实验四 形态学图像处理及图像分割的应用

课程名称：图像处理基础

实验学时：4 学时

综合性、设计性实验： ☑是 ☑否

面向专业和班级： 19 级软件工程（1 班、2 班、3 班、4 班、中澳班、卓越班）

学生人数：102

任务编制人：课程负责人

实验内容

一、 形态学图像处理（1）

二、 形态学图像处理（2）

三、 图像边缘检测与图像分割（1）

四、 图像边缘检测与图像分割（2）

一、形态学图像处理(1)

(一) 实验目的

1.学习常见的数学形态学运算基本方法;

2.了解腐蚀、膨胀、开运算、闭运算取得的效果。

(二) 实验要求

1.利用 MatLab 工具箱中关于数学形态学运算的函数，对本实验中指定图像进行处理。

(三) 实验内容与步骤

实验(1)：

1.步骤：

- ✓ 调入并显示图像 Plane2.jpg；
- ✓ 选取合适的阈值，得到二值化图像 Plane2-2.jpg；
- ✓ 设置结构元素；
- ✓ 对得到的二值图像 Plane2-2.jpg 进行腐蚀运算；
- ✓ 对得到的二值图像 Plane2-2.jpg 进行膨胀运算；
- ✓ 对得到的二值图像 Plane2-2.jpg 进行开运算；
- ✓ 对得到的二值图像 Plane2-2.jpg 进行闭运算；
- ✓ 将两种处理方法的结果作比较；

