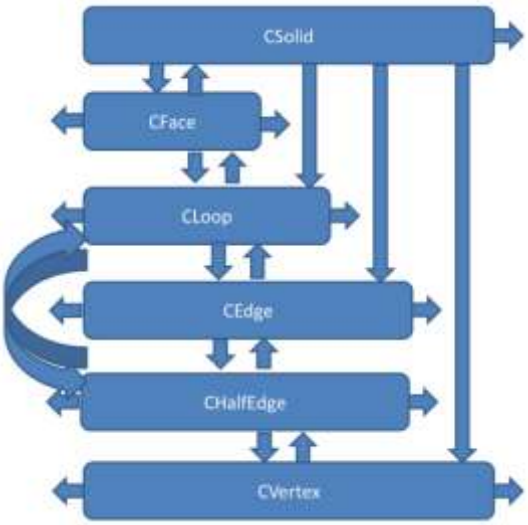
使用说明

姓名：Call 偶围城 学号：XXXXXXXX

一、 数据结构介绍

我的程序一共有九个类，其中 CSolid、CFace、CLoop、CEdge、CHalfEdge 和 CVertex 是用于实现半边数据结构的类，CCoordinate 是坐标类以方便程序中的各种坐标操作，CFileStream 是文件输入输出的类，所有的文件操作都在类里实现了，CEulerOperator 实现了五个欧拉操作和五个欧拉逆操作以及基于欧拉操作实现了 Sweeping 操作，其中五个欧拉逆操作由于没有实际需求没有去严格验证其正确性。

其中 CSolid、CFace、CLoop、CEdge、CHalfEdge 和 CVertex 六个类之间的相互关系如下图所示：



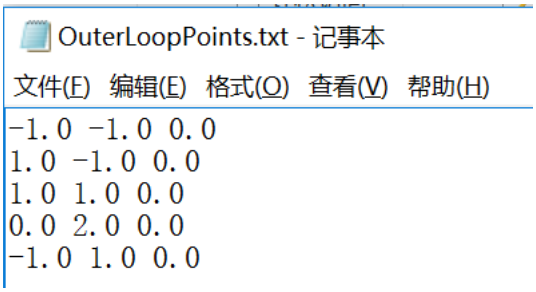
上图中向下的箭头指所属的第一个成员，向上的箭头指从属关系，向右的箭头指下一个同类，向左的箭头指前一个同类。其中 CSolid 中有各类元素的计数器，但是有些后来被弃用了，比如环的数目，开始设计的计数的方式和真实情况不一样，后来在需要统计时通过重新遍历数据结构得到相应的计数。CSolid 有指向 CEdge 的指针是为了方便查找。

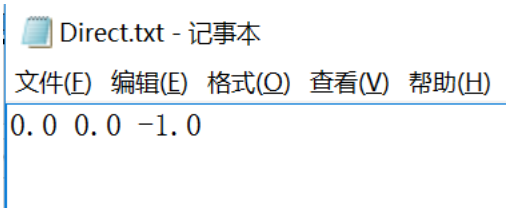
二、欧拉操作实现

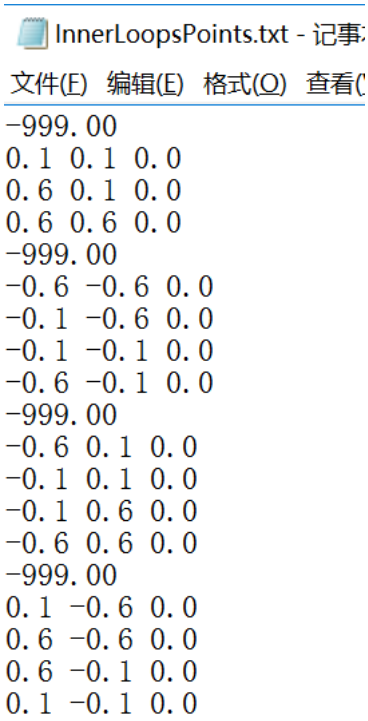
程序的欧拉操作是基于我的半边数据结构实现的，其中五个欧拉逆操作在后来的扫成中没有被用到。具体实现的过程可见程序 EulerOperator.cpp，这里不太好叙述。

