POLITECNICO DI TORINO

Department of Control and Computer Engineering
Master Degree in Cinema and Media Engineering

Master Thesis

Testing Virtual Reality Interaction Techniques and Metaphors: The VR InteracTest Approach

Supervisors
Prof. Andrea Bottino
Prof. Francesco Strada

Candidate
Anna Sansoè

Academic Year 2022-2023

A tutti quei
"Potrebbe, ma non si applica"

Abstract

Virtual Reality (VR) has introduced a new dimension of user interaction, with hand tracking emerging as an intuitive input method. This thesis investigate hand interactions in VR, presenting a Unity application designed to implement various tasks: object grabbing, keyboard typing, and object resizing. Through a comparative study, we analyze user interactions in terms of efficiency, user experience, and adaptability between hand tracking and controller-based interactions.

VR InteracTest is an experimental platform for testing different interaction methods and metaphors. The application consist of three meticulously designed scenes, which simulate real-world tasks, and offer users an opportunity to interact with each scene. The available methods of interaction are controllers and hand tracking, which can be combined with ray-casting or direct interaction as metaphor. Overall, VR InteracTest enables a comprehensive analysis of user behaviour across the two input modalities. Furthermore, the application includes data collection mechanisms to record user interactions, capturing metrics such as task completion time, accuracy, and user preferences. A focus group, comprising both university students and non-expert of VR, took part in the study, providing the broader insights.

In addition to its comprehensive analysis of interactions in VR, this Unity application aims to provide unique contribution to the field. In the expanding world of virtual reality, designers and developers continuously innovate with new interaction metaphors. However, the ability to objectively assess these new metaphors in comparison to established ones was missing. Therefore, this application bridges that gap by allowing researchers, designers, and developers to rigorously test and compare new interaction metaphors alongside already established ones. It provides a controlled environment where the metrics defining interaction metaphor and metrics can be objectively measured and compared. This level of objectivity is crucial in an industry where innovation often outpaces the ability to systematically evaluate and compare interaction approaches. By enabling structured evaluations of interaction metaphors, this application empowers VR creators with valuable insights into the effectiveness of their designs. It facilitates evidence-based decision-making in the development of VR applications, finally leading to more user-centric and efficient virtual experiences. Thus, VR InteracTest is not merely a research endeavour, but also a practical tool that addresses a pressing need in the VR community.

In conclusion, this thesis contributes to the evolving field of VR interaction by providing an in-depth exploration of hand-based interactions through a Unity application. The study outcomes underscore the importance of context-specific interaction metaphors and the necessity of considering user preferences and task requirements when designing VR applications. The collected data from user tests are a valuable resource for designers, researchers, and developers striving to create more effective and user-friendly VR experiences. As the VR landscape continues to evolve, the insights derived from this study guides the interaction paradigms refinement and enhance user engagement in virtual environments.

Summary

Virtual Reality (VR) has witnessed rapid advancements, providing new possibilities and challenges for interaction design. As VR applications become increasingly prevalent, understanding the effectiveness of different interaction systems becomes crucial. This thesis delves into Virtual Reality interaction systems, primarily focusing on developing and evaluating VR InteracTest—an application designed to aid developers and designers in refining interaction metaphors for VR environments.

This thesis addresses this need to objectively test the new interaction systems by developing and evaluating VR InteracTest. The research explores how users interact with VR environments using **controllers** and **bare hands**, aiming to optimise user experiences. The study involves twenty participants engaging with different interaction systems to evaluate their performance and determine user preferences. This extensive exploration provides insights into the efficiency of various VR interaction systems, focusing on user experiences and perceptions but mostly on evaluating from an objective point of view their performances; and collecting data using specific evaluation metrics.

The tasks to be performed with the various interaction methods were initially selected. This choice was made with an in-depth analysis of the existing state of the art. After identifying the tasks most commonly used in the literature, the initial tasks for VR InteracTest were selected. In particular, these are **Grabbing** objects, **Typing** on a keyboard and **Manipulating** objects. To make my study participants more comfortable, I decided to structure the application with different rooms. All the rooms were artificially furnished with 3D models, to ensure that people were pleasant within the virtual world surrounding them during the test. Furthermore, every test scene is preceded by a Tutorial Room. This choice was intended to ensure that users could understand the functioning of the different interaction systems.

The methodology involves a two-fold approach: objective performance metrics and subjective user evaluations. Twenty participants engaged in tasks using different interaction approaches. Specifically, the selected interaction systems are: Controller-Raycasting, Controller-Direct, Hand-Raycasting, and Hand-Direct. This interaction system was selected from a wide range of possibilities; during the state-of-the-art work, I had the chance to select the most common ones for the device I used to conduct the tests, the Oculus Quest 2. Performance metrics included task completion rates, errors, and task duration. Post-experience questionnaires and the System Usability Scale (SUS) were employed for subjective evaluations. Also, these two choices were driven by state-of-the-art research.

Since testing all these iteration methods would have taken too much of the participants' time, I decided to have each person test only two different interaction systems. The criteria with which I selected the interaction methods for each user are as follows to make it clear what the difference was between the input devices, I made sure that each person tried both the controllers and the hand tracking. Furthermore, to make people understand their personal feelings about the interaction metaphors, I ensured they tried the same metaphor with both controllers and bare hands. Therefore, the number of users that tried "Controller-Raycasting" was twenty because it was impossible to achieve the system "Controller-Direct" for the typing task, so those who had to try the type task with controllers and direct used controller Raycasting instead. On the other hand, the "Hand-Raycasting" interaction system was tested by ten participants, as for the interaction systems "Controller-Direct" and "Hand-Direct".

The results identify that the "Controller-Raycasting" system exhibited superior task completion rates and fewer errors, highlighting its efficiency. The outcomes of the subjective metrics are sustained by those extracted in the objective evaluation of the user performance. This particular system reached the highest score for the type task and the manipulation task in terms of completion time grabbed object for the first task, and written words for the second task. "Hand-Raycasting" demonstrated remarkable performance, while "Controller-Direct" and "Hand-Direct" systems presented more controversial user experiences. In fact, "Controller-Direct" scored the highest for the grab task in terms of completion time and total grabbed objects, while the Manipulation task had the highest completion time and lower number of scaled objects. Participants expressed a preference for controllers over bare hands, emphasising the perceived advantages of handheld devices. **70%** of the participants preferred controllers as interaction devices; this data is shown in Fig. 6.17. Interestingly, "Direct Interaction" was favoured by **55%** of participants, highlighting the importance of natural and straightforward interaction metaphors, as shown in Fig. 6.18.

Going into the details of the post-experience questionnaire outcomes, the "Controller-Raycasting" system consistently received favourable ratings for ease of use, suggesting its already recognised user-friendliness. The average result of SUS for "Controller-Raycasting" was a score of **73.62**. Indeed, for all the other methods, the scores were lower. Mainly, for the "Controller-Direct" system, the average result of all ten evaluations was **62.5**; for the "Hand-Raycasting" system, the average result of all ten evaluations was **57.75** and finally, for the "Hand-Direct" system, the average result of all ten evaluations was **63.75**. Summing up, "Hand-Raycasting" received positive feedback, even if some users found it less intuitive. "Hand-Direct" and "Controller-Direct" elicited mixed responses. More specifically, "Controller-Raycasting" secured the highest SUS score, indicating favourable perceptions of usability, as already anticipated previously. "Controller-Direct" demonstrated respectable usability, while "Hand-Raycasting" and "Hand-Direct" received lower scores, indicating space for improvement.

In conclusion, this research provides new insights into VR interaction systems, emphasising the need for adaptive approaches to accommodate diverse user preferences. Users' preference for the "Controller-Raycasting" interaction system underscores the significance

of handheld devices and haptic feedback. The study proposed in this thesis contributes to the ongoing discourse on optimising VR experiences, supporting user-centric designs and adaptive interaction methods. The findings offer practical guidance for future VR developments, promoting a comprehensive understanding of user needs in immersive technologies.

VR InteracTest will be available to the public. The choice to make it downloadable from GitHub and usable by most will allow for an increasingly expansive platform in the future. Adding more tasks other metaphors, and implementing new input devices will be possible. Through VR InteracTest, everyone will have the opportunity to compare their new interaction system with those already existing, to put an objectively tested interaction system on the market.

Index

List of figures

List of tables

Chapter 1

Introduction

1.1 What is Virtual Reality?

Virtual reality allows us to simulate a reality different from the one we know, a fictional reality. As early as 1962, there was talk of virtual reality with Morton Heilig's "Cinema of the Future", which engaged the viewer's senses realistically. The so-called SENSORAMA is shown in the following image, Fig. 1.1. It was a passive simulator of a motorcycle that presented real images to the user through a stereoscopic visor.

Fig. 1.1: The Sensorama [1]

Computer scientist Jaron Lanier coined the term 'Virtual Reality' (VR) in the 1980s. Lanier founded VPL Research, one of the first companies dedicated to VR development. Since then, Virtual Reality quickly became a tool with vast potential across various fields of knowledge, attracting an ever-growing number of users, primarily drawn to what could be perceived as an alternative lifestyle within an environment that provides ample room for creativity and interactivity, forging pathways into new worlds. For Jaron Lanier, the so-called VR is a tool capable of promoting new forms of shared creativity and new ways of relating. After that, in 1995, Nintendo introduced the Virtual Boy, a portable VR gaming console. It featured a stereoscopic 3D display, but its monochromatic graphics and discomfort limited its success. The 1990s saw significant interest in VR, with companies like Sega, Atari, and Virtuality producing VR devices. However, high costs, limited content, and bulky hardware led to declining popularity. Afterwards, the advances in computing power, graphics, and motion tracking reignited interest in VR. Oculus Rift, a kick-starter-funded project, played a pivotal role in this revival and was acquired by Facebook in 2014. From 2016 the release of consumer-grade VR headsets like the Oculus Rift, HTC Vive, and PlayStation VR marked a new era for VR. These headsets offered high-quality visuals, immersive experiences, and a growing library of VR content. VR nowadays found applications beyond gaming, such as in education, training, healthcare and architecture. Medical professionals use VR for simulations and therapies [3], while various industries increasingly adopt VR training.

Today, it is possible to experience virtual reality through headsets called Head Mounted Displays (HMDs). The user, immersed in a lifelike world, can also interact with objects within it and modify them, move them, and use them for specific purposes. Various input devices have been developed to achieve this, such as controllers, gloves, suits with sensors, and hand tracking.

Fig. 1.2: Examples of HMD | Meta

Another existing technology is Augmented Reality (AR). AR allows the overlay of digital elements, such as images, videos, or information, onto our perception of the physical reality surrounding us. Thanks to specialised applications and devices, Augmented Reality can recognise objects or the surrounding environment and provide users with additional information or interactive content. In this way, AR enhances our perception and interaction with the world.

While Augmented Reality enriches our interaction with the physical world, Virtual Reality, on the other hand, transports us entirely into digital worlds. This distinction is

crucial because it shapes the way we engage with and manipulate the environments we encounter.

1.2 Unity: the history

Fig. 1.3: Unity's Logo

Unity, a leading game development platform, has played a pivotal role in shaping the landscape of virtual reality (VR) applications. Unity's journey has been marked by innovation and a commitment to providing developers with the tools they need to create immersive and interactive experiences.

Unity initially gained popularity as a versatile game development engine, allowing developers to build games for various platforms. On November 12 2013, version 4.3 was published. This first version facilitated the development of two-dimensional games and a notable toolkit for creating toolkit customised GUIs. On March 3 2015, Unity 5 was presented since its publication by some video game developers. On May 2 2018, version 2018.1 was published with essential updates to the graphics engine. The 2019 version, in addition to the countless changes made to the graphics engine, makes it more professional and in step with the times and new functions. Recognising the potential of VR to revolutionise the way users engage with digital content, Unity incorporated VR support into its framework, presenting in the specific Nintendo Labo kit.

This strategic move opened up new possibilities for developers interested in exploring the immersive world of Virtual Reality. Unity's VR capabilities provided a comprehensive set of tools and resources, enabling developers to create VR applications and experiences across different industries, from gaming to education, healthcare, and beyond. The platform's user-friendly interface and extensive documentation made it accessible to developers of all levels of expertise. Unity's Asset Store further enriched the development experience by offering a vast array of pre-built assets, scripts, and plugins tailored for VR applications. These improvements accelerated development processes and fostered a collaborative ecosystem where developers could share and enhance each other's work. The success of Unity in the VR space is evident in the multitude of VR applications and experiences that have been developed using the platform. From games that transport players

to fantastical realms to educational simulations that offer immersive learning experiences, Unity has empowered developers to bring their VR visions to life.

In conclusion, Unity's evolution into a robust VR development platform has been instrumental in shaping the VR landscape. By providing accessible tools and fostering a supportive community, Unity has empowered countless developers to explore the limitless possibilities of virtual reality, contributing to the proliferation of diverse and innovative VR applications.

1.3 VR InteracTest: motivations

Fig. 1.4: VR InteracTest's Logo

As we just said, with the introduction of more affordable devices, the library of VR, AR and MR(Mixed reality), and Unity's platform's growth, contents have grown a lot. However, this sudden growth has brought with it some shortcomings. One of the lacks of this world is that programmers, designers and people who develop content for MR need a way to objectively compare their new systems of interactions introduced on the market.

VR InteracTest is an application that helps in this sense. Indeed, it gives a virtual place where developers can test their new interaction methods, with the existing ones.

For this work, I used Oculus Quest and Oculus Quest 2. These particular HMDs permit interaction with Controllers (Oculus Touch) and Hand Tracking. Traditionally, controllers have been the primary technology of interaction in virtual reality. These handheld devices, often resembling game controllers, allow users to navigate, point, and interact with objects within the virtual space. They provide precision and familiarity, making them a staple in VR experiences. Hand tracking technology uses 'Inside-out' tracking, meaning that on the HMD, four cameras face different directions, placed on the corners of the helmet. This system allows one to precisely track the motion of the hands and fingers without needing physical controllers or the help of markers.

Both methods are valid and have advantages; this is where the concept of interaction methods becomes critical. The hand-tracking approach promises a more intuitive and immersive interaction experience by allowing users to use their hands as they would in the real world, but it is difficult to give them feedback, which is possible with controllers. The Oculus Touch devices can give users haptic feedback.

As a development platform, I used Unity 2022.3.12f1. This particular version of Unity offers the possibility to use the XR Interaction Toolkit. The XR Interaction Toolkit package, provided by Unity, is a component-based system for crafting VR and AR experiences. It introduces a framework that exposes 3D and UI interactions through Unity input events. At its core, the system comprises fundamental Interactor and Interactable components and an Interaction Manager that binds the connection between these component types. Additionally, it incorporates components designed for locomotion and visual representation.

This thesis attempts to immerse itself in Virtual Reality's interaction by conducting a comprehensive study and comparison of these two methods: controllers and hand tracking. I search to understand their strengths and weaknesses, assessing efficiency, user experience, and adaptability by gaining insights into how users interact with virtual environments using these technologies.

1.3.1 VR InteracTest: overview

First, I conducted a meticulous study of the state-of-the-art, which will be exposed in the next chapter. This study helped me establish what kind of tasks and interaction metaphors are now the most common. The structure of the application was decided according to the state-of-the-art outcome.

Specifically, the chosen tasks are distant selection, typing and manipulation. In the first scene "distant selection" task was successfully implemented by constructing a scene that allows users to interact with different objects at a certain distance from their current position. In the second scene, the user has to interact with a VR keyboard. For the last task, I chose to surround the user with four stations on which various objects to be manipulated are placed.

The chosen metaphors are Ray-casting and Direct Interaction.

After completing the development of VR InteracTest, I conducted tests on 20 subjects. These test sessions helped me understand what kind of interaction system between the possible ones is better from an objective and subjective point of view.

In conclusion, this thesis embarks on a journey to explore and compare different interaction methods in Virtual Reality, highlighting their capabilities, limitations, and user preferences.

Chapter 2

State of the art

In this chapter, I present the state of the art of interaction methods developed in recent years by various research groups. The goal is to understand which methods have been most commonly used in recent years and to reproduce and study the effectiveness of the latter. This will be done through subjective evaluation via user tests and objective analysis using specific metrics. To conduct this study, three different elements are required: interaction metaphors, tasks, and a list of various devices on which to implement them. To find out which ones were most present in the literature, I also studied some already available literature reviews and surveys, like [4], [5].

2.1 Interaction metaphors

Interaction metaphors are used in virtual worlds to enable users to understand how to interact with the elements within them. A good interaction metaphor should be representative of the task, compatible with the user's knowledge, and in line with the physical constraints of the interface being used. The choice of interaction metaphor is a fundamental aspect of designing an XR (Extended Reality) system and significantly impacts its usability

Interaction metaphors exist not only for virtual reality but also divide into metaphors with 2 or 3 Degrees of Freedom (DOF) and those with 6 DOF. The former are controlled by devices like mice and joysticks, while the latter are handled by specific 6 DOF devices.

Interaction metaphors can be of various types, including metaphors for navigation, selection, and manipulation.

2.1.1 Metaphors for selection and manipulation

In a virtual world, to interact with an object, it is necessary to first select it and then manipulate or modify it. These two operations are often considered interconnected, but they can also be used individually. These metaphors need to be adapted to the input and output systems used in that specific virtual experience. There are numerous interaction metaphors for these types of tasks, and we will present some of them below.

Virtual hand

In this case, virtual hands are represented, and users can select and manipulate objects as if they were using their own hands. This type of metaphor is used in a lot of studies [6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25]. There are various ways to implement this type of interaction metaphor. One method is to use controllers to interact in the virtual world, and the representation can vary, showing only virtual hands, virtual hands holding the controllers, or neither being displayed. Another method is to use hands directly without controllers, tracked using hand tracking technology, made possible with HMDs (Head-Mounted Displays) equipped with external cameras or devices like Leap Motion. With controllers, it is possible to provide users with tactile feedback by making the controller vibrate upon collision. However, providing such feedback is more challenging with hand tracking technology.

Arm extension (Go-Go)

Arm extension is based on the Virtual Hand technique. In this case, the virtual arm's representation does not necessarily reflect the user's actual arm's dimensions. This approach overcomes the limitation of not being able to reach distant objects in the virtual world but introduces accuracy challenges. As the virtual arm extends, accuracy issues become more pronounced. Besides selection, this interaction metaphor can also be adapted for navigation. A recent study introduced a similar concept to the Go-Go is the article [14] with the FingerMapper. The FingerMapper is a new metaphor that operates as follows: in the real world, fingers are used, and in the virtual world, they control elongated arms. This solution is useful for making expansive movements in the virtual world without having to make extensive movements in the real world, allowing for minimal physical motion in the virtual space and enhancing safety. The Go-go metaphor is also used to compare it with other metaphor in [26], like Raycasting explained in the next paragraph.

Raycasting

Raycasting involves representing a virtual ray that emanates from the user and selects the nearest object it intersects with. This method is easy to implement, user-friendly, and efficient. However, it struggles to handle occlusion issues. This type of metaphor is the most used in VR. In fact, this type of system is studied in a lot of articles like: [27], [17], [28], [19], [29], [30], [31]. Is possible to achieve this metaphor is possible to use both controllers and virtual hands.

Speech

One method for selection is to use voice commands to select an object by uttering its name or a specific property. However, this metaphor faces challenges due to the wide variety of different voices and potential accent variations. This type of metaphors are explained in [32]. Is possible to achieve this type of metaphor in various ways.

Hand centred object manipulation extended ray-casting

The HOMER interaction metaphor utilises Raycasting for object selection. Once an object is selected, users can move closer to it to facilitate manipulation. Upon release, the view returns to its initial position, mimicking the virtual hand. This method is designed to simplify the manipulation of objects that are not in close proximity.

Scaled word grab

In this metaphor, objects are selected on the image plane, and after selection, the virtual world is scaled so that the object matches the size of the virtual hand.

World in miniature

As the name suggests, this metaphor represents the world in miniature. Objects are indirectly manipulated. Selection can occur through virtual hands, which are then mapped onto real-world objects.

2.1.2 Metaphors for navigation

Navigation comprises two components: movement and way-finding. Movement involves physically transitioning from one place to another, while way-finding is the cognitive or decision-making aspect of navigation. There are in literature some studies that had compared the accuracy of tracking movements with different HMDs [33].

Physical movement in a virtual environment can be implemented using various elements to which the virtual camera is attached. The possibilities for movement and navigation are limited by the system, as different systems offer varying degrees of freedom. There are immersive HMDs capable of tracking and translating the user's real-world movements into virtual movement (real walking). On the other hand, 3 DOF HMDs require different methods for indicating movement within the virtual world. Desktop VR systems rely on devices such as mice and joysticks for movement.

