CS 415: POINT Project Features

Fall 2022

POINT (Physics Outreach at Illinois through New Technologies) is developing an open-source, interactive VR experience targeted at teaching general relativity physics in an accessible and intuitive way. Production has already begun and some of the most basic features of gaming in VR have already been implemented. This includes:

- Hand models - a visual representation of hands at the location of the controller. These can interact with objects through the trigger button on the VR controller (labeled *Primary Index Trigger* on Fig. 1).
- Skybox - the background for any VR game, this is what you see when you look towards the horizon. In our case, we are using a skybox that looks like space so the user is surrounded by stars.
- Teleportation - the player can teleport by holding down the side trigger (labeled *Primary Hand Trigger* on Fig. 1), pointing to the area they want to jump to and then releasing.
- Basic grid deformation script - a script that when added as a component to a 3D model in Unity, will deform the vertex of the model with an equation simulating space curvature by another Unity object that has a RigidBody component (i.e. has a mass attribute).

You can download the base game from the project github: [https://github.com/POINT-VR/POINT-VR-Chapter-1]. You also need to join the project Discord for communication purposes: [https://discord.gg/HEGYsfYV8j]

Provided below is a list of project features that you can contribute to. Please select one of the potential features and let the POINT team know on the Discord in the `#CS-415` channel which feature you are going to work on.

We have office hours at which you will be able to discuss the project with us, test, and develop your current simulation. Our office hours are held in Loomis 257 on Tuesdays from 3-4:30 pm. We have some Meta Oculus Quest 2s and Quest Link cables to help you with testing.

1 Mesh Grid Deformation

Main task: Make a spatial grid of some kind that reacts to the radii and mass of an object. **Each team/person creates one type of grid.** We need to be able to:

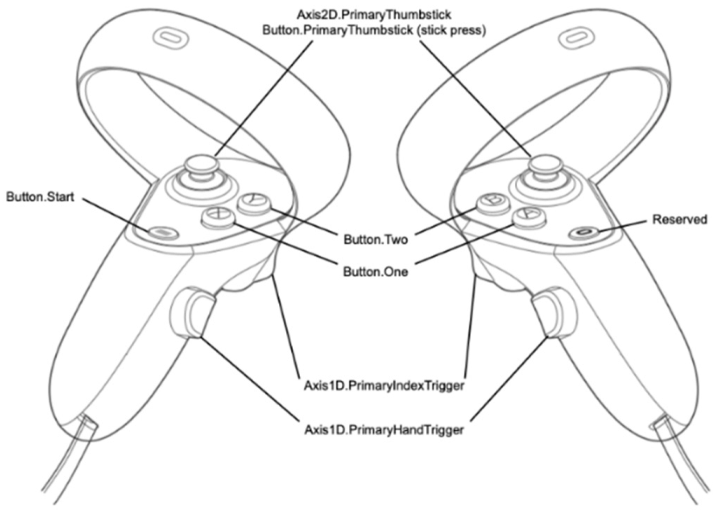

Axis2D.PrimaryThumbstick
Button.PrimaryThumbstick (stick press)
Button.Start
Button.Two
Button.One
Reserved
Axis1D.PrimaryIndexTrigger
Axis1D.PrimaryHandTrigger

Figure 1: A diagram showing the buttons available on Oculus Quest controllers.

- Turn the grid on and off, so it appears and disappears.
- Make the grid centered around the user.
- See the grid fading away in the distance.

  In reality, the grid extends infinitely in every direction. We need to find some way to make it look like this is true without crashing the program or overwhelming the user. Probably the best way to do that is to have the grid fade as it gets farther from the player.
- Deform the grid in the presence of an object.

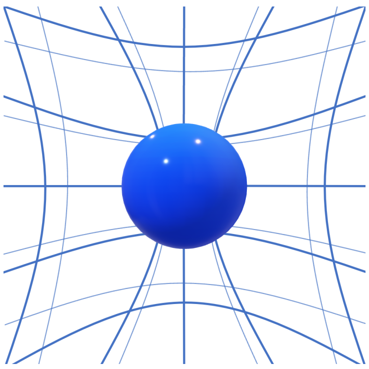
  The grid should be able to interact with the objects in the game. It must curve/deform according to some simple equations. Discuss with POINT team which equations to use. Check out the basic grid deformation script provided for ideas if you get stuck.

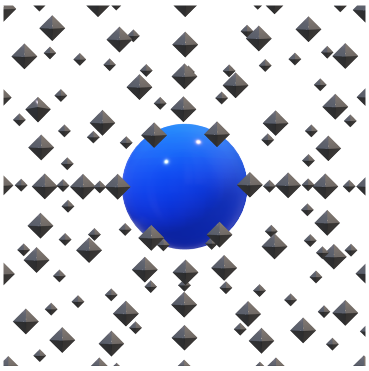
Some potential ideas for type of grid [Each group would only try to implement ONE type of grid]:

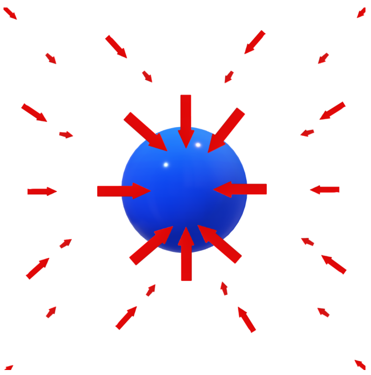
- Type 1: grid of cylinders like in Fig. 2a
- Type 2: show glyphs at points in space (points rather than lines) like in Fig. 2b
- Type 3: grid of arrows, whose direction can change like in Fig. 2c
- If you have other ideas for a representation, please suggest them!

