Development of a therapy game proof of concept using the Virtual Reality technology

Jose Ferrando Felix

Final Degree Work
Bachelor's Degree in
Video Game Design and Development
Universitat Jaume I

July 24, 2023

Supervised by: Zoe Valero Ramón, PhD.

To my grandfather, who always stood by my side until the very end

Acknowledgments

I would like to thank everyone who contributed directly or indirectly in this project.

To Albert, for trusting in the idea and lending his Virtual Reality set so that I could work on this project.

To Jose, for giving me the idea that would start it all.

To Gracia, Manuel and Hugo, for giving me their time to test the game and give their feedback.

To my Final Degree Work supervisor, Zoe Valero Ramón, PhD, for her contribution and feedback during the whole process of this work.

I also would like to thank Sergio Barrachina Mir and José Vte. Martí Avilés for their inspiring LaTeX template for writing the Final Degree Work report, which I have used as a starting point in writing this report.

Abstract

This document constitutes the Final Degree Work report of Jose Ferrando Felix, student in Video Game Design and Development.

The work consists of a proof of concept for a video game aimed at aiding patients in post-stroke rehabilitation through therapy gamification, serving as the first iteration of a medium to long-term project. In order to gamify therapy in the most realistic and organic way possible, Virtual Reality technology is utilized. The game presents a variety of minigames as exercise proposals that contribute to both the motor rehabilitation and neurorehabilitation aspects of the stroke recovery process.

Keywords — videogame, therapy, gamification, Virtual Reality, OpenXR, neurorehabilitation, motor rehabilitation

Contents

Chapter 1

Introduction

Contents

In this chapter, the motivations and the origins of the idea for this project are explained, along with the initially established objectives and the initial context in which the development of this project began

1.1 Work Motivation

Stroke is one of the leading causes of disability worldwide, affecting patients' mobility, limiting their daily life activities and leading to low overall quality of life. According to data from the Survey on Disability, Personal Autonomy and Dependency Situations published in April 2022 by the Instituto Nacional de Estadística (INE), more than 435,400 people live in Spain with Acquired Brain Injury, the leading cause being stroke, being the reason for more than 80% of cases. In Spain, 361,500 people have Acquired Brain Injury due to a stroke, representing an increase of 35,956 cases over the previous survey in the year 2008. Cranioencephalic traumatisms are the second most frequent cause of Acquired Brain Injury, with 73,900 people with Acquired Brain Injury as a consequence of a Cranioencephalic Trauma, even though a reduction in the number of cases of 16,620 with respect to 2008 [1].

Over the course of the last years, new technologies have emerged to help in stroke rehabilitation, such as robot-assisted training, non-invasive brain stimulation and Virtual Reality (VR) [2]. Even so, the lack of public resources and public places causes patients to seek help in private hospitals to start rehabilitation at a monthly cost of between 6.000 and 8.000 euros [3]. Due to the average wage of people from Spain, these prices are oftenly not even close to being affordable.

The idea behind this project is to give stroke patients another tool to reinforce their rehabilitation through video games, providing a different experience whose objectives are, among others, to make therapy more entertaining, and to lower the costs of post-stroke rehabilitation. To achieve this goal, it was concluded that the best way to do it would be using VR technology, since it is a booming technology with the potential to play an important role in rehabilitation [4]. It is important to keep in mind that the development of this project is a long-term idea, and therefore, the game version to be presented is a proof of concept. Although theoretical concepts have been discussed with experts in the field of medicine, the practical implementation and development have been carried out independently by the author. So, while the exercises are inspired by real rehabilitation activities, no bibliographic review or state-of-the-art research has been conducted to determine which techniques or exercises would be most suitable for the target users of this game.

1.2 Objectives

- Develop a video game with different minigames that can help patients in their rehabilitation by including activities that reinforce their cognitive and physical capabilities.
- Design a variety of minigames that have simple, yet entertaining and challenging mechanics for the players, so that they don't feel overwhelmed nor get frustrated when playing the game.
- Create a progression system, similar to a score system, so that players feel like they are progressing with their rehabilitation not only in the videogame, but also in real life.
- Learn to utilize the VR technology, thus making the therapy feel more realistic and organic.
- Carry out a proof of concept using VR technology and the proposed minigames.

1.3 Environment and Initial State

The idea arose at the time when the professors' Final Degree Work proposals came out. One of the proposals to be addressed consisted of developing of a serious video game using VR, meaning that it had to have some purpose beyond entertainment. This technology has always been a topic of personal interest, and this opportunity allowed the development of a project that was not only a means of entertainment, but also useful for people. After a discussion with a close relative specializing in the medical field, the potential of this technology to aid in post-stroke treatment was highlighted, as well as the potential for cost savings. The conclusion of this talk was the decision of the theme of the project.

After some research, it was decided that the platform for the project would be Unity 3D [5], not only because of its versatility, but also because it has a VR development library with many functionalities, both at a high and low level, supported by a wide community of VR developers: OpenXR [6], an open, royalty-free standard that simplifies the development of Augmented Reality and Virtual Reality (AR/VR) while covering development across a wide range of AR/VR devices. Finally, it only remained to establish which VR device would be used. A former college friend offered to lend me their device, the HTC Vive [7], to carry out the project.

In summary, the project involved the development of a VR video game where the Unity3D engine was used, with the help of the OpenXR library. Both the design and programming were done by the author of the work.

Chapter 2

Planning and resources evaluation

Contents

In this chapter, the followed planning to complete the project and the resources used to accomplish said purpose are explained.

2.1 Planning

This section shows the work that has been undertaken in order to carry out the project. It should be noted that the tasks were not carried out sequentially, but sometimes alternately or even in parallel. For better understanding, a Gantt chart is included to provide a visual reference of the work (See Figure 2.1).

- **Researching, understanding and information gathering (20 hours):** research for some articles that reaffirm the viability of the VR technology for therapy, conversations about rehabilitation and stroke therapy with a doctor, as well as a search for real exercises that serve as references for the minigames.
- **Game concept design (20 hours):** this refers to the early design of the game elements, such as the level designs, the mechanics within them, how these can gamify therapy and the progression system. This initial design serves as a foundation on which the rest of the game would be built as the project progresses.
- **Game development (180 hours):**
  - **Preliminary work (10 hours):** includes the creation of the project in Unity3D and the implementation of the OpenXR library, as well as the controls of the player using the VR input system.

  - **Main hub (10 hours):** the creation and programming of the main hub, where the player starts the game and can change some options or access to the minigames.
  - **First minigame(60 hours):** the creation and programming of the first minigame, which is inspired by the marshmallow roasting minigame from the videogame *Outer Wilds*.
  - **Second minigame (50 hours):** the creation and programming of the second minigame, based on first-person car racing games.
  - **Third minigame (35 hours):** the creation and programming of the third minigame, influenced by the *Simon Says* game.

- **Asset creation and search (5 hours):** use of 3D modelling programs to create objects or search for assets in the Unity Asset Store.
- **Game testing and feedback gathering (10 hours):** the testing of the game by patients who are in post-stroke therapy.
- **Project documentation (80 hours):** write the Final Degree Work report, as well as other necessary documents.

To understand this Gantt chart better, two periods of time can be appreciated. In the first half, the project flow went smoothly and had no noticeable issues. But as the second period started, around early April, things worsened. Due to the full-time internship, nearly half of the day was lost commuting, working, and returning. Combined with the workload of other courses, there was less and less time available to work on the project. This is why it can be observed that the project was delayed over time, extending it to almost July.

