Interactive Space Exploration Video Game in Augmented Reality using Unity Tool

Name: Shanuj Shekhar

Unity Version: 2018.4.16f1 Personal

Vuforia Version: 8.3.8

Hardware used:

Laptop Configurations: Inspiron 5580, Memory: 8 GB

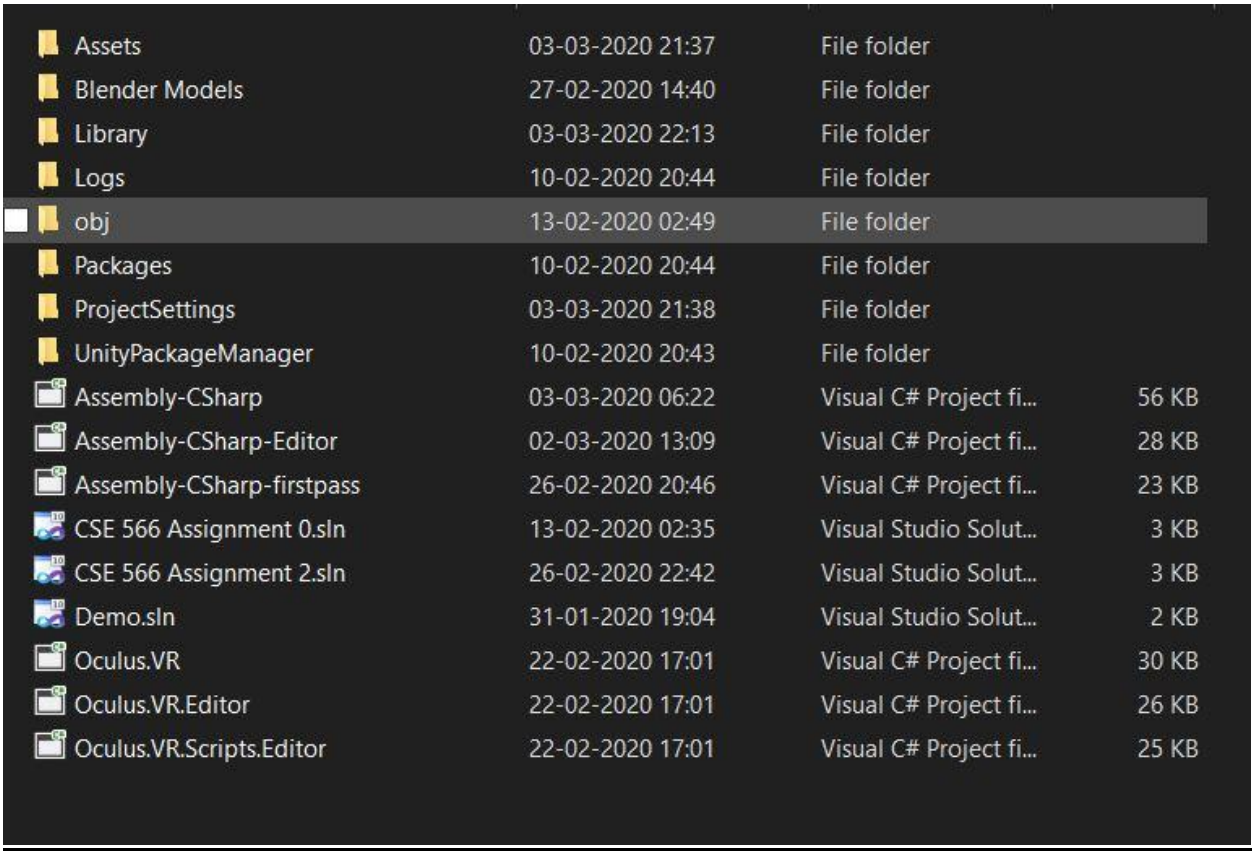
Processor: Intel(R) Core(TM) i7-8565U CPU @ 1.80GHz[Cores 4] [Logical/Core 2]

Operating System: Microsoft Windows 10 Home Single Language

Graphic Card Configurations: NVIDIA GeForce MX150, Shared Memory - 3999MB

Mobile Device Used: Google Pixel XL, Android Version 10, 128 GB ROM.

