1. 整体需求

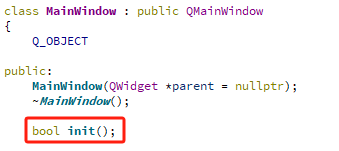
根据模块的功能需求，完成UI控件相关的功能，并给出对应的槽函数等接口。需要使用ui文件进行设计。

1. 整体代码规范
2. 命名方式：

* 不使用命名空间（using namespace），直接std:: …
* 命名要能体现该变量/函数结构体/类的作用
* 变量名：img\_src; （小写，\_分隔）
* 函数名：getTempData; （首字母小写，后续字母大写）
* 结构体\类名：WidgetDevice; （全部字母大写）

1. 模块化：

* 每个UI界面单独一个模块。
* 同时每个UI需要预留一个init()函数接口。



1. 注释：

尽量多注释，后续我们需要在槽函数中调用接口。

1. 编译环境：

MSVC，win10

1. 布局：

页面大小在1500\*900左右。不能太小，但也不能超出1080p屏幕大小（1920\*1080）。可先固定大小，后续如果可以做伸缩更好。

1. 时间节点：

完成时间大约在10月27日左右，如果来不及，可优先完成功能，样式可后续完善

界面开发顺序：设备调试——首页——温度检测——检测结果——数据库管理

1. 入参：你从UI控件读取数据给我

出参：我给你数据拿去UI显示

Qss可以借鉴resource文件夹里面

1. Github仓库：

<https://github.com/XCmafu/pcb_ui.git>

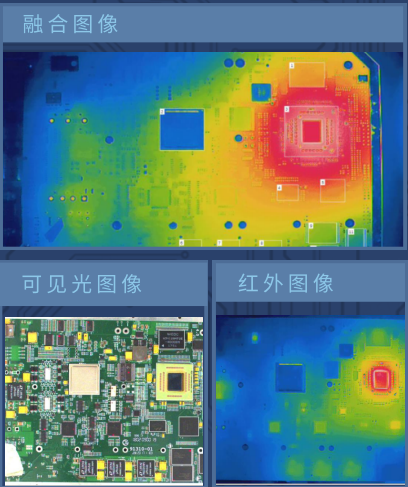
基本代码在basic分支

每增加一个新UI，就上传到一个新的分支，分支命名“update-月份-日期-新增ui名”

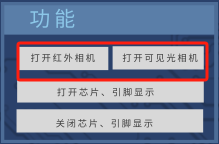
1. 首页
2. 三个图像显示部分：

原始图像均为opencv cv::Mat格式，可以转化成其他格式后（QImage）显示。同时这三幅图像需要双击放大。

原始图像接口每一次只能获取一张图像，所以该模块开启显示后，应该需要在循环中调用接口不断更新显示。



通过按钮进行打开和关闭图像显示，融合图像开始显示按钮是“开始故障检测”：

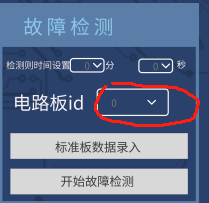


打开/关闭芯片引脚显示（可以换成勾选框）：区别是调用的后端接口不一样，但图像数据都是cv::Mat格式，所以需要给我一个标记让我知道调哪个接口。

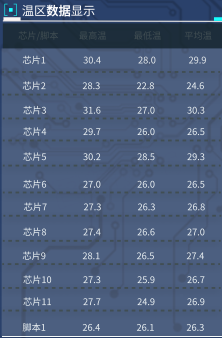
1. 故障检测部分：

入参：时间和电路板id（这里电路板id示意图画错了，不是combobox，应该是spinbox）

点击两个按钮都需要得到以上入参



1. 温区数据显示：

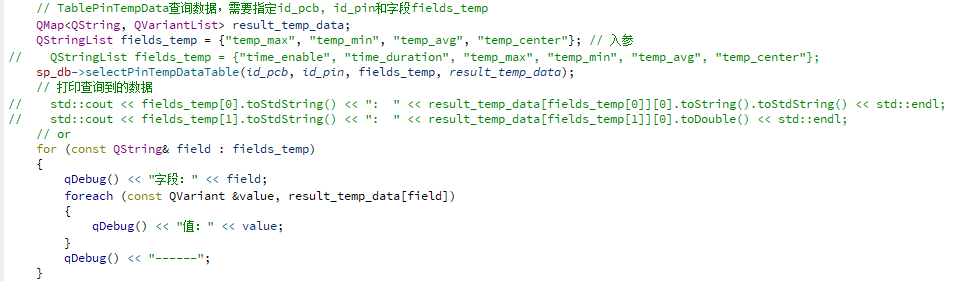


入参：故障检测部分输入数据。

出参：需要调用数据库接口：



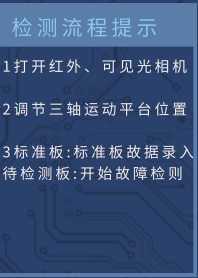
返回的数据类型是QMap<QString, QVariantList>，解析数据流程如下：



