武汉工程大学

计算机科学与工程学院

综合设计报告

**设计名称： 《图像处理与机器视觉》综合设计**

**设计题目： 图像处理软件设计与实现**

**专业班级： 2020智能科学与技术01班**

**学生学号： 2005120203**

**学生姓名： 方成杰**

**指导教师（职称）： 闵锋（副教授）**

**学业导师（职称）： 赵彤洲（副教授）**

**学生成绩：**

**完成时间： 2022年11月21日-11月24日**

武汉工程大学计算机科学与工程学院 制

说明：

1、报告中的第一、二、三项由综合设计负责人在综合设计开始前填写并发给每个学生。

2、学业导师负责批改学生的设计报告，并给出相应的得分。同时，就设计报告质量撰写评语。

3、指导教师就学生在设计期间的表现及设计完成情况分别给出相应的得分。同时，就此两项情况撰写评语。

4、设计的总评成绩由上述各部分累加得出，由指导教师汇总，并填写于报告的封面。

5、设计报告正文字数一般应不少于5000字，也可由综合设计负责人根据本项综合设计的具体情况酌情增加字数或内容。

6、此表格式为武汉工程大学计算机科学与工程学院提供的基本格式（适用于学院各项课程设计），各专业也可根据本项综合设计的特点及内容做适当的调整，并上报学院批准。

课程教学目标

本课程的教学目标是使学生掌握“数字图像处理”中的基础理论与算法，具备常见数学图像处理算法的设计与编程能力，能够利用数字图像处理思想和方法解决复杂问题求解的能力。培养学生独立地综合运用所学的基础理论、专业知识和基本技能，分析与解决实际工作中遇到的问题的能力；提高学生的工程素养和职业道德素质。通过上机操作，综合设计报告和汇报答辩，检验学生对所学知识的运用。

**（一）课程目标**

|  |  |  |  |
| --- | --- | --- | --- |
| **序号** | **课程目标** | **达成途径** | **评价判据** |
| 1 | 课程目标1：熟练运用本专业所需的数学、自然科学、工程基础和智能科学与技术的专业知识解决智能科学与技术领域的工程问题，如数据分析，人机交互，图像处理等。 | 达成途径：综合设计过程中对专业知识进行运用，分解问题，在报告中对专业知识进行应用。 | 依据代码、综合设计报告、汇报答辩进行评价 |
| 2 | 课程目标2：能够针对智能科学与技术领域的复杂工程问题提出解决方案，设计满足特定需求的系统和模块，并能够在设计环节中体现创新意识。 | 达成途径：综合设计过程中运用专业知识，查找文献，设计具体GUI界面并给出合理的解决方案，并在报告内容中体现出实现的软件效果。 | 依据代码、综合设计报告、汇报答辩进行评价 |
| 3 | 课程目标3：能够针对智能科学与技术复杂工程问题，开发、选择与使用恰当的技术、资源和信息技术工具，包括对智能科学与技术领域复杂工程问题的预测与模拟，并能够理解其局限性。 | 达成途径：通过互联网搜索，使用Visual Studio、OpenCV等集成编程开发环境编码和调试，在实际工程项目中实施及开发。 | 依据代码、综合设计报告、汇报答辩进行评价 |

**（二）所支撑的毕业要求及对应的指标点**

|  |  |  |  |
| --- | --- | --- | --- |
| **序号** | **所支撑的毕业要求** | **所支撑的毕业要求指标点** | **课程目标** |
| 1 | 毕业要求2. 问题分析：能够应用数学、自然科学、工程基础和智能科学与技术的专业知识，识别、表达和有效地分解智能系统理论与应用、图像处理及模式识别等相关领域的复杂工程问题，并通过文献查阅等多种方式对其进行分析，以获得有效结论。 | 指标点2-3: 掌握科技文献、资料的分类；能够通过图书馆、数据库、网上检索等多种方式快速、准确地检索相关信息，具备借助文献研究对复杂工程问题进行识别、表达、分析的能力。 | 要求理解数字图像处理的基本内涵，编程实现图像的打开、显示、保存、放大、缩小、旋转、翻转、反色、灰度化、裁剪等功能，对应课程目标1。 |
| 2 | 毕业要求3：设计/开发解决方案**——**能够针对智能科学与技术领域的复杂工程问题提出解决方案，设计满足特定需求的系统和模块，并能够在设计环节中体现创新意识；能够综合考虑其对社会、健康、安全、法律、文化及环境的影响。 | 指标点3-1: 理解满足实现智能系统软件功能的基本原理和方法，并设计实现硬件基础平台、功能单元或算法流程。 | 要求学生编程实现图像直方图均衡算法，平滑与锐化算法，形态学算法，点，线，边缘检测算法，分割算法等，对应课程目标2。 |
| 3 | 毕业要求5. 使用现代工具：能够针对智能系统理论与应用、图像处理及模式识别等领域的复杂工程问题，开发、选择与使用恰当的技术、资源、现代工程工具和信息技术工具，包括对智能系统理论与应用、图像处理及模式识别等领域复杂工程问题的预测与模拟，并能够理解其局限性。 | 指标点5-2: 能熟练运用学术资源、互联网引擎或者其他信息检索工具获取智能科学与技术领域理论与技术的最新进展信息。 | 要求学生编程实现在图像上添加直线、矩形、圆、字符、文字等功能，对应课程目标3。 |

成绩评定表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 序号 | 考核内容及权重 | 课程目标 | 评分标准 | | 实际  得分 | |
| 1 | 问题分析20分 | 课程目标1 | 对某具体GUI的设计过程进行分解，通过文献资料收集并分析。 | | 学业导师打分 |  |
| 2 | 设计/开发解决方案  60分 | 课程目标2 | 掌握基本的图像处理算法，设计具体GUI界面并给出合理的解决方案，并在报告内容中体现出实现的软件效果。 | | 指导导师打分 |  |
| 3 | 使用现代工具  20分 | 课程目标3 | 会使用Visual Studio、OpenCV等集成编程开发环境编码和调试。 | | 指导导师打分 |  |
|  |  |  |  | |  |  |
| 学业导师签字： | | | | 指导老师签字： | | |
| 总评成绩 | | | | |  | |

|  |
| --- |
| 一、综合设计目的、条件、任务和内容要求：  目的：进一步巩固图像处理与机器视觉的基本原理与算法，掌握一门计算机编程语言，能够进行数字图像处理的应用开发设计，提高分析问题，解决问题的能力。  条件：个人电脑，Windows操作系统，VC++， OpenCV。  任务：开发一款可视化界面的图像处理软件，实现基本的图像处理算法，功能齐全，交互性好。  内容要求：   1. 实现对图像文件的打开、显示、保存、退出等功能； 2. 实现对图像文件的放大、缩小、旋转、翻转、反色、灰度化、裁剪等功能； 3. 实现对图像文件的直方图统计和绘制，直方图均衡等功能； 4. 实现对图像文件的可调整邻域大小的平滑和锐化滤波等功能； 5. 实现对图像文件的膨胀、腐蚀、开运算、闭运算等功能； 6. 实现对图像文件的点、线、边缘检测等功能； 7. 实现对图像文件的给定阈值分割、全局自动阈值分割、OTSU、区域生长等功能； 8. 实现在图像文件上添加直线、矩形、圆、字符、文字等功能； |
| 二、进度安排：  本综合设计总计32学时，分4天完成。  周一：查阅资料、设计界面、确定方案，编写代码；  周二：实现图像处理软件的算法和功能；  周三：优化交互方式，调试和完善代码；  周四：完成功能演示和代码验收。 |
| 三、应收集资料及主要参考文献：  图像处理与机器视觉课程网站：<https://mooc1-1.chaoxing.com/course/214380417.html>  主要参考文献：  1、Milan Sonka, Vaclav Hlavac, Roger Boyle，图像处理、分析与机器视觉，清华大学出版社，2016．  2、张铮，徐超等，数字图像处理与机器视觉，人民邮电出版社，2014．  3、朱虹，数字图像处理基础与应用，清华大学出版社，2013．  4、Gary Bradski、Adrian Kaehle，学习OpenCV（中文版），清华大学出版社, 2009．  5、冈萨雷斯，数字图像处理（第四版），电子工业出版社，2020 |

