Google Cardboard & Virtual Reality

EEC 687

MIDTERM REPORT

Jaimal Bhairavia | 2668554

Arpit Patel | 2657673

October 26th, 2016

Abstract

Virtual Reality can be referred to an advance computing technology which provides the user with an environment wherein the user can experience his or her physical presence amidst it. It is like a realistic and mesmerizing 3D environment created using some interactive software. Unity is a one of the most popular platform for developing games for different platform such as Android, iOS, Xbox, PC and so on. Its sounds fascinating when it comes to developing a game. Our primary purpose is not limited to explore Virtual Reality, Google Cardboard and Unity but utilize the knowledge gained so far and making a Virtual Reality game that we call MazedOut.

1.0 Virtual Reality

In its mere form, virtual means something which you see but doesn't exist in real. Similarly, **Virtual reality (VR)** is a computer-generated environment that lets you experience a different reality with the sense of vision, hear, touch and smell in some cases. It includes simulating user's physical presence in this virtual environment. Instead of viewing a screen in front of them, users are immersed and able to interact with 3D worlds. VR are displayed either on a computer monitor, a projector screen or in most cases virtual reality headsets also called as **head-mounted display (HMD).** [1]

Digging deep, virtual reality is different than augmented reality in a manner explained further. In Augmented Reality, the computer uses sensors and algorithms to determine the position and orientation of a camera. AR technology then renders the 3D graphics as they would appear from the viewpoint of the camera, superimposing the computer-generated images over a user's view of the real world. Whereas, in Virtual Reality, rather than locating a real camera within a physical environment, the position of the user's eyes is located within the simulated environment. If the user's head moves, the graphics react accordingly. Rather than developing virtual objects and a real scene, VR technology creates a convincing, interactive world for the user. [2]

Speaking about the scope of VR, we feel it is the future. Gaming, Entertainment, Engineering, theme parks, and education and training are a few fields to name, where presently VR is implemented. PlayStation VR headsets, VR installations in roller coasters, incorporating virtual reality to add more dimension to design the prototype of devices related to automobile, aerospace and ground transport industries, using VR as a platform to train people to be pilots, doctors, etc., are real world example and with further developments, we would have a lot to look forward to.

2.0 Applications of VR

In the education field, VR is used by trainers to conduct experiments without stepping out in the real environment. For example, The Civil engineering department at our university has a VR Software/Hardware where in a person can sit in a real car and pretend to be driving with all its attributes and the monitor in from of the driver displays the driving environment and makes the driver feel like he is driving in the displayed conditions. This way, the trainers can utilize the virtual setup to experiment with different driving styles and issues related to it. Video games in VR are one of the most popular application which use the graphics, sound and input technology. Wii remote, Kinect and PlayStation are most popular VR platforms for games which can track and send motion input of the player. VR is also used in the Engineering like in CAD and printing the 3D objects. Its includes one more dimension to virtual prototyping which enables engineers to view the objects from different scenario so they can build and test things without going to actual scene [4]. The 3D views enable interactive viewing of different places. The best example are some museums that provide a walk through guide. VR is also being used in urban design and planning, the software developed in 2010 has ability to locate geometry coordinates and perform activities like designing the different views, surveying the design and testing different component. Software was capable enough to draw lines and curves on subdivision plats and surveying plans. Moreover, VR has been installed in different roller coasters to ameliorate the adrenaline rush on particular rides.

3.0 Google Cardboard

It is a wearable device developed by google in 2014 which acts as a gateway to experience virtual reality in a simple, fun and affordable way. It holds a smartphone in the back on which the actual VR application is running. This application is then viewed through the cardboard to get a VR experience. Cardboard is intended to low cost, to encourage interest and development in the Virtual Reality ground. The cardboard also has a clicker button which when pressed, the screen is tapped using the internal cushion connected to this button. We are going to use google cardboard as our display tool for the game we have developed.

