

实验报告

课程名称： **数字图像处理**

学生姓名：  **uknowho**  学 号：

学　 院： **计算机与信息工程学院**

专业年级： **2014级软件工程3班**

教 师：  **辛动军**

2017年12月

**目录**

[实验一：数字图像基本操作及灰度调整 3](#_Toc500320441)

[实验二：数字图像的空间域滤波 11](#_Toc500320442)

[实验三：数字图像的频域滤波 22](#_Toc500320443)

[实验四：图像分割与边缘检测 29](#_Toc500320444)

# 实验一：数字图像基本操作及灰度调整

1. **熟悉MATLAB语言中对图像数据读取，显示等基本函数**
2. 文件读取与信息显示：

load trees;

[X,map]=imread('forest.tif');

subplot(1,2,1)

subimage(X,map);

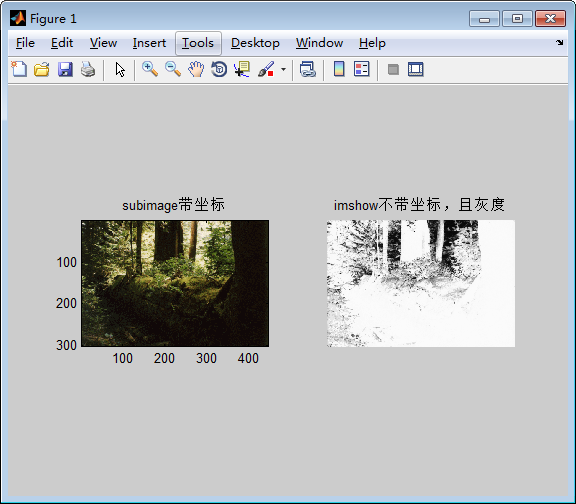
title('subimage带坐标')

I=imread('forest.tif');

subplot(1,2,2);

imshow(I);

title('imshow不带坐标，且灰度)



imfinfo('forest.tif')

ans =

Filename: 'D:\MATLAB\toolbox\images\imdemos\forest.tif'

FileModDate: '04-十二月-2000 13:57:58'

FileSize: 124888

Format: 'tif'

FormatVersion: []

Width: 447

Height: 301

BitDepth: 8

ColorType: 'indexed'

FormatSignature: [73 73 42 0]

ByteOrder: 'little-endian'

NewSubFileType: 0

BitsPerSample: 8

Compression: 'PackBits'

PhotometricInterpretation: 'RGB Palette'

StripOffsets: [17x1 double]

SamplesPerPixel: 1

RowsPerStrip: 18

StripByteCounts: [17x1 double]

XResolution: 72

YResolution: 72

ResolutionUnit: 'None'

Colormap: [256x3 double]

PlanarConfiguration: 'Chunky'

TileWidth: []

TileLength: []

TileOffsets: []

TileByteCounts: []

Orientation: 1

FillOrder: 1

GrayResponseUnit: 0.0100

MaxSampleValue: 255

MinSampleValue: 0

Thresholding: 1

Offset: 122964

ImageDescription: [1x49 char]

1. map颜色矩阵的修改

load trees;

[X,map]=imread('forest.tif');

subplot(1,2,1)

subimage(X,map);

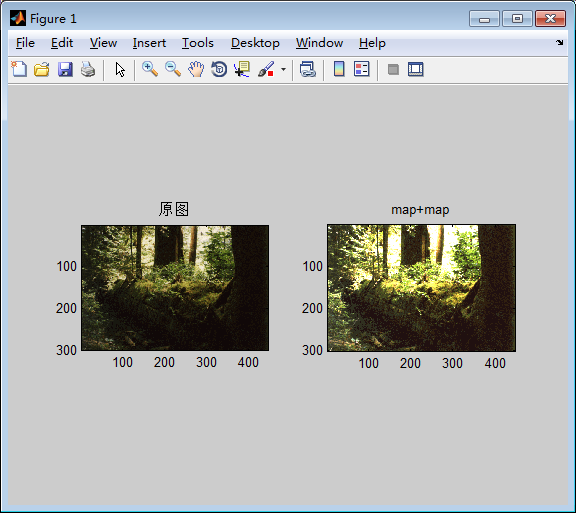
title('Ô­Í¼')

map1=map+map

subplot(1,2,2)

subimage(X,map1);

title('map+map')



1. 灰度图像的转化

rgb=imread('e:\pic.jpg');

grey=rgb2gray(rgb);

subplot(1,2,1);

subimage(rgb);

title('罗志翔：你好，色彩');

subplot(1,2,2);

subimage(grey);

title('灰色的罗志翔');



1. **图像灰度变换处理在图像增强的作用**

rgb=imread('e:\pic.jpg');

grey=rgb2gray(rgb);

subplot(2,2,1);

subimage(rgb);

title('罗志翔');

g1=imadjust(rgb,[0,1],[1,0]);

g2=imcomplement(g1);

g3=im2uint8(mat2gray(log(1+double(rgb))));

subplot(2,2,2);

subimage(g1);

title('反转的罗志翔');

subplot(2,2,3);

