数字图像处理

实验三 数字图像的空间域滤波（平滑和锐化）

专　　业：计算机科学与技术

年　　级：2020 级

学　　号：20201060330

姓　　名：胡诚皓

2022 年 10 月 13 日

实验三：数字图像的空间域滤波

1．实验目的

1) 掌握图像滤波的基本定义及目的。
2) 理解空间域滤波的基本原理及方法。
3) 掌握进行图像的空域滤波的方法。

2. 实验原理

1. 空间域增强

空间域滤波是在图像空间中借助模板对图像进行领域操作，处理图像每一个像素的取值都是根据模板对输入像素相应领域内的像素值进行计算得到的。各种空域滤波器根据功能主要分为平滑滤波器和锐化滤波器。平滑可用低通来实现，平滑的目的可分为两类：一类是模糊，目的是在提取较大的目标前去除太小的细节或将目标内的小肩端连接起来；另一类是消除噪声。锐化可用高通滤波来实现，锐化的目的是为了增强被模糊的细节。结合这两种分类方法，

空间滤波器都是基于模板卷积，其主要工作步骤是：

1) 将模板在图中移动，并将模板中心与图中某个像素位置重合；
2) 将模板上的系数与模板下对应的像素相乘；
3) 将所有乘积相加；
4) 将和（模板的输出响应）赋给图中对应模板中心位置的像素。

2. 平滑滤波器

1) 线性平滑滤波器

低通平滑滤波器也称为均值滤波器，这种滤波器的所有系数都是正数，对3×3的模板来说，最简单的是取所有系数为1，为了保持输出图像任然在原来图像的灰度值范围内，模板与象素邻域的乘积都要除以9。

MATLAB 提供了fspecial 函数生成滤波时所用的模板，并提供filter2 函数用指定的滤波器模板对图像进行运算。函数fspecial 的语法格式为：

```
h=fspecial(type);
h=fspecial(type,parameters);
```

其中参数 type 指定滤波器的种类，parameters 是与滤波器种类有关的具体参数。

类型	参数	说明
average	hsize	均值滤波，若邻域为方阵，则 hsize 为标量，否则由两元素向量 hsize 指定行数和列数
disk	radius	有(radius*2+1)个边的圆形均值滤波器
gaussian	hsize,sigma	标准差为 sigma，大小为 hsize 的高斯低通滤波器
laplacian	alpha	系数由 alpha(0.0-1.0)决定的二维拉普拉斯滤波
log	hsize,sigma	标准偏差为 sigma，大小为 hsize 的高斯滤波旋转对称拉氏算子
motion	len,theta	按角度 theta 移动 len 个像素的运动滤波器
prewitt	无	近似计算垂直梯度的水平边缘强调算子
sobel	无	近似计算垂直梯度光滑效应的水平边缘强调算子
unsharp	alpha	根据 alpha 决定的拉氏算子创建的掩模滤波器

表格 3.1 MATLAB **中预定义的滤波器种类**

MATLAB 提供了一个函数imnoise 来给图像增添噪声，其语法格式为：

```
J=imnoise(I,type);
J=imnoise(I,type,parameters);
```

参数type 指定噪声的种类，parameters 是与噪声种类有关的具体参数。参数的种类见表2.2。

类型	参数	说明
gaussian	m,v	均值为m，方差为v的高斯噪声
localvar	v	均值为0，方差为v的高斯白噪声
passion	无	泊松噪声
salt pepper	无	椒盐噪声
speckle	v	均值为0，方差为v的均匀分布随机噪声

表格 3.2 **噪声种类及参数说明**

2) 非线性平滑滤波器

中值滤波器是一种常用的非线性平滑滤波器，其滤波原理与均值滤波器方法类似，但计算的非加权求和，而是把领域中的图像的象素按灰度级进行排序，然后选择中间值作为输出象素值。

MATLAB 提供了medfilt2 函数来实现中值滤波，其语法格式为：

```
B=medfilt2(A, [m n]);
B=medfilt2(A);
```

