Coding Test

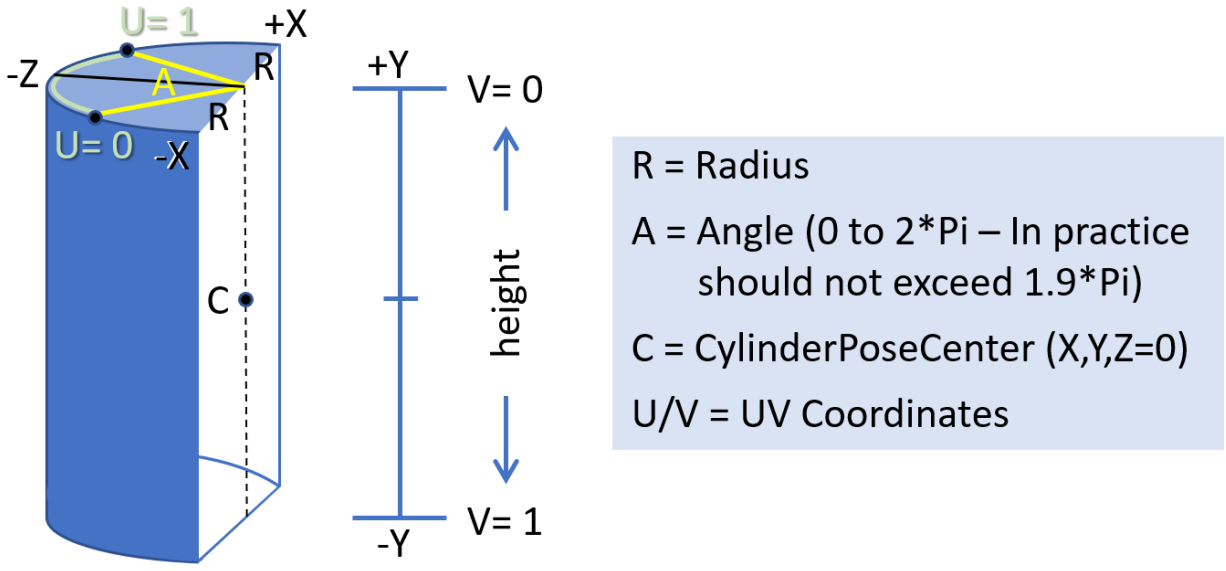
Cylindrical Eye-buffer mapping

Problem: In current VR systems, L-R eye-buffers are mapped on a rectangular surface and then distorted via barrel distortion to account for the pincushion distortion of the lens. In future systems, there is a proposal to enable non-rectilinear eye buffers by using cylindrical, spherical or similar higher order surfaces.

For this assignment, we'd want you to build a system which takes as input two high-resolution rectilinear eye buffers. You must then generate the final display buffers by applying the correct cylindrical transformation followed by a simplified barrel lens distortion correction transform using shaders.

The goal of this assignment is to simulate a simple VR pipeline. For mapping onto a cylinder, you'll have to mathematically generate a cylinder in the eye buffer space (using stored vertices or on the fly) and map the supplied eye buffer textures on them. Once this is done, we need to apply distortion correction using the barrel distortion equation listed below. You may choose any platform of your choice – Windows/Linux/Android and any graphics API – GL, Direct3D or any other. You may do it within a game engine too if you are comfortable that way. You may use any helper libraries (matrix multiplication, texture loader, glut etc.) as you please.

Please note the view must be from the **inside of the cylinder** looking at the cylinder with the viewpoint being common on both eyes – the viewing position should be along the axis of the cylinder at the midpoint in height. Adjust the height to exactly fill the entire screen. Please see the attached image.



Input:

i) Eye buffer sample images: The attached .jpg images are in 1440x1440 resolution. You may convert them to your format of choice to ingest them into your application.

ii) Distance between eyes (IPD): 64mm. Assume 1 world unit is 1 meter in the real world and convert accordingly.

iii) Frustum parameters - Assume the following:
   a. Near = 0.1, Far = 1000.0
   b. Aspect Ratio = 1.0
   c. Field of view = 80 degrees

iv) **Barrel distortion equation:** Assume the following equation:

Xd = Xo * (K0 + K1*r + K2*r^2 + K3*r^3 + K4*r^4 + K5*r^5 + K6*r^6)
Xd = Distorted point in normalized coordinates
Xo = Original input point
r = distance from the center of the frame.

Acceptable values for K0, K1 etc. are listed below
K0 = 1.0, K1 = 0.0, K2 = 0.22, K3 = 0.0, K4 = 0.25, K5 = 0.0, K6 = 0.0

For more information on this, look at: https://stanford.edu/class/ee267/lectures/lecture7.pdf
You may apply this transform in a fragment shader. Performance is not a huge concern.

Output:

i) Eye buffer after applying cylindrical projection.
ii) Distorted display buffer after applying barrel distortion
iii) A few lines on how your code works. What all was achieved and what approach you took.
iv) Your program could either run in real-time in which case, please provide a key binding to switch between a) Input buffers b) Eye buffer with cylindrical projection c) Final display buffer or your program could run on the command-line and dump these images out to disk based on a parameter supplied at execution time.

Please share a git repository with all the code and executables uploaded. Please also do supply build instructions if required.

If you have any queries, don't hesitate to get back to me.

Reference:

https://developer.oculus.com/documentation/pcsdk/latest/concepts/dg-render/