CS606 Computer Graphics
Spring 2020
Agam Kashyap
IMT2018004

Lighting and Quaternions

Problem Statement

The given task is to implement lighting from multiple sources present in a scene and to further implement rotations using quaternions. We must also implement two different types of shading i.e. Gourad Shading and Phong shading.

Approach

Lighting

I began by adding lighting to the scenery. The first step was to introduce directional lighting which introduced me to the normals and their use in determining the effect of lighting. Directional lighting acts on each of the faces, i.e. it calculates the dot product of the normal to a face with the direction of the light and we add this result to the color of the object.

Moving on I implemented point lighted on a single object. This is the case wherein we interpolate the normals for each of the points present on the mesh. This enables the effect of a more natural object reflecting light from it's surface. In this case, I passed the normals calculated into the fragment shader. The lighting that I calculated was the Blinn Phong model, wherein we pass in the position of the eye/camera as well.

The shader implemented at this stage corresponds to the Phong shading method, wherein we calculate the normals and the colour values in the fragment shader. I set the ambient color, diffusion color and specular color along with their corresponding coefficients to result in a GOLD type surface.

The next task was to implement the Gourad shading. This is nothing but performing all the colour calculations in the vertex shader and then passing the final calculated colour to the fragment shader.

2. What are your observations about the change in the shading model?
The approach of the two shading methods makes the difference in the two shading methods evident. In gourad shading since we are not interpolating the normals for each of the points, the specular part will not appear smooth, which means the faces will still be visible(also called flat shading). In Phong method of shading the surface will appear smooth (Fig 1). This is a defining difference between the two.

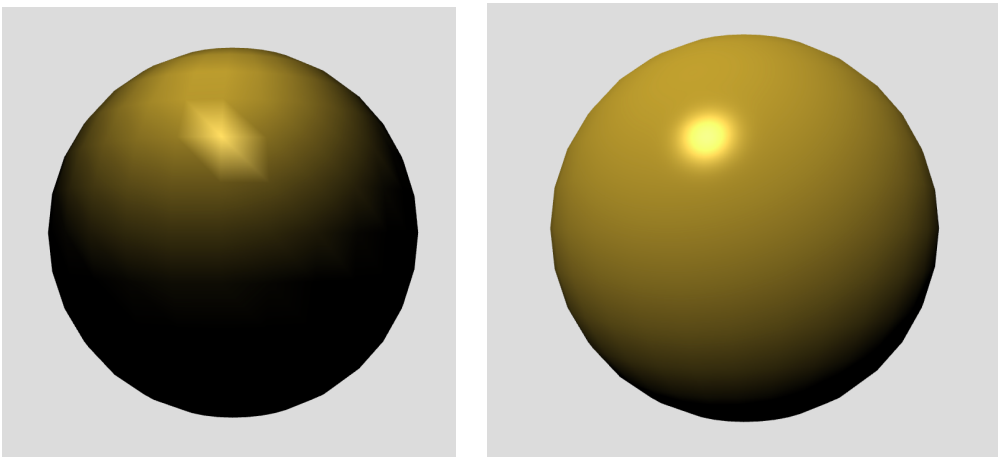

Figure 1: Left - Gourad shaded sphere; Right - Phong Shaded sphere;

After having implemented lighting, my next task was to translate the lights within the bounds of the bounding box of the figures. For finding the bounding box I just iterated over all the vertices of the mesh and found the maximum and the minimum values of x, y and z coordinates. I set the limit as 1.25 times these values found. While translating, if the light reaches one end of the limit it enters the box from the opposite end.

Initially, this is the stage where I got an output which was less desired. My canvas space initially covered a space of approximately 1200 pixels across(the camera was placed at a longer distance). Since the meshes were all small in size, I had to scale the objects by as high as 40 times. This increased the limits of the light positions by a very large amount. So the position of the light for an object was able to go as far as beyond the other objects. To solve this issue, I reduced the width of the viewspace by bringing the camera closer.

I also added attenuation, which required a lot of experimenting to get the proper coefficient values. I used the $1/(a + bd + cd^2)$ formula for attenuation calculation.

Quaternions

After having done the basic lighting, the next step was to implement the rotation using quaternions. I had already implemented a system of rotation for the cameras using mouse drag in all directions, though that used basic geometry to move a point along the sphere.

To implement the rotation using quaternions we need two things - the axis about which to rotate, and the angle to rotate the vector about that. For doing this, the basic logic remains the same as that of camera rotation. Say we drag our mouse from A(X_1,Y_1) to B(X_2, Y_2) on the screen. Now assume we have a sphere of unit radius present at the origin. We need to find the corresponding points on the sphere to A and B. Simple trick for doing this is to assume the screen is a plane present at some positive Z value. Now, the points on this sheet can be mapped onto the sphere by simply normalizing the coordinates A and B with z-coordinate as Z.

Now, we have two points A' and B' on the sphere, between which we have to perform the transformation. These two points are vectors with one endpoint as the center of the sphere. We can find the axis of rotation to go from A' to B' by taking the cross product between the two. The angle by which to rotate, we can find using the dot product of these two vectors. Hence, we now have the quaternion to perform the required rotation.

