计算机图形学课程设计报告书

数媒 1701 周展科 U201717132

一、选题

海军风格场景渲染

二、需求分析

使用 OpenGL API 和 GLSL 语言渲染海军风格的场景，需要实现模型导入、光照、纹理、透明物体、特效、交互等功能与效果。

二、设计与实现

整体场景设计：将物体放入天空盒中，天空盒选择傍晚夜空+平静湖面主题。物体组合中，最外面是一个旋转的透明球体，象征着保护罩，有星际战争的风格，保护罩里面有一个悬空的平台，其上有一架战斗机，战机里面有多个飞行员，长官则站在战机外面，等待指令下达。当玩家按下 o 键，保护罩闪烁并逐渐关闭，战斗机旋转 180° 后起飞，当战机飞离天空盒时，进入另外一个界面，显示”advanture begins”，表明探险就此开始。

具体的实现方法：

1．天空盒

绑定立方体贴图纹理，skybox 采样器就会自动填充上天空盒立方体贴图。绘制天空盒时，将它变为场景中的第一个渲染的物体，并且禁用深度写入。这样天空盒就会永远被绘制在其它物体的背后了。

2．模型导入

使用流行的模型导入库 Assimp 来加载模型数据，并使用 Model 和 Mesh 类来加载并使用导入后的模型。在后期修改模型的片元着色器，使之与带光源的场景融为一体。

3. 光照

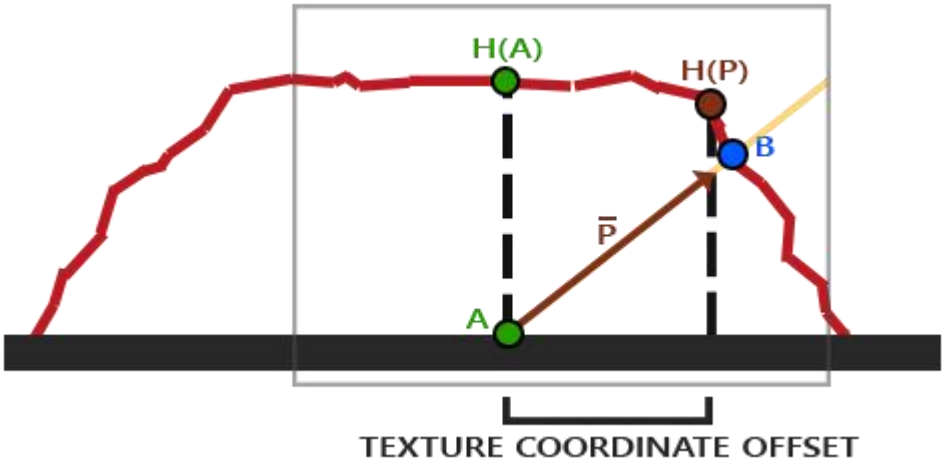
使用冯氏光照模型(Phong Lighting Model)绘制点光源，冯氏光照模型的主要结构由 3 个分量组成：环境(Ambient)、漫反射(Diffuse)和镜面(Specular)光照。循环渲染中将光源画出，并改变光源的位置，使之在水平面内旋转。

4.视差贴图

战斗机底下的平面采用视察贴图，会带来更好的效果。视差贴图技术和法线贴图差不多，能够极大提升表面细节，使之具有深度感。它也是利用了视错觉，然而对深度有着更好的表达，与法线贴图一起用能够产生难以置信的效果。

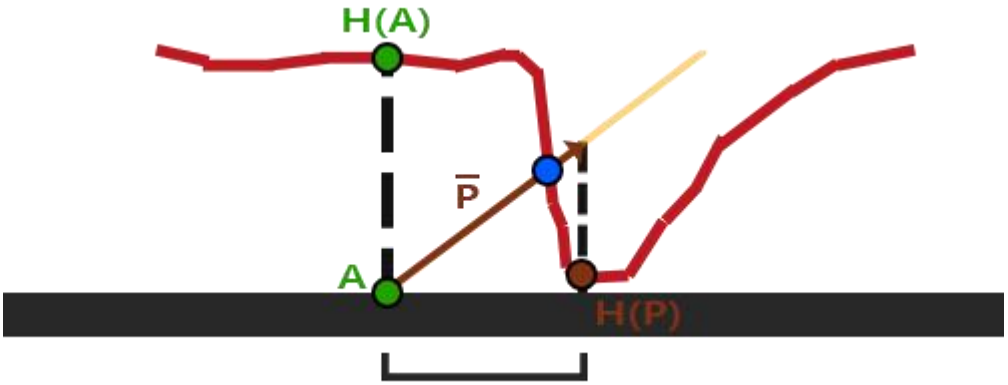
视差贴图背后的思想是修改纹理坐标使一个 fragment 的表面看起来比实际的更高或者更低，所有这些都根据观察方向和高度贴图。如果平面进行实际位移，观察者会在点 B 看到表面。然而我们的平面没有实际上进行位移，观察方向将在点 A 与平面接触。视差贴图的目的是，在 A 位置上的 fragment 不再使用点 A 的纹理坐标而是使用点 B 的。随后我们用点 B 的纹理坐标采样，观察者就像看到了点 B 一样。

