Lab5

Nidhi Kumari
M22AI822

Lab-5: VR App Development inducing motion sickness in Unity

Develop a VR app simulating motion sickness while measuring the threshold speed for inducing discomfort, dizziness, and nausea.
(i.e., speed-based object detection, simulated downward movement from a hill, etc.). Create some random objects to collide/interact with player movement to induce motion sickness.

STEPS:

Create a new Unity project for 360◦ VR video experience with Unity3D

Create a new project.
Select the appropriate Unity Editor version (2022.3 LTS).

- Choose the 3D Core Template.

Importing Required Packages and Assets for 360◦ VR video development in Unity

Project Setup:
To Develop a 360◦ VR Game in Unity.
Setting Up Google Cardboard SDK in Unity
• Go to Google Cardboard website > Developers > Unity - Android/iOS Documentation (Quickstart for
Google Cardboard for Unity) > Follow the steps given in document for how to use the Google Cardboard
XR Plugin for Unity to create your own Virtual Reality (VR) experiences.
• Go to Unity > create a normal 3D project (3D core) > Go to Window > Package manager > click +
sign in package manager > click - Add package from git url > copy link from google cardboard website
document: (https://github.com/googlevr/cardboard-xr-plugin.git) to the Package manager space and click on Add > wait for GIT package installation process > Google Cardboard XR plugin for Unity
is now installed in the packages.
• Google Cardboard XR Plugin for Unity: > Go to Samples > Import (wait for import process here) >
close the package manager window.
• Go to the Hello Cardboard Folder Under Assets > Sample > Google Cardboard XR Plugin for Unity
> Hello Cardboard > Go to Scenes folder > double click on Hello Cardboard demo to open it in Unity
project > move with cursor or keyboard shortcut to see the scene view of it.

Quickstart for Google Cardboard for Unity
Import the SDK and create a new project
Follow these steps to import the Unity SDK and create a new project.

1. Open Unity and create a new **3D** project.

2. In Unity, go to **Window** > **Package Manager**.

3. Click **+** and select **Add package from git URL**.

4. Paste https://github.com/googlevr/cardboard-xr-plugin.git into the text entry field.
   The package should be added to the installed packages.

5. Navigate to the **Google Cardboard XR Plugin for Unity** package. In the **Samples** section, choose **Import into Project**.
   The sample assets should be loaded into Assets/Samples/Google Cardboard/<version>/Hello Cardboard.

Configuring HelloCardboard scene
1. Navigate to Assets/Samples/Google Cardboard/<version>/Hello Cardboard/Scenes, select **Add Open Scenes**, and choose **HelloCardboard** to open the sample scene.

2. Open the **Layers** menu and select **Edit Layers...**.

3. Define a new layer called "Interactive".

4. Click on the **Treasure** GameObject to open the Inspector window. Set its layer to be "Interactive". If a pop up window appears asking if you want to set layer to Interactive for all child objects as well, click on "Yes, change children".

5. Click on the **Player > Camera > CardboardReticlePointer** GameObject to open the Inspector window. In the "Carboard reticle pointer" script, select "Interactive" as the **Reticle Interaction Layer Mask**.

Configuring Android project settings
Navigate to **File** > **Build Settings**.

1. Select **Android** and choose **Switch Platform**.

2. Select **Add Open Scenes** and choose **HelloCardboard**.

Player Settings
Resolution and Presentation

Navigate to **Project Settings** > **Player** > **Resolution and Presentation**.

1. Set the **Default Orientation** to **Landscape Left** or **Landscape Right**.

2. Disable **Optimized Frame Pacing**.

Other Settings

Navigate to **Project Settings** > **Player** > **Other Settings**.

1. Choose OpenGLES2, OpenGLES3, or Vulkan, or any combination of them in **Graphics APIs**.
2. Select Android 8.0 'Oreo' (API level 26) or higher in **Minimum API Level**.
3. Select API level 33 or higher in **Target API Level**.
4. Select IL2CPP in **Scripting Backend**.
5. Select desired architectures by choosing ARMv7, ARM64, or both in **Target Architectures**.
6. Select Require in **Internet Access**.
7. Specify your company domain under **Package Name**.
8. If Vulkan was selected as **Graphics API**:
    - Uncheck **Apply display rotation during rendering** checkbox in **Vulkan Settings**.
    - If the Unity version is 2021.2 or above, Select ETC2 in **Texture compression format**.

Publishing Settings

Navigate to **Project Settings** > **Player** > **Publishing Settings**.

1. In the **Build** section, select Custom Main Gradle Template and Custom Gradle Properties Template.
2. Add the following lines to the dependencies section of Assets/Plugins/Android/mainTemplate.gradle:

```
implementation 'androidx.appcompat:appcompat:1.6.1'
implementation 'com.google.android.gms:play-services-vision:20.1.3'
implementation 'com.google.android.material:material:1.6.1'
implementation 'com.google.protobuf:protobuf-javalite:3.19.4'
```

7. Add the following lines to Assets/Plugins/Android/gradleTemplate.properties:

```
android.enableJetifier=true
```

```
8. android.useAndroidX=true
```

XR Plug-in Management Settings

Navigate to **Project Settings** > **XR Plug-in Management**.

