编号



综合课程设计II 数字图像方向

课程设计报告

二级学院 计算机科学与工程学院

专 业 计算机科学与技术

班 级 118030701

学生姓名 年哲超 学号 11704060103

指导教师 傅由甲 王爱娟

时 间 2021年1月

目 录

[1 功能描述与分析 1](#_Toc60826850)

[1.1 功能描述 1](#_Toc60826851)

[1.2 功能分析 1](#_Toc60826852)

[1.3 组员分工 1](#_Toc60826853)

[2 图像合成模块的设计与实现 3](#_Toc60826854)

[2.1 功能说明 3](#_Toc60826855)

[2.2 功能设计 3](#_Toc60826856)

[2.3 算法设计 3](#_Toc60826857)

[2.4 模块测试 4](#_Toc60826858)

[3 灰度变换模块的设计与实现 6](#_Toc60826859)

[3.1 功能说明 6](#_Toc60826860)

[3.2 功能设计 6](#_Toc60826861)

[3.2.1 对数变换 6](#_Toc60826862)

[3.2.2 伽马变换 7](#_Toc60826863)

[3.2.3 灰度拉伸 7](#_Toc60826864)

[3.3 算法设计 8](#_Toc60826865)

[3.4 模块测试 9](#_Toc60826866)

[4 直方图模块的设计与实现 10](#_Toc60826867)

[4.1 功能说明 10](#_Toc60826868)

[4.2 功能设计 10](#_Toc60826869)

[4.3 算法设计 10](#_Toc60826870)

[4.4 模块测试 11](#_Toc60826871)

[5 图形化界面的设计与实现 12](#_Toc60826872)

[5.1 功能说明 12](#_Toc60826873)

[5.1.1 界面设计 12](#_Toc60826874)

[5.1.2 信息栏模块 13](#_Toc60826875)

[5.1.3 区域生长功能 13](#_Toc60826876)

[5.2 功能设计 13](#_Toc60826877)

[5.2.1 界面设计 13](#_Toc60826878)

[5.2.2 信息栏模块 15](#_Toc60826879)

[5.2.3 区域生长功能 15](#_Toc60826880)

[5.3 算法设计 16](#_Toc60826881)

[5.3.1 界面设计 16](#_Toc60826882)

[5.3.2 信息栏模块 17](#_Toc60826883)

[5.3.3 区域生长功能 17](#_Toc60826884)

[5.4 模块测试 18](#_Toc60826885)

[6 心得体会 18](#_Toc60826886)

1 功能描述与分析

1.1 功能描述

本次课程设计使用matlab语言编写实现了一个简单的数字图像处理集成系统，系统功能如下：

1、图像文件的打开、显示与保存。

2、图像几何变换，包括放大缩小、平移、旋转、翻转。

3、图像像素变换，包括图像合成、灰度变换（包括对数变换、伽马变换、灰度拉伸）、直方图计算及直方图均衡。

4、图像去噪实现，包括中值滤波、均值滤波。

5、图像模糊和锐化实现，包括空域滤波与频域滤波。

6、图像分割方法的实现，包括阈值法（类间最大距离、类间最大方差）、区域生长法等，分割结果以各个独立的区域呈现。

7、图像压缩与编码的实现，包括图像信息量、客观保真度的计算、霍夫曼编码压缩、无损预测编码压缩及灰度级量化压缩。

1.2 功能分析

本次设计最终产品为一个完整的包含图形化界面的小型数字图像处理平台，具有功能描述中的所有图像处理功能。

由于本次课程设计要求不能使用大部分库函数实现功能，故需要自行编写功能函数实现图像处理；其中部分功能函数的编写较为简单，例如图像集合变换与像素变换；但其他功能较为复杂，例如图像分割、图像压缩与编码；在实现这些复杂功能时，可以查阅库函数的实现方法、互联网相关资料、课程书籍等资料，再进行自行编写。

1.3 组员分工

本组成员：郭心语、杨浠、年哲超。

组长：郭心语

作为项目的管理人员和开发人员，负责整个项目的组织与协调，同时也是项目开发文档和GitHub远程仓库的主负责人。在项目开发方面实现图像的放大缩小，图像的模糊与锐化，以及图像的压缩与编码。并在项目完成后参与制作海报和宣讲ppt。

组员1：杨浠

作为项目的开发人员和测试人员，承担部分开发工作实现图像的平移、旋转、翻转，图像的去噪处理以及图像的分割，并且在完成后对已集成的项目进行测试，确保项目的稳定性，并在项目完成后参与制作海报和宣讲ppt。

