# 1 总体架构



## 1.1用户界面对应的文件

mainwindow.h

主窗口

CentralWidget.h

中心Widget，包含TabWidget、GLWidget、StrechLabel

TabWidget.h

标签页Widget，包含ObjTreeWidget，TabBackground

ObjTreeWidget.h

显示树状结构的Widget，这里只用于显示数据中的band的id，诸如band0，band1之类。

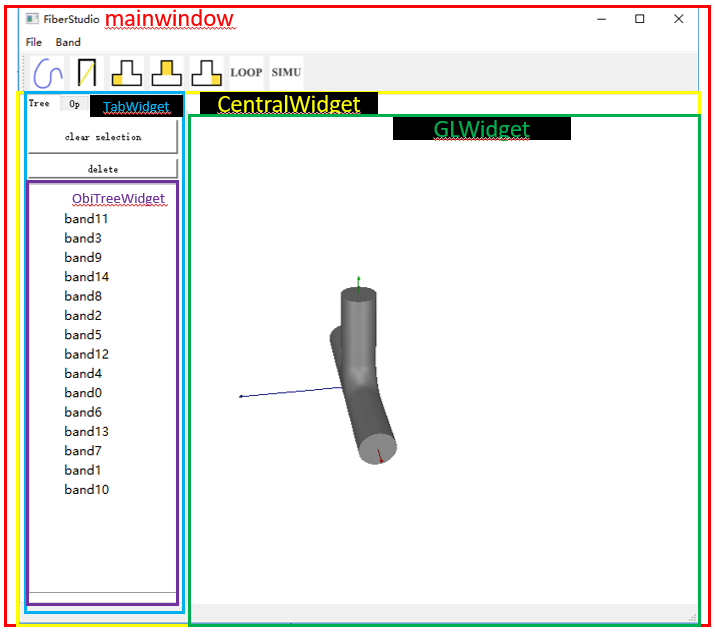


图1 用户界面说明图

其余用户界面文件不作说明。

## 1.2 控制器对应的文件

Controller.h

实现了各种功能。

**鼠标的对模型的拖动、旋转、缩放相关**

void processTranslation(QPoint mPos, QPoint mLastPos, glm::vec4 viewPort);

void processRotation(QPoint mPos, QPoint mLastPos, glm::vec4 viewPort);

void processScroll(double yOffset);

**添加设计点相关**

std::vector<HalfPoint> intersectionPointInTee(glm::vec3, glm::vec3);

std::vector<HalfPoint> intersectionPointInPipe(glm::vec3, glm::vec3);

QString addIntersectionPointInTee(glm::vec3 begin, glm::vec3 dir);

QString addIntersectionPointInPipe(glm::vec3 begin, glm::vec3 dir);

void processIntersectionPoint(glm::vec3 begin, glm::vec3 dir, glm::vec2 glXY);

**生成直管轨迹**

void Controller::genCylinderCurve(QString which)

void genLeftCurve();

void genUpCurve();

void genRightCurve();

