**Assignment Three**

**Report**

-

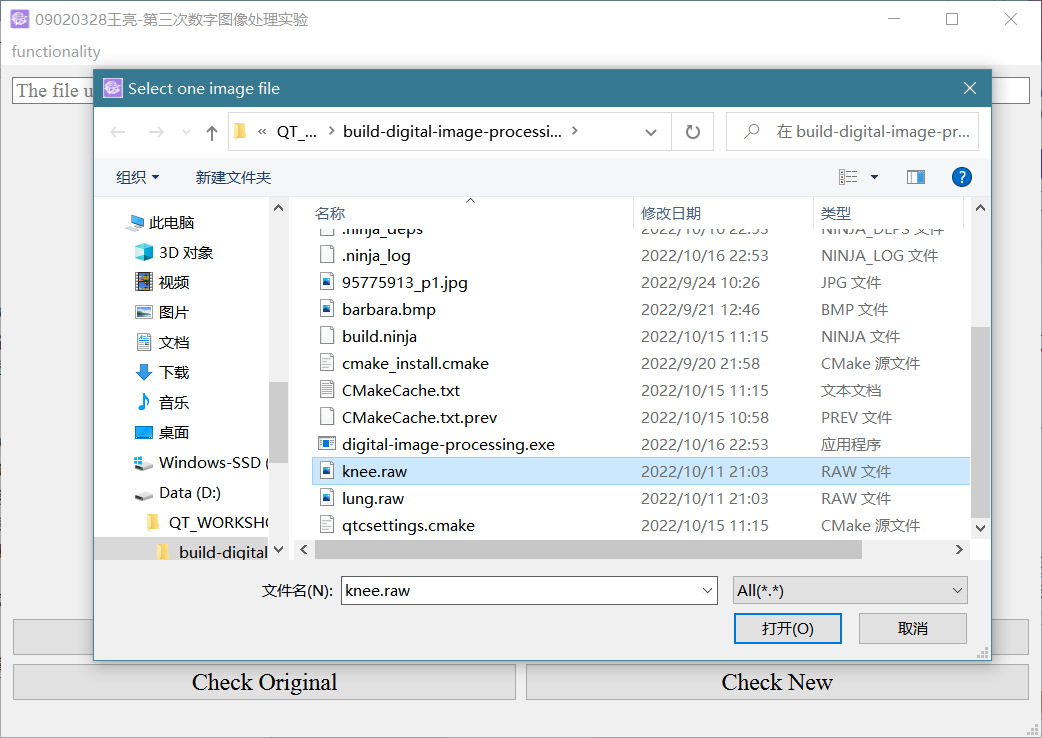
09020328 王亮

Note：本项目使用Qt6 + CMake，没有用QMake，没有.pro文件

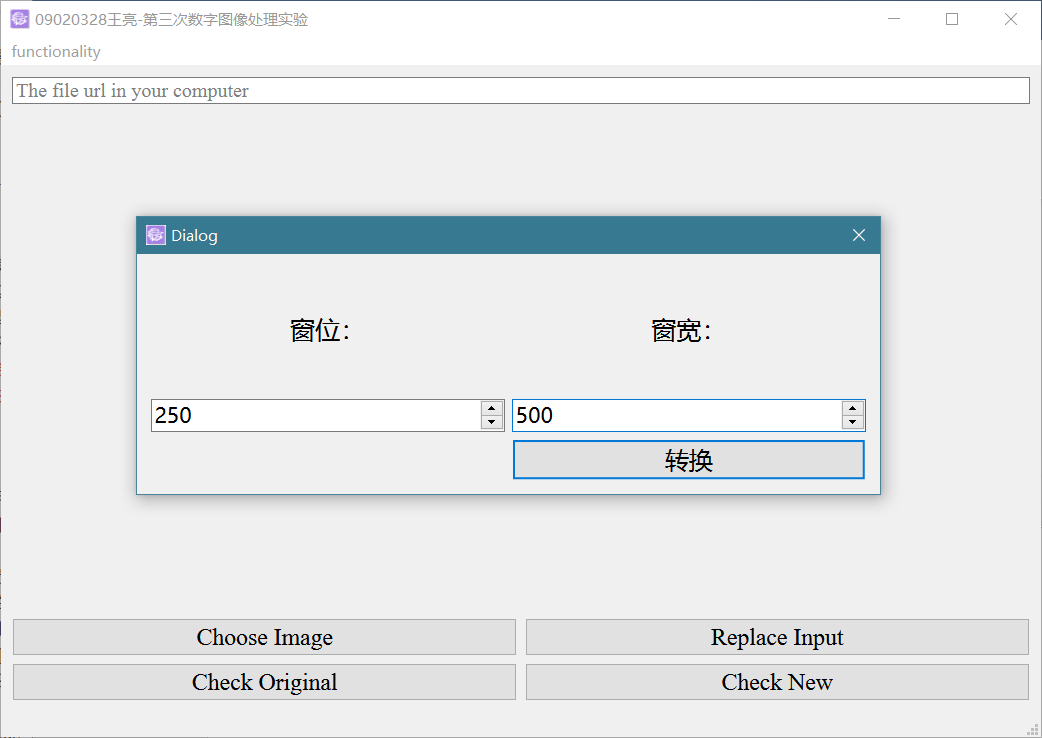
