清华校园震害预测程序 2.0 开发
——土木工程 CAE 课程大作业报告

1. 选题背景

区域建筑震害预测分析结果的可视化显示有利于使人们直观、迅速地掌握一次地震对分析区域的影响情况。

课题组已经开发了“清华校园建筑震害预测程序”，该程序可利用清华大学绝大部分校园建筑的坐标、层数、建造年代、结构类型等基本宏观信息，为各个建筑生成相应的剪切层模型，在用户选定的地震动下，对这些校园建筑进行弹塑性时程分析，并直观地展示整个清华校园建筑在地震作用下的位移响应时程和最终破坏状况，如图 1 所示。

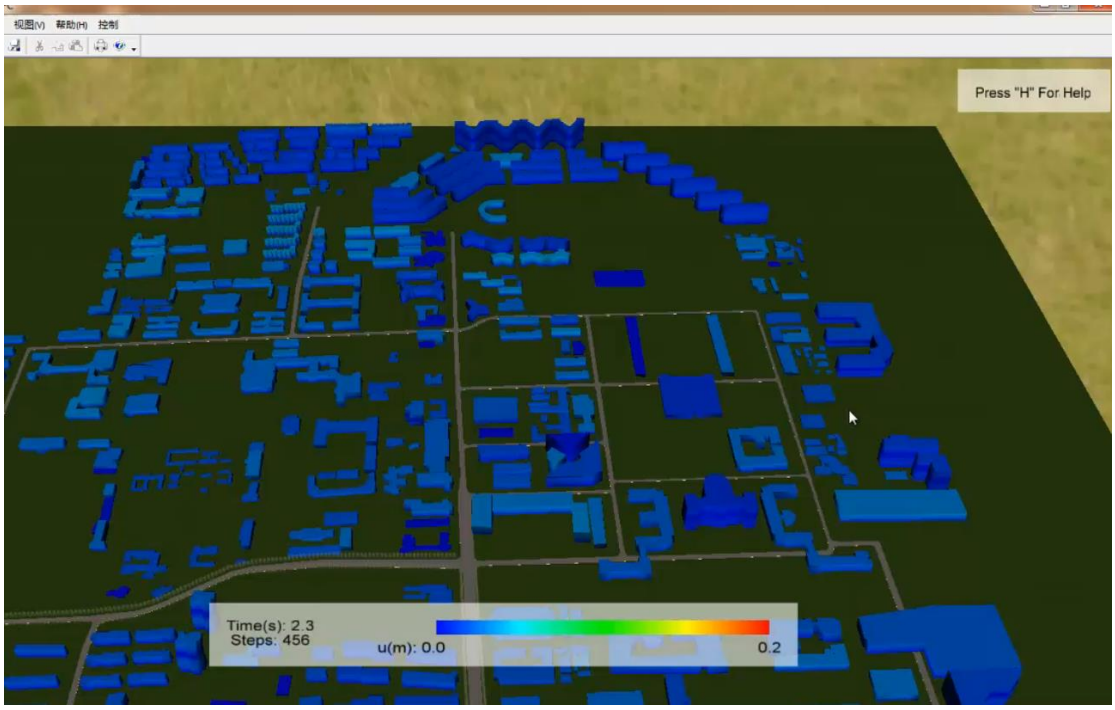

图 1 区域建筑震害预测分析结果的可视化显示

然而，该程序目前的可视化是 2.5D(平面轮廓向上拉起结构高度)的，为了提高震害结果的真实感，想利用已有 3ds Max 实现清华校园的 3D 可视化。同时之前的分析方法采用的是基于美国 HAZUS 的参数标定方法，对于中国建筑的适用性还需要进一步的研究，因此需要增加基于中国建筑抗震设计规范的参数标定方法。之前的分析模型是层模型，不能考虑高层建筑的弯曲变形，所以需要增加弯

