图形学第二次作业报告

软73 沈冠霖 2017013569

1.使用方法

运行环境：

- Windows10
- VS2017
- OpenGL 4.6.0
- GLU工具库 1.2.2.0

运行方法：

- 打开exe文件即可运行，查看场景
- 按下鼠标并拖动可以切换视角，鼠标左右移动实现视角旋转，上下移动实现视角上下平移
- 点击键盘的WASD/上下左右可以平移视角
- 查看控制台可以看到碰撞情况，每次碰撞都会在控制台打印碰撞信息

2.模块介绍

2.1.物体显示

我们总共显示三类物体：动态物体（2个球），静态物体（下方为梯形台，中间为长方形柱子，上方为球），边界（一个四边形）。我们的显示方法较为简单，先在初始化阶段存储物体等位置信息和材质信息，然后每次显示时先进行位置移动和碰撞处理，然后根据物体的材质信息设置当前opengl的材质信息，然后根据物体的位置信息，用opengl的多边形，球绘制函数绘制物体。

2.2.碰撞检测和处理

碰撞分为三类：

第一类是球和边界碰撞。检测只需要判断球心和边界是否距离小于半径，小于就是碰撞上了，否则没撞上。球和一个边界碰撞，只需要把对应方向速度取反即可。

第二类是球和球碰撞。这个检测只需要判断两个球心距离是否小于半径之和即可。速度更新的话，只需要径向速度交换，法向速度不变即可。

第三类是球和静态物体碰撞。因为静态物体较为复杂，我们采用其AABB包围盒进行碰撞检测和处理。检测方法是先用clamp方法求得包围盒表面和球心最近的点，之后判断这个点和球心等距离，小于半径就是碰撞了。

```
        //用clamping求AABB里离球最近的点
        float near_x = max(b.BoundingBoxDown.x, min(CurrentPlace.x,
b.BoundingBoxUp.x));
        float near_y = max(b.BoundingBoxDown.y, min(CurrentPlace.y,
b.BoundingBoxUp.y));
        float near_z = max(b.BoundingBoxDown.z, min(CurrentPlace.z,
b.BoundingBoxUp.z));
```

速度更新的话只需要判断这个最近点在哪几个平面上，然后把对应方向速度取反即可。

2.3.视角变换

我们通过绑定opengl的鼠标，键盘响应函数来实现响应用户输入，进行视角变换

```
    glutMouseFunc(OnMouseClick); //绑定鼠标点击函数
    glutMotionFunc(OnMouseMove); //绑定鼠标移动函数
    glutKeyboardFunc(OnKeyClick);//绑定键盘点击函数
    glutSpecialFunc(OnSpecialKeyClick);//绑定特殊键盘点击函数
```

我们实现了三种视角变换。

第一种变换是视角旋转，按下鼠标左右滑动可以实现XOZ平面的视角旋转，也就是修改相机所在的x，z坐标。

$$\theta = k_1 * t$$
$$x = rcos\theta, z = rsin\theta$$

第二种变换是高度变换，按下鼠标上下滑动可以实现y轴的高度变换，也就是修改相机所在等y坐标。

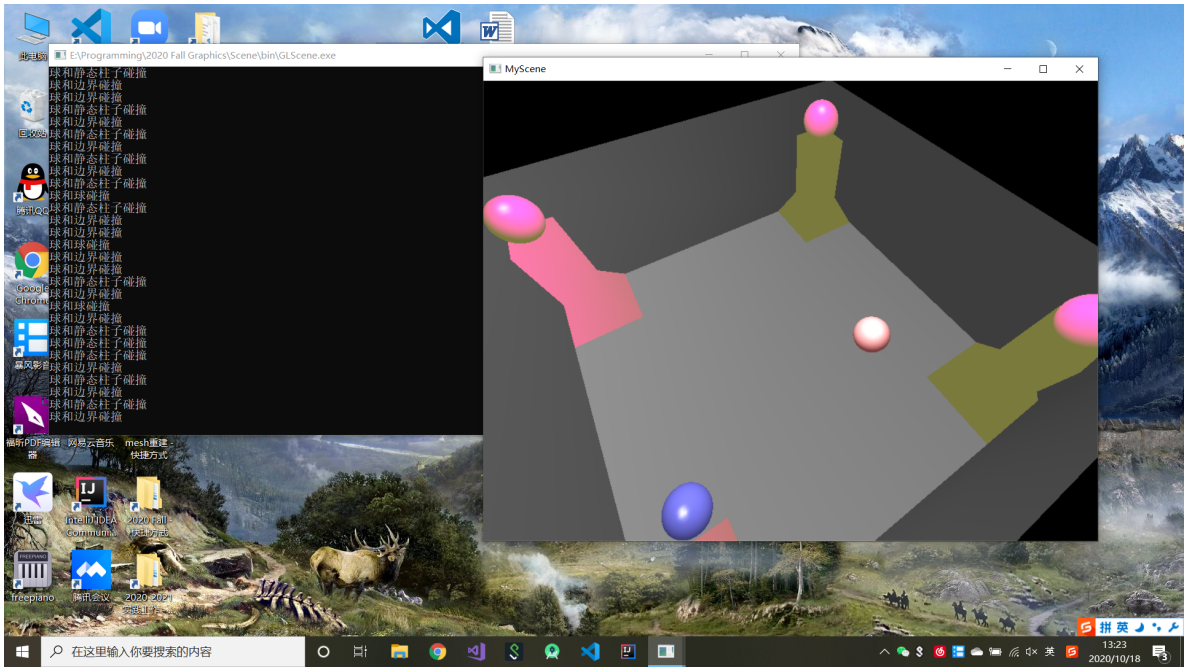
$$y = k2 * t$$

第三种变换是视角平移，按WASD/上下左右键能够实现视角平移，也就是同时修改相机位置和相机看向的点的x，z坐标。

$$LookAt = (k * t_x, 0, k * t_y)$$
$$x = k * t_x, z = k * t_z$$

3.结果

界面如图，右面的五个面是边界，四个柱子形状等物体是静态物体，两个球是动态物体。左面的控制台输出碰撞结果。



演示视频可以参见提交的视频文件

4.总结

这次实验，我了解了OpenGL的基本方法，包括物体绘制，光照和材质。也了解了图形学的视角，碰撞检测等方法，也体会到了图形学难以debug的特点。