# 钻孔视觉软件开发周报

## 2023.5.1-2023.5.7

### 工作进度

* + 1. 进行了代码架构的解耦优化（将控件从界面相关的结构中分离出去），目前为止软件的主功能框架基本已覆盖，并实现了部分数据流与绘制工具控件的功能；
    2. 目前大部分代码已提交到git（develop分支）：

[https://github.com/BRUHARE/ProbeVision](https://github.com/BRUHARE/ProbeVision/tree/develop)

绘图控件和封装后的算法在持续提交中。

\*目前仓库权限为Public，后续有必要修改为私有仓库。

### 风险预警

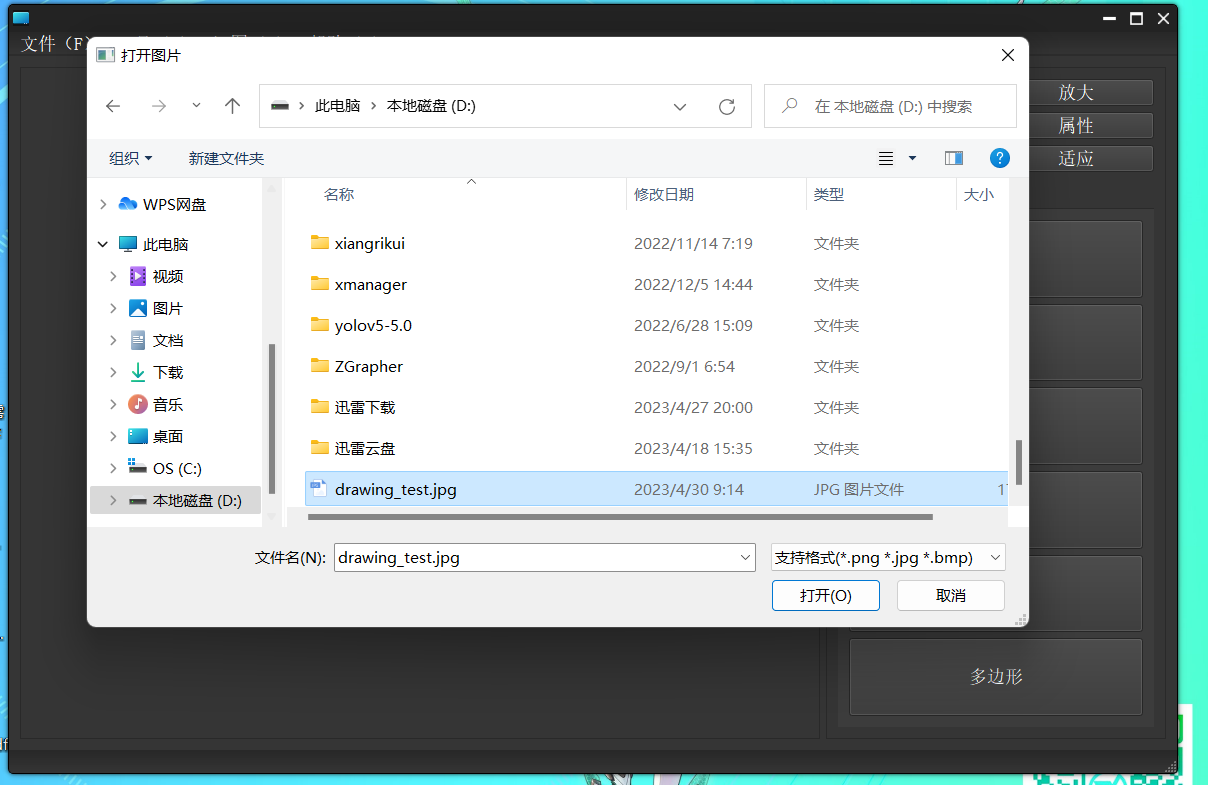
