成绩

中国农业大学

课程设计

(2020 -2021 学年春季学期)

题　　目： 基于 MATLAB 的不同大小黄色花朵占比计算

课程名称： 机器视觉与视频处理技术

任课教师： 孙红

班　　级： 自动 191

学　　号： 2019308130215

姓　　名： 孟令昶

目录

数字图像处理课大作业

——基于 Matlab 的不同大小黄色花朵占比计算

图像处理科学技术是信息获取、数据处理、社会生产以及人类生活中不可缺少的基本技术。在我们当今的信息社会中，数字图像处理科学在理论或实践上都存在着巨大的潜力。

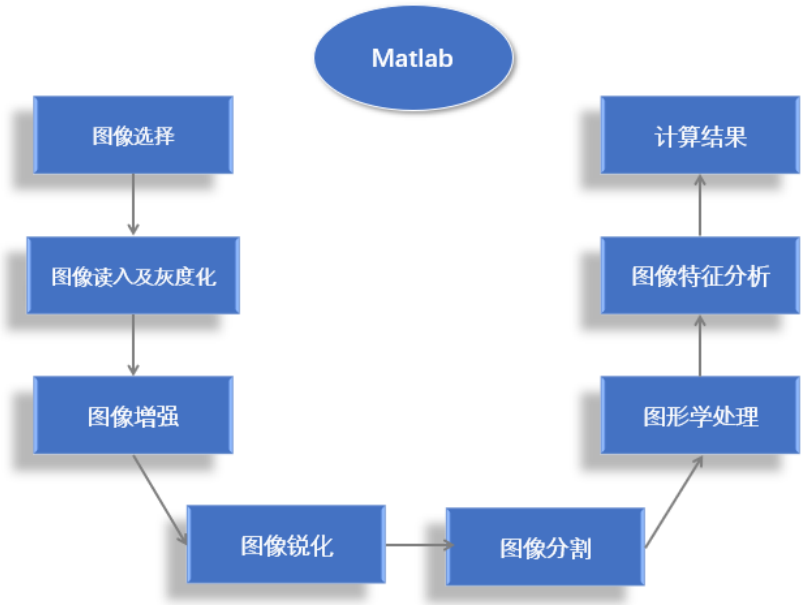
数字图像处理即利用数字计算机或者其他数字硬件，对从图像信息转换而得到的电信号进行某些数学运算，以提高图像的实用性。例如从卫星图片中提取目标物的特征参数，三维立体断层图像的重建等。总的来说，数字图像处理包括点运算、几何处理、图像增强、图像复原、图像形态学处理、图像编码、图像重建、模式识别等。

对于农业有关图像拍摄时受透视效应 “近大远小”影响，部分目标物体显示不完整，无法应用于实验，因此需要对图像进行主体分割计算，本文介绍了基于 Matlab 的不同大小黄色花朵占比计算的过程和方法，并给出了相应结论。

一、实验目的

基于 MATLAB 的数字图像处理方法，包括 RGB 彩色模型、梯度算法锐化、图像分割、图像特征提取等方法，来实现不同大小黄色花朵占比的计算，并给出相应结论。

二、实验过程流程图



三、过程实现

1、图像选择

我选择的是一幅油菜花照片，并将围绕它进行数字图像处理。

2.RGB 图像读入及灰度化

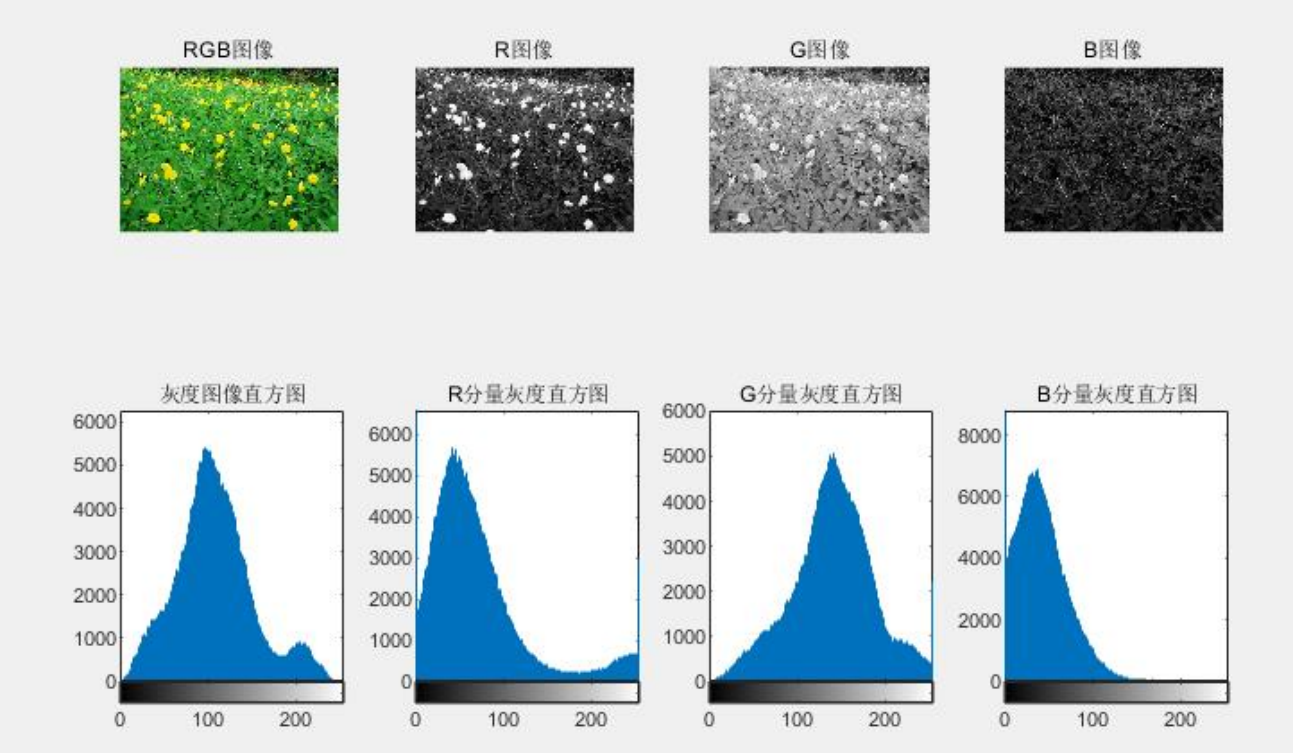
该部分主要是读入彩色图像，并提取出其 R、G、B 三分量，利用 rgb2gray 函数将原图像转化为灰度图像，读取并比较各分量的灰度直方图，选择与原直方图最接近的 B 分量进行后续处理。

（1） 部分程序代码

```
%读入一幅彩色图像
rgb=imread('777.jpg');
subplot(241),imshow(rgb,'InitialMagnification','fit'),title('RGB 图像');%显示原图像
r=rgb(:,:,1); g=rgb(:,:,2); b=rgb(:,:,3);%取出 RGB 的 R、G、B 各分量
subplot(242),imshow(r,'InitialMagnification','fit'),title('R 图像');%显示 R 分量
subplot(243),imshow(g,'InitialMagnification','fit'),title('G 图像');%显示 G 分量
subplot(244),imshow(b,'InitialMagnification','fit'),title('B 图像');%显示 B 分量

%图像灰度化并显示各分量灰度直方图
go=rgb2gray(rgb);
subplot(245),imhist(go),title('灰度图像直方图');
subplot(246),imhist(r),title('R 分量灰度直方图');
subplot(247),imhist(g),title('G 分量灰度直方图');
```

```
subplot(248),imhist(b),title('B 分量灰度直方图');
```

（2）处理结果



3. 图像预处理（图像增强）

该部分为了突出图像的有效信息，削弱某些干扰信息，分别对图像进行平滑、锐化等增强预处理，图像增强的方法有两大类:空域法和频域法。空域法是直接对图像的像素进行处理。频域法是以傅里叶变换为基础，对频谱进行处理。

（1） 图像平滑的空域法

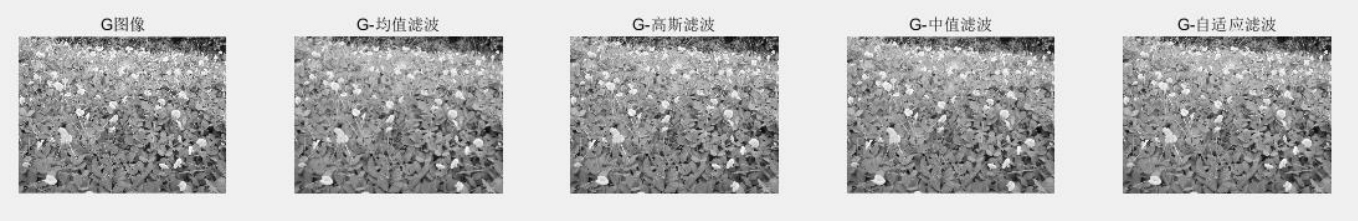
针对图像中的噪声，分别采用空间域均值滤波，高斯滤波，中值滤波，自适应滤波对图像三分量图像进行处理，并对比各种方式的处理效果。

（I） 部分程序代码

```
%R 分量图像平滑
figure, subplot(151),imshow(r,'InitialMagnification','fit'),title('R 图像');
I1=filter2(fspecial('average',3),r)/255; % 3×3 邻域均值滤波
subplot(152),imshow(I1,'InitialMagnification','fit'),title('R-均值滤波');
I2= imfilter(r,fspecial('gaussian',3)); % 3×3 高斯滤波
subplot(153),imshow(I2,'InitialMagnification','fit'),title('R-高斯滤波');
I3=medfilt2(r,[3 3]); % 3×3 中值滤波
subplot(154),imshow(I3,'InitialMagnification','fit'),title('R-中值滤波');
I4=wiener2(r,[3 3]); % 3×3 自适应滤波
subplot(155),imshow(I4,'InitialMagnification','fit'),title('R-自适应滤波');

%G 分量图像平滑
```

```
figure, subplot(151),imshow(g,'InitialMagnification','fit'),title('G 图像');

I1=filter2(fspecial('average',3),g)/255; % 3×3 邻域均值滤波
subplot(152),imshow(I1,'InitialMagnification','fit'),title('G-均值滤波');
I2= imfilter(g,fspecial('gaussian',3)); % 3×3 高斯滤波
subplot(153),imshow(I2,'InitialMagnification','fit'),title('G-高斯滤波');
I3=medfilt2(g,[3 3]); % 3×3 中值滤波
subplot(154),imshow(I3,'InitialMagnification','fit'),title('G-中值滤波');
I4=wiener2(g,[3 3]); % 3×3 自适应滤波
subplot(155),imshow(I4,'InitialMagnification','fit'),title('G-自适应滤波');

%B 分量图像平滑
figure, subplot(151),imshow(b,'InitialMagnification','fit'),title('B 图像');
I1=filter2(fspecial('average',3),b)/255; % 3×3 邻域均值滤波
subplot(152),imshow(I1,'InitialMagnification','fit'),title('B-均值滤波');
I2= imfilter(b,fspecial('gaussian',3)); % 3×3 高斯滤波
subplot(153),imshow(I2,'InitialMagnification','fit'),title('B-高斯滤波');
I3=medfilt2(b,[3 3]); % 3×3 中值滤波
subplot(154),imshow(I3,'InitialMagnification','fit'),title('B-中值滤波');
I4=wiener2(b,[3 3]); % 3×3 自适应滤波
subplot(155),imshow(I4,'InitialMagnification','fit'),title('B-自适应滤波');
```

（II） 处理结果



（III） 分析结果

对 RGB 三层图像进行空域平滑，可见不同的处理方法都能起到消除噪声的作用，均值滤波随着 filter 的增大，图像越来越模糊；高斯滤波去噪效果与均值滤波相差不大，但模糊程

