第二次作业

夏熙 3017218170

作业要求：
实现课上学的滤波器
给图像添加噪声
用滤波器进行图像复原
比较效果

1. 给图像添加噪声

```
%添加椒盐噪声
pSandP =imnoise(pGray,'salt & pepper',0.02);
subplot(3,2,3);
imshow(pSandP);
title('椒盐噪声');

 %添加高斯噪声
pGauss =imnoise(pGray,'gaussian',0.02); %加均值为0，方差为0.02的高斯噪声
subplot(3,2,4);
imshow(pGauss);
title('高斯噪声');
```

2. 滤波器

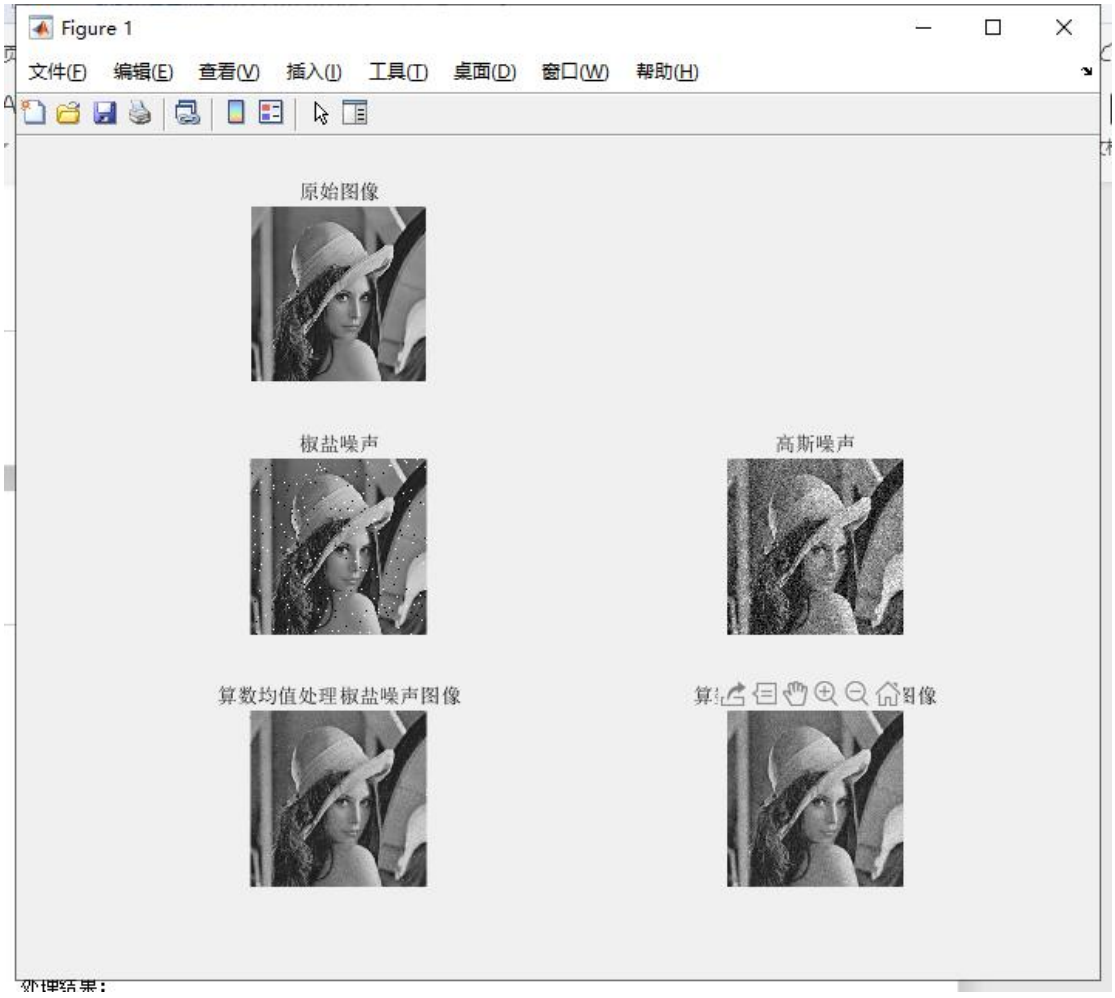
2.1 实现算数均值滤波器

```
function y = MVFilter(pic,n)  % 算数均值滤波器
    a(1:n,1:n)=1;   %a即n×n模板,元素全是1
    [height, width]=size(pic);   %输入图像是height x width的,且hight>n,width>n
    x1=double(pic);  %double(y)表示将参数y转为双精度浮点类型，如果y是字符，将返回字符的ASCII码值。在matlab
    x2=x1;
    for i=1:height-n+1
        for j=1:width-n+1
            c=x1(i:i+(n-1),j:j+(n-1)).*a; %取出x1中从(i,j)开始的n行n列元素与模板相乘
            s=sum(sum(c));                %求c矩阵中各元素之和
            x2(i+(n-1)/2,j+(n-1)/2)=s/(n*n); %将与模板运算后的各元素的均值赋给模板中心位置的元素
        end
    end
    %未被赋值的元素取原值
    y=uint8(x2);
end
```

```
%算术平均滤波J
pMVFilter = MVFilter(pSandP,3);
subplot(3,2,5);
imshow(pMVFilter);
title('算数均值处理椒盐噪声图像');

 %算术平均滤波G
pMVFilter = MVFilter(pGauss,3);
subplot(3,2,6);
imshow(pMVFilter);
title('算数均值处理高斯噪声图像');
```