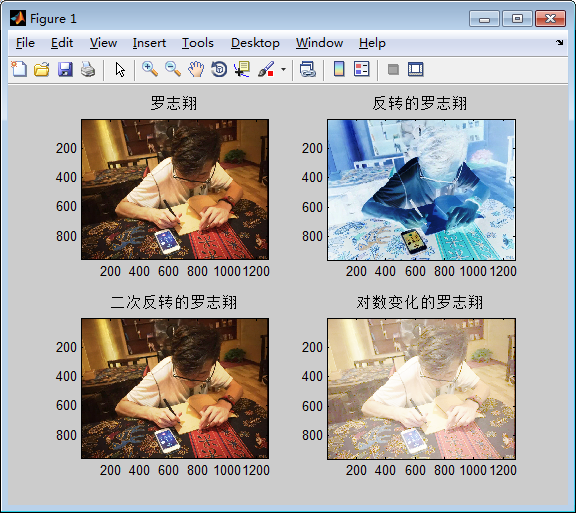
subimage(g2);

title('二次反转的罗志翔');

subplot(2,2,4);

subimage(g3);

title('对数变化的罗志翔')



1. **绘制图像灰度直方图的方法，对图像进行均衡化处理**
2. 图像灰度直方图的显示与灰度调整

灰度直方图显示:

rgb=imread('e:\pic.jpg');

grey=rgb2gray(rgb);

subplot(1,2,1);

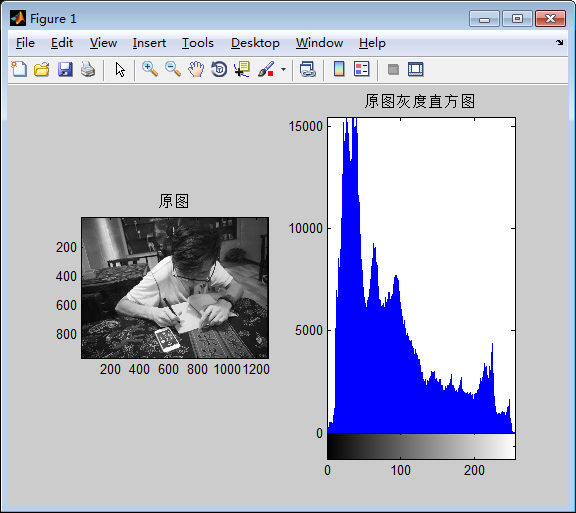
subimage(grey);

title(‘原图');

subplot(1,2,2)

imhist(grey); %绘制直方图

title('灰度直方图')



调整原图灰度范围为[0,0.5]

rgb=imread('e:\pic.jpg');

grey=rgb2gray(rgb);

J=imadjust(grey,[],[0,0.5])

subplot(1,2,1);

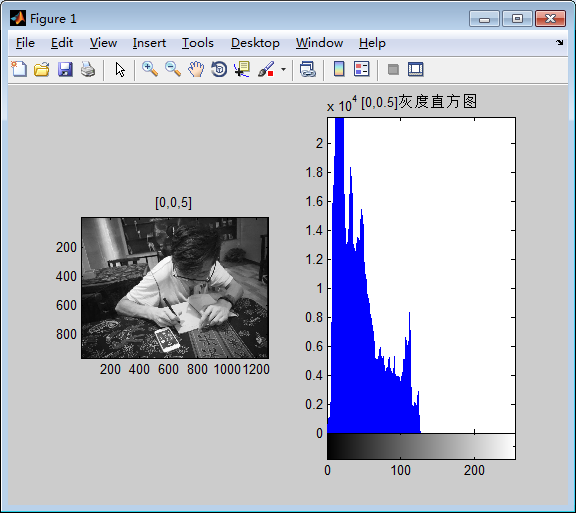
subimage(grey);

title('[0,0,5]');

subplot(1,2,2)

imhist(J); %绘制直方图

title('[0,0.5]灰度直方图');



将灰度值调整到[0,0.5]时，整个图像变暗，直方图横向压缩1倍。

1. 对B进行直方图均衡化处理，试比较与原图的异同。

rgb=imread('e:\pic.jpg');

grey=rgb2gray(rgb);

subplot(1,2,1);

subimage(grey);

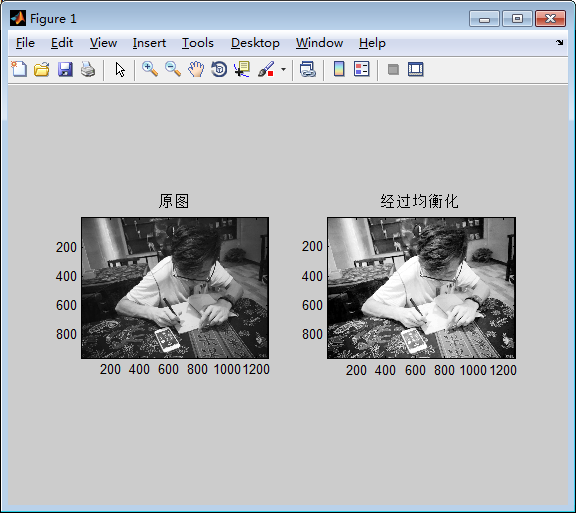
title('原图');

Q=histeq(grey);

