# GUI综合实验报告

## 实验目的

1. 了解Matlab制作GUI的步骤流程；
2. 为课内实验中的前5个实验制作一个Matlab的GUI，可视化操作各个实验的内容和显示结果。

## 实验原理

利用Matlab的GUI设计模块，将课内实验中的前5个实验聚合成一个可视化操作平台。前5个实验的GUI实现主要通过Matlab的GUI模块中的现成地按钮、显示界面、以及其他模块完成。当按键按下后，GUI调用回调函数，在回调函数中实现我们的所需要的操作。对于实验中可能用到的参数等，我们在文本框或者单选框或者列表中进行选择，并利用GUI模块的自带函数获得这些参数。利用这些基本操作，我们可以实现Matlab的GUI设计。

## 实验内容

本次GUI设计中，我们需要将课内实验中的前5个实验集成设计为一个GUI可视化界面，因此在具体介绍我的GUI的设计时，我将分成6个部分进行讲解，首先是GUI设计的大体思路，其次分别介绍5个实验具体如何集成为一个GUI可视化界面。

**3.1 可视化界面的大体设计**

因为要将5个实验集成在一个可视化界面上，所以我的设计思路为每一个实验设计一个模块来放置，达到清楚明了的效果。通过采用按钮组控件，我们就可以实现将每一个实验分开的效果，每一个实验所用到的控件都将放入对应的按钮组中。对于原始图片和显示图片的我们采用5个实验共用一组空间的方式。



图 1 GUI总体界面

GUI总体界面如图1所示，将界面划分成若干部分，每一个部分都对应了5个实验的其中一个。左上角则是原始图像和图像变换效果的展示，选择图片的按键将打开一个选择图片的界面供我们选择，而右侧清除按键则用来清除显示的图片。

**3.2 实验一的可视化界面设计**

实验一主要是一些对图像文件的操作和图像类型的转换，而图像文件的操作并不适合采用GUI可视化界面进行展示。因此对于实验一，我们主要在可视化界面中实现对图像类型的转换。

当我们选择实验一框中不同的按键时，我们将使原图转换成我们所选择的图像类型。这里我们主要在按键的回调函数中实现图像类型转换的操作。

**3.2.1 效果展示**

****

图 2 选择灰度图像时GUI显示效果



图 3 选择二值图像时GUI显示效果



图 4 选择索引图像时GUI显示效果

**3.2.2 源代码**

**%%灰度图像按键**

function gray\_figure\_Callback(hObject, eventdata, handles)

% hObject handle to binary\_figure (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

%ÏÔÊ¾»Ò¶ÈÍ¼Ïñ

global filename

global pathname

I=imread(fullfile(pathname,filename));

I=rgb2gray(I);

axes(handles.Fig2);

imshow(I);title('灰度图像');

**%%二值图像按键**

function binary\_figure\_Callback(hObject, eventdata, handles)

% hObject handle to binary\_figure (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

%ÏÔÊ¾¶þÖµÍ¼Ïñ

global filename

global pathname

I=imread(fullfile(pathname,filename));

I=im2bw(I,0.5);

axes(handles.Fig2);

imshow(I);title('二值图像');

**%%索引图像按键**

function ind\_figure\_Callback(hObject, eventdata, handles)

% hObject handle to ind\_figure (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

%ÏÔÊ¾Ë÷ÒýÍ¼Ïñ

global filename

global pathname

I=imread(fullfile(pathname,filename));

[I,cmap]=rgb2ind(I,64);

axes(handles.Fig2);

imshow(I,cmap);title('索引图像');

**3.2 实验二的可视化界面设计**

实验二主要实现图像的平移、镜像、缩放以及图像旋转等图像变换操作。与实验一不同的是在这实验二中有很多需要我们提前给定的参数，因此在设计GUI时我们不仅用到了按键以及按键的回调函数，也用到了文本框，在文本框中输入参数并传递给执行函数中。

**3.2.1 效果展示**

****

图 5 图像平移。在文本框中输入预设的平移量参数，然后点击平移按键，在右侧框中显示平移后的图片



图 6 图像镜像。点击垂直镜像或者水平镜像，在右侧框中显示镜像后的结果。此处我们展示了垂直镜像作为例子。



图 7 图像缩放。在文本框中输入预设的缩放倍数，点击图像缩放按键，在右侧框中显示缩放后的结果。为了效果更加明显，这里采用了局部缩放的方式，并以2倍为例。



图 8 图像旋转。在文本框中输入预设的旋转角度，点击图像旋转按键，在右侧框中显示旋转后的结果。这里我们以旋转60度为例。

**3.2.2 源代码**

**%%图像平移按键**

function figure\_move\_Callback(hObject, eventdata, handles)

