三维变换实验程序简要说明

一．综述：

三维变换实验目的&目标:

1)掌握比如button,comboBox,picture control工具在交互界面中的应用，掌握窗口分割，新的Dialog的创建，新的toolbar创建的流程，熟悉各类消息映射函数（比如Onsize,OnCreate,OnBnClicked,OnUpdate）。

2)了解掌握空间模型（包括形体和背景）的构建原理与绘制，了解空间模型数据结构和变换参数的设计。

3)掌握空间变换的矩阵计算，了解照相机模型，三维光照模型，掌握空间变换，空间投影变换，空间裁剪，光照计算，图像消隐，明暗过渡等原理。

4)熟悉整个程序的组成与架构，了解各个函数的调用关系，熟悉整个程序的运行流程。

实验内容:

空间变换，投影变换，空间体的绘制，视见变换，裁剪，光照计算，消隐，明暗过渡等。

空间基本变换通过变换矩阵进行实现。对于三维的空间体来说，其每一个点都具有三个坐标，那么可以通过4\*4的矩阵与原向量相乘实现空间体在x,y,z三个方向的移动以及对x,y,z三轴的旋转

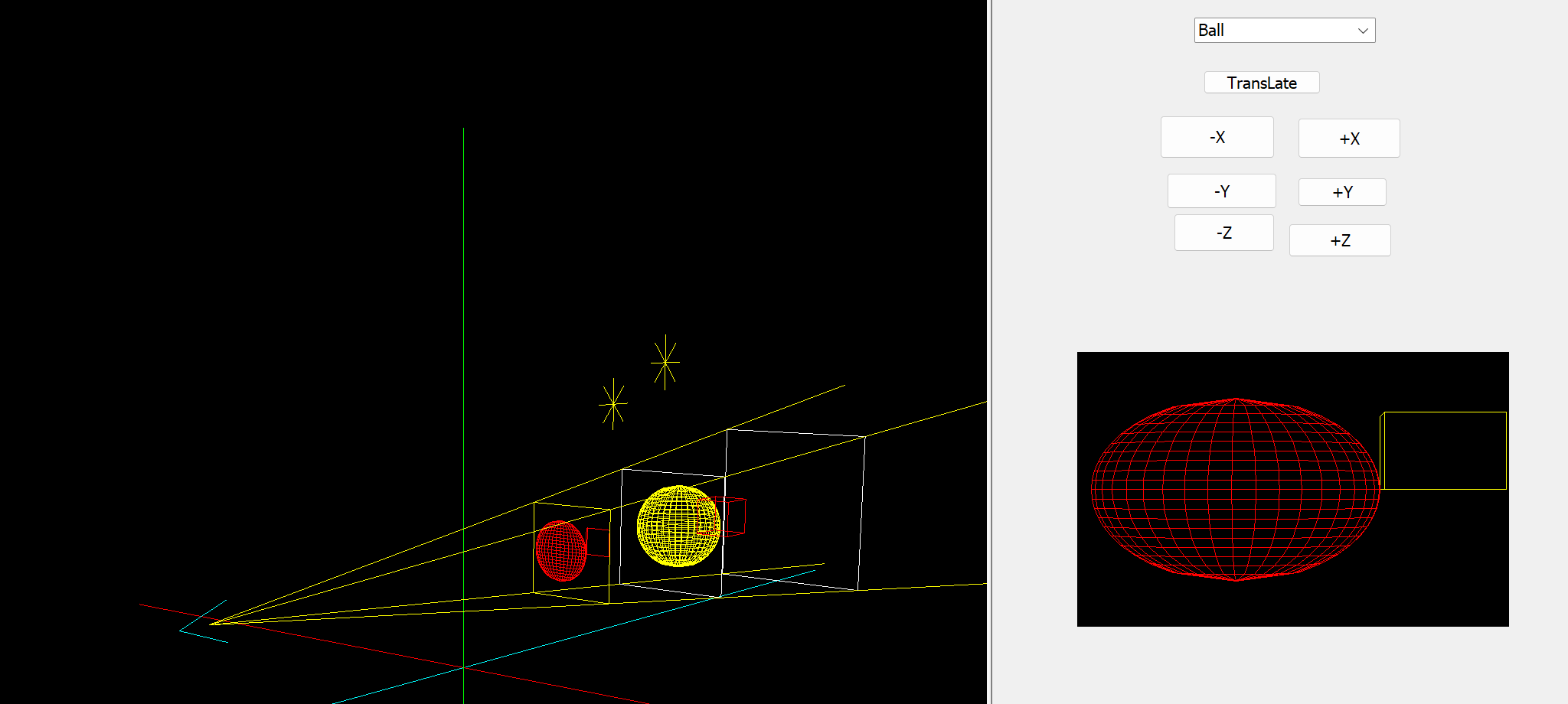
投影变换则根据视点的位置，将视点可见空间体的线与面投影下来。对于视点来说，其视角有限，因此要对空间体进行裁剪，将可见的部分投影到屏幕上。仅仅这样还是不够的，因为投影出来的效果不仅有可见的点和线，还有不可见的点和线，这样导致的效果就会有二义性，真实感不强，因此要引入消隐，在绘制空间体时要消除不可见的线或面，可以使用Z缓冲区算法进行消隐，即要改变某个像素的颜色时，要检测当前颜色的深度是否大于原来颜色的深度，如果大于，说明当前颜色更靠近视点，因此用当前颜色代替原来的颜色。简单光照模型有两种光源，一种是环境光，即周围环境对物体的光照，另一种是点光源直接照射到物体上。因为是简单光照模型，所以我们假定物体是不透明的，这样物体所表现的颜色就是其经光照反射出的颜色，其计算公式为：光照颜色 = (环境颜色 + 漫反射颜色 + 镜面反射颜色)\*衰减因子。而得到光照方程，就可以计算光照颜色，我们可以任取多边形上一点，利用光照方程计算出它的颜色，从而让用此颜色填充整个多边形，这样做的优点是每个多边形只需计算一次光照方程，速度快，但缺点也同时存在：1）相邻多边形颜色过渡不光滑2）曲面光照效果呈现不连续的光亮度跳跃变化，产生马赫带效应（光亮变化不连续的边界处出现过亮或过暗的条纹），针对这种情况，我们要引入明暗过渡算法进行处理。我们可以对于每一个多边形，计算其单位法向矢量（根据其平面方程），计算顶点的单位法向矢量（取包围该顶点的各多边形平面法向量的平均值），然后利用光照模型计算顶点颜色，在扫描线消隐算法中，对多边形颜色进行双线性插值，获得位于多边形内扫描线上各点的颜色，从而实现明暗过渡。

设计方案：通过变换矩阵实现空间体的移动和旋转，通过三维的Cohen-Sutherland算法实现多边形的裁剪，通过光照方程实现关照计算，通过Zbuffer算法实现消隐，通过Gouraud算法实现明暗过渡效果。

实现技术路线：使用Visual Studio2015集成开发环境，依托MFC框架，使用C++语言，借助OpenGL库进行绘图，利用多种算法，实现三维图形的多种变换。