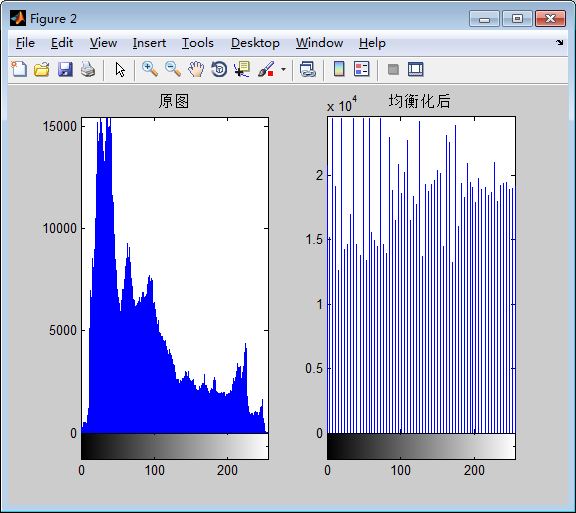
subplot(1,2,2);

subimage(Q);

title('经过均衡化')

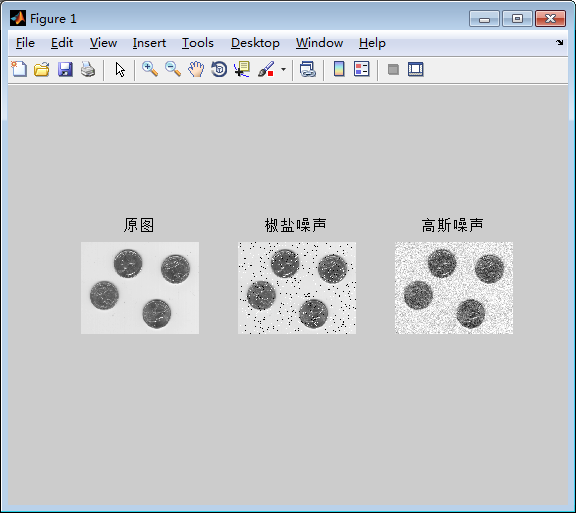


1. 对B进行如图所示的分段线形变换处理，试比较与直方图均衡化处理的异同。



# 实验二：数字图像的空间域滤波

1. **平滑空间滤波：**
2. 读出eight.tif这幅图像，给这幅图像分别加入椒盐噪声和高斯噪声后并与前一张图显示在同一图像窗口中。



1. 对加入噪声图像选用不同的平滑（低通）模板做运算，对比不同模板所形成的效果，要求在同一窗口中显示。

* 椒盐噪声+低通滤波：

I=imread('eight.tif')

J=imnoise(I,'salt & pepper',0.05);

H=fspecial('sobel');

Sobel=imfilter(J,H,'replicate');

subplot(221);

imshow(J);

title('原图');

subplot(222)

imshow(Sobel)

title('Sobel滤波');

H=fspecial('laplacian',0.4);

Lap=imfilter(J,H,'replicate');

subplot(223)

imshow(Lap)

title('Laplacian滤波');

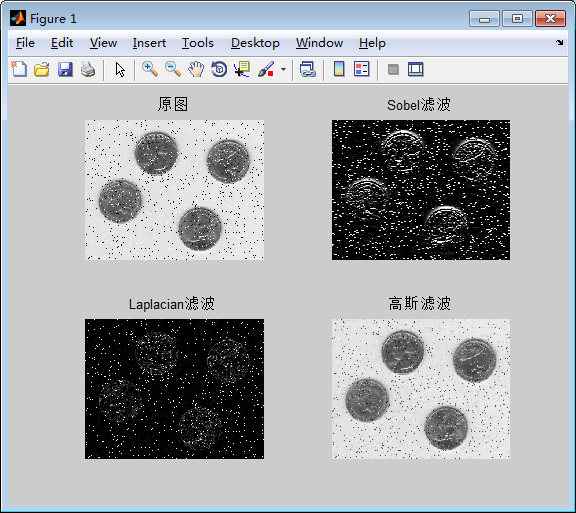
H=fspecial('gaussian',[3,3],0.5);

Gaussian=imfilter(J,H,'replicate');

subplot(224)

imshow(Gaussian)

title('高斯滤波');



* 高斯噪声+低通滤波

I=imread('eight.tif')

K=imnoise(I,'gaussian',0.01,0.01);

H=fspecial('sobel');

Sobel=imfilter(K,H,'replicate');

subplot(221);

imshow(K);

title('原图');

subplot(222)

imshow(Sobel)

title('Sobel滤波');

H=fspecial('laplacian',0.4);

Lap=imfilter(K,H,'replicate');

subplot(223)

imshow(Lap)

title('Laplacian滤波');

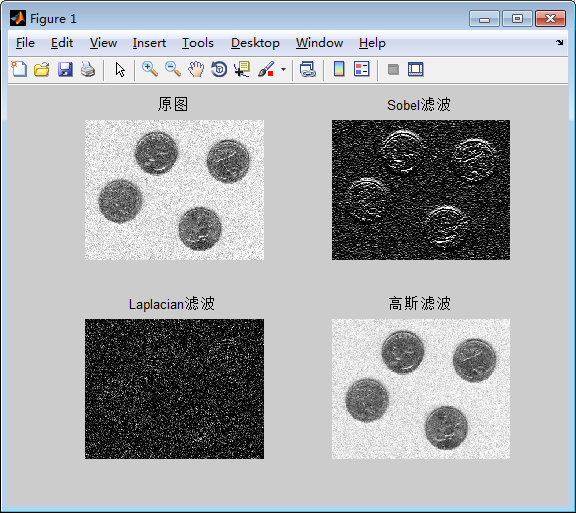
H=fspecial('gaussian',[3,3],0.5);

