東北大學

生物医学工程专业

实验报告

实验课程： 数字图像处理

班级： 1502 **姓名：** 尚麟静

学号： 20155467 **同组人：**

指导教师： 孙航

实验成绩（教师签字）：

中荷学院教学实验中心制

实验一

一、插值和采样

1 （a）读入图像 head.jpg 并显示。

（b）计算图像维度。

（c）此图像大小为 40cm*40cm，计算图像的采样距离。

（d）逻辑坐标（图像坐标）为（22, 54）、（126, 241）的点，其空间坐标是多少？

（e）求空间坐标为（14.2188, 5.3125）、（21.4063,34.5313）处的像素值。

（f）使用最邻近插值法，求在逻辑坐标（65.2, 193.7）和空间坐标（22.58, 7.24）处的像素值。

（g）使用双线性插值法，求逻辑坐标为（1.5,1.5）处的像素值。

```
2 -    img=imread('D:\宇宙第一ADC\____学习\数字图像处理\试验1\试验1\head.jpg');
3 -    s=size(img);
4 -    fprintf('图片维度为:');disp(s);
5 -    d=40/256;
6 -    fprintf('采样距离为:%f cm\n',d);
7 -    t1=[22,54]*d;
8 -    t2=[126,241]*d;
9 -    fprintf('逻辑坐标（图像坐标）为（22, 54）、（126, 241）的点，其空间坐标是');disp(t1);disp(t2);
10
11     %-------------(e)--------------%
12 -    x1=round(14.2188/d);
13 -    y1=round(5.3125/d);
14 -    x2=round(21.4063/d);
15 -    y2=round(34.5313/d);
16 -    fprintf('求空间坐标为（14.2188, 5.3125）、（21.4063,34.5313）处的像素值为%d和%d',img(x1,y1),img(x2,y2));
17
```

```
Command Window
  逻辑坐标（图像坐标）为（22, 54）、（126, 241）的点，其空间坐标是     3.4375     8.4375

     19.6875    37.6563

  求空间坐标为（14.2188, 5.3125）、（21.4063,34.5313）处的像素值为114和127最近插值法，在逻辑坐标（65.2, 193.7）的像素值为95
  最近插值法，在空间坐标（22.58, 7.24）处的像素值为99
  使用双线性插值法，逻辑坐标为（1.5,1.5）处的像素值为4
fx >>
```

MATLAB 代码：

```
%----------<一、插值和采样>--------%
img=imread('D:\宇宙第一 ADC\____学习\数字图像处理\试验 1\试验 1\head.jpg');
s=size(img);
fprintf('图片维度为:');disp(s);
```

```
d=40/256;
fprintf('采样距离为:%f cm\n',d);
t1=[22,54]*d;
t2=[126,241]*d;
fprintf('逻辑坐标（图像坐标）为（22, 54）、（126, 241）的点，其空间坐标是');disp(t1);disp(t2);

%-------------(e)--------------%
x1=round(14.2188/d);
y1=round(5.3125/d);
x2=round(21.4063/d);
y2=round(34.5313/d);
fprintf('求空间坐标为（14.2188, 5.3125）、（21.4063,34.5313）处的像素值为%d 和%d',img(x1,y1),img(x2,y2));

%-----------(f)------------%
%因为是最近插值法，在逻辑坐标（65.2, 193.7）的像素值为（65，193）处的像素值%
fprintf('最近插值法，在逻辑坐标（65.2, 193.7）的像素值为%d\n',img(65,193));
%同理可得空间坐标（22.58, 7.24）处的像素值为(22.58/d,7.24/d)的最近插值法的像素值%
temp1=round(22.58/d);
temp2=round(7.24/d);
fprintf('最近插值法，在空间坐标（22.58, 7.24）处的像素值为%d\n',img(temp1,temp2));

%-----------(g)------------%
C3=img(1,1);
C4=img(1,2);
C5=img(2,1);
C6=img(2,2);
C7=0.5*C3+0.5*C4;
C8=0.5*C5+0.5*C6;
C9=0.5*C7+0.5*C8;
fprintf('使用双线性插值法，逻辑坐标为（1.5,1.5）处的像素值为%d\n',C9);
```

输出结果：图片维度为:　　2

采样距离为:0.156250 cm
逻辑坐标（图像坐标）为（22, 54）、（126, 241）的点，其空间坐标是　　3.4375　　8.4375

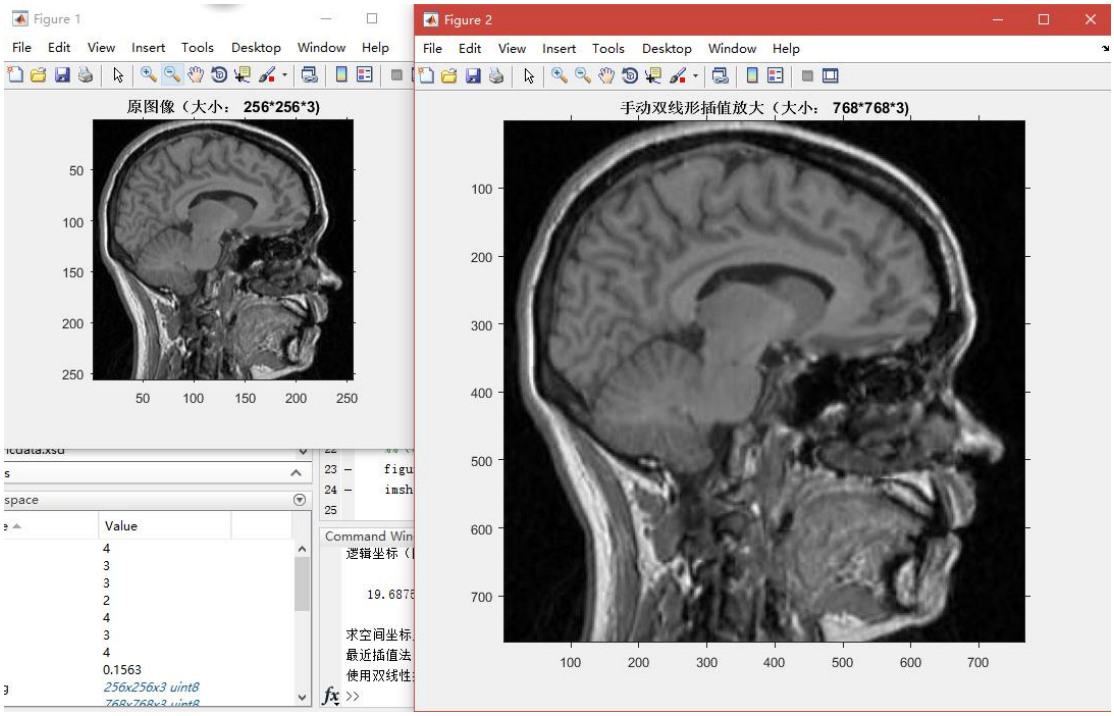
　　19.6875　　37.6563

求空间坐标为（14.2188, 5.3125）、（21.4063,34.5313）处的像素值为 114 和 127 最邻近插值法，在逻辑坐标（65.2, 193.7）的像素值为 95
最邻近插值法，在空间坐标（22.58, 7.24）处的像素值为 99
使用双线性插值法，逻辑坐标为（1.5,1.5）处的像素值为 4

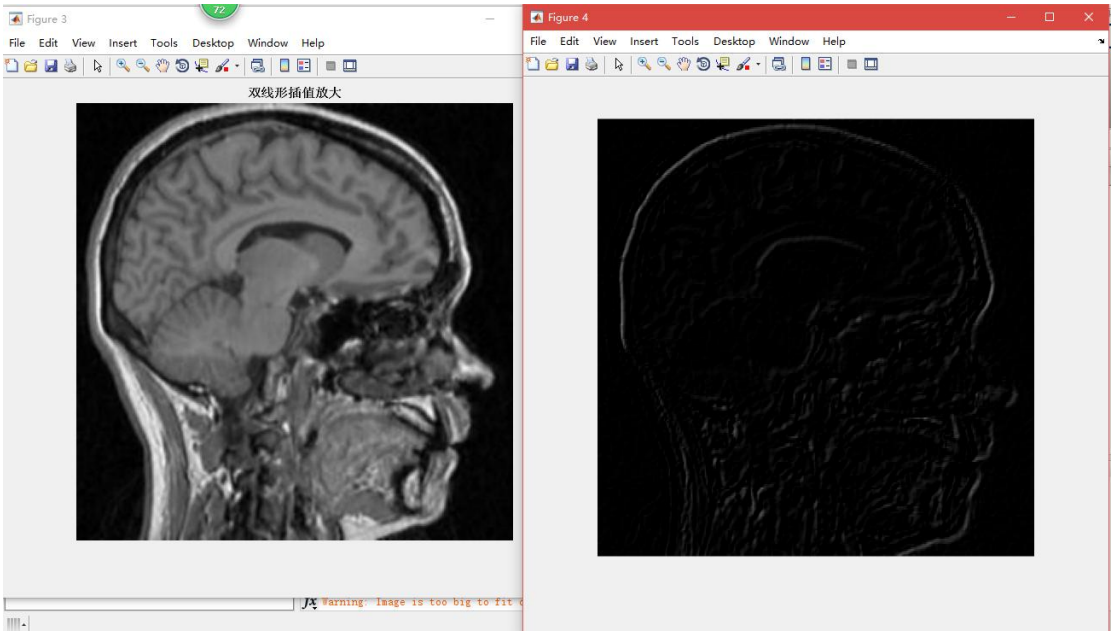
二、仿射变换

1 图像缩放

（a）编写程序代码，使用双线性差值方法把图像 head.jpg 放大 3 倍，显示原图像和放大后的图像。
（b）调用 Matlab 自带函数（imresize,bilinear）把图像 head.jpg 放大 3 倍，显示此图像。
（c）把以上两种方式得到的图像做差，显示图像结果。
（a）

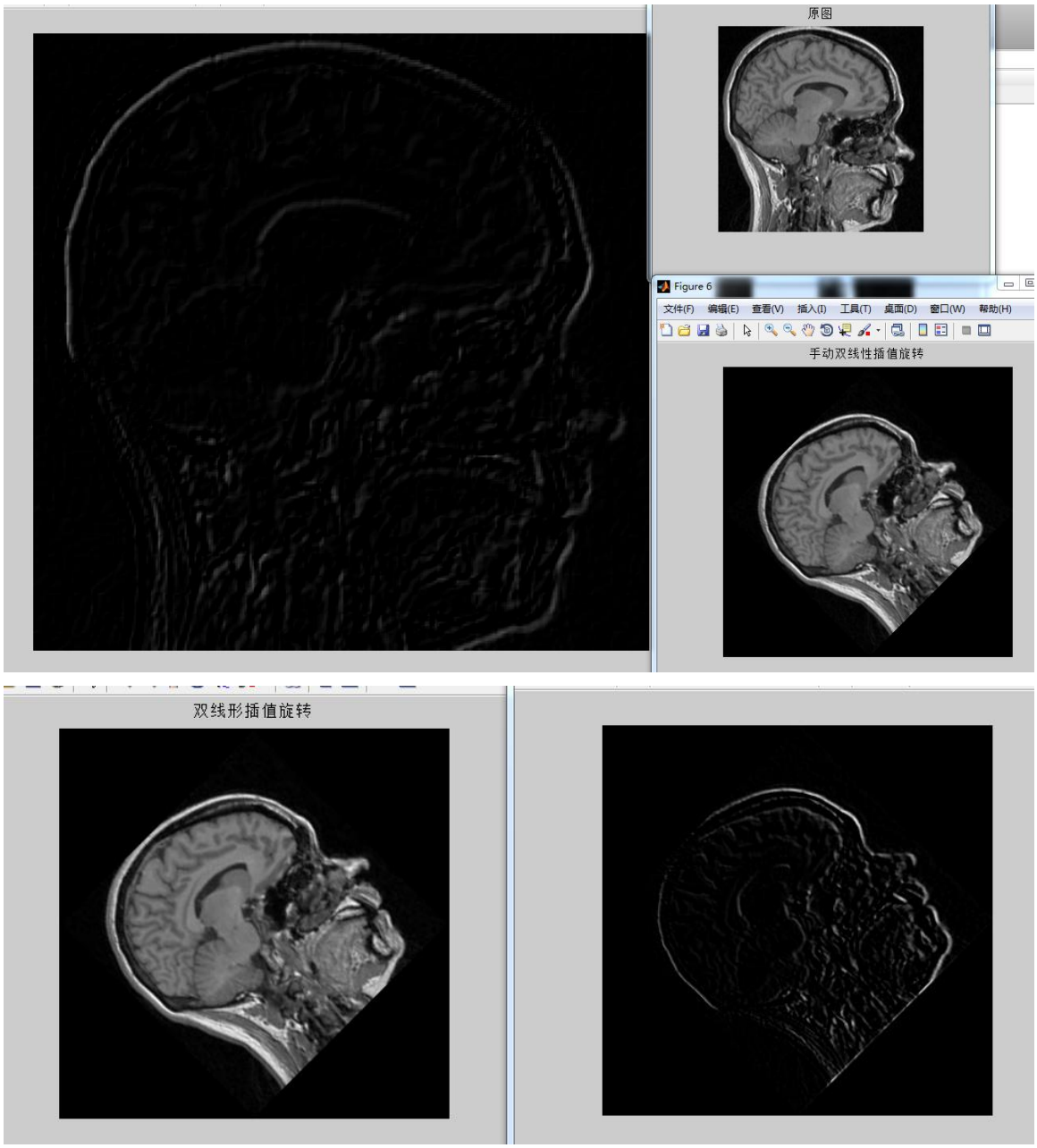


（b）（c）



2 图像旋转

（a）编写程序代码，使用双线性差值方法把图像 head.jpg 旋转 45 度，显示原图像和旋转后的图像。
（b）调用 Matlab 自带函数（imrotate,bilinear）把图像 head.jpg 旋转 45 度，显示此图像。
（c）把以上两种方式得到的图像做差，显示图像结果。

