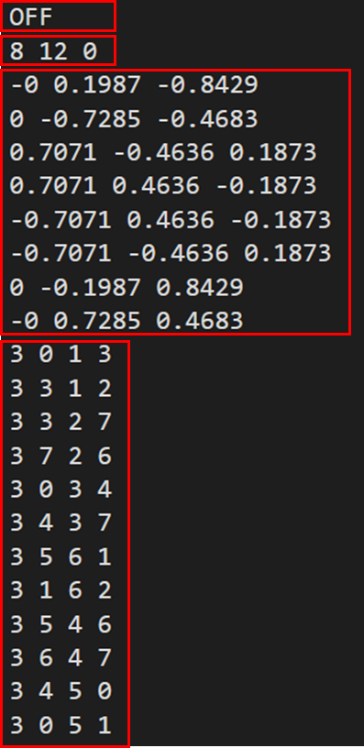
实验2.2 OFF格式的模型显示

1. 实验目的
2. 了解和熟悉OFF模型文件格式。
3. 掌握读取OFF模型文件。
4. 了解基本3D图元的绘制。
5. 了解深度测试技术。
6. 了解面剔除技术。
7. 理论背景
8. OFF格式文件

OFF，Object File Format，即物体文件格式，是一种三维模型文件格式。物体文件格式（.off）文件通过描述物体表面的多边形来表示一个模型的几何结构，这里的多边形可以有任意数量的顶点。本次实验提供了两个立方体的off文件，放在assets文件夹下。

Princeton Shape Benchmark中的 .off 文件遵循以下标准：

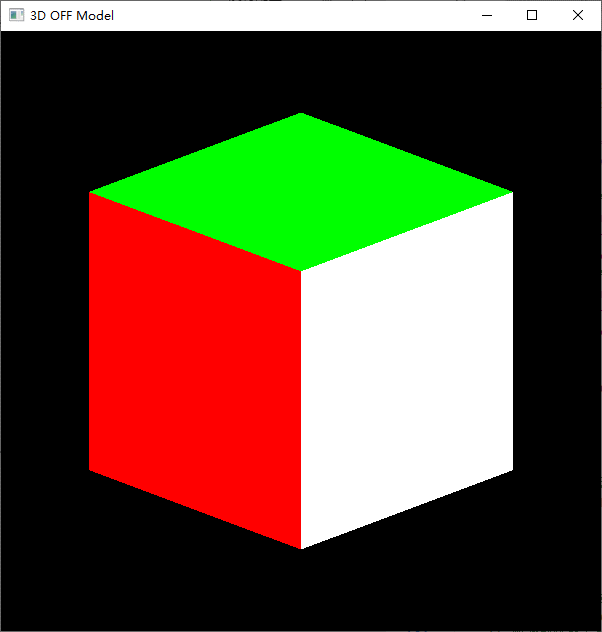
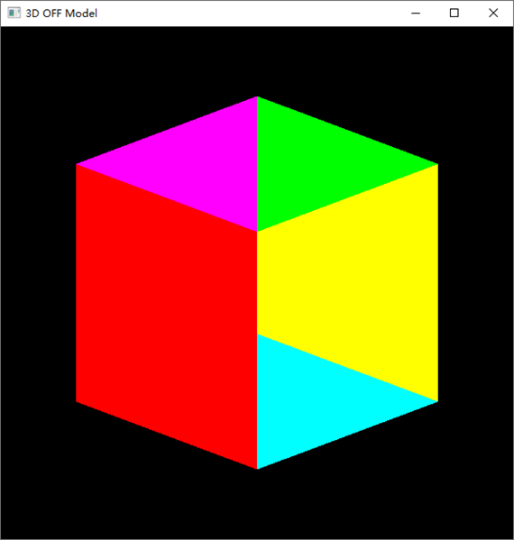
* OFF文件全是以OFF关键字开始的ASCII文件。
* 第二行说明顶点的数量、面片的数量、边的数量。边的数量可能会省略。
* 第三行开始是顶点列表，顶点按每行一个列出x、y、z坐标。
* 在顶点列表后，面片按照每行一个列表，对于每个面片，顶点的数量是指定的，接下来是顶点索引列表。比如图中有一行是 3 1 6 2，它表示该面片有3个顶点，由第1、6、2个顶点构成。