Gaussian=imfilter(K,H,'replicate');

subplot(224)

imshow(Gaussian)

title('高斯滤波');



1. 使用函数imfilter时，分别采用不同的填充方法（或边界选项，如零填充、’replicate’、’symmetric’、’circular’）进行低通滤波，显示处理后的图像。

originalRGB = imread('peppers.png');

h = fspecial('motion', 50, 45); %motion blurred

filteredRGB = imfilter(originalRGB, h);

subplot(321)

imshow(originalRGB)

title('原图');

subplot(322)

imshow(filteredRGB)

title('Motion Blurred')

boundaryReplicateRGB = imfilter(originalRGB, h, 'replicate');

subplot(323)

imshow(boundaryReplicateRGB);

title('Replicate')

boundarysymmetricRGB = imfilter(originalRGB, h, 'symmetric');

subplot(324)

imshow(boundarysymmetricRGB);

title('Symmetric')

boundarycircularRGB = imfilter(originalRGB, h, 'circular');

subplot(325)

imshow(boundarycircularRGB)

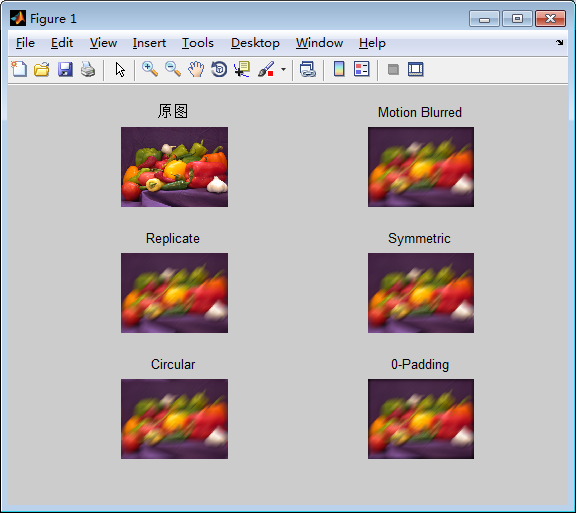
title('Circular')

boundary0RGB = imfilter(originalRGB, h, 0);

subplot(326)

imshow(boundary0RGB)

title('0-Padding')



1. 运用for循环，将加有椒盐噪声的图像进行10次，20次均值滤波，查看其特点,显示均值处理后的图像。

I=imread('eight.tif')

J=imnoise(I,'salt & pepper',0.05);

h=fspecial('average')

J1=imfilter(J,h)

for i=1:10

J1=imfilter(J,h)

end

for i=1:20

J2=imfilter(J,h)

end

subplot(131);

imshow(J);

title('椒盐噪声');

subplot(132);

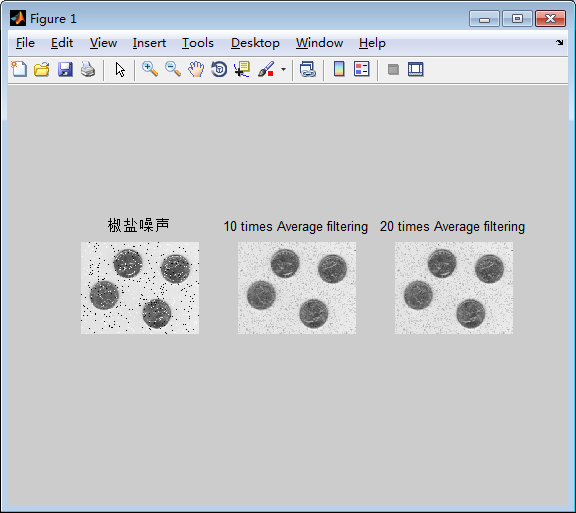
imshow(J1);

title('10 times Average filtering');

subplot(133);

imshow(J2);

title('20 times Average filtering');



1. 对加入椒盐噪声的图像分别采用均值滤波法，和中值滤波法对有噪声的图像做处理，要求在同一窗口中显示结果。

I=imread('eight.tif')

J=imnoise(I,'salt & pepper',0.05);

h=fspecial('average'); %Averaging Filtering

J1=imfilter(J,h);

J2=medfilt2(J); %Median Filtering

subplot(131);

imshow(J);

title('椒盐噪声');

subplot(132);

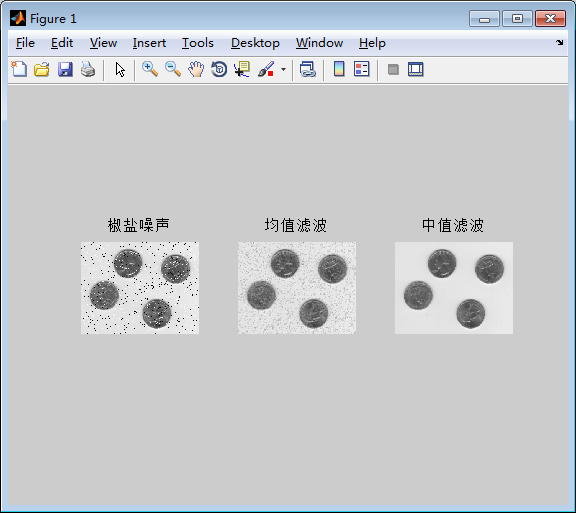
imshow(J1);

title('均值滤波');

subplot(133);

imshow(J2);

title('中值滤波');



