《数字图像处理I实践》

李军侠

南京信息工程大学计算机学院

实验一　图像基本变换

一、实验意义及目的

（1）了解和掌握图像处理工具 Matlab，熟悉基于 Matlab 的图像处理函数，并为下一步编程进行图像处理打下基础。

（2）理解色彩的概念，掌握图像代数运算，几何变换方法。

二、实验内容

打开一幅彩色图像 Image1，使用 Matlab 图像处理函数，对其进行下列变换：

（1）将 Image1 色彩通道互换，并显示效果；

（2）将 Image1 灰度化为 gray，并显示灰度化后图像；

（3）采用不同的插值方法实现 gray 的旋转、放大变换；

（4）打开另一幅彩色图像 Image2，和 Image1 进行代数运算，要求运用拼接、加减乘除等多种技术；

（5）实验要求中的拓展内容。

三、Matlab 相关函数介绍

（1）imread 函数

功能：实现多种类型图像文件的读取，如：BMP、GIF、JPEG、PNG、RAS 等。

调用格式：A = imread(filename, fmt)。filename 为图像文件名，可以是灰度图像，也可以是彩色图像，如果文件不在当前目录或不在 Matlab 目录下，则需要列全文件路径。fmt 为文件的扩展名，指定文件类型。A 为图像数据矩阵。

（2）imshow 函数

功能：显示图像。

调用格式：

imshow(I,n)：显示灰度图像 I，n 为要显示图像的灰度等级，整数，默认为 256。

Imshow(I,[LOW HIGH])：以规定的灰度级范围[LOW HIGH]来显示灰度图像 I，低于 LOW 值的显示为黑，高于 HIGH 值的显示为白，默认按 256 个灰度级显示。

imshow(RGB)：显示真彩色图像 RGB。

imshow(BW)：显示二值图像 BW。

imshow(X,map)：显示索引图像，X 为索引图像的数据矩阵，map 为其颜色映射表。

imshow filename：显示 filename 指定的图像，若文件包括多帧图像，则显示第一幅，且文件必须在 MATLAB 的当前目录下。

（3）imwrite 函数

功能：实现图像文件的保存。

调用格式：

imwrite(A,’filename’,fmt)：A 是要保存的图像数据矩阵，filename 是文件名，fmt 是文件格式。

imwrite(X,map,’filename’,fmt)：X 为索引图像的数据矩阵，map 为其颜色映射表。

（4）rgb2hsv 函数

功能：实现 RGB 数据图像向 HSV 数据图像的转换。

调用格式：

HSV = rgb2hsv(RGB)。RGB 为 RGB 彩色图像，为 3 维矩阵；HSV 为 3 维 HSV 图像矩阵，3 维依次为 H、S、V，取值均在[0,1]范围内。

（5）rgb2ycbcr 函数

YCBCR = rgb2ycbcr(RGB) ：实现 RGB 数据图像向 YCbCr 数据图像的转换。

（6）rgb2gray 函数

功能：彩色图像灰度化。

调用格式：

I = rgb2gray(RGB)：真彩色 RGB 图像变换为灰度图像 I。

NEWMAP = rgb2gray(MAP)：变换索引图像的调色板为灰度调色板。

（7）imrotate 函数

功能：实现图像旋转。

调用格式：

B = imrotate(A,ANGLE,METHOD,BBOX)：A 为要进行旋转的图像；ANGLE 为要旋转的角度（°）逆时针为正，顺时针为负；METHOD 为图像旋转插值方法，可取“'nearest', 'bilinear', 'bicubic'”，默认为 nearest；BBOX 指定返回图像大小，可取“crop”，输出图像 B 与输入图像 A 具有相同的大小，对旋转图像进行剪切以满足要求；可取“loose”，默认是，B 包含整个旋转后的图像。