1. 深度测试

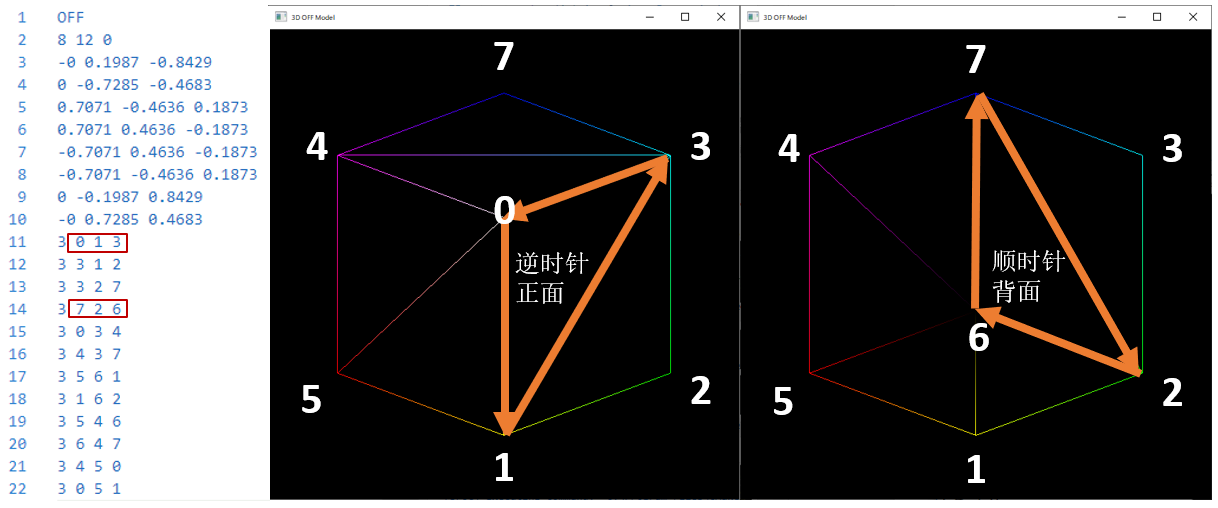
在绘制时，如果屏幕上当前像素要绘制新的候选颜色，只有对应物体比之前的物体更靠近观察者，我们才能绘制它。初始状态下，深度缓存的值是一个距视点尽可能远的最大值，而所有物体的深度值都要比这个值更靠近视点。每帧重绘场景时都要清除深度缓存数据。

