数字图像处理 Project

Project 04-01、Project 04-02 覃浩南
Project 04-03 检验谱平面旋转性质 孙奥远
Project 04-04 Project 04-05 施天予

Project 04-01

Two-Dimensional Fast Fourier Transform

原理

本题要求我们实现二维傅里叶变换，需满足题目的 5 个要求，并将其运用于频率域滤波。

频率域滤波基础 $f(x, y) \Leftrightarrow F(u, v)$

步骤:

(1) 用$(-1)^{x+y}$乘以输入图像, 做频谱中心化处理;

(2) 计算(1)结果的DFT, 即$F(u, v)$;

(3) 用滤波器函数$H(u, v)$乘以$F(u, v)$(在频谱域处理图像) - 滤波器函数下面讨论;

(4) 计算(3)中结果的反DFT;

(5) 得到(4)结果中的实部;

(6) 用$(-1)^{x+y}$乘以(5)中的结果

滤波和滤波器: 滤波顾名思义就是阻止或减少信号或图像中的某些频率成分. 滤波器(函数)就是能起到这样作用的函数. 一般表达式:

$$G(u, v) = H(u, v)F(u, v)$$

滤波后的结果图像可以从$G(u, v)$的反傅里叶变换得到.

$$g(x, y) = \Im^{-1}G(u, v)$$

原理

1. 将 F(0,0) 移动到 (M/2, N/2) 处：

根据二维离散傅里叶变换的平移不变性公式

$$f(x,y)e^{j2\pi(u_0x/M+v_0y/N)} \iff F(u-u_0,v-v_0)$$

$$f(x-x_0,y-y_0) \iff F(u,v)e^{-j2\pi(u_0x/M+v_0y/N)}$$

令 $(u_0,v_0)=(M/2,N/2)$，代入上述公式并使用欧拉公式推导，有

$$f(x,y)(-1)^{x+y} \iff F(u-M/2,v-N/2)$$

所以对 f(x,y) 乘上 $(-1)^{x+y}$ 就可以将 F(0,0) 移至中心 (M/2, N/2)

原理

2. 计算二维傅里叶变换和反变换：可以调用 MATLAB 内置函数 fft2 和 ifft2。

3. 这里我选用理想低通滤波器（ILPF）作为示例：

在以原点为圆心、以 D_0 为半径的圆内，无衰减地通过所有频率，而在该圆外“切断”所有频率的二维低通滤波器，称为理想低通滤波器(ILPF)；它由下面的函数确定：

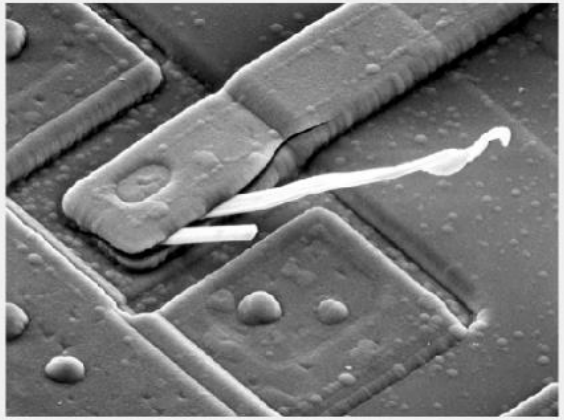
$$H(u,v)=\begin{cases}1, & D(u,v)\leqslant D_0\\ 0, & D(u,v)>D_0\end{cases} \tag{4.8-1}$$

其中，D_0是一个正常数，$D(u,v)$是频率域中点(u,v)与频率矩形中心的距离，即

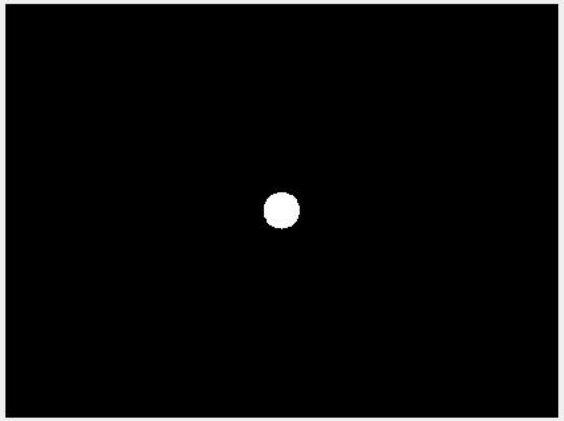
$$D(u,v)=\left[(u-P/2)^2+(v-Q/2)^2\right]^{1/2} \tag{4.8-2}$$