处理结果：



优点是能够一定程度去除噪声

缺点是存在着边缘模糊的问题。

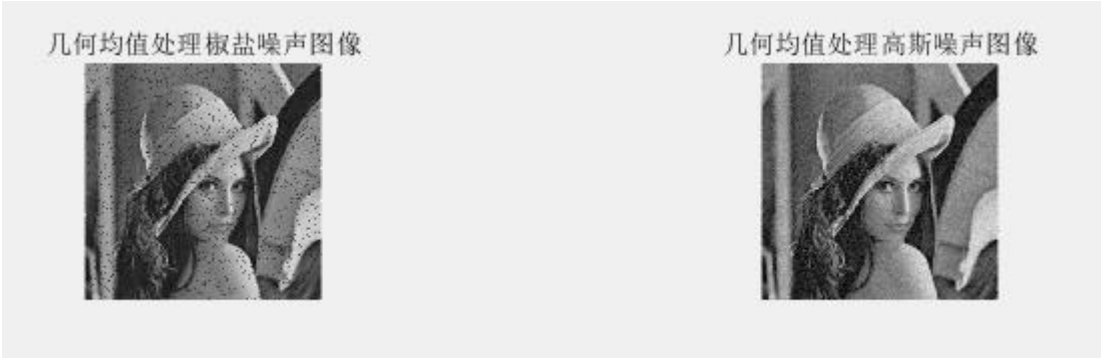
2.2 实现几何均值滤波器

```
function y = GMFilter(pic ,n)  % 几何均值滤波器
    [height, width]=size(pic);%%输入图像是height x width的
    x1=double(pic);
    F=x1+ones(height, width);  %防止有像素点为0而导致乘积为0
    for i=1:height-n+1
        for j=1:width-n+1
            P=F(i:i+n-1,j:j+n-1);
            X=prod(P(:));  %如果A是向量，prod(A)返回A向量的乘积。
                           %如果A是矩阵，prod(A)将A看作列向量，返回每一列元
            x1(i+(n-1)/2,j+(n-1)/2)=X.^(1/numel(P));%返回数组A中元素个数。若是
        end
    end
    y = uint8(x1);
end
```

```
%几何均值滤波J
pGMFilter = GMFilter(pSandP,3);
subplot(3,2,5);
imshow(pGMFilter);
title('几何均值处理椒盐噪声图像')

% 几何均值滤波G
pGMFilter = GMFilter(pGauss,3);
subplot(3,2,6);
imshow(pGMFilter);
title('几何均值处理高斯噪声图像')
```

效果：



可以看出和算术均值滤波器相比，几何均值滤波器能够更好的去除高斯噪声，并且能够更多的保留图像的边缘信息。

2.3 实现谐波均值滤波器

```
function y = HMFilter(pic)    % 3*3的谐波均值滤波器
[w,h]=size(pic);
x1=double(pic);
F=x1+ones(w,h);
fsize1=3;
fssize1=(fsize1-1)/2;
for x=1+fssize1:1:w-fssize1
    for y=1+fssize1:1:h-fssize1
            is=F(x-fssize1:1:x+fssize1,y-fssize1:1:y+fssize1);
            is=1./is;
            x1(x,y)=numel(is)/sum(is(:));
    end
end
    y = uint8(x1);
end
```

```
%谐波均值滤波器J
pHMFilter = HMFilter(pSandP);
subplot(3,2,5);
imshow(pHMFilter);
title('谐波均值处理椒盐噪声图像');

%谐波均值滤波器G
pHMFilter = HMFilter(pGauss);
subplot(3,2,6);
imshow(pHMFilter);
title('谐波均值处理高斯噪声图像');
```