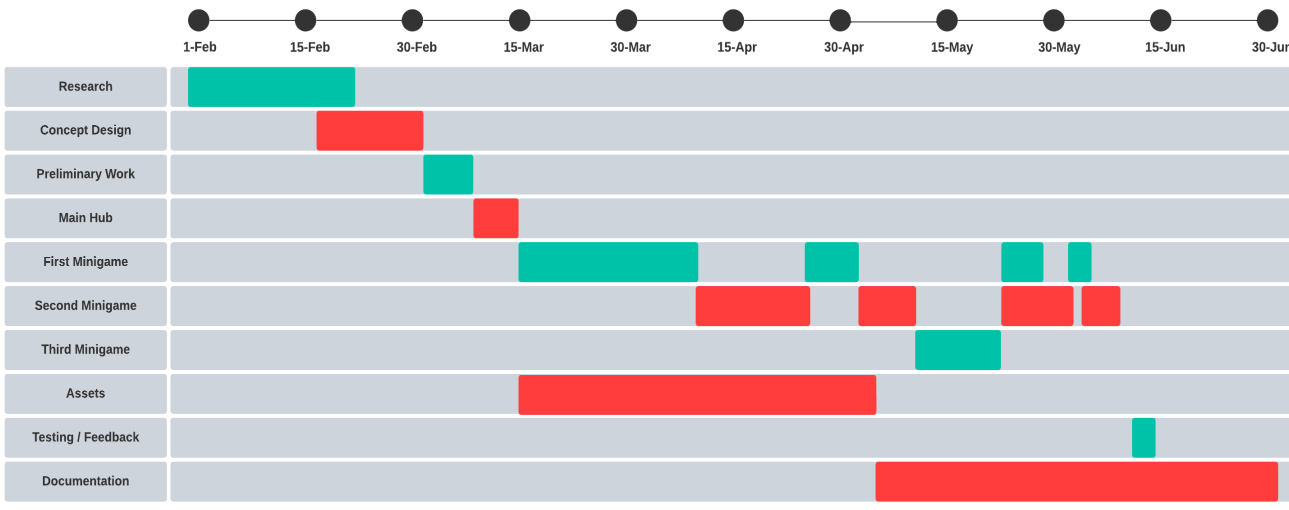


Figure 2.1: Gantt chart of the work done, made with Ludichart

2.2 Resource Evaluation

The assessment of project resources took into account the minimum requirements for using VR technology. Typically, when using a VR headset, minimum specifications for typical mid-range devices are recommended [8]. However, depending on the computer, it is possible to develop a VR project with requirements below the minimum. The laptop used in this case has a slightly lower graphics card than the minimum requirements, but the project can still be carried out without major issues.

- **Hardware:**
  - A computer that is compatible with VR technology. For this project, an MSI laptop with an Intel Core i7 7700HQ processor, 16GB of RAM, a GeForce GTX 1050Ti graphics card, and a 1TB HDD was used -> 1100€
  - HTC Vive, the VR set used to develop the project -> Free
- **Software:**
  - Unity 2021.3.0f1, the game engine used for the creation and development of the project -> Free
  - Visual Studio 2022, the program used to write the code for the project [9] -> Free
  - Blender 2.8, used for creating some of the 3D models in the project [10] -> Free
  - Unity Asset Store, the Unity asset store used to obtain most of the 3D models and other resources [11] -> Free
  - Overleaf, a web application for editing the document in LaTex format [12] -> Free
- **Others:**
  - Housing rent -> 600€
  - Invoices -> 150€
  - Internet service -> 35€

CHAPTER 3

System Analysis and Design

Contents

This chapter presents the game's operation, the requirements analysis, the design and architecture of the proposed work, as well as its interface design.

3.1 Game's Operation

Before explaining the requirements, the game's operation is explained for better understanding. Upon entering the game, the player is found in the main room. This room is the central lobby, and from it, the player can access all the minigames, configure certain game options, and exit the game. In this version of the project, the game has three minigames, which can be accessed through their respective portals. Each portal has a different color for every minigame The player can return to the central lobby.

Focusing on the minigames, each of them has three difficulty level: Easy, Normal, and Hard. The difficulty of the minigame can be selected inside the minigame before starting it. In order to clear the minigame, a series of requirements need to be met during the course of the game, such as achieving something a certain number of times or completing something in less than a certain amount of time. The minigames also have a scoring system, designed to encourage the player not only to meet the minimum requirements to unlock more challenging difficulties, but also to surpass themselves. At the end of the game, the score of the player is displayed, along with a screen that shows if they have met the requirements needed to beat the level.

Now, let's talk about each minigame. As mentioned earlier, this version of the project has three minigames. The first minigame aims to cook as many marshmallows as possible while burning the least amount. Each marshmallow must be cooked on a different fire, following different rules according to each woodfire. The idea behind this minigame was to create something that would help the patient strengthen their arm muscles, since in post-stroke therapy, there are exercises that aid in this by having the patient keep their arms suspended without resting them on any surface. This proposal posed a particular challenge as the mechanic had to be devised in order to gamify the concept of stretching the arms.

The second minigame consists of driving a car through an obstacle course while picking up different objects that give a score. The goal of this minigame is to reach the finish line in the shortest amount of time possible while also getting the highest score possible. The intention behind this minigame was to create a proposal that supported the neurorehabilitation aspect of post-stroke therapy by implementing a minigame that would work on both reaction capacity and sustained attention.

The third minigame is inspired by the game *Simon Says*, and it consists of following patterns or instructions. At the beginning of each round, a random sound is played, and the player must press the button corresponding to that sound. The goal is to get as many correct as possible, getting the highest score possible. This idea was thought to design a game that would enhance pattern recognition and memory skills.

3.2 Requirement Analysis

With the game's operation explained, both the functional and non-functional requirements can now be listed in this section.

3.2.1 Functional Requirements

The functional requirements that we can retrieve from the game's operation are the following:

- **R1:** the player can select level.
- **R2:** the player can move through the level.
- **R3:** the player can rotate the character.
- **R4:** the player can change settings of the game.
- **R5:** the player can select minigame difficulty.
- **R6:** the player can quit the game.

3.2.2 Non-functional Requirements

Subsequently, the non-functional requirements for this project are:

- **R6:** the game will be playable on PC.
- **R7:** the game will make use of low poly models.
- **R8:** the game mechanics of the game will be simple and easy to learn.
- **R9:** the game will be playable by anyone who is in possession of VR headsets compatible with OpenXR.
- **R10:** the game will serve as a proof of concept for post-stroke therapy.

3.3 System Design

In this section, the operational design of the system is presented through use case diagrams:

Requirement:	R1
Actor:	Player
Description:	The player selects a level by entering a portal
Preconditions:	1. The player must be in the main hub
Normal Sequence:	1. The player enters the game 2. The player moves to the portal of the minigame they want to play 3. The system loads the level assigned to that portal
Alternative sequence:	The system does not load the level, because the player has not moved to the portal

Table 3.1: Use case «Select Level»

Requirement:	R2
Actor:	Player
Description:	The player is able to move through the level
Preconditions:	None
Normal Sequence:	1. The player presses in any direction of the left VR controller pad 2. The player moves in the direction according to which zone they are pressing on the left trackpad
Alternative sequence:	The player does not move because they are colliding with something or not pressing the left trackpad

Table 3.2: Use case «Move»

Requirement:	R3
Actor:	Player
Description:	The player is able to rotate the character
Preconditions:	None
Normal Sequence:	1. The player presses in the left or right parts of the right VR controller pad 2. The player rotates in the direction according to which zone they are pressing on the right trackpad
Alternative sequence:	The player does not rotate because they not pressing the right trackpad

Table 3.3: Use case «Rotate»

Requirement:	R4
Actor:	Player
Description:	The player can change the settings of the game
Preconditions:	1. The player must be in the main hub
Normal Sequence:	1. The player goes to the main hub 2. The player moves in front of the panels corresponding to the settings 3. The player adjusts the settings of the game by interacting with the panels
Alternative sequence:	The player does not interact with the panels corresponding to the settings

Table 3.4: Use case «Change settings»

Requirement:	R5
Actor:	Player
Description:	The player can select the minigame difficulty
Preconditions:	1. The player must be inside a minigame
Normal Sequence:	1. The player goes to the main hub 2. The player accesses any minigame through their corresponding portal 3. The player adjusts the difficulty of the minigame by interacting with the corresponding panels
Alternative sequence:	The player does not change the difficulty of the minigame

Table 3.5: Use case «Select minigame difficulty»