组员2：年哲超

作为项目的开发人员和测试与质量保证人员，在项目的开发方面负责项目的应用界面，实现功能模块与界面的衔接工作，除此之外还负责实现图像文件的打开、显示与保存以及图像像素变换，包含图像合成、灰度变换、直方图模块，并承担项目的整合工作，并在整合过程中测试已集成的程序，并发现项目功能模块隐藏的Bug并整理出来, 并在项目完成后参与制作海报和宣讲ppt。

作为团队中负责整个项目整合工作的成员，需要对几乎所有的代码进行重构和整合，需要注意团队内部的交流，最有效地对其他成员所编写的函数进行重写，将其整合进最终系统中，防止无效工作的出现。

在进行团队交流的过程中，我们使用了git作为代码托管工具，其他团队成员完成了功能函数后，上传至代码库分支，我对其进行重构整合进系统后，由团队所有成员进行测试工作，测试完毕后关闭该分支，至此即完成了一个功能模块的开发实现。

作为项目整合人员，还需要频繁关注开发文档。当团队其他成员在进行测试过程中触发了软件bug后，会提交至开发文档；我需要对整合后的系统进行调试，找到触发bug的代码段，确定该部分的负责人，更新开发文档，通知相关人员进行代码修改。

2 图像合成模块的设计与实现

2.1 功能说明

在该系统中，图像合成功能主要是通过对两张图像进行加减乘除运算以实现二者的合成；功能实现的主要难点在于不同尺寸的图片无法进行运算，只能通过处理使二者变换为相同像素尺寸之后，才可以进行运算。

当用户使用该功能时，应能够对需要进行处理的图片进行所需的运算，当用户选取的图片由于其本身的原因无法进行处理时，应抛出异常并给用户反馈，保存用户工作区现场。

2.2 功能设计

在设计该功能的人机交互逻辑时，确定了如下的操作步骤：

1. 用户在主要工作区打开一张图片；
2. 用户点击选择图像合成菜单栏，并选择该菜单栏下的二级按钮，如图像加法；
3. 弹出文件选择框，用户选择需要进行处理的第二张图片；
4. 将第二张图片的大小进行拉伸或压缩，使其与主工作区已经打开的图片一致；
5. 对两张图片进行之前用户选择的操作，如图像加法；
6. 在主工作区显示处理后的图片；

2.3 算法设计

在实现该功能时，主要难点在于对图像进行重新缩放处理，得到所需大小的图像，再进行处理。该功能可以在matlab中使用imresize函数实现，课程设计需要自行编写函数实现。在解决该问题时，借鉴了系统之前实现的图片放大缩小功能的代码；先选取主工作区图片，获得行列数，新建一个该大小的矩阵；接着用双线性插值法计算，得到由选取待处理图片进行缩放后，每个像素点的灰度值。

实现图像缩放的部分核心代码如下：

for Ai=1:nr

for Aj=1:nc

Bi=(Ai-1)/S;

Bj=(Aj-1)/S;

i=fix(Bi); %向零方向取整，求出坐标Bi的整数部分

j=fix(Bj);

u=Bi-i; %求出坐标Bi的小数部分

v=Bj-j;

i=i+1; %这是在矩阵SB上计算的，不是在矩阵B上计算的，竖直方向上有平移量，加1对应B上的i值

j=j+1;

A(Ai,Aj)=(1-u)\*(1-v)\*SB(i,j)+u\*v\*SB(i+1,j+1)+u\*(1-v)\*SB(i+1,j)+(1-u)\*v\*SB(i,j+1); %双线性插值法计算A(Ai,Aj)

end

end

在两张图像大小矩阵相同的情况下，实现图像运算是十分简单的。只需遍历两张图片的所有矩阵点，对二者进行运算后赋值至新的空白矩阵，最终该空白矩阵即为用户所需结果。

图像运算核心代码如下：

[m,n] = size(q); %进行图像加法

s = zeros(m,n); %新建空白矩阵

for i = 1:m

for j = 1:n

s(i,j) = p(i,j) + q(i,j); %进行图像加法

end

end

2.4 模块测试

使用matlab工作区进行图像加法代码测试，结果如下：

表2-1 图像加法模块测试记录表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 测试用例编号 | Com001 | | 版本号 | | V 0.8 |
| 测试环境 | Matlab工作区 | | | | |
| 用例名称 | Einstein.tif 、lenna.tif | | | | |
| 测试步骤 | 使用lenna与einstein灰度图片进行图像加法测试 | | | | |
| 输入数据 | 两张尺寸不同的标准灰度图片 | | | | |
| 预期输出 | 正确的图像加法结果 | | | | |
| 实际输出 | 正确的图像加法结果 | | | | |
| 问题描述 | 无 | | | | |
| 设计人 | 年哲超 | 设计日期 | | 2021/12/30 | |
| 测试人 | 年哲超 | 测试日期 | | | 2021/12/30 |