1. Select Cardboard XR Plugin under **Plug-in Providers**.

Build your project
Navigate to **File** > **Build Settings**.

1. Select **Build**, or choose a device and select **Build and Run**.

Importing Unity Package
• Open Hello Cardboard Scene in the project window > go to Hierarchy of the Hello Cardboard scene
window

• Go to Hierarchy window > remove CubeRoom, Point light, and Treasure from the Hierarchy > Add (+
sign) > Light > Directional Light to the hierarchy > change the dark color skybox > G to the lighting
settings (see right bottom corner of the main scene window) > Lighting > Environment > skybox Material
> Choose default skybox light for the project.

VR App Development inducing motion sickness in Unity

• Download Polygon Nature 1.1 Unity package to create new environment for the project > Add this package
to the current project.
• Go to Polygon Nature Asset > Scene > open - DemoScene > now see on screen low poly environment
appears on the screen.

Setting Up Player Movement

• Go to Hierarchy > See the Player inside hierarchy > select camera of the Player > Tracked Pose Driver
Script is very important for the game to identify the head position, and rotation using mobile phones.
• First-person camera movement: Create an empty object in the hierarchy and give it a name playerbody
> drag and drop it to the Player > go to its inspector section > Add rigid body and Character controller

component to our empty game object named playerbody

Player Body Movement Script

- Create C# for player body movement:
- Drag and drop this 6) scrip to the playerbody object.

• Go to playerbody object inspector > drag and drop playercontroller script to the controller and character
controller given in inspector of playerbody.
• Measure the speed of player movement at which the user starts feeling motion sickness i.e. discomfort,
dizziness, nausea, vomiting etc.

playerBodyMovementScript

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class playerBodyMovementScript : MonoBehaviour
{
    public CharacterController controller;
    public CharacterController characterController;
    public float speed = 300f;
    public float gravity = -9.8f;
    public float Gravity = 9.8f;
    private float velocity = 0;

// for collision

    private Rigidbody rb;
    public float movementSpeed = 5f;
    // Start is called before the first frame update
    void Start()
    {
        characterController =
GetComponent<CharacterController>();
// for collision

        rb = GetComponent<Rigidbody>();

    }

    // Update is called once per frame
    void Update()
```

```
    {
        float x = Input.GetAxis("Horizontal");
        float y = Input.GetAxis("Vertical");

        Vector3 move = transform.right * x +
transform.forward * y;
        controller.Move(move * speed * Time.deltaTime);

        if(characterController.isGrounded){
            velocity = 0;
            velocity -= Gravity * Time.deltaTime;
        }
        else{
            characterController.Move(new Vector3(0,
velocity, 0));
        }

        Vector3 movement = new Vector3(x, 0.0f, y);
        rb.AddForce(movement * movementSpeed);
    }
// for collision
     void OnTriggerEnter(Collider other)
    {
        if (other.gameObject.CompareTag("Pickup"))
        {
            other.gameObject.SetActive(false); // Disable
the pickup object on collision
        }
    }

}
```

• Add some random objects in the scene to interact/collide with player and remove after some time.
• Add particle system for the object interaction task for motion sickness experience > Particle system
(Particle system)

4.1.5 Main Camera Movement Script

- Create a camera movement script for our main camera:

cameraMovement script

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class cameraMovement : MonoBehaviour
{
        public Transform FPcamera;
        public Transform cameraBody;
        public Transform PlayerBody;
        public float x;
        public float y;
        public float z;
    // Start is called before the first frame update
    void Start()
    {

    }

    // Update is called once per frame
    void Update()
    {
        PlayerBody.transform.rotation = Quaternion.Euler(0,
FPcamera.transform.rotation.eulerAngles.y,
FPcamera.transform.rotation.eulerAngles.z);

        cameraBody.position = PlayerBody.position + new
Vector3(x, y, z);

    }
}
```

- Drag and drop this main camera script to the main camera in the hierarchy window.
• go to main camera inspector > drag and drop Main camera inside to the FPcamera variable > drag and
drop Player class to the Camera body and > drag and drop player body to the player body variable in
the main camera inspector window.

• Play a Game and test whether it is working as per scripts, if not you can change and debug your controller
script as required.
- In the playerbody setting > tick mark on Is kinematic.

Built and run the VR game

• Open the Project Settings window (menu: Edit > Project Settings) > Select XR Plugin Management >
tick on Cardboard XR Plugin.
• Go to File > Build Settings > Switch to Android >
• Go to Player Settings > Go to Resolution and Presentation section > uncheck optimized frame pacing >
Native aspect ratio > Orientation - default orientation Landscape Left >
• Go to Player settings > Other settings > Grahpic API - remove Vulkan API > Minumum API level
- Android 7.0 Nougat (API level 24 or higher) > Target API level - Automated (highest installed) >
Scripting backend - IL2CPP > ARM 64 - tickmark on it. Please see figure 8 for the build settings.
• Go to Player settings > Publishing settings > Build > change three settings as given in figure 9.
- Enable Developer option in your mobile.
• Building the game and installing the APK on a mobile phone for testing.
• Try to run the game on your mobile and get a 3D VR experience using Google Cardboard any other
compatible device.

Url: https://mega.nz/folder/BY0m0LKb#chrwOCTNkkOcePlJ87j7pQ

Output: