Assignment 1 : Creating your own OpenGL program

author：夏苏安 student Number：18047482
email：xiasa@shanghaitech.edu.cn

1. Introduction

- [must] You are required to enable the multi-sample full-screen antialiasing functionality of OpenGL.

We search for "OpenGL multi-sampling" for more information online. And we find that we only need 2 lines of codes to complete it.

- [must] You are required to draw simple objects : triangle, tetrahedron, cubes and spheres.

After analysis, we need 3 vertices for a triangle, 4 triangles for a tetrahedron, 6 quadrangles for a cube and many samples of triangles for a sphere.

- [must] You are required to render 3D objects with Phong lighting model.

We use the build-in function in OpenGL to realize it easily.

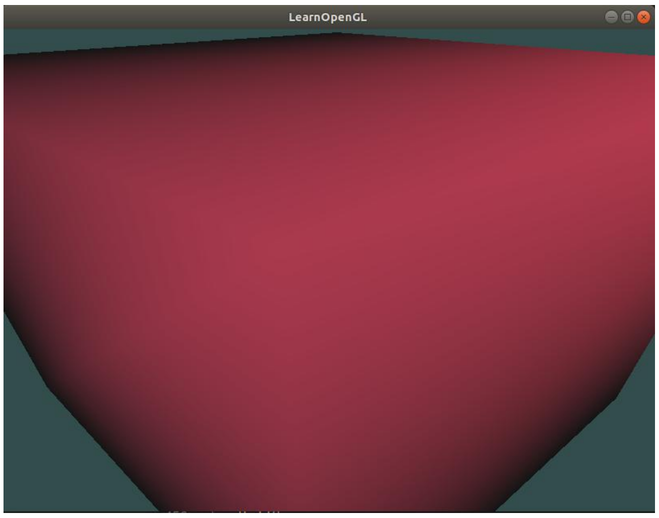
- [must] You are required to manipulate the camera by constructing the new camera matrix with gluLookAt(...) and use keyboard to control the camera: you can use keyboard to translate and rotate the camera so that you can walk in the virtual scene.

The keyboard input “WSAD” represents “forward back left right” four actions. Further more, there are 2 kinds of actions, first is along the forward vector and the other one is along the right vector. The mouse input represents the changes of yaw and pitch, which means changes of the direction vector.

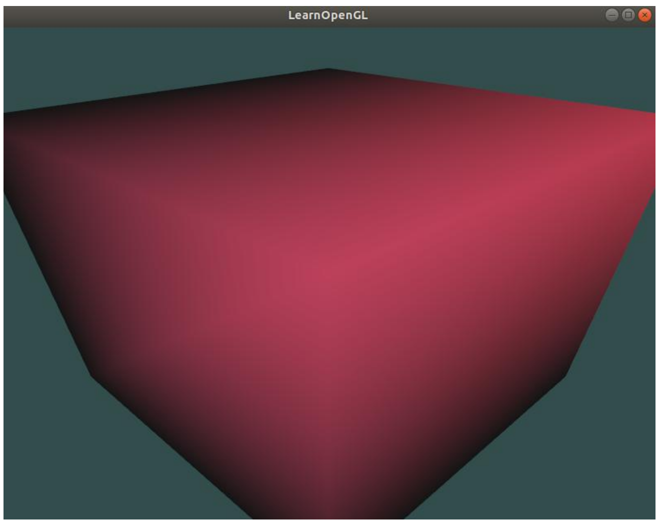
2. Implementation Details

- glfwWindowHint(.......); glEnable(.........);
- For each vertex, there is a normal vector as well. It well help render in the Phong lighting model.
  Also,
- Remember to glEnable(Normalize).
- In function setlight(), we will open the lighting and ligh0. we also need to define and pass the parameters of ambient, diffuse and specular, which decides the strength of each light.
- In function setmaterial(), we also need to define and pass the parameters of ambient, diffuse and specular.
- In function processinput(), we judge which input it is. And according to this, we call back the changePMV() function to only change the position of the camera. If it is W or S, we increase or decrease along the front vector. If it is A or D, we increase or decrease along the right vector.
- To capture and process mouse input, first we have to tell GLFW that it should hide the cursor and capture it: glfwSetInputMode(......). Then register it to OpenGL: glfwSetCursorPosCallback(.......);
- Then, we should define mouse input callbacks:
- In function mouse_callback(), we first calculate the offset of x and y, then adjust the offset so that it is valid. Next increase or decrease the yaw or pitch. At last decide the new direction vector.

