DIP HW4

题目

1. 使用教材上等式*5.6-11*实现一个模糊滤波器，并模糊图像使用a=b=0.1 T=1
2. 将平均值为0方差为500的高斯噪声加入模糊图像
3. 使用逆滤波器恢复模糊图像和模糊噪声图像
4. 使用参数维纳滤波器恢复模糊噪声图像，使用至少3个不同的参数，并将结果与3进行比较

算法描述

1. 模糊滤波器

$$H(u,v)=\frac{T}{\pi(ua+vb)}sin[\pi(ua+vb)]e^{-j\pi(ua+vb)}$$

2. 加入高斯噪声

```
INoise  = imnoise(zeros(M,N), 'gaussian', 0, 500);
FNoise = ifftshift(ifft2(INoise));
FGaussian = FBlurred + FNoise;
IGaussian = real(ifft2(ifftshift(FGaussian)));
```

3. 逆滤波器

   逆滤波器对噪声较小的退化图像可以获得较好的复原效果。复原过程为

$$\hat{F}(u,v)=G(u,v)/H(u,v)$$

```
%直接逆滤波
BlurredInv = FBlurred ./ H;
BlurredInvRestoration1 = real(ifft2(ifftshift(BlurredInv)));
```

   - 直接逆滤波存在一些问题：对于某些点(u, v), H(u, v)可能为0，因此不能直接做分母；或者可能存在一些amplified的噪声
   - 解决方法：伪逆滤波复原

$$H^{-1}=\begin{cases}\frac{1}{H(u,v)} & H(u,v)>\sigma\\ 0 & H(u,v)<=\sigma\end{cases}$$

加性噪声：施加圆形范围限制，使得在圆外的频率（噪声）被抑制

```
for a = 1:M
    for b = 1:N
        if sqrt((a-M/2))^2 + (b-M/2)^2 < r
            FGaussianInvR(a, b) = FGaussian(a, b)./H(a, b);
        end
    end
end
```

- 寻找最佳半径使得逆滤波图像峰值信噪比最小

4. 维纳滤波器

基本原理：寻找最佳复原图像使得均方差$e^2 = E\{|f - \hat{f}|\}$最小。复原图像函数如下：

$$\hat{F}(u,v) = \frac{|H(u,v)|^2}{|H(u,v)|^2 + S_\eta(u,v)/S_f(u,v))} * \frac{G(u,v)}{H(u,v)}$$

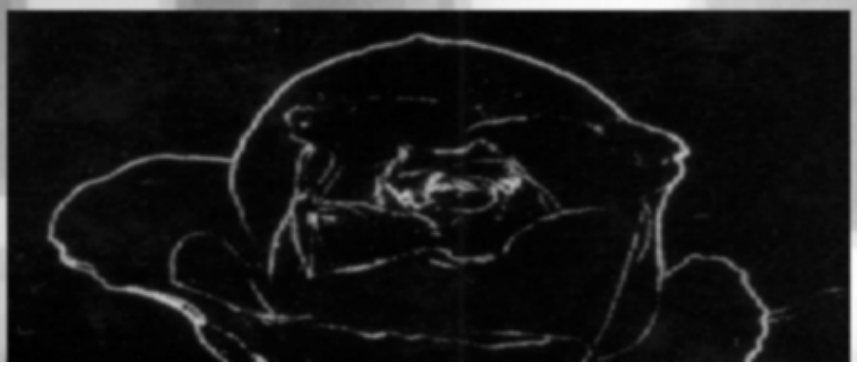
其中，H为退化函数，H^*为退还函数的复共轭，$|H|^2 = H^* * H$，S_η 为噪声功率谱，S_f为未退化图像功率谱。由于在实用中S_η 和S_f较难获得，令$K = \frac{S_\eta}{S_f}$ ，调节K值以得到最佳结果。

