

实验5

实验名称Transformation变换2

姓 名：雷昱

学 号：22920202204666

学 院：信息学院

专 业：软件工程

年 级：2020级

二〇二二年 **5** 月 **7** 日

建议阅读资料：

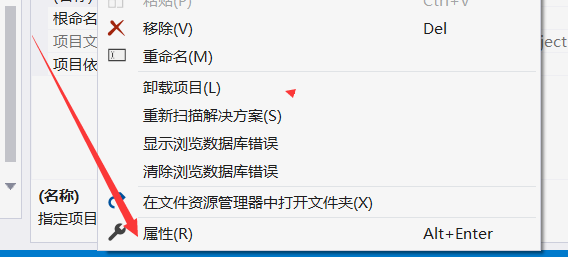
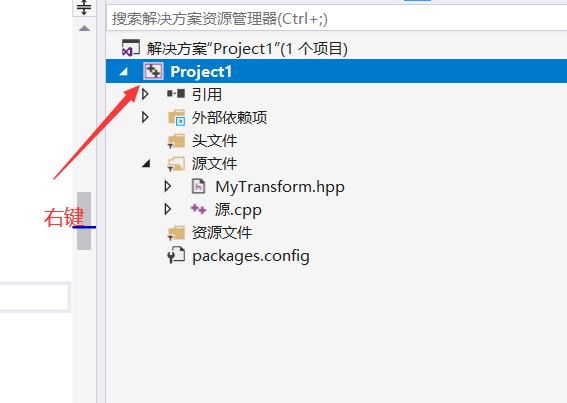
1. 课件
2. OpenGL编程基础 （第三版）

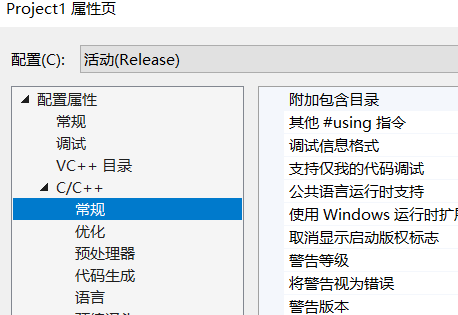
学习要求：

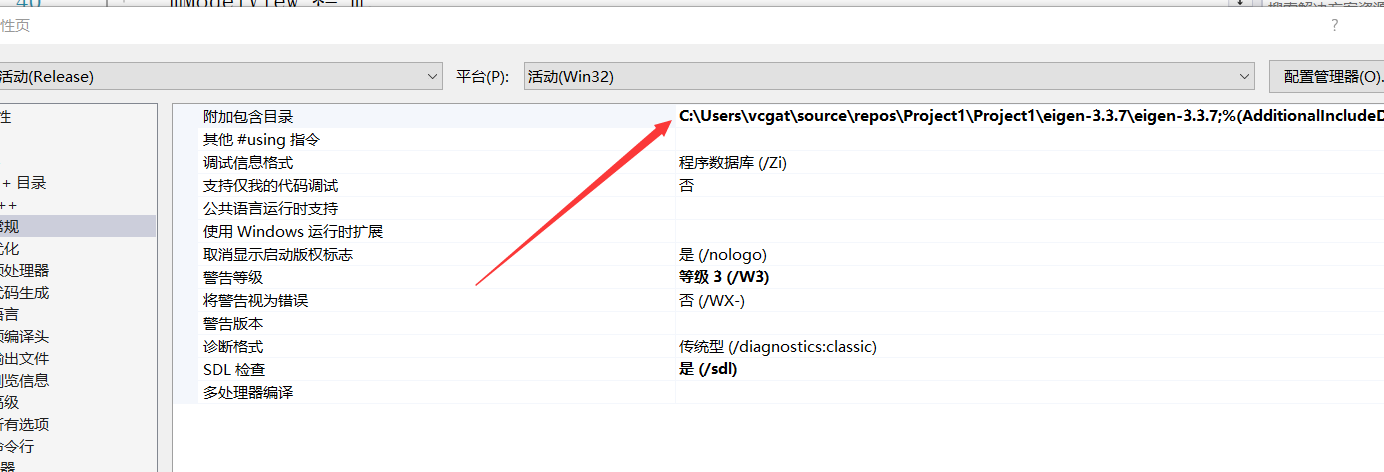
1. 掌握gluLookAt对应的矩阵，能自己实现OpenGL中gluLookAt功能
2. 掌握glOrtho对应的矩阵，能自己实现OpenGL的glOrtho功能
3. 掌握glViewport对应的矩阵，能自己实现OpenGL的glViewport功能

