CSE 566 – Virtual Reality Spring 2019
Assignment 1: Basic VR - Report

Jinghuan Shang – ID# 112076155

March 12, 2019

1 Important Links

Unity Project File: https://drive.google.com/open?id=1gloup2dACtpGdaUvJmRWZ2B5B2CzyjKQ
Video: https://drive.google.com/open?id=1jxV9XgUSXkxueC8cOdws_wP6-BcgU2Z6

2 Environment Set Up

Software used:

- Unity: 2017.4.21f1
- Google VR SDK for Unity v1.190.1[1]

Hardware used:

- VR Headset: VR SHINECON 3D VR Glasses
- Mobile Phone: HUAWEI P10 Plus with Android 8.0
- Controller: VR SHINECON Glass Remote Control

3 Directory Hierarchy

Only important files are mentioned.

```
VRLab1\—————————————————————————————Unity project directory
  Assets\———————————————————————————Assets used in the project
    lab1_first_trail.unity——————————Main scene file
    *.cs————————————————————————————C# script files
    [other directories]\————————————Imported assets files
```

4 Extra Features

4.1 Car Motion

The car in the scene will slowly move along the road around the building when the character(camera) teleports into the car. It can correctly turn directions if the current road ends.

4.2 Extra View in the Building

My building is designed and implement by myself with several 3d objects. Besides the basic view on the first floor, I set another place that the camera can teleport to on the second floor. Just press the button of teleporting into the building again if the current location is on the first floor. The scene on the second floor is an imitation from one of my favorite animation “Neon Genesis Evangelion”.

5 Implementation

5.1 House

The house, or say the building, is implemented by myself py several 3D objects. Due to not able to create complex 3D polygons for now, I just use many 3D cuboids. The material of the walls on the first floor is glass, which is the same as the car[2].

As for the walls on the second floor, I place my customize texts on it. The texts are actually picture resources serve as textures. The picture is designed manually by myself using a picture editor.

The lamp is from the asset[3]. I add an extra cylinder rod beneath it to raise it up. The lights in the house and the lamp are controlled by the sunlight. If the intensity of the sunlight is less than 0.5, lights will be turned on.The road is from the Kajaman’s Road - Free[4].

5.2 Airplane

The airplane (including model, texture and material) is from the asset store[5]. The plane can do all the Quaternion rotations. It will go back to its original place when the journey is finished. I control this by recording the original coordinates and remaining time in this round of motion. When time runs out, the plane will be moved to its original place and reset the timer(a float type variable).

The following are Quaternion rotations around X, Y, and Z axis respectively.

```
transform.Rotate(Vector3.right, turnSpeed * Time.deltaTime);
transform.Rotate(Vector3.up, turnSpeed * Time.deltaTime);
transform.Rotate(Vector3.forward, turnSpeed * Time.deltaTime);
```

5.3 Car

The car (including model, texture and material) is from the asset store[2].

