Stephen Frueh

CS 499

E-Portfolio

Software Design / Engineering Entry

For my software design and engineering entry into my portfolio, I chose to use my final project for CS 330. For this assignment, I created an application that draws 3D graphics and lets the user navigate around their 3D environment that the objects are being drawn in. Textures were added and the drawing environment in the application was set up and maintained using the OpenGL graphics library. Using the GLSL shading language, I wrote individual fragment and vertex shaders to run on the graphics card when processing the vertices used to draw the scene. The user can move the camera and orbit it around the object being drawn using a virtual camera. The user can also set the object they are drawing to move automatically over time, such as adding a small rotation so the object is constantly rotating.

I’ve decided to include this piece in my portfolio for a number of reasons. Mainly, I chose this piece because it encapsulates many aspects of developing software. It’s a complete application built from scratch, incorporating graphics programming and handling user input, which are important aspects to nearly any piece of software. It also demonstrates my ability to be able to learn and work with third party libraries, which are frequently required. Learning how to use OpenGL requires some skill to understand and efficiently use, especially considering how careful you must be with memory management. Also having to be very careful and aware of data storage and sizes, since sending vertex data to the graphics card for processing can be very difficult. The GLSL shading programming language is also tough to use beyond basic use, as well as being a new programming language to have to interact with altogether. Overall, it’s one of the most complete pieces of software with a variety of skills involved that I feel I can display in regards to design and engineering of a piece of software.

When looking at improving my work before submitting it into my portfolio, there were a few areas I specifically wanted to improve. The first would be documentation, mostly involving comments in some parts of the program. There were a few sections that went far too many lines of code without comments, which I went back and inserted. Some code I had put in for debugging, which involved checking the compile and link status of the individual shader programs was removed as well, though if the shaders were ever changed, checking their compile and link status is a good idea to have.

One of the bigger challenges I faced when working on this program initially was how hard it can be to work in OpenGL. When something isn’t working in regards to OpenGL, you don’t get the standard compiler warnings and errors that usually inform you of errors unless you actually mess up the syntax. It’s also rather difficult tracking which data is going and coming from the graphics cards using the buffer objects that handle storing the vertex data, so a normal debugger doesn’t do much good. Frequently, errors in OpenGL must be found by double and triple checking your code; the data you’re using to fill the buffers, how you tell the buffers the format of the data, and even whether things are done in the correct order.

Figuring it out and getting your OpenGL program to successfully run is incredibly rewarding, and something I’m proud to be able to do. This is why I chose this assignment to be included into my portfolio.