Date: Page: Weather:

解：原图像为 $B=\begin{bmatrix} 1 & 2 & 1 & 4 & 3 \\ 1 & 10 & 2 & 3 & 4 \\ 5 & 2 & 6 & 8 & 8 \\ 5 & 5 & 7 & 0 & 8 \\ 5 & 6 & 7 & 8 & 9 \end{bmatrix}$

用3×3的均值滤波器处理

$z_1=\frac{1+2+1+1+10+2+5+2+6}{9}=3.33$　　$z_2=\frac{2+1+4+10+2+3+2+6+8}{9}=4.22$

$z_3=\frac{1+4+3+2+3+4+6+8+8}{9}=4.33$　　$z_4=\frac{1+10+2+5+2+6+5+5+7}{9}=4.77$

$z_5=\frac{10+2+3+2+6+8+5+7+0}{9}=4.77$　　$z_6=\frac{2+3+4+6+8+8+7+0+8}{9}=5.11$

$z_7=\frac{5+2+6+5+5+7+5+6+7}{9}=5.33$　　$z_8=\frac{2+6+8+5+7+0+6+7+8}{9}=5.44$

$z_9=\frac{6+8+8+7+0+8+7+8+9}{9}=6.77$

故 处理后图像为 $\begin{bmatrix} 1 & 2 & 1 & 4 & 3 \\ 1 & 3.33 & 4.22 & 4.33 & 4 \\ 5 & 4.77 & 4.77 & 5.11 & 8 \\ 5 & 5.33 & 5.44 & 6.77 & 8 \\ 5 & 6 & 7 & 8 & 9 \end{bmatrix}$

用3×3的中值滤波器处理

$z_1=2$　　$z_2=3$　　$z_3=4$　　$z_4=5$　　$z_5=5$

$z_6=6$　　$z_7=5$　　$z_8=6$　　$z_9=8$

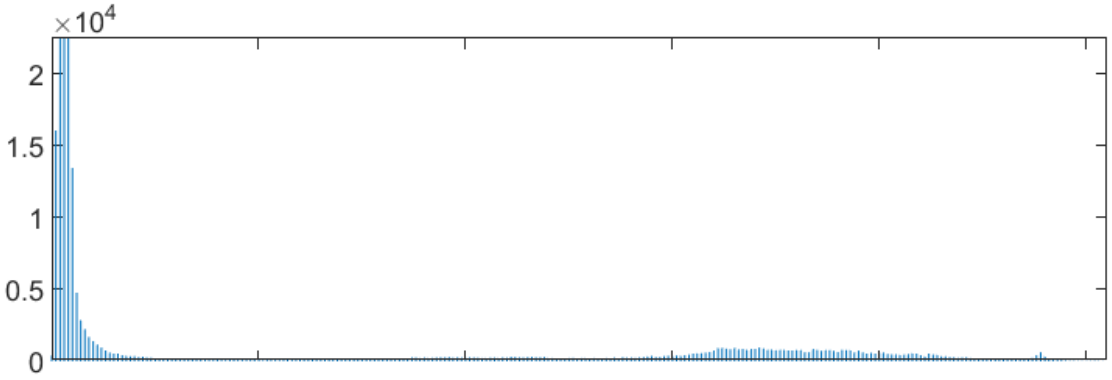
故 处理后图像为 $\begin{bmatrix} 1 & 2 & 1 & 4 & 3 \\ 1 & 2 & 3 & 4 & 4 \\ 5 & 5 & 5 & 6 & 8 \\ 5 & 5 & 6 & 8 & 8 \\ 5 & 6 & 7 & 8 & 9 \end{bmatrix}$

