CS380 Introduction to Computer Graphics
Homework Assignment #3

Let there be light and Blinn-Phong reflection model

20160042 Inyong Koo

In this document, I will explain how my implementation satisfies the given specifications. I will not copy the code itself, but rather explain in plain text. If you want to see how the actual implementation looks like, please refer to the code lines and comments of attached files.

1 Loading External 3D Model

I found a .3ds format model of an island and a lighthouse from Free3D. The file (`far.3ds`) was constructed of 34 meshes. I had to figure out their indices in order to set desirable colors of the material. The meshes were in Z-up system, so I transformed them to Y-up coordinates.

I also loaded the bunny object given by `scene.obj` to make my implementation more interesting.

2 Designing Lights Structure

Structure for various light sources are defined in both `Material.cpp` and `FragmentShader.glsl` files. Calculation of intensity by the light source is handled in fragment shaders.

2.1 Spotlight

I added properties regarding *direction* and *spotlight angle* to implement spotlight. If the angle ϕ between the to-light vector $\mathbf{l}$ and the direction vector $\mathbf{d}$ is smaller than the spotlight angle ϕ_s (or $\mathbf{l} \cdot \mathbf{d} > \cos\phi_s$), the intensity is given by the function of $\cos^e \phi$. Note that the addressed vectors are normalized. I set property `spotlight_angle` as the $\cos\phi_s$ (< 1) to simplify the calculation. I set the exponent e as euler number, e (2.718)!

2.2 Point light and Directional light

The specifications note that I may choose one from the point and the directional light source, but I implemented both (though I did not use point light source in my scene).

Point light The intensity by a point light source can be derived using the inverse square law. Since we know the position of the light source (stored in transform), we can calculate the distance between fragment and the light source without adding any new property.

Directional light Directional light can be easily implemented using the direction property we used to implement spotlight. The intensity by a directional light source is proportional to $\cos\phi$, which is $\mathbf{l} \cdot \mathbf{d}$.

(a) Phong shding

(b) Toon shading

Figure 1: Shader Programming

3 Shader programming

3.1 Phong shading

I implemented phong shading (per-fragment shading) using the function of Blinn-Phong reflection model.

$$I = k_a L_a + k_d L_d \max(\mathbf{l} \cdot \mathbf{v}, 0) + k_s L_s \max((\mathbf{n} \cdot \mathbf{h})^{\alpha}, 0)$$

I defined ambient, diffuse, specular reflectance and the shininess coefficient properties to the material. Also, I defined corresponding illuminance properties to the light source. The halfway vector $\mathbf{h}$ is defined as the normalization of $\mathbf{l} + \mathbf{v}$. For convininence, I didn't incorporate the distance term. (I didn't think it was necessary.) Also, I assumed that materials in my scene have same color for each reflection model.

When it comes to calculating intensity from a reflection model, the type of a light source is irrelevant to the reflection model. Thus, we just have to apply the effect by the source and the material indepentently.

3.2 Toon shading

I implemented a toon shader by first implementing a vertex shader with Gouraud shading, which is quite simple. (Just set fragment normal as the vertex normal). Then I used the Blinn-phong reflection model to calculate the intensity in similar way as I did in Phong shading. Lastly, I divided discrete levels of intensity to make it look 'cartoon-like'.

The results are shown in 1.

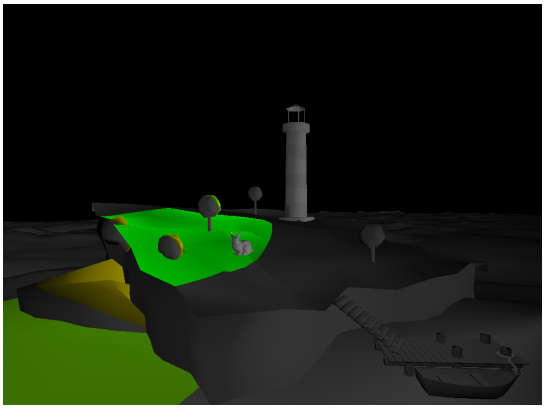
4 Creating scenes

I created a scenary of an island with a lighthouse. It came to my mind, that a day of a lighthouse keeper is an ever-repeating cycle. When the night arrives, the light gets on. When the dawn is drawn, the old man may finally sleep. (hmm... sounds familiar.) The cycle was expressed using two different light sources. The light from orbiting sun was implemented using directional light. (Though the sun may seem like a sphere - perhaps a point light source, It acts as a directional light since it's so far away.) The background color changes when the sun is below the horizon, so the sky of night and day will actually be different. When it's dark, the lighthouse emits a rotating spotlight.

The scenary is actually including two cycles - a 30 seconds cycle of day and night, and a 8 seconds cycle of rotating spotlight. You will see smooth animations - one implemented using keyframes, and one implemented using classic approach (which we used in HW 1 and 2.)

I also tried to create an exotic scenery with a message using MyMaterial, and it's shown in 2b.

(a) Phong shding



(b) Toon shading

Figure 2: Night Scenery

You can see that in MyMaterial, all material is black-and-white, except the area where the spotlight is shredded. Think of the life of a lighthouse keeper. The world may seem boring and gloomy, being trapped in a same routine in a small island. But one may seek beautiful things. Life can still be colorful - if you just keep the light of hope alive.

5 Creativity

- I tried to put a story in my scenery, related to the theme of 'cycle'.
- Though object animation was not mandatory, I implemented simple object animations to make my scenery look more alive.
- You can press 't' to stop the cycle - as long as you want. The bunny and the boat will keep moving.
- I implemented phong shading and toon shading insided MyMaterial and its shaders. You can press 0, 1, or 2 to change the mode to phong, toon and my shading.