度明显变小；中值滤波边缘保护的效果比较好，但窗口尺寸不好把握；自适应滤波的处理适应性比较强，处理效果也比较好。

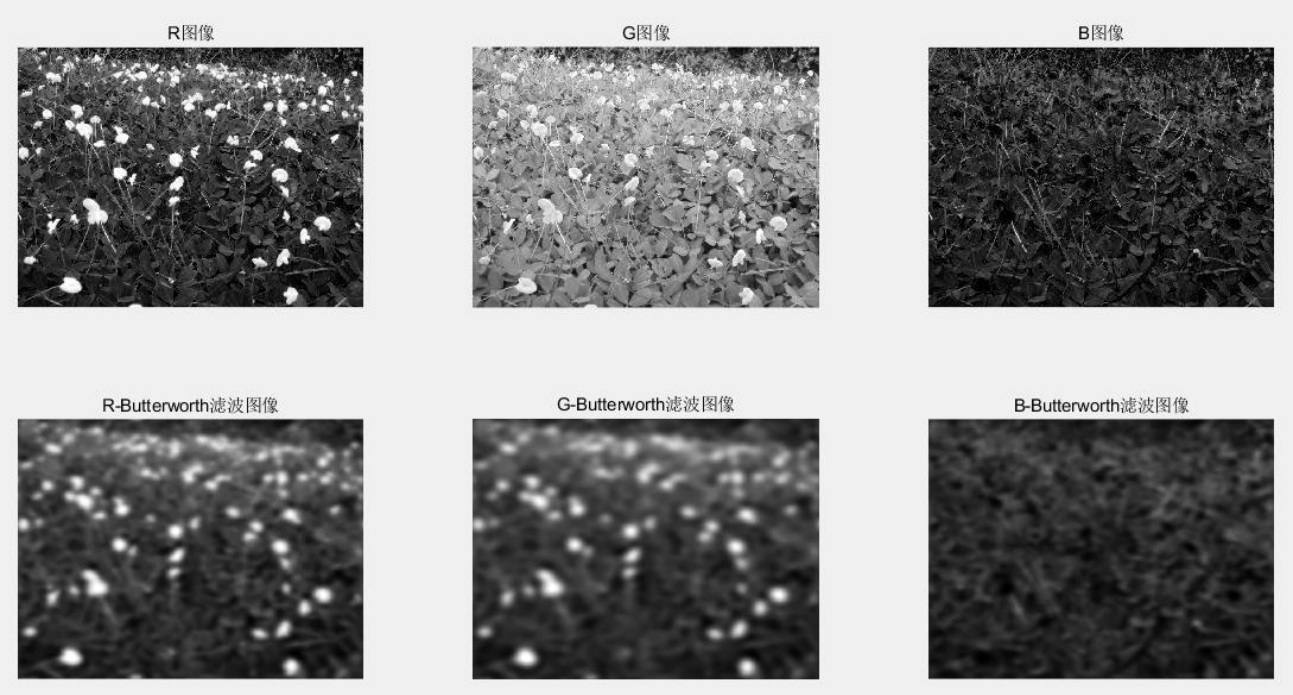
(2)图像平滑的频域法（Butterworth 低通滤波）

(I)部分程序代码：

```
rgb=imread('777.jpg');
r=rgb(:,:,1); g=rgb(:,:,2); b=rgb(:,:,3);
subplot(231),imshow(r,'InitialMagnification','fit'),title('RÍ¼Ïñ');
subplot(232),imshow(g,'InitialMagnification','fit'),title('GÍ¼Ïñ');
subplot(233),imshow(b,'InitialMagnification','fit'),title('BÍ¼Ïñ');

r=double(r);
f=fft2(r);
g=fftshift(f); %转换数据矩阵/数据矩阵平衡
[N,M]=size(g);
n=2;
d0=20;
n1=fix(M/2);
n2=fix(N/2); %求出频域原点(n1,n2)
for i=1:N
for j=1:M
d=sqrt((j-n1)^2+(i-n2)^2);
h=1/(1+0.414*(d/d0)^(2*n)); %2 阶 Butterworth 低通滤波器
g(i,j)=h*g(i,j);
end
end
g=ifftshift(g);
g=uint8(real(ifft2(g)));
subplot(234),imshow(g);
title('R-Butterworth 滤波图像');
g1=double(g);
f=fft2(g1);
g=fftshift(f); %转换数据矩阵/数据矩阵平衡
[N,M]=size(g);
n=2;
d0=20;
n1=fix(M/2);
n2=fix(N/2); %求出频域原点(n1,n2)
for i=1:N
for j=1:M
d=sqrt((j-n1)^2+(i-n2)^2);
h=1/(1+0.414*(d/d0)^(2*n)); %2 阶 Butterworth 低通滤波器
g(i,j)=h*g(i,j);
```

```
end
end
g=ifftshift(g);
g=uint8(real(ifft2(g)));
subplot(235),imshow(g);
title('G-Butterworth 滤波图像');
b=double(b);
f=fft2(b);
g=fftshift(f); %转换数据矩阵/数据矩阵平衡
[N,M]=size(g);
n=2;
d0=20;
n1=fix(M/2);
n2=fix(N/2); %求出频域原点(n1,n2)
for i=1:N
for j=1:M
d=sqrt((j-n1)^2+(i-n2)^2);
h=1/(1+0.414*(d/d0)^(2*n)); %2 阶 Butterworth 低通滤波器
g(i,j)=h*g(i,j);
end
end
g=ifftshift(g);
g=uint8(real(ifft2(g)));
subplot(236),imshow(g);
title('B-Butterworth 滤波图像');
```

（II）运行结果



（3）结果分析：

频域处理，虽然将时域卷积运算变成了频域的乘积运算，减轻了运算负担，但由于图像中高频分量比较杂乱，使处理后的图像明显变得模糊。而且由于图像平滑过程中会导致图像损失

一部分信息，所以在这种情况下，频域滤波处理效果不好，故可以直接对原图像进行后续处理。

4、RGB 图层锐化

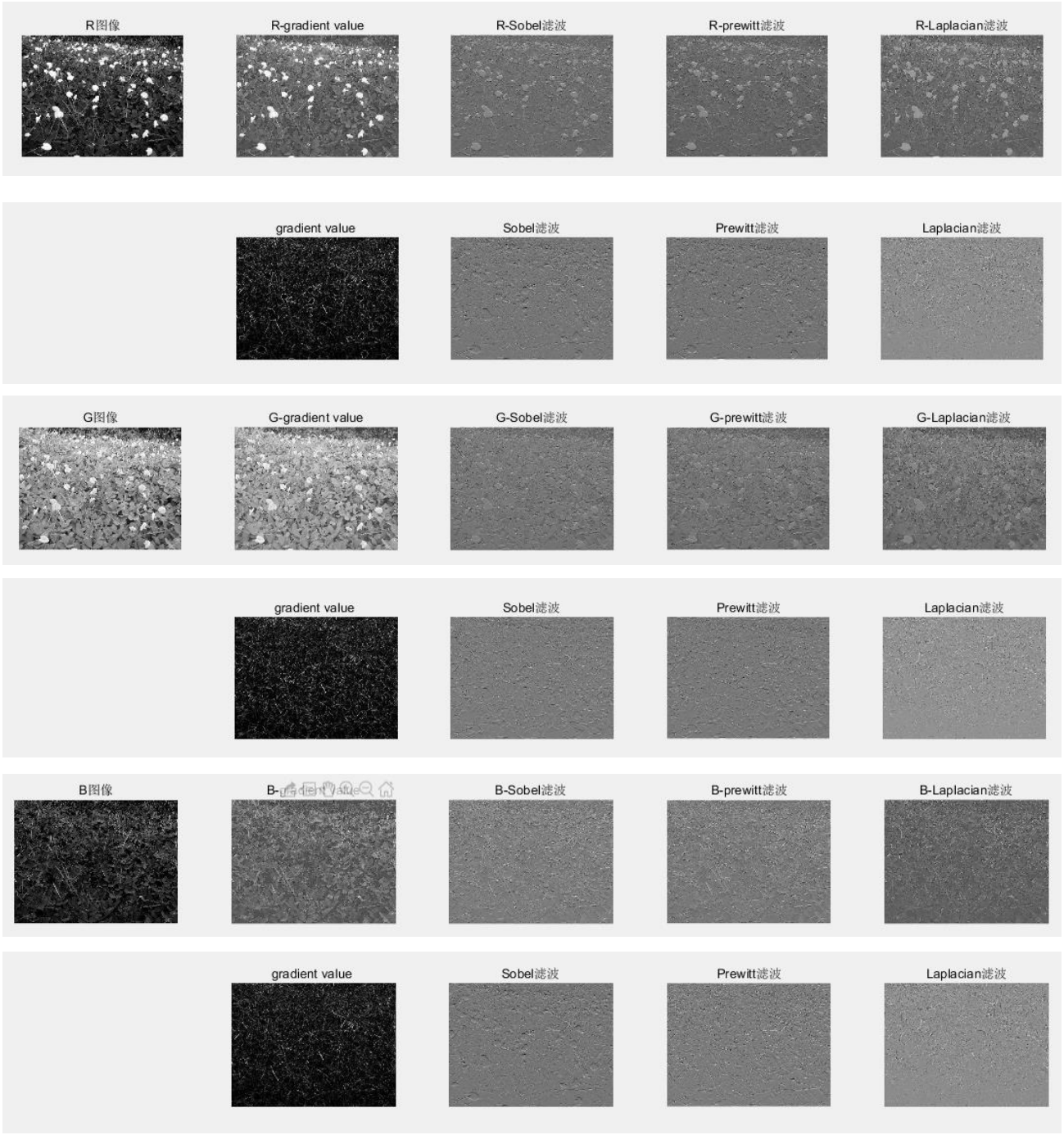
采用梯度算法、索博尔算法、蒲瑞维特、拉普拉斯算法对 R、G、B 图层分别锐化。最终的图像由图层与滤波后图像相减得到。