Requirement:	R6
Actor:	Player
Description:	The player can quit the game
Preconditions:	1. The player must be in the main hub
Normal Sequence:	1. The player goes to the panel that allows to exit the game 2. The player presses the button to exit the game 3. The system quits the application
Alternative sequence:	The player does not quit the application

Table 3.6: Use case «Quit the game»

The previous functional requirements are included in the following use case diagram, along with other actions, like grabbing an object or pressing a button, which are the main controls of the minigames

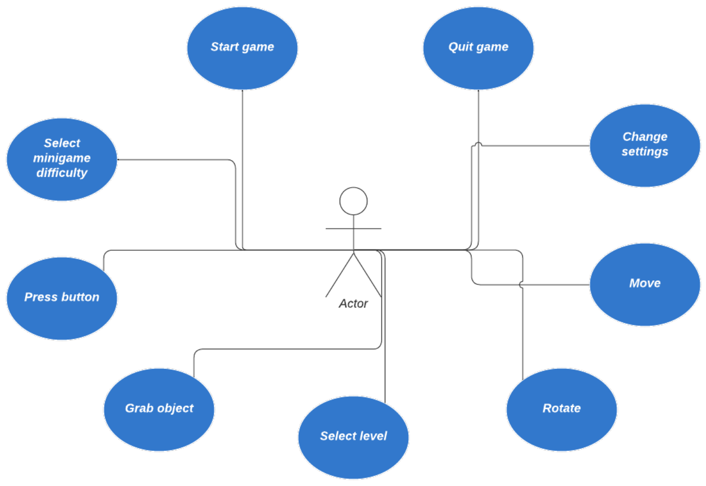


Figure 3.1: Case use diagram, made with Ludichart

3.4 System Architecture

The following system requirements have been extracted from both the Unity and the OpenXR plugin documentations. These are the minimum requirements that any computer needs to be compatible with the VR technology, and these must be met, otherwise the game cannot be played:

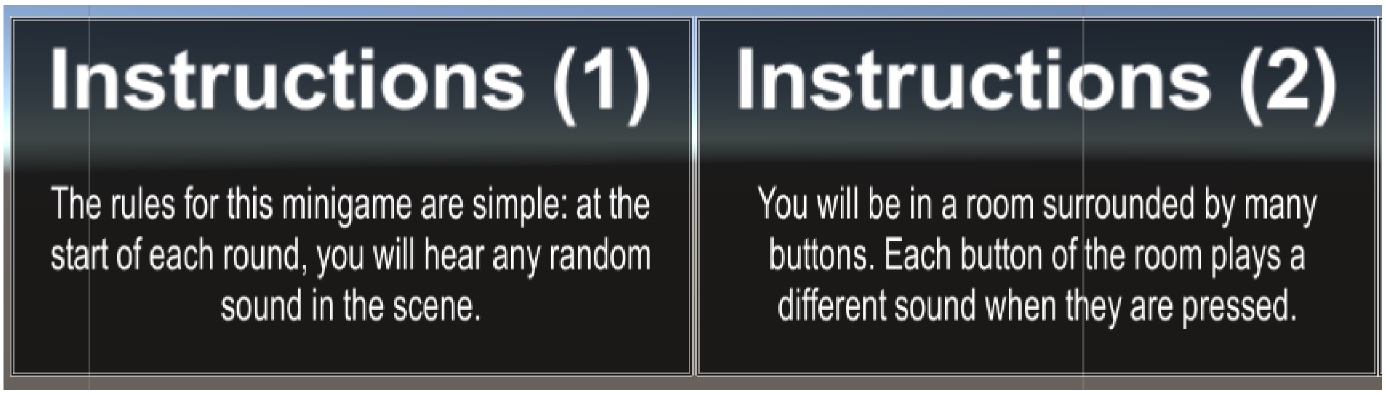
- At least, the operating system Windows 7 SP1+.
- At least, a CPU with x86 or x64 architecture with SSE2 instruction set support.
- At least, a NVIDIA GTX 1050Ti GPU, or an AMD Radeon RX GPU.
- DX11 Graphics API

3.5 Interface Design

Regarding the user interface section, there isn't much to comment on. The interface consists of text boxes located in both the central lobby and at the beginning of each level. Their purpose differs slightly, as there are three variations of these text boxes that serve different functions:

- The original text box, which aims to inform the player about the instructions on how to play or explain the rules of a mini-game. They can be found in both the central lobby and within each minigame (See Figure 3.2).
- The text box with a button, which is used to adjust the difficulty or restart the mini-games (See Figure 3.3).

The peculiarity of the second text box type is that they can be interacted by using your own hands, using the *XR Poke Interactor* (explained later in the section 4.1.6). In the case of the text box with a slider, it simply calls a function within the *Game Manager* that updates the game volume according to the value on the slider. As for the text boxes with buttons, they call a function witihn the *Minigame Manager* that updates the minigame's difficulty.


Instructions (1)

The rules for this minigame are simple: at the start of each round, you will hear any random sound in the scene.

Instructions (2)

You will be in a room surrounded by many buttons. Each button of the room plays a different sound when they are pressed.

Figure 3.2: Text boxes used to inform the player about to play a minigame

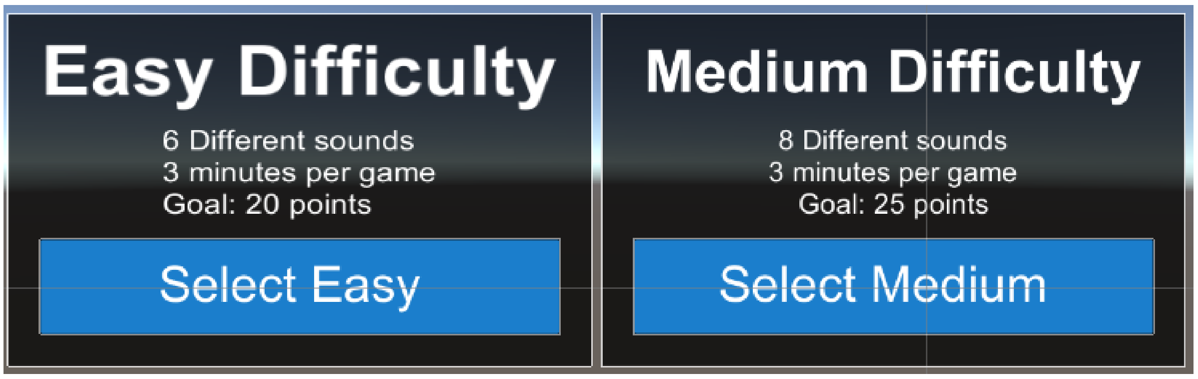


Figure 3.3: Text boxes used to adjust the minigame's difficulty

Chapter 4

Work Development and Results

Contents

4.1 Work Development

This chapter provides an explanation of how the project has been developed from beginning to end, alongside an assessment of the results and the feedback given by real patients, the supervisor, or even the author.

4.1.1 Preliminary Work

The first step in the project was to find a tool that would allow working with VR technology in Unity. After conducting research online, there were two options to choose from, considering that the VR device to be used in this project was the HTC Vive. On the one hand, there was the option to use the *SteamVR* [13] library, and on the other, there was the option to use the *OpenXR* library. SteamVR is both a digital platform and a plugindesigned for developing and distributing VR video games. Focusing on the latter, the plugin comes with plenty of implemented functionalities that help the developers create many games that are intended to be released on the digital platform, but the installation of their platform is required. Although the SteamVR library was a really good option to work with, selling this project was not initially a part of the project planning since one of the principles of the project was to help the people that needed it. So, the decision was made to look for another alternative. Unity works with OpenXR, so their library can be downloaded within the Unity Asset Store. OpenXR, unlike SteamVR, offers a unified API that allows developers to work on compatible applications across different platforms or devices, and does not require any other programs installed in the computer to work.

This second proposal sounded more appealing and versatile, as there may be VR devices in certain hospitals that are not necessarily the same as the HTC Vive, such as the Oculus Rift or the Oculus Meta Quest headsets, yet they can still be compatible with the project. Therefore, the decision was made to use OpenXR as the plugin to develop the project. Once the plugin was implemented in Unity, a series of tests were conducted to ensure everything was functioning correctly. It was now time to start working on the game.