二．程序交互页面设计/编辑

1.程序交互界面设计

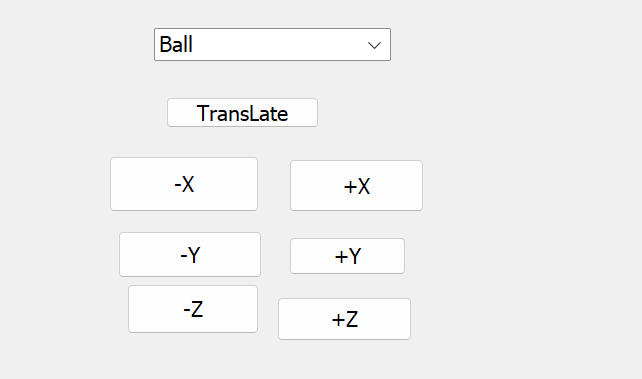


如图所示，程序交互界面由2两部分组成：三维空间场景显示区域和空间变换控制选择。

三维空间场景显示区域-对应cg3DtransView类，采用OpenGL库函数绘制。

空间变换交互控制选择-对应cgTaskControl类，包括：

1. 空间变换交互控制——ComboBox控件
2. 空间变换方式控制——Button控件
3. 空间变换方向选择——Button控件



2.程序交互页面编辑

项目资源中选择Dialog项，编辑生成空间变换交互控制选择。

1.1在资源视图里的DIALOG里面插入一个新的dialog,并修改其属性，style改为child，boder改为none

1.2添加一个CheckBox，给data选项添加内容作为程序运行后的待选项，修改caption和id

1.3添加控制方向和控制变换方式的button，修改它们的caption和id

1.4添加一个picturecontrol，作为投影的显示区域，修改caption和id。

1.5为这些控件添加消息映射函数

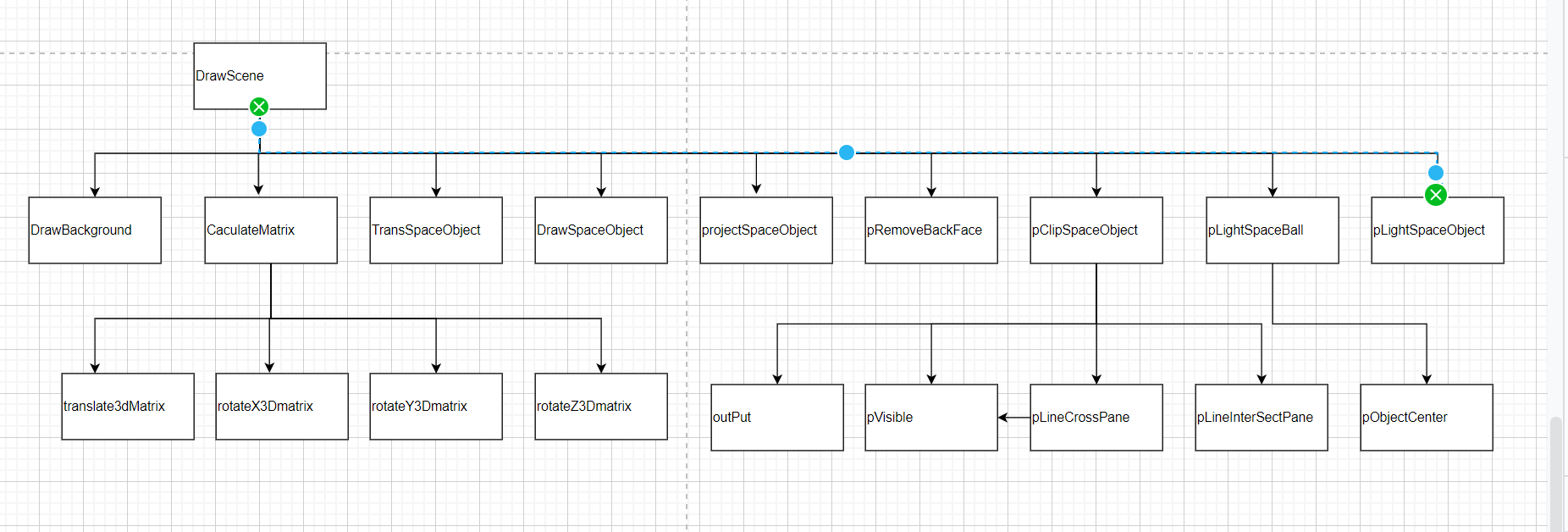
2.1在资源视图里点开Toolbar，增加光照，明暗过渡和动画的工具

2.2为它们增加消息映射函数。

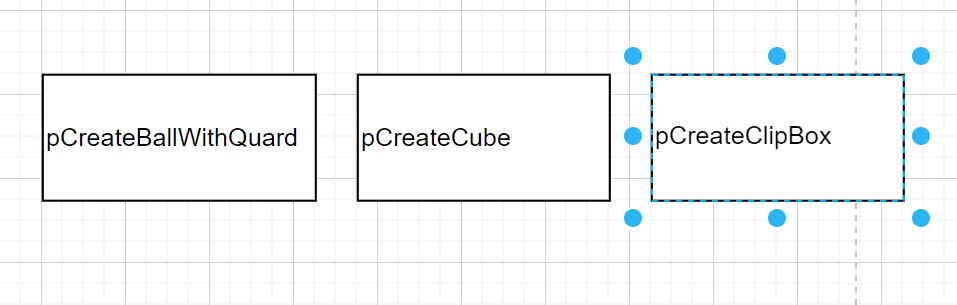
三．程序框架结构

三维变换程序由6部分组成，分别为cg3DtransApp,MainiFrm,cg3DtransDoc,cg3DtransVies,DrawScene,cgTaskControl.

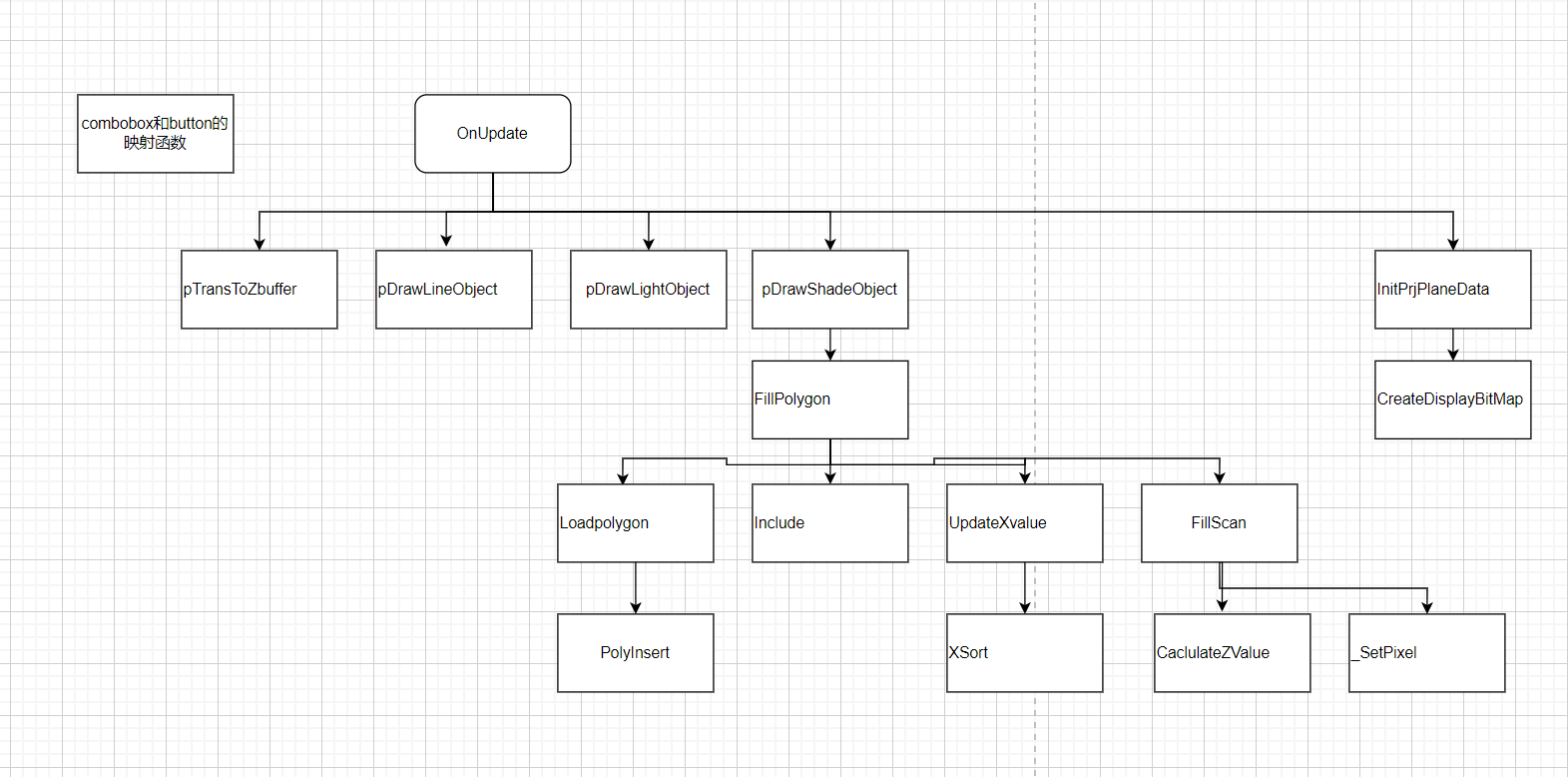
1. 功能具体描述:
2. MainFrm:创建程序的窗口，实现窗口的分割
3. DrawScene：实现背景的绘制，空间体的平移旋转变换，空间体的投影，裁剪以及光线照射。