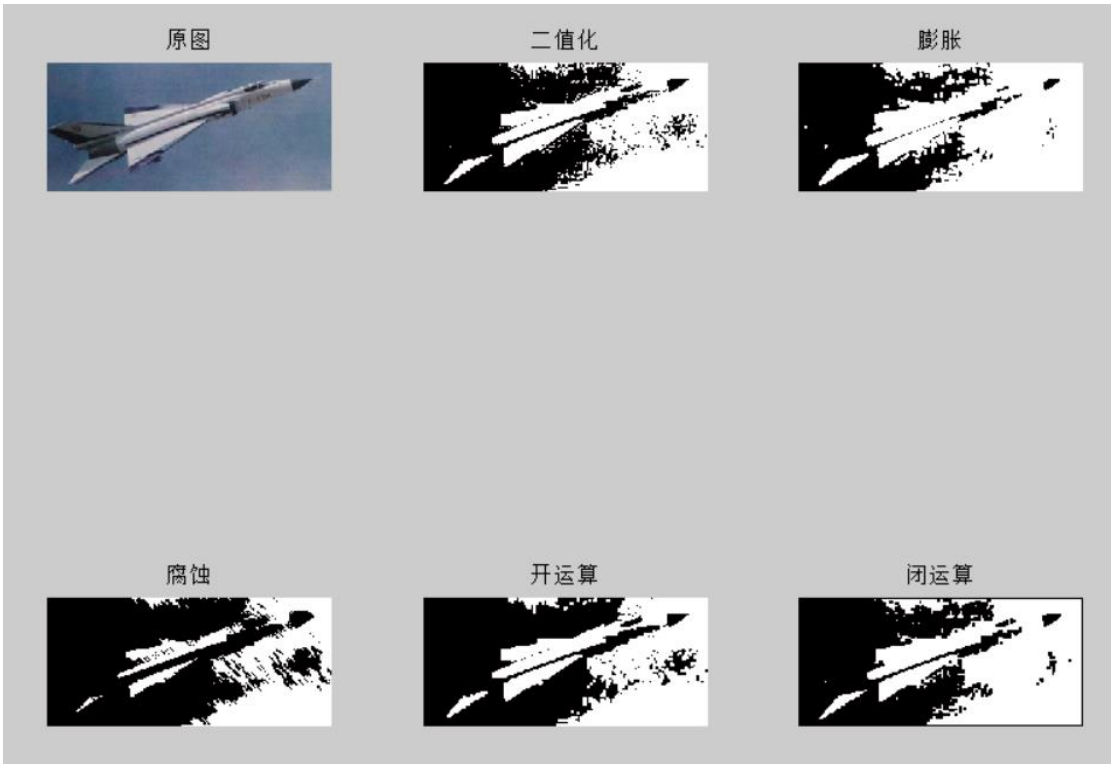
实验(1)采用图像 plane2. jpg

2.源代码：

```
I=imread('Plane2.jpg');
level = graythresh(I);                    %得到合适的阈值
bw = im2bw(I,level);                      %二值化
SE = strel('square',3);                   %设置膨胀结构元素
BW1 = imdilate(bw,SE);                    %膨胀
SE1 = strel('arbitrary',eye(5));          %设置腐蚀结构元素
BW2 = imerode(bw,SE1);                    %腐蚀
```

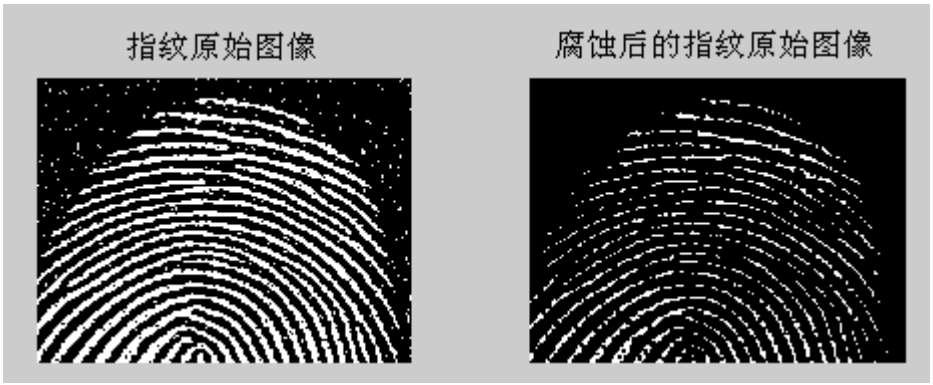
```
BW3 = bwmorph(bw, 'open');                  %开运算
BW4 = bwmorph(bw, 'close');                 %闭运算
imshow(I);
figure,imshow(bw);
figure,imshow(BW1);
figure,imshow(BW2);
figure,imshow(BW3);
figure,imshow(BW4);
```

