FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Improving Absolute Inputs for Interactive Surfaces in VR

Diogo Guimarães do Rosário

Mestrado em Engenharia Informática e Computação

Supervisor: Prof. Daniel Mendes

Second Supervisor: Prof. Teresa Matos

July 4, 2023

Improving Absolute Inputs for Interactive Surfaces in VR

Diogo Guimarães do Rosário

Mestrado em Engenharia Informática e Computação

Approved in oral examination by the committee:

President: Prof. Alexandre Valle
Referee: Prof. Paulo Dias
Referee: Prof. Daniel Mendes

July 4, 2023

Resumo

O principal objetivo da Realidade Virtual (VR) é criar uma experiência simulada e imersiva que transporte os usuários para um ambiente gerado por computador, geralmente usando dispositivos montados na cabeça (como HMDs) ou outros equipamentos sensoriais. A Realidade Virtual tem como objetivo fazer com que os usuários se sintam como se estivessem fisicamente presentes em um mundo virtual, interagindo e vivenciando-o como se fosse real.

No entanto, vários aspetos da Realidade Virtual ainda podem melhorar, de maneira a fornecer aos utilizadores uma experiência ainda mais rica. Um exemplo disto são tarefas que requerem a seleção precisa de alvos, usando superfícies interativas como dispositivo de input. Estas tarefas são especificamente difíceis visto que o utilizador não consegue ver a posição das suas mãos, relativamente à superfície interativa, tendo assim que utilizar os seus sentidos de propriocepção. Para além disso, outros fatores também contribuem para a elevada dificuldade destas tarefas como, por exemplo, latência e o limitado campo de visão dos óculos de realidade virtual, que fazem com que seja ainda mais difícil atingir alvos que estejam mais longe. Embora existam alguns trabalhos de pesquisa nesta área, a maioria das soluções recorre ao rastreamento de todos os dedos usando marcadores óticos.

Visto que não queremos recorrer ao uso de marcadores óticos em todos os dedos, pretendemos desenvolver uma técnica que nos permita selecionar múltiplos alvos de uma forma precisa em Realidade Virtual. Esperamos que esta técnica melhore significativamente a performance de seleção absoluta de alvos em RV e consequentemente permita um uso mais abundante de superfícies interativas como dispositivo de input em Realidade Virtual.

ACM Classification: Human-centered computing → Human computer interaction (HCI) → Interaction devices → Touch screens

Keywords: Virtual Reality, Interactive Surfaces

Abstract

The main goal of Virtual Reality (VR) is to create a simulated and immersive experience that transports users to a computer-generated environment, usually using head-mounted displays (HMDs) or other sensory equipment. VR aims to make users feel as if they are physically present in a virtual world, interacting with and experiencing it as if it were real.

However, several aspects of VR can still be improved upon, in order to provide users with an even more engaging and rich experience. An example of this is tasks that rely on accurately selecting targets while using interactive surfaces as input devices. These tasks are especially hard because the user cannot see the position of their hands in relation to the interactive surface used for input, having to rely on proprioception. Besides that, other factors also contribute to the difficulty of this task such as latency and a limited field of view on Head-Mounted Displays, making it even harder to select targets that are far away. Even though some exploration has been done in this area, most solutions to these problems resort to tracking every finger with optical markers for precise tracking.

Since we do not want to use optical markers on every finger, we aim to develop a technique that allows us to select multiple targets in Virtual Reality accurately. We hope that this technique significantly improves absolute target selection performance in VR and consequently allows for more common use of interactive surfaces in Virtual Reality.

ACM Classification: Human-centered computing → Human computer interaction (HCI) → Interaction devices → Touch screens

Keywords: Virtual Reality, Interactive Surfaces

Agradecimentos

To my supervisors, Daniel Mendes and Teresa Matos, for all the guidance during the project and their time in making sure I could succeed as best as I could in this project.

To my parents, aunt, and grandparents for their unconditional support in every moment. To my close friends Henrique and Davide for all their help during this whole process and for spending many days together at the laboratory working closely. To my friends Mike and Bruno for helping me relieve stress during these months of hard work.

To Diego, Amaia, Aritz, Reda, Mathias, Antoine, some of the best people I met during my exchange period, that were always present for me during some of the best months of my life.

To all my other friends, Fontão, Lucas, João Costa, João Alexandre, Ivo, Telmo, Tiago, César and Ana for being there when I needed it and supporting me during this process and the many years of my course.

To all the other people I met during my exchange, that also helped me become a better person.

Diogo Rosário

"The journey of a thousand miles must begin with a single step."

Lao Tzu

Contents

List of Figures

List of Tables

Listings

Abreviaturas e Símbolos

AI	Artificial Intelligence
CSV	Comma-separated values
HMD	Head-mounted display
ID	Identifier
POV	Point of view
RV	Realidade Virtual
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
UI	User interface
VE	Virtual Environment
VR	Virtual Reality

Chapter 1

Introduction

The use of virtual reality has been growing rapidly during the last decade, primarily due to the advancement of technology that enables VR. This technology provides us with a unique way to immerse users in environments that otherwise would not be possible and to create experiences that would be impossible to make in the real world. However, many aspects of the VR experience can still be improved to amplify the user experience further.

Interaction is critical in virtual reality. It is the way to connect to the virtual world, feeling present. However, interactivity is not always easy to achieve since there are a lot of factors to make it successful. Actions in Virtual Reality must be intuitive and meaningful, not to break immersion. Input devices are also limited in their actions, each having advantages.

1.1 Motivation and Problem

Interactive surfaces are being used in our everyday lives for multiple different contexts. These surfaces bring with them several advantages, with the main one being the fact that they provide an intuitive and natural interaction. Other advantages include supporting single and multi-touch actions allowing for more complex operations while combining great precision on touch. They also allow for making custom User Interfaces so that multiple buttons can be created for different actions. However, combining these surfaces with Virtual Reality still needs to be explored more.

Unfortunately, implementing interactive surfaces with Virtual Reality is not a trivial task. Several problems need to be addressed to make them usable. One of these problems is not knowing where the hands/fingers are relative to the device. In an ideal scenario, the user should feel the touch on the interactive surface at the same time the virtual touch event occurs, which means that the relative positions from the fingers/hands to the device in the real world should match with the ones in the virtual world. However, this is hard to solve since standard tracking solutions are subject to slight calibration errors. Without feedback, users could give the wrong inputs, leading to unwanted actions.

Effectively combining interactive surfaces with VR could benefit applications heavily since they would allow for more ways to interact with VR content, such as sliders or widgets.

1.2 Goals

This project focuses on finding new methods to implement and incorporate interactive surfaces in Virtual Reality, mainly for absolute target selection. As such, these methods should enhance the Virtual Reality experience by combining the advantages of these surfaces while minimizing their disadvantages when combined with using an HMD. We will aim for our technique to allow multi-touch events to simultaneously use both hands for input.

To figure out if we achieved our goals with the developed solution, we will try to answer the following research questions:

- Can our technique accurately estimate the contact points of the fingers with the surface?
- When the contact point estimation is wrong, can our technique still allow for successful inputs?
- How small can the targets be while maintaining an acceptable precision on target?

1.3 Document Structure

This document is divided into six chapters.

The first chapter 1 refers to the introduction to the document.

In chapter 2, we will analyze and discuss previous work on interactive surfaces in Virtual Reality. We will look into the use of touchscreens in Virtual Reality in section 2.1, hand tracking and how it is done in Virtual Reality in section 2.2, and absolute target selection in Virtual Reality in section 2.3. In section 2.4, we present the main findings of the analysis of the previous work and how they led us to our proposed solution.

In chapter 3, we will present our proposed solution and how we plan to develop it. Chapter 4 presents how we implemented our proposed solution and the choices made along the way. Chapter 5 will focus on the user testing conducted to evaluate our solution and the results of these same tests. Furthermore, we will take conclusions based on the data collected.

Finally, in chapter 6, we will present a quick summary of the document.

Chapter 2

Related Work

Interactive surfaces as an input method in Virtual Reality is still a largely unexplored area. Some work has been done in the area by using the capabilities of today's mobile phones and tablets. However, most focus on these devices' sensors to interact with the virtual world and not as much on the touchscreen capabilities.

2.1 Touchscreens in Virtual Reality

Touchscreens provide users with a more natural way to interact with Virtual Reality environments. This way of interaction enhances the sense of immersion, making it feel more like a real-world experience. One of the very first approaches that was explored was based on the placement of a touchscreen attached to the Head-Mounted display (HMD) to allow for direct inputs [11]. Although successful, this approach was very limited in that no feedback was given to the user, and the actions they could perform were very basic, such as dragging gestures on the touchscreen. Facetouch [9] is also based on the same principle, where a multi-touch surface was placed on the back of the HMD as shown in figure 2.1. By relying on proprioception, users could select targets in their point of view (POV) with visual feedback when their fingers touch the surface. This touchscreen was deemed viable for several actions, such as text entry and game controllers. However, some limitations of this prototype were the added weight on the user's head and the arms fatigue caused by the interaction being done at head length on the HMD. Mine et al. also explored the advantages of using proprioception while immersed in Virtual Environments by using hand controllers equipped with touchscreens for precise 2D inputs [15]. In contrast to Facetouch [9], the touchscreens were placed on hand controllers but used for the same simple actions, such as button clicks.

Several others iterated on this design by combining touchscreen inputs with 3D mid-air barehand gestures, as is the case with TabletInVR [23]. This work exploits the advantages of a tracked tablet while immersed in a Virtual Environment (VE). An interaction vocabulary was developed, mapping actions to gestures, such as dragging. The tablet used was tracked and rendered in the VE. Some actions possible with this app are rotating, scaling, and translating different 3D objects.

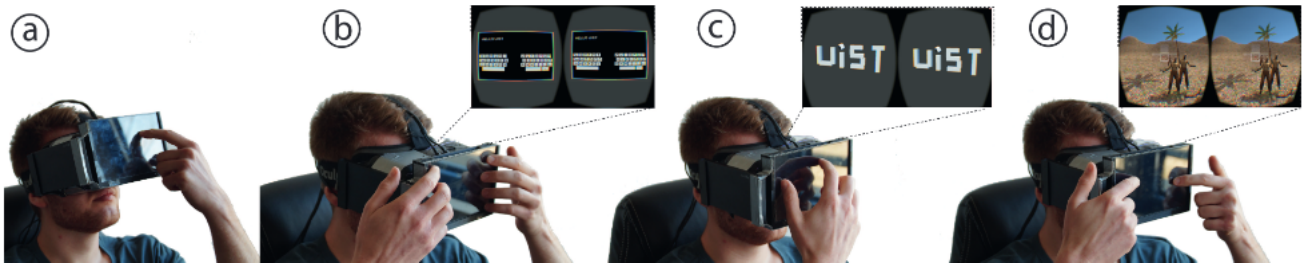

Figure 2.1: FaceTouch Demonstration
Adapted from: [9]

For these actions, the user can swipe, two-finger scale, and two-finger drag, respectively. A user study to test the system's overall usability was conducted, divided into a training phase and later a replication task where users should exercise all primary features to replicate a target model. Even though this work is not explicitly focused on interactive surfaces, results and user feedback showed that using 2D inputs on the tablet was beneficial for this context.

Following this work, others tried improving on this combination by using a tracked pen for better precision and different types of inputs for several different contexts, such as sketching [5] and spreadsheet interaction [6]. An example of the first is the case of VRSketchIn [5] where this combination of input devices was tried in depth by combining movements of the pen for inputs, but also the use of the tablet surface itself. The tablet used was tracked, as well as the pen used, and both were represented in the VE. It allows for sketching different objects and the possibility of manipulating them to create detailed objects easily. An example of this can be seen in the user study, where users were asked to draw a beach house on an island on the sea. The majority of the users opted to use the capabilities of the tablet and pen to draw surfaces like the island the house was sitting on or the sea. Some other objects, like the house, were built using the primitive extrusion feature and further operations on those primitives, such as scaling. Although these works use the touchscreen for some absolute inputs, they do not aim to fix the problems resulting from device calibration and miss-alignment of the virtual device and physical device, which can result in Z-depth or XY errors. Another similar work but in a different context was explored by Gesslein et al. [6], where the 3D space around the tablet displays extended spreadsheets views while immersed in VR environments. For this, several inputs are defined using a tracked tablet and pen capabilities. Some possible actions are pointing with the pen to select neighboring spreadsheets from the main one.