1. Cg3DtransApp:控制着程序的进行和入口
2. Cg3DtransDoc:生成空间体和空间约束立方体的坐标

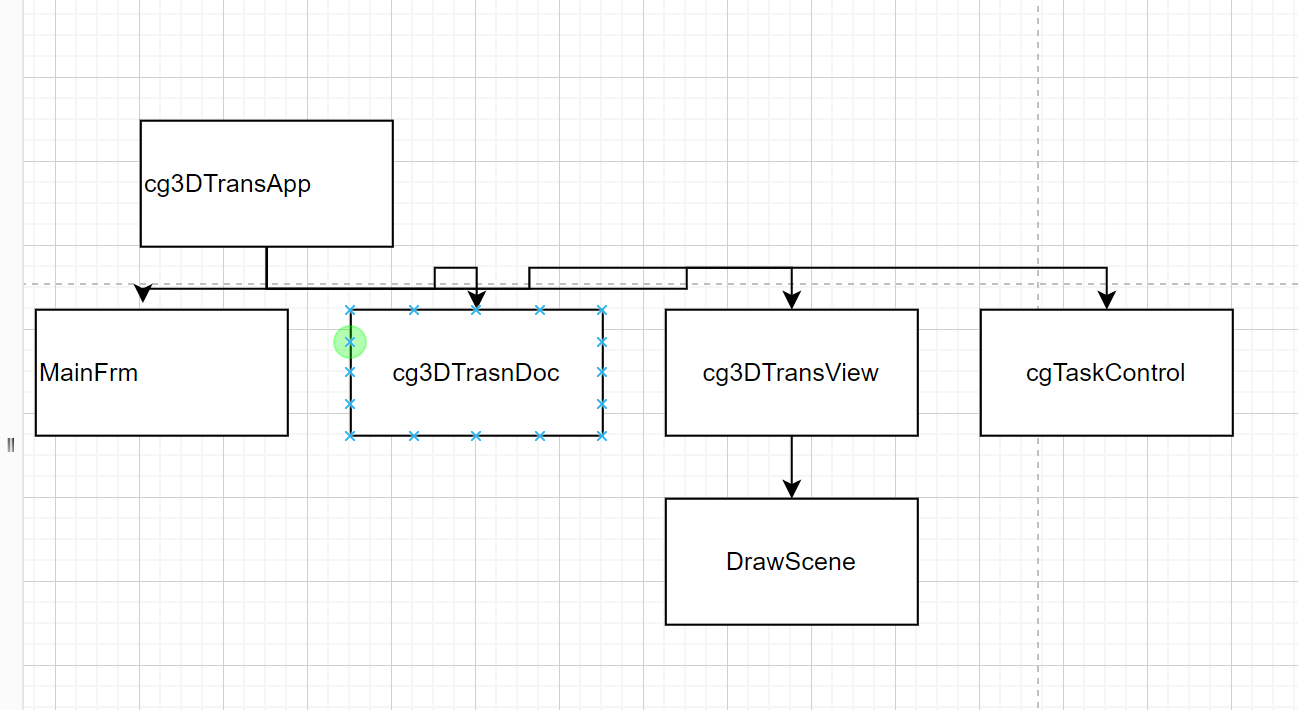


1. CgTaskControl:实现combobox和button的映射函数；实现消隐，图形绘制，打光，明暗过渡。



6）cg3DtransView:调用gl库实现绘图以及工具栏控件的映射函数

2.整体调用关系

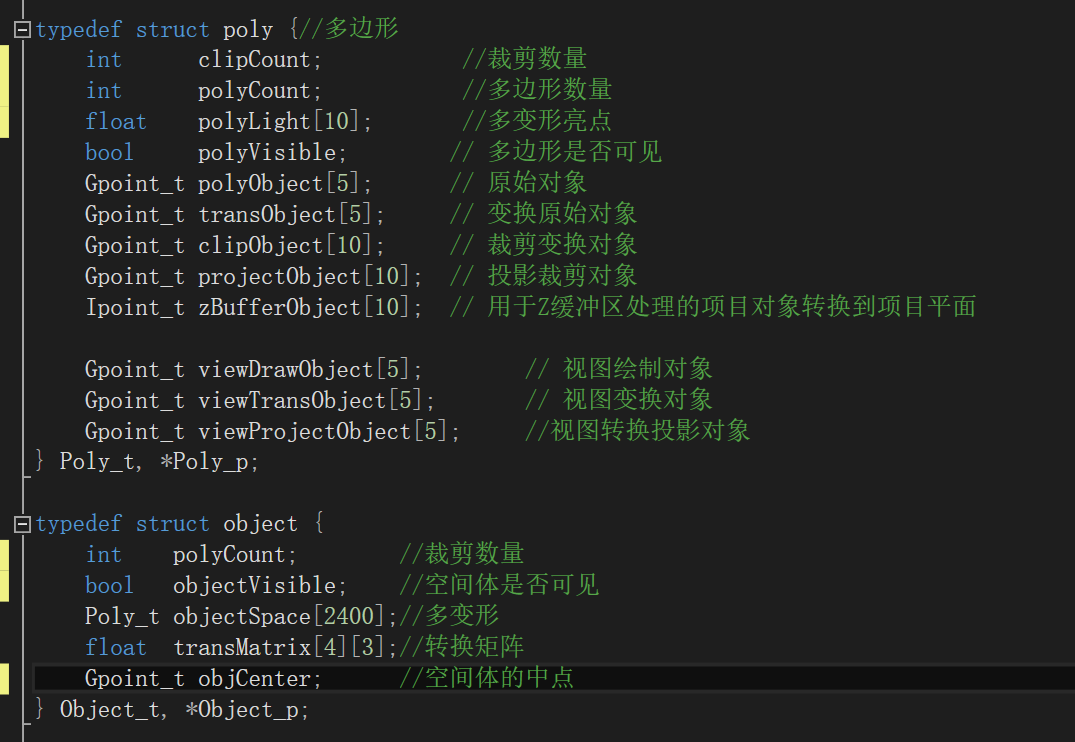


四．空间模型数据结构，变换参数设计说明

1.空间模型数据结构

TypeDefine.h定义描述空间模型数据结构

空间形体数据结构Object\_t，最多由2400个四边形组成。

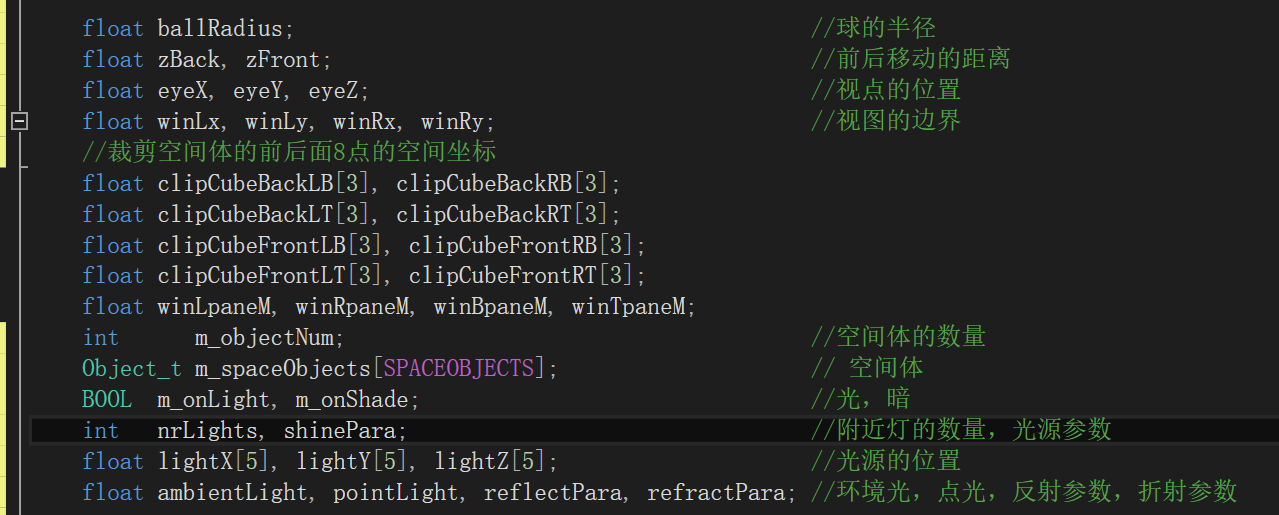


1. 空间变换参数

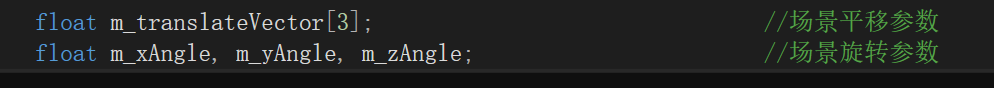
Cg3DtransDoc.h定义空间模型，OpenGL空间场景变换和空间变换交互控制选择参数