下面两幅图对比了开启深度测试的效果，左边是没有开启深度测试的结果，右边开启了深度测试技术。可见深度测试技术能带来更准确的渲染结果。

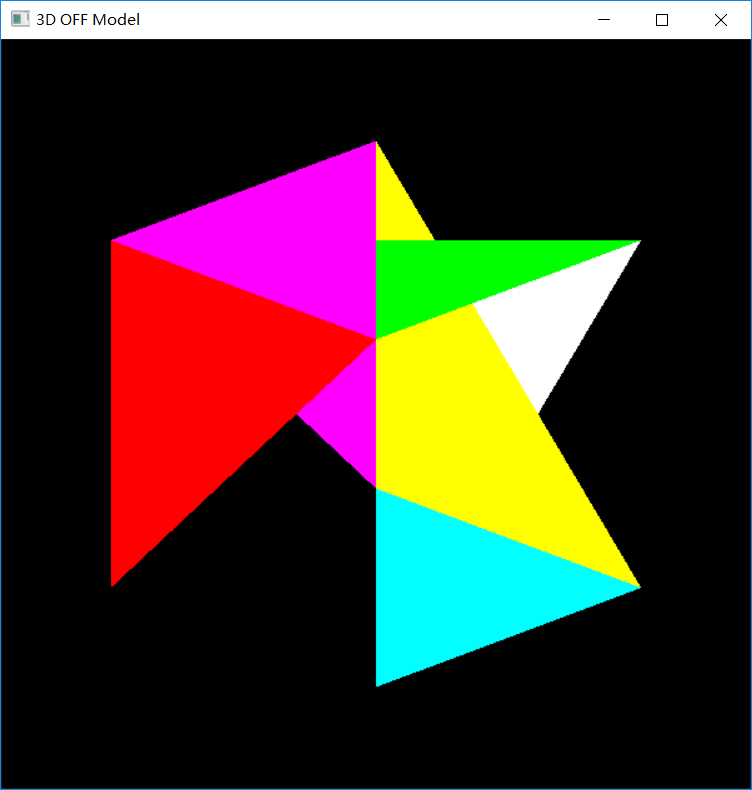
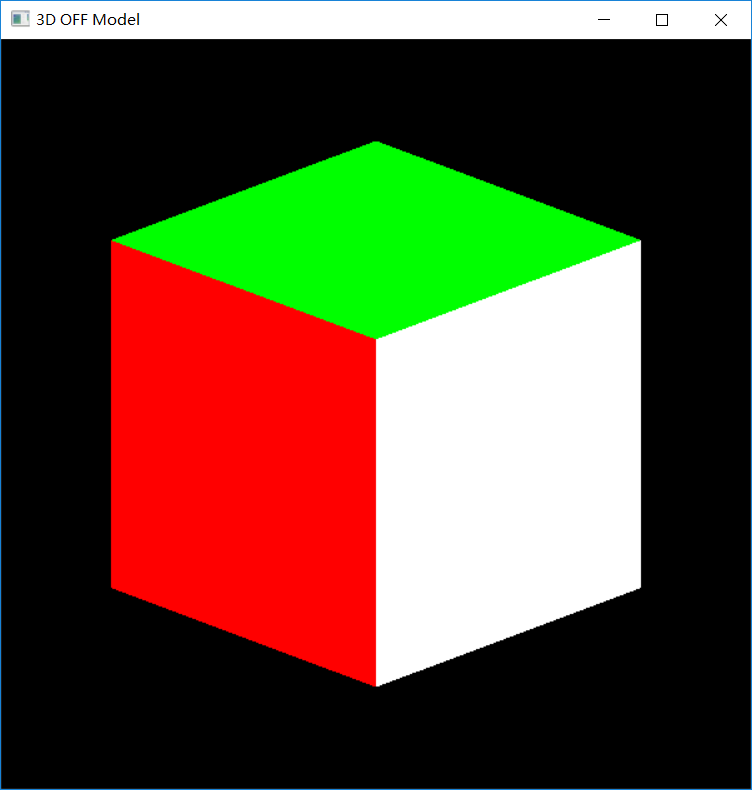


1. 面剔除

默认情况下，逆时针的顶点连接顺序被定义为三角形的正面。开启面剔除模式，会默认去除掉所有不是正面朝向的面，这样可以节约很多的处理性能。下面是实验提供的cube.off文件中的顶点编号和面片顺序。

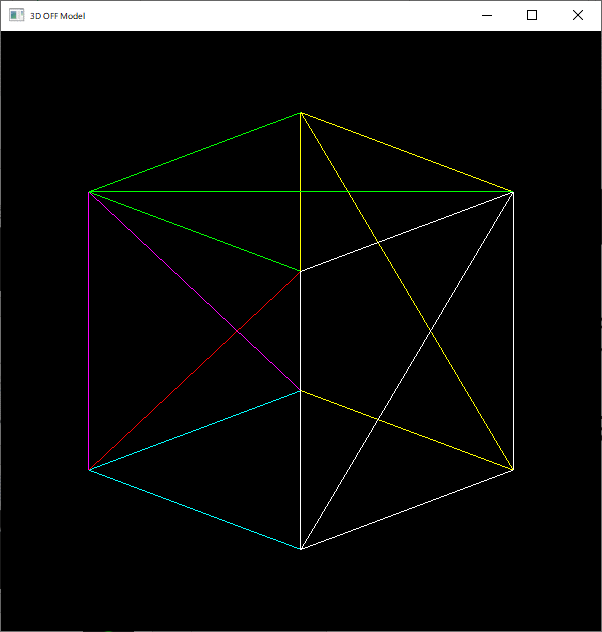


下面展示图中，左图为不开启深度测试而开启面剔除的效果（cube.off）。如果构成面片的点的顺序不统一，那么顺逆时针顺序不统一，无法用面剔除的方法代替深度测试，效果如右图（cube2.off）。



1. 绘制模式

每一个面都可以设置不同的绘制方式：填充、只绘制边缘轮廓线、只绘制顶点，其中“填充模式”是默认的方式。下图展示了使用线模式绘制的立方体。



1. 函数介绍

下面介绍如何开启前面提到的深度测试、面剔除等模式。

1. 设置绘制模式的基本函数

glEnable(GLenum capability)