**Task1. 自行实现视点、投影、视口函数，请严格按照以下要求实现：**

1. **创建空工程，配置freeglut库（之前已经教会大家）**
2. **配置eigen库（eigen用于矩阵和向量的计算）。做法是：将eigen库解压，同时在刚才创建的工程中设置eigen库路径。右键点击工程->属性->C/C++ 常规 -> 将eigen所在目录填到“附加包含目录”这一栏。**

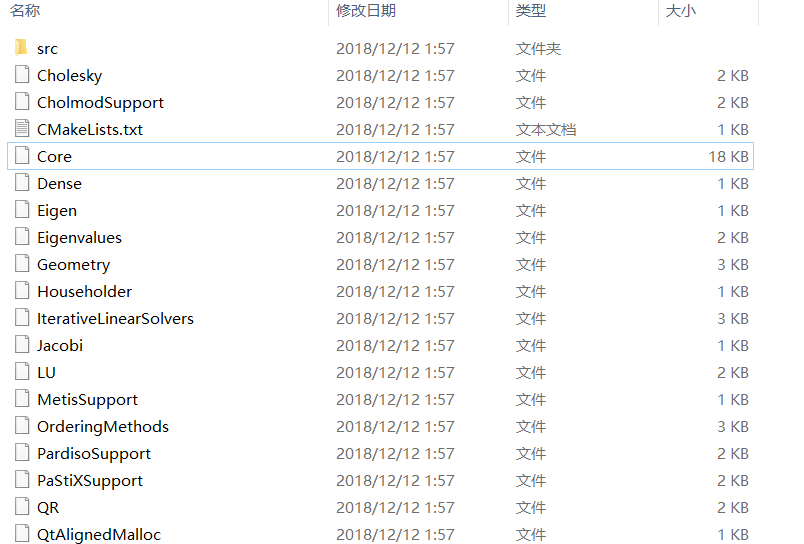






**注意：上述路径是我的路径，请根据自己的解压位置填写路径。**

**应该定位到这个目录的上一级：**



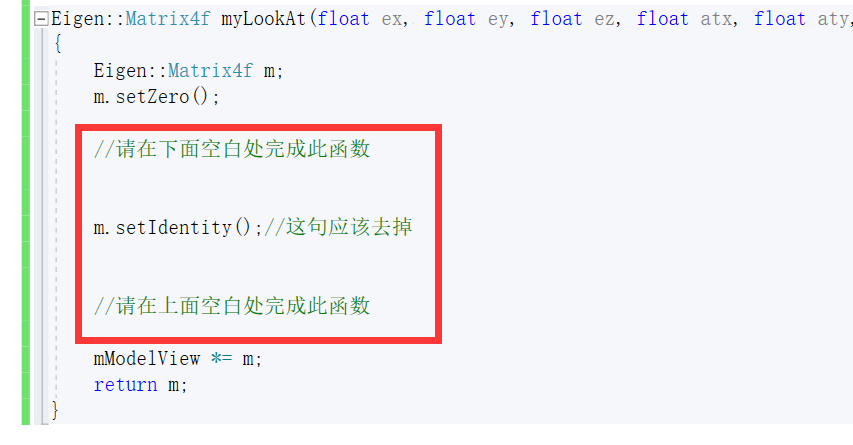
1. **将本次实验给大家的两个代码文件加入工程中，它们是exp5.cpp和MyTransform.hpp。在本次实验过程中，除非特别说明，不允许改动exp5.cpp的内容（也不必关心具体代码）。**
2. **不做任何更改，如果配置正确，你应该能看到一个旋转的彩色三角形。**
3. **在MyTransform.hpp中，请自行阅读并自己修改void Test()函数。此函数仅用于让你熟悉eigen中矩阵和向量的用法，你可以自己随意尝试。我们提供了矩阵、向量设置和乘法的基本代码，其他更多函数可参考eigen官方文档。**

