Virtual Reality – Project Report

Loïc Karl Droz – Rodrigo Soares Granja

Spring 2019

Abstract

The game we implemented for our project takes the form of a linear progression in which the player can use magic spells to interact with their environment, and also defeat some skeleton enemies.

We use the Oculus Rift VR headset for aiming, and the Leap Motion device for moving, and casting spells.

Gameplay

This section describes how the player uses the controllers to perform actions in the game, which actions are possible, and how they interact with the environment.

The player can perform the following actions :

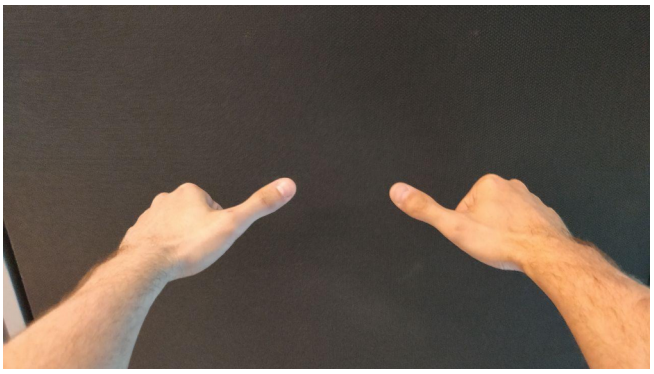


Fig. 1 : move forward

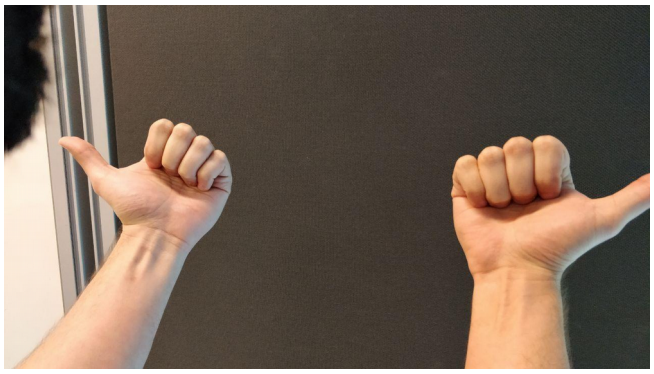


Fig. 2 : move backward

Walk forward, backward, sideways : relative to the direction the player is looking at. Using the Leap Motion (Fig. 1 - 4).

Fig. 3 : strafe left

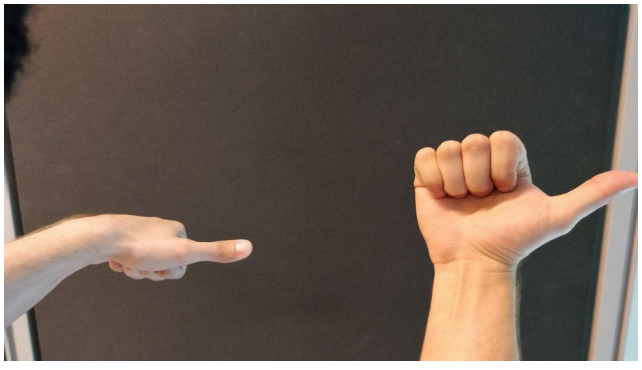


Fig. 4 : strafe right

Interactions : the player will bump into walls, doors and cranes.

Look around : changes the orientation of the player's look. Using the VR headset.

