数字图像处理 第二次作业

王亦飞 U202210371 人工智能 2204

[理论作业]

创建初始矩阵与输出矩阵

```
编辑器 - C:\Users\DELL\Documents\MATLAB\sztxcl2.m
sztxcl2.m
1  A=[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];
2  [m,n]=size(A);
3  % 初始化输出矩阵
4  mean_filtered = zeros(m, n);
5  median_filtered = zeros(m, n);
```

均值滤波

```
7  % 手动实现均值滤波
8  for i = 1:m
9      for j= 1:n
10         if i<2||i>m-1||j<2||j>n-1
11         % 边缘像素直接复制
12         mean_filtered (i, j) = A(i, j);
13         else
14         % 提取3x3的邻域
15         neighborhood = A(i-1:i+1, j-1:j+1);
16         % 计算平均值并赋值给中心像素
17         mean_filtered(i, j) = round (mean(neighborhood (:)));
18         end
19     end
20 end
```

中值滤波

```
22 % 手动实现中值滤波
23 for i = 1:m
24     for j = 1:n
25         if i<2||i>m-1||j<2||j>n-1
26             % 边缘像素直接复制
27             median_filtered(i,j)= A(i,j);
28         else
29             %提取3x3的邻域
30             neighborhood = A(i-1:i+1, j-1:j+1);
31             median_filtered(i, j) = median(neighborhood(:));
32         end
33     end
34 end
```

输出图像

```
35
36 %初始图像
37 A
38 %均值滤波后的图像
39 mean_filtered
40 %中值滤波后的图像
41 median_filtered
```

代码：

```
A=[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];
[m,n]=size(A);
% 初始化输出矩阵
mean_filtered = zeros(m, n);
median_filtered = zeros(m, n);

% 手动实现均值滤波
for i = 1:m
    for j= 1:n
        if i<2||i>m-1||j<2||j>n-1
        % 边缘像素直接复制
        mean_filtered (i, j) = A(i, j);
        else
        % 提取 3x3 的邻域
        neighborhood = A(i-1:i+1, j-1:j+1);
        % 计算平均值并斌值给中心像素
        mean_filtered(i, j) = round (mean(neighborhood (:)));
        end
    end
end

% 手动实现中值滤波
for i = 1:m
    for j = 1:n
        if i<2||i>m-1||j<2||j>n-1
            % 边缘像素直接复制
            median_filtered(i,j)= A(i,j);
        else
            %提取 3x3 的邻域
            neighborhood = A(i-1:i+1, j-1:j+1);
            median_filtered(i, j) = median(neighborhood(:));
        end
    end
end

%初始图像
A
%均值滤波后的图像
mean_filtered
%中值滤波后的图像
median_filtered
```

输出结果

编辑器 - C:\Users\DELL\Documents\MATLAB\sztxcl2.m

命令行窗口

```
>> sztxcl2

A =

     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


mean_filtered =

     1     2     1     4     3
     1     3     4     4     4
     5     5     5     5     8
     5     5     5     7     8
     5     6     7     8     9


median_filtered =

     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

fx >>
```

编程作业

读取图像并进行灰度化

编辑器 - C:\Users\DELL\Documents\MATLAB\sztxcl2_2.m

sztxcl2_2.m | sztxcl_1.m

```
I=imread ('C:\Users\DELL\Documents\MATLAB\418cda2b8a5fb22232e168608b4f618.png');
if size(I, 3) == 3
    I_gray = rgb2gray(I);
else
    I_gray = I;
end
```

计算灰度直方图

```
figure;
subplot (1,3,1);
imshow(I_gray);
title('Original Gray Image');
subplot (1,3,2);
imhist(I_gray);
title('Gray Histogram');
```