4.0 Software Development Kit

A software development kit (SDK) is typically a set of software development tools that supports the creation of applications for a certain software framework, hardware platform, computer system, video game console, operating system, or similar development platform. To create applications on any platform, one needs to download a specific software development kit. For example, the development of an Android app requires an SDK with Java, for iOS apps an iOS SDK with Swift. It may be something as simple as the implementation of one or more application programming interfaces (APIs) in the form of some libraries to interface to a particular programming language or to include sophisticated hardware that can communicate with a particular embedded system. [4]

Google provides distinct development kits for developing VR application on different platforms: SDK for android, SDK for iOS, SDK for Unity and SDK for Unreal. Android basically works on java codes. iOS application requires codes to be written in objective C. Unity is a cross platform game engine developed by Unity Technologies which can be used to develop 2D and 3D games for PC, Consoles and Mobile devices. The coding for Unity components is done in C#. Unreal engine is another game engine developed by Epic Games which is primarily used for developing first-person shooting games, other general games have also been developed on this platform. The code is written in C++. We have used the Google cardboard SDK for unity to develop the application.

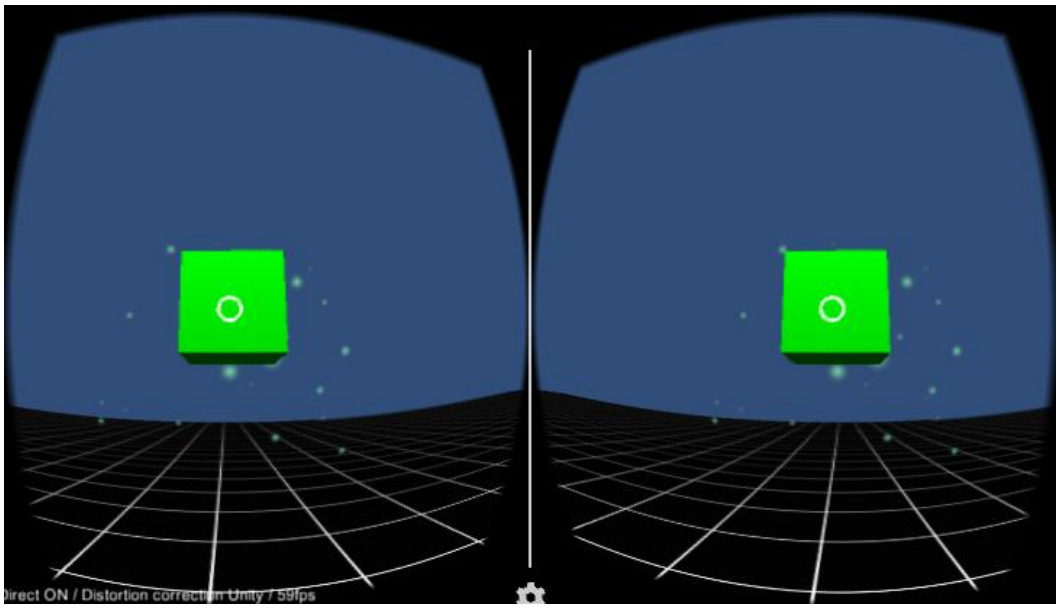
4.1 Google Cardboard SDK components

4.1.1 GVR main

Google Virtual reality Main is a pre-fabricated component of the SDK that replaces the normal main camera in the unity project. This prefab contains the main object known as GvrMain which has a script attached to it which controls the basic VR mode settings. Further it has a head object as a child. The head further has main camera as a child object with the stereo controller script attached to it. Finally, the main camera has left and right stereo eye camera.

4.1.2 GVR Reticle[5]

Google Virtual Reality Reticle is a prefab that adds an interactive gaze based reticle. It behaves like a small pointer in the local space that casts itself exactly in the position of the user's focal focus, and grows when it intersects with an object. GvrReticle is a class file which inherits from MonoBehaviour class file. It also has some functions of its own that can be utilized towards the development. From those function, we can adjust the reticle growth speed, enable/disable gaze, setting up event when user intersects an object with a ray projected form the camera.



