Assignment 6: Lighting and Shading a Picnic Table

In this assignment, you will build a Picnic Table with a shade and four stools around it. You will add normal as an attribute. You will apply proper lighting and shading.

Add objects to the scene:

1. Compile and run the attached program. You will see a pole (scaled cube) standing on the ground.
    a. Add normal as an attribute both for the plane and the pole. **1 point**
    Hints: See Week 8 example "CubeAmbientDiffuse."

2. **Add a shade:** **Total 3.5 (1.5 + 2) points**

    a. The shade can be constructed from a pyramid with a square base of 5 units and height 1 unit. This has been demonstrated in Figure 1. **OpenGL Tutorial Week 7** and **OpenGL Tutorial Week 8-I** will be helpful to start working on this part.

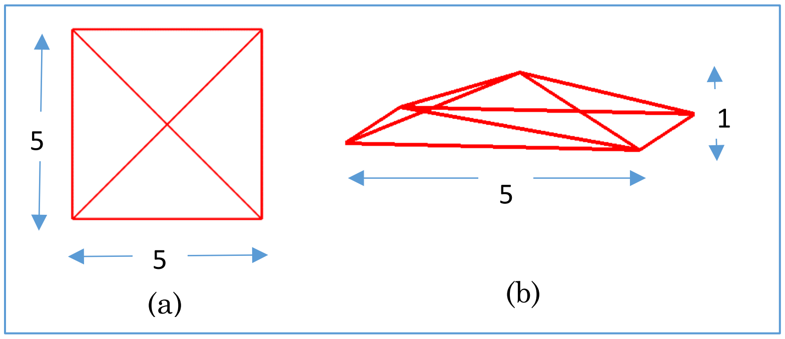

Figure 1: (a) Top view and (b) front view of the shade.

    b. In addition to the geometry of the shade, you also need to compute normals for the shade. I'll discuss about this in the class.

3. **Add a table:** **Total 2 (0.4 + 1.6) points**

    a. Place the tabletop at the center of the pole. Scale the cube to build a tabletop with dimension 3.0X3.0 and height 0.25.
    b. Add four legs to the table by applying appropriate transformation on cubes. Legs should touch the ground. Use shear transformation to make them look like Figure 2. We worked on shear matrix in Assignment 4. You have the flexibility to choose the shear factor. In addition to shear transformation, you need to apply other transformations, i.e., translation and rotation to place the legs properly under the table.
    Hints: See OpenGL Tutorial 11 in Week 7

4. **Add four stools:** **0.8 point**
   a. Add four stools each of dimension 2X2X2 onto the scene. Place them on four sides of the table at 3.0 from the center of the table. The stools should properly stand on the ground.

5. **Camera:** **0.2 point**
   Move the camera around the origin at a radius of 8.0 onto the XZ plane. Use glutTimer Func. While moving around the scene, camera will be placed at 2.75 along the positive Y axis. Switch between the top view and front view with the toggling of 't'/'T'.

Lighting and Shading **Total 5 points**

1. Add the following light and material parameters. Use them as uniform variables in shaders. **0.5 points**

```
vec4 light_position(5.0, 10.0, 5.0, 1.0);
vec3 light_ambient(0.3, 0.3, 0.3);
vec3 light_color(1.0, 1.0, 1.0);
vec3 material_color(0.9, 0.5, 0.3);
float shininess = 50.0f;
```

   Color of an object (defined by **material_color**) is not an attribute in this assignment; color is passed as uniform variables to shaders.

2. **Add the ambient, diffuse and specular components of the illumination model to the scene**

   After applying lighting and shading by considering different components of the illumination model and material properties, the properly shaded environment will look like Figure 2. The uploaded video will be helpful to understand the output better.

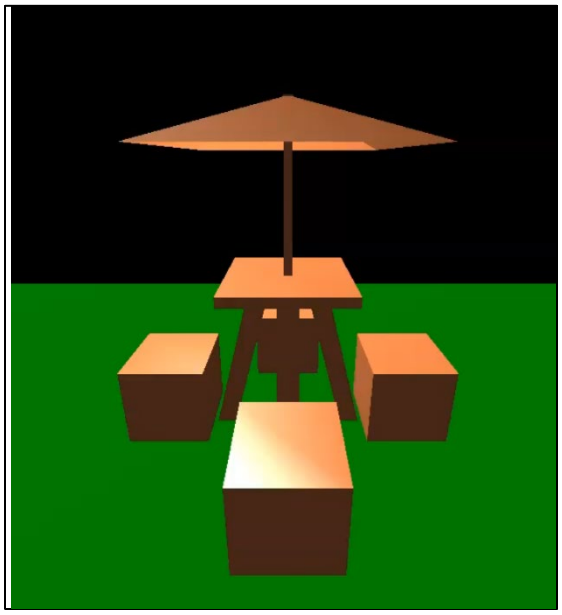

Figure 2: A shaded picnic table.

a. Add ambient component of the illumination model to the scene. **0.5 point**

b. For diffuse reflection, consider the light direction and per vertex normal. Do the eye space conversion of the vertices and normals by multiplying with the appropriate matrices. Calculate per vertex light direction. Compute light position in the eye space. **2 points**

c. For specular reflection, consider the view direction in addition to the direction of light. In the eye (camera) space, the viewer is located at the origin. Use half vector for specular reflection as discussed in the class. **1.5 points**

d. Calculate the combined light intensity to get the desired result. Consider material color to get the final color. **0.5 points**

Hints: Uploaded examples **CubeAmbient and CubeAmbientDiffuse** will be helpful to work on this part. I'll discuss about the specular component of the illumination model in the class.

Bonus part **2 points**

In the basic part, we computed per-triangle normals for vertices. Compute per-vertex normals. This necessitates unique vertex and normal attribute arrays that define the tringle primitives with the indices of attributes. So, use glDrawElements for this part. For a unique normal array, every vertex normal should be the vector addition of the normals of the triangles that share that vertex. This has been discussed in **CSCD377 Graphics Basics 21**. After you implement this part, you can compare the output of the basic part with the bonus part of the assignment. Look at the uploaded video.

Submission

Basic: Place your solution in a zipped file named with your last name followed by the first initial of your first name followed by 6 (ex: **CSCD377YasminS6Basic.zip**) and submit the solution via canvas. Thus, your zip should contain the following files:

- Cube.h, Cube.cpp, Plane.h, Plane.cpp, ShadeBasic.h, ShadeBasic.cpp, main.cpp, and shader files.

Bonus: If you would like to include the bonus part, submit the zipped file as **CSCD377YasminS6Bonus.zip**. Thus, your zip should contain the following files:

- Cube.h, Cube.cpp, Plane.h, Plane.cpp, ShadeBonus.h, ShadeBonus.cpp, main.cpp, and shader files.

Submission deadline is Tuesday, November 21, 11:59 pm. This assignment weighs 12.5% of this course.