1)空间模型





2)OpenGL空间场景变换参数

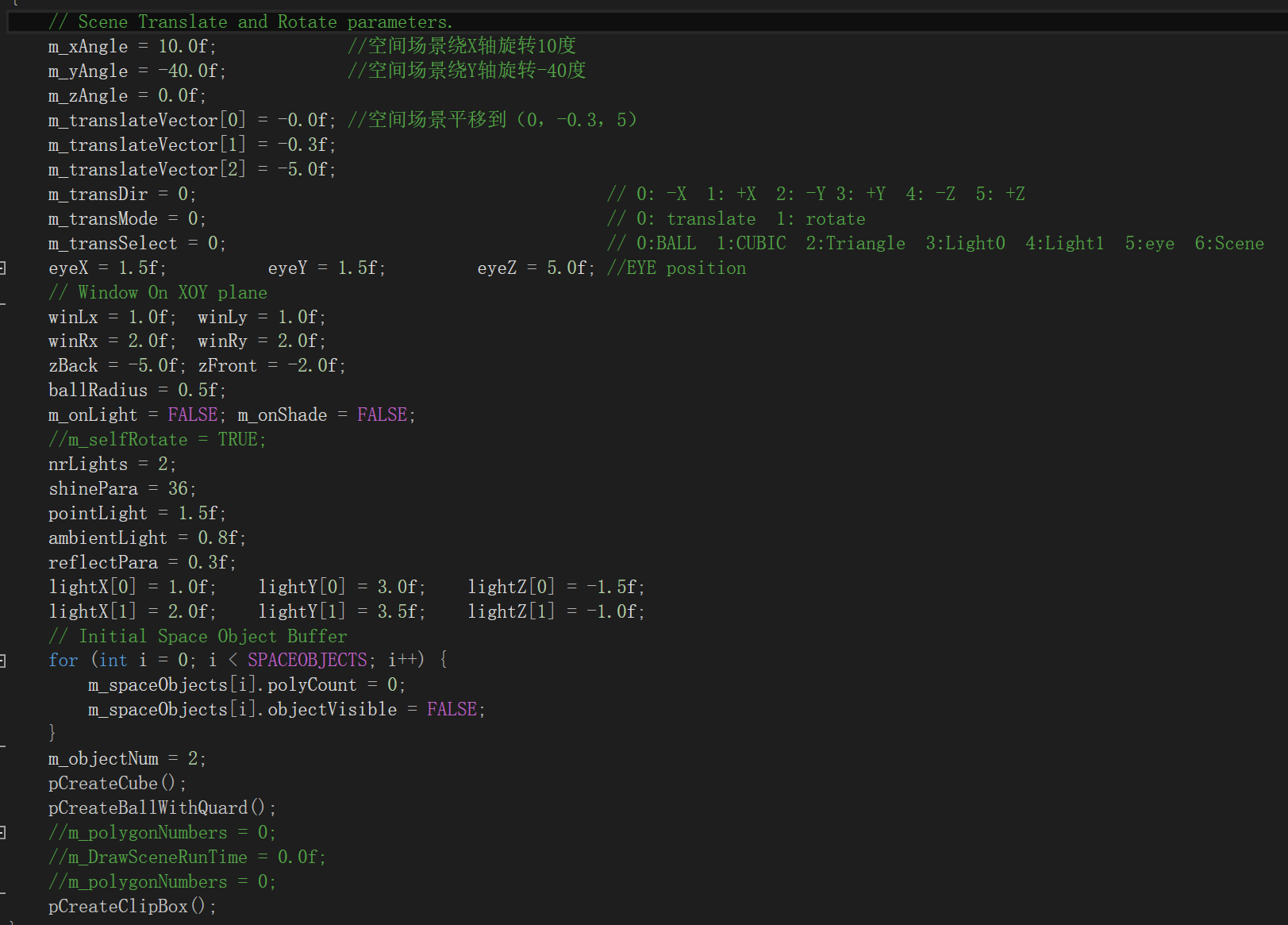


3）空间变换交互控制选择参数



1. 空间参数初始化

Cg3DtransDoc.cpp构造函数初始化空间变换参数

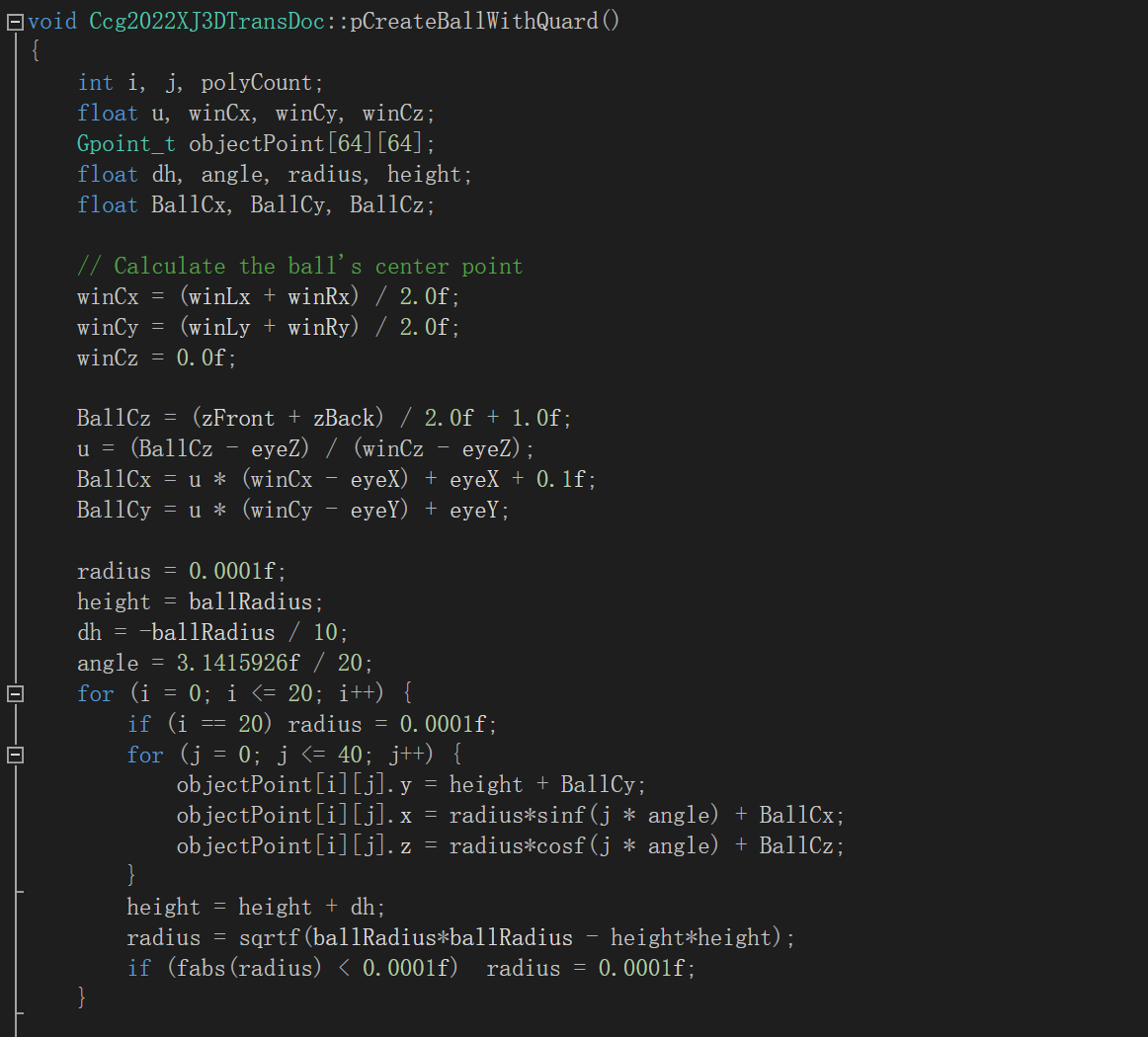


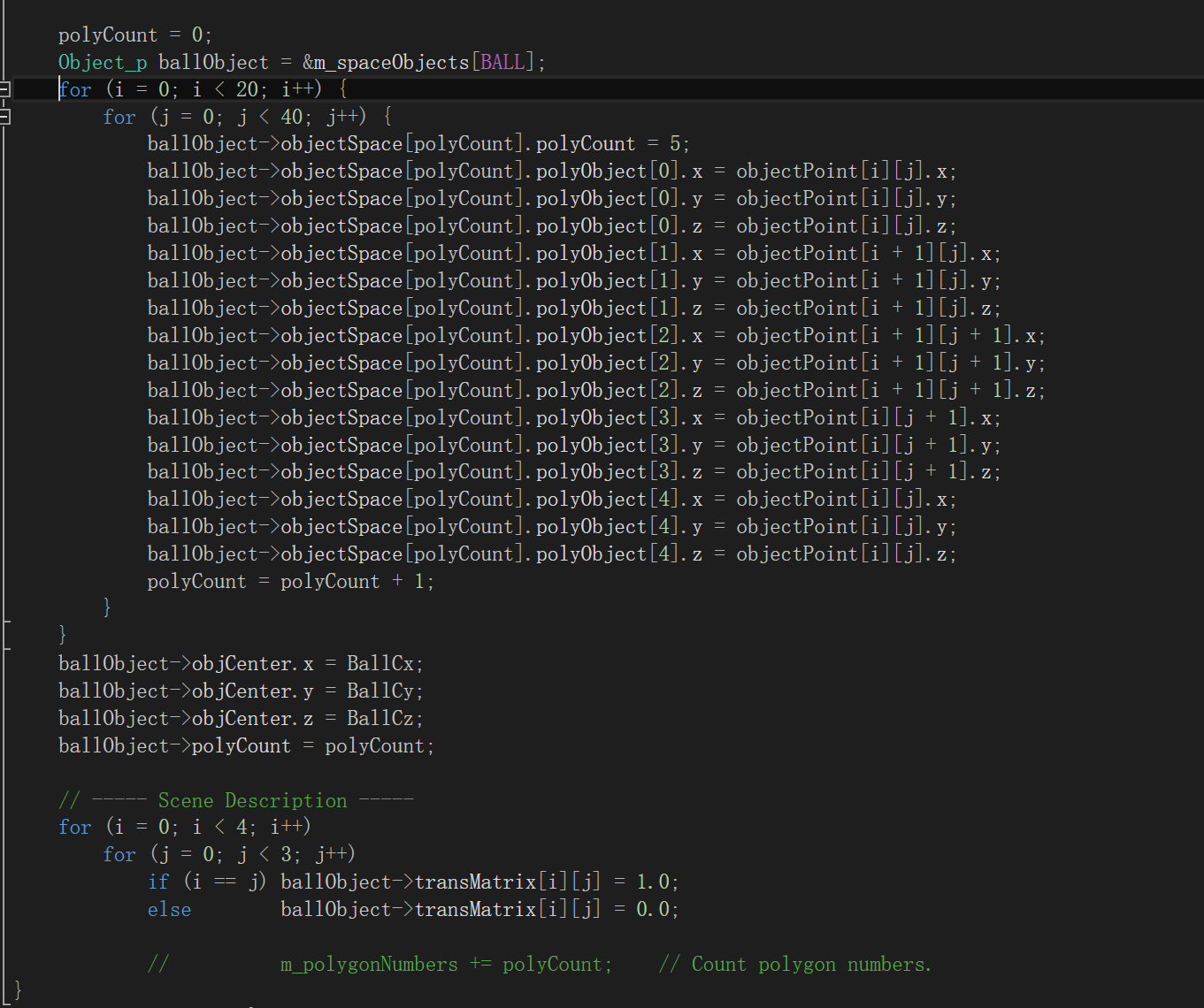
五．空间模型构建

空间模型由五部分组成：球，立方体，人眼，灯和裁剪约束体

1. 空间球面模型

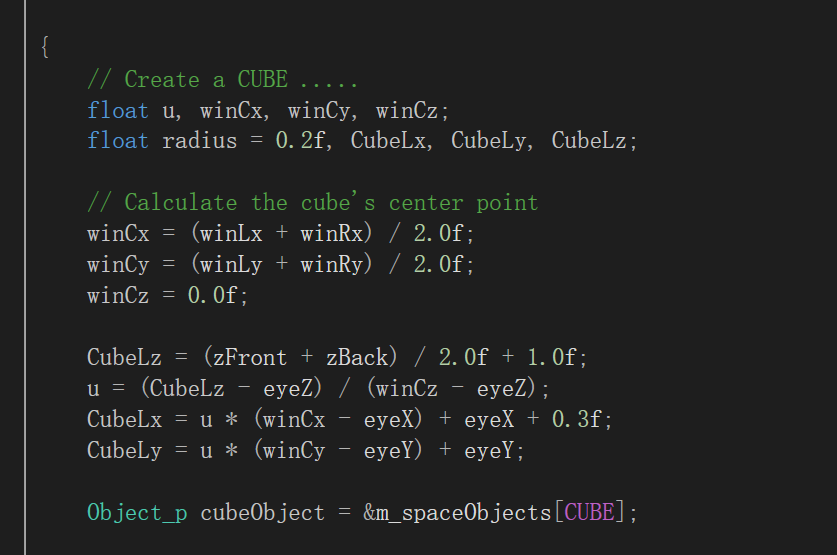
实现原理：通过球心坐标和半径，得到球面的位置；再根据角度参数，得到球面的经纬线；再根据由经纬线构成的四边形得到顶点数据，存到三角形数据结构中。