开启一个模式，mode可以取多种组合。比如要开启深度测试：



glDisable(GLenum capability)

与上面的Enable函数相反，用于关闭模式。

glClear(GLbitfield mask)

清除缓冲。比如同时清理颜色缓存和深度缓存：



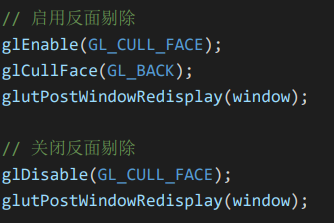
以上函数都可以使用“|”运算符组合不同的模式。

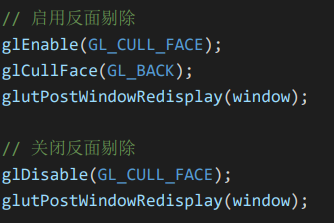
1. 面剔除函数

glCullFace(GLenum mode)

mode指明前面或后面的多边形是否要剔除（就是不显示）。可以使用的标识符有GL\_FRONT、GL\_BACK和GL\_FRONT\_AND\_BACK。初始值是GL\_BACK。这个函数指明当剔除功能启用时，前面或后面的多边形是否要剔除（由mode 指定的那一面）。

可以使用带GL\_CULL\_FACE参数的glEnable 和 glDisable函数来启用和关闭剔除功能。剔除功能默认是关闭的。比如下面的代码例子：





1. 深度测试函数

glEnable(GL\_DEPTH\_TEST);

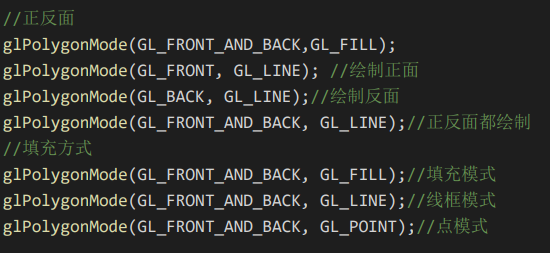
一旦开启深度测试，必须每一帧都清理depth-buffer，比如同时清理颜色缓存和深度缓存：



1. 绘制模式函数

glPolygonMode(GLenum face,GLenum mode)

face这个参数确定显示模式将适用于物体的哪些部分，控制多边形的正面和背面的绘图模式，mode这个参数确定选中的物体的面以何种方式显示（显示模式）。下面展示一段示例代码：



1. 实验内容

实验内容主要是设计并实现读取OFF文件的接口函数void read\_off(const std::string filename)。请修改本实验提供的main.cpp文件，按照下面的说明顺序完成实验

