**练 习 题 报 告**

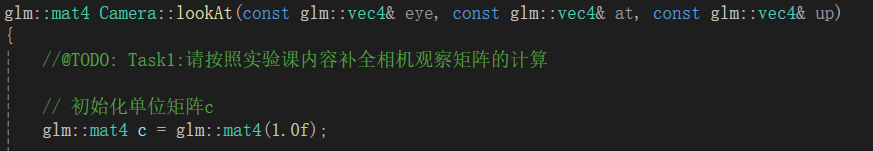
|  |
| --- |
| **课程名称 计算机图形学**  **项目名称 相机定位和投影**  **学 院 计算机与软件**  **专 业 计算机科学与技术**  **指导教师 熊卫丹**  **报 告 人 杜良衡 学号 2022150255** |

1. **练习目的**
2. 了解OpenGL中相机的模型视图变换的基本原理
3. 掌握OpenGL中相机观察变换矩阵的推导
4. 掌握在OpenGL中实现相机观察变换
5. 了解OpenGL中正交投影和透视投影变换
6. 了解在OpenGL中实现正交投影和透视投影变换。

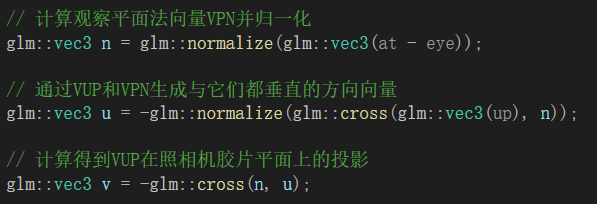
**二．练习完成过程及主要代码说明**

**Task-1：在Camera.cpp中完善lookAt函数和updateCamera函数**

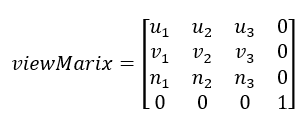
在lookAt函数中先初始化一个单位矩阵c作为相机观察矩阵

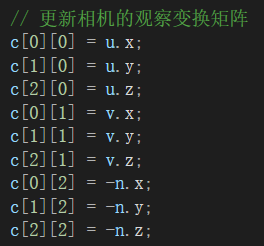
****

接着需要根据观察平面法向量VPN和观察正向向量VUP计算得到的3个向量n、u和v来更新相机的观察变换矩阵，也就是相机的局部坐标系。先计算出VPN再归一化得到向量n，再通过VUP和VPN生成与它们都垂直的方向向量u，最后计算得到VUP在照相机胶片平面上的投影v。这里的函数normalize和cross分别表示向量的归一化与叉乘。

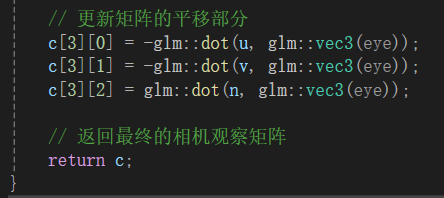
****

然后按照下图更新相机的观察矩阵c

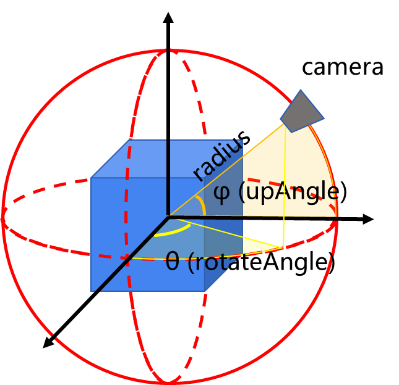
****

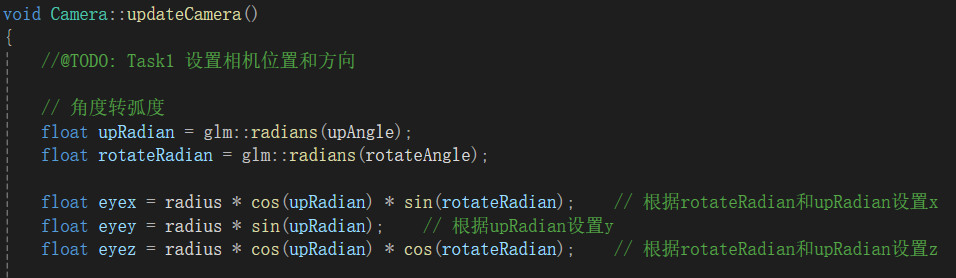
****

因为在最开始，需要将相机从坐标原点移动到视点，所以要更新观察矩阵的平移部分，这里通过将原始矩阵加上第四个维度，即齐次坐标的方式来实现。

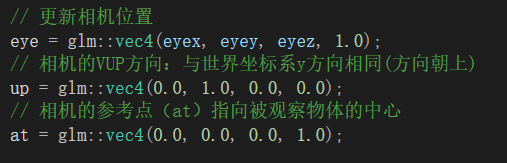
****

在updateCamera函数中根据角度与距离半径更新计算相机eye的位置，用于lookAt函数。先将角度upAngle和rotateAngle转为弧度，再根据下图利用三角函数与距离半径radius计算相机的x、y、z坐标。



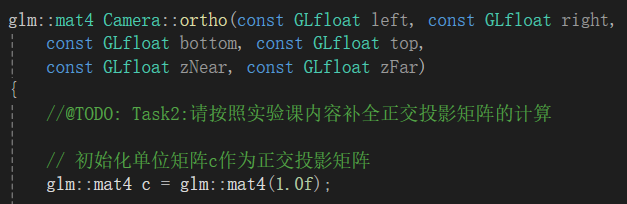
****

接着用计算得到的坐标更新相机位置，并设置相机的VUP方向与世界坐标系y方向相同(方向朝上)，相机的参考点（at）指向被观察物体的中心。这样就能保证相机围绕被观察物体旋转。