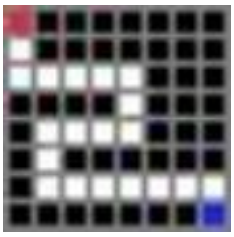
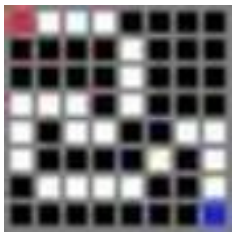
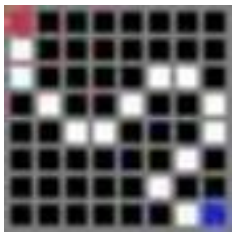
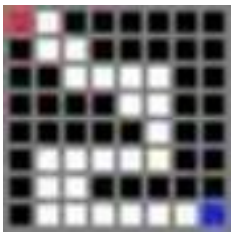


实验二

一 像素间的连通性

计算以下四幅图像中从红色位置经白色的前景区域到蓝色位置的最短路径。

提示：不是每个白色像素必须经过，而是路径中必须都是白色像素。

（a） （b） （c） （d）

连通性	a	b	c	d
4-path	20	/	/	20
8-path	14	20	14	13
m-path	20	23	14	20

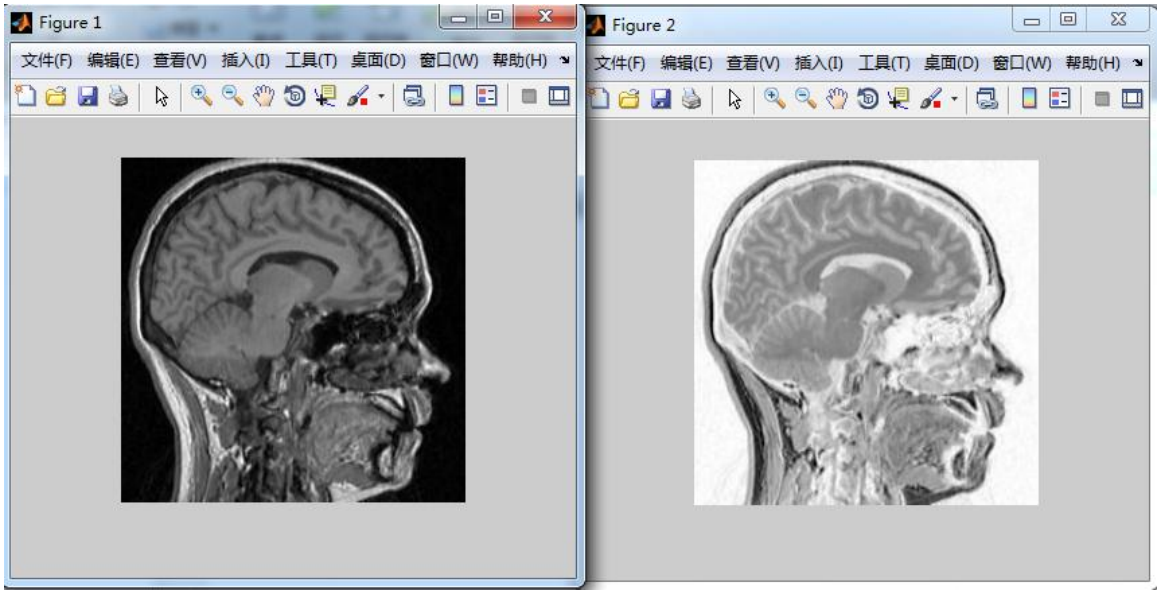
二 图像的灰度变换

1 反转变换

（a） 读入图像 head.jpg，显示原图像和反转后的图像。

MATLAB代码：

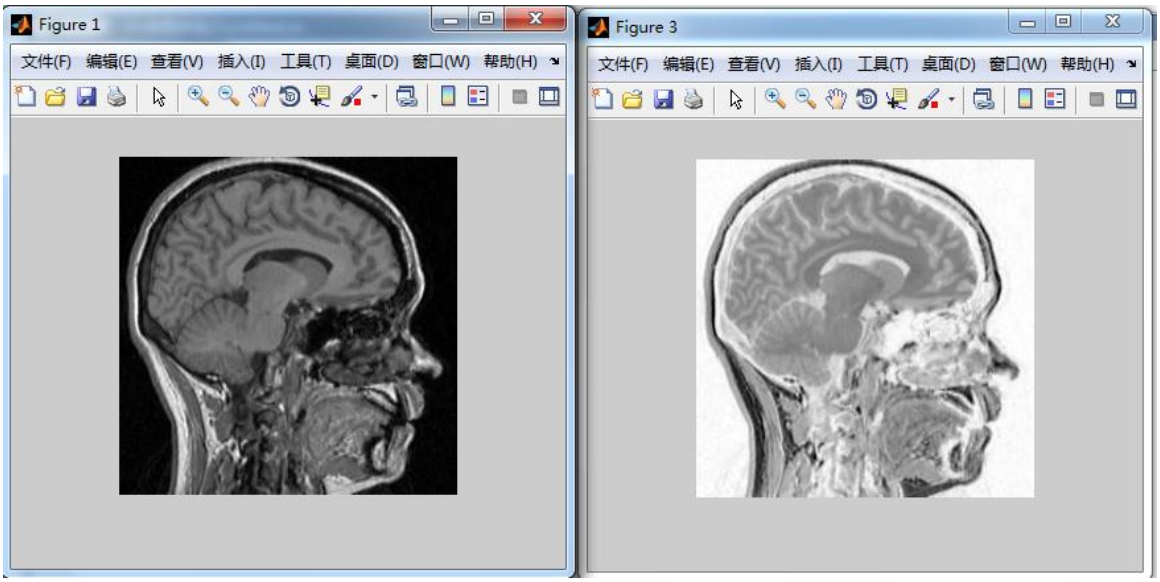
```
I=imread('C:\Users\sys\Desktop\head.jpg ');
imshow(I);
Id=double(I);
Id=-Id+255;
H=uint8(Id);
figure,imshow(H);
```



（b） 使用函数 imadjust 实现图像反转。

MATLAB 代码：

```
I=imread('C:\Users\sys\Desktop\head.jpg ');
imshow(I);
H=imadjust(I,[0,1],[1,0]);
figure,imshow(H);
```



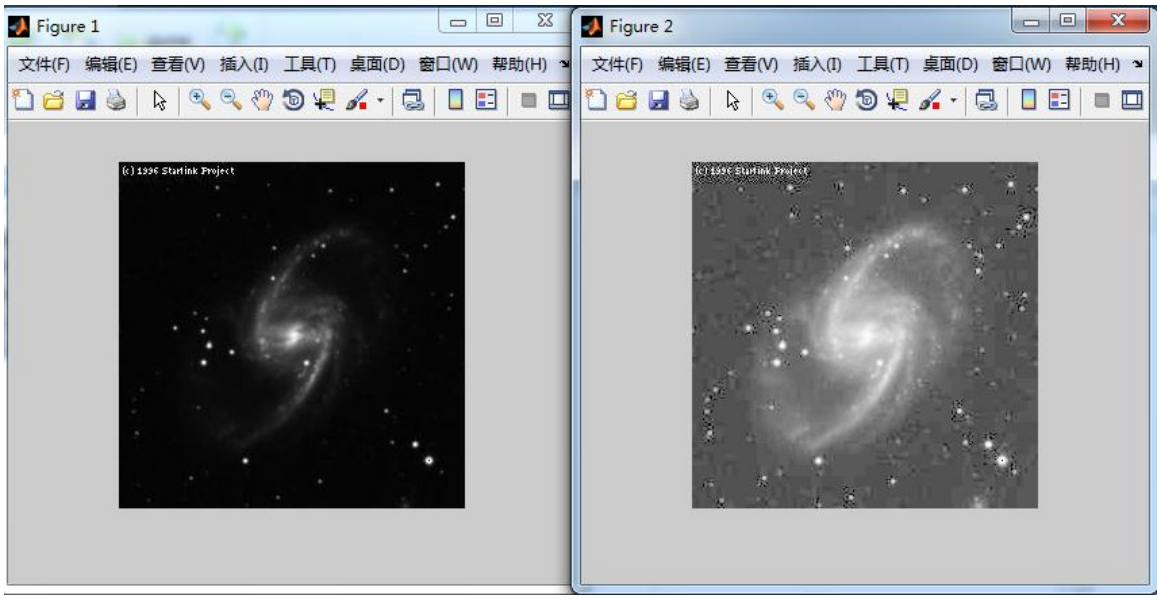
2 Log 变换

读入图像 cloud.jpg，显示原图像和 Log 变换后的图像。

MATLAB 代码：

```
I=imread('C:\Users\sys\Desktop\cloud.jpg ');
Id=double(I);
```

```
Il=im2uint8(mat2gray(log(Id+1)));
figure,imshow(I);
figure,imshow(Il,[]);
```



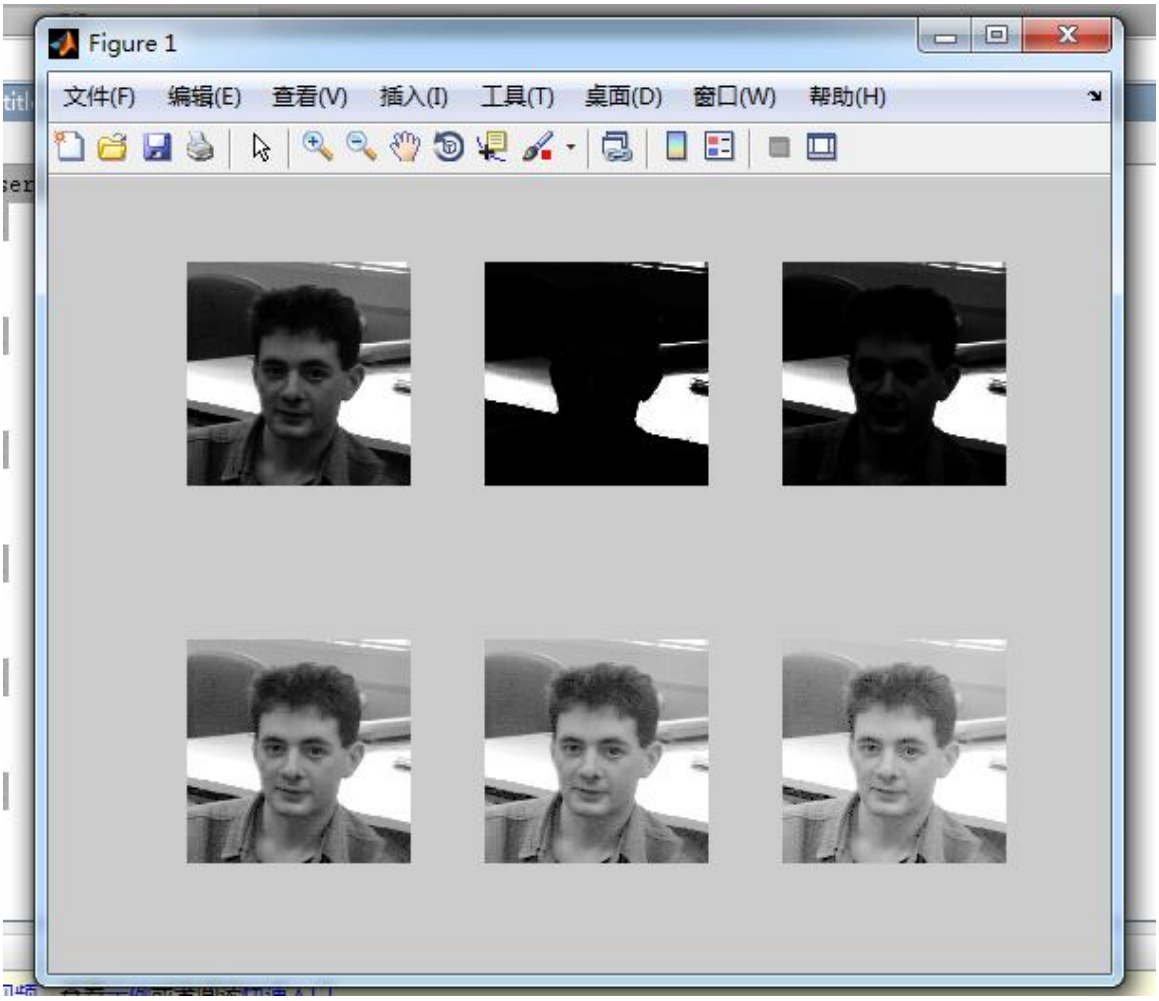
3 Gamma 变换

（a） 读入图像 man.jpg，显示原图像和 Gamma 变换后的图像，Gamma 的值分别为 5，2.3，0.5，0.4，0.3。

MATLAB 代码

```
I=imread('C:\Users\sys\Desktop\man.jpg ');
subplot(2,3,1);
imshow(I);
I=double(I);
subplot(2,3,2);
I1=I.^(5);              %5
imshow(I1,[]);
subplot(2,3,3);
I2=I.^(2.3);          %2.3
imshow(I2,[]);
subplot(2,3,4);
I3=I.^(0.5);          %0.5
imshow(I3,[]);
subplot(2,3,5);
I4=I.^(0.4);          %0.4
imshow(I4,[]);
subplot(2,3,6);
I5=I.^(0.3);          %0.3
```