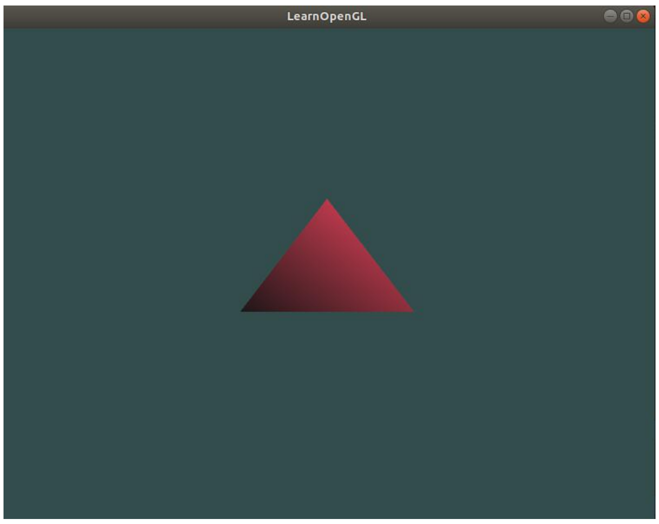
3. Result



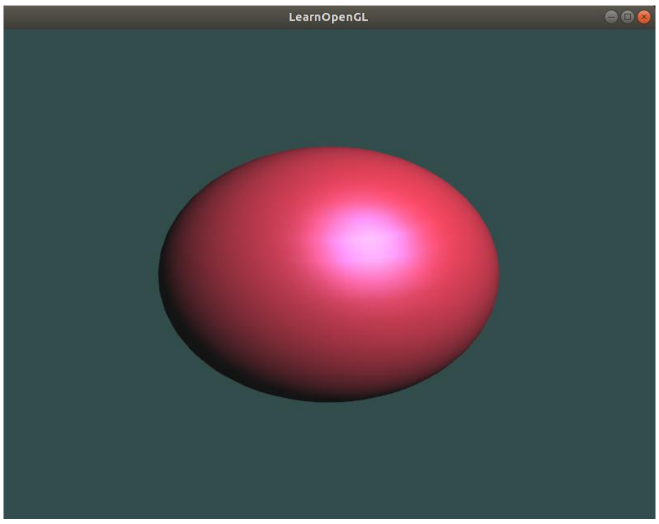
Disable the multi-sample



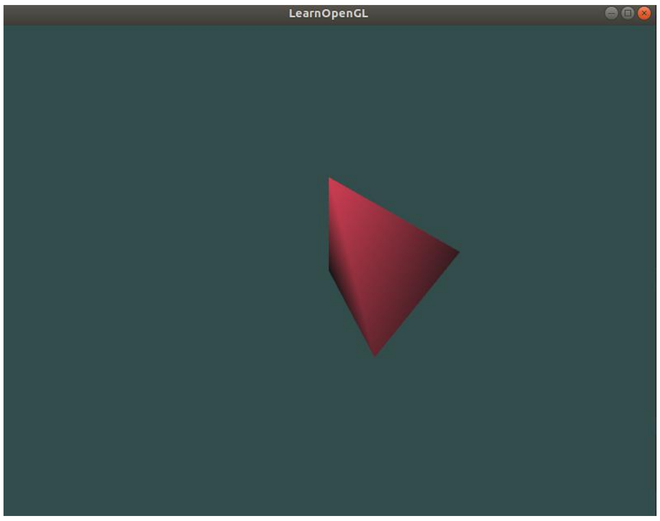
Enable the multi-sample



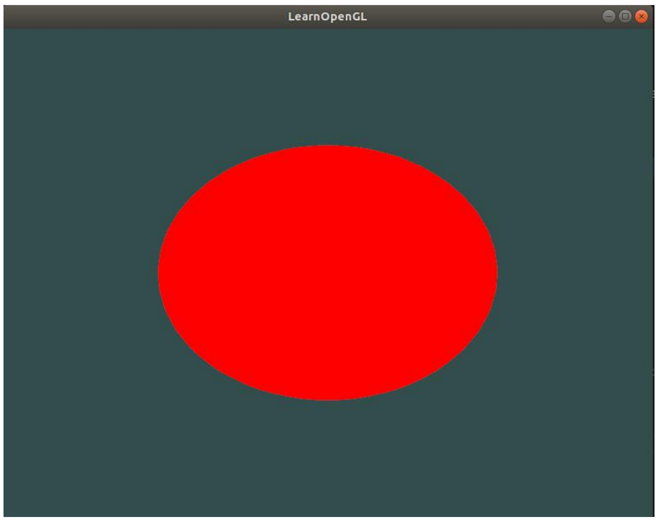
Triangle after rendering



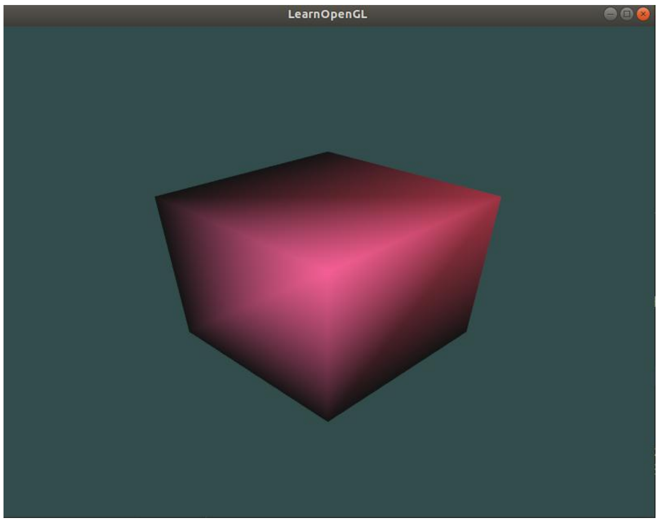
Shpere after rendering



Tetrahedron after rendering



Shpere without rendering



Cube after rendering

As for manipulating camera, we have shown in the video.