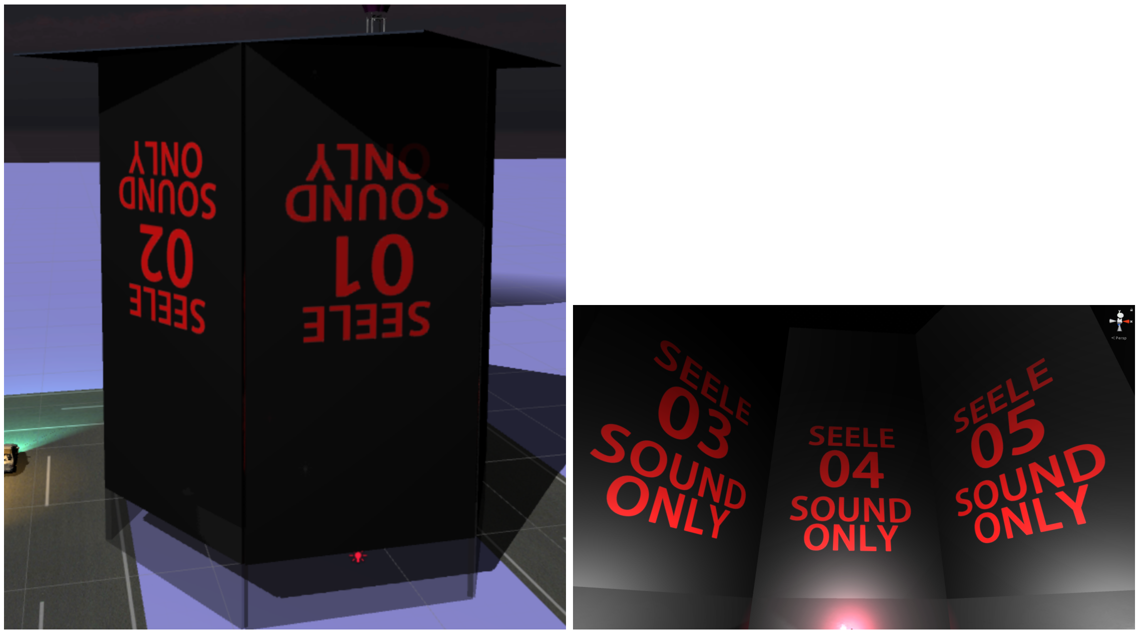

Figure 1: House and its second floor

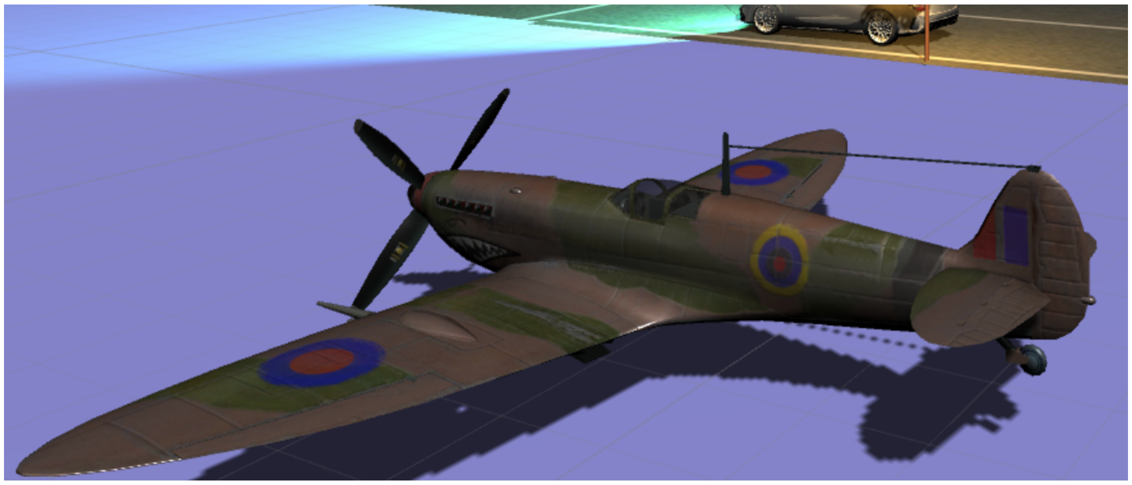

Figure 2: Airplane

The camera position in the car is set by applying a small offset based on the coordinate of the car. When car is moving, Player component gets the position of the car and makes a synced motion by applying same translation vectors to the Player and the car.

The four segments of car motion is maintained by a variable tracking its moving direction and a switch statement. In the switch statement, car is applied rotation translation properly when it will exceeds the current segment of road. The four coordinates indicating the boundaries in the direction of car motion is used to check these switch cases.

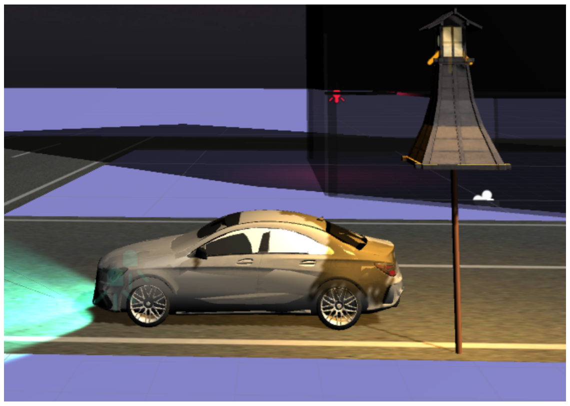

Figure 3: Car and Lamp

5.4 Hot Air Balloon

The hot air balloon (model and color design) is from the asset store[6].