计算离散傅里叶变换

```
%将灰度图像转换为双精度浮点数,进行傅里叶变换
I_double = double(I_gray);
%计算图像的二维快速傅里叶变换 (FFT)
I_fft = fft2(I_double);
% 将FFT结果的直流分量移到频谱的中心
I_fft_shift = fftshift(I_fft);
% 计算频谱的幅度
I_fft_magnitude = abs(I_fft_shift);
% 最示频谱的对数幅度图
figure;
imshow(log(1 + I_fft_magnitude), []);
title('FFT Magnitude Spectrum') ;
```

直方图均衡化

```
% 计算累积分布函数（CDF）
cdf = cumsum(counts) / numel(I_gray);
% 将 CDF 映射到新的灰度级
I_histeq = uint8(gray_levels(1 + (cdf * (numel(gray_levels) - 1))));
% 显示直方图均衡化后的图像
figure;
imshow(I_histeq);
title('Manual Histogram Equalization Result');
```

同态滤波

```
%设置同态滤波参数
rH=2; %高频增益
rL=0.2; %低频增益
c=1.5; %锐化系数
D0= 80; %截止频率
% 对数变换，将乘性噪声转换为加性噪声
I_log = log(double (I_gray) +1);
% 对图像进行零填充，以避免 wrap-around effects
[M, N] = size(I_log);
P = 2^nextpow2 (M);
Q = 2^nextpow2 (N) ;
I_padded = padarray(I_log, [P-M, Q-N], 'post');
%计算频域滤波器
[X,Y]=meshgrid(0:Q-1, 0:P-1);
D = sqrt ((X - round (P/2)).^2 + (Y-round (Q/2)).^2);
H = (rH - rL).* (1-exp(-C*(D.^2) / (D0^2))) + rL;
H = ifftshift(H);% 反中心化滤波器
```

```
% 对图像进行傅里叶变换
I_fft_padded = fft2(I_padded);
% 应用同态滤波器
I_filtered = I_fft_padded.* H;
% 进行逆傅里叶变换
I_filtered = ifft2(I_filtered);
% 收实部，并进行指数变换以恢复图像
I_homo = exp(real(I_filtered (1:M, 1:N)))-1;
% 显示同态滤波后的图像
figure;
imshow(I_homo);
title ('Manual Homomorphic Filtering Result');
```

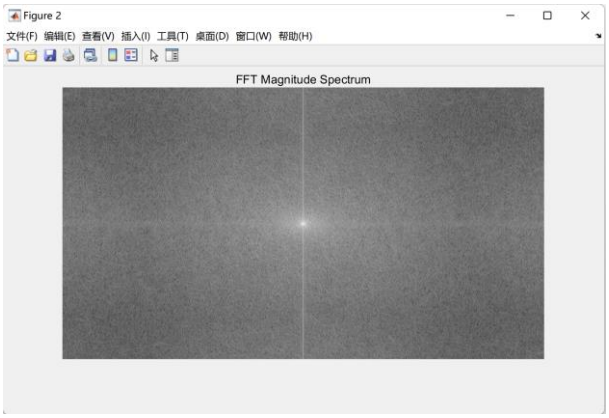
代码

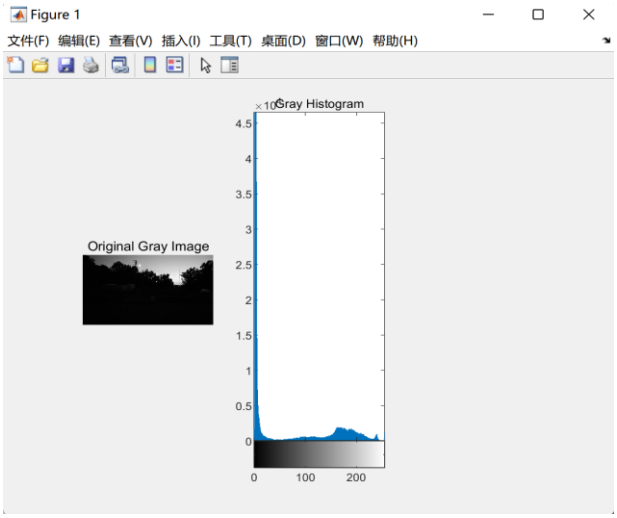
```
I=imread
('C:\Users\DELL\Documents\MATLAB\418cda2b8a5fb22232e168608b4f618.png');
if size(I, 3) == 3
    I_gray = rgb2gray(I);
else
    I_gray = I;
end

figure;
subplot (1,3,1);
imshow(I_gray);
title('Original Gray Image');
subplot (1,3,2);
imhist(I_gray);
title('Gray Histogram');

%将灰度图像转换为双精度浮点数,进行傅里叶变换
```

