

数字图像处理

第六次作业



摘要

本报告第一个任务是通过了解高斯噪声，椒盐噪声的产生方式，给目标图片加上噪声。并通过之前做过的各种滤波器进行滤波，最后分析各自的优缺点。第二个任务是推导维纳滤波器。所使用的编译环境是Matlab2018。

杨致远

电信钱51

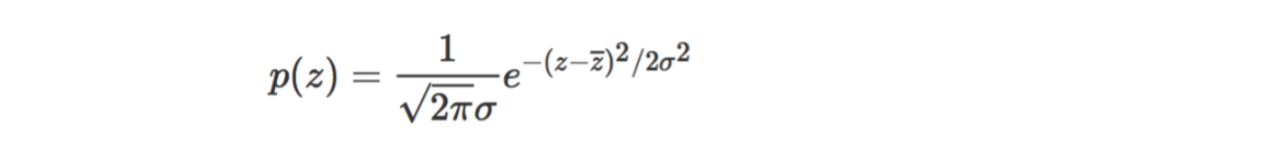
2150405061

2018/5/16

## 任务1—高斯噪声

**实验介绍：**

1. 首先什么是高斯噪声，是指噪声概率分布函数是高斯模型的一类噪声。高斯随机变量z的PDF公式为：



式中，z表示灰度值，Z为z的均值，sigma为z的标准差

当我们在给图片加噪声时，噪声描述子就是带入原图像的像素值到公式中算出对应的值然后加到原图像上即可。

1. 利用多中滤波器的恢复过程，因为在第四次作业中已经完成，所以在这里不再赘述。
2. 实际运用中，我们既可以自己构建高斯噪声函数，实现在原图的基础上加上噪声然后形成一幅新的图像；同样我们也可以利用matlab提供的现有函数imnoise()函数来进行构建。结果中我运用了两种方式来实现作业任务。

**结果展示：**



**结果讨论：**

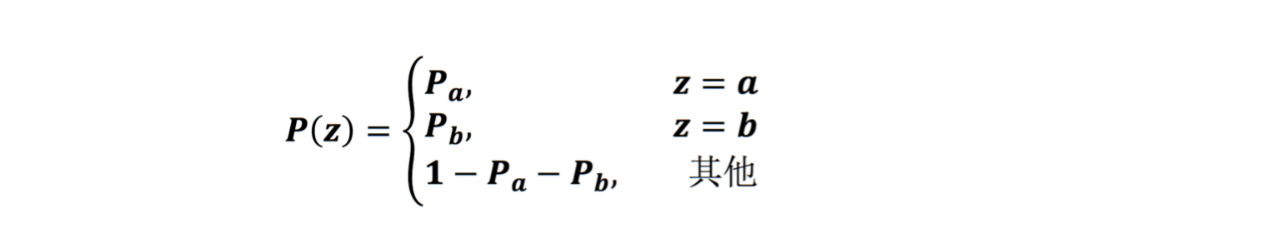
当高斯噪声均值不变为0时，随着方差增加，图像噪声越严重；当高斯噪声方差不变时，均值会影响到整个图像的灰度值，使整个图像变亮。

然后使用了高斯滤波和中值滤波，两者对图像均进行了恢复，但可以发现高斯滤波的效果会更好，恢复的图像细节更加丰富。

## 任务2—椒盐噪声

**实验介绍：**

1. 通过上面对高斯噪声的了解，椒盐噪声也类似，甚至比高斯噪声还要容易。原理是：



其中如果b>a，则灰度级b在图像中将显示为一个两点；反之，灰度级a在图像中将显示为一个暗点。若Pa和Pb为零，则脉冲噪声成为单极脉冲。通俗来讲也就是在途中随机生成一些点，这些点的灰度值为0或者255。