图2-1 图像加法测试

使用集成后的系统进行图像合成测试，结果如下：

表2-2 图像合成模块测试记录表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 测试用例编号 | Com002 | | 版本号 | | V 0.8 |
| 测试环境 | Matlab工作区 | | | | |
| 用例名称 | Einstein.tif 、lenna.tif | | | | |
| 测试步骤 | 使用lenna与einstein灰度图片进行图像合成测试 | | | | |
| 输入数据 | 两张尺寸不同的标准灰度图片 | | | | |
| 预期输出 | 正确的图像合成结果 | | | | |
| 实际输出 | 正确的图像合成结果 | | | | |
| 问题描述 | 无 | | | | |
| 设计人 | 年哲超 | 设计日期 | | 2021/12/31 | |
| 测试人 | 年哲超 | 测试日期 | | | 2021/12/31 |



图2-2 其他图像合成功能测试

经简单测试，可以初步认为在该部分编写的代码可以较为成功的实现图像合成功能，未发现问题。在模块全部整合完成后，会进行进一步测试。

3 灰度变换模块的设计与实现

3.1 功能说明

灰度变换是指根据某种目标条件按一定变换关系逐点改变源图像中每一个像素灰度值的方法。目的是为了改善画质，使图像的显示效果更加清晰。在设计开发过程中，选择了实现对数变换、伽马变换、灰度拉伸三个灰度变换模块。

在用户操作过程中，可以在文件选择框选取一张待处理图片，在主工作区打开，接着点选功能菜单栏的灰度变换按钮，对该图片进行灰度变换。

3.2 功能设计

3.2.1 对数变换

对数变换主要用于将图像的低灰度值部分扩展，将其高灰度值部分压缩，以达到强 调图像低灰度部分的目的。变换方法由下式给出。



这里的对数变换，底数为v+1，实际计算的时候，需要用换底公式。其输入为[0,1]其输出也为[0,1]。对于不同的底数，其对应的变换曲线如下图所示。（实际编码过程中将V设置为1）：

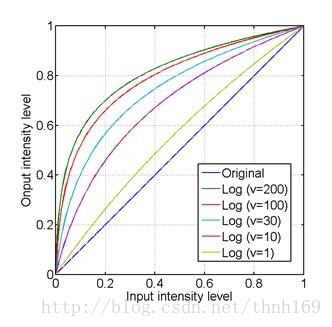


图3-1 对数变换曲线图

3.2.2 伽马变换

伽马变换主要用于图像的校正，将漂白的图片或者是过黑的图片，进行修正。伽马变换可以强调图像的某个部分,伽马变换也常常用于显示屏的校正，这是一个非常常用的变换。其变化所用数学式如下所示：



其输入为[0,1]，其输出也为[0,1]。对于不同的伽马值，其对应的变换曲线如下图所示（实际编码过程中C设置为1）。伽马变换曲线如下图所示：

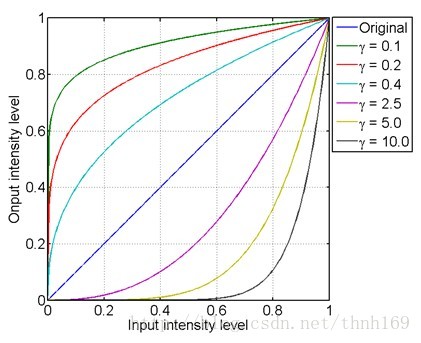
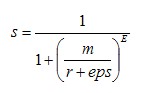


图3-2 伽马变换曲线图

3.2.3 灰度拉伸

灰度拉伸也用于强调图像的某个部分，与伽马变换与对数变换不同的是，灰度拉升可以改善图像的动态范围。可以将原来低对比度的图像拉伸为高对比度图像。实现灰度拉升的方法很多，其中最简单的一种就是线性拉伸。在本次设计过程中，通过实验比较了使用多种灰度拉伸公式处理的图像结果，最终选择了对图像处理效果最明显的一种，公式如下：



灰度拉伸可以简单理解为扩展图片的动态范围，即将[Min(r), Max(r)]范围的灰度点扩展到[0, 1]，在实际实验过程中，若使用matlab对其进行如上式的处理，会造成式中E的值极大，从而使得变换曲线斜率很大，扩展结果差，故在本次设计中取输出范围为[0.05, 0.95]。用这种方式确定的灰度拉伸输入输出曲线如下：

