The Euclidian Escape

VR Project

Alexandre CHAU & Clément BURGELIN

May 19, 2019

Introduction

The Euclidian Escape is a VR escape room game built with Unity for the HTC Vive headset. The player is trapped in a broken spaceship and must escape it. To achieve this goal, he must solve some riddles using in-game objects and the environment. The riddles are chained such that solving one unlocks another until the final exit.

Implementation

The games uses the VRTK framework and the Steam VR SDK to integrate the Vive headset and its controllers. To help us debug and run the game without any VR hardware, we also use the VRTK simulator which provides a mouse and keyboard interface to emulate and perform VR interactions.

The room

The room is a 2.6 x 2.35m space which roughly matches the calibrated play area of the HTC Vive in the INJ218 lab room. We wanted the player to be constrained in this space so that he/she feels locked inside. This however raised the following problem : as the player moves his headset around, he may be able to traverse walls if the physical space allows it. Usually, in a non-VR game, this behaviour would be fixed by teleporting the player back in front of the wall when a collision is detected. However this approach cannot be applied to this VR game, as each teleportation would "shift" the whole world coordinates relative to the physical position of the player, and the locomotion mechanics do not provide a way to teleport back. Hence, we followed the approch chosen by various VR game developers and fade the headset to complete black when a wall collision is detected, which effectively prevents the player to see anything if he isn't in the room, while staying compatible with the real-world space. The room contains a central console, which gives textual hints to the player.

Enigmas

All VR interactions in the game are based on the enigmas, and solving them requires interacting with the objects and the environment in the scene. The following descriptions detail their implementations, which contain spoilers of the game.

The floppy disk

The first enigma is very simple : the player spawns in the room and among the various objects, there is a floppy disk. The player is able to grab the floppy disk using the default grab key of the controller (a visual feedback is given to the player to indicate that the action is possible, by highlighting the object). When he inserts the object inside the floppy disk input of the console, the console boots up. This is implemented with a simple collision detection.

The maintenance password

When the previous enigma is solved, a blinking light is also activated. The console prompts the player to input the password to the maintenance room. The hint is given through the pattern of the blinking light : it lights up 4, 2, 1 and 5 times repeatedly. This is implemented in a custom script `Assets/Blink.cs` using the asynchronous primitive Invoke recursively with different timings. The player must touch the corresponding keys consecutively on the numpad using the controller, and the logic to subscribe to the touch events and check the password is implemented in `Assets/NumpadState.cs`. To ensure that the controller does not hit multiple keys at the same time (since its collider box is quite big), the hitbox of the keys is reduced to their centers.

The engine room

The previous password unlocks the maintenance / engine room. It is a non-euclidian change of the virtual space when the player rotates 360° in either direction. The hint is given on the back wall : the newly displayed sign will always point to the engine room in the direction of the current rotation, and the cockpit for the opposite. We implement the angular changes detection in `Assets/NonEuclidianGeometryChanges.cs`. This script polls the headset at each frame update and watches for Y-axis rotation changes by detecting changes between 4 quadrants (NE/NW/SE/SW). These triggers change the sign on the wall, and also (de-)activate the whole engine room, located in front of the cockpit room.

Unblocking the gears

In the engine room, the player will see 2 giant gears that are blocked by a green cube. He must destroy this cube by using the laser bit-gun that lies in the room. The gun can be grabbed and aimed using the VR controller. The intersection logic is implemented in `Assets/GunLazer.cs` : it uses ray-casting to determine the collision and trace the laser ray.

Escaping the spaceship

Unblocking the gears enables the final escape. The player must first return to the cockpit (as indicated by the console). When he looks back at the wall, he will see “Backward Exit” written on it. The escape also uses non-euclidian space properties : the player cannot escape by going forward through the door, he must go through it backwards. This is also implemented in the non-euclidian geometry changes script, that removes the back wall when the player faces forward. The player is then able to escape into space and float there indefinitely.

Extras

These objects are not necessary to solve the enigmas, but they add some VR interactions and are used to confuse the player with unuseful hints.

Portal radio

This radio is toggled when the player touches it and it plays the Portal radio tune.

Flashlight

This item can be grabbed and the light can be toggled with the “use” button of the controller.