其中，A 是原图象，B 是中值滤波后输出的图像。[m n]指定滤波模板的大小，如果不指定m和n,则模板大小默认为3×3。

3. 锐化滤波器

图像平滑往往使图像中的边界、轮廓变得模糊，为了减少这类不利效果的影响，需要利用图像锐化技术，使图像的边缘变得清晰。

最常用的线性锐化滤波器。这种滤波器的中心系数都是正的，而周围的系数都是负的，所有的系数之和为1。对3×3 的模板来说，典型的系数取值（8领域整数）为：

[-1 -1 -1;

-1 9 -1;

-1 -1 -1]

事实上这是拉普拉斯算子（取负数）加上原图像（即和8个相邻像素的差分结果加上原图像）。语句h=-fspecial('laplacian',0.5)得到的拉普拉斯算子为：

h = -0.3333 -0.3333 -0.3333

-0.3333 2.6667 -0.3333

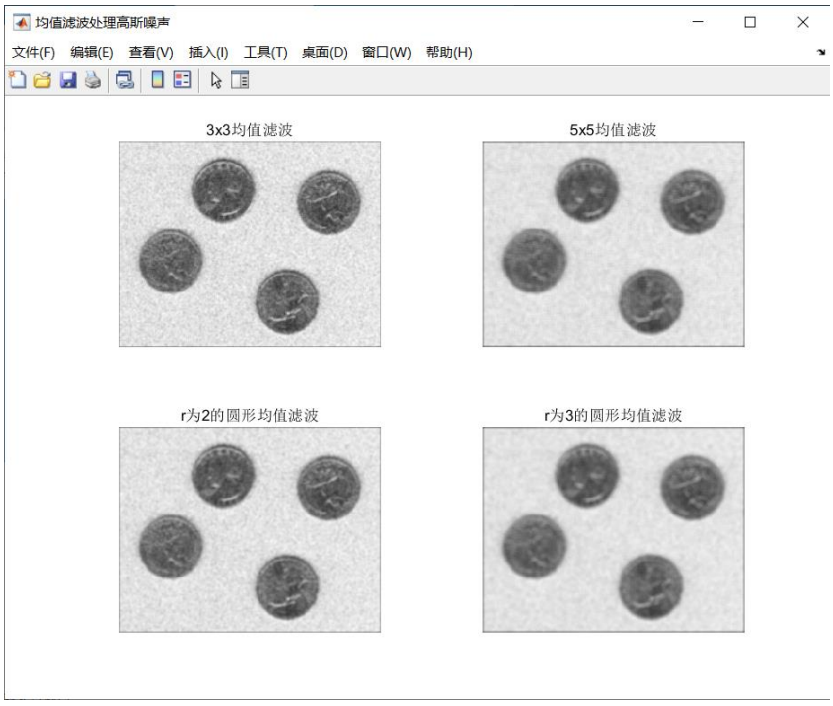
-0.3333 -0.3333 -0.3333

3、实验内容与要求

1. 平滑空间滤波

1) 读入一幅灰度图像，给这幅图像分别加入椒盐噪声和高斯噪声后并显示。

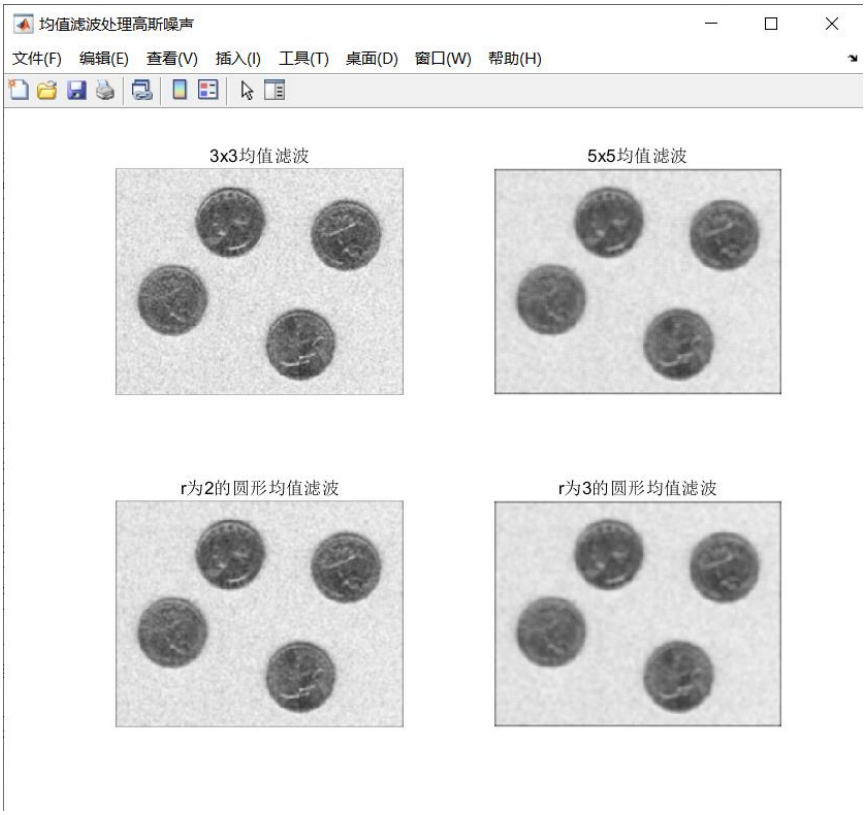
```
% 加入噪声
jiaoyan = imnoise(origin_std, "salt & pepper");
gaosi = imnoise(origin_std, "gaussian");
```

