Computer Graphics: HW03

Ming-Chang Chiu 100060007

May 15, 2015

1 Objective

In this assignment, we are supposed to modify the main.cpp and shader code to implement three lighting effects, which are directional light, spot light, and point light. Each effect are required to use the method of Gouraud shading, which interpolates the color of contiguous vertices to get the color of each pixel, to render the color of the model.

2 Implementation

My shader file is sample.vert. In general, there are three types of variables in GLSL, which are varying, uniform, and attribute variables.

Varying: Varying variables provide an interface between Vertex and Fragment Shader. Vertex Shaders compute values per vertex and fragment shaders compute values per fragment. If you define a varying variable in a vertex shader, its value will be interpolated (perspective-correct) over the primitive being rendered and you can access the interpolated value in the fragment shader.

Attribute: Vertex attributes are used to communicate from "outside" to the vertex shader. Unlike uniform variables, values are provided per vertex (and not globally for all vertices). There are built-in vertex attributes like the normal or the position, or you can specify your own vertex attribute like a tangent or another custom value. Attributes can't be defined in the fragment shader.

Uniform: Uniform variables are used to communicate with your vertex or fragment shader from "outside."

To communicate between shader and main.cpp, some types of functions which are provided by OpenGL are useful. First, we have to get the location of the uniform and attribute

variables defined in shader by glGetUniformLocation(), and then one can use functions like glUniform(), glAttribVertexPointer() etc,, to send values to desired shader's variables by assigning the counterpart locations.

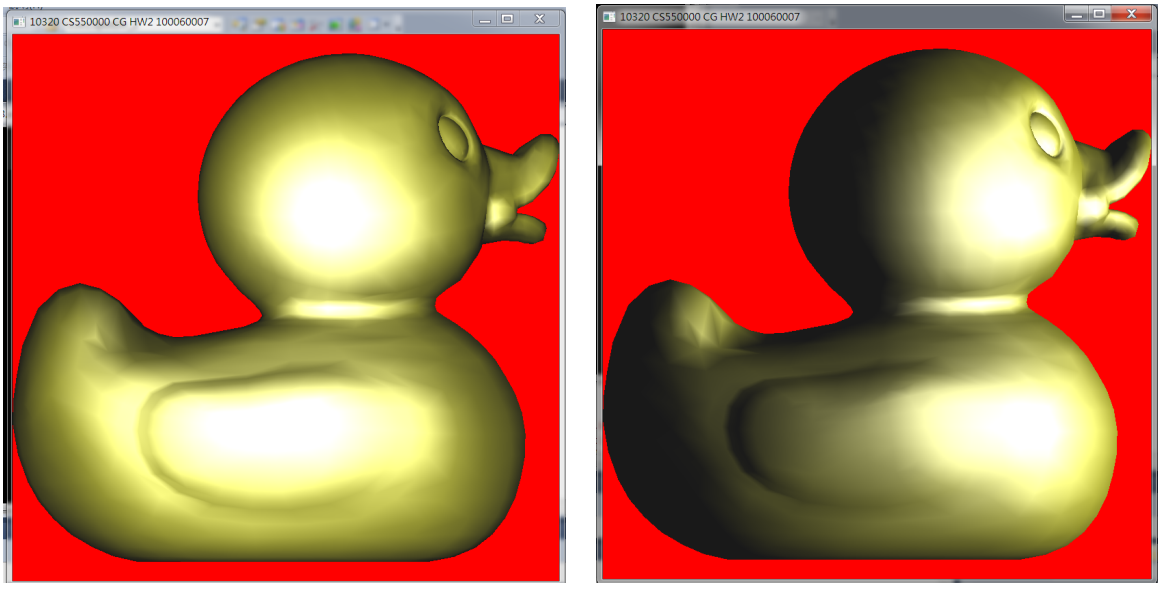
3 Usage

One can simply use mouse to control the position of model: click left button and the drag the model is to translate the position. click right button and then drag is to rotate the model. Scroll the middle button to change the light exponent.

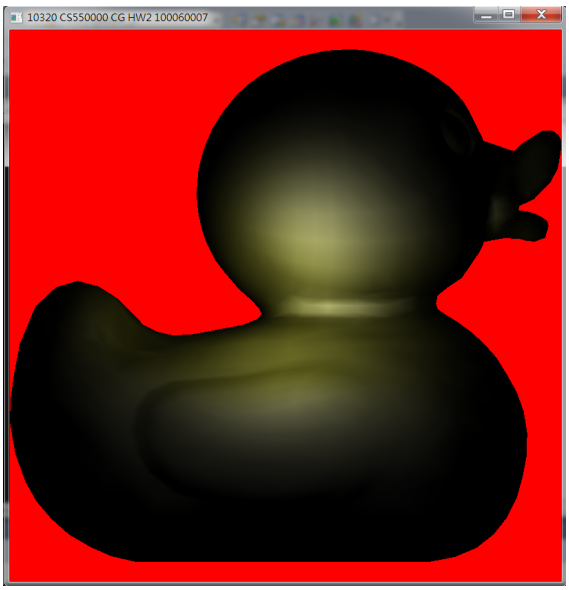
To change the position of light source, simply hit 'a', 'w', 'd' to shift left left by 0.5 for directional light, point light, spotlight respectively and 'A', 'W', 'D' to shift right. To turn on or off the light sources, hit '7' to do directional light, '8' to point light, '9' to spotlight. To increase or decrease the exponent of spot light, hit '4' to decrease, hit '6' to increase.