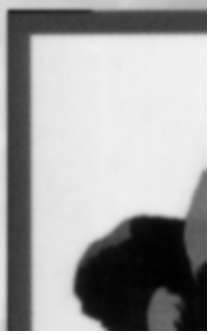
```
Hsq = (abs(H)).^2;
for i = 1:10
    K = i*0.1;
    FGaussianWiennerRestoration = Hsq ./ (Hsq + K) .* FGaussian ./H;
    IGaussianWiennerRestoration =
real(ifft2(ifftshift(FGaussianWiennerRestoration2)));
    imwrite(IGaussianWiennerRestoration, ['../outputs/wienner_', num2str(K)
,'.jpg'], 'jpg');
end
```

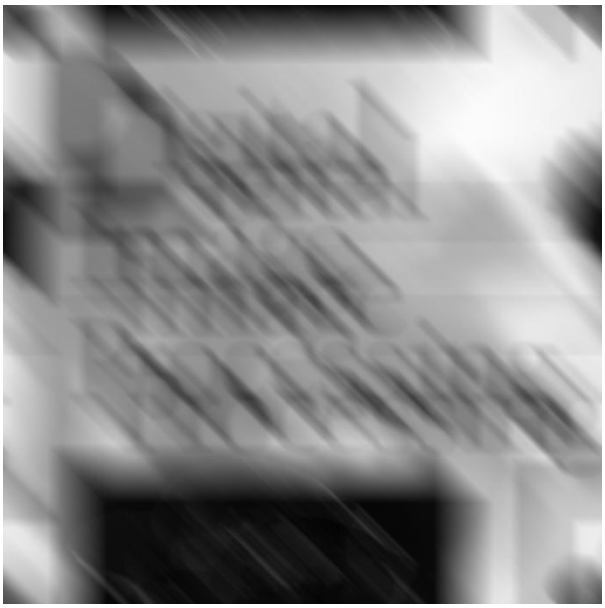
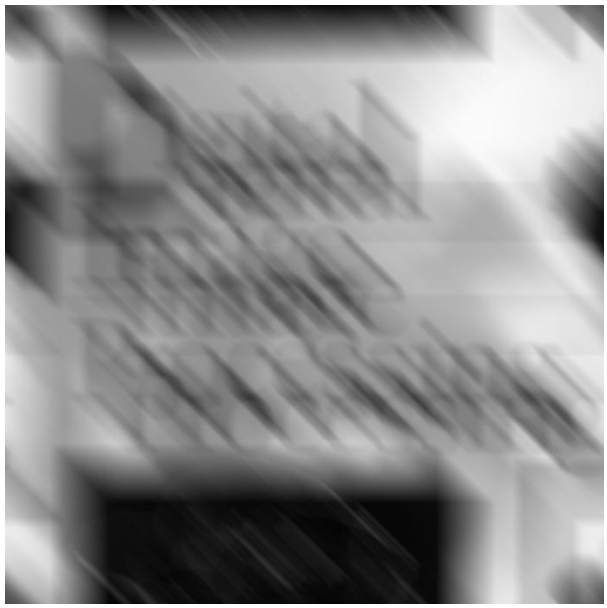
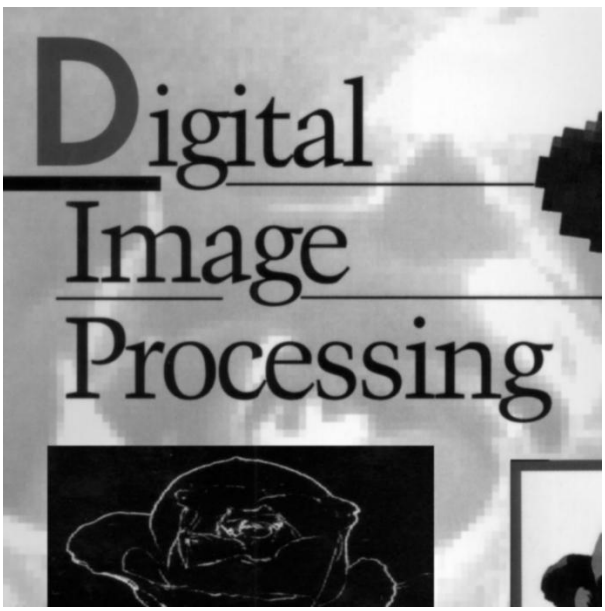
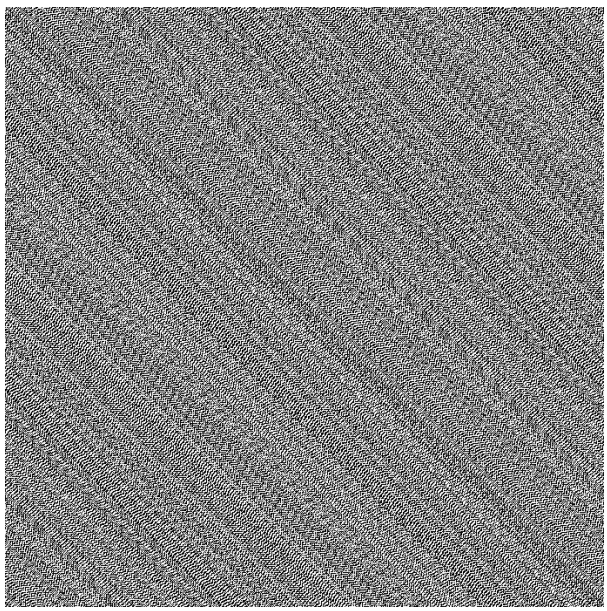
测试结果