实验结果

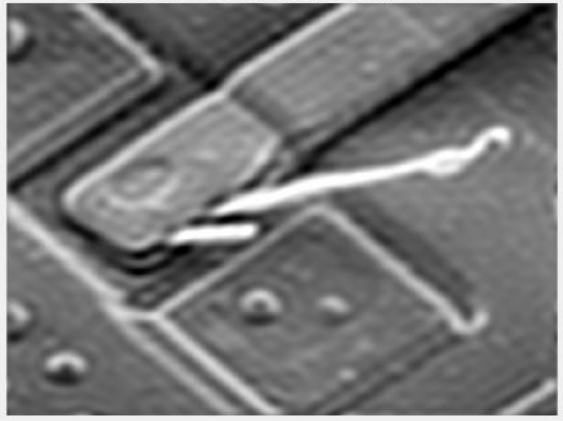
原图像



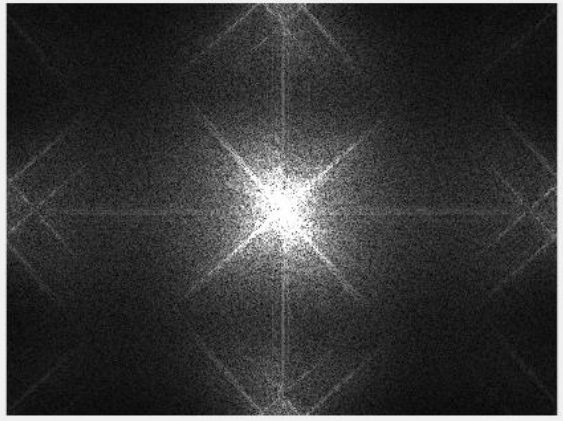
理想低通滤波器



取反变换的实部并乘以$(-1)^{(x+y)}$

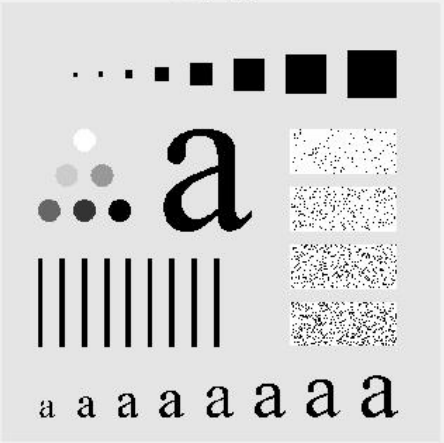


傅里叶频谱

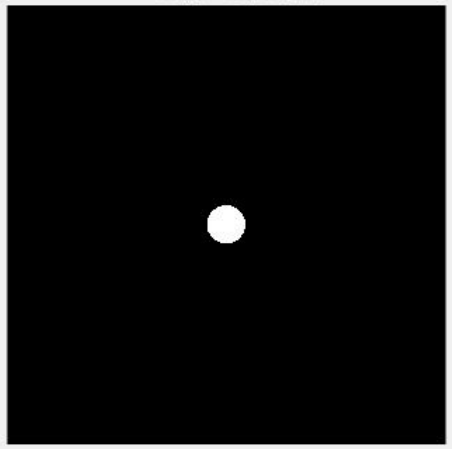


实验结果

原图像

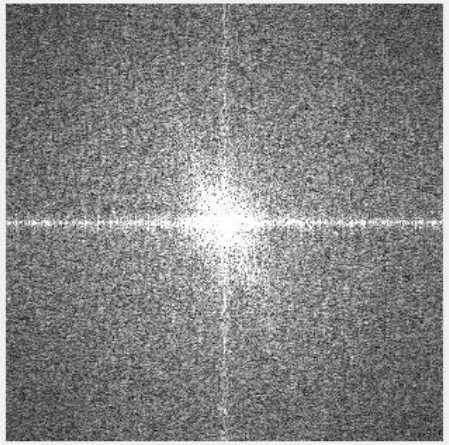


理想低通滤波器



取反变换的实部并乘以$(-1)^{(x+y)}$

傅里叶频谱



以上展示了傅里叶频谱以及反变换后的图像。

可以看到：傅里叶频谱的 F(0,0) 移到了中心；低通滤波器**模糊**了原图像，并且有明显的**振铃现象**。

```
f = imread('Fig0429(a)(blown_ic).tif');   % 读入图片
subplot(2,2,1); imshow(f); title('原图像');
fp = double(f);

% 用(-1)^(x+y)乘以原图像，再做傅里叶变换
fp = move_to_center(fp);
F = fft2(fp);

% 构建理想低通滤波器
[M,N] = size(f);
D0 = 30;    % 截止频率
H = zeros(M, N);
for u = 1:M
    for v = 1:N
        D = sqrt((u-M/2)^2 + (v-N/2)^2);
        if D <= D0
            H(u,v) = 1;
        else
            H(u,v) = 0;
        end
    end
end
```

```
subplot(2,2,2); imshow(H); title("理想低通滤波器");
G = H .* F;    % 滤波函数和图像的DFT对应元素点乘

% Compute the inverse Fourier transform
gp = ifft2(G);        % 傅里叶反变换

% Multiply the result by (-1)^(x+y) and take the real part.
gp = move_to_center(real(gp));
subplot(2,2,3); imshow(uint8(gp)); title("取反变换的实部并乘以(-
1)^{(x+y)}");

% Compute the spectrum
spec = spectrum(F);                %   根据第1步的结果计算复数的模
subplot(2,2,4); imshow(uint8(spec.^0.5)); title("傅里叶频谱");
```

```
% PROJECT 04-01 相关封装函数
function ret = move_to_center(img)
      [M,N] = size(img);
      [Y,X] = meshgrid(1:N,1:M);
      ones = (-1).^(X+Y);
      ret = ones.*img;
end

function ret = spectrum(X)
      ret = abs(X);       % 复数的模
end
```