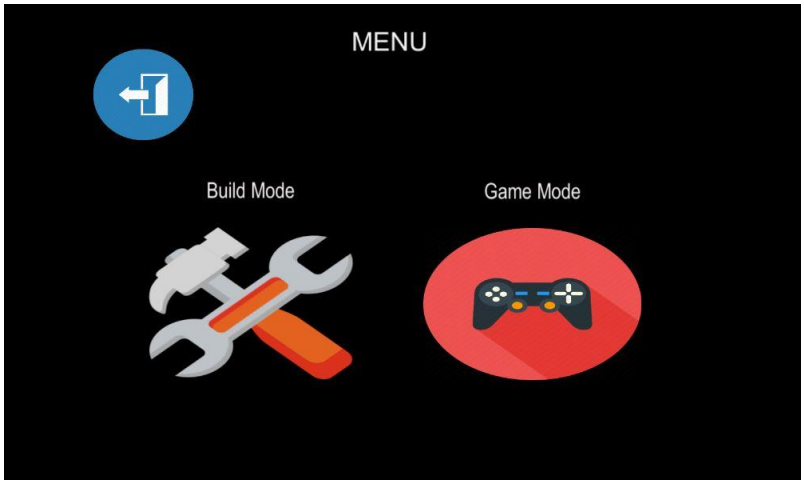
Directory Hierarchy:



Implementation Details:

I have made separate UIs for each mode.

- **Functionalities Used:**
  - I have used marker detection for plane detection.
  - All my objects in the library are the child of Marker (Image Target).
  - I have tried using ground plane detection using markerless detection but the plane was very unstable and 90% of the time, the plane was not detected. When it was detected, the plane always used to drift away. I have tried many workarounds, like using different android devices of versions 9, lighting of plane surfaces, choosing a bigger plane but nothing seems to work. I have also asked the TA of this course about this issue, but no solution so far, so I choose to do the Image Target option for plane detection.
- **ModesUI-** This UI comes at the start of the application. It has 3 buttons displayed on the screen.
  - The first is the **Exit** button which on click will exit the application.
  - The second button is for Build Mode which on click enables the **BuildUI** and performs its respective functions.
  - The third button is for **GameUI** which enables the user to go into the GameUI and drag cargo boxes into the greenhouse.
  - The figure belows the GameUI which has two buttons for build and game mode.
  - Also, I have extra functionality of exiting the application by clicking the blue exit button on top right of Game UI.



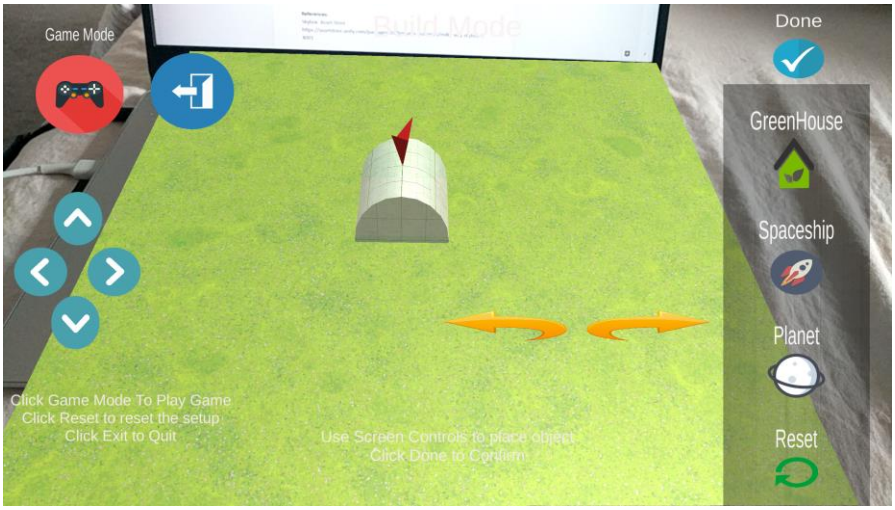
Game UI

- **Build Mode UI-**

Build UI

  - In my build mode, there is a library of objects from which you can choose to place in the real-world.
  - As you can see from the screenshot above, on the right hand side, I have a library of objects to place on the green plane detected.
  - You can switch to **Game Mode** by clicking the red button on top left of the screen.
  - **Placing Objects: PlaceObject.cs** script is used for placing every object. I have used Raycast in the script to select the object to be placed. This raycast goes from the camera of the device to the direction of the mouse or pointer clicked. Once this object is placed at desired position, the user has to click the **Done button** to confirm.