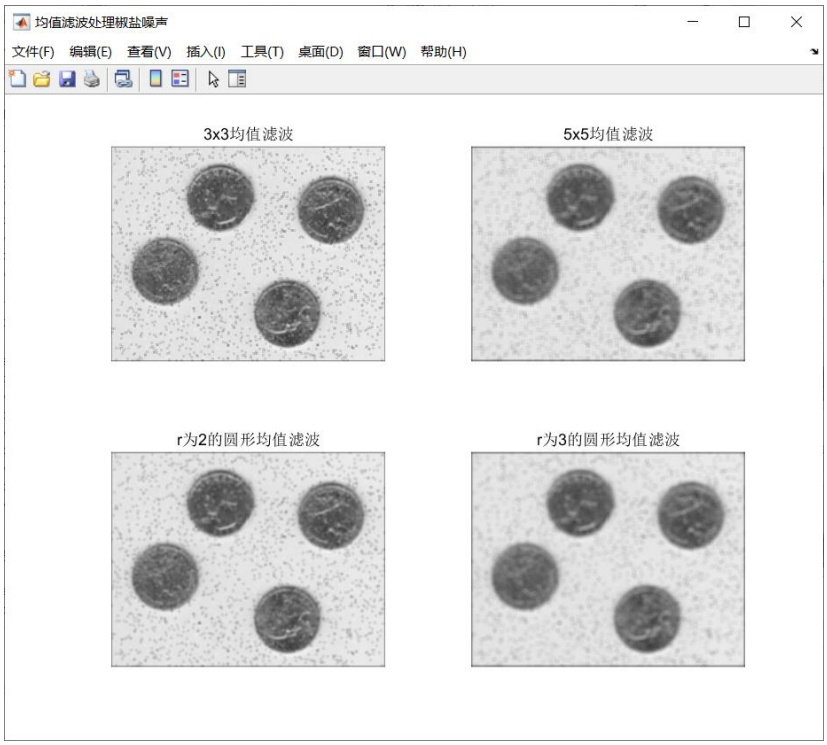


2) 对加入噪声图像选用不同的平滑（低通）模板做运算，对比不同模板所形成的效果，要求在同一窗口中显示。

```
% 3x3 均值滤波
avg3x3 = fspecial('average', 3);
```

