.1: One of the athletes playing the game during the user tests

The evaluation of the system was accomplished through passive observation of the user playing the game (fig. 4.1), and a posterior questionnaire and interview. The questionnaire evaluates the sense of presence and the feeling of motion sickness experienced by the user. It is composed of a combination of 22 questions from Witmer and Singer's presence questionnaire[58], and 6 questions regarding motion sickness symptoms. Of the 24 questions present in the Witmer and Singer's presence questionnaire revised by the UQO Cyberpsychology Lab[1], only the questions regarding haptic feedback were not included in the user tests questionnaire. The 6 motion sickness symptoms evaluated in the questionnaire are: general discomfort, stomach awareness, headache, eye strain, nausea, and dizziness. These are symptoms from the simulator sickness questionnaire created by Robert S. Kennedy et al. [22] that were considered to be more relevant to test in this study.

The presence questions have a seven-point scale format and the simulator sickness ones have a four-point scale format that represents how much the symptom is affecting the user (*1-None to 4-Severe*).

Unlike the student participants, the athletes answered a portuguese version of the questionnaire. For the presence questions, the adaptation to brazilian portuguese by Silva, G.R. et al. [37] was used, and for the simulator sickness symptoms M. R. de Carvalho et

[1] Available at: http://w3.uqo.ca/cyberpsy/docs/qaires/pres/PQ_va.pdf (last access: 18 Sep. 2018)

al.'s translation and cross-cultural adaptation[6] of the questionnaire to portuguese was used.

After the user finished filling the questionnaire, he/she answered to a small semi-structured interview. In it, he/she was asked about the background sound, how easy it was to find and interact with the menu, which locomotion method he/she had preferred and why, and what aspects of the game he/she felt needed improving. The athletes were also asked about how realist the environment and interactions felt.

4.1.2 Results

The results of the questionnaires are presented in appendix A. The results of the presence questionnaires are summarized in figures 4.2 and 4.3.

4.1.2.1 Presence

As already mentioned, the presence questionnaires' questions have a seven-point scale format. For most questions '7' is the best result, but in a few of them, such as "How much delay did you experience between your actions and expected outcomes?", due to the way the question is structured, the best result that can be obtained is '1'. Of all the presence questionnaire's questions, the ones in which '1' is the best possible result are numbers 14, 17, and 18.

Globally, the students' results were good. Among the questions for which a higher value means a better result, only the 5th question ("How natural was the mechanism which controlled movement through the environment?") had an average under the value of '4'. Among the questions for which a lower value means a better result, all had an average under the value of '4'. Thus, the presence results were positive although the locomotion methods could be improved.

The athletes' results were worst than the students' ones. This might be due to the fact that since the athletes know how the reality is, they have higher expectations. Therefore, they are harder to please since the way to control things in the virtual world and their behaviour is different than what they are expecting. Among the questions for which a lower value means a better result, all had an average under the value of '4'. Among the questions for which a higher value means a better result, some questions had an average under the value of '4', out of which the worst result was in the 5th question ("How natural was the mechanism which controlled movement through the environment?"). As can be observed in the graph from figure 4.3, the other questions in which a high value was desired that had an average under the value of '4' were questions 1, 2, 3, 6, 8, and 19. Out of these, the questions whose median value result was also under '4' were questions 1, 2, and 8. These questions relate to the ability to control events, the responsiveness of the environment to the actions performed, and the ability to anticipate what would happen in response to those actions. As mentioned, the bad results in these questions is most likely due to the athletes expecting the control and behaviour of the wheelchair

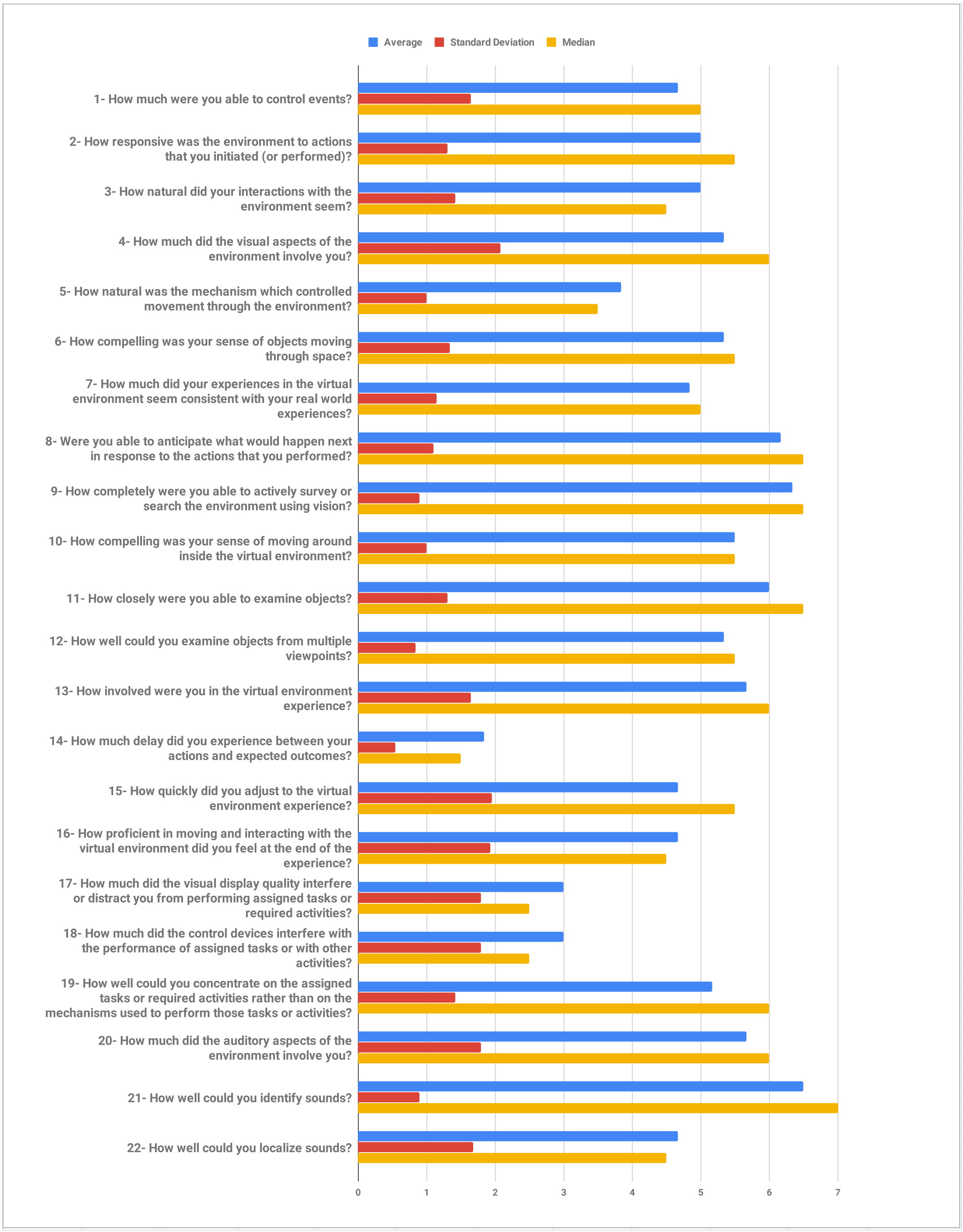


Figure 4.2: Presence questionnaire's students' results.

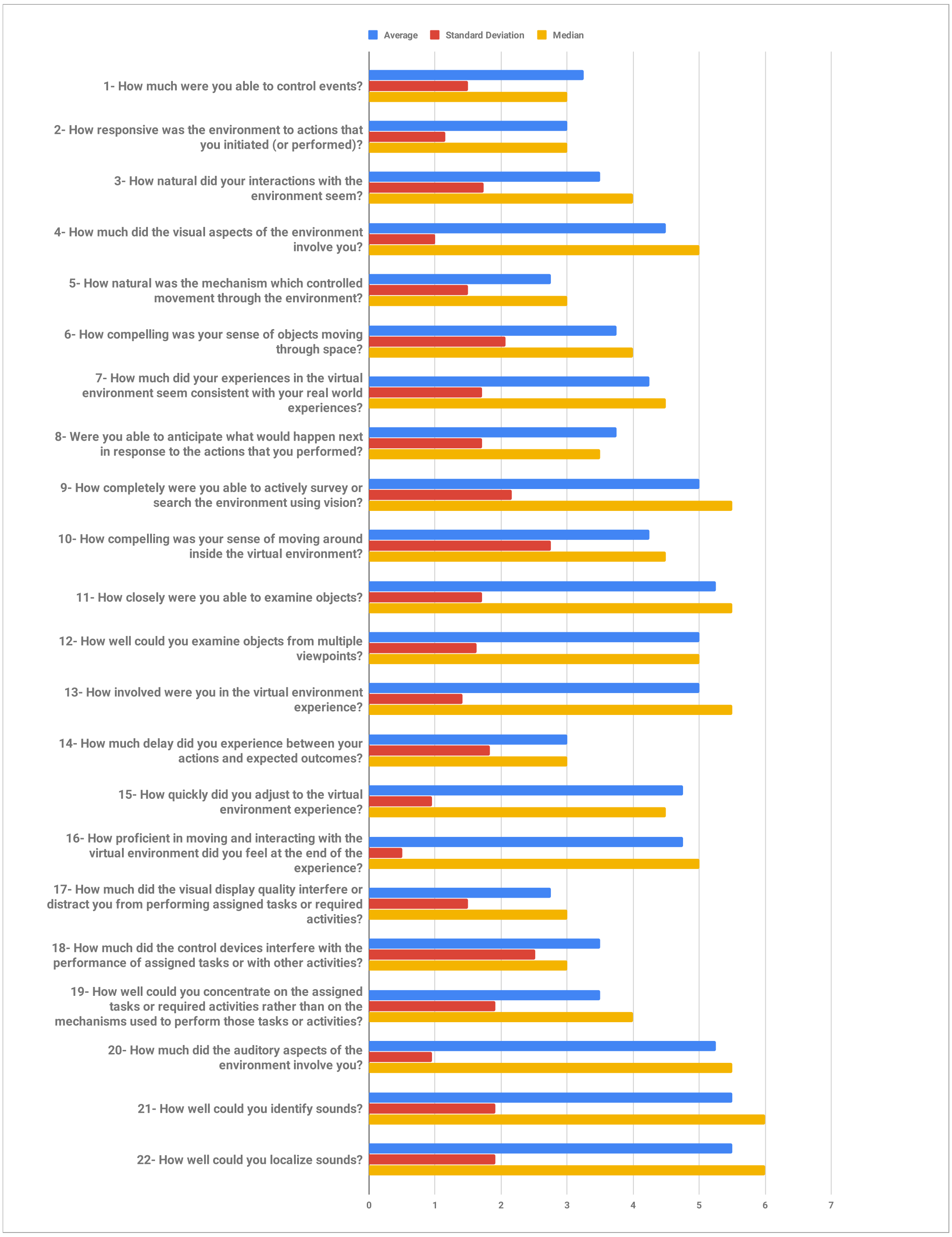


Figure 4.3: Presence questionnaire's athletes' results.

and other virtual elements to be more similar to the natural ones which they are familiar with. However, since the only input the system has relative to the user's body movement is the one provided by the headset and controllers, the rest of the body motions can not be obtained, therefore, for example, the virtual wheelchair can not be controlled by an athlete the same way it would in reality, where an athlete uses the entire body to give impulse to the wheelchair.

The questions regarding sound all got good results from both the students and the athletes. Thus, it can be concluded that the auditory aspects were successfully achieved. Besides the sounds, some other aspects that had good results in both the students' and the athletes' tests were regarding the ability to survey or search the environment using vision, and the ability to closely examine objects.

4.1.2.2 Motion Sickness

The results of the motion sickness questionnaires are summarized in figures 4.4 and 4.5. As already mentioned, these questionnaires have a four-point scale format, in which the higher the value attributed to a symptom, the more severe is its effect on the user.

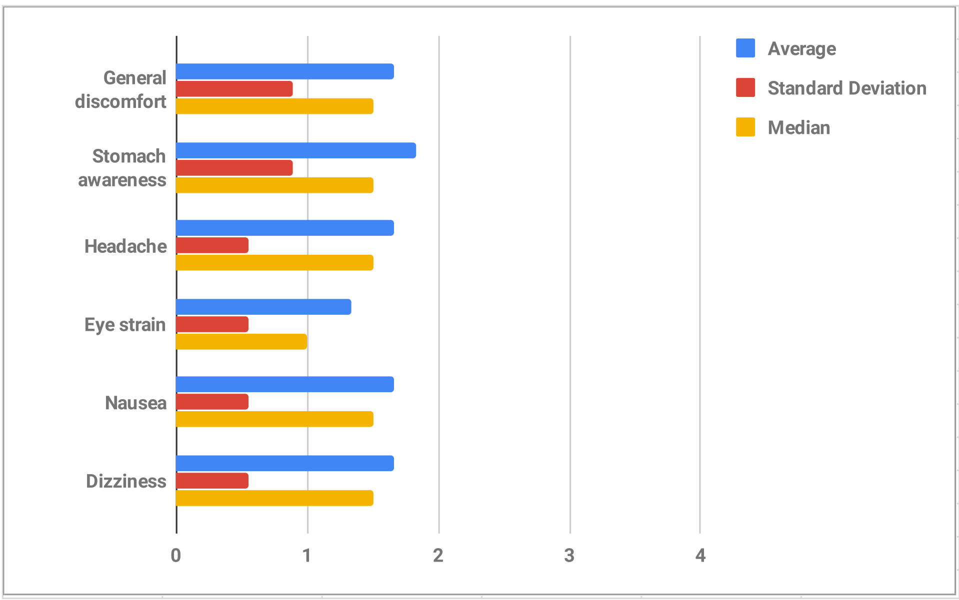


Figure 4.4: Motion sickness questionnaire's students' results.

Globally, the motion sickness questionnaire' results were good. All symptoms were evaluated with values under '2.5' by both students and athletes. In the students' results, the average value for each symptom is between '1' and '2'. However, in the athletes'

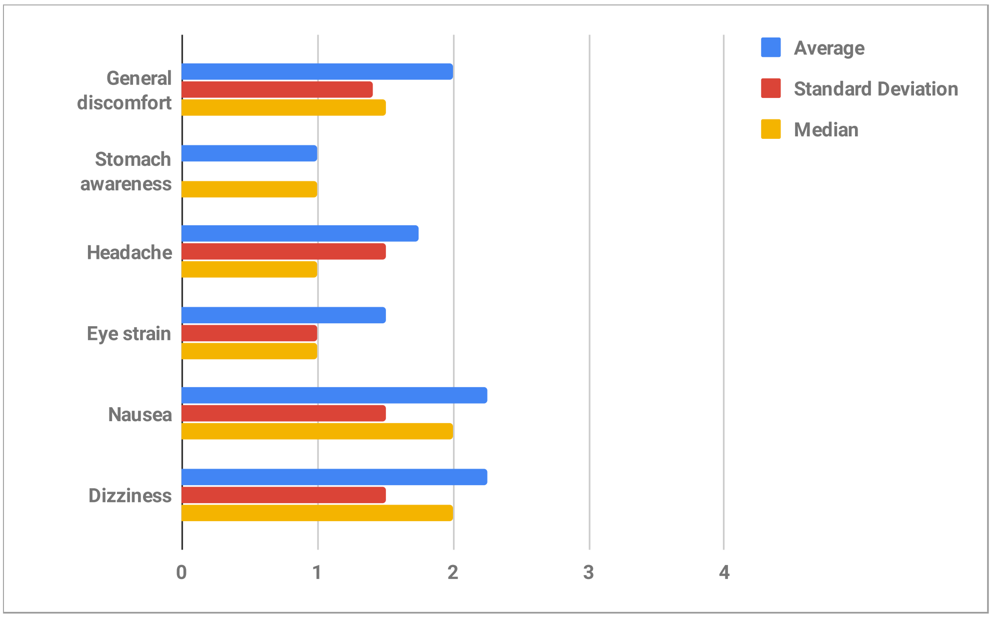


Figure 4.5: Motion sickness questionnaire's athletes' results.

results the average value for both *Nausea* and *Dizziness* surpasses the value of '2'. Hence, of the three symptom clusters considered in Robert S. Kennedy et al.'s simulator sickness questionnaire[22], symptoms of both the Nausea and the Disorientation (includes the *Dizziness* symptom) clusters have a value above '2'. Therefore, it was decided that, in the final user tests, the entirety of the simulator sickness questionnaire would be used, with all its symptoms. This was decided in order to better comprehend what symptoms result from the developed system and the causes of the motion sickness, as well as to be able to calculate a total severity index to access the overall extent of symptom severity.

4.1.2.3 Observation and Interview Results

As a result of these user tests, it was concluded that some corrections had to be made and some functionality needed to be added. Most of these conclusions came from the observation of the users' actions while testing the system and from the information they provided in the interviews. The athletes were the ones who raised the attention to most issues.

The athletes who tried using the virtual wheelchair as the way of locomotion mentioned that its motion was not realistic. One of those athletes said that when given an impulse the wheelchair's speed increased too quickly, and then when it was moving and no more impulse was given, the speed also decreased to quickly and the wheelchair came to a stop to soon. The athletes that tried maneuvering the wheelchair the most, as a result, ended up feeling motion sickness. Most of the users that tried using the virtual wheelchair for some time felt some degree of motion sickness and some of them even chose to finish the test earlier due to it. Therefore, after the tests, in an attempt to improve the wheelchair's movement, changes were made in order to decrease the velocity applied to the wheelchair as a result of pushing the wheels, thus leading to its speed not increasing so quickly. The friction of the wheels' material was also decreased so that the wheelchair will not stop so quickly and the impulse force needed to make it start moving is also reduced.

Still regarding the interaction with the virtual wheelchair, some users also pointed out that it is difficult to push the wheels while looking at the environment around them since they could not feel the wheels and, therefore, did not know whether they were holding the virtual wheel or not. To solve this issue some haptic feedback was later added. Currently while a controller is over one of the wheels and, therefore, the user can interact with it, that controller vibrates.

Most users commented that moving through the court by pushing the wheels felt more similar to what would happen in reality than using the teleport. However, many preferred using the teleport to avoid motion sickness and move faster to where they wished to go to.

Some of the athletes commented that the virtual wheelchair was similar to real wheelchairs they had seen before. Most also felt that the virtual court and the elements present in it, such as the ball and backboard, appeared to have the right dimensions.

Some issues were also raised regarding the ball interaction, as already mentioned in 3.2.4.3. Most people will instinctively try to catch a ball and throw it using both hands. However, in the system the user can only hold the ball with one hand at a time. Also, since at the time these tests occurred the user could only interact with the ball by grabbing it, and not by colliding with it and pushing it around, dribbling the ball was a difficult task to accomplish. Therefore, although the possibility of holding the ball with both hands at the same time was not implemented, the possibility of using the hand that is not grabbing the ball to assist in throwing the ball by colliding with it, and thus giving it support, was. After the hand collision was implemented as a solution to the problems found in the user tests, dribbling the ball, pushing it, and supporting it with the hands became possible.

Very few users managed to score and none of the athletes did it. Some mentioned that it is a bit difficult to know the strength and movement needed to make the ball reach the basket since it is not possible to feel the weight of the ball. Even so, some people said that it probably was just necessary to practice the throws in order to calibrate the strength used, since they slowly got better at it. Either way, to increase the chances of a user scoring, after the user tests the size of the backboards, and consequently the baskets, was slightly increased.

A few other issues were discovered and easily fixed, such as when a user would teleport too close to a wall he/she would go through that wall and fall of the court. This was solved by reducing the area the users can teleport to in a way that guarantees that they will always fully remain inside the court. Another of these issues was that when the NPC had the ball and was moving to a place from which to pass the ball to the user, and the user was turned towards a wall and too close to it, the NPC would not be able to move far enough in front of the user to get to a position considered good to pass the ball. Hence, the NPC would stay close to the wall, not moving and not passing the ball. To solve this, a change was made so that, in this case, the NPC would pass the ball, as explained in 3.2.4.4.

One of the athletes revealed that usually the athletes start by practicing only the wheelchair maneuvering and only the ball throwing alone, and only practice doing everything together later on. Therefore, after these tests, the wheelchair practice and the throws practice scenes were created.

Every user had a positive view on the background sound and the ball audio, finding it helpful in creating the sensation of immersion.