<http://eigen.tuxfamily.org/dox/group__QuickRefPage.html>

1. **完成MyTransform.hpp中如下函数：**

myFrustum，myOrtho，myLookAt，myViewPort

**要求仅在代码中提示的地方填入/修改代码，其余地方均不许改动！**



**不许引入任何头文件，不得改动函数原型。程序正确结果应该是一个旋转的三角形动画。**

**如果配置成功，但还没开始填入代码，您应该看不到任何图形。**

**说明：请按照myFrustum，myOrtho，myLookAt，myViewPort这个顺序完成代码填写。**

**Task1.1 将 myFrustum，myLookAt，myViewPort填写正确，运行exp5.cpp查看myFrustum的结果。**

**Task1.2 将myOrtho，myLookAt，myViewPort填写正确，在exp5.cpp中注释掉第56行，取消第57行的注释，观察myOrtho的效果；**

**所有代码都填写正确的话，您将看到两个三角形在窗口中做旋转。**

**标准程序结果已放在压缩包中。frustum.exe、ortho.exe分别展示了frustum.exe和ortho.exe的标准结果。（注：若无法复现标准结果，可将自己的结果和代码私发给助教，助教将评估已有结果和发送的结果哪个正确。若能够成功辅助助教老师更正结果，可以给予优秀作业。）**

**本次Task需提交2个可执行文件，Task1.1为使用myFrustum的结果，Task1.2为使用MyOrtho的效果。**

**作业提交说明：**

本次实验共有2个任务，分别是Task1.1，Task1.2。

提交方式为：*将代码、可执行文件、实验报告*放到一个文件夹中，命名为  
“您的学号\_姓名”，打包上传到ftp服务器中相应目录下。请确保提交的可执行文件可以运行（打分的重要依据）。每次实验作业的提交截止日期为下一次实验课前一天晚上。

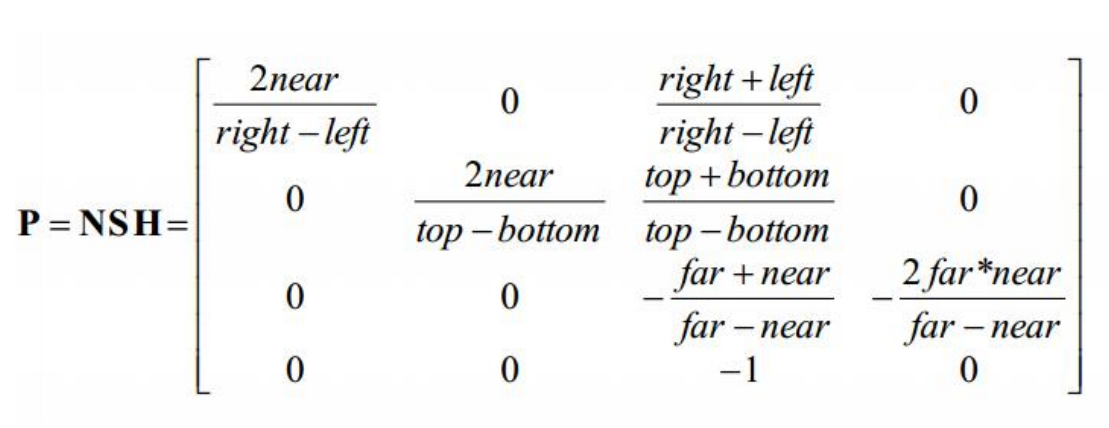
特别说明：本次实验2个任务（Task1.1，Task1.2）都需要提交。您需要提供一个完整的文档（不限格式），逐条说明您完成每一条任务的具体情况。