# **I功能实现展示**

1. 展示程序基本界面

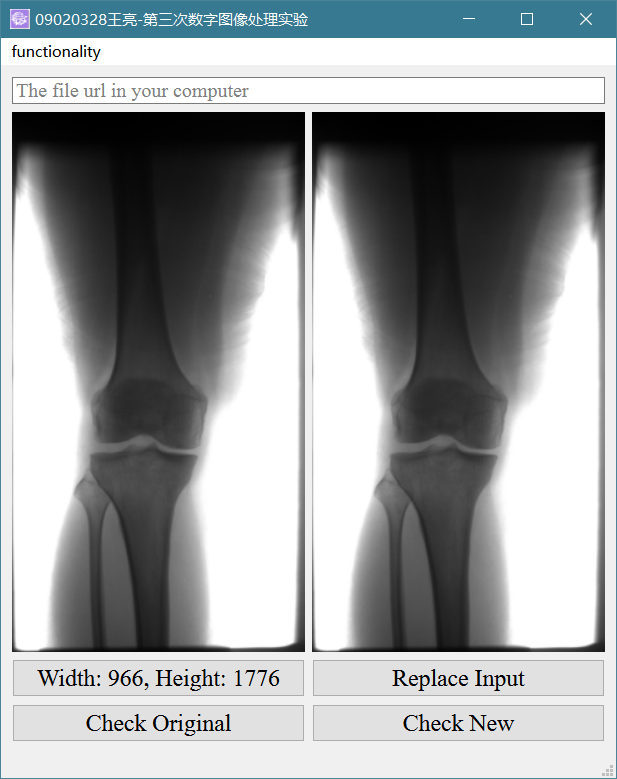
调用之前的选择文件功能，选择需要的raw文件



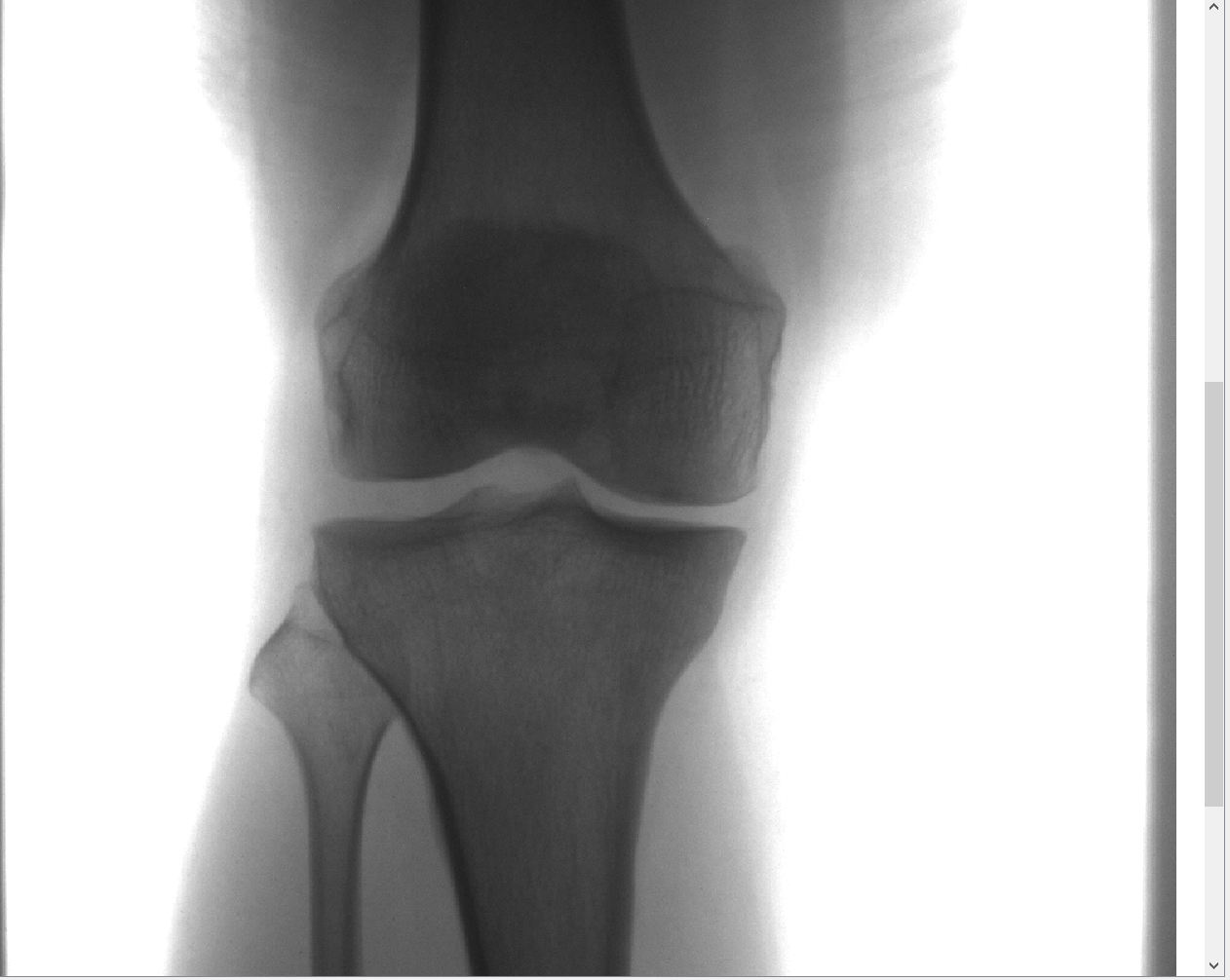
在弹窗中输入窗位 窗宽



