**功能函数**

function infostring = CreateImgInfoSting(~, filename, imginfo)

% Brief: 返回图像信息字符串

% Input: imginfo 图像信息

% Output infostring 图像信息字符串

% Usage Sample: DispImgInfo(app.file, imginfo)

switch imginfo.ColorType

case 'truecolor'

ColorType = '真彩色图';

case 'grayscale'

ColorType = '灰度图';

case 'indexed'

ColorType = '索引图';

otherwise

ColorType = '图像类型未知';

end

infostring = {strcat('图像名称:',32,32,filename),strcat('图像格式:',32,32,imginfo.Format),strcat('图像宽度:',32,32,num2str(imginfo.Width))...

strcat('图像高度:',32,32,num2str(imginfo.Height)),strcat('图像大小:',32,32,num2str(imginfo.FileSize),'字节')...

strcat('像素位数:',32,32,num2str(imginfo.BitDepth)),strcat('图像类型:',32,32,ColorType)};

end

function [MaxVal, MinVal, Avr, Var] = GetImgParams(~,ImgInput)

% Input: ImgInput 输入图像

% Output: MaxVal 图像当中的最大值

% MinIndex 图像当中最小值所在位置

% Avr 图像所有数据的平均值

% Var 图像所有数据方差

% Usage Sample: ......

[h,w,dims] = size(ImgInput);

%一定要对数据做处理,转换成double类型数据,否则计算结果误差较大,这是由于uint8类型数据的舍入误差引起的

temp = double(ImgInput);

MaxVal = max(max(max(temp)));

MinVal = min(min(min(temp)));

result = sum(sum(sum(temp)));

Avr = result / (h\*w\*dims);

Var = sum(sum(sum((temp-Avr).^2))) / (h\*w\*dims);

end

function hist = plotgrayhist(~,ImgInput)

% Input: ax 绘图区域

% ImgInput 输入图像

% Output: hist 输入图像的直方图数据,可用bar画图

% Usage Sample: ......

%求取图像大小

[h,w,dims] = size(ImgInput);

%将灰度矩阵+1避免后续hist(0)类型的索引出现

temp = ImgInput + 1;

%判断是否为RGB图像

if dims == 3

model = 1;

else

model = 2;

end

%初始化灰度直方图数组

hist = zeros(256,1);

switch model

case 1

for i=1:3

for j=1:w

for k=1:h

%统计灰度直方图

hist(temp(k,j,i)) = hist(temp(k,j,i))+1;

end

end

end

case 2

for j=1:w

for k=1:h

%统计灰度直方图

hist(temp(k,j)) = hist(temp(k,j))+1;

end

end

end

x = 0:1:255;

bar(x,hist);

end

function ImgOut = ImgShift(~,ImgInput,ShiftScale)

% Input: ImgInput 输入图像

% ShiftScale 水平垂直方向上的尺度,一个包含两个元素的向量

% 第一个元素决定左右方向上的平移,负号表示向左,正号表示向右

% 第二个元素决定上下方向上的平移,负号表示向下,正号表示向上

% Output: ImgOut 经过平移后的图像数据

% Usage Sample: ImgNew = ImgShift(MyImg, [26,5])

% 表示将MyImg图像向右平移26个单位,向上平移5个单位,得到ImgNew图像

%获取输入图像的尺寸

[h,w,dims] = size(ImgInput);

ImgOut = zeros(h, w, dims,'uint8');

if ShiftScale(1) >= 0 && ShiftScale(2) >= 0 %向右上平移

for i=1:h-ShiftScale(2)

for j=1:w-ShiftScale(1)

ImgOut(i,j+ShiftScale(1),:) = ImgInput(i+ShiftScale(2),j,:);

end

end

elseif ShiftScale(1) >= 0 && ShiftScale(2) <= 0 %向右下平移

for i=1:h-abs(ShiftScale(2))

for j=1:w-(ShiftScale(1))

ImgOut(i+abs(ShiftScale(2)),j+ShiftScale(1),:) = ImgInput(i,j,:);

end

end

elseif ShiftScale(1) <= 0 && ShiftScale(2) >= 0 %向左上平移

for i=1:h-ShiftScale(2)

for j=1:w-abs(ShiftScale(1))

ImgOut(i,j,:) = ImgInput(i+ShiftScale(2),j+abs(ShiftScale(1)),:);

end

end

elseif ShiftScale(1) <= 0 && ShiftScale(2) <= 0 %向左下平移

for i=1:h-abs(ShiftScale(2))

for j=1:w-abs(ShiftScale(1))

ImgOut(i+abs(ShiftScale(2)),j,:) = ImgInput(i,j+abs(ShiftScale(1)),:);

end

end

end

end

function ImgOut = ImgReverse(~, ImgInput, dir)

% Input: ImgInput 输入图像

% dir 颠倒方向

% 字符串"vertical"表示垂直方向

% 字符串"level"表示水平方向

% Output: ImgOut 经过颠倒后的图像数据

% Usage Sample: ImgNew = ImgReverse(MyImg, 'vertical')

% 表示将MyImg图像垂直翻转

[h,w,dims] = size(ImgInput);

ImgOut = zeros(h,w,dims,'uint8');

switch dir

case 'level'

ImgOut = ImgInput;

for i=1:3

temp = ImgOut(:,:,i)';

temp = flip(temp);

ImgOut(:,:,i) = temp';

end

case 'vertical'

ImgOut = flip(ImgInput);

end

end

function ImgOut = ImgShear(~,ImgInput,dir)

% Input: ImgInput 输入图像

% dir 切变方向

% 第一个元素决定水平方向上的切变,负号表示向左,正号表示向右

% 第二个元素决定垂直方向上的平移,负号表示向下,正号表示向上

% Output: ImgOut 经过切变后的图像数据

% Usage Sample: ImgNew = ImgShear(MyImg, [0,0.2])

% 表示将MyImg图像沿着垂直方向向上切变0.2个单位

%获取输入图像的尺寸

[h,w,dims] = size(ImgInput);

k1 = dir(1);

k2 = dir(2);

hnew = h + floor(abs(abs(k2))\*w);

wnew = w + floor(abs(abs(k1))\*h);

ImgOut = zeros(hnew, wnew,dims,'uint8');

%切变公式：

%向右上切变: x=x1+k1(h-y1);y=y1+k2(w-x1)

%向左下切变: x=x1+k1y1;y=y1+k2x1

%向右下切变: x=x1+k1(h-y1);y=y1+k2x1

%向左上切变: x=x1+k1y1;y=y1+k2(w-x1)

%其中h为原图像高度,w为原图像宽度,x,y分别为新图横纵坐标,x1,y1分别为原图横纵坐标,k1,k2分别为水平和垂直方向上的切变系数

for i=1:hnew

for j=1:wnew

%向右上切变，由新图像到原图像的映射

%x1=(x+k1y-k1h-k1k2w)/(1-k1k2);y1=(y+k2(x-w)-k1k2h)/(1-k1k2)

if k1 >= 0 && k2 >= 0

x = round((j+k1\*(i-h)-k1\*k2\*w) / (1-k1\*k2));

y = round((i+k2\*(j-w)-k1\*k2\*h) / (1-k1\*k2));

if x > 0 && x <= w && y > 0 && y <= h

ImgOut(i,j,:) = ImgInput(y,x,:); %最邻近插值

end

elseif k1 <= 0 && k2 <= 0 %向左上切变 公式为 x1=(x-k1y)/(1-k1k2);y1=(y-k2x)/(1-k1k2),

x = round((j-abs(k1)\*i) / (1-abs(k1\*k2)));

y = round((i-abs(k2)\*j) / (1-abs(k1\*k2)));

if x > 0 && x <= w && y > 0 && y <= h

ImgOut(i,j,:) = ImgInput(y,x,:); %最邻近插值

end

elseif k1 >= 0 && k2 <= 0 %向右下切变 公式为 x1=(x+k1y-k1h)/(1+k1k2);y1=(y-k2x+k1k2h)/(1+k1k2)

x = round((j+k1\*(i-h)) / (1+abs(k1\*k2)));

y = round((i+abs(k2)\*(k1\*h-j))/(1+abs(k1\*k2)));

if x > 0 && x <= w && y > 0 && y <= h

ImgOut(i,j,:) = ImgInput(y,x,:); %最邻近插值

end

elseif k1 <= 0 && k2 >= 0 %向左下切变 公式为 x1=(x-k1y+k1k2w)/(1+k1k2);y=y1+k(w-x)

x = round((j-abs(k1)\*i+abs(k1\*k2)\*w) / (1+abs(k1\*k2)));

y = round((i+k2\*j-abs(k2)\*w)/(1+abs(k1\*k2)));

if x > 0 && x <= w && y > 0 && y <= h

ImgOut(i,j,:) = ImgInput(y,x,:); %最邻近插值

end

end

end

end

end

function ImgOut = ImgRotate(~, ImgInput, deg)