（8）imresize 函数

功能：实现图像缩放。

调用格式：

B = imresize(A, SCALE,METHOD))：返回原图 A 的 SCALE 倍大小图像 B；

B = imresize(A, [NUMROWS NUMCOLS], METHOD))：对原图 A 进行比例缩放，返回图像 B 的行数 NUMROWS 和列数 NUMCOLS，如果二者为 NaN，表明 Matlab 自动调整了图像的缩放比例；

[Y, NEWMAP] = imresize(X, MAP, SCALE, METHOD))：对索引图像进行成比例缩放。

（9）imtransform 函数

功能：实现图像几何变换。

调用格式：

B = imtransform(A,TFORM,INTERP,param1,val1,param2,val2,…) ：对图像 A 实现空间变换，

TFORM 为 maketform 函数或 cp2tform 函数产生的结构；INTERP 为插值方法，可取“'nearest', 'bilinear', 'bicubic'”。

T = maketform(TRANSFORMTYPE,...)：产生转换结构；TRANSFORMTYPE 为变换类型，可以为“'affine', 'projective', 'custom', 'box', 'composite'”。

（10）fliplr 函数

B=fliplr(X)：实现二维矩阵 X 沿垂直轴的左右翻转。

（11）flipud 函数

B= flipud(X)：实现二维矩阵 X 上下翻转。

（12）flipdim 函数

B=flipdim(X,DIM)：使矩阵 X 按特定轴翻转，dim 指定翻转方式：为 1 表示按行翻转；为 2 表示按列翻转。

（13）permute 函数

B = permute(A,ORDER)：按照向量 ORDER 指定的顺序重排 A 的各维，B 中元素和 A 中元素完全相同，但在 A、B 访问同一个元素使用的下标不一样。order 中的元素必须各不相同。

（14）imadd 函数

C=imadd(A,B) ：实现两幅图像相加。

1）A、B 均为图像，则要求 B 和 A 的尺寸相等；若 B 是一个标量，则 C 表示对图像 A 整体加上某个值（对小数部分取整）。

2）假如 A 和 B 对应运算和大于 255，C 仍取 255，即截断处理；为避免截断，可以将 C 存储为 uint16，即 C=imadd(A,B,’uint16’)。

（15）imsubtract 函数

功能：实现两幅图像相减。

调用格式：

C=imsubtract(A,B)：差值结果小于 0 的赋值为 0，对 A、B 的要求同 imadd 相同。

C=imabsdiff(A,B)：差值结果取绝对值。

（16）immultiply 函数

C=immultiply(A,B)：实现两幅图像相乘。

（17）imdivide 函数

C=imdivide(A,B) ：实现两幅图像相除。

四、参考代码

参考代码中实现了彩色图像的灰度化、旋转、缩放两种几何变换以及镜像及拼接。

```
Image1=imread('peppers.jpg');
%红绿通道互换
```

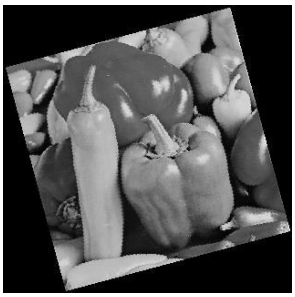
```
Image2=Image1;
Image2(:,:,1)=Image1(:,:,2);
Image2(:,:,2)=Image1(:,:,1);
imshow(Image2);
imwrite(Image2,'changecolor.jpg');
%灰度化
gray=rgb2gray(Image1);
figure；
subplot(121),imshow(Image1),title('Original Image');
subplot(122),imshow(gray),title('Gray Image');
imwrite(gray,'grayimage.jpg');
 %图像旋转
Newgray1=imrotate(gray,15);
Newgray2=imrotate(gray,15,'bilinear');
figure;
subplot(121),imshow(Newgray1),title('旋转15°（最邻近插值）');
subplot(122),imshow(Newgray2),title('旋转15°（双线性插值）');
imwrite(Newgray1,'rotate1.jpg');
imwrite(Newgray2,'rotate2.jpg');
 %图像缩放
Newgray3=imresize(gray,2.5,'nearest');
Newgray4=imresize(gray,2.5,'bilinear');
figure;
subplot(121),imshow(Newgray3),title('放大2.5倍（最邻近插值）');
subplot(122),imshow(Newgray4),title('放大2.5倍（双线性插值）');
imwrite(Newgray3,'scale1.jpg');
imwrite(Newgray4,'scale2.jpg');
 %图像镜像与拼接
Image2=imread('lotus.bmp');
HImage=flipdim(Image2,2);
VImage=flipdim(Image2,1);
CImage=flipdim(HImage,1);
[h w]=size(Image2);
NewImage=zeros(h*2,w*2,3);
NewImage=[Image2 HImage;VImage CImage];
```

```
figure,imshow(NewImage);
imwrite(NewImage,'newlotus.jpg');
```