无

## 2023.4.24-2023.4.30

### 工作进度

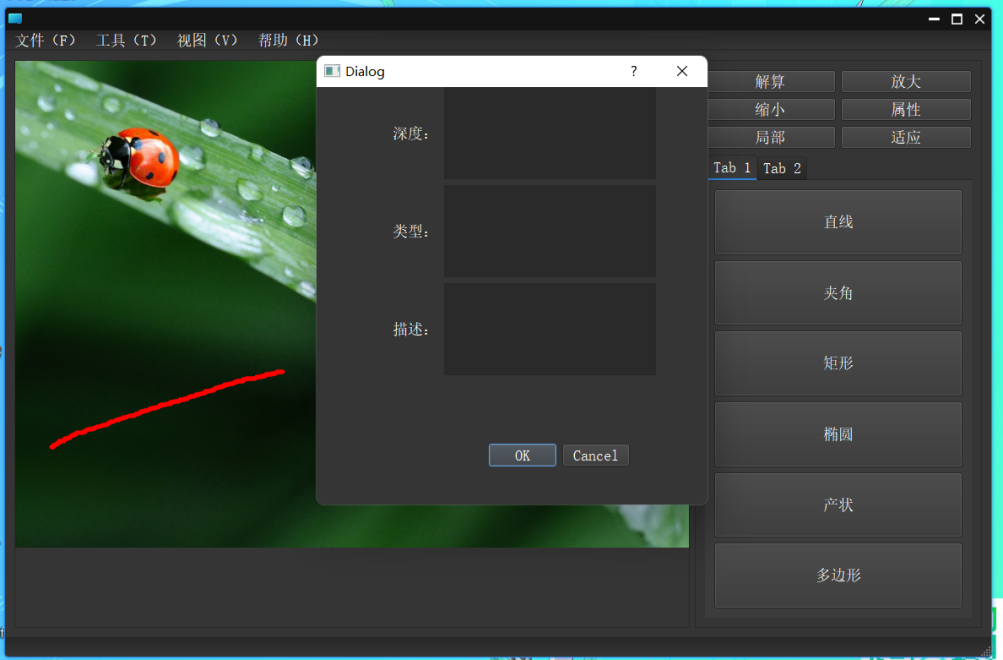
* + 1. 进行标注绘制功能的设计与开发：
* 框架中新增以下组件：data\_manager（原始数据的加载、现有工程的导入）、主画板以及对应的标注笔控件（直线、夹角、椭圆等）、标注信息（数据结构以及相关的序列化与反序列化）；具体的接口内部实现正在开发中，还未完全实现
* 底图绘制标注信息的设计：使用两层图层，底图为拼接结果，上层为标注信息，标注信息每次会先从文件中间下载，再重新绘制成上层的图片覆盖在底图上；
* 部分绘制功能的实现：
  + 导入工程文件（目前暂时是图片，后面会把标注文件和底图文件合并成一个单独的工程文件）：