4.1.2 Implementing Locomotion System

Once the project started, it was decided that the first thing to do was to implement player movement. With VR technology, three key aspects are considered when developing basic player movement: how the character moves, how the camera moves and how the character rotates. Typically, when a person plays conventional video games, there are many ways to address the above issues, and they are approached differently to accommodate the intentions of the developers and the needs of each game. For this project, it was decided to address the issue the following way: movement is performed by using inputs on the left controller of the VR set, the in-game camera represents the player's gaze, moving and rotating based on the headset, and rotation is done by using inputs on the right controller of the VR set.

So, to achieve all of this, the first step was to add an object to the scene and attach a script called *XR Origin* to it. This script transforms objects and trackable features to their final position, orientation, and scale in the Unity scene. It was initially estimated that setting up the project to represent the user's gaze through the headset would take some time. However, it was as easy as adding the XR Origin component to the object and assigning it a camera. This step was completed without any major issues. For movement, three additional components were added to the object. The first component is the *Locomotion System*, which receives user requests to move the object. The second component is the *Continuous Move Provider (Action Based)*, which listens to user input for movement. Lastly, the *Character Controller* component is added, which handles movement parameter adjustments such as character speed. By passing a list of inputs provided by the OpenXR plugin to the second script, the movement was essentially completed. The player could now move using the trackpad on the left controller. Regarding character rotation, there are two ways to rotate in VR: the first one is to perform a gradual rotation using the *Continuous Turn Provider (Action Based)* script, while the second way is to use the *Snap Turn Provider (Action Based)* script, where the player rotates a specific number of degrees. For the sake of comfort and to avoid potential patient discomfort, it was decided to use the first script. It was added to the object and assigned the corresponding input list from the right controller's trackpad.

After a quick test, it was found that everything seemed to work almost perfectly. The only issue encountered was related to the Character Controller. When the Character Controller was added, the object acquired a Capsule Collider, which is a cylindrical hitbox that allows the character to collide with other elements in the game, preventing it from passing through walls, for example. However, the capsule collider did not update its position or height while moving in the scene. This could be problematic when the player needs to crouch to overcome an obstacle, for instance. Fortunately, the solution was already provided by the plugin developers, and it did not require much research to find out that adding another component to the object, called *Character Controller Driver*, could fix this issue.

Once the character movement controls were established, it was necessary to implement the interaction controls. In VR, interactions involve grabbing and being grabbed. The character's hands should be capable of picking up objects in the environment. On the other hand, objects that can be interacted with need to communicate to the hands that they can be grabbed. The

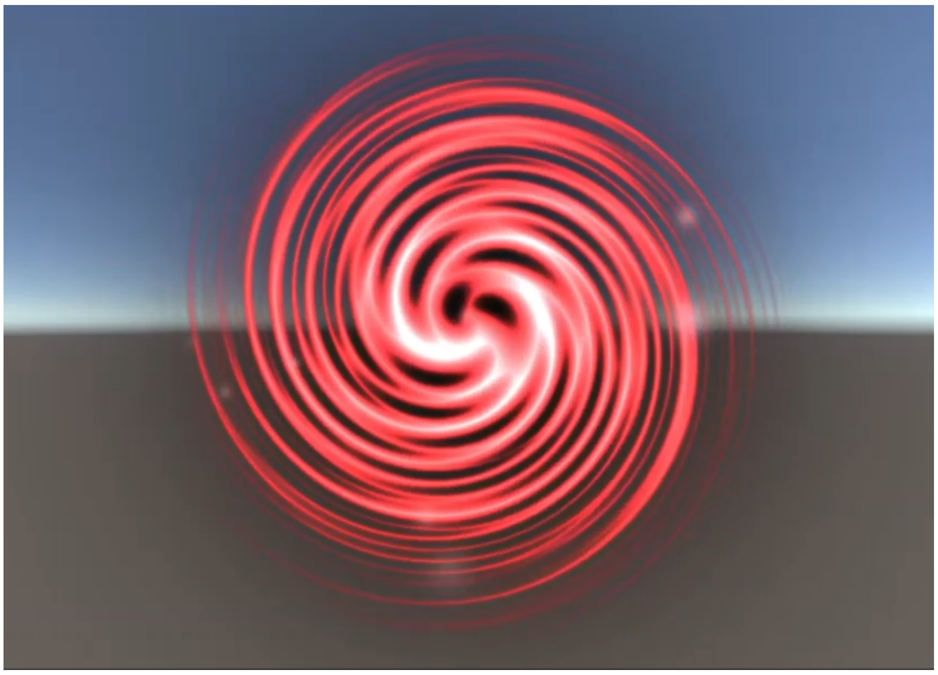


Figure 4.1: Final aspect of the portals

solution provided by OpenXR for this involves using *Interactors* and *Interactables*, which are scripts that enable these actions. When the player grabs an interactable object, it is placed in the character's hand. From now on, the *XR Direct Interactor* component is always present in the hands, which allows them to grab objects when the player presses the grab button and the hand is close to the object. The objects that the player interacts with have some form of the *XR Interactable* component attached to them.

4.1.3 Level Transitioning

Once it was confirmed that this idea worked, the character controls were ready to start developing the minigames. However, before diving into the minigame development, the method of transitioning the player to the minigames needed to be considered. After exploring various ideas, it was concluded that using portals, similar to those seen in science fiction, could create an interesting experience. To create these portals, Unity Shader Graph [14] was employed. Without delving into the specific details, the final result was achieved by experimenting with a *Worley* noise filter. The resulting shader could be applied as a material to an object. By adding a particle system, the final result looked quite appealing (See Figure 4.1):

Experimenting with *Unity Shader Graph* was quite enjoyable, so it was decided to continue creating new shaders to have a unique portal for each minigame. However, after creating a couple more shaders, the results were not sufficiently satisfying. Although they were not removed from the project, they were set aside in case they could be useful in the future. It is not uncommon for creative experimentation to yield mixed results. In this case, the decision was made to focus on other aspects of the project while keeping the shaders available for potential future use.

Later, a script called *Portal Collision* was added to the portals to facilitate the level transition. Each portal was assigned a unique identifier corresponding to the scene it should transport the player to. While the level change was functioning correctly, the transition felt abrupt and could potentially cause discomfort for the player. To address this, fade-in and fade-out transitions were implemented when exiting the current scene and entering the next one. The main difference in creating this transition was the approach used. Instead of using a canvas, a quad was positioned in front of the player's camera. Normally, this quad remains transparent and without colliders. By manipulating the alpha component of the quad, the desired transition effect could be achieved. This approach provided a smooth and immersive transition between scenes.

4.1.4 The First Minigame

In the following paragraphs, the different minigames are explained. Starting with the first one, the inspiration for this minigame came from the video game *'Outer Wilds'*. In this game, the user can roast marshmallows over a campfire, a mechanic that has some relevance to the game's narrative. So, the proposal to strengthen arm muscles consists of a marshmallow roasting minigame.

The initial idea for the minigame was that the player had to roast as many marshmallows as possible within a given time limit. To do this, the player would take marshmallows from a bag, skewer them on a stick, approach the campfire to cook them, and prevent them from burning. It was a somewhat simple proposal, but one that would serve the exercise purpose. Before delving into the development of the core mechanics, it was thought to create a suitable atmosphere for the minigame: a location set in the middle of nature, in the middle of the night, under the moonlight, seemed like an ideal scene for the minigame. To achieve this effect, a terrain was sculpted in Unity, the Skybox was changed, and an appropriate ambience was created for the purpose. After that, the marshmallow, the stick, and the campfire with which the marshmallow would be cooked were added in chronological order. The campfire was taken from the *Low Poly Fire* [15] package in the Unity Asset Store, while the others were modeled by the author (See Figure 4.2). The marshmallow was given a script called *Marshmallow Script*, which would be responsible for cooking it when it came into contact with the campfire, using the *OnTriggerEnter* method. To create the effect of cooking, the *Lerp* method was used, which is a linear interpolation between two values. This method can be used to interpolate between colors, so three colors were added: raw marshmallow, cooked marshmallow, and burnt marshmallow. The transition between a raw and cooked marshmallow would be a slow transition. Once cooked, if the player did not remove the marshmallow from the campfire, it would be overcooked, resulting in it getting burned and losing points. On the other hand, a collider was added to the end of the stick. This collider would act as a trigger, so when the tip of the stick passed close to the marshmallow, it would attach to the stick, simulating the action of skewering the marshmallow. This effect would be achieved using the *Attach Script*. As for the bag (See Figure 4.3), a script called *Spawner* was added to it, which would create an instance of the marshmallow every time it was picked up by either of the player's hands.