效果图：

可以看出谐波均值滤波器对盐噪声的处理效果较好，但是不适合胡椒噪声，并且处理高斯噪声的效果较好。

2.4 实现逆谐波均值滤波器

```
function y = CHFilter(pic)    % 3*3的逆谐波均值滤波器
f=pic;
[w,h]=size(f);
x1=double(pic);
resultImage=x1+ones(w,h);
Q1 = 1;
fsize1=3;
fssize1=(fsize1-1)/2;
for x=1+fssize1:1:w-fssize1
    for y=1+fssize1:1:h-fssize1
        is=f(x-fssize1:1:x+fssize1,y-fssize1:1:y+fssize1);
        resultImage(x,y) = sum(is(:).^(Q1))/sum(is(:).^(Q1-1));
    end
end
 y = uint8(resultImage);
end
```

```
  %逆谐波均值滤波器J
pCHFilter = CHFilter(pSandP);
subplot(3,2,5);
imshow(pCHFilter);
title('逆谐波均值处理椒盐噪声图像');

 %逆谐波均值滤波器G
pCHFilter = CHFilter(pGauss);
subplot(3,2,6);
imshow(pCHFilter);
title('逆谐波均值处理高斯噪声图像');
```

处理效果：

可见 Q 为正数时能够较好的消除胡椒噪声

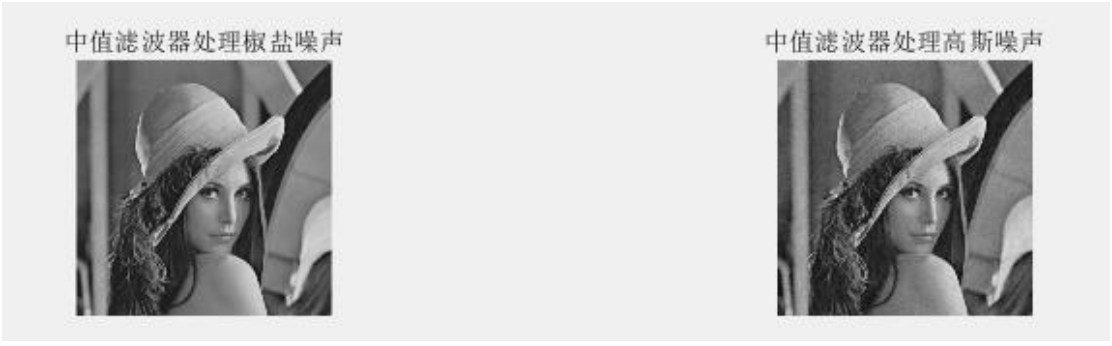
2.5 实现中值滤波器

```
    %中值滤波器J
pMFilter = MFilter(pSandP);
subplot(3,2,5);
imshow(pMFilter);
title('中值滤波器处理椒盐噪声');

%中值滤波器G
pMFilter = MFilter(pGauss);
subplot(3,2,6);
imshow(pMFilter);
title('中值滤波器处理高斯噪声');
```

```
function y = MFilter(pic)   % 中值滤波器
f = pic;
[w,h] = size(f);
fsize1=3;
fssize1=(fsize1-1)/2;
resultImage = f;
for x=1+fssize1:1:w-fssize1
    for y=1+fssize1:1:h-fssize1
        is=f(x-fssize1:1:x+fssize1,y-fssize1:1:y+fssize1);
        temp = sort(is(:));
        resultImage(x,y)= temp((numel(temp)+1)/2);
    end
end
y = resultImage;
end
```

处理效果：



可以看出中值滤波器的去噪能力较好，并且对椒盐噪声尤其有效

2.6 实现最大值滤波器

```
%最大值滤波器J
pMXFilter = MXFilter(pSandP);
subplot(3,2,5);
imshow(pMXFilter);
title('最大值滤波器处理椒盐噪声');
```

```
%最大值滤波器G
pMXFilter = MXFilter(pGauss);
subplot(3,2,6);
imshow(pMXFilter);
title('最大值滤波器处理高斯噪声');
```

```
function y = MXFilter(pic)  % 3*3最大值滤波器
f=pic;
[w,h]=size(f);
fsize1=3;
fssize1=(fsize1-1)/2;
resultImage = pic;
for x=1+fssize1:1:w-fssize1
    for y=1+fssize1:1:h-fssize1
        is=f(x-fssize1:1:x+fssize1,y-fssize1:1:y+fssize1);
        temp = is(:);
        resultImage(x,y)= max(temp);
    end
end
y = resultImage;
end
```