% hObject handle to figure\_move (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

% Æ½ÒÆÍ¼Ïñ

global filename

global pathname

I = imread(fullfile(pathname,filename));

I = rgb2gray(I);

X=get(handles.move\_x,'string');

Y=get(handles.move\_y,'string');

[M,N]=size(I);

I\_pingyi = zeros(M,N);

x\_move = str2num(X);

y\_move = str2num(Y);

move = [1,0,x\_move;0,1,y\_move;0,0,1];

for i = 1:M

for j = 1:N

temp = [i;j;1];

temp = move\*temp;

temp\_x = temp(1,1);

temp\_y = temp(2,1);

if temp\_x <= M && temp\_y <= N

I\_pingyi(temp\_x,temp\_y) = I(i,j);

end

end

end

I\_pingyi=uint8(I\_pingyi);

axes(handles.Fig2);

imshow(I\_pingyi);title('图像平移')

**%%图像镜像（以水平镜像为例）**

function horizontal\_mirror\_Callback(hObject, eventdata, handles)

% hObject handle to horizontal\_mirror (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

global filename

global pathname

I = imread(fullfile(pathname,filename));

I = rgb2gray(I);

[M,N]=size(I);

I\_shuiping = zeros(M,N);

shuiping = [1,0,0;0,-1,0;0,0,1];

for i = 1:M

for j = 1:N

item = [i;j;1];

item\_shuiping = shuiping\*item;

item\_x\_shuiping = item\_shuiping(1,1);

item\_y\_shuiping = N + item\_shuiping(2,1) + 1;

I\_shuiping(item\_x\_shuiping,item\_y\_shuiping) = I(i,j);

end

end

I\_shuiping=uint8(I\_shuiping);

axes(handles.Fig2);

imshow(I\_shuiping);title('Í¼ÏñË®Æ½¾µÏñ');

**%%图像缩放**

function shrink\_enlarge\_Callback(hObject, eventdata, handles)

% hObject handle to shrink\_enlarge (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

% Í¼ÏñËõ·Å

global filename

global pathname

I = imread(fullfile(pathname,filename));

I = rgb2gray(I);

[M,N]=size(I);

% µÃµ½ÊäÈë±¶Êý

Rate=get(handles.shrink\_enlarge\_rate,'string');

Rate=str2num(Rate);

I\_result=zeros(M,N);

% Èç¹ûÊäÈë´íÎó

if Rate<=0

axes(handles.Fig2);

imshow(I);title('ÇëÊäÈëÕýÈ·Ëõ·Å±¶Êý');

% ÎÞ·Å´óËõÐ¡

elseif Rate==1

axes(handles.Fig2);

imshow(I);title('·Å´ó±¶ÊýÎª1£¬ÎªÔ­Í¼Ïñ');

% Í¼Æ¬·Å´ó£¬ Í¼Æ¬Ö»ÏÔÊ¾×óÉÏ½Ç²¿·Ö

elseif Rate>1

I\_fangda = zeros(floor(M\*Rate),floor(N\*Rate));

[A,B] = size(I\_fangda);

for i = 1:A

for j = 1:B

x = floor(i/Rate);

y = floor(j/Rate);

if x == 0

x = x+1;

end

if y == 0

y = y+1;

end

if x == M

x = x-1;

end

if y == N

y = y-1;

end

u = i/Rate - x;

v = j/Rate - y;

I\_fangda(i,j) = u\*v\*I(x,y)+(1-u)\*v\*I(x+1,y)+u\*(1-v)\*I(x,y+1)+(1-u)\*(1-v)\*I(x+1,y+1);

end

end

I\_fangda=double(I\_fangda);

I\_result=I\_result+I\_fangda(1:M,1:N);

I\_result=uint8(I\_result);

axes(handles.Fig2);

imshow(I\_result);title('Í¼Ïñ·Å´ó');

% Í¼Æ¬ËõÐ¡£¬ÎÞÏÔÊ¾ÔòÒÔºÚÉ«Ìî³ä

else

I\_suoxiao = zeros(floor(M\*Rate),floor(N\*Rate));