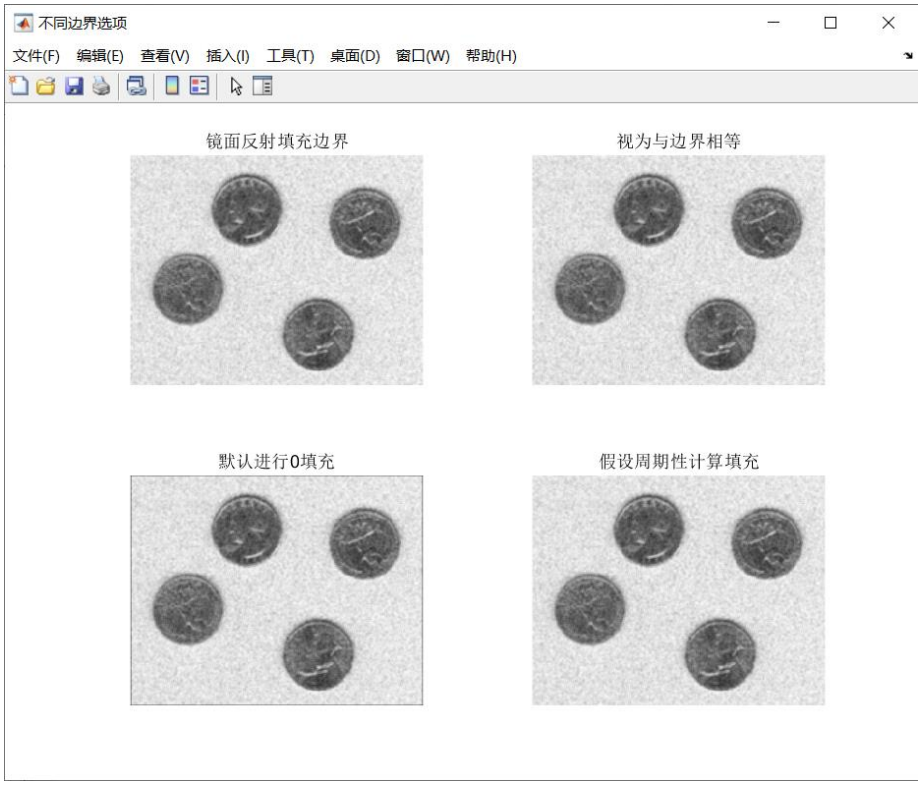
```
% 5x5 均值滤波
avg5x5 = fspecial('average', 5);

% r 为 2 的圆形均值滤波
round2 = fspecial('disk', 2);
% r 为 3 的圆形均值滤波
round3 = fspecial('disk', 3);
```





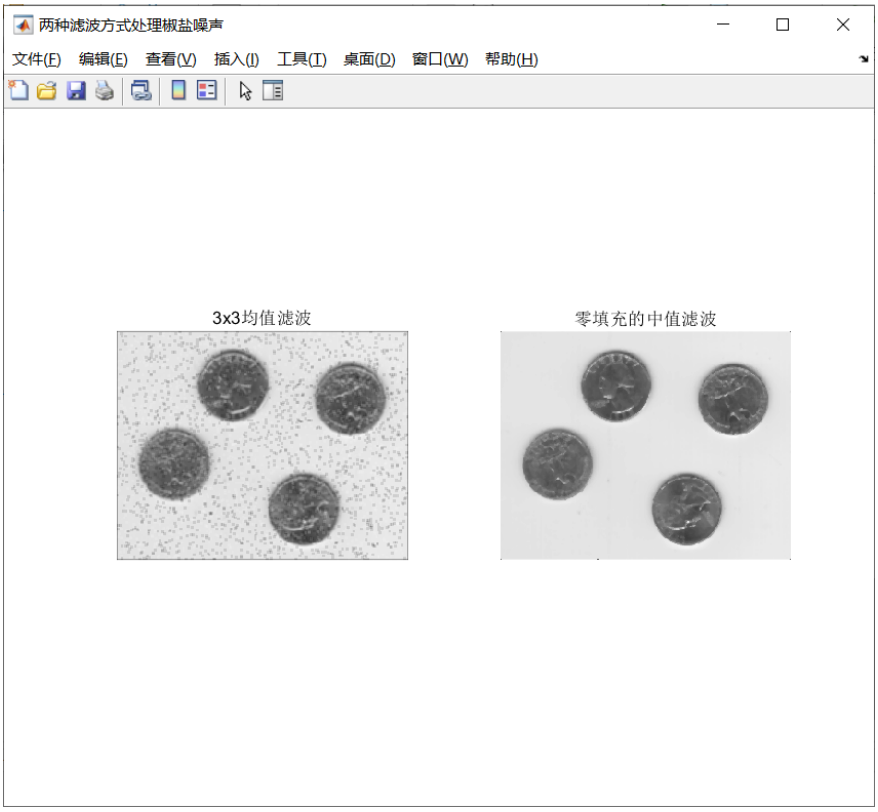
3) 使用函数 imfilter 时，分别采用不同的填充方法（或边界选项，如零填充、'replicate'、'symmetric'、'circular'）进行平滑滤波，显示处理后的图像。

```
imshow(imfilter(gaosi, avg3x3, 'symmetric')); title('镜面反射填充边界');
imshow(imfilter(gaosi, avg3x3, 'replicate')); title('视为与边界相等');
imshow(imfilter(gaosi, avg3x3)); title('默认进行 0 填充');
imshow(imfilter(gaosi, avg3x3, 'circular')); title('假设周期性计算填充');
```