4.2 Final User Tests

In the final user tests, the final state of the system, as described in the implementation section in 3.2, was evaluated. The user is presented with the initial "Moving Video Tracking" scene, followed by the "Main Menu" one. From there, the user is asked to enter a specific practice scene. After playing in that practice scene for around 10 minutes, the user is asked to change to another of the practice scenes, and after that to the last of the three practice scenes. The user plays in each of the these scenes for about 10 minutes. Finally, after playing in all three practices, the user is instructed to go to the "Classification Video" scene, and afterwards, he/she is asked to exit the system.

Since there are three practice scenes, there are six possible ways to order them. Considering that the order by which the users experience the scenes may affect the results, it was decided that an equal number of users would perform the tests by each of the six possible orders.

4.2.1 Participants and Evaluation Method

A total of 18 volunteers, aged from 19 to 26 years old, participated in this study. Their age average is of 21,667 and the standard deviation of 1,815. These participants were students in *Faculdade de Ciências e Tecnologia*, NOVA University of Lisbon. None of the volunteers had participated in the previous study, nor previously tried the developed system. Out of the 18 students, 13 were male and 5 were female, and only 3 of them had experience with VR and 4 had tried it once before. The other 11 had never tried VR previously.

As in the preliminary user tests, the evaluation of the system was made through passive observation of the user playing the game, and the realization of questionnaires and an interview. Each user answers 4 questionnaires, one after each 10 minute practice, that evaluates the feeling of motion sickness experienced by the user as a consequence of that practice, and one in the end, after exiting the game. This final questionnaire evaluates the sense of presence through the game and the video integration of 3D objects onto tracked 360 video.

In the ”Wheelchair Practice” the user was instructed to attempt to maneuver around the cones laid out through the court, in order to force the user to use the wheelchair to move through the court, and in the ”Throws Practice” the user was instructed to attempt to score and dribble the ball.

Attending to the fact that haptic feedback was added since the preliminary tests, the presence questionnaire in these tests included all 24 questions present in the previously mentioned revised presence questionnaire by UQO Cyberpsychology Lab[2].

Regarding the motion sickness questionnaire, as it was already mentioned in the results of the preliminary user tests, the entire Rovert S. Kennedy et al.'s simulator sickness questionnaire [22] was used, with all its 16 symptoms.

After the user finished filling the final questionnaire, he/she answered to a small interview in which he/she was asked about his/her thoughts relatively to the video integration of 3D objects onto the tracked 360 video, the interaction with that video, the movement and control of the virtual wheelchair, the interaction with the virtual ball, and what aspects of the system he/she felt needed improving.

The entire duration of a user test, including the filling of the questionnaires and answering of the interview was about an hour.

4.2.2 Results

The results of the motion sickness questionnaires and of the final questionnaire, that includes the presence questionnaire as well as the questions regarding the video integration, are present in appendix B. The results of the presence questionnaire are summarized in figure 4.6, and the results of the video integration questions are summarized in figure 4.7.

4.2.2.1 Presence

As mentioned in the results of the preliminary user tests, for most questions in the presence questionnaire '7' is the best result, but for a few of the questions (numbers 14, 17, and 18) the best possible result is '1'.

[2]As already mentioned, available at: http://w3.uqo.ca/cyberpsy/docs/qaires/pres/PQ_va.pdf (last access: 18 Sep. 2018)

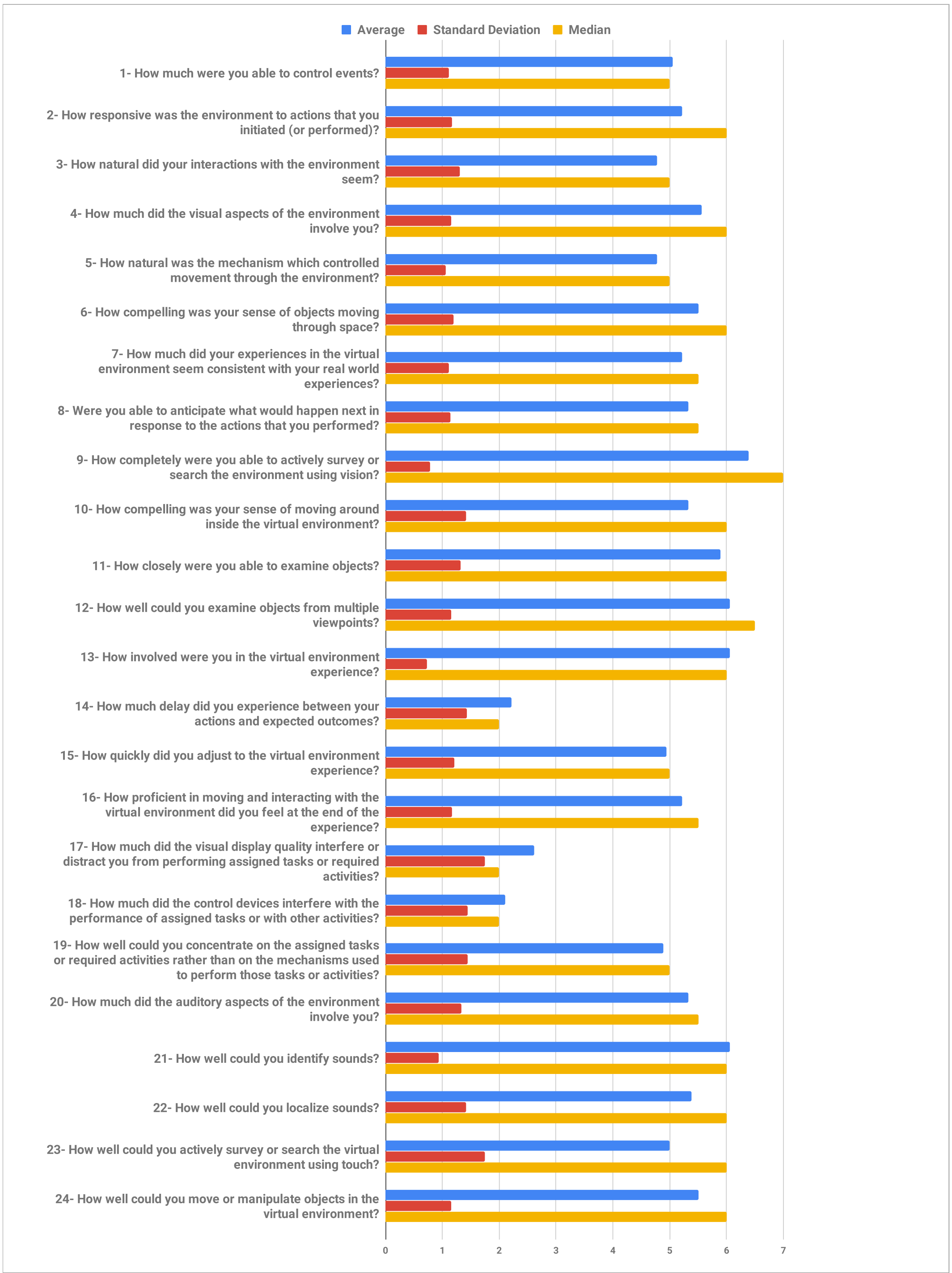


Figure 4.6: Presence questionnaire's results.

The results of the presence questionnaire were good. The questions in which a higher value means a better result all had an average and median higher than '4'. The questions for which a lower value means a better result, all had an average and median under '3'.

When comparing with the preliminary tests' students' results, it was observed that the results are similar in most questions and, in some, they have improved relatively to the previous ones, the most noticeable one being the result to the 5th question ("How natural was the mechanism which controlled movement through the environment?"). While in the preliminary results the average value for this question was of '3,833' and the median of '3,5', being under the medium value of '4', in the results of the final tests it rose to an average of '4,778' and median of '5', which are above the medium value. This improvement indicates that the locomotion methods have improved relatively to the way they were in the preliminary tests, which suggests that the changes in the wheelchair movement were successful.

The results of the haptic feedback questions, which were not present in the preliminary tests' presence questionnaire, were positive. Both questions ("How well could you actively survey or search the virtual environment using touch?" and "How well could you move or manipulate objects in the virtual environment?") had a median result of '6' and an average of at least '5'. This shows that the haptic feedback that was inserted in the system was successful.

4.2.2.2 Integration of 3D Objects onto Tracked 360 Video

The evaluation of the developed technique for the integration of computer-generated elements onto tracked 360 video, as observed in figure 4.7, shows promising results. Additionally, during the interviews, the users proved to be pleased with the integration result. When asked, most users' opinion was that the computer-generated spheres in the "Classification Video" scene were well integrated with the video. They said that they could notice that the spheres were not really from the video, due to the distinct image quality, since the 360 video's resolution was not very good. However, they referred that it was noticeable that the computer-generated elements go along with the video, and that the fact that the elements had a distinct resolution made them more prominent, which is helpful since it indicates that the elements are something that can be interacted with. Nearly every one immediately tried pointing to the spheres when starting in the scene. On the other hand, a lot of people did not try clicking in them, and most of those that tried took some time to realise that clicking in a sphere lead to keeping a canvas active in the scene even after the user stops pointing at the sphere. It was suggested that, besides changing the color of the sphere when it is clicked at, the canvas opacity could also be changed according to whether or not it is fixed in the scene.

Some users attempted to click in the players or the basketball ball in the video to see if it had interaction. They mentioned that they wished there were more actions possible to do in the video.

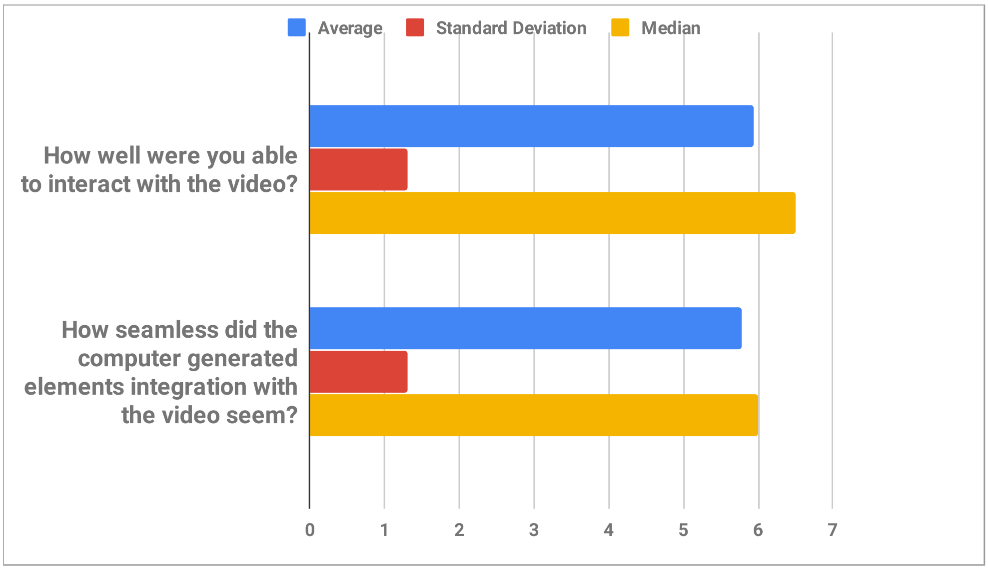


Figure 4.7: Video integration of 3D objects onto tracked 360 video questionnaire's results.

When asked about the "Moving Video Tracking" scene, in which a computer-generated basketball ball is present for interaction, most users said they did not recall what was in the scene. Of those who did, some said they noticed that the basketball was not real due to it appearing to be floating, but that that simply helped to know that it was something to be interacted with.

From both the questionnaire results and the users' feedback in the interview, it can be concluded that the method of integration of 3D computer-generated elements onto 360 video developed seems promising and should be further explored.

During the user tests some users, noticing the long loading time of the "Classification Video" scene, asked if it was normal or if there was some problem that was dealing it to take so much time. Of the other users, when inquired about it, most said they thought the loading took some time, but seemed normal. This shows that, in the future, more effort should be put in attempting to reduce the initial video processing time.

4.2.2.3 Motion Sickness

While the motion sickness questionnaires presented to the users have, like in the preliminary tests, a scale of '1' to '4', those values were adapted to a scale of '0' to '3' for the evaluation of the results. Thus, the graphs shown in 4.8 and 4.9 were made considering this interval. The reason behind this adaptation is that Robert S. Kennedy et al.'s [22] simulator sickness questionnaire and scoring used this scale. Therefore, and since in the evaluation of the motion sickness results of the final user tests it was decided to calculate the total severity index and other scores presented by Robert S. Kennedy et al., it was decided that the values should be adapted to be according to the original simulator sickness questionnaire.

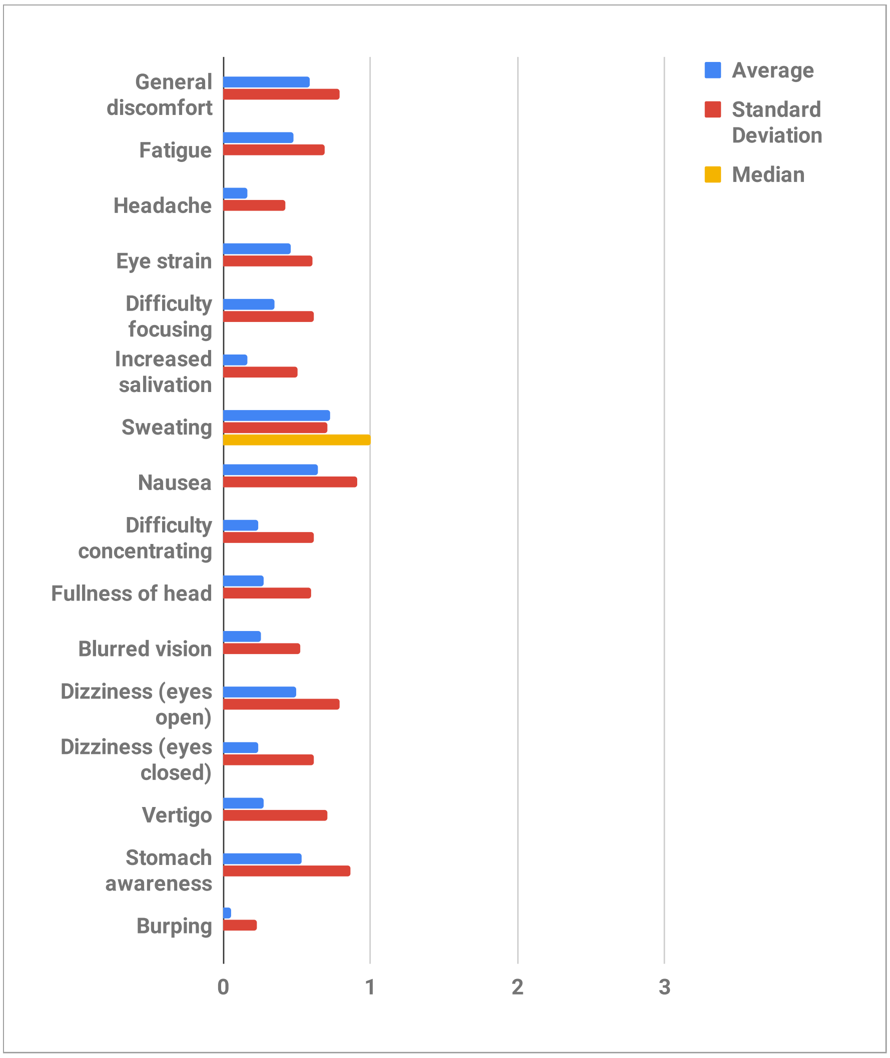


Figure 4.8: Global motion sickness questionnaire's results

The motion sickness questionnaires showed good results. From the results displayed in the figure in 4.8, which combines the answers given by the users to all the different practice scenes tested, it can be observed that the average calculated for each symptom are all under the value of '1', which is a very positive result.

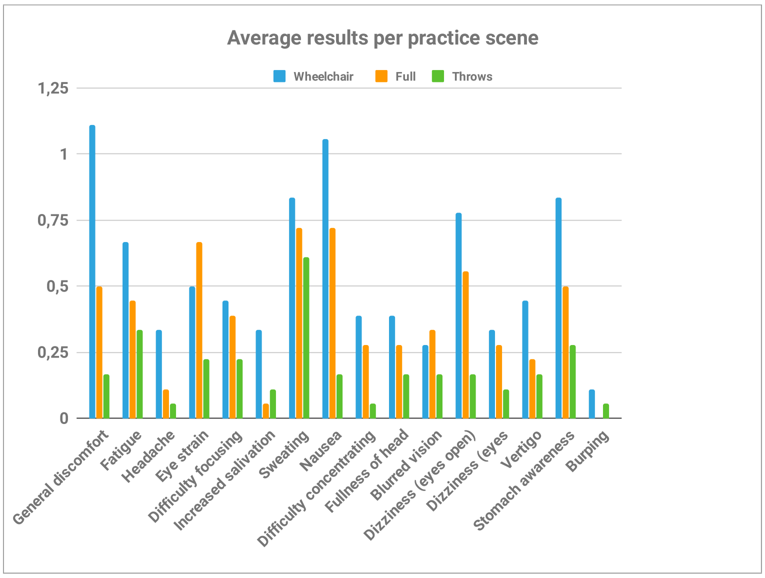


Figure 4.9: Motion sickness questionnaire's results per practice scene

Figure 4.9 shows the analyses of the questionnaire's results separated by practice scene. These results prove that, as expected, the wheelchair practice scene, in which the user is forced to move by maneuvering the virtual wheelchair, has the highest motion sickness symptoms values, and the throws practice scene, in which the user can only move by using teleport, has the lowest symptoms values. This confirms that the maneuvering of the wheelchair is one of the main instigators of motion sickness. Despite being the practice scene in which motion sickness was felt more, even the wheelchair practice results are positive. The only symptoms with an average result above '1' are *General Discomfort* and *Nausea*, and even those only surpass it slightly. This suggests that the virtual wheelchair maneuvering technique developed was successful, although it should still be improved further.

	N	O	D	TS
Global	28,26666667	19,37111111	35,57333333	30,19703704
Wheelchair	44,52	28,21444444	51,81333333	45,29555556
Full	26,5	20,63444444	38,66666667	30,95888889
Throws	13,78	9,264444444	16,24	14,33666667

Table 4.1: Motion sickness scores

As explained in 2.4.2 of the related work, the three subscales considered in Robert S. Kennedy et al.'s simulator sickness questionnaire are the Oculomotor (**O**), Disorientation (**D**), and Nausea (**N**). The Oculomotor includes the *Eyestrain*, *Difficulty Focusing*, *Blurred*

Vision and *Headache* symptoms. The Disorientation includes the *Dizziness* and *Vertigo* symptoms. Finally, the Nausea includes the *Nausea*, *Stomach Awareness*, *Increased Salivation* and *Burping* symptoms. The scores of each of these subscales were calculated for each of the practice scenes and for the global results, and are exhibited in table 4.1. The total severity index (**TS**) was also computed, using the already referred subscales, with the goal of obtaining a overall extent of symptom severity.

The calculated scores can be used to compare the motion sickness that results from playing in each practice scene. The wheelchair practice values are higher in all the scores (**N, O, D,** and **TS**), and the throws practice values are lower in all scores. These values help to show once more that the wheelchair practice is the one that induces more motion sickness and the throws practice the once that induces the least. Once again this shows that the most noticeable cause of motion sickness in the system is the maneuvering of the virtual wheelchair. The higher Disorientation scores points to issues with the fact that the user is seated still while he/she is provided visual cues that show him/her to be in motion.

The scores obtained can be compared with the ones from other VR systems. Robert S. Kennedy et al. [23] present the results of experiments with virtual environment (VE) devices which can be used for comparison. In these results, the total score for VE systems ranges from 19 to 55. Most of the VE systems showed less Oculomotor symptoms than the other two sets of symptoms, and all of them appear to exhibit a significant amount of Disorientation. This pattern of symptomatology is consistent with the one obtained from the final user tests. The **TS** values obtained are also according to the ones of the VE systems used for comparison, with all but the throws practice value being within the 19 to 55 range. The throws practice **TS** score is lower than 19, which indicates that it induces very little motion sickness in users.

The scores calculated in these final user tests may be used for comparison with future upgrades of the system to help determine whether the modifications reduced or, at least, did not increase the motion sickness rates.

