CS 498 Virtual Reality

MP2 Scripting and Depth Perception

Released 16 September 2015

Due 29 September 2015

Overview

This assignment will demonstrate the difference between rotation and position tracking. It will help you appreciate how much depth perception relies on both of these, and teach you how to enable and disable them. It'll also help you understand the rigid body transformations covered in class.

We have provided you with a .unitypackage containing a scene('MP2'). The provided scene contains an OVRCameraRig prefab, which doesn't allow player movement but still uses DK2's rotation and position trackers to modify camera position and rotation. The prefab contains scripts (OVRCameraRig & OVRManager) and gameobjects that you will extend in the following ways:

- Pressing tab should reset the camera position to (0,0,0).
- Pressing the R key should toggle rotation tracking on and off.
- Pressing the P key should toggle position tracking on and off. [EXTRA CREDIT]
- Pressing the S key should make gameobjects appear and disappear.
- Pressing the F key should flip the user 180 degrees so that he is looking behind where he was looking.
- Pressing the M key should make a certain cube either mirror or follow the user's movements.

Temporarily losing rotation and position tracking in this MP should help you understand how much they contribute towards presence in VR.

Part 1 Depth Perception and Relative Size

In your scene, there are three spherical gameobjects: Small, Medium, and Large that are children of a the parent gameobject called StimulusManager. Your task is to write a GenerateStimuli, a script that arranges these three gameobjects so that they appear to be of

equal sizes when viewed from camera position (0,0,0) with rotation tracking and position tracking disabled.

This script should activate when the user presses the S key. On activation, the script should do the following:

1. Make the 3 objects disappear.
2. Randomly assign the 3 stimuli to be in the left, center, or right position (from the perspective of the camera). There should be one stimulus in each position.
3. Determine where to place the stimuli so that they all appear to be of the same size from the camera's perspective (at position (0,0,0)).
4. Wait 3 seconds seconds and make the left stimulus appear.
5. Wait a further 3 seconds and make the center stimulus appear.
6. Wait a further 3 seconds and make the right stimulus appear.

Pressing the S key again should start this process over again.

Note on Point 3

Determining these positions is very non-trivial. We're therefore going to give full credit even if you precompute these positions and bake them into your submission.

This being said, there are some really interesting ways to compute these positions on the fly (not baked / hard coded). There are 10 points of extra credit for this MP if you choose to go with this route for your implementation.

Part 2 VR Mirror

Create a script named CameraFlipper that flips the OVRCameraRig 180 degrees when you press the F key, like you were turning around to look behind you. Now you should be facing a transparent window with a floating cube face on the other side. Think of this cube as your head. Write a script VRMirror to modify the position and rotation of this cube to match or mirror your own.

Pressing the M key should make the cube match your movements. This includes positional and rotational movements . Furthermore, note that this also means that the cube should be looking in the same direction the camera is (and as result, the user should see the back of the cube's head). For example, if you bring your face closer to the screen, the cube moves further away from the screen.

Pressing the M key again should make the cube mirror your movements (positional and rotational). This means that the cube is facing the camera (and as a result, the user should

see the cube's face). In this case, if you bring your face close to the screen, the cube moves closer to the screen as well. Imagine looking into a mirror to get an intuition of this.

Rubric

Criteria Name	**Points**	**Description for full credit**
Camera Manager Script 1	10%	Pressing tab resets the camera position to (0,0,0).
Camera Manager Script 2	10%	Pressing the R key toggles rotation tracking on and off.
Camera Manager Script 2	10% Extra	Pressing the P key toggles position tracking on and off.
Stimulus Manager Script 1	10%	Objects appear in the correct order, following the correct timing.
Stimulus Manager Script 2	10%	The script can be run repeatedly.
Perception	20%	Objects appear to be of equal size once the script has been run
Mirror Script 1	10%	Cube correctly matches your movements.
Mirror Script 2	10%	Cube correctly mirrors your movements.
Mirror Script 3	10%	Pressing the M key toggles the cube between mirroring and matching your movements.
Cool Math	10% Extra	Positions for the gameobjects being of the same size is computed on execution (not precomputed).

Submitting

Follow the method described in MP1 for submitting.

Additionally, if you're attempting the Extra Credit task, make sure you explicitly mention that in your README.txt.