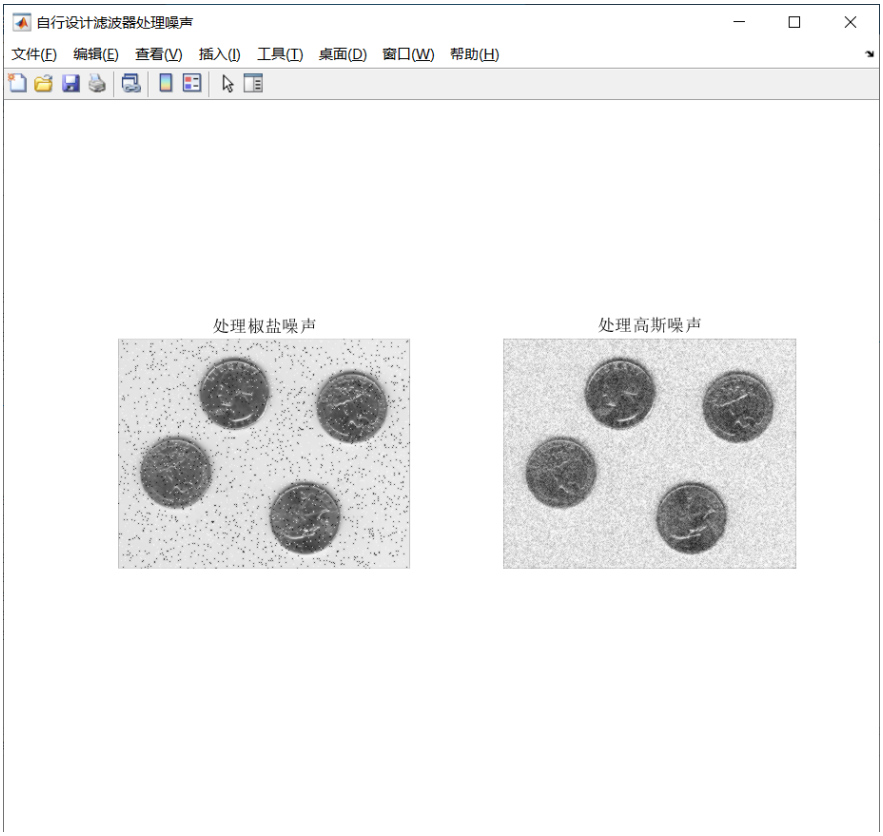
4) 对加入椒盐噪声的图像分别采用均值滤波法，和中值滤波法对有噪声的图像做处理，要求在同一窗口中显示结果。

```
imshow(imfilter(jiaoyan, avg3x3)); title('3x3 均值滤波');
imshow(medfilt2(jiaoyan)); title('零填充的中值滤波');
```



5) 自己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。

```
core = [0.05 0.05 0.05;
        0.05 0.6  0.05 ;
        0.05 0.05 0.05];
```



2. 锐化空间滤波

1) 读入一幅图像，采用 3×3 的拉普拉斯算子对其进行滤波。

```
lap = [1 1  1;
       1 -8 1;
       1 1  1];
imfilter(origin_std, -lap);
```

2) 编写自定义函数 w = genlaplacian(n)，自动产生任一奇数尺寸 n 的整数拉普拉斯算子，例如如 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]