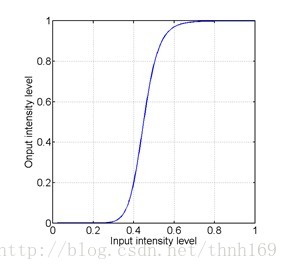


图3-3 灰度拉伸曲线图

3.3 算法设计

编写对数变换与伽马变换的函数难度不大，只需要设定好数值，对图像进行运算即可。核心代码如下：

f = imread('test.tif');

f = mat2gray(f,[0 255]);

v = 10;

g\_1 = log2(1 + v\*f)/log2(v+1);

C = 1;

Gamma = 0.4;

g2 = C\*(f.^Gamma);

figure();

subplot(1,3,1);imshow(f,[0 1]);

subplot(1,3,2);imshow(g\_1,[0 1]);

subplot(1,3,3);imshow(g\_2,[0 1]);

灰度拉伸函数编写的过程相比较为复杂，核心代码如下：

f = mat2gray(pImage,[0 255]);

[M,N] = size(f);

g = zeros(M,N);

Min\_f = min(min(f));

Max\_f = max(max(f));

m = (Min\_f + Max\_f)/2;

Out\_put\_min = 0.05;

Out\_put\_max = 0.95;

E\_1 = log(1/Out\_put\_min - 1)/log(m/(Min\_f+eps));

E\_2 = log(1/Out\_put\_max - 1)/log(m/(Max\_f+eps));

E = ceil(min(E\_1,E\_2)-1);

g = 1 ./(1 + (m ./ (f+ eps)).^E);

g = mat2gray(g,[1/(1+(m/eps)^E) 1/(1+(m/1+eps)^E)]);

imshow(g);

3.4 模块测试

对灰度变换模块进行测试，结果如下：

表3-1 灰度变换模块测试记录表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 测试用例编号 | Map001 | | 版本号 | | V 0.8 |
| 测试环境 | Matlab工作区 | | | | |
| 用例名称 | lenna.tif | | | | |
| 测试步骤 | 使用lenna灰度图片进行灰度映射测试 | | | | |
| 输入数据 | 标准灰度图片 | | | | |
| 预期输出 | 正确的灰度映射结果 | | | | |
| 实际输出 | 正确的灰度映射结果 | | | | |
| 问题描述 | 无 | | | | |
| 设计人 | 年哲超 | 设计日期 | | 2021/12/31 | |
| 测试人 | 年哲超 | 测试日期 | | | 2021/12/31 |

灰度映射测试结果如下：



图3-4 灰度映射测试结果图

4 直方图模块的设计与实现

4.1 功能说明

直方图显示模块主要用以显示图像灰度或RGB直方图。在本次系统设计过程中，确定了直方图的显示在界面右侧信息栏部分，在每次对图像进行处理后，都会刷新图像直方图，用户可以随时查看。则在每次操作后，都会重新调用一次绘制直方图的函数，对该部分算法的时间复杂度有较高的要求。

4.2 功能设计

在对该功能进行设计时，首先采用了遍历整张图片，统计各矩阵点灰度值的方法，对整个图片的灰度值进行统计，再进行绘制；但是后期经测试发现使用这种方式实现的函数由于需要遍历整张图片，使用了两重循环，故其的时间性能略差，用户操作会有较大的卡顿；在查阅资料后，改为使用bars函数、numel函数等进行实现。

4.3 算法设计

直方图计算的核心代码如下：

function bars=histogram(I)

%用==来提取某个灰度的像素

%并用sum来计算个数

tic

bars=zeros(1,256);

for value=0:255

bars(value+1)=sum(value==I(:));

end

bars=bars./numel(I);

toc

编码实现该函数的过程经过多次修改，最终的函数实现如上所示。在不断迭代修改函数的过程中，使用tic与toc命令进行设置计时器，以测试函数性能；通过这种方法，我们实现了时间复杂度较小的直方图绘制功能函数。

4.4 模块测试

对直方图绘制模块进行测试，结果如下：

表4-1 直方图模块测试表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 测试用例编号 | Hist001 | | 版本号 | | V 0.9 |
| 测试环境 | Matlab工作区 | | | | |
| 用例名称 | lenna.tif等 | | | | |
| 前提条件 | 使用tic与toc命令计时 | | | | |
| 测试步骤 | 在工作区使用多种灰度图片使用自编写的函数进行灰度直方图绘制，记录运行时间 | | | | |
| 输入数据 | 不同大小的标准灰度图像 | | | | |
| 预期输出 | 在较小时间内绘制图像直方图 | | | | |
| 实际输出 | 成功绘制图像直方图，时间复杂度高于预期 | | | | |
| 问题描述 | 函数时间性能较差 | | | | |
| 设计人 | 年哲超 | 设计日期 | | 2021/1/1 | |
| 测试人 | 年哲超 | 测试日期 | | | 2021/1/1 |
| 再测试人 | 年哲超 | 再测试日期 | | | 2021/1/3 |
| 问题修改摘要 | 重写了直方图绘制函数，减小时间复杂度 | | | | |
| 修改人 | 年哲超 | | 修改日期 | | 2021/1/3 |