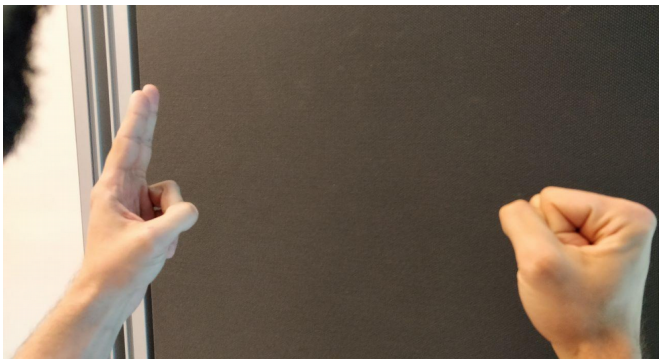


Fig. 5 : activate fire

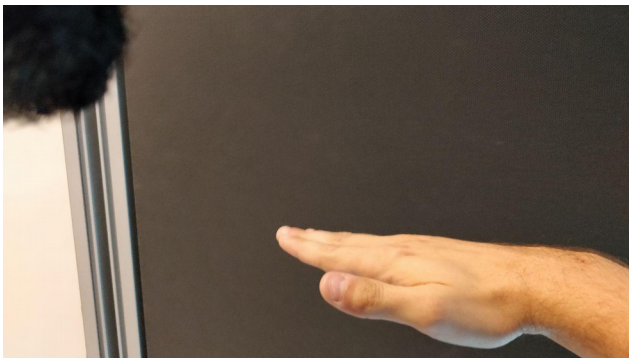


Fig. 6 : deactivate fireball

Interactions : dictates the direction of the player's forward and backward movements, as well as the direction of the fireballs (see below).

Use the right hand as a torch : illuminates the environment, and is the first step to casting a fireball (see below). Uses the Leap Motion (Fig. 5 - 6).

Interactions : illuminates the environment.

Cast a fireball : using the Leap Motion (Fig. 7 - 8).

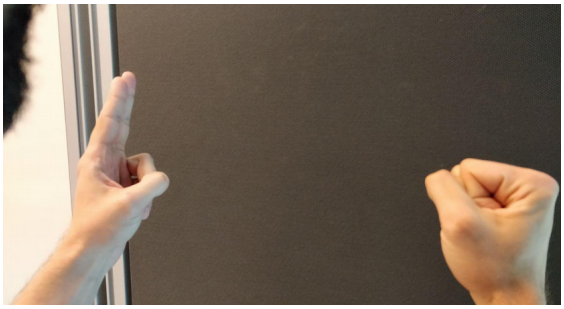


Fig. 7 : activate fire

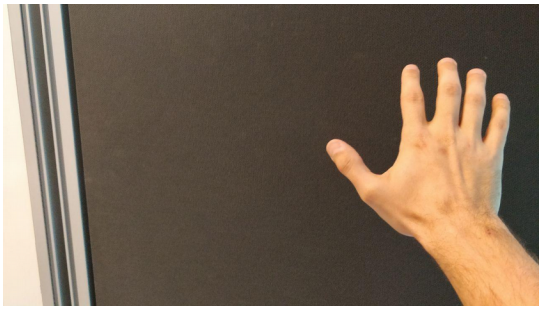


Fig. 8 : cast fireball

Interactions : triggers the fire ball-activator, destroys cranes, pops ice-skeletons out of existence, bumps with their spells and can therefore be used to deflect them.

Cast an iceball : using the Leap Motion (Fig. 9).

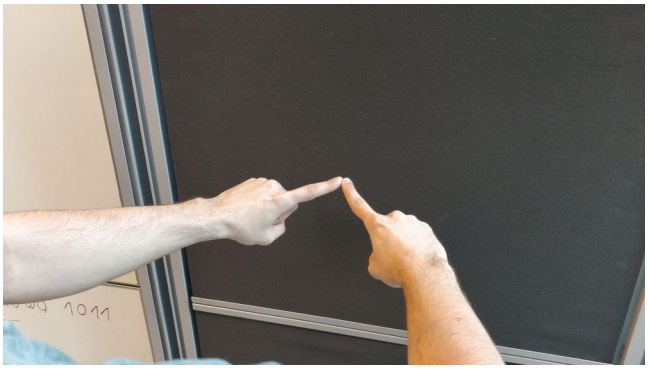


Fig. 9 : cast iceball

Interactions : triggers the ice ball-activator, extinguishes fires, pops fire-skeletons out of existence, bumps with their spells and can therefore be used to deflect them.

Deploy a shield : protects the player from enemies' attacks, using the Leap Motion (Fig. 10). Active as long as the hand motion below is detected.