[A,B]=size(I\_suoxiao);

suoxiao = [Rate,0,0;0,Rate,0;0,0,1];

for i = 1:M

for j = 1:N

temp = [i;j;1];

temp\_suoxiao = suoxiao\*temp;

temp\_x\_suoxiao = uint8(temp\_suoxiao(1,1));

temp\_y\_suoxiao = uint8(temp\_suoxiao(2,1));

if temp\_x\_suoxiao<=0

temp\_x\_suoxiao=1;

elseif temp\_x\_suoxiao>A

temp\_x\_suoxiao=A;

end

if temp\_y\_suoxiao<=0

temp\_y\_suoxiao=1;

elseif temp\_y\_suoxiao>B

temp\_y\_suoxiao=B;

end

I\_suoxiao(temp\_x\_suoxiao,temp\_y\_suoxiao) = I(i,j);

end

end

I\_suoxiao=double(I\_suoxiao);

I\_result(1:A,1:B)=I\_result(1:A,1:B)+I\_suoxiao;

I\_result=uint8(I\_result);

axes(handles.Fig2);

imshow(I\_result);title('Í¼ÏñËõÐ¡');

end

**%%图像旋转**

function rotate\_Callback(hObject, eventdata, handles)

% hObject handle to rotate (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

% Í¼ÏñµÄÐý×ª

global filename

global pathname

I = imread(fullfile(pathname,filename));

angle=get(handles.rotate\_angle,'string');

angle=str2num(angle);

[M,N]=size(I);

I\_xuanzhuan=zeros(M,N);

I\_xuanzhuan = imrotate(I,angle,'bilinear');

I\_xuanzhuan = uint8(I\_xuanzhuan);

axes(handles.Fig2);

imshow(I\_xuanzhuan);title('Í¼ÏñÐý×ª');

**3.3 实验三的可视化界面设计**

实验三主要是对图像的增强操作。这些增强操作分别有对数增强、指数增强、线性增强和直方图均衡化，这些操作并没有什么参数，因此对于实验三，主要采用按键来实现，在按键的回调函数中实现这四种图像增强操作。

**3.3.1 效果展示**

****

图 9 对数增强。点击GUI中对数增强按键，在右侧框中显示对数增强处理后的图像



图 10 指数增强。点击GUI中指数增强按键，右侧显示指数增强后的图像



图 11 线性增强。在GUI中点击线性增强按键，在右侧框图中显示线性增强后的图像



图 12 直方图均衡化。在GUI中点击直方图均衡化按键，在右侧框图中显示直方图均衡化后的图像

**3.3.2 源代码**

**%%对数增强**

function log\_enhance\_Callback(hObject, eventdata, handles)

% hObject handle to log\_enhance (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

% ¶ÔÊýÔöÇ¿

global filename

global pathname

I = imread(fullfile(pathname,filename));

I = rgb2gray(I);

I = Gray\_change(I,'Log');

axes(handles.Fig2);

imshow(I);title('Í¼Ïñ¶ÔÊýÔöÇ¿');

**%%指数增强**

function exp\_enhance\_Callback(hObject, eventdata, handles)

% hObject handle to exp\_enhance (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

% ¶ÔÊýÔöÇ¿

global filename

global pathname

I = imread(fullfile(pathname,filename));

I = rgb2gray(I);

I = Gray\_change(I,'Index');

axes(handles.Fig2);

imshow(I);title('Í¼ÏñÖ¸ÊýÔöÇ¿');

**%%线性增强**

function linear\_enhance\_Callback(hObject, eventdata, handles)

% hObject handle to linear\_enhance (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

% ÏßÐÔÔöÇ¿

global filename

global pathname

I = imread(fullfile(pathname,filename));

I = rgb2gray(I);

I = Gray\_change(I,'Linear');

axes(handles.Fig2);

imshow(I);title('Í¼ÏñÏßÐÔÔöÇ¿');

**%%直方图均衡化回调函数**

function averager\_hist\_Callback(hObject, eventdata, handles)

% hObject handle to averager\_hist (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

% Ö±·½Í¼¾ùºâ»¯

global filename

global pathname

I = imread(fullfile(pathname,filename));

I = rgb2gray(I);

I = Gray\_change(I,'Histogram\_Equalization');

axes(handles.Fig2);

imshow(I);title('Í¼ÏñÖ±·½Í¼¾ùºâ»¯');

**%%以上增强调用的函数**

function[Figure]=Gray\_change(Figure,type)

%% ±¾º¯ÊýÓÃÓÚÊµÏÖ¶ÔÍ¼Æ¬µÄ»Ò¶ÈÖµ½øÐÐ´¦Àí

% Inpute:

% path: Í¼Æ¬´æ´¢Â·¾¶

% type: ËùÓÃ»Ò¶ÈÖµ±ä»¯ÀàÐÍ£¬±¾º¯ÊýÌá¹©ÁËÒ»ÏÂ¼¸¸öÀàÐÍ£º

% ¡®Linear¡¯:»Ò¶ÈÖµÏßÐÔ±ä»¯

% ¡®Log¡¯:»Ò¶ÈÖµ·ÇÏßÐÔ¶ÔÊý±ä»¯

% ¡®Index¡¯:»Ò¶ÈÖµ·ÇÏßÐÔÖ¸Êý±ä»¯

% ¡®Histogram\_Equalization¡¯:Ö±·½Í¼¾ùºâ»¯

%#################################################################################

% Í¼Æ¬µÄÔØÈë

% Figure=imread(path);

% Figure=rgb2gray(Figure);

% figure(1);

% subplot(1,2,1);imshow(Figure);title('Original Figure');

[M,N]=size(Figure);

Figure=double(Figure);

% ½øÐÐÏßÐÔ»Ò¶È±ä»»

if strcmp(type,'Linear')

for i=1:M

for j=1:N

% ÏßÐÔº¯Êý¿ÉÒÔ×Ô¼º¶¨Òå

if Figure(i,j)<=20

Figure(i,j)=Figure(i,j);

elseif Figure(i,j)<=150

Figure(i,j)=(200-20)/(150-20)\*(Figure(i,j)-20)+20;

else

Figure(i,j)=(250-200)/(300-150)\*(Figure(i,j)-150)+200;

end

end

end

% Êä³ö½á¹û

Figure=uint8(Figure);

% subplot(1,2,2);imshow(Figure);title('Linear Gray Change');

% ###################################################

% ·ÇÏßÐÔ¶ÔÊý»Ò¶È±ä»»

elseif strcmp(type,'Log')

% ¶ÔÍ¼ÏñµÄÖµ½øÐÐ»Ò¶È±ä»¯

Figure=41\*log(Figure+1);

% Êä³ö½á¹û

Figure=uint8(Figure);

% subplot(1,2,2);imshow(Figure);title('Log Gray Change');

% ###################################################

% ·ÇÏßÐÔÖ¸Êý»Ò¶È±ä»»

elseif strcmp(type,'Index')

% ¶ÔÍ¼ÏñµÄÖµ½øÐÐ»Ò¶È±ä»¯

% Figure=Figure.\*Figure/255;

Figure=(Figure.^0.9);

% Êä³ö½á¹û

Figure=uint8(Figure);

% subplot(1,2,2);imshow(Figure);title('Index Gray Change');

% ###################################################

% Ö±·½Í¼¾ùºâ»¯

elseif strcmp(type,'Histogram\_Equalization')

freq = reshape(Figure, [1,M\*N]);

tab = tabulate(freq);

tab(:, 3) = cumsum(tab(:, 3));

% ½«Ô­ÏñËØ¸ù¾ÝÆµÊý×ª»»

for i = 1 : M

for j = 1:N

k = find(tab(:,1)==Figure(i,j));

Figure(i,j) = fix(tab(k(1),3)/100.0 \* 255);

end

end

Figure = uint8(Figure);

% ###################################################

% Òì³£ÊäÈë

else

disp('Please Input the right type,this function only support the Linear\NonLinear\Histogram\_Equalization');

end

end

**3.4 实验四的可视化界面设计**

实验四的内容是实现图像滤波。这个实验中主要需要在图像中加入噪声并对图像作中值滤波，并利用sobel和prewitt算子对图像进行处理并实现理想低通滤波器和巴特沃斯低通滤波器。对于噪声的加入，一共有三种噪声，分别为高斯噪声、椒盐噪声、泊松噪声，利用列表控件可以实现对三种噪声的选取。