# 目 录

[摘 要 II](#_Toc801)

[Abstract II](#_Toc28184)

[第一章 课题背景 1](#_Toc12681)

[1.1. 带图形用户界面的图像处理系统 1](#_Toc14183)

[第二章 设计简介及设计方案论述 3](#_Toc23981)

[2.1. 总体设计 3](#_Toc2535)

[2.2. 各模块组织结构设计 4](#_Toc5130)

[第三章 详细设计 7](#_Toc17757)

[3.1. 文件IO模块 7](#_Toc3488)

[3.2. 图像处理各模块 7](#_Toc32210)

[3.3. 画布UI模块 9](#_Toc31030)

[3.4. 绘制模块 11](#_Toc28239)

[3.5. 其他UI界面模块与支持 13](#_Toc14246)

[3.6. 基于PCA的多人脸身份识别模块 14](#_Toc1646)

[第四章 设计结果及分析 16](#_Toc27553)

[4.1. 文件IO模块设计结果 16](#_Toc4497)

[4.2. 图像处理各模块 17](#_Toc149)

[4.3. 画布UI模块 18](#_Toc11457)

[4.4. 绘制模块 19](#_Toc10633)

[4.5. 其他UI界面模块支持 20](#_Toc10362)

[4.6. 基于PCA的多人脸身份识别模块 21](#_Toc8250)

[总 结 22](#_Toc7670)

[致 谢 23](#_Toc28089)

[参考文献 24](#_Toc31671)

# 摘 要

21世纪是一个充满信息的时代，图像作为人类感知世界的视觉基础，是人类获取信息、表达信息和传递信息的重要手段。图像处理是用计算机对图像进行分析，以达到所需结果的技术。本文介绍一款自主设计的，基于Qt5实现GUI界面的图像处理可视化系统，集成了基本的图像处理操作的同时，还实现了基于主成分分析（PCA）的多目标人脸身份识别。

关键词：数字图像处理；图形用户界面；主成分分析；

# Abstract

The 21st century is an era full of information. As the visual basis of human perception of the world, image is an important means for human to obtain information, express information and transmit information. Image processing is a technology that uses a computer to analyze an image to achieve the desired results. This paper introduces a self-designed image processing visualization tool with friendly GUI interface, which not only integrates basic image processing operations, but also realizes multi-target face identity recognition based on principal component analysis (PCA).

**Keywords**：Digital image processing；Graphical user interface;；Principal component analysis；

# 课题背景

带图形用户界面的图像处理系统

项目背景

数字图像处理最早出现于20世纪50年代，当时的电子计算机已经发展到一定水平，人们开始利用计算机来处理图形和图像信息。数字图像处理作为一门学科大约形成于20世纪60年代初期。早期的图像处理的目的是改善图像的质量，它以人为对象，以改善人的视觉效果为目的。图像处理中，输入的是质量低的图像，输出的是改善质量后的图像，常用的图像处理方法有图像增强、复原、编码、压缩等。对图像进行处理(或加工、分析)的主要目的有三个方面:

1. 提高图像的视感质量,如进行图像的亮度、彩色变换,增强、抑制某些成分,对图像进行几何变换等,以改善图像的质量。
2. 提取图像中所包含的某些特征或特殊信息,这些被提取的特征或信息往往为计算机分析图像提供便利。提取特征或信息的过程是计算机或计算机视觉的预处理。提取的特征可以包括很多方面,如频域特征、灰度或颜色特征、边界特征、区域特征、纹理特征、形状特征、拓扑特征和关系结构等。
3. 图像数据的变换、编码和压缩,以便于图像的存储和传输。不管是何种目的的图像处理,都需要由计算机和图像专用设备组成的图像处理系统对图像数据进行输入、加工和输出。

随着计算机、多媒体和数据通信技术的高速发展，数字图像处理近年来得到了极大的重视和长足的发展，并在科学研究、工业生产、医疗卫生、教育、娱乐、管理和通信方面取得了广泛的应用。同时，人们对计算机图像处理软件的要求也越来越高，从而使得高速、可视化、多功能、智能化的高性能数字图像处理软件成为未来图像处理软件的发展方向。

项目基本理论依据

数字图像处理主要研究内容如下：

（1）图像变换：由于图像信号数据量巨大，直接在空间域中处理将涉及大量的计算，因此常采用图像变换的方法将空间域处理转换为变换域处理，以获得更有效的处理结果，其中最常使用的数字图像处理算法是小波分析，它可以进行“空间-尺度分析”和“多分辨率分析”；

（2）图像压缩编码：由于图像的庞大数据量，例如：动态图像、高分辨率图像等，使得存储和传输需要占据大量资源（空间和带宽），不便于后续识别时的特征提取，影响效率，因此在保证不失真的前提下采用编码压缩是图像处理中常用且成熟的技术；

（3）图像增强与复原：其目的是将图像有效部分或选择感兴趣部分通过定量强化，用图像强化技术改善图像质量，获取清晰轮廓及所需细节；然后，当了解图像品质下降（或退化）的原因后，建立“降质模型”，产生一个等价于理想成像系统所获得的图像，利用复原技术恢复原来图像，其整个过程是从图像到图像；

（4）图像重建：与图像增强和复原不同，它是从数据到图像的处理过程，通常是将图像重建与计算机图形学结合将数据转换为有真实感的高品质图像；

（5）图像分割：是数字图像处理中的关键技术之一，顾名思义即把图像分成区域的过程就是图像分割，是将图像中具有重要信息与意义的部位分割出来，它的难度在于图像不能完全依靠计算机自动分割，需将人工信息与之结合，因此如何将各种方法充分融合是目前提高准确性和有效性的主要内容，这也是未来研究的重点；

（6）图像描述与分析：图像描述是图像识别和理解的必要前提，当前对图像的描述不仅仅是二维形状（采用二值），还包括对三维物体的描述；而图像处理的应用目标就是通过对图像的分析获取事件的真相，那么图像分析就是抓住目标特征将其从图像到数据的转换过程；

（7）图像分类与理解：图像的分类属于一种基础性的识别模式，主要是在对图像进行复原、压缩等处理后，再进行分割与特点提取，进而决定如何分类；而图像理解则是在图像分析基础上进一步研究图像中各目标的性质和它们之间的相互联系，得出对图像内容含义的理解及原来客观场景的解释，并给出指导性决策。

本软件紧扣数字图像处理部分主要研究内容，实现图形变换、图像分割、图像识别、图像的增强与复原等功能的带用户界面可视化实时处理。

# 设计简介及设计方案论述

总体设计

软件将要实现基本目标

一款可视化界面的图像处理软件，功能齐全，交互性好。实现基本的图像处理算法，并对图像文件进行读写，具体实现功能如下：

1. 实现对图像文件的打开、显示、保存、退出等功能；
2. 实现对图像文件的放大、缩小、旋转、翻转、反色、灰度化、裁剪等功能；
3. 实现对图像文件的直方图统计和绘制，直方图均衡等功能；
4. 实现对图像文件的可调整邻域大小的平滑和锐化滤波等功能；
5. 实现对图像文件的膨胀、腐蚀、开运算、闭运算等功能；
6. 实现对图像文件的点、线、边缘检测等功能；
7. 实现对图像文件的给定阈值分割、全局自动阈值分割、OTSU、区域生长等功能；
8. 实现在图像文件上添加直线、矩形、圆、字符、文字等功能；