Another common context where these surfaces are tried is text entry. Previous work comparing different text entry methods in immersive VR proves that with a simple rendering of the users' fingertips, the performance was significantly improved when typing on a touchscreen keyboard [8]. However, this approach relies on retroreflective markers on every finger to track the hand, which is impractical. Similarly, in another project, several methods are compared for text typing tasks, and it is concluded that smartphones have some advantages compared to other methods, such as Raycasting or using a Gamepad for typing [3]. Smartphones had a lower error rate than

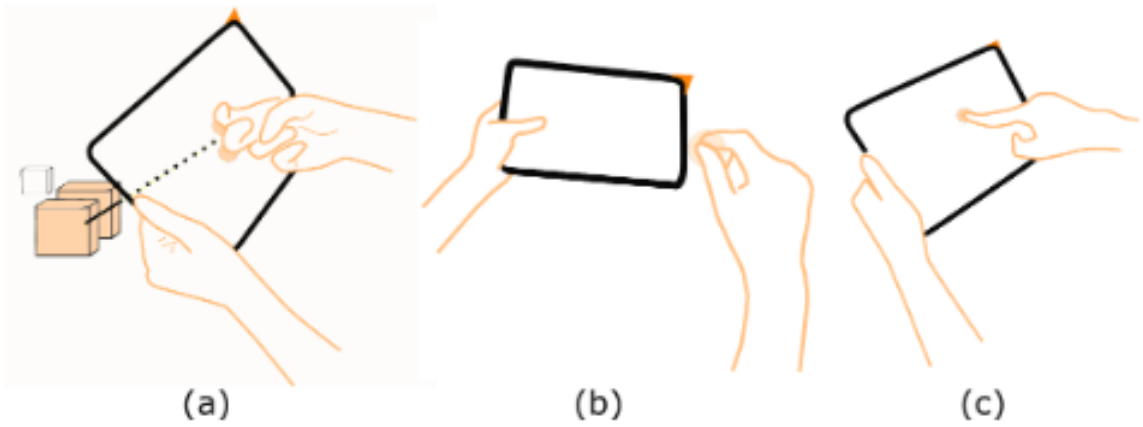

Figure 2.2: TabletInVR Input Demonstration
Adapted from: [23]

Raycasting but higher than using the Gamepad. However, typing speed was highest using the Raycasting method and lowest when using the Gamepad, with the smartphone in the middle of the other 2. The user study conducted afterward suggested that users can not only adapt their knowledge of everyday smartphone keyboards and transfer some of the typing performance to the virtual world but also improve their performance the more they type.

An issue with interactive surfaces is that we cannot detect which finger touches the surface. This is not always relevant, as we are more interested in detecting gestures and not specifically finger touches, but TapID [14] proposes a solution to this issue with the use of a wrist-worn wearable device that can accurately detect which finger touches the surface. Upon detecting a tap event, TapID forwards this input to a neural network classifier which will then output the likeliness of each finger being the cause for the tap. This wearable is combined with standard commercial hand-tracking methods, complementing each other to trigger inputs in the Virtual world.

Biener et al. [2] explored the use of touchscreens as a support for mobile information workers, more specifically, the use of touchscreens in VR for small, constrained spaces helping knowledge workers. For that, they resort to tracking the fingers and the device used, such as a tablet. The main focus of this work is to study how to interact with information beyond the physical touchscreen, in other words, how to go from 2D to 3D information while still using the same device. Several contexts are explored in the form of applications, and results prove that interactive surfaces were suitable for these activities, which included a window manager, medical imaging view, code version control, and others.

Related to device tracking but also interactive surfaces, there is also VRySmart [12], a Framework for Embedding Smart Devices into Virtual Reality. VRySmart works by representing the tracked smart device in the virtual world and using the sensors present in these smart devices to make actions in the virtual world. In order to track the smart device, optical tracking, and SLAM-based tracking are used. Many uses and functionalities were tested, and users that participated in user studies reported a good experience using Smart Devices as controllers in Virtual Reality,

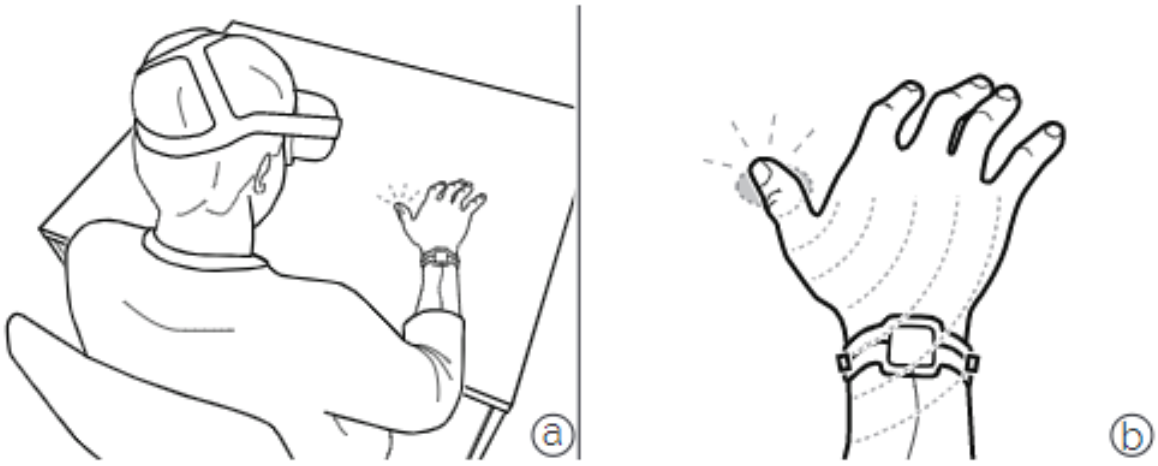

Figure 2.3: TapID device
Adapted from: [14]

which again shows the versatility and utility of these devices in Virtual Reality. TrackCap [16] opts for a similar approach to embed smart devices into Virtual Reality but instead uses the front smartphone camera to track a "cap" mounted on the HMD. This tracking is then used to determine the relative pose of the HMD in relation to the device without needing the common hand-tracking technologies used since only a line of sight between the smart device and the HMD is needed. TrackCap runs as a standalone service on the smartphone, communicating with the HMD via WiFi or Bluetooth.

More recently, a new approach was tried called Phonetroller [13]. It was implemented by placing a mirror above the touchscreen such that the front-facing camera captures the thumb's position. The captured image is then processed to get only the thumb image rendered in the virtual environment. A slight calibration is also applied for the thumb position due to mismatches between the perceived and actual contact points. Furthermore, a user study was conducted to evaluate different representations of the thumb, which demonstrated that fingernail-sized targets could be precisely hit by representing the thumbs with a shadow. This approach proved to be very successful, improving the users' performance on several tasks, such as text entry.

2.2 Hand Tracking in Virtual Reality

Most approaches rely on hand tracking to allow for visual feedback of the hands in the virtual world or to combine the use of around-device gestures with some simple touchscreen inputs such as dragging or pinching. Around-device gestures refer to the movements of the hands outside the HMD field of view but within range of other VR input devices like interactive surfaces.

It is specifically crucial that hand tracking is done as precisely as possible because a common problem is the fact that the contact point with the surface is not the same as the point that is tracked.

Figure 2.4: Phonetroller device
Adapted from: [13]

This is a problem because it makes tasks that rely on absolute inputs on an interactive surface very difficult and susceptible to errors.

Son et al. [22] explored this problem in depth and designed a regression method to correct this discrepancy and better estimate the actual contact point. The user study showed that the two-thumb touch typing method explored achieved one of the best performances in terms of words per minute compared to other approaches using interactive surfaces in Virtual Reality. However, a limitation of this work was that the fingers were tracked with reflective markers, which is not practical.

Previous work shows that the existing tracking methods could have a potential offset of 1 cm on touch [20, 19]. Touching The Droid aims to reduce this miss-alignment by using a dynamic calibration algorithm to improve touch accuracy [26]. The algorithm focuses on correcting Z-depth errors by first estimating the finger position and then using it to reduce the speed of the virtual finger when the distance to the virtual screen is less than a preset value. This should reduce the probability of penetration errors, which are errors where the virtual finger penetrates the virtual screen before physical touch happens.

DeepFisheye [18] offers a practical alternative to the more usual hand-tracking devices through a fisheye camera attached to the bottom of a touchscreen and an Artificial Intelligence (AI) algorithm. This AI is based on a neural network and uses depth estimation to predict hand postures. This project focuses on estimating the 3D position of the fingertips by applying deep learning

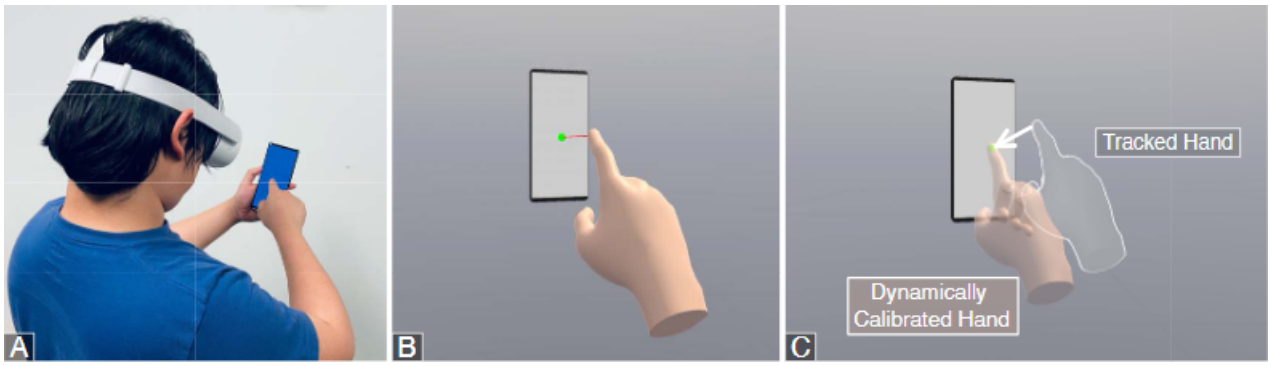

Figure 2.5: Touching The Droid Demonstration
Adapted from: [26]

methods to the images of the hand. It can also classify which finger is responsible for touch events and recognize hand postures. Figure 2.6 illustrates images of hands captured using the fisheye camera that was used to train the Artificial Intelligence algorithm.

How the hands are represented in the virtual world is also essential because it directly affects the user's ability to interact with and control the VE. The sense of presence and immersion in the virtual world can be improved with a natural and intuitive rendering of hands, making the experience more enjoyable and improving the overall usability of the VR system. The user's ability to manipulate virtual items or use virtual tools can be impacted by hand representation, which may affect productivity.

Grubert et al. [7] compared the performance of several hand representations for typing in VR on standard physical keyboards, eventually concluding that fingertip visualization and video inlay (blended video see-through of the user's hands) methods had a significantly lower error rate in the text entry tasks conducted in the user study, but no significant differences related to the speed of typing. Afonso et al. [1] investigated the effect of different avatars in selection tasks while using a tablet as an input device in an Immersive Virtual Environment. While no avatar (no representation of the hand) had better results related to the time it takes to complete a selection task, the use of an avatar for hand representation made it so users were lesser prone to errors. This is attributed to the fact that the tracking is not perfect. It is also concluded that the hand representations need not be entirely realistic, which may make the implementation easier. Similarly, Knierim et al. [10] investigated using different avatar hands for text typing tasks. They made use of retroreflective markers to track the hand, as well as track the physical keyboard they use. Once again, the results show that performance is very lacking compared to real-world text input. However, the representation of the hands in the virtual world increases typing performance, response time, and typing accuracy of inexperienced users. Concerning which type of hand avatar to use, results showed no significant differences in performance between the different hand avatar representations in most of the users. Inexperienced typists especially require more realistic hand representations, likely because they need more visual guidance to perform typing tasks. As such, the conclusions from this work suggest rendering realistic hands to obtain the best performance.