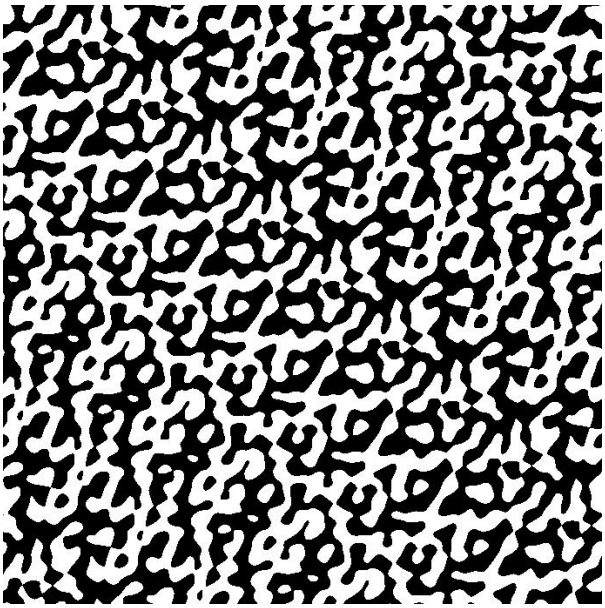
原始图像：

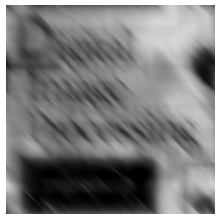
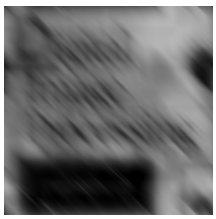
Digital Image Processing