Project 04-02

Fourier Spectrum and Average Value

原理

1. 使用 PROJECT 04-01中的函数 move_to_center 和内置的傅里叶变换函数 fft2，并不对结果取模，就可求得移到中心的傅里叶谱。

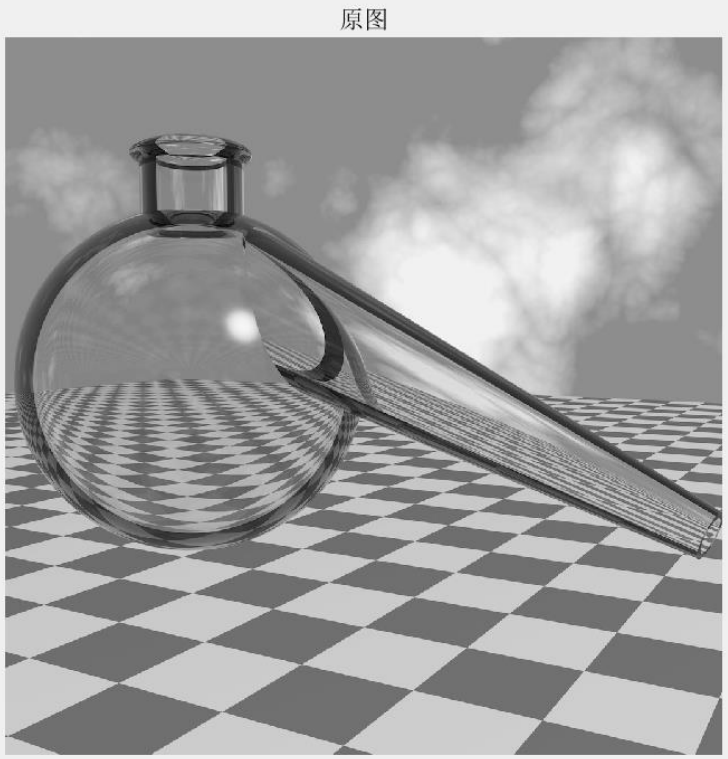
2. 求图像均值：

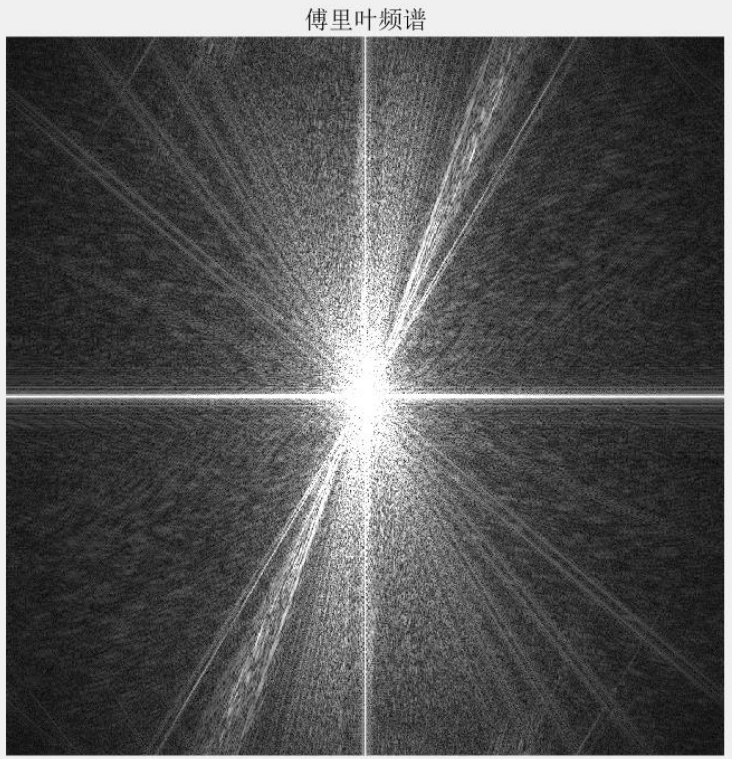
法一：对图像灰度值求和再除以 M*N 就可以得到图像的平均灰度值。

法二：由课本 P154 的公式可知，若不进行图像中心化直接对图像进行傅里叶变换，则图像均值为：

$$|\overline{f}(x,y)| = \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y) = \frac{1}{MN}F(0,0)$$

实验结果





MATLAB 程序中，两种方法计算出的平均灰度值均为 153.7417.

avg1	153.7417
avg2	153.7417

```
% a. compute its (centered) Fourier spectrum
f = imread('Fig0418(a)(ray_traced_bottle_original).tif');
fp = double(f);
F = fft2(move_to_center(fp));
spec = spectrum(F);

% b. Display the spectrum.
subplot(1,2,1); imshow(uint8(fp)); title('原图');
subplot(1,2,2); imshow(uint8(spec.^0.5)); title('傅里叶频谱');

% c compute the average value of the image
[M,N] = size(f);
s = sum(sum(f));          % 求和
avg1 = s / (M * N);       % 取平均

F2 = fft2(fp);    % 不乘(-1)^(x+y)，直接计算傅里叶变换
avg2 = F2(1,1) / (M * N);
```

Project 04-03

Lowpass Filtering

Lowpass Filtering Project04-03

问题描述

(a)实现Eq.(4.3-7)中的高斯低通滤波器
(b)下载图4.11(a)[与图4.18(a)相同]和低通对其进行滤波得到图4.18(c)。

解决思路

1.给定一副大小为$M * N$的输入图像f(x,y)，选择填充参数P和Q。

2.对f(x,y)添加必要数量的0，形成大小为$P * Q$的填充图像$f_p(x, y)$。

3.用$(-1)^{x+y}$乘以$f_p(x, y)$,移动到其变换的中心

4.计算步骤3得到的图像的DFT，得到F(u,v)

