AG231

Lab3 – Rev 5

**OVERVIEW**

This lab is designed introduce best practices for movement and avatar facing in VR

**IN-CLASS ASSIGNMENT**

* **You must use the PCs in Little 231 for this assignment**
* **This is an in-class assignment that is due by the end of the lab.**
* Setup the Odyssey with two controllers via the HMD Portal
* This lab continues with Unity project from Lab 2.
* Import the Lab03 Package.
* Use the scene provided in the Package.
  + Be sure to enable this scene in your Unity Build settings.
* Open the example scene that demonstrates an arc drawing routine.
* Take a look at the example script provided to you
* You are allowed to use these scripts in this lab
* You may write your own version for extra credit *(see below)*
* Continue your work with the scene provided in the Lab3 assets.
* Exit the application via the controller
* Implement so you can exit from both a standalone project and the Unity Editor.
* Render an arc instead of a single line for the ray cay under the following conditions and parameters
* When the controller trigger is pulled *(is down)*
* The controller cannot be holding a grabbable object
* Use the start and end points of the raycast for the arc
* The controller will not be able to grab or interact while the trigger is pulled
* Review the following video on Bezier curves.
* Bezier Curves Explained - 3:03 minutes
* <https://www.youtube.com/watch?v=pnYccz1Ha34>
* You may have a quiz on this material There may be a quiz on this before the end of class.

**LAB REQUIREMENTS**

**Movement via the controller**

* The vertical axis from the controller’s joystick will be used to move the player forward or backwards.
* *movementSpeed* is public float member of the class
* *movementDeadzone* is a public float member of the class
  + Set this default value of this member to 0.15f
  + This variable defines the zone where no movement will take place
* Write a method to control movement
  + Use the following function signature
    - *public void PerformMove(float value)*
  + The vertical axis of the controller will be used to indicate direction and how fast to move.
    - The axis value will be passed to the Move method.
  + Move only when the absolute value of the value parameter passed is greater or equal to the deadzone percentage.
* The direction of movement is relative to the direction the user is looking
  + This will be restricted to the global XZ plane.
  + Means to calculate this forward vector
    - Start with the Camera’s forward vector
    - Set the Y Component to 0
    - Normalize

**Turning via the controller**

* The horizontal axis from the controller’s joystick will be used to turn the player
* *turningAngle* is a public float member of the class
  + Set this default value of this member to 60.0f
  + When a turn is performed, this is the number of degrees to turn
* *turnActzone* is a public float member of the class
  + Set this default value of this member to 0.75f
  + This variable defines the zone in which a turn is activated
* *turnIsAvailable* is a private boolean member of the class
  + default value is true
  + Used internally to track and control when a turn can be performed.
* Write a method to control turning
  + Use the following function signature
    - *public void PerformTurn(float value)*
  + The horizontal axis of the controller will be passed to the Move method.
    - This will indicate the direction to turn
    - When a turn
* Turning will take place once when the absolute value of the horizontal axis is greater than or equal to *turnActzone*
  + The next turn will not take place until the value of the horizontal axis moves out of the the *turnActzone* first
  + Use *turnIsAvailable* boolean to control this behavior
  + This is the same functionality as using getButtonDown
* When a Turn takes place, the player prefab (with the camera and contollers) will turn a number of degrees as assigned to the *turningAngle* property and in the direction indicated by the Horizontal Axis.
  + You can get the direction by determining the sign (+1 or -1) of the horizontal axis

**Teleport mode**

* Teleport Mode extends upon the work performed in Lab 2.
* Locations that can be teleported to will have a *TeleportableLocation* script
  + *TeleportableLocation* inherits from *TargetableObject*
* A controller cannot enter teleport mode when it is holding a grabbable object
* When in Teleport mode, the controller will render an arc if it has a resulting hit.
  + A arc sample has been provided to you.
  + The start point will be from the controller
  + The end point will be the location of the raycast
  + This was a requirement of the in-lab assignment.
  + Use Colors when you render the arc to indicate different states
    - Raycast hit is a valid target, but not a *TeleportableLocation*
    - Raycast hit is a valid target that is a *TeleportableLocation*
    - No valid target was required in the previous lab.
* When the trigger, exit teleport mode
  + If there is valid target that is a *TeleportableLocation,* move to that location
  + Otherwise, do nothing.

**Extra Credit – A better teleportation mechanic**There are two extra credit options available to you. Performing the second option requires you to implement the first one.

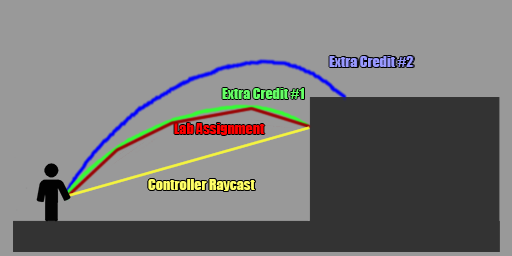
**Option #1:**

Replace the supplied arc draw with one that is built using a quadratic Bezier curve, or a parabolic curve. See the technical document for this lab for more information.

**Option #2:**

This extends extra credit option #1. This builds an arc that is capable to land on top of objects, sometimes in cases where you might not be able to directly see. See the technical document for this lab for more information.

**Teleportation Curve Option Summary**

****

**SUBMISSION**

Submit to Canvas a zip file with the following

* Remove standalone (installable) build from your prior labs
  + The size of your project will quickly get unwieldly if you don’t remove this folder.
* Structure your submission folder as follows  
   ***Your turn in folder  
   |--- Standalone, installable Lab03 build (Only latest version)  
   |--- Unity project ready to build and run from within Unity***
* Remove files that can be rebuilt from the Unity project to reduce the size of the project folder
* Name you zipfile like this <YourName>\_AG231\_Lab3