剪耦合模型来更好地考虑高层建筑的弯曲变形。为此大作业的题目选择为：清华校园震害预测程序 2.0 开发。

2. 研究内容

大作业的主要目的是，在原有的清华校园震害预测程序的基础之上，开发清华校园建筑震害预测程序 v2.0，主要工作包括：

1. 将 2.5D 的可视化模型换为 3D 的可视化模型；
2. 增加中国规范参数标定方法和弯剪耦合模型；
3. 增加双向地震动计算。

3. 具体实现

3.1 程序主界面的开发

清华校园建筑震害预测程序 1.0 的主界面如图 2 所示。2.0 版本的程序（图 3 所示）在此基础之上进行了更改。主要的更改有：

1. 增加中国规范参数方法；
2. 增加双向地震动计算；
3. 采用三维模型进行后处理展示。

程序会根据相应的选择项进行相对应程序的调用，主界面起到的是一个调用的作用。

图 2 清华校园震害预测程序 v1.0 界面

图 3 清华校园震害预测程序 v2.0 界面

3.2 计算方法和分析模型的改进

1.0 程序的计算方法为多自由度（MDOF）剪切层模型，该模型适用于中低层建筑，但是对于弯曲变形明显的高层建筑而言，不能进行很好地模拟，因此在中国规范参数标定方法中对模型进行了两个部分的改进：参数标定方法和分析模型。

3.2.1 参数标定方法

图 4 给出了基于中国规范参数标定方法的分析流程，图 5 给出了 MDOF 剪切模型参数标定方法。由于该套计算方法和分析模型已经开发好，不是本次作业的主要内容，2.0 的程序中直接调用了该套分析方法的可执行文件，具体的模型介绍和参数标定方法可以参考以下论文：

[1] Xiong C, Lu XZ, Lin XC, Xu Z, Ye LP, Parameter determination and damage assessment for THA-based regional seismic damage prediction of multi-story buildings, Journal of Earthquake Engineering, Accepted on Feb. 26, 2016. DOI: 10.1080/13632469.2016.1160009.

[2] Xiong C, Lu XZ, Guan H, Xu Z, A nonlinear computational model for regional seismic simulation of tall buildings, Bulletin of Earthquake Engineering, 2016, 14(4): 1047-1069.