2 Demonstrate three-dimensional space

Main task: Create three spatial axes that are centered on the user, that can be turned on and off. A tutorial-esque scene which will introduce the player to our description of the space and time dimensions. This includes:


(a) Type 1.


(b) Type 2.


(c) Type 3.

- Each of the three space axes appearing one by one in conjunction with each other, with some narration timed to the animation in the scene.
- The final 3D axes should be able to toggle on and off. In scripted events in the future, it should be possible to hide the 3D axes from view.
- After introducing the 3D spatial axes, the player should have a task to interact with these axes to facilitate understanding of using these to describe locations in 3D space.
  - One possible task would be to ask the player to drag an item to a specified location in 3D space, using the 3D axes as a guide.
  - Alternatively, a more guided approach would be to ask the player to move an object some number along each spatial axes along the grid lines. Some specified amount along the x-axis grid lines, some specified amount along the y-axis grid lines, and some specified amount along the z-axis grid lines.
  - To make these tasks easier to perform, it might be useful to demonstrate 3D space as a set of discretized grid lines, where objects snap to parts of the grid lines. Meaning if the player gets the object close enough to a whole number mark on the grid, it snaps into place there.

3 Create different mass and radii objects

Main task: Make one set of objects (spheres) that have different masses, same radii. Make a different set of objects (spheres) that have different radii, same masses. The player will be tasked with sorting these objects and making predictions about how they will deform the spacetime grid.

- The task needs to be user friendly. The instructions must be clear and the player should get quick and accurate feedback on if what they are doing is valid. The controls to use the feature should not be "picky".
- Need to present the objects to the player before the player is tasked to sort the objects. Some options to consider for when they first appear:

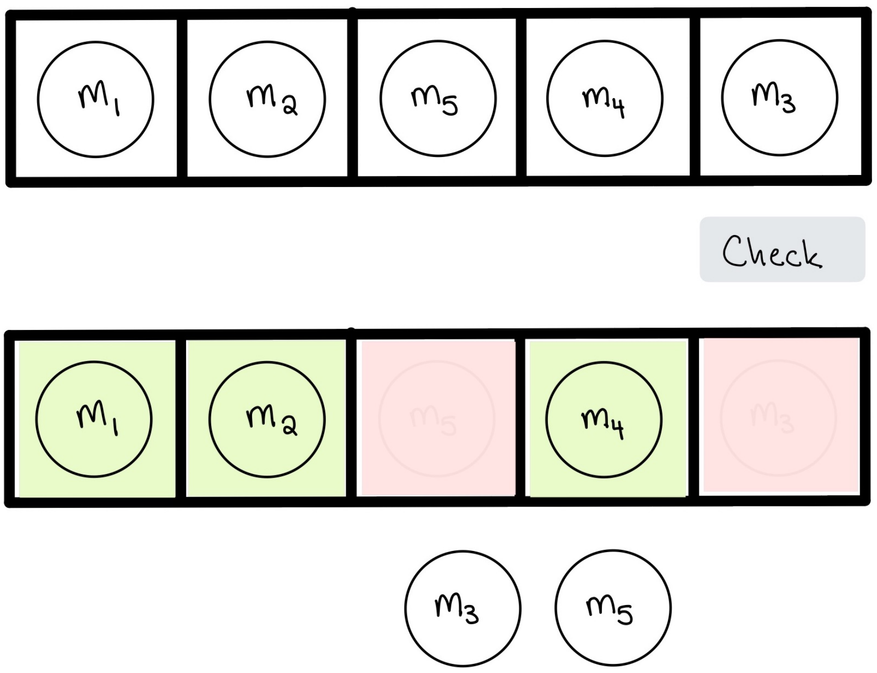


Figure 3: A diagram demonstrating the mass sorting task. In the top diagram the player can place the masses into the boxes in order and then check their submission with the button. In the bottom diagram, the program gives feedback.

  - Should they appear on the ground in a line?
  - Should the user pull them out of a box?
  - Should they be floating in mid air?
- Need an interface object that the player then sorts the objects into.
  - We recommend a box with different sections in it so the player can organize from least to most massive in a clear way as in Fig. 3. If you have other ideas, please suggest them.
  - Give a button for the player to use when they are ready to check their submission. Any box that is correct turns green and those masses stay in place. Wrong boxes turn red and masses in those boxes get bumped out.
- Make two objects with same mass, different radii.
  - Ask user to make prediction: which will curve the grid (the mesh grid in 4.) more?
  - Allow user to check submission. Can reuse that "check" button from before.