# CSE 190 Discussion 4

PA2: Levels of Immersion

# Agenda

- Project 2
  - Head Tracking Modes
  - Interocular Distance Adjustment
  - Tracking Lag
  - Rendering Lag
  - Tips for the Extra Credit

# Project 2

- Project 2 Due: May 3rd 2pm (This Friday!)
  - If you have scheduling conflicts, let us know
- Given features in Project 2 Starter Code
  - 2 textured cubes
  - Monoscopic skybox
- Features you need to implement:
  - Render a skybox in stereo
  - Change the rendering settings
  - More specifications in the assignment page.

# Viewing Modes

- If you are stuck can look here for last years starter code.
  - There are some comments on areas of code which will give you hints on how to modify the code to change the rendering settings
- These comments are only HINTS and it is up to you to figure out what to do with the information.
- To render mono, you can just render both eyes from the left side.

# Head Tracking Modes

- When you change tracking modes, you will need to save the previous eye poses. Then when you render, you will reuse either the rotation or position part of the data
- Remember:
  - Top left 3x3 part of the matrix = Rotational Data
  - Rightmost column = Positional Data

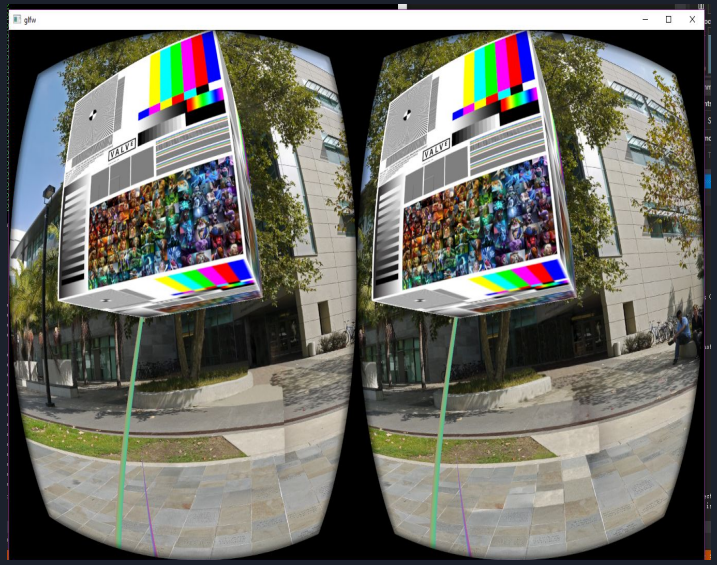
# Interocular Distance Adjustment

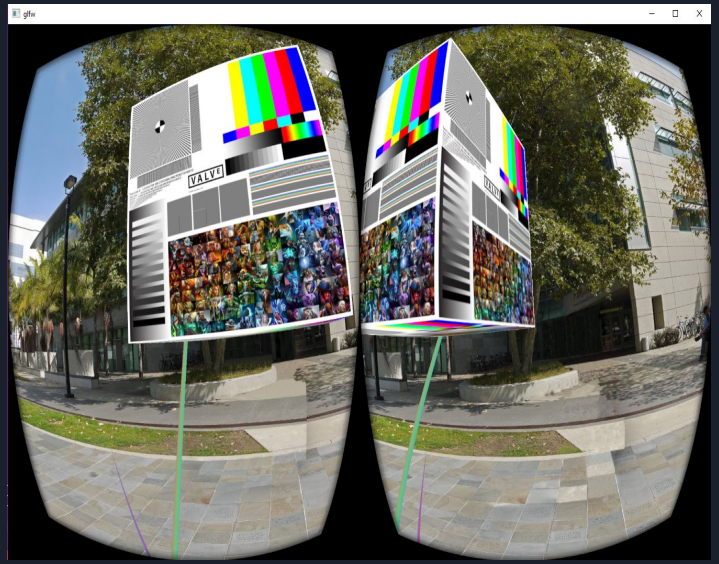
- You can see an example of setting the HmdToEyePose (or HmdToEyeOffset depending on the SDK version) in the RiftApp constructor.

```
//Setup initial IOD
IOD = std::abs(_viewScaleDesc.HmdToEyeOffset[0].x - _viewScaleDesc.HmdToEyeOffset[1].x);
```

- You will need to modify it to change the IOD
  - When assigning back, you might need to divide it by 2