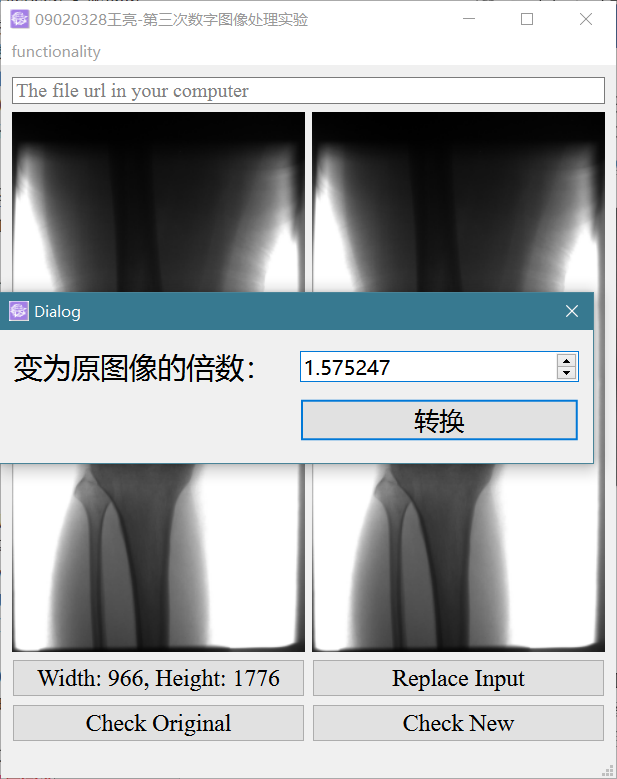
在预览界面显示缩略图像

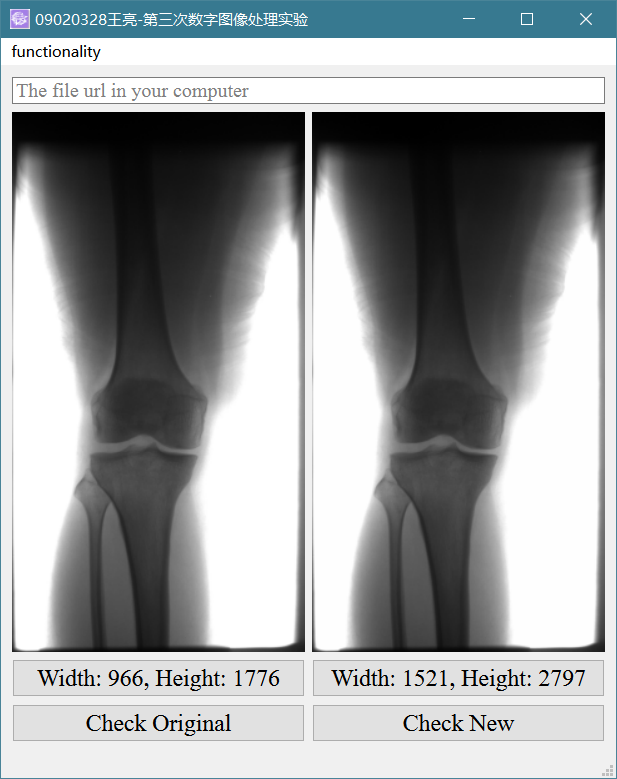


选择查看原图得到如下效果（屏幕高度有限，截图只能展示部分，实际中可通过拖条显示完全）：



对其进行小数倍的放缩

