Assignment – 3

Name: Ishan Mehta

Roll No: 2019309

The folder code_1, code_2 and code_3 contain the code for the question 1, question 2 and question 3 respectively.

Question 1)

The code for this question is in code_1 folder.

Part a)

The coordinates of vertices of triangles which form the sphere can be calculated by changing the parameters in the parametric equation of the sphere.

The parametric equation of a sphere of radius R is given by ->

$$X = Rsin(\phi)\cos(\theta)$$

$$Y = Rsin(\phi)\sin(\theta)$$

$$Z = Rcos(\phi)$$

Changing the value of θ while keeping value of ϕ constant will make a point move in a circle whose plane is parallel to the XY plane. This circle is uniquely defined by the parameter ϕ.

Part b)

The location of the point source is -> 30, 0, 0. This is stored in a variable light_pos.

The Colour of the point source is -> 1,0,0. This is stored in a variable light_color.

Light_pos and light_color are passed as uniform to the vertex shader from the c++ program.

Part c)

Diffuse lighting is calculated by the formula

$$diffused\_light = kd * light\_color * max(0, dot(light\_dir, normal));$$

Kd is the diffusion constant. (It is a scalar)

Light_color is the color of light. (It is a vec3)

Normal is the normal vector at the vertex (It is a vec3)

Light_dir is the vector pointing towards the light source from the vertex (It is a vec3)

The back of the object appears dark because of no ambient light.

Question 2)

The code for this question is in code_2 folder.

The three components of phong lighting are ambient, diffused and specular.

Ambient light -> It is assumed that every point in the scene is incident by some light ray. This means no point is completely dark. To simulate this effect, we give ambient light to each vertex making them lit with a very low intensity light.

$$ambient\_light = ka * light\_color;$$

Ka is the ambient constant (0<=ka<=1)

Diffused light -> It is assumed that all points in a surface scatter light almost in all directions. Also the intensity of light scattered depend on the angle of the normal at that point with light_dir vector defined above. Less the angle more is the intensity. If the angle is more than 90 degrees the intensity is considered to be 0. Mathematically this can be expressed as the following equation which uses dot product to which is proportional to this angle.

$$diffused\_light = kd * light\_color * max(0, dot(light\_dir, normal));$$

Specular light -> The view_dir is the vector pointing from the point vertex to the camera. If the angle between this vector and reflected angle at the vertex is less, then we can say that max light reaches the camera. This gives an effect of shiny white spots on the object surface.

$$specular\_light = ks * light\_color * pow(max(0, dot(h , normal)), 32);$$

The value of the dot product remains between 0 and 1, this is because h and normal are both unit vectors. The value α (here 32). Signifies how fast the value of this dot product will reduce to zero. Higher the value of α shorter the shiny white spot will be.

H is the half way vector => normalize(L + V)

L is the light_dir and,

V is the view_dir.

This gives a similar result and is easier to compute.

All these calculations are done in vertex shader. It is noted that the side of object not facing the light source is also lit. Also there is a shiny white spot on the surface of the object.

Question 3)

The code for this question is in code_3 folder.

Same calculations as in Questions 2, but this time they are done in the fragment shader to give a more smoother colouring effect.

Question 4)

The code for this question is in code_4 folder.

The colour of each vertex with normal n = (nx,ny,nz) is taken as C = ((nx + 1)/2, (ny + 1)/2, (nz + 1)/2)

This gives a rainbow like look on the entire sphere. This code is also in the fragment shader.