INFO 5340 / CS 5650
Virtual and Augmented Reality

Assignment 3

Objective

The objective of this assignment is to familiarize yourself with existing commercially available VR applications (part 1) and walk through the process of setting up and running a simple VR project in Unity (part 2).

You will create an interaction on your desktop device as shown in the solution screen recording. You can record your gameplay using OBS, which is already installed on the PCs in the Masters' Studio.

Assignment Task (Two Parts)

Part 1 | Evaluating Existing VR Applications

The task for part one of this assignment is to try two virtual reality applications and write an evaluation on each one. You will submit your written evaluation via GitHub with Part 2.

Application One | Google Earth VR

Open Steam and sign in with your personal account. Download and install Google Earth VR to the Steam client on one of the four designated Windows PCs in the Masters' Studio. Explore different areas for at least 15 minutes. Answer the following questions:

1. Where did you go and why?
2. Did you feel 'present' in those locations? Why or why not?
3. Did you feel motion sickness while exploring the world? If so, what do you think caused that feeling?
4. Do you consider the UI/UX in Google Earth VR to be intuitive? Why or why not?

Application Two | Rec Room

Open Steam and sign in with your personal account. Open Steam and sign in. Download and install Rec Room to the Steam client on one of the four designated Windows PCs in the

Masters' Studio. **Headphones are required for this experience, they can be found in the lockers and plugged into the cable hanging from the top of the Vive headset.** Make an account and customize your avatar. Go into the 'Rec Room' and socialize with other users. Enter and complete at least one 'activity' from the Rec Room (Paintball, Charades, Rec Royale, etc.). Answer the following questions:

1. Did you feel connected to your avatar after customizing them? Why or why not?

2. When socializing with others, did it feel as if you were connecting with them on the same level as if you were in the same physical space? Why or why not?

3. Do you consider the UI/UX in Rec Room to be intuitive? Why or why not?

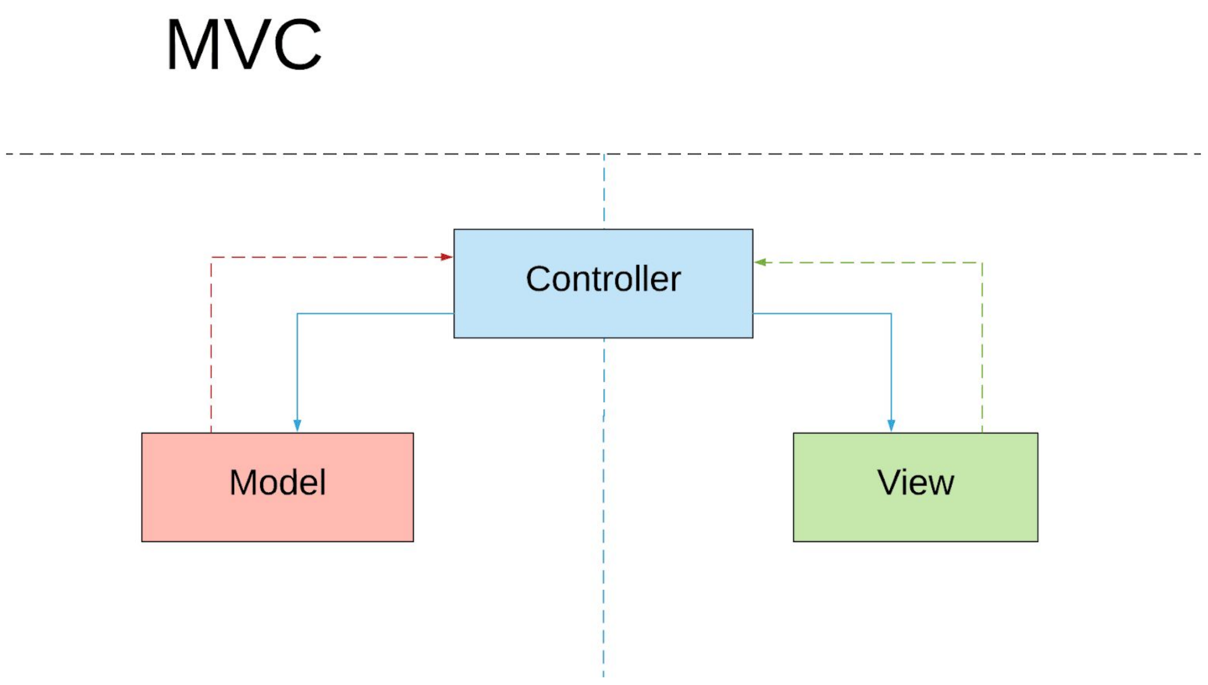
Part 2 | Building Your First VR Application