在实现上述功能的基础上，还将结合智能计算与模式识别的内容，实现基于PCA的对多人脸合照的人脸识别与身份检测。

软件结构图

根据上述需求分析，可将各模块功能总结出多个高度独立集成模块，如图2- 1所示。

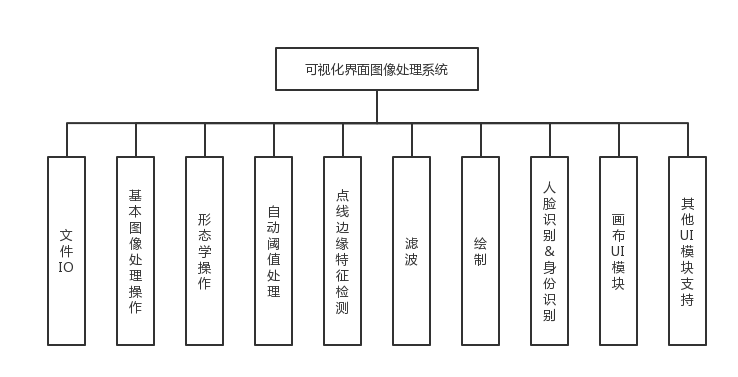


图2- 1 软件结构图

## 各模块组织结构设计

文件IO模块

文件功能模块主要实现对图像文件的打开、按指定格式保存等功能，调用操作系统的资源管理器可视化的设置目录和浏览要打开的图片，模块功能结构如图2- 2所示。

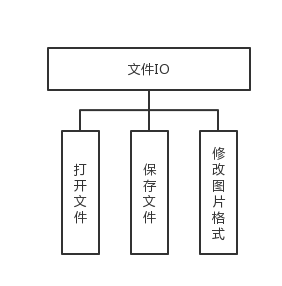


图2- 2 文件IO模块

图像处理各模块

在2.1小节中提到，将需求中对图像处理的操作分为各个模块。但实际上，这些功能都可以抽象成为一个输入为原始图片，输出为处理后结果的方法，封装各个方法并统一输入输出，调用时利用Python的对象特性，将要调用的方法传入，类似于SpringBoot中注入的思想，让接口统一独立扩展性强，模式如图2- 3所示。

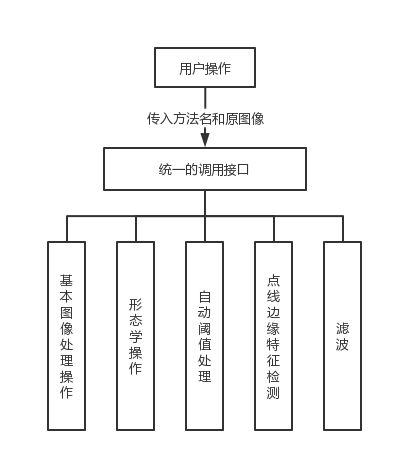


图2- 3 图像处理模块

画布UI模块

画布UI模块是继承重写PyQt5中的QGraphicsView类以实现鼠标拖拽，滚轮放大缩小，刷新显示普通图像和直方图，以及绘画接口供绘画模块使用，如图2- 4所示。

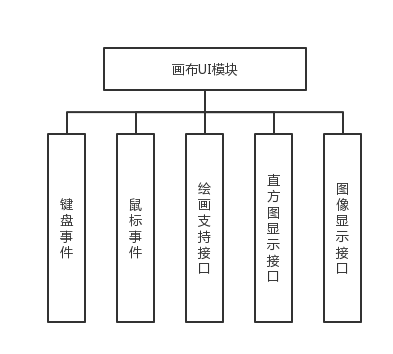


图2- 4 画布UI模块

绘制模块

绘制模块利用设计画布UI模块时留下的绘制接口，实现在画布区域直线，矩形，圆，字符的功能，并实现颜色和以像素为单位的大小调整，实现高精度低锯齿的渲染，如图2- 5所示。

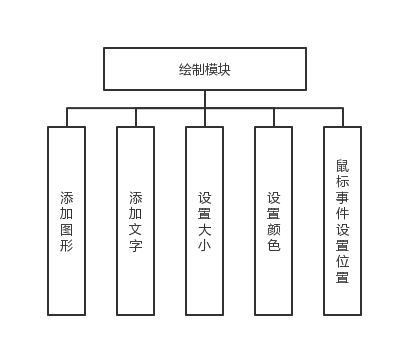


图2- 5 绘制模块

基于PCA的多人脸身份识别模块

这一部分利用自己做的数据集，实现人像合照的多目标人脸身份识别，并调用画布UI模块框出人脸与显示结果，结果如图2- 6所示。

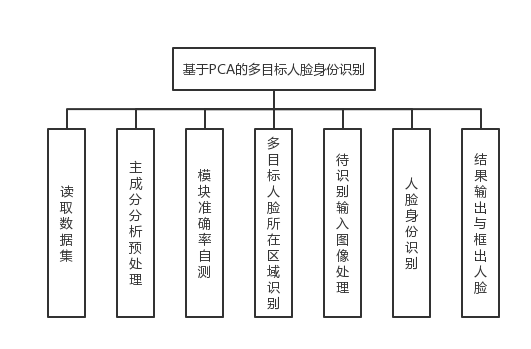


图2- 6 基于PCA的多人脸身份识别模块

其他UI界面模块支持

本模块组织并辅助所有其他模块运行，实现UI界面的构造和交互逻辑，并帮助用户执行各种操作，贯穿所有过程的运行中。包括了UI界面布局，槽函数与动作函数的调用，程序执行进度的监控与异常处理。

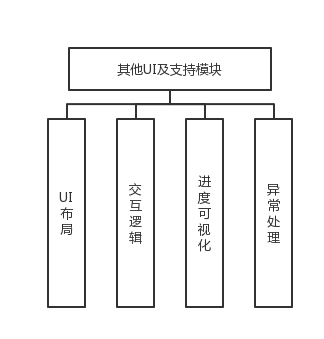


图2- 7 其他UI界面模块支持

# 详细设计

文件IO模块

调用Windows的资源管理器选择待操作或待保存文件的路径，调用OpenCv打开图像并将图像数据，更新内存中的原图像和处理结果图像并传给画布UI模块进行显示。文件操作过程中的交互由其他UI界面模块支持，代码实现如下：

# 文件打开

def open\_img(self):

self.start("打开")

fileName, \_ = QFileDialog.getOpenFileName(

self, '打开图像', '', '\*.png \*.jpg \*.bmp')

if fileName:

self.origin = self.result = np.array(cv2.imdecode(np.fromfile(fileName, dtype=np.uint8), -1))

self.origin\_img\_graphicsView.set\_img(self.origin)

self.result\_img\_graphicsView.set\_img(self.origin)

self.end()

return

# 文件保存

def save\_img(self):

result = self.result

if len(result):

filepath, \_ = QFileDialog.getSaveFileName(self, "保存处理结果", "/", '\*.png \*.jpg \*.bmp')

if filepath:

print(filepath)

cv2.imwrite(filepath, result)

图像处理各模块

调用各功能的接口函数设计

如第二章所述，所有的图像处理操作都由一个函数进行统一调用与管理，该函数还负责与其他模块通信和边界情况检测，及时更新图像，这样的设计大大减少代码冗余，具体设计如下：

def processing(self, method):

self.start("处理中")

# 图像不为空

if len(self.result):

self.result = method(self.result.copy())

self.result\_flash()

self.end()

有些功能需要指定参数（如滤波操作需要指定滤波核大小），相应的也有他们公用的接口函数支持。具体设计如下：

def filter(self, method):

self.start("滤波处理中")

if len(self.result):

size, ok = QInputDialog.getInt(self, 'kernel', '输入滤波核大小', 3, 3, 11, 2)

if ok and size >= 3 and size % 2:

self.result = method(self.result.copy(), size)

self.result\_flash()

else:

QMessageBox(QMessageBox.Warning, '意外输入', '核不合法！核应该 >=3 并 是奇数').exec\_()

self.end()