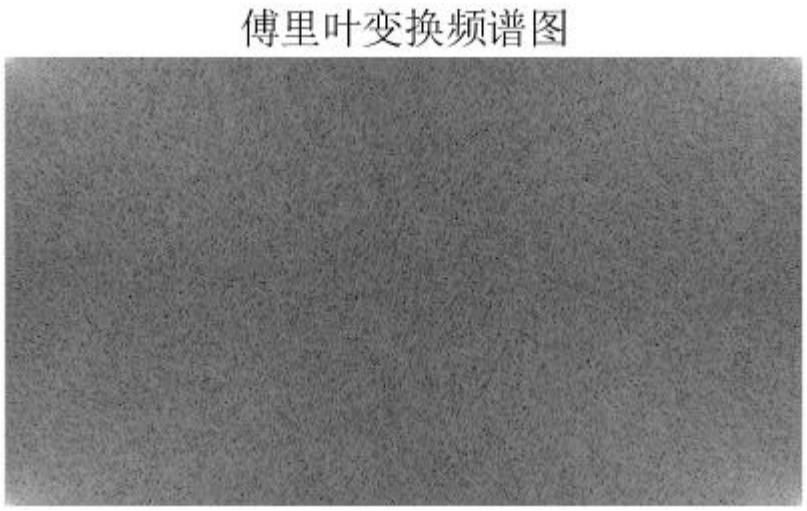
```
RGB = imread("微信图片_20241013160829.jpg");     %读取低照度图像

I = rgb2gray(RGB);
imwrite(I,'01.png')

subplot(2,1,1); imhist(I);     %计算灰度直方图

f = imread("01.png");
F = fft2(f);
F1 = log(abs(F)+1);    %取模并进行缩放
subplot(2,1,2); imshow(F1,[]); title('傅里叶变换频谱图'); %傅里叶变换频谱图
```





```m
% 首先读入灰度图像，并提取图像的高度和宽度
image = imread('01.png');
[height, width] = size(image);

% 然后统计每个灰度的像素值的累计数目
NumPixel = zeros(1,256);  % 建立一个256列的行向量，以统计各灰度级的像素个数
for i = 1 : height
   for j = 1 : width
       k = image(i,j);  % k是像素点(i,j)的灰度值
       % 因为NumPixel数组的下标是从1开始的，但是图像像素的取值范围是0~255
       % 所以用NumPixel(k+1)
       NumPixel(k+1) = NumPixel(k+1) + 1;  % 对应灰度值像素点数量加1
   end
end

% 用频数值计算频率
ProbPixel = zeros(1,256); % 统计各灰度级出现的频率
for i = 1 : 256
    ProbPixel(i) = NumPixel(i) / (height * width);
end

% 再用函数cumsum()来计算累积分布函数（CDF），并将频率（取值范围是0~1）映射到0~255的无符号整数
CumPixel = cumsum(ProbPixel);  % 这里的数组CumPixel大小也是1×256
CumPixel = uint8((256-1) .* CumPixel + 0.5);

% 在下列用作直方图均衡化实现的赋值语句右端，image(i,j)被用来作为CumPixel的索引
outImage = uint8(zeros(height, width));  % 预分配数组
for i = 1 : height
   for j = 1 : width
      outImage(i,j) = CumPixel(image(i,j)+1);  %防止出现数组下标非正整数
   end
end

% 显示直方图均衡化前后的图像，可以发现，与调用函数histeq()的效果基本一致
subplot(1,2,1); imshow(image); title('原图');
subplot(1,2,2); imshow(outImage); title('均衡化后');
```

```
I = imread('01.png');
figure(1),subplot(1,2,1),imshow(I); title('原图')

I = im2double(I);    %转换数据类型为double型
[Height,Width] = size(I);

CenterY = floor(Height/2);
CenterX = floor(Width/2);


I2 = log(I+1);  %取对数
FI = fft2(I2);   %傅里叶变换
FI = fftshift(FI);

rL = 0.4;
rH = 2.2;
c = 4.0;     %锐化参数
D0 = 10;


for Y = 1:Height
    for X = 1:Width
        D(Y,X) = sqrt(((Y-CenterY).^2+(X-CenterX).^2));        %频率域中点(u,v)与频率矩形中心的距离
        H(Y,X) = (rH-rL).*(1-exp(-c.*(D(Y,X)^2./D0^2)))+rL;    %高斯同态滤波
    end
end

FI = H.*FI;
FI = ifftshift(FI);     %对H做反中心化
FI = ifft2(FI);   %傅里叶逆变换

OUT = exp(FI)-1;    %取指数
figure(1),subplot(1,2,2),imshow(OUT,[]);title('同态滤波')
```