数字图像与视频处理
第六次作业

姓　　名：孔维乐

班　　级：自动化 65

学　　号：2160504122

提交日期：2019 年 4 月

摘要：图像的退化过程被建模为一个退化函数和一个加性噪声项。在给定退化后图像、退化函数以及加性噪声项后，就可以对图像进行复原。图像复原的目的是利用给定信息估算出一个尽可能接近原始图像的图像。

关键词：加性噪声　退化函数　图像复原

任务一

在测试图像上产生高斯噪声 lena 图-需能指定均值和方差；并用多种滤波器恢复图像，分析各自优缺点。

问题分析：

1）高斯噪声模型

高斯噪声是指它的概率密度函数服从高斯分布（即正态分布）的一类噪声。如果一个噪声，它的幅度分布服从高斯分布，而它的功率谱密度又是均匀分布的，则称它为高斯白噪声。高斯白噪声的二阶矩不相关，一阶矩为常数，是指先后信号在时间上的相关性。

概率密度函数 PDF：

$$p(z)=\frac{1}{\sqrt{2\pi}\sigma}e^{-(z-\bar{z})^2/2\sigma^2}$$

其中 z 表示灰度值，σ 表示 z 的标准差。标准差的平方 σ 2 称为 z 的方差。

2）均值滤波

均值滤波是一种线性平均滤波器，它通过求窗口内所有像素的平均值来得到中心像素点的像素值。这样的好处是可以有效的平滑图像，降低图像的尖锐程度，降低噪声。但缺点是不能消除噪声。

3）中值滤波

中值滤波也是一种很常用的数字滤波器，它通过找窗内的所有像素值的中值然后赋给中心像素点，然后得到输出图像，这样做的好处是，它不创造新的像素值，只是取周围像素值作为它的输出，这一方法可以有效的消除脉冲噪声，而且可以很好的保护图像尖锐的边缘。

实验结果：

原始图像

加入高斯噪声后的图像

均值滤波后图像

中值滤波后图像

结果分析：

1）在原始图像的加噪声环节，当高斯噪声均值不变时，随着方差的增加，图像的噪声越严重；当高斯噪声的方差不变时，均值会影响到整个图像的灰度值，使整个图像变亮。与理论上均值和方差对图像的影响一致。

2）算数均值滤波器和中值滤波器均可以对加噪图像恢复、降低噪声。均值滤波器在降低图像噪声的同时也模糊了图像。

任务二

在测试图像 lena 图加入椒盐噪声（椒和盐噪声密度均是 0.1）；用学过的滤波器恢复图像；在使用反谐波分析 Q 大于 0 和小于 0 的作用。

问题分析：

1）椒盐噪声模型

椒盐噪声是数字图像中的常见噪声，一般是由图像传感器、传输信道及解码处理等产生的黑白相见的亮暗点噪声，椒盐噪声常由图像切割产生。椒盐噪声是指两种噪声：盐噪声（salt noise）及椒噪声（pepper noise）。盐噪声一般是白色噪声，椒噪声一般是黑色噪声，前者高灰度噪声，后者属于低灰度噪声，一般两种噪声同时出现，呈现在图像上就是黑白杂点。图像去除脉冲干扰及椒盐噪声最常用的算法是中值滤波，图像模拟添加椒盐噪声是通过随机获取像素值点并设置为高亮点来实现的。

$$p(z)=\begin{cases} Pa, z=a \\ Pb, z=b \\ 1-Pa-Pb, \text{其他} \end{cases}$$

2）逆谐波均值滤波器

$$\hat{f}(x,y)=\frac{\sum_{(s,t)\in S_{xy}} g(s,t)^{Q+1}}{\sum_{(s,t)\in S_{xy}} g(s,t)^{Q}}$$

当 Q>0 时，g(s,t)Q 对 g(s,t)有增强作用，由于“胡椒”噪声值较小（0），对加权平均结果影响较小，所以滤波后噪声点处(x,y)取值和周围其他值更接近，有利于消除“胡椒”噪声。

当 Q<0 时，g(s,t)Q 对 g(s,t)有削弱作用，由于“盐”噪声值较大（255），取倒数后较小，对加权平均结果影响较小，所以滤波后噪声点处(x,y)取值和周围其他值更接近，有利于消除“盐”噪声。

实验结果：

原始图像

加入椒盐噪声后的图像

均值滤波后图像

中值滤波后图像

负阶数逆谐波滤波后图像

正阶数逆谐波滤波后图像

任务三

推导维纳滤波器并实现下边要求；

(a) 实现模糊滤波器如方程 Eq. (5.6-11).

(b) 模糊 lena 图像：45 度方向，T=1；

(c) 再模糊的 lena 图像中增加高斯噪声，均值= 0 ，方差=10 pixels 以产生模糊图像；

(d)分别利用方程 Eq. (5.8-6)和(5.9-4)，恢复图像；并分析算法的优缺点。

问题分析：

1）维纳滤波器推导

原图像： $f(x,y)$

退化函数： $g(x,y)=h(x,y)*f(x,y)+\eta(x,y)$

恢复图像： $\hat{f}(x,y)$

最小均方误差： $e^2(x,y)=E\left|e(x,y)\right|^2$

其中： $e(x,y)=f(x,y)-\hat{f}(x,y)$

$$E[e(n_1,n_2)g^*(m_1,m_2)]=0$$

故：
$$\begin{aligned}&E[f(n_1,n_2)g^*(m_1,m_2)]=E[(w(n_1,n_2)*g(n_1,n_2))g^*(m_1,m_2)]\\&=\sum_{k_1=-\infty}^{+\infty}\sum_{k_2=-\infty}^{+\infty}w(k_1,k_2)E[g(n_1-k_1,n_2-k_2)g^*(m_1,m_2)]\end{aligned}$$

令 $R_{fg}(n_1-m_1,n_2-m_2)=\sum_{k_1=-\infty}^{+\infty}\sum_{k_2=-\infty}^{+\infty}w(k_1,k_2)E[g(n_1-k_1,n_2-k_2)g^*(m_1,m_2)]$

$$R_{fg}=w(n_1,n_2)*R_g(n_1,n_2)$$

故： $W(w_1,w_2)=\dfrac{P_{fg}(w_1,w_2)}{P_g(w_1,w_2)}$

因为 f 噪声 n 无关

故： $E[f(n_1,n_2)\eta^*(m_1,m_2)]=E[(f(n_1,n_2)]E[\eta^*(m_1,m_2)]$

$$W(w_1,w_2)=\frac{P_f(w_1,w_2)}{P_f(w_1,w_2)+P_\eta(w_1,w_2)}$$

$$\hat{F}(w_1,w_2)=W'(w_1,w_2)G(w_1,w_2)$$

证得： $W'(w_1,w_2)=\dfrac{1}{H(w_1,w_2)}\dfrac{|F(w_1,w_2)|^2}{|F(w_1,w_2)|^2+|\dfrac{N(w_1,w_2)}{H(w_1,w_2)}|^2}$

2）模糊滤波器方程

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

3）维　滤波恢复模型

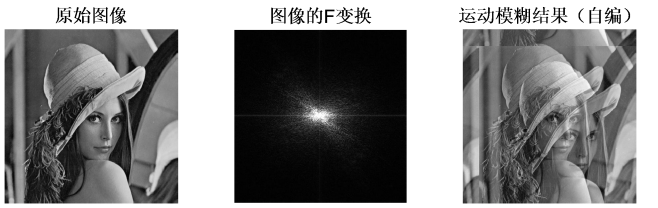
纳

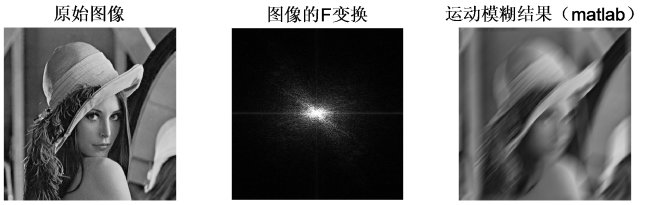
$$\hat{F}(u,v)=[\frac{1}{H(u,v)}\frac{|H(u,v)|^2}{|H(u,v)|^2+K}]G(u,v)$$