In this part you will use Unity to build a simple interactive VR application. Follow the guide below to complete all the tasks. **Tasks are in bold and highlighted in blue.**

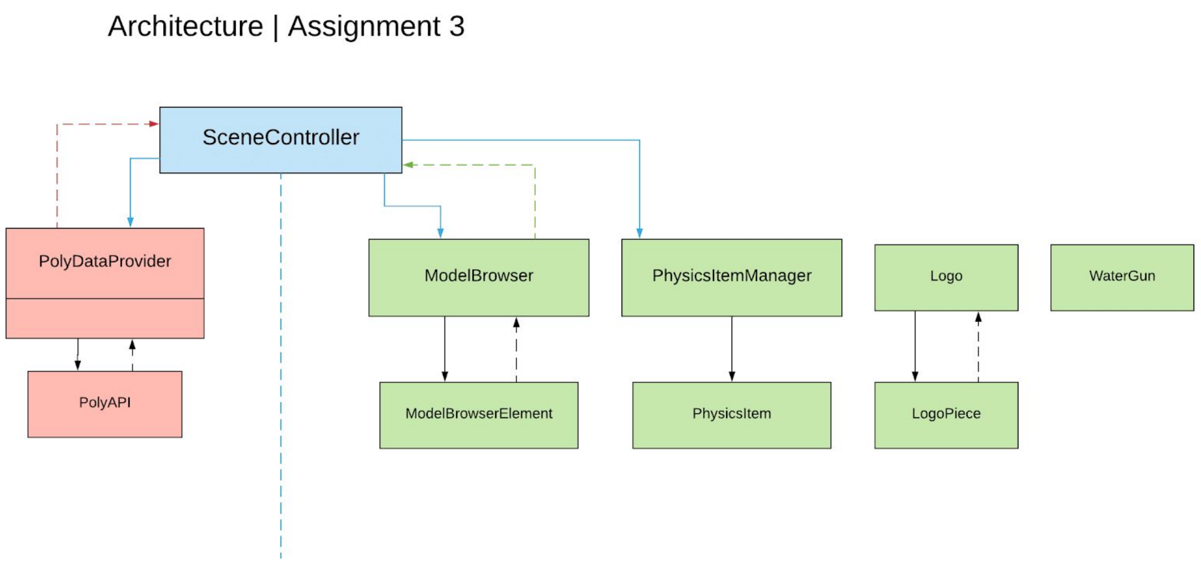
1 | Project Setup and Overview

Log into the computer using your NetID. Clone assignment-3 onto the VR computers in the Masters' Studio and open the Unity Project. Check the console for any errors relating to unsafe code. *If you see an error relating to unsafe code open the project's Player Settings and uncheck then recheck the 'Allow unsafe code' checkbox located under 'Other Settings'*.

Open the MainScene (Assets/Scenes/MainScene), you will only be using this scene for the assignment. The scripts in this scene are set up to follow MVC (Model-view-controller) architecture.



Below is a high-level architectural diagram of the scene's classes:



Solid arrows represent a reference, either by SerializedField or GetComponent.

Dotted arrows represent information being 'bubbled up' via Actions/Delegates (without a direct reference). You can read more about actions here.

This information is provided to help you better understand the relationship between classes, you will not need to modify the scene architecture in any way.