**结果展示：**



**结果讨论：**

从图中可以看到椒盐噪声产生的效果，直观上讲就是在图片中穿插了很多或黑或白的点。然后通过中值和高斯进行滤波，滤波效果对比可以看到中值滤波的效果会更好。

## 任务3—维纳滤波器

**实验介绍：**

1. （5.6-11）公式为：

要求是45度方向，也就是T=1，a=b=0.1;然后同第五次作业的卷积变换，对原始图像做傅里叶变换得到F，F和H作乘法，然后进行傅里叶反变换得到变换后的图像。

1. （5.8-6）公式为：

(5.9-4)公式为：

维纳滤波器又称为最小均方差滤波器，是为了使得f(x,y)和(x,y)的均方误差最小。

推导维纳滤波：

首先是任务目标是让协方差最小：

应用拉格朗日乘数法：

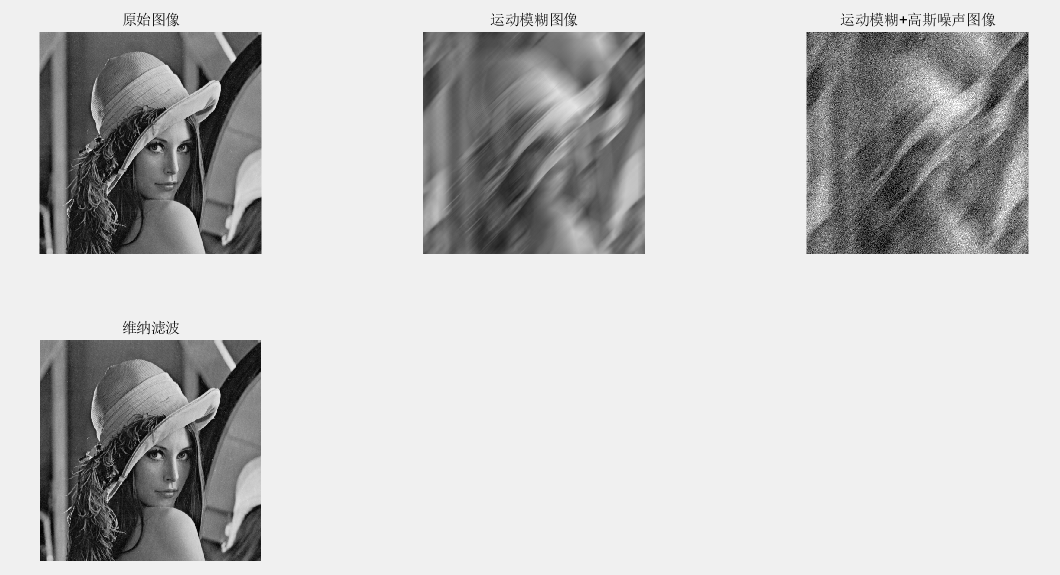
对上式求导，令导数为0：

然后做一个变换，得到

然后就得到了上面维纳方程的两个公式！

自己编写了一个维纳滤波器函数

**结果展示：**



**结果讨论：**

通过维纳滤波，较好地恢复了原始图像。

## 代码：

%高斯噪声

clear;clc;

picture = imread('lena.bmp');

subplot(3,3,1:3);

imshow(picture);

title('原始图像');

% y = gaussmf(x,[sig c])

% y1 = exp(-((x-c)^2)/(2\*sig^2))

picture\_noise1 = imnoise(picture, 'gaussian', 0, 0.01); %现有函数能产生高斯噪声

subplot(3,3,4);

imshow(picture\_noise1);

title('高斯噪声-均值0-方差0.01');

picture\_noise2 = imnoise(picture, 'gaussian', 0, 0.1); %现有函数能产生高斯噪声

subplot(3,3,5);

imshow(picture\_noise2);

title('高斯噪声-均值0-方差0.1');

picture\_noise3 = imnoise(picture, 'gaussian', 0.5, 0.01); %现有函数能产生高斯噪声

subplot(3,3,6);

imshow(picture\_noise3);

title('高斯噪声-均值0.5-方差0.01');

picture1 = medfilt2(picture\_noise1,[5 5]);

subplot(3,3,7);

imshow(picture1);

title('中值滤波');

picture2 = imgaussfilt(picture\_noise1, 1.5); %高斯滤波，指定标准差为1.5

subplot(3,3,8);

imshow(picture2);

title('高斯滤波');

%椒盐噪声

clear;clc;

picture = imread('lena.bmp');

subplot(3,3,1:3);

imshow(picture);

title('原始图像');

picture\_noise1 = imnoise(picture,'salt & pepper',0.1); %现有函数能产生高斯噪声

subplot(3,3,5);

imshow(picture\_noise1);

title('椒盐噪声');

picture1 = medfilt2(picture\_noise1,[5 5]);

subplot(3,3,7);

imshow(picture1);

title('中值滤波');

picture2 = imgaussfilt(picture\_noise1, 1.5);

subplot(3,3,8);

imshow(picture2);

title('高斯滤波');

%%维纳滤波

function J = mydeconvwnr(I, PSF, NSR)

sizeI = size(I);

H = psf2otf(PSF, sizeI);

S\_u = NSR;

S\_x = 1;

denom = abs(H).^2;

denom = denom .\* S\_x;

denom = denom + S\_u;

clear S\_u

denom = max(denom, sqrt(eps));

G = conj(H) .\* S\_x;

clear H S\_x

G = G ./ denom;

clear denom

J = ifftn(G .\* fftn(I));

clear G

J = real(J);

clear;clc;

picture = imread('lena.bmp');

subplot(2,3,1);

imshow(picture);

title('原始图像');

picture = double(picture);

P = fft2(picture);

P = fftshift(P);

[U, V] = size(P);

a = 0.1; b = 0.1; T = 1;

for u = 1:U

for v = 1:V

H(u,v) = (T/(pi\*(u\*a+v\*b)))\*sin(pi\*(u\*a+v\*b))\*exp(-sqrt(-1)\*pi\*(u\*a+v\*b));

G(u,v) = H(u,v)\*P(u,v);

end

end

G = ifftshift(G);

g = ifft2(G);

g=256.\*g./max(max(g));

g=uint8(real(g));

subplot(2,3,2);

change = fspecial('motion',50,45);

picture1 = imfilter(g,change,'circular','conv');

imshow(picture1);

title('运动模糊图像');

picture2 = imnoise(picture1,'gaussian',0,0.01);

subplot(2,3,3);

imshow(picture2);

title('运动模糊+高斯噪声图像');

pp = im2double(imread('lena.bmp'));

blurred = imfilter(pp, change, 'conv', 'circular');

picture3 = mydeconvwnr(blurred, change, 0);

subplot(2,3,4);

imshow(picture3);

title('维纳滤波');

## 参考文献

[1] 冈萨雷斯.数字图像处理（第三版）北京：电子工业出版社，2011

[2] 维基百科