4）约束最小二乘方滤波模型

$$\hat{F}(u,v)=[\frac{H^*(u,v)}{|H(u,v)|^2+\gamma|P(u,v)|^2}]G(u,v)$$

实验结果：





结果分析：

1）首先分别通过自己编写的模糊函数和 MATLAB 中提供的 imfilter 和 fspecial 函数配合使用对图像 lena 进行了模糊滤波，得到的模糊

效果还是基本一致的。之后调用 imnoise 函数对两幅图像加入高斯噪声。

2）再分别使用自己编写的函数和 MATLAB 中提供的 deconvwnr 函数进行维纳滤波。调用 MATLAB 中函数滤波后的图像得到了一定的改善，运动模粉的影响基本被消除，但噪声的影响仍然较大，导致图像质量下降:对于自己编写的维纳滤波函数，难点在于寻找令信噪比最大的 K 值报告中显示了部分 K 值对应的滤波结果，其中 K=0.026.为信噪比最大时的滤波结果，从结果看，视觉上的效果并不是很理想，要想达到更好的效果可能需要寻找更加合适的 K 值。

3)最后采用 MATLAB 提供的 decovreg 函数进行约束最小二乘方滤波。从滤波后的结果看，约束最小二乘方滤波得到了比维纳滤波更好的结果，尤其是对噪声的滤除。但是对于最小二乘方滤波的参数选取仍然存在一些疑问。

