COSC4370 Spring 2020 HW3 - 3D Viewing and Shaders

March 24, 2020
Due: April 3, 11:59 PM, 2020

1 Introduction

In this assignment, we will practice 3D viewing and dive a little more deeply into OpenGL by implementing the Phong shader model.

2 Setup

Since this homework assignment is more complicated than the last, you will need a few libraries installed on your system: GLUT (which you should have already installed for HW2), GLEW, GLFW, and GLM.

On Ubuntu/Debian, first run sudo add-apt-repository ppa:keithw/glfw3 in order to add a repository containing the GLFW3 library.

On Ubuntu and other Linux variants, these libraries can be installed with a one-liner at the terminal: sudo apt-get install libglew-dev libglfw3-dev libglm-dev . (Note: on some Linux variants, the package names might end in devel rather than dev; check your distribution's package database to find the correct package.)

On OS X: GLEW: If you have Homebrew installed, you can run brew install glew, or if you have Macports, you can run sudo port install glew +universal and sudo port install libsdl +universal . GLFW: If you have Homebrew installed, you can run brew install glfw3 . With Macports, you can run sudo port install glfw . GLM: If you have Homebrew installed, you can run brew install glm . With Macports, you can run sudo port install glm .

On Windows: You can download GLEW from https://sourceforge.net/projects/glew/files/glew/1.13.0/ (select the download that ends in win32.zip) and GLFW from http://www.glfw.org/download.html (you will most likley want the 64-bit Windows binaries). The header-only (no compilation necessary) GLM library can be downloaded from https://github.com/g-truc/glm/releases. In your Visual Studio project, you will need to add the appropriate include directories and library directories for each of these libraries. Some help with this can be found at http://www.41post.com/5178/programming/opengl-configuring-glfw-and-glew-in-visual-cplusplus-express.

3 Compiling and Running the Code

For Linux and OS X, we have included a Makefile that will automatically compile the homework, assuming you have the correct libraries installed. Just run make in a terminal. The program that is generated is named hw3.

On Windows, you can use Visual Studio in the usual way to compile and run your program.

Note that the files needed for compilation include main.cpp, Camera.h, and Shader.h. Your vertex and fragment shader files are loaded by OpenGL at runtime; you do not need to compile them with the other

files.

Note that the program takes no command line arguments etc. - you can just compile and run.

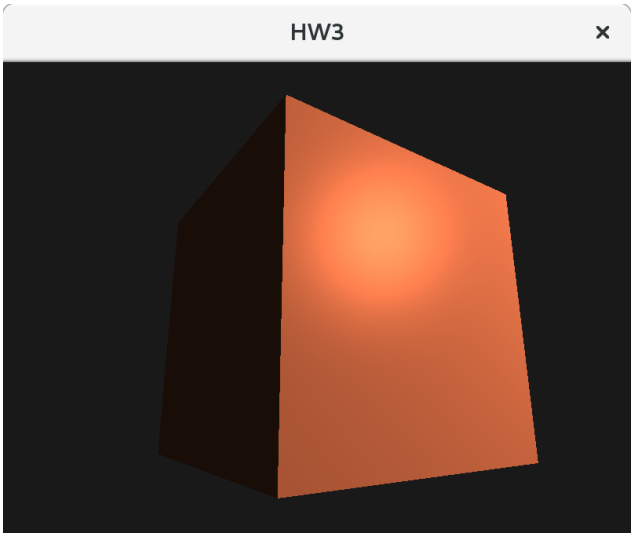
4 Note

We have provided ample starter code. Among the features we've included, you can pan and shift the camera. Moving the mouse will rotate the camera (note: on some machines, the code might be very sensitive to the mouse movement). Using the W+A+S+D keys will shift the camera. You can press the Escape key to quit the program.

5 The Main Assignment

The goal of this assignment is to implement the the 3D viewing and Phong shading model. To view the object from the camera, you will complete the GetViewMatrix() function in Camera.h and the projection matrix in main.cpp.

You will write the vertex and fragment shaders for the Phong model to shade a simple cube, whose geometry is constructed in main.cpp; stubs for the shaders are provided in phong.vs and phong.frag, respectively. If you implement everything correctly, you should be able to reproduce an image like the following:



6 Tips

Initially, you will get a black screen. We recommend that you first attempt to get a solid red (or other color) cube visible. Once you have that, then you can proceed to build up your Phong model, and you can debug as you progress.

7 Deliverables

Submit all deliverables to your Github reporsitory.

- Code for your shaders (phong.vs and phong.frag)

- Camera.h and main.cpp
- A screenshot (preferably .png) of your result