计算机图形学

（Computer Graphics）

实验题目：OpenGL 中的建模和视图显示功能

目录：

实验报告：

1. 实验内容描述

本次实验使用的模型是一个音乐盒的旋转轴，是本人在大一课程《工程制图》中使用 solidworks 建立的模型。它的特点是整体是一个中间细，两头粗的圆柱，头部是做了锯齿状的齿用于辅助联动，头部中心有突出的一根细圆柱作为旋转的轴，尾部中心有个圆台型凹陷用于辅助固定。

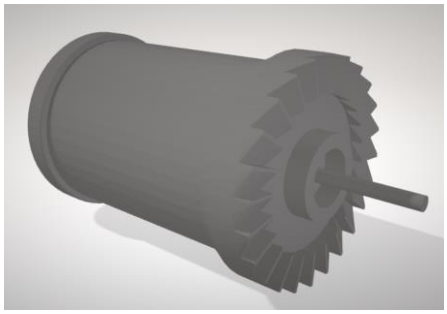

图 1 模型的 obj 文件打开结果（前侧视图）

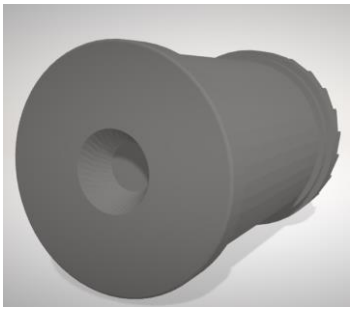

图 2 后侧视图

2. 实验功能算法描述，即模型是如何建立、如何载入、如何观察的

模型建立是通过使用 solidworks 进行 3D 建模建立的，然后将模型使用 ScanTo3D 插件，另存储为.obj 类型的文件。

模型载入是使用了自己实现的一个 obj 文件（仅含顶点数据 v 和面数据 f）的载入器类，通过文件读写，将所有的顶点数据 v 读入一个数组，将所有的面的顶点索引数据读入到另一个数组，然后在主程序中使用类中的 getVertexBuffer 方法和 getIndexBuffer 方法将数据绑定到顶点缓冲器和索引缓冲器中，然后进行索引绘制。

```
struct Vertex
{
    Vector3f m_pos;
    //Vector2f m_tex;

    Vertex() {}

    /* ... */
    Vertex(Vector3f pos) { ... }
};

class ObjLoader
{
private:
    vector<Vertex> VertexBuffer;
    vector<unsigned int> IndexBuffer;
public:
    void load(const char* filename);
    vector<Vertex> getVertexBuffer();
    vector<unsigned int> getIndexBuffer();
};
```

图 3 ObjLoader 类和顶点结构体

关于模型的观察，使用了一个 Camera 类，通过设置 Camera 类中的三个向量(position,tartget,up)，相机坐标系的第三个向量(right)可以通过 target 向量和 up 向量做叉乘得到。有了这三个向量之后，就可以把世界坐标系中的物体投影到相机坐标系，利用的矩阵计算如下，其中 U 为 right 向量，V 为 up 向量，N 为 target 向量

$$\begin{bmatrix} U_x & U_y & U_z & 0 \\ V_x & V_y & V_z & 0 \\ N_x & N_y & N_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_{world} \\ Y_{world} \\ Z_{world} \\ 1 \end{bmatrix} = \begin{bmatrix} X_{camera} \\ Y_{camera} \\ Z_{camera} \\ 1 \end{bmatrix}$$

最后，为了让模型看上去更加真实，我在程序后指定了隐藏一些本来看不到的面。

```
//在渲染中把正常视角看不到的面隐藏
glFrontFace(GL_CW);
glCullFace(GL_BACK);
glEnable(GL_CULL_FACE);
```

图 4 隐藏正常视角看不到的面

3. 实验 shader 程序描述，即 vertex shader 和 fragment shader 的程序代码及说明

shader.fs

```
#version 330

in vec4 Color;

out vec4 FragColor;

void main()
{
    FragColor = Color;
}
```

图 5 片元着色器

shader.vs

```
#version 330

layout(location = 0) in vec3 Position;

uniform mat4 gWVP;

out vec4 Color;

void main()
{
    gl_Position = gWVP * vec4(Position, 1.0);
    //TexCoord0 = TexCoord;
    Color = vec4(clamp(Position,0.0, 1.0), 1.0);
    //Color = vec4(0.0, 1.0, 0.0, 1.0);
}
```

图 6 顶点着色器

在顶点着色器(shader.vs)中，我们接收来自绑定的顶点数据，然后通过 uniform variables(统一变量)来接收来自我们camera类提供的相机坐标系的变换矩阵。然后进行坐标变换运算（矩阵与顶点数据相乘），另外我们额外制定一个输出变量 Color，这个主要是希望给各个不同的点赋予不一样的颜色，如果模型的所有的点都是同一种颜色，由于没有加入纹理，没有加入光照，不利于模型观察，因此，根据点的坐标 xyz，转换成 rgb 三个通道的值，这样可以做到各个位置的颜色同，方便观察。

在片元着色器(shader.fs)中，我们接收来自顶点着色器传过来的各点的颜色，然后只要将其输出即可。

4. 实验结果，要贴实验结果图

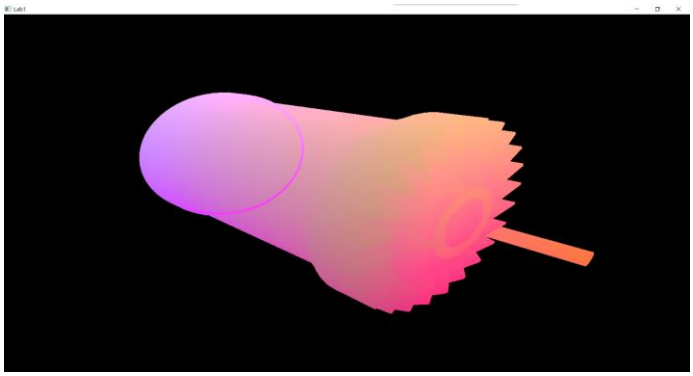

图 7 实验结果图（前侧视图）

图 8 实验结果图(侧后视图)

5. 心得体会

在本次实验中，实现了建模-》读入模型-》更改视角进行观察-》利用着色器渲染整个过程，了解了物体利用变换矩阵在世界坐标系中的变换，物体如何从世界坐标系投影到相机坐标系，了解 shader 应该如何应用到程序中，受益匪浅。期待后期对模型添加纹理和光照后的效果！