With this, the game's core mechanics were already in place: the player would pick up a marshmallow from the bag, place it on the stick, and cook it, preventing it from burning. This was what was shown to the supervisor initially. After receiving her feedback, the importance of encouraging the use of both arms throughout the game was highlighted, as it would not make much sense to have an exercise to strengthen arm muscles that didn't involve both arms.

This led to the second iteration of the minigame development. After some time to brainstorm ideas, it was discovered that the fire asset package included not just one, but five different

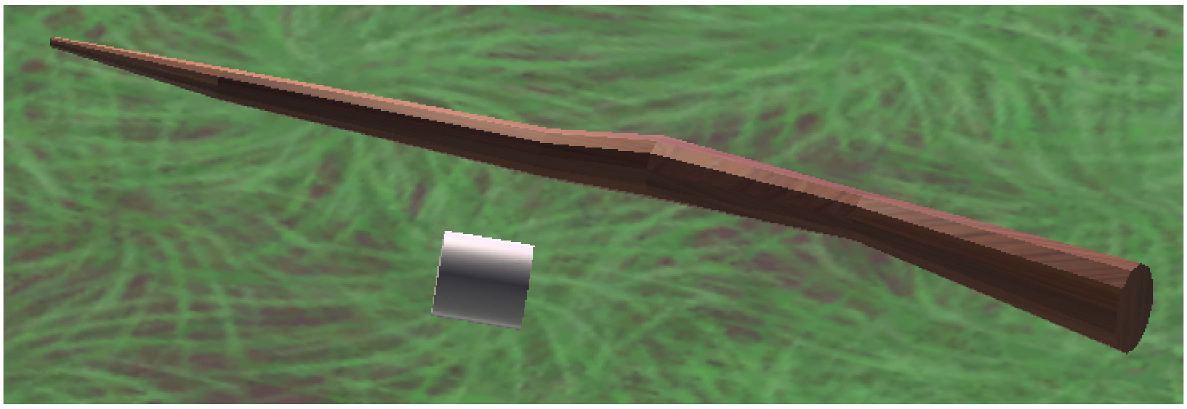


Figure 4.2: The stick and the marshmallow

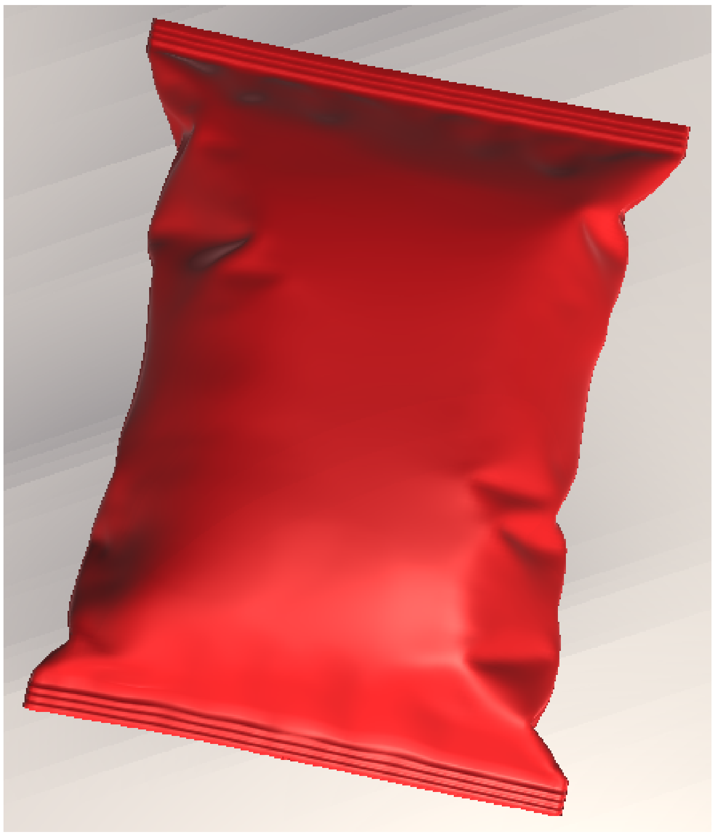


Figure 4.3: The bag where the marshmallows are spawned

fireplaces, each with a different color. This sparked the idea for the new approach: instead of having just one fireplace, there would be five, each with a unique cooking mechanic:

- Orange Fireplace: The regular fireplace with no special mechanics.
- Green Fireplace: The fireplace would cook the marshmallow at a different speed each time a marshmallow was cooked in it.
- Blue Fireplace: The fireplace would quickly cook the marshmallow and then burn it, but then it would reverse the process, uncooking it, constantly alternating between the states.
- Pink Fireplace: The marshmallow could only be cooked if the stick was held with the left arm.
- Purple Fireplace: The marshmallow could only be cooked if the stick was held with the right arm.

Implementing the mechanics for the blue and green fire pits was not very difficult. It only required tweaking some parameters based on what was done for the central fire pit. The challenge arose with the pink and purple fire pits. Detecting which arm was holding the stick proved to be quite complicated, as there were no tutorials or readily available resources online for implementing such a mechanic. After nearly a week of trial and error, arm detection was achieved using a method that is currently considered obsolete, according to the documentation. This method, implemented in the *Gettting Grabbed* script, involves identifying the interactor object (of type *XRBaseInteractor*) using the *OnSelectEntered* method from the OpenXR plugin. From there, the name of the object containing the interactor can be compared to determine whether the object is being grabbed by the left or right controller. While the arm detection currently works, there are plans to review the implemented methodology and redo it using an updated version supported by OpenXR.

Once it was possible to detect which arm was grabbing the object, this information was stored in the *Stick Script*, which was originally designed for other purposes but currently only stores this information. This variable is then passed to the fire pits and checked alongside the marshmallows. This proposal still has a design flaw: each fire pit has a different mechanic, but nothing is stopping the player from cooking the marshmallow in their preferred fire pit and ignoring the rest. To address this issue, it was suggested to assign colors to the marshmallows, specifically the same colors as the fire pits, so that each marshmallow must be cooked in its respective fire pit. To further prevent the player from only playing with marshmallows of one color, the bag that generates the marshmallows was modified to randomly generate them, deciding the color of each marshmallow at the moment they are instantiated in the scene. With all of this, the marshmallow roasting mechanic was implemented.

The next things to do were the scoring system and the timer. For the sake of simplicity, the scoring system works this way: if the marshmallow is cooked properly and the player eats it by grabbing the marshmallow and pressing the trigger button from the controller, they will add a hundred points to their score. However, if the player exceeds the cooking time, thus burning the marshmallow, instead of adding points to their score, it will substract fifty points. If the marshmallow is still uncooked, the player will not be able to eat it, so it will not add nor substract any point from their score. For the timer, a circular slider was used which empties during the course of the minigame. It also displays the remaining time inside the object (See Figure 4.4).

Once the timer drops to zero, the game will end, and it will show the stats the player achieved during the session. In the left canvas, the points scored, the current record and if the player has beat their record is displayed. In the right canvas, stats like the amount of cooked and burnt marshmallows are displayed, along with a text that tells the player if they have beat the game in that difficulty level. From this point, the player can decide wether they want to restart the minigame by pressing the button from a panel below the canvas, or leave the minigame by walking back through the portal, being the system used in the three minigames when they are finished.