2 | Customize The Environment

[2.1] Assign a custom skybox material to the scene's lighting window. You can find five skyboxes to choose from in the Vendor/Allsky/ folder. **[2.2] Modify the fog color in the lighting window.**

[2.3] Customize the color of the floor using the existing FloorMaterial.

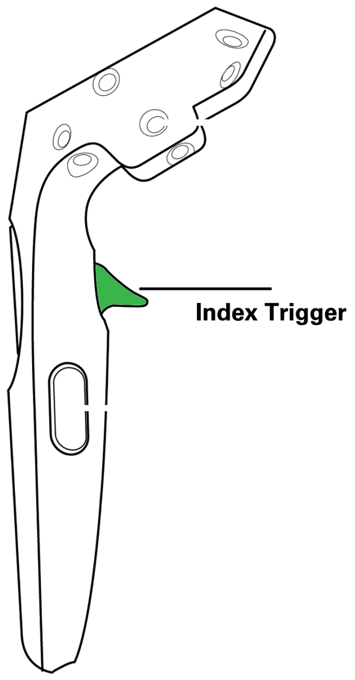
3 | SteamVR Setup

Drag the SteamVR Player prefab into the scene as a child of the 'Cameras' GameObject, it can be found in "Assets/SteamVR/InteractionSystem/Core/Prefabs". Ensure the Player transform is recentered under the 'Cameras' GameObject, shortcut function for doing so can be found under the menu in GameObject > Reset PSR.

Disable the 'Main Camera' GameObject.

Ensure you have Steam running, sign into your account if you have not done so already. Open SteamVR by clicking the [VR] icon in the top right corner of Steam, or finding SteamVR in your library under 'Tools'.

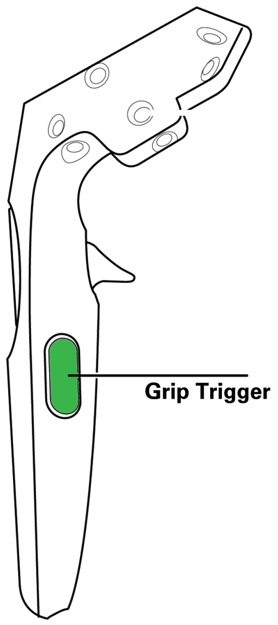
Hit play in the editor and put on the Vive headset, you should be able to walk around the environment and see your controllers/hands (if they are powered on). Walk over to the virtual table and pick up the cube and water gun using the index trigger.



3.1 | SteamVR Input Actions

You will now need to enable use of the grip trigger to pick up objects using SteamVR input actions. To do so you will need to utilize SteamVR Input. Open the SteamVR Input window (Window > SteamVR Input). Click the plus icon under the ‘In’ list. Name the new action “GrabGrip”, leave the rest of the settings as the default. Click ‘Save and generate’ at the bottom of the window.

Hit play in the editor and put on the Vive headset, you should now be able to pick up the virtual objects using the grip trigger.



4 | Importing a Poly asset

Using the Poly Toolkit for Unity your task is to import a Blocks model into the scene. Start by opening the Poly Toolkit window from the menu bar (Poly > Browse Assets). Set ‘Asset type’ to ‘Blocks’ and select an asset you like. Write down the attribution information somewhere for later use (Title & Author). Click ‘Import into project’, this will place the asset in your Assets folder and Instantiate it in the scene.

[4.1] Position the object so it is floating slightly above the table, but not intersecting with the table. Scale the GameObject so that it is roughly the same size of the cube that is on the table.

Add the attribution information from the previous step in the ‘YourAttributionText’ GameObject’s Text component. **[4.2] Customize the font of the Text component using the fonts located in Assets/Materials/Fonts/.**

5 | Poly asset physics

[5.1] Add physics to your Poly asset so that it falls on the table: Add a RigidBody component to your Poly asset’s GameObject. Add a MeshCollider component to each child of the asset GameObject that has a MeshFilter. Check the ‘convex’ checkbox on the MeshCollider components.

Hit ‘Play’ in the Editor and ensure that the GameObject falls and rests on the table. (Viewable from the scene window, no need to put on the headset).