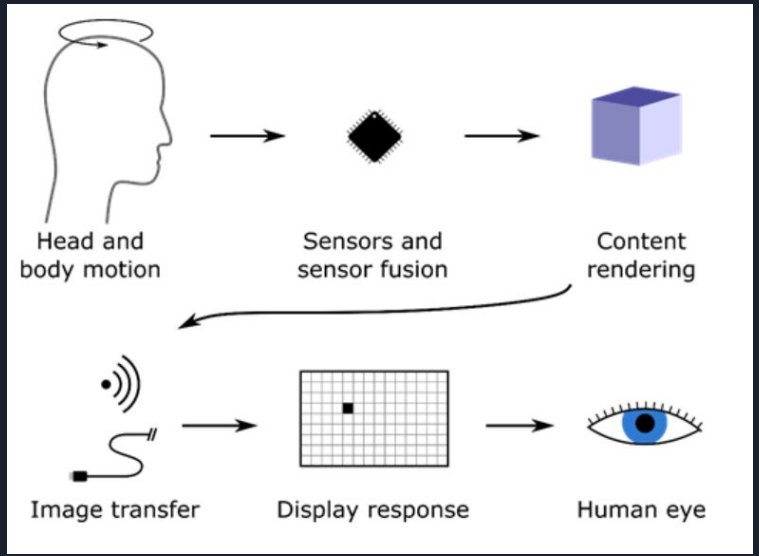
# Interocular Distance Adjustment



# Tracking Lag

- Simulate sensors taking a long time to process tracking (for both HMD and controllers)
- Store Camera matrices in a ring buffer and render scenes with the same Camera Matrix for multiple frames
- When implemented should see sphere for `handController` lagging
  - Like sphere is your hands shadow



# Rendering Lag

- Simulate what it looks like if it takes longer than 1/90th of a second to render a frame
- Need to either:
  - Render the exact same image for multiple frames re-using the camera matrix
  - Wait multiple frames before rendering as usual
- So tracking information will go unused for several frames

# Tracking & Rendering Lag Time Scheme

# Some Technical/Testing Tips

## Technical Tips

- ovrPosef has a Position(Vector3) and Rotation(Quaternion) you might want to look into
- There are various toGlm function to help translate the various ovr data types to glm data types

## Testing Tips

- When testing locking the position/rotation, try either only rotating or moving your head to see if it works.
- To test stereo/mono/inverted, try looking directly at the edge of a cube

# Extra Credit

# Extra Credit - Stereo Image Viewer

- It should just be two “regular, non-panoramic” pictures
  - Set camera FOV to as close to 90 degrees as possible
  - Uses the Oculus HMD like a ViewMaster Image viewer
- Similar to creating a 3D skybox, you will need to render two different textures to your cube to create the stereoscopic effect

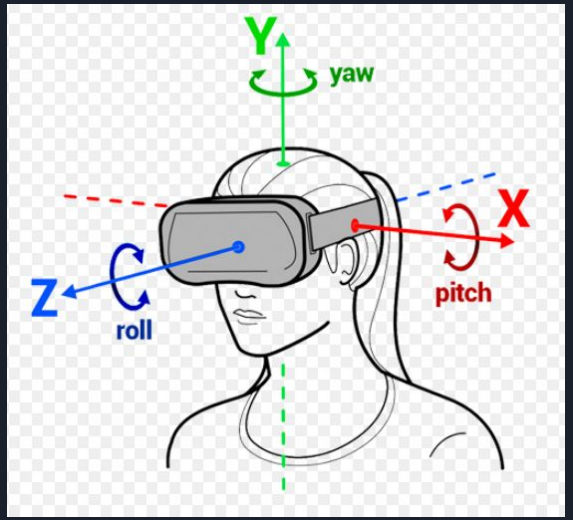
# Extra Credit - Custom Sky Box

- Create your own monoscopic skybox.
- Convert your 360 degree panorama picture into a cubemap:
  - https://jaxry.github.io/panorama-to-cubemap/

- Make it an alternative option when “X” button is pressed
  - It will just be a skybox in mono
  - The device can be borrowed at the Media Lab
  - Can also use the Google StreetView App

# Extra Credit - Super Rotation

- Doubles the movement of your yaw rotation (left and right)
  - ie. rotating  your head to the right 45 degrees, causes your view to look 90 degrees to the right
- No magnification on roll/pitch
  - When looking up, you are still looking above.
  - If the head is only tilting, it should be no effect of super rotation as well.



# Extra Credit - Smooth Controller Tracking

- Averaging the tracking data and beforing rendering the controller using a moving average
- Moving average is basically finding the average between n item in a list
- The larger the window size...
  - the smoother the controller movement will be
  - BUT will also introduce some lag
- Note:
  - Make sure you are applying the smoothing to the position ONLY

# Extra Credit - Smooth Controller Tracking

- Moving Average example:
  - n = 3, meaning we will average over 3 frames
  - Average over those 3 values
  - Use this average value as the position for the controller


Average = (4+2+7)/3

# Extra Credit - Smooth Controller Tracking

- Moving Average example (cont.)
  - Add the additional tracking data
  - Average over those 3 values
  - Use this average value as the position for the controller


Average = (2+7+1)/3

# QUESTIONS?