Real walking

Real walking involves physical movement, generally achieved with 6 DOF HMDs. Users move as if they were walking in the real world, which helps reduce motion sickness. However, this metaphor has limitations, as the user's virtual movement is constrained by the physical space they occupy. Additionally, it can be dangerous if the room where real walking is conducted is not free of obstacles, as users wearing HMDs may not see real-world impediments.

Walk in place

In this interaction metaphor for navigation, users walk in place to move within the virtual world. The direction of movement is determined by the current view direction. Changing the head's position changes the walking direction. This type of interaction can be achieved

by attaching rings to the user's belt, ankles, or shoes. However, it may result in unnatural movement and feedback for users and can be physically demanding over extended periods. It also lacks data on walking speed and gait changes. This issue can be addressed by using external trackers or analysing accelerometer data.

Arm swing

Arm Swing is similar to walk in place, but in this case, users use their arms to move. This method provides better feedback for users, as arm movement while walking is more natural.

Teleportation

With teleportation, users wearing an HMD aim a controller or pointing device toward their desired destination. By pressing or releasing a specific button or trigger on the controller, they are instantly teleported to that location.

Pointing

Using a hand tracker held in hand, users indicate the walking direction continuously. This technique allows the decoupling of view direction from movement direction. However, it can be cumbersome due to the weight of the device. There is also the possibility to use the hands for moving as explained in [34].

Grabbing the air

In this interaction metaphor, users move as if they were "pulling a rope" to navigate through space. This method requires a mechanism to detect the start and end of the gesture.

Map based travel

This is a target-based technique. Users select their destination on a 2D map, and movement can occur either instantly or be represented through an animation.

These interaction metaphors and navigation methods offer different ways for users to engage with virtual environments. Each has its advantages and limitations, and the choice of which to use depends on the specific application and user experience goals.

The various metaphors mentioned have been implemented across different studies using diverse methods of interaction, including controllers, hand-tracking, gloves, and more. Also the devices were various: Oculus Quest 1, Oculus Quest 2, HTC Vive, RGB Cameras, Leap Motions and more.

2.2 Tasks for Virtual reality

Virtual reality and Augmented Reality are part of what is generically referred to as Extended Reality (XR). For any virtual experience, it is essential to establish a purpose. Indeed, the scope of the application, for instance, gaming, exploration, and education, undoubtedly has a vital role in distinguishing a particular virtual reality experience from another one.

The existing and available literature on VR applications offers countless tasks that can be implemented differently depending on the devices involved. The following table 2.1 offers a general overview of several tasks implemented differently within several experimental studies and applications. Notice that in Table 2.1, the acronyms UMSR and BMSR refer to Unimanual Metaphor with Scaled Replica and Bimanual Metaphor with Scaled Replica, respectively.

Evaluation metrics of literature

This section of my thesis highlights the evaluation metrics employed in the existing literature to assess the efficacy and influence of novel interaction systems. It is essential to clarify that I am now specifying the evaluation and comparison methods among various interaction metaphors already used to evaluate and quantitatively analyse new interaction systems. The most relevant metrics to evaluate the interaction systems in the literature are listed below, which are provided with some valuable references to allow the reader to gather more related details.

- Computational efficiency: evaluating with a frame rate analysis [6]
- Visually quantify the grasping performance, analysing each finger position and how it fits the object mesh [7]
- Completion time: Time took for a participant to complete trial successfully [27], [14], [29]
- Error in the object collection and monitoring (selection in movement) [27]
- Physical Motion: distance moved by the user's head in each trial [27]
- Spatial accuracy: position error and orientations error [29]
- Numbers of interactions [29]
- Movement time [35]
- The numbers of attempts [35]
- Numbers of collisions [35]
- Number of grabbed cubes, number of dropped cubes, number of stacked cubes, number of unstacked cubes, etc. [36]

To choose the metrics for my study, I took inspiration from these types of metrics. In particular, the ones that I selected because of relevant importance for my application are the number of interactions, the number of grabbed objects, the error in object collections and the completion time. The following parts of this thesis will exhaustively analyse the selected performance metrics and discuss their selection.

Task	Gesture
Selection	Pinch gesture [12]
	Retrieve virtual objects through either gesture or menu [13]
	Hit a moving disk to select an object and move it [27]
	Target selection, with attach and direct interaction [14]
	Pressing a trigger button, which displays details regarding an art piece
	Holding the trigger button and dragging it with the controller. [30]
	Reaching a target object with the rubber slider metaphor [25]
Manipulation	Scale the shape of a rectangle to fit in a establish shape [15]
	Match the vertex colours as quickly or precisely as possible with virtual hands [29]
	Transform an object to resemble a goal object in position, orientation, and scale [37]
	Manipulate a cube into a wire frame target to match the vertex colour quickly [35]
	Manipulate a cube minimising distance and collisions with the tunnel walls [35]
	Manipulate objects for an Intubation training. BMSR & UMSR [23]
	Manipulation of remote objects in AR environment with different system [31]
	Manipulate objects with the Baloon metaphor [26]
Digitation	Digit some number from a keyboard with all different interaction mode [8]
	Digit on a virtual keyboard or piano, on a table for having haptic feedback [11]
	Digit on keyboard to answer quiz's questions, or select from multiple choices [12]
Lego assembly	Pinch gesture and release
Grabbing	Grasping an object using fingers and palm
	Grasping an axe to hit a tower of cubes that subsequently collapses [6]
	Grabbing object like in real life [7]
	Grabbing balls from a basket with different types of interactions [8]
	Pick four coins and place those in a chest [27]
	Pick up some balls and put them in a chest [14]
	Grab and move an object from the shared workspace to the personal workspace [28]
	Path moved thorough the grasping gesture point by moving or rotating the hand [21]
	PC assembly learning, chemistry learning, interacting with 3D cubic elements [22]
	Different grabbing scenario. For assembly tools[24]
	A cube into a cubical hole in a blue wall as quickly and as accurately as possible [35]
	Pick up and place task with three different level of complexity with virtual hands [36]
	Mid air interaction to grab objects [26]
	Target reaching [25]
Rotation	Rotate objects with the Handlebar metaphor [26]

Table 2.1: List of tasks and gesture

Chapter 3

VR InteracTest: interaction testing environment

This chapter introduces VR InteracTest, a new virtual reality (VR) app created to explore how people interact with different things in VR and how well those interactions match what users want. I'll explain the app's main goals, what we're trying to find out, how we're doing it, and what users can expect when they first set it up.

3.1 Environment overview

At the beginning of the application, users are presented with a scene that forms the start of their VR experience, in Fig. 3.1. This initial scene is crucial as it bridges the immersive world they are about to enter. In this scene, users can choose from various specific tasks, interaction methods, and interaction metaphors that will define their VR journey. One of the main decisions users must make is selecting a task. The available tasks are object grabbing, typing on a keyboard, and object manipulation. The decision to include these tasks was guided by a comprehensive review of current VR research, as outlined in the second chapter. In addition to task selection, users can choose the interaction method and interaction metaphor. The interaction method can be selected from VR controllers or hand tracking. The selection of the interaction metaphor is another pivotal decision. Users can choose between direct interaction and ray-casting, allowing them to customise their virtual interaction experience.

Considerable attention was paid to environmental design to ensure users feel comfortable and immersed in VR, that is really important as said in [38]. The initial scene is a carefully crafted room that replicates real-world elements. It is furnished with various elements, such as walls, windows, and a sofa, all designed to evoke a sense of familiarity and cosiness. This environment is engineered to provide users with a seamless transition from the physical world to the virtual one, thus enhancing the overall user experience.

In summary, this chapter offers an in-depth look at the fundamental elements of the VR InteracTest application. These factors, from the user's initial scene and task selection to

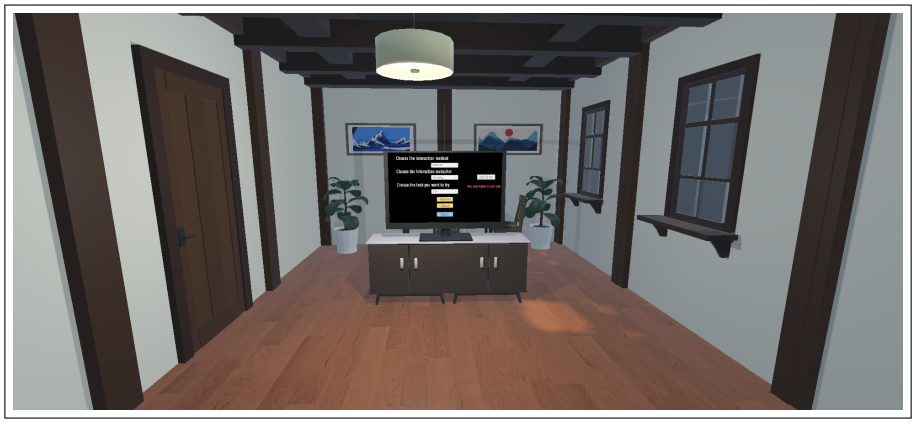


Fig. 3.1: Start Screen of VR InteracTest

interaction method and metaphor choices, create an immersive and tailored VR experience for participants.

3.2 Application scenes

An interactive tutorial precedes each scene within the experience. In these tutorials, a dialogue is presented on the screen to instruct the user on what to do. The tutorial's most valuable aspect is teaching the user about the interaction system they chose in the starting scene. For instance, if the user selects controllers as their method of interaction, the tutorial shows, with the help of an image, the correct button to press on the controller to achieve a specific goal. Alternatively, if the user chose bare hands as their method of interaction, the image demonstrates the correct hand gesture for interaction in the VR scene.

The user must achieve specific goals or objectives to progress through the tutorial. These tutorials are designed to familiarise users with the different interaction systems of the following test scenes.

3.2.1 The grab scene

The tutorial that precedes this test scene begins with only a cube placed on the table in the virtual room, in Fig. 3.2. The first goal of this tutorial for the user is to understand how to pick up the object with the particular system of interaction that he selected in the starting scene. On the table, there is also a TV screen displaying a phrase explaining what to do, an image that helps the user understand how to grab the object based on the chosen metaphor and interaction type. The second step of these tutorials is to grab the object and bring it to the bin, letting it fall inside. Every time the user reaches the goal, there is a sound feedback with a victory sound, and a message is displayed, saying, "Great job!". This stratagem helps the user feel involved in the game. If the user selects controllers, there is also haptic feedback. Another form of feedback is visual and audio

obtained through the "Interaction Affordance" developed by Unity. If you hover over the object with your hands, ray, or controller, it changes colour and produces a hover sound.

Fig. 3.2: Tutorial's Room - Grab task

For the grab task, I developed a scene where the user has to grab 25 different objects placed on a table, 3.3. The objects are trash that must be thrown into different bins for waste sorting.

Fig. 3.3: Grab task test scene

Like in every test scene, a timer is present, hidden on the wall above the door of the VR room. This timer has a clock sound, encouraging the user to complete the task quickly. However, it is strategically hidden from the user's direct view to avoid causing too much anxiety.

This element is also practical when the user encounters problems with the chosen interaction system, preventing them from indefinitely getting stuck in the scene. After

conducting preliminary tests with individuals who are no longer part of the final study, I set the timer for every test scene to three minutes based on their results.

As mentioned earlier, I provided feedback to the users using the "Interaction Affordance" developed by Unity. In these test scenes, whenever the user points with the ray of the ray-cast metaphor or reaches the object with the controllers or hands, the colour of the object they want to select becomes darker, accompanied by the "Hover sound". When the object is selected, it plays a "Pop sound". When the user reaches the bin with the corresponding trash element, there is a "Win sound," if an object falls, a "Fail sound" is triggered. In this particular case, a statement informs the user that the object has fallen. This specific statement appears only for two seconds. To prevent the user from wasting time, whenever an object falls on the floor, it reappears on the table in its original position when the test scene is loaded. In Figure 3.4 are shown all the 25 objects to grab and throw in the bins.

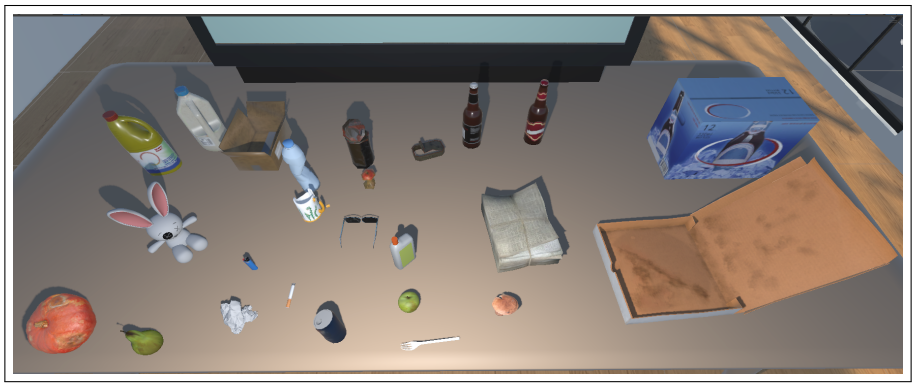


Fig. 3.4: Objects to grab - Grab task

Messages are displayed on the TV screen placed on the table in this room, to help users understand at which stage of the test they are. Different messages show the total number of objects collected, with the first sentence displaying the total since the beginning and the others showing the objects for each category of trash collected.

When the user successfully collects the twenty-five different objects or when the timer finishes, a menu will appear on the TV screen, as shown in Fig. 3.5, replacing the previous statement. This menu is structured similarly to the one present in the Start Scene. It features three drop-down menus. The first one allows the user to choose the type of interaction, either controllers or bare hands. The second one lets them select the interaction metaphor, either ray-casting or direct interaction. The last one allows them to choose the type of task. Combining these choices will load the next scene they want to try when the "Load Scene" button is pressed. Additionally, there is an option to go back to the Start scene. A button designed in the shape of a home is available. A "Back to menu" message will appear when the user hovers it.

Fig. 3.5: End Menu - Grab task

3.2.2 The type scene

The tutorial preceding this test scene starts with an Input field labelled “Enter text..”. The first step in this tutorial is to press on it, making the VR Keyboard Fig. 3.6 appear on the table in front of the user. Moving forward, the user has to type their name into the input field and submit the answer by pressing the “Enter” button. It’s possible to see the room’s aspect in Figure 3.7. Various visual feedback elements are present in this scene. When the user submits the correct answer, the sentence “Valid input” in green appears for two seconds. Conversely, if the user submits an incorrect answer, a sentence reading “Invalid input” in red appears for two seconds. These are the same feedback cues that will be present in the test scene. All these sentences are associated with specific sound feedback. Also, the keyboard has sound feedback on pressing buttons. There are different types of clips, depending on which button the user is pressing. After learning how to use the keyboard with the chosen interaction system, instructions on how to play in the test scene are provided, and then the user can enter the test scene.

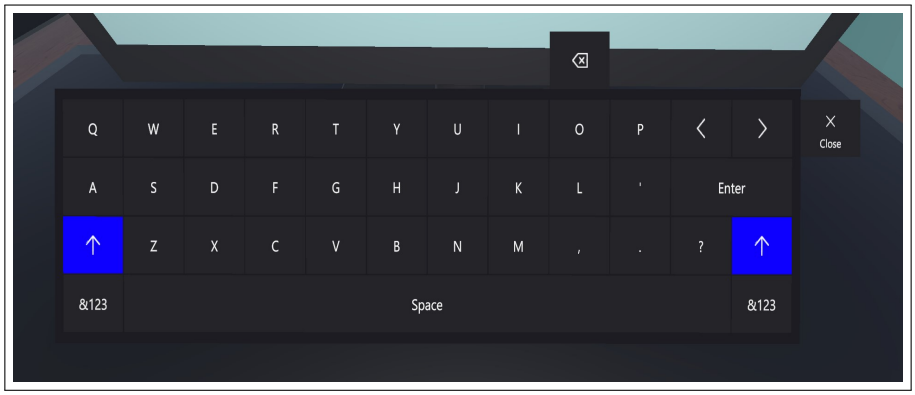


Fig. 3.6: MRTK-Keyboard, [2]

Fig. 3.7: Tutorial's Room - Type task

In the test scene for this task, the user has to write the word in capital letters.

- The capital of France is PARIS.
- In the solar system, there are EIGHT planets.
- In a week, there are SEVEN days.
- The colour you get by mixing blue and yellow is GREEN.
- The largest planet in our solar system is JUPITER.
- The Earth's natural satellite is called the MOON.
- In a year, there are TWELVE months.
- The opposite of right is LEFT.

Fig. 3.8: Type task test scene

Also, for this task, there is a three-minute timer. When the user completes all the requested words or when the timer ends, the End Menu, similar to the one described before, appears.

Fig. 3.9: End Menu - Type task

3.2.3 The manipulation scene

For the third task, I chose to focus on the Manipulation task. This task involves first selecting the object and then resizing it. The tutorial that precedes this test scene begins with only a pyramid placed on the table in the virtual room. The initial goal of this tutorial is for the user to understand how to pick up the object with the particular system of interaction selected in the starting scene. The objects must be grabbed with two hands for the manipulation task. The second step of these tutorials is to resize the pyramid and make it bigger. To achieve this, the user, after selecting it with both hands, needs to move the hands away. The last step of this tutorial is to resize the pyramid to make it smaller by moving the hands closer. Every time the user reaches the goal, there is a sound feedback with a victory sound, and a message is displayed, saying things like "Great job!". Also, for this last task, haptic feedback is provided when a user uses controllers.

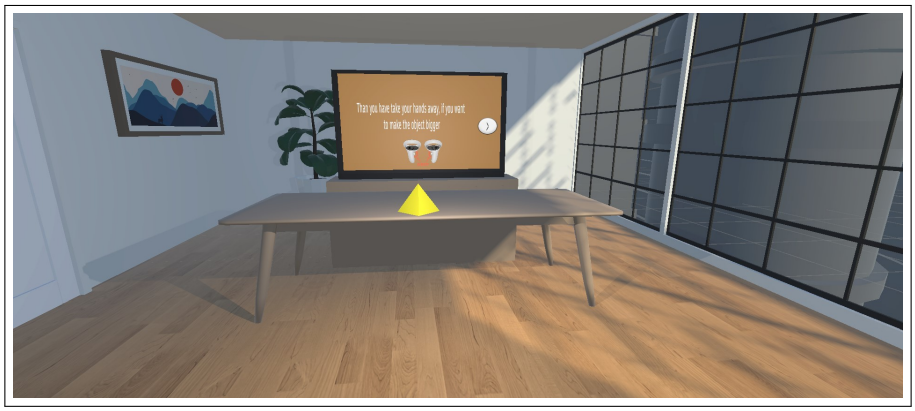


Fig. 3.10: Tutorial's Room - Manipulate task

The Test scene is loaded after completing the tutorial and pressing the "Start Scene" button.

In the test scene, the user sees four different tables placed all around. On every table is a different TV screen, each showing instructions for manipulating the specific objects on that table. The first table has two different cubes Fig. 3.12a. One is bigger and yellow, with the word "Reference" on it. The other one is smaller and red. For this station, the user has to make the red cube the same size as the reference one. When the user makes it bigger, the manipulable object becomes green, the instruction sentence disappears, and a "You did it!" message appears with a winning sound.

On another table, there are two different keys (Fig. 3.12c). One is small and yellow with "Reference" written on it. Near it is a second key coloured in grey. The user has to make the second key the same size as the reference one. The behaviour of the scene is the same; indeed, the object becomes green when resized, the instruction disappears, and the "You did it" message is displayed. An extra goal for this table is to put it in the keyhole to open a nearby door.

A hat is on the third table, Fig. 3.12b. The reference hat is yellow. The manipulable one is tiny and red. When the user resizes the manipulable hat, it becomes green, and

the user has an additional goal to place it on his virtual head.

As the last goal of this test, there are four different books on the last table, all grey, Fig. 3.12d. In this case, there is no reference because the objective is to make them small enough to fit in the drawers of a chest of drawers. When a book is small enough, it changes colour to green.

When all the "You did it" messages appear, a victory sound plays, and the End Menu appears on the screen, Fig. 3.11, just like in the previous scenes. In this case, the scene also includes the timer from the previous scenes, with identical functions.