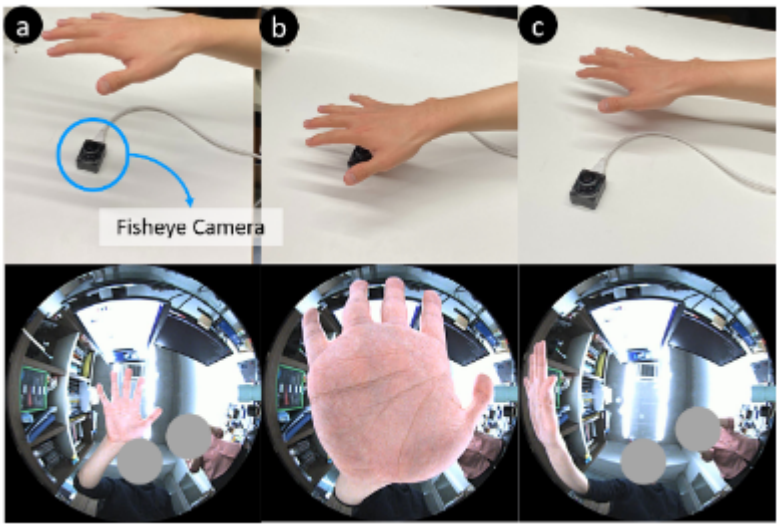

Figure 2.6: DeepFisheye camera and hand images captured
Adapted from: [18]

Figure 2.7: Different hand avatar representations
Adapted from: [10]

As stated before, TabletInVR [23] combines the use of touchscreens with 3D mid-air barehand gestures. It relies on hand tracking to represent the hands inside the virtual world. This hand tracking is done with the help of a Leap Motion device [1], a commonly used device for hand tracking. However, there are also other ways to incorporate hand tracking in VR. An example is the CAVE system, where users can see their real hands and their real hands if hand tracking is implemented with motion controllers. However, previous work shows that the majority of users not only prefer using the more commonly used options such as the HMD, and their performance is better for several tasks such as pick-and-place and text typing [17]. Besides that, CAVE systems are not appropriate for individual use since they are very spatially demanding and expensive.

Vosinakis et al. [24] analyzed several different techniques to give feedback to users while performing bare-hand interaction. For that purpose, they resort to common hand-tracking options like the Leap Motion device and Oculus Rift. No feedback technique had a significant advantage over the others. However, user satisfaction reported that "object halo" was preferred. This technique makes it so that when you grab an object, a halo is displayed around it. Some limitations reported were the fact that LEAP motion often produced tracking errors or misinterpretations of hand gestures, which resulted in the majority of errors.

2.3 Absolute target selection in Virtual Reality

Absolute target selection in VR refers to a method of selecting targets in a virtual reality environment through precise and direct hand or controller input. Text typing is an example of a task that relies upon absolute target selection. However, text typing is not trivial in Virtual reality. The fact that we are immersed makes it so that we are immersed in a VE, and as such, we are unable to see the device we are typing in. When needing text input, most applications resort to either pointing at a virtual keyboard in the virtual environment, head movements, or speech-to-text.

Walker et al. [25] explored using physical keyboards in combination with HMDs. The approach used to solve the fact that we are unable to see the hands relies on a virtual keyboard providing live feedback coupled with an autocorrection algorithm.

TextBox [4] provides a novel approach by equipping the HMD with keys and touch sensors. Figure 2.8 illustrates this device. These two will form a keyboard layout with six rows (1 for each touch sensor) and five columns (1 for each key). Although interesting, this approach did not significantly improve text typing performance. The user studies conducted showed that users could improve their performance after a while, which demonstrates learnability.

As stated before, Son et al. [22] explored ways to improve the performance of text typing in VR by adding virtual cursors that try to accurately represent the contact point of the fingers with the input surface. Son et al. [21] investigated the impact of hover feedback and grip stability on the performance of two-thumb touch typing in Virtual Reality. Although hover feedback improved the efficiency of typing tasks, the typing speed was only 19.19 words per minute which is still very far away from the speeds achieved in the real world. After 2 hours of practice, speeds could reach

[1] https://www.ultraleap.com/product/leap-motion-controller/

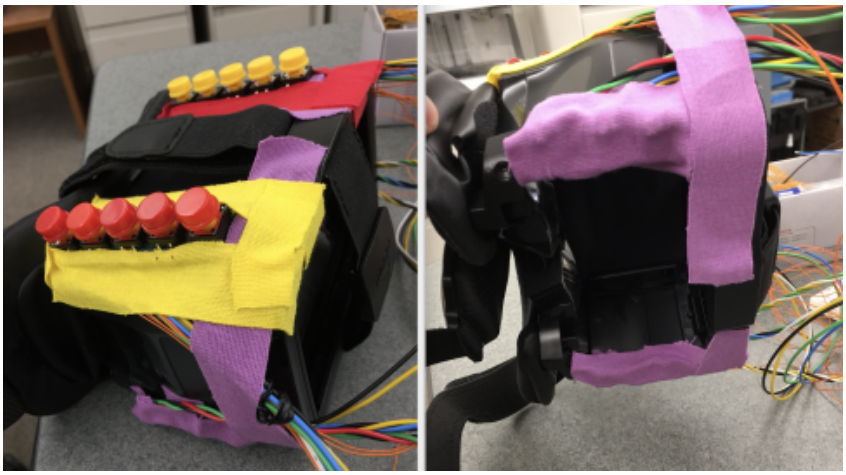

Figure 2.8: TextBox Device
Adapted from: [4]

around 30 words per minute which is an improvement but still below the smartphone text entry speeds of around 43 words per minute.

2.4 Discussion

Table 2.1 summarizes much of the work done concerning the use of touchscreens and Interactive Surfaces in Virtual Reality. We decided to classify these works into several categories. The first column is related to **Hand Tracking**, whether it is present or not, and how it is done. The **Device Tracking** column specifies whether or not device tracking is present and what devices are being tracked. **Type of Input** characterizes if the inputs used are relative, absolute, or mixed. Relative inputs mean that the interactive surface is used for actions such as gestures where the input positioning in the screen is irrelevant. An example of a relative input is a dragging gesture on a tablet surface to move a cursor in the virtual world. In contrast, absolute inputs mean that we care about the actual position where the touch occurs. Mixed inputs mean that both are present. This work will focus on absolute inputs and ways of solving the issues stated before. The **Multi-touch** column is present to distinguish the works that allow for multi-touch events or just single-touch. A multi-touch event occurs when more than one finger touches the surface. Finally, we have the **Tasks** column that gives us information about what tasks and contexts these works were tried on.

The table shows that most approaches use standard hand-tracking methods like the LEAP motion device. Although these devices provide us with reasonable accuracy, we have seen that it is insufficient for precise touch input since they are subject to errors related to miss-alignment [20, 19]. Touching the Droid [26], although with some limitations, uses these same tracking methods but also tries to correct the errors with a dynamic calibration algorithm.

From the table, we can also see that a lot of the work done tries to combine the use of a tracked pen for more precision. Although the pen provides a minor point of contact with the surface and, in theory, better precision for smaller targets than a finger, it is still subject to the calibration and

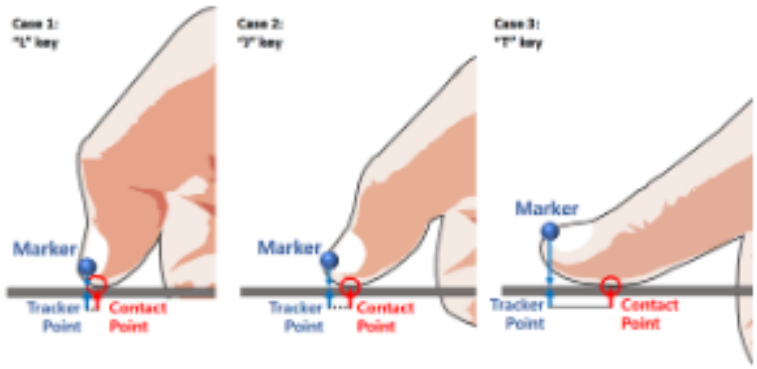

Figure 2.9: Discrepancy between contact point and tracked point
Adapted from: [22]

perception errors mentioned before. As such, this constitutes a problem. Besides that, the pen only allows for single-touch actions.

Concerning the tasks executed in these works, we can see that it is common to evaluate the use of interactive surfaces in relation to text typing. This is because most of these works try to take advantage of users' familiarity with tablets/smartphones to replicate a virtual keyboard. However, we can see that results are still far from the expected, with performance being way lower than real-world typing [8]. Relative input works opt for a more varied set of tasks. TabletInVR [23] defines an interaction space where they combine relative inputs, such as pinching for some actions and some absolute inputs to, for example, open menus. However, the precision necessary for such absolute actions is not very high. Although VRSketchIn [5] uses some absolute inputs, we consider most of them relative because they are used to paint strokes on a virtual world, as if moving a virtual cursor. Similarly, we have the work developed by Gesslein et al. [6], where most of the inputs are relative, but some are considered absolute, even though they do not require much precision.

The works that use absolute inputs, especially for text typing tasks, rely heavily on fully tracking the hands using retroreflective markers [3, 22]. As such, the examined research work further supports the idea that we can explore the use of absolute and multi-touch inputs concerning absolute target selection, such as text typing, without using reflective markers for hand tracking.

Table 2.1: Summary of the works already done in the area

Source	Hand tracking	Device tracking	Type of Input	Multi-touch	Tasks
TabletinVR [23]	LEAP	Tablet + Pen	Mixed	Yes	Create, Select, Delete, Transform, Modify 3D models
VRSketchIn [5]	Reflective markers	Tablet + Pen	Mixed	Yes	Select, Add, Subtract, Rotate, Translate, Scale primitives for 3D sketching
Mine et al. [15]	No	No	Absolute	No	Menu navigation, Object manipulation
Boustila et al. [3]	No	No	Relative	Yes	Text typing
Phonetroller [13]	Phone camera	Phone	Absolute	No	Text typing, freely experience several applications
Gesslein et al. [6]	No	Tablet + Pen	Mixed	Yes	Spreadsheet actions (Creating functions,...)
Grubert et al. [8]	Reflective markers	No	Absolute	No	Text typing
VRySmart [12]	No	Phone	Relative	.	Several different uses (Messaging app, ...)
Lee et al. [11]	No	No	Relative	Yes	Selection methods: Two-Finger and Drag-n-Tap
FaceTouch [9]	No	No	Mixed	Yes	Several different uses (Text Entry, Game controllers, ...)
TouchingTheDroid [26]	LEAP	No	Absolute	No	Target acquisition
Son et al. [22]	Reflective markers	Phone	Absolute	Yes	Text typing

Chapter 3

Design of the proposed solution

In this chapter, we delve into the problem and present our proposed solution based on an extensive review of relevant literature. We aim to provide a comprehensive understanding of the problem and highlight crucial aspects that play a pivotal role in the success of our solution.

3.1 Proposed Solution

Our proposed solution will provide the user with an approximation of the finger's contact point with the surface based on the position of their hands and the device, similar to the approach proposed by [22]. This approximation must make it so that when the touch occurs, it is coherent with the feedback we provided.

In order to successfully answer all the proposed research questions, we will develop a technique that predicts the contact point of the fingers with the surfaces, allowing users to rely on proprioception to improve absolute target selection performance in VR without tracking every finger. Similarly to [26], we will use the fingertip position to generate this predicted contact point.