% Input: ImgInput 输入图像

% deg 旋转度数,大于0表示顺时针,小于零表示逆时针

% Output: ImgOut 经过旋转后的图像数据

% Usage Sample: ImgNew = ImgRotate(MyImg, -45)

%获取输入图像的尺寸

[h,w,dims] = size(ImgInput);

cosdeg = cosd(deg); sindeg = sind(deg);

%根据旋转中心及旋转角度计算旋转后的图像大小

hnew = round(h \* abs(cosdeg) + w \* abs(sindeg));

wnew = round(h \* abs(sindeg) + w \* abs(cosdeg));

ImgOut = zeros(hnew,wnew,dims,'uint8');

%旋转矩阵

rot = [cosdeg, -sindeg

sindeg, cosdeg];

%旋转原理说明：

%由旋转公式可得：

% x = x0\*cosθ - y0\*sinθ

% y = x0\*sinθ + y0\*cosθ

% rot = [cosθ -sinθ

% sinθ cosθ]即为旋转因子

% [x;y] = rot \* [x0;y0] => [x0;y0] = inv(rot) \* [x;y] 或 rot \ [x;y]

%而它是在原点为(0,0)下的旋转因子,在对图像作旋转时,选取图像中心点为旋转中心较为方便

%此时以(w/2,h/2)为原点,所以此时有 x0 = xr - w/2, y0 = yr - h/2

%同理,其得到的x,y也是以变换后的图像中心为原点的坐标,而旋转后的图像中心为(wnew/2,hnew/2),

%所以有 x = xnr - wnew/2, y = ynr - hnew/2, 所以将图像都以中心为原点后可以统一旋转点，

%得到(x,y)与(x0,y0)的关系,再根据坐标转换把他们转换到以(0,0)(或(1,1))为原点的原始图像中去,

%就能得到两幅旋转前后的原始图像之间的映射关系,再利用反映射方法,在ImgInput里面寻找由

%ImgOut反映射回去的点是否存在,若存在,即得到一组映射点,即找对点位,同理将所有点的对应位置找完整,即可得到旋转后的图像

for i = 1:wnew

for j = 1:hnew

%根据新坐标系下的点位推算原坐标点位

temp = rot \ [i-wnew/2; j-hnew/2];

x = round(temp(1,1))+round(w/2);

y = round(temp(2,1))+round(h/2);

if y > 0 && y <= h && x > 0 && x <= w

ImgOut(j, i, :) = ImgInput(y, x, :); %按照最近邻插值法获取图像数据

end

end

end

end

function ImgOut = ImgZoom(~,ImgInput,scale)

% Brief: 图像缩放及双线性差值

% Input: filename 输入图像名称

% scale 缩放比例

% Output: output 经过缩放及插值的图像

% Usage Sample:

%

%获取图像的尺寸

[m,n,dims] = size(ImgInput);

%创建输出矩阵

if scale <= 0

disp('缩放倍数要大于0！');

else

m\_new = round(m \* scale);

n\_new = round(n \* scale);

output = zeros(m\_new,n\_new,dims);

end

%扩展原图像矩阵

imgx = zeros(m+2,n+2,dims);

%填充中心

imgx(2:m+1,2:n+1,:) = ImgInput;

%填充上下边界两行

imgx(1,2:n+1,:)=ImgInput(1,:,:);imgx(m+2,2:n+1,:)=ImgInput(m,:,:);

%填充左右边界两列

imgx(2:m+1,1,:)=ImgInput(:,1,:);imgx(2:m+1,n+2,:)=ImgInput(:,n,:);

%填充左上角单元格、右上角单元格

imgx(1,1,:) = ImgInput(1,1,:);imgx(1,n+2,:) = ImgInput(1,n,:);

%填充左下角单元格、右下角单元格

imgx(m+2,1,:) = ImgInput(m,1,:);imgx(m+2,n+2,:) = ImgInput(m,n,:);

for oj = 1:n\_new % 对图像进行按列逐元素扫描

for oi = 1:m\_new

ii = (oi-1)/scale; jj = (oj-1)/scale;

i = floor(ii); j = floor(jj); % 向下取整

u = ii - i; v = jj - j;

i = i + 1; j = j + 1;

%公式：f(oi,oj) =

%(1-u)\*(1-v)\*f(i,j)+(1-u)\*v\*f(i,j)+u\*(1-v)\*f(i+1,j,:)+u\*v\*f(i+1,j+1,:)

output(oi,oj,:) = (1-u)\*(1-v)\*imgx(i,j,:) +(1-u)\*v\*imgx(i,j+1,:)...

+ u\*(1-v)\*imgx(i+1,j,:) + u\*v\*imgx(i+1,j+1,:);

end

end

for oj = 1:n\_new % 对图像进行按列逐元素扫描

for oi = 1:m\_new

end

end

ImgOut = uint8(output);

end

function Params = ImgTransParamsParse(~,ParamSting,mode)

% Brief: 图像变换参数解析

% Input: ParamSting 图像变换参数源数据(字符串)

% mode 图像变换类型

% Output: Params 解析后的参数

% Usage Sample: Params = ImgTransParamsParse('100,200','shift')

% 解析图像平移参数,向右平移100，向上平移200

switch mode

case 1

temp = split(ParamSting,',');

if length(temp) == 2 && ~isempty(temp{1}) && ~isempty(temp{2})

Params = [str2double(temp{1}),str2double(temp{2})];

else

Params = [0,0];

end

case 2

Params = str2double(ParamSting);

end

end

function ImgOut = ImgGraying(~,img)

% Brief: 图像灰度化

% Input: img 输入图像

% Output: ImgOut 输出图像

% Usage Sample:

dims = ndims(img);

if dims == 3

ImgOut = 0.2989 \* double(img(:,:,1)) + 0.5870 \* double(img(:,:,2)) + 0.1140 \* double(img(:,:,3));

else

ImgOut = img;

end

ImgOut = uint8(ImgOut);

end

function ImgOut = ImgEqualing(~,img)

% Brief: 图像灰均衡化

% Input: img 输入图像

% Output: ImgOut 输出图像

% Usage Sample:

[M,N,dims]=size(img);

grey=zeros(1,256);

prob\_accum=zeros(1,256);

eq\_img = zeros(M,N,dims);

