Brett Stelly

October 19, 2018

**Computer Graphics**

**Cameras and Projections**

**CameraApplication class that derives from the Application class**

**Pre-conditions:** In order to create a class for the Camera application that derives from an application class, you will first need to make an application class with functions for startup, shutdown, update, draw, clearScreen, and run. The startup, shutdown, update, and draw functions should be made to be pure virtual functions, so that they can be overridden.

Include the application header in the CameraApplication header. The CameraApplication class inherits from the Application class and should override functions for all of the pure virtual functions in the application class. The CameraApplication class has access to all non-pure virtual functions, and so, I do not have to overwrite them. Instead, the CameraApplication class will use the code that was already written for those methods. This gives me the ability to create different applications, that derive from the Application class, with different functionalities if desired.

**Camera class used by CameraApplication class**

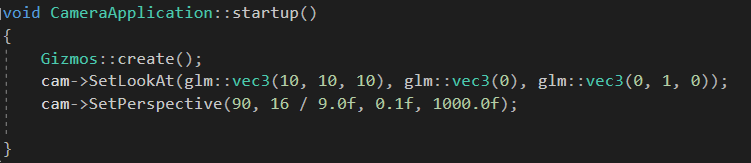
**Pre-conditions:** The CameraApplication class must be created and deriving from the Application class. You will also need to have linked the glfw3 header file and the glm libraries to the project. The glfw3 header file gives access to the OpenGL library and the glm libraries are OpenGL mathematic libraries for doing calculations. These files can be found online.

Now, create the Camera class itself.

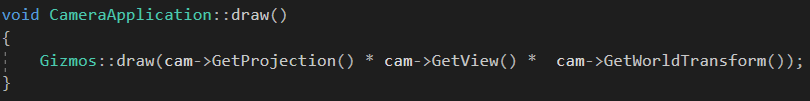
In order for the CameraApplication class to make



zyp function, I call Gizmos create function. Then, I set the camera’s viewTransform member variable by calling setLookAt. Next, I set the camera’s projectionView member variable by calling SetPerspective.



In order to actually see through the camera, I call Gizmos’ draw function and give it the Camera object’s, projectionTransform, viewTransform, and worldTransform.



**Rendering Geometry**

**Direct Lighting**

**Textures**