这个技巧就是描述如何从点 A 得到点 B 的纹理坐标。视差贴图尝试通过对从 fragment 到观察者的方向向量 $\bar{V}$进行缩放的方式解决这个问题，缩放的大小是 A 处 fragment 的高度。所以我们将 $\bar{V}$的长度缩放为高度贴图在点 A 处 H(A)采样得来的值。下图展示了经缩放得到的向量 $\bar{P}$：



我们随后选出 $\bar{P}$以及这个向量与平面对齐的坐标作为纹理坐标的偏移量。这能工作是因为向量 $\bar{P}$是使用从高度贴图得到的高度值计算出来的，所以一个 fragment 的高度越高位移的量越大。

这个技巧在大多数时候都没问题，但点 B 是粗略估算得到的。当表面的高度变化很快的时候，看起来就不会真实，因为向量 $\bar{P}$ 最终不会和 B 接近，就像下图这样：



视差贴图的另一个问题是，当表面被任意旋转以后很难指出从 $\bar{P}$ 获取哪一个坐标。我们在视差贴图中使用了另一个坐标空间，这个空间 $\bar{P}$ 向量的 x 和 y 元素总是与纹理表面对齐。如果你看了法线贴图教程，你也许猜到了，我们实现它的方法，是的，我们还是在切线空间中实现视差贴图。

将 fragment 到观察者的向量 $\bar{V}$ 转换到切线空间中，经变换的 $\bar{P}$ 向量的 x 和 y 元素将于表面的切线和副切线向量对齐。由于切线和副切线向量与表面纹理坐标的方向相同，我们可以用 $\bar{P}$ 的 x 和 y 元素作为纹理坐标的偏移量，这样就不用考虑表面的方向了。

5.透明物体

在纹理中使用 alpha 通道，开启颜色混合和深度测试，将一张 png 格式的图片贴在球面上，使之拥有保护罩的效果。

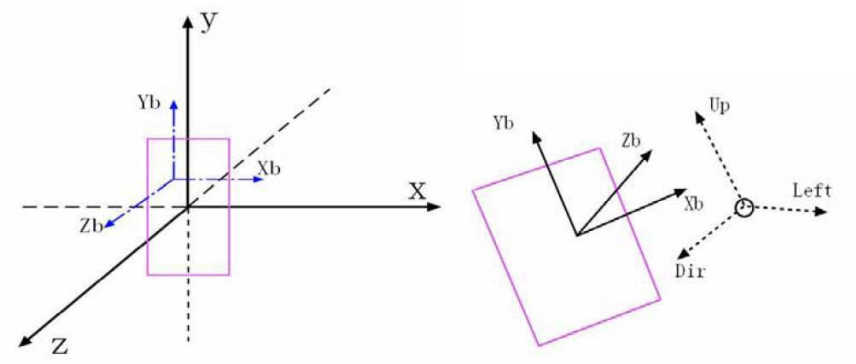
6.特效

场景中主要应用了三种特效，分别是视差贴图、公告板技术和文本渲染。视察贴图不再累述，下面细说公告板技术和文本渲染。

6.1 公告板技术(Billboard)

Billboard 为面对着摄影机的一种面片，在绘制 BillBoard 的时候，首先都假定 BillBoard 为一个面对着某个方向的简单的四边形。如图 1，红色的四边形为我们的 BillBoard 没有变换前的位置。在本文中，我把所有的 BillBoard 的朝向都定为+z。即[0,0,1]。因此，我们可以得到我们把图 1 蓝色的三个方向定义为 Billboard 局部坐标系的三个正交基，Xb = [1,0,0] ， Yb = [0,1,0]， Zb =

