Graphics Lab Project

Project Name: 3D Table Lamp

Submitted To:
Ms. Shuvra Chakraborty
Lecturer
Ms. Ismat Rahman
Lecturer

Submitted By:
Salsabl Ahmed Khan (Roll-11)
Fariha Moomtaheen (Roll-23)

Department of Computer Science and Engineering,
University of Dhaka.

Contents:

1.Project Overview:

3D Table Lamp encapsulates the 3d appearance of a lamp along with lighting and shadow effects. There are multiple static light sources to show different effects of light and shadow. There is also one user interactive light which can be circled around the lamp to show the effect of lighting from different angles.

This project also shows the functions of rotation, translation and perspective.

The project is developed using OpenGL graphics API and C++ programming language.

2. Purpose of this project:

The purpose of making this project is to visualize the lighting effects of a table lamp in a better way. Again by making this project we get to know graphical work and many useful API in this language. And we have used various built in functions. Thus we get a vivid idea of the effective and proper use of OpenGL.

3. Library And Utility Toolkit:

We have used OpenGL library and GLUT utility toolkit to accomplish this task.

Open Graphics Library (OpenGL) is a cross-language, cross platform application programming interface (API) for rendering 2D and 3D vector graphics. The API is typically used to interact with a graphics processing unit (GPU), to achieve hardware accelerated rendering.

The OpenGL utility toolkit (GLUT) is a library of utilities for OpenGL programs, which primarily perform system-level I/O with the host operating system. Functions performed include window definition, window control and monitoring of keyboard and mouse input.

4.Project Designing and Planning:

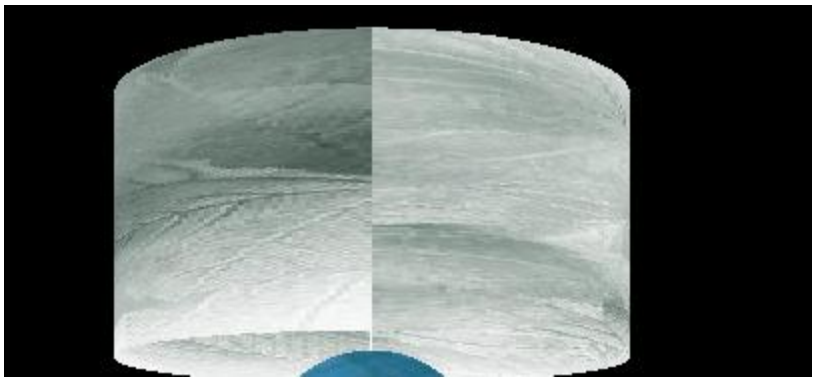
This project is nice work of pipelining in OpenGL, Light effect, Textures. We used many library to accomplish the task along with the 3D modelling and texturing object.

4.1 Structure of the Table Lamp:

A table lamp consists of multiple parts, that have been drawn here with 3d simulation.

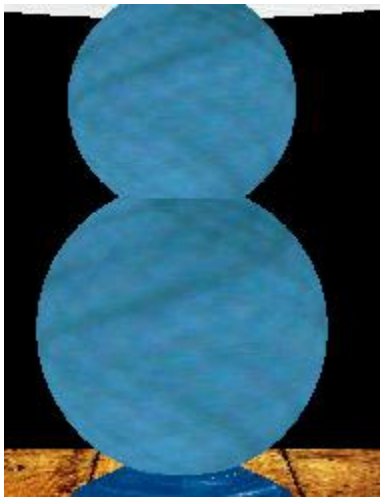
4.1.1 Lamp Shade:

The lamp shade has been drawn using glut cylinder function. We have also used texture to bring the effect of cloth.