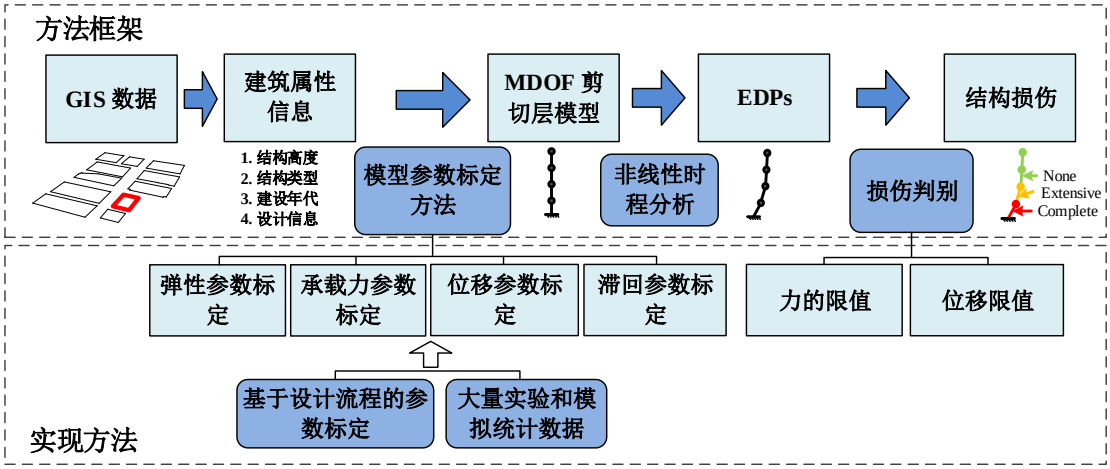

图 4 中国规范参数标定方法的分析流程

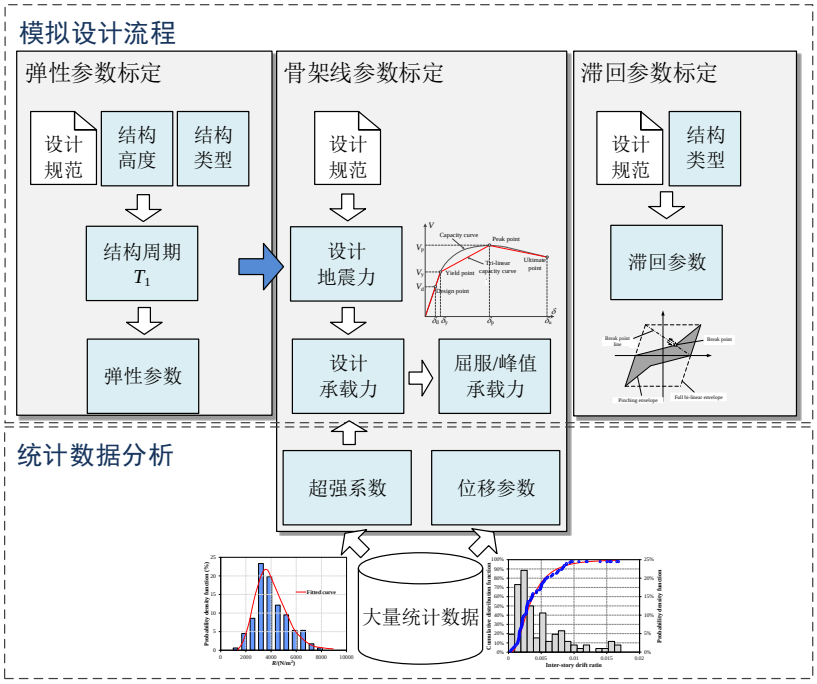

图 5 MDOF 剪切模型参数标定方法

3.2.2 分析模型

分析模型在原有的多自由度剪切层模型的基础上增加了 MDOF 弯剪耦合模型，如图 6 所示。

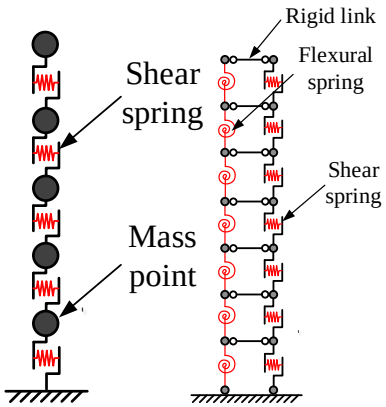

图 6 MDOF 剪切层模型和弯剪耦合模型

3.3 可视化模型的更改

原有的可视化程序采用的是 2.5D 的模型，该模型直接基于建筑平面外围轮廓，根据建筑高度进行拉伸而得到。2.0 版本的程序采用 3D 模型进行后处理展示。整个实现过程如图 7 所示。

3.3.1 3D 模型的前处理

首先根据获取 3Ds Max 模型导出单栋建筑的 dae 格式（Digital Asset Exchange）的文件，并将实地考察得到建筑基本属性（建筑名称、结构类型、层数、高度、建造年代、建筑面积和建筑功能）赋予给该栋建筑。

3.3.2 3D 模型的解析

由于导出的 dae 文件并没有层的概念，这里需要对其进行基于楼层标高的切片操作，将建筑分出层。同时解析 dae 文件，得到建筑外表面多边形点的坐标。该部分程序已经实现，2.0 程序开发过程中直接采用了该程序。