* + 绘制标注图形，松开鼠标后自动弹出标注保存窗口，保存后标注信息进行落盘：



* + 组件架构还有一些小调整，完成后这两天会将代码提交Git。

### 风险预警

无

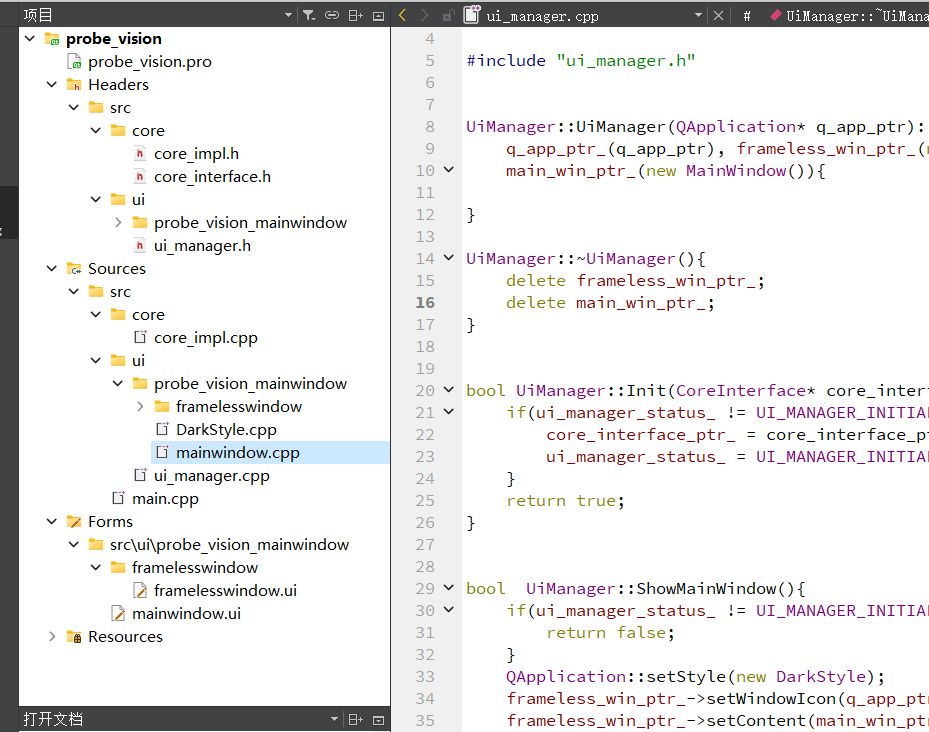
## 2023.4.17-2023.4.23

### 工作进度

* + 1. 更换了qt自带的主界面，改成了无边框主窗口，并修改优化了界面以及控件的风格：



* + 1. 已将新的ui实现加入到框架中，正在进行主业务框架的实现，这一周会实现完初步的画板标注功能：



### 风险预警

无

## 2023.4.9-2023.4.16工作周报

### 工作进度

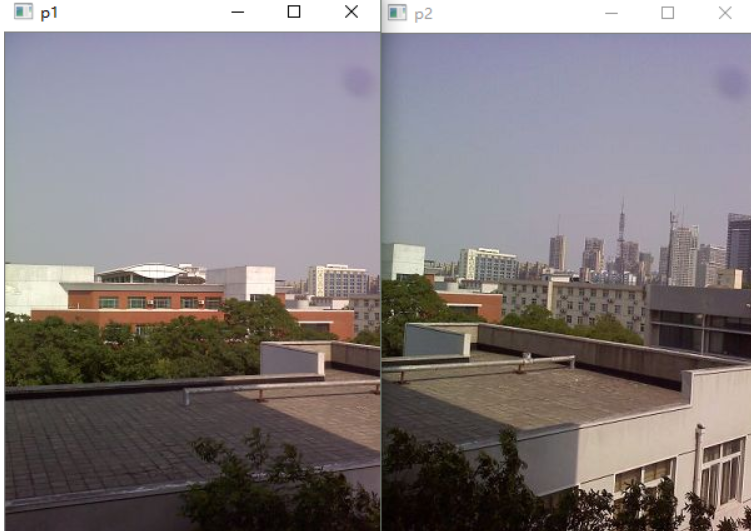
* + 1. 拼接算法调研

经调研后会封装两种拼接方式：

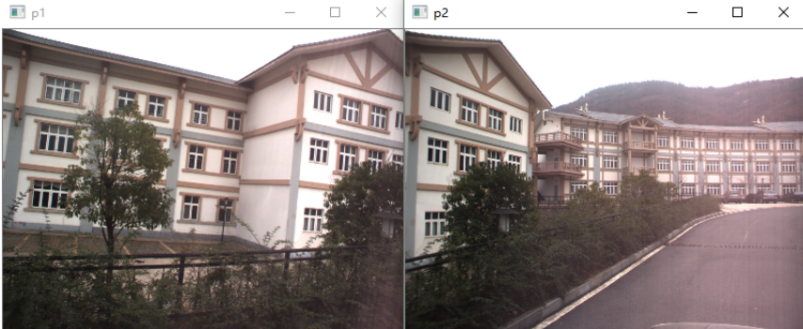
* 根据特征点匹配进行拼接

大概流程如下：

* + 提取特征点，可以选择提Surf性能和效果比较均衡；
  + 计算特征点的描述子；
  + 使用knnMatch进行描述子的匹配，选出可靠的匹配对；
  + 根据匹配对计算最佳单应矩阵（H阵），opencv中有findHomography接口实现；
  + 将其中一张图（选新加入的图）使用H阵做透视矩阵变换，得到了目标透视图；
  + 将第一张原图拷贝到透视图上即可（提前计算好目标图片的尺寸，同时拼接后的图需要做去裂缝处理，效果会更自然）；