（1）部分程序代码

```
%R 层锐化：
r=im2double(r); %双精度化处理
[rX,rY]=gradient(r); %返回矩阵 I 梯度值的 X 和 Y 分量
r1=sqrt(rX.*rX+rY.*rY); %得到梯度算法结果图像
figure,subplot(252),imshow(r1,[]);title('gradient value ');
h2=fspecial('sobel');%选择索伯尔算法
r2=imfilter(r,h2);%采用索伯尔算法滤波
subplot(253),imshow(r2,[]);title('Sobel 滤波');% 显示索伯尔算法结果图像
h3=fspecial('prewitt');%选择蒲瑞维特算法
r3=imfilter(r,h3);%采用蒲瑞维特算法滤波
subplot(254),imshow(r3,[]);title('Prewitt 滤波');% 显示蒲瑞维特算法结果图像
h4=fspecial('laplacian'); %选择滤波算法为拉普拉 斯算法
r4=imfilter(r,h4);%采用拉普拉斯算法滤波
subplot(255),imshow(r4,[]);title('Laplacian 滤波'); %在中下区域显示拉普拉斯算法结果图像
%图层相减
subplot(256),imshow(r,'InitialMagnification','fit'),title('R图像');
r5=r-r1;
subplot(257),imshow(r5,[]);title('R-gradient value');
r6=r-r2;
subplot(258),imshow(r6,[]);title('R-Sobel 滤波');
r7=r-r3;
subplot(259),imshow(r7,[]);title('R-prewitt 滤波');
r8=r-r4;
subplot(2,5,10),imshow(r8,[]);title('R-Laplacian 滤波');

%G 层锐化：
g=im2double(g); %双精度化处理
[gX,gY]=gradient(g); %返回矩阵 I 梯度值的 X 和 Y 分量
g1=sqrt(gX.*gX+gY.*gY); %得到梯度算法结果图像
figure,subplot(252),imshow(g1,[]);title('gradient value ');
h2=fspecial('sobel');%选择索伯尔算法
g2=imfilter(g,h2);%采用索伯尔算法滤波
subplot(253),imshow(g2,[]);title('Sobel 滤波');% 显示索伯尔算法结果图像
h3=fspecial('prewitt');%选择蒲瑞维特算法
g3=imfilter(g,h3);%采用蒲瑞维特算法滤波
subplot(254),imshow(g3,[]);title('Prewitt 滤波');% 显示蒲瑞维特算法结果图像
```

```
h4=fspecial('laplacian'); %选择滤波算法为拉普拉 斯算法
g4=imfilter(g,h4);%采用拉普拉斯算法滤波
subplot(255),imshow(g4,[]);title('Laplacian 滤波'); %在中下区域显示拉普拉斯算法结果图像
%图层相减
subplot(256),imshow(g,'InitialMagnification','fit'),title('G图像');
g5=g-g1;
subplot(257),imshow(g5,[]);title('G-gradient value');
g6=g-g2;
subplot(258),imshow(g6,[]);title('G-Sobel 滤波');
g7=g-g3;
subplot(259),imshow(g7,[]);title('G-prewitt 滤波');
g8=g-g4;
subplot(2,5,10),imshow(g8,[]);title('G-Laplacian 滤波');

%B 层锐化：
b=im2double(b); %双精度化处理
[bX,bY]=gradient(b); %返回矩阵 I 梯度值的 X 和 Y 分量
b1=sqrt(bX.*bX+bY.*bY); %得到梯度算法结果图像
figure,subplot(252),imshow(b1,[]);title('gradient value ');
h2=fspecial('sobel');%选择索伯尔算法
b2=imfilter(b,h2);%采用索伯尔算法滤波
subplot(253),imshow(r2,[]);title('Sobel 滤波');% 显示索伯尔算法结果图像
h3=fspecial('prewitt');%选择蒲瑞维特算法
b3=imfilter(b,h3);%采用蒲瑞维特算法滤波
subplot(254),imshow(b3,[]);title('Prewitt 滤波');% 显示蒲瑞维特算法结果图像
h4=fspecial('laplacian'); %选择滤波算法为拉普拉 斯算法
b4=imfilter(b,h4);%采用拉普拉斯算法滤波
subplot(255),imshow(b4,[]);title('Laplacian 滤波'); %在中下区域显示拉普拉斯算法结果图像
%图层相减
subplot(256),imshow(b,'InitialMagnification','fit'),title('B图像');
b5=b-b1;
subplot(257),imshow(b5,[]);title('B-gradient value');
b6=b-b2;
subplot(258),imshow(b6,[]);title('B-Sobel 滤波');
b7=b-b3;
subplot(259),imshow(b7,[]);title('B-prewitt 滤波');
b8=b-b4;
subplot(2,5,10),imshow(b8,[]);title('B-Laplacian 滤波');
```

（2）锐化结果



（3）结果分析

一阶微分的 Roberts、Sobel、Prewitt 算法由于图像的灰度变化不是特别明显，仅一阶算子可能找不到边界，锐化后基本失去利用价值；在上述情况下，利用二阶微分算子，如 Laplace、LoG 算子，对噪声比较敏感，能找出一阶微分算子无法处理的边界，对比不同算子的处理结果，可以看出梯度算法的处理效果远好于其他方法，并且 R、G 层上的细节信息被清晰的显现出来，因此可采用梯度算法的结果进行之后的处理。

5、图像分割

为了把目标提取出来，可以通过分析三通道直方图的双峰特性，其中 R 通道的直方图有双峰现象，有可能将花朵从背景中分割出来；也可以通过图片的颜色特性，设置阈值提取目标颜色区域，目标花朵是均为黄色，有可能通过颜色特征提取。

（1）阈值的确定

（i）迭代思想的自动阈值法：

代码：

```
I=double(r5);
T=(min(I(:))+max(I(:)))/2; %选择灰度中值作为初始阈值T
done=false;
while ~done
I1=I>=T; %利用阈值T把图像分割成两个区域
T1=(mean(I(I1))+mean(I(~I1)))/2; %计算均值作为新阈值
done=abs(T-T1)<0.5; T=T1; %与上个阈值差小于给定值
end
figure,imshow(I1);
```

处理结果：

```
T =

    0.3345
```

采用这种方法，由于黄花与茎秆的灰度值相近，噪声过多，很难将其从背景中分割出来

（ii）最大类间方差法

代码

```
level= graythresh(r5);%采用最大类间方差法自动求取阈值
I1=imbinarize(r5,level);%利用所得到的阈值分割图像
```

```
figure,imshow(I1);%显示分割后的二值图像
```

结果

```
level =

    0.4549
```

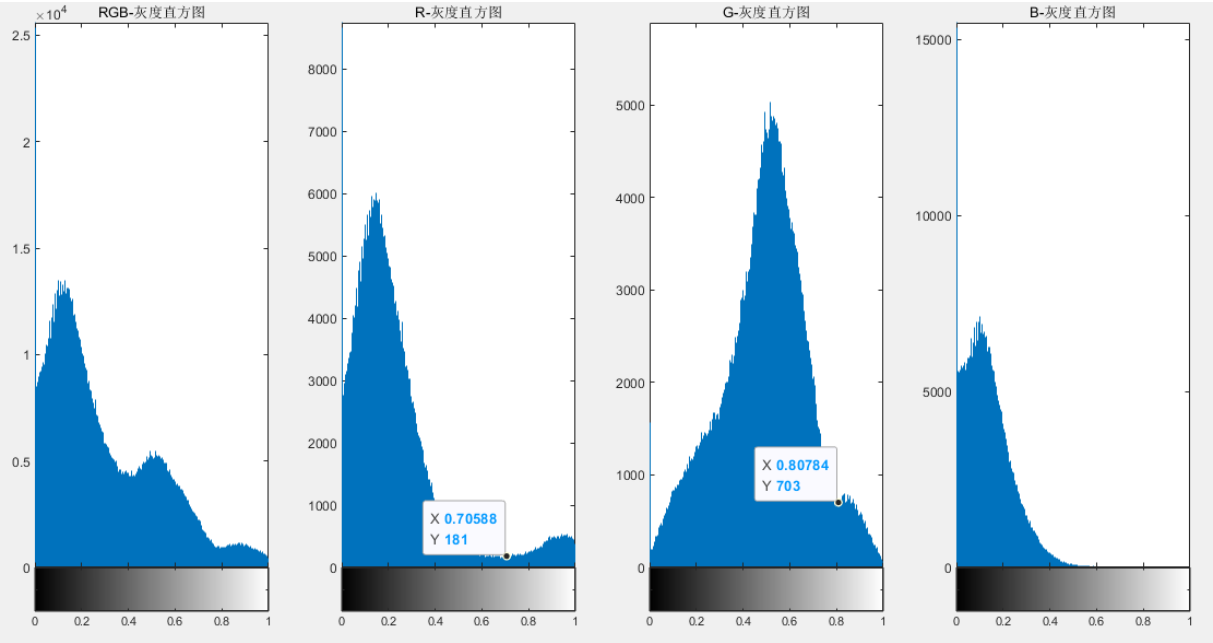
最大类间方差法减少了绝大多数噪声，处理效果明显提高，但仍然不能将结果直接用于后续处理

（iii）利用直方图人工确定阈值

先得到锐化处理的直方图，由图可得阈值设为 0.75；

（2）基于直方图的图像分割

首先把锐化后的 RGB 三层叠加，即叠加 R-gradient value, G-gradient value, B-gradient value 三层。然后基于 R、G 层，进行分割，先得到直方图，由图可得阈值设为 0.75；



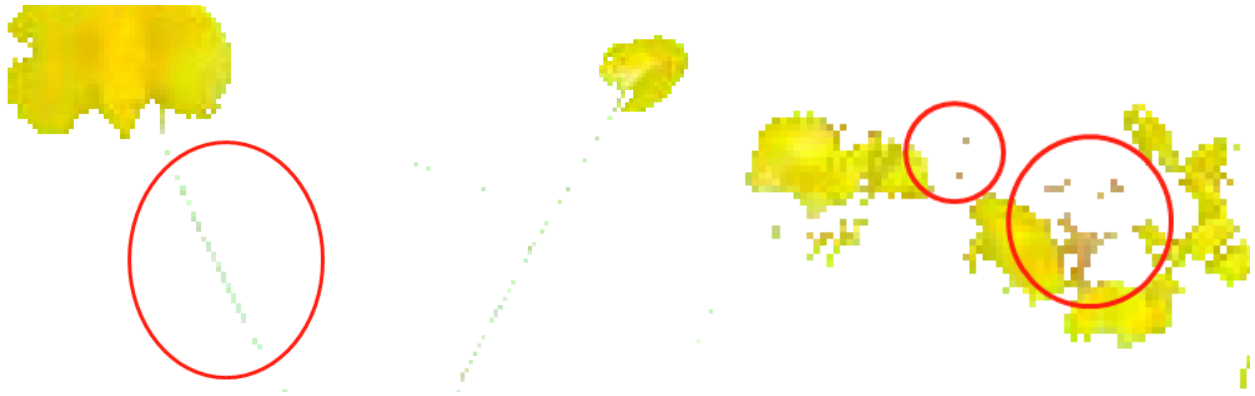
（i）部分程序代码

```
rgb1(:,:,1)=r5;
rgb1(:,:,2)=g5;
rgb1(:,:,3)=b5;
figure;subplot(141);imhist(rgb1);title('RGB-灰度直方图')
subplot(142);imhist(r5);title('R-灰度直方图');
subplot(143);imhist(g5);title('G-灰度直方图');
subplot(144);imhist(b5);title('B-灰度直方图');
%分割
[M,N]=size(r5);
for i=1:M
    for j=1:N
        if(r5(i,j)<0.75)
          rgb1(i,j,1)=255; rgb1(i,j,2)=255; rgb1(i,j,3)=255;
        end
    end
end
figure;imshow(rgb1);
```

（ii）处理结果

即在 B 层中当灰度值小于 0.75 时，将 R\G\B 三个图层中的灰度值都设置为 255，也就是变成白色，从而将花朵提取出来。但是这种直方图分割方法会残留一些绿色噪声（花朵颈部,如下图所示），还需后续处理。

（iii）直方图分割噪声展示



（3）基于 RGB 模型的颜色分割法

（i）部分程序代码

```
a=imread('777.jpg');
r=a(:,:,1); g=a(:,:,2); b=a(:,:,3);
b=a;
[m,n,d]=size(a);
for i=1:m
```

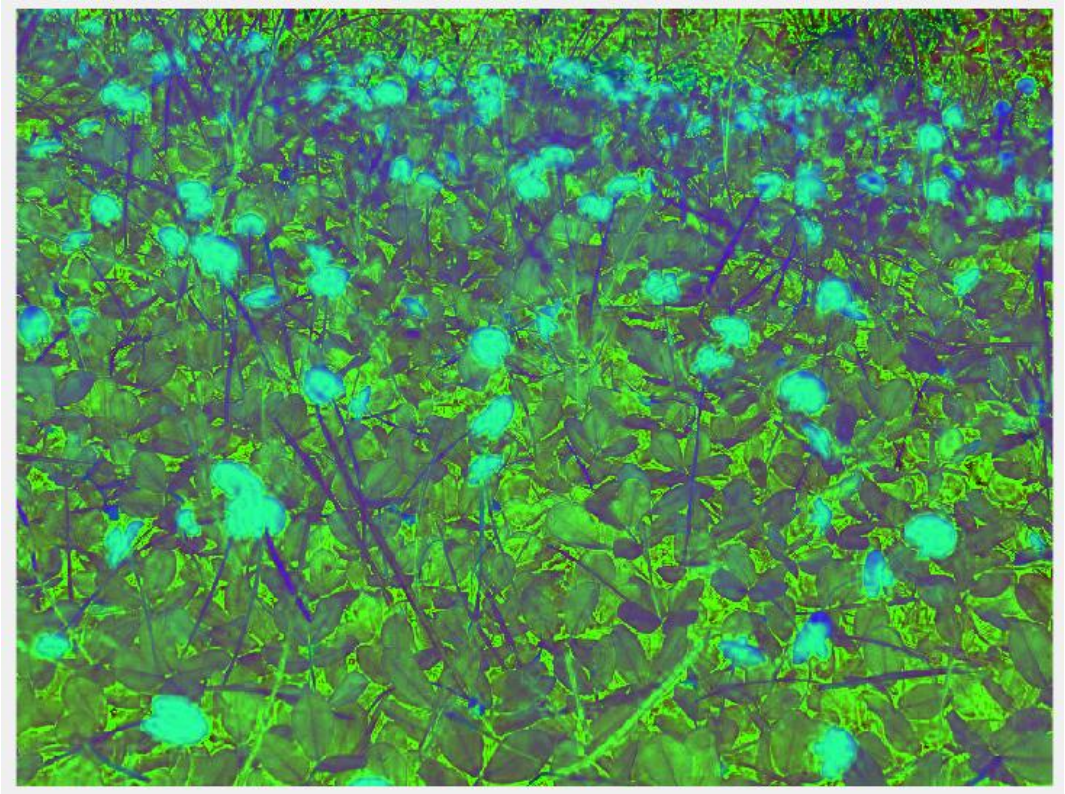
```
    for j=1:n
        if(a(i,j,1)>180&& a(i,j,1)<=255)&&(a(i,j,2)>140 &&
a(i,j,2)<=255)&&(a(i,j,3)>=0 && a(i,j,3)<75)
            b(i,j,1)=a(i,j,1);
            b(i,j,2)=a(i,j,2);
            b(i,j,3)=a(i,j,3);
        else
            b(i,j,1)=255;
            b(i,j,2)=255;
            b(i,j,3)=255;
        end
    end
end
figure,imshow(b);
```