4.2.2.4 Observation and Interview Results

Several things can be concluded as consequence of the observation of the users' actions while testing the system and from their feedback and suggestions during the interviews. Of particular importance is the rising of aspects that could still be improved.

Regarding the menus, nearly every user had no problem in finding and interacting with them, both the main menu and the application menu present within the video and practice scenes. The few who, at first, had trouble knowing how to select an option of the application menu eventually figured it out and afterwards had no issues.

As already explained in 3.2.3 and 3.2.5.1, when the user chooses to go to the "Full Practice" scene he/she is presented with a video of the basketball court before actually going into the practice scene, which does not happen with the other practice scenes. The presence of this video appears to have initially confused some users, who were unsure whether the practice had already started and if they were supposed to do something. However, most simply interpreted the video as being a loading screen and did not find it strange at all. Either way the users realised that the court present in the video was the same as the computer-generated one in the practice scene, and most assumed the video was to give time for the users to notice the environment they were going to play in and help them situate themselves in it.

The virtual environment present in the practice scenes appears to have been successful, since the users were pleased with it, remarking that it seemed natural and was immersive. When asked, most said they had not noticed that some walls were made with pictures and others with video clips. However, some users mentioned that one of the walls (a dark one) was weird, partially because pixels could be noticed in it. The environment sounds were also mentioned as helping causing the sense of immersion, and a user suggested that sound should be added to the wheelchair's movements.

A fact that contributed to the success of the environment was the presence of interactable elements such as the basketball ball and the traffic cones. Most users mentioned that they enjoyed the cones, especially those who tried to grab and throw them. Some users also said they wished there were other similar elements with which they could interact. Several users also tried to interact with the NPC by attempting to touch him or push him around, and were disappointed when that had no effect.

While most people had a positive view on the NPC, liking the idea of having a character to play with and which constantly gives them the ball, a short amount of them got scared by him when he appeared too close to them. A few also disliked him stealing the ball from them when he gets close. This issue rises when the NPC is trying to defend its backboard. While positioning himself between the user and the backboard, when the user is close to the backboard, the NPC ends up getting too close to the user. This could be easily solved by increasing the minimum distance the NPC must keep from the user while defending the backboard.

Some of the test participants also believe that the NPC should not be always moving around and passing the ball and suggested that it should be possible to make the NPC take a break, so that the user could rest or try practicing by himself/herself for a bit.

Regarding the interaction with the basketball ball, the participants were pleased with the ball's behaviour, saying that its movements are very realistic, specially in the way it bounces when it hits the floor or other obstacles. However, nearly every user had trouble with trying to dribble the ball since it very quickly ended up jumping away from them. Despite that, the users did not think this to be a issue with the ball's behaviour, with some saying that it is simply hard to hit the ball in the right moment and with the right force due to not being possible to feel the ball, and others saying that it seams realistic for the ball to easily get away since it is hard to dribble while sitting. Besides, the ball sometimes hits the wheelchair and that makes it bounce away.

Several users initially had trouble catching the ball in the rebound and when the NPC passes it to them, since the ball would collide with their hand before they managed to grab it. However, after some time of practice, nearly all users mastered the ability to catch the ball when they wanted to. A situation that a few participants called attention to when catching the ball is that it sometimes ends up being inside the user's character's arm while held, which is unnatural.

Most users either tried holding and throwing the ball with only one hand from the start, or quickly noticed that it was only possible to grab the ball with only one hand at a time and started doing that with no problem. It was only after being told that it was possible to use the hand not holding the ball to assist in giving impulse when throwing it, that some tried doing that, throwing with both hands. Those who tried it, managed to master it after some practice.

Some people felt that it was very hard to score, although most said they also had difficulty doing it in reality. Of the 18 participants in the study, 7 managed to score, with the highest number of successful throws in a practice being of 5. Therefore, the scoring could be facilitated, maybe by creating the possibility of playing in a easier level in which it would be easier to score, but it is already considered to be successfully implemented.

Despite complaining at first that it was hard to aim to score due to the difficulty of knowing the right amount of force to apply to the ball, which they can not feel, as they practiced, the users' aim increasingly improved and by the end of the throws practice nearly all throws were hitting close to the net.

During the throws practice, a few users initially had some difficulty grabbing a new ball from the spawning ball. This happened due to the fact that only one hand is considered to be over the ball at a time, and only that hand is able to grab a new ball. When a user tries to grab a new ball with one hand, but has the other closer to the spawning ball, he/she often will have trouble. Most user understood the source of the problem by themselves and, afterwards, had no further trouble.

Some users suggested adding haptic feedback to the basketball ball in order to help feel when their hands are in contact with it. A lot of users also suggested creating a tutorial where they could learn, for example, how to throw the ball using both hands, and the ways of locomotion that exist to move around the court.

After finding one of the locomotion methods, most users only learned about the other when they either got to a practice in which the method they already knew was not available, or they were informed that another method existed.

When asked, most participants said they found the teleport easier to use, but that the wheelchair maneuvering felt more realistic. While some preferred using the teleport due to making it easy to quickly travel big distances and not causing motion sickness, other found the wheelchair method to be more enjoyable and natural. For small adjustments in their position and orientation, nearly all users preferred to maneuver the wheelchair.

Very few users discovered by themselves that they could change their orientation by using the controller. However, after being informed that it was a possibility, most found how it was done without any difficulties. Thus, this is one thing that should be mentioned in the tutorial that should be created.

The maneuvering of the wheelchair was difficult at first for some users to control, with some having the issue of it moving faster than they expected and others the issue of it moving too slow. However, after practicing, the users got increasingly more skilled at knowing the right movement and speed to to apply to the wheels in order to create the needed impulse. By the end of the wheelchair practice most users had mastered the control of the wheelchair, with only a few still having issues with the rotational movements. The wheels haptic feedback allowed the participants to push the wheelchair without constantly having to look at the wheels.

While moving the arms around in a wider range, some participants hit objects in the real world, which breaks immersion. This problem happened due to the limited space available to perform the user tests.

An error detected during this study was that sometimes, when the user's character collides with the wall in a certain angle, the system crashes. Two different participants noticed this issue, and one of them managed to replicate it. No solution has been found to this issue so far.

Some participants suggested to improve the system by creating new functionality such as the possibility to customize the user's character and its wheelchair, and adding celebratory sounds and something like, for example, fireworks when the user scores.

Chapter 5

Conclusions and Future Work

This chapter presents the conclusions from the work developed in this thesis, and describes some possibilities and ideas for improving the solution in the future.

5.1 Conclusions

This work resulted in the creation of a virtual reality game about wheelchair basketball, which was the sport this thesis focused on. This game is composed by both 360 videos with didactic information, and a simulator of the sport, that allows the user to practice the sport as if he/she were one of the players.

The system developed has the goal to help people better understand paralympic sports. It does so by allowing the users to perceive the sport from an athlete's perspective. While playing in the simulation, the user has a virtual body of an athlete and is capable of moving through the court, interact with his/her virtual wheelchair, and grab and throw balls.

A challenge in the development of the system was the integration of 360 video with computer-generated elements. To achieve this integration different techniques were applied: fading from video to virtual, use of videos as textures, and the placement of computer-generated elements onto tracked 360 video. For the later, a color detection algorithm had to be implemented in order to track a mark throughout the video frames to know the position in the video that the computer-generated element must overlay in each frame.

From the feedback given by the users during the user tests, it can be concluded that the integration techniques applied were successful, helping in the creation of a virtual environment more realistic and immersive. The placement of computer-generated elements onto tracked 360 video seems to be promising and should be better explored in the future. Improvements to its current implementation should be performed, particularly regarding the amount of time taken to process the video before it starts playing.

An iterative approach was adopted in the design and development of the system. There were two main iterations. After each of these iterations user tests were performed to evaluate the sense of presence, the motion sickness, and the usability of the system. From one iteration to the other, the results showed improvement.

The user test results were positive and showed that the users enjoyed the system developed. Some motion sickness was observed to result from using the system, however the motion sickness values were within the expected range for a virtual reality system. Even so, this aspect can still be improved in the future. The total severity scores calculated to measure the amount of motion sickness experienced by users can be used to compare with future versions of the system.

It can be concluded that the game engine chosen to implement the system (Unity) was an appropriate choice, having allowed for the easy implementation of all required functionalities.

Summing up, the goals of this thesis were met, and all contributions present in section 1.4 were successfully achieved. Even so, both the simulator and the integration techniques developed still have a lot of room for improvement.

5.2 Future Work

Although the system was successfully created, some functionalities should still be further improved. Besides that, in the future, more techniques could be applied and other characteristics added. The use of virtual reality as a medium to tell a story should also be better explored.

One of the things that should be improved is the movement of the virtual wheelchair, in order to reduce the motion sickness resulting from it, and to make it appear more realistic.

Another aspect to improve is the behaviour of the NPC. An attack mode should be added to it, enabling the character to participate in all aspects of a game. Once the NPC is also able to attack, and therefore, try to score, the game should be upgraded so that the user can practice the sport in an actual game scenario, with multiple players divided in two teams. A script should be created to manage the teams, defining which characters belong to which team, and in which backboard each team should score. This information will affect to which character a player will try to pass the ball to.

The possibility of getting points whenever a player scores and keeping track of the score of each team should also be developed. It would also make sense to add a goal to the wheelchair practice scene to motivate the user. For example, elements could be placed along the way the user is expected to go through in his/her practice and, as the user goes through those elements, they are collected and points are gained.

When the user is practicing together with NPCs, as suggested by some participants of the user tests, it should be possible to decide to make the NPCs take a break so that they are not always playing the game, therefore, giving the user time to also take a break or try praticing without the NPCs. Thus the option of taking a break should be added to the menu.

Another suggestion given by some users was that a tutorial should be added to help users learn the different ways to move through the court, and inform them of all the features available.

In the future, the system is also expected to be extended to include other paralympic sports besides wheelchair basketball, such as boccia, for example.

Additionally the system should be improved once further resources, specially more videos, are obtained. For example, the court environment would benefit from using higher quality videos for the walls and ceiling.

Relatively to the integration of 3D objects onto tracked 360 video, other more sophisticated examples should be created. More interesting possibilities of interaction with the video using this technique should also be developed, rather then simply making texts appear or being used to change scenes. For example, the classification video could be used for the user to select the class of the athlete he/she wants to play as, therefore, changing the player character. Within the scene of interaction with this video, instead of using the spheres as the 3D computer-generated objects that appear above the athletes' heads, other objects with a more realistic look could be used, since some participants of the user tests mentioned that what made it obvious that the spheres did not belong in the video was their unrealistic look. The integration technique's implementation itself could also be improved, reducing the initial processing time of the video so that it does not take so long to be ready to be played.

Finally, it would also be interesting to attempt using another technique of 360 video integration with the computer-generated environment. Video of a court with an audience present would be recorded, at least one with the audience passive, another with them celebrating, and one of them sad. In all videos the audience would be sitting in the same place and start with each person in approximately the same position. These videos would then be used to create the environment of a scene in which the user can practice the sport, and be played in response to the user's actions. For example, during most of the time the video of the audience passive would be playing then, once the user's team scored, the video of the audience celebrating would start playing instead. When the adversary team scores, the video of the audience being unhappy would play. This way, the video of the environment would change according to the user's actions, therefore it would be responsive to the user's behaviour, increasing immersion.

At some point in the future, some other hardware could also be added to the system in order to obtain more input, such as the user's body's orientation, and provide additional feedback to the user. A wheelchair simulator, such as the ones mentioned in section 2.2.2 of the related work, using an actual physical wheelchair could be used to allow the user to better interact with the virtual wheelchair, while also providing force feedback. Other hardware capable of, for example, providing haptic feedback to the user's hands would also be helpful. It could be used to allow the user to feel the ball or other objects when these are grabbed, and help determining the force being applied to the ball when throwing it.

Lastly, the writing of a paper about the work developed, and its submission to a conference, is a on going process.

Bibliography

[1] R. M. Baños, C. Botella, I. Rubió, S. Quero, A. García-Palacios, and M. Alcañiz. "Presence and emotions in virtual environments: The influence of stereoscopy." In: *CyberPsychology & Behavior: the impact of the Internet, multimedia and virtual reality on behavior and society* 11.1 (2008), pp. 1–8.

[2] M. Bessa, M. Melo, D. Narciso, L. Barbosa, and J. Vasconcelos-Raposo. "Does 3D 360 video enhance user's VR experience?: An Evaluation Study." In: *Proceedings of the XVII International Conference on Human Computer Interaction*. ACM. 2016, p. 16.

[3] L. Bradley. *VR Wheelchair Basketball Game Created To Raise Paralympic Awareness*. Accessed: 16 Aug. 2018. 2018. URL: https://www.sporttechie.com/the-img-studio-creates-fully-immersive-vr-wheelchair-basketball-game/.

[4] S. Bruck and P. A. Watters. "Estimating cybersickness of simulated motion using the simulator sickness questionnaire (SSQ): A controlled study." In: *Computer Graphics, Imaging and Visualization, 2009. CGIV'09. Sixth International Conference on*. IEEE. 2009, pp. 486–488.

[5] G. Calleja. "Immersion in virtual worlds." In: *The Oxford handbook of virtuality* (2014), pp. 222–236.

[6] M. R. de Carvalho, R. T. da Costa, and A. E. Nardi. "Simulator Sickness Questionnaire: tradução e adaptação transcultural." In: *J Bras Psiquiatr* 60.4 (2011), pp. 247–52.

[7] A. Covaci, A.-H. Olivier, and F. Multon. "Visual perspective and feedback guidance for vr free-throw training." In: *IEEE computer graphics and applications* 35.5 (2015), pp. 55–65.

[8] L. Crichlow, G. Fernie, J. Campos, and P. Grant. "A FULL MOTION MANUAL WHEELCHAIR SIMULATOR FOR REHABILITATION RESEARCH." In: ().

[9] C. Cruz-Neira, D. J. Sandin, and T. A. DeFanti. "Surround-screen projection-based virtual reality: the design and implementation of the CAVE." In: *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*. ACM. 1993, pp. 135–142.

[10] G. Developers. *Google VR, Cardboard*. Accessed: 24 Jan. 2018. URL: https://developers.google.com/vr/discover/cardboard.

[11] A. Dix, J. E. Finlay, G. D. Abowd, and R. Beale. “Human-Computer Interaction.” In: Prentice Hall, 2003. Chap. 9.

[12] S. I. Entertainment. *Playstation VR*. Accessed: 26 Jan. 2018. URL: https://www.playstation.com/pt-pt/explore/playstation-vr/.

[13] G. C. Foundation. *The Now: What is 360 Video?* Accessed: 31 Jan. 2018. URL: https://www.gcflearnfree.org/thenow/what-is-360-video/1/.

[14] P. J. Gianaros, E. R. Muth, J. T. Mordkoff, M. E. Levine, and R. M. Stern. “A questionnaire for the assessment of the multiple dimensions of motion sickness.” In: *Aviation, space, and environmental medicine* 72.2 (2001), p. 115.

[15] I. Gonzalez. *IMG Studio Develops First VR Basketball Game for Adaptive, Paralympic Sports*. Accessed: 16 Aug. 2018. 2018. URL: https://www.startupssanantonio.com/img-studio-develops-first-vr-basketball-game-for-adaptive-paralympic-sports/.

[16] S. Gradl, B. M. Eskofier, D. Eskofier, C. Mutschler, and S. Otto. “Virtual and augmented reality in sports: an overview and acceptance study.” In: *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct*. ACM. 2016, pp. 885–888.

[17] M Grant, C Harrison, and B Conway. “Wheelchair simulation.” In: *Cambridge Workshop Series on Universal Access and Assistive Technology*. 2004.

[18] R. Heijdens and H. B. Lee. *360 Video & Virtual Reality (VR) in JW Player Part 1 : State of the Industry*. Accessed: 11 Feb. 2018. 2016. URL: https://www.jwplayer.com/blog/360-vr-part1-state-of-the-industry/.

[19] V. Interrante, B. Ries, and L. Anderson. “Seven league boots: A new metaphor for augmented locomotion through moderately large scale immersive virtual environments.” In: *3D User Interfaces, 2007. 3DUI’07. IEEE Symposium on*. IEEE. 2007.

[20] M. Journalism. *Gone Gitmo*. Accessed: 8 Feb. 2018. URL: http://www.multiplejournalism.org/case/gone-gitmo.

[21] V. Kalivarapu, A. MacAllister, M. Hoover, S. Sridhar, J. Schlueter, A. Civitate, P. Thompkins, J. Smith, J. Hoyle, J. Oliver, et al. “Game-day football visualization experience on dissimilar virtual reality platforms.” In: *SPIE/IS&T Electronic Imaging*. International Society for Optics and Photonics. 2015, pp. 939202–939202.

[22] R. S. Kennedy, N. E. Lane, K. S. Berbaum, and M. G. Lilienthal. “Simulator sickness questionnaire: An enhanced method for quantifying simulator sickness.” In: *The international journal of aviation psychology* 3.3 (1993), pp. 203–220.

[23] R. S. Kennedy, J. M. Drexler, D. E. Compton, K. M. Stanney, D. S. Lanham, and D. L. Harm. "Configural Scoring of Simulator Sickness, Cybersickness and Space Adaptation Syndrome: Similarities and Differences." In: *Virtual and adaptive environments: Applications, implications, and human performance issues* (2003), p. 247.

[24] A. Kronqvist, J. Jokinen, and R. Rousi. "Evaluating the Authenticity of Virtual Environments." In: *Advances in Human-Computer Interaction* 2016 (2016), p. 3.

[25] H. C. Miles, S. R. Pop, S. J. Watt, G. P. Lawrence, and N. W. John. "A review of virtual environments for training in ball sports." In: *Computers & Graphics* 36.6 (2012), pp. 714–726.

[26] D. C. Niehorster, L. Li, and M. Lappe. "The Accuracy and Precision of Position and Orientation Tracking in the HTC Vive Virtual Reality System for Scientific Research." In: *i-Perception* 8.3 (2017), p. 2041669517708205.

[27] J. Nielsen and R. Molich. "Heuristic evaluation of user interfaces." In: *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM. 1990, pp. 249–256.

[28] R. do Oculus. *Oculus*. Accessed: 25 Jan. 2018. URL: https://www.oculus.com/.

[29] T. Pithon, T. Weiss, S. Richir, and E. Klinger. "Wheelchair simulators: A review." In: *Technology and Disability* 21.1, 2 (2009), pp. 1–10.

[30] L. Prasuethsut. *HTC Vive review*. Accessed: 10 Feb. 2018. URL: https://www.wareable.com/vr/htc-vive-review.

[31] S Pravenaa and R Menaka. "A methodical review on image stitching and video stitching techniques." In: *International Journal of Applied Engineering Research* 11.5 (2016), pp. 3442–3448.

[32] J. D. Ribeiro, B. M. Faria, A. P. Moreira, and L. P. Reis. "Realistic Boccia Game Simulator Adapted for People with Disabilities or Motor Disorders: Architecture and Preliminary Usability Study." In: *World Conference on Information Systems and Technologies*. Springer. 2017, pp. 165–176.

[33] P. Salamin, D. Thalmann, and F. Vexo. "The benefits of third-person perspective in virtual and augmented reality?" In: *Proceedings of the ACM symposium on Virtual reality software and technology*. ACM. 2006, pp. 27–30.

[34] T. Schubert, F. Friedmann, and H. Regenbrecht. *igroup presence questionnaire (IPQ) overview*. Accessed: 24 Jan. 2018. URL: http://www.igroup.org/pq/ipq/index.php.

[35] M. J. Schuemie, P. Van Der Straaten, M. Krijn, and C. A. Van Der Mast. "Research on presence in virtual reality: A survey." In: *CyberPsychology & Behavior* 4.2 (2001), pp. 183–201.

[36] B. S. Sidhu. *Light Field Capture - A Paradigm Shift for Cinematography, Visual FX & Live-Action Virtual Reality*. Accessed: 31 Jan. 2018. 2015. URL: https://virtualrealitypop.com/light-field-capture-a-paradigm-shift-for-cinematography-visual-effects-and-live-action-vr-a946055adc8f1.

[37] G. R. Silva, J. C. Donat, M. M. Rigoli, F. R. de Oliveira, and C. H. Kristensen. "A questionnaire for measuring presence in virtual environments: factor analysis of the presence questionnaire and adaptation into Brazilian Portuguese." In: *Virtual Reality* 20.4 (2016), pp. 237–242.