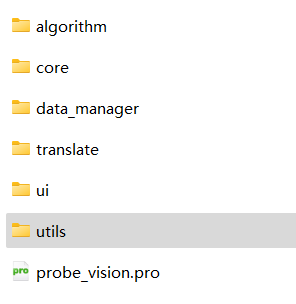
* 使用opencv提供的stitch算法进行拼接（未测试过，看网上的评价效果是较好的，但是拼接速度慢）





可以看到拼接的结果有黑边，这里要单独处理

* + 1. 软件框架搭建



* Algorithm：包含图像展开算法，拼拼接算法，三维重建算法；
* Data\_manager：管理数据流，包括视频文件的加载，工程文件的封装（包含拼接后的图片以及图片附带的标注信息），工程文件的序列化与反序列化；
* Translate: 多语言文件，用于支持多语言切换；
* Ui：界面控件管理，与业务与算法解耦，只负责显示，并与其他模块进行交互；
* Utils: 工具模块，提供诸如单例、模板工厂等公用类；
* Core：主业务模块，负责实例化各个模块，并串联业务逻辑.
  + 1. Ui功能开发

进行了控件的选取与布局器的设置,保证窗口变化时各控件进行自适应,还未进行美化(icon和背景都还没有进行设置)



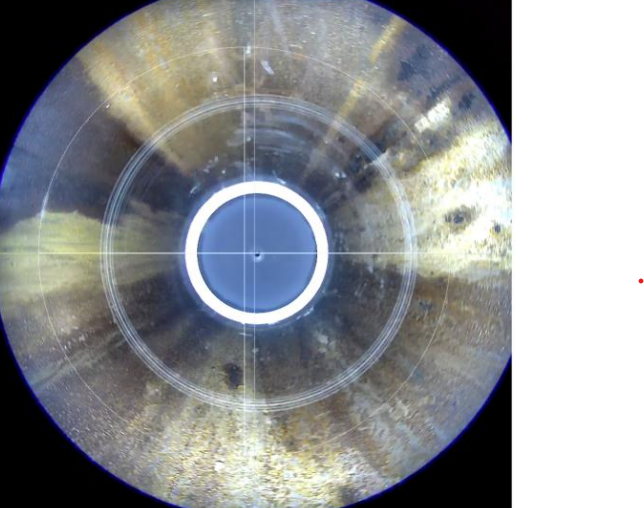
### 风险预警

无

## 2023.4.1-2023.4.8工作周报

### 工作进度

* + 1. 鱼眼相机展开算法实现（已完成了一版），下面是一些细节
* 源数据：根据目前可供参考的钻头相机数据，应该是做一个圆环转矩形的操作：



探头数据