3.3.3 基于 OSG 的 3D 可视化

3D 可视化是 2.0 程序开发中的主要环节。以下进行较为具体的介绍。

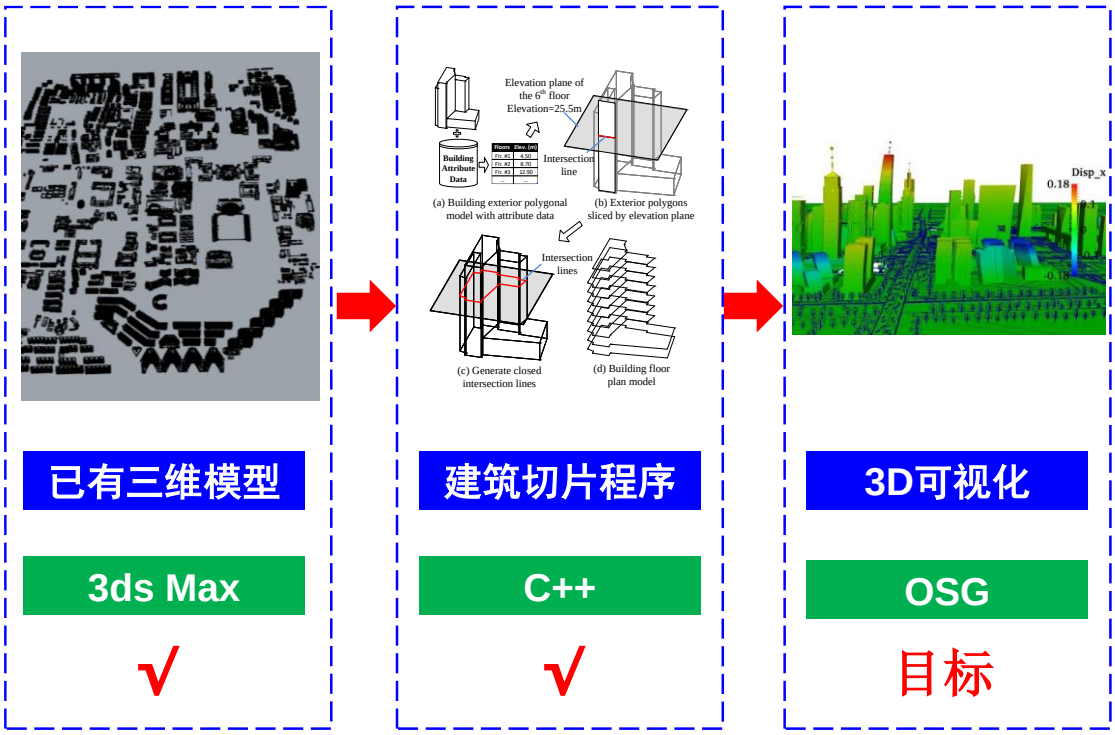

图 7 3D 可视化实现方法

3D 可视化的开发是基于 OpenSceneGraph（OSG）平台的。OSG (Open Scene Graph) 是一个基于 OpenGL 的开源三维图形引擎，它对 OpenGL 实现了一定程度的封装，因此开发者不需要关心底层的 OpenGL 代码，而可以基于 OSG 提供的库，更方便的进行编程开发。

（1） 程序框架

程序沿用之前 1.0 版本的可视化程序的框架，如图 8 所示。

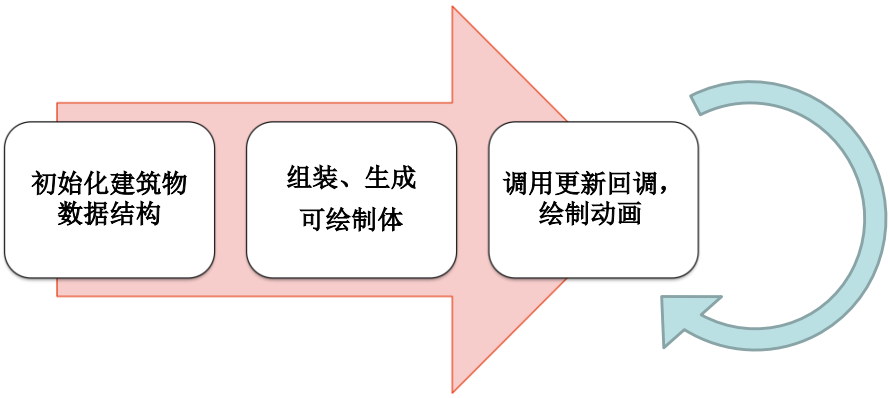

图 8 程序框架

（2） 场景组织结构

OSG 的场景组织结构，即 OSG 中的数据存储与组织形式。掌握 OSG 的场景组织结构，才能合理、高效地设计程序的数据结构，存储和管理各个建筑的信息及其显示。本程序的场景组织形式如图 9 所示。

图 9 可视化程序的场景组织形式

（3） 绘制建筑

采用上述的场景组织形式，首先需要绘制建筑多边形的图元，其在程序中实现的代码如表 1 所示。

vvv[]和 vv[]数组分别为所绘制多边形的颜色数组和顶点数组。采用 setColorArray()和 setVertexArray()函数即可将相应的颜色数组和顶点数组赋予给建筑多边形图元。通过 DrawElementsUInt()函数创建多边形索引数组，多边形的维数由 noc[][]数组确定，通过 addPrimitiveSet()函数即可将索引数组添加到几何图元，再通过 addDrawable()函数即可将绘制好的图元添加的叶节点。

接下来按照场景组织形式，将相应的节点添加到相应的位置，就可以完成整个可视化场景的定义。

表 1 绘制建筑多边形代码

```
osg::ref_ptr<osg::Geometry> wall=new osg::Geometry;
vvv[i]->push_back(osg::Vec4((*color)[i].x(),(*color)[i].y(),(*color)[i].z(),(*color)[i].w()));
wall->setColorArray(vvv[i]);
wall->setColorBinding(osg::Geometry::BIND_OVERALL);
for (int l=0;l<noc[j][k];l++)
{vv[i]->push_back(osg::Vec3((*v)[cc].x(),(*v)[cc].y(),(*v)[cc].z()));
cc++;}
wall->setVertexArray(vv[i]);
osg::ref_ptr<osg::DrawElementsUInt> poly= new
        osg::DrawElementsUInt(osg::PrimitiveSet::POLYGON,0,noc[j][k]);
for (int m=0;m<noc[j][k];m++)
{poly->push_back(m);}
wall->addPrimitiveSet(poly.get());
geode->addDrawable(wall.get());
```