[38] M. Slater. "A note on presence terminology." In: *Presence connect* 3.3 (2003), pp. 1–5.

[39] M. Slater and M. V. Sanchez-Vives. "Enhancing our lives with immersive virtual reality." In: *Frontiers in Robotics and AI* 3 (2016), p. 74.

[40] M. Slater and S. Wilbur. "A framework for immersive virtual environments (FIVE): Speculations on the role of presence in virtual environments." In: *Presence: Teleoperators and virtual environments* 6.6 (1997), pp. 603–616.

[41] M. Slater, B. Spanlang, M. V. Sanchez-Vives, and O. Blanke. "First person experience of body transfer in virtual reality." In: *PloS one* 5.5 (2010), e10564.

[42] D. Software. *360 Video Format: Video Audio Formats for 360 VR Video*. Accessed: 01 Feb. 2018. URL: https://www.macxdvd.com/online-video/360-video-format-vr-audio.htm.

[43] D. Software. *What Are the Virtual Reality Video Formats/Codecs You'll Find and Choose?* Accessed: 01 Feb. 2018. URL: https://www.winxdvd.com/resource/best-virtual-reality-video-formats.htm.

[44] L. Y. Sørensen and J. P. Hansen. "A low-cost virtual reality wheelchair simulator." In: *Proceedings of the 10th International Conference on PErvasive Technologies Related to Assistive Environments*. ACM. 2017, pp. 242–243.

[45] T. I. Studio. *CASE STUDY: VR WHEELCHAIR BASKETBALL*. Accessed: 16 Aug. 2018. 2018. URL: https://theimgstudio.com/virtual-reality/.

[46] J. Studios. *The Cinematic VR Field Guide: A Guide to Best Practices for Shooting in 360°*. Available at: https://www.jauntvr.com/cdn/uploads/jaunt-vr-field-guide.pdf. 2017.

[47] A. Sutcliffe and B. Gault. "Heuristic evaluation of virtual reality applications." In: *Interacting with computers* 16.4 (2004), pp. 831–849.

[48] M. Swider. *HTC Vive vs Oculus Rift: which VR headset is better?* Accessed: 26 Jan. 2018. URL: http://www.techradar.com/news/wearables/htc-vive-vs-oculus-rift-1301375/.

[49] U. Technologies. *Unity - Manual: Sprites*. Accessed: 5 Feb. 2018. URL: https://docs.unity3d.com/Manual/Sprites.html.

[50] U. Technologies. *Unity User Manual*. Accessed: 28 Aug. 2018. URL: https://docs.unity3d.com/Manual/.

[51] T. Tsen. "Video Stitching Literature Review." In: (2014).

[52] UNIT9. *O2: Wear The Rose*. Accessed: 7 Feb. 2018. URL: https://www.unit9.com/project/wear-the-rose/.

[53] M. Usoh, E. Catena, S. Arman, and M. Slater. "Using presence questionnaires in reality." In: *Presence: Teleoperators and Virtual Environments* 9.5 (2000), pp. 497–503.

[54] VIVE. *Discover Virtual Reality Beyond Imagination*. Accessed: 26 Jan. 2018. URL: https://www.vive.com/us//.

[55] VRfix. *Inject 360 video metadata on a mobile for YouTube or Facebook*. Accessed: 11 Feb. 2018. URL: http://vrfix.me/.

[56] Waves. *Ambisonics Explained: A Guide for Sound Engineers*. Accessed: 31 Jan. 2018. 2017. URL: https://www.waves.com/ambisonics-explained-guide-for-sound-engineers.

[57] B. G. Witmer and M. J. Singer. "Measuring presence in virtual environments: A presence questionnaire." In: *Presence: Teleoperators and virtual environments* 7.3 (1998), pp. 225–240.

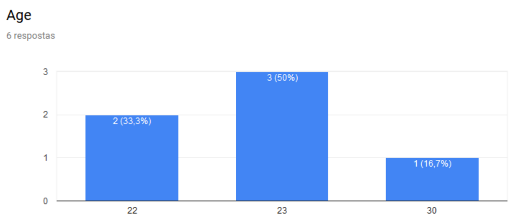
[58] B. G. Witmer, C. J. Jerome, and M. J. Singer. "The factor structure of the presence questionnaire." In: *Presence: Teleoperators and Virtual Environments* 14.3 (2005), pp. 298–312.

[59] H. Zhang. "Head-mounted display-based intuitive virtual reality training system for the mining industry." In: *International Journal of Mining Science and Technology* (2017).

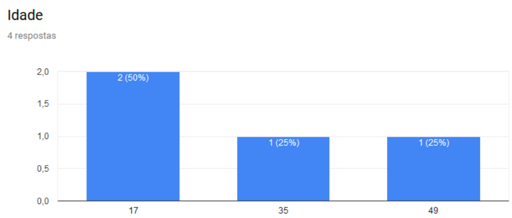
APPENDIX

A

Questionnaire Results of Preliminary Tests

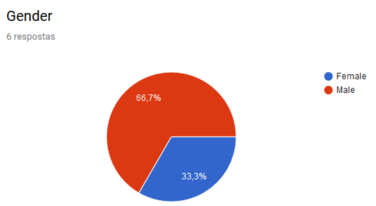


(a) Students' results

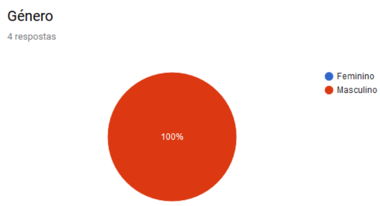


(b) Athletes' results

Figure A.1: Age

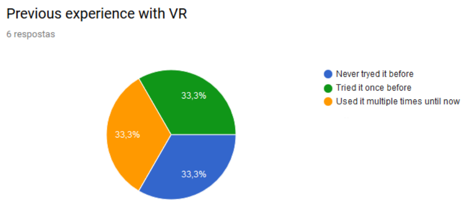


(a) Students' results

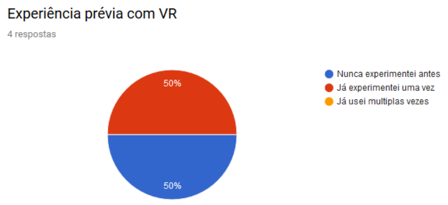


(b) Athletes' results

Figure A.2: Gender

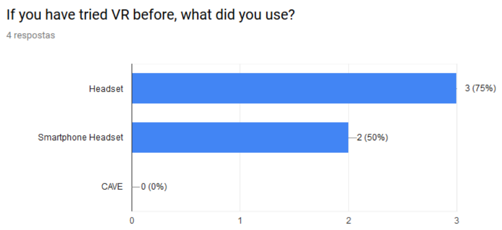


(a) Students' results

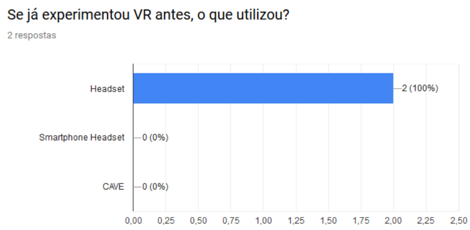


(b) Athletes' results

Figure A.3: Previous experience with VR

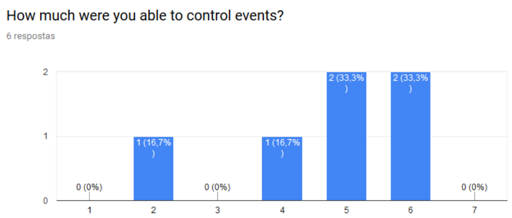


(a) Students' results

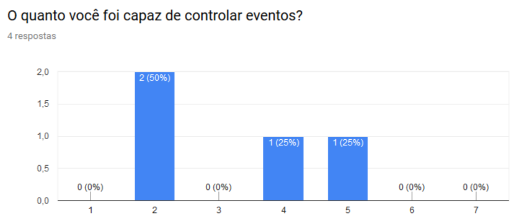


(b) Athletes' results

Figure A.4: If you have tried VR before, what did you use?

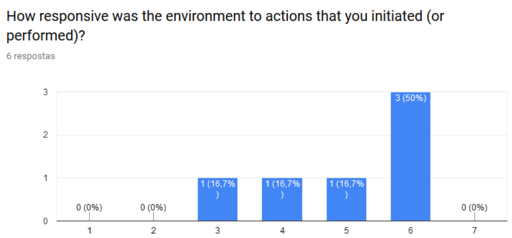


(a) Students' results

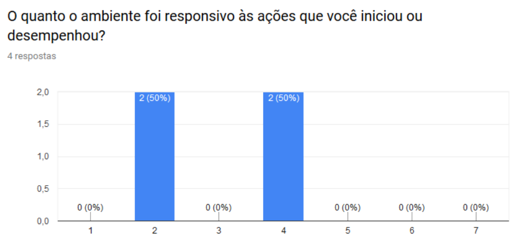


(b) Athletes' results

Figure A.5: How much were you able to control events?

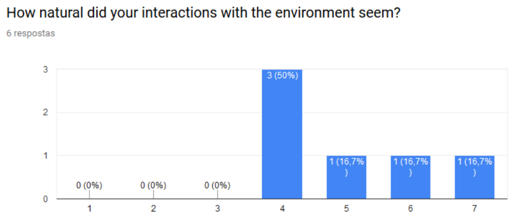


(a) Students' results

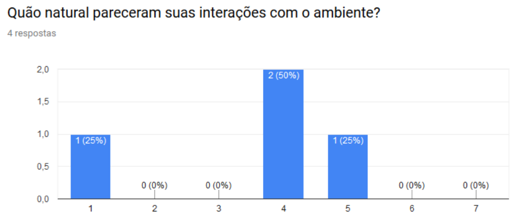


(b) Athletes' results

Figure A.6: How responsive was the environment to actions that you initiated (or performed)?

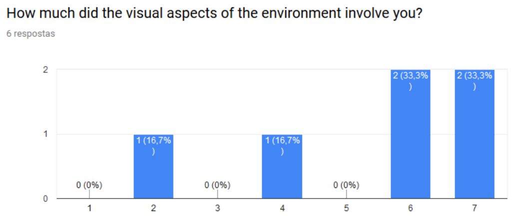


(a) Students' results

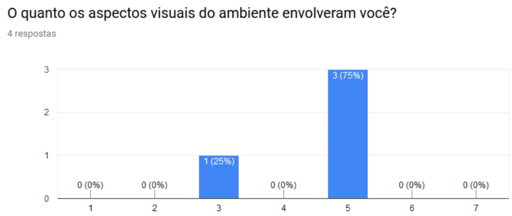


(b) Athletes' results

Figure A.7: How natural did your interactions with the environment seem?

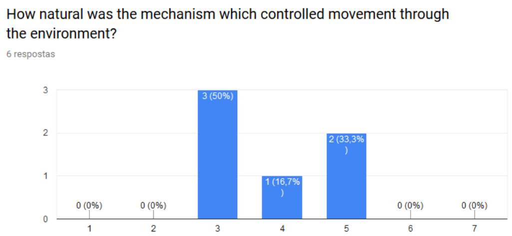


(a) Students' results

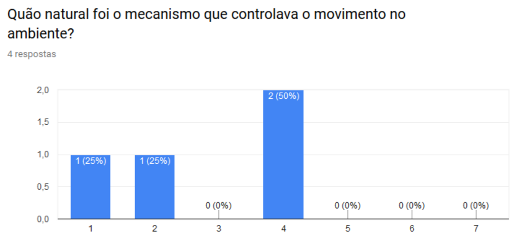


(b) Athletes' results

Figure A.8: How much did the visual aspects of the environment involve you?

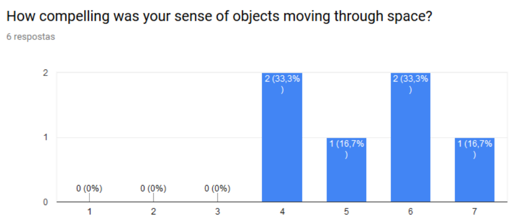


(a) Students' results

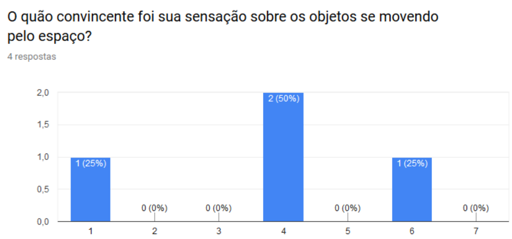


(b) Athletes' results

Figure A.9: How natural was the mechanism which controlled movement through the environment?

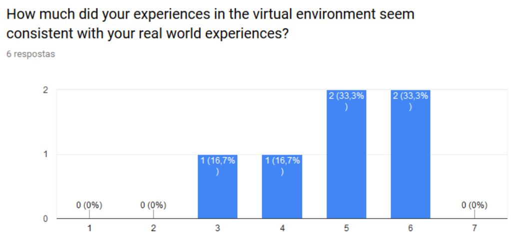


(a) Students' results

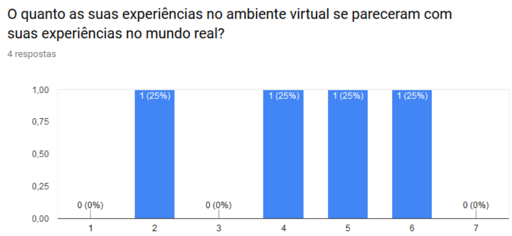


(b) Athletes' results

Figure A.10: How compelling was your sense of objects moving through space?

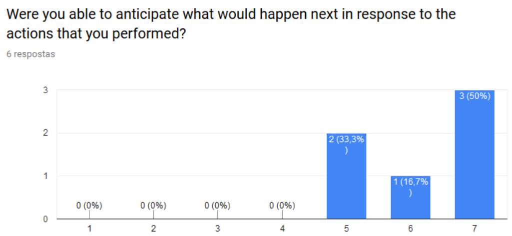


(a) Students' results

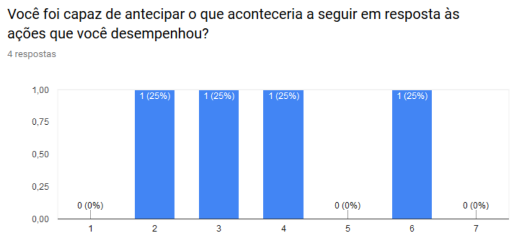


(b) Athletes' results

Figure A.11: How much did your experiences in the virtual environment seem consistent with your real world experiences?

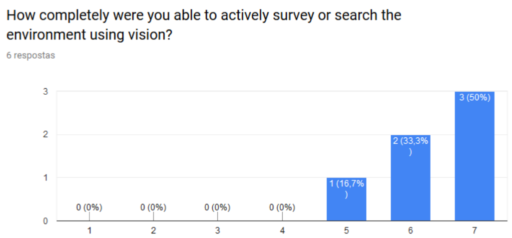


(a) Students' results

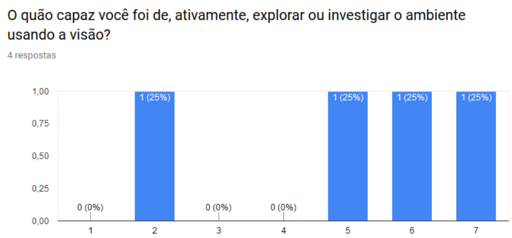


(b) Athletes' results

Figure A.12: Were you able to anticipate what would happen next in response to the actions that you performed?

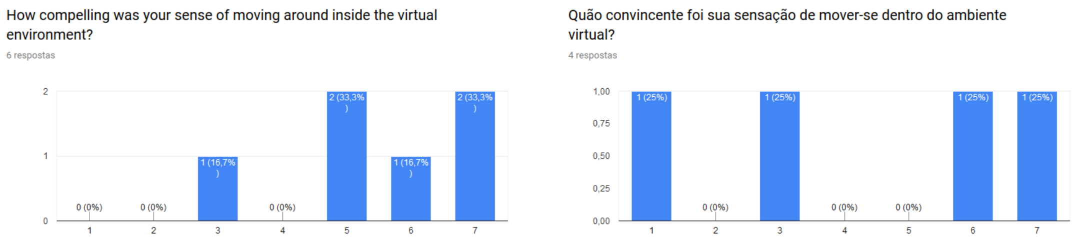


(a) Students' results



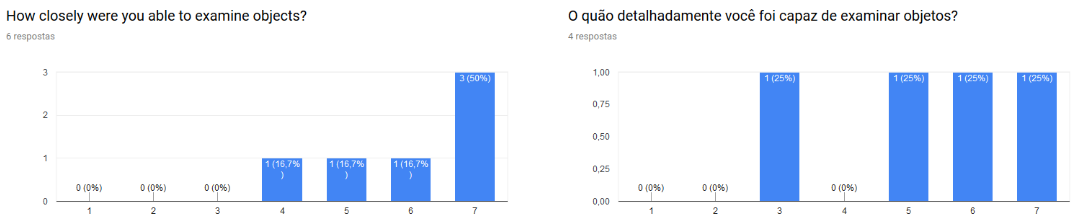
(b) Athletes' results

Figure A.13: How completely were you able to actively survey or search the environment using vision?



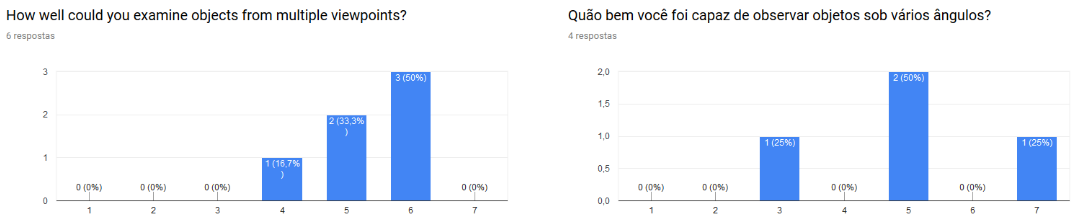
(a) Students' results (b) Athletes' results

Figure A.14: How compelling was your sense of moving around inside the virtual environment?



(a) Students' results (b) Athletes' results

Figure A.15: How closely were you able to examine objects?

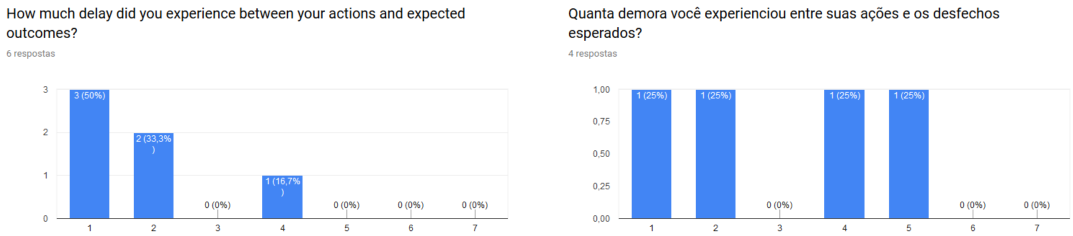


(a) Students' results (b) Athletes' results

Figure A.16: How well could you examine objects from multiple viewpoints?

(a) Students' results (b) Athletes' results

Figure A.17: How involved were you in the virtual environment experience?

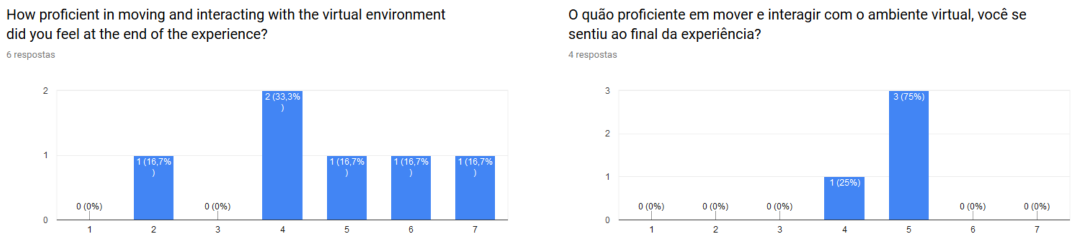


(a) Students' results (b) Athletes' results

Figure A.18: How much delay did you experience between your actions and expected outcomes?