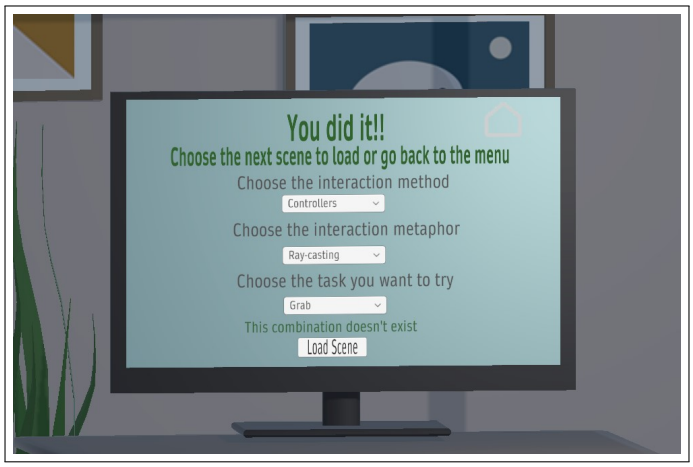
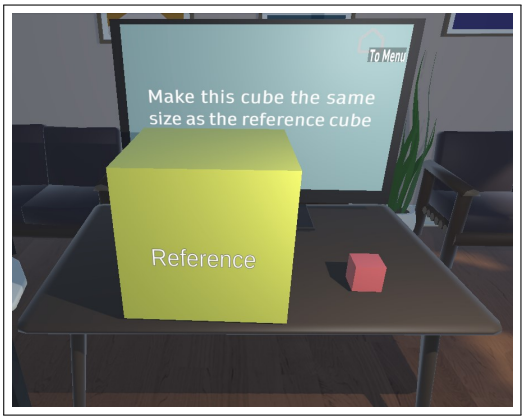
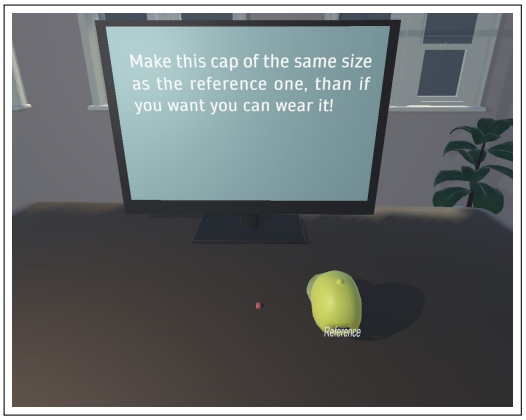


Fig. 3.11: End Menu - Manipulate task



(a) table with the cube



(b) table with the hat

(c) table with the key

(d) table with the books

Fig. 3.12: Manipulation task stations

3.3 Data collection

This thesis aims to build an application for VR developers. This app will collect data to objectively compare existing interaction systems and what developers intend to introduce in the market. I developed a method to record users' experience data to achieve this. During the literature review, I identified some metrics other people used for their studies.

To save all the data, I developed a script in C# language named CSVManager. This script can create a CSV file for each scene the user faces. In detail, the script first checks if a file already exists. If it does, it adds new lines to the existing table; if not, it creates a new file. The generated file has a specific name based on the scene that calls the CSVManager. It produces a file named "SceneOne.csv" for the grabbing object task. For the type task, it generates a file called "SceneTwo.csv". Lastly, the generated file is titled "SceneThree.csv" for the manipulation task. To write all this .csv, I used the "persistentDataPath", the standard method to access Oculus's internal memory. So once the user finishes one of the tasks in the VR InteracTest app, this system saves the data of that experience. In

cases where more than one player or the same player experiences a scene multiple times, the user index is incremented. It represents the third element saved in the third column of every CSV file. In the other columns of every specific file, other specific data will be written, which I will explain in the following subsection of this chapter.

3.3.1 Data saved for the scene with the grab task

For the task of Grabbing objects, I decided to save some specific data. In particular, like in all the other scenes, I save the type of interaction method as controllers or bare hands and the interaction metaphor ray-casting or direct interaction. Saving this data was simple because I developed eleven scenes, one for each interaction system for each combination of tasks, interaction method and interaction metaphor, excluding the combination of direct interaction, controllers and type tasks that cannot be achieved. So, in the script that controls the entire scene, a method called "BackToMenu" can save all data. Table 3.1 summarises the data extracted from the conducted test sessions, representing a subset of the data collected due to space constraints in this template. In addition to this represented data, there are Time, Object's Name and an Interaction Type column for each of the twenty-five objects in the scene. For the same reason, I shortened certain expressions; the acronyms are explained in the bulleted list following the table.

Scene	Method	Metaphor	#User	CO	FO	ST	ET	Time	ON	IT
1	Controllers	RC	1	25	1	21:06:44	21:07:50	21:06:48	Rabbit	SG
1	Hands	RC	1	25	1	21:07:56	21:09:02	21:07:59	Pizza	SG
1	Controllers	Direct	1	25	4	22:18:03	22:19:45	22:18:12	Beer	SG
1	Hands	Direct	1	24	6	22:27:00	22:30:00	22:18:12	Beer	SG

Table 3.1: Example of a SceneOne.csv file generated from the experience of one user

Abbreviated words:

- CO = Collected Objects,
- FO = Fallen Objects,
- ST = Start Time,
- ET = End Time,
- ON = Object's Name,
- IT = Interaction Type,
- RC = Ray-casting,
- SG = Start Grabbing,

As I explained, the actual tables are much more complete with data. The number of collected objects is helpful to determine whether or not the user finished collecting all the objects in the scene. The count of fallen objects during the performance is functional for objectively assessing whether the user makes more mistakes with a specific interaction system or another. Instead, the Start and End Times serve to understand how long the user takes to complete throwing all twenty-five objects in the scene or knowing if the timer expired. In this last scenario, the duration is three minutes. In addition to the previous representation, every object has another Interaction Type besides Start Grabbing and End Grabbing. These specific parameters (Start Grabbing and End Grabbing), along with the Time column, enable calculating the average time it takes for a user to target the correct bin after the initial object selection successfully.

3.3.2 Data saved for the scene with the type task

The second analysed task is the Typing task. As previously mentioned, this scene also saves the type of interaction method, whether it's between controllers or bare hands, and the interaction metaphor, either ray-casting or direct interaction. In the script that controls the entire scene, a method called "BackToMenu" can save all the data. In this case, I wanted to save different parameters from the previous one, but something is still in common. The standard parameters between the previous and this scene are the Start and End times. Table 3.2 summarises the data extracted from the conducted test sessions, representing a subset of the data collected due to space constraints in this template.

Scene	Method	Metaphor	User#	WA	CA	TCanc	TClick	StartTime	EndTime
2	Controllers	RC	1	0	8	5	61	19:04:28	19:06:02
2	Hands	Direct	1	0	7	3	54	19:16:08	19:17:38
2	Hands	RC	2	1	8	1	57	20:06:01	20:08:23

Table 3.2: Example of a SceneTwo.csv file generated from the experience of two users

Abbreviated words:

- WA = Wrong Answer,
- CA = Correct Answer,
- TCanc = Total Cancelled letters,
- TClick = Total clicked button,
- RC = Ray-casting,

As I declared in the previous paragraph, the Scene Number, the Method, the Metaphor, and the User Number are the same as the other scenes. In this specific case, I also note the number of wrong answers to understand the number of errors made by the user. The number of correct answers helps me understand if the user completed all the words I

requested in this scene. The number of Total Canc, instead, lets me know how precise the user's interaction system is in that particular try. In the end, the number of total clicks on the keyboard allows me to understand the correct number of letters I require and how many clicks it took the user to finish the scene.

The additional data in the actual table for this scene includes two repeated columns for each of the eight sentences. The first column represents the index of the sentence. The second column represents the timestamp with which the user starts responding. This last parameter helps me understand how long the user takes to write the specific word.

3.3.3 Data saved for the scene with the manipulation task

The saved data are structured similarly to other scenes for the scene involving the manipulation of objects. As always, we have the first column indicating the Scene Number, the second column indicating the method of interaction, the third column indicating the interaction's metaphor, and the fourth column indicating the user number. The functionality of these elements remains consistent, including the seventh and eighth columns representing the Start Time and End Time of the scene. The difference between these two elements provides the duration of the test.

Similarly, I collect specific data in this scenario as outlined in the summary Table 3.3 below. Abbreviated words:

Scene	Method	Metaphor	#User	ScaledObjects	FO	ST	ET
SceneThree.csv	Controllers	Raycasting	1	4	3	20:24:12	20:27:42
SceneThree.csv	Hands	Raycasting	1	4	7	20:39:00	20:41:31
SceneThree.csv	Controllers	Direct	2	4	5	22:25:00	22:27:05
SceneThree.csv	Hands	Direct	2	4	0	22:50:41	22:53:07

Table 3.3: Example of a SceneThree.csv file generated from the experience of two users

- FO = Fallen Objects.
- ST = Start Time.
- ET = End Time.

As explained earlier, the actual tables contain much more comprehensive data. The number of scaled objects helps determine whether the user successfully scaled all the objects in the scene. The count of fallen objects during the performance is functional for objectively assessing whether the user makes more mistakes with a specific interaction system. If the user cannot scale all the objects, the timer expires in three minutes.

In addition to the data represented in Table 3.3, I collect the Start Scaling and the End Scaling time for each object. This solution allows me to determine how much time a user needs to scale a single object.

3.3.4 Data Analysis

In this chapter, we will dive into how I analysed the data using MATLAB, building on what we discussed in the previous section about the data itself. As we move from exploring the data set to processing the numbers, this chapter will explain my steps and methods to make sense of the information. I chose MATLAB because it's a powerful tool that's good at handling the data we have. I will take you through the process, explaining each step and why I did it, all to find significant patterns and trends in the data.

As I explained in the previous chapter, the data that I extract from the performance of each user generates three different CSV files. The first thing I do in the MATLAB file is to read each file and generate four different tables for the possible fourth system of interaction. For the SceneTwo.csv file, three different systems of interaction are possible, because the "Controllers - Direct Interaction" is not possible in this case.

Listing 3.1: Extraction of four tables of each interaction system | SceneOne.csv

```
SceneOne = readtable("/MATLAB Drive/01SceneOne.xlsx");

ControllersRaycasting = table;
ControllersDirect = table;
HandTrackingRaycasting = table;
HandTrackingDirect = table;

for ii = 1:size(SceneOne,1)
    if (strcmp(char(SceneOne(ii,2).Var2),'Controllers') &&
       strcmp(char(SceneOne(ii,3).Var3),'Raycasting'))
           ControllersRaycasting(size(ControllersRaycasting
              ,1)+1,:) = SceneOne(ii,:);
    end
    if (strcmp(char(SceneOne(ii,2).Var2),"Controllers") &&
       strcmp(char(SceneOne(ii,3).Var3),"Direct"))
          ControllersDirect(size(ControllersDirect,1)+1,:) =
             SceneOne(ii,:);
    end
    if (strcmp(char(SceneOne(ii,2).Var2),"Hands") && strcmp(
       char(SceneOne(ii,3).Var3),"Raycasting"))
           HandTrackingRaycasting(size(HandTrackingRaycasting
              ,1)+1,:) = SceneOne(ii,:);
    end
    if (strcmp(char(SceneOne(ii,2).Var2),"Hands") && strcmp(
       char(SceneOne(ii,3).Var3),"Direct"))
           HandTrackingDirect(size(HandTrackingDirect,1)+1,:)
               = SceneOne(ii,:);
    end
end
```

Listing 3.2: Extraction of four tables of each interaction system | SceneTwo.csv

```
SceneTwo = readtable("/MATLAB Drive/02SceneTwo.xlsx");

ControllersRaycasting = table;
HandTrackingRaycasting = table;
HandTrackingDirect = table;

for ii = 1:size(SceneTwo,1)
    if (strcmp(char(SceneTwo(ii,2).Var2),'Controllers') &&
       strcmp(char(SceneTwo(ii,3).Var3),'RayCasting'))
           ControllersRaycasting(size(ControllersRaycasting
              ,1)+1,:) = SceneTwo(ii,:);
    end
    if (strcmp(char(SceneTwo(ii,2).Var2),"Hand") && strcmp(
       char(SceneTwo(ii,3).Var3),"RayCasting"))
           HandTrackingRaycasting(size(HandTrackingRaycasting
              ,1)+1,:) = SceneTwo(ii,:);
    end
    if (strcmp(char(SceneTwo(ii,2).Var2),"Hand") && strcmp(
       char(SceneTwo(ii,3).Var3),"Direct"))
           HandTrackingDirect(size(HandTrackingDirect,1)+1,:)
               = SceneTwo(ii,:);
    end
end
```

Listing 3.3: Extraction of four tables of each interaction system | SceneThree.csv

```
SceneThree = readtable("/MATLAB Drive/SceneThree.csv");

ControllersRaycasting = table;
ControllersDirect = table;
HandTrackingRaycasting = table;
HandTrackingDirect = table;

for ii = 1:size(SceneThree,1)
    if (strcmp(char(SceneThree(ii,2).InteractionMethod),'
       Controllers') && strcmp(char(SceneThree(ii,3).
       InteractionMetaphor),'Raycasting'))
           ControllersRaycasting(size(ControllersRaycasting
              ,1)+1,:) = SceneThree(ii,:);
    end
    if (strcmp(char(SceneThree(ii,2).InteractionMethod),"
       Controllers") && strcmp(char(SceneThree(ii,3).
       InteractionMetaphor),"Direct"))
          ControllersDirect(size(ControllersDirect,1)+1,:) =
             SceneThree(ii,:);
    end
    if (strcmp(char(SceneThree(ii,2).InteractionMethod),"
       Hands") && strcmp(char(SceneThree(ii,3).
       InteractionMetaphor),"Raycasting"))
           HandTrackingRaycasting(size(HandTrackingRaycasting
              ,1)+1,:) = SceneThree(ii,:);
    end
    if (strcmp(char(SceneThree(ii,2).InteractionMethod),"
       Hands") && strcmp(char(SceneThree(ii,3).
       InteractionMetaphor),"Direct"))
           HandTrackingDirect(size(HandTrackingDirect,1)+1,:)
               = SceneThree(ii,:);
    end
end
```

After creating separate tables for each interaction system, I conducted a detailed analysis of the data of the respective utilised systems.

For the Grab task, I extracted the average time to collect objects and the number of fallen objects. I generated figures based on this data, which will be presented in the sixth chapter of this thesis.

Regarding the Type task, I captured the number of wrong answers, the frequency of users clicking the cancel button, and the total number of button clicks. Utilising MATLAB, I produced figures to represent these results visually.

Similarly, for the Manipulation task, I recorded the number of incorrect answers, instances of users clicking the cancel button, and the overall number of button clicks. Again,

MATLAB facilitated the generation of figures to illustrate the outcomes.

Listing 3.4: Calculate the duration of the scene for each interaction system | SceneOne.csv

```
ControllersRaycastingDurations = duration();
for ii = 1:size(ControllersRaycasting,1)
    startTime = datetime(char(ControllersRaycasting(ii,7).
       Var7), 'InputFormat', 'dd-MMM-yyyy HH:mm:ss');
    endTime =  datetime(char(ControllersRaycasting(ii,8).Var8
       ), 'InputFormat', 'dd-MMM-yyyy HH:mm:ss');
    ControllersRaycastingDurations(ii) = endTime-startTime;
end
meanControllersRaycastingDurations = mean(
   ControllersRaycastingDurations);

ControllersDirectDurations = duration();
for ii = 1:size(ControllersDirect,1)
    startTime = datetime(char(ControllersDirect(ii,7).Var7),
       'InputFormat','dd-MMM-yyyy HH:mm:ss');
    endTime =  datetime(char(ControllersDirect(ii,8).Var8), '
       InputFormat','dd-MMM-yyyy HH:mm:ss');
    ControllersDirectDurations(ii) = endTime-startTime;
end
meanControllersDirectDurations = mean(
   ControllersDirectDurations);

HandtrackingRaycastingDurations = duration();
for ii = 1:size(HandTrackingRaycasting,1)
    startTime = datetime(char(HandTrackingRaycasting(ii,7).
       Var7), 'InputFormat', 'dd-MMM-yyyy HH:mm:ss');
    endTime =  datetime(char(HandTrackingRaycasting(ii,8).
       Var8), 'InputFormat', 'dd-MMM-yyyy HH:mm:ss');
    HandtrackingRaycastingDurations(ii) = endTime-startTime;
end
meanHandtrackingRaycastingDurations = mean(
   HandtrackingRaycastingDurations);

HandtrackingDirectDurations = duration();
for ii = 1:size(HandTrackingDirect,1)
    startTime = datetime(char(HandTrackingDirect(ii,7).Var7),
        'InputFormat', 'dd-MMM-yyyy HH:mm:ss');
    endTime =  datetime(char(HandTrackingDirect(ii,8).Var8),
       'InputFormat', 'dd-MMM-yyyy HH:mm:ss');
    HandtrackingDirectDurations(ii) = endTime-startTime;
end
```

```
meanHandtrackingDirectDurations = mean(
   HandtrackingDirectDurations);

disp(['ControllersRaycasting: ', char(
   meanControllersRaycastingDurations)]);
disp(['ControllersDirect: ', char(
   meanControllersDirectDurations)]);
disp(['HandTrackingRaycasting: ', char(
   meanHandtrackingRaycastingDurations)]);
disp(['HandTrackingDirect: ', char(
   meanHandtrackingDirectDurations)]);
```

Listing 3.5: Calculate the duration of the scene for each interaction system | SceneTwo.csv

```
%Calculate the average time for each combination.
ControllersRaycastingDurations = duration();
for ii = 1:size(ControllersRaycasting,1)
    startTime = datetime(char(ControllersRaycasting(ii,9).
       Var9), 'InputFormat', 'dd-MMM-yyyy HH:mm:ss');
    endTime =  datetime(char(ControllersRaycasting(ii,10).
       Var10), 'InputFormat', 'dd-MMM-yyyy HH:mm:ss');
    ControllersRaycastingDurations(ii) = endTime-startTime;
end
meanControllersRaycastingDurations = mean(
   ControllersRaycastingDurations);

HandtrackingRaycastingDurations = duration();
for ii = 1:size(HandTrackingRaycasting,1)
    startTime = datetime(char(HandTrackingRaycasting(ii,9).
       Var9), 'InputFormat', 'dd-MMM-yyyy HH:mm:ss');
    endTime =  datetime(char(HandTrackingRaycasting(ii,10).
       Var10), 'InputFormat',  'dd-MMM-yyyy HH:mm:ss');
    HandtrackingRaycastingDurations(ii) = endTime-startTime;
end
meanHandtrackingRaycastingDurations = mean(
   HandtrackingRaycastingDurations);

HandtrackingDirectDurations = duration();
for ii = 1:size(HandTrackingDirect,1)
    startTime = datetime(char(HandTrackingDirect(ii,9).Var9),
        'InputFormat',  'dd-MMM-yyyy HH:mm:ss');
    endTime =  datetime(char(HandTrackingDirect(ii,10).Var10)
       , 'InputFormat',  'dd-MMM-yyyy HH:mm:ss');
    HandtrackingDirectDurations(ii) = endTime-startTime;
end
```

```
meanHandtrackingDirectDurations = mean(
   HandtrackingDirectDurations);

disp(['ControllersRaycasting: ', char(
   meanControllersRaycastingDurations)]);
disp(['HandTrackingRaycasting: ', char(
   meanHandtrackingRaycastingDurations)]);
disp(['HandTrackingDirect: ', char(
   meanHandtrackingDirectDurations)]);
```

Listing 3.6: Calculate the duration of the scene for each interaction system | SceneThree.csv

```
%Calculate the average time for each combination.
ControllersRaycastingDurations = duration();
for ii = 1:size(ControllersRaycasting,1)
    startTime = datetime(char(ControllersRaycasting(ii,7).
       StartTime), 'InputFormat', 'dd-MMM-yyyy HH:mm:ss', '
       Locale', 'it_IT');
    endTime = datetime(char(ControllersRaycasting(ii,8).
       EndTime), 'InputFormat', 'dd-MMM-yyyy HH:mm:ss', '
       Locale', 'it_IT');
    ControllersRaycastingDurations(ii) = endTime-startTime;
end
meanControllersRaycastingDurations = mean(
   ControllersRaycastingDurations);

ControllersDirectDurations = duration();
for ii = 1:size(ControllersDirect,1)
    startTime = datetime(char(ControllersDirect(ii,7).
       StartTime), 'InputFormat', 'dd-MMM-yyyy HH:mm:ss', '
       Locale', 'en_US');
    startTime = datetime(datenum(startTime), 'ConvertFrom', '
       datenum', 'Format', 'dd/MM/yyyy HH:mm:ss');
    tempTime = datetime(char(ControllersDirect(ii,8).EndTime)
       , 'InputFormat', 'dd-MMM-yyyy HH:mm:ss', 'Locale', '
       en_US');
    endTime = datetime(datenum(tempTime), 'ConvertFrom', '
       datenum', 'Format', 'dd/MM/yyyy HH:mm:ss');

    ControllersDirectDurations(ii) = endTime-startTime;
end
meanControllersDirectDurations = mean(
   ControllersDirectDurations);
```

```
HandtrackingRaycastingDurations = duration();
for ii = 1:size(HandTrackingRaycasting,1)
tempTime = datetime(char(HandTrackingRaycasting(ii,7).
   StartTime), 'InputFormat', 'dd-MMM-yyyy HH:mm:ss', 'Locale
   ', 'en_US');
startTime = datetime(datenum(tempTime), 'ConvertFrom', '
   datenum', 'Format', 'dd/MM/yyyy HH:mm:ss');
tempTime = datetime(char(HandTrackingRaycasting(ii,8).EndTime
   ), 'InputFormat', 'dd-MMM-yyyy HH:mm:ss', 'Locale', 'en_US
   ');
endTime = datetime(datenum(tempTime), 'ConvertFrom', 'datenum
   ', 'Format', 'dd/MM/yyyy HH:mm:ss');

    HandtrackingRaycastingDurations(ii) = endTime-startTime;
end
meanHandtrackingRaycastingDurations = mean(
   HandtrackingRaycastingDurations);

HandtrackingDirectDurations = duration();
for ii = 1:size(HandTrackingDirect,1)
    tempTime = datetime(char(HandTrackingDirect(ii,7).
       StartTime), 'InputFormat', 'dd-MMM-yyyy HH:mm:ss', '
       Locale', 'en_US');
    startTime = datetime(datenum(tempTime), 'ConvertFrom', '
       datenum', 'Format', 'dd/MM/yyyy HH:mm:ss');
    tempTime = datetime(char(HandTrackingDirect(ii,8).EndTime
       ), 'InputFormat', 'dd-MMM-yyyy HH:mm:ss', 'Locale', '
       en_US');
    endTime = datetime(datenum(tempTime), 'ConvertFrom', '
       datenum', 'Format', 'dd/MM/yyyy HH:mm:ss');
    HandtrackingDirectDurations(ii) = endTime-startTime;
end
meanHandtrackingDirectDurations = mean(
   HandtrackingDirectDurations);

disp(['ControllersRaycasting: ', char(
   meanControllersRaycastingDurations)]);
disp(['ControllersDirect: ', char(
   meanControllersDirectDurations)]);
disp(['HandTrackingRaycasting: ', char(
   meanHandtrackingRaycastingDurations)]);
disp(['HandTrackingDirect: ', char(
   meanHandtrackingDirectDurations)]);
```

In the sixth chapter, I will show the results of the data I collected.