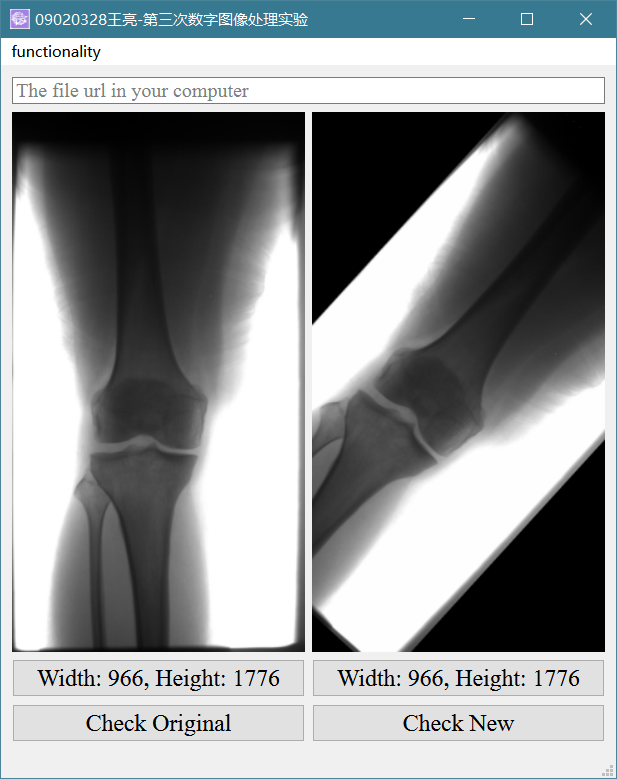






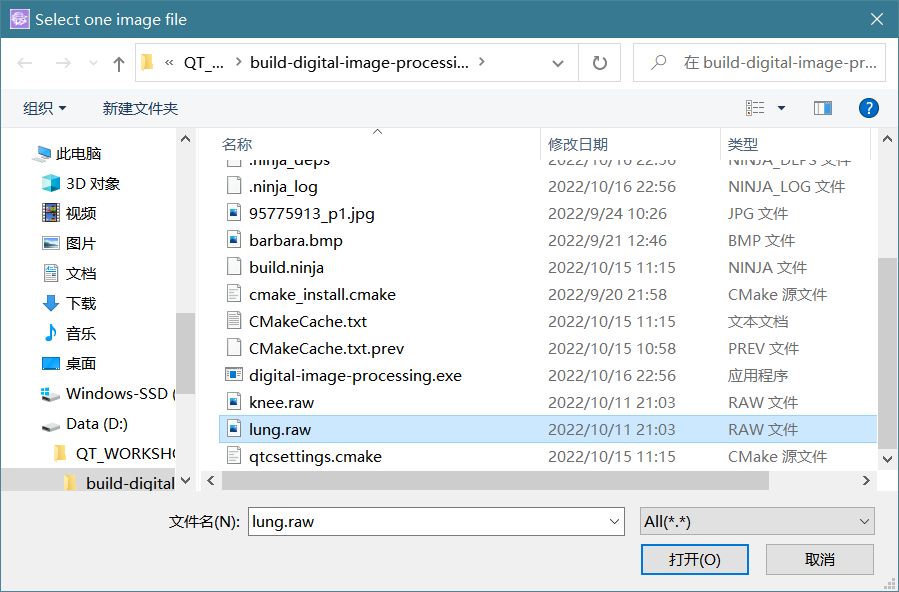
以几何中心做带小数度数的顺时针旋转

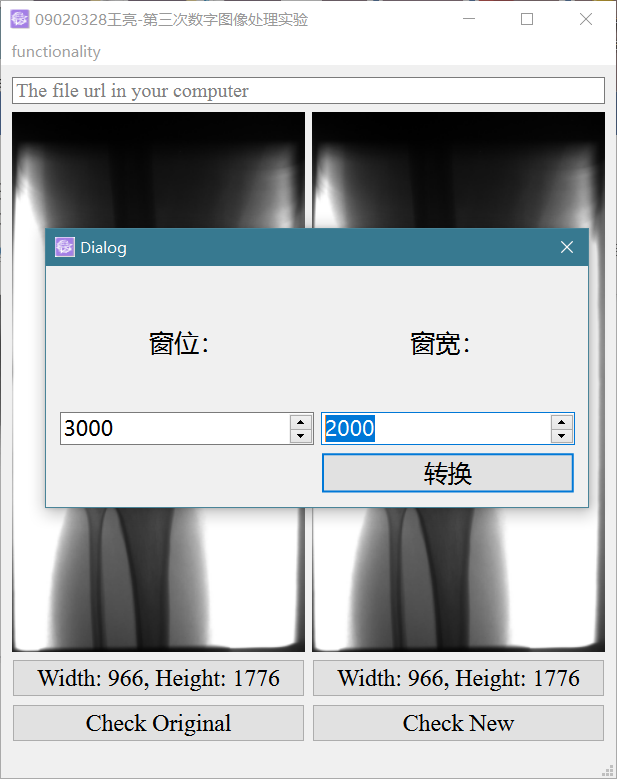


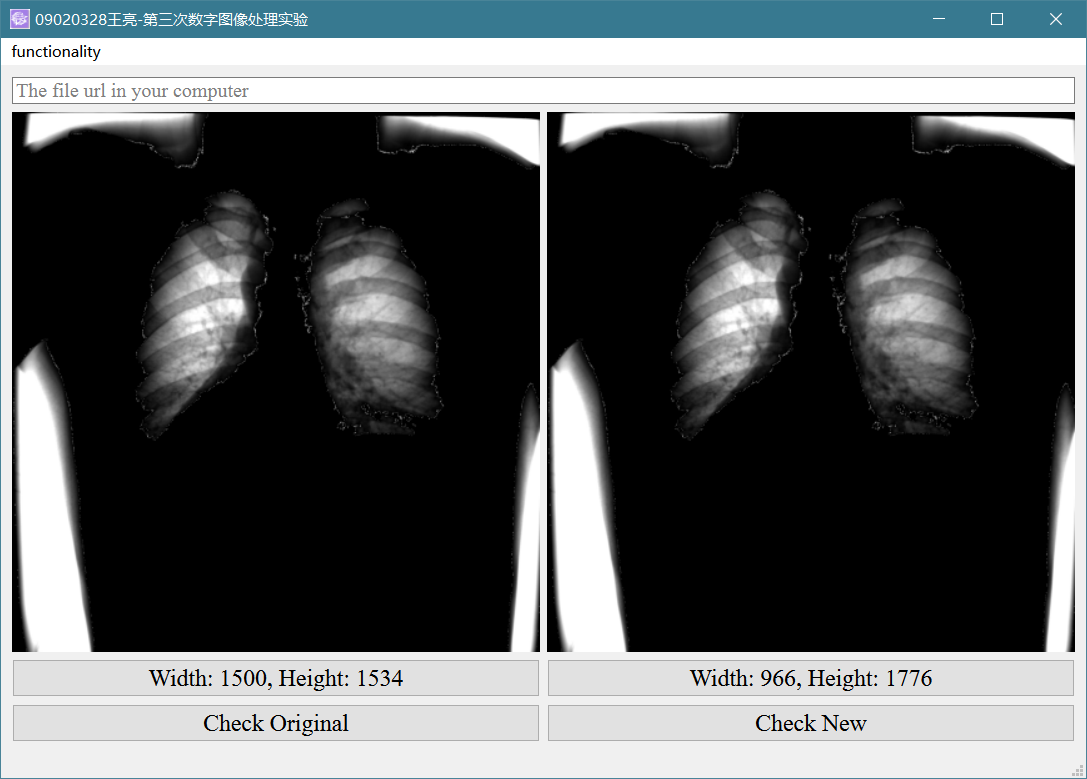




选择lung.raw









2048 4096版本



旋转缩放同理，不再赘述

# **II功能实现原理**

1. 文件读取

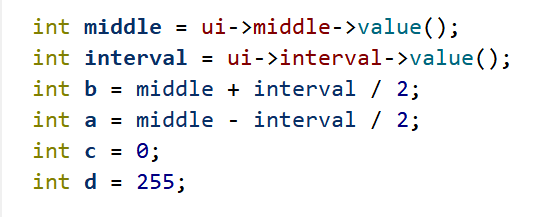
以大小为4的char数组作buffer读取数据，由于intel式的字节顺序问题做移位调整，其中注意移位前先转无符号char，再转为无符号long类型计算后输入到width/height中（同为unsigned long）。方便起见，采用QVector装取图像数据，元数据宽高存在MainWindow中便于管理、复用，由于实际有效灰度范围只有12位，所以与unsigned short 的 4095进行按位与，取出最低12位