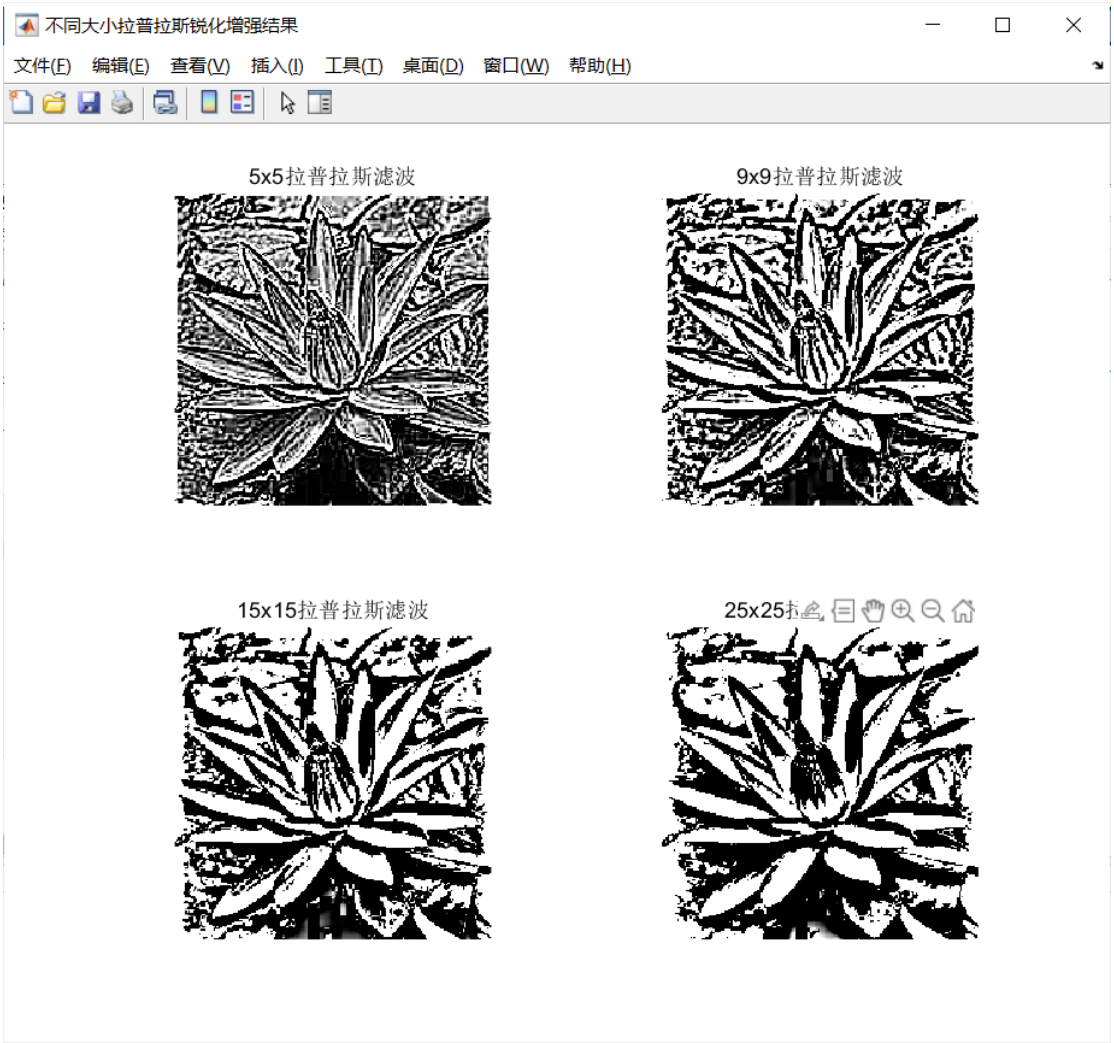
```
function laplacian_core = genlaplacian(n)
    error = MException("myComponent:inputError", '参数不能为偶数');
    if (rem(n, 2) == 0)
        laplacian_core = nan;
        throw(error);
    end
    laplacian_core = ones(n, n);
    laplacian_core(ceil(n/2), ceil(n/2)) = -n*n+1;
end
```

```
命令行窗口
>> genlaplacian(5)

ans =

     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

fx >>
```

3) 分别采用 5×5，9×9，15×15 和 25×25 大小的拉普拉斯算子对图像进行锐化滤波，并利用式 $g(x,y)=f(x,y)-\nabla^2 f(x,y)$ 完成图像的锐化增强，观察其有何不同，要求在同一窗口中显示。

```
w1 = -genlaplacian(5);
w2 = -genlaplacian(9);
w3 = -genlaplacian(15);
w4 = -genlaplacian(25);

% 先得到边缘图像
wf1 = imfilter(flower_std, w1);
wf2 = imfilter(flower_std, w2);
wf3 = imfilter(flower_std, w3);
wf4 = imfilter(flower_std, w4);
```

```
% 叠加上原图以展示最终结果
imshow(flower_std+wf1);
imshow(flower_std+wf2);
imshow(flower_std+wf3);
imshow(flower_std+wf4);
```