图像处理方法设计

由于图像处理函数很多都大同小异，这里举几个典型例子来说明图像处理函数的设计，如OTSU全局阈值划分的实现，无论代码逻辑多么复杂，设计上都保证输入输出都是一张图片，具体设计如下：

def OTSU(img):

img = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY)

maxCore = 0

threshold = 0

H, W = img.shape

histogram = np.zeros(256)

for i in range(H):

for j in range(W):

histogram[img[i, j]] += 1

averageAll = np.sum(np.dot(histogram, np.array([n for n in range(256)])))

averageAll = averageAll / np.sum(np.array([n for n in range(256)]))

# 找令类间方差最大的k值

for i in range(1, 255):

# C1均值

mean1 = np.sum(np.dot(histogram[0:i], np.array([n for n in range(i)])))

mean1 = mean1 / np.sum(histogram[0:i])

# C2均值

mean2 = np.sum(np.dot(histogram[i:256], np.array([n for n in range(i, 256)])))

mean2 = mean2 / np.sum(histogram[i:256])

# 公式计算

score = sum(histogram[0:i]) \* ((averageAll - mean1) \*\* 2) + sum(histogram[i:256]) \* ((averageAll - mean2) \*\* 2)

if maxCore < score: # 记录最大值

maxCore = score

threshold = i

dst = cv2.threshold(img, threshold, 255, cv2.THRESH\_BINARY)[1]

return dst

画布UI模块

前面提到，画布UI模块通过继承重写PyQt5中的QGraphicsView类实现。设计中新建了一个文件用来存放重写后的类，并通过Qt中的“提升”操作来构造自定义的画布。具体设计如下：

class View(QGraphicsView):

def \_\_init\_\_(self, parent=None):

super(View, self).\_\_init\_\_(parent)

self.setRenderHints(QPainter.Antialiasing | # 抗锯齿

QPainter.HighQualityAntialiasing | # 高品质抗锯齿

QPainter.TextAntialiasing | # 文字抗锯齿

QPainter.SmoothPixmapTransform) # 图元变换平滑

# 设置放大缩小时跟随鼠标

self.setTransformationAnchor(QGraphicsView.AnchorUnderMouse)

self.setResizeAnchor(QGraphicsView.AnchorUnderMouse)

self.setDragMode(QGraphicsView.ScrollHandDrag)

# 设置拖拽模式

self.setDragMode(self.RubberBandDrag)

self.scene = GraphicsScene(self)

self.setScene(self.scene)

def set\_img(self, img):

del self.scene

if not type(img) is np.ndarray:

self.scene = QGraphicsScene()

self.scene.addWidget(img)

self.setVerticalScrollBarPolicy(Qt.ScrollBarAlwaysOff)

self.setHorizontalScrollBarPolicy(Qt.ScrollBarAlwaysOff)

else:

self.scene = GraphicsScene(self)

img = img.astype(np.uint8)

img = cv2.cvtColor(img, cv2.COLOR\_BGR2RGB)

rows, cols, channels = img.shape

self.scene.setSceneRect(0, 0, cols, rows)

bytesPerLine = channels \* cols

QImg = QImage(img, cols, rows, bytesPerLine, QImage.Format\_RGB888)

pix = QPixmap.fromImage(QImg)

item = QGraphicsPixmapItem(pix)

self.scene.addItem(item)

self.setScene(self.scene)

# 滚轮与键盘事件

def wheelEvent(self, event):

zoomInFactor = 1.25

zoomOutFactor = 1 / zoomInFactor

if event.angleDelta().y() > 0:

zoomFactor = zoomInFactor

else:

zoomFactor = zoomOutFactor

self.scale(zoomFactor, zoomFactor)

def keyPressEvent(self, event) -> None:

if event.key() == Qt.Key\_Control:

self.setDragMode(self.ScrollHandDrag)

def keyReleaseEvent(self, event) -> None:

if event.key() == Qt.Key\_Control:

self.setDragMode(self.RubberBandDrag)

绘制模块

绘制模块主要通过添加和设置QGraphicsItem（图元）来实现，通过鼠标响应事件，获取起始点与终止点的位置坐标，通过鼠标移动事件，实现动态拖拽放置。通过调用颜色和整数对话框，来设置画笔的颜色与大小，通过一个枚举类型来设置初始化图元内容，具体设计如下：

class Draw(Enum):

# 为序列值指定value值

eraser = 0

line = 1

rect = 2

circle = 3

text = 4

pen = 5

def paint\_graph\_init(self, type\_num):

color, size = self.getFont()

self.paint = True

self.paint\_type = type\_num

if type\_num == Draw.line:

self.paint\_item = QGraphicsLineItem()

self.set\_method = self.paint\_item.setLine

elif type\_num == Draw.rect:

self.paint\_item = QGraphicsRectItem()

self.set\_method = self.paint\_item.setRect

self.paint\_item.setBrush(QBrush(color))

elif type\_num == Draw.circle:

self.paint\_item = QGraphicsEllipseItem()

self.set\_method = self.paint\_item.setRect

self.paint\_item.setBrush(QBrush(color))

elif type\_num == Draw.text:

self.paint\_item = QGraphicsSimpleTextItem()

self.set\_method = self.paint\_item.setPos

font = QFont()

font.setPixelSize(size)

self.paint\_item.setFont(font)

self.paint\_item.setBrush(QBrush(color))

text, ok = QInputDialog.getText(self.parent, '文本', '输入添加内容')

if ok and text:

self.paint\_item.setText(text)

else:

self.paint\_item.setText("")

# override

def mousePressEvent(self, event):

if self.clicked:

self.clicked = False

self.paint = False

if self.paint:

self.clicked = True

self.start\_pos = event.scenePos()

print(self.start\_pos.x())

self.addItem(self.paint\_item)

if self.paint\_type == Draw.text:

self.paint\_item.setPos(self.start\_pos)

def mouseMoveEvent(self, event):

if self.clicked:

self.end\_pos = event.scenePos()

print("self.end\_pos=", self.end\_pos)

if self.paint\_type == Draw.line:

self.set\_method(self.start\_pos.x(), self.start\_pos.y(), self.end\_pos.x(), self.end\_pos.y())

elif self.paint\_type == Draw.circle or self.paint\_type == Draw.rect:

self.set\_method(self.start\_pos.x(), self.start\_pos.y(),

self.end\_pos.x() - self.start\_pos.x(), self.end\_pos.y() - self.start\_pos.y())

elif self.paint\_type == Draw.text:

self.set\_method(self.end\_pos)

else:

return

def mouseReleaseEvent(self, event):

self.clicked = False

self.paint = False

def getFont(self):

color = QColorDialog.getColor()

size, ok = QInputDialog.getInt(self.parent, '大小', '输入希望添加元素的大小(pix)')

if ok and size > 0:

return color, size

else:

size = 20

return color, size

其他UI界面模块与支持

基础UI界面设计

基础的UI框架由QtDesigner可视化搭建，设计的UI界面及按钮槽函数如下所示：



图3- 1 基础UI界面

程序进度可视化实现

通过继承重写Qt中QProgressBar（进度条）实现向用户实时显示程序状态，缓解用户在等待期间的不适。通过在程序执行的关键时间点调用来更新状态，具体设计如下：

class ProgressBar(QProgressBar):

def \_\_init\_\_(self, parent=None, step=8):

super().\_\_init\_\_(parent)

self.step = step

self.setRange(0, step) # 设置进度条的范围

def done(self):

self.setMaximum(self.step)

self.setValue(self.step)

def busy(self):

self.setMaximum(0)

self.setMinimum(0)

基于PCA的多人脸身份识别模块

这部分与图像处理模块部分相似，也是输入图像，输出结果，在主函数外再封装一个用于PCA的类，具体设计如下：

def inquire\_result(index):

return Dictionary[index][1]

class PCA:

def \_\_init\_\_(self):

self.xTrain\_, self.yTrain, self.pathSet, count = loadImageSet()

self.num\_train = self.xTrain\_.shape[0]

self.xTrain, self.data\_mean, self.V = self.pca(count)

def pca(self, k):

data = self.xTrain\_

data = np.float32(np.mat(data))

rows, cols = data.shape # 取大小

data\_mean = np.mean(data, 0) # 求均值

# 将data\_mean拉成与data一样的矩阵(行复制rows份)便于减法

Z = data - np.tile(data\_mean, (rows, 1))