[0,0,1].可见，它和世界坐标系的基是一样的。



（图 1）　　（图 2）

现在我们来计算如何将它旋转到面对着摄影机。如图 2，红色的为我们的 BillBoard。有圆圈表示的是我们的 Camera，虚线为 Camera 的三个方向(均为单位化向量)。假设我们的变换矩阵为 M。因为，在这里我们只计算 Billboard 旋转矩阵。则在 BillBoard 坐标系中的点 P1 = [1,0,0] , P2= [0,1,0], P3=[0,0,1]. 经过 M 变换后在世界空间的坐标为:

$P1' = Left, P2' = Up, P3' = -Dir$,（注，因为右手系中，如 openGL，Dir 的方向和+Z 方向相反，而左手系的 D3D 则刚好相同，所以如果是左手系则为 $P1' = Left, P2' = Up, P3' = Dir$ ）。所以，很容易得到(L = Left , U = Up , D = Dir)：

$\begin{pmatrix}1,0,0\\0,1,0\\0,0,1\end{pmatrix} \times M = \begin{pmatrix}L.x,L.y,l.z\\U.x,U.y,U.z\\D.x,D.y,D.z\end{pmatrix}$,所以 $\begin{pmatrix}1,0,0\\0,1,0\\0,0,1\end{pmatrix} = \begin{pmatrix}L.x,L.y,l.z\\U.x,U.y,U.z\\D.x,D.y,D.z\end{pmatrix} \times M^{-1}$ ，因为 L,U,D 为三个正交单位向量，所以， $M^{-1} = \begin{pmatrix}L.x,L.y,l.z\\U.x,U.y,U.z\\D.x,D.y,D.z\end{pmatrix}^T$ 。

推导过 View 矩阵的人都看的出来，这个矩阵就是 View 矩阵的左上角 3x3 部分。

因为 Billboard 的 Local 坐标系为世界坐标系，所以 M 就是 Billboard 的旋转矩阵。即 M = Inverse(Rotation(ViewMatrix))，用这个矩阵，再加上平移矩阵，可以把一个 Billboard 从如图 1 的坐标中变换到观察坐标中。

6.2 文本渲染

使用一个非常受欢迎的跨平台字体库——FreeType，FreeType 所做的事就是加载 TrueType 字体并为每一个字形生成位图以及计算几个度量值(Metric)。我们可以提取出它生成的位图作为字形的纹理，并使用这些度量值定位字符的字形。

要加载一个字体，我们只需要初始化 FreeType 库，并且将这个字体加载为一个 FreeType 称之为 Face 的东西，我加载了字体文件 Expansiva.otf。

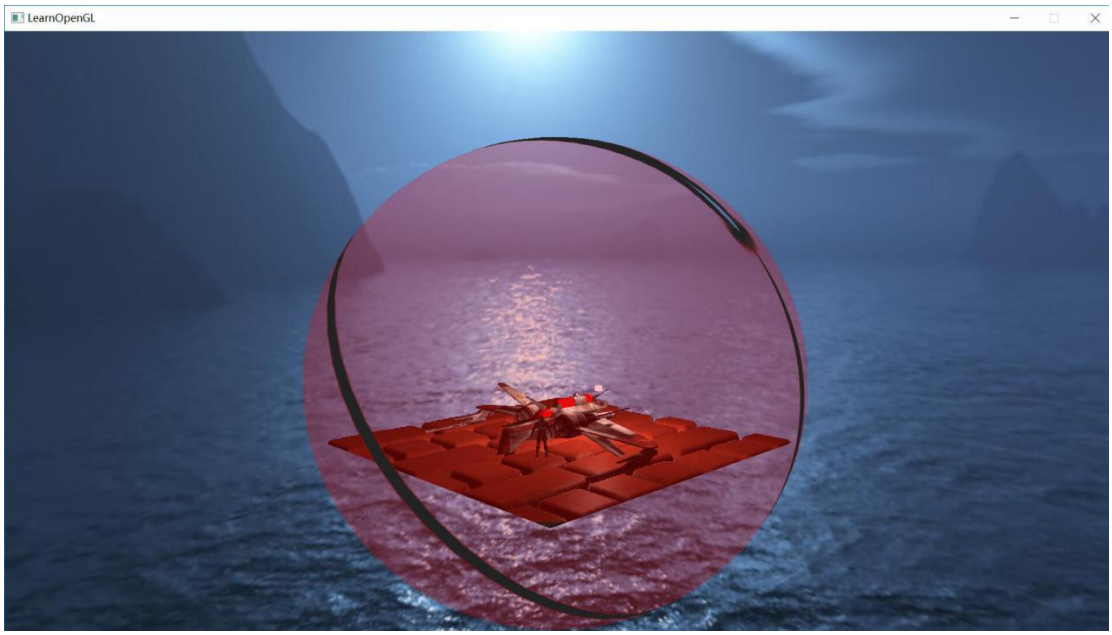
在需要渲染字符时，我们可以加载一个字符字形，获取它的度量值，并生成一个纹理，但每一帧都这样做会非常没有效率。我们应将这些生成的数据储存在程序的某一个地方，在需要渲染字符的时候再去调用。我们会定义一个非常方便的结构体，并将这些结构体存储在一个 map 中。

