数字图像处理作业三

夏厚 PB18051031

2021 年 5 月 3 日

1 实验原理

1.1 逆滤波

当图像退化函数 $H(u,v)$ 已经给出，最简单的复原方法就是直接做逆滤波。使用退化函数除图像的傅里叶变换 $G(u,v)$ 来计算原始图像傅里叶变换的估计 $\hat{F}(u,v)$，即

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

考虑到图像中的噪声，所以 $G(u,v) = H(u,v)F(u,v) + N(u,v)$，得到

$$\hat{F}(u,v) = F(u,v) + \frac{N(u,v)}{H(u,v)}$$

从该表达式可以看到，即使知道退化函数，也不能准确地复原未退化地图像，因为 $N(u,v)$ 未知。此外，如果退化函数是零或者是非常小的值，则 $N(u,v)/H(u,v)$ 将会很大，使得后者很容易支配估计值 $\hat{F}(u,v)$。

为了解决退化函数为零或者很小的情况，可以限制滤波的频率，使其接近原点。因为我们知道 $H(0,0)$ 通常是 $H(u,v)$ 的最高值。因此，通过将频率限制在原点附近分析，就减少了遇到零值的概率。退化函数是

$$H(u,v) = e^{-k[(u-M/2)^2+(v-N/2)^2]^{5/6}}$$

式中 k=0.0025。对应大气湍流模型的剧烈湍流。常数 M/2 和 N/2 是偏移值；它们使函数居中，以便与居中的傅里叶变换对应。

1.2 维纳滤波

运动模糊的退化函数

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

其中 T 为移动的时间，a 为 x 方向的最大移动距离占整个图片尺寸的比例，b 为 y 方向的最大移动距离占整个图片尺寸的比例。

维纳滤波是一种综合退化函数和噪声统计特征进行复原处理的方法。该方法建立在图像和噪声都是随机变量的基础上，目标是找到未污染图像 f 的一个估计 $\hat{f}$，使它们之间的均方误差最小。误差度量为：

$$e^2=E(f-\hat{f})^2$$

经常使用下面的表达式来近似估计：

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

式中，K 是一个加到 $|H(u,v)|^2$ 的所有项上的特定常数。K 值的选择是为了找到最好的视觉效果。

2 实验内容与结果分析

2.1 重现图 5.28 的结果

图 1: 退化的图像与全逆滤波图像

图 2: 半径受限逆滤波图像与维纳滤波图像

对于实验所给的原图，在程序中首先对图像以退化函数 $H(u,v)$ 进行退化，并叠加一个均值为 0，方差为 0.001 的高斯噪声。

$$H(u,v) = e^{-k[(u-M/2)^2+(v-N/2)^2]^{5/6}}$$

然后对退化后的图像进行三种恢复，分别是：全逆滤波、半径受限的逆滤波和维纳滤波。其中半径受限逆滤波是对 $G(u,v)/H(u,v)$ 的结果进行截取，对这个比值应用一个阶数为 10 的 butterworth 低通滤波器。实验中尝试，对于所加的高斯噪声下，截取半径为 33 左右时效果较好。而书本上在半径为 70 时效果最好，这是因为所加噪声可能不相同。截取半径小时，图像丢失高频成分，变得模糊；截取半径大时，图像开始退化，这是因为噪声开始支配着结果。

```
1          clc;
2          clear;
3          close all;
4
5          filename = 'demo-1'; %测试图像1
6          %filename = 'demo-2'; %测试图像2
7          im = (imread([filename, '.jpg']));
8          im = im2gray(im);
9          [M,N]=size(im);
10         F=fftshift(fft2(im));
11
```

```
12          for u=1:M
13              for v=1:N
14                  H(u,v)=exp(-0.0025*((u-M/2)^2+(v-N/2)^2)^(5/6));
15              end
16          end
17
18          %k=0.0025的剧烈湍流
19          T=F.*H;
20          I1=real(ifft2(fftshift(T)));
21          subplot(2,2,1);imshow(im2uint8(mat2gray(I1)));
22          title('剧烈湍流');
23          imwrite(I1, sprintf('result/%s.jpg', '湍流'));
24
25          %加入高斯噪声
26          I1=imnoise(uint8(im),'gaussian',0,0.001);
27
28          G=fftshift(fft2(I1));
29          F_hat=G./H;
30          I2=real(ifft2(fftshift(F_hat)));
31          subplot(2,2,2);imshow(uint8(255.*mat2gray(I2)));
32          title('全逆滤波');
33          imwrite(I2, sprintf('result/%s.jpg', '全逆'));
34
35          %butterworth滤波
36          for u=1:M
37              for v=1:N
38                  d=sqrt((u-M/2)^2+(v-N/2)^2);
39                  h=1/(1+0.414*(d/34)^(2*10));
40                  F_hat(u,v)=h*F_hat(u,v);
41              end
42          end
43          I2=real(ifft2(fftshift(F_hat)));
44          subplot(2,2,3);imshow(im2uint8(mat2gray(I2)));
```

```
45          title('半径受限逆滤波');
46          imwrite(I2, sprintf('result/%s.jpg', '逆滤波'));
47
48          %维纳滤波
49          K=0.1;
50          for u=1:M
51              for v=1:N
52                  H0(u,v)=(abs(H(u,v)))^2;
53                  H(u,v)=H0(u,v)/(H(u,v)*(H0(u,v)+K));
54              end
55          end
56          F=G.*H;
57          I3=real(ifft2(fftshift(F)));
58          subplot(2,2,4);imshow(im2uint8(mat2gray(I3)));
59          title('维纳滤波');
60          imwrite(I3, sprintf('result/%s.jpg', '维纳'));
```