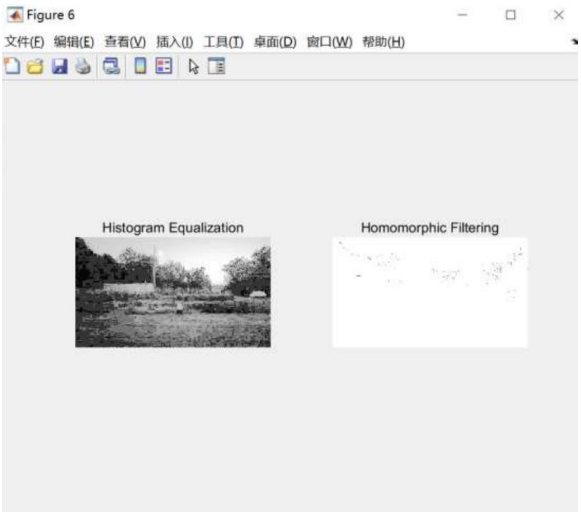
```
I_double = double(I_gray);
%计算图像的二维快速傅里叶变换 (FFT)
I_fft = fft2(I_double);
% 将 FFT 结果的直流分量移到频谱的中心
I_fft_shift = fftshift(I_fft);
% 计算频谱的幅度
I_fft_magnitude = abs(I_fft_shift);
% 最示频谱的对数幅度图
figure;
imshow(log(1 + I_fft_magnitude), []);
title('FFT Magnitude Spectrum') ;

% 计算累积分布函数（CDF）
cdf = cumsum(counts) / numel(I_gray);
% 将 CDF 映射到新的灰度级
I_histeq = uint8(gray_levels(1 + (cdf * (numel(gray_levels) - 1))));
% 显示直方图均衡化后的图像
figure;
imshow(I_histeq);
title('Manual Histogram Equalization Result');

%设置同态滤波参数
rH=2; %高频增益
rL=0.2; %低频增益
c=1.5; %锐化系数
D0= 80; %截止频率
% 对数变换，将乘性噪声转换为加性噪声
I_log = log(double (I_gray) +1);
% 对图像进行零填充，以避免 wrap-around effects
[M, N] = size(I_log);
P = 2^nextpow2 (M);
Q = 2^nextpow2 (N) ;
I_padded = padarray(I_log, [P-M, Q-N], 'post');
%计算频域滤波器
[X,Y]=meshgrid(0:Q-1, 0:P-1);
D = sqrt ((X - round (P/2)).^2 + (Y-round (Q/2)).^2);
H = (rH - rL).* (1-exp(-C*(D.^2) / (D0^2))) + rL;
H = ifftshift(H);% 反中心化滤波器

% 对图像进行傅里叶变换
I_fft_padded = fft2(I_padded);
% 应用同态滤波器
I_filtered = I_fft_padded.* H;
% 进行逆傅里叶变换
```

```
I_filtered = ifft2(I_filtered);
% 收实部，并进行指数变换以恢复图像
I_homo = exp(real(I_filtered (1:M, 1:N)))-1;
% 显示同态滤波后的图像
figure;
imshow(I_homo);
title ('Manual Homomorphic Filtering Result');
```

输出结果







电子签名