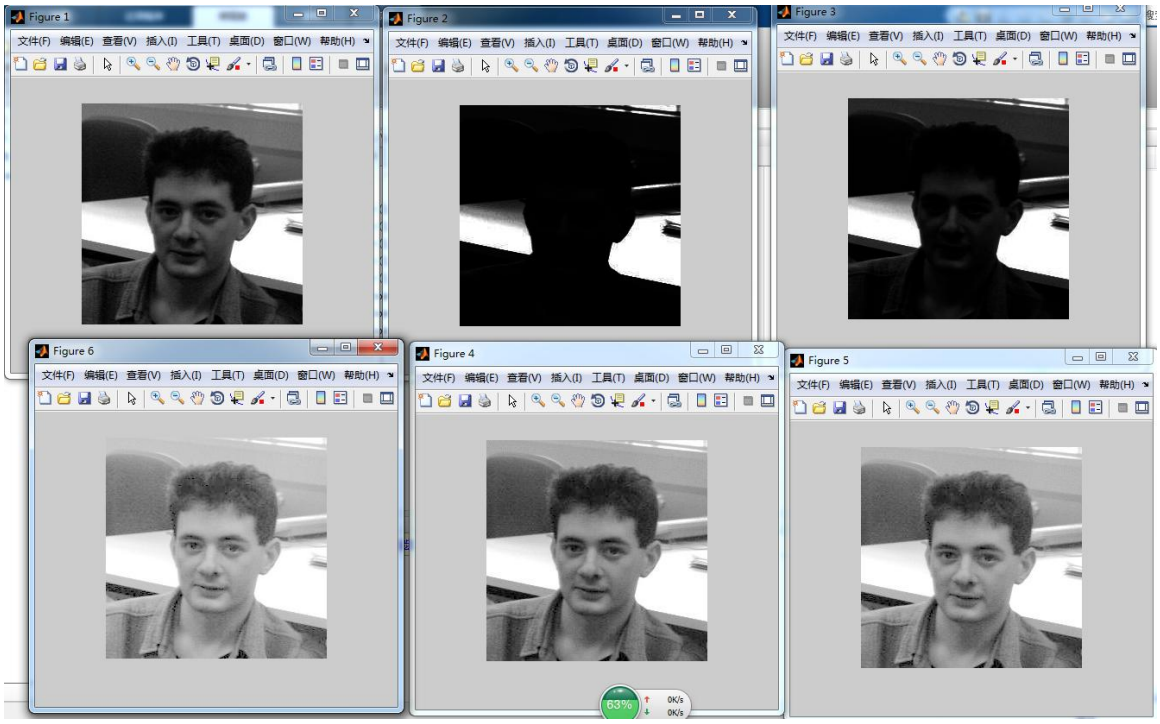
imshow(I5,[]);



（b） 使用函数 imadjust 实现图像的 Gamma 变换。

MATLAB 代码：

```
I=imread('C:\Users\sys\Desktop\man.jpg ');
imshow(I);
I1=imadjust(I,[],[],5);
I2=imadjust(I,[],[],2.3);
I3=imadjust(I,[],[],0.5);
I4=imadjust(I,[],[],0.4);
I5=imadjust(I,[],[],0.3);
figure,imshow(I1);
figure,imshow(I2);
figure,imshow(I3);
figure,imshow(I4);
figure,imshow(I5);
```

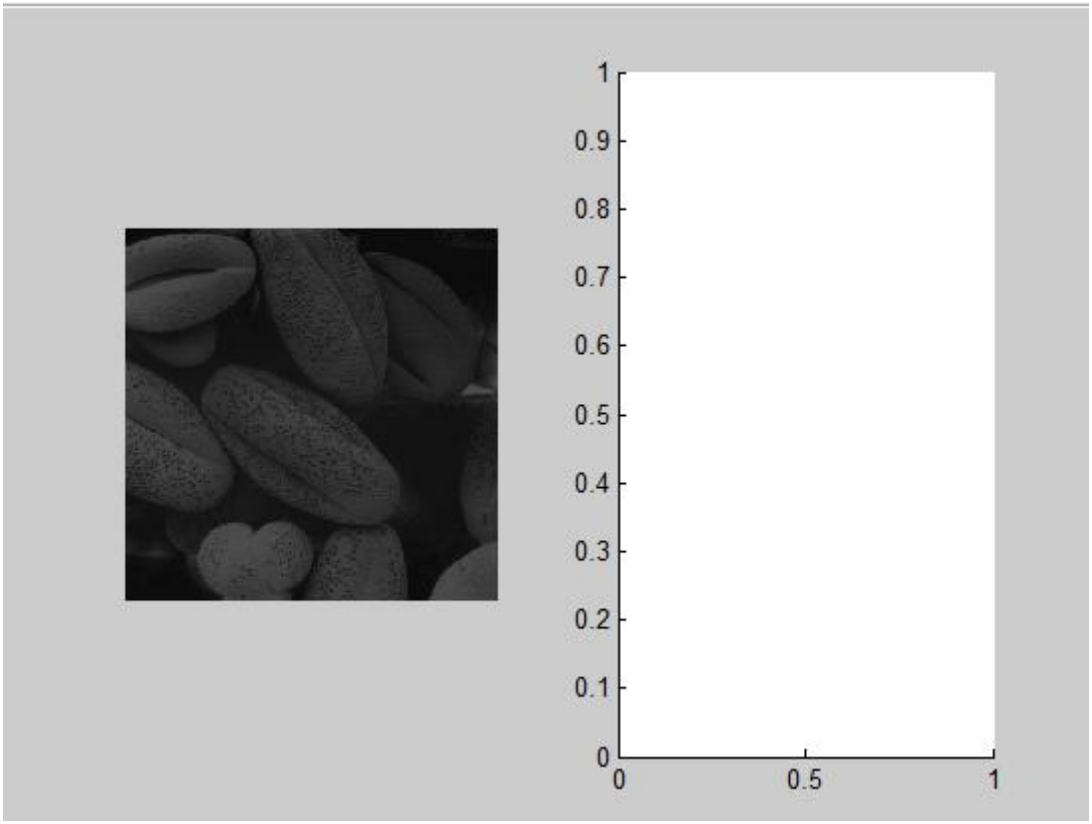


4 分段线性变换

读入图像 pollen.jpg，显示原图像和映射到整个灰度级后的图像。

MATLAB 代码：

```
I=imread('C:\Users\sys\Desktop\pollen.tif ');
subplot(1,2,1);
imshow(p);
subplot(1,2,2);
h=max(max(I));
l=min(min(I));
h1=double(h)/255;
l1=double(l)/255;
I1=imadjust(I,[11,h1],[0,1]);
imshow(I1);
```

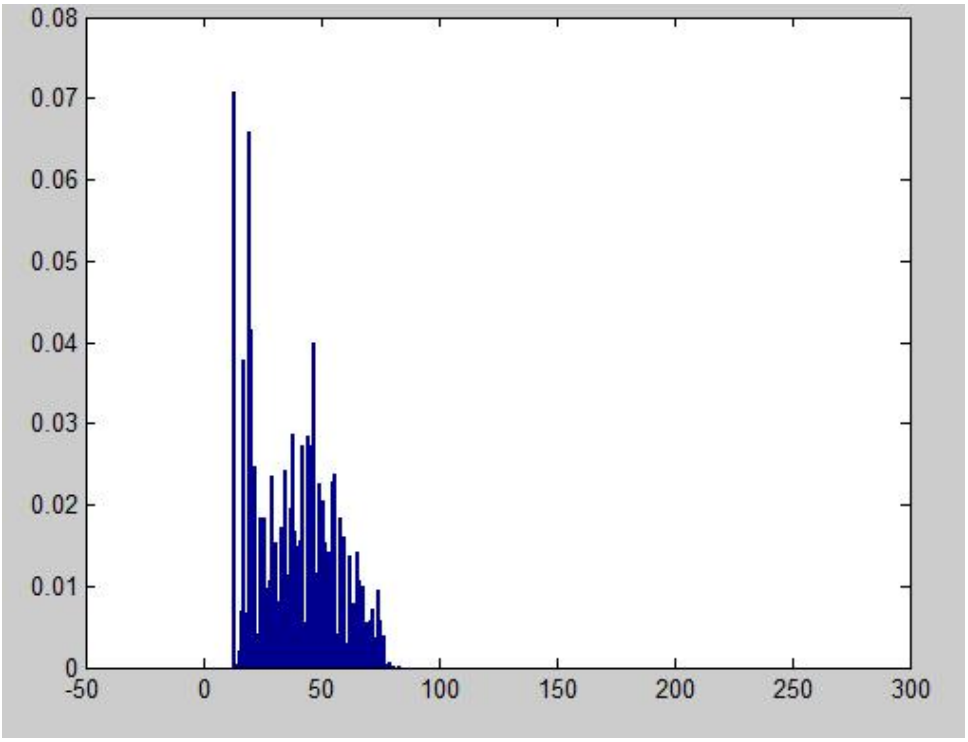


实验三

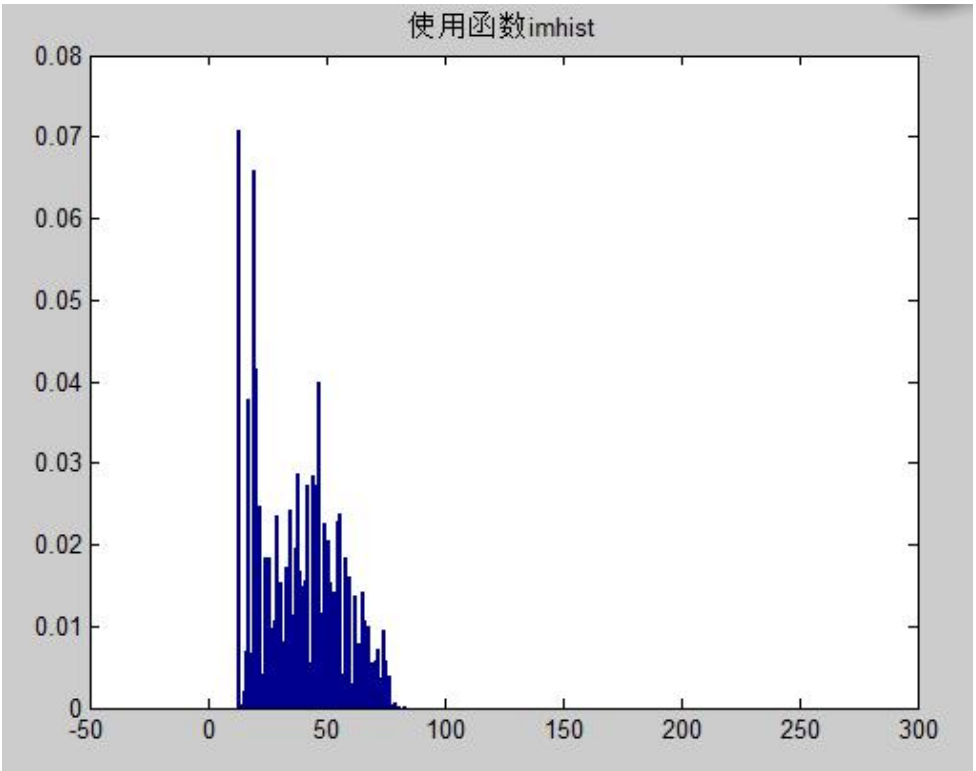
一 直方图均衡

1 读入图像 pollen.jpg。

（a） 编写程序代码画出直方图。



（b） 使用函数 imhist 画出直方图。



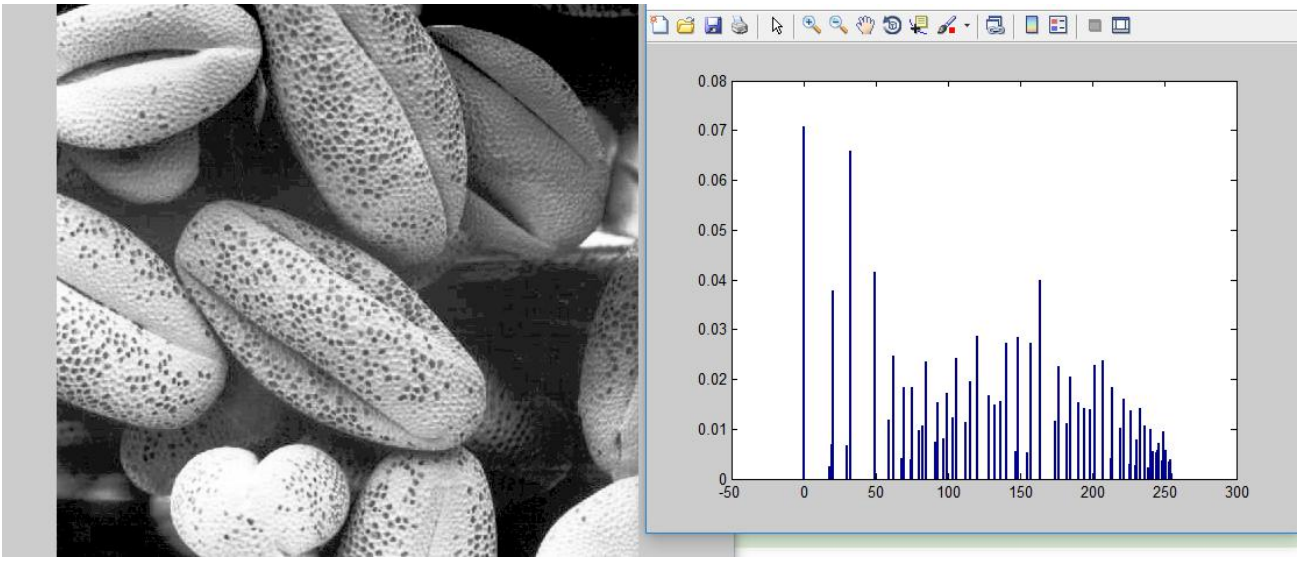
代码实现：

```
I=imread('C:\Users\尚麟静\Desktop\pollen.tif');
[m,n]=size(I);
l=m*n;
```

```
I2=zeros(1,256);
for k=0:255
    image(k+1)=length(find(I==k))/l;
end
figure,bar(0:1:255,I2) ;
I3=imhist(I)/l
figure,bar(0:1:255,I3) ;
```