    - **GreenHouse:**
      - When the greenhouse from the library of objects is selected, a red marker arrow is displayed on the greenhouse, the UI for greenhouse is displayed. Since the greenhouse can only be translated on the plane, so I have put colliders on all the sides of the plane so the greenhouse cannot be moved outside the plane. (**CollisionCheck.cs** script)



Greenhouse Placement UI

- From the figure above, the red marker is above the greenhouse to depict this is the current selected object. On the left side, the controls are for translation of greenhouse on the plane in x-y direction only and on the right side, those two yellow arrows are for rotating the object about the normal direction of the plane. Click Done once confirmed.

■ **Spaceship:**

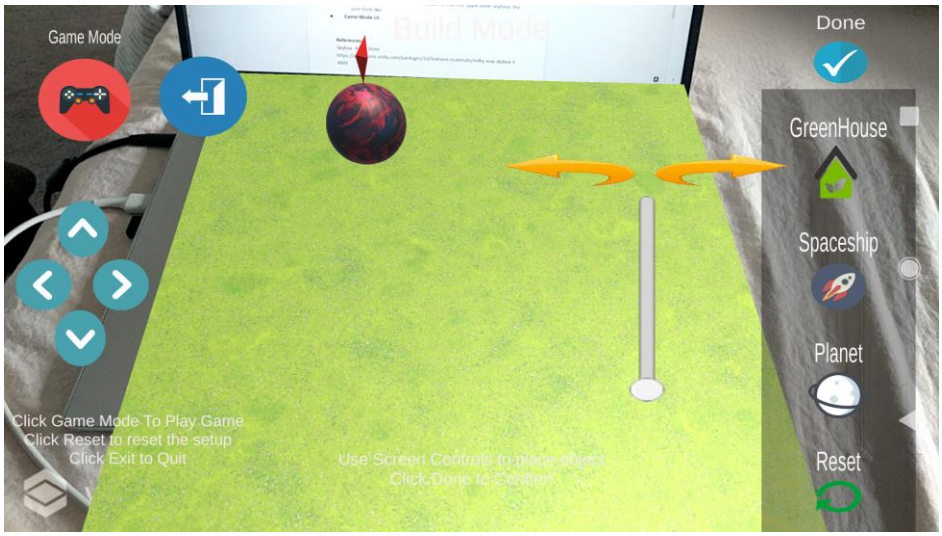
- When the spaceship from the library of objects is selected, a red marker arrow is displayed on the spaceship, the UI for spaceship is displayed. Since the spaceship can be translated anywhere (perpendicular to the plane) but on the plane.

Spaceship Placement UI

- From the figure above, the red marker is above the spaceship to depict this is the current selected object. On the left side, the controls are for translation of spaceship on the plane perpendicular to the green plane in x-y direction only and on the right side, there is a slider to change the height of the spaceship from the plane. Click Done once confirmed.

■ **Planet:**

- When the planet from the library of objects is selected, a red marker arrow is displayed on the planet, the UI for the planet is displayed. The planet can be placed anywhere in the real world except the plane surface.