注：如果QMap<QString, QVariantList>不好弄，可以先直接创建double类型的数据显示上去。

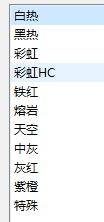
1. 检测流程部分：

文字显示即可



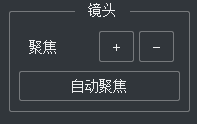
1. 设备调试
2. 红外图像控制
   1. 切换伪彩

Combobox：顺序为：白热、黑热、彩虹、彩虹HC、铁红、熔岩、天空、中灰、灰红、紫橙、特殊。（序号从0开始，默认是2彩虹）



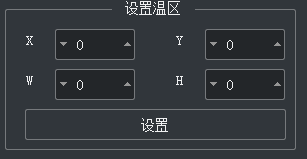
* 1. 镜头：

三个按钮槽函数各自调用不同后端接口。



* 1. 设置温区

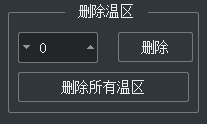
入参：X范围为（0~640）、Y范围为（0~512）、W范围为（0~640-X）、H 范围为（0~512-X）。



* 1. 删除温区

单个温区删除入参：温区编号

删除所有温区不需要入参



* 1. 读取温区温度：

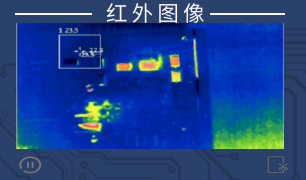


入参：温区编号

出参：四个温度（double类型）



1. 红外图像显示



* 1. 播放按钮：

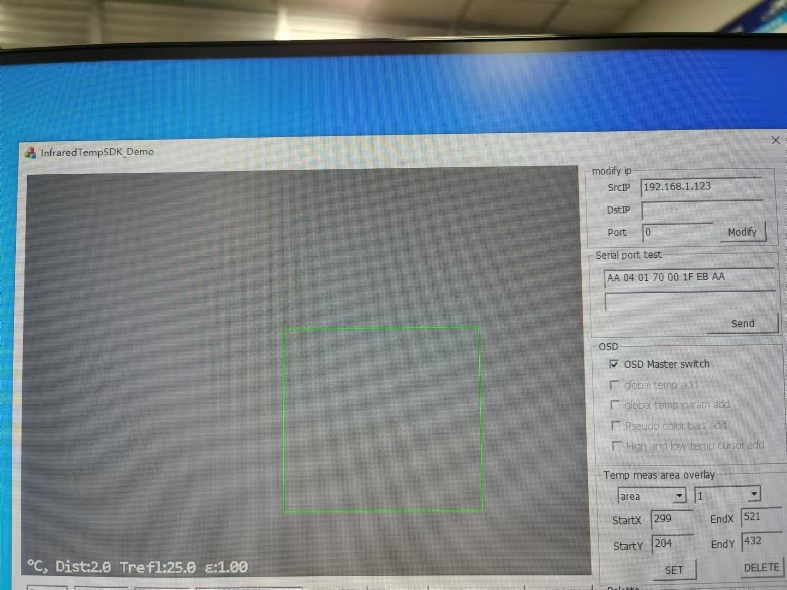
原始图像均为opencv cv::Mat格式，请转化成相应格式后显示

原始图像接口每一次只能获取一张图像，所以该模块开启显示后，应该需要在循环中调用接口不断更新显示。

* 1. 截图按钮：
  2. 双击放大：

放大后的大小为原始图像（640\*512）大小

注：放大后需要有鼠标拖动框选功能，如下例子



能够获取鼠标框选的大小，也即是上面设置温区的四个入参，并有其功能。

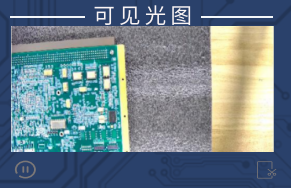
1. 可见光图像控制



四个模块，能有槽函数即可，我来调相应的后端接口

曝光时间那spinbox和horizontalScrollBar同时变化。范围为（100-5000）

1. 可见光图像显示



* 1. 播放按钮：

原始图像均为opencv cv::Mat格式，请转化成相应格式后显示

原始图像接口每一次只能获取一张图像，所以该模块开启显示后，应该需要在循环中调用接口不断更新显示。

* 1. 截图按钮：
  2. 双击放大：

放大后的大小为原始图像（3840\*2160）大小，需要等比例缩小

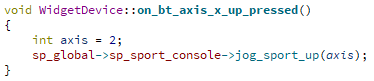