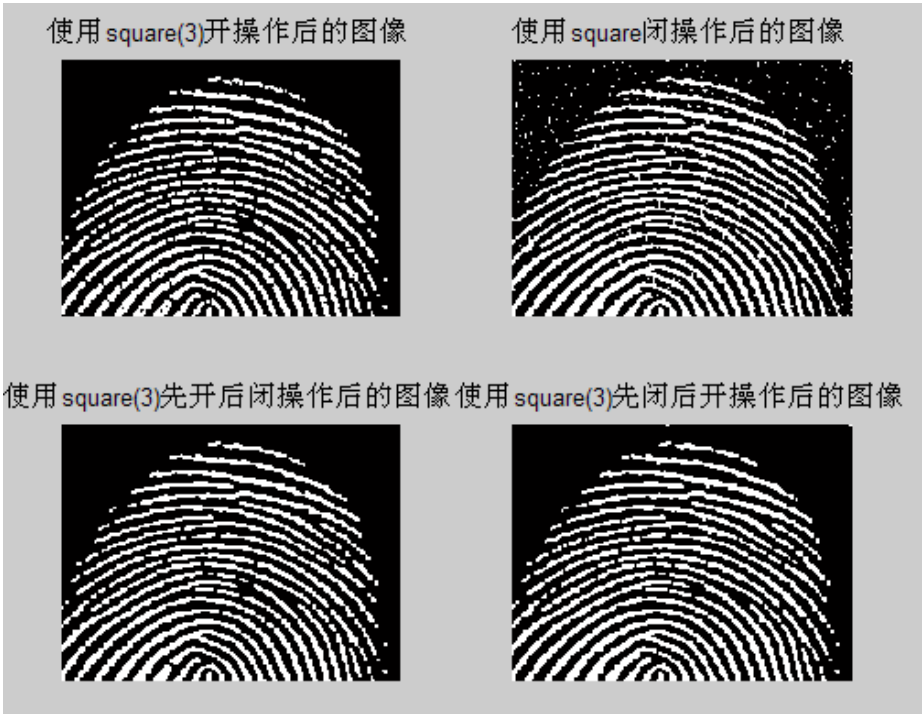
3.效果图：



实验(2)：

要求：首先读入指纹原始图像，然后根据剩下五幅图像提示，按照顺序，写出实现的步骤和 matlab 代码。





1.步骤：
2.代码：
3.结果图：

（四）思考题/问答题

1. 结合实验内容，评价腐蚀运算与膨胀运算的效果。
2. 结合实验内容，评价开运算与闭运算的效果。
3. 腐蚀、膨胀、开、闭运算的适用条件是什么？

二、形态学图像处理(2)

（一）实验内容与步骤

灰度级重构原图像

最后的重构结果


```
%% 使用重构删除复杂图像的背景
clc
clear
f=imread('.\images\dipum_images_ch09\Fig0930(a)(calculator).tif');
subplot(221),imshow(f);
title('灰度级重构原图像');

f_obr=imreconstruct(imerode(f,ones(1,71)),f);
subplot(222),imshow(f_obr);
title('经开运算重构图');

f_o=imopen(f,ones(1,71));
```

```
subplot(223),imshow(f_o);
title('经开运算后图');

f_thr=imsubtract(f,f_obr);
subplot(224),imshow(f_thr);
title('顶帽运算重构图')
432%使用重构删除复杂图像的背景 1:

f_th=imsubtract(f,f_o)
figure,subplot(221),imshow(f_th);
title('经顶帽运算图');

g_obr=imreconstruct(imerode(f_thr,ones(1,11)),f_thr);
subplot(222),imshow(g_obr);
title('用水平线对 f_thr 经开运算后重构图');

g_obrd=imdilate(g_obr,ones(1,2));
subplot(223),imshow(g_obrd);
title('使用水平线对上图进行膨胀');

f2=imreconstruct(min(g_obrd,f_thr),f_thr);
subplot(224),imshow(f2);
title('最后的重构结果');
449%使用重构删除复杂图像的背景 2:
```

（二）思考题/问答题

1.结合实验内容，介绍每一个函数的功能以及在本实验中的作用。

三、图像边缘检测和图像分割(1)

(一) 实验目的

1、对图像的边缘进行提取，通过对边缘的分析来分析图像的特征。

(二) 实验要求

（1）首先，了解一些术语的定义：

边缘点：图像中具有坐标[i,j]且处在强度显著变化的位置上的点。

边缘段：对应于边缘点坐标[i,j]及其方位θ,边缘的方位可能是梯度角。

边缘检测器：从图像中提取边缘（边缘点和边缘段）集合的算法。

轮廓：边缘列表，或者是一条表示边缘列表的拟合曲线。

边缘连接：从无序边缘表形成有序边缘表的过程，习惯上，边缘表的表示采用顺时针方向来排序。

边缘跟踪：一个用来确定轮廓的图像（指滤波后的图像）搜索过程。

边缘就是图像中包含的对象的边界所对应的位置。物体的边缘以图像局部特性的不连续性的形式出现的，例如，灰度值的突变，颜色的突变，纹理结构的突变等。从本质上说，边缘就意味着一个区域的终结和另外一个区域的开始。图像边缘信息在图像分析和人的视觉中十分重要，是图像识别中提取图像特征的一个重要属性。

边缘检测（edge detection）在图像处理和对象识别领域中都是一个重要的基本问题。由于边缘的灰度不连续性，可以使用求导数的方法检测到。最早的边缘检测方法都是基于像素的数值导数的运算。本实验主要是对图像依次进行 Sobel 算子，Prewitt 算子,Roberts 算子，Laplace 算子和 Canny 算子运算，比较处理结果。