5.生成一个实的，对称的滤波函数H(u,v)，其大小为$P * Q$,中心在(P/2,Q/2)处，用阵列相乘得到乘积G(u,v) = H(u,v)F(u,v)。

6.对处理后的图像进行IDFT，得到结果的实部

7.用$(-1)^{x+y}$乘以步骤6得到图像$g_p(x, y)$

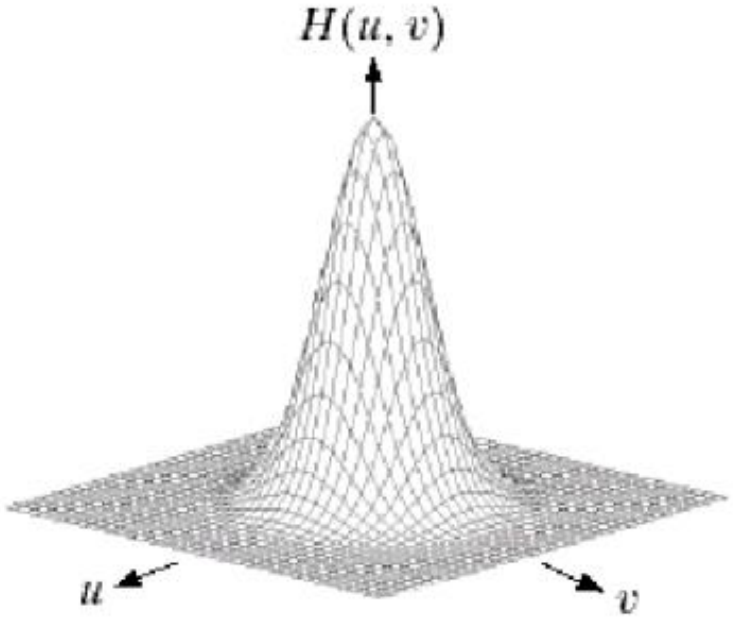
8.从$g_p(x, y)$的左上象限提取$M * N$区域，得到最终处理结果g(x,y)