3.2 Requirements Specification

To successfully implement our proposed solution and address the research questions at hand, it is crucial to establish a comprehensive set of requirements. These requirements will guide the development process and ensure that the solution meets the necessary criteria. As such, we can define the list of requirements:

- **Markerless Hand Tracking**: The solution should achieve accurate hand tracking without using any markers. It should provide a reliable estimation of the user's finger contact points in the virtual reality (VR) environment, allowing for natural and intuitive interactions. This requirement is extremely important since we want to minimize the need for external tools at maximum.

- **Practicality**: The solution should prioritize practicality for day-to-day VR usage. It should offer a user-friendly experience without the need for cumbersome and intrusive reflective markers that may hinder the overall immersion and convenience.
- **Robust Calibration**: The technique should be robust against calibration errors that are inherent to the chosen tracking device. It should account for variations and inconsistencies in the calibration process, ensuring reliable and consistent hand-tracking performance.
- **Interaction with Interactive Surface**: The solution should support using an interactive surface as an input device. It should allow users to seamlessly interact with the VE using the tracked hands and interactive surfaces. The solution should accurately use touch input data captured by these surfaces and translate them into corresponding actions within the VE.
- **Tracking of the Interactive Surface**: The proposed solution should incorporate tracking mechanisms for the surface itself to enable interaction with the interactive surface. The tracking is important so that the interactive surface assumes a natural position in the virtual world compared to the user's position.
- **Virtual Representation of the Interactive Surface in the Virtual Environment**: To create a coherent and immersive user experience, the proposed solution should provide a virtual representation of the interactive surface within the virtual environment. This representation lets users perceive the interactive surface and its position with other virtual objects. The representation should be as similar as possible to the real representation of the object.

3.3 Architecture

The main architecture relies on several components. One of the main components is the Application, which runs inside Unity. This application will manage all the input data necessary from the Interactive surface position, the Hand position, and the touch data from the interactive surface. The Interactive surface position will come from a VIVE Tracker, which will be placed together with the input device to supply our application with 3D position data. Furthermore, the Hand data will come from a LEAP Motion device, which will be responsible for tracking the hands of the user. The touch data will come directly from the interactive surface. All this data is then processed in the application, and the result will be outputted to the rendered, which will then send the image back to the HMD for the user to see it.

For a visual representation of the implementation architecture, refer to figure 3.1.

3.4 Development Methodology

To facilitate the development of our prototype, we opted for an iterative approach consisting of multiple development cycles, similar to Scrum methodologies. During each stage, an initial requirement-gathering phase was needed, followed by prototyping, and only after implementing

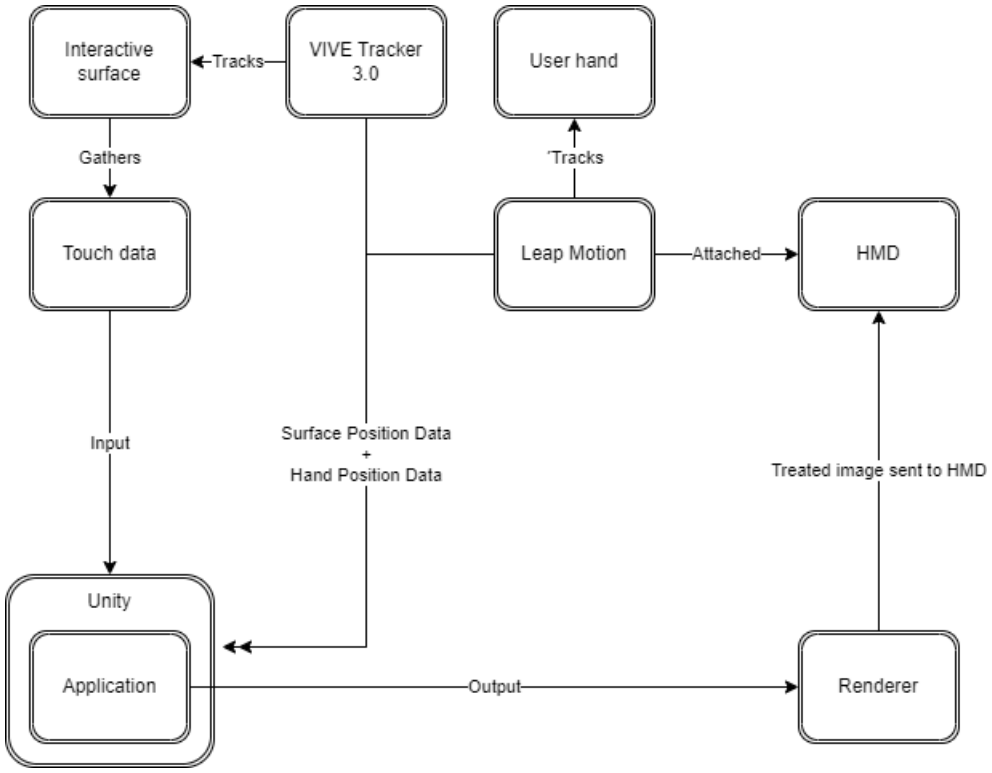

Figure 3.1: Architecture

and testing. With these three steps concluded it was enough to move forward into the next development cycle until all the necessary components were ready for user testing.

Chapter 4

Development of the proposed solution

In this chapter, we present how we implemented our proposed solution. Our solution will provide the users with a visual representation of the contact point of their finger's with the input surface, which will then be used to trigger the inputs in the virtual world.

4.1 Visual representation of interactions

The functionality of our solution heavily depends on providing visual cues to the user, conveying information about ongoing interactions. We opted for a simple representation of the finger contact point represented by a blue sphere for the right hand and a green sphere for the left hand, as shown in figure 4.1. While using this technique, we opted not to represent the hands in the Virtual World. This was due to the feedback obtained from preliminary testing, where users said there was too much visual information simultaneously without a significant advantage. Furthermore, the fact that we use a projection makes it so we fix the Y-axis errors that may result from misscalibration since the projection is always present. This way, haptic feedback is always coherent with touches in the Virtual World.

If the LEAP Motion device lost tracking of one or both hands, the spheres would not move anymore and turn grey. The position of the spheres is relative to the plane defined by the device, which is being tracked by a VIVE tracker 3.0. These representations are calculated by defining a vector between the tip of the finger used for the touches and the device's normal vector. This is then used to translate the sphere until it reaches the plane, as shown in code snippet in 4.1. In line 7 of the code below, the variable upShift is used to translate the spheres slightly upwards so that they do not stay inside the 3D representation of the input device.

```
1 private void ProjectOnMobile(bool isLeft,Vector3 thumbTipPosition)
2     {
3         Transform mobilePhonePlane = device.transform;
4 
5         var up = mobilePhonePlane.forward;
```

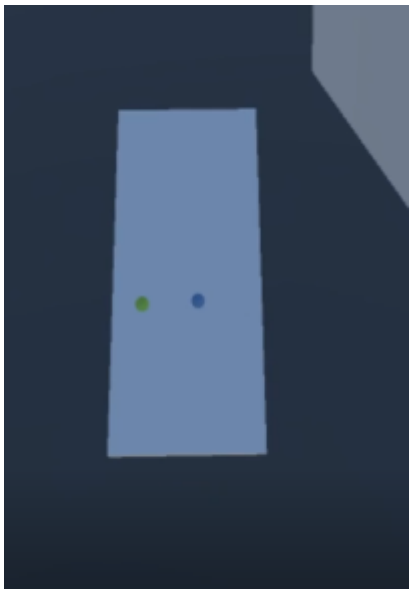

Figure 4.1: How the user sees the Projection technique in VR

```
6         Vector3 targetPos = Vector3.ProjectOnPlane(thumbTipPosition, up) + Vector3.
              Dot(mobilePhonePlane.position, up) * up;
7         Vector3 upShift = -up * this.upShift;
8
9         if (isLeft)
10        {
11            LeftHandProjection.transform.localPosition = targetPos + upShift;
12            LeftHandProjection.GetComponent<Renderer>().material.color = new Color
                  (174/255f, 243/255f, 89/255f);
13        }
14        else
15        {
16            RightHandProjection.transform.localPosition = targetPos + upShift;
17            RightHandProjection.GetComponent<Renderer>().material.color = new Color
                  (114/255f, 159/255f, 207/255f);
18        }
19
20    }
```

Listing 4.1: Code snippet for calculating projection position

After every click on the input device, an input event is triggered in the virtual world. We used Unity's built-in collision detection engine to trigger these inputs. Each projection had an attribute that determined with which object it was colliding. When a collision was detected, this attribute would change to have a reference to the object that had collided. To decide which sphere (right hand or left hand) to use, the Euclidian distance was calculated between the actual touch's coordinates and each sphere's coordinates. The one with the less distance would be used to trigger the inputs.

Figure 4.2: Headset used with LEAP Motion Device attached

4.2 Choice of Hand-Tracking Device

To initiate the implementation of our solution, we faced the crucial decision of determining which device would provide us with accurate hand position data. Initially, we explored the possibility of utilizing the cameras already present on the HTC VIVE Pro 2 to track the hands. This approach aimed to eliminate the need for external devices, providing a seamless user experience. However, upon testing, we encountered significant challenges with the accuracy of the data obtained. The fingers were frequently distorted, rendering this approach unreliable and impractical.

This presented a significant setback, especially considering that one of the interactive surfaces chosen for our solution was a smartphone that required the user to hold it. When holding it, the device partially covers some of the fingers, making tracking them unreliable. The finger-tracking data severely affected the usability and effectiveness of the solution in this scenario. Consequently, an alternative approach was necessary. We decided to incorporate the use of a LEAP Motion Device as the new tracking device. This LEAP Motion Device would be attached to the front of the HMD, as shown in figure 4.2

After implementing the LEAP Motion Device into our solution, we conducted a calibration process to ensure accurate hand tracking. Once calibrated, the LEAP Motion Device provided us with robust hand-tracking data, significantly improving the accuracy and reliability of the solution.

However, it is important to note that using the LEAP Motion Device did not eliminate all challenges. A specific challenge arose when the user held the smartphone with both hands simultaneously with the thumbs above the screen. In this configuration, some fingers would be occluded from the view of the LEAP Motion Device, which was attached to the front of the HMD. While

this occlusion presented a limitation, the positional data of the thumbs remained unaffected. Based on this observation, we decided to trigger inputs solely based on the positional data of the finger used for contact, which was the thumb for the smartphone and the index for the larger touchscreen, instead of allowing any finger to trigger inputs. This way, we ensured reliable and consistent interaction with the interactive surfaces.

4.3 Initial calibration of the LEAP Motion Device

Since the LEAP Motion device was attached to the front of the HMD, it was necessary to perform an initial calibration. This process was done using the first **N** touches, an adjustable field before the running of the application. Every time a touch was performed on any part of the input surface, the distance between the tip of the finger used for the touch and the coordinates of the actual touch was calculated. After all the touches necessary for calibration, the average was calculated, and the offsets were applied to the LEAP Motion object in Unity. The user would then evaluate the final calibration to check for eventual errors, and the process would be repeated until the user felt the calibration was accurate enough. This process allowed us to improve the performance of our solution by providing us with slightly more accurate data on the hands.

4.4 Development tools and hardware

We will use Unity Game Engine to aid the development of the proposed solution. Unity offers plenty of support for VR applications without the need for external plugins, as well as complete documentation. Besides that, it is compatible with multiple devices. Previous experience working with Unity also influenced this decision, although not in a Virtual Reality context.

For hardware, we will use a VR headset equipped with a LEAP Motion device responsible for providing our application with information on the hands' position and posture. Furthermore, we will use two interactive surfaces as input devices: a Smartphone and a touchscreen monitor.

4.5 Choice of Interactive Surfaces

In addition to selecting the appropriate hand-tracking device, we recognized the importance of choosing suitable interactive surfaces for our solution. To ensure versatility and practicality, we incorporated two different types of interactive surfaces into our implementation.

The first interactive surface we chose was a smartphone that is readily available to most individuals in their daily lives. This choice aimed to reflect real-world usage scenarios and make our solution accessible to a broader user base.