处理效果：

可以看出最大值滤波器适合处理胡椒噪声，但不适合处理盐噪声，

2.7 实现最小值滤波器

```
function y = MNFilter(pic)  %  3*3最小值滤波器
 f=pic;
[w,h]=size(f);
fsize1=3;
fssize1=(fsize1-1)/2;
resultImage = pic;
for x=1+fssize1:1:w-fssize1
    for y=1+fssize1:1:h-fssize1
        is=f(x-fssize1:1:x+fssize1,y-fssize1:1:y+fssize1);
        temp = is(:);
        resultImage(x,y)= min(temp);
    end
end
y = resultImage;
end
```

```
%最小值滤波器J
pMNFilter = MNFilter(pSandP);
subplot(3,2,5);
imshow(pMNFilter);
title('最小值滤波器处理椒盐噪声');
```

```
%最小值滤波器G
pMNFilter = MNFilter(pGauss);
subplot(3,2,6);
imshow(pMNFilter);
title('最小值滤波器处理高斯噪声');
```

处理效果：

可以看出最小值滤波器适合处理盐噪声，但不适合处理胡椒噪声，

2.8 实现中点滤波器

```
function y = MPFilter(pic)  % 3*3中点滤波器
f=pic;
[w,h]=size(f);
fsize1=3;
fssize1=(fsize1-1)/2;
resultImage = pic;
for x=1+fssize1:1:w-fssize1
    for y=1+fssize1:1:h-fssize1
        is=f(x-fssize1:1:x+fssize1,y-fssize1:1:y+fssize1);
        temp = sort(is(:));
        resultImage(x,y)= (max(temp) + min(temp))/2;
    end
end
y = resultImage;
end
```

```
%中点滤波器J
pMPFilter = MPFilter(pSandP);
subplot(3,2,5);
imshow(pMPFilter);
title('中点滤波器处理椒盐噪声');

%中点滤波器G
pMPFilter = MPFilter(pGauss);
subplot(3,2,6);
imshow(pMPFilter);
title('中点滤波器处理高斯噪声');
```

效果：

可见中点滤波器对于高斯噪声的去噪效果较好，但不适合椒盐噪声

2.9 修正后的阿尔法均值滤波器：

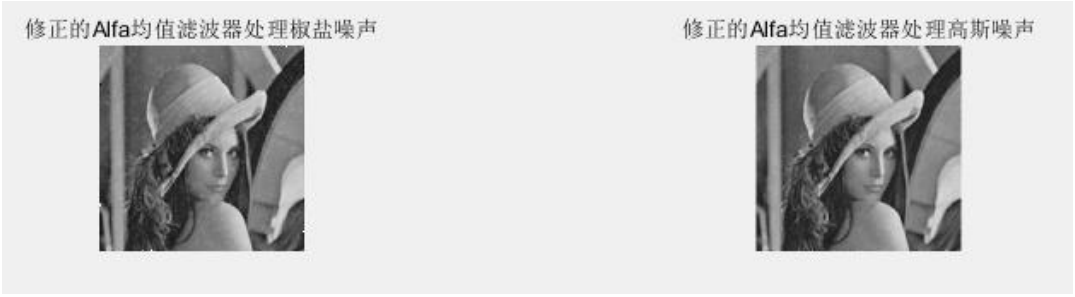
```
function y = AFilter(pic)    % 5*5的修正的阿尔法均值滤波器
f = pic;
[w,h]=size(f);
n = 5; %模板大小
d = 3; %去掉最值灰度值的个数
resultImage = f;
for i = 1:w-n+1
    for j = 1:h-n+1
        g3 = f(i:i+n-1,j:j+n-1);
        g3 = sort(g3(:)); %对邻域内的像素点进行排序
        min_num = ceil(d/2); %去掉最小灰度值的个数
        max_num = floor(d/2); %去掉最大灰度值的个数
        %去掉d个最值灰度级后求算术均值
        g3(1:min_num) = zeros(min_num,1);
        g3(n-max_num+1:n) = zeros(max_num,1);
        s3 = sum(g3);
        %中心点的值用子图像的算术均值代替
        resultImage(i+(n-1)/2,j+(n-1)/2) = s3/(n*n-d);
    end
end
y = uint8(resultImage);
end
```

```
    %修正的阿尔法均值滤波器J
pAFilter = AFilter(pSandP);
subplot(3,2,5);
imshow(pAFilter);
title('修正的Alfa均值滤波器处理椒盐噪声');

%修正的阿尔法均值滤波器G
pAFilter = AFilter(pGauss);
subplot(3,2,6);
imshow(pAFilter);
title('修正的Alfa均值滤波器处理高斯噪声');
```