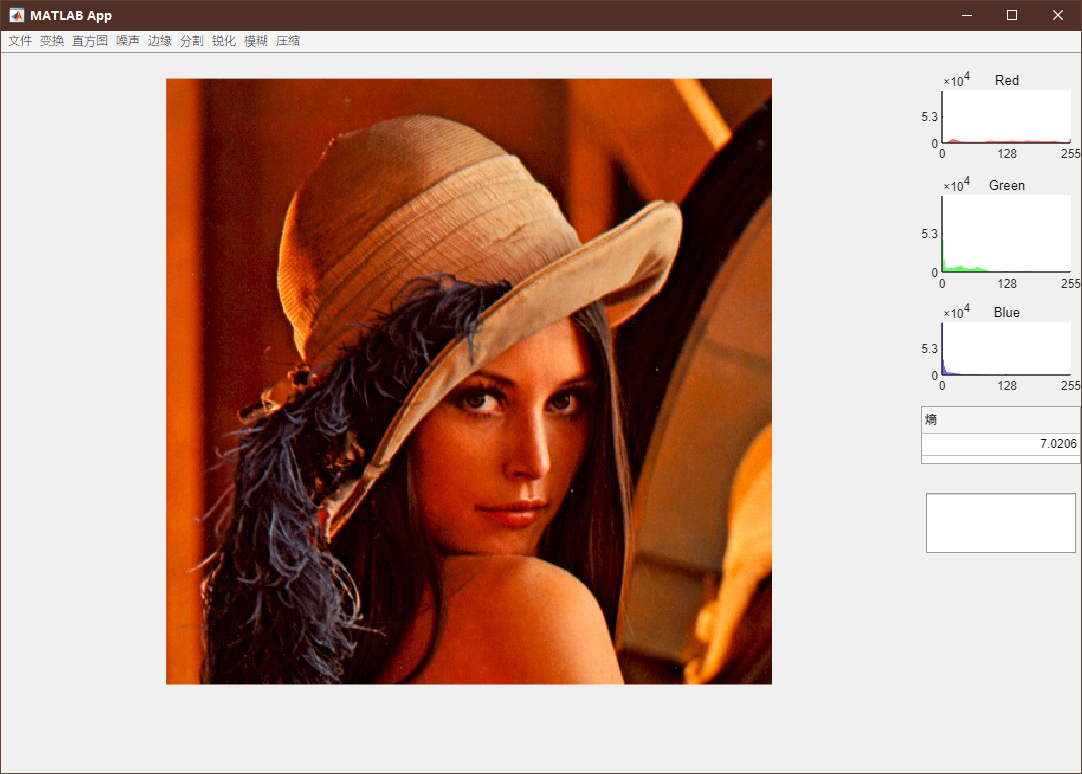


图4-1 直方图模块最终效果图

5 图形化界面的设计与实现

5.1 功能说明

本次课程设计中，我的主要工作是实现软件的图形化界面。主要工作内容如下。

5.1.1 界面设计

在对整体的图形化界面设计的过程中，团队内所有成员都进行了讨论，参照了市面上大多数主流图形编辑软件，最终确定了图形界面整体开发思路与操作逻辑如下：

1. 图形化界面主要部分为操作区，显示正在处理的图片；
2. 主要功能通过菜单栏的方式实现，用户点击菜单栏按钮则触发功能函数；
3. 界面右侧为软件信息栏，显示图形直方图与计算得到的熵值，每次对图形进行处理后重新调用该部分的函数进行刷新；
4. 图形化界面的编写需要注意异常处理，若软件使用过程中出现操作错误或未发现的bug，以用户友好的方式弹出提示框；

5.1.2 信息栏模块

在图形界面设计过程中，确定了软件界面右侧为软件信息栏，显示已打开图像文件的各种信息，如直方图等；在每次对图像进行处理后会调用这部分的功能函数重新计算，刷新信息栏的值；由于该功能的调用次数很多，为防止用户操作卡顿，则对信息栏参数计算函数的时间复杂度要求较高；在实际实现过程中，经历了多次函数重写，最终较为成功地实现了该功能。

5.1.3 区域生长功能

在本次软件设计中实现区域生长功能时，团队成员进行了讨论，得到结论：在用户的实际使用时，若使用输入灰度值参数选取生长种子点的方式，则要求用户对需要处理的图像十分熟悉，了解其内部各点的灰度值，才能正确使用该功能分割出其希望分割的区域，软件使用友好度较差；团队讨论决定将该功能设计为使用鼠标选取图像某点，提取该点灰度值作为种子点再进行区域生长的方式，设计实现该功能。在软件的最终版本中实现了该功能。