7.交互

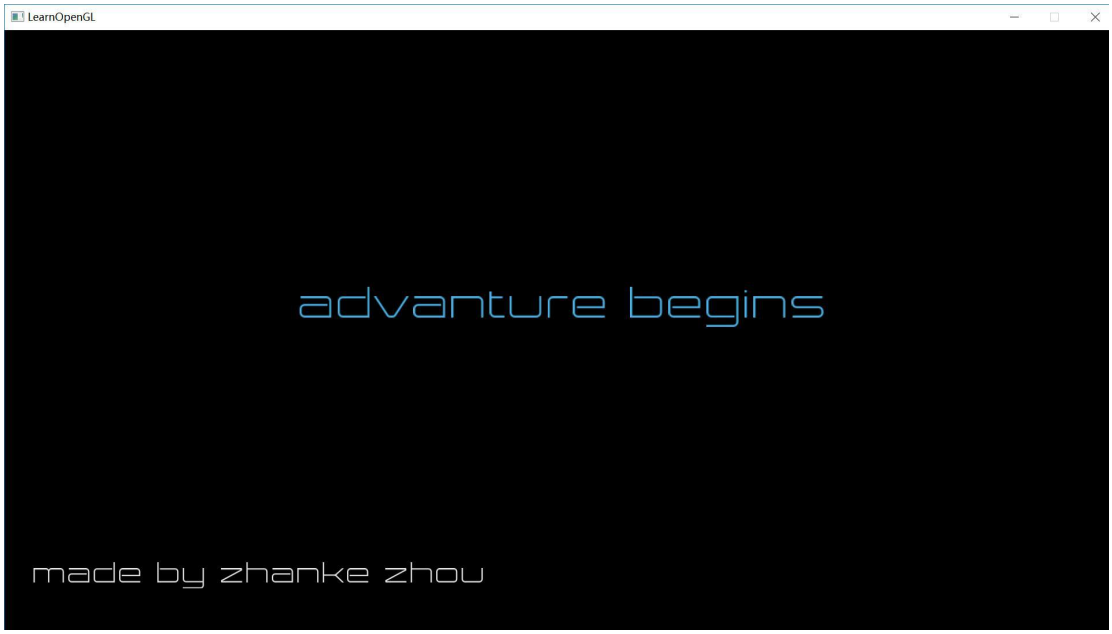
通过添加标记类全局变量，并重写键盘相应函数，当用户按下”o”键时，场景进入起飞模式，保护罩闪烁并逐渐关闭，战斗机旋转 180° 后起飞，当战机飞离天空盒时，进入另外一个界面，显示”advanture begins”，表明探险就此开始。

最终的效果图为：

效果图一.保护罩内部



效果图二. 保护罩整体与天空盒



效果图三. 结束界面

四、心得与体会

本次课程设计，我认为最大的难点在于设计一个整体感较强的场景，如果单凭物体的堆砌和特效的叠加，整个渲染的场景违和感会很强，与观众难以产生心灵上的共振。为了解决这个问题，我首先在草图上画出了大致的场景布局，设想所要达到的效果，然后在各个资源网站上寻找合适的模型和贴图，尽量挑选风格相近并符合场景主题的，接下来才是编码实现场景和调试。

在编码过程中我遇到了不少问题，一些是因为代码块之间存在污染和影响，一些是因为模型自身和硬件自身的固有问题，绝大部分问题都得以解决，这也给我留下了宝贵的调试经验。

总的来说，本次课设与设想的完成度比较高，收获颇丰。课程设计既是对已学知识的巩固与运用，又是对未知知识的探索与尝试，对我的编码能力与化理论为实际应用的能力有很大的促进与激励。

最后，非常感谢万琳老师及闫铠助教的解疑和指导。

References:

1. 《Billboard 的推导》潘李亮 2005-1-11
2. LearnOpenGL 中文官网： https://learnopengl-cn.github.io/