1. 设计平滑空间滤波器，并将其对噪声图像进行处理，显示处理后的图像。

I=imread('eight.tif')

J=imnoise(I,'salt & pepper',0.05);

domain=[0 0 8 0 0;

0 0 8 0 0;

8 8 8 8 8;

0 0 8 0 0;

0 0 8 0 0];

K1= ordfilt2(J,5,domain);

subplot(121)

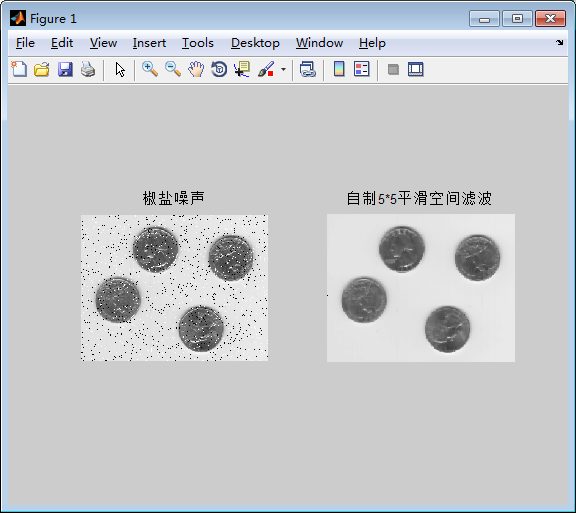
imshow(J);

title('椒盐噪声');

subplot(122)

imshow(K1)

title('自制5\*5平滑空间滤波')



1. **锐化空间滤波**
2. 采用3×3的拉普拉斯算子w = [ 1, 1, 1; 1 – 8 1; 1, 1, 1]滤波

I=imread('moon.tif');

T=double(I);

subplot(1,2,1)

imshow(T,[])

title('Original Image');

w =[1,1,1;

1,-8,1;

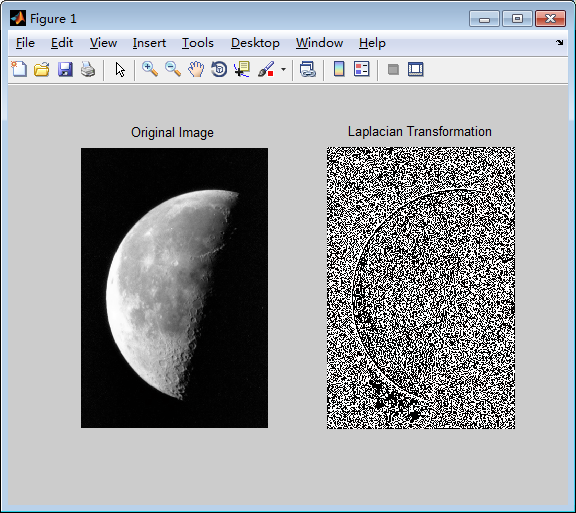
1,1,1];

K=conv2(T,w,'same');

subplot(1,2,2)

imshow(K)

title('Laplacian Transformation');



1. 编写函数w = genlaplacian(n)，自动产生任一奇数尺寸n的拉普拉斯算子，如5×5的拉普拉斯算子：

w = [ 1 1 1 1 1

1 1 1 1 1

1 1 -24 1 1

1 1 1 1 1

1. 1 1 1 1]
2. 采用不同的梯度算子对blurry\_moon.tif进行锐化滤波，并比较其效果

imshow([I,map])

title('Ori')

I=double(I);

[Gx,Gy]=gradient(I); % gradient calculation

G=sqrt(Gx.\*Gx+Gy.\*Gy); % matrix

J1=G; % gradient1

subplot(232)

imshow(J1,map);

title('gradient1')

J2=I; % gradient2

K=find(G>=7);

J2(K)=G(K);

subplot(233)

imshow(J2,map);

title('gradient2')

J3=I; % gradient3

K=find(G>=7);

J3(K)=255;

subplot(234)

imshow(J3,map);

title('gradient3')

J4=I; % gradient4

K=find(G<=7);

J4(K)=255;

subplot(235)

imshow(J4,map);

title('gradient4')

J5=I; % gradient5

K=find(G<=7);

J5(K)=0;

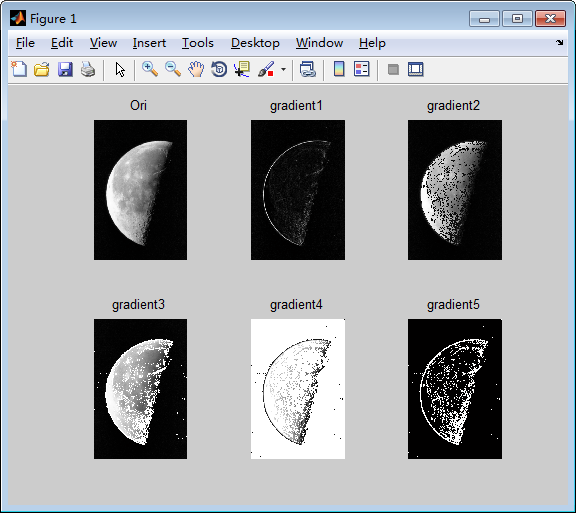
Q=find(G>=7);