边缘检测有三个共性准则，

1，好的检测结果，或者说对边缘的误测率尽可能低，就是在图像边缘出现的地方检测结果中不应该没有；另一方面不要出现虚假的边缘。

2，对边缘的定位要准确，也就是我们标记出的边缘位置要和图像上真正边缘的中心位置充分接近。

3，对同一边缘要有尽可能低的响应次数，也就是检测响应最好是单像素的。

(三) 对图像进行各种算子运算

本实验中主要是对图像依次进行 Sobel 算子，Prewitt 算子,Roberts 算子，Laplace 算

子和 Canny 算子运算。

（四）Matlab 代码:

```
clear all;
close all;
warning off all;

I = imread('cameraman.jpg');
%%没有噪声时的检测结果
BW_sobel = edge(I,'sobel');
BW_prewitt = edge(I,'prewitt');
BW_roberts = edge(I,'roberts');
BW_laplace = edge(I,'log');
BW_canny = edge(I,'canny');
figure(1);
subplot(2,3,1),imshow(I),xlabel('原始图像');
subplot(2,3,2),imshow(BW_sobel),xlabel('sobel 检测');
subplot(2,3,3),imshow(BW_prewitt),xlabel('prewitt 检测');
subplot(2,3,4),imshow(BW_roberts),xlabel('roberts 检测');
subplot(2,3,5),imshow(BW_laplace),xlabel('laplace 检测');
subplot(2,3,6),imshow(BW_canny),xlabel('canny 检测');

%%加入高斯噪声（μ=0，σ^2=0.01）检测结果
I_g1 = imnoise(I,'gaussian',0,0.01);
BW_sobel = edge(I_g1,'sobel');
BW_prewitt = edge(I_g1,'prewitt');
BW_roberts = edge(I_g1,'roberts');
BW_laplace = edge(I_g1,'log');
BW_canny = edge(I_g1,'canny');
figure(2);
subplot(2,3,1),imshow(I_g1),xlabel('加入高斯噪声(μ=0，σ^2=0.01)图像');
subplot(2,3,2),imshow(BW_sobel),xlabel('sobel 检测');
subplot(2,3,3),imshow(BW_prewitt),xlabel('prewitt 检测');
subplot(2,3,4),imshow(BW_roberts),xlabel('roberts 检测');
subplot(2,3,5),imshow(BW_laplace),xlabel('laplace 检测');
subplot(2,3,6),imshow(BW_canny),xlabel('canny 检测');

%%加入高斯噪声（μ=0，σ^2=0.02）检测结果
I_g2 = imnoise(I,'gaussian',0,0.02);
BW_sobel = edge(I_g2,'sobel');
BW_prewitt = edge(I_g2,'prewitt');
BW_roberts = edge(I_g2,'roberts');
BW_laplace = edge(I_g2,'log');
BW_canny = edge(I_g2,'canny');
```

```
figure(3);
subplot(2,3,1),imshow(I_g2),xlabel('加入高斯噪声(μ=0, σ^2=0.02)图像');
subplot(2,3,2),imshow(BW_sobel),xlabel('sobel 检测');
subplot(2,3,3),imshow(BW_prewitt),xlabel('prewitt 检测');
subplot(2,3,4),imshow(BW_roberts),xlabel('roberts 检测');
subplot(2,3,5),imshow(BW_laplace),xlabel('laplace 检测');
subplot(2,3,6),imshow(BW_canny),xlabel('canny 检测');
```

（五）实验结果：

原始图像

sobel检测

prewitt检测

roberts检测

laplace检测

canny检测

加入高斯噪声(μ=0，σ^2=0.01)图像

sobel检测

prewitt检测

roberts检测

laplace检测

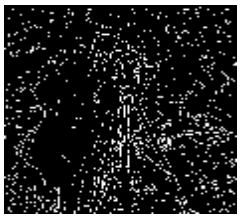
canny检测

加入高斯噪声(μ=0，σ^2=0.02)图像

sobel检测

prewitt检测

roberts检测



laplace检测

canny检测

(六) 实验分析:

通过对上述几种算子的研究，我们可以发现，几种算子的优缺点？

Prewit t 算子和 Sobel 算子……

Robert s 算子……

Laplace 算子……

Canny 算子……

通过上述实验结果我们可以发现，在加入高斯噪声以后，几种算子的性能情况？

四、图像边缘检测和图像分割(2)