Chapter 4

Application usage

In the fourth chapter, I will explain how scene selection works. Additionally, I will describe the extras around the application, such as the 'About,' 'Options,' and 'Quit' sections.

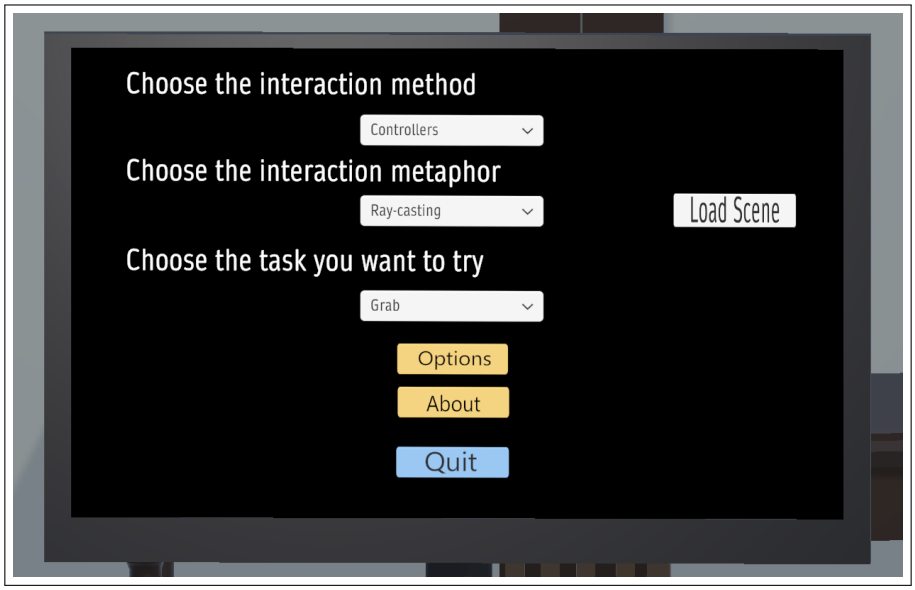
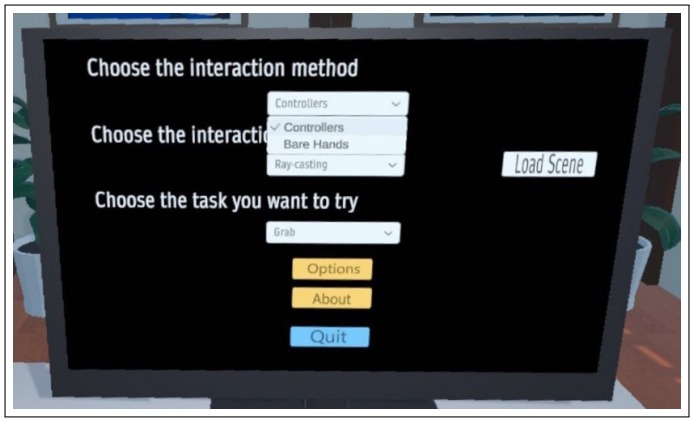


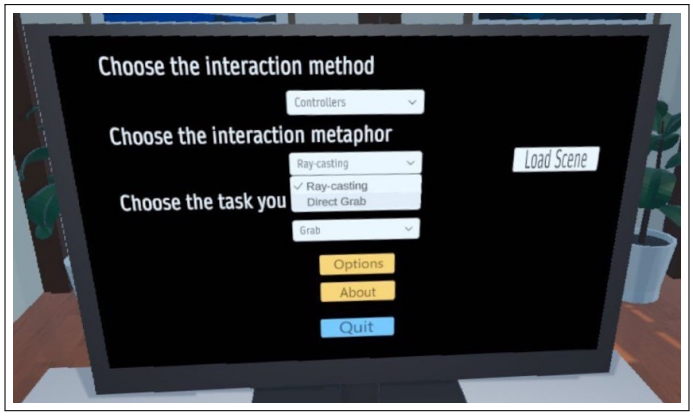
Fig. 4.1: Start Screen - Scene Selection

4.1 Scene selection

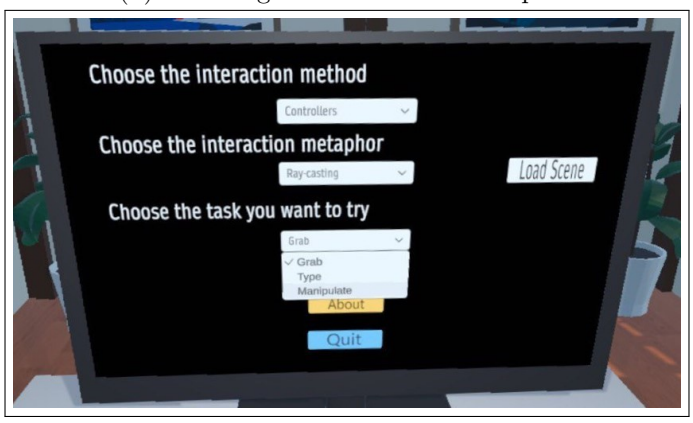
For the scene selection, I opted to use three different drop-down menus. This decision was driven by the need to offer users a wide range of options without filling the interface with individual buttons for each possibility. Above the first drop-down menu, there is the written "Choose the interaction method". In this menu, users can select between controllers or bare hands. With the second drop-down menu, the interaction metaphor can be selected for the experience. In this case, the possible choices are ray-casting or direct interaction; "Choose the interaction metaphor" is the caption on the multiple-choice menu. Finally, the third drop-down menu lets you choose the corresponding task to start the experience. The written above this last drop-down is "Choose the task you want to try". Next to these three drop-down menus, there's a button "Load Scene". When users press it, they receive auditory feedback as a sound. This loading mechanism is also present at the end of every test scene. Another particular feature I have added is a "Fader screen". This particular object allows for a smoother transition between scenes. After pressing the 'Load Scene' button, users experience a brief two-second blackout before the scene tutorial begins. This intentional pause aims to prevent sudden changes that could potentially unsettle the user.



(a) Choosing the Interaction Method



(b) Choosing the Interaction Metaphor



(c) Choosing the Task

Fig. 4.2: The drop-down menus. (a) Choosing the Interaction Method (b) Choosing the Interaction Metaphor (c) Choosing the Task

The list of possible scenes that can be loaded is:

1. Grab task, controllers and ray-casting.
2. Grab task, bare hands and ray-casting.
3. Grab task, controllers and direct interaction.
4. Grab task, bare hands and direct interaction.
5. Type task, controllers and ray-casting.
6. Type task, bare hands and ray-casting.
7. Type task, bare hands and direct interaction.
8. Manipulation task, controllers and ray-casting.
9. Manipulation task, bare hands and ray-casting.
10. Manipulation task, controllers and direct interaction.
11. Manipulation task, bare hands and direct interaction.

If a user tries to load the combination "Type task, controllers and direct interaction' the message "this combination does not exist" appears, and the error sound is played.

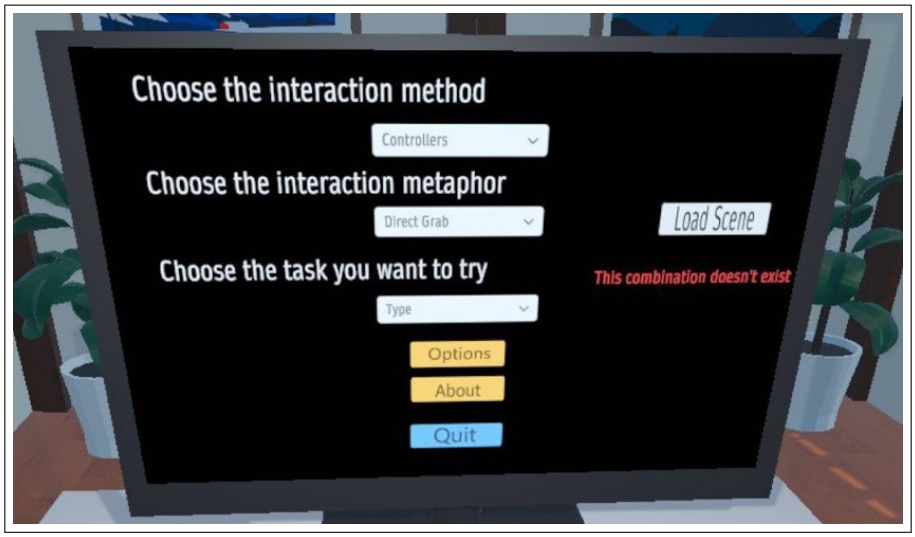


Fig. 4.3: Not valid selection of interaction system

4.2 Extras

In the Home of VR InteracTest app, I wanted to add some extras for the users beyond the scene selection mechanism. Indeed, I designed three different buttons below the drop-down menus described in the previous paragraph: the About button, the Options button and the Quit button.

4.2.1 Options, About and Quit

The Options button is like the typical Settings button of the applications. The user can press this button to load another screen on the TV. When this button is clicked, all the previous elements on the TV screen disappear and are replaced by another menu. In this section, the user can set the volume level. And there is also the possibility to change how the view is loaded in the HMD. The choices in this case are Snap Turn and Continuous Turn.

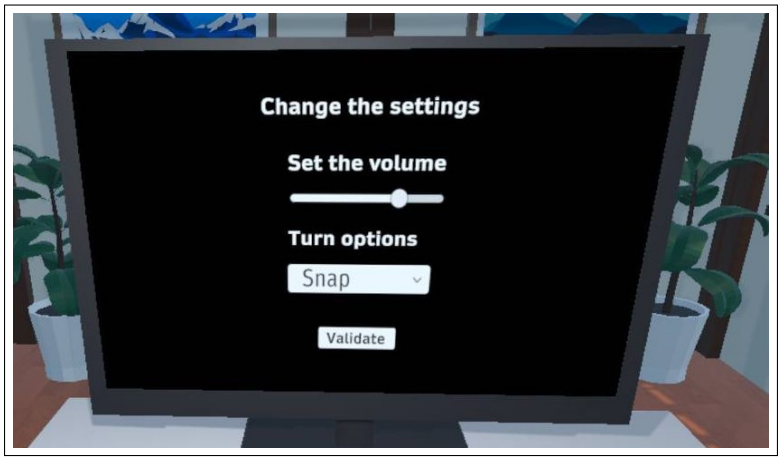


Fig. 4.4: The "Option" section of VR InteracTest

Snap Turn typically refers to a virtual reality (VR) locomotion technique. Snap Turning allows users to rotate their view in fixed increments, providing a more comfortable way to navigate VR environments without needing continuous physical turning.

The idea is that instead of smoothly turning around, the view "snaps" to predefined angles, which can help reduce motion sickness for some users and provide a more controlled experience.

Continuous Turn in Unity, especially in virtual reality (VR), refers to a locomotion technique where the user can smoothly rotate their view in any direction by continuously inputting a control, such as moving a joystick or rotating a thumbstick. Unlike Snap Turn, which rotates the view in fixed increments, Continuous Turn provides a more fluid and continuous rotation.

Continuous Turn is designed to simulate natural head movement and provides a more immersive experience for users comfortable with this locomotion form. However, continuous rotation can potentially induce motion sickness in some users, and developers often

provide options for users to choose between continuous or snap turning based on their preferences.

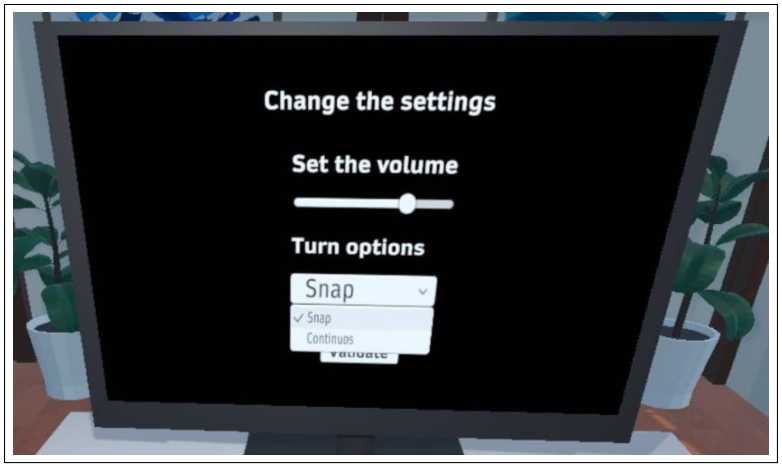


Fig. 4.5: Snap or Continuous turn - Drop-down menu

The other extra of this screen is the About section. When this button is pressed, all the drop-down menus and sentences on the Start screen described above disappear. This section of the start scene presents the text 'This application was made by Anna Sansoè in 2023 for her thesis project.' As customary in many applications, I wanted to add a credit section to acknowledge the author.

Fig. 4.6: The "About" section of VR InteracTest

The Quit button is helpful to close the up. But this button also has the utility to save and close the generated CSV files.

Chapter 5

Experiment

The purpose of this work is primarily to put on the market an application that helps developers, designers etc, in their works. VR InteracTest assists them in comparing their metaphors and the methods of interaction they have invented and would like to put on the market. This is why I conducted a study with twenty people to test my application and ensure that it is valid and truly serves the purpose for which it was designed. This chapter presents the whole experiment procedure, highlighting my decisions during the design phases. In particular, this chapter is divided into two parts. First, a description of the experiment goals is provided. Second, I describe the design of each experiment and the reasoning behind it. Moreover, the second part also provides insights on the pre and post-experience questionnaires.

5.1 Experiment goals

The experiment's goals are to test in two different ways, subjectively and objectively, the interaction metaphors and methods that I have meticulously chosen, with some users. To do that, I conducted the study with twenty subjects. Some of them were VR-experts, while others never experienced VR before my test. This configuration of testing groups aligns with those usually adopted in the current literature regarding numbers, background mixtures, and the number of proposed tasks to each user. Specifically, I organised the study in such a way as to let users try either two or three interaction systems, repeating two times the three tasks. Nevertheless, each user was asked to experience the application with both bare hands and controllers. Such an insight guarantees the most comprehensive evaluation and comparison between the proposed interaction system.

The subjective aspects that I intended to study focused on the personal feelings derived from the experience. In particular, I wanted to know from the people I selected if they perceived an essential difference between the diverse system of interaction that I provided.

Concerning the objective aspects of the tested interaction systems, I wanted to investigate the number of errors the user committed with the specific system and the time to finish a specific goal.

5.2 Experiment design

The experiment that I conducted is divided into three different steps. The first one is an introductory questionnaire. This first step is a demographic questionnaire, with which I define the age brackets of the subjects I have chosen; I understand the educational level percentages and the level of expertise of the participants. The second step of the experiment was the experience of VR InteracTest, the users had to choose the interaction system I was telling them. At the end of the entire experience, the subjects had to submit the answers to my second questionnaire. With this second survey, I intended to find out which one of the proposed systems was the most appreciated.

To submit both questionnaires to the subjects, I used Google Forms.

5.2.1 Introductory questionnaire

As I anticipated in the paragraph above, the introductory questionnaire is a demographic survey. This first question sheet consists of the following requests:

Q1. Provide your age.

Q2. Provide your sex.

Q3. Level of instruction.

Q4. Do you have any interest or involvement in video games, in general?

Q5. What are your favourite gaming platforms, or which ones do you usually play?

Q6. How much time do you spend playing video games on these platforms on average during the week?

Q7. Have you ever used virtual reality before?

Q8. What kind of VR devices have you used in the past?

Q9. Have you ever participated in a VR study or test before?

Q10. Have you ever experienced motion sickness?

The questions are a mixture of open and closed-ended questions. Specifically, Q1, Q2, and Q3 are closed-ended questions; the others are open-ended. The questionnaire has been developed ad-hoc to have a more comprehensive overview of the previous user experience. The purpose of this form was to understand, in addition to purely demographic questions, the experience level acquired before taking the test with my application. This demographic data is essential because if some people are non-experts, like in this case, the designers and developers of VR interaction systems can discover what kind of system is more straightforward as the first approach in Virtual Reality. Oppositely, having some VR experts is helpful to understand whether a newly implemented system is too complex for them, it means that this specific system needs to be validated.

5.2.2 The experience

In this chapter, we'll go through what the participants went through. I ensured the headset was in "Cast" mode at the beginning of each test session. This trick allowed me to observe user movements in both the virtual and real worlds, enabling me to assist them at any moment. Then, the Room-scale Boundary of Oculus Quest were drawn by me and also the VR InteracTest app was started. Following this setup, the participant put on the Oculus headset and adjusted it on the head to their preferred comfort level.

After that, I explained the basics of interactions in VR, especially if it was their first time in a virtual world. This quick tutorial covered essential controls and gestures, setting the stage for a smoother experience.

Before diving into the virtual experience, participants were informed about the interaction system of interaction they should use. To achieve this, I employed a table, outlined below 5.1, to keep track of the methods already utilised, those yet to be employed, and which were specific to the user in that model. The method I used to choose the sequence of interaction systems allows me to have participants use controllers first and then bare hands, or vice versa. The interaction metaphors are typically kept the same to enable users to assess their effectiveness based on the medium they interact in VR. Another criterion I maintained during the test sessions was to alternate users, having some start with controllers and others begin with bare hands. In the Table 5.1:

- Task 1 stands for the Grab action.
- Task 2 stands for the Type action.
- Task 3 stands for the Manipulate action.

Returning to the description of the actual experience. The participant was instructed to complete all three initial tasks using the first interaction method before redoing them with the second interaction method. If the interaction metaphor occurred was "Direct Interaction", the second task of Typing was impossible to load because, in Virtual Reality, you can not push a button by pressing it directly with the controller. In this case, the "Controller and Direct Interaction" was replaced with "Controller and Ray-casting".

The duration of this experimental session was approximately 30 minutes. The time to complete the experience also depends significantly on the participants' prior experience and abilities.

After completing the entire experience, the participant was invited to complete the second questionnaire. The request to complete the questionnaire immediately following the experience was driven by the need to capture the impressions formed during the test. Delaying the completion of the questionnaire may have allowed the passage of time to influence those initial and immediate sensations.