The hot air balloon's trajectory is similar to the car, where this only contains 3 segments of motion. The middle part, where the balloon is simultaneously moving forward and upward, is implemented by apply a translation vector with positive values on both X and Y axes.

5.5 Day and Night

Day and night changing is implemented by modifying the intensity and the rotation angle of the directional light. The intensity of the 'Sun' is designed by equation

$$L(t) = \begin{cases} \frac{1-0.005}{150}t + 0.005, & t \in [0,150) \\ 1, & t \in [150,300) \\ 1 - \frac{1-0.005}{150}(t-300), & t \in [300,450) \\ 0.005, & t \in [450,600) \end{cases}$$

where t is cycling between $[0,600]$, corresponding to 10 minutes in real world. $t \in [150,450)$ is day and $t \in [0,150) \cup [450,600)$ is night. The rotation of the "Sun" $R_{Sun}(t)$ is in uniform speed. If represented in Euler angle, it starts from $(0,0,\gamma)$ to $(180,180,\gamma)$, where γ is an arbitrary angle, implemented by

```
transform.Rotate(Vector3.up * angularSpeed * Time.deltaTime * 10 + Vector3.right *
  angularSpeed * Time.deltaTime * 10);
```

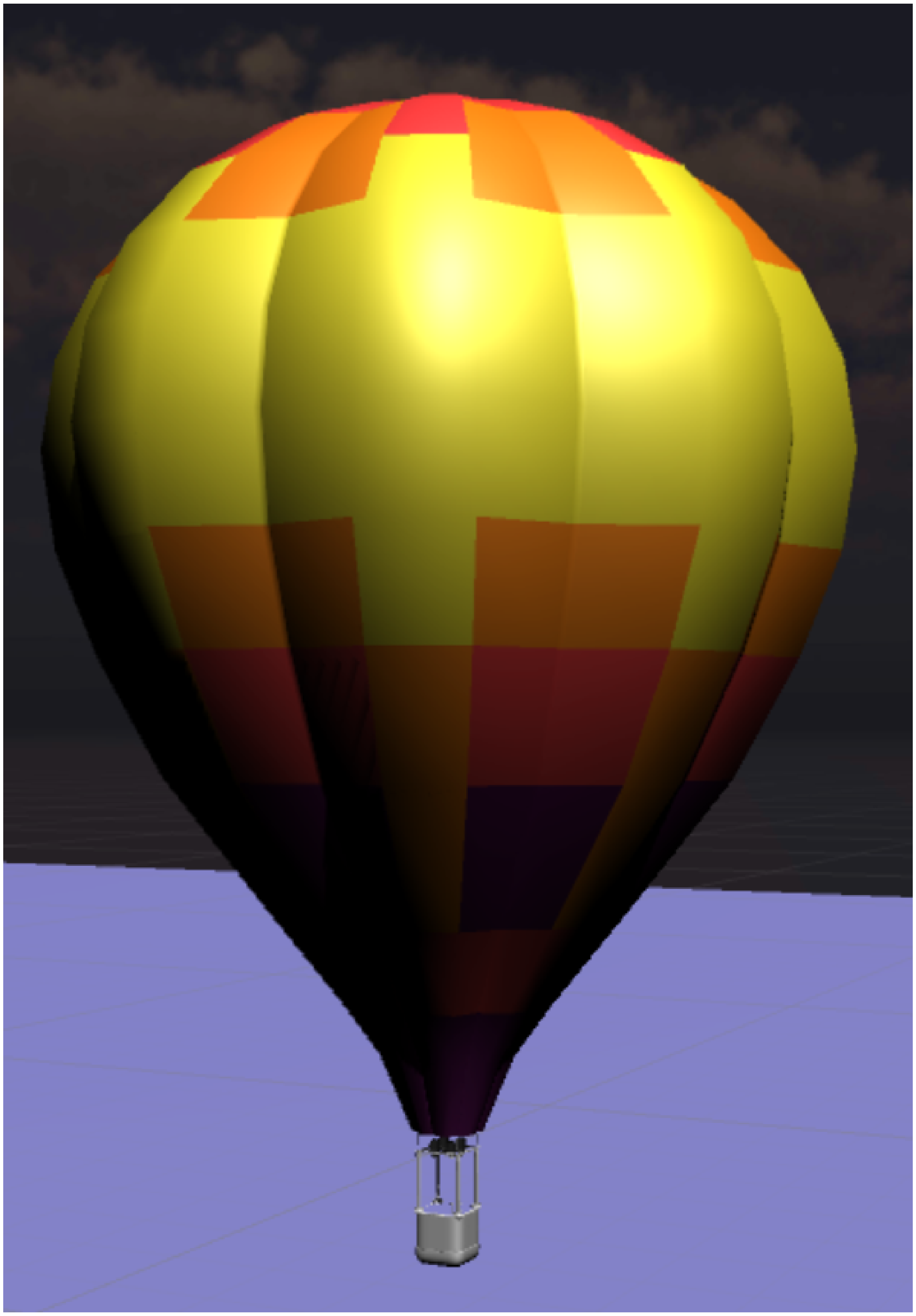

Figure 4: Hot Air Balloon

This rotation simulates the sunlight changing during the day. When $t > 600$, $R_{Sun}(t)$ is reset to $(0, 0, \gamma)$

The "Moon" is stationary in the sky with no rotations. It only has $\frac{1}{2}$ brightness of the Sun. When the Sun's intensity is less than a threshold (0.3 in my project), the moon's intensity will gradually go up. Their sum will be exactly 0.3 in case of sudden flash when the moon is "lighted". During the day, the Moon's intensity is 0.

There is another important component called "Skybox" in the scene. It simulates the sky above our heads. The Skybox has a parameter "Exposure" can be explained as how bright it be in our camera. So when turns to the night, it will also be darker. The value of Exposure is always equals to the sum of the intensities of the Moon and the Sun.

5.6 Interaction and Camera