2 读入图像 pollen.jpg。

（a） 编写程序代码画出均衡化后直方图，并显示此变化后的图像。



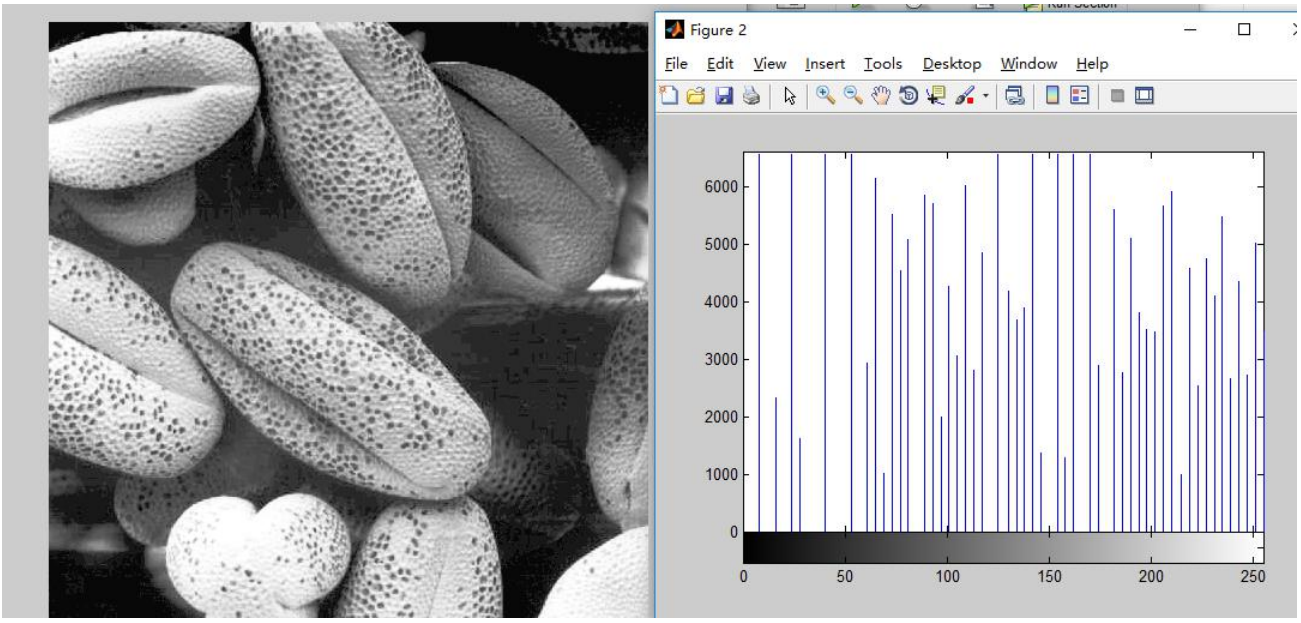
代码实现：

```
I=imread('C:\Users\尚麟静\Desktop\pollen.tif');
[m,n]=size(I);
  r=max(I(:));
  t=min(I(:));
  r=double(r);t=double(t);
I0=zeros(1,256);
I1=zeros(1,101);
I2=zeros(m,n);
l=m*n;
for k=0:r
    I1(k+1)=255*length(find(I<=k))/l;
end
I1=round(I1)

for x=1:n
    for y=1:m
        w=I(x,y);
        I2(x,y)=I1(w+1);
```

```
    end
end
I2=uint8(I2);
figure,imshow(I2);
for k=0:255
    I(k+1)=length(find(I2==k))/l;
end
figure,bar(0:255,I0);
```

（b）使用函数 imhist、histeq 画出直方图，并显示此变化后的图像。



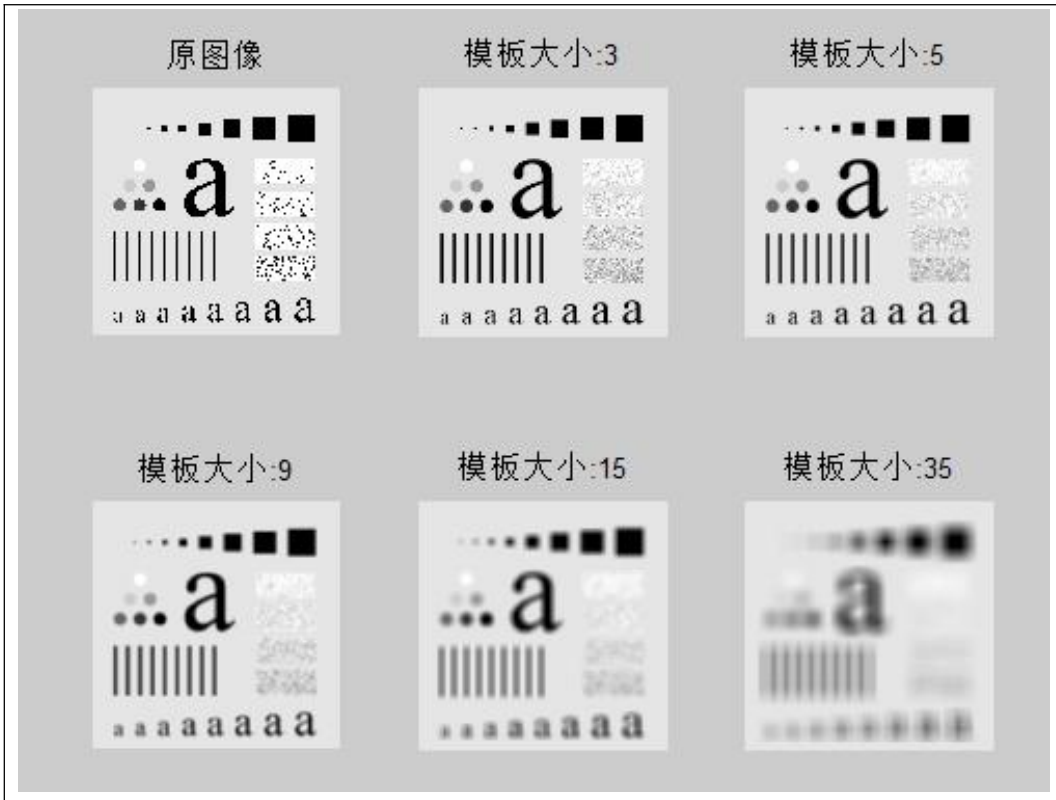
代码实现：

```
I=imread('C:\Users\尚麟静\Desktop\pollen.tif');
I1=histeq(I);
imshow(I1);
figure,imhist(I1);
```

二 平滑空间滤波器

1 均值滤波器

（a）使用方形均值滤波器对图像 a.tiff 进行平滑处理，模板大小分别为 3、5、9、15 和 35，显示原图像和平滑后的图像，图像的边界填充方式为边界重复。



代码实现：

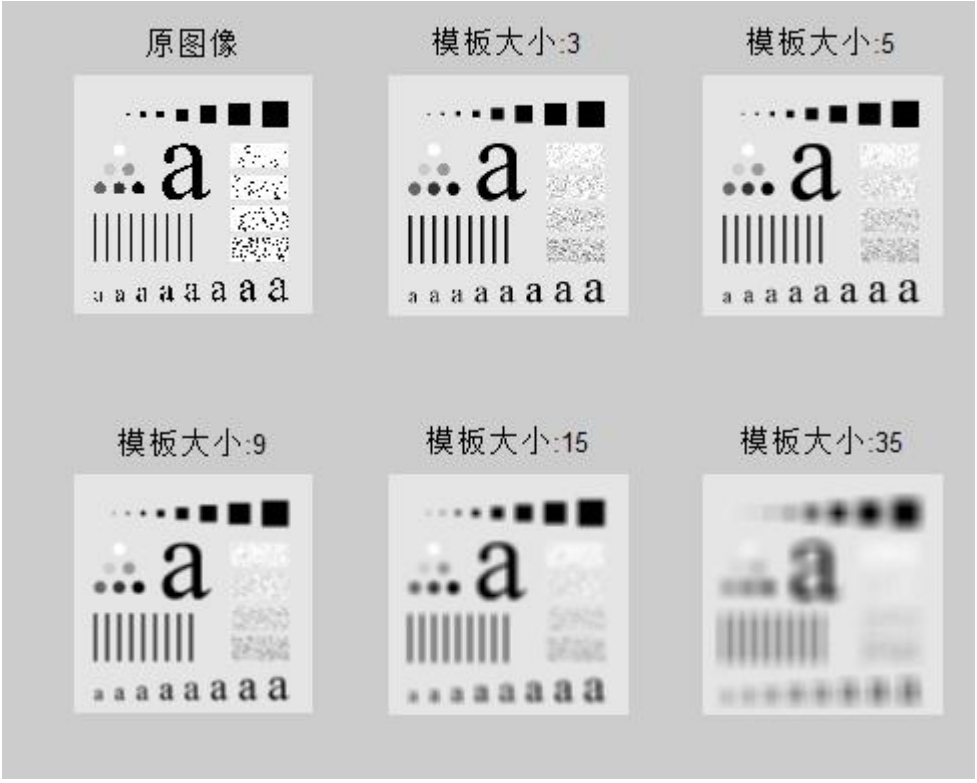
```
I=imread('C:\Users\尚麟静\Desktop\a.tif');
n =3; n1=5;n2=9;n3=15;n4=35;
a = ones(n);
b = ones(n1);
c = ones(n2);
d = ones(n3);
e = ones(n4);
[height, width] = size(I);
x1 = ones(height+2,width+2);
x2 = double(I);    x3 = double(I);
x4 = double(I);    x5 = double(I); x6 = double(I);
x1(1,2:width+1)=I(1,:);
x1(height+2,2:width+1)=I(height,:);
x1(2:height+1,2:width+1)=I(:,:);
x1(:,1)=x1(:,2);
x1(:,width+2)=x1(:,width+1);
for i = 1:height-n+1
    for j = 1:width-n+1
        m = x1(i:i+n-1,j:j+n-1).*a;
        s = sum(sum(m));
        x2(i+(n-1)/2,j+(n-1)/2) = s/(n*n);
    end
end
for i = 1:height-n1+1
```

```
    for j = 1:width-n1+1
        m1 = x1(i:i+n1-1,j:j+n1-1).*b;
        s = sum(sum(m1));
        x3(i+(n1-1)/2,j+(n1-1)/2) = s/(n1*n1);
    end
end
for i = 1:height-n2+1
    for j = 1:width-n2+1
        m2 = x1(i:i+n2-1,j:j+n2-1).*c;
        s = sum(sum(m2));
        x4(i+(n2-1)/2,j+(n2-1)/2) = s/(n2*n2);
    end
end
for i = 1:height-n3+1
    for j = 1:width-n3+1
        m3 = x1(i:i+n3-1,j:j+n3-1).*d;
        s = sum(sum(m3));
        x5(i+(n3-1)/2,j+(n3-1)/2) = s/(n3*n3);
    end
end

for i = 1:height-n4+1
    for j = 1:width-n4+1
        m4 = x1(i:i+n4-1,j:j+n4-1).*e;
        s = sum(sum(m4));
        x6(i+(n4-1)/2,j+(n4-1)/2) = s/(n4*n4);
    end
end

A = uint8(x2);
B = uint8(x3);
C = uint8(x4);
D = uint8(x5);
E = uint8(x6);
A()
subplot(2,3,1), imshow(I),title('原图像');
subplot(2,3,2), imshow(A),title('模板大小:3');
subplot(2,3,3), imshow(B),title('模板大小:5');
subplot(2,3,4), imshow(C),title('模板大小:9');
subplot(2,3,5), imshow(D),title('模板大小:15');
subplot(2,3,6), imshow(E),title('模板大小:35');
```

（b）使用自带函数（fspecial、imfilter）对图像 a.tiff 进行均值滤波，模板大小和图像的边界填充方式都与上问相同。



代码实现：

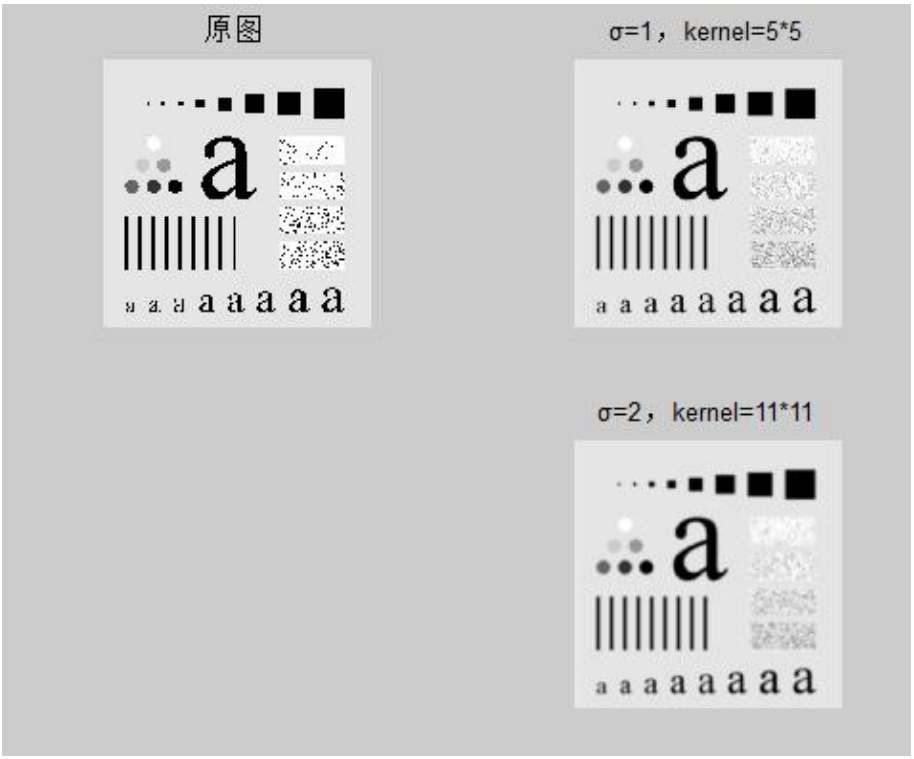
```
I=imread('C:\Users\尚麟静\Desktop\a.tif');
h1 = fspecial('average',3);
h2 = fspecial('average',5);
h3 = fspecial('average',9);
h4 = fspecial('average',15);
h5 = fspecial('average',35);
image1=imfilter(I,h1,'replicate');
image2=imfilter(I,h2,'replicate');
image3=imfilter(I,h3,'replicate');
image4=imfilter(I,h4,'replicate');
image5=imfilter(I,h5,'replicate');
subplot(2,3,1), imshow(I),title('原图像');
subplot(2,3,2), imshow(image1),title('模板大小:3');
subplot(2,3,3), imshow(image2),title('模板大小:5');
subplot(2,3,4), imshow(image3),title('模板大小:9');
subplot(2,3,5), imshow(image4),title('模板大小:15');
```

subplot(2,3,6), imshow(image5),title('模板大小:35');