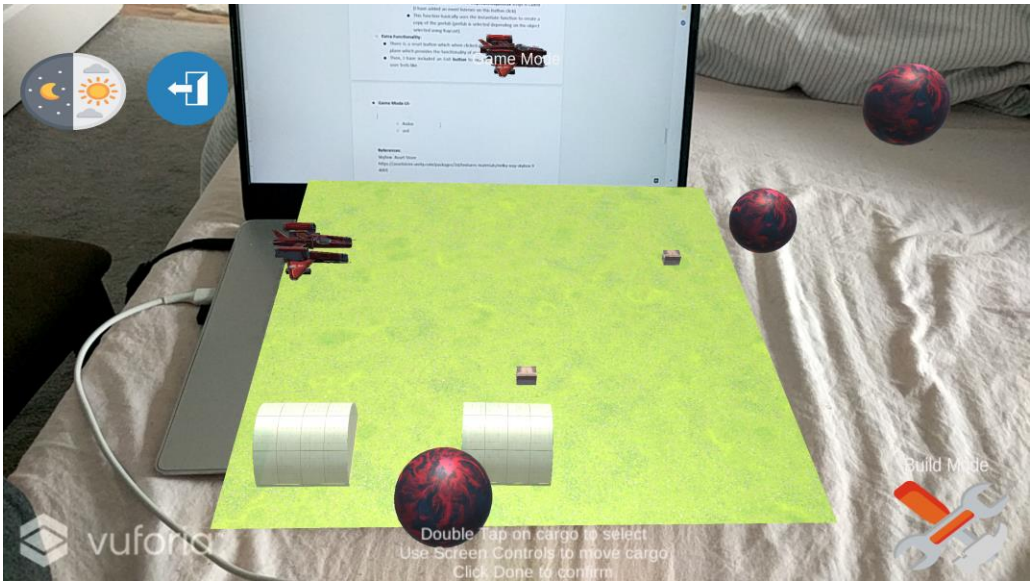


Planet Placement UI

- From the figure above, the red marker is above the planet to depict this is the current selected object. On the left side, the controls are for translation of the planet anywhere in the real world and on the right side, there is a slider to change the height of the spaceship from the plane. Also, those two yellow arrows are for rotating the object about the normal direction of the plane. Click Done once confirmed.

- **Duplicate Objects:**
  - I have made a prefab for every object in the library, i.e. prefabs are for greenhouse, spaceship and planet.
  - When the user clicks on any of the objects from the library, AddObject() function in the **DuplicateObjects.cs** script is called (I have added an event listener on this button click).
  - This function basically uses the Instantiate function to create a copy of the prefab (prefab is selected depending on the object selected using Raycast).
- **Extra Functionality:**
  - There is a reset button which when clicked destroys all the objects on the plane which provides the functionality of starting over. (**DestroyObjects.cs**)
  - Then, I have included an Exit **button** to exit the application anytime the user feels like.

- **Game Mode UI-**



Game UI

  - In this UI, there is a Switch to Build Mode button, on the bottom right side which when clicked will switch to Build UI.
  - **Spaceship & Planet Motion:** When the game starts, the planets are in constant circular motion around the center of the horizontal plane (Green Plane Detected). Spaceships are also in constant circular motion around the plane's center axis. These motions are handled in **Motion.cs** script.
  - **Cargo Drop:** This functionality is included in **Motion.cs** script. When the game starts, I have made a function **objectMotion()** which gets called every frame when the game starts. After every 13 seconds, a new cargo object is created using the prefab of cargo using the instantiate function just below the
  - There is a Toggle Day and Night button on the top left side. When this button is clicked, the directional light which acts as the scene light. This light is on initially when the game starts.

References:

Skybox- Asset Store
https://assetstore.unity.com/packages/2d/textures-materials/milky-way-skybox-94001
Earth, Moon, Mars, PlanetCSE566 - Asset Store
https://assetstore.unity.com/packages/2d/textures-materials/sky/earth-planets-skyboxes-53752

Spaceship
https://assetstore.unity.com/packages/3d/vehicles/space/hi-rez-spaceships-creator-free-sample-153363

CargoShip
https://www.3dcadbrowser.com/3d-model/barge

Cart
https://assetstore.unity.com/packages/3d/environments/fantasy/wooden-cart-65835

Plant
https://www.youtube.com/watch?v=JoW1XWi4Lw0

Portal
https://www.turbosquid.com/3d-models/3d-sci-fi-teleporter/860716

MiniMap
https://www.youtube.com/watch?v=28JTTXqMvOU
https://www.youtube.com/watch?v=faXQQ5b6GEs
https://www.youtube.com/watch?v=28JTTXqMvOU

Oculus Joystick
https://developer.oculus.com/documentation/unity/unity-ovrinput/?locale=en_US&device=QUEST
https://forum.unity.com/threads/hi-i-am-trying-to-get-movment-of-the-ovrplayercontroller-with-the-gear-vr-trackpad.507240/

Grass Texture
https://assetstore.unity.com/?category=3d&free=true&q=grass&orderBy=0