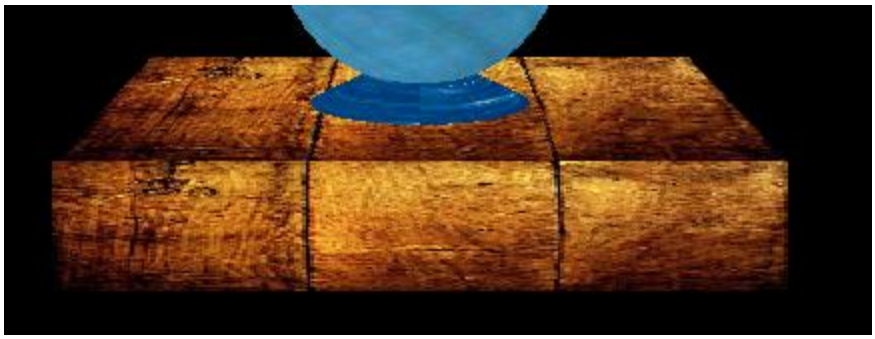
4.1.2: Body of Lamp:

The body was drawn with two spheres drawn on top of each other and again texture was used to make it look realistic.



4.1.3: Base of Lamp and Platform:

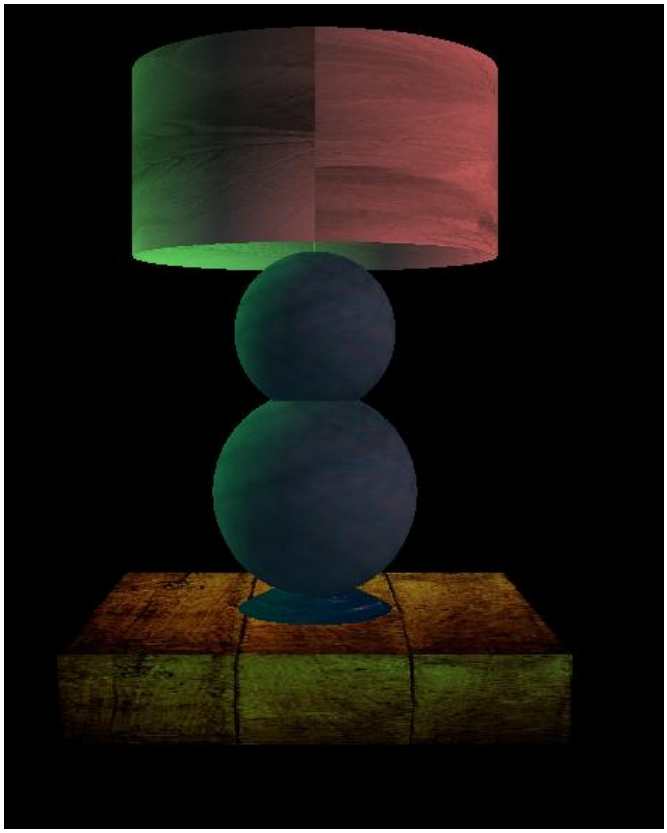
The base of lamp was again drawn using glut cylinder and the platform was drawn with the help of 6 GL_QUADS.



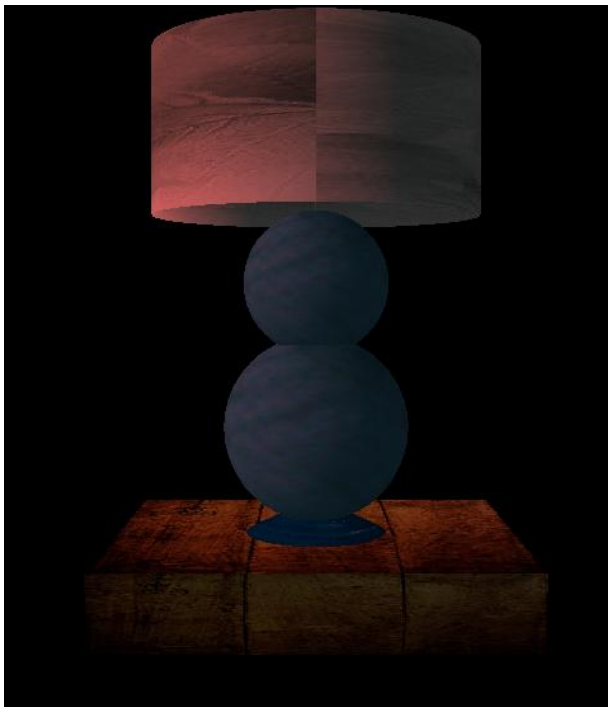
4.2 Lighting:

We have used various lighting and shadow effects to simulate the effects of a table lamp.

