CAP 4730 Computer Graphics
Dr. Corey Toler-Franklin

GLSL Programing
DUE: April 5th, 11:59 pm

Overview | Details | Resources | Getting Help | Submitting

Overview

You may complete this project alone or in groups of two.

Shader programs are essential components of the modern OpenGL pipeline. You will use the OpenGL shader language, GLSL, to implement shader programs that process high level algorithms efficiently using the graphics processing unit (GPU). You will be evaluated on (1) source code completion and correctness 30% (2) reflectance direction calculation 10% (3) cube map 20% (4) texture lookup 20% (5) result image 10% and (6) written report 10%.

Details

Getting Started

Extract the file proj4_glslshaders.zip to your local folder. The source code framework is in the *src* folder. Datasets (meshes, textures and scenes) are found in the *data* folder. Other helpful materials are in the *docs* folder. To build the application on linux, type *make* in the *src/mainsrc* folder. This also generates a visual studio c++ solution file. Use any platform but make sure your code compiles and runs on Ubuntu Virtual Box before you submit (see *docs* for instructions). Included is libst (by Syoyo Fujita) from prior assignments. The program is executed from the main function in src/mainsrc/main.cpp. **Note that librt is not included and a utilizes.cpp and utilities.h file are added to the project (and Makefile) in src/mainsrc/**.

Your goal is to learn shader programing. You will focus most of your time on implementing the shader program in the specified shader files. The code framework provides infrastructure for compiling, loading and running your shaders. The *STShaderProgram* is the main class for managing your shader. Examples of vertex and fragment shaders can be found in *mainsrc/kernels*.

Figure 1: Reflections: Environment Lighting

Environment lighting uses texture-mapping (instead of point lights) to implement complex scene lighting. This is different from other examples that compute reflectance by plugging parameters into a specific lighting equation. To implement a specular (mirror-like) reflection under environment lighting, you will compute a reflection ray at a point, and perform a texture look-up to determine the object's color at the specified point.

Implement the main() function in the vertex and fragment shader files, and complete functions in main.cpp that build the cubemap. Follow the **TO DO: proj4_GLSLshaders** notes and refer to the lecture notes and section 11.6 of the textbook for more details. To implement environment lighting, OpenGL uses a cubemap that stores six images on a cube (the environment). The environment light intensity in a certain direction is determined by sampling a point along a 3-D vector that intersects the cube map. Implement texture look-up in the shader function *getEnvColor*. Simply look-up the color of the environment in the direction of the viewing ray. To implementation the St. Peter's cube-box, initialize the variable *cmapFiles*, initialize the shaders in the function *SetUpEnvironmentMap* and complete the *GenerateCubeMap* and *LoadCubeFace* functions. The interior of the cube map is the background. Remember (for realism) to look-up the background environment intensity for rays that do not hit the model. Given a viewing direction and a normal direction, compute the specular reflection direction using the GLSL built-in reflectance equation. Use the result to sample the cube map. Implement texture lookup in *envmap.vert* and *envmap.frag*. Compute the specular reflection direction in *reflectionmap.frag* and *reflectionmap.vert* (located in *mainsrc/kernels*). Fig. 1 shows an example. Save two views of your reflectance result. Use the rudimentary spin function provided to flip the model about the x and y axes (mouse moves with left button down).

Resources

The openGL Programming Guide, https://www.opengl.org/sdk/ is a good reference. Appendix A also provides information about GLUT and GLEW. Chapter 12 of the textbook, Fundamentals in Computer Graphics, covers meshes. Information about the *.obj* file format is in the *docs* folder.

If you are not in the CISE department, and would like computer lab access, you can register for a CISE account at https://www.cise.ufl.edu/help/account. The source packet proj1_mesh.zip will run on machines in the computer labs on the first floor of the CSE building. Labs are open 24 hours (see https://www.cise.ufl.edu/help/access). **Remember to back-up your work regularly!!!** Use version control to store your work. DO NOT PUBLISIZE SOLUTIONS.

Getting Help

Source Code Please do not re-invent the wheel. Use the source code framework (and comments) and review the notes in the docs folder.

Discussion Group Post questions to Canvas (everyone benefits in this case), or send me an email at ctoler@cise.ufl.edu. I will check both daily.

Office Hours CSE 332 (or lab CSE 319) MWF 11:45am to 12:30pm.

Collaborating Credit outside sources and follow University policies on academic integrity.

Submitting

Upload your proj4_glslshaders.zip file to Canvas. It should include:

- Source code with make file
- One written report that includes:
  - reflection environment map result (two views)
  - a few lines that explain your implementation
  - anything you would like us to know (how to run your code, bugs, difficulties)
- original reflection environment map result from shader