# 说明文档

## 实验环境说明：

硬件环境为AMD 3080 CPU，

软件环境为CLion软件，C++编程软件。

## 用户界面使用说明

参见UI使用说明.docx

## 数据结构说明

#### 基础数据结构:

datatype/data.h：我自己构建数据分析库的一部分，主要负责生成大型的矩阵，在本项目中使用了该文件中的Lmatrix数据结构，负责Buffer数据的生成。

datatype/Model.h, datatype/Project.h, datatype/view.h,分别代表model矩阵，project矩阵，view矩阵。

datatype/viewport.h,：因为project之后的节点数据（-1，1）之间，将这些节点数据转换为窗口坐标的矩阵。

datatype/pipeline.h: 为了在我的代码中体现pipeline的思想，所以我使用该pipeline，统一管理以上所有矩阵，并处理vertex数据，相当于vertex shader的作用。 这样即方便使用openmp优化，又方便处理。

Vec.h: 由于在光栅化的过程中，出现了大量的4x4 matrix， 4x1 vector,因此我写了个数据结构专门对这些计算做并行的优化（使用openmp）。

以上是基础的数据结构，我并没有使用任何数据计算的库，纯自己造了一个。 我自己的数据分析库的地址为<https://github.com/lidan233/MyEigen>，目前已经实现了LU矩阵迭代分解，高斯消元法，以及附带得多种trick，支持迭代求解矩阵。

为了方便多种数据类型的光栅化

geom/line.h, geom/Point.h, geom/triangle.h 专门负责这些数据结构的光栅化算法。

util文件夹中存在多个ZBuffer的数据结构以及Octree等，名字非常明显，这里不做赘述。

#### 加速

因为数值分析库是我自己的，基础数据结构也是我自己的，所以我针对数据结构的计算使用openmp做了并行计算。有两点值得说一下：

1. 使用cmake获得处理器数目，自动调节加速线程数目。
2. 使用pipeline的思想,在vertex处理过程中，在矩阵计算过程中，在基础计算，插值计算等计算过程中，均使用openmp做了加速，最大程度上避免了锁的使用。