Warm-up assignment:
Programming simple graphics program with OpenGL

NAME: HAIZHAO DAI
STUDENT NUMBER: 2019533084
EMAIL: DAIHZH@SHANGHAITECH.EDU.CN

1 INTRODUCTION

In this project, we will (1) load mesh objects from files and draw the meshes; (2) render objects with a Phong lighting model; (3) manipulate a camera and control it.

2 IMPLEMENTATION DETAILS

Here I will show how to make a bunny model step by step.

2.1 Load mesh object and draw it

2.1.1 Load mesh data. First step is to load the mesh data from the file. The format of file is given so I will not mention about that. Notice that between each part of the data, there is one blank line. Using 'std::getline' to handle it. Also, all the data type ('float', 'GLuint', 'Vertex') are plain old data, which indicates that allocate space for 'std::vector' first is a preferable and fast way.

2.1.2 Draw mesh. Following with LearnOpenGL's tutorial, it is easy to draw mesh simply by creating vertex array objects(VAO), vertex buffer objects(VBO) and putting data into OpenGL render pipeline. As shown in fig.1, with simple vertex shader and fragment shader, I placed the bunny on the middle of the screen with proper size and used its vertex normals to generate color. The result is shown later.

2.2 Render objects with a Phong lighting model

Phong lighting model consists of 3 kinds of lighting: ambient, diffuse, specular. I decomposed and implemented them one by one. This part of implementation is done in fragment shaders.

2.2.1 Ambient lighting. A simple model of global illumination, which imitates weak environment lighting. So just a weak ambient strength is enough. Ambient lighting is view-independent and light-sources-independent.

2.2.2 Diffuse lighting. Diffuse lighting creates a rough face for object just by assuming reflectance is isotropic. In calculation, we need to let rays react with face normal so that we can get the actuall diffuse strength at that point. Diffuse lighting is view-independent and light-sources-dependent.

2.2.3 Specular lighting. Specular lighting creates a metal-like face for object by calculating the similarity of the direction of incident rays and reflected rays. Specular lighting is view-dependent and light-sources-dependent.

As shown in fig.2, the bunny is colorful now.

2.3 Camera and controlling

Camera is related to transformation matrices. Use encapsulated funtion of OpenGL, we are able to control the scene just we control the camera. My camera can walk and run 6 directions, zooming, and rotating when dragging with mouse.

As shown in fig.3, the bunny turns back.

3 RESULTS

3.1 Load mesh object and draw it

First created bunny, its color is defined with one minus its vertex normal.

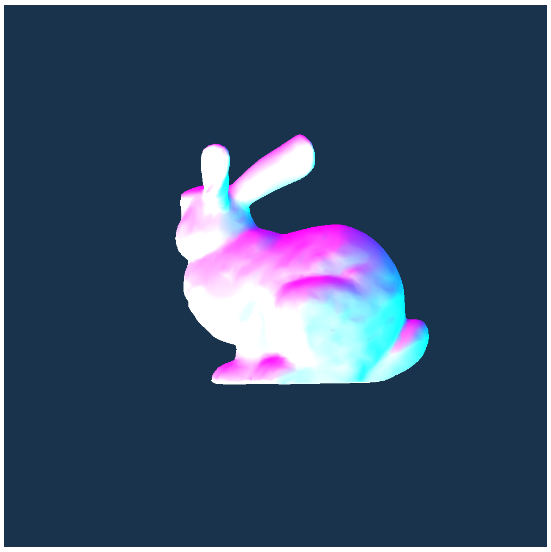

Fig. 1. Bunny1

3.2 Render objects with a Phong lighting model

I created 3 light sources with red, green, and blue color. And they rotate constantly on XY, XZ, and YZ planes.

© 2021
Computer Graphics I Course Assignment

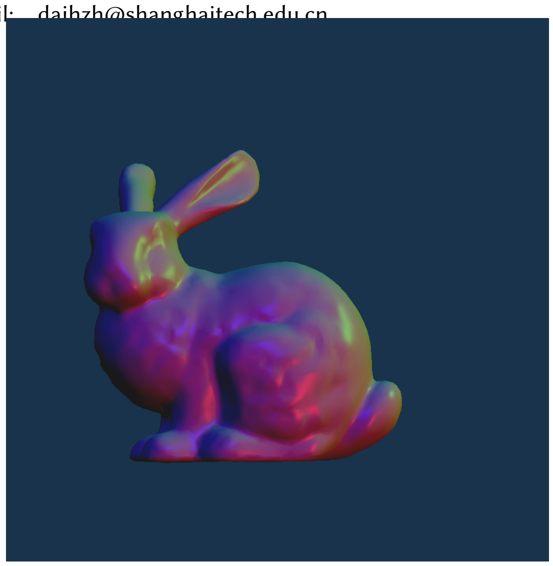

Fig. 2. Bunny2

3.3 Camera and controlling

Another view of Bunny.

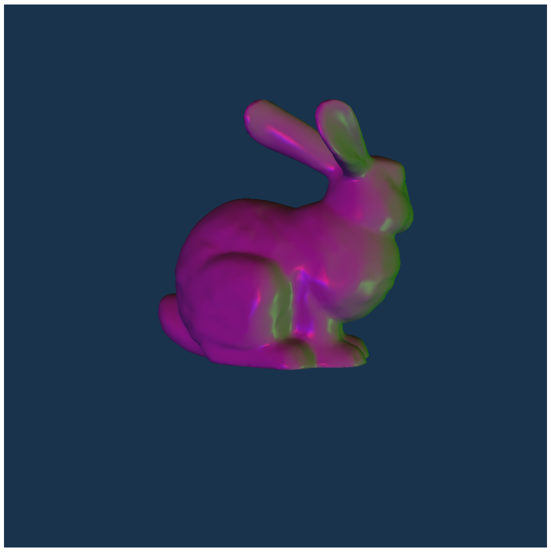

Fig. 3. Bunny3