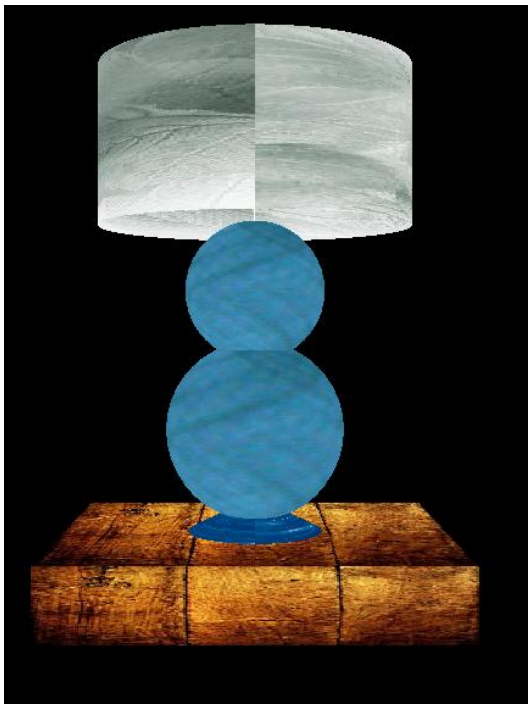
There are 3 static lightsources to the left, right and front of the lamp in colors green, red and blue respectively to show the effect of lighting on an object from different directions and the corresponding shadows. To do this, while drawing each surface we have found the normal for each vertex of that surface. This is because OpenGl needs to know a normal for every vertex it is lighting. OpenGL does its lighting computations at vertices for efficiency reasons.



Now the red light source can also be circled around the lamp in an anticlockwise direction when the user presses the button ‘i’. This is done to show the effect of the same light source from different directions of an object. Here we show when the red light is circled to the left of the lamp in an anticlockwise direction.

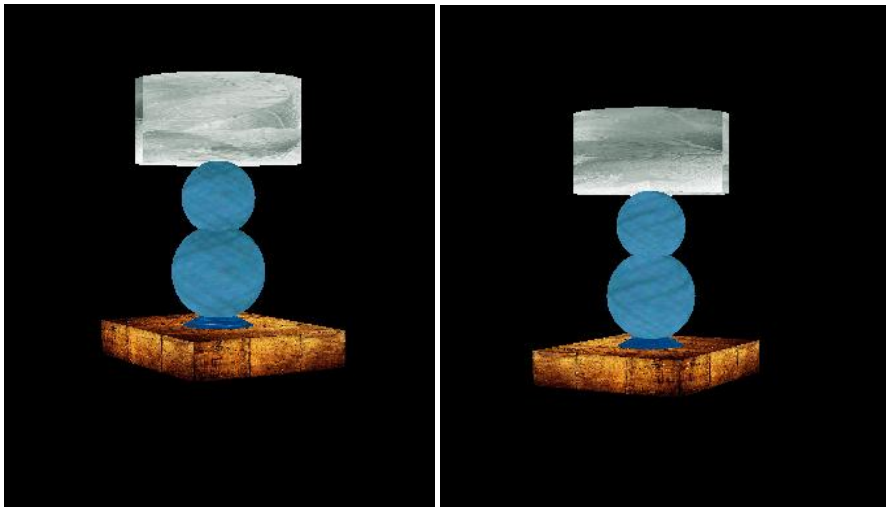


Now we show the effect of light and shadow when the lamp is turned “on” and all other lights are turned off.

