CSCI 3260 Principles of Computer Graphics

Assignment One: Creating a 3D Scene

Due Time: 11:59pm, Oct 07 2016 (Friday)

Late penalty: 10% per day.

Fail the course if you copy

I. Introduction

This first programming assignment will introduce you to the OpenGL graphics programming interface and programmable pipeline. In this programming assignment, you will be creating different 3D objects to model interesting shapes. The objective of this assignment is to apply you understanding of the computer graphics theories, OpenGL programming library and give you an introduction to the programmable pipeline.

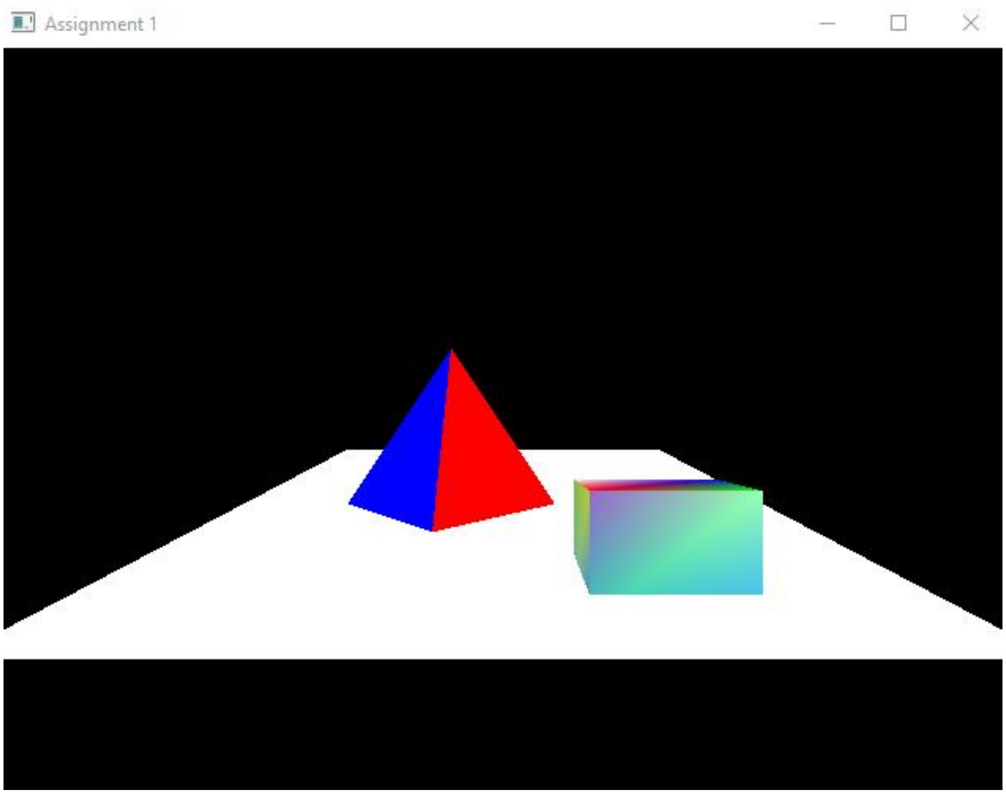

Fig. 1 The scene drawn by the demo program.

Your goal is to design a 3D scene, which consists of a ground, at least two different geometric polyhedrons, such as sphere, tetrahedron, cube, etc., and you should be able to apply arbitrary transformations including translation, rotation, and scaling to them. The user should be able to use the keyboard (and/or the mouse) to translate, rotate, and scale the object. The object color, window size, window title and window background color are all up to you. In order to make your scene more realistic, you should use the perspective projection instead of orthographic projection. You are recommended to draw objects with indexing. Your 3D scene shall not be limited by the demo program.

II. Implementation Details

In this assignment package, we have provided you with two shader programs (i.e., *VertexShader.glsl & FragmentShader.glsl*) and a template program (i.e., *submit.cpp*) which include the necessary functions you are going to use and callback functions in the GLUT interface toolkit. Use this template as the basis for your implementation. You need to design your own function to process the keyboard events, and you should also submit a file like *readme.txt* to specify the keyboard (and/or mouse) events you designed in your program. Otherwise, the mark for related items will be deducted.

All programs should meet reasonable programming standards: header comment, in-line comments, good modularity, clear printout, and efficiency.

Basic Requirements:

1. OpenGL code should be written using programmable pipeline instead of fixed pipeline with OpenGL 3.0+;
2. Draw at least two solid geometric primitives in the 3D scene above a ground;
3. Ensure at least one object is drawn with indexing;
4. Create at least three keyboard and/or mouse events;
5. Design diverse objects transformations, such as rotation, translation and scaling;
6. Use perspective projection to draw the scene and enable depth test to realize occlusion;

Additional self-design requirements:

You are free to add objects, move them, organize them, and whatever you wish to make your scene interesting.

III. Grading Scheme

Your assignment will be graded by the following marking scheme:

Basic (80%)	
Ground	10%
At least two different solid geometric primitives, at least one object is drawn with indexing	20%
At least three keyboard (and/or mouse) events	15%
Object transformation (rotation, translation, scaling)	15%
Depth test	10%
Perspective projection	10%
Advanced (20%)	
Complex and meaningful objects constructed by different primitives	10%
Interesting and creative interactions	10%
Total:	**100%**

Note: no grade will be given if the program is incomplete or fails compilation or using fixed pipeline.

IV. Guidelines to submit programming assignments

1) You are suggested to write your programs on Windows, since there will be enough technical support. If you developed the program in other platforms, make sure your program can be compiled and executed on Windows as the program will only be tested on this platform. The official IDE is Visual Studio C++ 2015.

2) Modify the provided *submit.cpp & VertexShader.glsl & FragmentShader.glsl*, and provide all your code in this file. No other additional .cpp or .h files are allowed. Type your full name and student ID in *submit.cpp*. ***Missing such essential information will lead to mark deduction (up to 10 points).***

3) We only accept OpenGL code written in programmable pipeline. No points will be given if your solution is written in fixed pipeline.

4) Zip the source code file (i.e. *submit.cpp & VertexShader.glsl & FragmentShader.glsl*), the executable file (i.e., *submit.exe*), and the readme file (i.e., *readme.txt*) in a .zip. Name it with your own student id (e.g. *1155012345.zip*). There should be exactly ***five*** files in your submitted package.

5) Submit your assignment via eLearn Blackboard. (https://elearn.cuhk.edu.hk/)

6) Please submit your assignment before 11:59 p.m. of the due date. Late submission will be penalized by 10% deduction per day.

7) In case of multiple submissions, only the latest one will be considered.

8) ***Fail the course if you copy.***