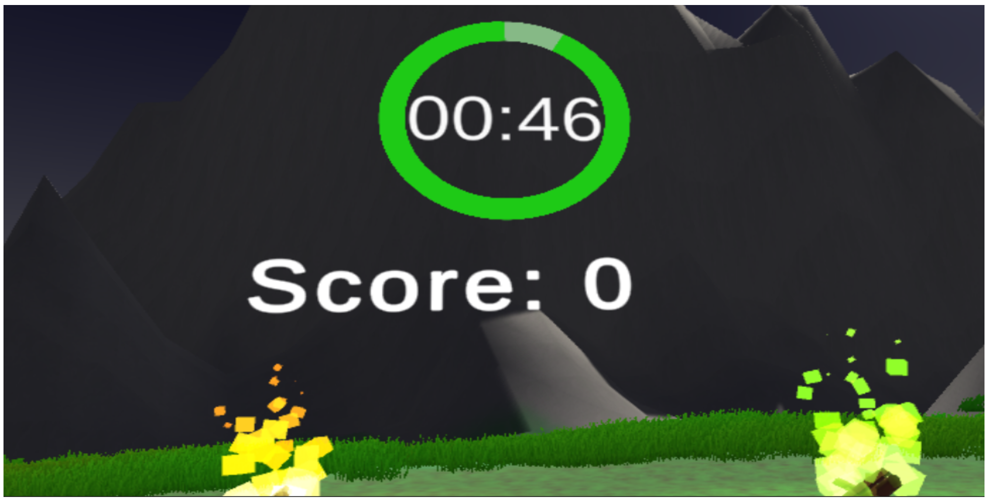


Figure 4.4: A screenshot of the time left and the current score of the player

4.1.5 The Second Minigame

For the second minigame, after considering several options and based on the author's preference, the decision was made to create a driving minigame. The concept of this minigame was for the player to drive a car and reach the finish line while avoiding collisions with obstacles in the environment. The player would be pursued by some malevolent entity, and if it caught up with the player, they would lose the game, creating the typical obstacle race scenario.

The implementation of the minigame began with the car. It was imported from an asset in the Unity Asset Store, specifically from the *Car'Toon: The Sport Car with interior* package [16]. This car was empty inside, making it ideal for the desired experience of the minigame, as it allowed for a driver's perspective (See Figure). The car also came with a set of scripts that added more realistic physics to the driving, but the input system used in these scripts was the traditional one, making it incompatible with the OpenXR input system. This was the first obstacle encountered in this minigame, so it was necessary to edit the scripts to make the car work and, at the same time, be compatible with OpenXR. The first part of resolving this problem involved a rather simple idea that allowed for simplifying the game. Instead of using an input for the car to move forward, the car would automatically move forward, allowing the player to focus on the other control. After some adjustments in the *Car User Control* script, the car was able to move forward.

The second obstacle was to make the car turn. Having the car turn by pressing a button on the controllers was an easy solution to implement, but considering the concept that the player is driving a car, it was a less realistic and immersive outcome. The author had a clear vision of what they wanted to achieve, which was to make the player turn the car as anyone would in real life: by turning the steering wheel. However, as the saying goes, easier said than done. This task

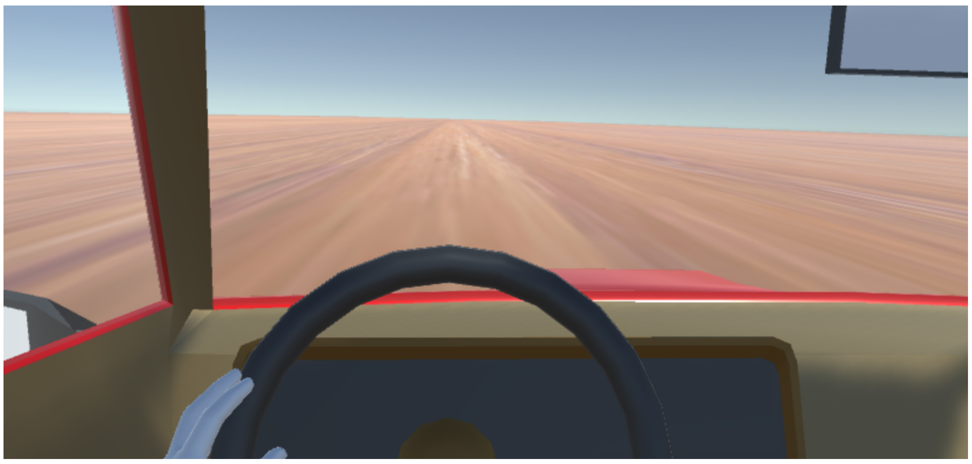


Figure 4.5: Player's point of view in the driving minigame

consumed more time than expected, but eventually, an acceptable implementation was achieved. A Hinge Joint component was added to the steering wheel object, which restricts the movement or rotation of an object to mimic the behavior of a hinge. This allowed the steering wheel to rotate only along a specific axis. Then, invisible colliders were added to the steering wheel, which is what the user actually grabs with their hands. While it may sound counterintuitive, this approach improved the physics of the steering wheel.

Initially, with this setup, it would be possible to manually turn the steering wheel. However, another problem arose: as mentioned earlier, when the player grabs something, the object they are grabbing is automatically placed in their hand. So, after some more time, the issue was resolved by creating a variation of the *XR Grab Interactable* script. This new script is called *XR Offset Grab Interactable*, and it essentially prevents the object from being placed in the player's hand, allowing it to stay in its original position.

After presenting this work to the supervisor, the concept of the minigame was discussed. While the idea of a chase was not necessarily bad, we agreed that if the player lost the game, despite the controls being simple, it could lead to feelings of frustration and potentially affect their mood, ultimately demotivating them during the rehabilitation session. Therefore, it was decided to remove both the chase element and the condition of defeat from the game design. After the meeting, some ideas were brainstormed to add some complexity to the game and create an incentive for the player to continue playing. It was decided to introduce a scoring system divided into two parts: object collection and a timer. This would motivate the player to achieve a high score by collecting objects and encourage them to reach the goal in the shortest time possible.

Regarding the object collection, the objects would be gems (See Figure 4.6). The gem models were imported from the *Low poly Gems* package in the Unity Asset Store [17]. Each gem would have an associated value, meaning that some gems would be worth more than others. These gems would be scattered throughout each track, and when the player drives over them with the car, they would be collected, adding their value to the total score of the game, which can be seen in the copilot seat of the car. The difference in gem values influenced the design of the tracks, incorporating branching paths with different gems at each exit. This would challenge the player's decision-making skills within a limited time frame, forcing them to choose which path would yield the greatest benefit. As for the timer, it actually functions as a cronometer rather than as a timer, located above the stick shift of the car. This way, the player can check the time they are spending on the track.

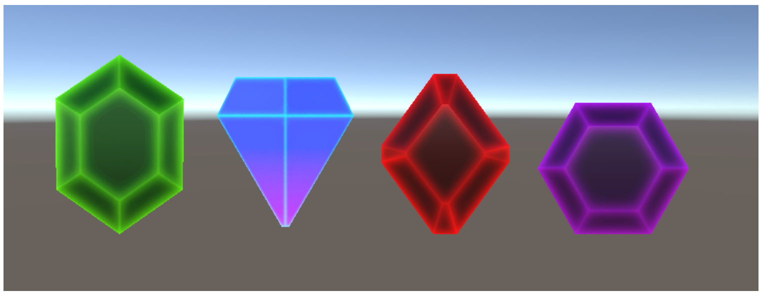


Figure 4.6: A screenshot of the gems used in the minigame

When they reach the end of the track, the player will be taken out of the car, so that they can check more comfortably the stats achieved in the session, being also given the option to restart the minigame or leave.