The second interactive surface we selected was a larger touchscreen device. This choice provided us with a different input modality and expanded the possibilities for interaction within the virtual environment.

Although our main focus was not specifically on the influence of different interactive surfaces on the overall solution, evaluating their viability and effectiveness was still important. During the user testing phase, we collected valuable feedback and insights regarding using these different input devices. This feedback allowed us to assess each interactive surface option's usability, user satisfaction, and practicality.

4.6 Device Tracking

The devices used to implement our solution were being tracked using a VIVE Tracker 3.0. In the case of the smartphone, this tracker was attached at the bottom of the phone as shown in figure 4.3. This position was chosen because the phone was hard to hold if the tracker was positioned in the middle area of the smartphone. However, this problem was still present if the tracker was positioned at the bottom, and it was impossible to position it at the top due to the cameras present on the mobile phone. To overcome this problem, it was decided to use the phone upside down, and as such, the values of the Y coordinate of every touch were inverted, being calculated the following way:

$$touchY = maxY - actualTouchY$$

Meanwhile, while using the larger touchscreen, the tracker was placed in the middle of the back part of the touchscreen.

The placement of the trackers forced us to make minor adjustments in the placement of the objects in the virtual world since their position relative to the "middle" of the input device had to be calculated to match the positions. As such, these distances were accurately measured so that the trackers would always maintain an accurate position.

4.7 Visual representation of the input devices in the Virtual World

To maintain all the principles of VR and for the user to be as immersed as possible, the virtual representation of the input device needed to be as close as possible to the one being used by the users in the real world. As such, the most important thing was that the device in the virtual world had the same dimensions as the real one. Our implementation allows for the size of the virtual device to be set before running the application, and this was used to define these measures based on the technical sheet of the input devices used.

4.8 Communication

Two different apps are created to facilitate the development of the solution, representing a server and a client. The server app was running the whole VR environment and was responsible for interpreting the messages sent from the client application. The client application was running on the input device. It was responsible for sending a message containing the device resolution

Figure 4.3: Positioning of the tracker attached to the smartphone

when first starting and the coordinates of every touch on the input surface. All the communication involved used the UDP sockets. This method was chosen because the UDP protocol is reliable rather than TCP. Besides, UDP allows for fast and efficient communication without the overhead of establishing and maintaining a connection.

A simple format was defined for all the messages transmitted. The initial message containing the device resolution was defined as **Size:(ScreenWidth,ScreenHeight)** while the touch messages had to incorporate an ID due to the fact that we allow for multi-touch events. Besides that, the touch events have different phases, which are defined as **Begin**, **Moved**, and **End**. As such, three different formats for the touch event messages are defined below:

- **ID:id;Begin-(x,y)**
- **ID:id;Moved-(x,y)**
- **ID:id;End**

4.9 Additional implemented methods

To conduct a comparative analysis of our solution, we employed another method, which we will refer to as the Baseline Method. The purpose of this method was to establish a simple and straightforward approach for hand representation in the virtual environment. To achieve this, we relied on the information the LEAP Motion Device provided to display the user's hand.

However, while representing the hand in the Virtual World, a problem arose when it came to determining the point of contact between the finger and the interactive surface within the virtual world. Since it is not trivial to determine the contact point between the hand representation and the input device representation because we would have to detect the collision between a specific part of the finger (contact point) and a part of the virtual representation of the input device an alternative approach was required for input triggering. As a result, we decided that the inputs would be triggered at the precise location of the user's touch on the interactive surface itself. It is important to note that this approach is more susceptible to calibration errors than our developed solution.

Figure 4.4 showcases the visual representation of the Baseline Method as perceived by the user in the virtual world. This depiction provides a visual reference for how the hand is represented based on the data received from the LEAP Motion Device. The lack of explicit finger contact points is evident in this representation, highlighting the need for an alternative approach to input triggering.

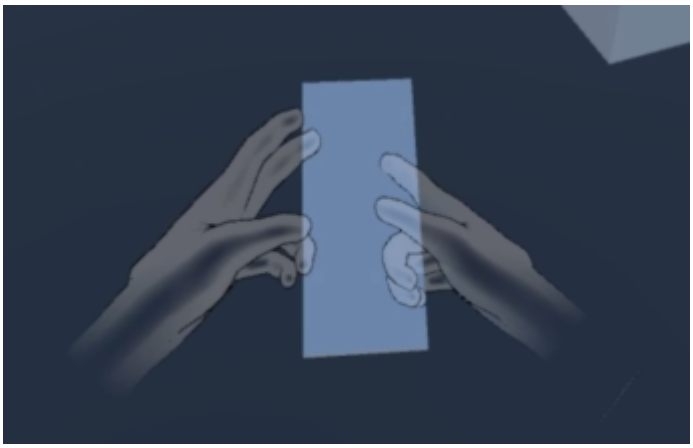

Figure 4.4: How the user sees the Baseline technique in VR

Chapter 5

User Testing

We conducted a series of experiments to evaluate our solution's effectiveness accurately. Participants were asked to play a Whac-A-Mole-like game where they had to click the moles that appeared as fast as possible, allowing us to measure quantitative and qualitative metrics. With these tests, we intend to test the effectiveness of our proposed solution.

5.1 Setup

The hardware used consisted of an HTC VIVE Pro headset, with a LEAP Motion Device attached in the front at a 15º angle, as shown in figure 4.2.

Additionally, we opted for two types of interactive surfaces for the inputs, a Xiaomi Redmi Note 10 and a Dell ST220T both being tracked using an HTC VIVE Tracker 3.0.

While performing the necessary tasks, the participants would sit while using the Dell touchscreen and stand while using the phone. This configuration was necessary due to the trackers attached to the back of the device. The sensors performed worse while using the smartphone seated, sometimes losing smartphone tracking. This is not an issue while using the touchscreen because we can place it in a place where these trackers are in perfect view of the cameras used to track.

5.2 Tasks

To analyze both techniques' and target size's effects, a set of tasks must be defined for the experiment. The experiment consisted of a simple task similar to the popular Whac-A-Mole game, where the user would try to click on targets that would show up in the virtual representation of the input device inside the virtual world. Figures 5.2 and 5.3 show the game's appearance in the Virtual World.

A grid would be generated, and the targets would appear randomly on cells of this grid. After successfully touching one of the targets, they would change position to another grid cell. The maximum targets at a time were set at two to try to force users to use multi-touch inputs. We opted

Figure 5.1: Participant during the user experiment

to define six tasks in total, to test the effect of the sizes of the grid chosen and the two techniques in question. It is important to note that the size of the grid refers to the total number of cells, and not the actual size of the targets. As such, the Small grid will have bigger targets than the Medium grid. Table 5.1 summarizes the tasks defined.

Each task had a set duration of 30 seconds.

5.3 Choice of grid sizes for user testing

To properly evaluate the size of the targets on our solution, we decided to opt for three different grid sizes, which we defined as Small, Medium, and Big. While using the smartphone, offsets of 0,2 cm and 0,1 cm were used for the length and height, while grids had a size of 6*4 cells, 9*6 cells, and 16*9 cells, respectively. These chosen sizes made it so that the sizes of the targets corresponded to 1.65 cm in length and 2.4 cm in height for the Small grid, 1.1 cm in length and

Task Name	Grid Size	Technique
T1Baseline	Small (S)	Baseline
T1Projection	Small (S)	Projection
T2Baseline	Medium (M)	Baseline
T2Projection	Medium (M)	Projection
T3Baseline	Big (L)	Baseline
T3Projection	Big (L)	Projection

Table 5.1: Tasks defined

Figure 5.2: Small grid with the whac-a-mole game running on the Smartphone

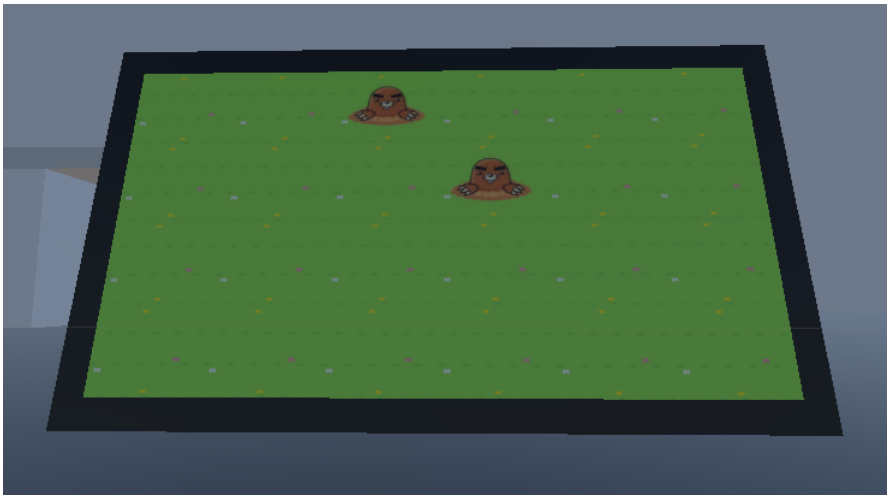

Figure 5.3: Small grid with the whac-a-mole game running on the Touchscreen

	Small Grid	Medium Grid	Big Grid
Number of cells (width * height)	6*4	9*6	16*9
Cell size (width * height)	1.65 cm * 2.4 cm	1.1 cm * 1.6 cm	0.73 cm * 0.9 cm

Table 5.2: Information about the grids for the smartphone

1.6 cm in height for the Medium grid, and 0.73 cm in length and 0.9 cm in height for the Big grid. Table 5.2 summarizes this information for the smartphone.

Meanwhile, on the touchscreen, the grids change to 4*6 cells, 6*9 cells, and 9*16 cells, respectively. These chosen sizes made it so that the sizes of the targets corresponded to 7.33 cm in length and 5.75 cm in height for the Small grid, 4.88 cm in length and 3.83 cm in height for the Medium grid, and 2.75 cm in length and 2.55 cm in height for the Big grid. Table 5.3 summarizes this information for the touchscreen.

5.4 Test Structure

Before commencing the experiments, the user was introduced to the test session. Two initial declarations were shown to the user, in which he would consent to willingly participate in the experiment and allow for a photo to be taken during the testing. After, an initial pre-test questionnaire would be filed, containing demographic data and information about the previous use of VR and interactive surfaces. The user would then be shown how to perform touch inputs on both devices and receive a brief explanation of the Whac-a-mole game. Finally, the user would be introduced to the VR environment and given some time to get comfortable with all the equipment. After completing all the necessary pre-test steps, the users verbally indicated they were ready to start the test tasks. Then, the test tasks would be triggered by keyboard inputs by the person conducting the tests. Before the tasks were executed, the participants were subject to the calibration process for the LEAP Motion Device. After completing the tasks, the user would fill in an after-test questionnaire regarding the analyzed techniques and the grid sizes.

We employed a Latin square design to ensure fair evaluation and minimize any potential bias. This design allowed us to determine the sequence in which the users would utilize the implemented methods. The different sequences included changing the grid sizes and also the techniques, but only on one of the devices. After, this sequence would be repeated on the second device. By doing so, we ensured that every combination was evaluated thoroughly while also eliminating any systematic influence that could arise from testing the methods in a particular order.

	Small Grid	Medium Grid	Big Grid
Number of cells (width * height)	4*6	6*9	9*16
Cell size (width * height)	7.33 cm * 5.75 cm	4.88 cm * 3.83 cm	2.75 cm * 2.55 cm

Table 5.3: Information about the grids for the touchscreen

Task Name	Grid Size	Technique	Device
T1Baseline	Small (S)	Baseline	Smartphone
T1Projection	Small (S)	Projection	Smartphone
T2Baseline	Medium (M)	Baseline	Smartphone
T2Projection	Medium (M)	Projection	Smartphone
T3Baseline	Big (L)	Baseline	Smartphone
T3Projection	Big (L)	Projection	Smartphone
T4Baseline	Small (S)	Baseline	Touchscreen
T4Projection	Small (S)	Projection	Touchscreen
T5Baseline	Medium (M)	Baseline	Touchscreen
T5Projection	Medium (M)	Projection	Touchscreen
T6Baseline	Big (L)	Baseline	Touchscreen
T6Projection	Big (L)	Projection	Touchscreen

Table 5.4: Full list of tasks

Since each task was repeated twice (one time for the smartphone and another for the touchscreen), the final set of tasks can be seen in table 5.4

5.5 Metrics

We gathered data from each experiment to comprehensively compare the methods. This data collection process involved capturing quantitative and qualitative user information. By obtaining a combination of numerical measurements and subjective feedback, we aimed to understand each method's performance and user experience.

