计算机图形学报告 HW7

姓名：朱俊凯

学号：16340315

要求

Basic:

1. 实现方向光源的 Shadowing Mapping:

   要求场景中至少有一个 object 和一块平面(用于显示 shadow)

   光源的投影方式任选其一即可

   在报告里结合代码，解释 Shadowing Mapping 算法

2. 修改 GUI

Bonus:

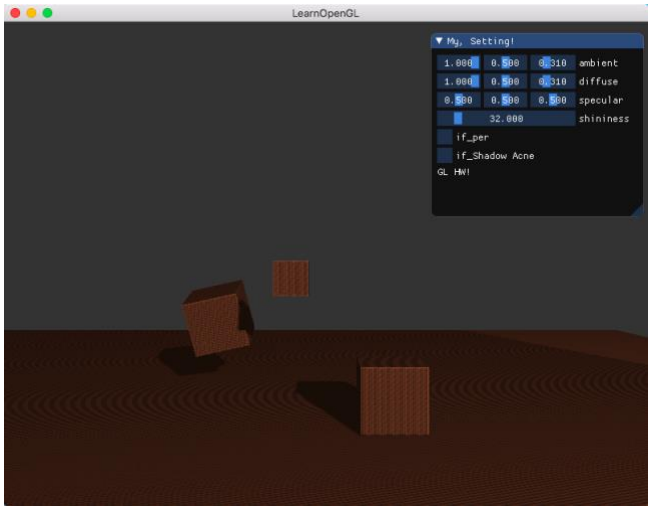
1. 实现光源在正交/透视两种投影下的 Shadowing Mapping
2. 优化 Shadowing Mapping (可结合 References 链接，或其他方法。优化方式越多越好，在报告里说明，有加分)

报告：

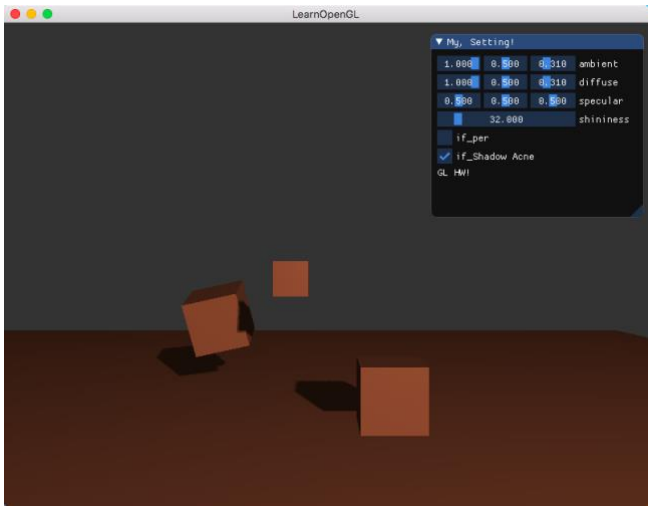
1. 把运行结果截图贴到报告里，并回答作业里提出的问题。

---gif 在文件夹内 ---

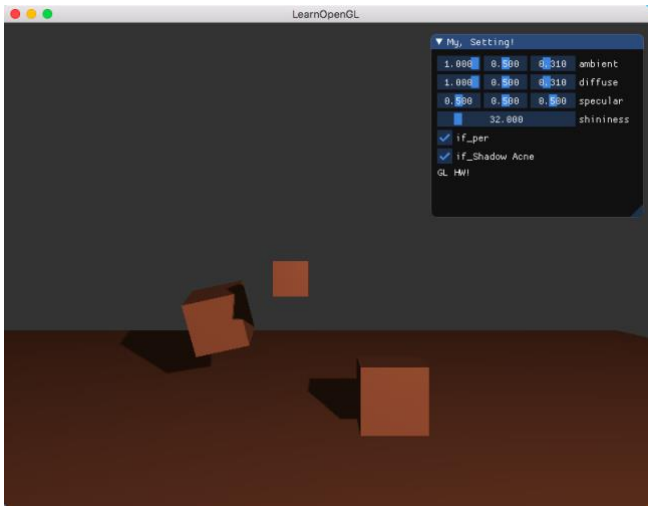
正交投影+没有阴影失真处理



正交投影+阴影失真处理



透视投影



解释 Shadowing Mapping 算法

基本上就是先在光源位置利用一次之前的相机渲染的方法，就可以获得相对于光源的各物体的深度信息，就可以获得这样的一个深度映射，然后通过这个深度来判断每个点是不是在相应像素离光源最近的点，及是否处在阴影之中，然后在正常渲染时根据这个信息，渲染出阴影即可。

阴影失真优化:阴影偏移

加入一个 bias，根据法向量和光源方向去计算一个值，和 0.005 取一个最大值，也就是可以让那些相邻的像素之间差值不操过这个差值，就让他们不存在遮挡关系。

悬浮问题：

由于之前的阴影偏移的方法会使得一些离立方体向光面附近的地板误判为相邻

的像素，不产生阴影，使得物体有悬浮感，但我使用了官网去除隐藏面的方法无法解决悬浮问题，可能是我个人问题。

PCF：

个人理解上，有点像卷积的感觉，利用周围 9 个像素来决定该像素的阴影值

2. 报告里简要说明实现思路，以及主要 function/algorithm 的解释。

1.首先为渲染的深度贴图创建一个帧缓冲对象 FBO

2.创建一个 2D 纹理，用于把渲染结果储存到这个纹理中

3.把这个纹理作为帧缓冲的深度缓冲

4.在光源的位置上运用之前相机的方法去渲染，就可以获得场景的深度，将深度值渲染到纹理的帧缓冲中

5.然后开始正常的场景渲染，将刚刚的深度纹理传入到着色器之中

6.根据这个深度纹理，提取判断每个像素是否在阴影之中。

7.根据深度再做一些阴影优化，得到合适的阴影信息。

8.渲染场景

正交/透视的区别：

1.在光源处分别使用正交/透视矩阵

2.在正常渲染时，需要对透视的使用时，将非线性深度值转变为线性的