（ii）处理结果

可以看出，利用 RGB 模型的颜色提取，减少了部分绿色噪声，但受设置阈值的影响，部分花心黄色也会损失，边缘仍有些许绿色噪声，但整体效果优于直方图分割。

（4）利用 HSI 参数模型进行分割

经过 rgb 到 his 的变换，图像变成下图结果



分析直方图可知，根据 H 分量有明显双峰特性，可以将目标从背景中分离出来

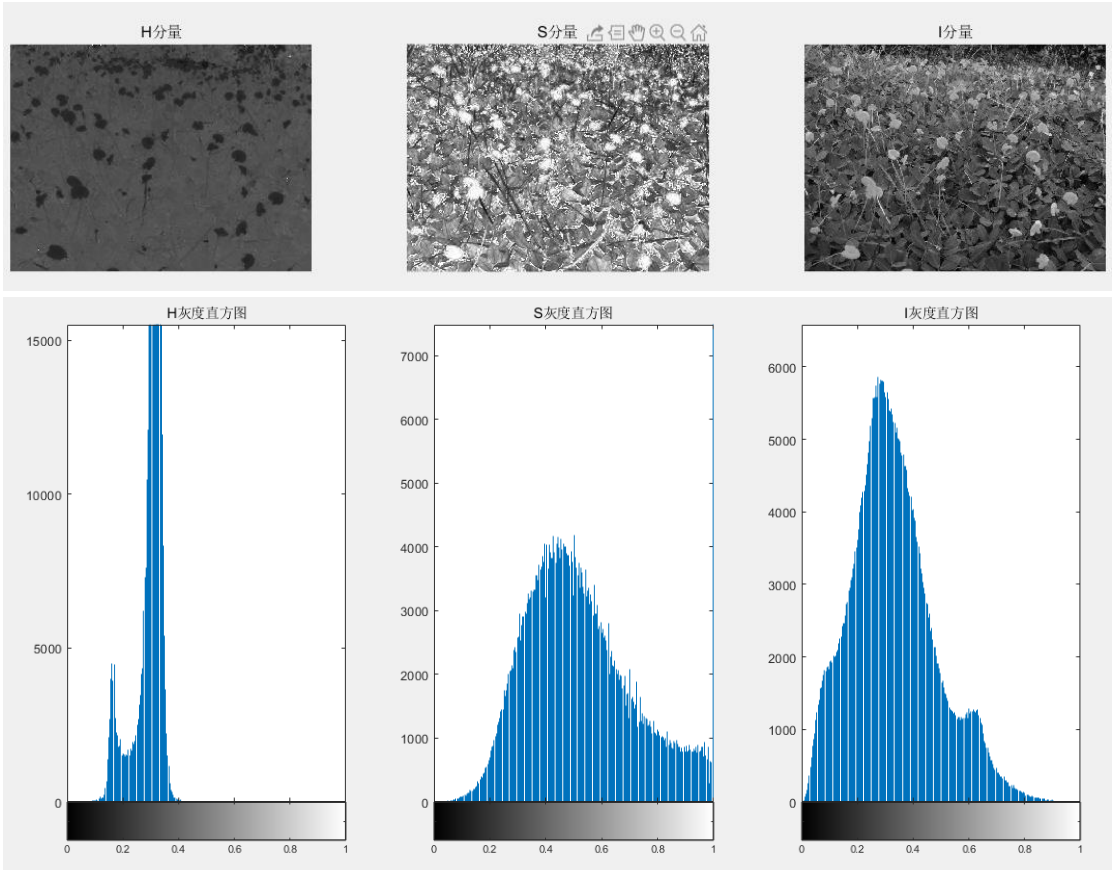
（i）部分程序代码

```
rgb = imread('777.jpg');
% 抽取图像分量rgb = im2double(rgb);
r = rgb(:, :, 1);
g = rgb(:, :, 2);
b = rgb(:, :, 3);
% 执行转换方程
num = 0.5*((r - g) + (r - b));
den = sqrt((r - g).^2 + (r - b).*(g - b));
theta = acos(num./(den + eps));
H = theta;
H(b > g) = 2*pi - H(b > g);
H = H/(2*pi);
num = min(min(r, g), b);
den = r + g + b;
den(den == 0) = eps;
S = 1 - 3.* num./den;
H(S == 0) = 0;
I = (r + g + b)/3;
%将3个分量联合成为一个HSI图像
hsi = cat(3, H, S, I);
hsi1 = H;
hsi2 = S;
hsi3 = I;
figure,subplot(131),imshow(hsi1);title('H分量');
```

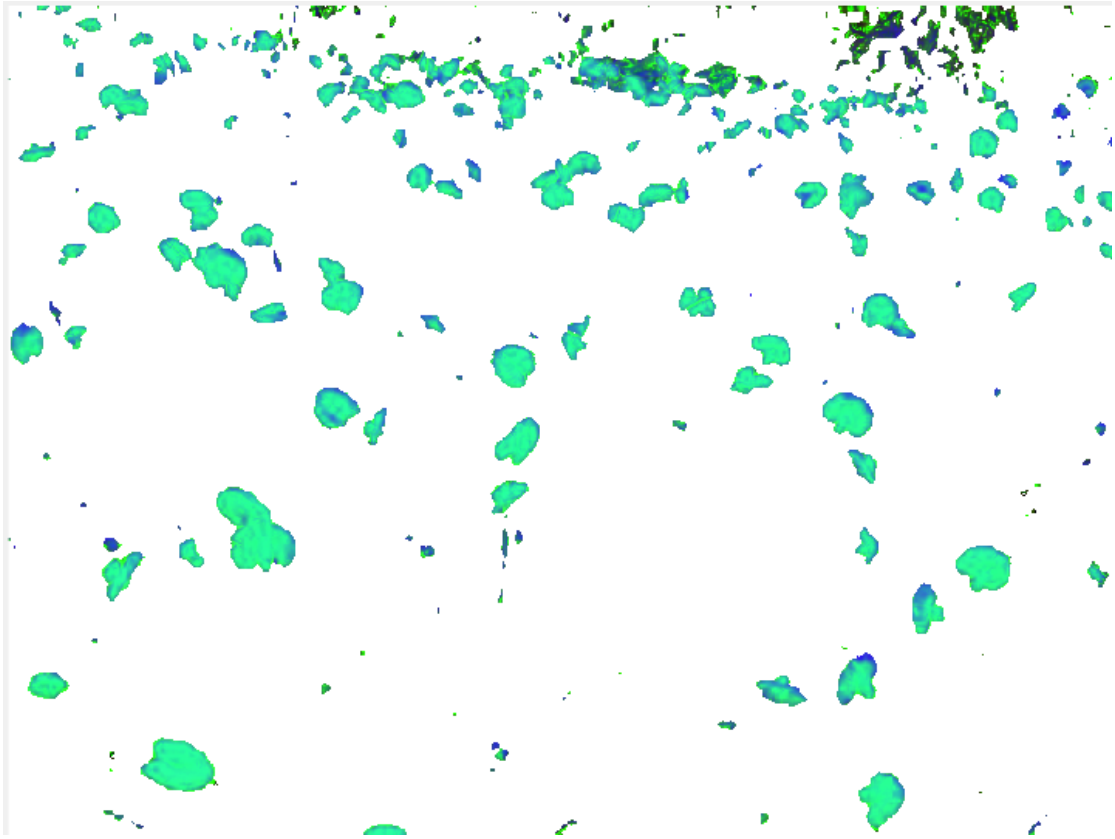
```
subplot(132),imshow(hsi2);title('S分量');
subplot(133),imshow(hsi3);title('I分量');
figure,subplot(131),imhist(hsi1);title('H灰度直方图');
subplot(132),imhist(hsi2);title('S灰度直方图');
subplot(133),imhist(hsi3);title('I灰度直方图');
%分割
rgb1(:,:,1)=hsi1;
rgb1(:,:,2)=hsi2;
rgb1(:,:,3)=hsi3;
[M,N]=size(hsi1);
for i=1:M
    for j=1:N
        if(hsi1(i,j)>0.20392)
            rgb1(i,j,1)=255; rgb1(i,j,2)=255; rgb1(i,j,3)=255;
        end
    end
end
figure;imshow(rgb1);
```

（ii）结果

各分量及其直方图展示



分割结果



对比 RGB 模型的分割结果，HIS 模型的分割明显有部分噪声残留，虽然局部未出现孔洞和缝隙，这点优于 RGB 模型，但整体效果不如 RGB 模型

6. 图形学处理

对图像的图形学处理，是对腐蚀、膨胀两个过程的不同顺序组合。先腐蚀后膨胀的开运算，可以去除图像噪声，但原有图像大小略有压缩；先膨胀后腐蚀的闭运算，可以填补前景物体上的小孔、连接临近物体。

由于 rgb 提取的黄色花朵有的部分有缺孔，有的边缘仍有绿色噪声，故可以用开、闭运算进行处理；由于黄花的形状不规则，故结构元素选择扁圆形 disk，球形 sphere，方形 square 等尝试，取处理效果最好的进行计算。