2　Gaussian 滤波器

（a）编写程序代码，建σ=1，kernel=5*5 和σ=2，kernel=11*11 的 Gaussian 函数。

（b）使用以上两个 Gaussian 滤波器，对图像 a.tiff 进行平滑处理，图像的边界填充方式为边界重复。



代码实现：

```
I=imread('C:\Users\尚麟静\Desktop\a.tif');
sigma1 = 1;sigma2 = 2;
gausFilter1 = fspecial('gaussian',[5 5],sigma1);
gausFilter2 = fspecial('gaussian',[11 11],sigma2);
image1=imfilter(I,gausFilter1,'replicate');
image2=imfilter(I,gausFilter2,'replicate');
subplot(2,2,1),imshow(I),title('原图');
subplot(2,2,2),imshow(image1),title(' σ =1，kernel=5*5');
subplot(2,2,4),imshow(image2),title(' σ =2，kernel=11*11');
```

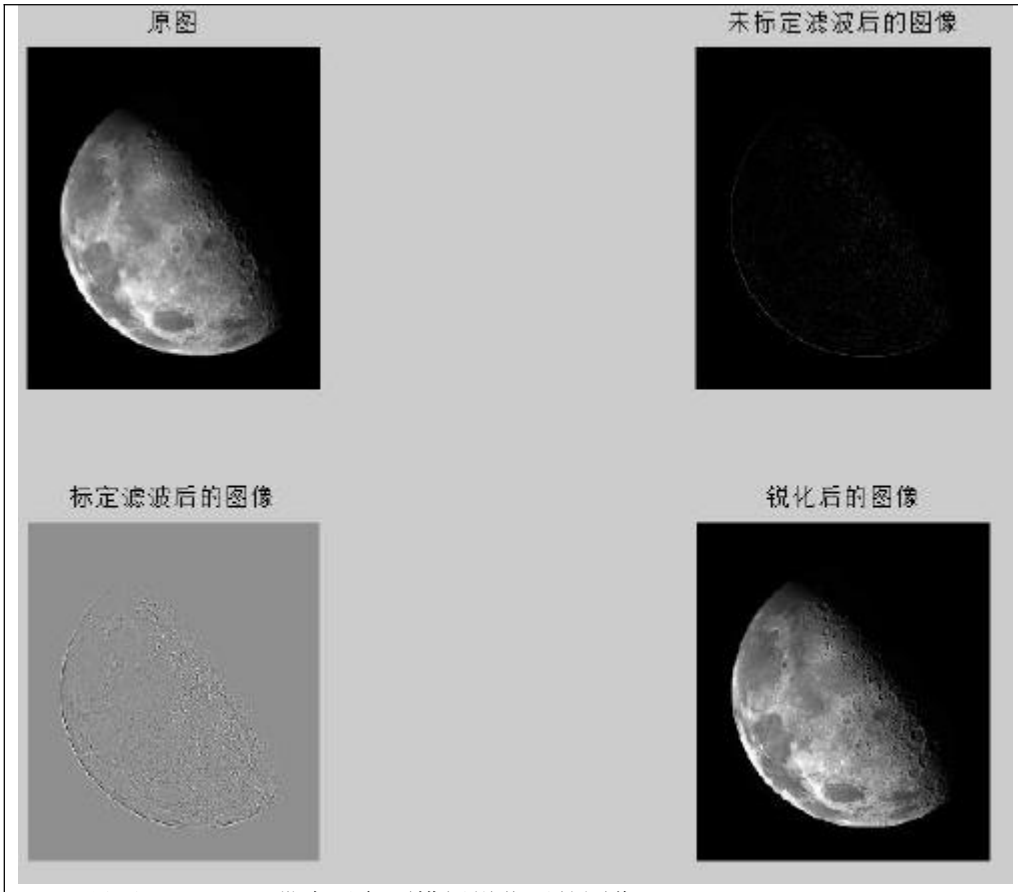
实验四

一 、锐化空间滤波器

1 .Laplacian 滤波器。

（a）读入图像 moon.tif，编写程序代码，实现（b）~（e）。
（b）显示未标定的 Laplacian 滤波后的图像。
（c）显示标定的 Laplacian 滤波后的图像。
（d）显示 Laplacian 模板锐化后的图像。
（e）显示 Laplacian 带有对角项模板锐化后的图像。
（f）使用自带函数（fspecial、imfilter）实现（b）~（e）的结果。
MATLAB 代码：

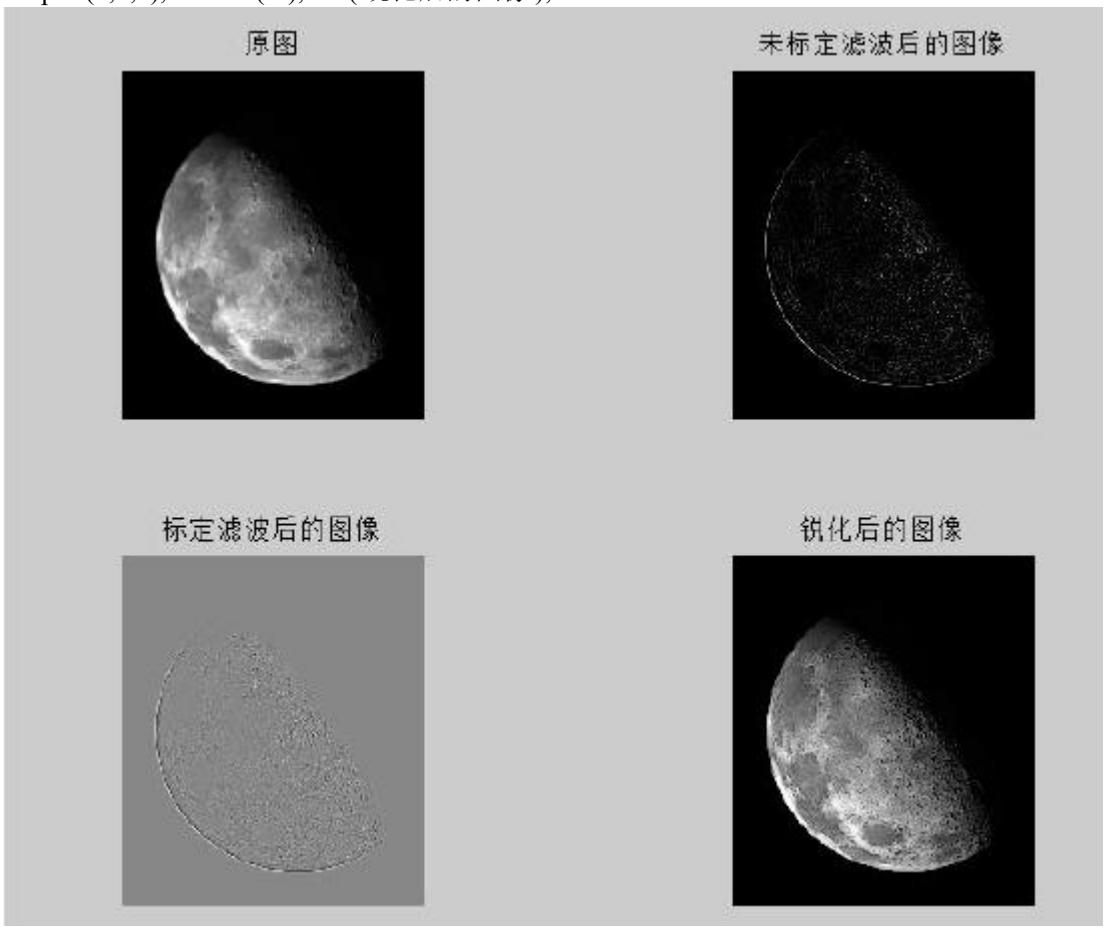
```
I=imread('C:\Users\Desktop\20155467\moon.tif');
yan=zeros(3,3);
yan=[0,1,0;
     1,-4,1;
     0,1,0];
[m,n]=size(I);
I2=zeros(m+2,n+2);
I2(2:m+1,2:n+1)=I;
Id2=double(I2);
Id=double(I);
for x=1:m
     for y=1:n
t=Id2(x:x+2,y:y+2).*yan;
s=sum(sum(t));
Id(x,y)=s;
     end
end
I3=Id;
miner=min(Id(:));
gap=max(Id(:))-min(Id(:))
for x=1:m
     for y=1:n
f=Id(x,y)-miner;
I3(x,y)=255*(f/gap);
     end
end
I4=uint8(I3);
I5=uint8(Id);
I6=I-I5;
subplot(2,2,1),imshow(I),title('原图');
subplot(2,2,2),imshow(I5),title('未标定滤波后的图像');
subplot(2,2,3),imshow(I4),title('标定滤波后的图像');
subplot(2,2,4),imshow(I6),title('锐化后的图像');
```



（e）显示 Laplacian 带有对角项模板锐化后的图像。

```
I=imread('C:\Users\Desktop\20155467\moon.tif');
yan=zeros(3,3);
yan=[1,1,1;
     1,-8,1;
     1,1,1];
[m,n]=size(I);
I2=zeros(m+2,n+2);
I2(2:m+1,2:n+1)=I;
Id2=double(I2);
Id=double(I);
for x=1:m
     for y=1:n
t=Id2(x:x+2,y:y+2).*yan;
s=sum(sum(t));
Id(x,y)=s;
     end
end
I3=Id;
miner=min(Id(:));
gap=max(Id(:))-min(Id(:))
for x=1:m
     for y=1:n
```

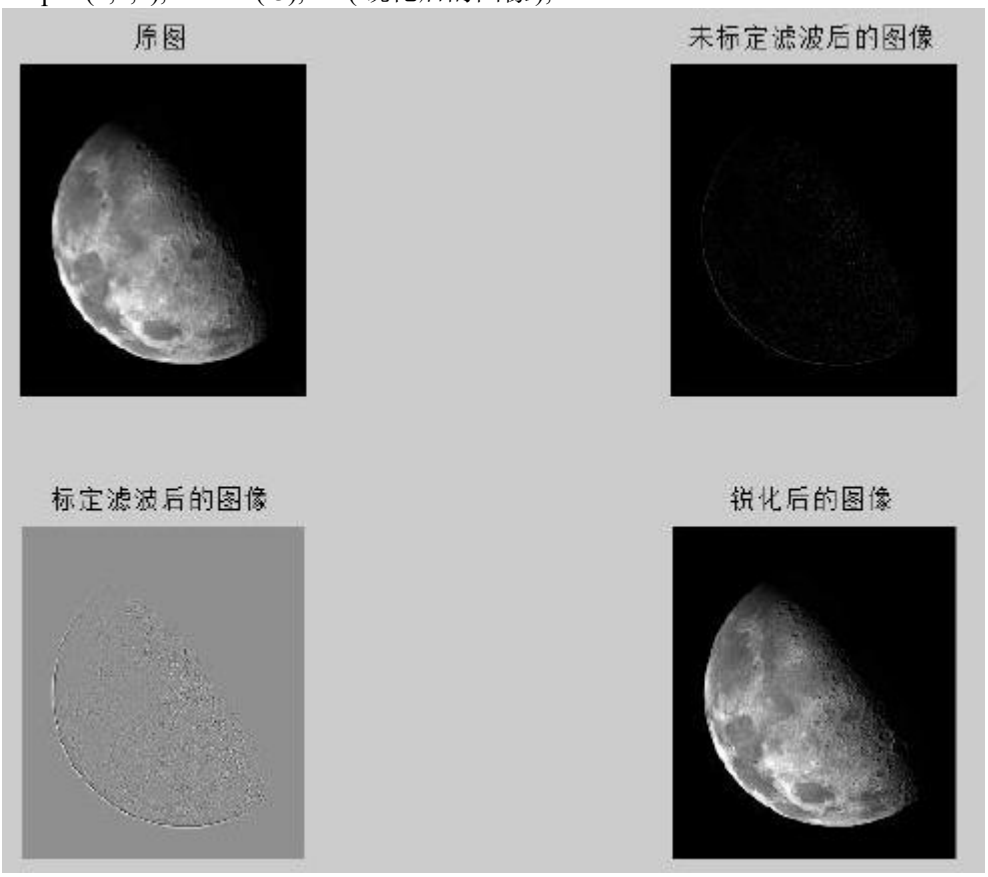
```
f=Id(x,y)-miner;
I3(x,y)=255*(f/gap);
      end
end
I4=uint8(I3);
I5=uint8(Id);
I6=I-I5;
subplot(2,2,1),imshow(I),title('原图');
subplot(2,2,2),imshow(I5),title('未标定滤波后的图像');
subplot(2,2,3),imshow(I4),title('标定滤波后的图像');
subplot(2,2,4),imshow(I6),title('锐化后的图像');
```



（f）使用自带函数（fspecial、imfilter）实现（b）~（e）的结果。

```
I=imread('C:\Users\Desktop\20155467\moon.tif');
[m,n]=size(I);
Id=double(I);
h=fspecial('laplacian',0);
I2=imfilter(Id,h,'replicate');
I5=mat2gray(I2);
I6=uint8(I2);
I3=I-I6;
I5=im2uint8(I5);
subplot(2,2,1),imshow(I),title('原图');
subplot(2,2,2),imshow(I6),title('未标定滤波后的图像');
```

subplot(2,2,3),imshow(I5),title('标定滤波后的图像');
subplot(2,2,4),imshow(I3),title('锐化后的图像');