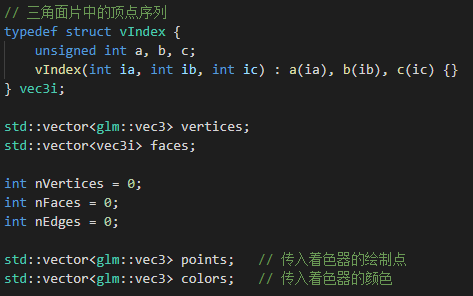
1. 创建工作项目

参考之前的实验流程创建工程，用自己常用编辑器打开项目。

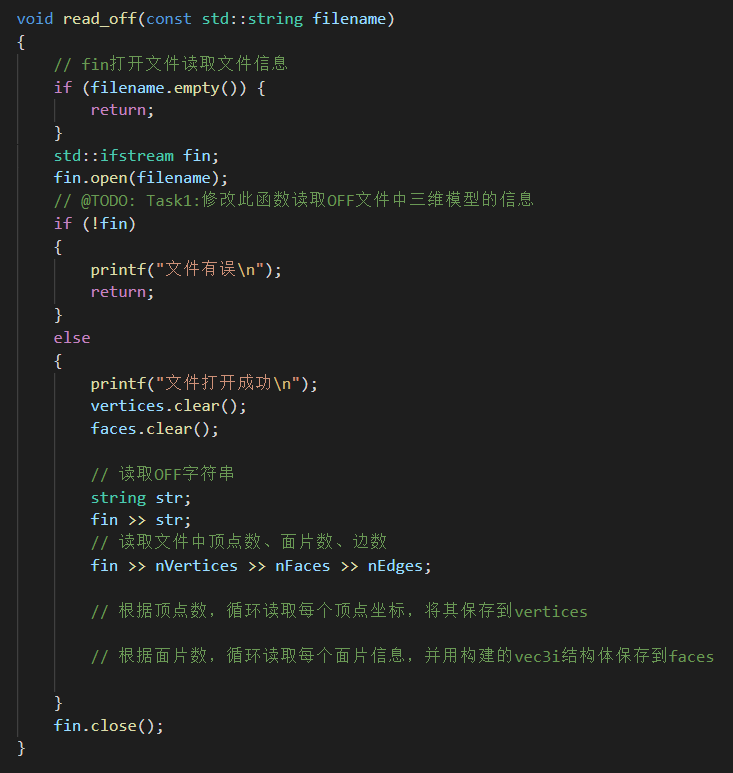
1. 完成read\_off函数读取OFF文件，并存储信息到外部变量中

为了存储顶点下标（正整数），我们先定义了一个vec3i结构体，以及4个向量容器：vertices、faces、points和colors。

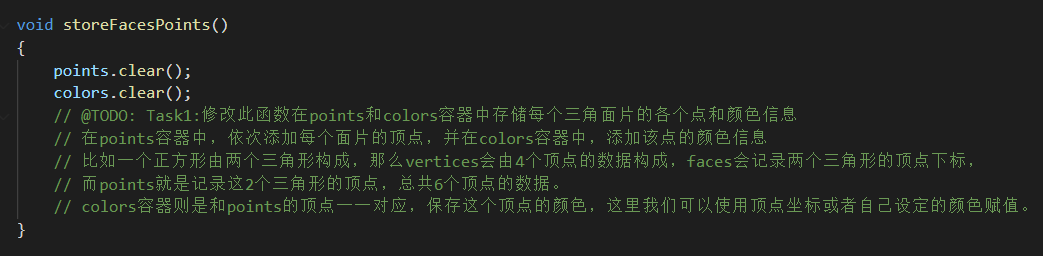
Vertices容器用于存储off文件中的顶点坐标，faces容器存储off文件中的顶点下标。而points是根据vertices和faces数据进一步获得的所有面片上顶点的数据，比如一个正方形由两个三角形构成，那么vertices会由4个顶点的数据构成，faces会记录两个三角形的顶点下标，而points就是记录这2个三角形的顶点，总共6个顶点的数据。Colors容器则是和points的顶点一一对应，保存这个顶点的颜色，这里我们可以使用顶点坐标或者自己设定的颜色赋值，main.cpp开头定义了一些颜色vertex\_colors以供使用。



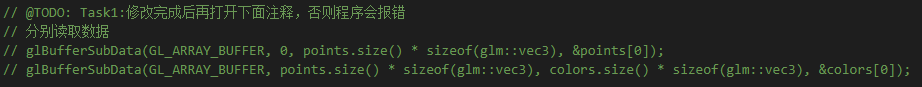
1. 在read\_off()中将读取的顶点数据保存到vertices和faces中：