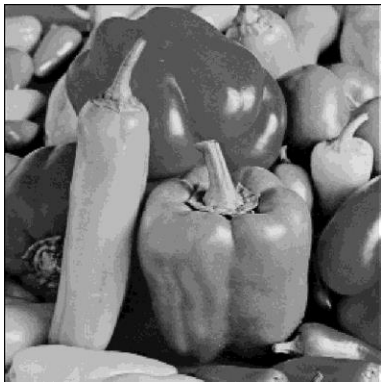
程序的运行效果如下：


（a）红绿色彩通道互换


（b）灰度化


（c）旋转（最邻近插值）


（d）旋转（双线性插值）


（e）放大（最邻近插值）


（f）放大（双线性插值）


（g）图像镜像与拼接

图1 实验一参考代码运行效果

peppers.jpg 和 lotus.bmp 为 Matlab 目录下的图像文件。

五、实验要求

1.熟悉 Matlab 函数，读懂参考代码；

2.拓展内容：

（1）将彩色图像采用不同的灰度化方法实现灰度化；

（2）将彩色图像变换到 YCbCr、HSV 空间，熟悉各分量数据并显示；

（3）不采用 Matlab 函数，自行设计基于双线性插值的图像放大程序。

六、实验考核

实验报告

七、实验报告

实验结束后，撰写实验报告，实验报告主题部分应包括：算法原理、程序流程、算法各部分主要函数代码以及功能注释、运行结果四部分，按照撰写情况打分。

实验二 图像增强

一、实验意义及目的

（1）进一步掌握图像处理工具 Matlab，熟悉基于 Matlab 的图像处理函数。

（2）掌握各种图像增强方法。

二、实验内容

打开一幅彩色图像 Image1，使用 Matlab 图像处理函数，对其进行下列变换：

（1）将 Image1 灰度化为 gray，统计并显示其灰度直方图；

（2）对 gray 进行分段线性变换；

（3）对 gray 进行直方图均衡化；

（4）对 gray 进行伪彩色增强；

（5）对 gray 添加噪声并平滑；

（6）对 gray 利用 Sobel 算子锐化；

（7）实验要求中的拓展内容。

三、Matlab 相关函数介绍

（1）imhist 函数

功能：统计变显示图像的直方图。

调用格式：

imhist(I)：显示图像 I 的直方图。

imhist(I, n)：显示图像 I 的直方图，n 指定直方图中的列数。

[COUNTS,X] = imhist(...)：返回直方图数据向量 COUNTS 和相应的色彩值向量 X。

（2）histeq 函数

功能：直方图均衡化

调用格式：

J = histeq(I,hgram)：将图像 I 的直方图变成用户指定的向量 hgram，hgram 中的各元素值域为[0,1]；

J = histeq(I,N)：对原始图像 I 进行直方图均衡化，N 为输出图像的灰度技术，默认 N 为 64。

（3）imadjust 函数

功能：调整图像灰度值或颜色映射表，增加图像的对比度。

调用格式：

J = imadjust(I,[LOW_IN; HIGH_IN],[LOW_OUT; HIGH_OUT],GAMMA)：调整图像 I 的灰度值；[LOW_IN; HIGH_IN]指定原始图像中要变换的灰度范围；[LOW_OUT; HIGH_OUT]指定变换后的灰