6 | Poly asset Vive interaction

[6.1] Add the capability to pick up and throw your Poly asset:
Add an ‘Interactable’ component to your Poly asset GameObject.
Add a ‘Velocity Estimator’ component to the GameObject.
Add a ‘Throwable’ component to the GameObject.
Change ‘Release Velocity Style’ to ‘Short Estimation’, this will enable more realistic throwing by estimating the velocity of your hand at the time of release. Check the ‘Restore Original Parent’ checkbox.

Hit ‘Play’ in the Editor, put on the Vive headset and ensure you can grab and throw the asset with the Vive controller, using the grip or trigger buttons.

7 | ModelBrowser setup

Find the ModelBrowser GameObject in the scene and set it to active. Open the SceneController script. **[7.1] Call the ‘LoadPolyAssets’ function from inside the ‘Start’ function.**

If you enter play mode in the editor you should now see the ModelBrowser fill with thumbnails from Google Poly.

Using the same instructions from step 3.1, add a SteamVR input action named “InteractUI”, save and generate the changes.

Select the ModelBrowser GameObject and assign its reference to the attribution text (The one titled ‘ModelBrowser_AttributionText’ not the one from task 4.2). Open the ModelBrowser script. **[7.2] Complete the ‘DisplayAttributionText’ function so that it displays the attribution text contained in the ModelViewData parameter.** It should be formatted like so: “*name* by *authorName*”.

You should now be able to point at browser elements with the Vive controllers and select them with the attribution text on top of the browser updated after each click. *NOTE: Some models will not load due to an error with the Poly Toolkit, you may need to account for this in future steps.

8 | PhysicsItemManager Setup

Find the ‘PhysicsItemSpawnAnchor’ GameObject in the scene and assign its transform to the PhysicsItemManager’s serialized transform field. The following will change the spawn position of physics items from inside the floor to above the half pipe.

Open the PhysicsItemManager script.
[8.1] Fill in the ‘SetupGameObject’ function so that it does the following:
- Adds a ‘PhysicsItem’ component to the ‘gameObject’ parameter
- Sets the parents of the ‘gameObject’ parameter’s transform to the PhysicsItemManager’s transform.
- Sets the ‘gameObject’ parameter’s transform position to the position of the ‘itemSpawnAnchorTransform’.

9 | Enabling physics for PhysicsItem

[9.1] Now let’s make sure our *physics* items follow the laws of *physics*! The items need to fall from their spawn position and collide with the half pipe. Open the ‘PhysicsItem’ script. Fill in the ‘AddPhysics’ function so it does the following:
- Adds a Rigidbody component to the GameObject the script is attached to, and assigns the return value to the global ‘rigidbody’ variable
- Adds a MeshCollider to each child of the GameObject the script is attached to, only if it has a MeshFilter
- Make sure ‘convex’ is set to true on each MeshCollider

Hit ‘Play’ in the Editor, put on the Vive headset and ensure you can spawn items by pointing at the menu items with the Vive controller, and clicking the trigger button. The physics items should fall onto the half pipe.

10 | Making the half pipe slippery

Create a new PhysicMaterial in your Assets folder under Assets/Materials/PhysicMaterials/. Select the HalfPipe_Mesh GameObject. Assign your PhysicMaterial into the MeshCollider’s ‘Material’ serialized field. **[10.1] Adjust the values of the PhysicMaterial so that spawned assets slide and land on the platform.** (Friction values and FrictionCombine value).