5.5.1 Quantitative data

For the quantitative data, we measured the time taken to hit every target, the number of times the tracking of the hand was lost, and the number of targets generated and hit. Besides that, the total number of touches was also recorded. With this data, we could calculate a new metric which is the amount of errors. In this context, an error is a touch that does not successfully hit the target. The amount of errors can be defined by the following formula:

$$\frac{nHits - nTouches}{nTouches} * 100$$

To log this data, a script was running along with the application. Initially, the **userID** was set before running the application, and all the data would start being saved after the tasks were started by the keyboard inputs, with these same files being saved in a separate CSV file with the user's ID when the task was finished. All this data was logged and measured automatically by the system while performing the tasks, with each event being assigned a timestamp.

5.5.2 Qualitative data

For the subjective data, we wanted to evaluate how comfortable the user was using both methods, how hard it was to use both techniques, and the different grid sizes. Due to the subjective nature of the data, which can vary from user to user, we employed a standardized questionnaire for data collection. This questionnaire was separated into three parts, the first being a pre-test with information about the participant and the other two related to each of the techniques analyzed. The several questions in this form are stated below:

- Pre-test:
  - Age
  - Gender
  - Field of Study/Area of Work
  - How often do you use interactive surfaces (e.g. smartphone/touchscreen) in your day-to-day life? (1-5 scale with 1 being "Never" and 5 being "Very Frequently")
  - How often do you use interactive surfaces (e.g. smartphone/touchscreen) as an input device in virtual reality? (1-5 scale with 1 being "Never" and 5 being "Very Frequently")
  - Do you own a device with an interactive surface (e.g. smartphone/tablet)? (Yes/No)
- After-test with both Developed Technique and Baseline Method (Repeated twice, one after the Projection Technique and the other after the Baseline Method):
  - How intuitive was the use of the device in Virtual Reality using this technique (Finger Projection)? (1-5 scale with 1 being "Not intuitive" and 5 being "Very intuitive")
  - How difficult was it to use the device using this technique (Finger Projection)? (1-5 scale with 1 being "Very hard" and 5 being "Very easy")
  - Did you feel the provided feedback was enough to perform the tasks? (Yes/No)
  - Did you feel uncomfortable using the smartphone/touchscreen? (Yes/No)
  - Did you feel present in the virtual world? (Yes/No)
  - Which grid was easiest to use? (Big/Medium/Small)
  - Which grid was hardest to use? (Big/Medium/Small)
  - Overall experience (1-5 scale with 1 being "Very bad" and 5 being "Very good")

5.6 Participants

There were twenty-four participants in total, of which seventeen were male, and six were female. One of the participants opted not to mention their gender. All participants willingly agreed to participate in the experiment. Sixteen of the users were aged between 21 and 30 years, with

seven others over 50 years old and only one under 18. Nineteen of the participants reported never having used interactive surfaces before in a VR context. Seventeen participants reported using interactive surfaces very frequently in their day-to-day life, with four others using these devices frequently. Two users reported using interactive surfaces not very frequently in their daily lives. When compared to VR, nineteen of the users reported never having used this type of surface for inputs in VR, with only one reporting very frequent use. All the users possessed a device containing some interactive surface.

5.7 Results

We performed the Shapiro-Wilk test for all the following tests presented to assess the normality of the data in question. Subsequently, to compare the outcomes of two distinct techniques, we employed a Paired-Samples T-Test to identify significant differences when the data exhibited a normal distribution. Conversely, we employed a Wilcoxon Signed Rank test when the data did not conform to a normal distribution. If statistically significant differences were found in both instances, post hoc analyses were conducted using the Mann-Whitney U test to determine the methods that displayed statistically significant differences.

5.7.1 Average time to hit a target

The box-plot graph in Figure 5.4 shows the average time to hit a target for each method and each task in seconds. From this graph, we can see that, on average, tasks 1, 2, and 3 have a higher average time to hit a target. This is because the smartphone is a smaller device than the touchscreen, and, as such, the targets are also smaller. Besides, the average time increases from tasks 1 to 2 and to 3 due to the targets getting smaller between these tasks. The same happens between tasks 4 to 5 and to 6. For the first task (T1), the Wilcoxon Signed Rank test showed that the average time to hit a target with each technique had statistically significant differences ($p < 0{,}001$) while comparing techniques. Posthoc tests using the Mann-Whitney U test applied to the average time in T1 using the Projection technique versus the same metric using the Baseline technique proved that the Projection technique (2,21s) was slower than the Baseline technique (1,23s) for a p-value of $< 0{,}05$. For the second task (T2), the Wilcoxon Signed Rank test showed that the average time to hit a target with each technique had statistically significant differences ($p < 0{,}005$) while comparing techniques. Posthoc tests using the Mann-Whitney U test proved that the Projection technique (2,27s) was slower than the Baseline technique (1,47s) for a p-value of $< 0{,}05$. For the remaining tasks (T3, T4, T5, T6), no statistically significant differences were found using both methods. After analyzing these results, we can see that the Baseline technique outperformed the Projection technique in tasks 1 and 2, representing the small and medium-sized grid while using the smartphone in terms of average time to hit a target. This might be because the users performed more exploratory touches using the Baseline technique while trying to hit the targets. Due to the slight discrepancy between real touch and perceived touch, users tried to touch more around the target area rather than the actual target.

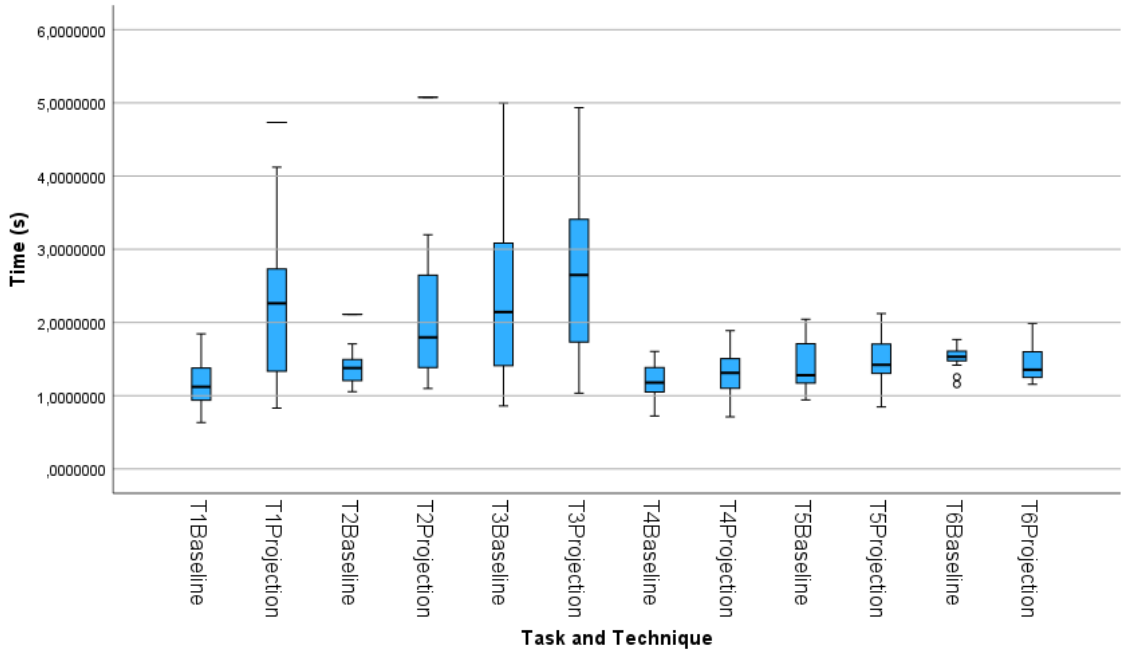

Figure 5.4: In seconds, Average time to hit each target using the two methods on each task.

5.7.2 Amount of errors

The bar graph in Figure 5.5 shows the average error rate for each method and each task in percentage. By analyzing the graph, we can see that in every task, the average error rate is always higher while using the Baseline method instead of the developed technique. For the third task (T3), the Wilcoxon Signed Rank test showed that the average error rate with each technique had statistically significant differences ($p < 0,05$) while comparing the amount of errors that each technique had. Posthoc tests using the Mann-Whitney U test proved that the Projection technique (83,9%) was less error-prone than the Baseline technique (90,0%) for a p-value of $< 0,05$. For the sixth task (T6), the Wilcoxon Signed Rank test showed that the average error rate with each technique had statistically significant differences ($p < 0,05$) while comparing the amount of errors that each technique had. Posthoc tests using the Mann-Whitney U test proved that the Projection technique (58,8%) was less error-prone than the Baseline technique (72,2%) for a p-value of $< 0,05$.

5.7.3 Analysis of Quantitivate Results

Even though our technique performed slightly worse regarding average time to hit the target, we must also consider the number of errors committed in every task. The reason for this is that while using the Baseline Method, most of the users performed a lot more exploratory touches in the area around the targets due to the slight discrepancy between perceived touch and actual touch, which eventually resulted in a bigger amount of errors, but also a lower average time since the users would make a lot more touches in general. This is especially noticeable in the number of errors in tasks 3 and 6, which correspond to the big-sized grids. Since the targets were smaller, this

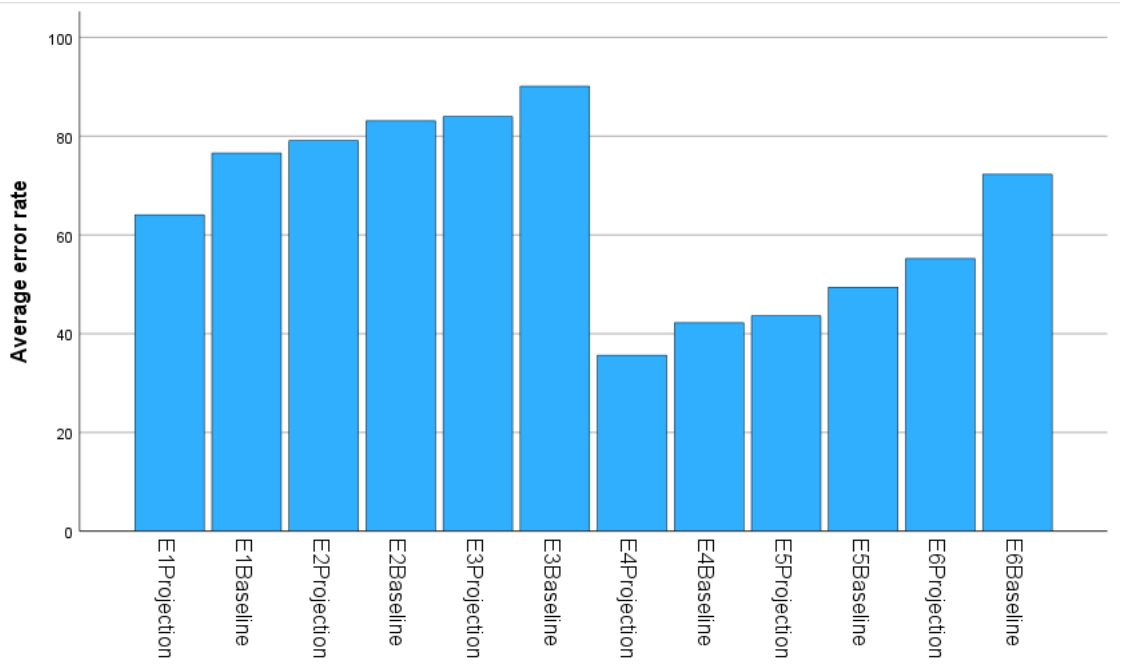

Figure 5.5: Average error rate using the two methods on each task, in percentage.

discrepancy would be a lot more noticeable, with most users reporting that even though they were hitting the target visually, the actual touch would be slightly offset in some direction, resulting in a miss and not a hit on the targets.