效果：

（窗口大小 5*5，d=5）



（窗口大小 5*5，d=3）

（窗口大小 3*3，d=3）

可以看出修正后的阿尔法均值滤波器对于椒盐噪声和高斯噪声都具有较好的处理效果，选择合适的窗口大小和 d 的大小也很重要，不同的取值效果有较大的差异

2.10 自适应滤波器

实现：

```
function y = ADFilter(pic) %自适应滤波器
[w,h]=size(pic);
gImg=double(pic);
gNewImg=gImg;
fsize1=5;
fssize1=(fsize1-1)/2;

    for i=1+fssize1:w-fssize1
        for j=1+fssize1:h-fssize1
                gTemp = gImg(i-fssize1:i+fssize1,j-fssize1:j+fssize1);
                avg = mean(gTemp(:));
                v = var(gTemp(:));
                gNewImg(i,j)=gImg(i,j)-255*255*0.02*0.02/v*(gImg(i,j)-avg);
        end
    end
    y=uint8(gNewImg);
end
```

```
 %自适应滤波器J
pADFilter = ADFilter(pSandP);
subplot(5,6,25);
imshow(pADFilter);
title('自适应滤波器处理椒盐噪声');

%自适应滤波器G
pADFilter = ADFilter(pGauss);
subplot(5,6,26);
imshow(pADFilter);
title('自适应滤波器处理高斯噪声');
```

处理效果：

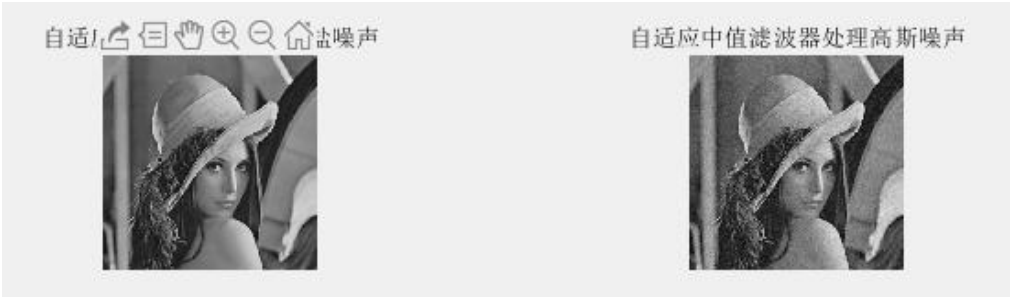
2.11 自适应中值滤波器

```
function y = AMFilter(f2) %自适应中值滤波器
f14 = f2;
alreadyPro = false(size(f2)); %看是否完成进程 falsea函数产生和f2同样大小的全0逻辑矩阵
Smax=5;       %最大窗口尺寸
for s = 3:2:Smax    %起始窗口尺寸设为3
    %得到特定的灰度值
    zmin = ordfilt2(f2, 1, ones(s, s), 'symmetric');
    zmax = ordfilt2(f2, s * s, ones(s, s), 'symmetric');
    zmed = medfilt2(f2, [s s], 'symmetric');
    %进程A
    processA = (zmed > zmin) & (zmax > zmed) & ~alreadyPro;
    %进程B
    processB = (f2 > zmin) & (zmax > f2);
    outZxy  = processA & processB;
    outZmed = processA & ~processB;
    f14(outZxy) = f2(outZxy);
    f14(outZmed) = zmed(outZmed);
    alreadyPro = alreadyPro | processA;
    if all(alreadyPro(:))  %如果alreadyPro的所有值都变为1则跳出for循环
       break;
    end
end
f14(~alreadyPro) = zmed(~alreadyPro);
y = uint8(f14);
end
```

```
    %自适应中值滤波器J
pAMFilter = AMFilter(pSandP);
subplot(3,2,5);
imshow(pAMFilter);
title('自适应中值滤波器处理椒盐噪声');

%自适应中值滤波器G
pAMFilter = AMFilter(pGauss);
subplot(3,2,6);
imshow(pAMFilter);
title('自适应中值滤波器处理高斯噪声');
```

效果：



自适应中值滤波对椒盐噪声和高斯噪声都有较好的处理效果