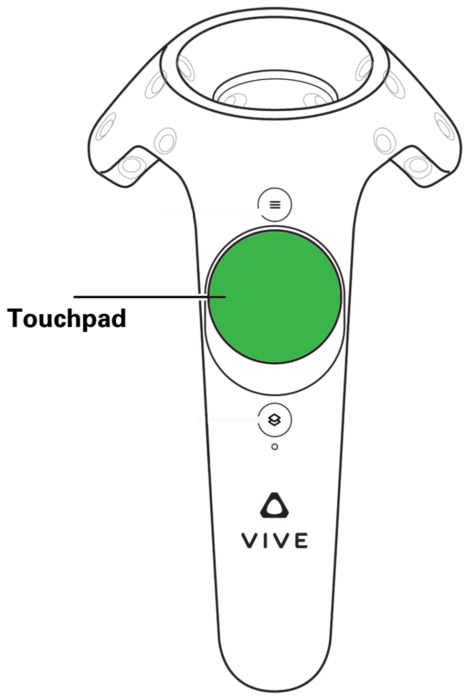
11 | Making the PhysicsItem throwable

Open the ‘PhysicsItem’ script. **[11.1] Fill in the ‘MakeThrowable’ function so it does the following:**
- Adds an ‘Interactable’ component to the GameObject the script is attached to
- Adds a ‘Velocity Estimator’ component to the GameObject the script is attached to
- Adds a ‘Throwable’ component to the GameObject the script is attached to and modify the following values:
    - Change ‘releaseVelocityStyle to ‘ReleaseStyle.ShortEstimation’
    - Set ‘restoreOriginalParent’ to ‘true’

Enter play mode and ensure you can spawn assets by pointing at the menu items with the Vive controller, and clicking the trigger button. You should now be able to pick up and throw the GameObjects.

12 | Physics item scaling

Now let's add the ability to scale the currently held physics item using the Vive controller's touchpad.



Using the same instructions from step 3.1, add a SteamVR input action called "ScaleItem", but this time its Type should be Vector2. Provide a localized string. Save and generate the changes. You will need to open the binding UI and assign an input source for the action (Trackpad). For further guidelines go here.

Open the PhysicsItem script and find the UpdateScale function. **[12.1] Fill the function so that it scales the script's attached transform based on the user's input along the Y axis of the assigned input source.** This logic should scale the (x, y, z) values of the localScale Vector linearly with the axis' delta, keep in mind this can be both positive and negative so you may need to restrict the minimum localScale.

13 | Physics item animation reset

Open the PhysicsItem script and find the ReturnToOrigin function. **[13.1] Fill in the function so that it does the following:**

- Waits three seconds
- Disables the rigidbody (helper function included)
- Animates to the original spawn position, rotation, and scale
- Enables the rigidbody (helper function included)

When you throw a PhysicsItem on the floor, it should now start animating back to the top of the slide after three seconds, then fall onto the slide.

14 | Logo functionality

Find the ‘CornellTech_Logo’ GameObject in the scene and set it to active. Open the LogoPiece script. Find the RandomizeColor function. **[14.1] Fill it in so it assigns a random color to the global material variable.**

Logo pieces should now change color when they collide with something (try throwing an object at it).

Open the Logo script. **[14.2] Add the necessary functionality so that the following behaviour is performed:**

- After OnCollisionEntered is called five times, call FallApart
- FallApart should loop through the ‘pieces’ array and call AddRigidbody only if the piece element in the array has the ‘ShouldFall’ property set to true.

The logo should now fall apart after it collides with objects five times.

Bonus Tasks

1) When you pick up a PhysicsItem make it change color. When you release the GameObject it should return to its original color.
2) Add scrolling functionality to the UI of the ModelBrowser so that it can display more than 12 elements. Load more than 12 elements by modifying the SceneController script.
3) Modify the LogoPiece script so that it only changes color when it collides with a PhysicsItem, not when it collides with the floor, platform, or another logo piece.

Reference

SteamVR

SteamVR is Valve’s SDK for developing and running virtual reality applications for various headsets and operating systems. SteamVR is built on top of OpenVR, an open source SDK built by Valve. Steam is required to run SteamVR.

To find answers to various questions you can visit the SteamVR developer discussion forums here.

SteamVR Unity Plugin 2.0

As of 9/21/18 the SteamVR plugin in the Unity Asset Store has been updated to version 2.0, which overhauled the existing input system, making it easier for developers to account for different controllers and input configurations. You can read more about the new plugin here and here.

SteamVR Input

For this project we will use the new SteamVR Interaction System, you can learn more here: https://steamcommunity.com/sharedfiles/filedetails/?id=1416820276

Good luck!