User #	**Task 1**	**Task 2**	**Task 3**
1	Controller and Ray Hand and Ray	Controller and Ray Hand and Ray	Controller and Ray Hand and Ray
2	Controller and Direct Hand and Direct	Controller and Ray Hand and Direct	Controller and Direct Hand and Direct
3	Controller and Ray Hand and Ray	Controller and Ray Hand and Ray	Controller and Ray Hand and Ray
4	Controller and Direct Hand and Direct	Controller and Ray Hand and Direct	Controller and Direct Hand and Direct
5	Controller and Ray Hand and Ray	Controller and Ray Hand and Ray	Controller and Ray Hand and Ray
6	Controller and Direct Hand and Direct	Controller and Ray Hand and Direct	Controller and Direct Hand and Direct
7	Controller and Ray Hand and Ray	Controller and Ray Hand and Ray	Controller and Ray Hand and Ray
8	Controller and Direct Hand and Direct	Controller and Ray Hand and Direct	Controller and Direct Hand and Direct
9	Controller and Ray Hand and Ray	Controller and Ray Hand and Ray	Controller and Ray Hand and Ray
10	Controller and Direct Hand and Direct	Controller and Ray Hand and Direct	Controller and Direct Hand and Direct
11	Controller and Ray Hand and Ray	Controller and Ray Hand and Ray	Controller and Ray Hand and Ray
12	Controller and Direct Hand and Direct	Controller and Ray Hand and Direct	Controller and Direct Hand and Direct
13	Hand and Ray Controller and Ray	Hand and Ray Controller and Ray	Hand and Ray Controller and Ray
14	Controller and Direct Hand and Direct	Controller and Ray Hand and Direct	Controller and Direct Hand and Direct
15	Controller and Ray Hand and Ray	Controller and Ray Hand and Ray	Controller and Ray Hand and Ray
16	Controller and Direct Hand and Direct	Controller and Ray Hand and Direct	Controller and Direct Hand and Direct
17	Controller and Ray Hand and Ray	Controller and Ray Hand and Ray	Controller and Ray Hand and Ray
18	Controller and Direct Hand and Direct	Controller and Ray Hand and Direct	Controller and Direct Hand and Direct
19	Controller and Ray Hand and Ray	Controller and Ray Hand and Ray	Controller and Ray Hand and Ray
20	Controller and Direct Hand and Direct	Controller and Ray Hand and Direct	Controller and Direct Hand and Direct

Table 5.1: Users Interaction System for the experiment

5.2.3 Post-experience questionnaire

This paragraph provides a detailed overview of the questionnaire's design, the considered variables, and the underlying rationale behind methodological choices.

The Post-experience questionnaire starts with the first section composed of these three questions:

Q1. Which interaction method do you prefer between "Controller" and "Hands" in VR?

Q2. Which interaction metaphor did you find most intuitive and effective for completing tasks?

Q3. Do you have something that you really don't like about this app?

Q1 and Q2 are closed-ended questions. Specifically, for Q1, the predefined answers are "Controllers" or "Bare Hands", and for Q2, are "Ray-casting" or Direct interaction". Both those questions are followed by the question "Why?" and the participants were able to write whatever they want to. Also, for the last question (Q3), the subjects can write whatever they want with an open-ended response.

After the initial section, the second section is presented. In particular, this second part of the questionnaire consists of Likert scale questions, where participants are asked to provide their responses on a scale from 1 to 5, indicating their level of agreement or disagreement with each statement. Moreover, I requested to rate the complexity of the tasks on a scale from 1 (very simple) to 5 (very complex) related to the specific system of interaction. It was specified that each participant should respond only to the statements corresponding to the individual interaction systems used with their respective tasks. The second section consisted of the following statements:

Q1. The Grab task with controllers and ray-casting

Q2. The Type task with controllers and ray-casting

Q3. The Manipulation task with controllers and ray-casting

Q4. The Grab task with controllers and direct interaction

Q5. The Manipulation task with controllers and direct interaction

Q6. The Grab task with bare hands and ray-casting

Q7. The Type task with bare hands and ray-casting

Q8. The Manipulation task with bare hands and ray-casting

Q9. The Grab task with bare hands and direct interaction

Q10. The Type of task with bare hands and direct interaction

Q11. The Manipulation task with bare hands and direct interaction

To help the people compile this part of the questionnaire, I allowed them to consult Table 5.1. This stratagem was necessary because each participant experienced only six out of the eleven possible scenes, as I explained in the previous paragraph.

The last section of the questionnaire was the System Usability Scale (SUS), [39]. I founded diferrent studies that evaluated their interactions systems using SUS: [8], [27], [14], [40], [15], [29], [21], [23], [31], [36]. The System Usability Scale (SUS) is a widely used questionnaire for evaluating the usability of a system, product, or service. It was developed by John Brooke in 1986 and has since become a standard tool in usability testing and user experience research.

The SUS consists of a 10-item questionnaire with a five-point Likert scale ranging from "Strongly Disagree" to "Strongly Agree." The questions assess various usability aspects, including the user's perception of the system's complexity, ease of use, and overall user satisfaction.

The scores from each question are converted, summed, and then multiplied by 2.5 to obtain a usability score ranging from 0 to 100. A higher score indicates a higher perceived usability. The SUS is valuable for obtaining a quick and reliable measure of the subjective usability of a system from the user's perspective. It's often used with other usability testing methods to provide a comprehensive view of the user experience.

The ten System Usability Scale questions are:

Q1. I think that I would like to use this system frequently.

Q2. I found the system unnecessarily complex.

Q3. I thought the system was easy to use.

Q4. I think that I would need the support of a technical person to be able to use this system.

Q5. I found the various functions in this system were well integrated.

Q6. I thought there was too much inconsistency in this system.

Q7. I would imagine that most people would learn to use this system very quickly.

Q8. I found the system very cumbersome to use.

Q9. I felt very confident using the system.

Q10. I needed to learn a lot of things before I could get going with this system.

Within the questionnaire were four distinct SUS surveys, one for each interaction system: controller & ray-casting, bare hand & ray-casting, controllers & direct interaction, and bare hands & direct interactions. In this case, each participant was instructed to complete only the SUS corresponding to the specific interaction system they used to perform the tasks. This tailored approach ensures a focused and meaningful assessment based on each participant's unique experience.

This post-experience questionnaire required at least ten minutes to be completed. The whole experience with post questionnaires and VR experience is known to be 40 minutes.

Chapter 6

Results & discussion

Within this chapter, I delve into the outcomes derived from the testing conducted with the participation of twenty individuals. The results presented here encapsulate the diverse responses and performances across various tasks. This comprehensive analysis provides valuable insights into the effectiveness and user experience associated with the implemented interaction systems.

6.1 Experiment's outcome

Within this exploration, I have organised the findings into three distinct subsections to provide a structured presentation of the outcomes. The initial subsection delves into the "Results of the Preliminary Questionnaire", offering insights into participants' initial expectations and perceptions. Following this, the focus shifts to the "Results of the VR InteracTest Experience", where a detailed examination of participant activities and interactions within the VR application is presented. Lastly, the chapter concludes by examining the "Results of the Post-Experience Questionnaires", shedding light on participants' reactions and opinions following their immersive interaction.

6.1.1 Results of the Preliminary Questionnaire

In this section, I present the questionnaire results before the experience, providing an overview of participants' initial expectations and perceptions. The questionnaire is detailed in Chapter 5.2.1.

The first question provided information on the age range of the participants, spanning from 24 to 31 years, as shown in Fig. 6.1. The majority of the participants are 26 years old.

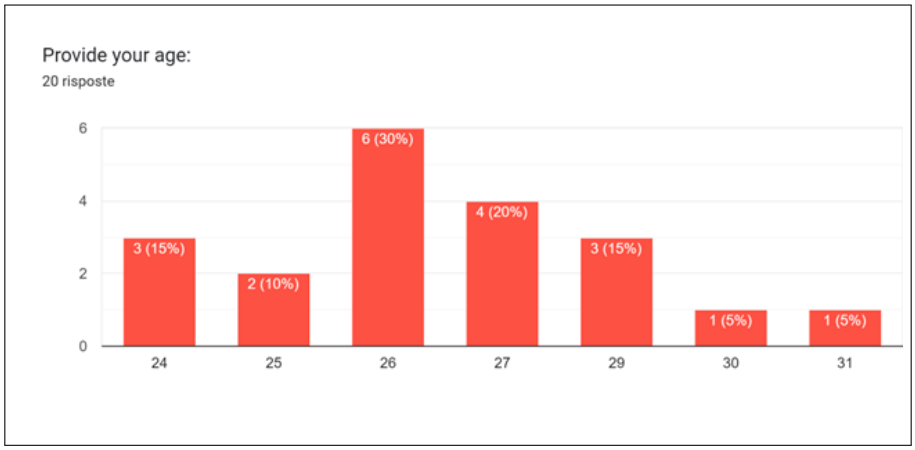


Fig. 6.1: Age of the participants

The second question I asked was: "Provide your sex". 50% of the twenty participants in my study were men. One participant chose not to specify their sex and the remaining identified as female. These data are presented in the pie chart in the Fig. 6.2.

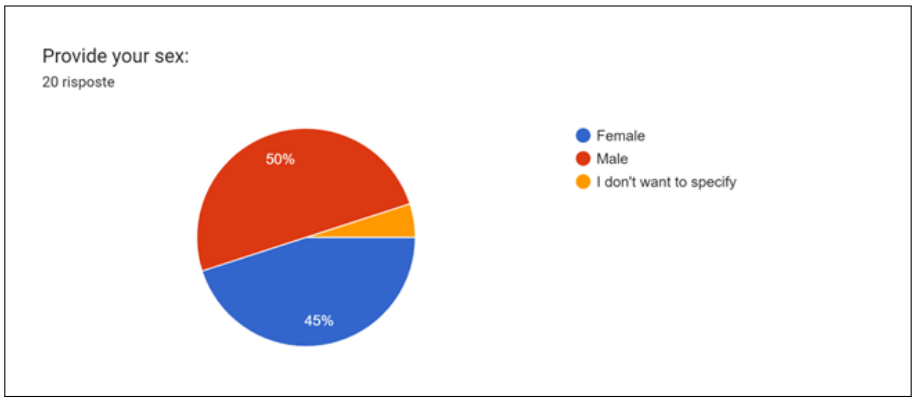


Fig. 6.2: Sex of the participants

With the third question, I can know the participants' instruction level. The results of this question are presented in the Fig. 6.3. As we can see, most people have a Master's degree.

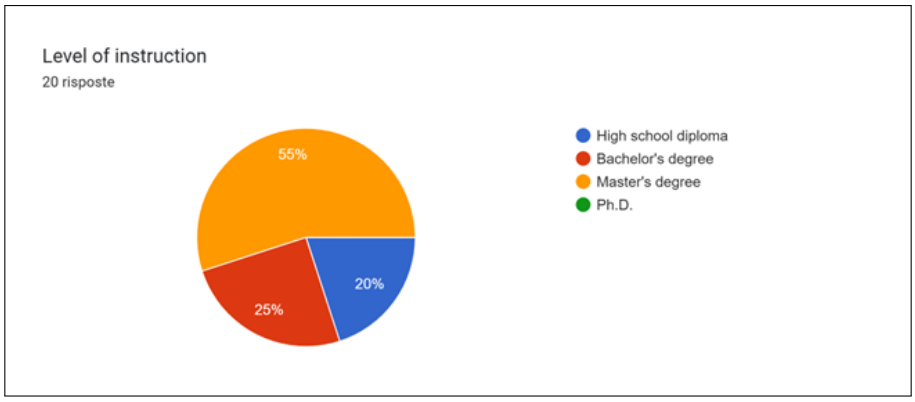


Fig. 6.3: Level of instruction of the participants

In the second part of the Introductory Questionnaire, I inquired about participants' general interest in video games. Most participants responded affirmatively, as depicted in Fig. 6.4. This question helped me understand that most participants who tried my application are generally familiar with gamepads or joysticks and game dynamics. So, the chosen population is heterogeneous in this respect.

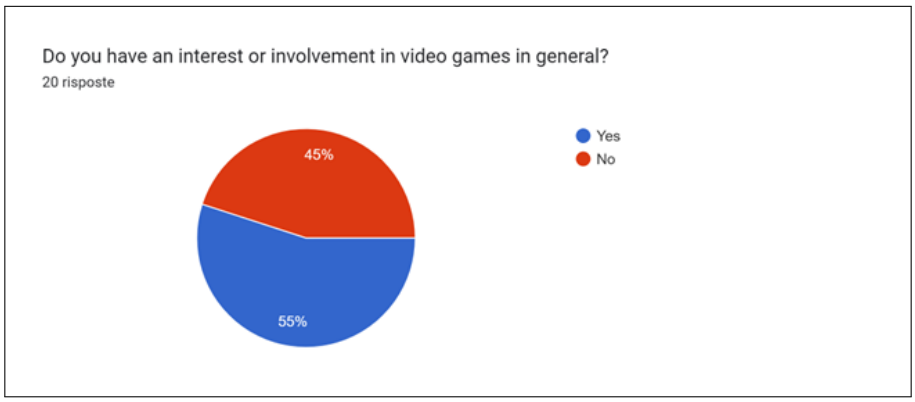


Fig. 6.4: Participants who play Video Games

Following that, I asked if they had ever tried VR (Fig. 6.5) and, related to that, if they had ever participated in a VR study (Fig. 6.6) or experienced motion sickness using VR (Fig. 6.7).

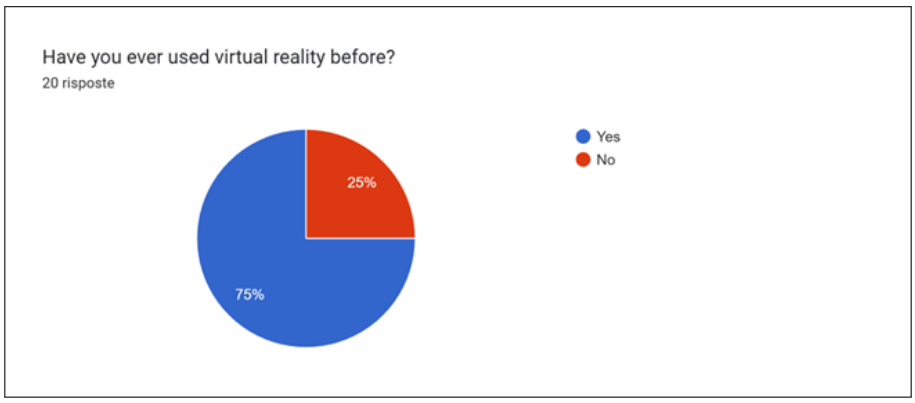


Fig. 6.5: Participants that tried VR

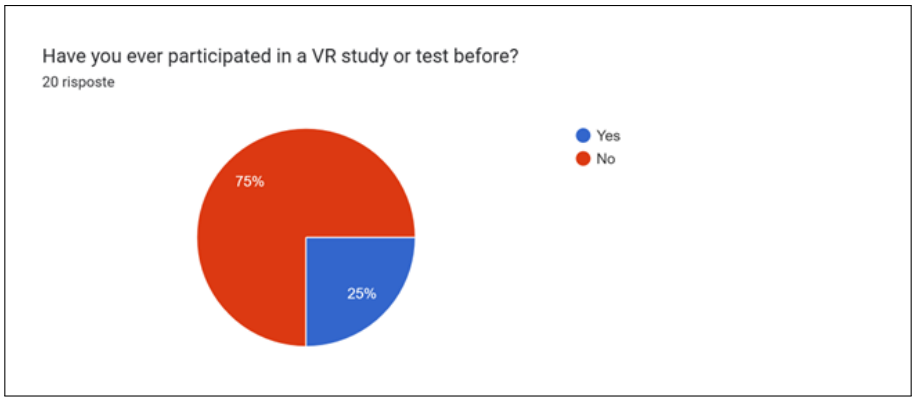


Fig. 6.6: Participants that participated in a VR Study

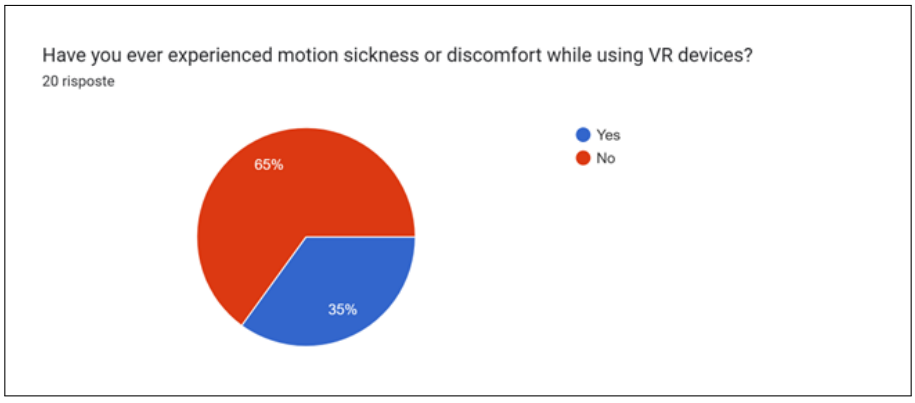


Fig. 6.7: Participants that ever experienced motion sickness

6.1.2 Results of the VR InteracTest Experience

In this thesis section, I will present the results obtained by having twenty individuals experience VR InteracTest. Since testing all tasks with every possible interaction system proved too time-consuming for the participants, I exposed them to only two interaction systems. The selection criteria were consistent for all twenty participants. Each participant experienced both interaction methods, each associated with a single metaphor. The interaction systems employed with the participants are detailed in Table 5.1. As evident and as I have emphasised throughout this exposition, achieving the combination "Controller with Direct interaction for the Type task" is impossible. Due to this limitation, I opted to have users perform the second task twice, but when the interaction method involved controllers, the "Controller and Raycasting" system was utilised. Consequently, we observe 20 recurrences for this interaction system and ten recurrences for all others. Chapter 3 provides a detailed exposition of the data saved for each task.

Results for the Grab task

For this first task, I initially intended to present the average duration users took to collect the twenty-five objects using each interaction method. The mean duration for each interaction system is exposed below in Table 6.1.

Interaction System	Duration
Controller and Ray-casting	00:01:56
Controllers and Direct Interaction	00:01:49
Bare Hands and Ray-casting	00:02:08
Bare Hands and Direct Interaction	00:02:42

Table 6.1: Average duration of the scene with the grab task.

As can be observed from the data exposed in Table 6.1, the lowest average duration among the interaction systems is that of the "Controllers and Direct" system. So, from an objective point of view, concerning this evaluation metric, the most efficient interaction system is the "Controllers and Direct Interaction". This calculation, like all subsequent calculations, was derived using MATLAB. The code iterates through the data sets, each corresponding to a different interaction system, extracted as I explained in Listing 3.1. For each row in this data-sets, it extracts start and end time information from the corresponding columns. The code then converts these time values to datetime objects and calculates the duration by subtracting the start and end times. The resulting duration is stored in an array named ControllersRaycastingDurations. In summary, the code is designed to calculate and store the duration between start and end times for each row in the specified data sets.

As a second evaluation metric for the interaction systems, I decided to take the Number of Collected Objects. With this metric, I'm able to know with which interaction system is simpler to finish the proposed task shown by the figures Fig. 6.8 and Fig. 6.9, ten participants used each system of interaction. More precisely, ten used both "Controller and Raycasting" and "Hand Tracking and Raycasting"; the others used both "Controller and

Direct" and "Hand Tracking and Direct". Comparing the four histograms, we can notice that 25 objects are collected from 9 users out of 10 just with the system of "Controller-Direct", with a mean of Collected Objects equal to 24.7. For the "Controller-Raycasting" system, eight users out of ten collected 25 objects, one user collected 24 objects, and another collected 23 objects, with a mean of Collected Objects equal to 24.7. As for hand-based systems, the means are 23.6 for "Hand Tracking - Raycasting" and 21.2 for "Hand Tracking - Direct". It's possible to see the grabbed objects for "Hand Tracking - Direct" in Fig. 6.9.