Now, a subtask here was to ensure that the previous performed rotation is saved, else the object would always reorient itself to the starting orientation. For achieving this, I stored the previous rotation matrix and multiplied the new rotation matrix(created from the quaternion) before sending it to be multiplied to the objects.

As explained earlier, the task of using the mouse as a trackball was a simple task of just keeping track of the starting point of mousedown A and the other point B.

Multiple Lights

The last part was a bit tricky, given that it required restructuring the code. We need to access the position of all the lights and the colours of them as well. For enabling this I created a separate class of Light wherein I created a struct containing all of these values. The position values of the lights were automatically updated in this struct while translating. I created an array of structs of all the light sources and passed it to each of the mesh objects. In the shader I created a struct for holding all these values to make it more modular. The rest of the process remained the same. All we had to do now was just keep adding the resultant colours from all of these light sources.

For turning the lights on/off I used a simple boolean to indicate whether a certain light is on or not and only added the resulting colour from it to the final Color if it was on.

1. What are your observations of the distance attenuation terms used for lighting?
As mentioned earlier, this part required a lot of testing with the attenuation coefficients. But, the basic logic behind this term is to come up with a fucntion which is inversely proportional to the distance of a point from the light source. As the distance would increase the output would reduce. One other point to keep in mind, is that the attenuation term must be between 0 and 1.

3. You are now able to generate different sizes of specular highlights using different settings for lighting, shading, and materials. When do you see focussed sharper and smaller specular highlights, and when do you see larger ones?
What I understand from lighting is the colour that it's different sections hold and the position. So, if the light source is closer to the surface of the object, the specular highlight will be smaller and sharper. The colour of this highlight will be determined by the specular colour.

But, the distance is not enough to determine the same, we also need to set the

shininess value for the object. As the shininess increases the spot becomes more and more focused, and it becomes more spread out when the shininess decreases.

The material properties are set by defining the coefficients for ambient(Ka), specular(Ks) and diffusion(Kd). I created two set of materials, one is the GOLD for which the features are `ka : 1.0, kd : 1.0, ks : 1.0, Ambient color : (0.0, 0.0, 0.0), Diffuse Color : (0.752, 0.606, 0.226), Specula rColor : (0.628, 0.556, 0.366) , Shine : 200` while another is MATTE for which the features are `(ka : 0.33, kd : 0.27, ks : 0, Ambient color : (0.81, 0.81, 0.81), Diffuse Color : (0.666, 0, 0.8), Specular Color : (1.0,1.0,1.0), Shine : 20`

The Difference between the two can be seen in Fig 2.

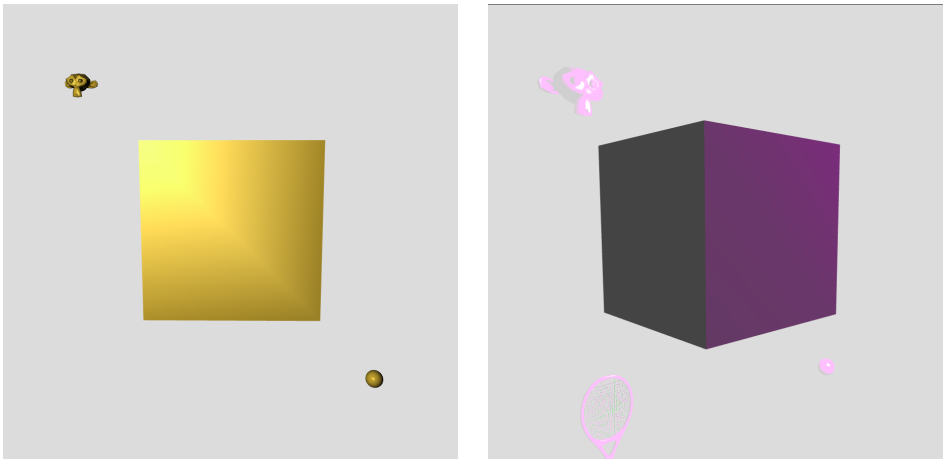

Figure 2: Left - Gold; Right - Matter;

In terms of shading model chosen, the phong method will result in sharper specular highlights as compared to the gourad shading model.

4. What are your comments about your choice of mesh models for this assignment?
This was an important factor, because, for obvious reasons, I did not make the more complex meshes such as the monkey or the tennis racket. Before deciding on these meshes, I had downloaded other meshes, but in some of them, I was not able to see any light based output. It remained black. The reason for this, which I understood later was that in certain mesh OBJ files, there are no normals. Since the light value is calculated as the dot product of the normal with the direction of the light, this value will always be zero if the normal is non existent.

Another important factor was choosing meshes which had curved surfaces so that I could realise the difference in the shading and different type of lightings.

Conclusion

At the end of this assignment I was able to get a clear understanding of the two types of shading model and the difference between the two. This was also useful in

helping me understand the three properties, specular, ambient and diffuse of light.