# 滤波器的

## 一、添加噪声

主要的噪声有高斯噪声、瑞利噪声、伽马噪声、指数噪声、均匀噪声、椒盐噪声。给一个图像加噪声主要使用matlab中的im2double函数对图像作归一化处理，然后根据相应的噪声分布函数生成噪声图像，把噪声图像加到原始图像中就可以了。高斯噪声和椒盐噪声的添加可以使用imnoise函数完成。

img= im2double(rgb2gray(imread('pic.jpg')));%im2double作归一化处理

sigma=25;%添加噪声的标准差

J\_Gaussian=imnoise(img,'gaussian',0,(sigma/255)^2); %添加均值为0，标准差为sigma高斯噪声

imwrite(J\_Gaussian,'gaussian.png');%输出高斯噪声图

J\_salt=imnoise(img,'salt & pepper',0.07); %添加7%的椒盐噪声

imwrite(J\_salt,'salt&pepper.png');

%生成瑞利噪声

[M,N]=size(img);

a=0;

b=0.08;

B=1;

N\_Ray1=a+b\*raylrnd(B,M,N);

%将噪声叠加到图像上

J\_rayl=img+N\_Ray1;

imwrite(J\_rayl,'rayl.png');

% 伽马噪声

a=0;

b=0.08;

A=1;

B=2;

N\_Gam=a+b\*gamrnd(A,B,[M,N]);

J\_Gamma=img+N\_Gam;

imwrite(J\_Gamma,'gamma.png');

% 指数噪声

a=0;

b=0.08;

mu=2;

N\_exp=a+b\*exprnd(mu,[M,N]);

J\_exp=img+N\_exp;

imwrite(J\_exp,'exp.png');

% 均匀分布噪声

a=0;

b=0.08;

A=0;

B=2;

N\_unif=a+b\*unifrnd(A,B,[M,N]);

J\_unif=img+N\_unif;

imwrite(J\_unif,'uniform.png');

subplot(2,3,1);

imshow(J\_Gaussian);

title('高斯噪声');

subplot(2,3,2);

imshow(J\_rayl,[]);

title('瑞利噪声');

subplot(2,3,3);

imshow(J\_Gamma,[]);

title('伽马噪声');

subplot(2,3,4);

imshow(J\_exp,[]);

title('指数噪声');

subplot(2,3,5);

imshow(J\_unif,[]);

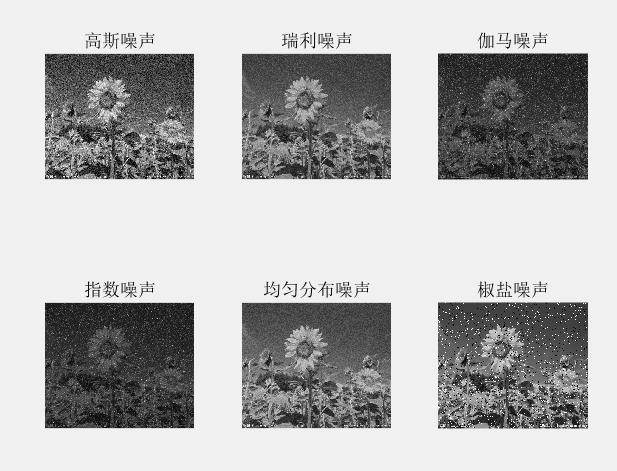
title('均匀分布噪声');

subplot(2,3,6);

imshow(J\_salt,[]);

title('椒盐噪声');

最终效果：



同时，对上述代码进行如下修改，可以实现给彩色图像加噪声

img= im2double(imread('pic.jpg'));%im2double作归一化处理

[M,N,count]=size(img);



## 二、滤波器的实现

### 2.1均值滤波器

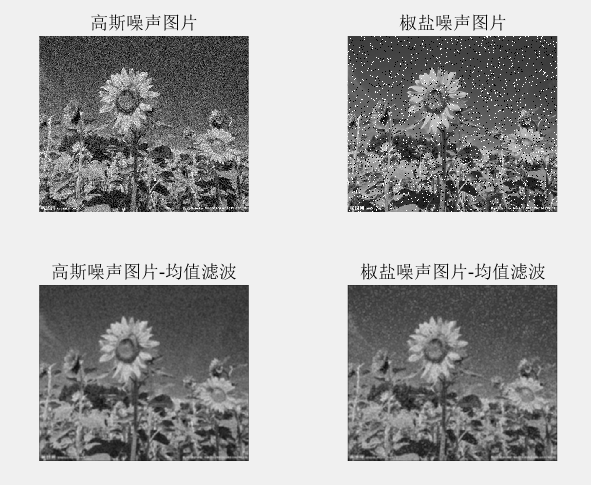
#### 2.1.1算术均值滤波器

算数均值滤波器是对周围的m\*n的像素点求平均值，作为这一点的输出，实现过程中定义了3\*3的滤波盒大小，以下是实现代码，为了清楚的描述算数均值滤波器的matlab实现，我只在文档中提供关键代码，首先，使用fspecial创建均值滤波器的算子，然后使用imfilter对图像进行滤波。Fdata就是滤波后的图像。

h=fspecial('average',5);%创建一个均值(5\*5)