度范围；低于 LOW_IN 、高于 HIGH_IN 的采取截取式；都可以使用空的矩阵[]，默认值是[0 1]；GAMMA 为标量，指定描述值 I 和值 J 关系的曲线形状，如果小于 1，此映射偏重更高数值（明亮）输出，如果 gamma 大于 1，此映射偏重更低数值（灰暗）输出，如果省略此参数，默认为（线性映射）。

NEWMAP = imadjust(MAP,[LOW_IN; HIGH_IN],[LOW_OUT; HIGH_OUT],GAMMA)：调整索引图像的颜色表 map，其他参量同上。

RGB2 = imadjust(RGB1,...)：对 RGB 图像 RGB1 的 R、G、B 分量进行调整。

（4）imnoise 函数

J = imnoise(I,type,parameters)：按指定类型在图像 I 上添加噪声；type 表示噪声类型，parameters 为其所对应参数，可取值如表 1 所示：

表 1 imnoise 函数参数表

type 参数	描述	parameters
‘gaussian’	高斯白噪声，均值和方差均为常数	均值和方差，默认为 0 和 0.01
‘localvar’	零均值高斯白噪声，方差取决于灰度	方差，维数与图像 I 相同
‘poisson’	泊松噪声	均值
‘salt & pepper’	椒盐噪声	噪声密度，默认为 0.05
‘speckle’	乘法噪声	方差，默认为 0.04

（5）fspecial 函数

h=fspecial(type,parameters)：创建指定类型和参数的二维滤波器 h，如表 2 所示：

表 2 fspecial 函数参数表

type	parameters	含义
‘average’	n	均值滤波器，n 为模板尺寸，默认为[3 3]
‘disk’	radius	圆形均值滤波器，radius 为半径，默认为 0.5
‘gaussian’	hsize,sigma	高斯低通滤波器，参数为滤波器大小和标准差，默认为 0.5
‘laplacian’	alpha	近似于二维 laplacian 算子，alpha∈[0,1],默认为 0.2
‘log’	n,sigma	log 算子, n 为模板尺寸，默认为[3 3] ,sigma 为滤波器标准差，默认为 0.5
‘motion’	len,theta	按角度 theta 移动 len 像素的运动滤波器，len 默认为 9，theta 默认为 0
‘prewitt’	无	Prewitt 水平边缘强化滤波器
‘sobel’	无	Sobel 水平边缘强化滤波器
‘unsharp’	alpha	由 alpha 决定的 laplacian 算子，alpha∈[0,1],默认为 0.2

（6）filter2 函数

Y = filter2(B,X,shape)：使用二维 FIR 滤波器 B 对矩阵 X 进行滤波；shape 指定返回值 Y 的形式，‘full’：Y 的维数大于 X；‘same’：Y 的维数等于 X；‘valid’：Y 的维数小于 X；默认为 same。

（7）imfilter 函数

B=imfilter(A,H,option1,option2,…)：根据指定的属性 option1，option2…等，使用多维滤波器 H 对图像 A 进行滤波，H 常由函数 fspecial 输出得到。属性参数如表 3 所示：

表 3 imfilter 函数参数表

参数类型	参数	含义
边界选项	‘X’	输入图像的外部边界通过 X 来扩展，默认的 X=0
	‘symmetric’	输入图像的外部边界通过镜像反射其内部边界来扩展
	‘replicate’	输入图像的外部边界通过复制内部边界的值的来扩张
	‘circular’	输入图像的外部边界通过假设输入图像是周期函数来扩张
输出大小选项	‘same’	输入图像和输出图像同样大小，默认操作
	‘full’	输出图像比输入图像大
滤波方式选项	‘corr’	使用相关进行滤波
	‘conv’	使用卷积进行滤波

