COSE436 Interactive Visualization (fall 2023)
Instructor: Prof. Won-Ki Jeong
Due date: Oct 8, 2023, 11:59 pm.

Assignment 1: Simple OpenGL Viewer (100 pts)

In this assignment, you will implement a simple OpenGL viewer that supports user input (keyboard and mouse interactions). The final viewer should be able to render basic glut polygon models as shown in Figure 1.

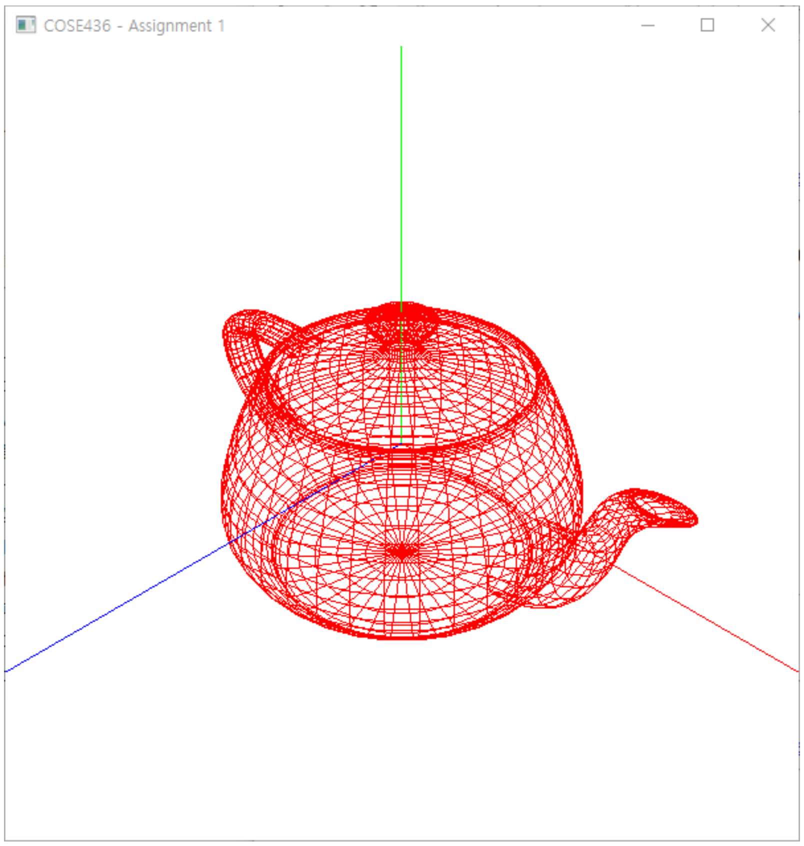

Figure 1. Example of a simple OpenGL viewer

The skeleton glut code is provided. You will need to add more OpenGL code so that you can visualize glut models. Note that you may change `main.cpp` and `vshader.vert` only (submit those two files). Here is the list of required functions you need to implement.

- (10 pts) Your viewer should be able to render several glut-provided 3D models in a wireframe mode, such as **glutWireTeapot()**, as shown in Figure 1. You should implement a keyboard callback "m" to switch between 3D models (e.g., sphere, cube, etc). Your viewer should provide at least three different models. Note that you do not need to implement surface shading in this assignment.

- (10 pts) Reference frame rendering (x,y, and z axis) using red, blue, green colors, respectively.
- (30 pts) Orthogonal and perspective projection using vertex shader. You should use **Ortho()** or **Perspective()** function in the matrix class to create a projection matrix and pass it to the vertex shader. You should implement keyboard callbacks: "o" for orthogonal and "p" for perspective projection to switch between projection modes. Since glut 3D models are placed at the origin, it might be helpful to change the eye location away from the origin. For this, you may use **LookAt()** function to create a viewing matrix.
- (50 pts) Virtual trackball - zooming, panning, and rotation using a mouse. You must use the matric class (mat.h) and pass the model-view matrix to the vertex shader. The trackball should work accurately. Make sure the left-mouse button is rotation, the middle-mouse button is panning (translation), and the right-mouse button is zooming.

The provided skeleton code is tested on a Windows PC and Microsoft Visual Studio. Use CMake to generate a solution file for Visual Studio. CMake is a platform-independent project generation tool (http://www.cmake.org/).

You should submit **main.cpp and vshader.vert only** in a single zip file. The code must be compiled without additional external library other than the provided ones. Note that I will not debug your code, so it is your responsibility to make the code working correctly (make sure to keep the original skeleton files and test your codes with the unmodified skeleton files). Make sure you compile your code in 64bit mode(x64) because the included freeglut library is compiled in 64bit mode.

Good luck and have fun!