（1） 仅开运算

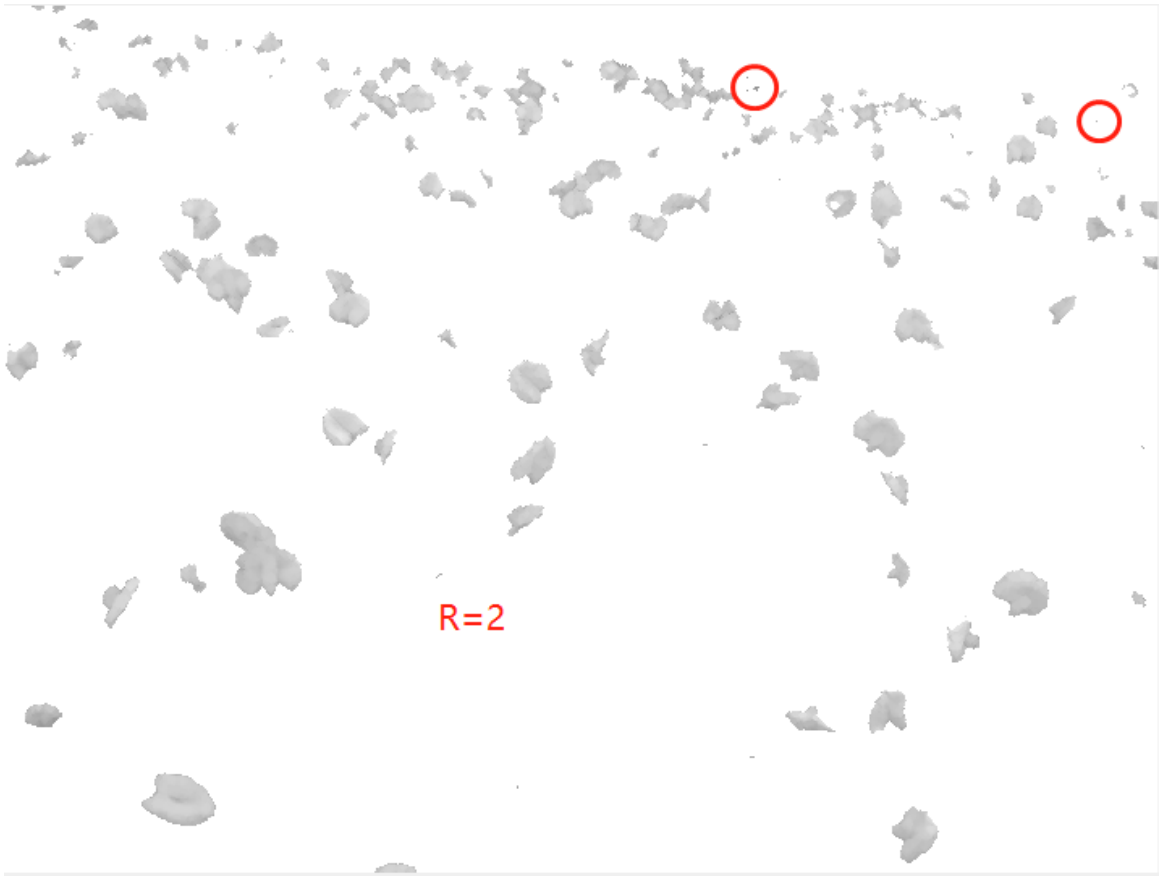
（i）部分程序代码：（上接分割部分）

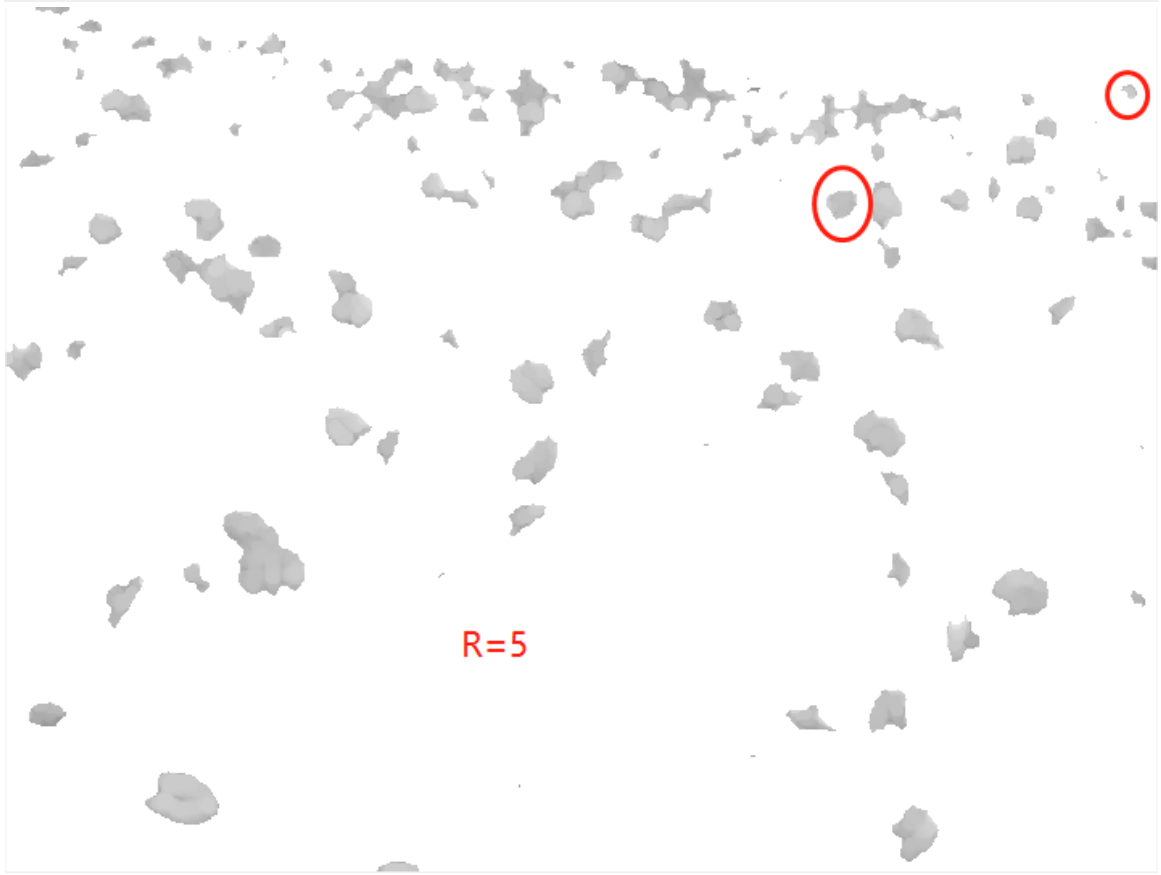
以 r=2 的扁圆形结构元素代码为例

```
go=rgb2gray(b);
se=strel('disk',2); %采用扁圆形结构元素，半径r=2
i3 =imopen(go,se) ;
figure;imshow(i3);
```