三、文件读入介绍

这里我一共有三个读入文件：OuterLoopPoints.txt，InnerLoopsPoints.txt 和 Direct.txt 分别存储着围成外环的点坐标，围成众多内环的点坐标和扫成方向的向量。



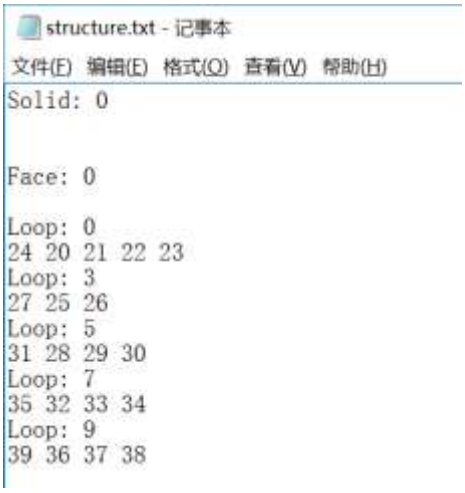




其中有四点需要说明一下。第一，InnerLoopsPoints.txt 文件中-999.00 是两个环之间的间隔标志；第二，请保证围成环的点逆时针顺序输入；第三，请保证环之间没有重叠部分；第四，环可以拥有任意大于等于三的顶点。

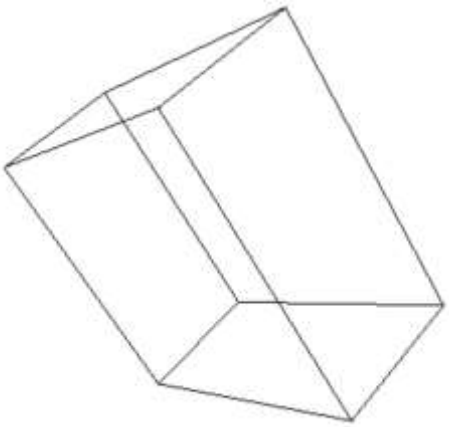
四、Structure.txt 文件介绍

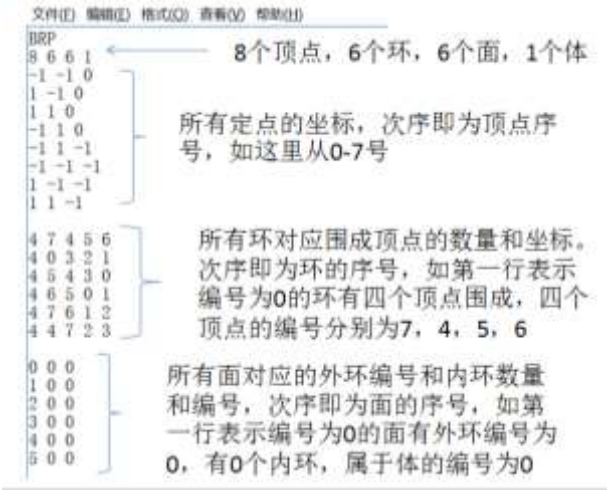
Structure.txt 是我在编程过程中为了方便查看建立过程是否正确所输出的文件，其中 Solid，Face，Loop 后的都是相应的编号，而 Loop 下的是围成环的点的坐标。其具体格式如下：



五、Brp 文件介绍

这里 brp 文件我是按照框架所给的结构建立的，这里以简单的长方体为例，所建立的 brp 文件内容如下：





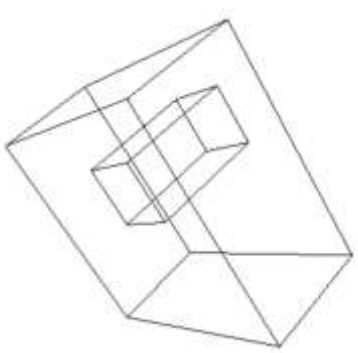
如果面有内环的话，内环的编号会跟在内环数量之后，如 0 2 4 5 0 代表着编号为 x 的面外环编号为 0，有两个内环，两个内环的编号分别为 4 和 5，属于体的编号为 0。