%统计各个灰度级别像素个数

for k=1:dims

for i=1:M

for j=1:N

grey(img(i,j,k)+1)=grey(img(i,j,k)+1)+1;

end

end

grey\_p=grey/(M\*N);%灰度分布概率

%灰度累计的分布概率

for a=2:256

prob\_accum(a)=prob\_accum(a-1)+grey\_p(a);

end

%将原来 f 的灰度值映射到新的灰度值

for i = 1:M

for j = 1:N

%灰度分布概率乘以255得到具体的灰度值

eq\_img(i,j,k)=floor(prob\_accum(img(i,j,k)+1)\*255);

end

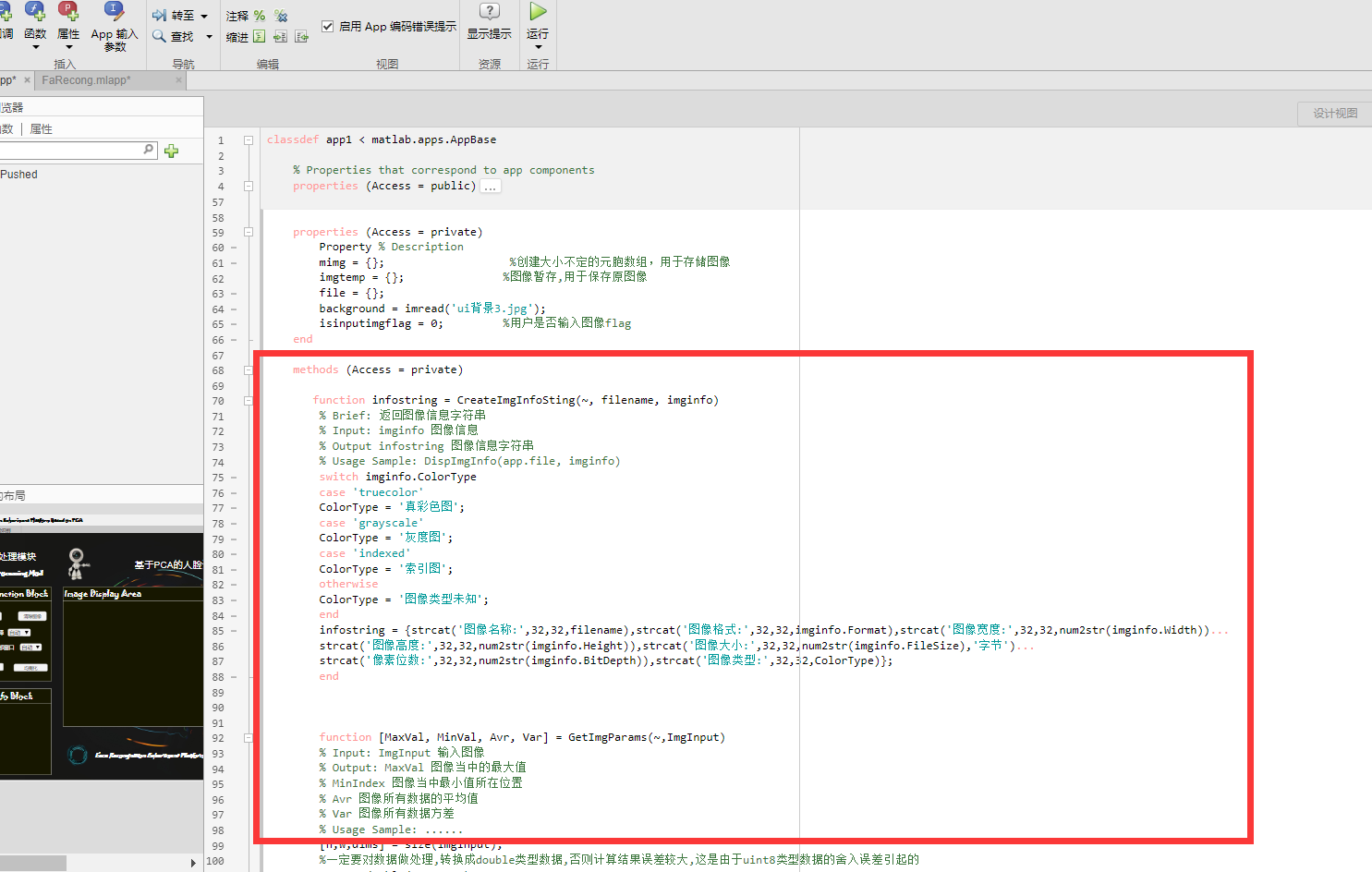
end

end

ImgOut=uint8(eq\_img);

end

不知道复制到哪里去可以参考这里图片。由此，所需要用到的函数和属性都已经准备好，接下来开始添加回调函数。



4.输入图像：



如图所示添加相应的回调函数，然后将下面的代码粘贴到函数当中去：

[app.file, path] = uigetfile({'\*.jpg';'\*.png';'\*.jpeg';'\*.bmp';'\*.\*'},'Choose your file','MultiSelect','on'); %读取文件后执行下列程序

figure(app.UIFigure);

if iscell(app.file) == 1 %判断是否输入多个图像

app.img = {}; %创建元胞数组

for i=1:length(app.file)

app.img{i} = imread(strcat(path, app.file{i}));

end

imginfo = imfinfo(strcat(path, app.file{1}));

axes1 = subplot(1,1,1,'Parent',app.ImageDisplayAreaPanel); %创建作图区域

imshow(app.img{1},'Parent',axes1,'InitialMagnification','fit');

app.ImgInfo.Text = app.CreateImgInfoSting(app.img{1},imginfo);

app.isinputimgflag = 1;

app.imgtemp = app.img{1};

elseif app.file ~= 0 %避免点击按钮但是未输入图像的情况

app.img = imread(strcat(path, app.file));

imginfo = imfinfo(strcat(path, app.file));

axes1 = subplot(1,1,1,'Parent',app.ImageDisplayAreaPanel);

imshow(app.img,'Parent',axes1,'InitialMagnification','fit'); %显示图像

app.ImgInfo.Text = app.CreateImgInfoSting(app.file,imginfo);

app.isinputimgflag = 1;

app.imgtemp = app.img;

end

结果如图所示：



5.清除图像：



将下面的代码粘贴到回调函数中去：

if app.isinputimgflag == 1

app.img = 0;

app.imgtemp = 0;

axes1 = subplot(1,1,1,'Parent',app.ImageDisplayAreaPanel);

imshow(app.background,'Parent',axes1,'InitialMagnification','fit'); %显示图像

axes2 = subplot(1,1,1,'Parent',app.ImageDisplayAreaPanel\_2);

imshow(app.background,'Parent',axes2,'InitialMagnification','fit'); %显示图像

app.ImgInfo.Text = 'Please input your image ~';

app.ImgParams.Text = 'Please input your image ~';

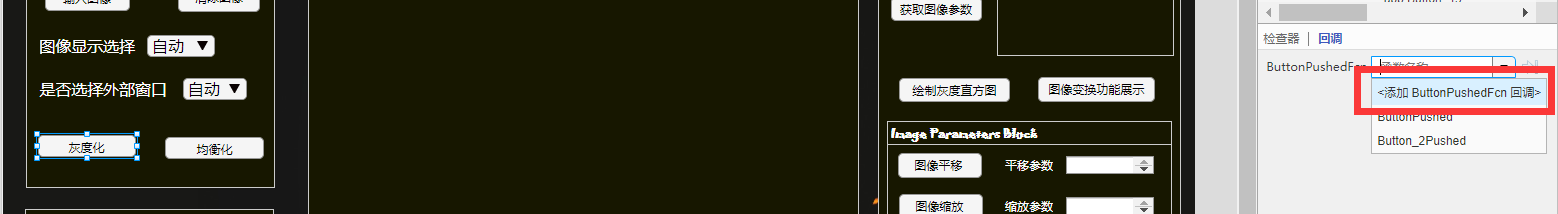
app.isinputimgflag = 0;

end

得到如下结果：



6.灰度化：



代码如下：

if app.isinputimgflag == 1

%图像灰度化并作图,多个输入

if iscell(app.img) == 1

%划分区域显示图像

app.img{1} = app.imgtemp;

app.img{1} = app.ImgGraying(app.img{1});

switch app.ImgDispModDropDown.Value

case "自动"

axes1 = subplot(1,2,1,'Parent',app.ImageDisplayAreaPanel);

imshow(app.imgtemp,'Parent',axes1,'InitialMagnification','fit'); %显示灰度化前的图像

title('原图',"Parent",axes1,"FontName","楷体","FontSize",16,"Color",[1,1,1],"FontWeight","bold");

axes2 = subplot(1,2,2,'Parent',app.ImageDisplayAreaPanel);

imshow(app.img{1},'Parent',axes2,'InitialMagnification','fit'); %显示灰度化后的图像

title('灰度化图像',"Parent",axes2,"FontName","楷体","FontSize",16,"Color",[1,1,1],"FontWeight","bold");

case "原图"

axes = subplot(1,1,1,'Parent',app.ImageDisplayAreaPanel);