J5(Q)=255;

subplot(236)

imshow(J5,map);

title('gradient5')



# 实验三：数字图像的频域滤波

1. **傅立叶变换**
2. 读出woman.tif这幅图像，对其进行快速傅立叶变换，分别显示其幅度图像和相位图像。

F=imread('e:\cameraman.tif');

subplot(221)

imshow(F)

title('ori')

F1=fft2(F);

F2=log(1+abs(F1)); %amplitude spectrum

subplot(222)

imshow(F2,[])

title('amplitude spectrum')

F3=fftshift(F1);

subplot(223)

imshow(log(1+abs(F3)),[]);

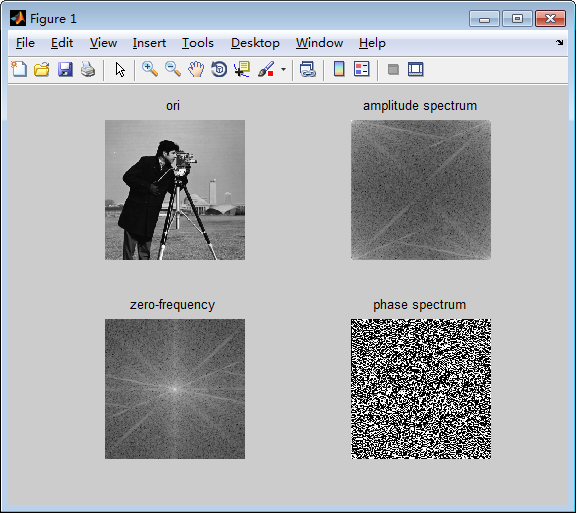
title('zero-frequency')

F4=angle(F1); %phase spectrum

subplot(224)

imshow(F4)

title('phase spectrum')



1. 仅对相位部分进行傅立叶反变换后查看结果图像。

F=imread('e:\cameraman.tif');

subplot(221)

imshow(F)

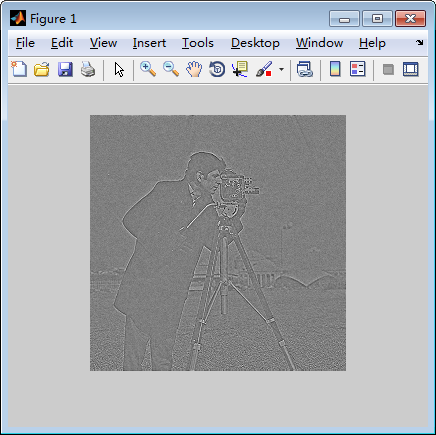
title('ori')

F1=fft2(F);

i=sqrt(-1);

f2=ifft2(exp(i\*angle(F1)));

imshow(real(f2),[]);

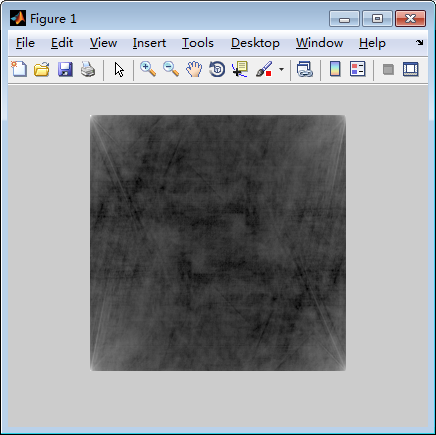


1. 仅对幅度部分进行傅立叶反变换后查看结果图像。

F=imread('e:\cameraman.tif');

f1=ifft2(abs(F1));

imshow(log(1+abs(f1)),[]);



1. 将图像的傅立叶变换*F*置为其共轭后进行反变换，比较新生成图像与原始图像的差异。

F=imread('e:\cameraman.tif');

F1=fft2(F);

F2=log(1+abs(F1)); %amplitude spectrum

F3=fftshift(F1);

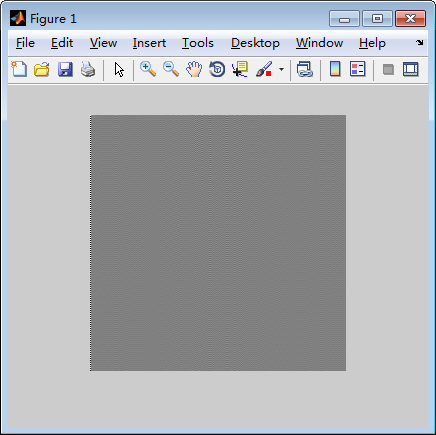
F4=angle(F1); %phase spectrum

F5=-F4

F6= double(F3\*exp(F4)); %the complex conjugate of the fourier transform

F7=ifft2(F6); %inverse fourier transform

imshow(real(F7),[]);