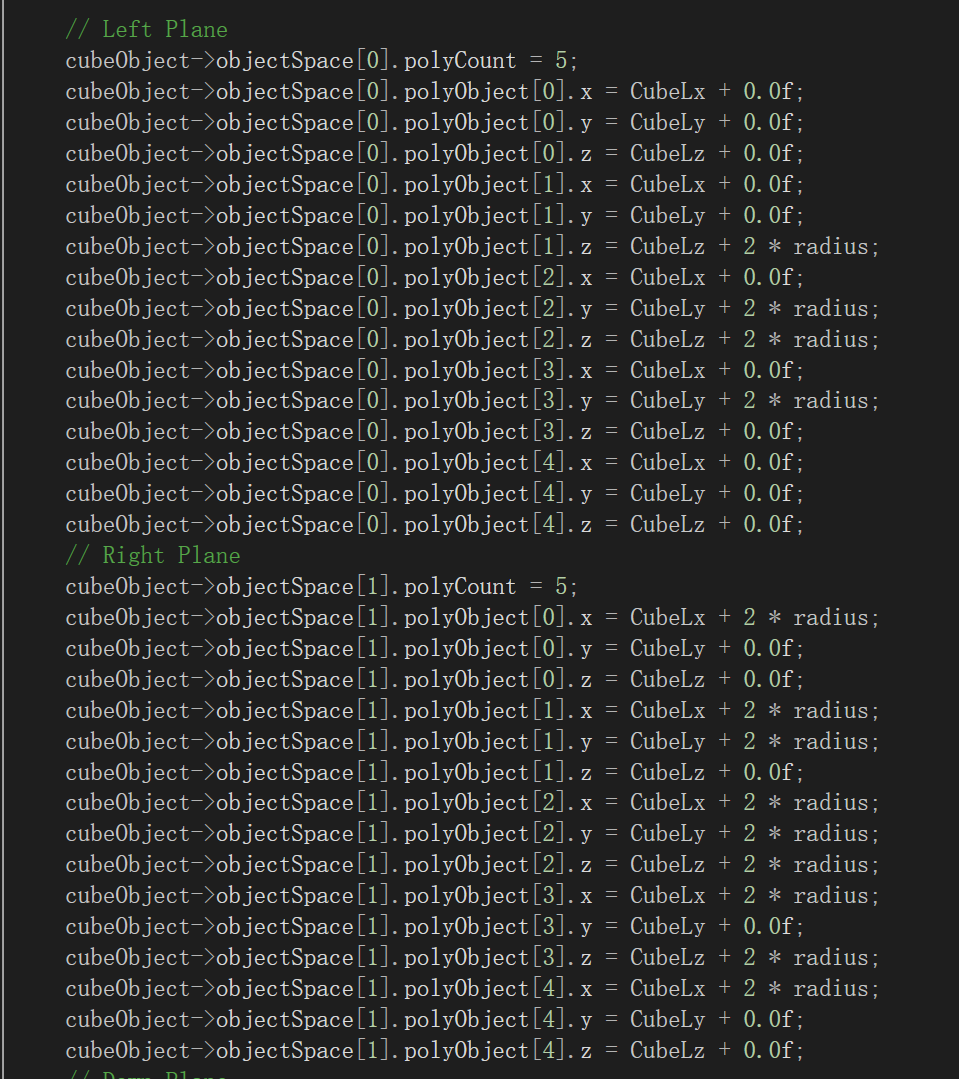


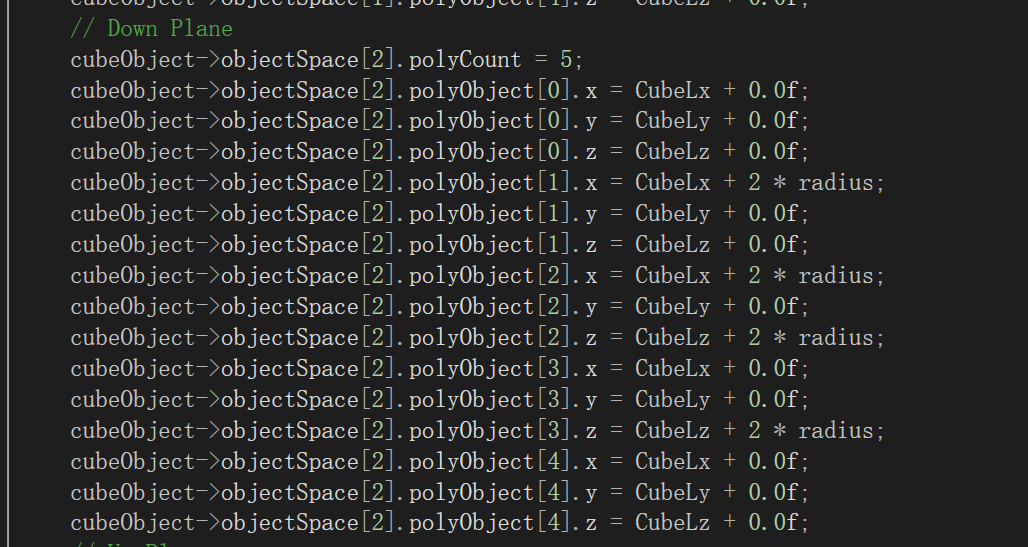


1. 空间立方体模型

实现原理：根据四边形边长和中心坐标，计算出立方体的6个面，将顶点存入数据结构中。

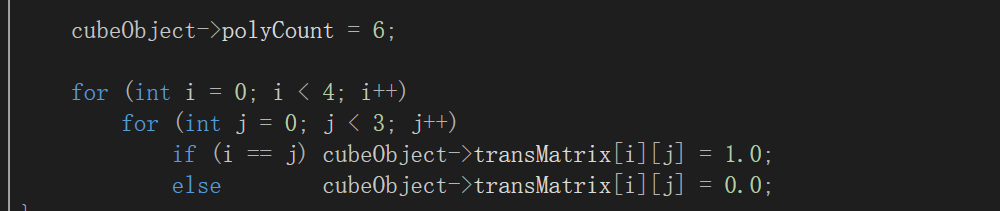






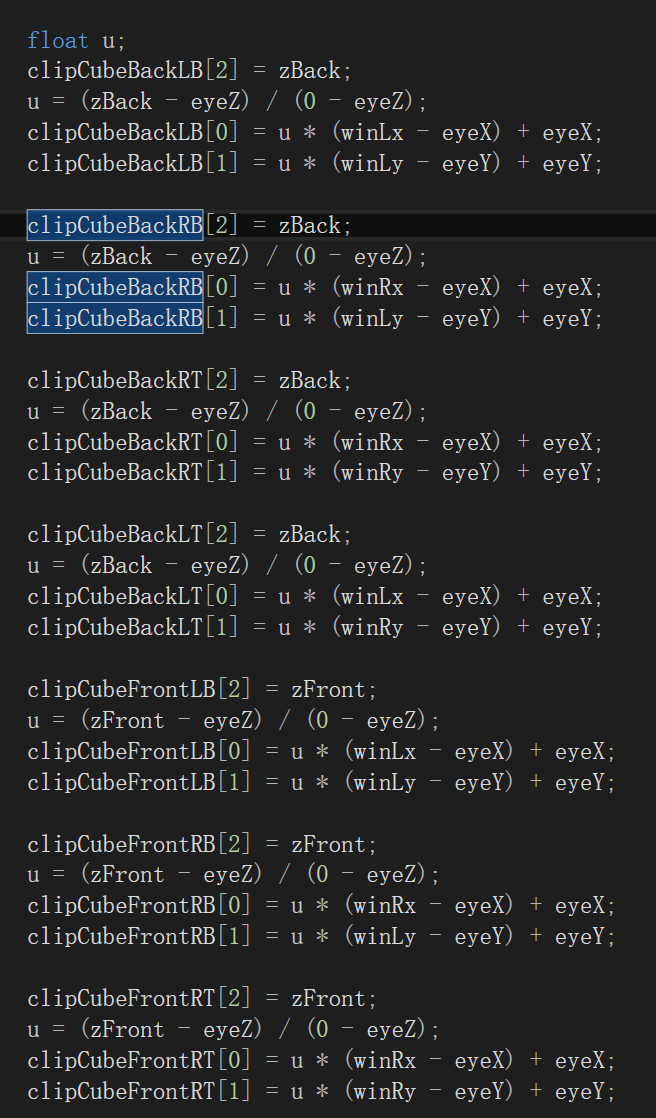


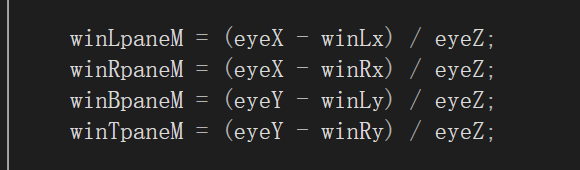




1. 裁剪约束体模型

实现原理：由视点和窗口可见域四个顶点的连线构成四棱放射锥，加上前后面就构成裁剪约束体。根据裁剪约束体，就可以计算出空间裁剪后的目标从而绘制图形，从而实现裁剪操作。





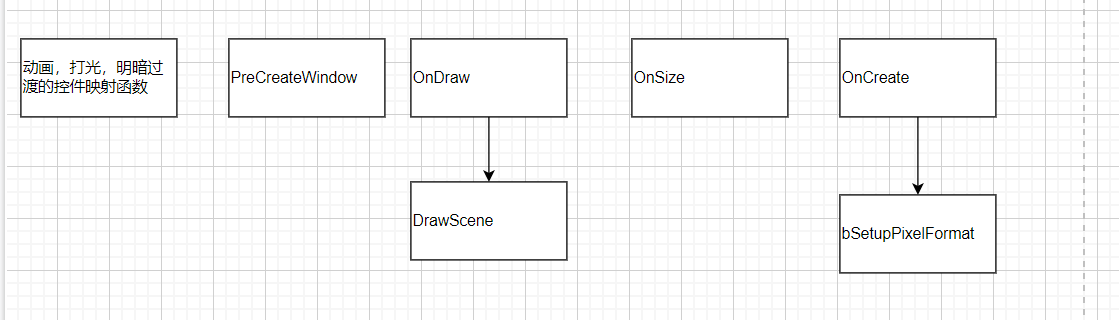
六．空间变换绘制

Cg3DtransView类和DrawScene实现空间变换绘制。

1. CCg3DTransView类

综述：调用GL库进行绘图以及实现控件的映射函数

组成以及调用关系：



1. CCg3DTransView.h 中定义成员变量。

CClientDC \*m\_pDC; // 视区设备描述

CRectm\_viewRect; // 视区尺寸参量

2） CCg3DTransView.h 中定义成员函数。

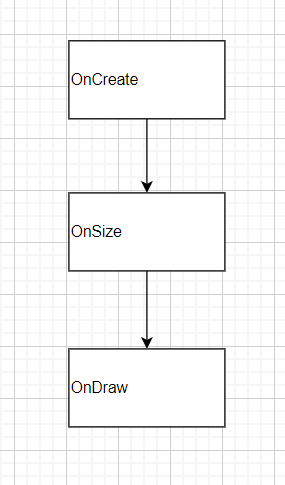
void DrawScene(); // 空间场景绘制函数

BOOL bSetupPixelFormat(); // 空间场景显示格式设置函数

3)重载 WM\_SIZE 和 WM\_CREATE 消息处理函数

afx\_msg void OnSize(UINT nType, int cx, int cy); afx\_msgintOnCreate(LPCREATESTRUCT lpCreateStruct);

处理流程说明：



OnCreate函数进行窗口的建立，OnSize函数进行一些参数设定，OnDraw函数进行窗口的绘制。

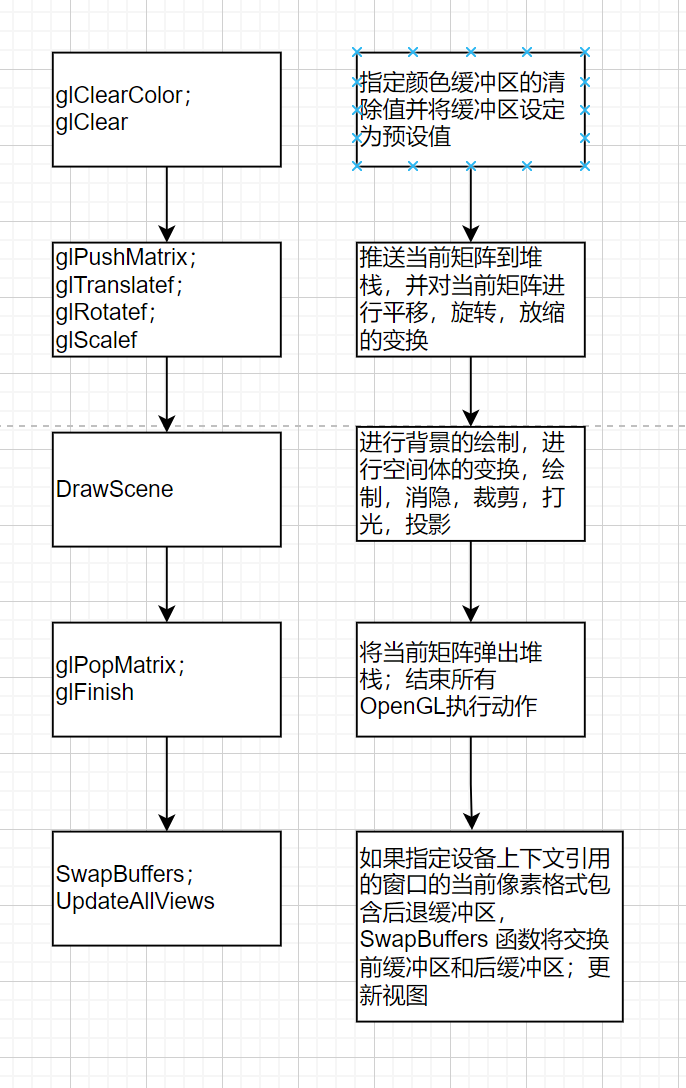
4)空间场景显示格式设置函数

BOOL CCg3DTransView::bSetupPixelFormat()