D, V = np.linalg.eig(Z \* Z.T) # 特征值与特征向量

# 注意矩阵索引 V[:, :k] 取每一行前k列[0, k)

V1 = V[:, :k] # 取前k个特征向量

V1 = Z.T \* V1

for i in range(k): # 特征向量归一化

# 求范数，即 √x1²+x2²+...+xn² 归一化

# 第i列的所有元素

divisor = np.linalg.norm(V1[:, i])

if divisor != 0:

V1[:, i] /= divisor

return np.array(Z \* V1), data\_mean, V1

def img\_process(self, img):

img = cv2.cvtColor(img, cv2.COLOR\_RGB2GRAY)

data = np.array(cv2.resize(img, IMAGE\_SIZE)).flatten()

return np.array((data - self.data\_mean) \* self.V)

def test\_identify(self):

# np.tile 将data\_mean拉成与data一样的矩阵(行复制num\_test份)便于减法

# 得到测试脸在特征向量下的数据

xTest\_, yTest = loadTestSet()

num\_test = xTest\_.shape[0]

xTest = np.array((xTest\_ - np.tile(self.data\_mean, (num\_test, 1))) \* self.V)

yPredict = [self.yTrain[np.sum((self.xTrain - np.tile(d, (self.num\_train, 1))) \*\* 2, 1).argmin()] for d in

xTest]

print(u'欧式距离法识别率: %.2f%%' % ((yPredict == np.array(yTest)).mean() \* 100))

def identify(self, sample):

index = np.sum((self.xTrain - np.tile(sample, (self.num\_train, 1))) \*\* 2, 1).argmin()

img = cv2.imread(self.pathSet[index])

print("识别结果:" + Dictionary[self.yTrain[index]][1])

return img, Dictionary[self.yTrain[index]][1]

def face\_detect(self, img):

img\_gray = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY) # 图片灰度化

img\_gray = cv2.equalizeHist(img\_gray) # 直方图均衡化

face\_cascade = cv2.CascadeClassifier('..\lbpcascade\_animeface.xml') # 加载级联分类器

faces = face\_cascade.detectMultiScale(img\_gray,

scaleFactor=1.01,

minNeighbors=1,

minSize=(250, 250))