Besides this, we can also analyze the effect of the grid sizes on the number of errors for both the touchscreen and smartphone combined. The boxplot in figure 5.6 demonstrates that the number of errors increases the smaller the targets are, with a more noticeable difference on the Big grid. This matches the feedback given by most of the users that participated in the study, that the targets in the Big grid, especially on the smartphone, were indeed too small and hard to hit no matter the technique.

5.7.4 Overall rating of technique

The bar graph in Figure 5.7 shows the median rating users gave to the overall experience of each technique. This graph shows that most users prefer the Projection technique, with a median of 4 out of 5, compared to the Baseline method, with 3 out of 5.

5.7.5 Intuitiveness of technique

Related to the question on the intuitiveness of each technique, a Chi-squared test was applied showing a statistically significant difference between the methods. Posthoc tests using the Mann-Whitney U test showed that most users preferred the Projection Technique (Average of 4,25 out of 5) over the Baseline Method (Average of 3,58 out of 5).

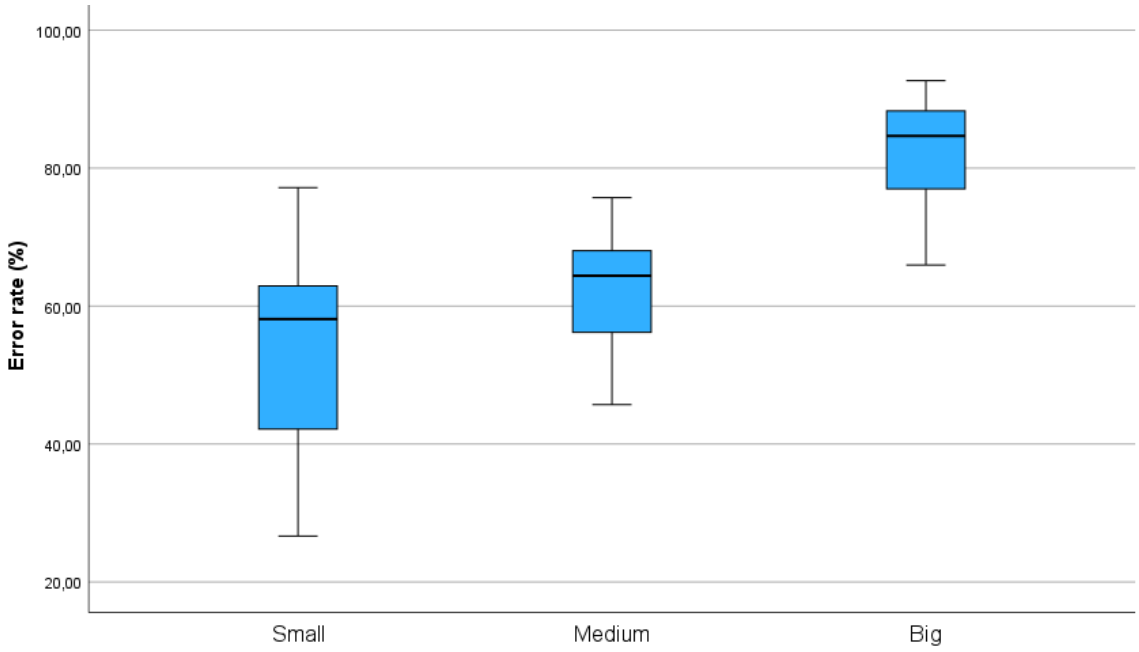

Figure 5.6: Error rate for each grid size, in percentage.

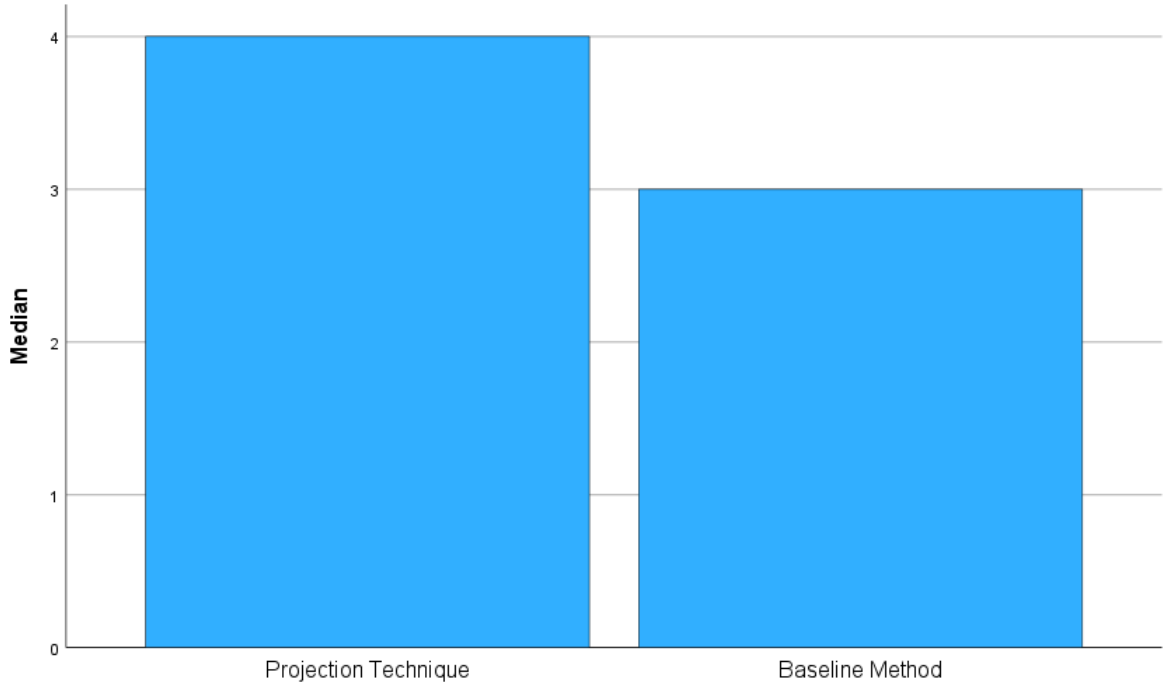

Figure 5.7: Median rating of each technique

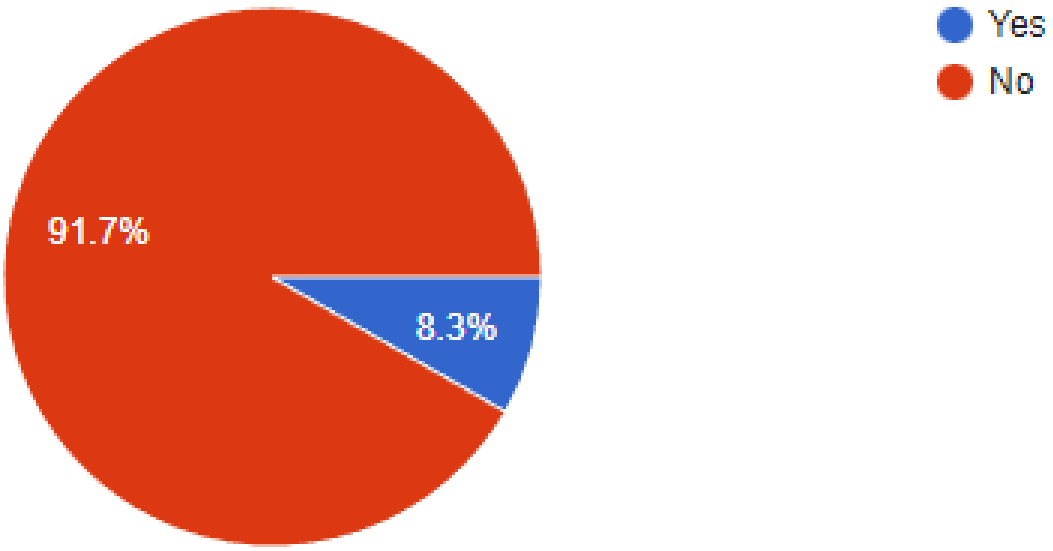

Figure 5.8: Did you feel uncomfortable using the Projection Technique?

5.7.6 Difficulty of technique

Related to the question on the difficulty of each technique, a Chi-squared test was applied showing a statistically significant difference between the methods. Posthoc tests using the Mann-Whitney U test showed that most users felt the Projection Technique (Average of 3,54 out of 5 with 5 being Very Easy) was easier than the Baseline Method (Average of 2.67 out of 5 with 5 being Very Easy).

5.7.7 How uncomfortable is each technique

When asked if they felt uncomfortable using the input device, only 2 (8,3%) out of 24 participants responded with "Yes" while using the Projection technique, which can be seen in figure 5.8. Comparatively, in figure 5.9, we can see that the number of "Yes" increased to 7 (29,2%). When asked about why, the participants mentioned that this had to do with how the hand was represented in the virtual world. As mentioned before, the LEAP Motion Device cannot detect the fingers which are occluded behind the input device while holding it. However, it can detect the thumbs correctly, and as such, they are represented well. The current implementation of the system sometimes has an unnatural representation of the hands, when the fingers are not visible to the LEAP Motion Device causing a lot of distractions for the users. Figure 4.4 exemplifies this problem. We can see that the fingers used to hold the phone clip the representation of the smartphone sometimes causing the screen to be occluded or harder to see.

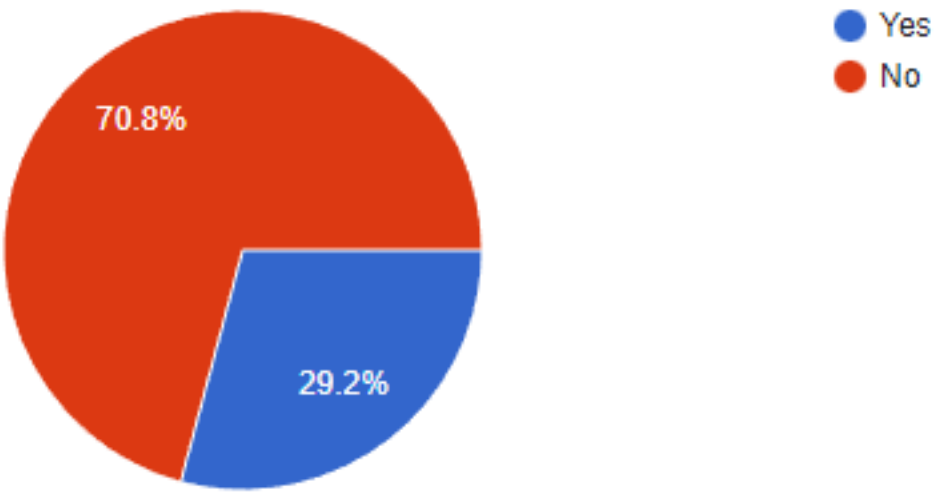

Figure 5.9: Did you feel uncomfortable using the Baseline Method?

5.7.8 Analysis of Qualitative Results

From the analysis of the previously shown data from the user questionnaires, we can see that most users preferred the Projection technique, classifying it as more intuitive and comfortable. Some users, however, preferred the Baseline technique, stating that it gave a better depth perception than the Projection technique. It is also important to note the feedback from the users that after a period of time using the Baseline technique with the same calibration, they could figure out the slight discrepancy between the real touches and the perceived touches in VR. As such, they started correcting their own touches.

5.8 Discussion

After analyzing subjective and objective data, we can combine the two. First of all, even though the Baseline technique proved to be faster than the Projection, it also proved to be more error-prone due to the calibration mistakes the devices are inherently subject to. Moreover, the feedback provided by the users clearly favored the Projection technique, with participants claiming it was simpler, more intuitive, and more comfortable to use.