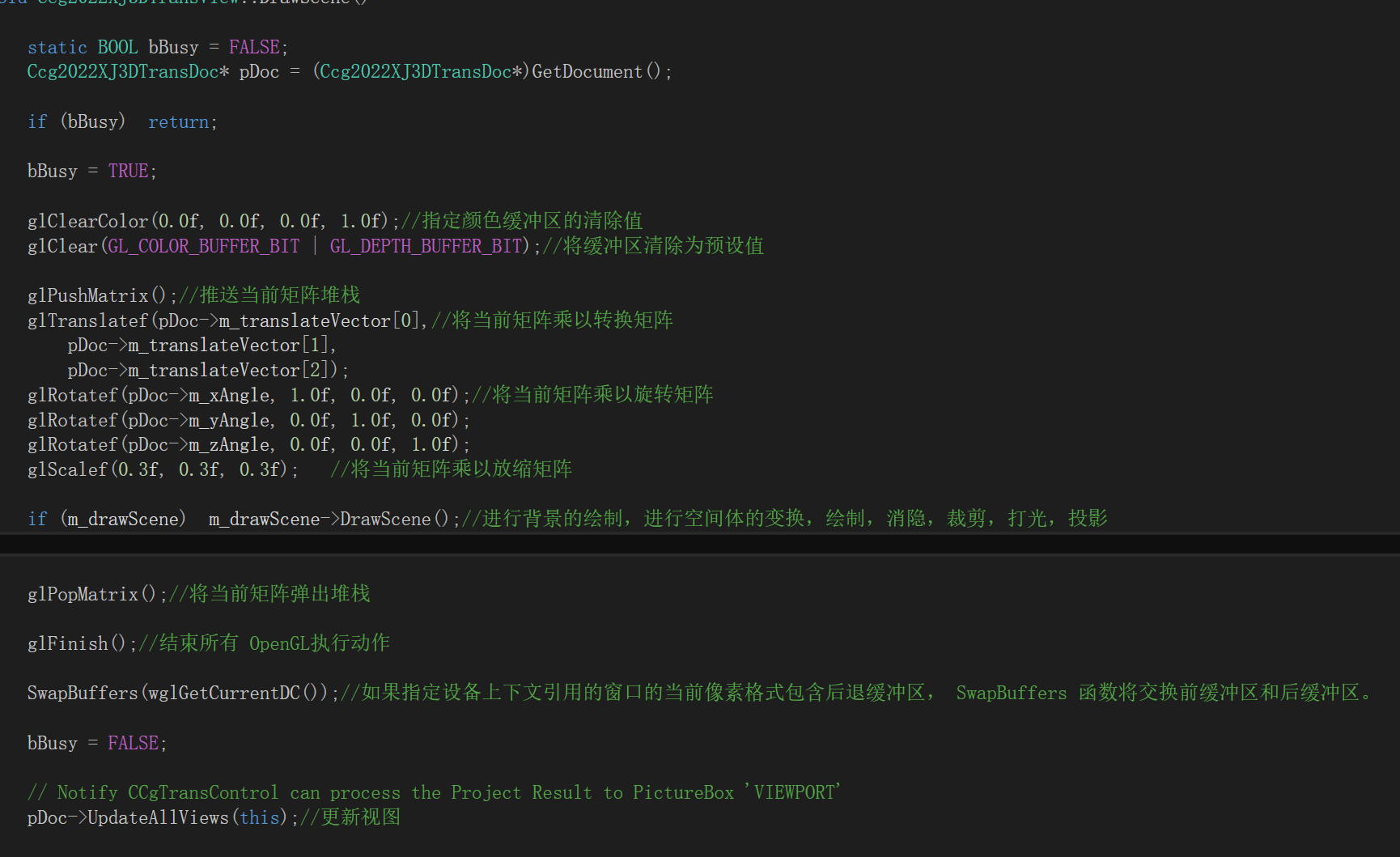
处理流程：先使用一个结构体设置像素的格式（比如支持的窗口，支持的OpenGL,RGBA的格式等等），然后尝试将设备上下文支持的相应像素格式与给定像素格式规范匹配，最后再将指定设备上下文的像素格式设置为 索引指定的格式。

5)空间场景绘制

处理流程：



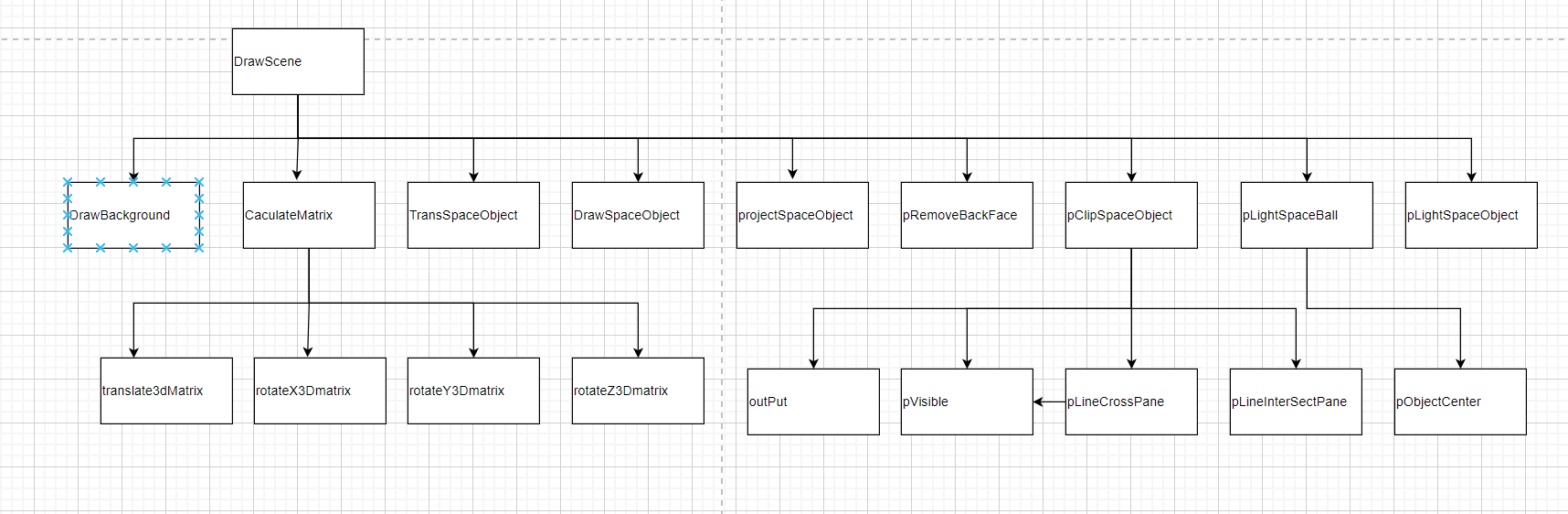
函数实现：



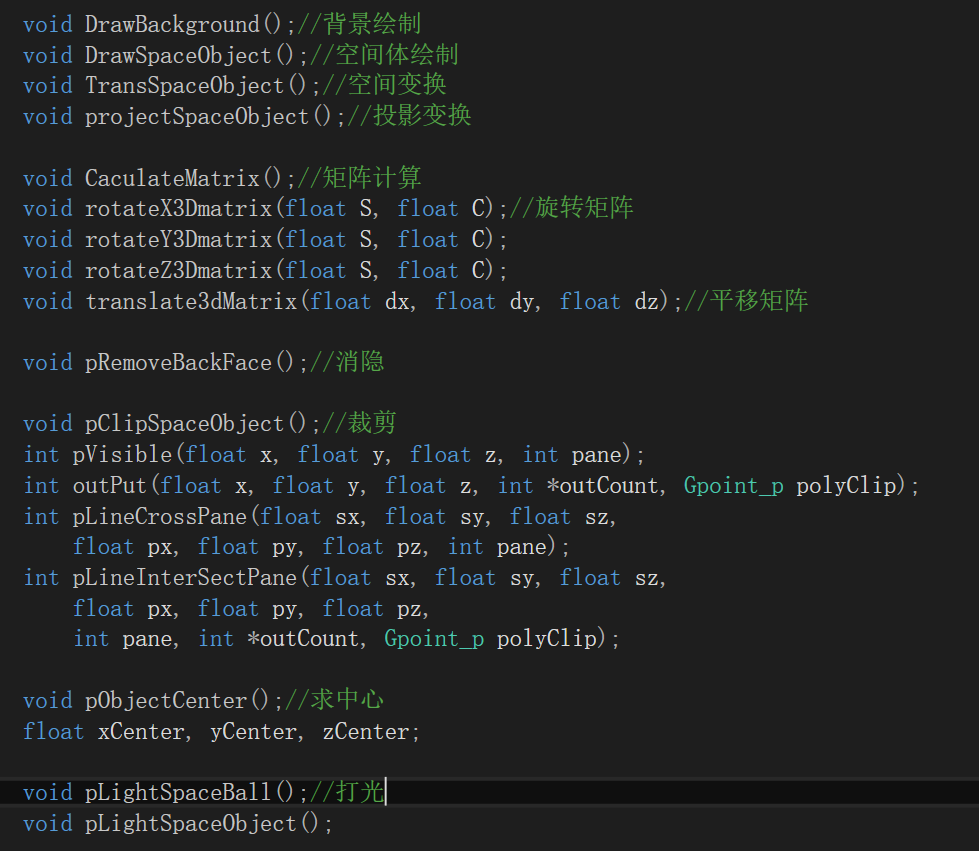
1. cDrawScene类:

综述:进行背景的绘制，空间体的空间变换，绘制，消隐，裁剪，光照，投影。

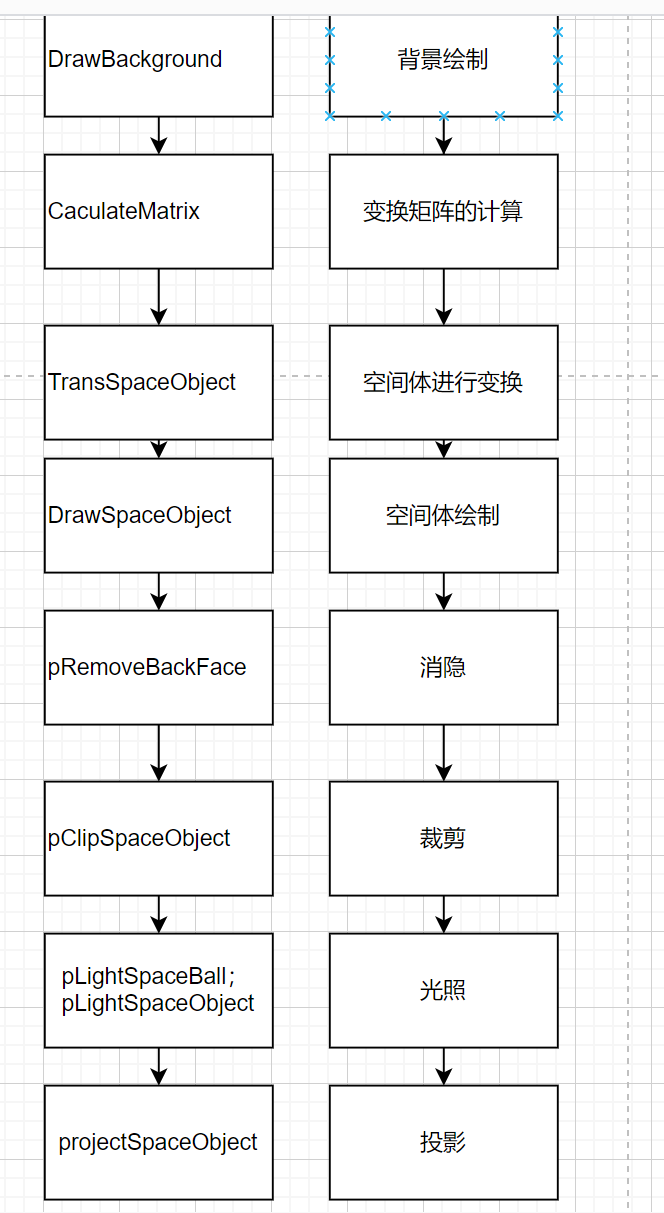
组成和调用关系



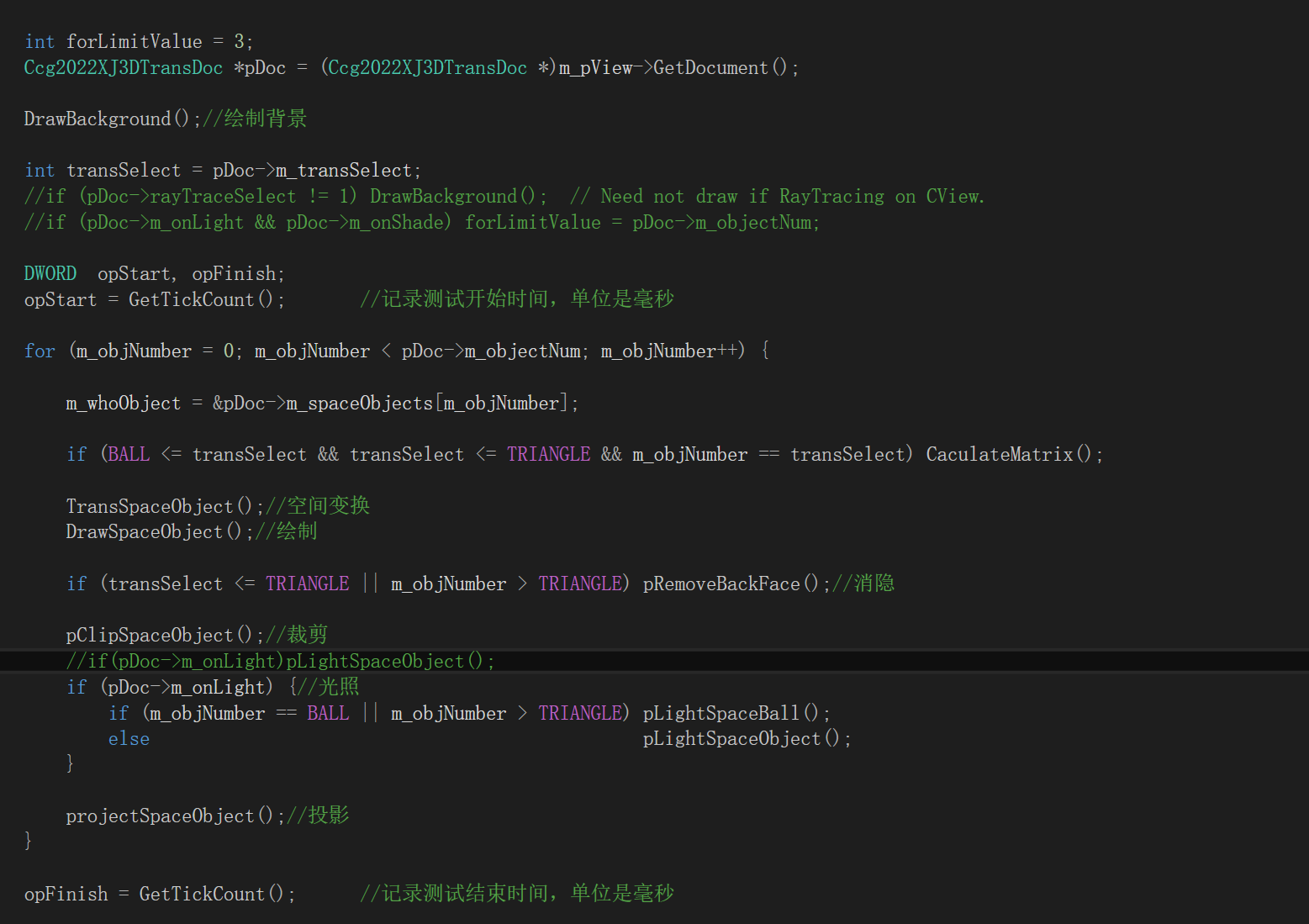
1. CdrawScene成员函数:



1. 空间场景变换绘制流程



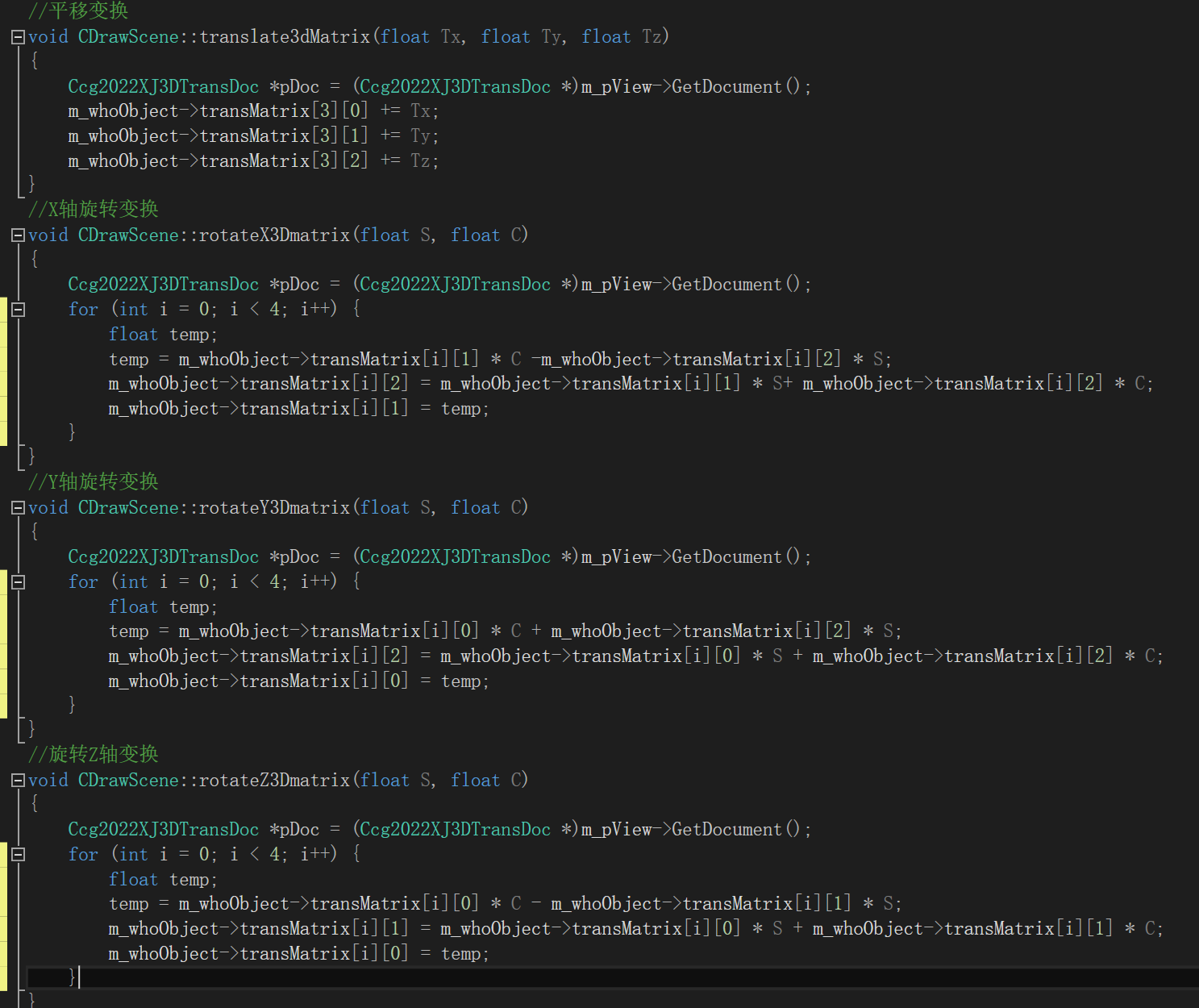
实现：



1. 空间基本变换矩阵计算

基本原理：将平移对应为相应的变换矩阵，原来的位置矩阵再与变换矩阵进行运算得到最终的位置矩阵，即最终的坐标。

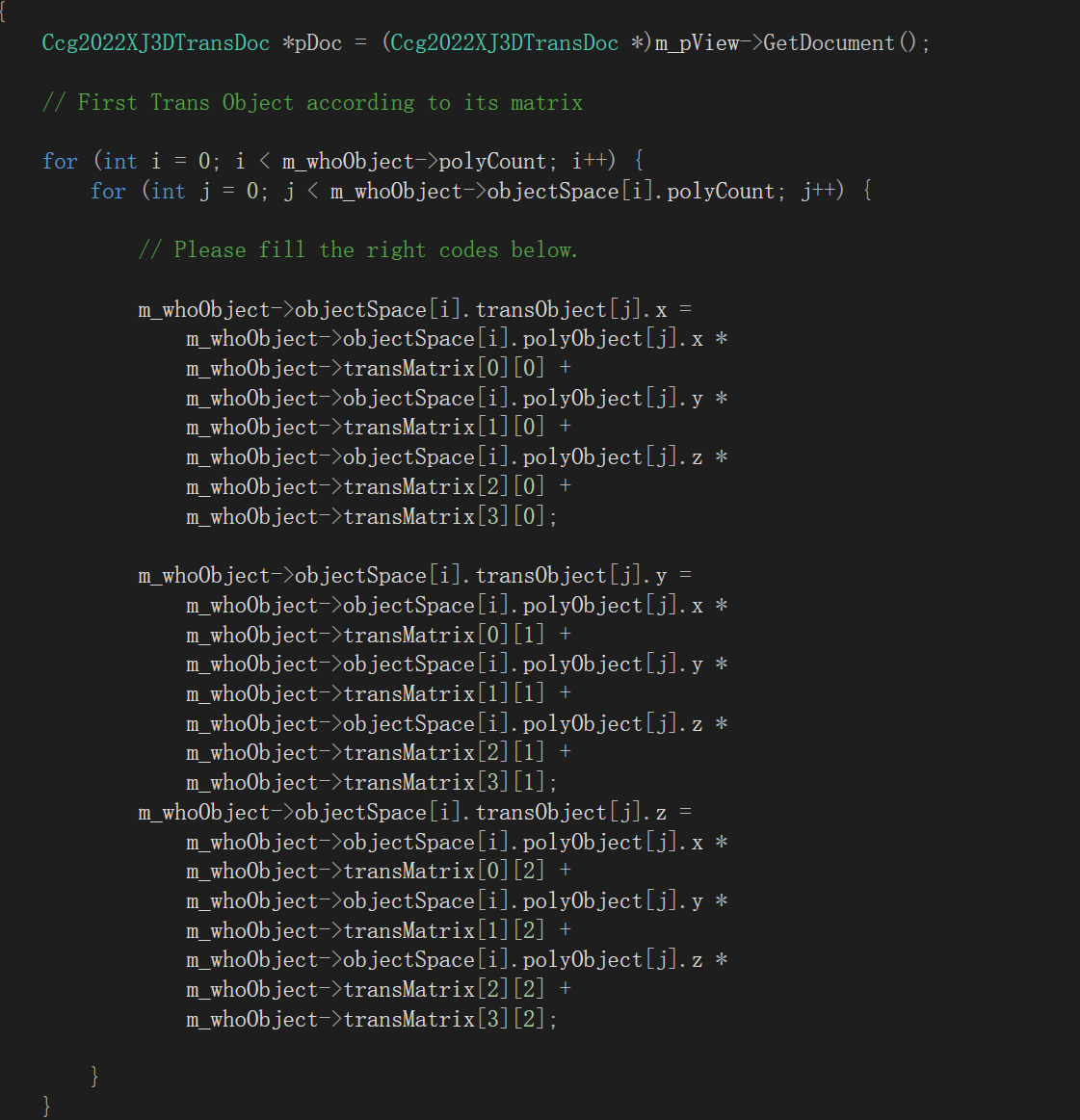
实现方法:



4)空间变换

原理：将坐标矩阵乘上变换矩阵就会得到最终的坐标矩阵

实现:



5)空间投影变换

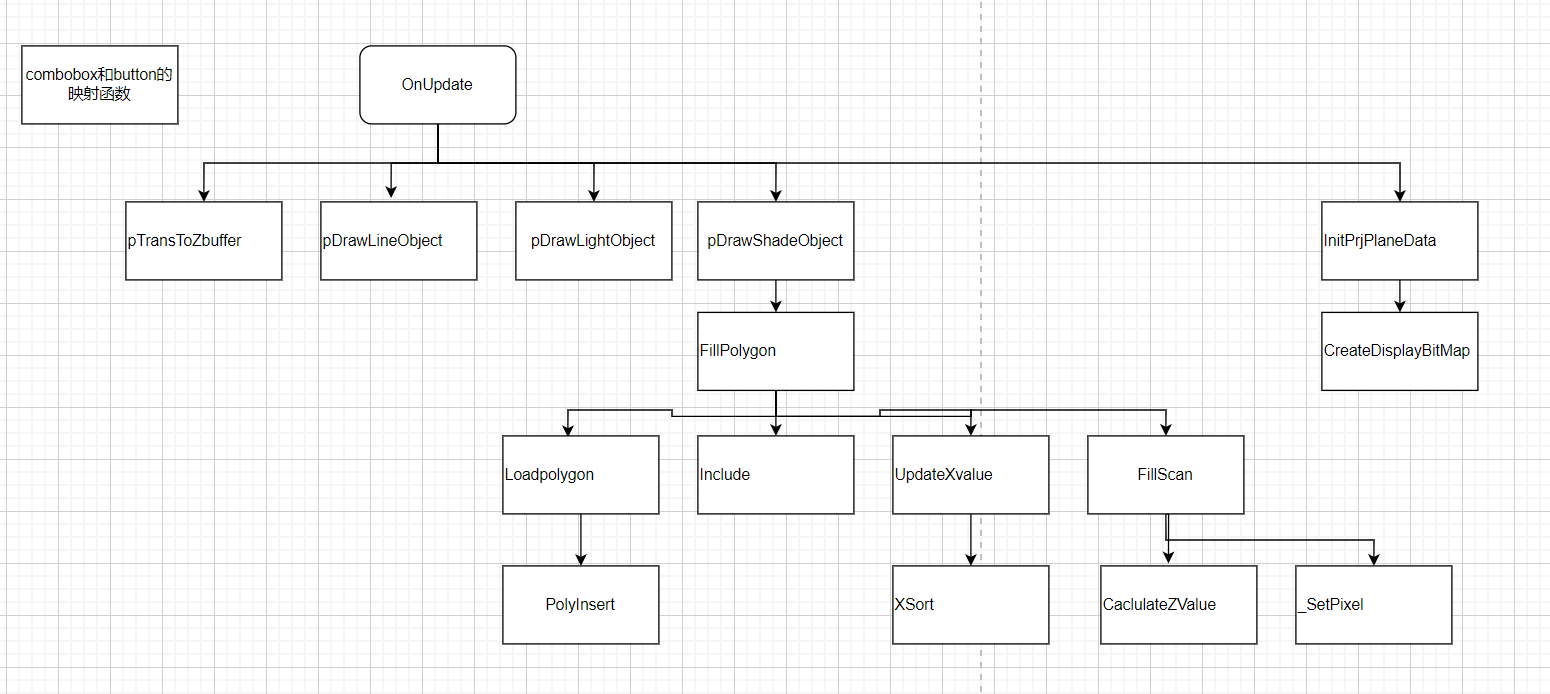
原理：X,Y坐标根据对应的比例投影到平面上，Z坐标则置为0；而后进行绘制。



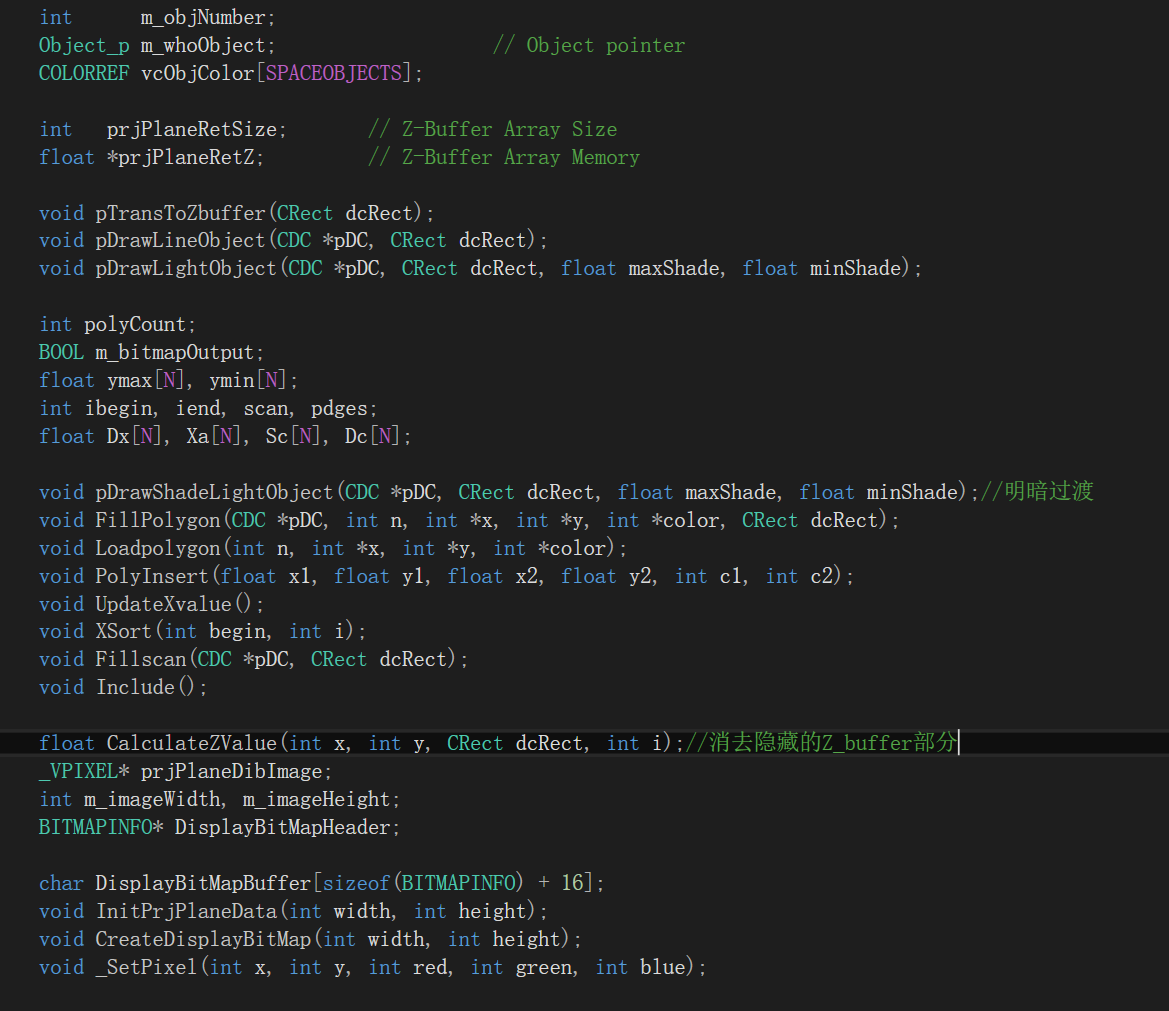
7.空间交互变换与视见变换

综述：实现各button与commobox控件的映射函数；实现明暗消隐。

组成与调用关系



参数与实现方法：



八．空间照相机模型视见变换

实现方法：