1. 完成storeFacesPoints函数，存储用于着色器中顶点和颜色信息



1. 在init()和key\_callback ()中启用被注释的代码

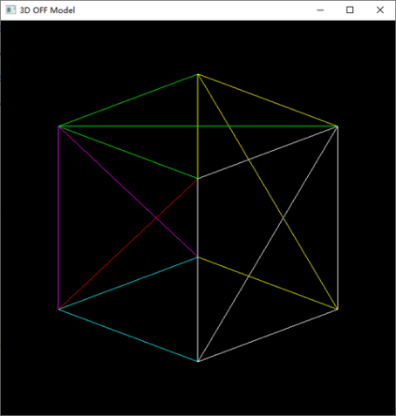
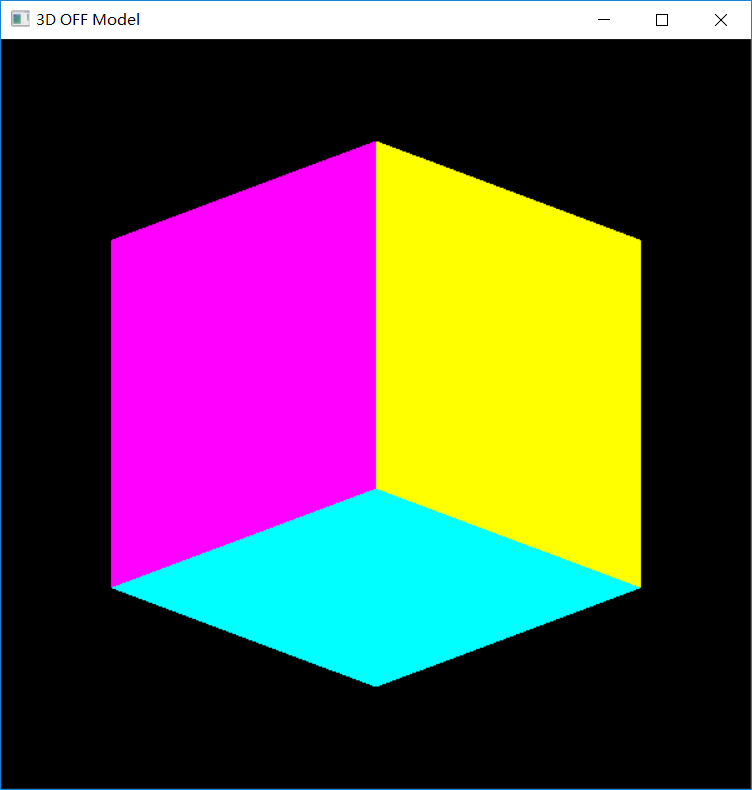
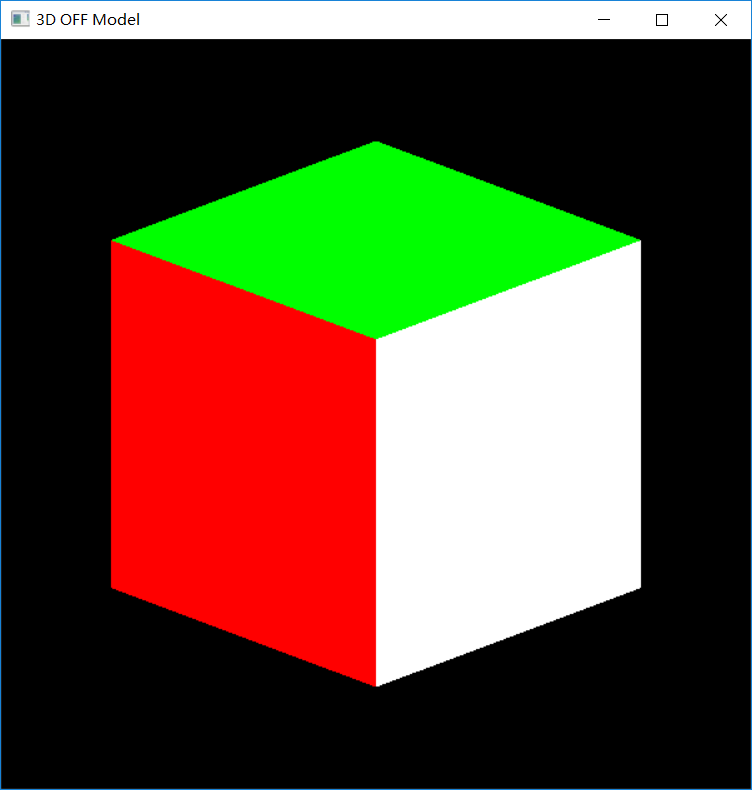


1. 参考前面提到函数控制深度测试、面剔除和绘制模式的切换

在main.cpp中，display()、key\_callback ()、main()中找到@TODO标记的位置，根据注释补充代码。

1. 课堂练习
2. OFF模型读取，复制顶点数据到OpenGL缓存
3. 开启窗口的深度测试和深度缓存清理。实现按键1和!分别控制深度测试的开启和关闭。
4. 实现按键2和@分别控制背面面片的剔除和恢复
5. 实现按键3和#分别控制正向面片的剔除和恢复
6. 实现按键4和$分别控制线模式的启动和关闭

最终得到下面的结果：开启深度测试或者背面剔除、开启正面剔除、开启线模式



如果为每个顶点赋上不同的颜色可以得到一个插值的彩色立方体