（8）medfilt2 函数

B = medfilt2(A,[m n])：用[m n]大小的滤波器对图像 A 进行中值滤波，输出图像为 B，滤波器大小默认为 3×3。

（9）edge 函数

BW = edge(I)：对灰度或二值图像 I 进行边缘检测，检测后图像为二值图像 BW，边界处取值为 1，否则为 0。缺省情况下，edge 函数使用 Sobel 算子检测边缘，也可以指定算子。

BW = edge(I,'sobel')；BW = edge(I,'prewitt')；BW = edge(I,'roberts')；

BW = edge(I,'log')；BW = edge(I,'canny')引号内为指定算子。

BW = edge(I,'sobel',thresh)：thresh 指定保留边缘的阈值，若为 0，edge 函数自动选择该值。

四、参考代码

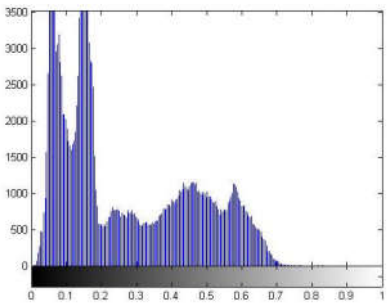
参考代码中实现了彩色图像的灰度化、灰度图像的直方图统计、分段线性变换、直方图均衡化、伪彩色增强、添加噪声与中值滤波、Sobel 算子滤波。

```
Image1=im2double(imread('lotus.bmp'));
gray=rgb2gray(Image1);
imhist(gray);
[h,w]=size(gray);
NewImage1=zeros(h,w);
a=80/256; b=180/256; c=30/256; d=220/256;
for x=1:w
    for y=1:h
        if gray(y,x)<a
            NewImage1(y,x)=gray(y,x)*c/a;
        elseif gray(y,x)<b
```

```
                NewImage1(y,x)=(gray(y,x)-a)*(d-c)/(b-a)+c;
            else
                NewImage1(y,x)=(gray(y,x)-b)*(255-d)/(255-b)+d;
            end
        end
end
figure,imshow(NewImage1),title('分段线性变换');

NewImage2=histeq(gray);
figure,imshow(NewImage2),title('直方图均衡化');

NewImage3=zeros(h,w,3);
for x=1:w
    for y=1:h
        if gray(y,x)<64/256
            NewImage3(y,x,1)=0;
            NewImage3(y,x,2)=4*gray(y,x);
            NewImage3(y,x,3)=1;
        elseif gray(y,x)<128/256
            NewImage3(y,x,1)=0;
            NewImage3(y,x,2)=1;
            NewImage3(y,x,3)=2-4*gray(y,x);
        elseif gray(y,x)<192/256
            NewImage3(y,x,1)=4*gray(y,x)-2;
            NewImage3(y,x,2)=1;
            NewImage3(y,x,3)=0;
        else
            NewImage3(y,x,1)=1;
            NewImage3(y,x,2)=4-4*gray(y,x);
            NewImage3(y,x,3)=0;
        end
    end
end
figure,imshow(NewImage3);
```

```
noiseIsp=imnoise(gray,'salt & pepper',0.1);
noiseIg=imnoise(gray,'gaussian');
result1=medfilt2(noiseIsp);
result2=medfilt2(noiseIg);
figure;
subplot(121),imshow(result1),title('椒盐噪声3×3中值滤波');
subplot(122),imshow(result2),title('高斯噪声3×3中值滤波');

H1=[-1 -2 -1;0 0 0;1 2 1];
H2=[-1 0 1;-2 0 2;-1 0 1];
R1=imfilter(gray,H1);
R2=imfilter(gray,H2);
edgeImage=abs(R1)+abs(R2);
sharpImage=gray+edgeImage;
figure；
subplot(121),imshow(edgeImage),title('Sobel梯度图像');
subplot(122),imshow(sharpImage),title('Sobel锐化图像');
```