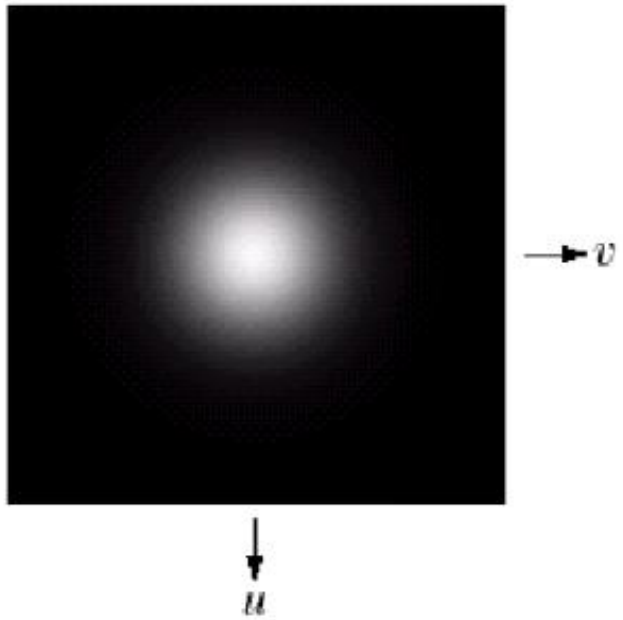
解决思路

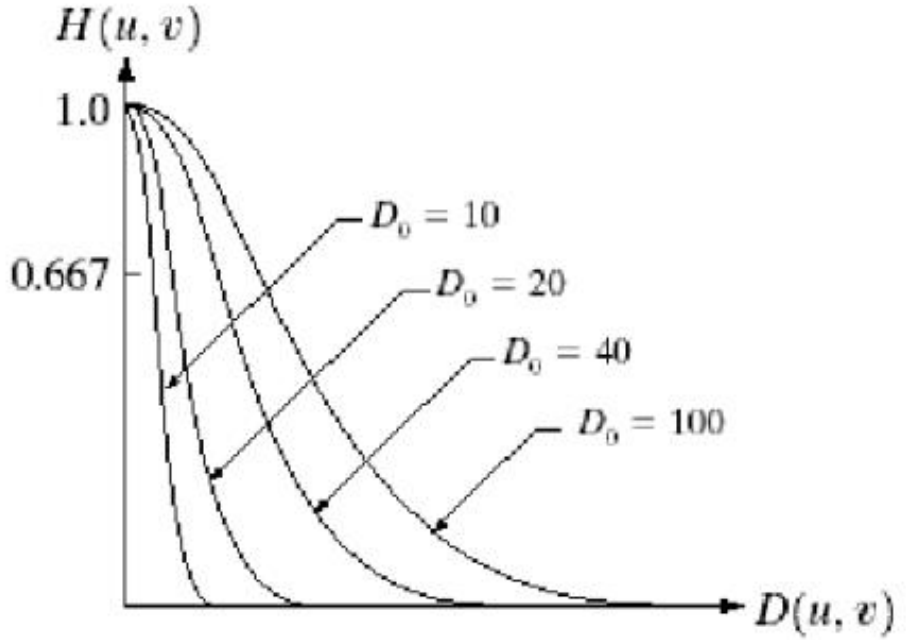
高斯低通滤波中的滤波器函数

$$H(u,v) = e^{-D^2(u,v)/2D_0^2}$$

其中D(u,v)是距频率矩形中心的距离，D_0为截止频率







代码展示

```
f = imread('Fig0418(a).tif');
f = double(f);
figure(1); imshow(uint8(f)); title('原图像');

fp = padding(f);%对原图像进行填充
fp = centralized(fp);%中心化处理

[P,Q] = size(fp);
[M,N] = size(f);
H = zeros(P,Q);%滤波器的大小要与填充后的图像的大小一致
D0 = 10;        %截止频率
for i=1:P       %计算高斯低通滤波器
    for j=1:Q
        H(i,j) = exp((-(i-M)^2-(j-N)^2)/(2*D0^2)); %D(i,j) = (i,j)到图像中心(M,N)的距离
    end
end
```

代码展示