（ii）处理结果

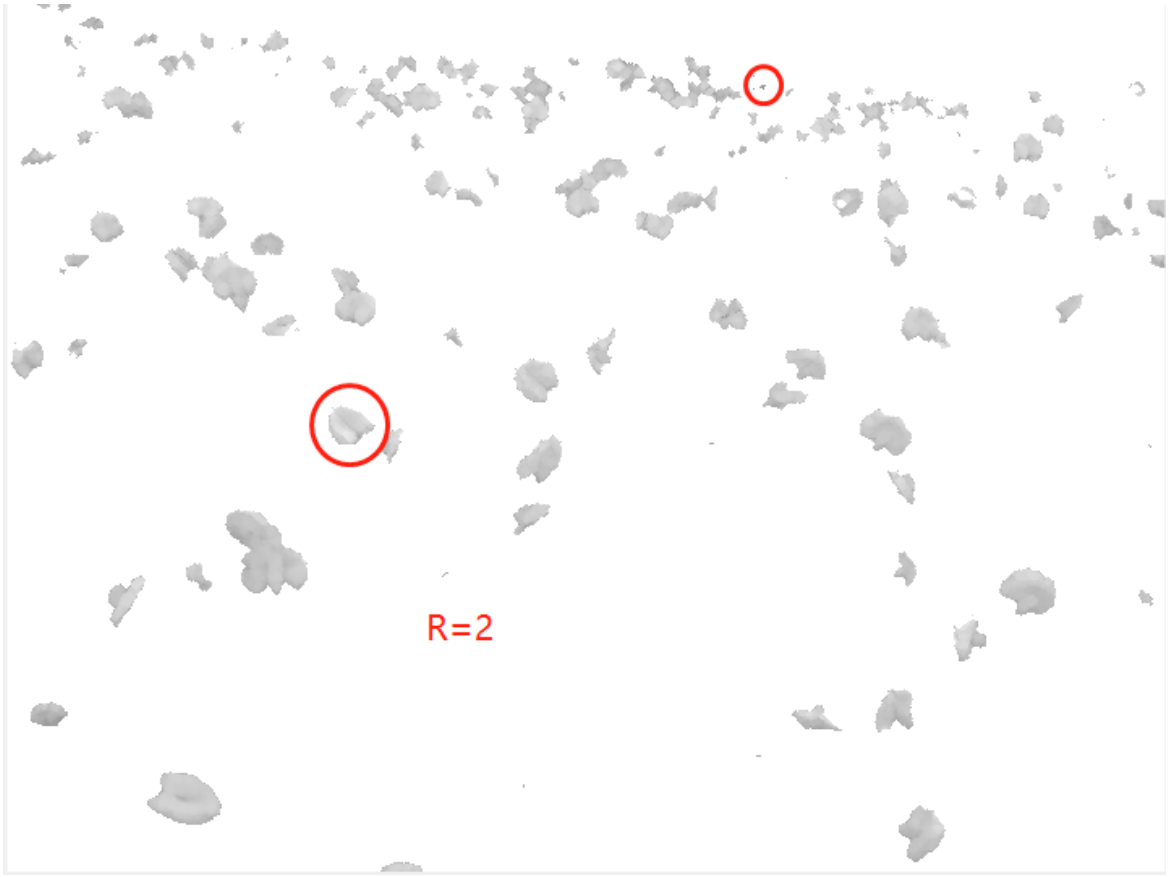
① 扁圆形disk：

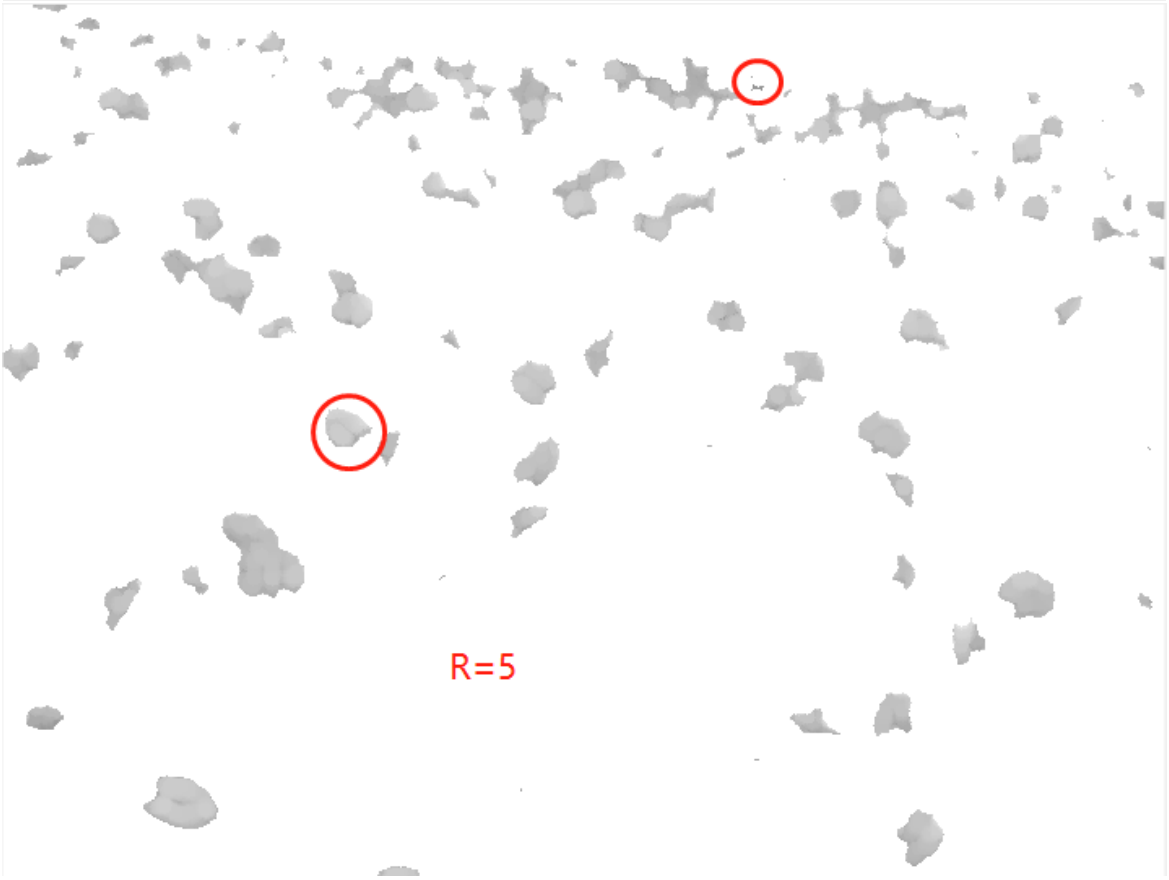




R=2 时，部分噪声没有去除，R=5 处理效果比较好，但过度填充了部分孔洞

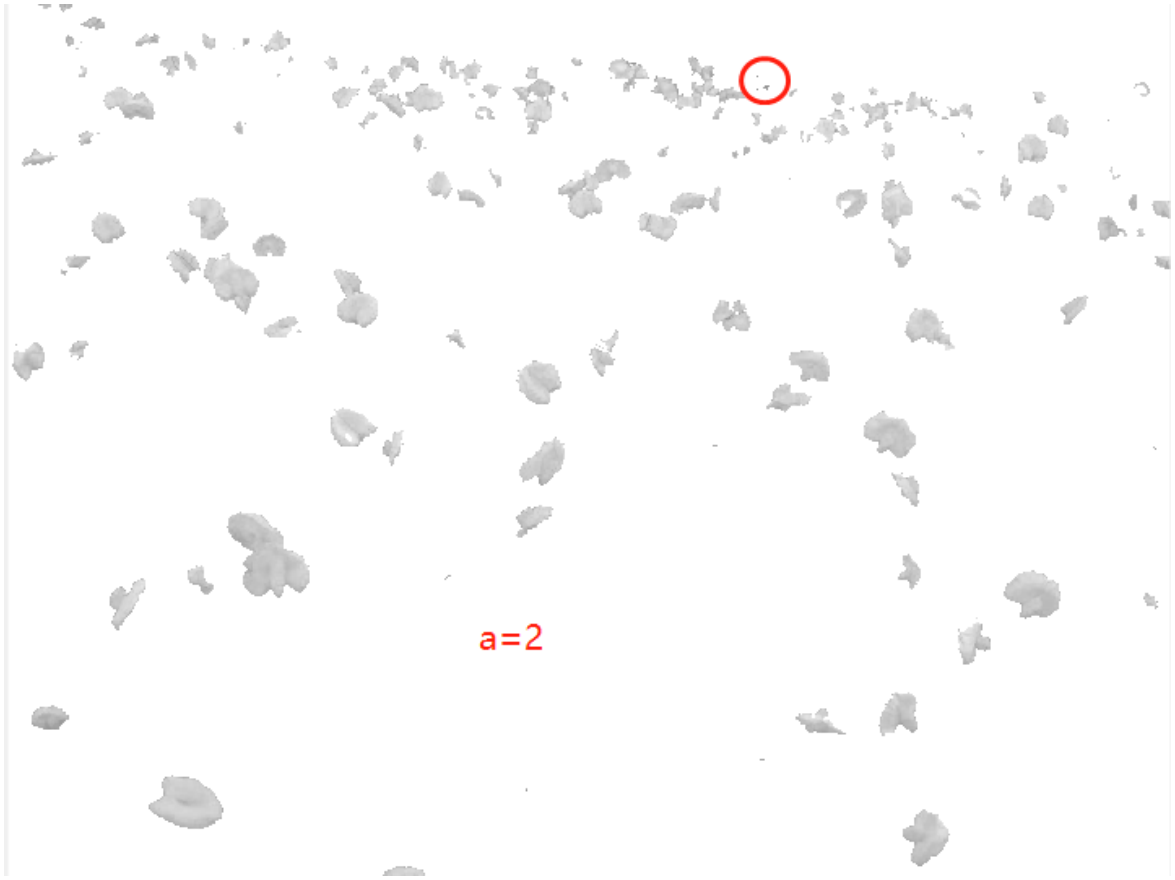
② 球形 sphere：

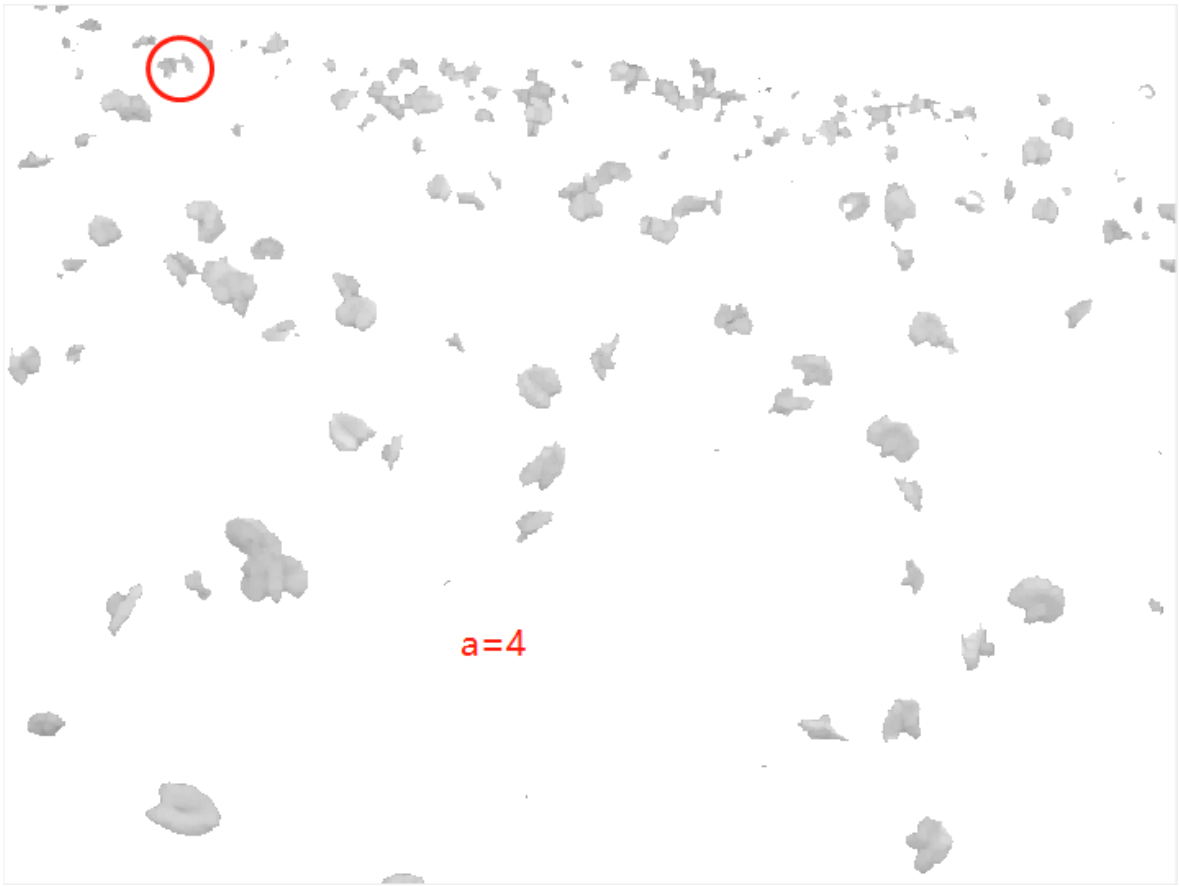




R=2 时，部分噪声没有去除，而且还填充了一些孔洞，随着 R 的增大，噪声被过度放大，放大了误差，处理效果不好

③ 方形 square：





采用方形结构元素，较好的保存了原图像信息，也能填充部分孔洞，整体效果比较好

（2） 闭运算

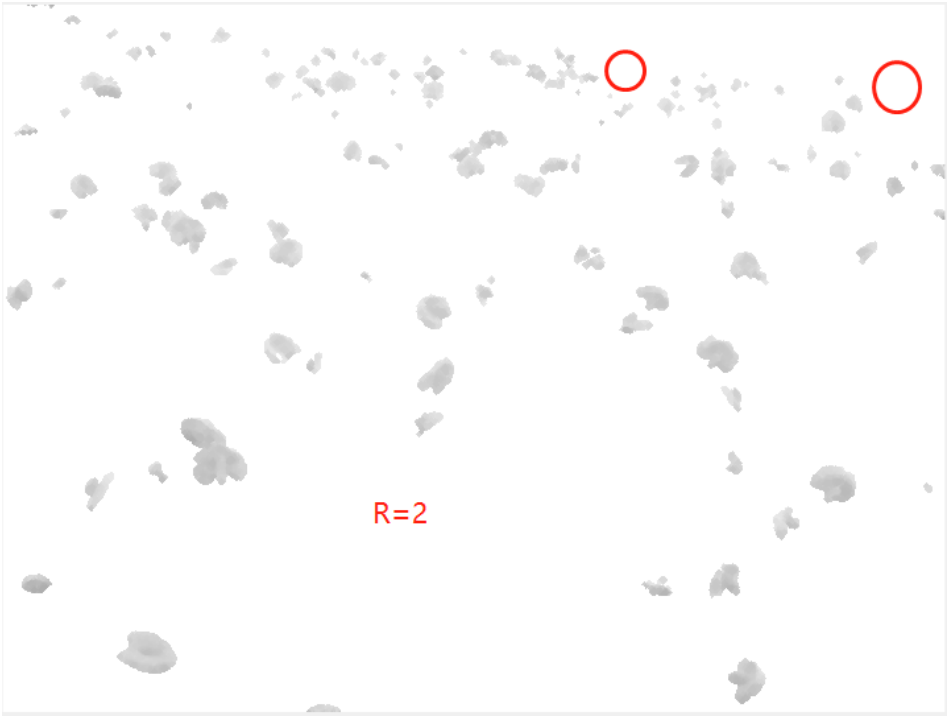
（i）部分程序代码：（上接分割部分）

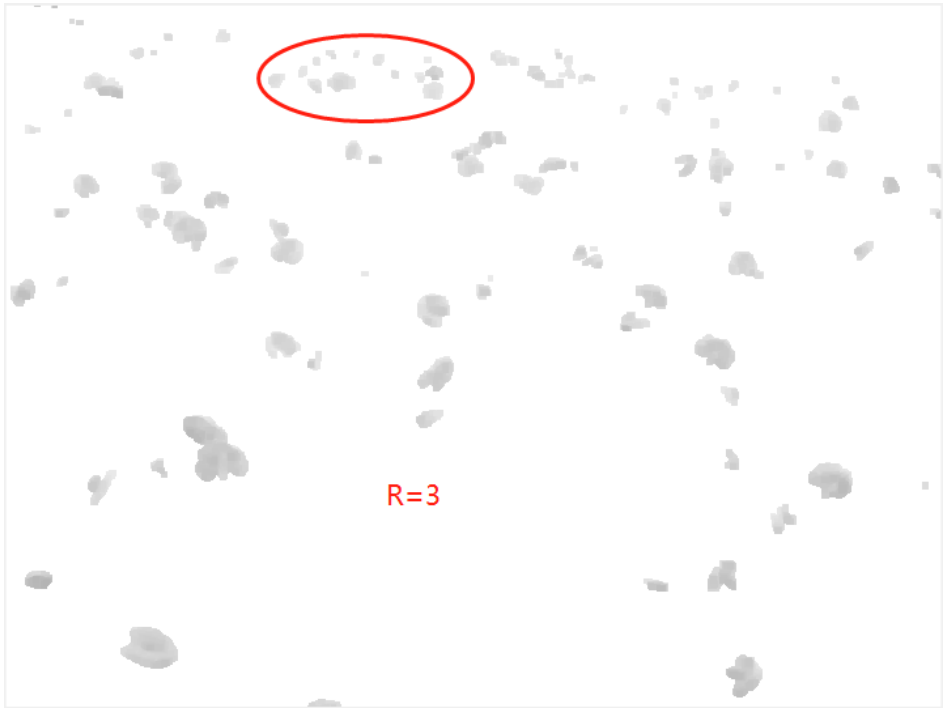
以 r=2 的扁圆形结构元素代码为例

```
go=rgb2gray(b);
se=strel('disk',2);
i3 =imclose(go,se) ;
figure;imshow(i3);
```