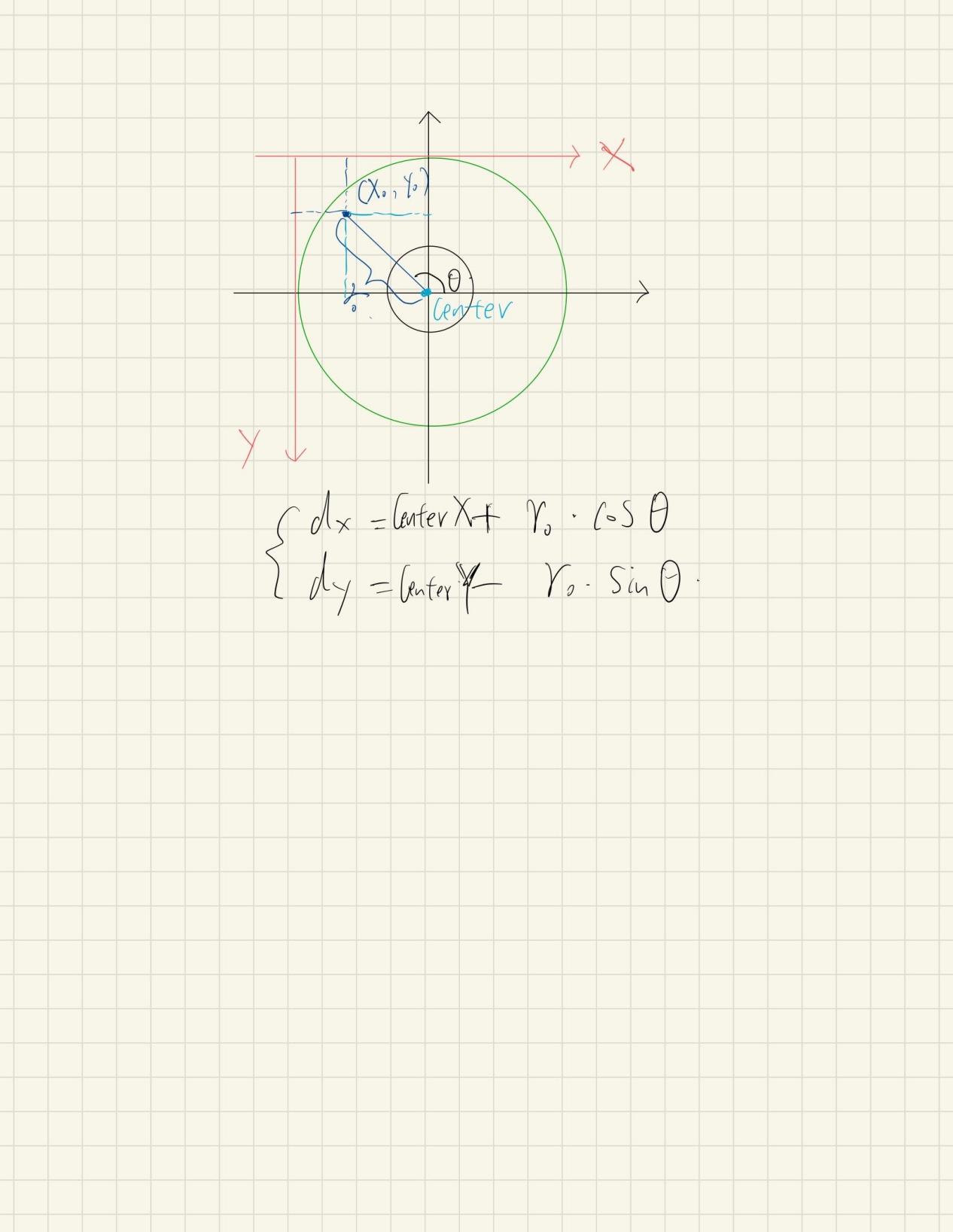
* 转换算法：

使用霍夫圆检测出鱼眼相机的有效区域：



提取该有效圆区域，预处理结束；

* 目标矩阵大小为：圆周长\*圆环宽度（中间不取同心圆时为圆半径），将圆环从最外层向内遍历，填满目标矩阵的每一行，遍历时通过将极坐标（原点为霍夫圆的中心）转为直角坐标，可以找到输入数据的对应坐标从而取到像素值：

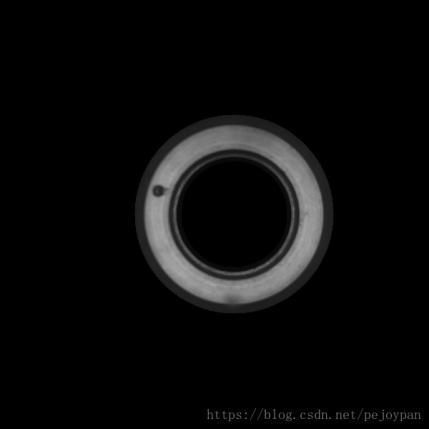


同时，由于越靠近圆心的位置畸变程度越高（信息丢失程度高），采用双线性插值的方式进行补充。

* 展开效果如下（图片为网上找到的资源，公开数据集中没有找到俯视纵深类型的数据）：

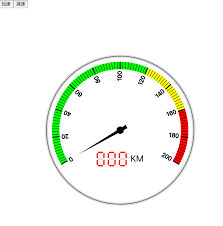


* 钻孔数据是一个更规范的同心圆场景，因此转换出来效果会更规范，类似于一下两个例子：



转为

20181016134501246



转为

d9d7ba973eed4da4bdcc4407b351de79

同时，约靠近圆边缘的数据效果越好，因此对于钻孔视觉的场景会在中心制作一个大小合适的mask，只保留效果较好的数据，mask的大小需要拿到真实数据后来进行设计。

* + 1. 梳理了QT框架的核心机制，搜集整理了一批控件库，代码框架的设计进行了一部分，预计再有一到两周完整代码框架设计。

### 风险预警

无