2.2 重现图 5.29 的结果

首先对所给原图，按实验原理所述的运动模糊退化函数，进行运动模糊化，并加入一个加性高斯噪声。对退化的图像分别使用直接逆滤波与维纳滤波。

可以看到噪声较大时，逆滤波产生了一幅不太适用的图像。滤波后的噪声非常强，以至于其结构就在模糊滤波器的方向上。维纳滤波的结果也并不完美，但能较好的去掉运动带来的模糊。噪声较小时，直接逆滤波噪声明显，且在运动模糊的方向上呈现，但图像也能清晰可见，逆滤波对于消除运动模糊的能力还是很强的，只是受限与噪声。而维纳滤波的结果很好。

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

```
1           clear;close all;clc;
2           I = im2double(imread(['demo-2','.jpg']));
3           [M,N] = size(I);
```

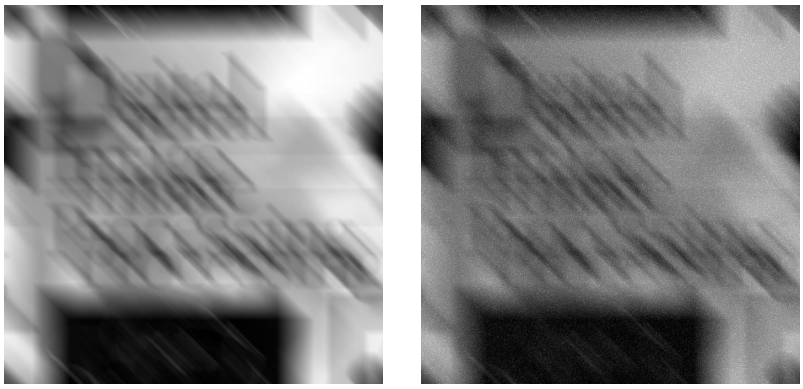

图 3: 运动模糊图像与运动模糊加噪图像

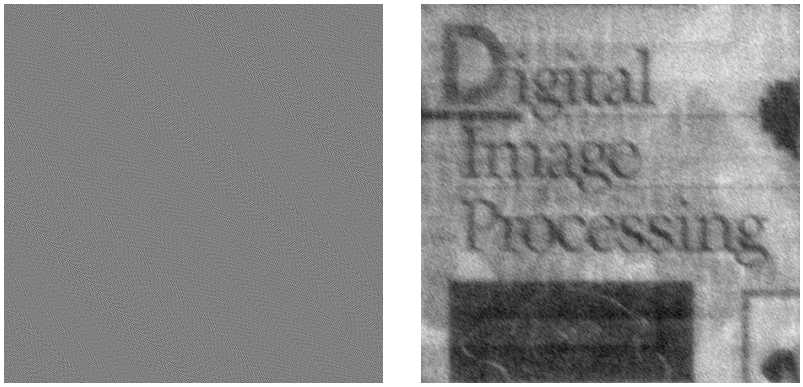

图 4: 逆滤波图像与逆滤波图像

```
%运动模糊退化函数
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(-1i*pi.*A);
H(A==0)=T;

%运动模糊图像
F=fftshift(fft2(I));
FBlurred=F.*H;
IBlurred =real(ifft2(ifftshift(FBlurred)));
```

```
15          subplot(1,4,1), imshow(uint8(255.*mat2gray(IBlurred)));
16          title('运动模糊图像');
17          %imwrite(IBlurred,sprintf('result/%s.jpg','运动模糊'));
18
19          %加入高斯噪声
20          noise=imnoise(zeros(M),'gaussian',0,0.0001);
21          FNoise=fftshift(fft2(noise));
22          FBlurred_Noised=FNoise+FBlurred;
23          IBlurred_Noised=real(ifft2(ifftshift(FBlurred_Noised)));
24          subplot(1,4,2), imshow(uint8(255.*mat2gray(IBlurred_Noised)));
25          title('运动模糊加噪图像');
26          imwrite(IBlurred_Noised,sprintf('result/%s.jpg','运动模糊加噪'));
27
28          %逆滤波
29          FDeblurred=FBlurred_Noised./H;
30          IDeblurred=real(ifft2(ifftshift(FDeblurred)));
31          subplot(1,4,3), imshow(uint8(255.*mat2gray(IDeblurred)));
32          title('逆滤波');
33          % imwrite(IDeblurred,sprintf('result/%s.jpg','逆滤波'));
34
35          %维纳滤波
36          K=0.005;
37          for u=1 : 480
38              for v=1 : 480
39                  H0(u,v)=(abs(H(u,v)))^2;
40                  H(u,v)=H0(u,v)/(H(u,v)*(H0(u,v)+K));
41              end
42          end
43          F=FBlurred_Noised.*H;
44          I3=real(ifft2(fftshift(F)));
45          subplot(1,4,4);imshow(im2uint8(mat2gray(I3)));
46          title('维纳滤波');
47          % imwrite(I3, sprintf('result/%s.jpg', '维纳滤波'));
```