1. 灰度映射核心原理

Middle为窗位 interval为窗宽

[a, b]为原区间 [c, d]为目标区间



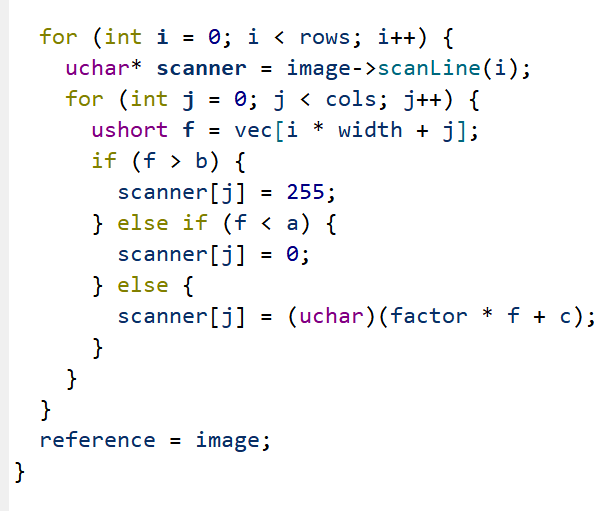
对于从[a, b]区间映射至[c, d]区间可以采用如下公式：

于是：

计算系数



赋值给新图像（超过原本灰度窗范围见如下处理）

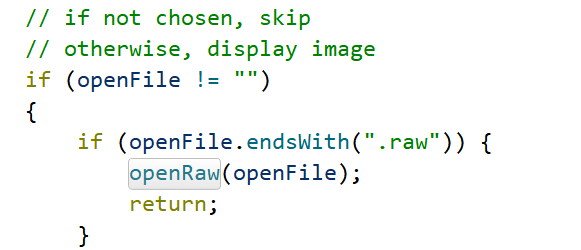


# **III 程序设计总览**

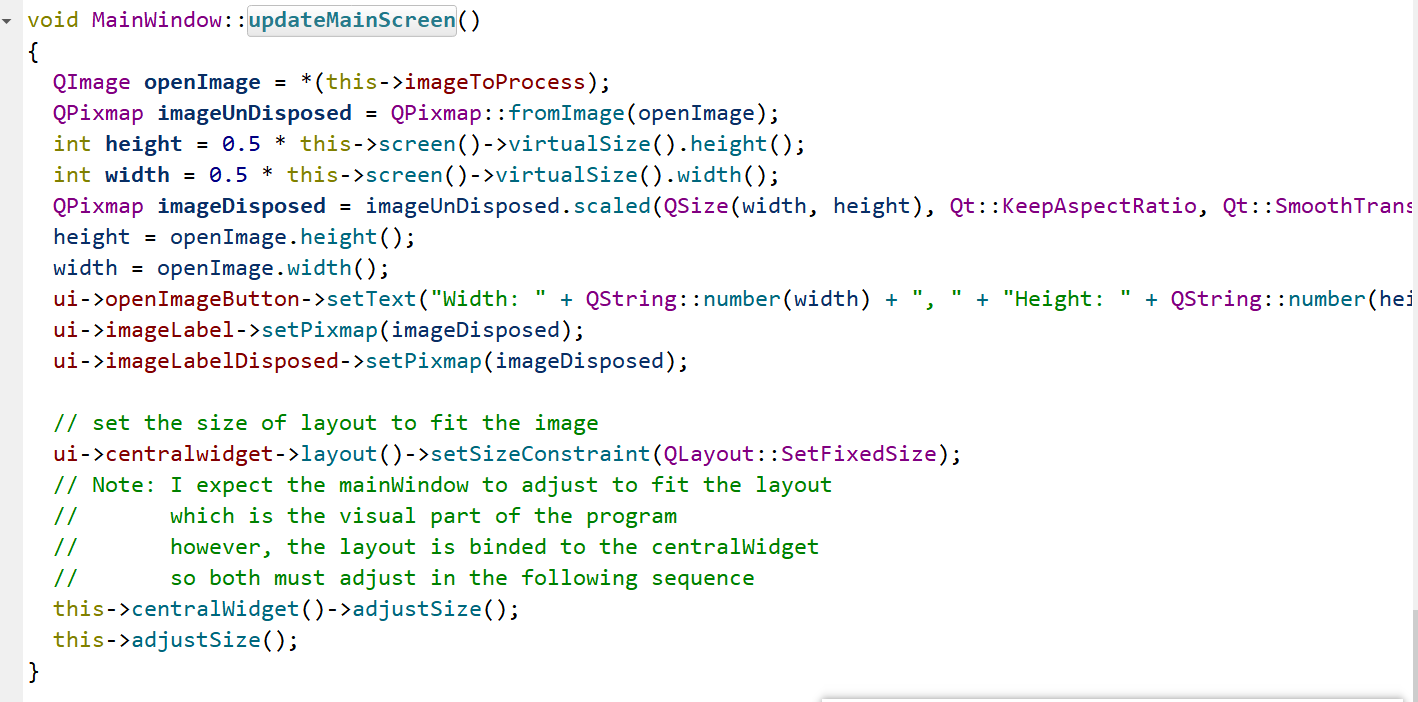
功能扩展遵循一贯的原则，主窗口负责子窗口的调用，逻辑完全由子函数实现，对原有代码完全无侵入性，只增不改。