4.1.3 GVR View[6]

It's mostly used to track the movement of the head and provide the most recently updated view to user. This class is also inherited from MonoBehaviour class. It's designed to work in full screen mode with a full landscape or reverse landscape orientation. Also, it can be implemented in two different views which are Renderer and Stereo Renderer. Gvr View always has a stereo render because the stereo pair when viewed through the lens in the cardboard looks like one single image.

5.0 Exploring 3D View with Unity:

5.1 Rigid body physics

As objects in a game are strongly connected with real world objects in terms of depicting them, Unity uses Nvidia's PhyX engine. The game objects which are directly connected with physics requires high computation power and processing speed. There is separate physics engine which controls all rigid body components which may have some properties like Mass, Gravity, Velocity and Friction.

5.2 Skybox

Skybox is an asset which includes a sky wrapped around the entire global space that can be seen by user while plying open roof games. It has several properties such as color, exposure, brightness, rotation etc. The most important property of the skybox is, it can be seen from all non-opaque objects in the scene.

5.3 Collision detection

Collision detection of the reticle with the object becomes an important aspect when we talk about games because there are moving objects in world space and events attached to it. In Unity, we can assign Collider component which has invisible net on edges and reports if any collision occurs. A simple example is one from our demo app. The event of the cube changing color when the reticle intersects with the cube is because of the collision being detected and a certain action been taken on that event.

5.4 Physics Raycaster

In Unity, there is event system which requires a function that detects the dyanamic input events which records them to be used by the event system in the scene. For this setup, we use the physics raycaster which is generally used for 3D objects. PhysicsRaycaster class can be implemented from BaseRaycaster class and it has four properties such as Depth, to create the 3D environment, Event Camera, Event Mask that allows the events to be captured and Final Event Mask.

5.5 Gaze Input Module

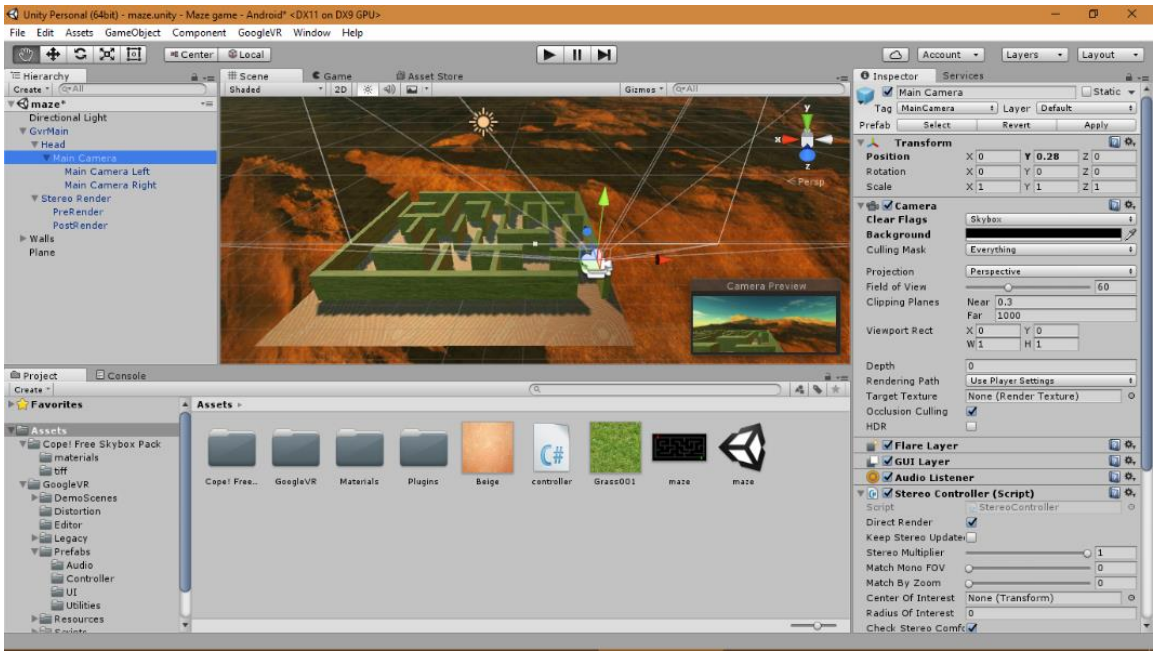
It is a script which provides an implemention of unity's BaseInputModule class. Gaze Input Module allows selecting canvas based UI elements by looking at them or by touching the screen when they are in the focal focus of the user. It supports several events like enter, exit, up, down, delete select and update select with respect to the event system.