In my project, I use the Player component in Google VR SDK for Unity[1] to develop my camera and changing of views driven by gyroscope inside the phone.

As for interaction, I utilize 4 keys on the joystick, **A**, **B**, **Y**, and **X**. The receive is implemented by codes like

```
Input.GetKeyDown(KeyCode.Joystick1Button0)
```

Interactions are described in the following list.

A Teleport into the building. Press **A** when in the building again to teleport to the second floor.

B Teleport into the car. The car will start its motion along the road, carrying the camera.

Y Teleport into the bracket of the hot air balloon. The camera will also follow the motion of hot air balloon.

X Teleport back to the initial point where the game starts.

In order to easy testing, I also bind them with corresponding characters on the keyboard. I set a variable to track the current mode. For example, 1 and 5 are for in the building, and 2 for in the car. The variable will help me decide whether and how to update the position of the camera under different modes.

6 Acknowledgment

I would like to thank Xueying Bai and Ying Lu for discussions during this assignment. I also appreciate all the authors of free assets available in the Unity Asset Store.

References

[1] Google vr sdk for unity v1.190.1. [Online]. Available: https://github.com/googlevr/gvr-unity-sdk/releases

[2] Low-poly civilian vehicle 5. [Online]. Available: https://assetstore.unity.com/packages/3d/vehicles/land/low-poly-civilian-vehicle-5-124987

[3] Japanese street lamp. [Online]. Available: https://assetstore.unity.com/packages/3d/props/exterior/japanese-street-lamp-68184

[4] Kajaman’s road - free. [Online]. Available: https://assetstore.unity.com/packages/3d/environments/roadways/kajaman-s-roads-free-52628

[5] Super spitfire. [Online]. Available: https://assetstore.unity.com/packages/3d/vehicles/air/super-spitfire-53217

[6] Hot air balloon 3d model. [Online]. Available: https://free3d.com/zh/3d-model/hot-air-balloon-54348.html