imshow(app.imgtemp,'Parent',axes,'InitialMagnification','fit'); %显示灰度化前的图像

title('原图',"Parent",axes,"FontName","楷体","FontSize",16,"Color",[1,1,1],"FontWeight","bold");

otherwise

axes = subplot(1,1,1,'Parent',app.ImageDisplayAreaPanel);

imshow(app.img{1},'Parent',axes,'InitialMagnification','fit'); %显示灰度化后的图像

title('灰度化图像',"Parent",axes,"FontName","楷体","FontSize",16,"Color",[1,1,1],"FontWeight","bold");

end

%是否选择外部窗口

if app.IsChooseFigureDropDown.Value == "是"

switch app.ImgDispModDropDown.Value

case "自动"

figure;

subplot(1,2,1);imshow(app.imgtemp);

title('原图');

subplot(1,2,2);imshow(app.img{1});

title('灰度化图像');

case "原图"

figure;

imshow(app.imgtemp);title('原图');

title('原图');

otherwise

figure;

imshow(app.img{1});

title('灰度化的图像');

end

end

%单个输入

else

app.img = app.imgtemp;

app.img = app.ImgGraying(app.img);

switch app.ImgDispModDropDown.Value

case "自动"

axes1 = subplot(1,2,1,'Parent',app.ImageDisplayAreaPanel);

imshow(app.imgtemp,'Parent',axes1,'InitialMagnification','fit'); %显示灰度化前的图像

title('原图',"Parent",axes1,"FontName","楷体","FontSize",16,"Color",[1,1,1],"FontWeight","bold");

axes2 = subplot(1,2,2,'Parent',app.ImageDisplayAreaPanel);

imshow(app.img,'Parent',axes2,'InitialMagnification','fit'); %显示灰度化后的图像

title('灰度化图像',"Parent",axes2,"FontName","楷体","FontSize",16,"Color",[1,1,1],"FontWeight","bold");

case "原图"

axes = subplot(1,1,1,'Parent',app.ImageDisplayAreaPanel);

imshow(app.imgtemp,'Parent',axes,'InitialMagnification','fit'); %显示灰度化前的图像

title('原图',"Parent",axes,"FontName","楷体","FontSize",16,"Color",[1,1,1],"FontWeight","bold");

otherwise

axes = subplot(1,1,1,'Parent',app.ImageDisplayAreaPanel);

imshow(app.img,'Parent',axes,'InitialMagnification','fit'); %显示灰度化后的图像

title('灰度化图像',"Parent",axes,"FontName","楷体","FontSize",16,"Color",[1,1,1],"FontWeight","bold");

end

%是否选择外部窗口

if app.IsChooseFigureDropDown.Value == "是"

switch app.ImgDispModDropDown.Value

case "自动"

figure;

subplot(1,2,1);imshow(app.imgtemp);

title('原图');

subplot(1,2,2);imshow(app.img);

title('灰度化图像');

case "原图"

figure;

imshow(app.imgtemp);title('原图');

title('原图');

otherwise

figure;

imshow(app.img);

title('灰度化图像');

end

end

end

end

结果如图所示：



7.均衡化：

添加回调函数，将下面的代码添加进去：

if app.isinputimgflag == 1

%图像均衡化并作图,多个输入

if iscell(app.img) == 1

%划分区域显示图像

app.img{1} = app.imgtemp;

app.img{1} = app.ImgEqualing(app.img{1});

switch app.ImgDispModDropDown.Value

case "自动"

axes1 = subplot(1,2,1,'Parent',app.ImageDisplayAreaPanel);

imshow(app.imgtemp,'Parent',axes1,'InitialMagnification','fit'); %显示均衡化前的图像

title('原图',"Parent",axes1,"FontName","楷体","FontSize",16,"Color",[1,1,1],"FontWeight","bold");

axes2 = subplot(1,2,2,'Parent',app.ImageDisplayAreaPanel);

imshow(app.img{1},'Parent',axes2,'InitialMagnification','fit'); %显示均衡化后的图像

title('均衡化图像',"Parent",axes2,"FontName","楷体","FontSize",16,"Color",[1,1,1],"FontWeight","bold");

case "原图"

axes = subplot(1,1,1,'Parent',app.ImageDisplayAreaPanel);

imshow(app.imgtemp,'Parent',axes,'InitialMagnification','fit'); %显示均衡化前的图像

title('原图',"Parent",axes,"FontName","楷体","FontSize",16,"Color",[1,1,1],"FontWeight","bold");

otherwise

axes = subplot(1,1,1,'Parent',app.ImageDisplayAreaPanel);

imshow(app.img{1},'Parent',axes,'InitialMagnification','fit'); %显示均衡化后的图像

title('均衡化图像',"Parent",axes,"FontName","楷体","FontSize",16,"Color",[1,1,1],"FontWeight","bold");

end

%是否选择外部窗口

if app.IsChooseFigureDropDown.Value == "是"

switch app.ImgDispModDropDown.Value

case "自动"

figure;

subplot(1,2,1);imshow(app.imgtemp);

title('原图');

subplot(1,2,2);imshow(app.img{1});

title('均衡化图像');

case "原图"

figure;

imshow(app.imgtemp);title('原图');

title('原图');

otherwise

figure;

imshow(app.img{1});

title('均衡化的图像');

end

end

%单个输入

else

app.img = app.imgtemp;

app.img = app.ImgEqualing(app.img);

switch app.ImgDispModDropDown.Value

case "自动"

axes1 = subplot(1,2,1,'Parent',app.ImageDisplayAreaPanel);

imshow(app.imgtemp,'Parent',axes1,'InitialMagnification','fit'); %显示均衡化前的图像

title('原图',"Parent",axes1,"FontName","楷体","FontSize",16,"Color",[1,1,1],"FontWeight","bold");

axes2 = subplot(1,2,2,'Parent',app.ImageDisplayAreaPanel);

imshow(app.img,'Parent',axes2,'InitialMagnification','fit'); %显示均衡化后的图像

title('均衡化图像',"Parent",axes2,"FontName","楷体","FontSize",16,"Color",[1,1,1],"FontWeight","bold");

case "原图"

axes = subplot(1,1,1,'Parent',app.ImageDisplayAreaPanel);

imshow(app.imgtemp,'Parent',axes,'InitialMagnification','fit'); %显示均衡化前的图像

title('原图',"Parent",axes,"FontName","楷体","FontSize",16,"Color",[1,1,1],"FontWeight","bold");

otherwise

axes = subplot(1,1,1,'Parent',app.ImageDisplayAreaPanel);

imshow(app.img,'Parent',axes,'InitialMagnification','fit'); %显示均衡化后的图像

title('均衡化图像',"Parent",axes,"FontName","楷体","FontSize",16,"Color",[1,1,1],"FontWeight","bold");

end

%是否选择外部窗口

if app.IsChooseFigureDropDown.Value == "是"

switch app.ImgDispModDropDown.Value

case "自动"

figure;

subplot(1,2,1);imshow(app.imgtemp);

title('原图');

subplot(1,2,2);imshow(app.img);

title('均衡化图像');

case "原图"

figure;

imshow(app.imgtemp);title('原图');

title('原图');

otherwise

figure;

imshow(app.img);

title('均衡化图像');

end

end

end

end

结果如下：



8.获取图像参数：



代码：

if app.isinputimgflag == 1

if iscell(app.img) == 1

[MaxVal, MinVal, Avr, Var] = GetImgParams(app,app.img{1});

elseif app.isinputimgflag == 1

[MaxVal, MinVal, Avr, Var] = GetImgParams(app,app.img);

end

ImgParamStrings = {strcat('最大灰度值:',32,32,num2str(MaxVal)),strcat('最小灰度值:',32,32,num2str(MinVal))...