6.0 MazedOut

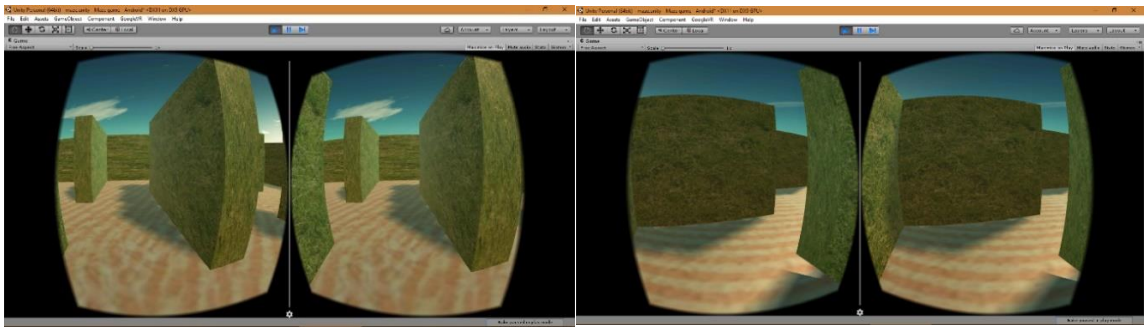
We have named our application MazedOut. The gameplay includes locating and collecting all different kind of objects existing in the maze and exit the maze in the least time possible. Every iteration of the game played would have the scored displayed after the game finishes. We can increase the complexity in terms of the scale of the maze, time, objects to be collected & avoided and so on.

7.0 Work completed under MazedOut

The application developed comprises of a structure like a maze. The character in the first person view travels through this maze. For now, we have incorporated the character to move automatically. This can also be called as an auto walk feature. Another feature implemented is that the character stops walking when you look at the ground. So, when the user lifts the vision from the ground, the character resumes walking from the same position. Also, if the character comes out of the maze successfully, the game starts over again from the start point. This also avoids the character from exiting the local space. All of this was achieved by writing different function in a script in coded in C#. The project and the game windows for app are shown below.