1. 运动平台

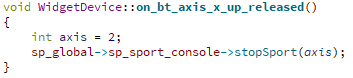


* 1. Jop模式：

X、y、z三轴，鼠标点击就一直不断调用某个接口，直到鼠标释放再调一个接口。

例如x轴：





* 1. 点位模式：

入参：

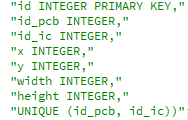
轴（x轴、y轴、z轴）、

位置（x轴范围0~300，y轴范围0~385，z轴范围0~400）

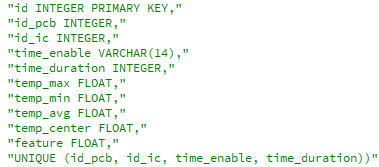
1. 数据库管理

表结构：

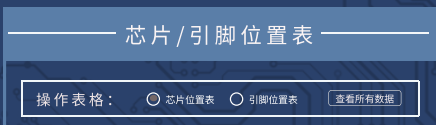
位置表：



温度数据表：



1. 选择表，需要给我一个标记，知道是选的哪个表



1. 查看数据



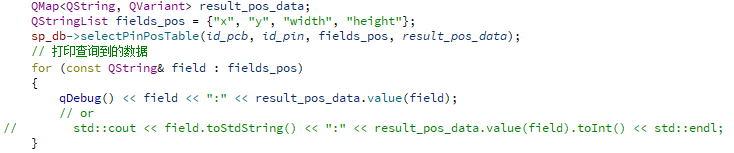
调用接口如下



位置表入参：pcb\_id（必填）、ic\_id（必填）、位置x（可选）、位置y（可选）、高度h（可选）、宽度w（可选），结构可如下：



位置表出参：类型为QMap<QString, QVariant>& result，解析过程如下：



注：如果QMap<QString, QVariant>不好弄，可以先直接创建相应类型的数据显示上去。

注：查询后，点击某条记录可将该记录的数据填充到修改数据和删除数据框中。

1. 修改数据：



入参：pcb\_id（必填不可修改）、ic\_id（必填不可修改）、位置x（必填可修改）、位置y（必填可修改）、高度h（必填可修改）、宽度w（必填可修改）

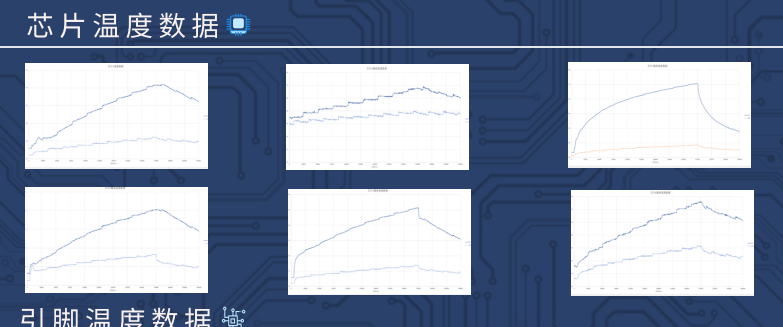
这里示意图的Combobox可能不正确，需要改成spinbox

1. 删除数据：



入参：pcb\_id（必填）、ic\_id（必填）

1. 温度监测
2. 芯片温度数据



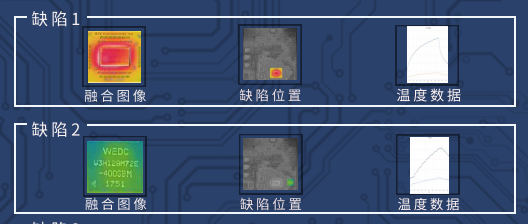
入参：两个std::vector<std::vector<std::pair<int, double>> >对应两条线。横坐标为时间int类型，纵坐标为温度double类型。

一行只展示2~4张表。

1. 引脚温度数据

同上。

1. 检测结果
2. 缺陷部分：



入参：目前暂定，std::vector<std::vector<std::vector<cv::Mat>>>

1. 缺陷结果部分：

入参：同上

导出结果：把缺陷和相应的图导出到本地word文档里面