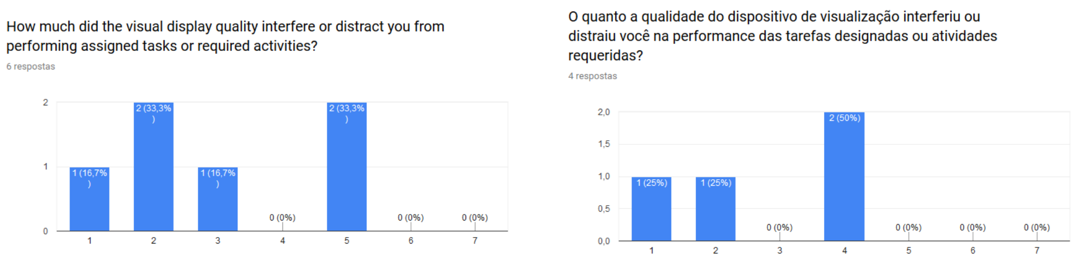
(a) Students' results (b) Athletes' results

Figure A.19: How quickly did you adjust to the virtual environment experience?



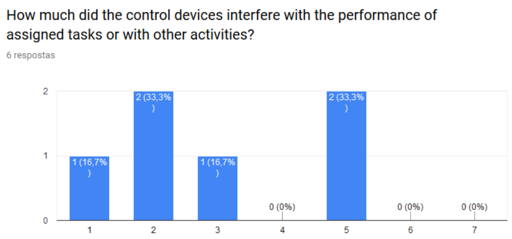
(a) Students' results (b) Athletes' results

Figure A.20: How proficient in moving and interacting with the virtual environment did you feel at the end of the experience?

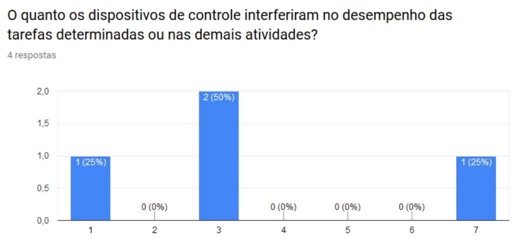


(a) Students' results (b) Athletes' results

Figure A.21: How much did the visual display quality interfere or distract you from performing assigned tasks or required activities?

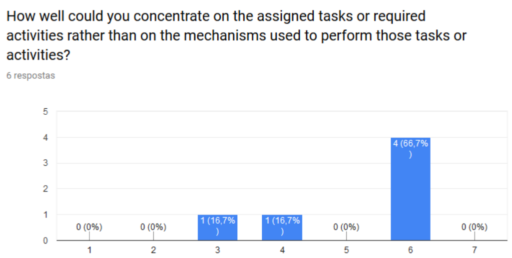


(a) Students' results

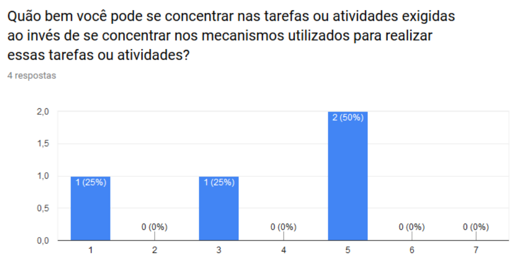


(b) Athletes' results

Figure A.22: How much did the control devices interfere with the performance of assigned tasks or with other activities?

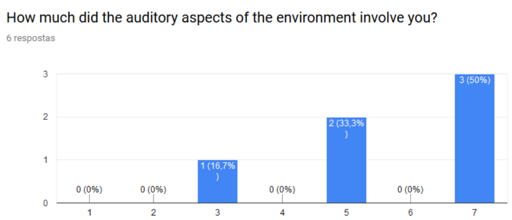


(a) Students' results

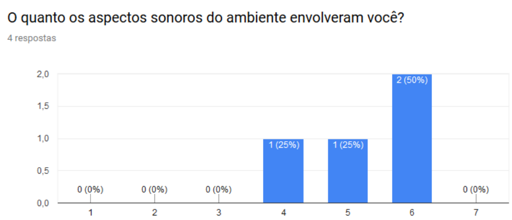


(b) Athletes' results

Figure A.23: How well could you concentrate on the assigned tasks or required activities rather than on the mechanisms used to perform those tasks or activities?

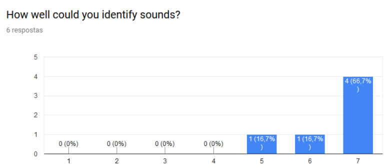


(a) Students' results

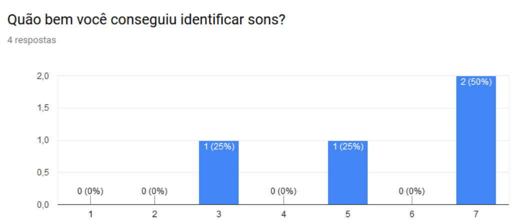


(b) Athletes' results

Figure A.24: How much did the auditory aspects of the environment involve you?

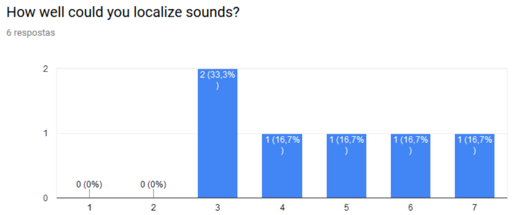


(a) Students' results

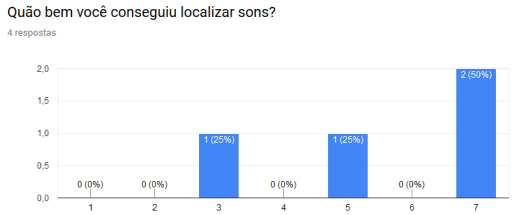


(b) Athletes' results

Figure A.25: How well could you identify sounds?

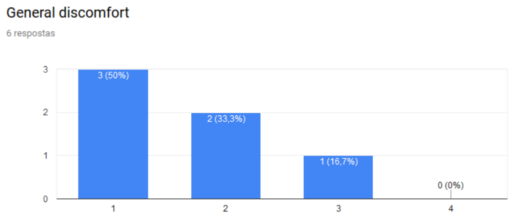


(a) Students' results

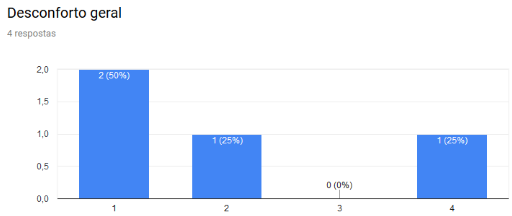


(b) Athletes' results

Figure A.26: How well could you localize sounds?

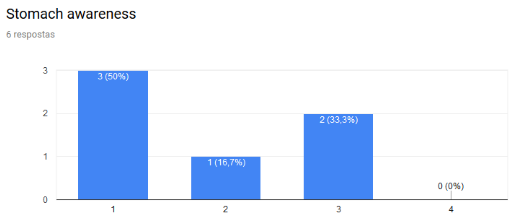


(a) Students' results

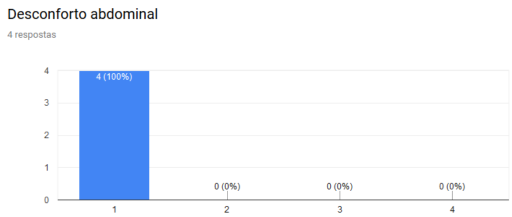


(b) Athletes' results

Figure A.27: General discomfort

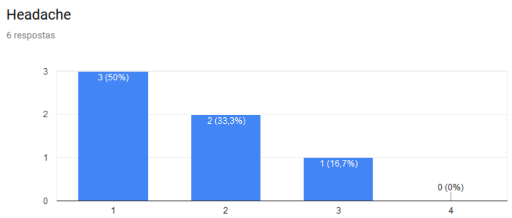


(a) Students' results

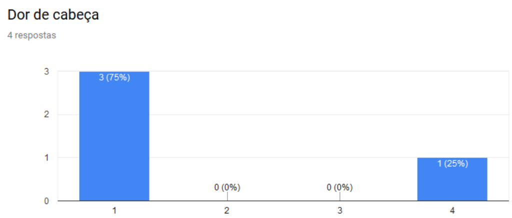


(b) Athletes' results

Figure A.28: Stomach awareness

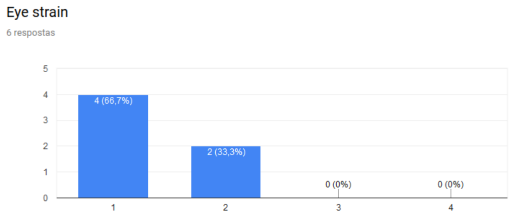


(a) Students' results

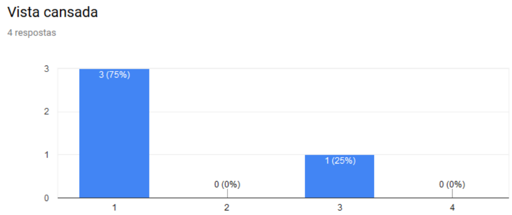


(b) Athletes' results

Figure A.29: Headache

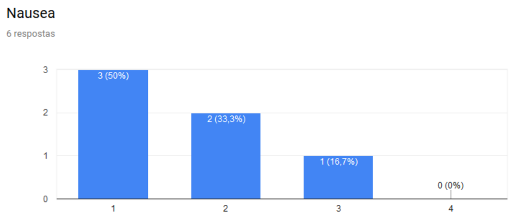


(a) Students' results

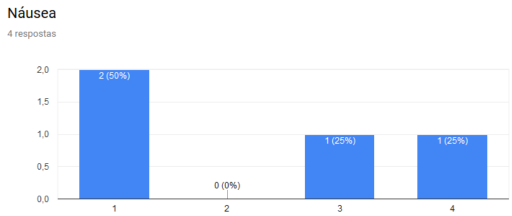


(b) Athletes' results

Figure A.30: Eye strain

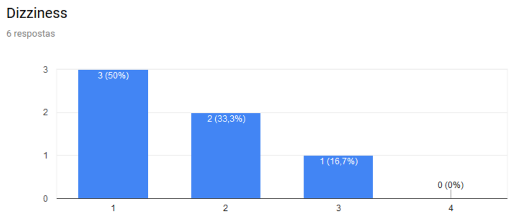


(a) Students' results

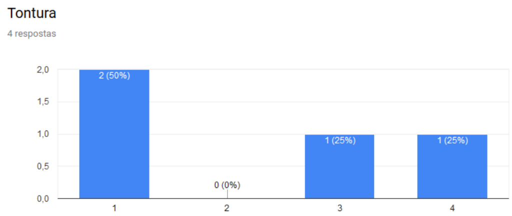


(b) Athletes' results

Figure A.31: Nausea



(a) Students' results



(b) Athletes' results

Figure A.32: Dizziness

Appendix B

Questionnaire Results of Final Tests

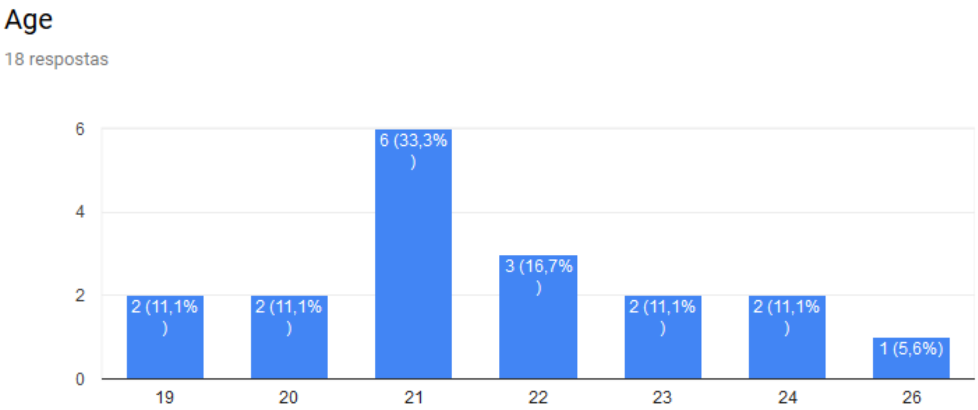


Figure B.1: Age

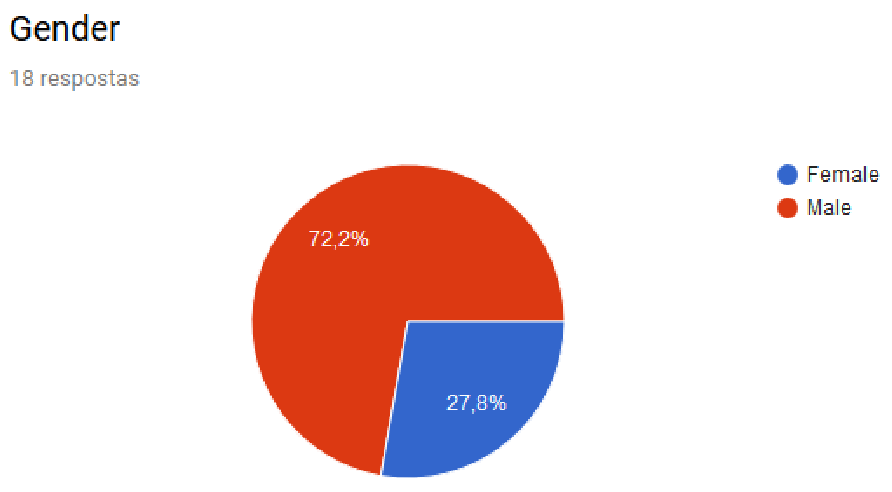


Figure B.2: Gender

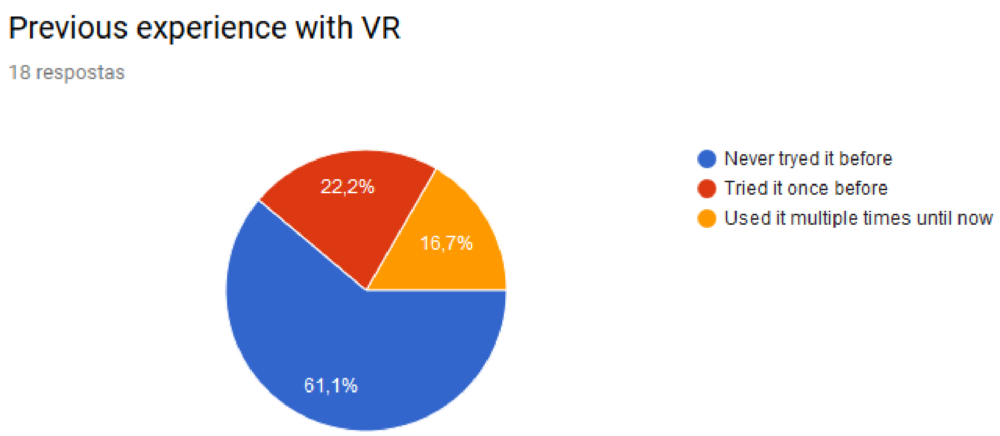


Figure B.3: Previous experience with VR

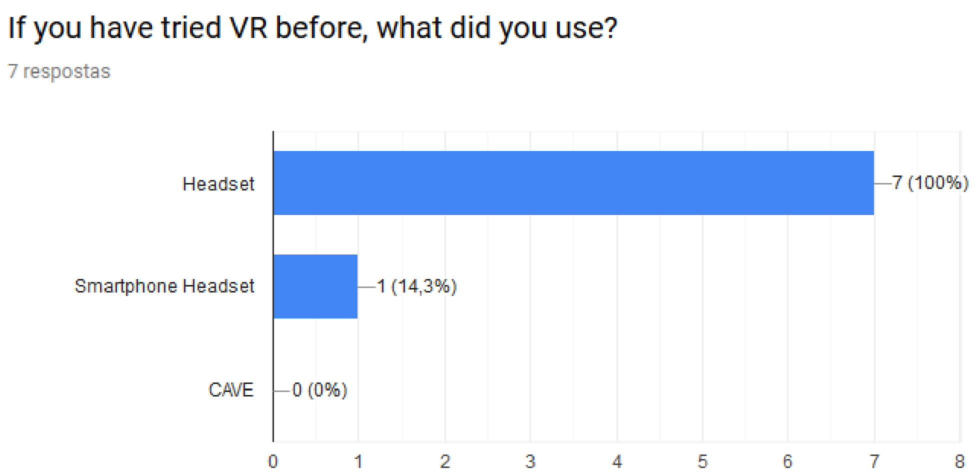


Figure B.4: If you have tried VR before, what did you use?

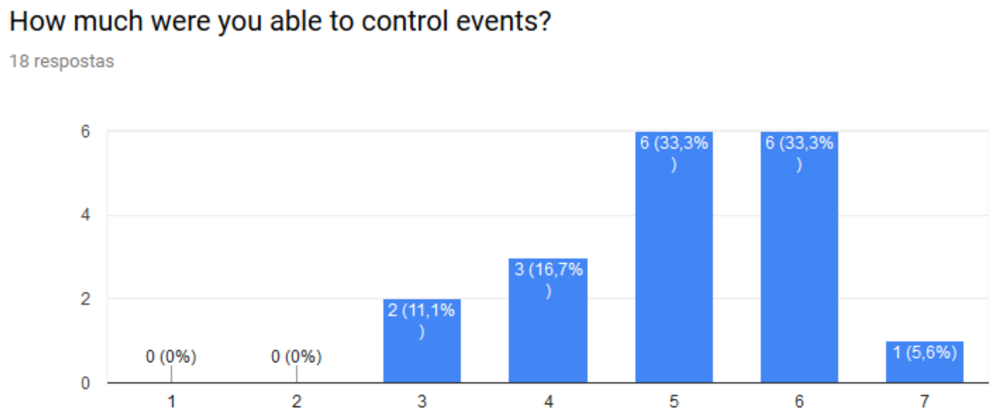


Figure B.5: How much were you able to control events?

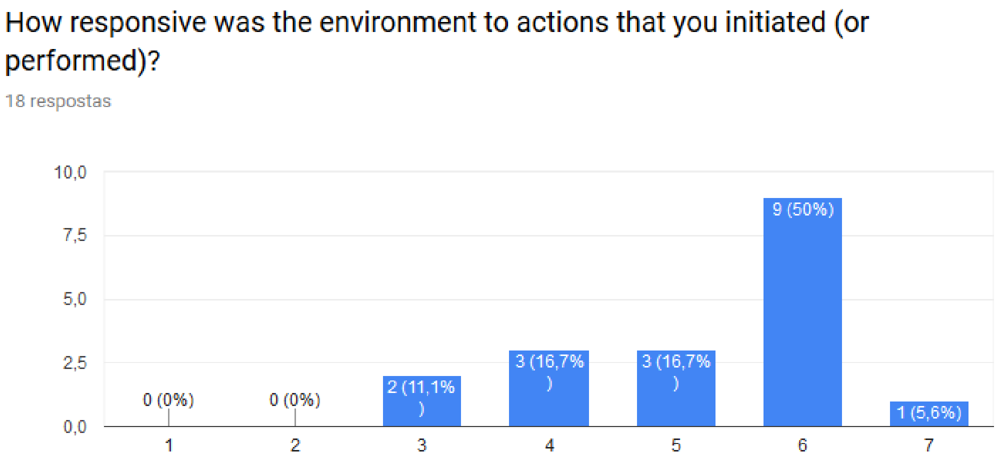


Figure B.6: How responsive was the environment to actions that you initiated (or performed)?

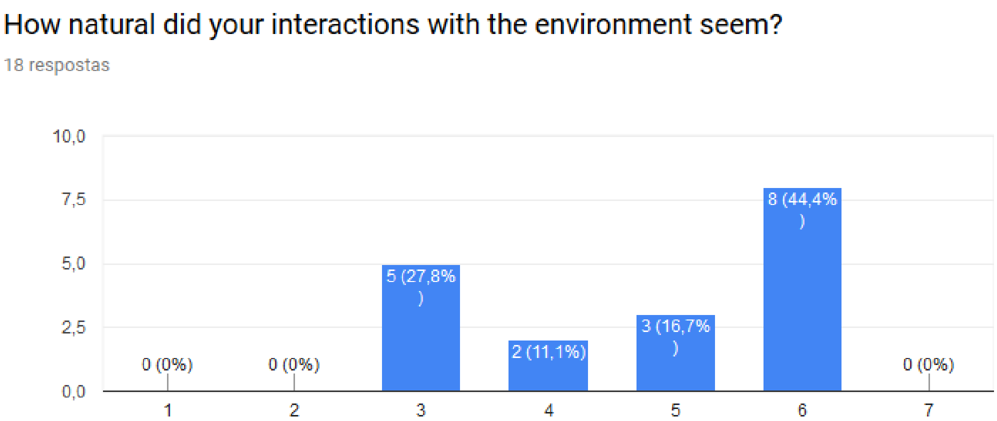


Figure B.7: How natural did your interactions with the environment seem?