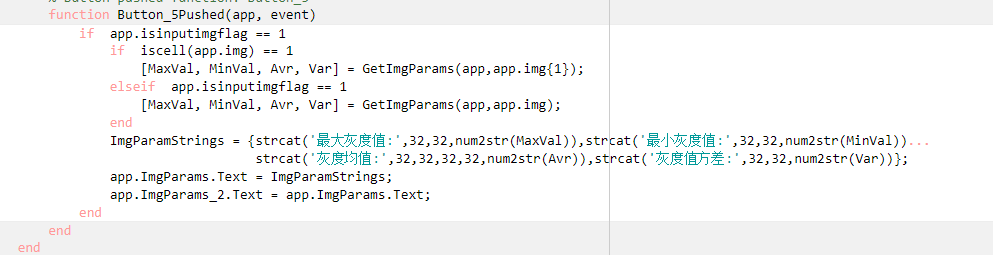
strcat('灰度均值:',32,32,32,32,num2str(Avr)),strcat('灰度值方差:',32,32,num2str(Var))};

app.ImgParams.Text = ImgParamStrings;

app.ImgParams\_2.Text = app.ImgParams.Text;

end

结果如下：



9. 绘制灰度直方图

代码：

if iscell(app.img) == 1 && app.isinputimgflag == 1

figure;app.plotgrayhist(app.img{1});

title(strcat(app.file,'灰度直方图'));xlabel('灰度值(0~255)');ylabel('个数');

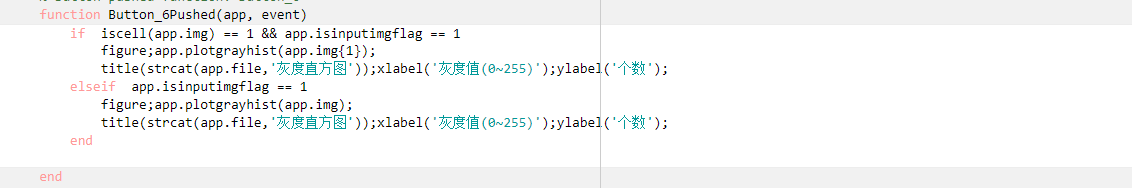
elseif app.isinputimgflag == 1

figure;app.plotgrayhist(app.img);

title(strcat(app.file,'灰度直方图'));xlabel('灰度值(0~255)');ylabel('个数');

end

结果如下：



10.图像变化功能展示：



代码：

if app.isinputimgflag == 1

figure;

subplot(2,3,1);imshow(app.imgtemp);title('原图');

subplot(2,3,2);imshow(app.ImgShift(app.imgtemp,[100,200]));title('平移图像(100,200)');

subplot(2,3,3);imshow(app.ImgZoom(app.imgtemp,1.5));title('放大1.5倍的图像');

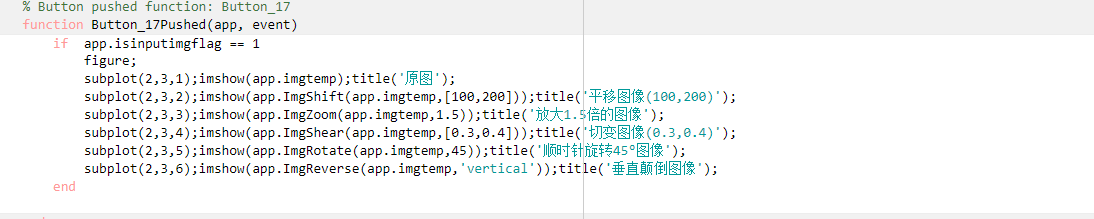
subplot(2,3,4);imshow(app.ImgShear(app.imgtemp,[0.3,0.4]));title('切变图像(0.3,0.4)');

subplot(2,3,5);imshow(app.ImgRotate(app.imgtemp,45));title('顺时针旋转45°图像');

subplot(2,3,6);imshow(app.ImgReverse(app.imgtemp,'vertical'));title('垂直颠倒图像');

end

结果如下：



11.图像平移：



代码：

if app.isinputimgflag == 1

%判断输入参数是否为空

if isempty(app.ImgShiftParams.Value) == 1

params = [0,0];

else

params = app.ImgTransParamsParse(app.ImgShiftParams.Value,1);

end

%平移图像并作图

if iscell(app.img) == 1

%划分区域显示图像

app.img{1} = app.imgtemp;

app.img{1} = app.ImgShift(app.img{1},params);

switch app.ImgDispModDropDown.Value

case "自动"

axes1 = subplot(1,2,1,'Parent',app.ImageDisplayAreaPanel);

imshow(app.imgtemp,'Parent',axes1,'InitialMagnification','fit'); %显示平移前的图像

title('原图',"Parent",axes1,"FontName","楷体","FontSize",14,"Color",[1,1,1],"FontWeight","bold");

axes2 = subplot(1,2,2,'Parent',app.ImageDisplayAreaPanel);

imshow(app.img{1},'Parent',axes2,'InitialMagnification','fit'); %显示平移后的图像

title('平移后的图像',"Parent",axes2,"FontName","楷体","FontSize",14,"Color",[1,1,1],"FontWeight","bold");

case "原图"

axes1 = subplot(1,1,1,'Parent',app.ImageDisplayAreaPanel);

imshow(app.imgtemp,'Parent',axes1,'InitialMagnification','fit'); %显示平移前的图像

title('原图',"Parent",axes1,"FontName","楷体","FontSize",14,"Color",[1,1,1],"FontWeight","bold");

otherwise

axes1 = subplot(1,1,1,'Parent',app.ImageDisplayAreaPanel);

imshow(app.img{1},'Parent',axes1,'InitialMagnification','fit'); %显示平移前的图像

title('平移后的图像',"Parent",axes1,"FontName","楷体","FontSize",14,"Color",[1,1,1],"FontWeight","bold");

end

%是否调用外部窗口

if app.IsChooseFigureDropDown == "是"

switch app.ImgDispModDropDown.Value

case "自动"

figure;

subplot(1,2,1);imshow(app.imgtemp);title('原图');

subplot(1,2,2);imshow(app.img{1});title('平移后的图像');

case "原图"

figure;

imshow(app.imgtemp);title('原图');

otherwise

figure;

imshow(app.img{1});title('平移后的图像');

end

end

else

app.img = app.imgtemp;

app.img = app.ImgShift(app.img,params);

%根据用户需求采用不同的方式显示图像

switch app.ImgDispModDropDown.Value

case "自动"

axes1 = subplot(1,2,1,'Parent',app.ImageDisplayAreaPanel);

imshow(app.imgtemp,'Parent',axes1,'InitialMagnification','fit'); %显示平移前的图像

title('原图',"Parent",axes1,"FontName","楷体","FontSize",14,"Color",[1,1,1],"FontWeight","bold");

axes2 = subplot(1,2,2,'Parent',app.ImageDisplayAreaPanel);

imshow(app.img,'Parent',axes2,'InitialMagnification','fit'); %显示平移后的图像

title('平移后的图像',"Parent",axes2,"FontName","楷体","FontSize",14,"Color",[1,1,1],"FontWeight","bold");

case "原图"

axes1 = subplot(1,1,1,'Parent',app.ImageDisplayAreaPanel);

imshow(app.imgtemp,'Parent',axes1,'InitialMagnification','fit'); %显示平移前的图像

title('原图',"Parent",axes1,"FontName","楷体","FontSize",14,"Color",[1,1,1],"FontWeight","bold");

otherwise

axes1 = subplot(1,1,1,'Parent',app.ImageDisplayAreaPanel);

imshow(app.img,'Parent',axes1,'InitialMagnification','fit'); %显示平移后的图像

title('平移后的图像',"Parent",axes1,"FontName","楷体","FontSize",14,"Color",[1,1,1],"FontWeight","bold");

end

%是否选择外部窗口

if app.IsChooseFigureDropDown.Value == "是"

switch app.ImgDispModDropDown.Value

case "自动"

figure;

subplot(1,2,1);imshow(app.imgtemp);title('原图');

subplot(1,2,2);imshow(app.img);title('平移后的图像');

case "原图"

figure;

imshow(app.imgtemp);title('原图');

otherwise

figure;

imshow(app.img);title('平移后的图像');

end

end

end

end

结果：



12.图像缩放



代码：

if app.isinputimgflag == 1

%判断输入参数是否为空

if isempty(app.ImgZoomParams.Value) == 1

params = 1;