4 Results

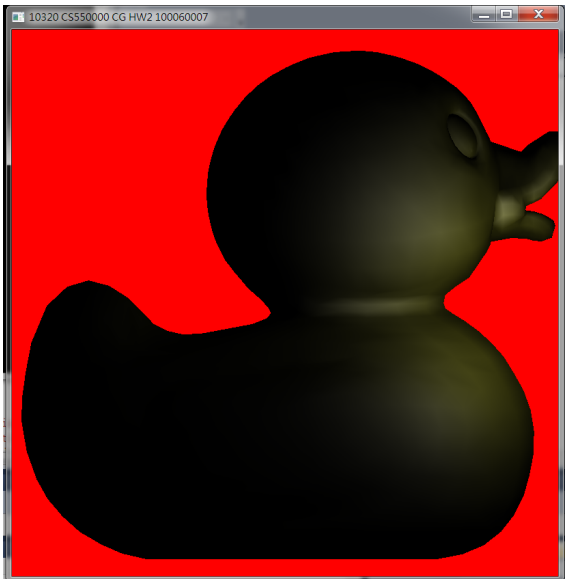
Default light is directional light.



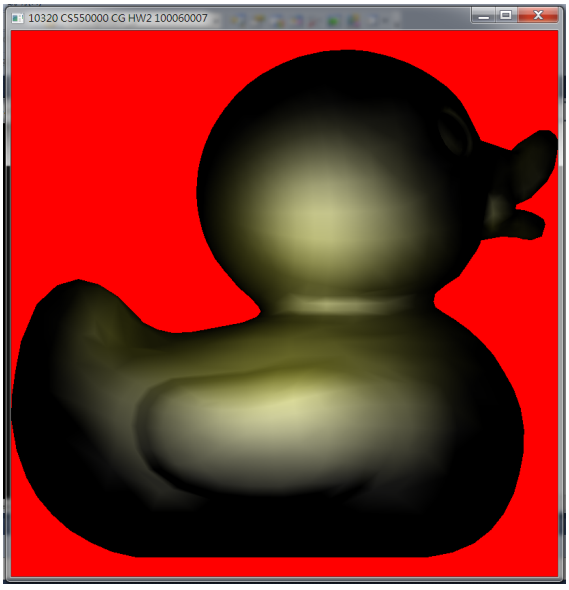
(a) Directional Light (b) Shifted Directional Light



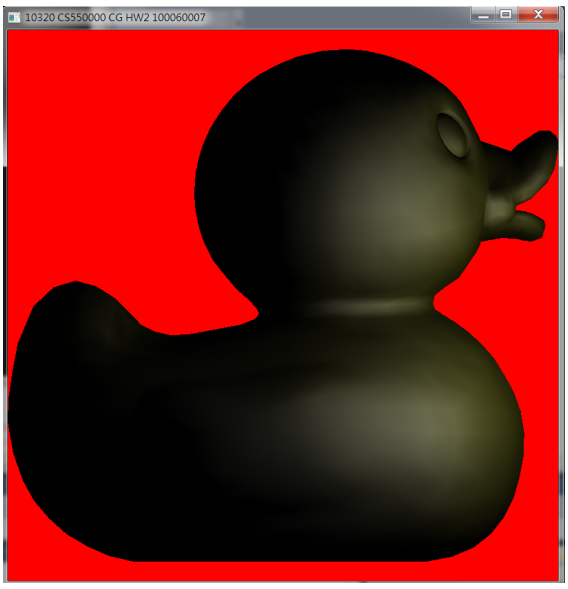
(c) Spotlight



(d) Shifted Spotlight



(e) Point Light



(f) Shifted Point Light

5 Challenges

Using wrong type of function: When sending uniform variables, sometimes I forgot to change the name of the function, like glUniform4fv(), but actually glUniform1i() is the desired one.

Assigning wrong material parameter value: When assigning material parameter, one

would easily assign wrong group parameter to shader. The solution is to create variables or multi-dimensional array to store the values of each group and then send them to shader group-by-group, like I did in renderScene(). Doing so we have to draw the number of triangles in each group for each group, that is, three times of the number of triangles of vertices in that group.