for (x, y, w, h) in faces: # 遍历所有检测到的动漫脸

img = cv2.rectangle(img, (x, y), (x + w, y + h), (0, 0, 255), 5) # 绘制矩形框

return img, faces

# 设计结果及分析

文件IO模块设计结果

交互良好的指引用户打开和保存文件，运行结果如下图4- 1图4- 2所示。

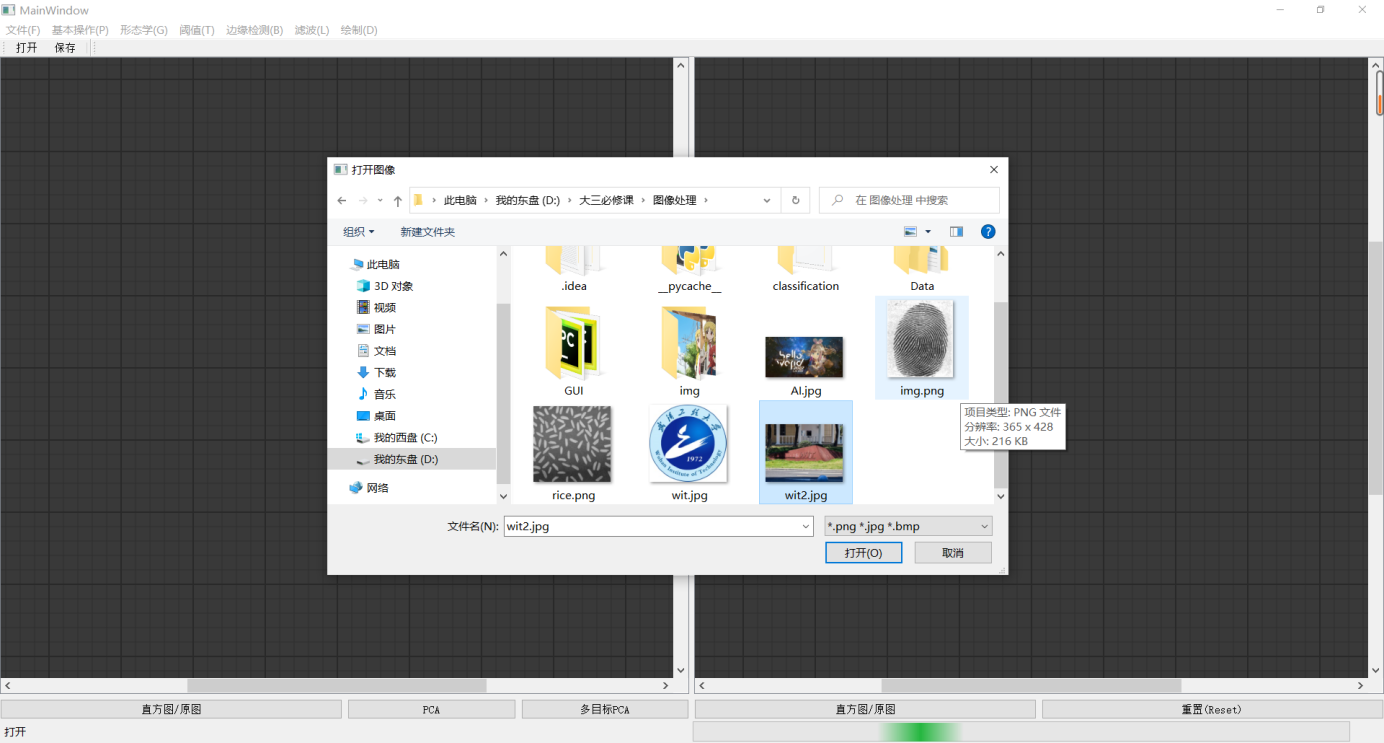


图4- 1 打开文件

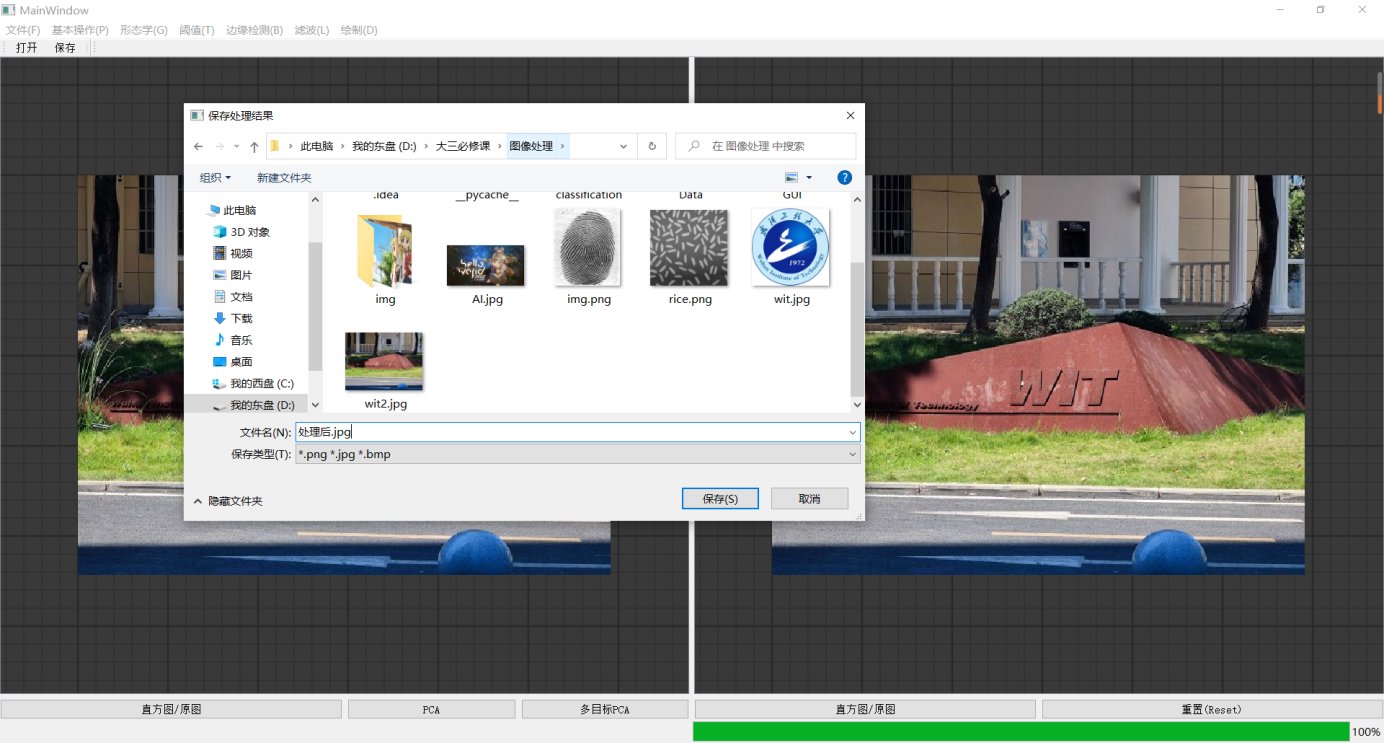


图4- 2 保存文件

图像处理各模块

用户通过点击上侧工具栏中各类中的图像处理方法，可以调用相应的图像处理功能，原图显示在左侧，处理后的图片显示在右侧，并且支持对处理后的图片继续进行新的处理，下面以边直方图均衡化为例，运行结果如图4- 3图4- 4所示。