else

params = app.ImgTransParamsParse(app.ImgZoomParams.Value,2);

end

%缩放图像并作图,多个输入

if iscell(app.img) == 1

%划分区域显示图像

app.img{1} = app.imgtemp;

app.img{1} = app.ImgZoom(app.img{1},params);

switch app.ImgDispModDropDown.Value

case "自动"

axes1 = subplot(1,2,1,'Parent',app.ImageDisplayAreaPanel);

imshow(app.imgtemp,'Parent',axes1,'InitialMagnification','fit'); %显示缩放前的图像

[h,w,~] = size(app.imgtemp);

title(strcat('图像高度:',num2str(h),32,'图像宽度:',num2str(w)),"Parent",axes1,"FontName","楷体","FontSize",16,"Color",[1,1,1],"FontWeight","bold");

axes2 = subplot(1,2,2,'Parent',app.ImageDisplayAreaPanel);

imshow(app.img{1},'Parent',axes2,'InitialMagnification','fit'); %显示缩放后的图像

[h,w,~] = size(app.img{1});

title(strcat('图像高度:',num2str(h),32,'图像宽度:',num2str(w)),"Parent",axes2,"FontName","楷体","FontSize",16,"Color",[1,1,1],"FontWeight","bold");

case "原图"

axes1 = subplot(1,1,1,'Parent',app.ImageDisplayAreaPanel);

imshow(app.imgtemp,'Parent',axes1,'InitialMagnification','fit'); %显示缩放前的图像

[h,w,~] = size(app.imgtemp);

title(strcat('图像高度:',num2str(h),32,'图像宽度:',num2str(w)),"Parent",axes1,"FontName","楷体","FontSize",16,"Color",[1,1,1],"FontWeight","bold");

otherwise

axes1 = subplot(1,1,1,'Parent',app.ImageDisplayAreaPanel);

imshow(app.img{1},'Parent',axes1,'InitialMagnification','fit'); %显示缩放后的图像

[h,w,~] = size(app.img{1});

title(strcat('图像高度:',num2str(h),32,'图像宽度:',num2str(w)),"Parent",axes1,"FontName","楷体","FontSize",16,"Color",[1,1,1],"FontWeight","bold");

end

%是否选择外部窗口

if app.IsChooseFigureDropDown.Value == "是"

switch app.ImgDispModDropDown.Value

case "自动"

figure;

subplot(1,2,1);imshow(app.imgtemp);

[h,w,~] = size(app.imgtemp);

title(strcat('图像高度:',num2str(h),32,'图像宽度:',num2str(w)));

subplot(1,2,2);imshow(app.img{1});

[h,w,~] = size(app.img{1});

title(strcat('图像高度:',num2str(h),32,'图像宽度:',num2str(w)));

case "原图"

figure;

imshow(app.imgtemp);title('原图');

[h,w,~] = size(app.imgtemp);

title(strcat('图像高度:',num2str(h),32,'图像宽度:',num2str(w)));

otherwise

figure;

imshow(app.img{1});

[h,w,~] = size(app.img{1});

title(strcat('图像高度:',num2str(h),32,'图像宽度:',num2str(w)));

end

end

%单个输入

else

app.img = app.imgtemp;

app.img = app.ImgZoom(app.img,params);

switch app.ImgDispModDropDown.Value

case "自动"

axes1 = subplot(1,2,1,'Parent',app.ImageDisplayAreaPanel);

imshow(app.imgtemp,'Parent',axes1,'InitialMagnification','fit'); %显示缩放前的图像

[h,w,~] = size(app.imgtemp);

title(strcat('图像高度:',num2str(h),32,'图像宽度:',num2str(w)),"Parent",axes1,"FontName","楷体","FontSize",14,"Color",[1,1,1],"FontWeight","bold"); %显示图像宽度和高度

axes2 = subplot(1,2,2,'Parent',app.ImageDisplayAreaPanel);

imshow(app.img,'Parent',axes2,'InitialMagnification','fit'); %显示缩放后的图像

[h,w,~] = size(app.img);

title(strcat('图像高度:',num2str(h),32,'图像宽度:',num2str(w)),"Parent",axes2,"FontName","楷体","FontSize",14,"Color",[1,1,1],"FontWeight","bold");

case "原图"

axes = subplot(1,1,1,'Parent',app.ImageDisplayAreaPanel);

imshow(app.imgtemp,'Parent',axes,'InitialMagnification','fit'); %显示缩放前的图像

[h,w,~] = size(app.imgtemp);

title(strcat('图像高度:',num2str(h),32,'图像宽度:',num2str(w)),"Parent",axes,"FontName","楷体","FontSize",16,"Color",[1,1,1],"FontWeight","bold");

otherwise

axes = subplot(1,1,1,'Parent',app.ImageDisplayAreaPanel);

imshow(app.img,'Parent',axes,'InitialMagnification','fit'); %显示缩放后的图像

[h,w,~] = size(app.img);

title(strcat('图像高度:',num2str(h),32,'图像宽度:',num2str(w)),"Parent",axes,"FontName","楷体","FontSize",16,"Color",[1,1,1],"FontWeight","bold");

end

%是否选择外部窗口

if app.IsChooseFigureDropDown.Value == "是"

switch app.ImgDispModDropDown.Value

case "自动"

figure;

subplot(1,2,1);imshow(app.imgtemp);

[h,w,~] = size(app.imgtemp);

title(strcat('图像高度:',num2str(h),32,'图像宽度:',num2str(w)));

subplot(1,2,2);imshow(app.img);

[h,w,~] = size(app.img);

title(strcat('图像高度:',num2str(h),32,'图像宽度:',num2str(w)));

case "原图"

figure;

imshow(app.imgtemp);title('原图');

[h,w,~] = size(app.imgtemp);

title(strcat('图像高度:',num2str(h),32,'图像宽度:',num2str(w)));

otherwise

figure;

imshow(app.img);

[h,w,~] = size(app.img);

title(strcat('图像高度:',num2str(h),32,'图像宽度:',num2str(w)));

end

end

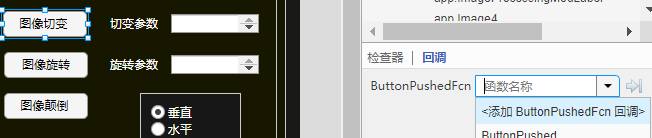
end

end

结果：



13.图像切变：



代码：

if app.isinputimgflag == 1

%判断输入参数是否为空

if isempty(app.ImgShearParams.Value) == 1

params = [0,0];

else

params = app.ImgTransParamsParse(app.ImgShearParams.Value,1);

end

%切变图像并作图,多个输入

if iscell(app.img) == 1

%划分区域显示图像

app.img{1} = app.imgtemp;

app.img{1} = app.ImgShear(app.img{1},params);

switch app.ImgDispModDropDown.Value

case "自动"

axes1 = subplot(1,2,1,'Parent',app.ImageDisplayAreaPanel);

imshow(app.imgtemp,'Parent',axes1,'InitialMagnification','fit'); %显示切变前的图像

title('原图',"Parent",axes1,"FontName","楷体","FontSize",16,"Color",[1,1,1],"FontWeight","bold");

axes2 = subplot(1,2,2,'Parent',app.ImageDisplayAreaPanel);

imshow(app.img{1},'Parent',axes2,'InitialMagnification','fit'); %显示切变后的图像

title('切变后的图像',"Parent",axes2,"FontName","楷体","FontSize",16,"Color",[1,1,1],"FontWeight","bold");

case "原图"

axes = subplot(1,1,1,'Parent',app.ImageDisplayAreaPanel);

imshow(app.imgtemp,'Parent',axes,'InitialMagnification','fit'); %显示切变前的图像

title('原图',"Parent",axes,"FontName","楷体","FontSize",16,"Color",[1,1,1],"FontWeight","bold");

otherwise

axes = subplot(1,1,1,'Parent',app.ImageDisplayAreaPanel);