程序的运行效果如下：



（a）灰度直方图

（b）分段线性变换

（c）直方图均衡化

（d）伪彩色增强

（e）抑制椒盐噪声

（f）抑制高斯噪声

（g）Sobel梯度图像

（h）Sobel锐化图像

图2 实验二参考代码运行效果

五、实验要求

1. 熟悉 Matlab 函数，读懂参考代码；
2. 拓展内容：

（1）对以上处理变换参数，查看处理效果；

（2）更改伪彩色增强方法为热金属编码或彩虹编码；

（3）设计不同的平滑滤波、锐化滤波方法，查看处理效果；

（4）自行设计方法，实现对彩色图像增强处理。

六、实验考核

实验报告

七、实验报告

实验结束后，撰写实验报告，实验报告主题部分应包括：算法原理、程序流程、算法各部分主要函数代码以及功能注释、运行结果四部分，按照撰写情况打分。

实验三 图像分割与描述

一、实验意义及目的

（1）进一步掌握图像处理工具 Matlab，熟悉基于 Matlab 的图像处理函数。

（2）掌握图像分割方法，熟悉常用图像描述方法。

二、实验内容

打开一幅图像 Image，使用 Matlab 图像处理函数，对其进行下列变换：

（1）将 Image 灰度化为 gray，对其进行阈值分割转换为 BW；

（2）对 BW 进行数学形态学滤波；

（3）对 BW 进行边缘跟踪，用红色线在图中标出；

（4）计算各区域边界点的傅里叶描绘子并用四分之一点重建边界；

（5）实验要求中的拓展内容。

三、Matlab 相关函数介绍

（1）im2bw 函数

BW = im2bw(I, level)：以 level 为阈值把灰度图像 I 转变为二值图像。

（2）graythresh 函数

level = graythresh(I)，使用 Otsu 方法获取阈值，level 被归一化到[0,1]区间。

（3）imbinarize 函数

BW = imbinarize(I)：采用基于 OTSU 方法的全局阈值实现灰度图像 I 的二值化。

BW = imbinarize(I, METHOD) ：采用 METHOD 指定的方法获取阈值实现灰度图像 I 的二值化。METHOD 可选'global'和'adaptive'，前者指定 OTSU 方法，后者采用局部自适应阈值方法。

（4）bwboundaries 函数

搜索二值图像 BW 的外边界和内边界。

[B,L,N,A] = bwboundaries(BW,CONN,OPTIONS) ：函数视 BW 中为 0 的元素为背景像素点，为 1 的元素为待提取边界目标。B 中的每个元素均为 Q×2 矩阵，矩阵中每一行包含边界像素点的行坐标和列坐标，Q 为边界所含像素点的个数。L，标识矩阵，标识二值图像中被边界所划分的区域；N，区域的数目 N；A，被划分的区域的邻接关系。CONN 取 4，搜索中采用 4 连通方法，默认取 8，即 8 连通方法。OPTIONS 指定算法的搜索方式，默认为‘holes’，搜索目标的内外边界，‘noholes’只搜索目标的外边界。

（5）bwtraceboundary 函数

跟踪二值图像 BW 中目标轮廓。

B = bwtraceboundary(BW,P,FSTEP)：目标区域取值非 0；参数 P 是初始跟踪点的行列坐标二元矢量；FSTEP 表示初始查找方向，用于寻找对象中与 P 相连的下一个像素，可取'N'、'NE'、'E'、'SE'、'S'、'SW'、'W'、'NW'；返回值 B 为边界坐标值，是一个 Q×2 矩阵。

（6）strel 函数

创建形态学结构元素。

SE = strel(shape,parameters)，创建一个由 shape 指定的结构元素，其中 shape 的种类有：arbitrary、pair、diamond、periodicline、disk、rectangle、line、square、octagon，参数 parameters 一般控制 SE 的大小。