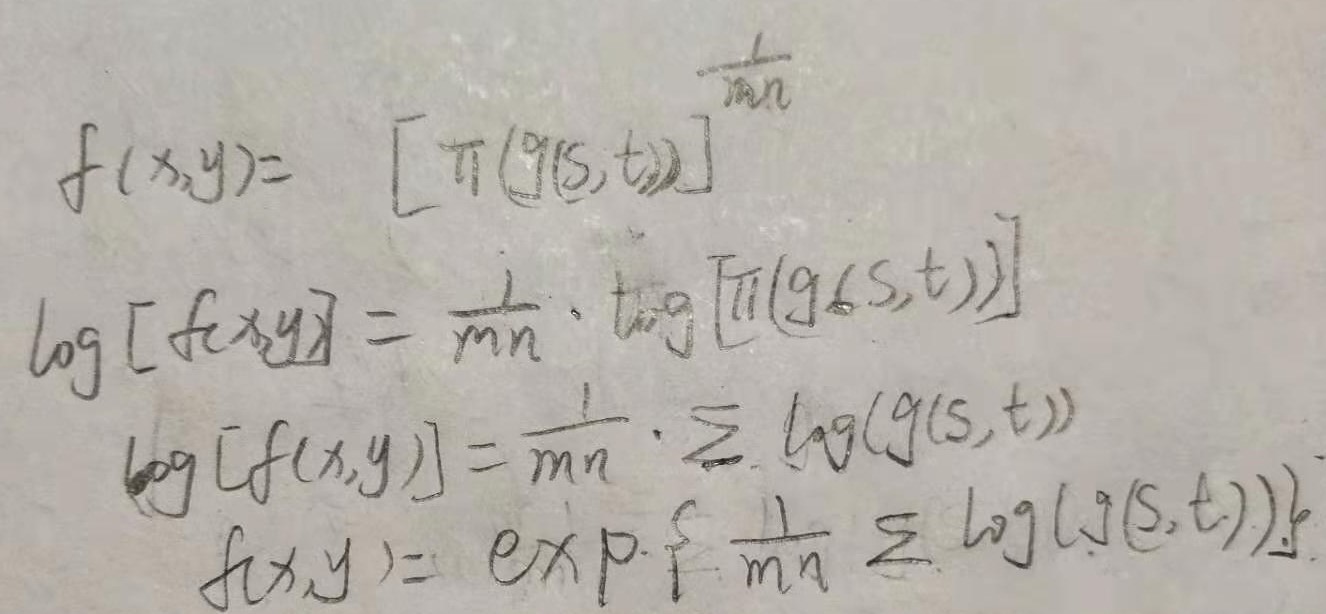
fdata=imfilter(gaussian,h);%前面是图片，后面是模板

以下是对加过高斯噪声、椒盐噪声的图片进行均值滤波之后得到的效果图片。详细代码在test02.m中。可以看到图像在滤波之后变得模糊了许多，但是也起到了不错的去噪效果，然而，考虑到这个去噪效果是以牺牲清晰度为代价的，在实际应用中，不常用这种方式进行滤波。



#### 2.1.2几何均值滤波器

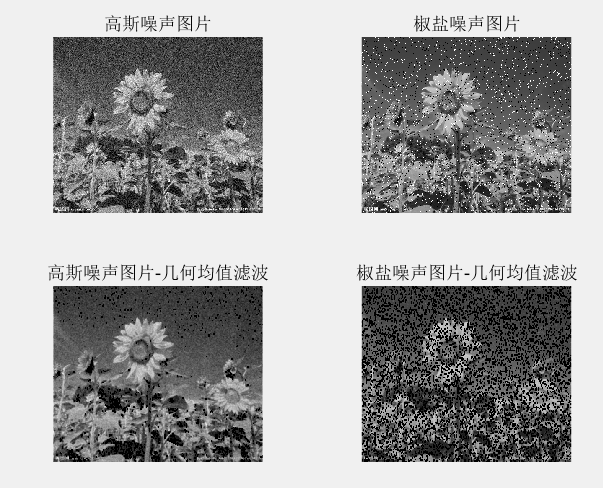
基于以上对算数均值滤波器的描述，对几何均值滤波器的公式进行如下变换，可以很容易根据算数均值滤波器实现几何均值滤波器。