由于raw文件打开同样属于文件读取部分（事实上此函数约定，自身调用后必然保证imageToProcess（原图）与imageProcessed（待处理图）不再为nullptr，必为有效值），在打开文件的函数进行调用，并将逻辑抽出为单独函数（内容为上方QFile开头的截图）。

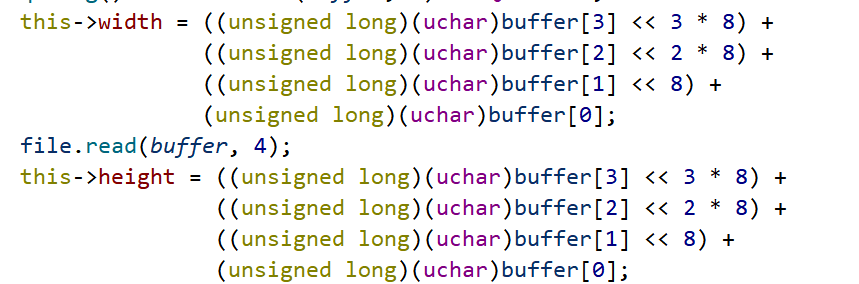


考虑文件读取后的缩略图更新问题（有助于交互体验），将更新主界面缩略图逻辑抽出

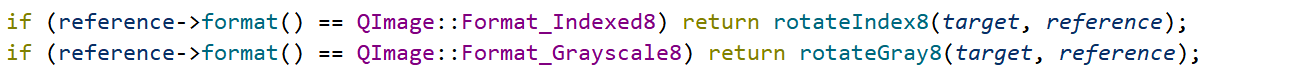


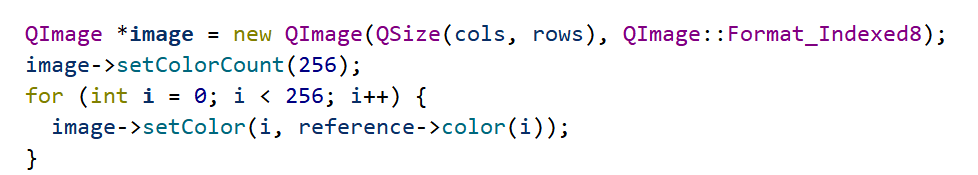
# **设计糟粕与实验总结**

读取过程的类型转换写得相当粗糙，重复度极大，我也想不到怎么改进。



由于一开始的bmp图与现在图片读取不同（bmp读取即为Format\_Indexed8，不能当Grayscale8处理），需要采取分支判断。而二者代码具有大量重复，但由于区别不只在于获取像素的方式还在于格式，以及index8的颜色表设置问题，实际上并不能简单通过一个变量装不同的像素获取函数解决复用，便只得复制粘贴再改了。





不了解QImage底层的实现，在写入图像过程中，需要采用scanLine获取到每一行后再做偏移量修改像素，如果直接采取i \* width + j的偏移量方式会出现图像变形，具体原因不明。

