计算机图形学作业报告

【1】调用 OpenGL 库函数绘制一个球面

一、主要库：glut 库

二、主要结构/类

无

三、主要函数

a. main():主函数

b. init():初始化函数，主要用于整体的环境设置（包括光照和物体材质等）

c. myDisplay()：主要用于绘制球面

d. reshape(int w, int h)：主要用于保证拉伸窗口时模型不随着窗口变形而变形

四、实现思路

主要调用了 glut 库的函数 glutSolidSphere 来绘制一个球体。并且添加了光照与材质设置使得球体看上去更加真实。

五、操作

运行后即可看到绘制的球体

六、其他说明

暂无

【2】可改变视角从各个方向观察该模型

一、主要库：glut 库

二、主要结构/类

a. (struct)Poisition_3D:三维坐标点

```
{
    GLfloat x, y, z;
}
```

三、主要函数

void keyDown(unsigned char key, int x, int y)：处理键盘操作事件

void mouseMove(int x, int y)：处理鼠标移动事件

void mouseDown(int button, int state, int x, int y)：处理鼠标按下的事件

void idle()：变换观察点位置

四、实现思路

主要：通过观察点的变换使得观察者产生物体发生变换

详细：

1．缩放：计算出物体和观察者之间的位置的矢量，并沿着该矢量的方向对观察点进行移动。放大则距离减小，沿着正方向移动缩小则距离增加，沿着矢量的负方向；

2．平移：物体变化位置其实可以等效于观察者的位置以及灯光位置位置的沿着反方向的转换，也就是将负方向的位移添加到观察位置与灯光位置。

3．旋转：物体是直接在物体上旋转实现的，主要通过调用 glRotatef()函数实现

五、操作

运行后，使用左键拖拽可实现缩放效果，使用 w\s\a\d\z\x 对物体进行 3 个方向的平移，使用 q\e\r 对物体进行旋转。

六、其他说明

为了演示效果，本程序使用的是立方体进行测试

【3】写一个 Loader 读入三维模型

一、主要库：glut 库

二、主要结构/类

a. (struct) coordinate：三维坐标

```
{
    GLfloat x, y, z;
}
```

b. (struct) face：面索引以及面的属性

```
{
    int facenum;   //面的编号
    bool four;     //是否有四个点
    int faces[4];  //顶点索引
}
```

c. (class) ObjModel：存储模型所有数据的类

```
{
    std::string modelName;                  //模型的名字
    std::vector<coordinate*> vertex;  //存储的顶点的集合
    std::vector<face*> faces;               //存储的面的索引的集合
    std::vector<coordinate*> normals;       //存储的法向量的集合
    ……
    void ObjModel::Load();        //读取这个模型
    int ObjModel::Draw();                   // 绘制模型
    void ObjModel::Release();               // 释放模型所存内存空间
}
```

d. (struct)glutWindow:存储 glut 窗口的属性

```
{
    int width;      //窗口大小
    int height;
    char* title;    //窗口名
    ……
}
```

三、主要函数

void ObjModel::Load():读取模型

int ObjModel::Draw():绘制模型

四、实现思路

读取：从 obj 文件存储数据的格式出发，逐行读取数据。如果是开头是 v 并且后面是空格的话，就是一个顶点，则将其存储到新建一个顶点，将其地址存储到顶点向量中；如果是开头是 vn，就是一个法向量，同理，将新向量的地址存储到法向量的向量中；如果是开头是 f，就是一个面的数据，则先判断这个面有几个顶点，记录是一个三角面还是一个四边面，把新建的 face 结构的地址存储到面向量中。

绘制：遍历该模型存储的每一个面，判断其形状后再根据其中所存储的面索引调取相应位置的顶点和法向量，然后根据法向量和顶点的位置绘制出这个面。

五、操作

运行后即可看到读取的 obj 模型

六、其他说明

可以替换成其他模型，但是注意 obj 文件的格式：面的索引的格式必须以’//’分割，而不能用’/’或空格分割（正确的例子："f 443//401 319//401 311//401 313//401"）

三角形面和四边形面都是可行的。

【4】建立球面顶点与导入模型顶点之间的映射，实现球面向任意模型的变形过程动画

一、**主要库**：glut 库

二、主要结构/类

（同上）

三、主要函数

void ObjModel::copyFrom(ObjModel old)：从另一个模型复制其数据到这个模型

void ObjModel::unitVertex()：将模型的所有顶点单位化（即是模变为 1）

void ObjModel::trans(ObjModel start, ObjModel end, int current, int total)：变化模型

coordinate calculateNormalTRI(coordinate* coord1, coordinate* coord2, coordinate* coord3):计算法向量

GLfloat calmid(GLfloat a, GLfloat b, int per, int total)：计算中间值

void unitCoord(coordinate* coord):单位化坐标

四、实现思路

核心思路是将物体的顶点单位化，使得其可以全部落在球面上；然后只需要将这些顶点再逐帧移动到原来的位置。

根据以上思路，新建了两个 ObjModel 的变量。一个用于存储单位化后模型的信息，另外一个用于存储变化中模型的信息。而在变化的时候由于面所存储的向量是不会改变而顶点位置会发生变化，所以其面的法向量会产生变化，所以在每次变化顶点的时候必须重新计算每一次的法向量。

五、操作

鼠标左键拖曳缩放；鼠标右键拖曳旋转；w/s/a/d 点击向四个方向移动

模型变换：运行后按下空格键开始。

六、其他说明

这里提供交互旋转操作的时候提供了另一种旋转物体的方法使得可以使用鼠标拖

曳达到旋转的效果，但是由于是用球坐标实现的，所以不能旋转太大角度（>180）。

同时，由于是顶点一对一映射，所以当顶点过少的时候可能不能达到比较理想的球面效果。