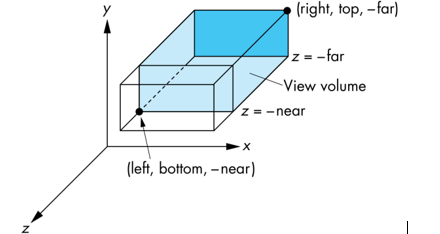


**Task-2：在Camera.cpp中完善ortho函数，在main.cpp中完善display\_3函数**

在计算正交投影的ortho函数中先初始化一个单位矩阵c作为正交投影矩阵。

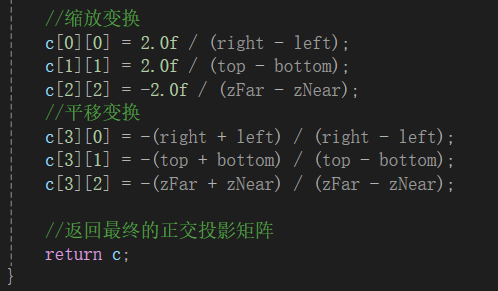


在OpenGL中使用的正交投影是定义在一个平行六面体的视景体中。如下图所示，该六面体由六个参数决定，分别为左右裁剪平面（left和right），上下裁剪平面（top和bottom），远近裁剪平面（near和far）。

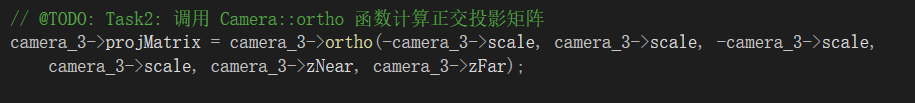


而在OpenGL渲染过程中，需要将定义的正交投影视景体通过平移和旋转变换到标准视景体中。因此，需要按照下图所示矩阵来更新正交投影矩阵c，并返回最终得到的矩阵。

****

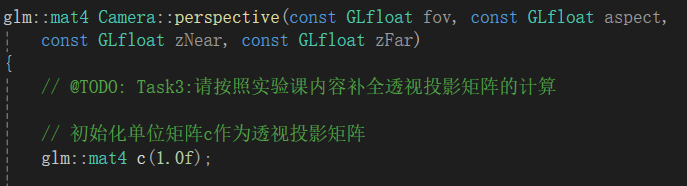


在main.cpp的display\_3函数中需要调用Camera::ortho函数来计算正交投影矩阵。其参数依次为左右边界、底边界、顶边界以及远近裁剪平面。

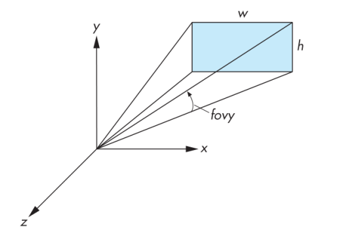


**Task-3：在Camera.cpp中完善perspective函数，在main.cpp中完善display\_4函数**

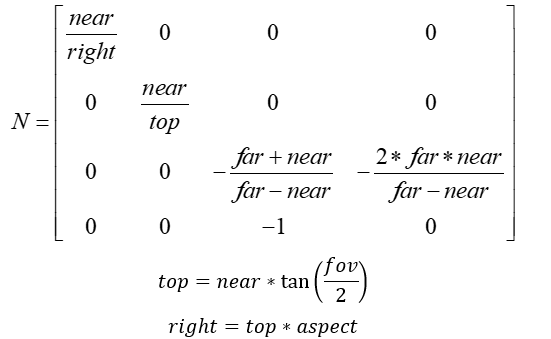
在计算透视投影的perspective函数中先初始化一个单位矩阵c作为透视投影矩阵。

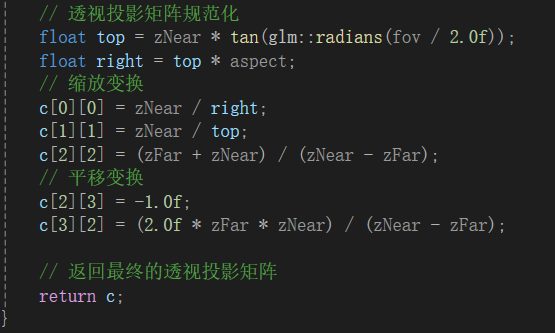


对于透视投影也需要设置视景体来裁剪三维物体。在OpenGl中可以通过视域（FoV，Field of View）来定义视景体，其由视角（Field of View），投影平面长宽比（aspect）和远近裁剪平面（near和far）决定。

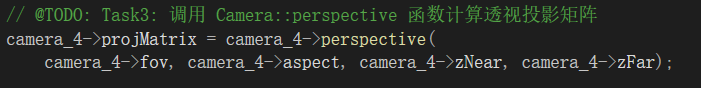


而透视投影同样也需要执行投影规范化过程。按照下图所示矩阵来更新透视投影矩阵c，并返回最终得到的矩阵。



****

在main.cpp的display\_4函数中调用Camera::perspective函数来计算透视投影矩阵。其参数依次为视角、投影平面长宽比以及远近裁剪平面。

****

**最终效果展示**