六、效果展示

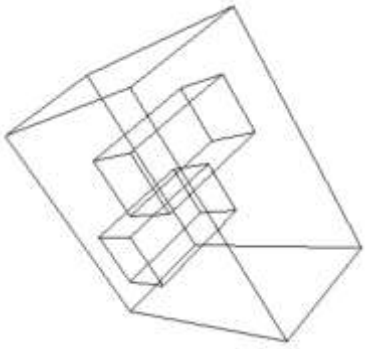
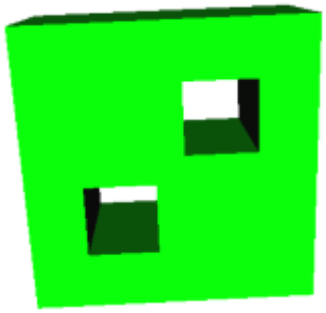
这里展示我的部分测试数据，如果想进一步测试改变测试数据即可，然后用框架所给的程序打开新生成的 brp 文件即可。这里我也实现了仅仅显示线框结构的方法，运行我的程序即可。

其中线框显示我的视点设置在（4，1.5，2），视线方向为（1，1，0）

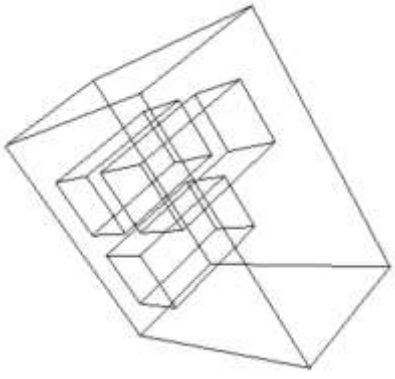
一个环：



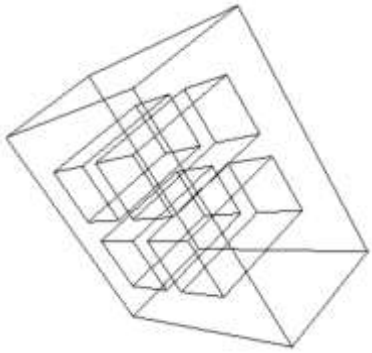

两个环：

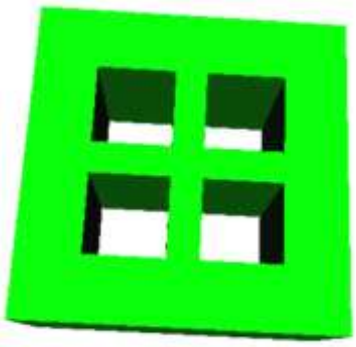



三个环：

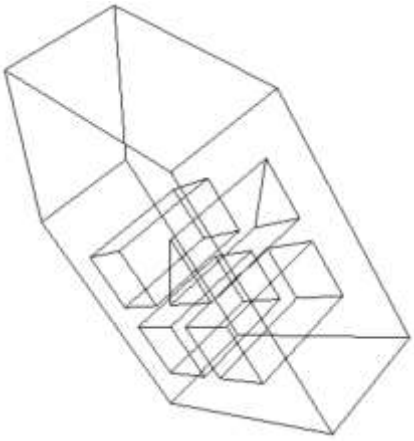



四个环：





非四边形环测试：



七、环境配置

我使用的 IDE 是 VS2010，配置了 OpenGL，需要的配置文件在压缩包内，具体配置方法如下：

1. 将 glut.lib 和 glut32.lib 这两个静态函数库复制到文件目录的 lib 文件夹下 X:\Program Files (x86)\Microsoft Visual Studio 12.0\VC\lib

2. 将 glut.dll 和 glut32.dll 这两个动态库文件放到操作系统目录下面的 C:\Windows\system32 文件夹内（32 位系统）或C:\Windows\SysWOW64(64 位系统）

3. 将解压得到的头文件 glut.h 复制到 X:\Program Files (x86)\Microsoft Visual Studio 12.0\VC\include\GL，如果在 incluce 目录下没有 GL 文件夹，则需要手动创建

4. 打开项目属性，设置包含目录和库目录，分别添加 gl 文件夹和 lib 文件夹所在的目录

在文档的最后我要特别感谢高老师一学期以来的指导，同时也要感谢制作渲染框架的师兄师姐。