实验步骤如下：

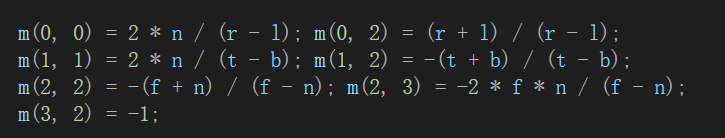
1. myFrustum函数实现

①透视变换转成正交变换

②矩阵相乘计算后myFrustum矩阵所示

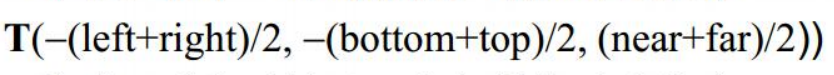


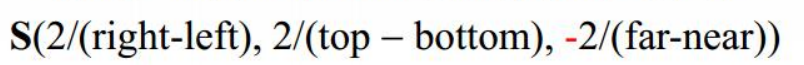
③代码实现



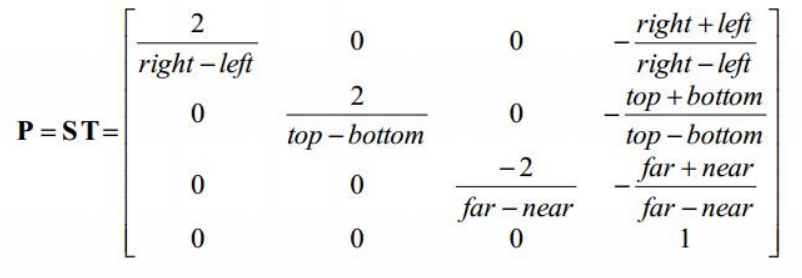
1. myOrtho函数实现

①先平移，把中心移到原点

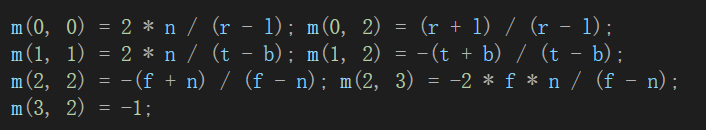
 ②再缩放，进行缩放使视景体边长为2



③矩阵相乘计算后myOrtho矩阵所示



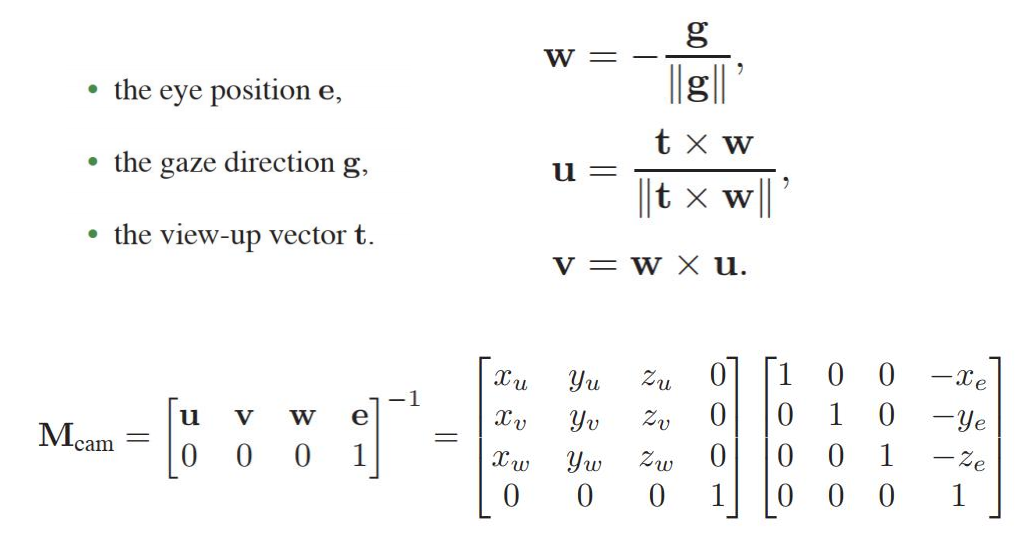