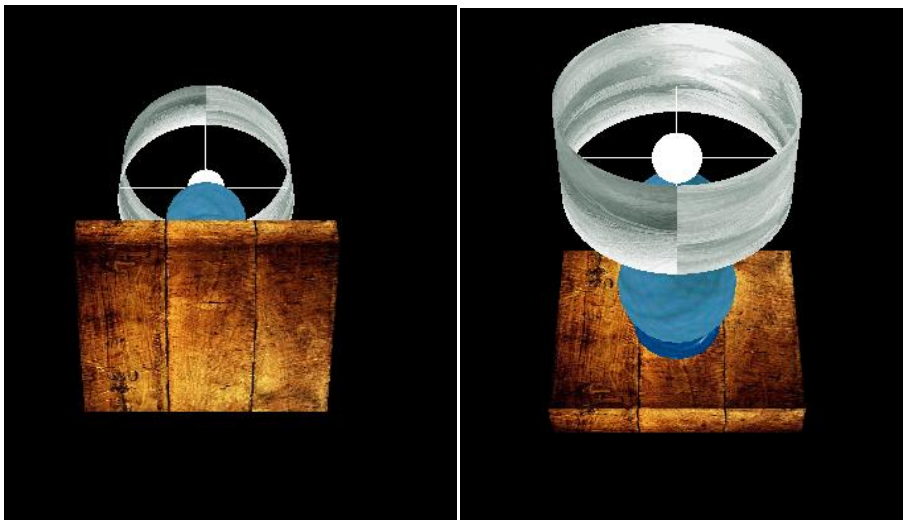


4.3 Rotation:

We also rotate the lamp about X axis in both clockwise and anticlockwise direction when the user presses the buttons ‘1’ and ‘2’.



The lamp is rotated about Y Axis in a similar way when the user presses buttons ‘3’ and ‘4’.



5. Code Generaton:

5.1 : Structure of Lamp:

The code for the structure of the lamp is as follows:

5.1.1 Lamp Shade:

```
glColor3f(1,1,1);
glPushMatrix();
glEnable(GL_TEXTURE_2D);
glTexEnvf(GL_TEXTURE_ENV, GL_TEXTURE_ENV_MODE, GL_MODULATE );
glHint(GL_PERSPECTIVE_CORRECTION_HINT, GL_NICEST);
glBindTexture(GL_TEXTURE_2D, texName[0]);
drawCylinder(4.6,7.8,7,7);
glDisable(GL_TEXTURE_2D);
glPopMatrix();
```

5.1.2 Body of Lamp:

```
glPushMatrix();
    glEnable(GL_TEXTURE_2D);
        glTranslated(0,-4.5,0);
        glRotatef(90,1.0f,0.0f,0.0f);
        glTexEnvf(GL_TEXTURE_ENV, GL_TEXTURE_ENV_MODE, GL_MODULATE );
        glHint(GL_PERSPECTIVE_CORRECTION_HINT, GL_NICEST);
        glBindTexture(GL_TEXTURE_2D, texName[2]);
        gluQuadricDrawStyle(obj1, GLU_FILL);
        gluQuadricNormals(obj1, GLU_SMOOTH);
        gluQuadricTexture(obj1,1);
        gluSphere(obj1,4,20,20);
    glDisable(GL_TEXTURE_2D);
    glPopMatrix();
```

5.1.3 Platform:

```
glBindTexture(GL_TEXTURE_2D, texName[4]);
glBegin(GL_QUADS);
        glNormal3f(0.0f, 1.0f, 0.0f);
        glTexCoord2f(0.0, 0.0); glVertex3f(-8.5, -11, -7);
        glTexCoord2f(0.0, 1.0); glVertex3f(8.5, -11, -7);
        glTexCoord2f(1.0, 1.0); glVertex3f(8.5, -11, 7);
        glTexCoord2f(1.0, 0.0); glVertex3f(-8.5, -11, 7);
glEnd();
```