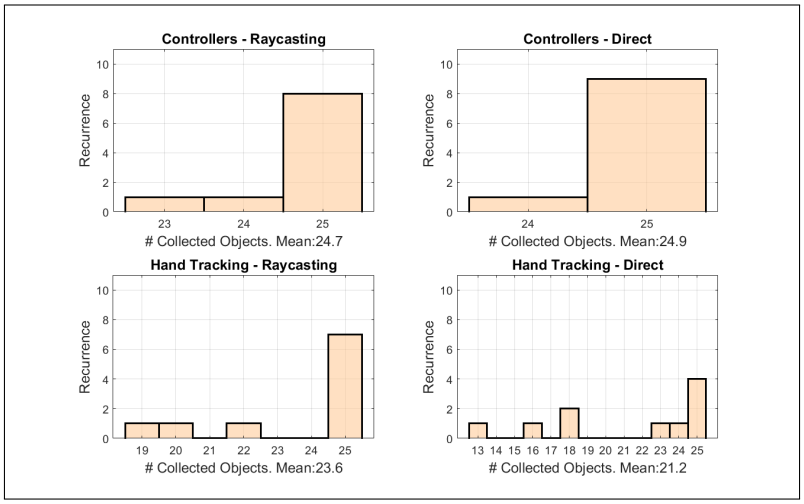


Fig. 6.8: Collected Objects | Scene with the Grab task

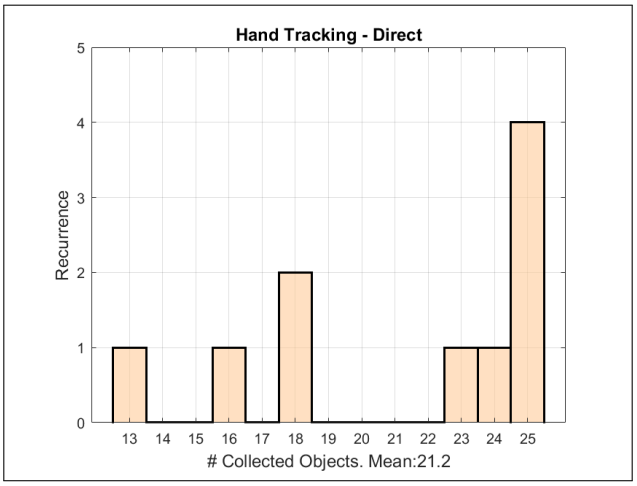


Fig. 6.9: Collected objects | Bare Hands and Direct Interaction

As the last evaluation metric for this task, I collected the Fallen Objects' numbers.

With this metric, I can know with which interaction system it is possible to finish the proposed task without many errors. As evidenced by the figure below Fig. 6.10, ten participants used each interaction system. More precisely, the system "Controller and Raycasting" averaged 5.6 dropped objects, reaching a peak of 25 objects for a single user. For both hand-based methods, the average number of fallen objects is 6.3. Once again, the lowest average is achieved with the "Controller-Direct" interaction system, which, in this case, is equal to 4.7.

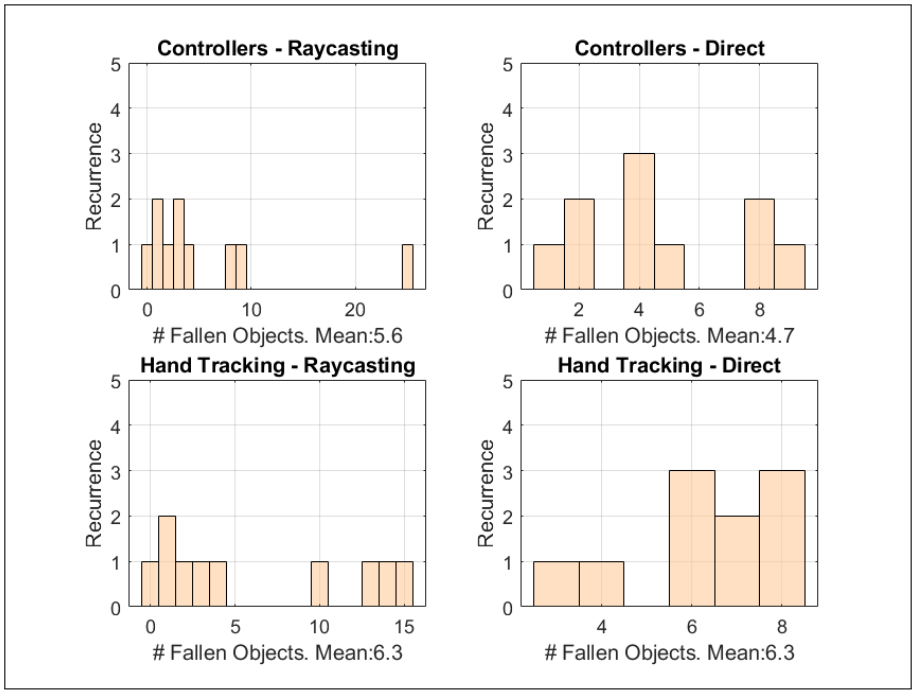


Fig. 6.10: Fallen Objects | Scene with the Grab task

To conclude this initial paragraph, regarding the grab task, from an objective standpoint, the most effective interaction system is the "Controller-Direct". Indeed, for every collected metric, it has recorded the best averages compared to the other three systems, including the average duration.

Results for the Type task

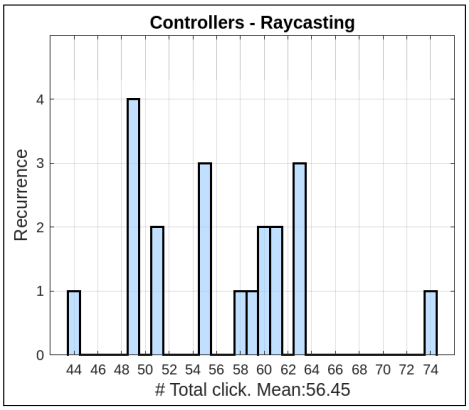
For this second task, I initially intended to present the average duration users took to write eight words using each interaction system. In this case, the valid systems to interact with the keyboard were "Controller - Raycasting", "Hand Tracking - Raycasting", and "Hand Tracking - Direct". The "Controller - Direct" system was impossible to achieve, so I decided that all users should use the "Controller - Raycasting" to do the Type task twice. There were 20 recurrences for this system; indeed, there were 10 for the other systems. The mean duration for each interaction system is exposed below in Table 6.2.

Interaction System	Duration
Controller and Ray-casting	00:01:47
Bare Hands and Ray-casting	00:02:07
Bare Hands and Direct Interaction	00:01:54

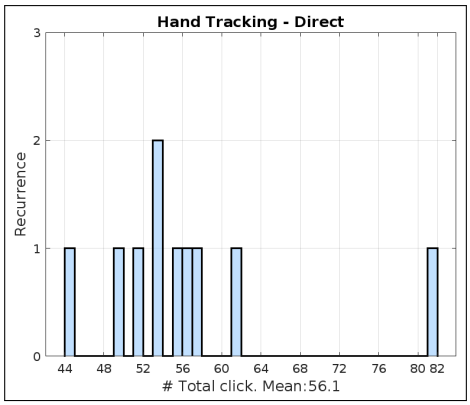
Table 6.2: Average duration of the scene with the type task.

This initial metric reveals that the average duration is shorter with the "Controller - Raycasting" system.

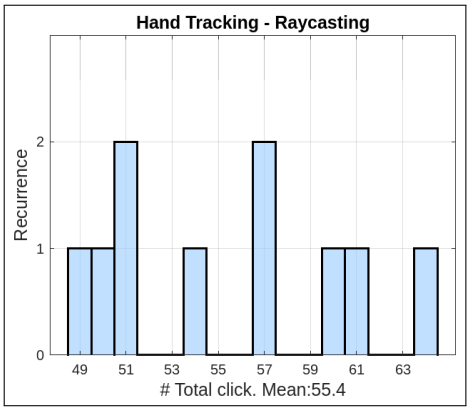
The second metric I chose to evaluate the possible interaction systems for this particular task objectively is the Number of times keys were pressed on the keyboard, regardless of the type of key pressed. The results are exposed on the histograms extracted with MATLAB in Fig. 6.11.



(a) Controllers and Ray-casting



(b) Hands and Direct



(c) Bare Hands and Raycasting

Fig. 6.11: Total clicks on the keyboard for different interaction systems.

I included another metric to objectively evaluate the best interaction system for this type of task — the Number of times users pressed the Backspace button to delete a

typed character. I chose this data point because it can indicate how accurate a given interaction system is. A well-functioning system will result in users making fewer errors and vice versa. The Fig. 6.12 show this data for each interaction system. It is possible to observe that the hand systems have approximately the same value; conversely, the system "Controller-Raycasting" has the lower mean of the three systems. This situation indicates that "Controller-Raycasting" is the best interaction system for this metric.

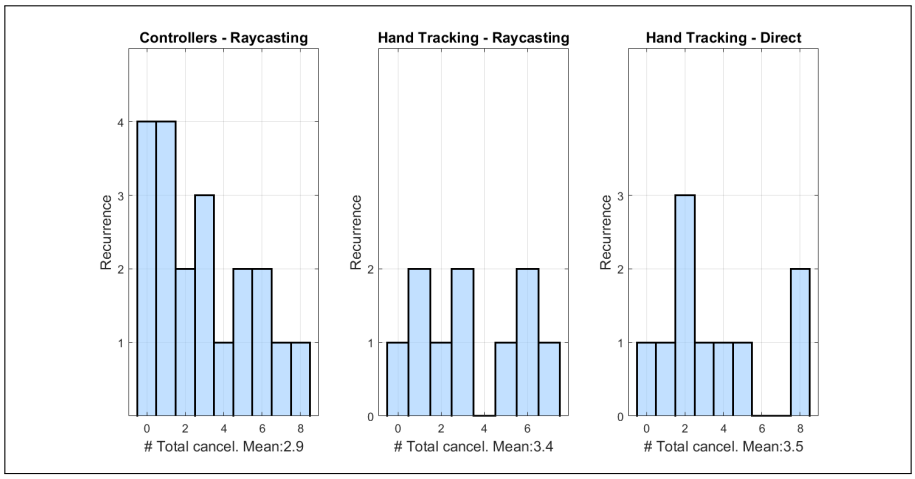


Fig. 6.12: Number of times the Backspace key has been pressed | Type task

I've collected the Number of Right answers to know how many participants have completed the scene, writing all the words correctly or not. It is immediately apparent that with "Controller - Raycasting", every participant, twenty people, has correctly written all the requested words. On the other hand, for the other two systems, just four people didn't write all the answers. It's possible to see these results in Fig. 6.13 below.

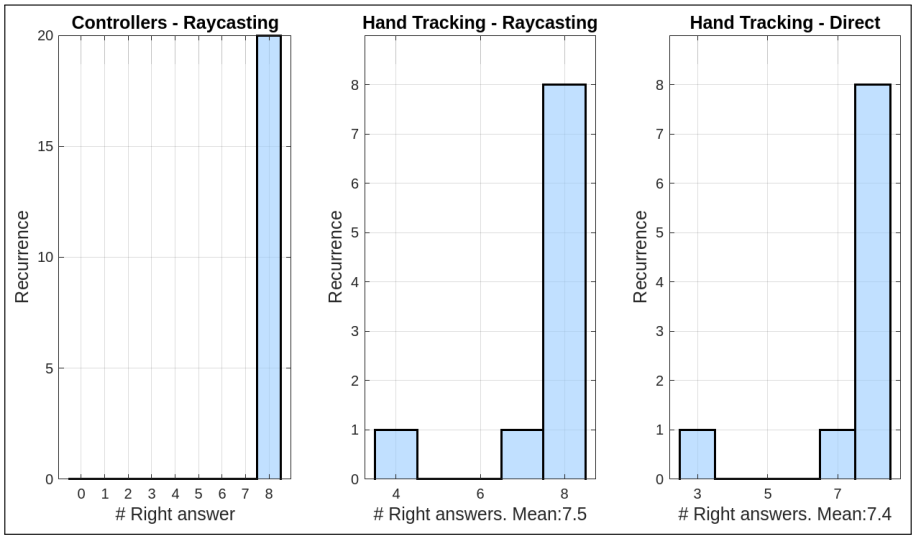


Fig. 6.13: Right Answers | Scene with the Type task

The last metric I decided to collect is the Number of Wrong Answers the participants submitted. The results are shown in Fig. 6.14. It's possible to see that in this case, "Controller - Raycasting" has a lower mean, with 0.45 instead of 0.6 and 0.8 for the hands-based systems.

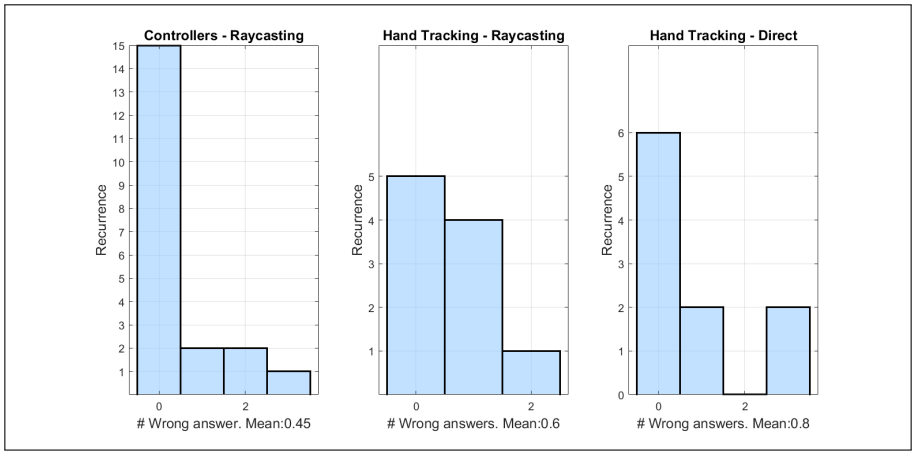


Fig. 6.14: Wrong Answers | Scene with the Type task

In conclusion, for this particular task, it's evident that the system "Hand Tracking - Raycasting" has the lower means regarding button clicks. Another noteworthy observation regarding this metric is that we can trace it back to the number of required clicks. When the averages deviate from this known value, it becomes apparent that, given there are eight words with a total of 41 letters, and considering that pressing the Enter button is necessary to submit the answer every time, 49 button presses are needed to finish the

task. Thus, it's possible to deduce that the delta with the actual presses is either 6 or 5 buttons more. However, it's worth noting that for the other metrics, the most performant system is the "Controller - Raycasting". In particular, the most important thing to notice is that 20 out of 20 completed the task with "Controller - Raycasting" and not with the other two systems.

Results for the Manipulation task

Lastly, let's analyse the data collected during the Manipulation. The mean duration for each interaction system is exposed below in Table 6.3. It's possible to notice that both "Controller and Raycasting" and "Bare Hands and Raycasting" have a mean duration of two minutes. Indeed, "Controller and Raycasting" has the higher mean duration.

Interaction System	Duration
Controller and Ray-casting	00:02:00
Controllers and Direct Interaction	00:02:24
Bare Hands and Ray-casting	00:02:00
Bare Hands and Direct Interaction	00:02:06

Table 6.3: Average duration of the scene with the grab task

As the first metric for this task, like I previously did for the other two, I grabbed the Number of total scaled objects. This is because this scene aimed to scale all the objects in the four tables. So if the Scale object equals four, the scene is completed. It's possible to see the results in the Fig. 6.15. For the system "Controller and Raycasting", just one person didn't scale all the objects, and the mean is 3.9. For the system "Controller and Direct", three people didn't scale all the objects, and the mean is 3.6. For the "Hand Tracking and Raycasting" system, three people didn't scale all the objects, and the mean is 3.5. For the "Hand Tracking and Direct" system, three people didn't scale all the objects, and the mean is 3.4. So, for this metric, "Controller and Raycasting" is the most efficient system of interaction to manipulate objects. The second metric I collected is the Number of Fallen Objects. As for the Grab task, I wanted to know this data to analyse for every single system how it is easy to loose the object during the manipulation task. I wanted to collect the errors that this system led to. As shown in Fig 6.16, the highest means of fallen objects is reached with the "Hand Tracking and Direct" system, with a value of 3.4 fallen objects. The lowest is reached with "Controller and Direct", with a value of 3.4 fallen objects.

For this particular task, the "Controller and Raycasting" is the most efficient system to finish scaling all the objects.

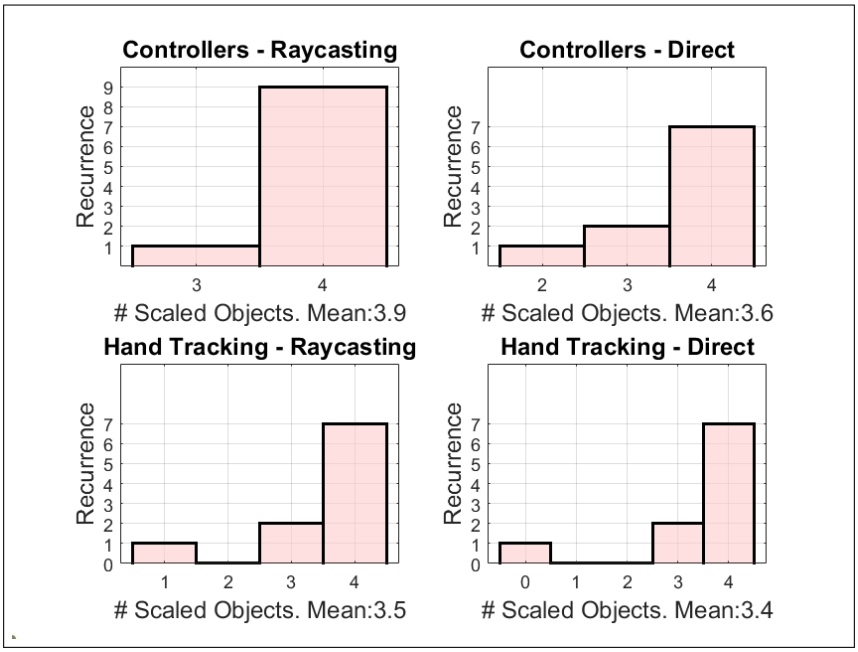


Fig. 6.15: Scaled Objects | Scene with the Manipulation task

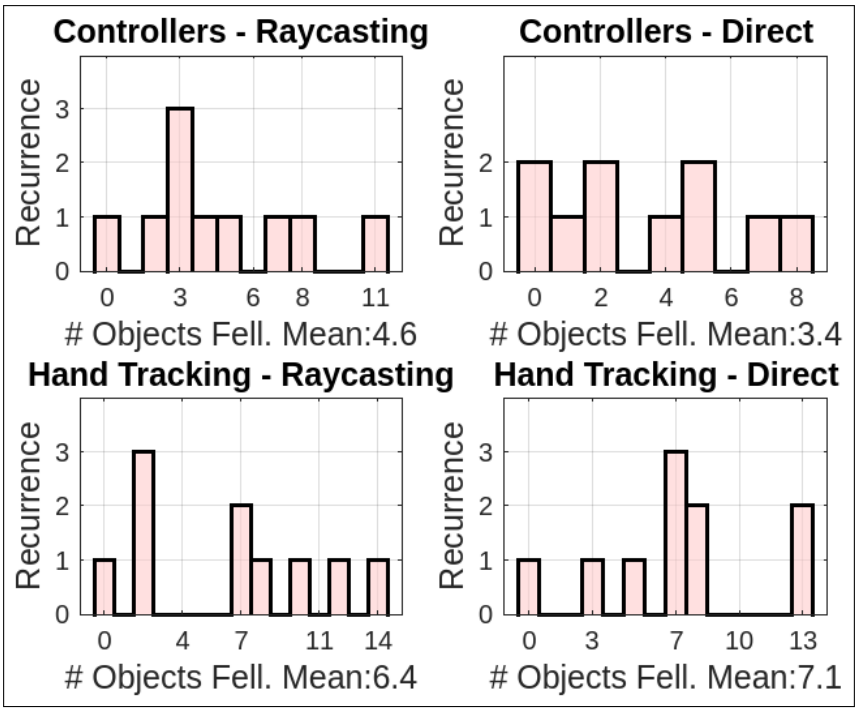


Fig. 6.16: Fallen Objects | Scene with the Manipulation task

6.1.3 Results of the Post-Experience Questionnaires

As discussed in Chapter 5, after the experience, all the participants had to answer the Post-Experience Questionnaires; here, I will explain the results. In the previous paragraph, I analysed the objective point of view to understand the efficiency of a particular interaction system. Instead, this section aims to analyse the subjective point of view of what type of interaction system people like the most.

To start, I asked the people what they preferred between Controllers or Hands. I have obtained the majority of the votes for Controllers. As shown in Fig. 6.17.