④代码实现



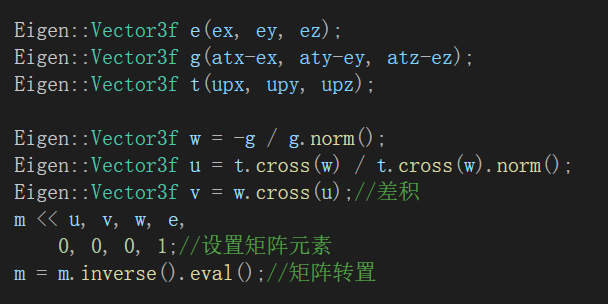
1. myLookAt函数实现

①相机变换

②myLookAt矩阵所示

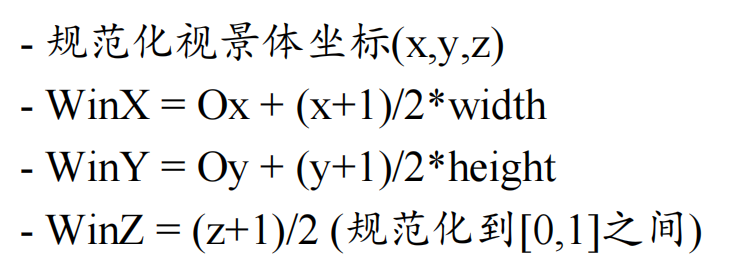


③代码实现



1. myViewPort函数实现

①根据规范化视景体计算

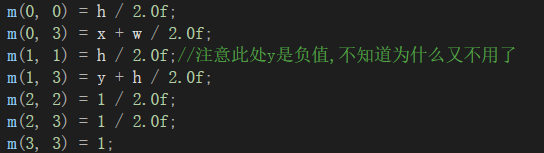


注意此处WinY坐标和窗口系统的坐标朝向是反的

②myViewPort矩阵所示

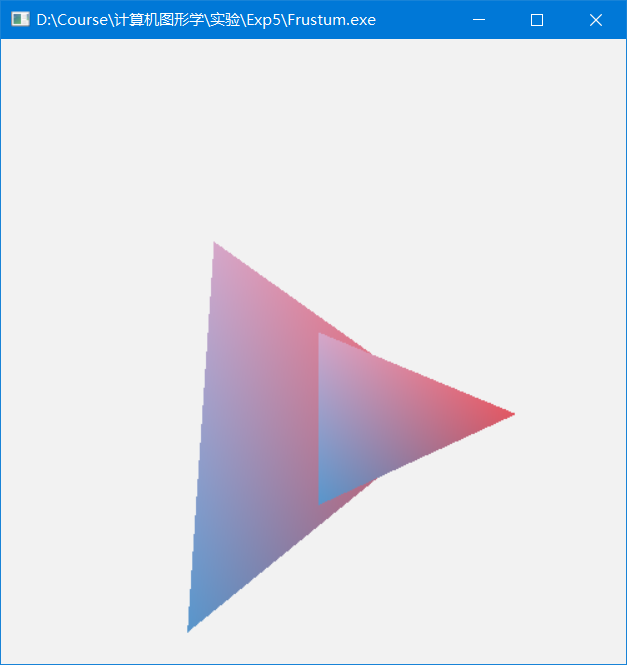
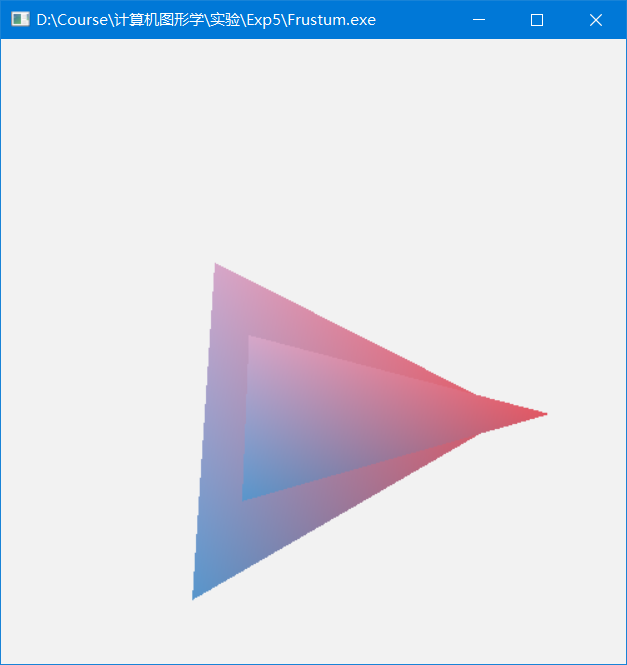


③代码实现



1. 结果展示

myFrustum



myOrtho