2 非锐化掩蔽。

（a）读入图像 dipxe.tif。

（b）显示使用 gaussian 滤波模糊后的图像。

（c）显示非锐化模板的图像。

（d）显示非锐化掩蔽的结果。

（e）显示使用高提升滤波的结果。

```
I=imread('C:\Users\Desktop\20155467\dipxe.tif ');
parameters=[5 5];
Id=double(I);
gau = fspecial('gaussian',parameters,3);
I1=imfilter(Id,gau,'replicate');
I3=Id-I1;
I6=Id+I3;
i=4.5*I3;
I8=Id+i;
I4=mat2gray(I3);
I2=uint8(I1);
I5=im2uint8(I4);
I7=mat2gray(I6);
I7=im2uint8(I7);
I9=mat2gray(I8);
I9=im2uint8(I9);
```

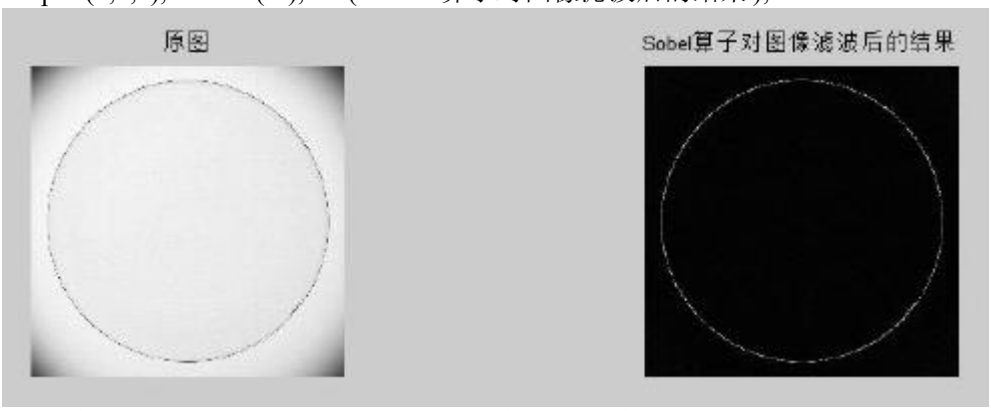
```
subplot(3,2,1),imshow(I),title('原图');
subplot(3,2,2),imshow(I2),title('使用 gaussian 滤波模糊后的图像');
subplot(3,2,3),imshow(I5),title('非锐化模板的图像');
subplot(3,2,4),imshow(I7),title('锐化后的图像');
subplot(3,2,5),imshow(I9),title('使用高提升滤波的结果');
```

3 Sobel 算子

（a）读入图像 contact.tif，编写程序代码，显示 Sobel 算子对图像滤波后的结果。

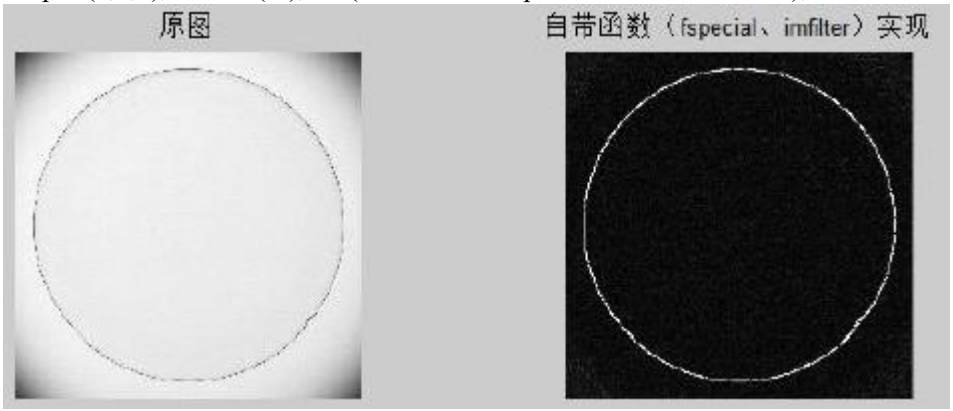
```
I=imread('C:\Users\Desktop\20155467\contact.tif');
yan1=[-1,-2,-1;
        0,0,0;
        1,2,1];
yan2=[-1,0,1;
      -2,0,2;
      -1,0,1];
[m,n]=size(I);
I1=double(I);
I2=zeros(m+2,n+2)
I2(2:m+1,2:n+1)=I;
Id=double(I2);
Id1=double(I);
Id2=double(I);
for x=1:m
    for y=1:n
f=Id(x:x+2,y:y+2).*yan1;
s=sum(sum(f));
Id1(x,y)=s;
    end
end
for x=1:m
```

```
        for y=1:n
f=Id(x:x+2,y:y+2).*yan2;
s=sum(sum(f));
Id2(x,y)=s;
        end
end
Id1=abs(Id1(:,:));
Id2=abs(Id2(:,:));
Id3=Id1+Id2;
I3=mat2gray(Id3);
I4=im2uint8(I3);
subplot(2,2,1),imshow(I),title('原图');
subplot(2,2,2),imshow(I4),title('Sobel 算子对图像滤波后的结果');
```



（b）使用自带函数（fspecial、imfilter）实现上述结果。

```
I=imread('C:\Users\Desktop\201155467\contact.png');
Id=double(I);
f = -fspecial('sobel');
g=f';
I1=imfilter(Id,f,'replicate');
I2=imfilter(Id,f','replicate');
I3=abs(I1)+abs(I2);
I3=uint8(I3);
subplot(2,2,1),imshow(I),title('原图');
subplot(2,2,2),imshow(I3),title('自带函数（fspecial、imfilter）实现');
```

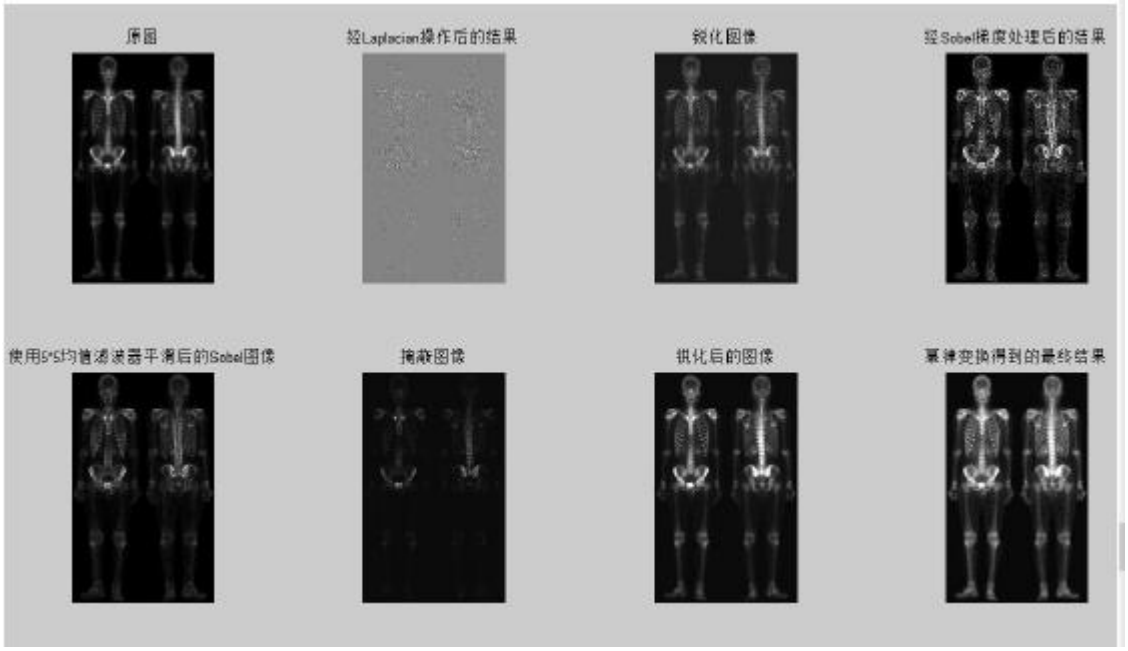


4 混合空间增强法

（a）读入图像 bone.tif。
（b）显示经 Laplacian 操作后的结果。
（c）显示将（a）和（b）相加得到的锐化图像。
（d）显示将（a）图经 Sobel 梯度处理后的结果。
（e）显示使用 5*5 均值滤波器平滑后的 Sobel 图像。
（f）显示由（c）和（e）相乘形成的掩蔽图像。
（g）显示由（a）和（f）求和得到的锐化后的图像。
（h）显示对图（g）应用幂律变换得到的最终结果。
MATLAB 代码：

```
I=imread('C:\Users\Desktop\20155467\bone.png');
parameters=[5 5];
[m,n]=size(I);
Id=double(I);
f1=-fspecial('laplacian',0);
f2=-fspecial('sobel');
f3=f2';
f4= fspecial('gaussian',parameters,3);
I2=imfilter(Id,f1,'replicate');
I4=imfilter(Id,f2,'replicate');
I5=imfilter(Id,f3,'replicate');
I6=abs(I4)+abs(I5);
I7=imfilter(I6,f4,'replicate');
I3=I2+Id;
I8=I7.*I3;
I2=mat2gray(I2);
I2=im2uint8(I2);
I3=mat2gray(I3);
I3=im2uint8(I3);
I6=uint8(I6);
I7=mat2gray(I7);
I7=im2uint8(I7);
I8=mat2gray(I8);
I8=im2uint8(I8);
I9=I+I8;
I10=double(I9);
I10=I10.^0.5;
I10=mat2gray(I10);
I10=im2uint8(I10);
subplot(2,4,1),imshow(I),title('原图');
subplot(2,4,2),imshow(I2),title('经 Laplacian 操作后的结果');
subplot(2,4,3),imshow(I3),title('锐化图像');
subplot(2,4,4),imshow(I6),title('经 Sobel 梯度处理后的结果');
subplot(2,4,5),imshow(I7),title('使用 5*5 均值滤波器平滑后的 Sobel 图像');
subplot(2,4,6),imshow(I8),title('掩蔽图像');
subplot(2,4,7),imshow(I9),title('锐化后的图像');
```

subplot(2,4,8),imshow(I10),title('幂律变换得到的最终结果');



二腐蚀和膨胀

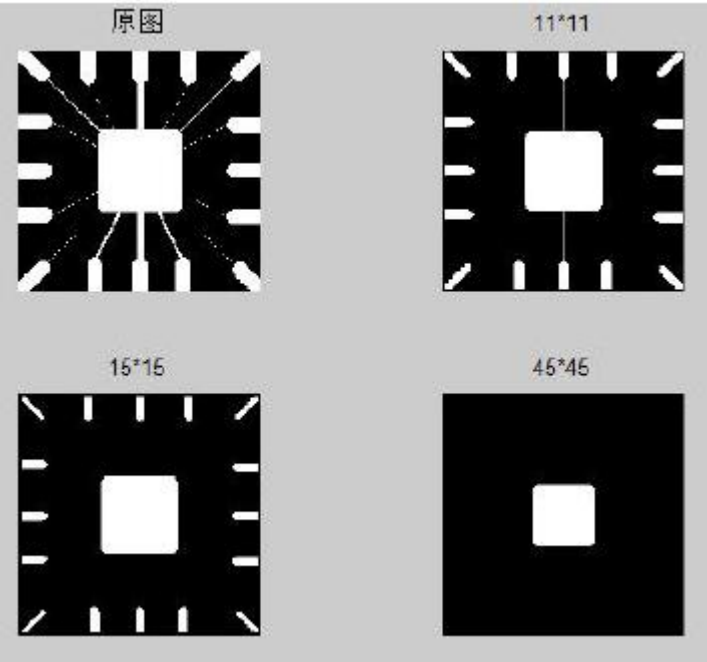
1 腐蚀

（a）读入图像 wirebond.tif，编写程序代码，结构元素分别为 11*11、15*15、45*45 且元素都是 1 的方形结构腐蚀该图像，显示结果。

MATLAB 代码：

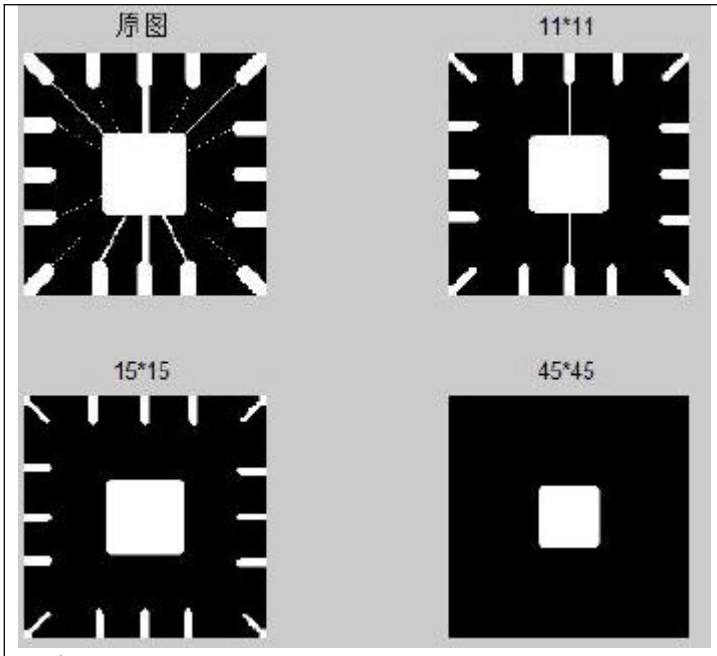
```
I=imread('C:\Users\Desktop\20155467\wirebond.png');
[M,N]=size(I);
I1=mat2gray(I);
I2=zeros(M,N);I3=zeros(M,N);I4=zeros(M,N)
yan1=ones(11,11);
for m=1:M-10
   for n=1:N-10
       if(I1(m:m+10,n:n+10)==yan1)
I2(m+5,n+5)=1;
       else
       end
   end
end
yan2=ones(15,15);
for m=1:M-14
   for n=1:N-14
       if(I1(m:m+14,n:n+14)==yan2)
         I3(m+7,n+7)=1;
       end
   end
```

```
end
yan2=ones(45,45);
for m=1:M-44
   for n=1:N-44
      if(I1(m:m+44,n:n+44)==yan2)
        I4(m+22,n+22)=1;
      else
      end
   end
end
I2=im2uint8(I2);
I3=im2uint8(I3);
I4=im2uint8(I4);
```