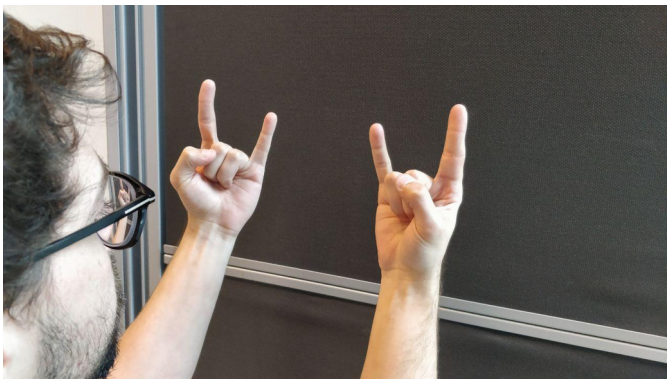
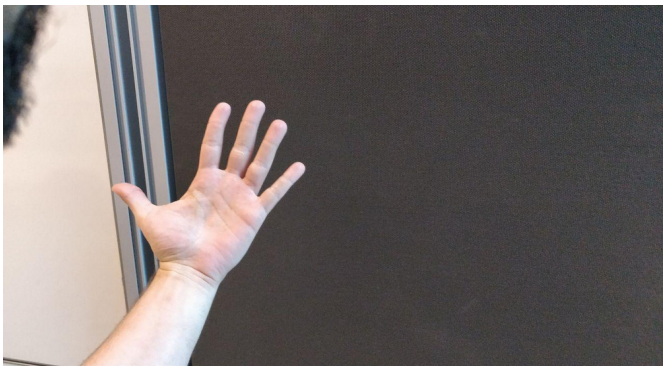


Fig. 10 : deploy shield

Interactions : prevents enemies' ice and fireball from reaching the player.



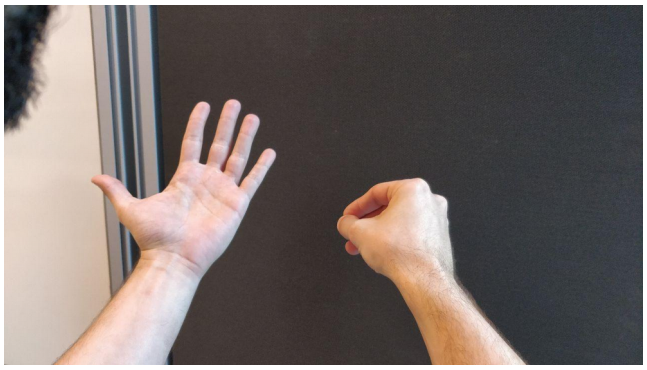


Fig. 11 : open inventoryFig. 12 : grab and place cube cube in the inventory

Open/close/interact with inventory : using the Leap Motion (Fig. 11 – 12).

Interactions : allows gathering the three labyrinth cubes (see below).

Game environment

The game is divided in three main areas. The first area, which is also the game's starting point, is a hall in which the player is faced with two balls, one made of fire, one made of ice, and a closed door (Fig. 13). The player has to shoot the ice-ball with ice, and the fire-ball with fire, which will then open the door and allow the player to proceed through the second area.