**3.4.1 效果展示**



图 13 图像中值滤波。以椒盐噪声的滤波为例，首先在列表框中选择椒盐噪声，然后点击添加噪声实现噪声的添加，再点击中值滤波，得到使用中值滤波得到的图像



图 14 图像锐化。以基于Sobel算子的图像锐化为例，点击GUI中的Sobel算子，右侧图像显示锐化后的图像



图 15 理想低通滤波器。首先我们在文本框中输入滤波的阈值，这里以5为例，然后点击理想低通滤波器按键，则显示理想低通滤波处理后的图像。

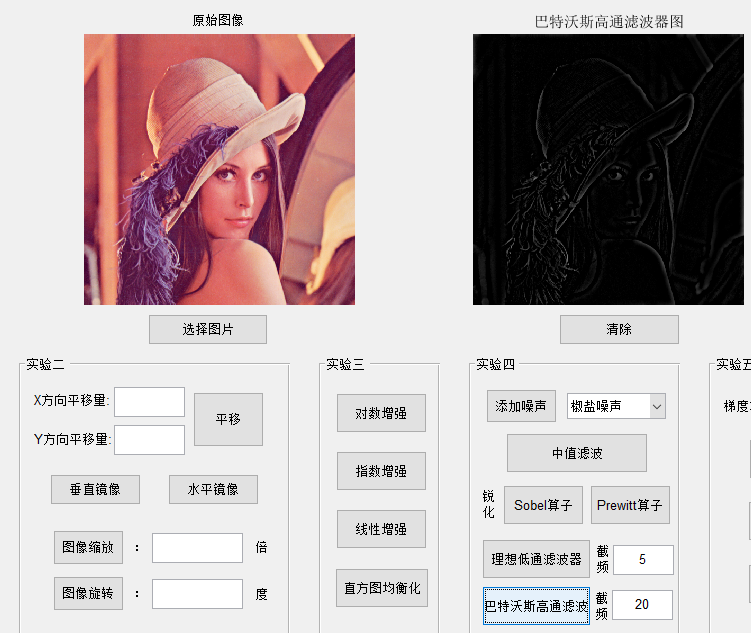


图 16 巴特沃斯高通滤波器。首先我们在文本框中输入滤波的阈值，这里以20为例，然后点击巴特沃斯高通滤波器按键，显示巴特沃斯高通滤波处理后的图像。

**3.4.2 源代码**

**%%添加噪声回调函数**

function add\_noise\_Callback(hObject, eventdata, handles)

% hObject handle to add\_noise (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

% Ìí¼ÓÈýÖÖÀàÐÍµÄÔëÉù

global filename

global pathname

I = imread(fullfile(pathname,filename));

noise\_type=get(handles.noise\_type,'Value');

if noise\_type==1

I=imnoise(I,'gaussian');

axes(handles.Fig2);

imshow(I);title('Ìí¼Ó¸ßË¹ÔëÉù');

elseif noise\_type==2

I=imnoise(I,'salt & pepper',0.02);

axes(handles.Fig2);

imshow(I);title('Ìí¼Ó½·ÑÎÔëÉù');

else

I=imnoise(I,'poisson');

axes(handles.Fig2);

imshow(I);title('Ìí¼Ó²´ËÉÔëÉù');

end

**%%中值滤波回调函数**

function medium\_filter\_Callback(hObject, eventdata, handles)

% hObject handle to medium\_filter (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

% ÖÐÖµÂË²¨

global filename

global pathname

I = imread(fullfile(pathname,filename));

[hight, width] = size(I);

noise\_type=get(handles.noise\_type,'Value');

if noise\_type==1

I=imnoise(I,'gaussian');

elseif noise\_type==2

I=imnoise(I,'salt & pepper',0.02);

else

I=imnoise(I,'poisson');

end

fsize = 3;

pad = (fsize - 1) / 2;

for i = 1+pad : hight-pad

for j = 1+pad : width-pad

filter = I(i-pad:i+pad, j-pad:j+pad);

I(i,j) = median(filter(:));

end

end

% I=Median\_filter(I,3);

axes(handles.Fig2);

imshow(I);title('ÖÐÖµÂË²¨');

**%%锐化操作回调函数（以Sobel算子为例）**

function sobel\_Callback(hObject, eventdata, handles)

% hObject handle to sobel (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

% sobelËã×ÓÈñ»¯

global filename

global pathname

img = imread(fullfile(pathname,filename));

img = rgb2gray(img);

k\_shape = [3, 3]; % set kernel size

padding = (k\_shape(1)-1) / 2; % padding size

h\_kernel = [1,2,1;0,0,0;-1,-2,-1]; % horizon sobel kernel

v\_kernel = [-1,0,1;-2,0,2;-1,0,1]; % vertical sobel kernel

[hight, width] = size(img);

norm\_img = zeros(hight + padding\*2, width + padding\*2);

% generate new image

norm\_img(1+padding:hight+padding, 1+padding:width+padding) = double(img);

new\_norm = zeros(hight + padding\*2, width + padding\*2);

for i = (1+padding) : (hight+padding)

for j = (1+padding) : (width+padding)