（b）使用自带函数（imerode）实现上述结果。

```
I=imread('C:\Users\Desktop\20155467\wirebond.png');
yan1=ones(11,11);
yan2=ones(15,15);
yan3=ones(45,45);
I1=imerode(I,yan1);
I2=imerode(I,yan2);
I3=imerode(I,yan3);
subplot(2,2,1),imshow(I),title('原图');
subplot(2,2,2),imshow(I1),title('11*11');
subplot(2,2,3),imshow(I2),title('15*15');
subplot(2,2,4),imshow(I3),title('45*45');
```

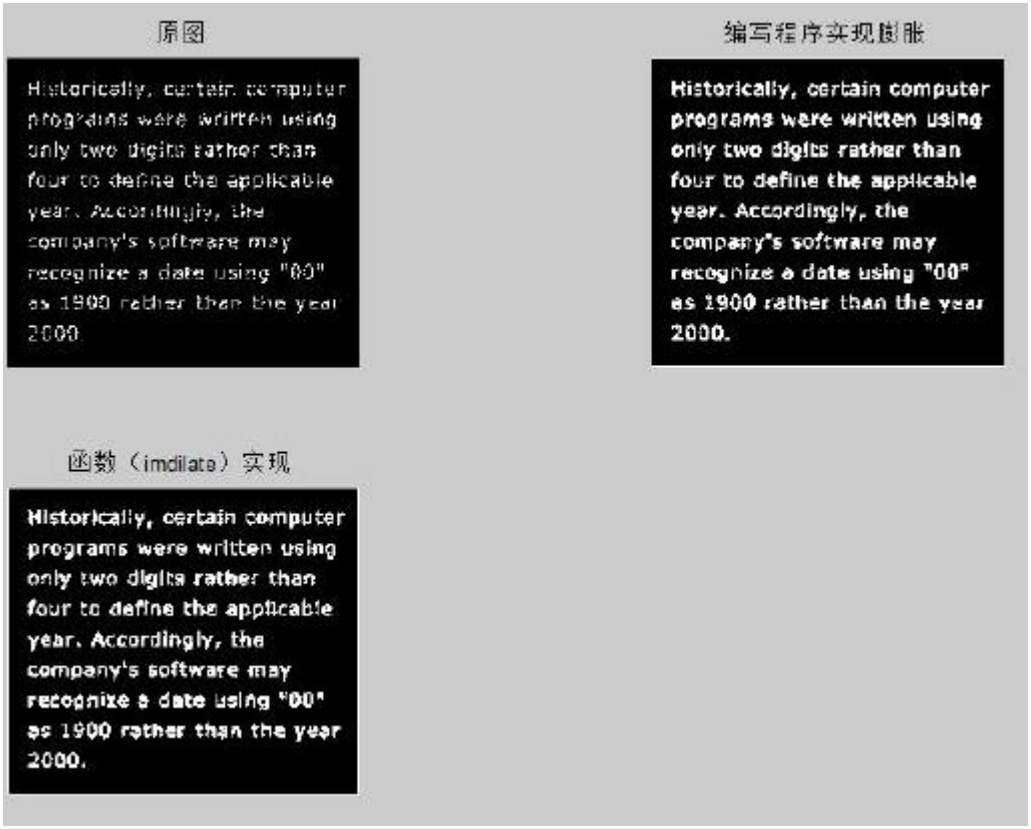


2 膨胀

（a）读入图像 text_gaps.tif，编写程序代码，使用结构元素[0 1 0;1 1 1;0 1 0]膨胀该图像，显示结果。

（b）使用自带函数（imdilate）实现上述结果。

```
I=imread('C:\Users\Desktop\20155467\text_gaps.png');
[M,N]=size(I);
I0=mat2gray(I);
I1=zeros(M+2,N+2);
yan=[0 1 0;1 1 1;0 1 0];
for m=1:M
    for n=1:N
        if(I0(m,n)==1)
            I1(m+1,n:n+2)=1;
            I1(m:m+2,n+1)=1;
        end
    end
end
I1=im2uint8(I1);
I3=I1(2:M+1,2:N+1);
I2=imdilate(I,yan);
subplot(2,2,1),imshow(I),title('原图');
subplot(2,2,2),imshow(I3),title('编写程序实现膨胀');
```



实验五

一 开操作和闭操作

（a）读入图像 noisy_fingerprint.tif 并显示。

（b）使用结构元[1 1 1;1 1 1;1 1 1]对（a）腐蚀，显示结果。

（c）使用相同的结构元对（b）膨胀，显示结果。

（d）使用相同的结构元对（c）膨胀，显示结果。

（e）使用相同的结构元对（d）腐蚀，显示结 果。

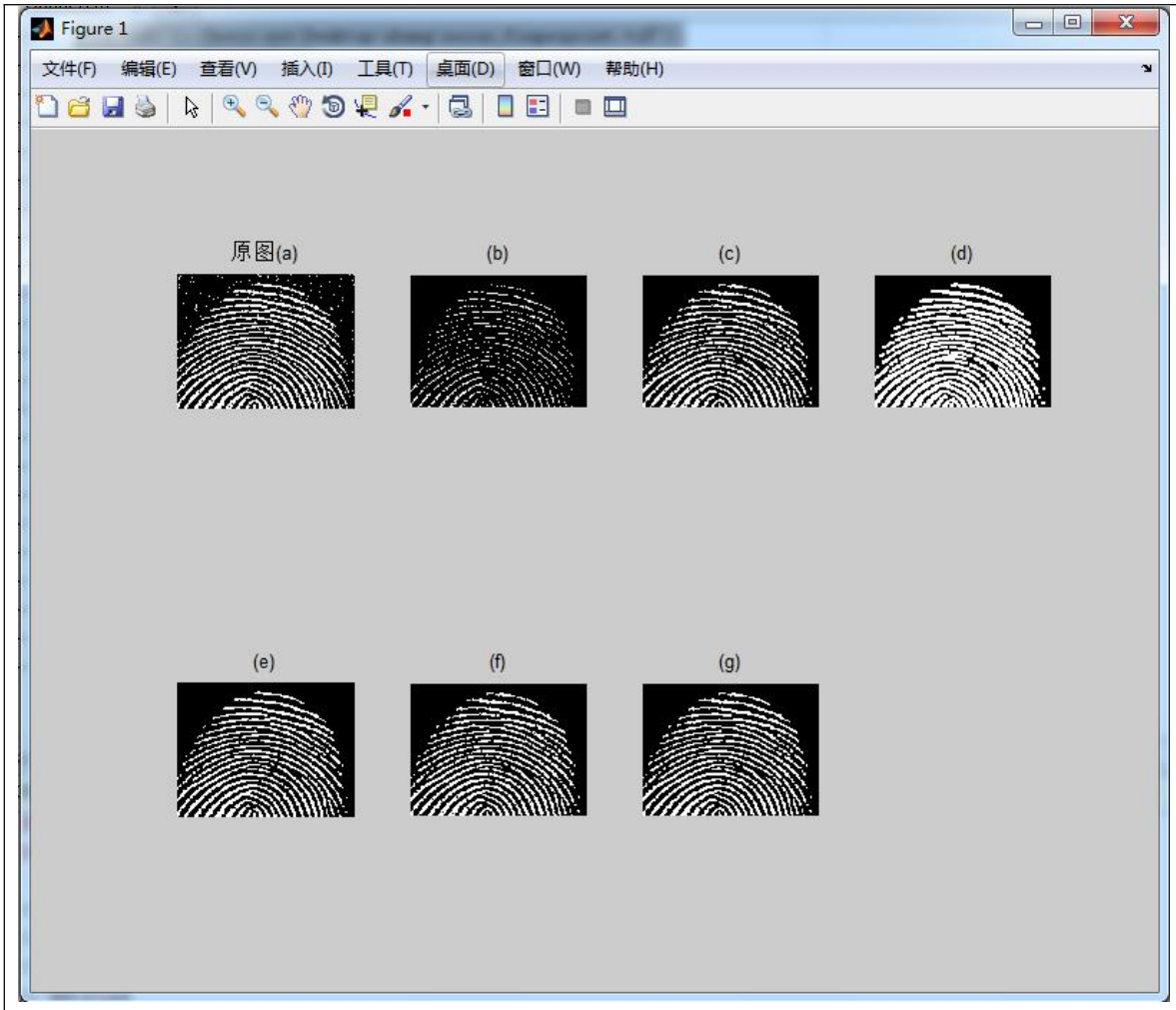
（f）调用函数 imopen，使用相同的结构元对（a）开操作，显示结果，并与（c）结果进行对比。

（g）调用函数 imclose，使用相同的结构元对（f）闭操作，显示结果，并与（e）结果进行对比。

Matlab 代码

```
I=imread('C:\Users\sys\Desktop\shang\noisy_fingerprint.tif');
sha=[1 1 1;1 1 1;1 1 1];
```

```
[m,n]=size(I);
I1=imerode(I,sha);
subplot(2,4,1);imshow(I);
title('原图(a)');
subplot(2,4,2);imshow(I1);
title('(b)')
I2=imdilate(I1,sha);
subplot(2,4,3);imshow(I2);
title('(c)')
I3=imdilate(I2,sha);
subplot(2,4,4);imshow(I3);
title('(d)')
I4=imerode(I3,sha);
subplot(2,4,5);imshow(I4);
title('(e)')
I5=imopen(I,sha);
subplot(2,4,6);imshow(I5);
title('(f)')
I6=imclose(I5,sha);
subplot(2,4,7);imshow(I6);
title('(g)')
```



二 击中或击不中变换

读入图像 small-squares.tif，定位有东、南邻域像素（这些是“击中”）和没有东北、北、西北、西和西南邻域像素（这些是“击不中”）的前景像素。

（a）建立“击中”和“击不中”的掩模。

（b）“击中”掩模对图像腐蚀，显示结果。

（c）“击不中”掩模对图像的补集腐蚀，显示结果。

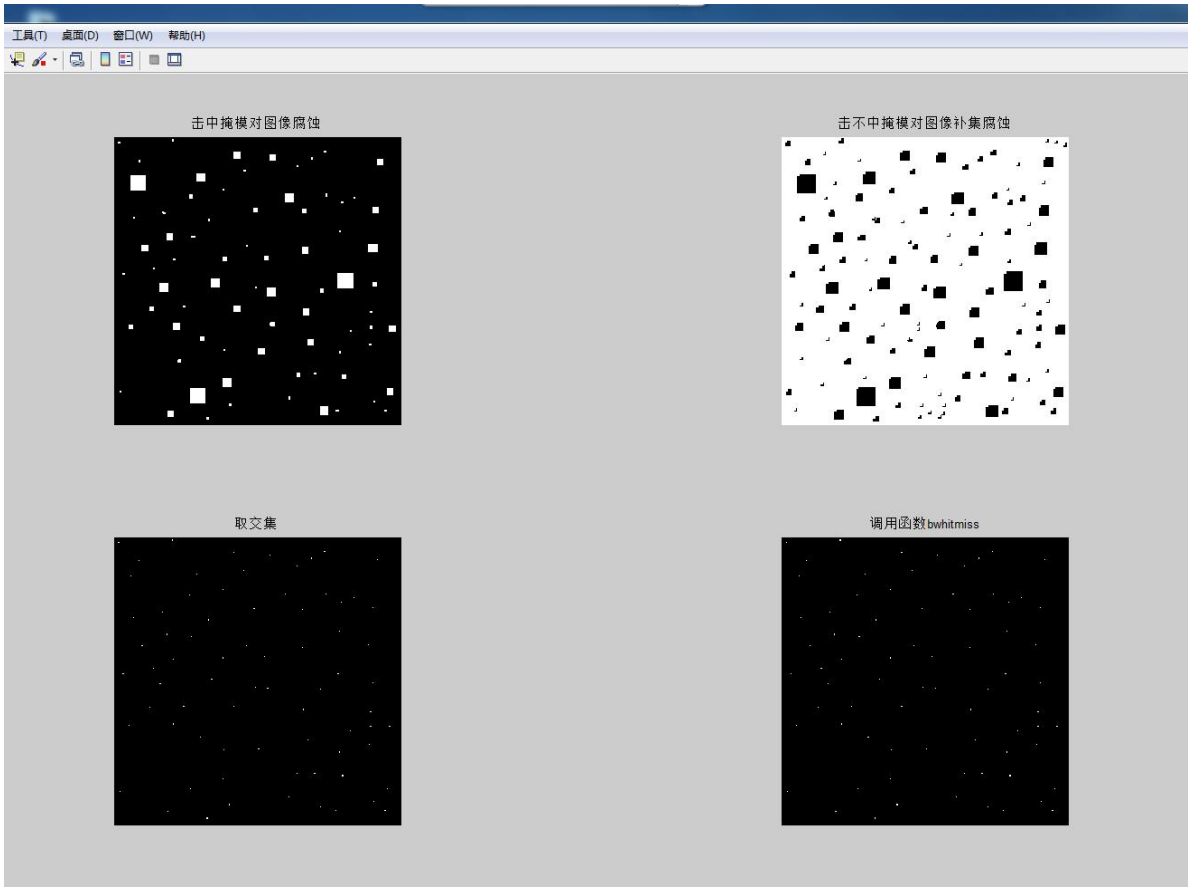
（d）取（b）和（c）的交集，显示结果。

（e）调用函数 bwhitmiss，得到与（d）相同的结果。

MATLAB 代码

```
I=imread('C:\Users\sys\Desktop\shang\small-squares.tif');
mo1=[0 0 0
     0 1 1
     0 1 1];
mo2=[1 1 1
```

```
    1  0  0
    1  0  0];
I1=imerode(I,mo1);
subplot(2,2,1);imshow(I1);title('击中掩模对图像腐蚀')
If=~I;
I2=imerode(If,mo2);
subplot(2,2,2);imshow(I2);title('击不中掩模对图像补集腐蚀)
I1=logical(I1);
I2=logical(I2);
I3=immultiply(I1,I2);
subplot(2,2,3);imshow(I3);title('取交集')
I4=bwhitmiss(I,mo1,mo2);
subplot(2,2,4);imshow(I4);title('调用函数bwhitmiss')
```