1. **平滑频域滤波**
2. 设计理想低通滤波器、巴特沃斯低通滤波器和高斯低通滤波器

**理想低通滤波器**

I=imread('e:\cameraman.tif');

f=double(I); % chage into double as MATLAB doesn’t suppor calculation

% of image in unsigned int type

g=fft2(f); % fourier transform

g=fftshift(g); % zero-frequency area centralized

[M,N]=size(g);

d0=100; %cutoff frequency

m=fix(M/2); n=fix(N/2);

for i=1:M

for j=1:N

d=sqrt((i-m)^2+(j-n)^2);

if(d<=d0)

h=1;

else h=0;

end

result(i,j)=h\*g(i,j);

end

end

result=ifftshift(result);

J1=ifft2(result);

J2=uint8(real(J1));

imshow(J2)



**巴特沃斯低通滤波器（二阶）**

I=imread('cameraman.tif');

f=double(I);

g=fft2(f);

g=fftshift(g);

[M,N]=size(g);

nn=2; % 2-grade Butterworth lowpss filter

d0=100;

m=fix(M/2); n=fix(N/2);

for i=1:M

for j=1:N

d=sqrt((i-m)^2+(j-n)^2);

h=1/(1+0.414\*(d/d0)^(2\*nn)); % filter transform function

%h=1./(1+(d./d0).^(2\*n))

%h=exp(-(d.^2)./(2\*(d0^2)));

result(i,j)=h\*g(i,j);

end

end

result=ifftshift(result);

J1=ifft2(result);

J2=uint8(real(J1));

imshow(J2);



**高斯低通滤波**

I=imread('cameraman.tif');

f=double(I);

g=fft2(f);

g=fftshift(g);

[M,N]=size(g);

d0=100;

m=fix(M/2); n=fix(N/2);

for i=1:M

for j=1:N

d=sqrt((i-m)^2+(j-n)^2);

h=exp(-(d.^2)./(2\*(d0^2))); % gaussian filter transform

result(i,j)=h\*g(i,j);

end

end

result=ifftshift(result);

J1=ifft2(result);

J2=uint8(real(J1));



# 实验四：图像分割与边缘检测

1. 分别用Roberts,Sobel和拉普拉斯高斯算子对图像进行边缘检测。比较三种算子处理的不同之处；

I=imread('eight.tif');

imshow(I)

BW1=edge(I,'roberts');

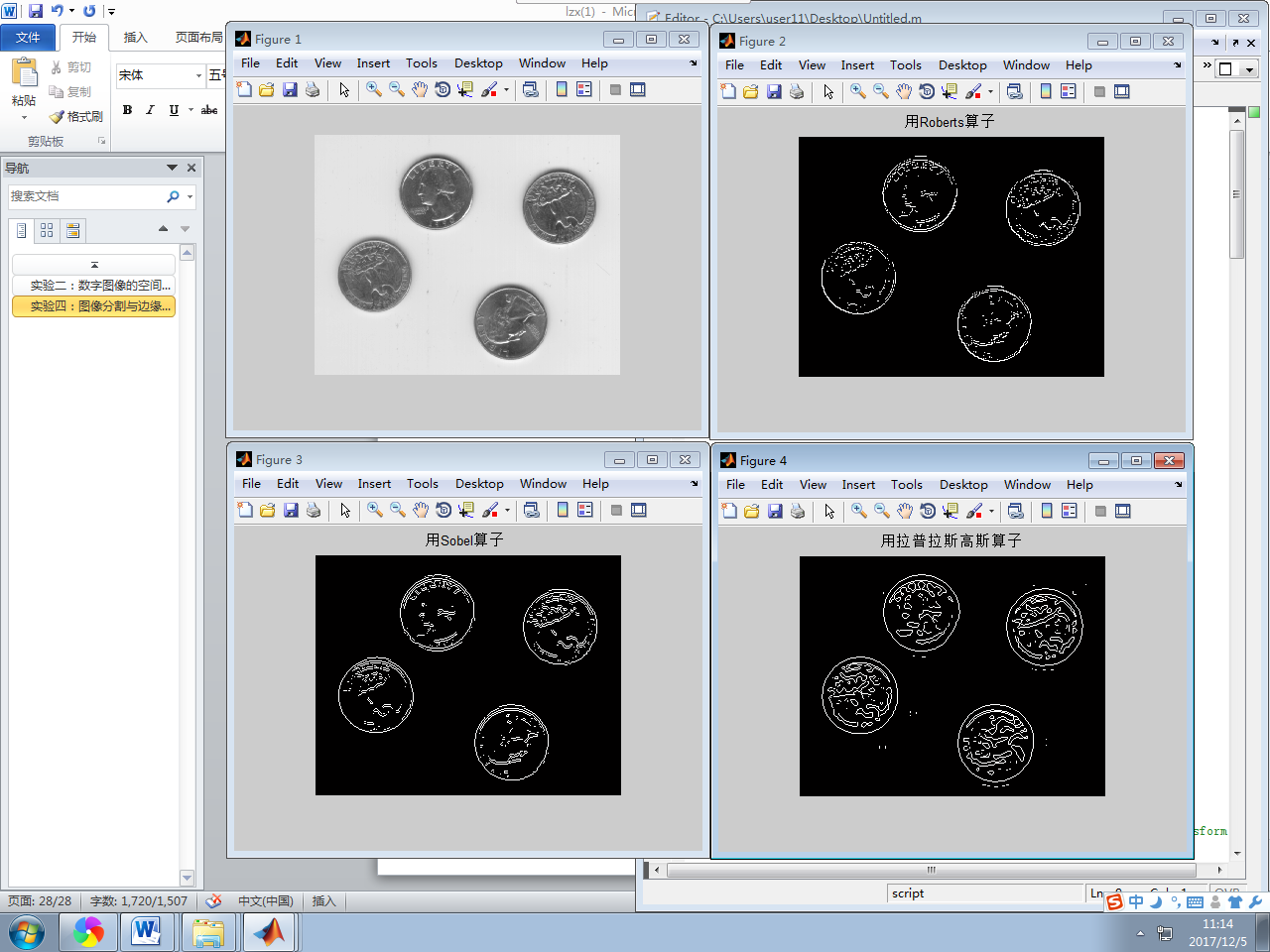
figure ,imshow(BW1),title('用Roberts算子')