（4） OSG 更新回调

建筑地震下位移响应要求每帧都能更新画面（即更新节点坐标并重新绘制），从而体现动画效果。

逐帧操作有至少两种实现方式。

第一，可以通过调用 osgViewer::Viewer::frame()方法直接控制每帧的渲染与前进，并调用节点的遍历函数，对所有节点进行遍历访问。这种方式允许用户进行更灵活、更复杂的程序设计，例如实现快进、回退等功能，但要求用户手动编写的程序部分也相对较多。

第二，可以直接重写 osg::Drawable::UpdateCallback::update()方法，定义每帧需要进行的更新操作。程序将自动进行每帧的渲染，且每次渲染时，将自动调用节点访问器遍历所有节点，并执行可绘制体的 update()方法。例如，如果需要让某个顶点沿 x 轴平移，且每帧移动一个单位，只需要将 update()方法重写成表 2 形式。

表 2 更新回调代码

```
void update(osg::NodeVistor *nv, osg::Drawable *drawable)
{osg::Geometry *geom=dynamic_cast<osg:Geometry*>(drawable);
osg::Vec3Array*vertices=geom->getVertexArray();
vertices[0].x()+=1;
vertices->dirty();
}
```

这种方式下，用户只需重写 update()方法即可完成逐帧更新的操作，需要手动编写的代码相对较少，较适合于初学者或较简单的程序的实现。