用于锐化的源图像(作为输入)

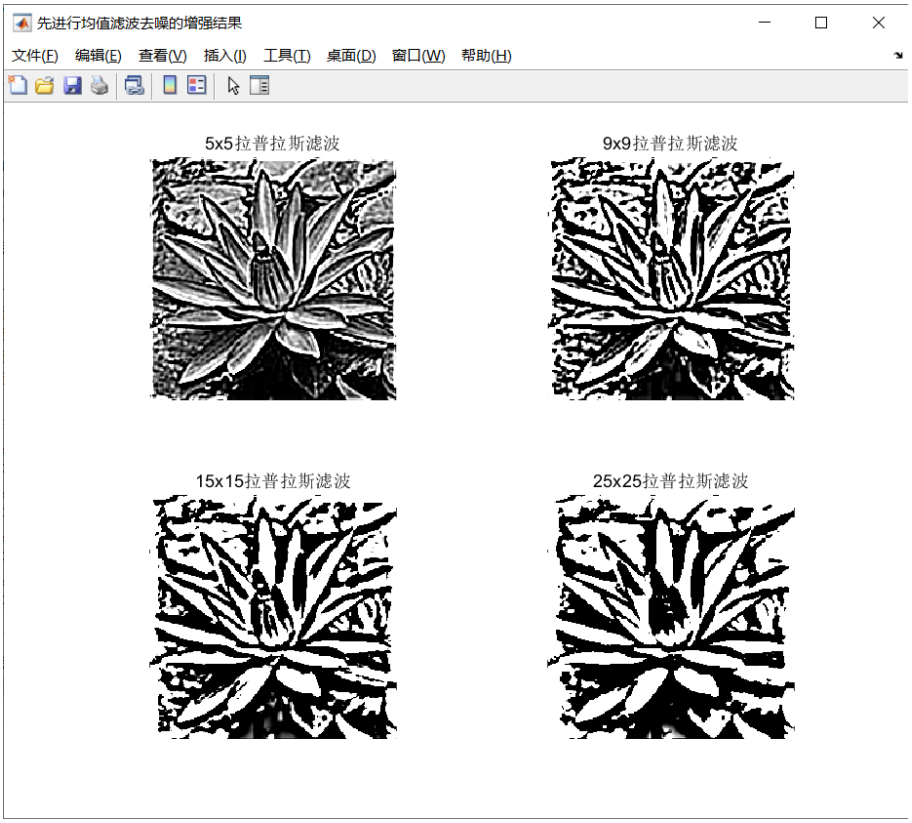
4、实验分析：

在本次实验中，对图像平滑去噪和边缘锐化增强进行了实践。去噪是一个非常常用的操作，在进行其他图像处理的操作之前往往会首先对原图像进行去噪，以优化后面处理的效果。重点使用拉普拉斯算子对图像进行边缘锐化的操作。拉普拉斯算子的本质是通过二阶导数得到图像像素值突变的边缘部分，事实上也是对边缘提取后再叠加到原图像上。

使用 3*3 的拉普拉斯锐化模板直接对荷花图像进行边缘锐化增强后，会发现图片的边缘部分会变得特别不自然，图片的整体形状甚至都有些破坏。这其中有两个原因，一个是原图本身的噪声过大，另一个是拉普拉斯算子自身的缺陷。