% execute grad\_norm filter

h\_norm = sum(sum(norm\_img(i-padding : i+padding, j-padding : j+padding) .\* h\_kernel));

v\_norm = sum(sum(norm\_img(i-padding : i+padding, j-padding : j+padding) .\* v\_kernel));

new\_norm(i, j) = abs(h\_norm) + abs(v\_norm);

end

end

new\_norm = uint8(new\_norm(1+padding:hight+padding, 1+padding:width+padding));

% I=Sharpen\_filter(I,'Sobel');

axes(handles.Fig2);

imshow(new\_norm);title('SoberËã×Ó');

**%%理想低通滤波器**

function low\_pass\_Callback(hObject, eventdata, handles)

% hObject handle to low\_pass (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

% µÍÍ¨ÂË²¨Æ÷

global filename

global pathname

I=imread(fullfile(pathname,filename));

data=get(handles.low\_pass\_data, 'String');

data=str2num(data);

I=Low\_pass\_filter(I,data);

axes(handles.Fig2);

imshow(I);title('µÍÍ¨ÂË²¨Æ÷Í¼');

**%%巴特沃斯高通滤波器**

function high\_pass\_Callback(hObject, eventdata, handles)

% hObject handle to high\_pass (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

% °ÍÌØÎÖË¹¸ßÍ¨ÂË²¨Æ÷

global filename

global pathname

I=imread(fullfile(pathname,filename));

data=get(handles.high\_pass\_data, 'String');

data=str2num(data);

I=Butterworth\_high\_pass\_filter(I,data);

axes(handles.Fig2);

imshow(I);title('°ÍÌØÎÖË¹¸ßÍ¨ÂË²¨Æ÷Í¼');

**3.5 实验五的可视化界面设计**

实验五的实验内容为图像分割。在这个实验中我们主要实现3个任务，利用三种算子实现边缘检测，对一张二值图像作边缘跟踪以及实现分水岭算法。其中要利用三种边缘检测算子，因此我们使用列表框来实现三种边缘检测的选择，然后再调用回调函数。