imshow(app.img{1},'Parent',axes,'InitialMagnification','fit'); %显示切变后的图像

title('切变后的图像',"Parent",axes,"FontName","楷体","FontSize",16,"Color",[1,1,1],"FontWeight","bold");

end

%是否选择外部窗口

if app.IsChooseFigureDropDown.Value == "是"

switch app.ImgDispModDropDown.Value

case "自动"

figure;

subplot(1,2,1);imshow(app.imgtemp);

title('原图');

subplot(1,2,2);imshow(app.img{1});t

title('切变后的图像');

case "原图"

figure;

imshow(app.imgtemp);title('原图');

title('原图');

otherwise

figure;

imshow(app.img{1});

title('切变后的图像');

end

end

%单个输入

else

app.img = app.imgtemp;

app.img = app.ImgShear(app.img,params);

switch app.ImgDispModDropDown.Value

case "自动"

axes1 = subplot(1,2,1,'Parent',app.ImageDisplayAreaPanel);

imshow(app.imgtemp,'Parent',axes1,'InitialMagnification','fit'); %显示切变前的图像

title('原图',"Parent",axes1,"FontName","楷体","FontSize",16,"Color",[1,1,1],"FontWeight","bold");

axes2 = subplot(1,2,2,'Parent',app.ImageDisplayAreaPanel);

imshow(app.img,'Parent',axes2,'InitialMagnification','fit'); %显示切变后的图像

title('切变后的图像',"Parent",axes2,"FontName","楷体","FontSize",16,"Color",[1,1,1],"FontWeight","bold");

case "原图"

axes = subplot(1,1,1,'Parent',app.ImageDisplayAreaPanel);

imshow(app.imgtemp,'Parent',axes,'InitialMagnification','fit'); %显示切变前的图像

title('原图',"Parent",axes,"FontName","楷体","FontSize",16,"Color",[1,1,1],"FontWeight","bold");

otherwise

axes = subplot(1,1,1,'Parent',app.ImageDisplayAreaPanel);

imshow(app.img,'Parent',axes,'InitialMagnification','fit'); %显示切变后的图像

title('切变后的图像',"Parent",axes,"FontName","楷体","FontSize",16,"Color",[1,1,1],"FontWeight","bold");

end

%是否选择外部窗口

if app.IsChooseFigureDropDown.Value == "是"

switch app.ImgDispModDropDown.Value

case "自动"

figure;

subplot(1,2,1);imshow(app.imgtemp);

title('原图');

subplot(1,2,2);imshow(app.img);

title('切变后的图像');

case "原图"

figure;

imshow(app.imgtemp);title('原图');

title('原图');

otherwise

figure;

imshow(app.img);

title('切变后的图像');

end

end

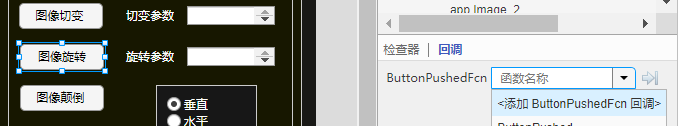
end

end

结果：



14.图像旋转：



代码：

if app.isinputimgflag == 1

%判断输入参数是否为空

if isempty(app.ImgRotateParams.Value) == 1

params = 0;

else

params = app.ImgTransParamsParse(app.ImgRotateParams.Value,2);

end

%缩放图像并作图,多个输入

if iscell(app.img) == 1

%划分区域显示图像

app.img{1} = app.imgtemp;

app.img{1} = app.ImgRotate(app.img{1},params);

switch app.ImgDispModDropDown.Value

case "自动"

axes1 = subplot(1,2,1,'Parent',app.ImageDisplayAreaPanel);

imshow(app.imgtemp,'Parent',axes1,'InitialMagnification','fit'); %显示旋转前的图像

title('原图',"Parent",axes1,"FontName","楷体","FontSize",16,"Color",[1,1,1],"FontWeight","bold");

axes2 = subplot(1,2,2,'Parent',app.ImageDisplayAreaPanel);

imshow(app.img{1},'Parent',axes2,'InitialMagnification','fit'); %显示旋转后的图像

title('旋转后的图像',"Parent",axes2,"FontName","楷体","FontSize",16,"Color",[1,1,1],"FontWeight","bold");

case "原图"

axes = subplot(1,1,1,'Parent',app.ImageDisplayAreaPanel);

imshow(app.imgtemp,'Parent',axes,'InitialMagnification','fit'); %显示旋转前的图像

title('原图',"Parent",axes,"FontName","楷体","FontSize",16,"Color",[1,1,1],"FontWeight","bold");

otherwise

axes = subplot(1,1,1,'Parent',app.ImageDisplayAreaPanel);

imshow(app.img{1},'Parent',axes,'InitialMagnification','fit'); %显示旋转后的图像

title('旋转后的图像',"Parent",axes,"FontName","楷体","FontSize",16,"Color",[1,1,1],"FontWeight","bold");

end

%是否选择外部窗口

if app.IsChooseFigureDropDown.Value == "是"

switch app.ImgDispModDropDown.Value

case "自动"

figure;

subplot(1,2,1);imshow(app.imgtemp);

title('原图');

subplot(1,2,2);imshow(app.img{1});

title('旋转后的图像');

case "原图"

figure;

imshow(app.imgtemp);title('原图');

title('原图');

otherwise

figure;

imshow(app.img{1});

title('旋转后的图像');

end

end

%单个输入

else

app.img = app.imgtemp;

app.img = app.ImgRotate(app.img,params);

switch app.ImgDispModDropDown.Value

case "自动"

axes1 = subplot(1,2,1,'Parent',app.ImageDisplayAreaPanel);

imshow(app.imgtemp,'Parent',axes1,'InitialMagnification','fit'); %显示旋转前的图像

title('原图',"Parent",axes1,"FontName","楷体","FontSize",16,"Color",[1,1,1],"FontWeight","bold");

axes2 = subplot(1,2,2,'Parent',app.ImageDisplayAreaPanel);

imshow(app.img,'Parent',axes2,'InitialMagnification','fit'); %显示旋转后的图像

title('旋转后的图像',"Parent",axes2,"FontName","楷体","FontSize",16,"Color",[1,1,1],"FontWeight","bold");

case "原图"

axes = subplot(1,1,1,'Parent',app.ImageDisplayAreaPanel);

imshow(app.imgtemp,'Parent',axes,'InitialMagnification','fit'); %显示旋转前的图像

title('原图',"Parent",axes,"FontName","楷体","FontSize",16,"Color",[1,1,1],"FontWeight","bold");

otherwise

axes = subplot(1,1,1,'Parent',app.ImageDisplayAreaPanel);

imshow(app.img,'Parent',axes,'InitialMagnification','fit'); %显示旋转后的图像

title('旋转后的图像',"Parent",axes,"FontName","楷体","FontSize",16,"Color",[1,1,1],"FontWeight","bold");

end

%是否选择外部窗口

if app.IsChooseFigureDropDown.Value == "是"

switch app.ImgDispModDropDown.Value

case "自动"

figure;

subplot(1,2,1);imshow(app.imgtemp);

title('原图');

subplot(1,2,2);imshow(app.img);

title('旋转后的图像');

case "原图"

figure;

imshow(app.imgtemp);title('原图');

title('原图');

otherwise

figure;

imshow(app.img);

title('旋转后的图像');

end

end

end

end

结果：



15.图像颠倒：



代码：

if app.isinputimgflag == 1

if app.ImgReverseModeButton.SelectedObject.Text == "水平"

params = 'level';

else

params = 'vertical';

end

%缩放图像并作图,多个输入

if iscell(app.img) == 1

%划分区域显示图像

app.img{1} = app.imgtemp;

app.img{1} = app.ImgReverse(app.img{1},params);

switch app.ImgDispModDropDown.Value

case "自动"

axes1 = subplot(1,2,1,'Parent',app.ImageDisplayAreaPanel);

imshow(app.imgtemp,'Parent',axes1,'InitialMagnification','fit'); %显示灰度化图像

title('原图',"Parent",axes1,"FontName","楷体","FontSize",16,"Color",[1,1,1],"FontWeight","bold");