5.2 功能设计

5.2.1 界面设计

在进行界面设计时，编码主要思路如下：

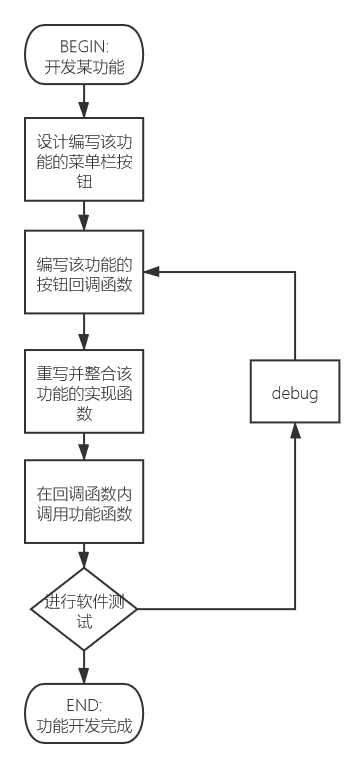


图5-1 界面设计开发思路

软件最终版本的主要界面示意图如下。

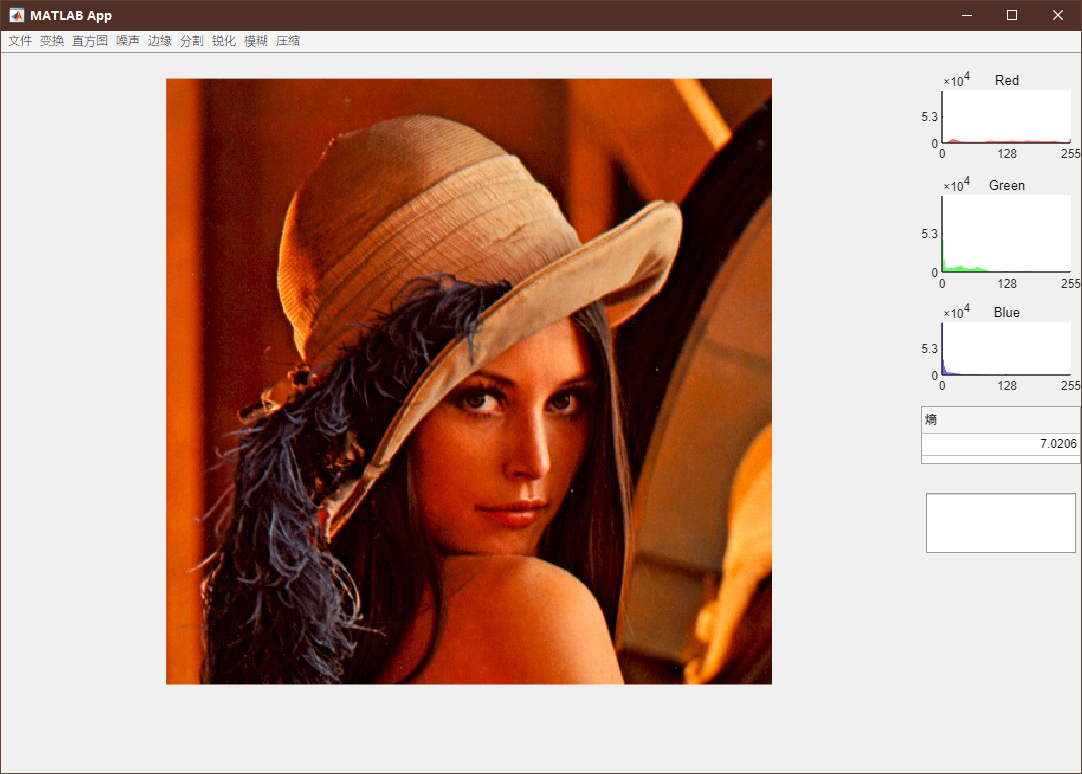


图5-2 软件图形化界面

5.2.2 信息栏模块

信息栏模块主要显示了图像直方图、图像熵值、对图像进行霍夫曼编码后的码字表三种信息；其中直方图模块的功能函数本人负责编写，另外两个由团队其他成员进行编写。

在实现该功能时，主要的问题在于matlab的app designer工具的回调函数只能接收类似指针的地址信息；故在设计实现时，采用了一个全局变量app.Image主工作区显示的正在处理的图片，在每次调用功能函数刷新右侧信息栏时，在回调函数中使用一个新的变量pImage将全局变量app.Image调入内存，再进行处理。