5.2 Texture Mapping:

We have wrapped each part of the lamp with texture to give it a 3d look. We first loaded the image to our program, converted it to a 2d array by writing the RGB values of each pixel, then we convert this into a linear array and this linear array is transformed into texture and bound with every shape.

```
for(r=0; r<imageheight; r++)
      {
            for(c=0; c<imagewidth; c++)
            {
                  fread(&blueValue, sizeof(char), 1, fp);
                  fread(&greenValue, sizeof(char), 1, fp);
                  fread(&redValue , sizeof(char), 1, fp);
                  image[index][r][c][0] = (GLubyte)blueValue;
                  image[index][r][c][1] = (GLubyte)greenValue;
                  image[index][r][c][2] = (GLubyte)redValue;
                  image[index][r][c][3] = (GLubyte)255;
            }
      }
```

5.3 Lighting Effect:

Parameter Name	Default Value	Meaning
GL_AMBIENT	(0.0, 0.0, 0.0, 1.0)	ambient RGBA intensity of light
GL_DIFFUSE	(1.0, 1.0, 1.0, 1.0)	diffuse RGBA intensity of light
GL_SPECULAR	(1.0, 1.0, 1.0, 1.0)	specular RGBA intensity of light
GL_POSITION	(0.0, 0.0, 1.0, 0.0)	(*x*, *y*, *z*, *w*) position of light
GL_SPOT_DIRECTION	(0.0, 0.0, -1.0)	(*x*, *y*, *z*) direction of spotlight
GL_SPOT_EXPONENT	0.0	spotlight exponent
GL_SPOT_CUTOFF	180.0	spotlight cutoff angle
GL_CONSTANT_ATTENUATION	1.0	constant attenuation factor
GL_LINEAR_ATTENUATION	0.0	linear attenuation factor
GL_QUADRATIC_ATTENUATION	0.0	quadratic attenuation factor

The function LightingParameter() is shown below:

```
const GLfloat light_ambient[]  = { 0.5f, 0.5f, 0.5f, 0.0f };
 const GLfloat light_diffuse[]  = { 1.0f, 1.0f, 1.0f, 0.0f };
 const GLfloat light_position[] = { 0.0f, 7.0f, 0.0f, 0.0f };
 const GLfloat mat_ambient[]    = { 1.0f, 1.0f, 1.0f, 1.0f };
 const GLfloat mat_diffuse[]    = { 1.0f, 1.0f, 1.0f, 0.0f };
 const GLfloat high_shininess[] = { 50.0f };
 glLightfv(GL_LIGHT1, GL_AMBIENT,  light_ambient);
 glLightfv(GL_LIGHT0, GL_DIFFUSE,  light_diffuse);
 glLightfv(GL_LIGHT0, GL_POSITION, light_position);
 glMaterialfv(GL_FRONT_AND_BACK, GL_AMBIENT,   mat_ambient);
 glMaterialfv(GL_FRONT_AND_BACK, GL_DIFFUSE,   mat_diffuse);
 glMaterialfv(GL_FRONT_AND_BACK, GL_SHININESS, high_shininess);
```

5.4 Key Listener:

We have user interaction for multiple purpose such as turning the lights “on” and “off”, rotating the lamp about X and Y Axis and projection.

```
switch (key) {
case 27:
   exit(0);
case '1':
  X = X + 1;
  break;
case '2':
  X = X - 1;
  break;
```

```
case 'a':
        glEnable(GL_COLOR_MATERIAL);
        glEnable(GL_LIGHTING);
        glEnable(GL_LIGHT0);
        LightingParameters();
        break;
 case 's':
        glDisable(GL_LIGHT0);
        break;
```

6 Future Plan:

1. We will try to rotate all three outer light sources in every direction possible, to show the effect of lighting on an object better.
2. We will try to do better implementation of projection.

7 Conclusion:

While doing this project we faced various problems and due to overcoming these problems we learned various aspects of graphics. For example, we learned how built in functions for texture mapping uses BMP image, how we have to bind the image and how the API reads the image.

We have also learned various aspects of lighting and shadows. We have learned that OpenGL needs the normal of all the vertices of a surface to computate lighting.

Overall, we have gotten a better perspective of graphics design through this project.