axes2 = subplot(1,2,2,'Parent',app.ImageDisplayAreaPanel);

imshow(app.img{1},'Parent',axes2,'InitialMagnification','fit'); %显示颠倒后的图像

title('颠倒后的图像',"Parent",axes2,"FontName","楷体","FontSize",16,"Color",[1,1,1],"FontWeight","bold");

case "原图"

axes = subplot(1,1,1,'Parent',app.ImageDisplayAreaPanel);

imshow(app.imgtemp,'Parent',axes,'InitialMagnification','fit'); %显示颠倒前的图像

title('原图',"Parent",axes,"FontName","楷体","FontSize",16,"Color",[1,1,1],"FontWeight","bold");

otherwise

axes = subplot(1,1,1,'Parent',app.ImageDisplayAreaPanel);

imshow(app.img{1},'Parent',axes,'InitialMagnification','fit'); %显示颠倒后的图像

title('颠倒后的图像',"Parent",axes,"FontName","楷体","FontSize",16,"Color",[1,1,1],"FontWeight","bold");

end

%是否选择外部窗口

if app.IsChooseFigureDropDown.Value == "是"

switch app.ImgDispModDropDown.Value

case "自动"

figure;

subplot(1,2,1);imshow(app.imgtemp);

title('原图');

subplot(1,2,2);imshow(app.img{1});

title('颠倒后的图像');

case "原图"

figure;

imshow(app.imgtemp);title('原图');

title('原图');

otherwise

figure;

imshow(app.img{1});

title('颠倒后的图像');

end

end

%单个输入

else

app.img = app.imgtemp;

app.img = app.ImgReverse(app.img,params);

switch app.ImgDispModDropDown.Value

case "自动"

axes1 = subplot(1,2,1,'Parent',app.ImageDisplayAreaPanel);

imshow(app.imgtemp,'Parent',axes1,'InitialMagnification','fit'); %显示颠倒前的图像

title('原图',"Parent",axes1,"FontName","楷体","FontSize",16,"Color",[1,1,1],"FontWeight","bold");

axes2 = subplot(1,2,2,'Parent',app.ImageDisplayAreaPanel);

imshow(app.img,'Parent',axes2,'InitialMagnification','fit'); %显示颠倒后的图像

title('灰度化图像',"Parent",axes2,"FontName","楷体","FontSize",16,"Color",[1,1,1],"FontWeight","bold");

case "原图"

axes = subplot(1,1,1,'Parent',app.ImageDisplayAreaPanel);

imshow(app.imgtemp,'Parent',axes,'InitialMagnification','fit'); %显示颠倒前的图像

title('原图',"Parent",axes,"FontName","楷体","FontSize",16,"Color",[1,1,1],"FontWeight","bold");

otherwise

axes = subplot(1,1,1,'Parent',app.ImageDisplayAreaPanel);

imshow(app.img,'Parent',axes,'InitialMagnification','fit'); %显示颠倒后的图像

title('灰度化图像',"Parent",axes,"FontName","楷体","FontSize",16,"Color",[1,1,1],"FontWeight","bold");

end

%是否选择外部窗口

if app.IsChooseFigureDropDown.Value == "是"

switch app.ImgDispModDropDown.Value

case "自动"

figure;

subplot(1,2,1);imshow(app.imgtemp);

title('原图');

subplot(1,2,2);imshow(app.img);

title('灰度化图像');

case "原图"

figure;

imshow(app.imgtemp);title('原图');

title('原图');

otherwise

figure;

imshow(app.img);

title('灰度化图像');

end

end

end

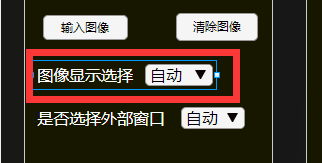
end

结果：



到这里，当你运行软件时会出现很多错误，这是因为控件的名字跟代码不匹配，做如下更改即可：

1.图像显示选择：

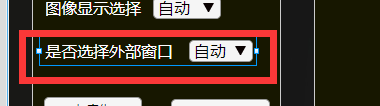


将该控件名称改为app.ImgDispModDropDown.



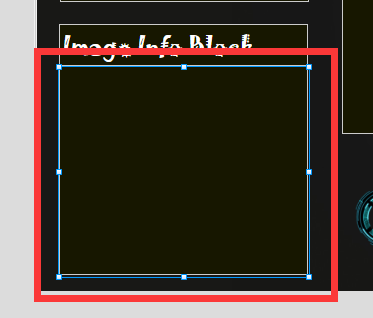
在右边红色框处的部分修改名称，下同。

2.是否选择外部窗口：



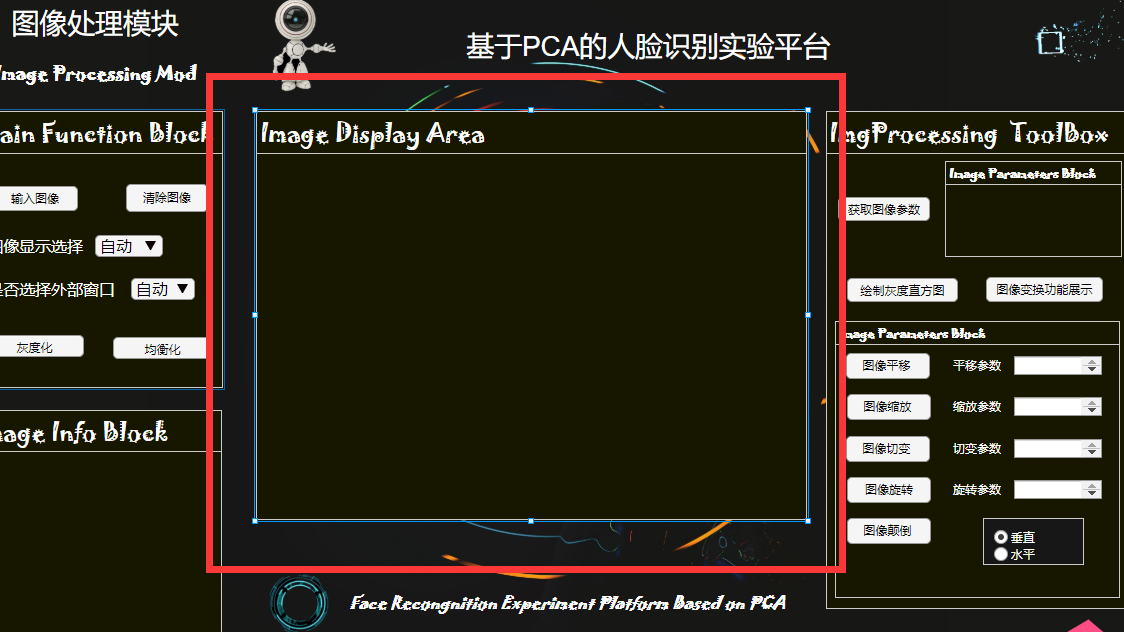
名称改为app.IsChooseFigureDropDown

3.图像信息栏：



这个控件一定要选中内部的标签Label，将标签的名称改为app.ImgInfo，不要更改面板Panel的名称！

4.图像显示区



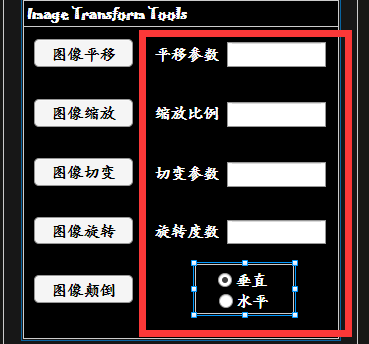
改名为app.ImageDisplayAreaPanel。

5.图像参数栏：



改名为app.ImgParams（注意是改里面的标签Label）。

6.输入参数文本框：



依次改名为:

app.ImgShiftParams, app.ImgZoomParams, app.ImgShearParams

app.ImgRotateParams, app.ImgReverseModeButton,

最后一个app.ImgReverseModeButton中，有两个选项也要改名字，“垂直”选项改名为app.vertical，“水平”选项改名为app.level

到此，全部更改工作完成，可以正常运行程序实现功能。