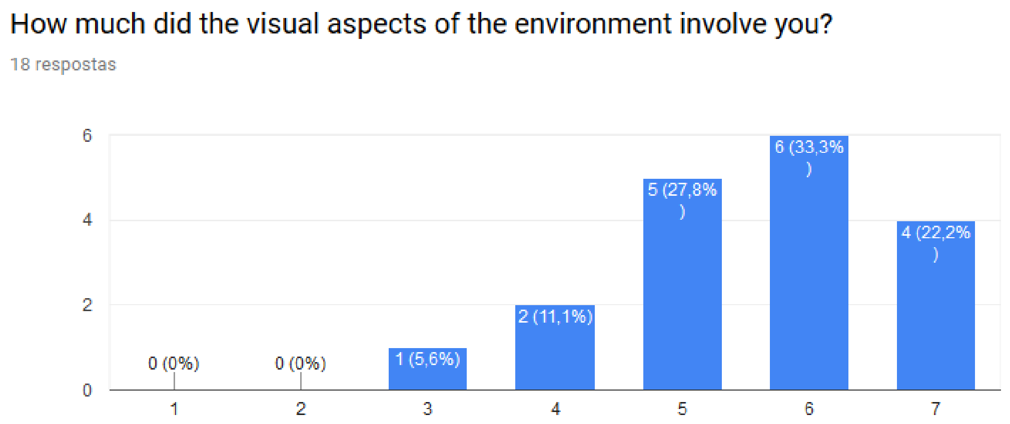


Figure B.8: How much did the visual aspects of the environment involve you?

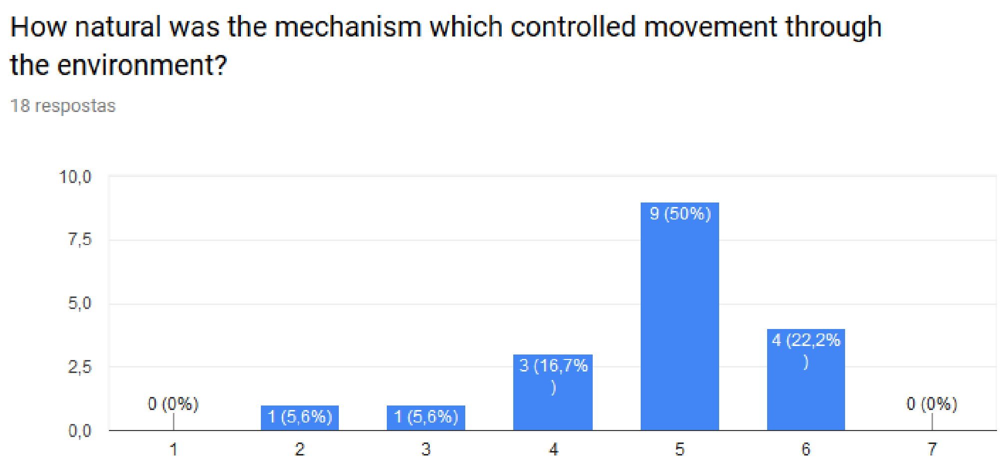


Figure B.9: How natural was the mechanism which controlled movement through the environment?

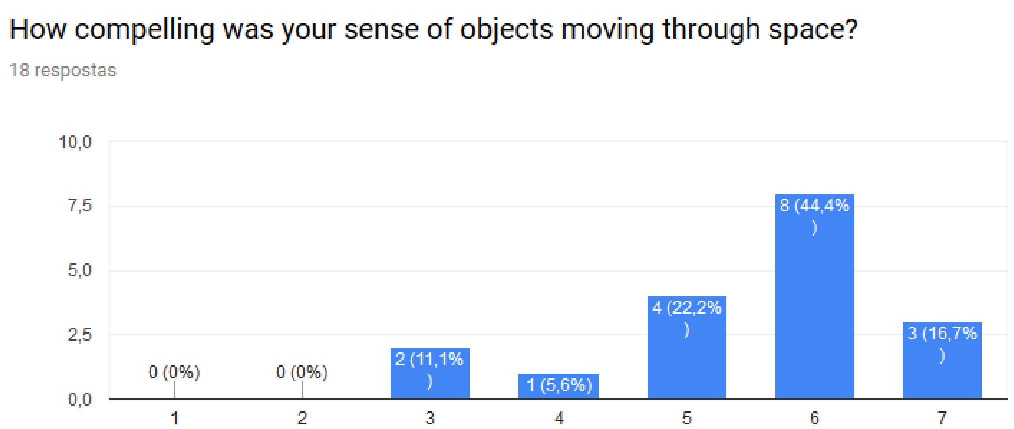


Figure B.10: How compelling was your sense of objects moving through space?

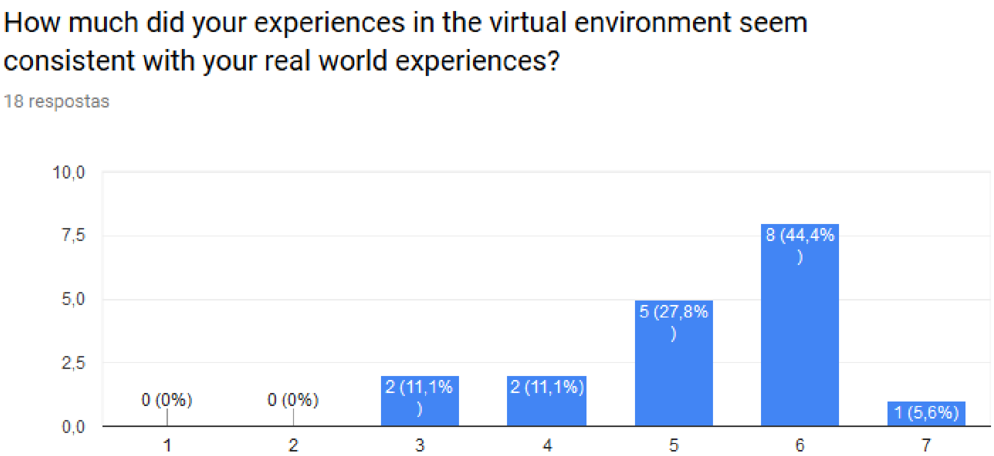


Figure B.11: How much did your experiences in the virtual environment seem consistent with your real world experiences?

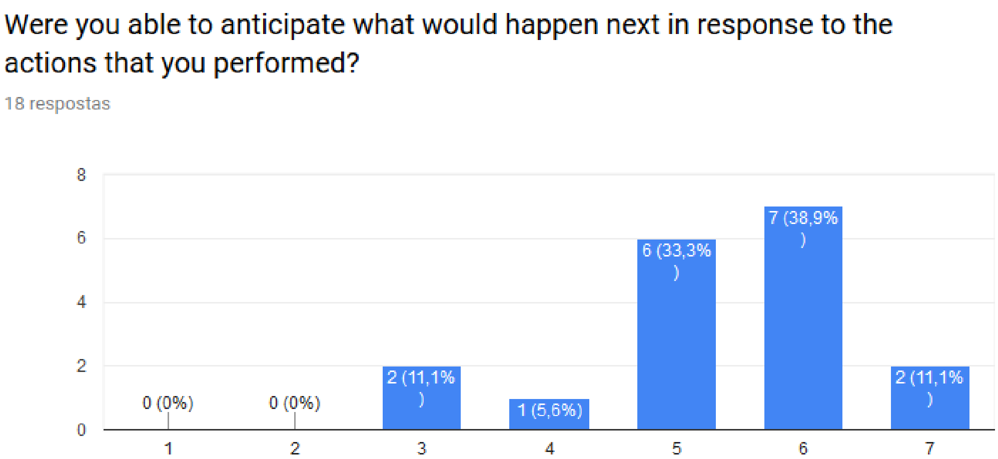


Figure B.12: Were you able to anticipate what would happen next in response to the actions that you performed?

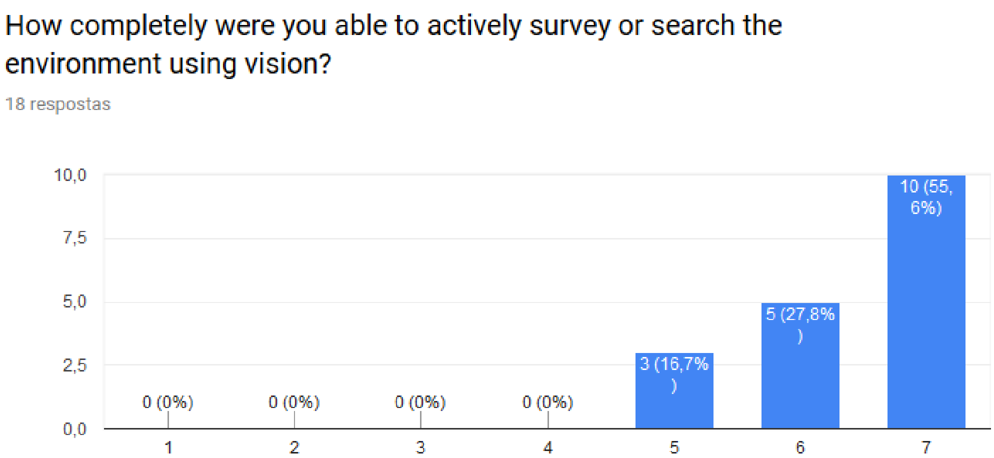


Figure B.13: How completely were you able to actively survey or search the environment using vision?

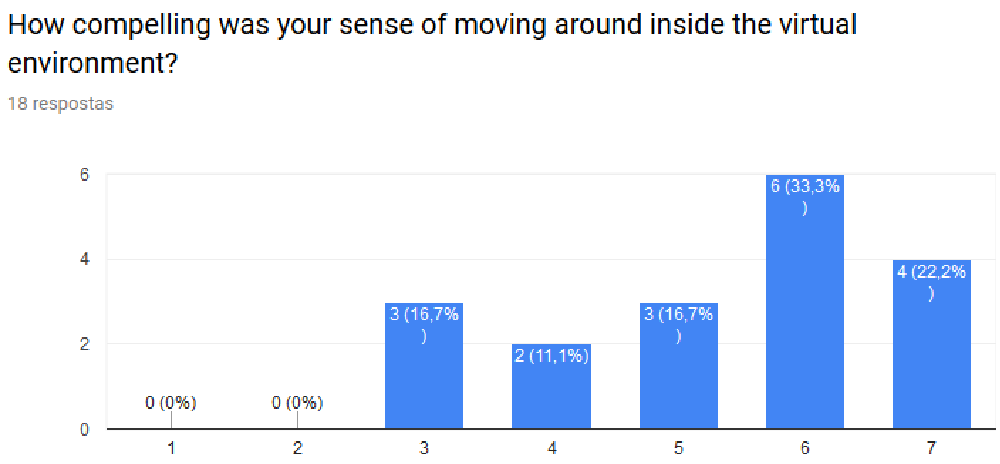


Figure B.14: How compelling was your sense of moving around inside the virtual environment?

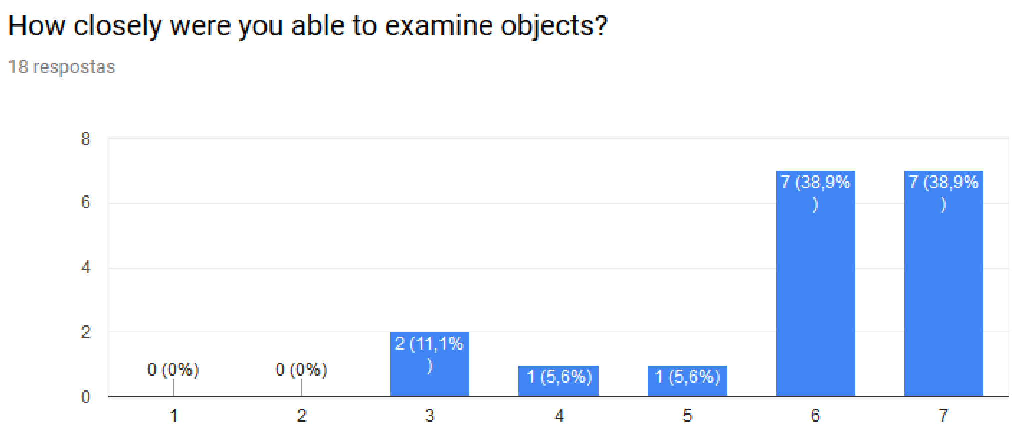


Figure B.15: How closely were you able to examine objects?

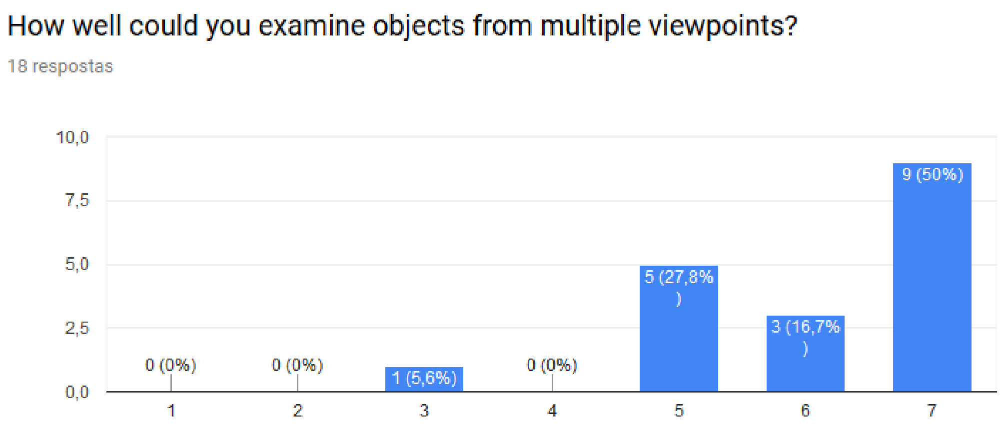


Figure B.16: How well could you examine objects from multiple viewpoints?

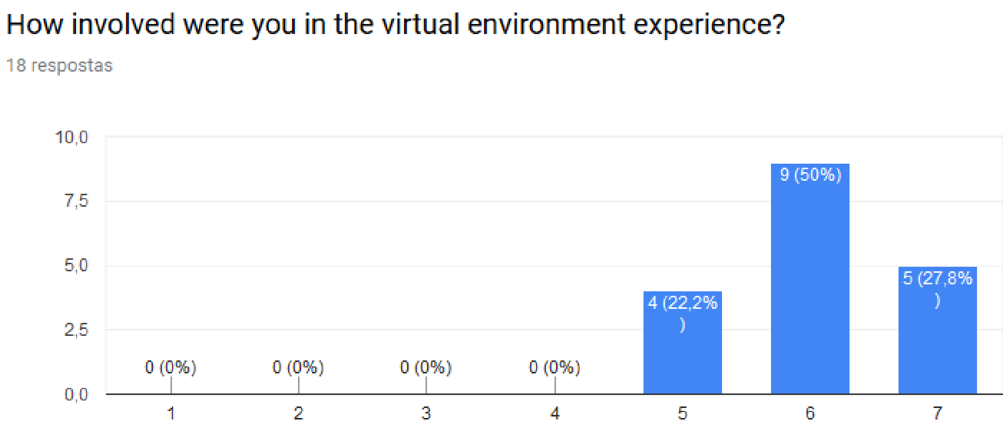


Figure B.17: How involved were you in the virtual environment experience?

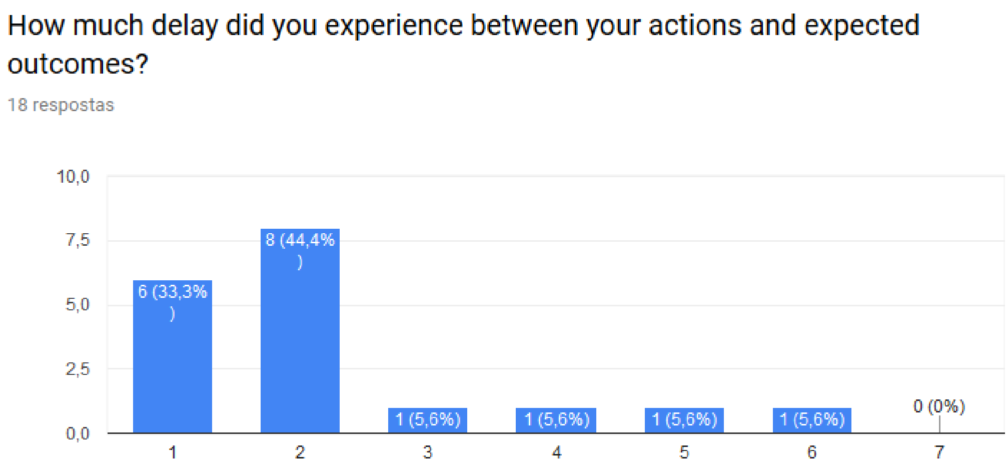


Figure B.18: How much delay did you experience between your actions and expected outcomes?

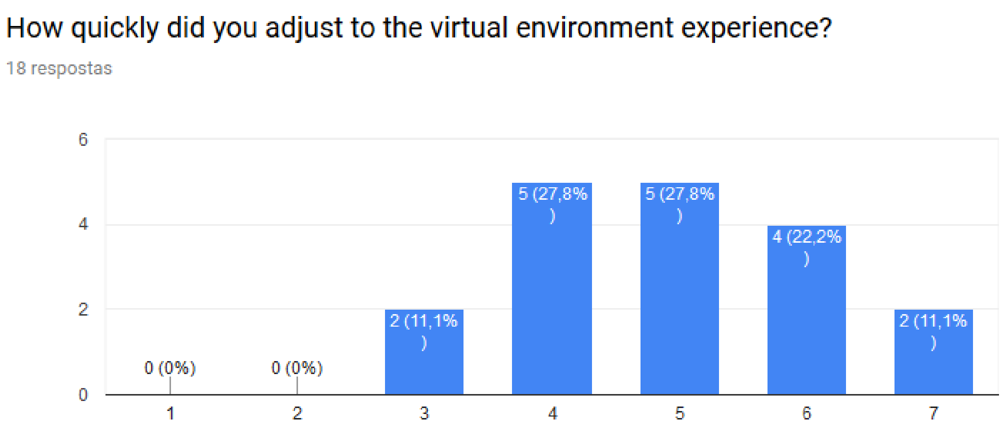


Figure B.19: How quickly did you adjust to the virtual environment experience?

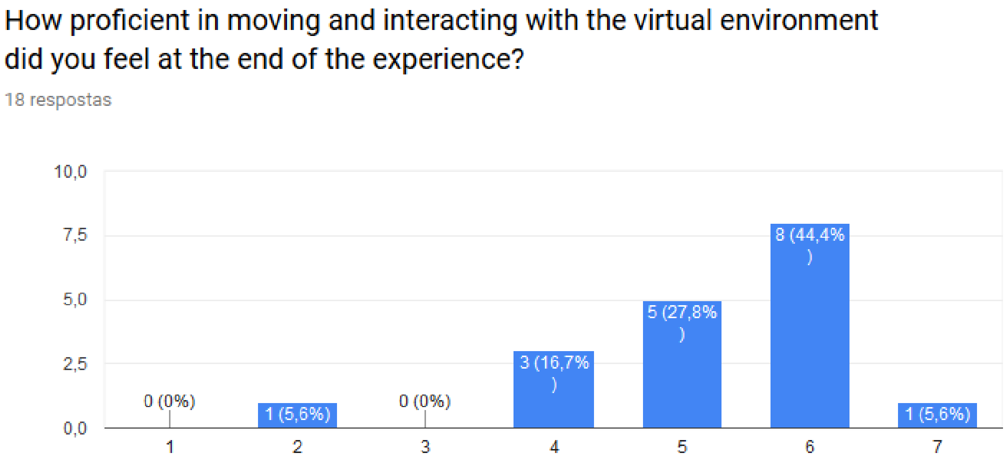


Figure B.20: How proficient in moving and interacting with the virtual environment did you feel at the end of the experience?

