**Quidditch VR Project: Game Description**

**Modelling:**

The only asset package used was the “Standard Assets” package from the unity Asset Store, which was used to create the mountains, grass and trees. The glove that shows the right hand of the player is also from the “Standard Assets” package. The enemy was modelled with the default shapes that are available in Unity (a *Sphere* and a *Particle System* superimposed).

All other assets (Quidditch Pitch, Broom, Quaffle, Hoops), were fully modelled with Blender, only getting inspiration of different images from Google. UV Mapping was used to map textures onto the modelled objects in Blender. Blender models are provided in a separate folder.

**Interaction:**

The interaction with the environment is designed to be as realistic as possible, with the left hand simulating the broom and the right hand in charge of holding and throwing the quaffles. The controls of movement were implemented with a vector that goes from the belly to the left controller, since like that the movement of the user has to resemble the movement of somebody on a flying broom. The physics of the quaffles (impact, gravity, interaction with the players velocity) were also implemented.

The reaction from the Virtual Environment is displayed via haptic feedbacks to the player.

**Game Design:**

The player’s task is to score 30 points. To do so the player has to fly around the pitch with his broom, grabbing a quaffles from the ground and throwing them into the hoops, getting 10 points each time he scores. But victory won’t be an easy job, since the player will be followed by an enemy (fire ghost), who will make him loose a life each time he gets caught.

**C# Scripts:**

The game uses 6 scripts that we implemented with what we learned from doing several different tutorials on how to move in 3D space, how to interact with objects, etc.

The names of the scripts with a brief description of them are provided below:

* **Flying\_new2**:

1.) Each time the pad of the left controller is clicked, switch state of player from moving to not moving or vice versa.

2.) If the player is moving, the direction is defined by the vector that goes from the belly (head - bellyToHead) to the left controller. The velocity is proportional to the magnitude of that vector.

3.) If the left trigger is pressed and held, mySpeed becomes SuperSpeed.

* **Hand**:

1.) The script Hand is attached to the right controller

2.) The player can only interact (hold) objects that have the Tag "Interactable" and when it is "colliding" with the object collider

3.) When holding the trigger, the player catches the object, when not holding, the object falls.

4.) If more than one object is close to the player, only the nearest object will be grabbed.

* **Haptics**: If left trigger pushed (SuperSpeed activated), haptic feedback to player, to simulate high speed
* **Interactable**: This script is used so that objects with the Tag "Interactable" (see quaffle object) can interact with the "Hand" (left controller)
* **Scoring**: This script is used to account the points when a ball collides with the collider inside a hoop (+10 points) and the ball reappears somewhere random in the pitch. When the player gets to the winCount (by default =30), the text "You Won!" is displayed in the sky.
* **ThrowBall**: This script is attached to the enemy. The name is “ThrowBall” since at first we wanted to use it to make the quaffles follow the player but in the end used it for the enemy.

1.) The enemy has a lifetime of 10 seconds. During its lifetime it will always move towards the player's position with "dir". When its lifetime equals zero it will not look for the player anymore and will keep going with the last saved direction towards the player "lastDir".

2.) If the enemy collides with the player, a haptic feedback in both controllers will be displayed and the players will lose a life (the number of lives the player has are displayed in the sky over the hoops). If the player loses all lives, the text "Game Over" will be displayed. And the enemy will reappear in a random position (see "SetEnemyBack()")