BW2=edge(I,'sobel');

figure,imshow(BW2),title('用Sobel算子 ')

BW3=edge(I,'log');

figure,imshow(BW3),title('用拉普拉斯高斯算子')



1. 设计一个检测图3-2中边缘的程序，要求结果类似图3-3，并附原理说明。

利用双峰法

i=imread('e:\cameraman.tif');

subplot(1,2,1);

imhist(i);

title('原始图像直方图');

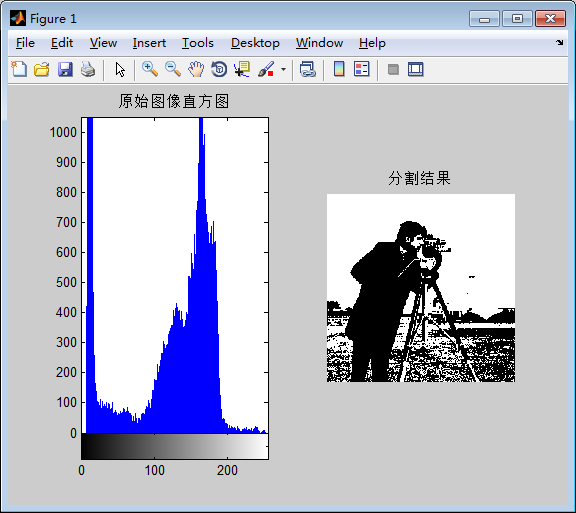
thread=130/255;

subplot(1,2,2);

i3=im2bw(i,thread);

imshow(i3);

title('分割结果');



1. 任选一种阈值法进行图像分割.

选用双峰法对图像进行分割：

i=imread('e:\blood.bmp');

subplot(1,2,1);

imhist(i);

title('原始图像直方图');

thread=50/255;

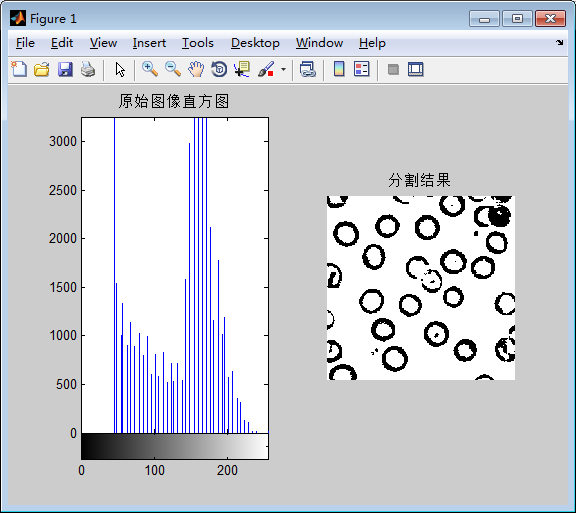
subplot(1,2,2);

i3=im2bw(i,thread);

imshow(i3);

title('分割结果');





1. 彩色分割。在RGB空间中，可以用一个球体来确定某种颜色的聚类，从而提取图像中该聚类内的颜色区域。分割出给定图片中的任意目标。

clear all

RGB = imread('mandrill.bmp');

R = double(RGB(:,:,1));

G = double(RGB(:,:,2));

B = double(RGB(:,:,3));

figure(1);

imshow(RGB);title('原始图像','fontname','宋体');

%从图像中选择感兴趣颜色的一个像素

[y,x] = ginput(1);

x = uint16(x);

y = uint16(y);

%求选择像素3\*3邻域内各分量均值

Cr = 0;

Cg = 0;

Cb = 0;

for nx = x-1:x+1

for ny = y-1:y+1

Cr = Cr + R(nx,ny);

Cg = Cg + G(nx,ny);

Cb = Cb + B(nx,ny);

end

end

Cr = Cr/9;

Cg = Cg/9;

Cb = Cb/9;

[m n ] = size(R);

%颜色分层 设置颜色聚类球体半径为30

for i = 1:m

for j = 1:n

if ( ( ( R(i,j) - Cr)^2 + (G(i,j) - Cg)^2+(B(i,j) - Cb)^2 ) > 30^2)

R(i,j) = 255;

G(i,j) = 255;

B(i,j) = 255;

end

end

end

R = uint8(R);

G = uint8(G);

B = uint8(B);

RGBnew = cat(3,R,G,B) ;

figure(2);

imshow(RGBnew);

title('彩色分割结果','fontname','宋体');