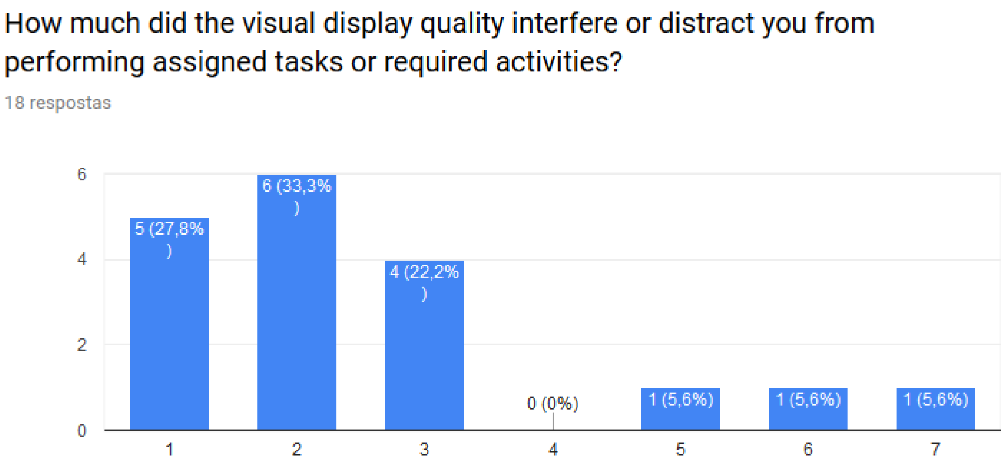


Figure B.21: How much did the visual display quality interfere or distract you from performing assigned tasks or required activities?

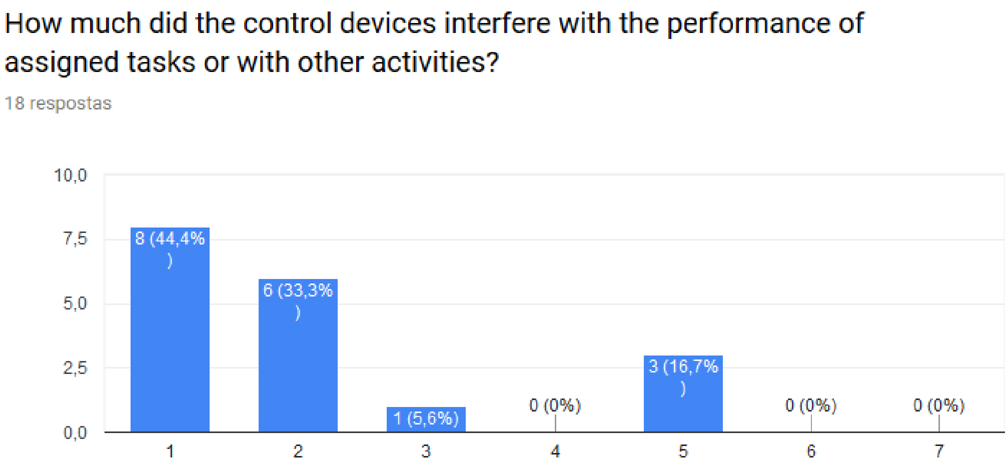


Figure B.22: How much did the control devices interfere with the performance of assigned tasks or with other activities?

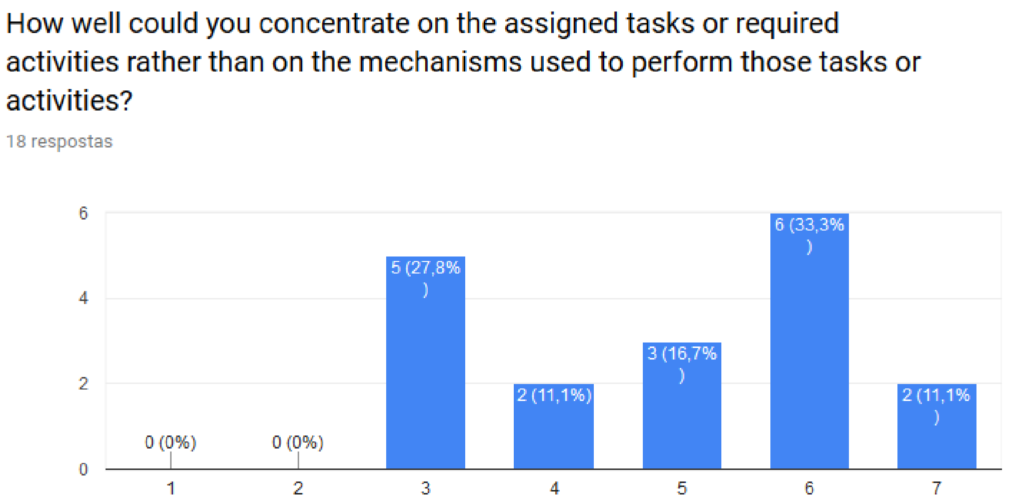


Figure B.23: How well could you concentrate on the assigned tasks or required activities rather than on the mechanisms used to perform those tasks or activities?

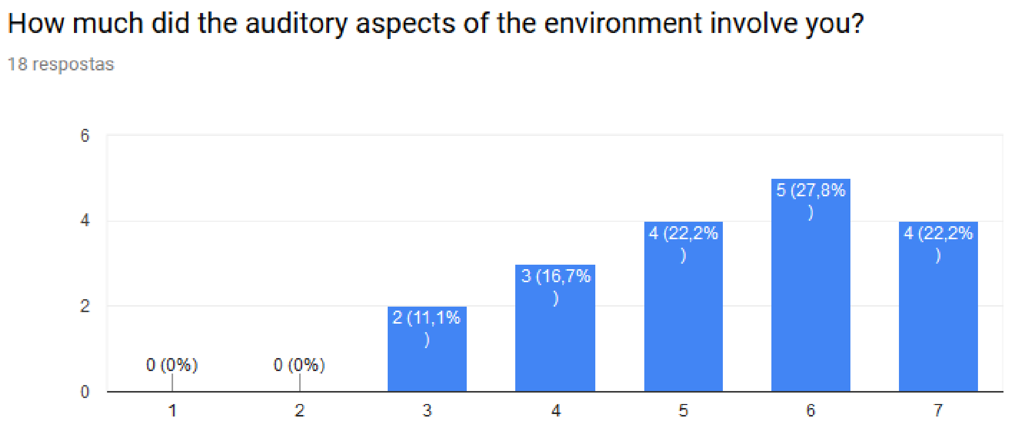


Figure B.24: How much did the auditory aspects of the environment involve you?

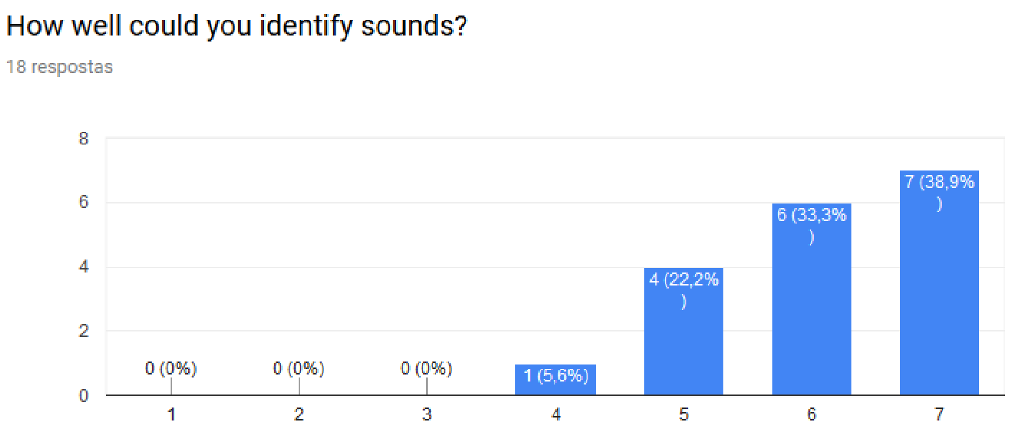


Figure B.25: How well could you identify sounds?

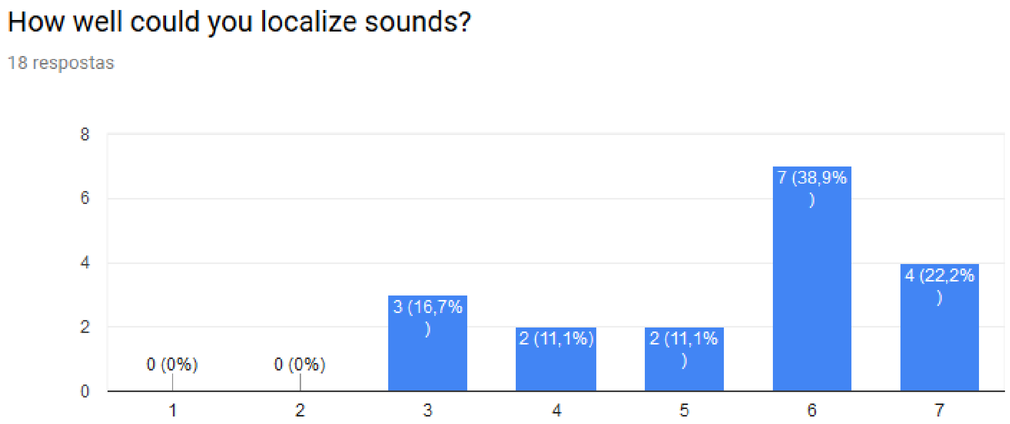


Figure B.26: How well could you localize sounds?

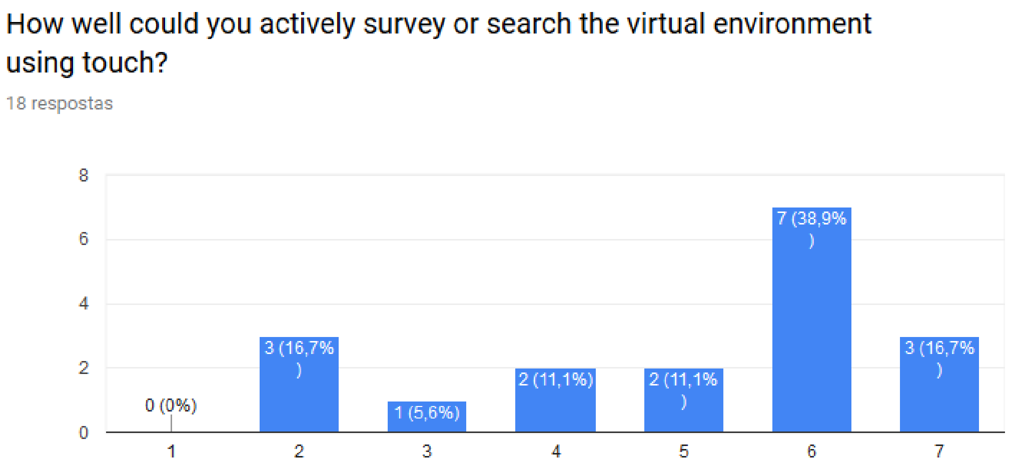


Figure B.27: How well could you actively survey or search the virtual environment using touch?

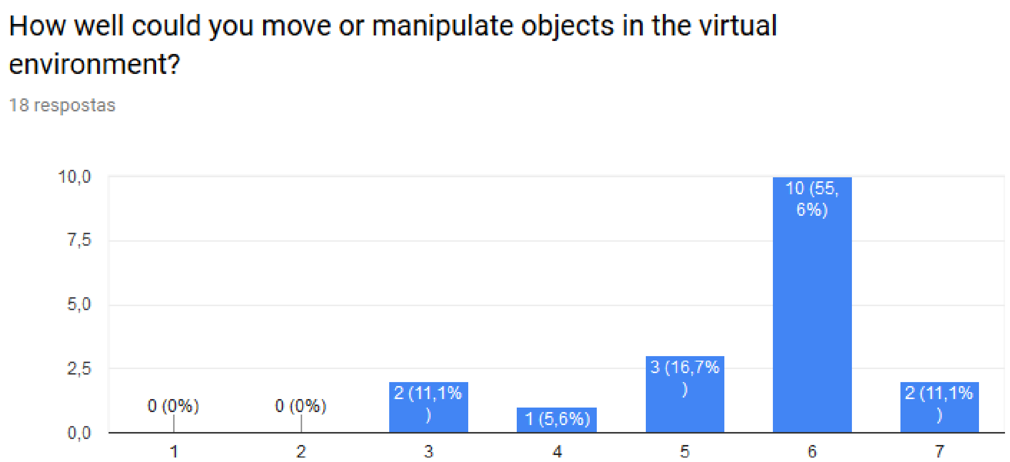


Figure B.28: How well could you move or manipulate objects in the virtual environment?

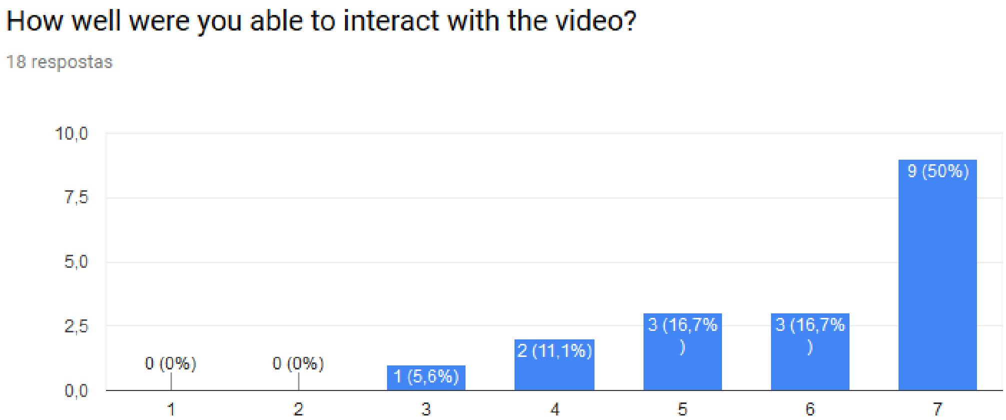


Figure B.29: How well were you able to interact with the video?

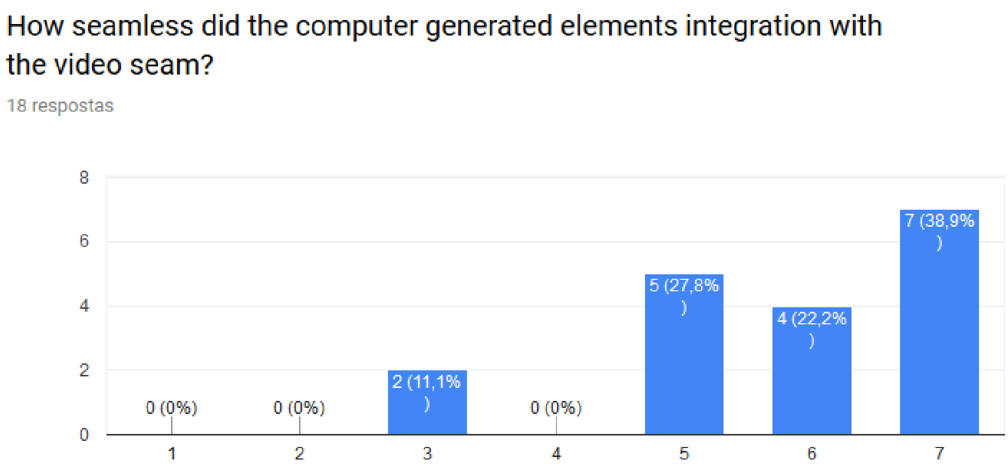


Figure B.30: How seamless did the computer generated elements integration with the video seem?

