3D Project Part 1

Overview

In this project, you will develop a basic 3D application that will showcase several features of 3D Graphics using OpenGL. You are required to use the accompanied project as the code base for this project. Please do not use any other codebase.

Please watch the 3D Project 1 Demo.mp4 from the shared Google Drive first. You can also try the attached executable demo in the SuCourse. This is an example of what we are expecting from you.

Grading
Scenes 1, 2, 3: 15 points each,
Scenes 4, 5: 20 Points each,
Scene 6: 30 Points (with bonus)

What to Submit
You should submit an executable and a report. The report should have a screenshot of each scene followed by an explanation of how you have achieved it.
Note: Change the window title with your name.

Scene System

You will create a program with multiple scenes, each rendering something different. These scenes should be shown when the related key is pressed. The keys you will use are Q, W, E, R, T, and Y in order, e.g. when the E key is pressed 3rd scene should be shown.

In each scene (except the 5th and 6th), the same four meshes must be rendered. For the meshes on the upper side, you must render a Sphere on the left, and a Torus on the right. They must be generated with the functions `ParameticHalfCircle` and `ParameticCircle` respectively, with 16×16 samples. The meshes on the bottom are left to your imagination, they can be any shape, does not have to be the ones on the demo. Also, these meshes must rotate around themselves as in the demo. To be exact, the rotation axis must be `(1, 1, 0)`, and the rotation angle must be `current_time*10 degrees`.

First Scene

You will render the meshes as wireframes. The color of the wireframe should be white, `(1, 1, 1)`.
Hint: You may utilize `glPolygonMode` for this.

Second Scene

You must create a shader that shows the normal vectors as colors.

Third Scene

You must use the Blinn-Phong reflection model to shade the meshes. All the meshes must have a gray surface color, `(0.5, 0.5, 0.5)` with a shininess value of `64`. There must be a single *directional light* as the only light source. Its direction should be `(-1, -1, 1)`, from the near left upper corner to the origin, and its color must be `(0.4, 0.4, 0.4)`. The *ambient light* is `(0.5, 0.5, 0.5)`.

Fourth Scene

You should again use the Blinn-Phong shading. However, each mesh must have its own surface color: gray `(0.5, 0.5, 0.5)`, red `(1, 0, 0)`, green `(0, 1, 0)`, and blue `(0, 0, 1)`. You must use the shininess values `128` and `32` for the Sphere and Torus respectively. The other meshes' shininess values are up to you. While keeping the previous *ambient light* and *directional light* as it is, a mouse-controlled *point light* must be added. Its color must be `(0.5, 0.5, 0.5)`, and its position must be computed as `(mouse_x, mouse_y, -1)`.

Fifth Scene

You must develop a small game as in the demo. One sphere must be in the position of the mouse, while the other sphere is chasing the first one. To keep things consistent, you must calculate the position of the chasing sphere with the below function:

```
chasing_pos = glm::mix(mouse_pos, chasing_pos, 0.99)
```

You must use the Blinn-Phong shading, but the light sources are up to you. Both of the spheres should be the Sphere you have generated for the first scene and scaled by `0.3`. If the distance between the mouse-controlled sphere and the chasing sphere is larger than `0.3 * 2`, the mouse-controlled sphere should be green, otherwise red. (When you catch the sphere it blushes)

Sixth Scene

Do whatever you desire, try to impress yourself, but report it in detail.