AnimateController.h

负责动画控制相关。

FromPipeCurveController.h

在直管上添加完设计点后，生成的轨迹会在T形结的边界上有交点。此文件根据这个交点生成四个对称的T形结轨迹。

NewCurveController.h

根据T形结上的设计点，生成四个对称的T形结轨迹。

**各类文件中的功能会有用不到的，但是没有删除。请自行斟酌。**

**“控制器”的概念并未全程贯穿。因此有些用户界面文件的代码包含“业务逻辑”，这实际上违反了MVC原则。请自行斟酌。**

## 1.3数据对应的文件

Data.h

最重要的文件，数据全在这个类中。

包括：

std::shared\_ptr<Camera2> camera;//相机的位置和姿态

std::shared\_ptr<Node> root;//对象树的根节点

BoundingBox box;//包围盒的大小

DataState state;//作用不大，用于更新UI按钮

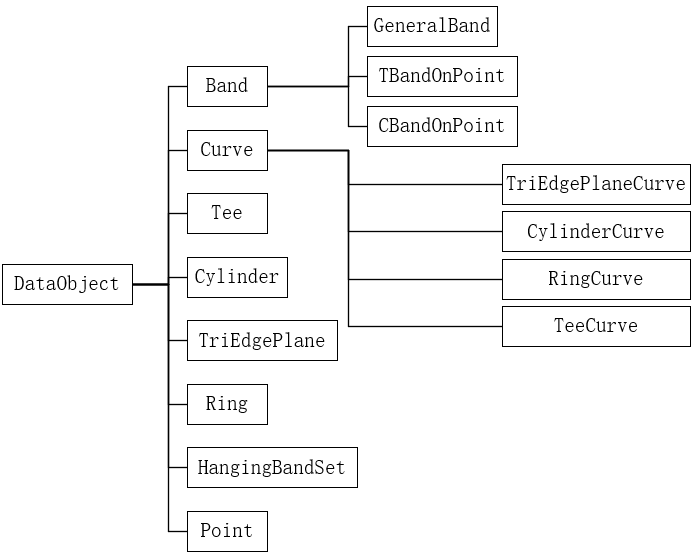
AxisIni axiss;//缠绕机配置

EnvelopData envelop;//包络面数据

float m\_bandWidth;//带宽

IdGenerator idGenerator;//负责记录id编号

# 2 数据类型的层次关系



# 3 头文件简要说明

AnimateController.h

控制动画播放进度

AxisConfigDialog.h

用于新建或导入缠绕机配置

AxisIni.h

记录缠绕机配置，包括各轴的名称和各轴的偏移量。

Axiss.h

用于在GLWidget中显示XYZ轴。

Band.h

轨迹（含带宽）的基类，包括两个端点（BandEnd类型），两个边界（vector<CurvePtr>类型），轨迹的位置和切向量（PosDirVec类型）以及法向量（DirVec类型），QStringVec m\_strs表示轨迹点处于哪个面片之上。

boundingbox.h

定义了包围盒，包围盒主要用于确定旋转中心。

Camera2.h

定义了相机的位置姿态，开角，平行光源。可通过查阅OpenGL原理了解。

CBandOnPoint.h

全称是CylinderBandOnPoint，由直管上的设计点所生成的Band。

CentralWidget.h

中央Widget

closePathDialog.h

用于显示连接的进度。

closePathThread.h

用于连接轨迹。

Color.h

颜色类，被用于定义各显示要素的颜色。

config.h

用于定义着色器路径，不同电脑上的配置不同。

Controller.h

包含了主要的“业务逻辑”。

CPPara.h

包含CPPara类和SPara类。CPPara类是curve point parameter，包含轨迹点的位置u,v和缠绕方向uAng。SPPara类是surface parameter，只有两个参数坐标u,v。