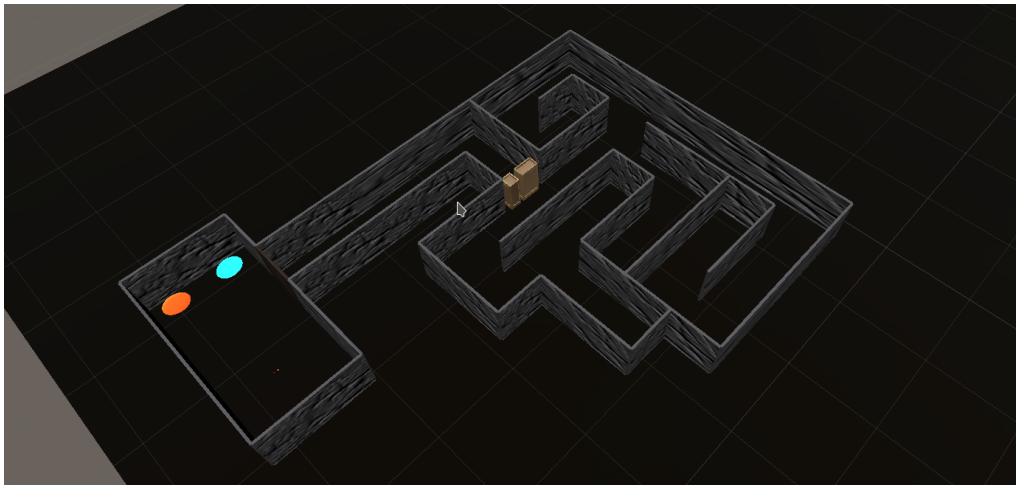


Fig. 13 : first and second areas of the game, also called the labyrinth

The second area is a labyrinth. The player has access to an inventory hidden in their hand accessible using a simple motion (see above). The player must gather three small floating cubes hidden in the labyrinth to be teleported to the third area. Once the player is close enough to any of them, they can grab them with their right hand, and drop them in the inventory. Progression is hindered by two obstacles : wooden crates which can be destroyed by the player's fireballs, and fire which can be put out by the player's iceballs and will remove health points if the player gets too close. The player starts with 10 health points. If the player's health point count drops to 0, they will respawn in the first area.

Fig. 14 : the third area, also called the arena

The third area (Fig. 14) is an arena in which the player has to knock out skeletons casting spells at them. Skeletons can be of two types : fire and ice, and can be knocked out respectively by ice and fire. Each hit by a skeleton will deplete 1 health point. If the player's health point count drops to 0, they will respawn in the arena. If the player manages to knock out all skeletons in the arena, a portal in the arena will open, and allow the player to be teleported to the end (Fig. 15).

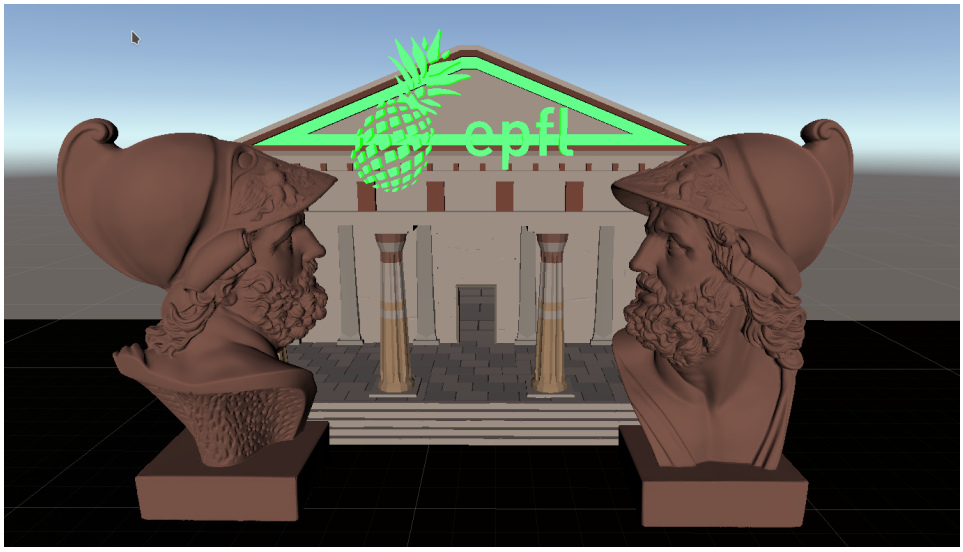


Fig. 15 : the end of the game

Implementation details

We used Prefabs found on the asset store for the major assets of the game, such as the Arena, the Skeletons, and the textures.

We have used LeapMotion Orion software and development kit.

Gestures implementation. We use a combination of leap motion detectors such as : extended fingers detector, palm direction detector, finger direction detector and proximity detector. These are independant detectors that allow us to check small gestures such as a closed fist, a palm facing the user, etc. For some of our gestures we needed to combine these detectors and for this purpose we use Detector Logic Gates. These use boolean logic (AND, OR, etc).

In the detectors that check a direction there are 2 important angles, On and OFF angles. These angles determine if a user gesture has an angle that is close enough to the verified direction. We use this to calibrate the gestures. We have calibrated the game to work with our gestures and our motions but quickly realised that the game required more calibration so it identifies more broad or innacurate gestures. While we could easily do the gestures for moving forward/backward during the demo, the assistants were struggling to do so. Because of this, we have now augmented the ON angles for the gestures involved in user motion. We have also added physical keyboard input for moving forward and backwards in the case where these gestures fail to be identified. To move forward just press up key and down key for backwards.