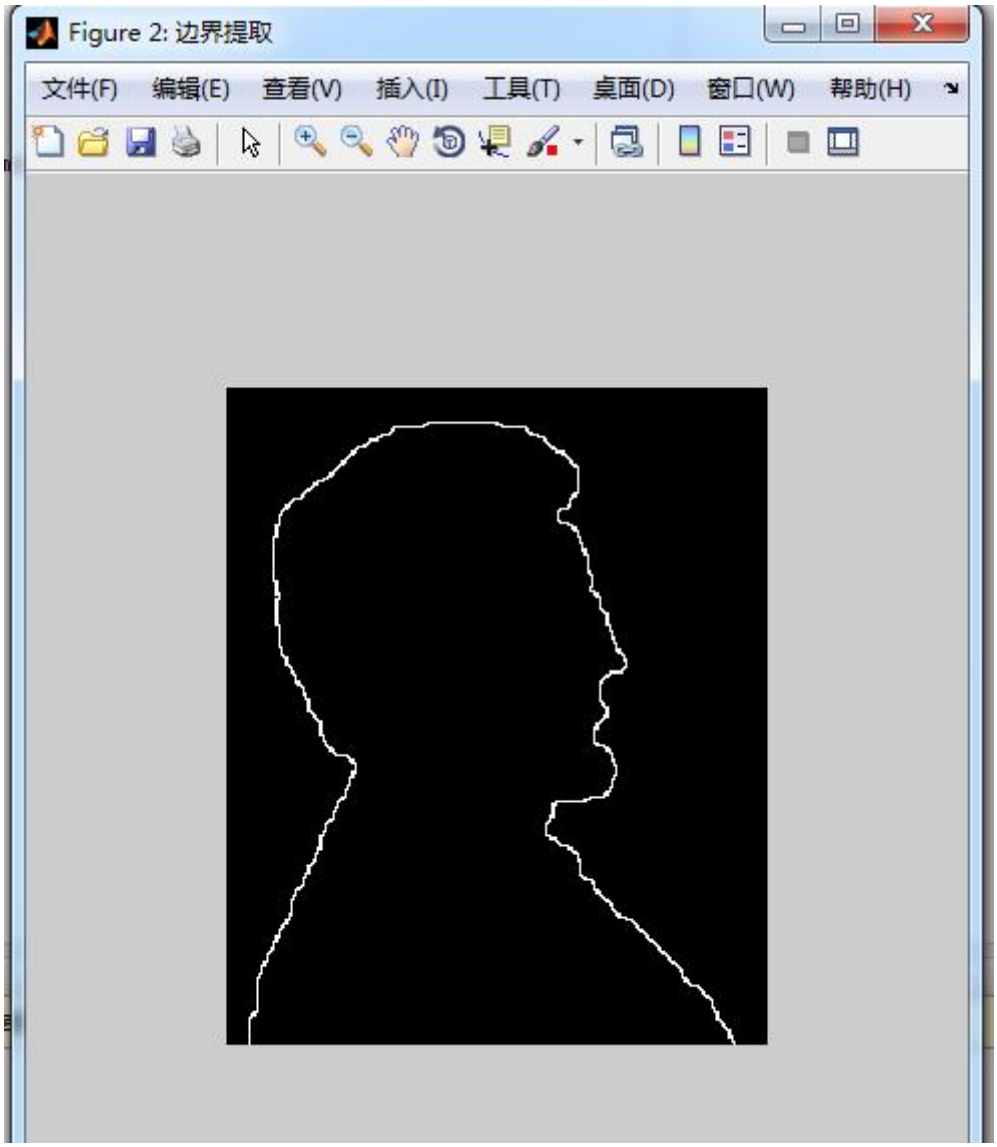
三 边界提取

读入图像 penny.tif，选择合适的掩模提取图像边界。

MATLAB 代码：

```
I=imread('C:\Users\sys\Desktop\shang\penny.tif');
yan=ones(3,3);
I2=imerode(I,yan);
```

```
I=double(I);
I3=I-I2;
figure('name','边界提取');imshow(I3);
```



四 重建开操作

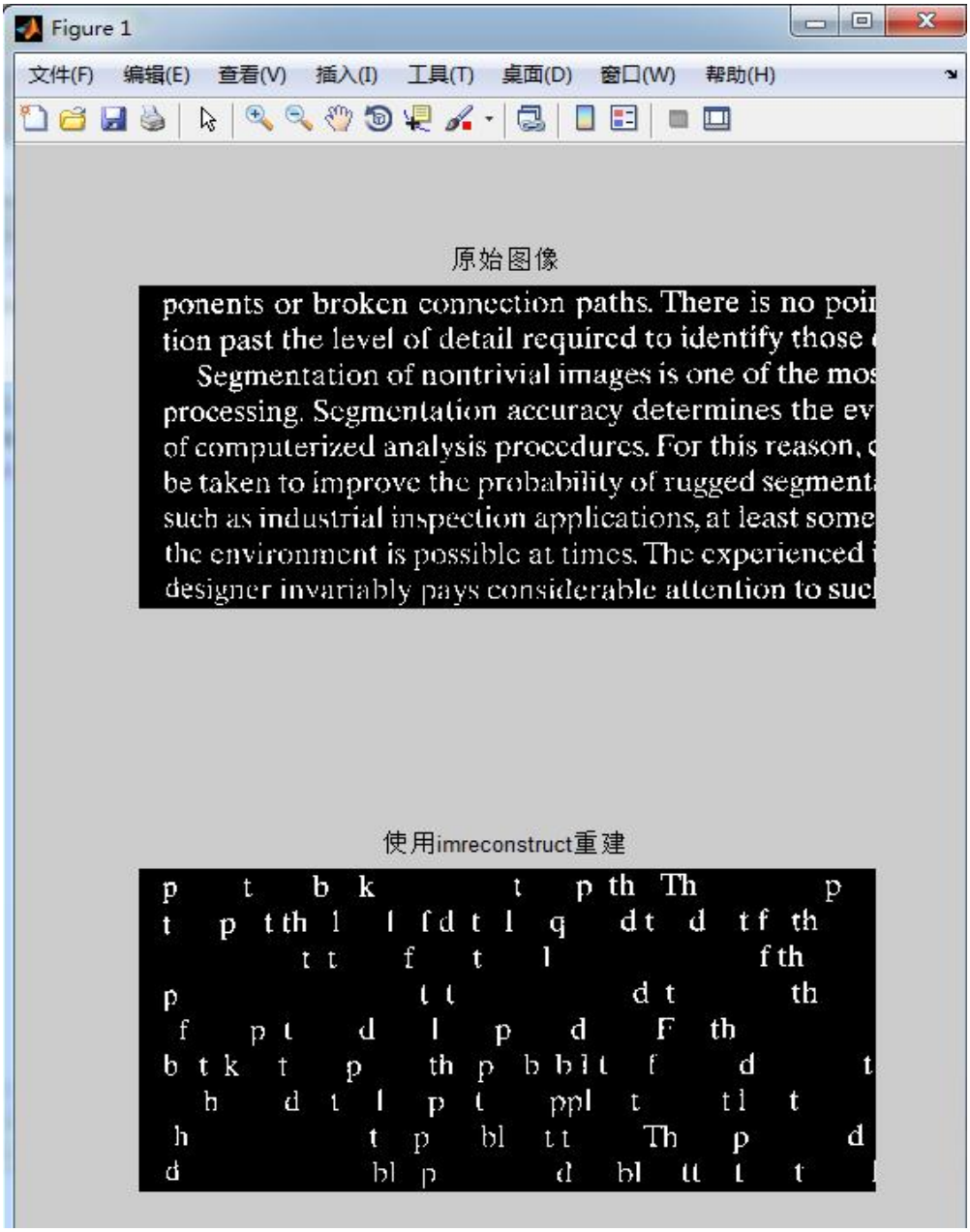
读入图像 text_image.tif，提取该图像中含有长竖线的文字。

提示：可调用函数 imreconstruct。

MATLAB 代码

```
I=imread('C:\Users\sys\Desktop\shang\text_image.tif');
subplot(2,1,1),imshow(I);
title('原始图像')
I2=imerode(I,ones(51,1));
I3=imreconstruct(I2,I);
```

```
subplot(2,1,2),imshow(I3);
title('使用imreconstruct重建')
```



五 灰度级形态学

1 腐蚀和膨胀

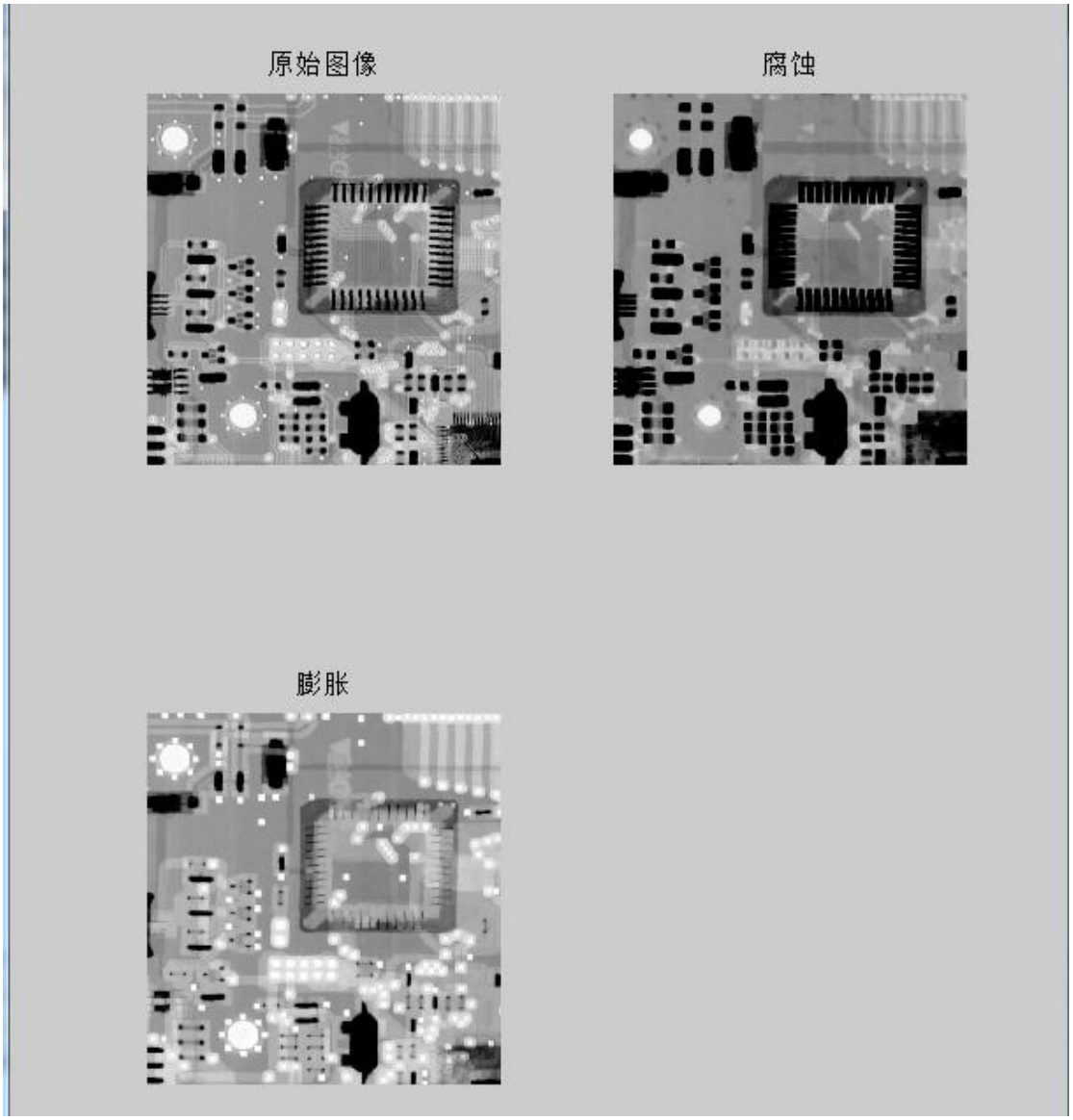
（a）读入图像 ckt_board.tif 并显示。

（b）使用半径为 3 个像素的圆盘结构元对（a）腐蚀，显示结果。

（c）使用半径为 3 个像素的圆盘结构元对（a）膨胀，显示结果。

（d）分别写出腐蚀和膨胀对图像的影响。

提示：可调用函数 strel 建立结构元。



MATLAB 代码

```
I=imread('C:\Users\sys\Desktop\shang\ckt_board.tif');
se_disk=strel('disk',3)
I1=imerode(I,se_disk);
I2=imdilate(I,se_disk);
subplot(2,2,1),imshow(I),title('原始图像')
subplot(2,2,2),imshow(I1),title('腐蚀')
subplot(2,2,3),imshow(I2),title('膨胀')
```

腐蚀使图片黑色部分变多白色部分变少，膨胀使白色部分变多黑色部分变少

2 开操作和闭操作

（a）读入图像 ckt_board.tif 并显示。

（b）使用半径为 3 个像素的圆盘结构元对（a）开操作，显示结果。

（c）使用半径为 3 个像素的圆盘结构元对（a）闭操作，显示结果。

（d）分别写出开操作和闭操作对图像的影响。

提示：可调用函数 strel 建立结构元。

MATLAB 代码

```
I=imread('C:\Users\sys\Desktop\shang\ckt_board.tif');
se_disk=strel('disk',3)
I3=imopen(I,se_disk);
I4=imclose(I,se_disk);
subplot(2,2,1),imshow(I),title('原始图像')
subplot(2,2,2),imshow(I3),title('开操作')
subplot(2,2,3),imshow(I4),title('闭操作')
```

开操作使轮廓线更光滑，闭操作消弥狭窄的间断，填补断裂。