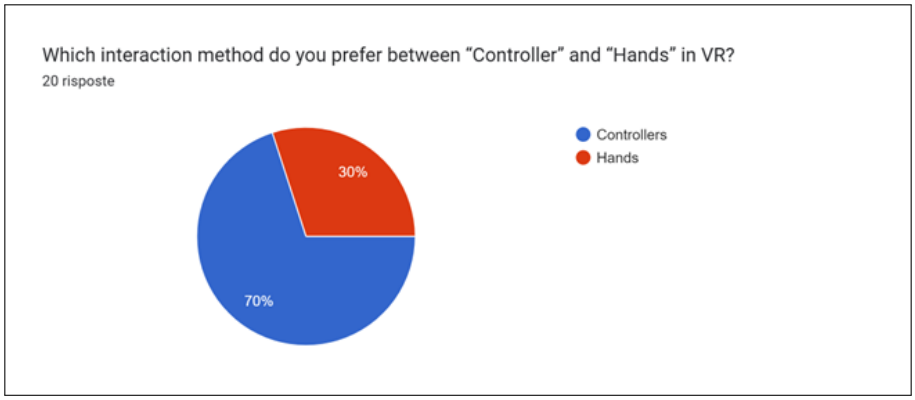


Fig. 6.17: The most preferred interaction method in VR

As the second question, I have asked what is preferred between Raycasting or Direct Interaction. The 55% of participants preferred Direct Interaction instead of Raycasting Fig. 6.18.

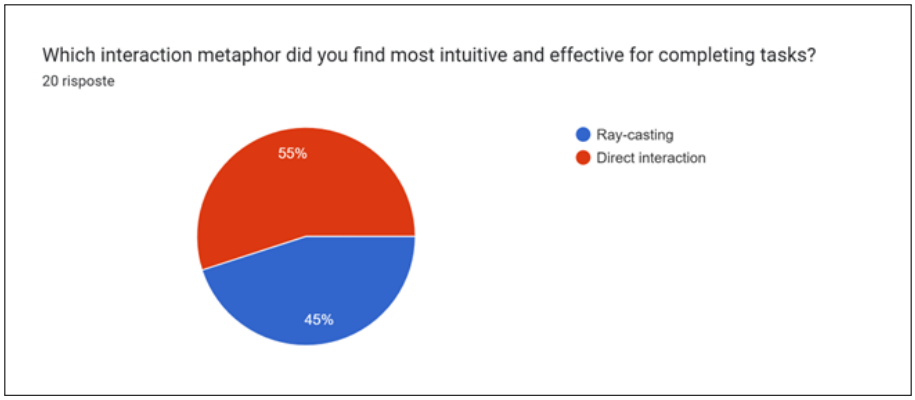
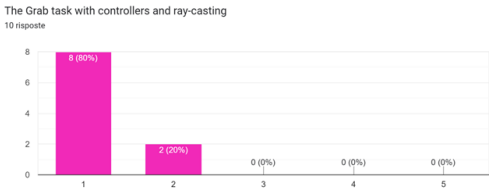


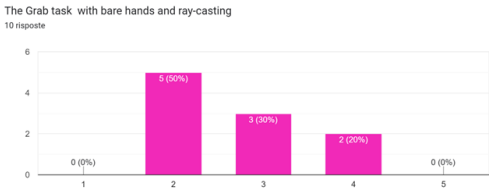
Fig. 6.18: The most preferred interaction metaphor in VR

After that, I asked participants to rate the interaction system from one to five, where one means "Very easy" and five "Extremely difficult". The results are reported below in Fig. 6.19, Fig. 6.20, Fig. 6.21, Fig. 6.22 and Fig. 6.23. With these questions, I aimed to

understand which interaction system was preferred by the participants, focusing on ease of use. In comparing the two interaction systems they used to complete the same task.



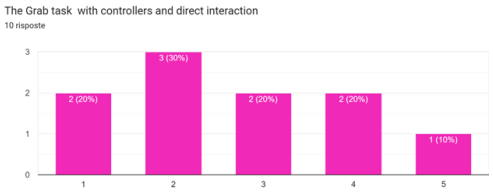
(a) Controller - Raycasting | Grab task



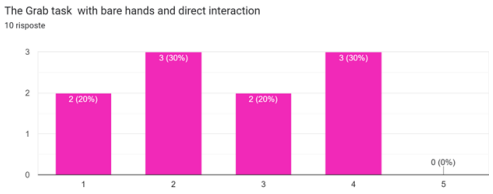
(b) Hands - Raycasting | Grab task

Fig. 6.19: Rating of Interaction Systems for the Grab task | Raycasting Metaphor

About those who tackled the Grab task with "Controller-Raycasting" and "Hand-Raycasting" (Fig. 6.19), 80% rated the first interaction method very easy, while 20% rated it as a 2. As for the second interaction method, 50% rated it as 2, 30% as 3, and 20% as fairly challenging. This result indicates that between the two, the "Controller-Raycasting" interaction method is perceived as the simpler one.



(a) Controller - Direct | Grab task



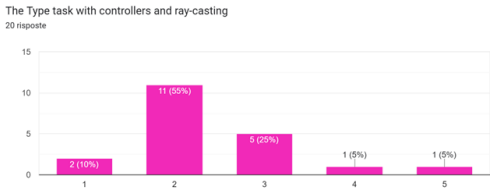
(b) Hands - Direct | Grab task

Fig. 6.20: Rating of Interaction Systems for the Grab task | Direct Interaction

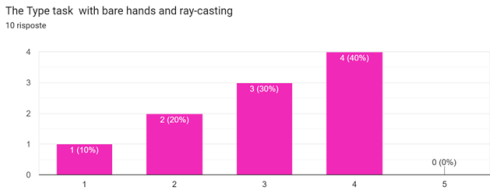
For those who grabbed 25 objects with the direct interaction metaphor (Fig. 6.20), the evaluations were less homogeneous than the previous ones. For the "Controller-Direct" system, 20% rated it as 2, 30% as 3, 20% as 2, 20% as 4, and 10% as 1 (very difficult). Instead, for the "Hand-Direct" system, 20% rated it as 2, 30% as 3, 20% as 2 and 30% as 4. So, no one has rated the "Hand-Direct" system as extremely difficult, but the rating is almost the same between the two systems. Comparing both Raycasting and Direct, I think it is possible to say that the one that was perceived as easiest to use is one more time "Controller-Raycasting".

Regarding the scene with the Type task (Fig. 6.21), all participants used the "Controller-Raycasting" system, and then only 10 used the "Hand-Raycasting" system, and 10 used the "Hand-Direct" system. The "Controller-Raycasting" system was rated by two people as 1, by eleven people as 2, by five people as 3, by one person as 4, and by one person as 5. This result means that only two people found it a problematic interaction system. On the other hand, the "Hand-Raycasting" system was rated as easy to use by one person, 2 by

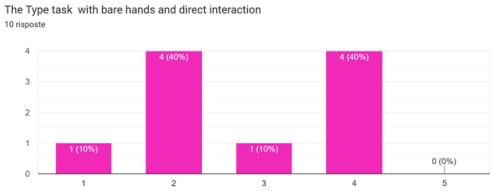
two people, 3 by three people, and 4 by four people. So, in this case, four people did not find this system very user-friendly. Finally, for those who used the "Hand-Direct" system, similarly to the previous one, four people rated it as a difficult system, one person rated it as neither difficult nor complex, four people as a simple system, and one person as a very simple system to use. In conclusion, based on the results obtained from this survey, the easiest-to-use system for this type of task is once again the "Controller-Raycasting".



(a) Controller - Raycasting | Type task



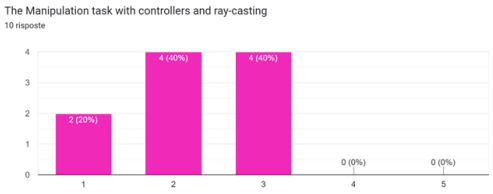
(b) Hands - Raycasting | Type task



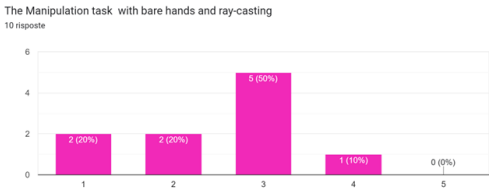
(c) Hands - Direct | Type task

Fig. 6.21: Rating of Interaction Systems for the Type task

Lastly, let's analyse the last task proposed in VR InteracTest: Manipulation. In this case, ten participants also tried this task using Raycasting as a metaphor, while the other ten used Direct Interaction.



(a) Controller - Raycasting | Manipulation

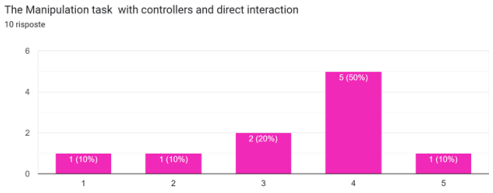


(b) Hands - Raycasting | Manipulation

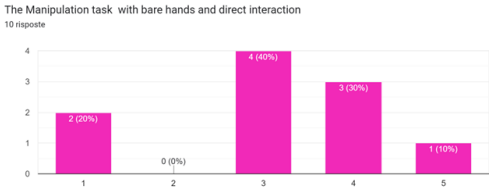
Fig. 6.22: Rating of Interaction Systems for the Manipulation task | Raycasting Metaphor

For those who tried "Controller-Raycasting" and "Hand-Raycasting", the results of the survey are shown in Fig. 6.22. It's possible to notice that 40% rated the first interaction system as neither difficult nor easy, another 40% rated it as easy to use, and 20% rated it very easy to use. As regards the "Hand-Raycasting" interaction system, 20% rated it very

easy, 20% rated it easy to use, 50% rated it as neither difficult nor easy, and one person rated it as a difficult-to-use system. So, in this case, the evaluations of that interaction system for this particular task were barely the same. But it's easy to notice how, once again, the "Controller-Raycasting" system did not receive ratings surpassing the score of 3.



(a) Controller - Direct | Manipulation



(b) Hands - Direct | Manipulation

Fig. 6.23: Rating of Interaction Systems for the Manipulation task | Direct Interaction

For those who tried with "Controller-Direct" and "Hand-Direct", the results of the survey are shown in Fig. 6.23. It's possible to notice that 50% rated the first interaction system as difficult to use, another 1% rated it as extremely difficult to use, 20% as neither difficult nor easy, and finally, 10% rated it as both easy and very easy to use. Regarding the "Hand-Direct" interaction system, 20% rated it as very easy, 40% rated it as neither difficult nor easy, 30% rated it as difficult to use and 10% as extremely difficult. So, comparing these two systems, the one with which users encountered more difficulties is the "Controller-Direct" interaction system.

In conclusion, considering all the tasks and systems evaluated, the "Controller-Raycasting" system emerged as the most positively rated in terms of ease of use and user preference, followed by "Hand-Raycasting". The "Hand-Direct" system received more varied ratings, with some users finding it easy to use and others rating it as more complex. Additionally, the "Controller-Direct" system received less positive ratings, with some users finding it challenging to use.

Results of the System Usability Scale

In the last section of this chapter, I present the results of the SUS related to each interaction system employed in this VR application.

I have explained how the SUS works in Chapter 5. As mentioned earlier, participants must answer ten questions and provide ratings from 1 to 5 after the experience. In summary, the calculations to assess the usability of a given interaction system, in this case in VR, are as follows:

1. For the odd questions, it's necessary to subtract 1 from the given answer (Q1, Q3, Q5, Q7, Q9),

2. For the even questions, it's required to subtract the score assigned to the question from 5 (Q2, Q4, Q6, Q8, Q10),

3. Add up all the modified scores obtained from individual questions,
4. Multiply the total sum by 2.5,
5. Now you have the final score for the SUS participant, which can range from 0 to 100,
6. Calculate the average of all the scores received for various interaction systems, considering that each system has multiple responses.

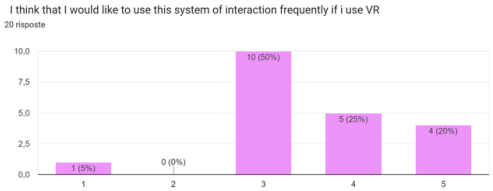
I have done this procedure for each system. The results of the responses given by the participants are shown in Figures 6.24, 6.25, 6.26 and 6.27. Each participant has only responded to the SUS related to their personal interaction methods. As explained in previous sections, 20 individuals evaluated the "Controller-Raycasting" system, while the other systems were assessed by 10 participants each.

For the "Controller-Raycasting" system, the average result of all twenty evaluations was 73.62. For the "Controller-Direct" system, the average result of all ten evaluations was 62.5. For the "Hand-Raycasting" system, the average result of all ten evaluations was 57.75. For the "Hand-Direct" system, the average result of all ten evaluations was 63.75.

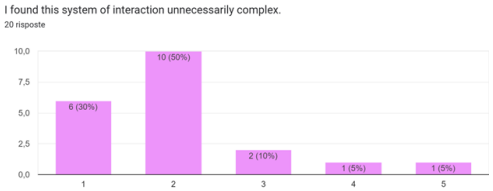
In conclusion, the usability evaluations of the various interaction systems revealed valuable insights into user experiences in virtual reality. The "Controller-Raycasting" system received the highest average score of 73.62 among the participants, indicating a favourable perception of its usability. The "Controller-Direct" system also demonstrated a respectable average result of 62.5, while the "Hand-Raycasting" and "Hand-Direct" systems received average scores of 57.75 and 63.75, respectively.

These findings suggest that the "Controller-Raycasting" and "Controller-Direct" systems exhibit higher levels of usability compared to the "Hand-Raycasting" system. The differences in user experiences between the controller-based and hand-based interaction methods underscore the importance of considering diverse user preferences and needs when designing VR systems. In my opinion, one of the most significant differences between these two interaction methods is the haptic feedback provided by the controller, which is impossible to receive with bare hands. Furthermore, as specified in the previous sections of this chapter, there is heterogeneity in the use of controllers, with half of the participants not accustomed to this interaction system. This aspect suggests that exploring VR worlds using hands might be more challenging.

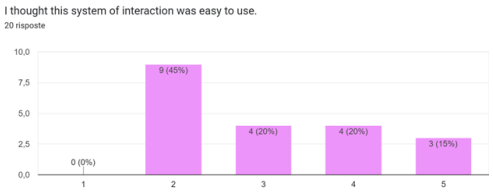
The results contribute to the ongoing discourse on optimising VR interaction methods, providing valuable guidance for future developments in immersive technologies. Acknowledging the diverse user preferences and adapting interaction systems to accommodate varying comfort levels and preferences is essential.



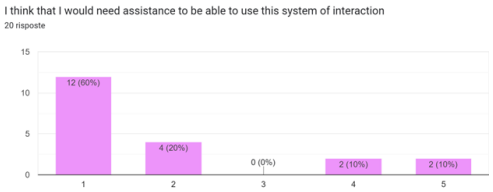
(a) Question 3



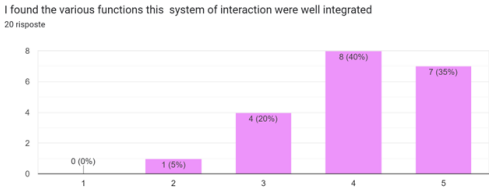
(b) Question 2



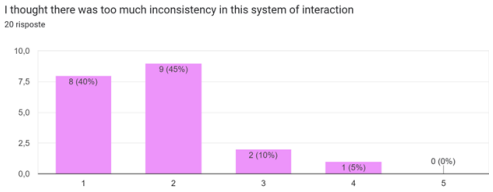
(c) Question 3



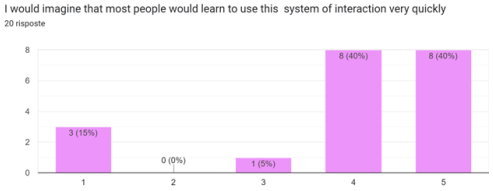
(d) Question 4



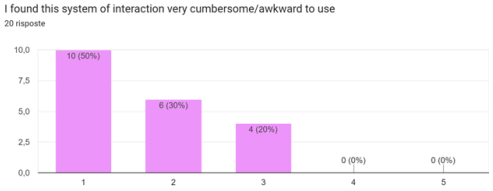
(e) Question 5



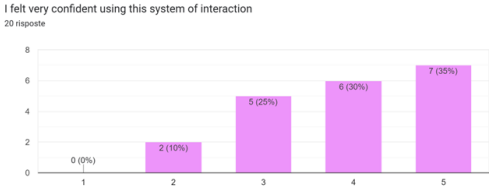
(f) Question 6



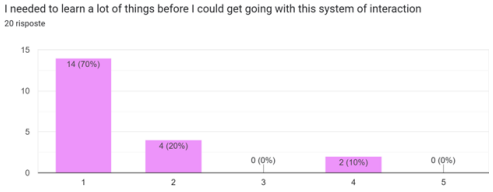
(g) Question 7



(h) Question 8

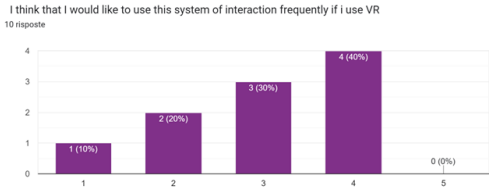


(i) Question 9

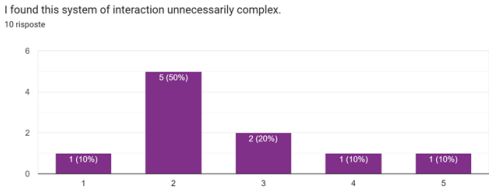


(j) Question 10

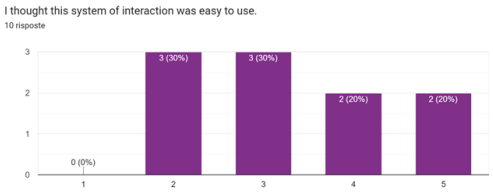
Fig. 6.24: System Usability Scale for “Controller-Raycasting” interaction system



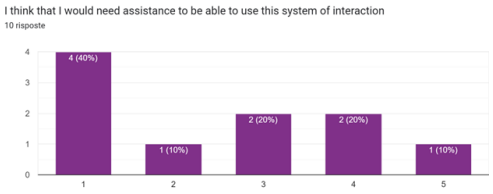
(a) Question 3



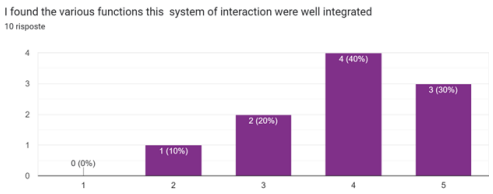
(b) Question 2



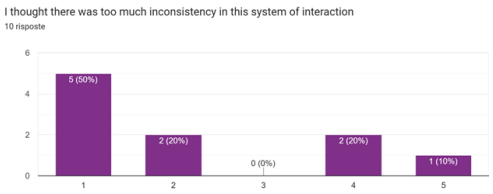
(c) Question 3



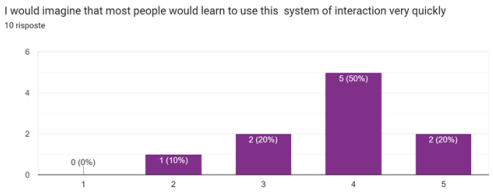
(d) Question 4



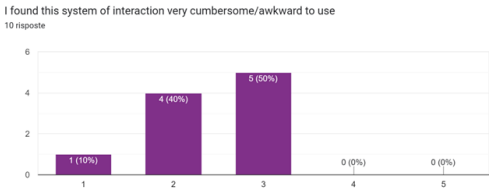
(e) Question 5



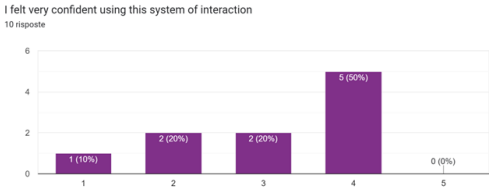
(f) Question 6



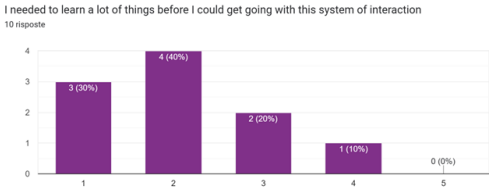
(g) Question 7



(h) Question 8

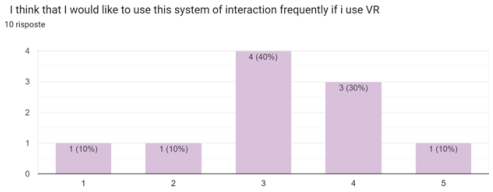


(i) Question 9

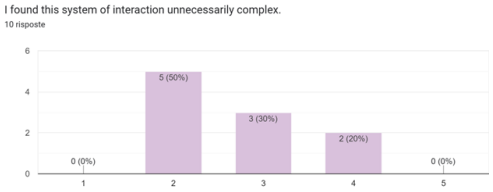


(j) Question 10

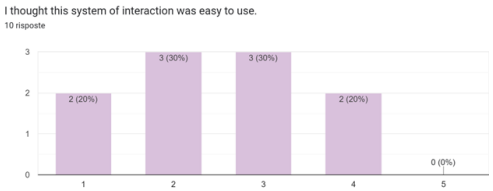
Fig. 6.25: System Usability Scale for “Controller-Direct” interaction system