（ii）处理结果

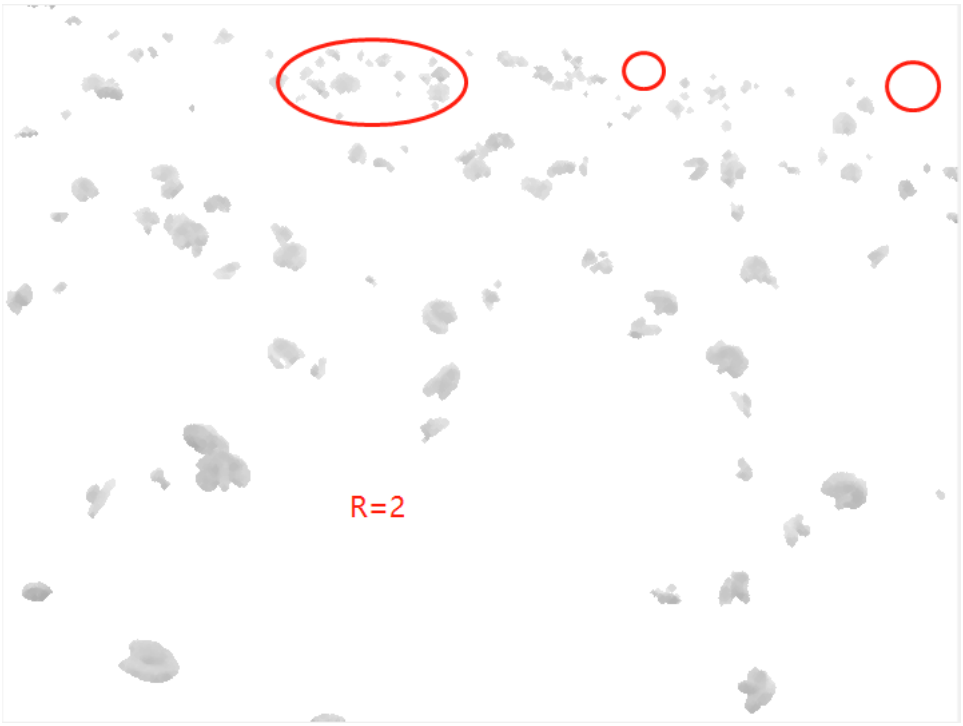
① 扁圆形 disk：





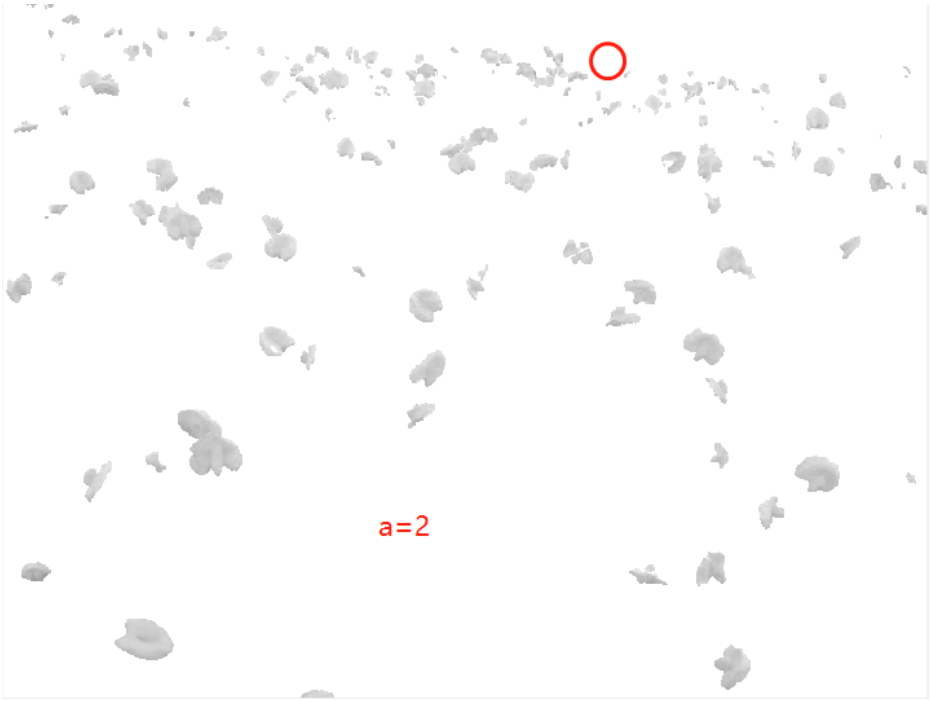
采用扁圆形结构元素的闭运算，明显的破坏了原有图像内容，处理效果不佳

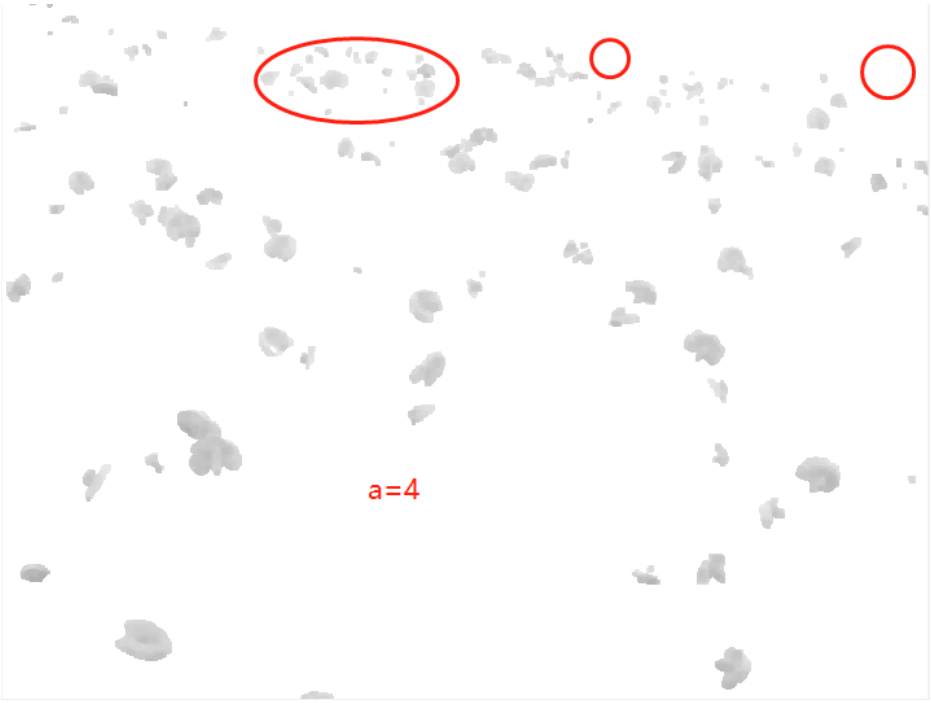
② 球形 sphere：



更换结构元素也出现上述问题

③ 方形 square：





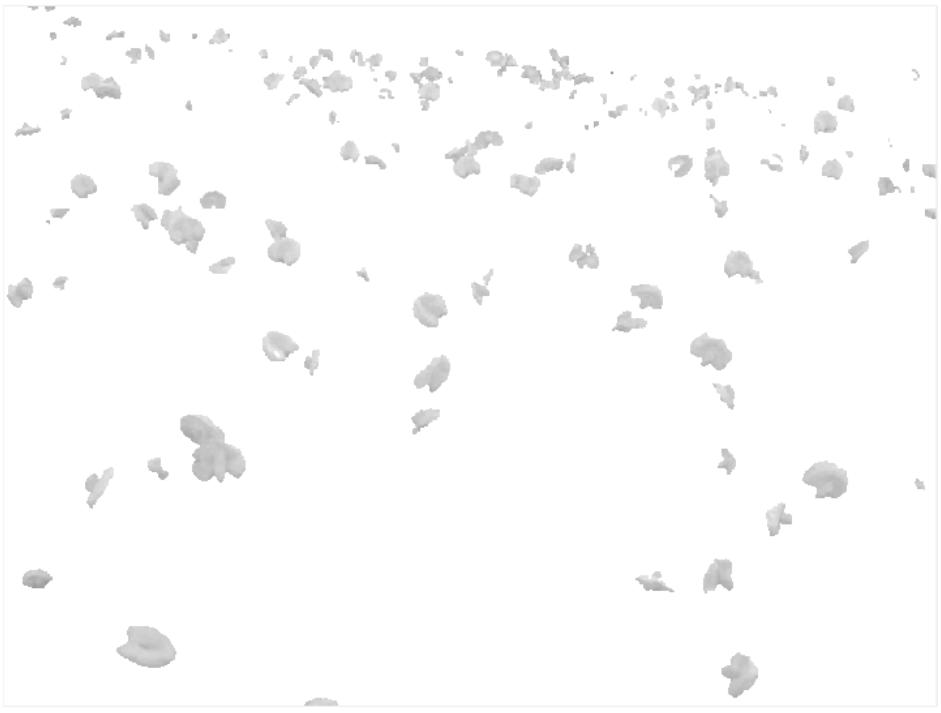
方形结构元素，a=2 时处理后去掉部分噪声，随着 a 的增大，也不会破坏图像原有内容；

（3） 先开运算后闭运算

（i）部分程序代码（以表现最好的方形结构元素为例）

```
go=rgb2gray(b);
se=strel('square',2);
i3 =imopen(go,se) ;%开运算
figure;imshow(i3);
se=strel('square',2);
i4 =imclose(i3,se) ;%闭运算
figure;imshow(i4);
```

（ii）处理结果



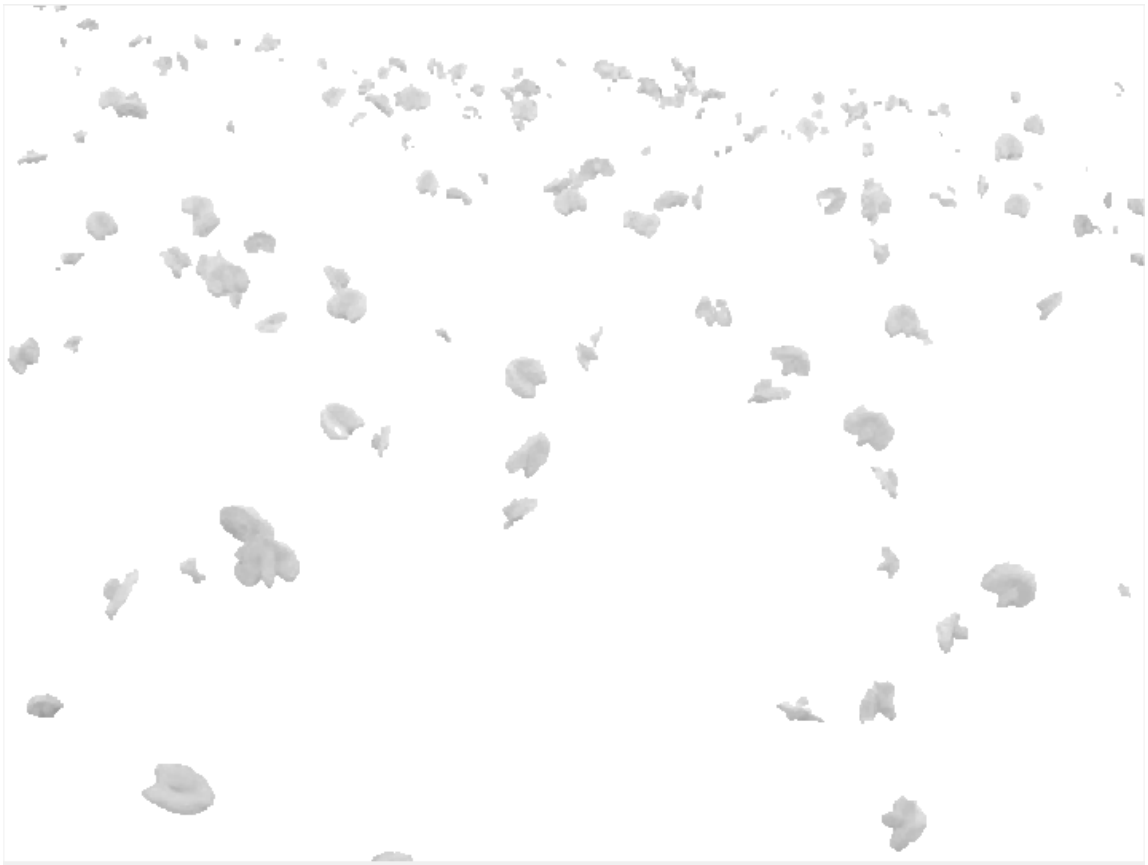
这种处理较好的保留了原有图像的细节信息，同时也去除了一些噪声，但有孔洞没有填补，总体处理效果好于单独的开闭运算处理；