（7）imdilate 函数

IM2 = imdilate(IM, SE)：膨胀图像 IM，返回膨胀后的图像 IM2，SE 是结构元素。

（8）imerode 函数

IM2 = imerode(IM,SE)：腐蚀图像 IM，返回腐蚀后的图像 IM2，SE 是结构元素。

（9）imopen 函数

IM2 = imopen(IM,SE)：对图像 IM 进行开运算，返回图像为 IM2，SE 是结构元素。

（10）imclose 函数

IM2 = imclose(IM,SE)：对图像 IM 进行闭运算，返回图像为 IM2，SE 是结构元素。

（11）fft 和 fft2 函数

fft(X)：对序列 X 进行 DFT 运算。

fft2(X)：对矩阵 X 进行二维 DFT 运算。

（12）ifft 和 ifft2 函数

ifft(X)：对 X 进行 IDFT 运算。

ifft2(F)：对 F 进行二维 IDFT 运算。

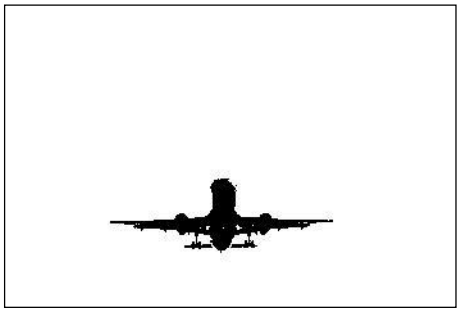
四、参考代码

参考代码中实现了图像的阈值分割、数学形态学滤波、边缘跟踪、傅里叶描绘子计算及重建。

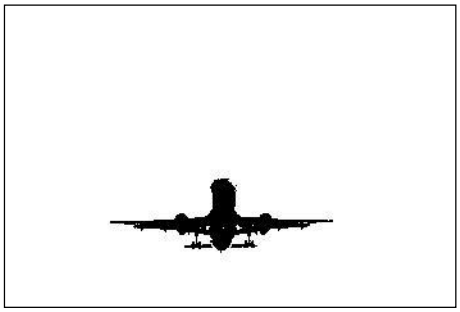
```
Image1=im2double(imread('plane.jpg'));
gray=rgb2gray(Image1);
T=graythresh(gray);
BW=im2bw(gray,T);
figure,imshow(BW),title('二值化图像');

SE=strel('square',3);
Morph=imopen(BW,SE);
Morph=imclose(Morph,SE);
figure,imshow(Morph),title('形态学滤波');
```

```
[B L]=bwboundaries(1-Morph);
figure,imshow(L),title('划分的区域');
hold on;
for i=1:length(B)
    boundary=B{i};
    plot(boundary(:,2),boundary(:,1),'r','LineWidth',2);
end

M=zeros(length(B));
for k=1:length(B)
    N=length(B{k});
    if N/2~=round(N/2)
        B{k}(end+1,:)=B{k}(end,:);
        N=N+1;
    end
    M(k)=[N*3/4];
end
S=zeros(size(Morph));
figure,imshow(S);
hold on;
for k=1:length(B)
    z=B{k}(:,2)+1i*B{k}(:,1);
    Z=fft(z);
    [Y I]=sort(abs(Z));
    for count=1:M(k)
        Z(I(count))=0;
    end
    zz=ifft(Z);
    plot(real(zz),imag(zz),'w');
end
```