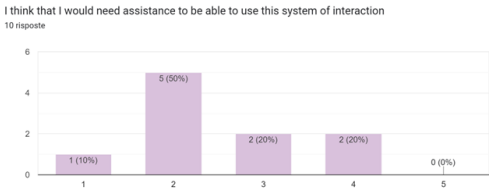
(a) Question 3



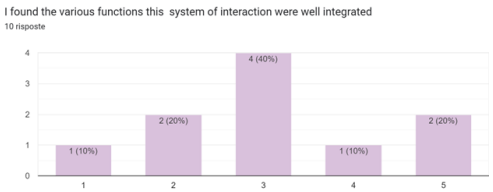
(b) Question 2



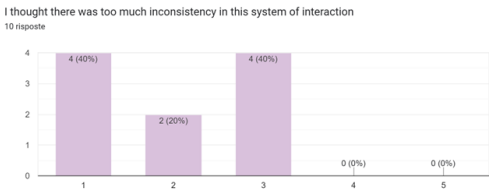
(c) Question 3



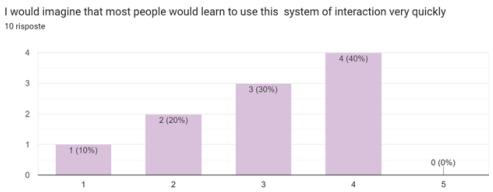
(d) Question 4



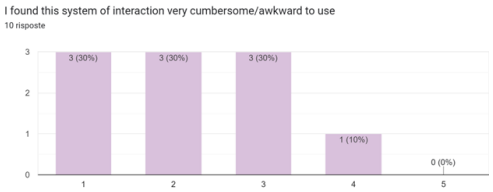
(e) Question 5



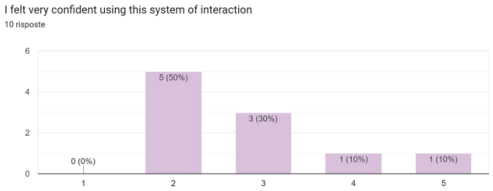
(f) Question 6



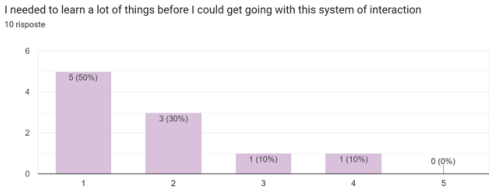
(g) Question 7



(h) Question 8

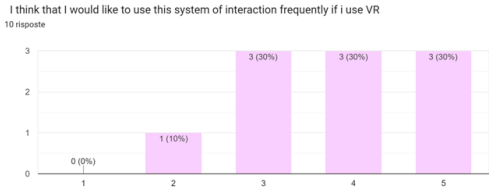


(i) Question 9

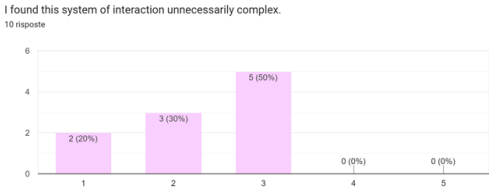


(j) Question 10

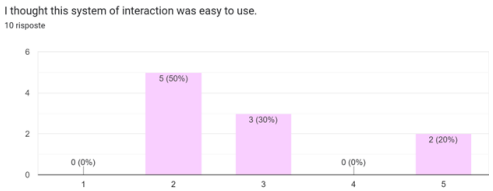
Fig. 6.26: System Usability Scale for “Hand-Raycasting” interaction system



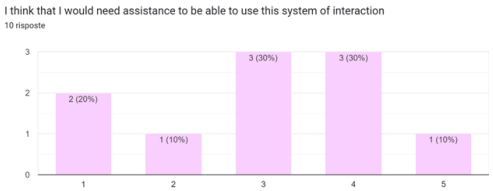
(a) Question 3



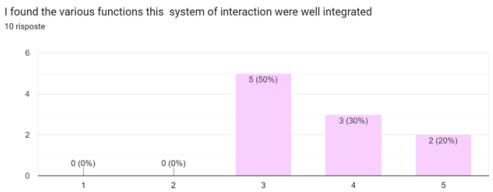
(b) Question 2



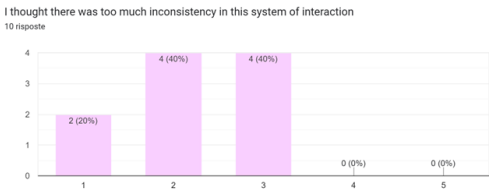
(c) Question 3



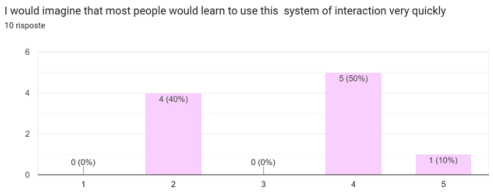
(d) Question 4



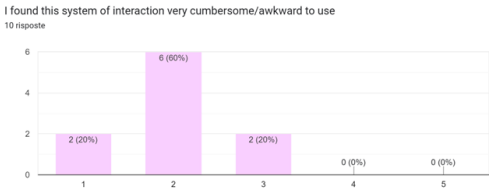
(e) Question 5



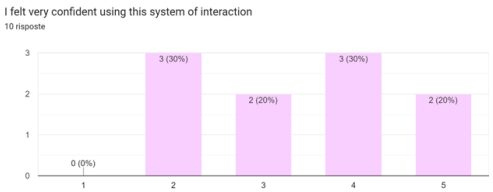
(f) Question 6



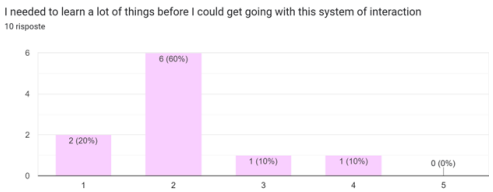
(g) Question 7



(h) Question 8



(i) Question 9



(j) Question 10

Fig. 6.27: System Usability Scale for “Hand-Raycasting” interaction system

Chapter 7

Conclusion

This work introduces my application, VR InteracTest, to bring an innovative solution to the market of Virtual Reality. In fact, in recent years, the VR world has had a rapid increase in the development of the products. The increase in VR applications is due to the Unity or Unreal Engine platforms, which are accessible and partly allow people to create whatever they want. For this reason, the field of research has also increased.

This thesis work began with detailed state-of-the-art research. From this study, it's possible to notice that, nowadays, there is a lot of research on new techniques of interactions. A standard evaluation method needs to be added to all of these studies. VR InteracTest is an application that tries to delete this lack. The development of VR applications, particularly VR InteracTest, was driven by the need to facilitate the work of developers and designers by providing a platform for comparing interaction metaphors and methods.

I have chosen three of the most common tasks from the state-of-the-art study: grabbing objects, typing on a keyboard and manipulating objects. Regarding the interaction systems, I have chosen two methods, namely controllers and hand-tracking, and two different metaphors, i.e., raycasting and virtual hands. The primary objective is to evaluate the system interaction effectiveness and user preferences.

To test the different interaction systems, VR InteracTest can save data from the user experience. I decided to use some metrics to evaluate the effectiveness of the different systems. The mean time spent completing the task is the standard metric for all the tasks. More specifically, for the first task, I also saved the number of collected objects, the number of fallen objects, the first time a single object is selected, and the time the object is thrown in the specific bin. For the type task, I decided to collect the number of total clicks on every button of the keyboard, the number of times that the Backspace is pressed, the time the user starts answering each question, and the time it has finished. Finally, for the manipulation task, I collected the number of times the objects fell, the in which the user started scaling, and we they ended.

The "Controller-Raycasting" system consistently emerged as the most performing across various tasks, showcasing higher completion rates and lower error instances. "Hand-Raycasting" followed closely, while "Controller-Direct" and "Hand-Direct" exhibited more diverse user experiences, with some users finding them challenging. Most participants

preferred controllers over bare hands, indicating that the tangible feedback provided by controllers plays a crucial role in user satisfaction. The preference for "Direct Interaction" over "Raycasting" suggests that users find direct manipulation more intuitive and effective. The System Usability Scale (SUS) results reinforced the positive user experiences with the "Controller-Raycasting" system, with a notably higher average score than other systems. The findings underscore the importance of considering diverse user preferences when designing VR systems, especially acknowledging the impact of haptic feedback.

The thesis highlights the significance of designing VR interaction systems that align with user expectations and preferences. Integrating haptic feedback, especially in hand controllers, can contribute to improved user experiences.

Future VR applications should undergo thorough usability testing, considering both objective performance metrics and subjective user feedback. Iterative design processes, informed by user preferences, can lead to more user-friendly and effective VR systems. Acknowledging the diversity in user experience preferences is crucial for creating inclusive VR applications that cater to a broad audience. Future research could explore the factors influencing individual preferences in VR interaction. In conclusion, this research provides valuable insights into the dynamics of VR interaction systems, emphasising the importance of considering both objective performance metrics and subjective user preferences. The outcomes contribute to the ongoing discourse on optimising VR applications, paving the way for more user-centric and immersive virtual experiences.

VR InteracTest will be available to the public. The choice to make it downloadable from GitHub and usable by most will allow for an increasingly expansive platform in the future. Adding more tasks other metaphors, and implementing new input devices will be possible. Through VR InteracTest, everyone will have the opportunity to compare their new interaction system with those already existing, to put an objectively tested interaction system on the market.

Bibliography

[1] V. Italia, "Il sensorama: Realtà virtuale." https://www.vr-italia.org/il-sensorama-realta-virtuale/, Novembre 15, 2023.

[2] Ayfel, "Mrtk-keyboard." https://github.com/Ayfel/MRTK-Keyboard.

[3] C. M. Craig, J. Stafford, A. Egorova, C. McCabe, and M. Matthews, "Can we use the oculus quest vr headset and controllers to reliably assess balance stability?," *Diagnostics*, vol. 12, no. 6, p. 1409, 2022.

[4] J. Hertel, S. Karaosmanoglu, S. Schmidt, J. Bräker, M. Semmann, and F. Steinicke, "A taxonomy of interaction techniques for immersive augmented reality based on an iterative literature review," in *2021 IEEE international symposium on mixed and augmented reality (ISMAR)*, pp. 431–440, IEEE, 2021.

[5] E. Bouzbib, G. Bailly, S. Haliyo, and P. Frey, ""can i touch this?": Survey of virtual reality interactions via haptic solutions: Revue de littérature des interactions en réalité virtuelle par le biais de solutions haptiques," in *Proceedings of the 32nd Conference on l'Interaction Homme-Machine*, pp. 1–16, 2021.

[6] M. Höll, M. Oberweger, C. Arth, and V. Lepetit, "Efficient physics-based implementation for realistic hand-object interaction in virtual reality," in *2018 IEEE conference on virtual reality and 3D user interfaces (VR)*, pp. 175–182, IEEE, 2018.

[7] S. Oprea, P. Martinez-Gonzalez, A. Garcia-Garcia, J. A. Castro-Vargas, S. Orts-Escolano, and J. Garcia-Rodriguez, "A visually realistic grasping system for object manipulation and interaction in virtual reality environments," *Computers & Graphics*, vol. 83, pp. 77–86, 2019.

[8] J.-N. Voigt-Antons, T. Kojic, D. Ali, and S. Möller, "Influence of hand tracking as a way of interaction in virtual reality on user experience," in *2020 Twelfth International Conference on Quality of Multimedia Experience (QoMEX)*, pp. 1–4, IEEE, 2020.

[9] S. Esmaeili, B. Benda, and E. D. Ragan, "Detection of scaled hand interactions in virtual reality: The effects of motion direction and task complexity," in *2020 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, pp. 453–462, IEEE, 2020.

[10] S. Han, B. Liu, R. Cabezas, C. D. Twigg, P. Zhang, J. Petkau, T.-H. Yu, C.-J. Tai, M. Akbay, Z. Wang, *et al.*, "Megatrack: monochrome egocentric articulated hand-tracking for virtual reality," *ACM Transactions on Graphics (ToG)*, vol. 39, no. 4, pp. 87–1, 2020.

[11] M. Meier, P. Streli, A. Fender, and C. Holz, "Tapid: Rapid touch interaction in virtual reality using wearable sensing," in *2021 IEEE Virtual Reality and 3D User Interfaces (VR)*, pp. 519–528, IEEE, 2021.

[12] X. Xu, X. Pan, D. Kilroy, A. Kumar, E. Mangina, and A. G. Campbell, "Using hmd-based hand tracking virtual reality in canine anatomy summative assessment: a user study," in *2022 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pp. 287–296, IEEE, 2022.

[13] S. Pei, A. Chen, J. Lee, and Y. Zhang, "Hand interfaces: Using hands to imitate objects in ar/vr for expressive interactions," in *Proceedings of the 2022 CHI conference on human factors in computing systems*, pp. 1–16, 2022.

[14] W.-J. Tseng, S. Huron, E. Lecolinet, and J. Gugenheimer, "Fingermapper: Mapping finger motions onto virtual arms to enable safe virtual reality interaction in confined spaces," in *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, pp. 1–14, 2023.

[15] R. Alkemade, F. J. Verbeek, and S. G. Lukosch, "On the efficiency of a vr hand gesture-based interface for 3d object manipulations in conceptual design," *International Journal of Human–Computer Interaction*, vol. 33, no. 11, pp. 882–901, 2017.

[16] C. R. Austin, B. Ens, K. A. Satriadi, and B. Jenny, "Elicitation study investigating hand and foot gesture interaction for immersive maps in augmented reality," *Cartography and Geographic Information Science*, vol. 47, no. 3, pp. 214–228, 2020.

[17] N. Capece, U. Erra, D. Malandrino, M. M. North, and M. Gruosso, "Evaluation of virtual reality interaction techniques: the case of 3d graph," *arXiv preprint arXiv:2302.05660*, 2023.

[18] J. Wang, F. Mueller, F. Bernard, S. Sorli, O. Sotnychenko, N. Qian, M. A. Otaduy, D. Casas, and C. Theobalt, "Rgb2hands: real-time tracking of 3d hand interactions from monocular rgb video," *ACM Transactions on Graphics (ToG)*, vol. 39, no. 6, pp. 1–16, 2020.

[19] B. Spittle, M. Frutos-Pascual, C. Creed, and I. Williams, "A review of interaction techniques for immersive environments," *IEEE Transactions on Visualization and Computer Graphics*, 2022.

[20] N. Capece, G. Manfredi, V. Macellaro, and P. Carratù, "An easy hand gesture recognition system for xr-based collaborative purposes," in *2022 IEEE International Conference on Metrology for Extended Reality, Artificial Intelligence and Neural Engineering (MetroXRAINE)*, pp. 121–126, IEEE, 2022.

[21] N. Capece, U. Erra, G. Manfredi, and R. Di Bello, "Boidvr: An agent simulation environment based on freehand and virtual reality," *IEEE Computer Graphics and Applications*, vol. 42, no. 6, pp. 107–115, 2022.

[22] M. Z. Iqbal, E. Mangina, and A. G. Campbell, "Exploring the real-time touchless hand interaction and intelligent agents in augmented reality learning applications," in *2021 7th International Conference of the Immersive Learning Research Network (iLRN)*, pp. 1–8, IEEE, 2021.

[23] C. Khundam, V. Vorachart, P. Preeyawongsakul, W. Hosap, and F. Noël, "A comparative study of interaction time and usability of using controllers and hand tracking in virtual reality training," in *Informatics*, p. 60, MDPI, 2021.

[24] M. Noghabaei and K. Han, "Object manipulation in immersive virtual environments: Hand motion tracking technology and snap-to-fit function," *Automation in Construction*, vol. 124, p. 103594, 2021.

[25] D. Abdlkarim, M. Di Luca, P. Aves, S.-H. Yeo, R. C. Miall, P. Holland, and J. M. Galea, "A methodological framework to assess the accuracy of virtual reality hand-tracking systems: A case study with the oculus quest 2," *BioRxiv*, pp. 2022–02, 2022.

[26] D. Mendes, F. M. Caputo, A. Giachetti, A. Ferreira, and J. Jorge, "A survey on 3d virtual object manipulation: From the desktop to immersive virtual environments," in *Computer graphics forum*, vol. 38, pp. 21–45, Wiley Online Library, 2019.

[27] J. Schjerlund, K. Hornbæk, and J. Bergström, "Ovrlap: Perceiving multiple locations simultaneously to improve interaction in vr," in *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, pp. 1–13, 2022.

[28] C. Peng, Y. Dong, and L. Cao, "Freehand interaction in virtual reality: bimanual gestures for cross-workspace interaction," in *Proceedings of the 27th ACM Symposium on Virtual Reality Software and Technology*, pp. 1–2, 2021.

[29] X. Zhang, W. He, M. Billinghurst, L. Yang, S. Feng, and D. Liu, "Design and evaluation of bare-hand interaction for precise manipulation of distant objects in ar," *International Journal of Human–Computer Interaction*, pp. 1–15, 2022.

[30] A. Cheymol, G. Fouché, L. Gramoli, Y. Hirao, E. Hummel, M. Mavromatis, Y. Moullec, F. Argelaguet, and F. Nouviale, "The rubber slider metaphor: Visualisation of temporal and geolocated data," in *2022 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*, pp. 904–905, IEEE, 2022.

[31] H. Ro, J.-H. Byun, Y. J. Park, N. K. Lee, and T.-D. Han, "Ar pointer: Advanced ray-casting interface using laser pointer metaphor for object manipulation in 3d augmented reality environment," *Applied Sciences*, vol. 9, no. 15, p. 3078, 2019.

[32] S. M. Nizam, R. Z. Abidin, N. C. Hashim, M. C. Lam, H. Arshad, and N. Majid, "A review of multimodal interaction technique in augmented reality environment," *Int. J. Adv. Sci. Eng. Inf. Technol*, vol. 8, no. 4-2, p. 1460, 2018.

[33] V. Holzwarth, J. Gisler, C. Hirt, and A. Kunz, "Comparing the accuracy and precision of steamvr tracking 2.0 and oculus quest 2 in a room scale setup," in *2021 the 5th International conference on virtual and augmented reality simulations*, pp. 42–46, 2021.

[34] C. Khundam, "First person movement control with palm normal and hand gesture interaction in virtual reality," in *2015 12th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, pp. 325–330, IEEE, 2015.

[35] C.-Y. Lee, W.-A. Hsieh, D. Brickler, S. V. Babu, and J.-H. Chuang, "Design and empirical evaluation of a novel near-field interaction metaphor on distant object manipulation in vr," in *Proceedings of the 2021 ACM Symposium on Spatial User Interaction*, pp. 1–11, 2021.

[36] R. Serrano, P. Morillo, S. Casas, and C. Cruz-Neira, "An empirical evaluation of two natural hand interaction systems in augmented reality," *Multimedia Tools and Applications*, vol. 81, no. 22, pp. 31657–31683, 2022.

[37] M. Krichenbauer, G. Yamamoto, T. Taketom, C. Sandor, and H. Kato, "Augmented reality versus virtual reality for 3d object manipulation," *IEEE transactions on visualization and computer graphics*, vol. 24, no. 2, pp. 1038–1048, 2017.

[38] A. G. Sutcliffe, C. Poullis, A. Gregoriades, I. Katsouri, A. Tzanavari, and K. Herakleous, "Reflecting on the design process for virtual reality applications," *International Journal of Human–Computer Interaction*, vol. 35, no. 2, pp. 168–179, 2019.

[39] J. Brooke, "Sus: a "quick and dirty'usability," *Usability evaluation in industry*, vol. 189, no. 3, pp. 189–194, 1996.

[40] P. Wang, X. Bai, M. Billinghurst, S. Zhang, D. Han, M. Sun, Z. Wang, H. Lv, and S. Han, "Haptic feedback helps me? a vr-sar remote collaborative system with tangible interaction," *International Journal of Human–Computer Interaction*, vol. 36, no. 13, pp. 1242–1257, 2020.