4.1.6 The Third Minigame

For the third minigame, the mechanics are the simplest of all. The concept for this minigame was clear from the beginning: a button pressing game. Drawing inspiration from *'Simon Says'*, the game would involve playing a random sound in the scene, and the player would have to press the button associated with that sound. Multiple rounds would occur within a set time limit. To begin, the button functionality was implemented. The button consists of three parts: the base, the visual part of the button, and the interactable part of the button. When the player presses the interactable part, the visual part is updated using the *Button Follow Visual* script. When the button reaches its lowest position, the appearance of the button changes to provide visual feedback that it has been pressed (see Figure 4.7), and to execute the code responsible for playing the associated sound. To implement the sound playback mechanic, at the beginning of each round, the minigame manager randomly selects an index from a list of sounds and plays the sound associated with that index. Then, that index is stored as a variable to compare with the button pressed by the player. When the player presses a button, the sound of that button is played, and the button's identifier is sent to the manager. If the button identifier and the index stored in the manager match, the player earns a point. However, if the player presses the wrong button and the identifiers don't match, the player loses a point. In either case, after updating the score displayed in the bottom left corner of the screen, the game proceeds to the next round, playing another sound.

Before starting the game, the player has to cross a door after selecting any difficulty. This will take them to the stage corresponding to the difficulty level they have chosen. To give a little help to the player, before starting the minigame they can check the sounds from every button by pressing them. Doing so will not be punished since they have not started the minigame yet. To begin the session, they will have to press the button that is located at the center of the stage. The type of sounds vary depending on the difficulty level, being less clear to recognise or differentiate as the difficulty increases, while adding more sounds to the stage so that the player has to memorise more buttons. It was thought that giving them the chance to play a little with the buttons to familiarize themselves with the sounds was a good idea for the possible inconveniences that the sounds on the higher difficulties could cause.

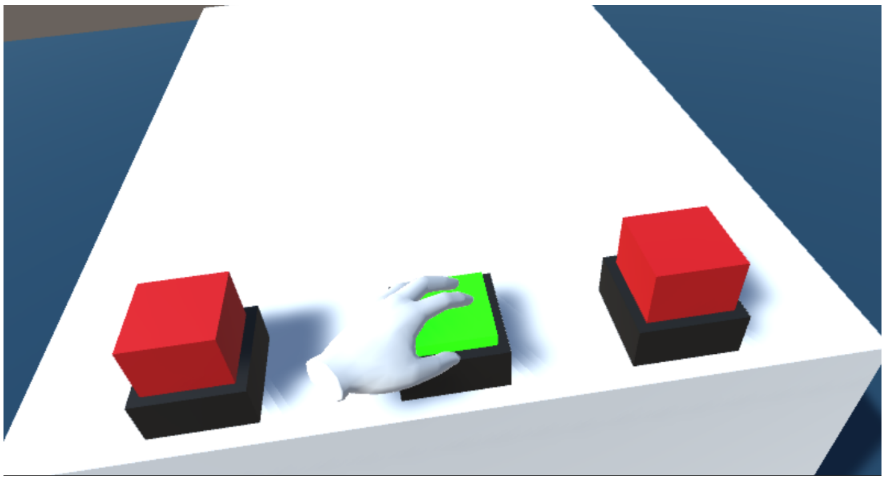


Figure 4.7: The buttons turn green when they are pressed

4.1.7 The Testing

After the first iteration of game development, an important part of the project arrived. With the project's first prototype practically done, it was time to contact the hospital, find patients, and organize a testing session. This part was considered one of the most crucial aspect of the project, as the feedback that both patients in the rehabilitation process and the doctors themselves could provide was, for the author, the ultimate goal of the project. Any contribution made, no matter how minor, would be considered highly valuable and crucial to continue developing a game that could help people who have suffered a stroke.

It should be noted that fortunately for this case, the process of contacting and organizing was quite informal, as a close family member was part of the medical team at a hospital. This relative was the person with whom the conversation took place and the idea for this project originated. They took it upon themselves to seek out patients among their own caseload who might be interested in being part of the project's testing, and after some time, they put the author in contact with three patients. This contact was indirect, but it involved sending the patients an informal letter explaining the author of the project, what the project was about, what would be done during the testing session, and asking if they would be willing to contribute their input to the project. All three individuals agreed to be part of the experience, so calmly, discussions began with the doctors regarding the patients' availability, reserving a space for the trial, the necessary equipment, as well as other more trivial matters.

On the day of the testing, the author went to the hospital with the VR device. After meeting with the two supervising doctors and having a lengthy discussion, they guided the author to the reserved room. Once everything was set up, we were informed that the condition of one of the patients had worsened. This meant that the testing session would be conducted with two patients instead of three. Fortunately, the author was later informed that the patient in question is currently in stable condition.

After the first patient arrived, a conversation was held to create a relaxed atmosphere and get to know the person better. Then, she moved on to the game testing. Initially, the patient had difficulty adapting to the controls since it was a new experience for her. However, with the help of the doctors and the author, and after some time, she was able to get started with the game. The patient was explained how to play each minigame, and a demonstration was even given to ensure she understood the mechanics. She mentioned that it didn't seem difficult to her.

In the first minigame, the patient encountered several things that bothered her. When she tried to pick up a marshmallow from the bag, sometimes it would roll off the table and fall to the floor, preventing her from grabbing it. It was explained to her that this occurred because during the development of the minigame, physics such as gravity were added to prevent the object from floating in the air when generated. It was mainly done to add a realistic touch, but it was understood that it could cause inconvenience during gameplay. Also, during the cooking phase, the patient asked how she would know when the marshmallow was cooked. She was informed that it would be cooked when the marshmallow turned gray. However, she still seemed unsure, so a quick change was made to make the cooked marshmallow's color golden instead of gray. She mentioned that it was easier for her to differentiate it that way.

In the second minigame, the patient didn't seem to have much difficulty driving, even after admitting feeling a bit exhausted from the first minigame. After explaining how the minigame worked, she asked if there was a way to apply braking to the car because she didn't want to crash it. She was informed that there were no brakes, but if she felt it would be beneficial, it was a change that could be made. During the testing, the patient was asked if she could recognize the gems she had to collect, to which she replied that she believed so. She was then asked if making the gems shine would help her recognize them better, and she responded that if it allowed her to see them from a distance, then probably it would.

Regarding the third minigame, the patient didn't seem to have any apparent difficulty during the course of the game. She even mentioned that she was having fun, despite making mistakes many times due to her "aging memory."

After completing the testing with the patient, she was asked about whether she found it difficult or enjoyable, and if she would like to undergo rehabilitation in this way. She mentioned that the minigames were quite simple, but she found it more challenging to adapt to using the VR headset than the actual game. However, she expressed that it was an experience she enjoyed a lot, and she would like to continue testing the game if it progressed further. After some more informal conversation, she was escorted back to her room, and the second patient was brought in.

The procedure with the second patient was basically the same, having a conversation to get to know them better, explaining the project, and then getting started. In the first minigame, the patient was initially amazed by the scenario, but that feeling changed when he started looking upward. He mentioned that he thought he had broken the game, and he was asked to briefly remove the headset. The author discovered that, for unknown reasons, there was a strange effect occurring when looking upwards, which was quite bothersome. However, it was observed that the effect didn't occur in the rest of the scenes. The trial was paused for a few minutes, and a temporary fix was implemented by changing the Skybox of the scene to a different one. Apparently, this resolved the issue, so the headset was returned to the patient to resume the trial. Even with the changes made during the previous session, the patient still seemed unsure of when the marshmallow was ready. This feedback was taken into account, and it was considered important to come up with something to clearly indicate when the marshmallow was cooked.

Without much else to add about the first minigame, the patient began the testing of the second one. The mechanics and how the minigame worked were explained to him, and when it came to the part with the gems and scoring, the patient asked if he could preview the gems in advance. The gems were shown directly from the asset's prefabs, but to avoid having to do this procedure each time, the author came up with an idea to prevent future confusion. The patient was also asked, after the gems were presented, if he believed he would recognize the gems better from a distance if their brightness was increased, to which he confirmed. As a comment from the patient, he mentioned that he enjoyed the experience of driving again since, due to his condition, it was impossible for him to drive a car. This made it his favorite minigame out of the three tested.

Regarding the third minigame, the patient also seemed to have a pleasant time. He appeared to fare better than the previous patient, but it is important to remember that each patient's situation and condition are different, so it is necessary to consider individual factors.