When analyzing all the data related to the sizes of the targets, it is clear that the smaller the targets, the more errors the users committed. However, this difference is more noticeable when comparing the medium-sized grid with the big-sized grid rather than the medium-sized grid with the small-sized grid. This is in line with the user feedback that stated that the targets on the big-sized grid were harder to hit, especially while using the smartphone. Although our tests did not focus on analyzing the different types of interactive surfaces, we used two different types to test the solution's viability on both these devices. The data shows us that tasks such as text typing on

the smartphone could be very error-prone due to the number of errors recorded in the user tests. This high amount of errors could imply that tasks that rely on hitting small targets are not very feasible. In conclusion, after analyzing all the results, we can see that our solution, although not optimal in every context, was the preferred method by the users, especially regarding comfort and ease of use.

Chapter 6

Conclusions and future work

Current VR solutions have shown immense potential in helping humans in several activities, such as education or training and simulation in fields such as military and aviation. However, several downsides of the VR experience still can be experienced, such as fatigue using controllers and others.

The use of interactive surfaces as input devices can be a way to improve these aspects of the experience by allowing for different inputs to be done with little to no physical effort. These surfaces bring benefits with them, such as natural and interactive interaction, haptic feedback, and precise inputs. However, there are still some problems attached to their use in VR based on the current technology limitations related to hand tracking, making it so inputs can not be as precise as they are when not used for Virtual Reality actions. Therefore, there is a need to improve the use of these devices as input devices in VR to use their full capabilities better.

In this dissertation, we developed a technique to improve absolute inputs using interactive surfaces in Virtual Reality, ultimately aiming to improve VR experiences. Our developed technique, the Projection technique, was compared with another method called Baseline Method to perform absolute inputs in VR using interactive surfaces. The Baseline Method used a simple representation of the hands in the VE and had no processing of any data related to its position, using just the coordinates of the inputs on the surface for triggering inputs in VR. Although slower, we concluded that our method was more precise in several contexts and reported by users as more comfortable and intuitive to use than the other method. As such, results show that our technique can be suitable for incorporating interactive surfaces as input devices in VR to perform absolute inputs.

As mentioned above, the performance of our technique was not optimal in terms of the time users took to hit targets on the virtual screen. Something that could eventually improve the precision of the Projection Technique would be improving the calculation of the position of the input's projection on the virtual interactive surface in the VR environment. Our current calculation only considers the position of the fingertip provided to us by the LEAP Motion device. It does not consider the posture of the fingers, for example. The posture of the finger can significantly condition the actual point of contact, reducing the pre-existing discrepancy between the position in the

virtual and real world.

It is also important to test and explore better ways to determine which of the projections to use since our technique only accounts for the Euclidian distance between actual touches and projection positions.

Below, we present some new things that could be explored as well using this dissertation as a starting point. The Projection technique was only tested in the general context of hitting targets and not a more specific context, such as text typing, which is a task that could benefit a lot from these proposed new interaction techniques. Our tests explored hitting small targets, which is one of the challenges of text typing. Future work could explore this context in more depth.

These improvements can potentially improve the presented technique, which allows VR users to easily interact with a device that allows for a natural and precise interaction.

References

[1] Luis Afonso, Paulo Dias, Carlos Ferreira, and Beatriz Sousa Santos. Effect of hand-avatar in a selection task using a tablet as input device in an immersive virtual environment. In *2017 IEEE Symposium on 3D User Interfaces (3DUI)*, pages 247–248. IEEE, 2017.

[2] Verena Biener, Daniel Schneider, Travis Gesslein, Alexander Otte, Bastian Kuth, Per Ola Kristensson, Eyal Ofek, Michel Pahud, and Jens Grubert. Breaking the screen: Interaction across touchscreen boundaries in virtual reality for mobile knowledge workers. *arXiv preprint arXiv:2008.04559*, 2020.

[3] Sabah Boustila, Thomas Guégan, Kazuki Takashima, and Yoshifumi Kitamura. Text typing in vr using smartphones touchscreen and hmd. In *2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, pages 860–861. IEEE, 2019.

[4] Ruei-Che Chang. Textbox: Exploring text-entry on head-mounted displays.

[5] Tobias Drey, Jan Gugenheimer, Julian Karlbauer, Maximilian Milo, and Enrico Rukzio. Vrsketchin: Exploring the design space of pen and tablet interaction for 3d sketching in virtual reality. In *Proceedings of the 2020 CHI conference on human factors in computing systems*, pages 1–14, 2020.

[6] Travis Gesslein, Verena Biener, Philipp Gagel, Daniel Schneider, Per Ola Kristensson, Eyal Ofek, Michel Pahud, and Jens Grubert. Pen-based interaction with spreadsheets in mobile virtual reality. In *2020 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 361–373. IEEE, 2020.

[7] Jens Grubert, Lukas Witzani, Eyal Ofek, Michel Pahud, Matthias Kranz, and Per Ola Kristensson. Effects of hand representations for typing in virtual reality. In *2018 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, pages 151–158. IEEE, 2018.

[8] Jens Grubert, Lukas Witzani, Eyal Ofek, Michel Pahud, Matthias Kranz, and Per Ola Kristensson. Text entry in immersive head-mounted display-based virtual reality using standard keyboards. In *2018 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, pages 159–166. IEEE, 2018.

[9] Jan Gugenheimer, David Dobbelstein, Christian Winkler, Gabriel Haas, and Enrico Rukzio. Facetouch: Enabling touch interaction in display fixed uis for mobile virtual reality. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, pages 49–60, 2016.

[10] Pascal Knierim, Valentin Schwind, Anna Maria Feit, Florian Nieuwenhuizen, and Niels Henze. Physical keyboards in virtual reality: Analysis of typing performance and effects of avatar hands. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, pages 1–9, 2018.

[11] Jihyun Lee, Byungmoon Kim, Bongwon Suh, and Eunyee Koh. Exploring the front touch interface for virtual reality headsets. In *Proceedings of the 2016 chi conference extended abstracts on human factors in computing systems*, pages 2585–2591, 2016.

[12] Akhmajon Makhsadov, Donald Degraen, André Zenner, Felix Kosmalla, Kamila Mushkina, and Antonio Krüger. Vrysmart: a framework for embedding smart devices in virtual reality. In *CHI Conference on Human Factors in Computing Systems Extended Abstracts*, pages 1–8, 2022.

[13] Fabrice Matulic, Aditya Ganeshan, Hiroshi Fujiwara, and Daniel Vogel. Phonetroller: Visual representations of fingers for precise touch input with mobile phones in vr. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, pages 1–13, 2021.

[14] Manuel Meier, Paul Streli, Andreas Fender, and Christian Holz. Tapid: Rapid touch interaction in virtual reality using wearable sensing. In *2021 IEEE Virtual Reality and 3D User Interfaces (VR)*, pages 519–528. IEEE, 2021.

[15] Mark Mine, Arun Yoganandan, and Dane Coffey. Making vr work: building a real-world immersive modeling application in the virtual world. In *Proceedings of the 2nd ACM symposium on Spatial user interaction*, pages 80–89, 2014.

[16] Peter Mohr, Markus Tatzgern, Tobias Langlotz, Andreas Lang, Dieter Schmalstieg, and Denis Kalkofen. Trackcap: Enabling smartphones for 3d interaction on mobile head-mounted displays. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, pages 1–11, 2019.

[17] Guillermo Molina, Jesús Gimeno, Cristina Portalés, and Sergio Casas. A comparative analysis of two immersive virtual reality systems in the integration and visualization of natural hand interaction. *Multimedia Tools and Applications*, 81(6):7733–7758, 2022.

[18] Keunwoo Park, Sunbum Kim, Youngwoo Yoon, Tae-Kyun Kim, and Geehyuk Lee. Deepfisheye: Near-surface multi-finger tracking technology using fisheye camera. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*, pages 1132–1146, 2020.

[19] Daniel Schneider, Verena Biener, Alexander Otte, Travis Gesslein, Philipp Gagel, Cuauhtli Campos, Klen Čopič Pucihar, Matjazz Kljun, Eyal Ofek, Michel Pahud, et al. Accuracy evaluation of touch tasks in commodity virtual and augmented reality head-mounted displays. In *Symposium on Spatial User Interaction*, pages 1–11, 2021.

[20] Daniel Schneider, Alexander Otte, Axel Simon Kublin, Alexander Martschenko, Per Ola Kristensson, Eyal Ofek, Michel Pahud, and Jens Grubert. Accuracy of commodity finger tracking systems for virtual reality head-mounted displays. In *2020 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*, pages 804–805. IEEE, 2020.

[21] Jeongmin Son, Sunggeun Ahn, Sunbum Kim, and Geehyuk Lee. Improving two-thumb touchpad typing in virtual reality. In *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems*, pages 1–6, 2019.

[22] Jeongmin Son, Sunggeun Ahn, Sunbum Kim, and Geehyuk Lee. Effect of contact points feedback on two-thumb touch typing in virtual reality. In *CHI Conference on Human Factors in Computing Systems Extended Abstracts*, pages 1–6, 2022.

[23] Hemant Bhaskar Surale, Aakar Gupta, Mark Hancock, and Daniel Vogel. Tabletinvr: Exploring the design space for using a multi-touch tablet in virtual reality. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, pages 1–13, 2019.

[24] Spyros Vosinakis and Panayiotis Koutsabasis. Evaluation of visual feedback techniques for virtual grasping with bare hands using leap motion and oculus rift. *Virtual Reality*, 22(1):47–62, 2018.

[25] James Walker, Bochao Li, Keith Vertanen, and Scott Kuhl. Efficient typing on a visually occluded physical keyboard. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, pages 5457–5461, 2017.

[26] Fengyuan Zhu, Zhuoyue Lyu, Mauricio Sousa, and Tovi Grossman. Touching the droid: Understanding and improving touch precision with mobile devices in virtual reality.

Appendix A

Consent forms for the user tests

Declaração de Consentimento de Direitos de Imagem

No âmbito da realização da tese de Mestrado em Engenharia Informática e Computação da Faculdade de Engenharia da Universidade do Porto, intitulada **Improving Absolute Inputs for Interactive Surfaces in VR**, realizada pelo estudante Diogo Guimarães do Rosário, orientada pelo Prof. Daniel Filipe Martins Tavares Mendes e sob a co-orientação da Prof. Teresa Carla de Canha e Matos, eu abaixo assinado declaro que autorizo a filmagem da minha imagem, bem como a difundi-la no contexto de investigação acima mencionado.

A presente autorização é concedida a **título gratuito.**

Porto, __ de _____ de 20__

_____

(Participante ou seu representante)

DECLARAÇÃO DE CONSENTIMENTO

(Baseada na declaração de Helsínquia)

No âmbito da realização da tese de Mestrado em Engenharia Informática e Computação da Faculdade de Engenharia da Universidade do Porto, intitulada **Improving Absolute Inputs for Interactive Surfaces in VR**, realizada pelo estudante Diogo Guimarães do Rosário, orientada pelo Prof. Daniel Filipe Martins Tavares Mendes e sob a co-orientação da Prof. Teresa Carla de Canha e Matos, eu abaixo assinado declaro que compreendi a explicação que me foi fornecida acerca do estudo no qual irei participar, nomeadamente o carácter voluntário dessa participação, tendo-me sido dada a oportunidade de fazer as perguntas que julguei necessárias.

Tomei conhecimento de que a informação ou explicação que me foi prestada versou os objetivos, os métodos, o eventual desconforto e a ausência de riscos para a minha saúde, e que será assegurada a máxima confidencialidade dos dados.

Explicaram-me, ainda, que poderei abandonar o estudo em qualquer momento, sem que daí advenham quaisquer desvantagens.

Por isso, consinto participar no estudo e na recolha de imagens necessárias, respondendo a todas as questões propostas.

Porto, __ de _____ de 20__

_____

(Participante ou seu representante)