**3.5.1 效果展示**

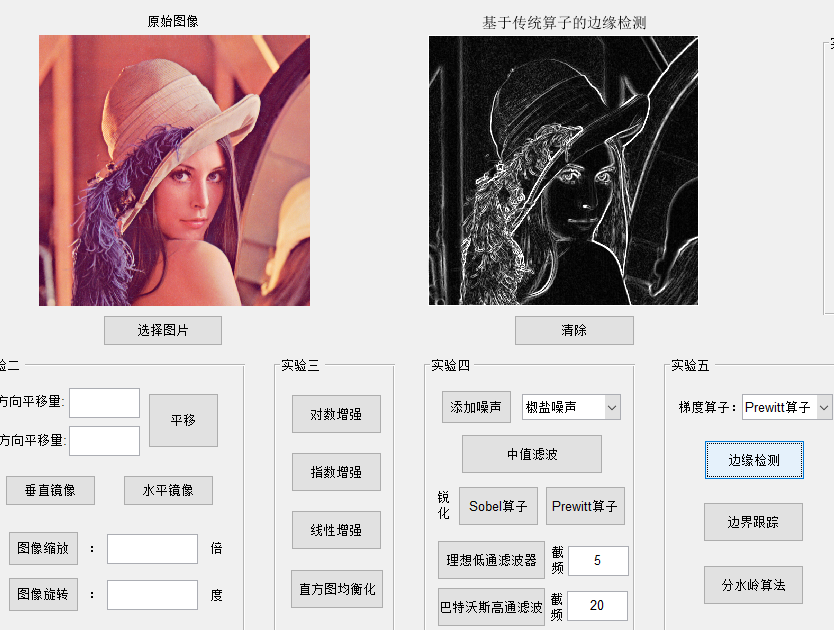


图 17 图像边缘检测。以基于Prewitt算子的图像边缘检测为例，首先我们在列表中选择Prewitt算子，然后点击边缘检测按键，得到边缘检测图像。

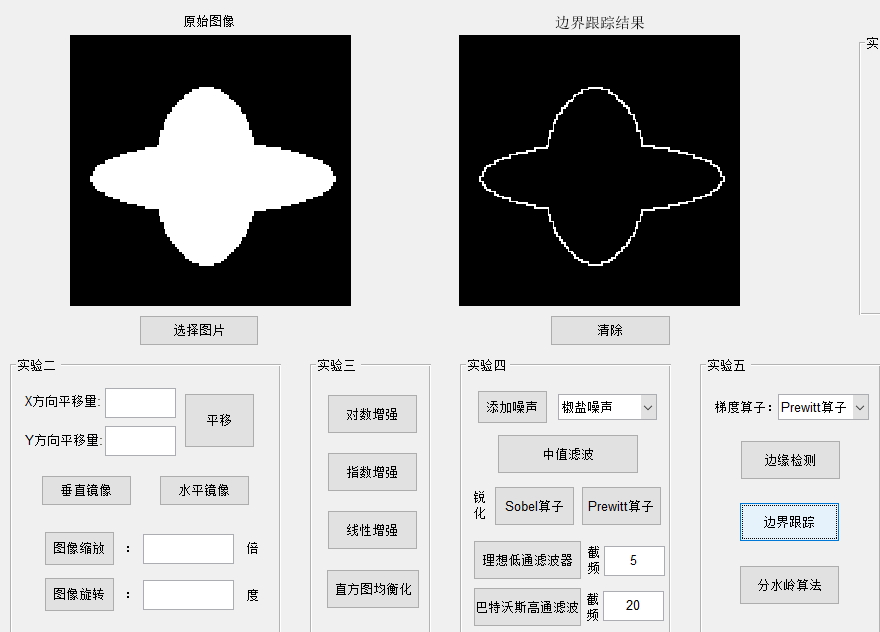


图 18 二值图像的边缘跟踪。首先我们在GUI中导入一张封闭形状的二值图像，然后点击边界跟踪按键，显示边界跟踪的处理效果。



图 19 分水岭算法实现。首先我们在GUI中导入一张合适的图像，然后点击分水岭算法按键，显示采用分水岭算法处理后的图像。

**3.5.2 源代码**

**%%边缘检测回调函数**

function boundary\_test\_Callback(hObject, eventdata, handles)

% hObject handle to boundary\_test (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

% »ùÓÚËã×ÓµÄ±ßÔµ¼ì²â

global filename

global pathname

I=imread(fullfile(pathname,filename));

[hight, width, ~] = size(I);

type\_operator=get(handles.operator\_type, 'Value');

if type\_operator==1

rob\_h = [-1,0;0,1];

rob\_v = [0,-1;1,0]; % Roberts template

rob\_img = double(I);

new = double(I);

for k = 1:3

for i = 1:hight-1

for j = 1:width-1

grad1 = sum(sum(rob\_img(i:i+1,j:j+1,k) .\* rob\_h));

grad2 = sum(sum(rob\_img(i:i+1,j:j+1,k) .\* rob\_v));

new(i,j,k) = abs(grad1) + abs(grad2);

end

end

end

new = uint8(new);

% I=Sharpen\_filter(I,'Robert');

elseif type\_operator==2

sob\_h = [1,2,1;0,0,0;-1,-2,-1];

sob\_v = [-1,0,1;-2,0,2;-1,0,1]; % Sober template

sob\_img = zeros(hight+2, width+2, 3);

sob\_img(2:hight+1, 2:width+1,:) = double(I);

new = zeros(hight+2, width+2, 3);

for k = 1:3

for i = 2:hight+1

for j = 2:width+1

grad1 = sum(sum(sob\_img(i-1:i+1,j-1:j+1,k).\*sob\_h));

grad2 = sum(sum(sob\_img(i-1:i+1,j-1:j+1,k).\*sob\_v));

new(i,j,k) = abs(grad1) + abs(grad2);

end

end

end

new = uint8(new(2:hight+1,2:width+1));

% I=Sharpen\_filter(I,'Sobel');

else

prew\_h = [-1,-1,-1;0,0,0;1,1,1];

prew\_v = [-1,0,1;-1,0,1;-1,0,1]; % prewitt template

prew\_img = zeros(hight+2, width+2, 3);

prew\_img(2:hight+1, 2:width+1, :) = double(I);

new = zeros(hight+2, width+2, 3);

for k = 1:3

for i = 2:hight+1

for j = 2:width+1

grad1 = sum(sum(prew\_img(i-1:i+1,j-1:j+1,k).\*prew\_h));

grad2 = sum(sum(prew\_img(i-1:i+1,j-1:j+1,k).\*prew\_v));

new(i,j,k) = abs(grad1) + abs(grad2);

end

end

end

new = uint8(new(2:hight+1,2:width+1));

% I=Sharpen\_filter(I,'Prewitt');

end

axes(handles.Fig2);

imshow(new);title('»ùÓÚ´«Í³Ëã×ÓµÄ±ßÔµ¼ì²â');