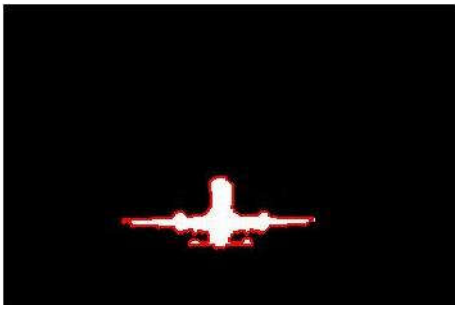
程序运行效果如下：



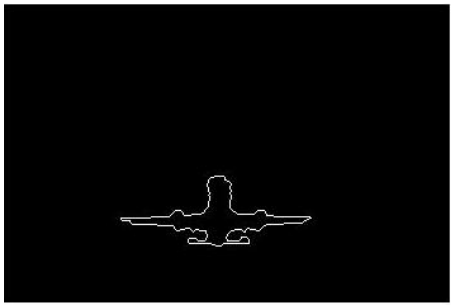
（a）二值化图像



（b）形态学滤波



（c）划分的区域



（d）重建图像

图3 实验三参考代码运行效果

五、实验要求

1. 熟悉 Matlab 函数，读懂参考代码；

2. 拓展内容：

（1）尝试不同的阈值选择方法，实现灰度图像二值化；

（2）变换参数实现形态学滤波，查看滤波效果；

（3）更改重建边界点数，查看效果；

（4）自行设计方法实现图像分割，并计算分割区域相关参数。

六、实验考核

实验报告

七、实验报告

实验结束后，撰写实验报告，实验报告主题部分应包括：算法原理、程序流程、算法各部分主要函数代码以及功能注释、运行结果四部分，按照撰写情况打分。

实验四 图像综合处理

一、实验意义及目的

充分利用所学各种图像处理技术，实现对图像的综合处理，加深对基础知识的理解和应用。

二、实验内容

在下列题目中任意选择一个题目完成实验。

1. 目标与背景的分割与提取 1

主要要求：提取红苹果

建议方法：

（1）将已知图像进行消噪处理

（2）对彩色图像进行目标和背景分析

（3）通过阈值法将图像进行分割

（4）提取目标

难点：

（1）确定目标区域的特征；

（2）边界修复与区域分割。

2. 目标与背景的分割与提取 2

主要要求：从图像中检索出篮球。

建议方法：

（1）对待检测图片进行色彩空间变换；

（2）利用色度信息进行分割；

（3）区域修复；

（4）目标提取。

难点：

篮球上的黑色花纹对边缘检测、色彩检测均造成干扰。

3. 硬币检测及计数

主要要求：白色背景上扔几枚不同面值的硬币，拍摄图像，通过图像处理，获取硬币的数目及

金额。

建议方法：

（1）图像分割；

（2）边缘检测、滤波去噪；

（3）连通区域检测，判断硬币个数；

（4）边缘图像通过霍夫变换检测圆，进而检测硬币金额。

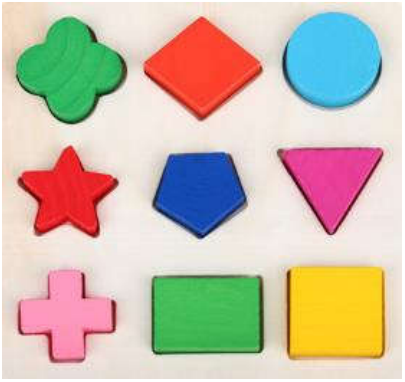
难点：

（1）硬币有重叠的处理；

（2）在拍摄距离未知、硬币种类未知的情况下，如何判断硬币的面值。

4. 形状检测认知

主要要求：实现图中九种形状检测。



建议方法：

（1）目标分割。图中目标与背景色彩及亮度差别较大，轮廓清晰，可以采用 Canny 边缘检测、形态滤波、区域填充的方式实现分割；

（2）特征参数提取，如色彩特征、边界特征、形状特征等；

（3）利用角点数、长宽比、圆度、色彩等特征区别各形状。

难点：

（1）选取合适的特征；

（2）特征参数比对的阈值选择。

三、实验考核

实验报告

四、实验报告

实验结束后，撰写实验报告，实验报告主题部分应包括：算法原理、程序流程、算法各部分主要函数代码以及功能注释、运行结果四部分，按照撰写情况打分。