Computer Graphics

Assignment 4 – Shaders and lighting

Familiarize yourself with lighting calculations and the programming sessions on the shader language and using shaders in a program. You can of course use any programming session recording and any shaders and code that have been added to the course website. Understand how these work and add them as needed into your own code or base your project on them.

Lighting calculations in the OpenGL Shading Language

The main purpose of the OpenGL shaders is to decide the position and color of each pixel that is rendered on to the screen. Transformation matrices are built and used to decide the positions but there are several ways to decide the color. The most common tool in 3D pipeline rendering is the lighting model.
You will need to build a lighting model, implement the lighting calculations in the shaders and handle any variables needed to run them correctly.

The lighting model needs to include a material for what is being rendered and more than one light. Apart from that there are many choices that can be made, and no one light model is better than another as long as you can understand and argue why the choices have been made.

A full lighting model would have *diffuse*, *specular* & *ambient* colors for every single light as well as a global ambience color. It would then have *diffuse*, *specular* (plus *shininess* value), *ambient* and *emmission* colors for the material being rendered. It would also have a *position* or *direction* for each light while the position for the material comes from the vertex geometry being sent through the shader each time. ***While this is the most complex and flexible lighting model, it's not necessarily the best for you nor will it necessarily give better marks than any other.***

Another possibility is to have a single color for each light and use it both for specular and diffuse. Then you could skip ambience per light and only use a global ambience. Materials will almost invariably need separate values for specular and diffuse, but it might be possible to use the diffuse color for ambience as well. The shininess is part of the material, and is important.

In many cases you don't have to add the emmission at all, if you're not using it, and in any case, when it's used it's often used alone, and so a boolean switch would be fine to decide whether the material should use lighting calculations or skip them entirely and only use the flat color (whether it's called emmission or just u_color).

Now you can choose whether to do the lighting calculations per vertex or per pixel/fragment. Make sure you understand the difference and what information needs to flow between the vertex shader and the fragment shader in each case, regardless of which you choose.

Then you have to decide how many lights you will use, whether each of them can be directional or have a position, or if one or two of them are fixed directional and the rest have positions. More flexibility here can be cool, but it's also more complex and often not even necessary or helpful. If your shader offers many lights, make sure they can be switched off when not in use, so the calculations aren't unnecessarily expensive.

Directional vs. Position lights
The difference here is whether the coordinates given for the light are used as the s-vector (directional) or if the s-vector is calculated by subtracting the vertex position from the light position

(position light). One way to make lights offer both possibilities is to use homogenous coordinates, where the fourth coordinate is 0 for a directional light (vector) and 1 for a position light (point). Then your shader checks this fourth coordinate to know if it should calculate the s-vector or simply read it straight from the coordinates.

Always make sure that your lighting model isn't too complicated for what you plan on using and showcasing in your program. *Adding flexibility for later use is great, but just make sure it doesn't become so cumbersome that it interferes with what you're currently doing.*

Returning the shader

The shaders themselves will be part of the project that you return in Assigment 3 (3D Maze). In this project *you only return a .pdf report describing the shader* and how it works.

The report

The report needs at least two chapters. In the first one you will describe you lighting model, what's in it and why you have decided to include and/or exclude certain things.

The second chapter will go through the entire code of your shaders and describe what each line/part of line/set of lines does. Explain every variable and where and how it's used.

Make sure you describe the relationship between your vertex and fragment shaders, even if you end up doing most of your work only in one of them, and this description might be a general description of shaders, describe this in your own words. Make sure that you get across the fact that you understand your shaders, both what happens within them and how that affects what happens around them in OpenGL.

Grading

A lighting model with one directional light and two position lights could be sufficient for a program like this. It would have to be fairly flexible in terms of colors and light types and you would showcase its flexibility in your program. If your report then explains how it works and also how its limited and what could be done to build upon it, then that will easily give full marks, given that the implementation isn't flawed or the descriptions vague or incorrect.

You may wish to make your lighting model more complex later for the last programming assignment, so if you do that now, while your working on this assignment, you might very well both get extra points here and get points for that same work again in the last programming assignment. Just sayin'.