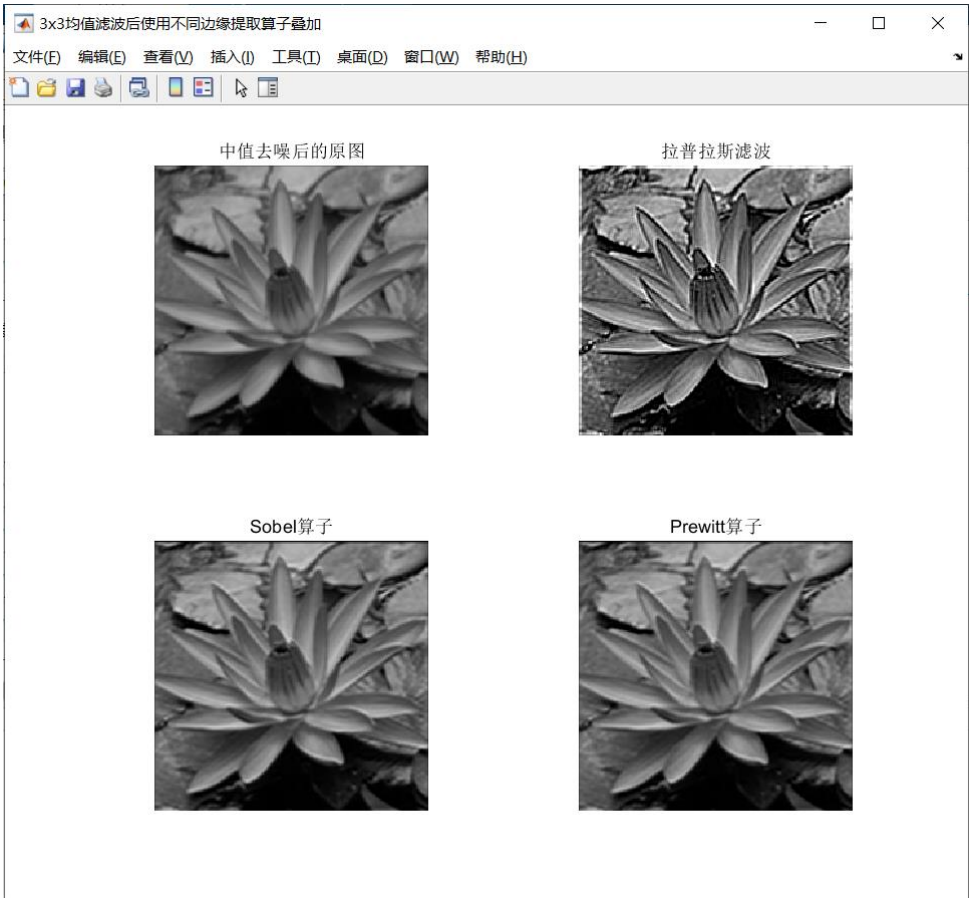
针对第一个问题，此处使用 3*3 的均值滤波先对原图像进行去噪，再分别使用 5*5、9*9、15*15、25*25 的拉普拉斯锐化模板进行边缘锐化增强。

```
dec_flower = imfilter(flower_std, fspecial('average', 3));
wf1 = imfilter(dec_flower, w1);
wf2 = imfilter(dec_flower, w2);
wf3 = imfilter(dec_flower, w3);
wf4 = imfilter(dec_flower, w4);

imshow(dec_flower+wf1); title('5x5 拉普拉斯滤波');
imshow(dec_flower+wf2); title('9x9 拉普拉斯滤波');
imshow(dec_flower+wf3); title('15x15 拉普拉斯滤波');
imshow(dec_flower+wf4); title('25x25 拉普拉斯滤波');
```



针对第二个问题，由于使用拉普拉斯算子进行边缘锐化的实质是提取边缘，那么其他的边缘提取算子也可以应用在边缘锐化上。对于均值滤波去噪后的图像，分别叠加由 Sobel、Prewitt 得到的边缘。

```
sob = fspecial('sobel')/4;
prew = fspecial('prewitt')/3;

subplot(2, 2, 1);
imshow(dec_flower); title('中值去噪后的原图');
subplot(2, 2, 2);
imshow(imfilter(dec_flower, -lap)+dec_flower); title('拉普拉斯滤波');
subplot(2, 2, 3);
imshow(imfilter(dec_flower, sob)+dec_flower); title('Sobel 算子')
subplot(2, 2, 4);
imshow(imfilter(dec_flower, prew)+dec_flower); title('Prewitt 算子')
```



观察后可以发现,相较于拉普拉斯滤波,使用 Sobel 算子和 Prewitt 算子得到的结果显然更为自然。另外可以观察到，使用 Sobel 算子得到的图像的边缘似乎更为平滑一些。推测可能是由于这幅荷花图像的边缘水平与垂直方向的因素更多一些。