Project Window



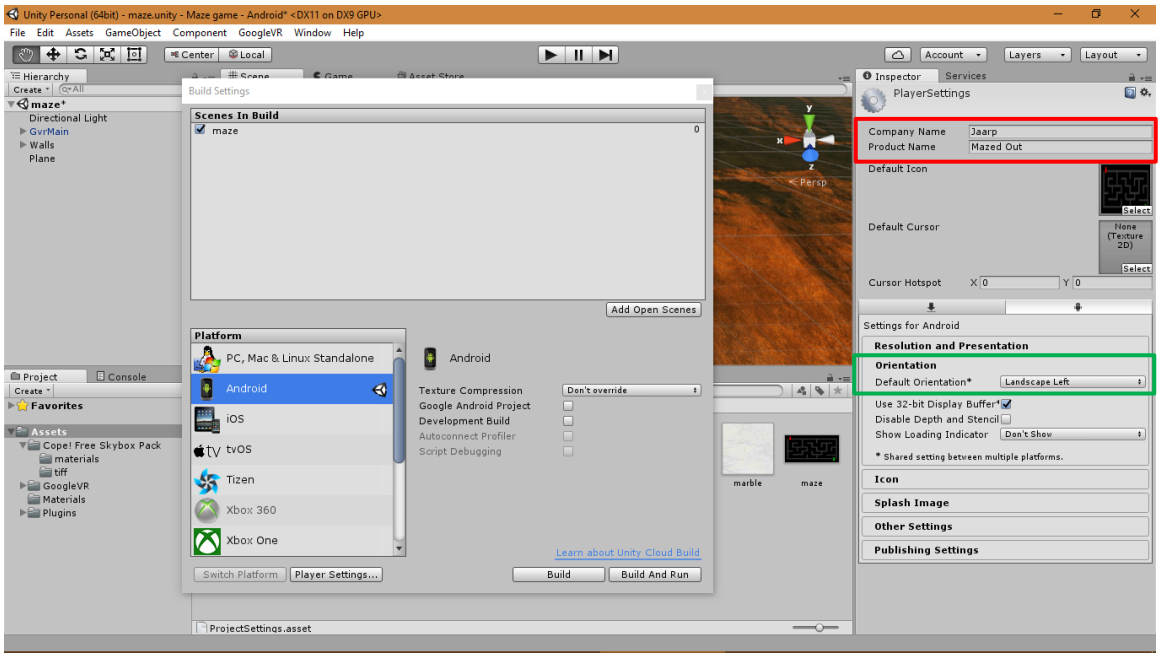
Game windows

Features to be further implemented include locating coins & gifts present in the maze and capturing them using the click button in the cardboard. The cardboard is designed in such a manner that this click button presses the screen. Hence, it works exactly like a finger pressing a point on the screen. Eventually, collecting the coins and gifts while navigating to the end of the maze in the minimum time would complete the game. Also, after the completing the game, the time taken to completed the game would be displayed with the score in accordance to the number of coins & gifts collected.

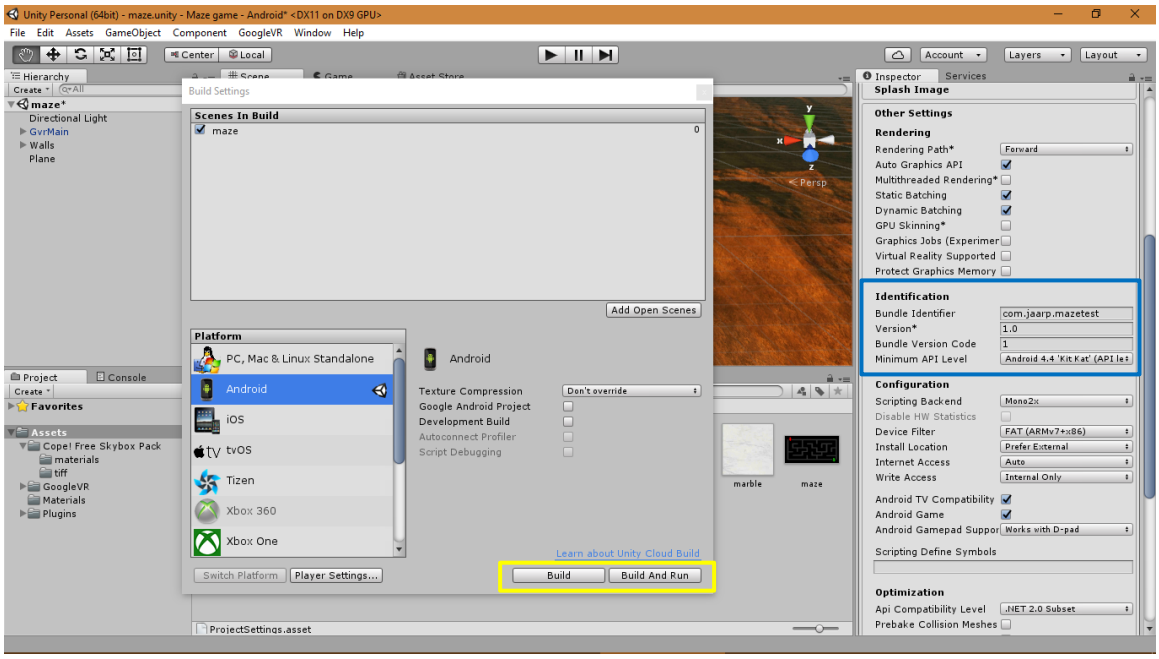
8.0 Building the application for android:

Any application developed in unity can be built for a variety of different platforms. Since we are working with the cardboard, our concentration is going to be on mobile devices. Here we are going to demonstrate the method to build an application for an android operating device. The player settings need to be edited to achieve successful operation of exporting and running of the application. Firstly, one needs to enter the name of the company and then the product name which would be the name of the application. The portion highlighted in Red depicts these settings.

Next is the orientation setting that needs to be set to landscape left since the phone must be placed in the cardboard and it'll be in the landscape mode. This setting is visible in the workspace highlighted in green



Player settings



Player settings

The most important part of the player setting is the identification setting. They are highlighted in the blue portion. where in the developer needs to define a bundle identifier whose format is pre-defined. The version depicts the version of the application. This is like different versions of software's that we use. For example, apple released iOS 10 in

September and the second version of the software i.e. 10.0.2 in oct fixing a few bugs. Since this is the first version, we have kept this field as 1.0 and in further development would be changing this as required. The minimum API Level indicates the minimum version of the android OS the application would support. Since google advices to build a VR app on a minimum API level of Android 4.4, we have tried implementing that.

Unity gives the developer two options for building the app. The portion in the yellow area depicts the two options. Build is used to build an apk file of the final application which can be transferred to the phone. Alternatively, build and run is used to run the app on the phone while it is connected to the computer via unity. The former option is great when the application needs no modifications and is ready to be exported to the device. Whereas the idea behind the latter option is to facilitate the developer with a hands-on experience of how the app would perform on the device and make any changes needed. Hence, the developer can modify or edit the app and its attributes before creating the final apk file.

9.0 Issues encountered during development

As every project development procedure encounters a fair share of problems to achieve the desired, similarly we came across a few problems too.

- Starting with the reticle function, we couldn't make it work correctly until we created an event system and then a gaze input module to detect the collision and take an action if the event occurred.
- The next was the auto stop feature implemented. It seemed easy in the beginning but the script had to be adjusted per the coordinates of the ground. A lot of trial and error went into solving this.
- Another problem that persisted the longest was exporting the application onto the device. Since it was a technical issue, we tried a list of different things. We tried using different devices with the same OS version. But that dint work. So, we tried a device with a higher android OS version (5.0). For some reason the performance of the app was not as desired on this version. Finally, we could export it in a device working on marshmallow (Android 6.0). We tried playing the game on 2 different devices with the 6.0 OS and it worked completely fine.
- Making the game object opaque by changing their properties is also a problem faced by. We are trying to work on it and hopefully solve the problem before the final version of the game is developed.

10.0 Future work under MazedOut

- Expscale of the maze to fit in several objects.
- Facilitate the user locate coins & gifts in the maze and capture them using the click button in the cardboard.
- On completion of the game, display the score in accordance to the number of objects collected and the time taken to finish.
- Determine the parameters to calculate the score.

References

1) http://www.cnet.com/special-reports/vr101/
2) http://www.marxentlabs.com/what-is-virtual-reality-definition-and-examples/
3) https://en.wikipedia.org/wiki/Virtual_reality
4) https://en.wikipedia.org/wiki/Software_development_kit
5) https://developers.google.com/vr/unity/reference/class/gvr-reticle
6) https://developers.google.com/vr/android/reference/com/google/vr/sdk/base/GvrView