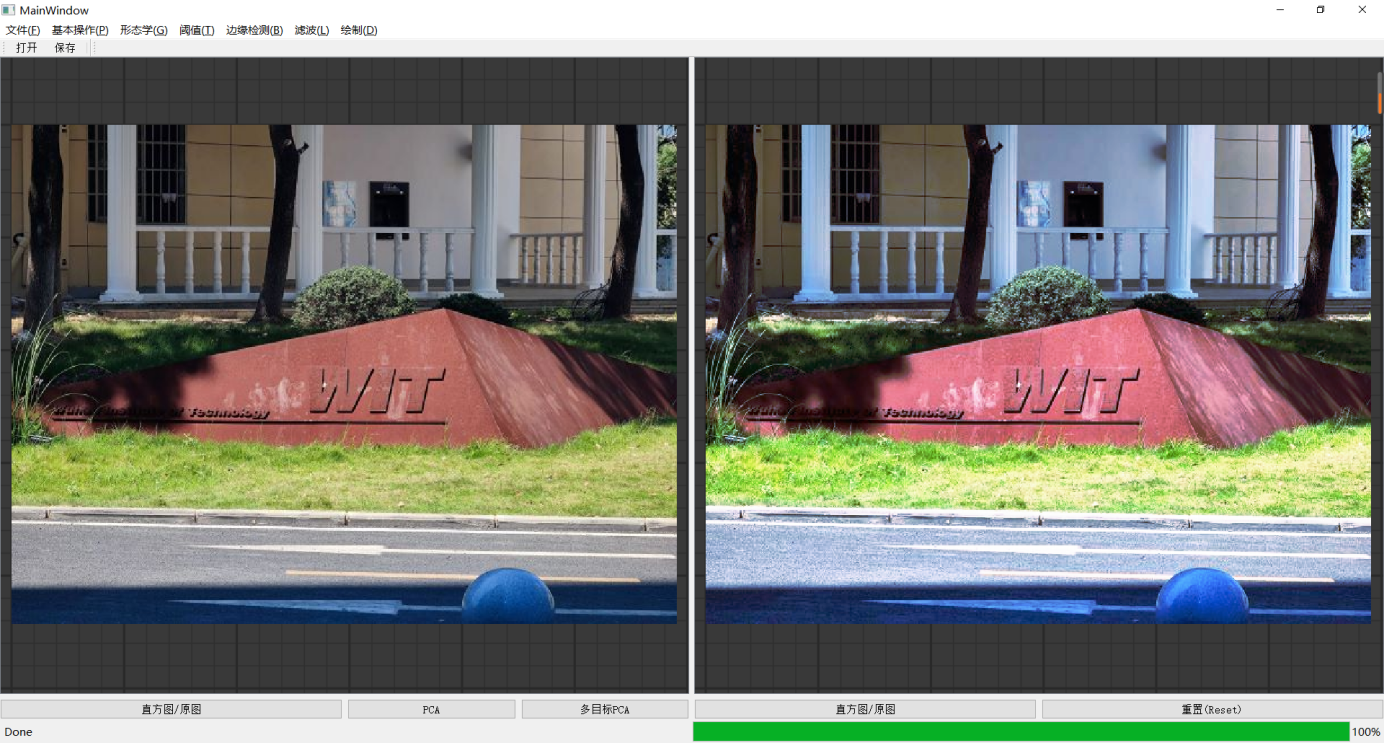


图4- 3 直方图均衡化

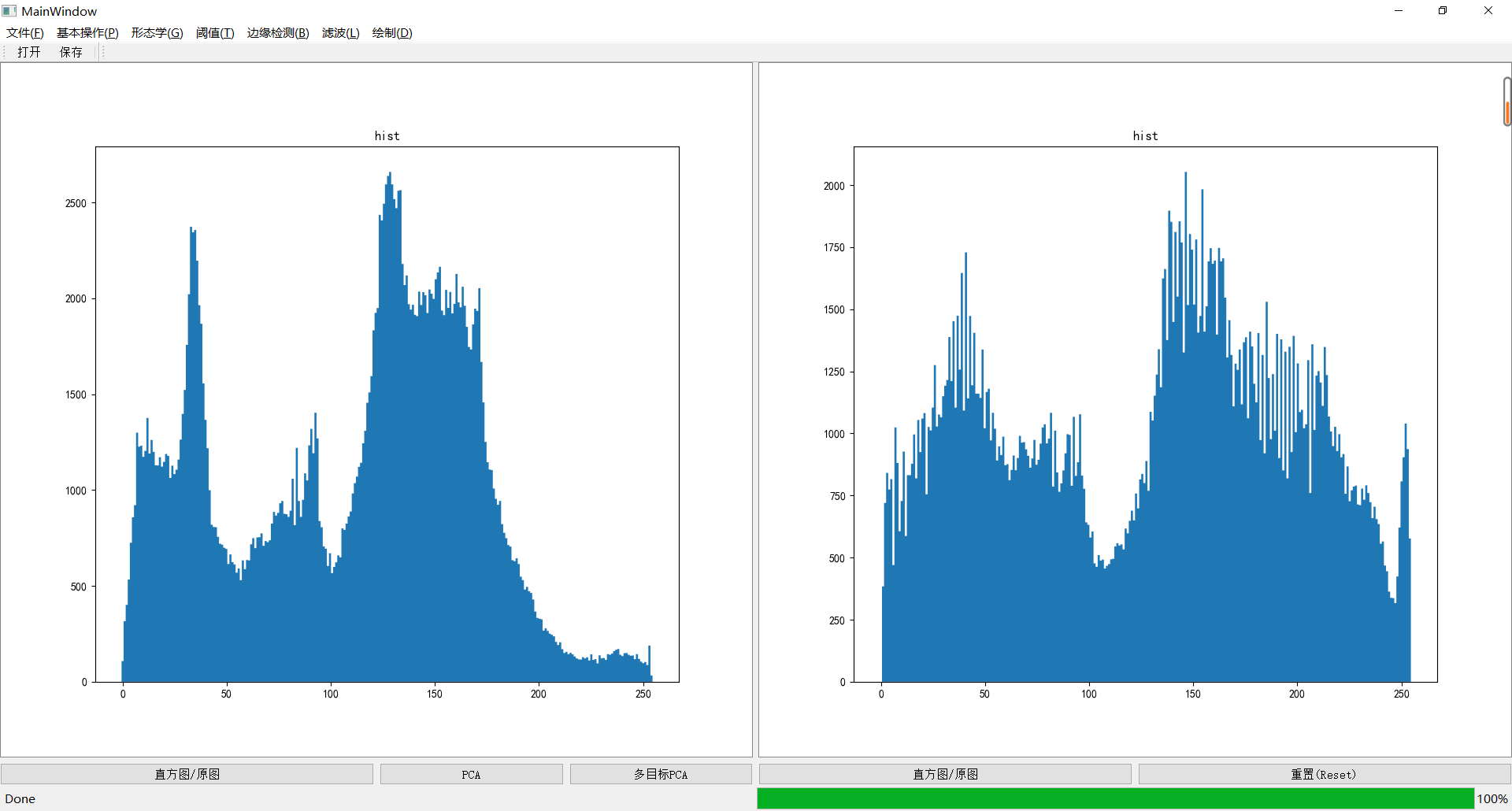


图4- 4 直方图显示

画布UI模块

画布初始状态为灰底黑线网格，在鼠标在画布内时情况下按下Ctrl键可切换到拖拽模式，滚动滚轮可以在鼠标位置进行缩放操作，点击画布底下的按钮可以重置画布以及切换原图和直方图，以拖拽和放大为例，运行结果如图4- 5图4- 6所示。