因此，本程序中选择的是上述第二种方法实现逐帧更新节点坐标并重绘建筑体，从而实现建筑地震下位移响应的显示。

4. 成果展示

详细的程序演示以及操作说明可以参考附件，这里给出了 3D 可视化结果，如图 10、11 和 12 所示。

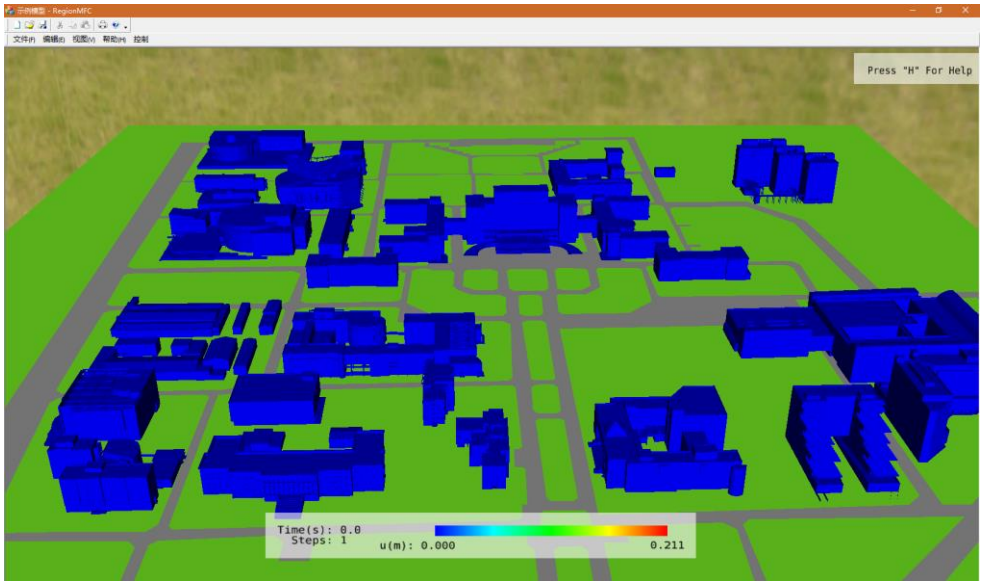

图 10 查看分析结果

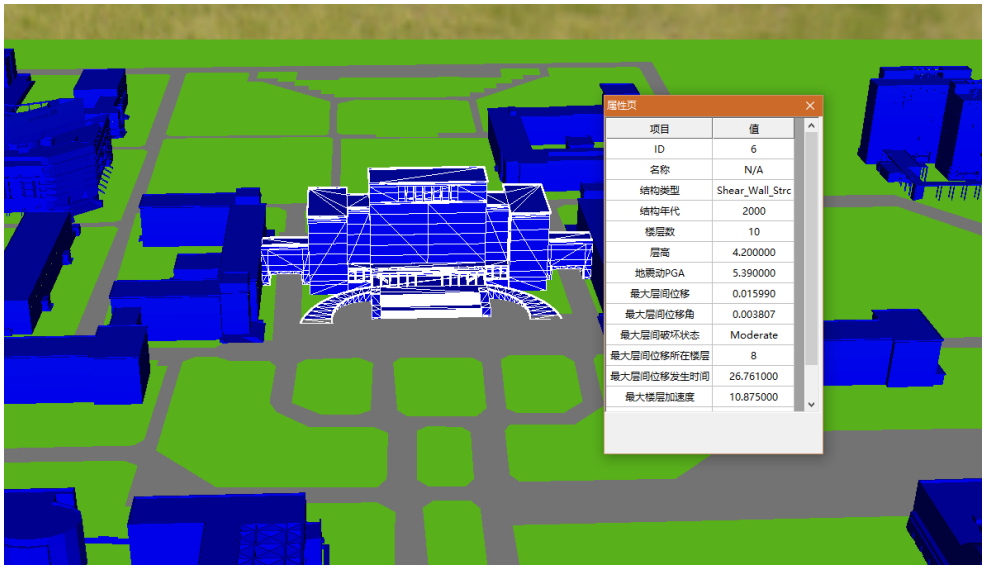

图 11 单击建筑可查看建筑的破坏状态等信息

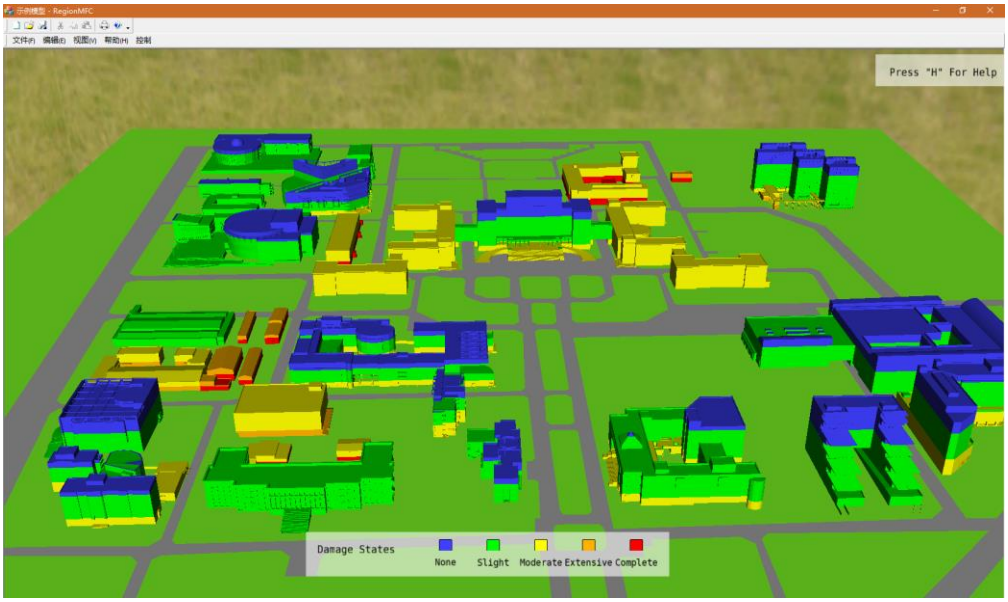

图 12 整个区域的破坏情况

5. 心得体会与课程建议

5.1 研究体会

在此之前，从未接触过图形学，对 OpenGL 和 OSG 也从未了解过，前期学习 OSG 花了很多时间。第一遍看 OSG 书的时候，非常难懂，在考虑是不是要从底层的 OpenGL 开始学起。犹豫之际回头看之前看过的书，发现慢慢懂了一些。再加上看之前清华校园 1.0 的程序，带着目的去学习，效率就上去了。

之前也从未接触过一个大型项目的开发，所以在写程序的过程中遇到了很多问题，有的卡了好几天，就是不能解决，也找不到人问，非常难受，但是当问题真真解决了，才能明白柳暗花明又一村的真正含义。

整个项目开发过程中让我明白了：代码是永远不会骗人的！遇到的错误越多，离编程高手就越近。

5.2 课程建议

(1) 大作业环节对于学生的锻炼非常有帮助！！！

(2) 老师根据自己的经历来讲课很有代入感；

(3) 建议平时来点作业，巩固学习的知识，比如 OpenGL 编一个小程序。