5.2.3 区域生长功能

在开发区域生长功能的过程中，由团队的杨浠成员编写了区域生长的功能函数，完成后提交至远程仓库，由我进行整合。

在整合过程中，经查阅资料得知，由于app designer工具不够成熟，暂时不支持在独立app内调用impixel进行像素点选取。

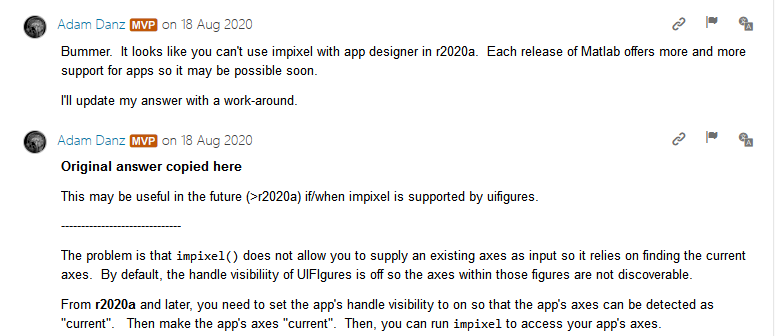


图5-3 matlab官方社群截图

在之后的开发过程中，我们使用了重新弹出绘图窗口，在新的窗口中调用impixel工具进行像素点选取的方法；用这种方法实现不但解决了无法在app内部调用impixel工具选取像素点的问题，而且为用户提供了分割前后的图形对比，操作友好型更高。

5.3 算法设计

5.3.1 界面设计

根据界面开发流程，使用图片打开功能举例描述编码过程如下：

编写菜单栏按钮：

properties (Access = public)

UIFigure matlab.ui.Figure

File matlab.ui.container.Menu

end

function createComponents(app)

% Create File

app.File = uimenu(app.UIFigure);

app.File.Text = '文件';

% Create open

app.open = uimenu(app.File);

app.open.MenuSelectedFcn = createCallbackFcn(app, @openFile, true);

app.open.Text = '打开图片';

end

重写功能函数，在其中补充异常处理部分，修改部分变量名称：

function updateImage(app,imagefile)

try

app.Image = imread(imagefile);

catch ME

uialert(app.UIFigure, ME.message, 'Image Error'); %异常处理

return;

end

end

编写按钮回调函数：

function openFile(app, event)

filterspec = {'\*.jpg;\*.tif;\*.png;\*.gif','All Image Files'};

[f, p] = uigetfile(filterspec);

if (ischar(p))

fname = [p f];

updateImage(app, fname);

showHist(app);

end

end

5.3.2 信息栏模块

与上文描述的信息栏功能设计过程相对应，编码过程如下：

初始化存储主工作区图片的全局变量：

properties (Access = private)

Image % the Processing Image

end

在回调函数中将该全局变量调入内存，调用功能函数进行处理，刷新信息栏模块，再改变全局变量的值：（以图像垂直翻转功能举例）

function reverseMenu(app, event)

pImage = app.Image; %将图片调入内存

pImage = CustomReverseY(app,pImage); %调用功能函数

app.Image = pImage; %更新全局变量

imshow(app.Image,'Parent',app.ImageAxes); %刷新主工作区图片

showHist(app); %刷新直方图

end

5.3.3 区域生长功能

在编码实现区域生长功能的过程中，直接新建窗口，调用impixel工具，在新的窗口内进行像素点的选择；虽然最后实现使用的代码十分简洁，但解决该问题查找资料与测试花费了很多精力与时间。

function Menu\_24Selected(app, event)

pImage = app.Image;

if( ~( size(pImage,3)-3 ))

pImage = rgb2gray(pImage); %转化为单通道灰度图

end

[y, x, ~] = impixel(app.Image); %选取像素点

pImage = grow2(app,pImage,x,y); %调用区域生长函数

app.Image = pImage;

imshow(app.Image,'Parent',app.ImageAxes);

showHist(app);