Leap Motion Orion provides modules such as Interaction Module and Core Module. Interaction module allows us to more easily implement grasp interactions. We just need to add an Interaction Manager as a child in the LeapRig (myRig in our case) and add an Interaction Behaviour script to the object we want to grasp.

Anchors were harder to implement and finally we based our implementation on one of the examples given in the module. There are 3 collectables that have Interaction Behaviour and Anchorable Behaviour scripts. These allow for the objects to be grasped and to be anchored. Then we added Attachment Hands to our Rig. This allows for objects to be attached to some joint in the hand. We added anchors to the object palm in the Attachment Hands of the left hand and set them in the Anchorable Behaviour of the corresponding anchorable object. We used the same tweens (scripts that move the objects according to the anchors) as in the example.

The anchors will only appear if the palm if facing the user and this is done by the Palm Facing Script also provided by the example.

Finally when all objects are in their corresponding anchor (attached) the player is teleported to the arena. Since he no longer uses the anchors and anchorable objects we just deactivate them. We wrote a script that is given all collectibles and uses their AnchorableBehaviour component to check if they are anchored or not.

One of our spells is to spawn fire in a hand. This fire is attached to the palm of the corresponding hand using the Attachment Hands, the spell only activates or deactivates it.

We wrote scripts for every shoot event. We have a script for shooting fireballs and one for shooting ice projectiles. In the case of the fireball, it will spawn at the coresponding palm position in the direction the user is facing (it uses Camera.main.forward). The ice case is a bit more complicated, we spawn an ice capsule at the camera position, create a vector direction from this point to the index finger tip and set it as the direction velocity for the rigidbody. This allows the user to direct the projectile. We added a round crosshair so the user knows exactly where the fireball will go.

We also have a script for opening the door (it only opens) which uses Vector3.Lerp to smoothly change the position from OpenPosition to ClosedPosition when the fireball hits the orange ball and the ice hits the blue ball.

We have destructible objects such as crates, fire and the skeletons. We implemented them in such a way that only certain spells may destroy them : crates are destructible by fire, fire by ice and skeleton by corresponding spell (ice shooting skeleton disappears by fire, fire shooting skeleton disappears by ice). For this purpose we use tags and check them in script. Fireballs have tag « Fire » and ice has tag « Ice ».

The rig has a capsule collider and rigidbody and this prevents it from going through walls and obstacles which also have rigidbodies and capsule colliders. Our rig is not influenced by gravity.

The interactable objects are light emitters and have point lights with respective color so the user knows where they are. We also implemented an attraction between right index finger and the interactable object when these are close enough. We used leap motion's Proximity Detector to do this. Proximity will activate a script that uses Vector3.Lerp to move the object towards the fingerTip. The only way for the script to know these positions is for us to pass them to the script. The user can play with it, pinch it, tap it, etc, and it will always come back to the finger if it is close enough.

We tried to lower the ambient light to the minimum so the labyrinth is dark and the user is required to spawn fire, which also emits light. When the user ends the game we activate a directional light so he can see the end game temple.

Conclusion and afterthoughts

We aimed for a certain experience where the user has complete freedom of hands and is not constrained by the Oculus' controllers. This means that some features such as teleportation or movement are completely dependant on gestures. It would have been much easier to add controllers and use these for teleportation (using VRTK and SteamVR packages) but we welcomed the challenge. Finally, we have movement that works quite well for us and we have a satisfying experience. We have recalibrated the movement gestures to be more flexible and permissive and we hope other users have the same satisfying experience as we do. For future work we would implement a small tutorial with clear instructions for each gesture.

On a more personal note, we found this project enriching because it was an opportunity for us to learn the basics of VR games with the Oculus Rift, as well as the Leap Motion which is arguably a more unusual VR device, at the time of writing. It was also a good opportunity for us to get started with Unity3D, which goes beyond the scope of VR but rather video game design and development in general.