因此，几何均值滤波器实现代码如下：

Result=exp(imfilter(log(img),fspecial('average',5)))；

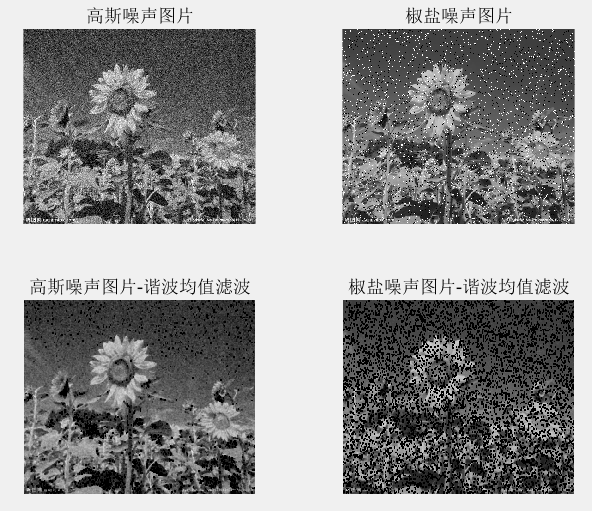
Result是滤波之后的图像，对原始img进行log操作之后采用均值滤波之后，再对结果进行exp操作，最后得到的结果就是几何均值滤波器的结果。如下是几何均值滤波器对椒盐噪声、高斯噪声的效果，图像相比算数均值滤波器清晰了许多，但是图像莫名多出了许多的黑点，可见效果不是很好。（详细代码在test03.m）



#### 2.1.3谐波均值滤波器

基于谐波滤波器的公式，可以根据算数均值滤波器变换实现该滤波器，详细代码如下：

Result=1./(imfilter(1./img,fspecial('average',5)))；下图是谐波滤波器对高斯噪声、椒盐噪声的去除前后效果对比，可以看到，它对于“盐” 噪声效果较好，但不适于“胡椒”噪声。它善于处理高斯噪声。代码在（test04.m）



#### 2.1.4逆谐波均值滤波器

这种滤波器适合减少或是在实际中消除椒盐噪声的影响。当*Q*值为正数时，滤波器用于消除“胡椒”噪声;当*Q*值为负数时，滤波器用于消除“盐”噪声。逆谐波的实现与前面的滤波器有些不同，但是还是可以用算数均值滤波器实现，但是需要两个算数均值滤波器，先对图像中的每个元素求Q+1次幂，然后求算数均值滤波器，得到一个图像矩阵，然后对原图像中每个元素求Q次幂，再求算数均值滤波器，得到一个图像矩阵，把两个对应元素相除之后，得到的图像矩阵就是逆谐波均值滤波器之后的效果，其主要代码如下，其中-1是Q+1,-2是Q。

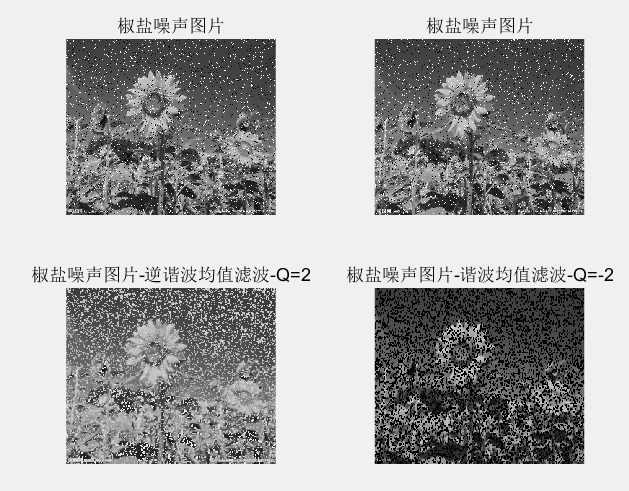
h=fspecial('average',3);%创建一个均值(5\*5)

fdata1=imfilter(salt.^-1,h);%前面是图片，后面是模板

fdata2=imfilter(salt.^-2,h);

fdata=fdata1./fdata2;

使用Q=-2和Q=2对椒盐噪声图像处理，对比结果可以看到，Q=2(正值)的时候，图像中黑色的胡椒噪声看不见了，但是盐噪声还在，而Q=-2的时候，正好相反，图像中的椒噪声很明显，基本没有去除，但是盐噪声已经被去除了。详细代码（test05.m）



### 2.2统计排序滤波器

#### 2.2.1中值滤波器

#### 2.2.2最大值和最小值滤波器

#### 2.2.3中点滤波器

#### 2.2.4修正后的阿尔法均值滤波器

### 2.3自适应中值滤波器