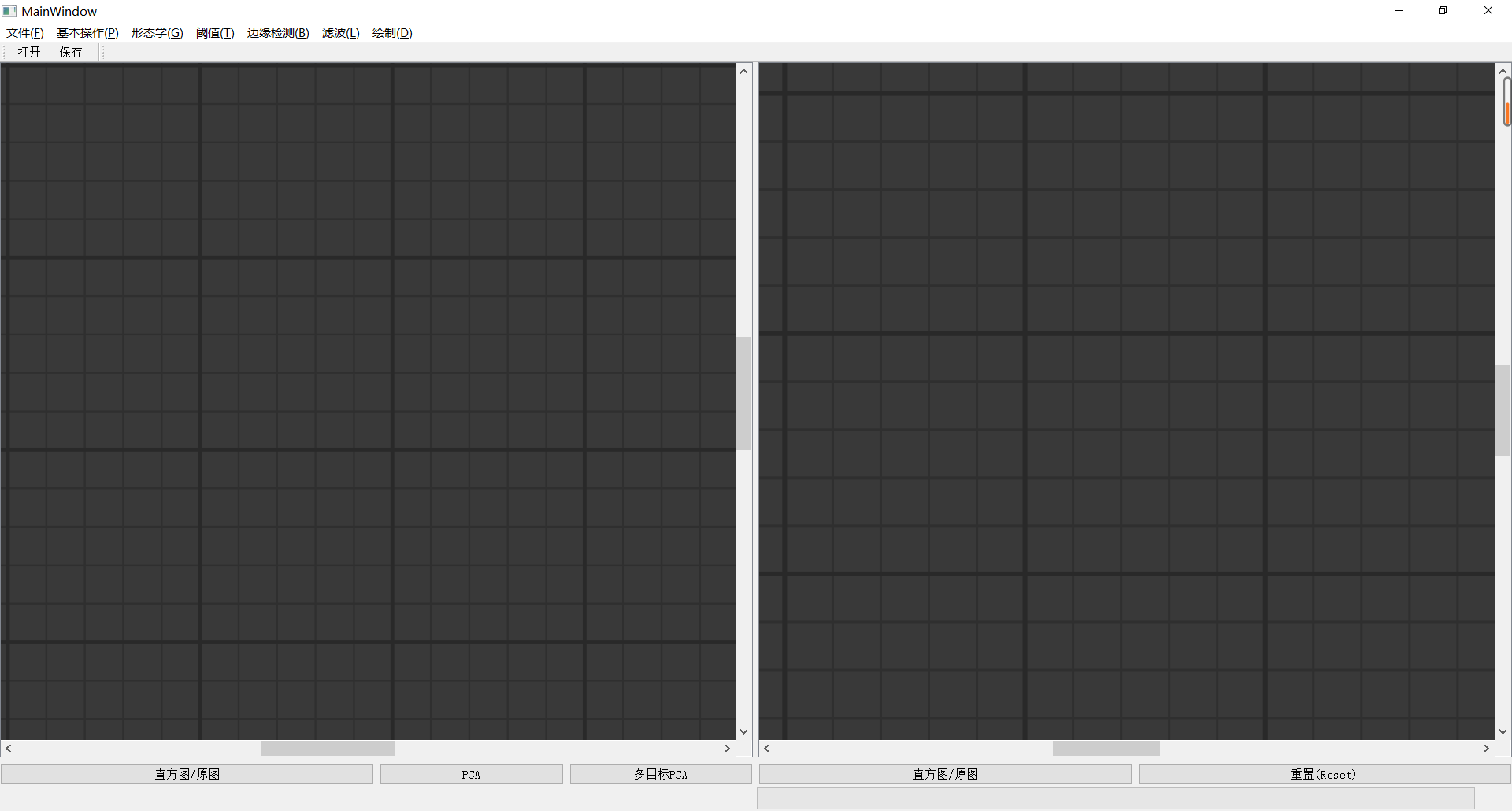


图4- 5 画布初始状态（放大）

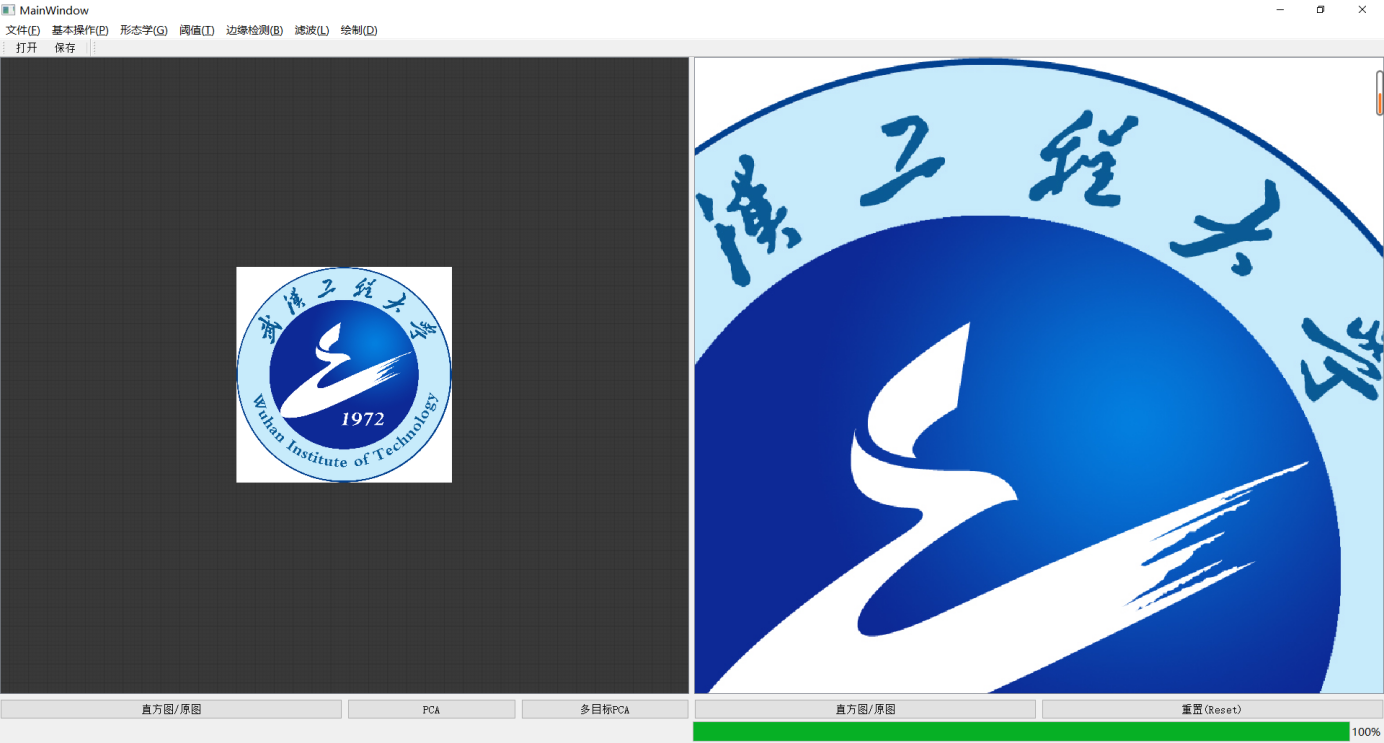


图4- 6 画布鼠标操作

绘制模块

用户通过上方选择绘制功能，指定颜色和大小后，直接在画布上拖拽鼠标来确定图形绘制，状态栏会提示此时绘制状态，绘制过程直观便捷精确，运行结果如图4- 7图4- 8所示。

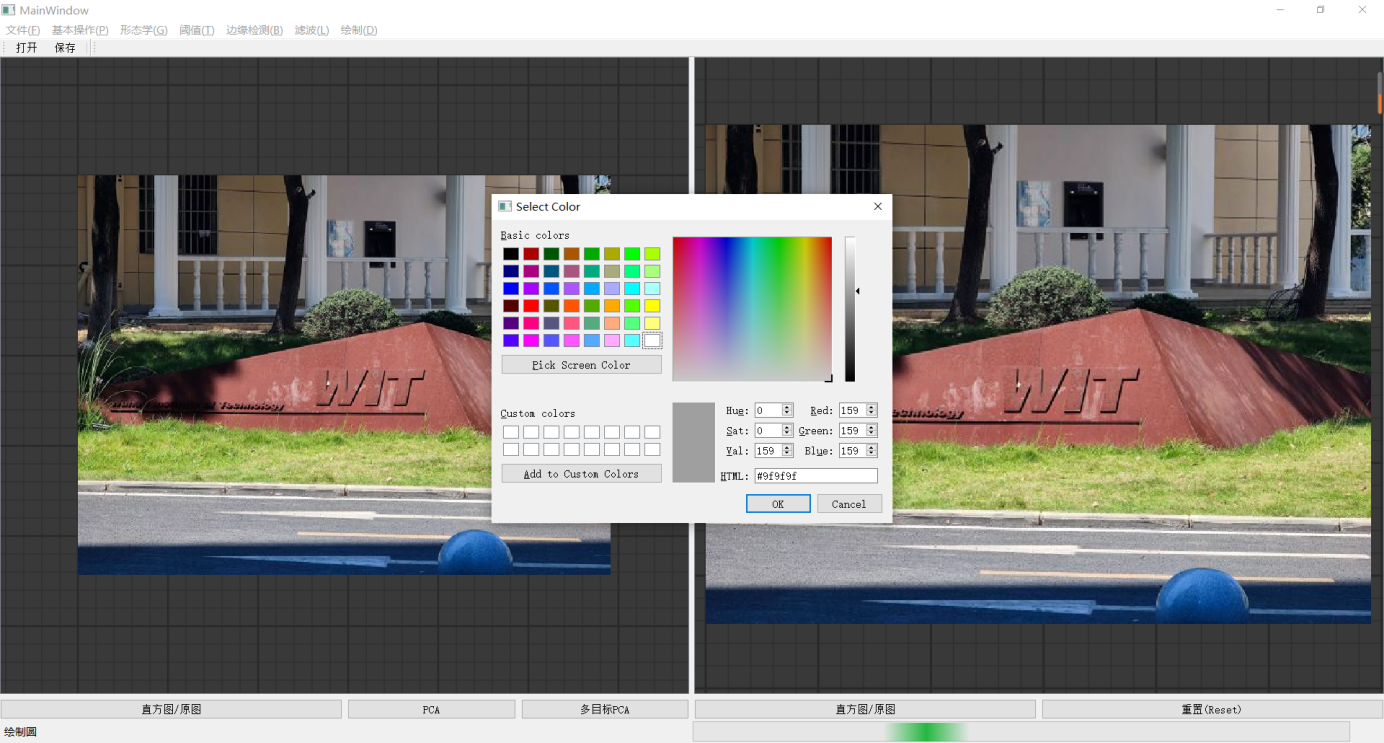


图4- 7 颜色选择



图4- 8 绘制效果

其他UI界面模块支持

该模块承担着UI构建和处理调用逻辑，包括进度条，界面各种按钮，贯穿在上面的几个模块运行中，这里着重展示UI支持模块对用户非法输入的处理和对意外发生的致命异常紧急回溯处理，例如在滤波处理中，核应该大于等于3并是奇数，运行结果如图4- 9图4- 10所示。



图4- 9 非法输入处理



图4- 10 致命错误处理

基于PCA的多人脸身份识别模块

软件事先构建了数据集，用户通过点击PCA按钮进行人脸身份识别操作，两个按钮分别是自动识别人脸和手动框选，由于疫情原因这里采用动漫人脸作为示例，运行结果如图4- 11图4- 12所示。



图4- 11 手动单人脸识别



图4- 12 多人脸自动识别

# 总 结

此次课程设计顺利的达成预定的所有要求和目标，还结合《软件工程》与《智能计算》课程，实践了结构化需求分析与设计，模块化构建系统体系，以及模式识别的经典方法，百感交集，收获颇丰！

此次图像处理软件综合设计完成的过程不能说是一帆风顺，先后经历各种技术上的和疫情的困难与险阻，有值得骄傲的地方，也有些许不足与遗憾，能顺利完成此次课设，不仅是我一个人的努力，还是智能科学与技术教研室老师和武汉工程大学教务处等部门共同的成果。

回顾这个设计过程，我学到了许多书本上没有学到的知识，丰富了自己的实践经验，扩展了本专业的知识面，同时熟悉了使用Qt进行桌面端开发的工具与流程，受益非浅！此次课程设计将会成为我本科期间宝贵的经验和难忘的回忆！

# 致 谢

这次综合设计的完成，不止是我自己的努力，同时也有老师的指导，还有周围同学的帮助，以及那些无私奉献的计算机前辈。通过这次综合设计，我想我成长了很多，不只是磨练了我的专业技能，也使我更加确定了我今后的目标，在计算机领域继续钻研深造。在此我要感谢教授本门课程的闵锋老师与我的导师赵彤洲老师，我会谨记老师们的教诲！

通过这次实践，我学习了很多新知识，也对很多以前的东西有了更深的记忆与理解。漫漫求学路，过程很快乐。我要感谢智能科学与技术教研室的老师们。从他们那里学到了许多珍贵的知识和科学严谨的学术态度，令我受益良多。同时还要感谢学院给了我一个可以认真学习的环境和机会。

# 参考文献

1. 李俊山,李旭辉.数字图像处理 [M].清华大学出版社,2007:195-223.
2. 阮秋琦,阮宇智等.数字图像处理 (第二版) (美)冈萨雷斯 [M]. 电子 工业出版社,2003:252-284.
3. 朱虹.数字图像处理基础 [M].科学出版社,2005:164-166.
4. 章毓晋.图像处理和分析 [M].清华大学出版社,1999:254-278.
5. 闫敬文.数字图像处理MATLAB版 [M].国防工业出版社,2007.2.