（4） 先闭运算后开运算

（i）部分程序代码

```
go=rgb2gray(b);
se=strel('square',2);
i3 =imclose(go,se) ;%闭运算
figure;imshow(i3);
se=strel('square',2);
i4 =imopen(i3,se) ;%开运算
figure;imshow(i4);
```

（ii）处理结果



这种处理较好的保留了原有图像的细节信息，在填补了部分孔洞的同时，也去除了一些噪声，处理效果好于单独的开闭运算处理；

7. 图像特征提取

为了区分图中黄色花朵的不同大小，需要计算图中各区域的面积，并按面积数值归类出图中大、中、小三类花朵

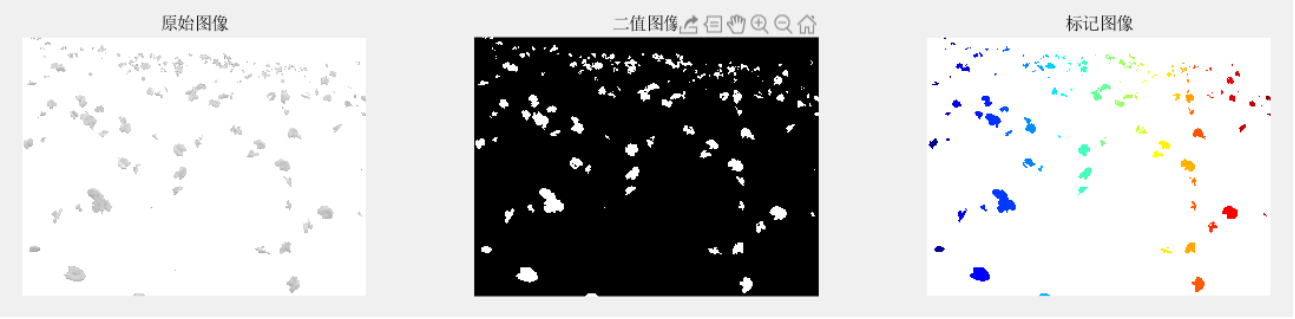
（1） 部分程序代码

```
I=rgb2gray(b);
figure,subplot(131),imshow(I),title('原始图像');% 显示该灰度图像
B=imbinarize(I,graythresh(I));% 对该灰度图像进行自动阈值
```

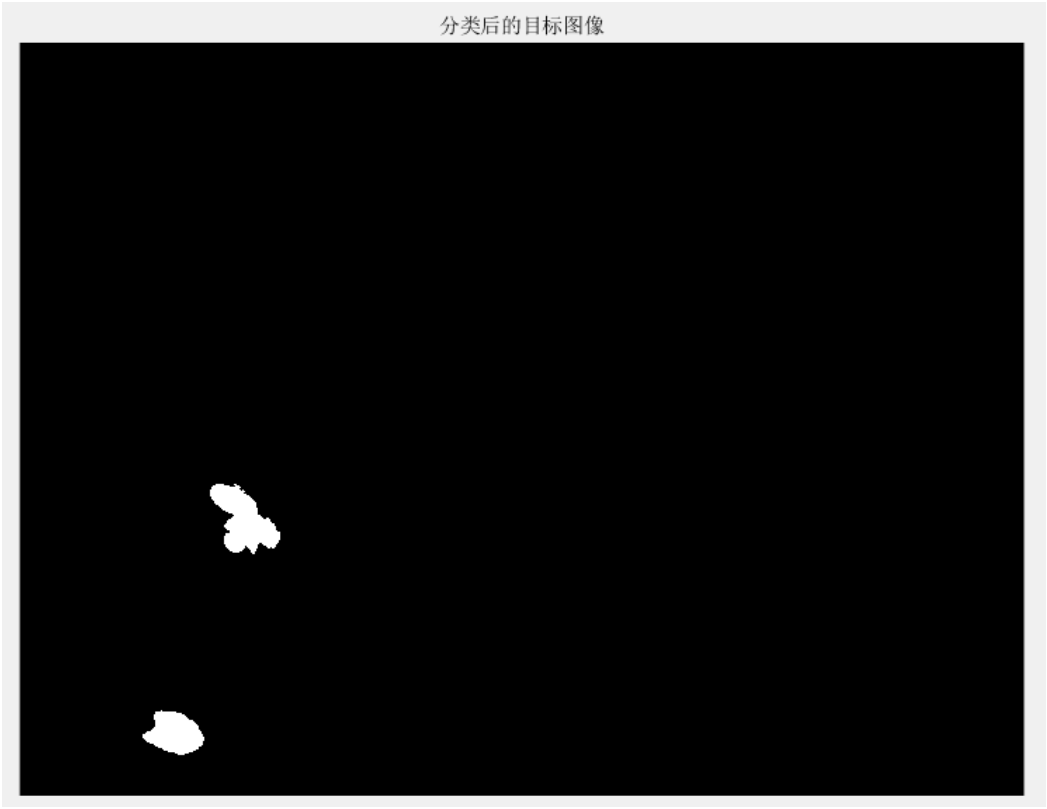
```
B=~B;% 二值图像取反·´
subplot(132),imshow(B),title('二值图像');
B=imerode(B,strel('disk',1));% 腐蚀运算
[x,n]=bwlabel(B);% 区域标记
stats=regionprops(x, 'all');% 求出面积、重心等19个特征参数
X=label2rgb(x); %对标记图像做伪彩色变换
subplot(133),imshow(X),title('标记图像');
B1=zeros(size(B));% 设定并初始化一个与二值图像B大小相同的图像文件B1
for i=1:n
    disp(stats(i).Area);
end

for i=1:n
    ar=stats(i).Area;
    if ar>1000 %此部分设置阈值，用于区分大中小三类
        [p,q]=find(x==i);
        B2=bwselect(B,q,p,8);% 获取只有该区域的二值图像
        B1=(B1)|(B2);
    end
end
figure,imshow(B1);title('分类后的目标图像')
```

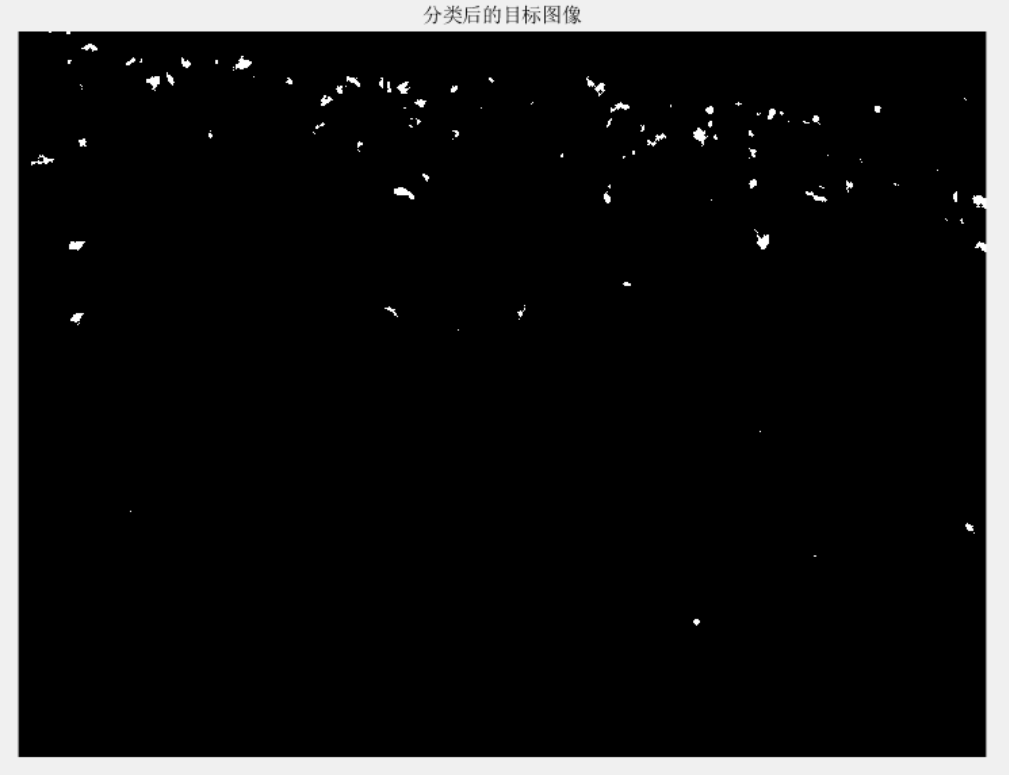
（2） 结果（一部分）

```
   269   122    13    10
   356    33    52   121
   316   729   235   638
     1  1694     2    10
    28    12   361   449
     1     1   154   292
  1150     6    88    12
     5   210    17    24
    78     1    32     1
   220     2    15    38
    35    81   185     7
   388   122     1   142
```



由于面积值最大 1000 以上，小的只有 100 甚至 10 以下，大多数在 100 到 1000 之间，故以此为依据划分成大中小三类， 图像分别为





7. 计算结果

首先计算所有黄色花朵在图中的占比

（1）初次分割后计算

```
[H,L] = size(b);
Num=0; %统计白色像素个数
for i=1:H
```

```
  for j=1:L
     if b(i,j)==255 %记录白色像素
      Num=Num+1;
     end
  end
end
t1=Num/(H*L);
```

即统计图中白色像素个数

黄花占比计算结果为

```
t1 =

    0.9464
```

（2）图形学处理后计算

```
[H,L] = size(i4);
Num=0;
for i=1:H
  for j=1:L
     if i4(i,j)==255
      Num=Num+1;
     end
  end
end
t2=Num/(H*L);
```

黄花占比计算结果为

```
t2 =

    0.9453
```

再计算大中小三类花朵占比

```
[H,L] = size(B1);
Num=0;
for i=1:H
  for j=1:L
     if B1(i,j)==0
      Num=Num+1;
     end
  end
end
t3=Num/(H*L);%改变参数，计算方法相同，1-t3为大花朵占比，1-t4为中花朵占比，1-t5为噪点占比
```

```
t3 =

    0.9941


t4 =

    0.9708


t5 =

    0.9954
```

8．最终计算，得出结论

在黄色花朵图像后期处理中，程序运算可知，图片尺寸 H*L=600*800=480000，白色像素占总图像的 0.9453，所以黄色花朵在图像中占比为 1-0.9453=0.0547。

在分类计算结果中，大花朵占比 1-0.9941=0.0059，中花朵占比 1-0.9708=0.0292，小花朵占比 1-0.9954=0.0046。

四、课程感悟

之前的课外学习中接触过 python 图像处理的有关库，因此对这门课很感兴趣。通过这门课，我了解到了图像处理的原理、算法，入门了一直想学的 Matlab，还能深入体会信号与系统等其他专业课知识的具体应用，真的让我收获满满。

此外，老师上课之前会复习巩固，课堂之中会请同学交流，在这个过程中加深了对知识的认识，我觉得这种教学方法很 nice，当然老师也很 nice，依旧延续了信电老师们严谨不失亲切的特点，让我每节课都能全神贯注。

做作业的过程比较长久，有空就跑一跑试一试，虽然觉得自己做的这个题目有点小儿科，但是还是通过自己的思考，解决了不少问题，这个过程成就感超强，毕竟浮于表面都是风光，沉下心来自有答案。

写到这就算大作业结束啦，但是考试还要继续努力鸭，卷出好成绩！

感谢相遇，祝老师工作顺利，天天开心！