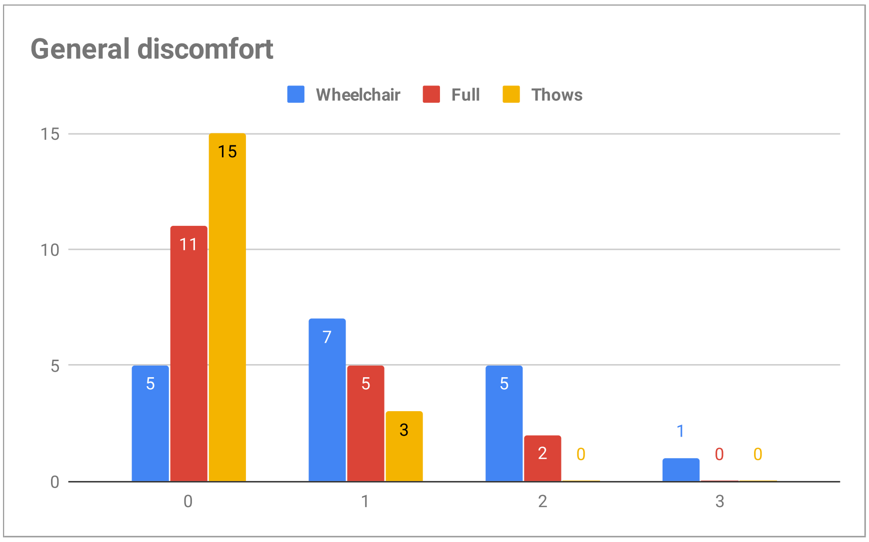


Figure B.31: General discomfort

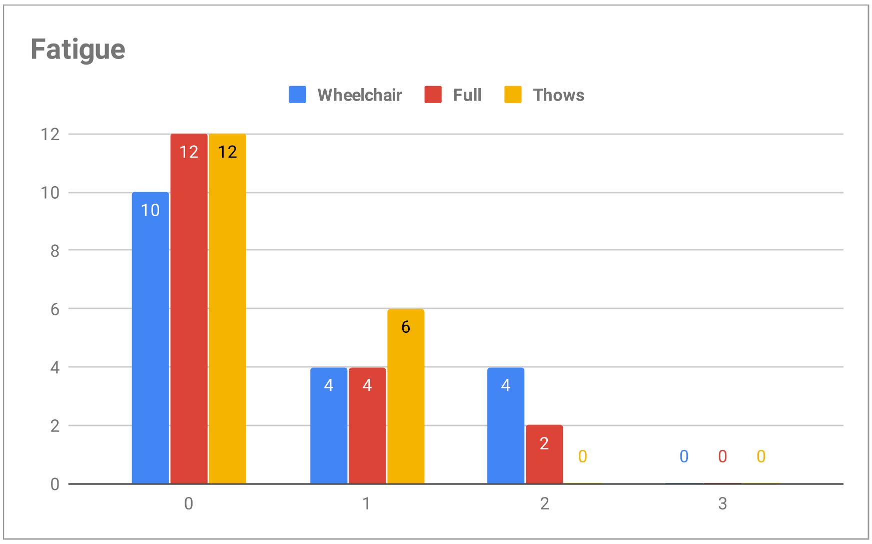


Figure B.32: Fatigue

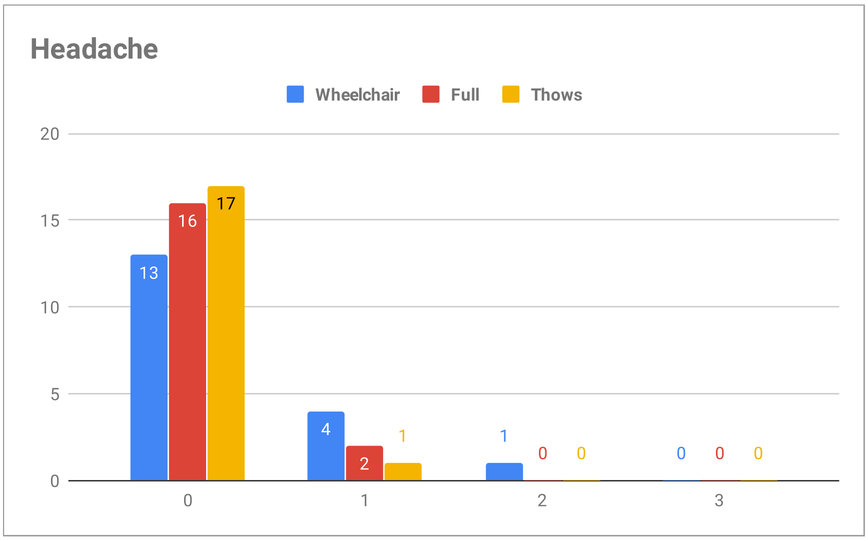


Figure B.33: Headache

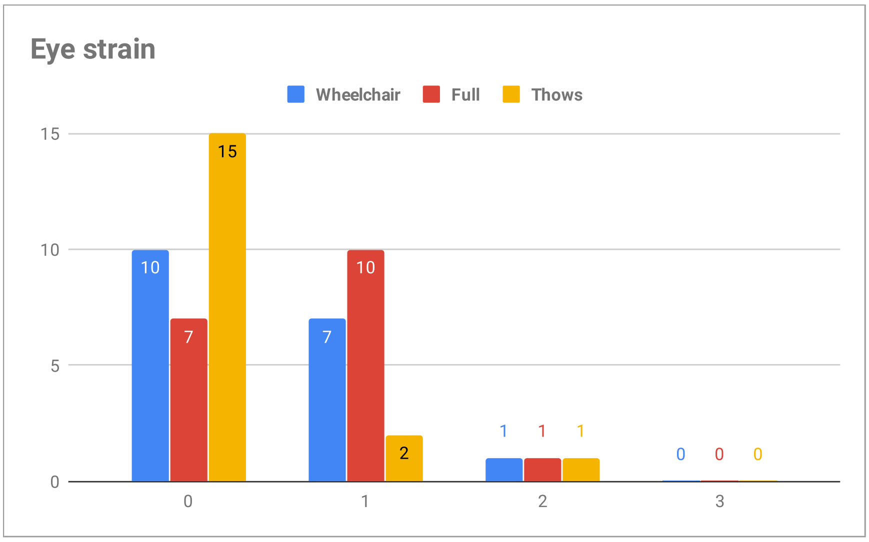


Figure B.34: Eye strain

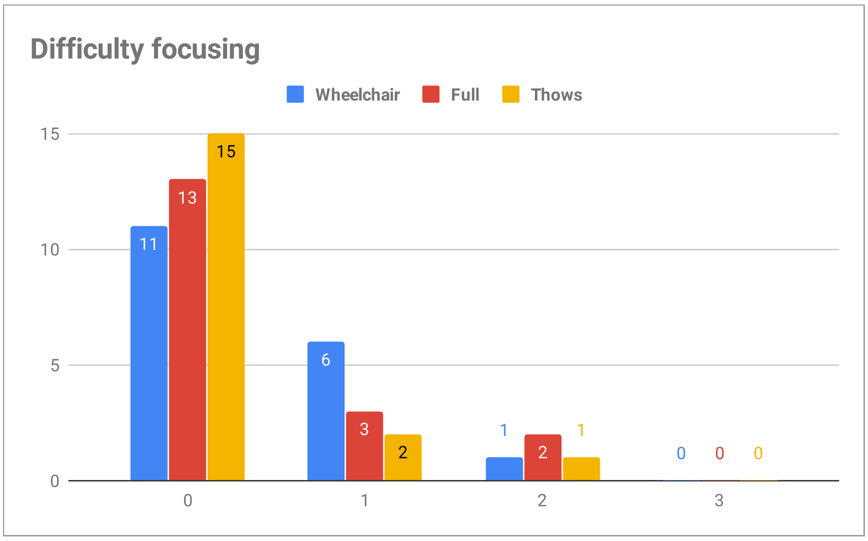


Figure B.35: Difficulty focusing

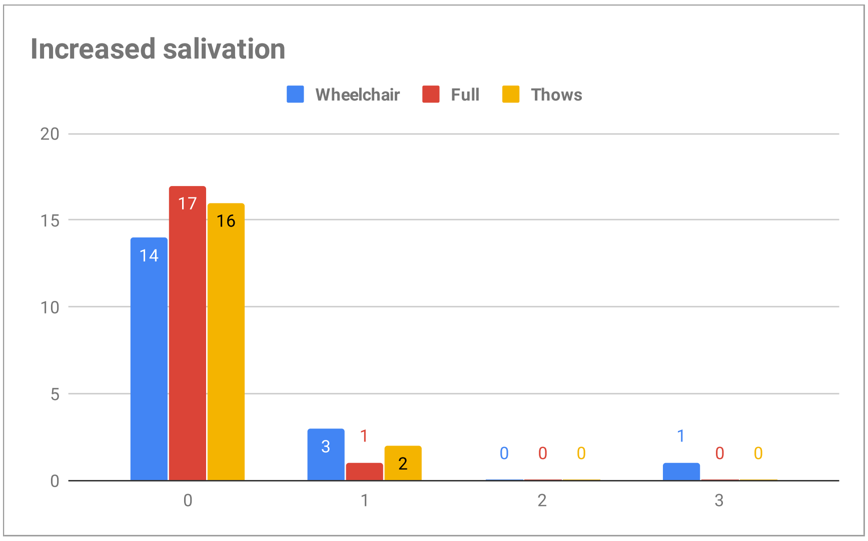


Figure B.36: Increased salivation

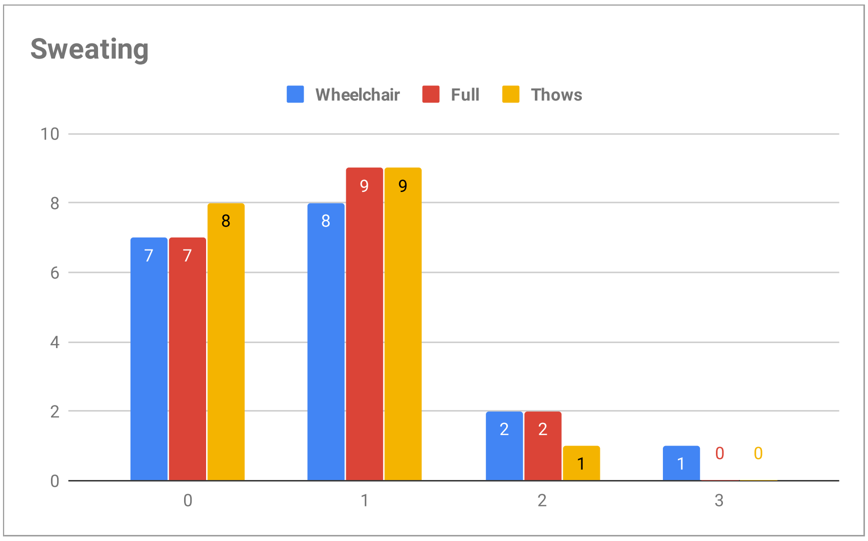


Figure B.37: Sweating

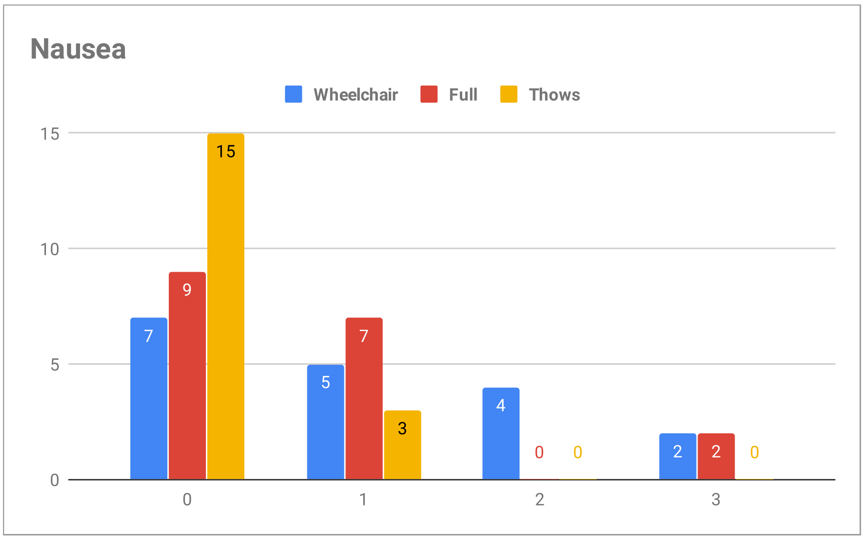


Figure B.38: Nausea

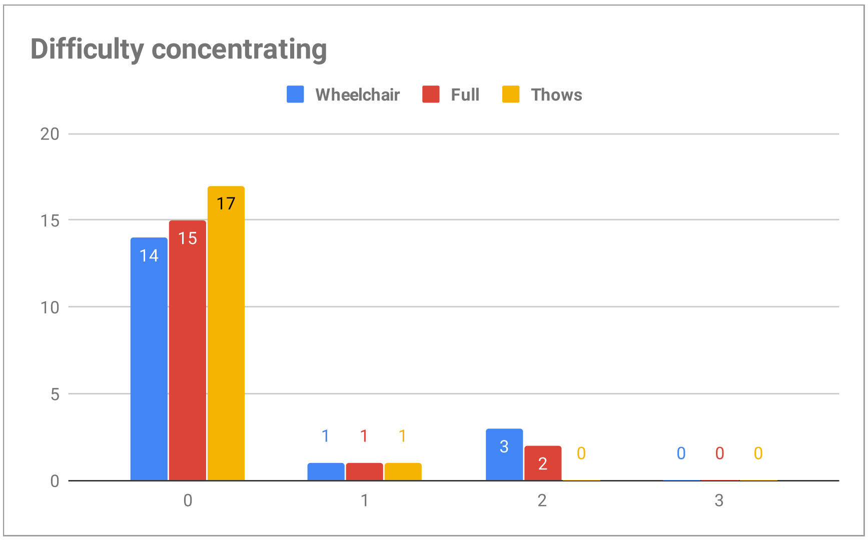


Figure B.39: Difficulty concentrating

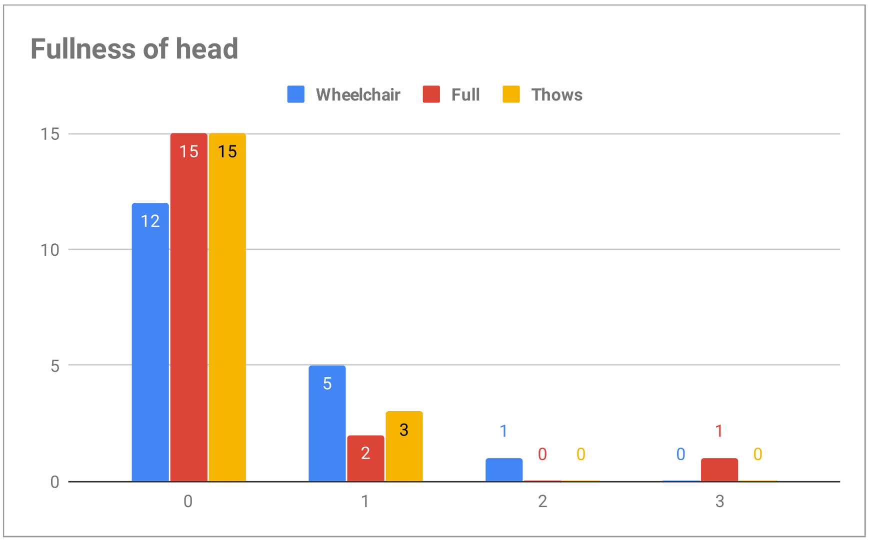


Figure B.40: Fullness of head

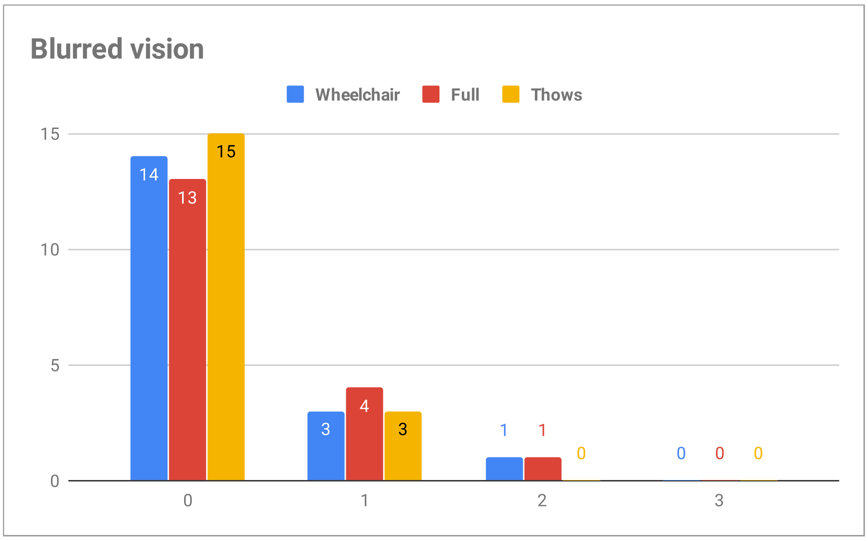


Figure B.41: Blurred vision

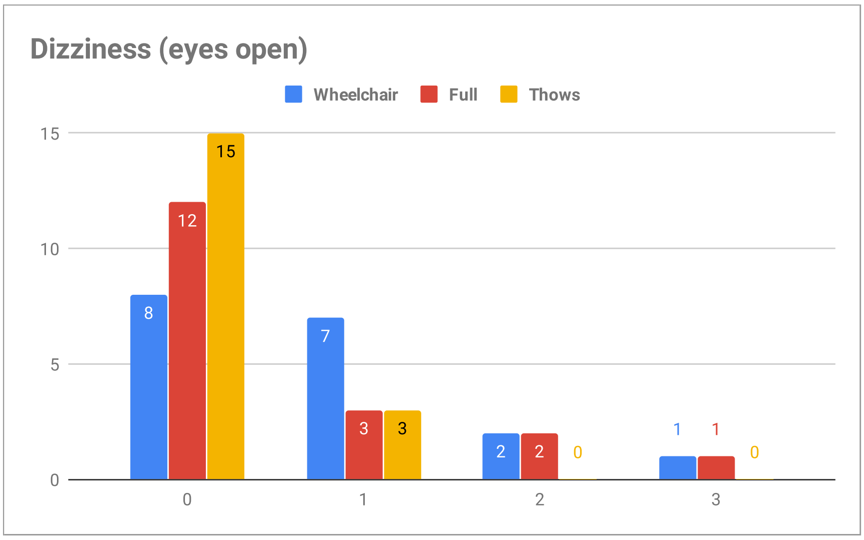


Figure B.42: Dizziness (eyes open)

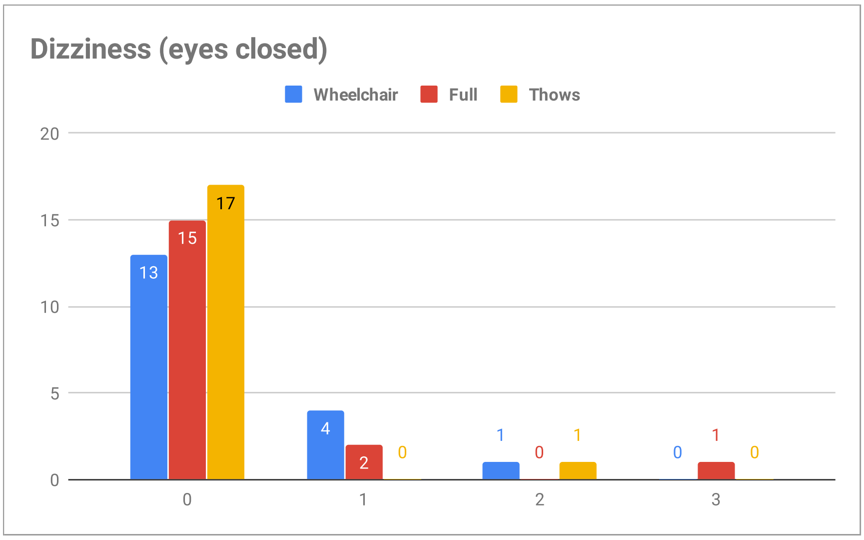


Figure B.43: Dizziness (eyes closed)

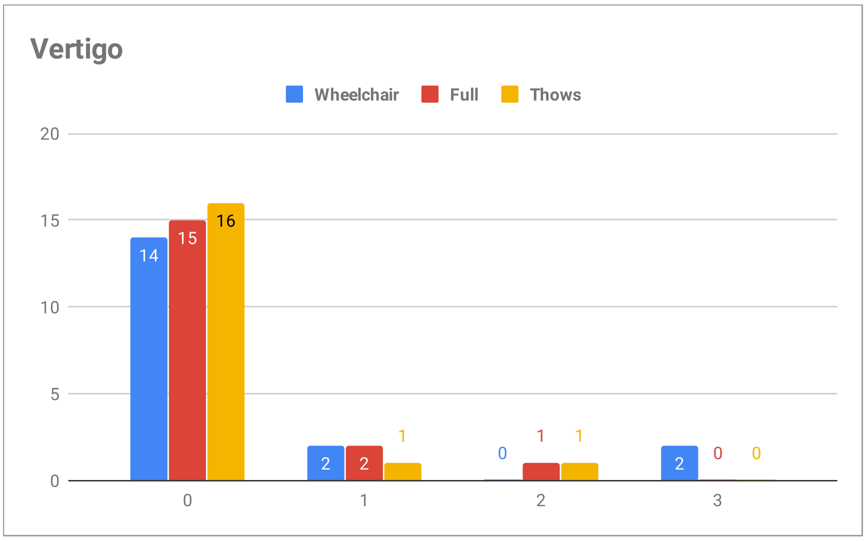


Figure B.44: Vertigo

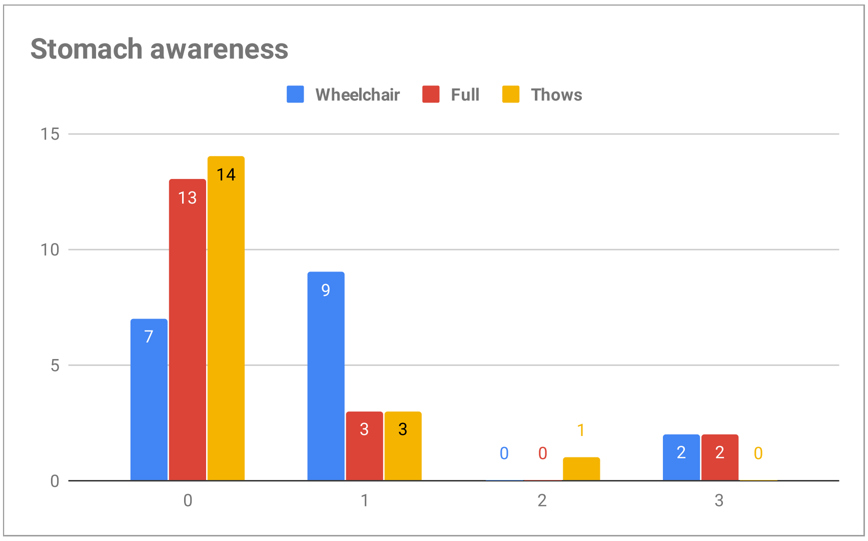


Figure B.45: Stomach awareness

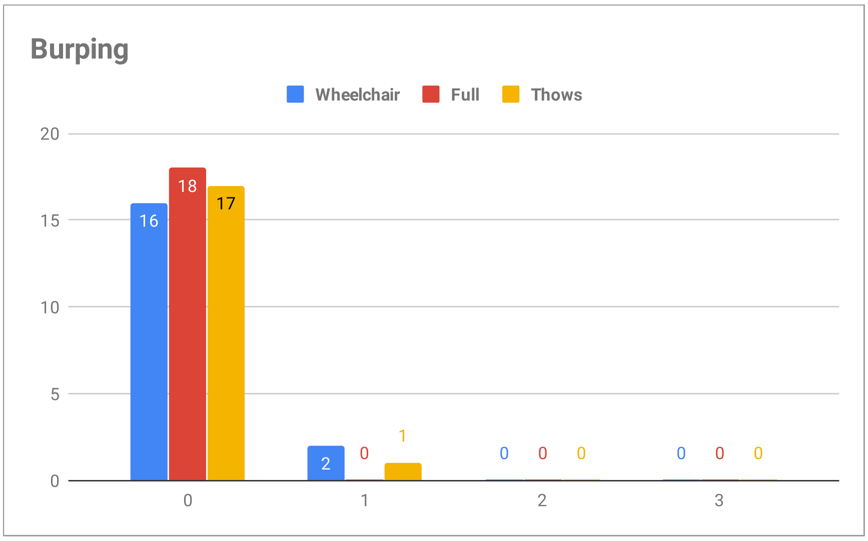


Figure B.46: Burping