运动模糊图像	模糊-噪声图像
	
运动模糊 直接逆滤波	**模糊-噪声 直接逆滤波**
	
模糊-噪声 逆滤波 R = 38.10	**模糊-噪声 维纳滤波 k = 0.1**

运动模糊图像	模糊-噪声图像
	

k =0.01	k =0.02	k =0.03	k =0.15	k =0.2
				
k =0.2	k =0.3	k =0.35	k =0.38	k =0.39
				

源码

```
close all;

img = imread('../book_cover.jpg');
subplot(2, 3, 1), imshow(img), title('原始图像');
[M,N] = size(img);

I = im2double(img);% [0,1]
j = complex(0, -1);
%% 计算运动模糊矩阵H
T=1;a=0.1;b=0.1;
v=[-M/2:M/2-1];
```

```
u=v';
A=repmat(a.*u,1,M)+repmat(b.*v,M,1);
H=T/pi./A.*sin(pi.*A).*exp(j*pi.*A);
H(A==0)=T;% replace NAN

F=fftshift(fft2(I));
FBlurred=F.*H;

IBlurred =real(ifft2(ifftshift(FBlurred)));
subplot(2, 3, 2), imshow(uint8(255.*mat2gray(IBlurred)))
title("运动模糊");
imwrite(IBlurred, '../outputs/blurred.jpg', 'jpg');

%% 加入高斯噪声
INoise  = imnoise(zeros(M,N), 'gaussian', 0, 500);
FNoise = ifftshift(ifft2(INoise));
FGaussian = FBlurred + FNoise;
IGaussian = real(ifft2(ifftshift(FGaussian)));

subplot(2, 3, 3), imshow(IGaussian), title('高斯噪声');
imwrite(IGaussian, '../outputs/blurred_noise.jpg', 'jpg');

%% 直接逆滤波
BlurredInv = FBlurred ./ H;
BlurredInvRestoration1 = real(ifft2(ifftshift(BlurredInv)));
subplot(2, 3, 4), imshow(BlurredInvRestoration1),title('模糊图像逆滤波');
imwrite(BlurredInvRestoration1, '../outputs/inv_blurred.jpg', 'jpg');

FGaussianInvRestoration = FGaussian ./ H;
IGaussianInvRestoration = real(ifft2(ifftshift(FGaussianInvRestoration)));
subplot(2, 3, 5), imshow(IGaussianInvRestoration), title('模糊-噪声图像逆滤波');
imwrite(IGaussianInvRestoration, '../outputs/inv_blurred_noise.jpg', 'jpg');

%% 寻找最佳半径
maxPsnr = 0;
bestR = 0;
for r = 20:1:50
    FGaussianInvR = zeros(M, N);
    for a = 1:M
        for b = 1:N
            if sqrt((a-M/2)^2 + (b-N/2)^2) < r
                FGaussianInvR(a, b) = FGaussian(a, b)./H(a, b);
            end
        end
    end
    %figure, imshow(real(ifft2(ifftshift(FGaussianInvR))));
    % 计算峰值信噪比
    IGaussianInvR = real(ifft2(ifftshift(FGaussianInvR)));
    psnr = calPsnr(IGaussianInvR, I);

    if psnr > maxPsnr
        maxPsnr = psnr;
        bestR = r;
```

```
    end
end
maxPsnr
bestR
FGaussianInvR = zeros(M, N);
for a = 1:M
    for b = 1:N
        if sqrt((a-M/2)^2 + (b-N/2)^2) < bestR
            FGaussianInvR(a, b) = FGaussian(a, b)./H(a, b);
        end
    end
end
%figure, imshow(FGaussianInvR);
IGaussianInvR = real(ifft2(ifftshift(FGaussianInvR)));
subplot(2, 3, 6), imshow(uint8(255.*mat2gray(IGaussianInvR))), title('模糊-噪声图像半径抑制逆滤波');
imwrite(IGaussianInvR, ['../outputs/inv_blurred_noise', int2str(bestR), '.jpg'], 'jpg');

%% 维纳滤波
Hsq = (abs(H)).^2;
for K = 0:0.01:0.4
    FGaussianWiennerRestoration = Hsq ./ (Hsq + K) .* FGaussian ./H;
    IGaussianWiennerRestoration = real(ifft2(ifftshift(FGaussianWiennerRestoration)));
    imwrite(IGaussianWiennerRestoration, ['../outputs/wienner_', num2str(K),'.jpg'], 'jpg');
end

figure, imshow(IGaussianWiennerRestoration), title("Wienner");


```