源代码：

```
%task1
I=imread('lena.bmp');
subplot(2,2,1)
imshow(I);
title('原始图像');
new_I=imnoise(I,'gaussian',0,0.01);
subplot(2,2,2)
imshow(new_I);
title('加入高斯噪声后的图像');

%滤波模板的选择
n=3;
A=fspecial('average',n);
I1=filter2(A,new_I)/255;
I2=medfilt2(new_I,[n,n]);
subplot(2,2,3)
imshow(I1);
title('均值滤波后图像');
subplot(2,2,4)
imshow(I2);
title('中值滤波后图像');
%task2
I=imread('lena.bmp');
figure(1)
subplot(2,3,1)
imshow(I);
title('原始图像');
new_I=double(imnoise(I,'salt & pepper',0.1));
subplot(2,3,2)
imshow(new_I,[]);
title('加入椒盐噪声后的图像');

%滤波模板的选择
n=3;
A=fspecial('average',n);
I1=filter2(A,new_I)/255;
I2=medfilt2(new_I,[n,n]);
subplot(2,3,3)
imshow(I1);
title('均值滤波后图像');
subplot(2,3,4)
imshow(I2,[]);
```

```
title('中值滤波后图像');

%滤波器阶数
Q1=-1.5;
Q2=1.5;
%I3=imfilter(new_I.^(Q+1),fspecial('average',3))./imfilter(new_I.
^Q,fspecial('average',3));

[M,N]=size(new_I);
I3= new_I(:,:);
for x=1+(n-1)/2:1:M-(n-1)/2
    for y=1+(n-1)/2:1:N-(n-1)/2
        pI=new_I(x-(n-1)/2:1:x+(n-1)/2,y-(n-1)/2:1:y+(n-1)/2);
        I3(x,y) = sum(pI(:).^(Q1+1))/sum(pI(:).^(Q1));
        I4(x,y)=sum(pI(:).^(Q2+1))/sum(pI(:).^(Q2));
    end
end

subplot(2,3,5);
imshow(I3,[]);
title('负阶数逆谐波滤波后图像');
subplot(2,3,6);
imshow(I4,[]);
title('正阶数逆谐波滤波后图像');
%task3（MATLAB）
clear all
clc
I=imread('lena.bmp');
H=fspecial('motion',50,45);
I1=imfilter(I,H,'circular','conv');
subplot(2,3,1)
imshow(I);
title('原始图像');
f=double(I);
F=fft2(f);
F=fftshift(F);
A=abs(F);
A=(A-min(min(A)))/(max(max(A))-min(min(A)))*255;
subplot(2,3,2);
imshow(A);
title('图像的 F 变换');
subplot(2,3,3)
imshow(I1);
title('运动模糊结果（matlab）');
```

```
I2=imnoise(I1,'gaussian',0,0.01);
subplot(2,3,4)
imshow(I2);
title('加入高斯噪声后的图像');
noise=imnoise(zeros(size(I)),'gaussian',0,0.01);
NSR=sum(noise(:).^2)/sum(im2double(I(:)).^2);
I3=deconvwnr(I2,H,NSR);
subplot(2,3,5);
imshow(I3);
title('维纳滤波结果（matlab）');
V=0.01;
NP=V*prod(size(I2));
I4=deconvreg(I2,H,NP);
subplot(2,3,6);
imshow(I4);
title('最小二乘滤波结果（matlab）');
%task3（自编）
clear all
clc
I=imread('lena.bmp');
subplot(2,3,1);
imshow(I);
title('原始图像');
f=double(I);
F=fft2(f);
F=fftshift(F);
A=abs(F);
A=(A-min(min(A)))/(max(max(A))-min(min(A)))*255;
subplot(2,3,2);
imshow(A);
title('图像的 F 变换');
[P0,Q0]=size(F);
a=0.1;
b=0.1;
T=1;
K=0.026;

for u=1:1:P0
    for v=1:1:Q0

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)*F(u,v);
    end
```

```
end
G=ifftshift(G);
g=ifft2(G);
g=256.*g./max(max(g));
%g=(g-min(min(g)))/(max(max(g))-min(min(g)))*255;
g=uint8(real(g));
subplot(2,3,3);
imshow(g);
title('运动模糊结果（自编）');

new_I=imnoise(g,'gaussian',0,0.01);
subplot(2,3,4)
imshow(new_I);
title('加入高斯噪声后的图像');


g1=double(new_I);
G1=fft2(g1);
G1=fftshift(G1);
[P1,Q1]=size(G1);
for u=1:1:P1
    for v=1:1:Q1

F1(u,v)=((1/H(u,v))*(abs(H(u,v)))^2/((abs(H(u,v)))^2+K))*G1(u,v);
    end
end
F1=ifftshift(F1);
f1=ifft2(F1);
f1=256.*f1./max(max(f1));
%f1=(f1-min(min(f1)))/(max(max(f1))-min(min(f1)))*255;
f1=uint8(real(f1));
subplot(2,3,5);
imshow(f1);
title('维纳滤波结果（自编）');

K2=0.00001;
g2=double(new_I);
G2=fft2(g2);
G2=fftshift(G2);
[P2,Q2]=size(G2);
p0=[0 -1 0;-1 4 -1;0 -1 0];
p=zeros(P2,Q2);
p(1:3,1:3)=p0;
P=fft2(p);
```

```
P=fftshift(P);
%P=psf2otf(p0,[P2,Q2]);
for u=1:1:P2
    for v=1:1:Q2

F2(u,v)=(1/H(u,v))*(abs(H(u,v)))^2/((abs(H(u,v)))^2+K2*(abs(P(u,v)
))^2)*G2(u,v);
    end
end
F2=ifftshift(F2);
f2=ifft2(F2);
f2=256.*f2./max(max(f2));
f2=uint8(real(f2));
subplot(2,3,6);
imshow(f2);
title('最小二乘滤波结果（自编）');
```