（一）实验目的

（1）了解并掌握图像分割的基本原理；

（2）编写程序使用 Hough 变换处理图像，进行线检测；

（3）编写程序使用阈值处理方法进行图像分割，根据实验结果分析效果；

（4）总结实验过程：方案、编程、调试、结果、分析、结论。

（二）实验内容和要求

(1)对 256 级灰度的数字图像 camera.bmp（如图 1 所示）进行 Hough 变换进行线检测，显示处理前、后图像；思考如何利用 Hough 变换进行圆检测；

图 1 实验图像 camera.bmp

图 2 实验图像 car.bmp

（2）对图像 car.bmp(如图 2 所示）分别利用不同的阈值处理方法(直方图阈值法，最佳阈值法，分水岭分割法)进行图像中汽车及车牌的分割，显示处理前、后图像；思考不同的阈值处理算法对分割效果的影响？

（三）实验代码：

第一题源程序：

```
clear all;
RGB = imread('camera.bmp');
I=RGB;
%I  = rgb2gray(RGB);
BW = edge(I,'canny');    % 利用Canny算子提取图像边缘
[H,T,R] = hough(BW,'RhoResolution',0.5,'ThetaResolution',0.5);
subplot(121)
imshow(H,[],'InitialMagnification','fit'), axis on, axis normal
xlabel('\T'), ylabel('\R')
p = houghpeaks(H,5,'threshold',ceil(0.3*max(H(:))));
%找到5个较明显的Hough变换峰值
hold on
```

```
plot(T(p(:,2)),R(p(:,1)),'s','color','white');
lines = houghlines(BW,T,R,p,'FillGap',10,'MinLength',10);
  %查找并链接线段
figure, imshow(BW), hold on  %在二值图中叠加显示这些线段
for k = 1:length(lines)
    xy = [lines(k).point1; lines(k).point2];
    plot(xy(:,1),xy(:,2),'LineWidth',2,'Color','green');
end
```

第二题源程序：

```
(1)%用直方图阈值法
I=imread('car.bmp');
subplot(2,2,1);
imshow(I);
title('原图像');
I1=rgb2gray(I);
subplot(2,2,2);
imhist(I1);
title('直方图');
subplot(2,2,3);
I2=im2bw(I1,165/415);
imshow(I2);
title('分割后的图像');

(2)%用Otsu算法
I=imread('car.bmp');
subplot(2,1,1);
imshow(I);
title('原图像');
subplot(2,1,2);
level=graythresh(I);
BW=im2bw(I,level);
imshow(BW);
title('分割后的图像');

(3)%分水岭分割算法
f=imread('car.bmp');
imshow(f);
g=im2bw(f, graythresh(f));
figure,imshow(g);
gc=~g;
D=bwdist(gc);
L=watershed(-D);
w=L==0;
```

```
g2=g&~w;
figure,imshow(g2);
```

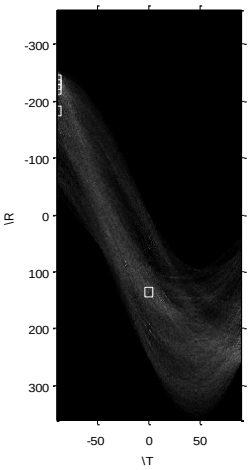
(四) 实验分析

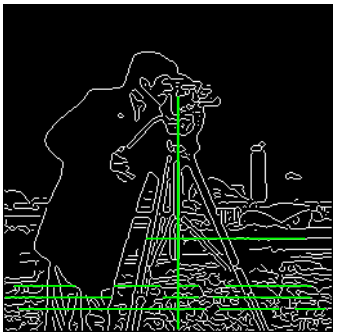
(1)第一题

Hough 变换是最常用的直线提取方法，它的基本思想是：

Hough 变换对圆的检测：

第一题结果图





(2)第二题

实验原理：

结果图：

思考：

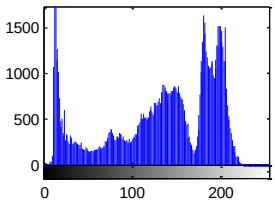
直方图阈值法：

Otsu 法：

a、直方图阈值法:

原图像

直方图



分割后的图像

b、Otsu 法:

原图像

分割后的图像

c、分水岭分割法:（请自行补充）