Curve.h

轨迹的基类

CyCurveAssist.h

圆柱面轨迹的辅助类

Cylinder.h

圆柱面类

CylinderAssist.h

圆柱面辅助类

CylinderCurve.h

圆柱面轨迹类

Data.h

最主要的数据类，包含所有的数据。

DataObject.h

很多类的基类。

DataState.h

当Data类的对象被更新时，需要更新DataState的状态为changed。刷新UI时，需要查询DataState的状态。（不是很重要）

Edge.h

面片的边界类

EnvelopAssist.h

包络面的辅助类

EnvelopData.h

包含包络面的数据

EnvelopIniDialog.h

包络面的配置对话框

FromPipeCurveController.h

完成如下业务逻辑：在直管上添加完设计点后，生成的轨迹会在T形结的边界上有交点。此文件根据这个交点生成四个对称的T形结轨迹。

FromPipeCurveTab.h

在直管上添加设计点的标签页。

GenCurveProgressDialog.h

由T形结轨迹生成直管轨迹的的进度对话框。

GeneralBand.h

最一般意义的Band。

GenGeneralBandThread.h

生成GeneralBand的线程类。

GLMemory.h

主要为了提供OpenGL显示的基本元素。

GLProgram.h

封装的着色器程序。

glwidget.h

显示OpenGL环境的窗口。

GuiConnector.h

很重要，GUI内部Widget之间的互相调用，通过此类完成。

HalfPoint.h

包含点的三维位置和属于哪个面片的信息。

HangingBandSet.h

悬纱相关。

IdGenerator.h

用于给出id的编号。

LeftCylinderAssist.h

用于生成连接轨迹的辅助类。（名称中的Left不是很合适）

LeftRibbonLayout.h

用于CentralWidget的layout。

LineLight.h

定义了平行光。

mainwindow.h

主界面。

Mesh.h

很重要，最基本的显示片。

ModelReader.h

用于读取STL文件

MoveData.h

定义了后处理中的解算数据。

MoveDataTab.h

略

NameDeclear.h

包含大部分类型的声明。

NewCurveController.h

完成如下业务逻辑：根据T形结上的设计点，生成四个对称的T形结轨迹。

NewCurveTab.h

略

Node.h

对象树的节点。

numpy.h

一些计算相关的函数。

ObjModel.h

略

ObjTreeWidget.h

略

OpenBandThread.h

导入Band的线程类。

PerspectiveMat.h

和相机相关。

Point.h

定义了设计点。

PointSym.h

定义了对称设计点。

PosDir.h

定义了PosDir类型，包含位置和方向的三维坐标。

Ring.h

圆环类

RingAssist.h

圆环类的辅助类

RingCurve.h

圆环面上的轨迹

RingCurveAssist.h

圆环面上轨迹的辅助类

SimulationTab.h

仿真的标签页。

StrechLabel.h

鼠标拖动此Label可以延伸TabWidget。

TabBackground.h

略

TabWidget.h

略

TBandOnPoint.h

T形结上的Band

TCurve.h

T形结上的轨迹。

TCurveAssist.h

T形结轨迹的辅助类。

Tee.h

三通管

TeeCurve.h

三通管上的轨迹

TeeCurveAssist.h

三通管轨迹的辅助类

TeePara.h

三通管的参数

TeeParameterDialog.h

三通管的参数化对话框

TreeItem.h

略

TriCurveAssist.h

平面轨迹的辅助类

TriEdgePlane.h

平面

TriEdgePlaneAssist.h

平面的辅助类

TriEdgePlaneCurve.h

平面轨迹

utility.h

包含许多常用功能

vertex.h

定义了vertex类型，主要用于OpenGL中。

meshFragShader.fs

片段着色器。

pointFragShader.fs

没用到

vertexShader.vs

顶点着色器

# 4 存在的问题

## 4.1 带宽显示问题

带宽的显示是通过中心轨迹向两个方向偏移半个带宽实现的。偏移实际上是计算测地距离。由于采用的是数值算法，计算会有偏差，导致在面片的交汇处的带宽显示存在问题，如图4-1所示。

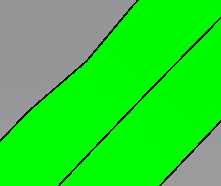


图4-1 带宽显示问题

## 4.2 模型统一问题

实际缠绕时，存在后处理模具比轨迹设计模具更长的需求，如图4-2所示。而目前软件中轨迹设计和后处理所用的三通管模型一致，这会造成不便。目前，只能先设计轨迹，然后保存。再之后根据后处理要求，修改模具尺寸。再将轨迹导入，进行后处理。

后续工作中可以为后处理模具单独设置一个模型。

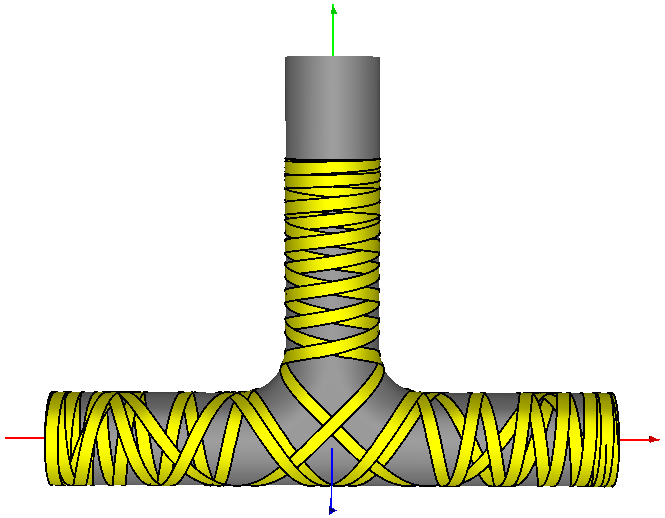


图4-2 更长的后处理模具

## 4.3 栈溢出问题

规划轨迹时，有时会出现栈溢出问题，即无穷次的调用，导致栈的空间不够。原因应该是在边界判断中出现了问题，解决方法有赖于进一步的分析。可通过降低轨迹点之间的距离减少此问题的出现频率，但代价是计算时间变长。