**%%边界跟踪回调函数**

function boundary\_following\_Callback(hObject, eventdata, handles)

% hObject handle to boundary\_following (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

% ±ß½ç¸ú×Ù

global filename

global pathname

I=imread(fullfile(pathname,filename));

I=boundary\_tracking(I);

axes(handles.Fig2);

imshow(I);title('±ß½ç¸ú×Ù½á¹û');

**%%边界跟踪函数**

function[g]=boundary\_tracking(f)

offsetr=[-1,0,1,0]; % ÐÐÁÐËÄÁìÓòË÷Òý

offsetc=[0,1,0,-1];

next\_search\_dir\_table=[4,1,2,3]; % ÉèÖÃÏÂÒ»¸öËÑË÷·½Ïò

next\_dir\_table=[2,3,4,1];

start=-1;

boundary=-2;

[rv,cv]=find((f(2:end-1,:)>0)&(f(1:end-2,:)==0)); % Ñ°ÕÒ·ûºÏÌõ¼þµÄÐÐÁÐË÷Òý£¬ÕÒµ½ÆðÊ¼µã

rv=rv+1;

startr=rv(1); % ÉèÖÃÐÐÁÐÆðÊ¼µÄÐÐÁÐË÷Òý

startc=cv(1);

f=im2double(f); % Í¼Æ¬¸ñÊ½×ª³É¿É´¦ÀíÄ£Ê½

f(startr,startc)=start;

cur\_p=[startr,startc];

init\_departure\_dir=-1;

done=0;

next\_dir=2; % ³õÊ¼ËÑË÷·½Ïò

while ~done

dir=next\_dir;

for i=1:length(offsetr) % ËÄÁìÓòÑ°ÕÒÏÂÒ»¸ö±ßÔµµã

offset=[offsetr(dir),offsetc(dir)];

neighbor=cur\_p+offset;

if(f(neighbor(1),neighbor(2))~=0) % ÁÚ¾ÓÊôÓÚ±ßÔµµã

if((f(cur\_p(1),cur\_p(2))==start) & (init\_departure\_dir==-1))

init\_departure\_dir=dir;

elseif((f(cur\_p(1),cur\_p(2))==start) & (init\_departure\_dir==dir)) % ½áÊøÌõ¼þ

done=1;

break;

end

next\_dir=next\_search\_dir\_table(dir);

if(f(neighbor(1),neighbor(2))~=start)

f(neighbor(1),neighbor(2))=boundary;

end

cur\_p=neighbor; % ¸úÐÂµ±Ç°½Úµã

break;

end

dir=next\_dir\_table(dir); % Èç¹ûÎÞ±ßÔµµã¼ÌÐø

end

end

bi=find(f==boundary);

f(:)=0;

f(bi)=255;

f(startr,startc)=255;

g=im2bw(f);

end

**%%分水岭算法回调函数**

function watershed\_Callback(hObject, eventdata, handles)

% hObject handle to watershed (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

global filename

global pathname

I = imread(fullfile(pathname, filename));

I = double(I);

hv = fspecial('prewitt');

hh = hv.';

gv = abs(imfilter(I, hv, 'replicate'));

gh = abs(imfilter(I, hh, 'replicate'));

g = sqrt(gv.^2 + gh.^2);

df = bwdist(I);

L = watershed(df);

em = L==0;

im = imextendedmax(I,20);

g2 = imimposemin(g, im|em);

L2 = watershed(g2);

wr2 = L2==0;

I(wr2) = 255;

axes(handles.Fig2);

imshow(uint8(I));title('·ÖË®Áë·Ö¸î');

## 四、心得感悟

通过对Matlab的GUI制作，首先回顾了课内实验的前五次实验内容，是对前五次课内实验的一次总结。在图形交互界面中实现此前的课内实验，我加深了对此前5次实验的认识，对每个实验利用了什么原理，如何实验进行了复习，也明白了同一种方法可能在不同场景应用会得到不同的效果。而且如果要将这些实验制作成GUI，对代码运行的速度、时空复杂度有更高的要求，需要我们对代码进行优化。

其次第二个方面也明白了Matlab是如何制作一个GUI图形交互界面。与其他语言制作GUI的方式不同，Matlab提供了可视化界面让我们直接在界面上操作。一些控件可以通过直接拖动来实现控制，这大大简化了GUI设计的过程。其次对于每个控件，我们都可以用Matlab提供的回调函数来进行控制，在回调函数中添加每个实验的实现代码。