Connor Tsui

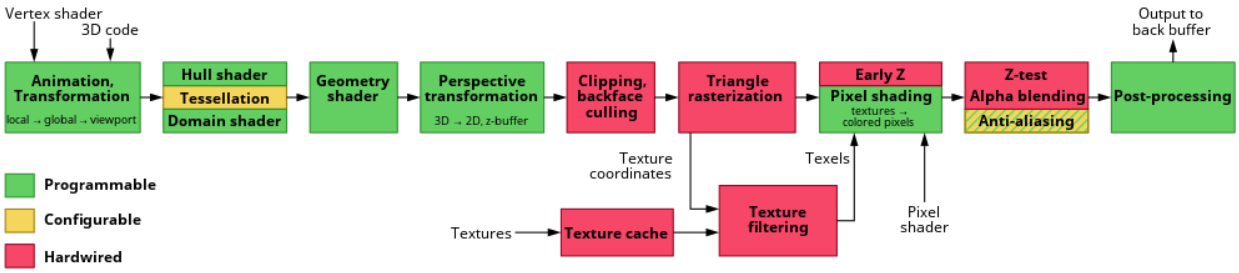
Project Proposal for MVP: (Since I'm done with MVP this is just an explanation)
3D Game Engine from scratch

Description:
3D Game Engine using cmu-112-graphics because I like to suffer. Basically all linear algebra, matrices, and lots of math.

Competitive Analysis:
People have definitely made full 3D graphical engines from scratch before, however not many people are stupid enough to do it in Python, and I have seen none do it in tkinter, and of course who would make themselves do it in a deprecated version of the worse graphics module?

Algorithmic Plan: The most difficult part of this project was the 3d transforms, 3d projection, and by far the hardest thing was frustum culling. Almost all of these things I completed, though they are not the most efficient given the limitations of tkinter and vanilla Python.

This diagram of the graphics pipeline best represents most of what I did with this engine,



This engine implements Animation/Transformation, Perspective Transform, Clipping/Backface Culling, Rasterization (through tkinter), and pseudo z-testing, which are almost all the things that are possible within vanilla Python without recreating an entire 2D graphics engine from scratch.

Version Control: Using Github with this project since October 8, 2021
https://github.com/cjtsui/Carnegie-Mellon-Fall-2021/tree/main/15-112/Term%20Project%20MVP

Module List:

numpy as np

Citation: 3D Engine From Scratch (ChiliTomatoNoodle)
www.youtube.com/watch?v=uehGqieEbus&list=PLqCJpWy5Fohe8ucwhksiv9hTF5sfid8lA
This series was how I learned how to do all of this, but because everything was written in C I was only able to use the concepts brought up in his videos.

POST-MVP DESIGN PROPOSAL

Name: Porta112

Description: Recreation of the game Portal by Valve built using PyOpenGL. Might not be a full replication of Portal but close enough to be recognizable. Controls using PyGame, and math done by numpy and pyrr.

Competitive Analysis: With regards to Portal, people have recreated it many times, but mostly on the back of Unity. A few people have recreated Portal from scratch by building their own game engine, but these people are very few and certainly none of them have made it in Python (because why would anyone do that to themselves).

Citation: OpenGL with Python (GetIntoGameDev)
https://www.youtube.com/playlist?list=PLn3eTxaOtL2PDnEVNwOgZFm5xYPr4dUoR
Repo: https://github.com/amengede/getIntoGameDev
Without these tutorials I would not be able to do anything. As of 11/16, a large majority of my code is almost exactly the same as his with edits on the style and naming, but I expect the majority of the code to be mine within the next week.

Structural Plan: I have no idea yet since I don't think I've realized the monumental task I've put myself up to, so until I have portals actually working I will not have a set structure.

Algorithmic Plan: Obviously the most difficult part of Portal are the portals, but another dimension to the portals (haha get it) is the physics behind it. In Portal, momentum is conserved when you go through a portal, which is a large part of the game. If I am unable to figure out the momentum part within 2 weeks then so be it, but I will do my best to complete both things.

Timeline Plan: haha typing go brrrr

Version Control Plan: Using Github with this project since October 8, 2021
https://github.com/cjtsui/Carnegie-Mellon-Fall-2021/tree/main/15-112/Term%20Project%20PyOpenGL

Module List:
PyOpenGL, PyGame, numpy, pyrr, pathlib

Storyboard: https://www.youtube.com/watch?v=0qcED35LL8I

TP2 Update:
For portals to work how most people expect portals to work, I need to hire either a physicist or someone really good with linear algebra because I'll need to either use quaternions or I need to use one of these nonsense methods:
https://en.wikipedia.org/wiki/Rotation_matrix#Conversion_from_rotation_matrix_to_axis%E2%80%93angle
Or
http://www.euclideanspace.com/maths/geometry/rotations/conversions/angleToEuler/index.htm

So basically I give up on that and I'm just going to hardcode 6 walls and maintain momentum through only those 6 walls. Yes that means I have to hardcode 15 different possibilities (6 choose 2) and no I'm not happy.

The texture overlay (portal effect) is my next and probably final goal. If I can figure this out I think I am done.

Timeline Plan:
Figure out the portal effect and if I can do that I'm done

Storyboard: trust me
https://www.youtube.com/watch?v=AZMSAzZ76EU

TP3 Update:
Literally nothing changed except that I've realized portals require gimbal locks and are way harder than I thought to even hardcode.