end

区域生长功能最终在图形化的实现方式如下图：

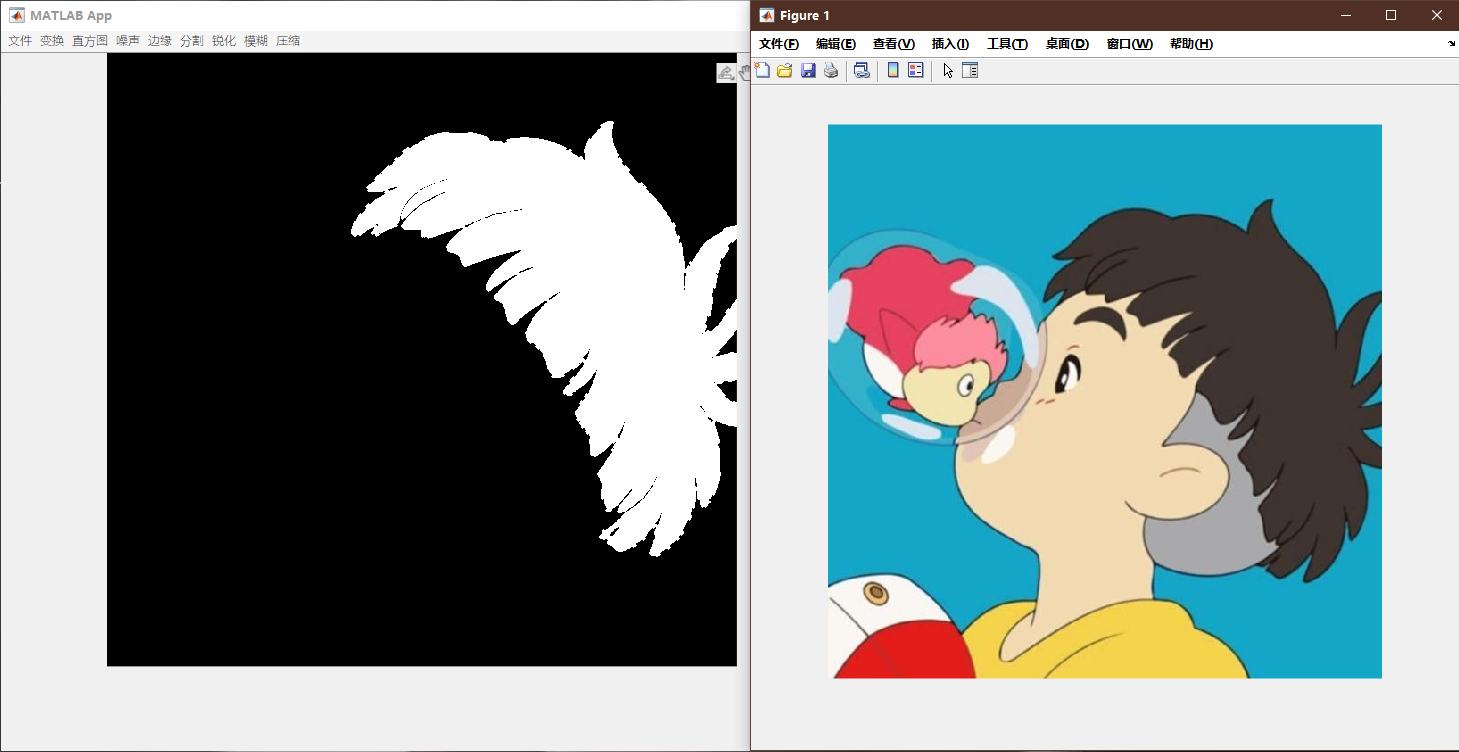


图5-4 区域生长功能最终实现方式

5.4 模块测试

在完成了整个系统的开发后，我们对系统的图形化界面进行了完整的测试，并修改了测试过程中触发的bug。

表5-1 图形界面测试

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 测试用例编号 | UI001 | | 版本号 | | V 0.9 |
| 测试环境 | Windows操作系统 | | | | |
| 测试步骤 | 使用测试图片集在系统软件中进行各种处理 | | | | |
| 输入数据 | 包含不同格式、大小的各种灰度图片、RGB图片 | | | | |
| 预期输出 | 正确对图形进行处理后的图片 | | | | |
| 实际输出 | 正确对图形进行处理后的图片 | | | | |
| 问题描述 | 无 | | | | |
| 设计人 | 杨浠 | 设计日期 | | 2021/1/3 | |
| 测试人 | 杨浠 | 测试日期 | | | 2021/1/3 |
| 再测试人 | 郭心语 | 再测试日期 | | | 2021/1/4 |

6 心得体会

在本次课程设计中，本团队对数字图像处理平台进行了较为高效的开发，最终开实现了一个简易的图像处理软件。

在开发过程中我通过文献查阅、实验、经验提炼等多种手段，结合专业知识设计了较为新颖的解决方案。

在系统开发过程中，我不但运用了本学期学到的数字图像处理的知识，还综合运用了本专业之前所学习的许多其他知识，例如面向对象编程、数据结构等。

在团队合作时，我们的沟通十分高效。本团队使用了git工具，很好地做到了不同代码的版本控制；同时，在进行软件测试的过程中，使用在线开发文档，发现bug后及时找到该部分的负责人，通知其进行代码修改；我们在本次课程设计中也收获了珍贵的友谊。