After completing the testing session with him, the same questions were asked to the second patient. He responded that it had been a rejuvenating experience (likely because he was able to drive again), that he had a good time, and although he wouldn't do the entire rehabilitation using VR, he wouldn't mind trying other rehabilitation methods like this occasionally. After talking a bit more with him and accompanying him back to his room, the author had a discussion with the doctors. They were asked for their opinions on the proposed minigames, and in summary, they responded:

- The first minigame appears to be a good support for muscle reinforcement exercises for the arms, and the gamified approach is original, but they cannot make any definitive claims without scientific evidence to support it.
- The second minigame has an entertaining concept, but of the three minigames, it deviates the most from the aspects worked on in neurorehabilitation. However, with some work, it could become a good proposal.
- The third minigame could be used effectively for memory exercises as it currently stands, and it seems that the patients have enjoyed it quite a lot.

After some more time discussing the patients's feedback and the author's proposals for improving the game with the doctors, the testing session was finally concluded. Undoubtedly, it was an experience that will be hard to forget.

4.2 Results

After the testing session at the hospital, the first iteration of the project can be considered concluded. Looking back, the work done has diverged significantly from the author's initial ideas, probably for better or for worse.

Using VR and learning about this technology has been a success. Regardless of the simplicity of the minigames proposed in the conceptual trial, the organic and realistic interaction provided by this technology could not have been achieved otherwise, making this objective accomplished. Furthermore, since this project has involved both patients and specialized doctors, receiving firsthand feedback from the people for whom this work is intended adds value to the project, which is highly appreciated. They have been able to demonstrate that, to a greater or lesser extent, the minigames designed for their rehabilitation are enjoyable, sparking their intrinsic motivation to continue trying the game. They do not feel frustrated when playing the game, and above all, it can be beneficial for their recovery.

On the other hand, due to various external factors to the project, the aspect of progression could be improved and polished, but it is nothing that cannot be addressed in the next stage of the game's development.

Chapter 5

Conclusions and Future Work

Contents

In this chapter, the conclusions of the work developed and its future extensions are shown in order to provide a better understanding of what will happen with the project.

5.1 Conclusions

This project has been one of the most bittersweet experiences of my university career, both professionally and personally. I have learned a lot of new things by using VR technology, but it came at the cost of experiencing high levels of stress due to the combination of studies, internships, my final year project, and other personal aspects.

VR is a technology that, seeing how it is currently gaining more and more traction in the market, can provide the audience with an unprecedented experience, further blurring the line between reality and fiction. Its potential for application beyond entertainment was always one of the characteristics that sparked my desire to work on projects using this technology. Undoubtedly, undertaking this project was one of the best decisions I have ever made. Thanks to this project, I learned how to delve into the documentation of a technology that appears to be more than what it actually is. I believe that the addition of a VR course by the university will surely be an enriching experience for everyone.

On the other hand, I believe that this project is an original proposal, as it combines the entertainment and medicine fields through a very organic medium like VR. I wanted to show my parents that video games can go beyond mere entertainment and can serve a greater purpose. And so, as mentioned at the beginning of the article, after discussing with my cousin what application a project made with VR could have, I decided to focus this idea on therapy.

In regards to the objectives I set at the beginning of the project, I would say that I am satisfied with what I have achieved. The first objective was to create a VR game with the purpose of aiding patients in their rehabilitation, and based on both their feedback and the feedback from doctors, my project is heading in the right direction. The second objective was to create simple mechanics that also presented a balanced challenge, and I believe I have been able to accomplish that. Perhaps the objective of creating a robust progression system is the one I haven't fully achieved as I would have liked, due to time constraints towards the end of the project. However, I think I have compensated for it professionally by self-learning and effectively utilizing a new technology, which was the fourth objective. With all this considered, I can say that I have successfully developed a satisfactory proof of concept, which was the last objective of the project.

I believe that the work I have done has been sufficient, considering the context that I was working with a technology that had not been taught before in my degree. Therefore, I had to learn it from scratch. However, the artistic or visual aspect is something that can still be highlighted as needing improvement. Unfortunately, I am not skilled in that area, and I even had to inform both in this article and to my professor from the beginning that my project would focus more on functionality rather than making the game visually appealing. But it is also true that if I had dedicated more time to the visual aspect, I would not have been able to present anything, even in the second delivery.

With all that being said, the last thing left to add is that if someone were to ask me if doing this project was worth it, I would say that every second of this project has been worth it.

5.2 Future work

The final result I have achieved after the presented outcome still aligns with what I had in mind from the beginning. While it may have been mentioned that some ideas were discarded, it is not entirely accurate. These ideas are documented and saved, but due to time constraints or technical aspects, they were not implemented. However, if the project continues, sooner or later they will take shape in the game.

One thing I want to do when I have more free time, is to implement the corrections I have gathered from the testing's feedback. Furthermore, I would like to add other features, such as a companion pet that helps the player get accustomed to the game controls and VR, serving as an introductory tutorial and accompanying the player throughout the game. I also have plans to include a "free play" mode, creating a recreational area with various interactable elements of different shapes, so that players can engage with the game beyond their therapy sessions.

My intention with this project is to keep working on it, the doctors and I are keeping in touch to schedule another testing session in the future. I also want to explore opportunities on social media or conferences to seek funding or find collaborators. While I don't have plans to sell the game, it is true that you cannot survive on thin air. If I could get the attention of a public institution and secure resources for the project, I believe that would be satisfying enough for me to keep working on this game.

Bibliography

[1] INE. El daño cerebral adquirido en cifras. https://fedace.org/cifras_dano_cerebral. Accessed: 2023-02-22.

[2] S. M. Hatem, G. Saussez, M. Della Faille, V. Prist, X. Zhang, D. Dispa, and Bleyenheuft. Rehabilitation of motor function after stroke: A multiple systematic review focused on techniques to stimulate upper extremity recovery. 10, 2016. Accessed: 2023-02-22.

[3] Nieves Salinas. Rehabilitation of motor function after stroke: A multiple systematic review focused on techniques to stimulate upper extremity recovery. Available at https://www.epe.es/es/sanidad/20220905/ictus-sanidad-publica-sanidad-privada-14396785 (2022/09/05). Accessed: 2023-02-23.

[4] Francesca Marchionne. Virtual reality rehabilitation – the future of physical therapy. Available at https://imotions.com/blog/insights/research-insights/virtual-reality-rehabilitation/ (2019/12/17). Accessed: 2023-04-15.

[5] Unity technologies. Available at https://unity.com/es. Accessed: 2023-02-26.

[6] Openxr. Available at https://www.khronos.org/openxr/. Accessed: 2023-02-26.

[7] Htc vive. Available at https://www.htc.com/es/virtual-reality/. Accessed: 2023-02-26.

[8] What are the system requirements? Available at https://www.vive.com/eu/support/cosmos/category_howto/what-are-the-system-requirements.html. Accessed: 2023-06-02.

[9] Microsoft visual studio. Available at https://visualstudio.microsoft.com/es/. Accessed: 2023-02-26.

[10] Blender. Available at https://www.blender.org/. Accessed: 2023-03-23.

[11] Unity asset store. Available at https://assetstore.unity.com/. Accessed: 2023-03-05.

[12] Overleaf, latex online editor. Available at https://es.overleaf.com/. Accessed: 2023-04-12.

[13] Steam vr. Available at https://www.steamvr.com/. Accessed: 2023-02-26.

[14] Unity Technologies. Unity shader graph. Available at https://unity.com/features/shader-graph. Accessed: 2023-03-05.

[15] Indian Ocean Assets. Low poly fire. Asset download available at https://assetstore.unity.com/packages/vfx/particles/fire-explosions/low-poly-fire-244190. Accessed: 2023-03-17.

[16] Pixim. Car'toon : The sport car with interior. Asset download available at `https://assetstore.unity.com/packages/3d/vehicles/land/car-toon-the-sport-car-with-interior-62697`. Accessed: 2023-04-14.

[17] Tridentcorp. Low poly gems. Asset download available at `https://assetstore.unity.com/packages/3d/props/low-poly-gems-245515`. Accessed: 2023-05-24.