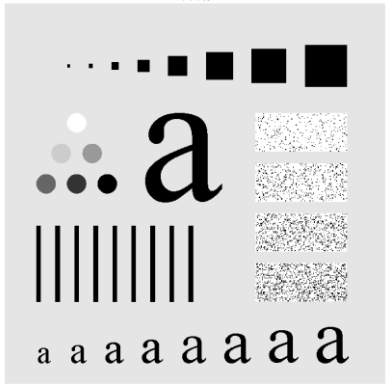
```
FLY = fft2(fp); %计算傅里叶变换
G = H.*FLY;     %计算滤波图像
g = real(ifft2(G)); %傅里叶反变换取实部
g = ones.*g;        %去中心化
g = g(1:M,1:N);

figure(2); imshow(H); title('高斯低通滤波器');
figure(3); imshow(uint8(g)); title('结果图像');
```

代码展示

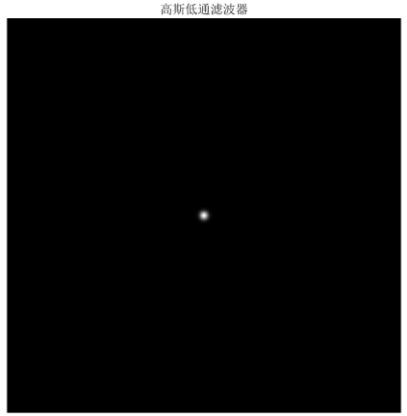
```
function fp=padding(f)%对原始图像进行零填充
    [M,N] = size(f);
    P = 2*M;
    Q = 2*N;
    fp = zeros(P,Q);
    fp(1:M,1:N) = f(1:M,1:N);
end

function center_fp = centralized(fp)%对图像fp进行中心化处理
    [P,Q] = size(fp);
    [Y,X] = meshgrid(1:Q,1:P);    %X,Y均为P*Q的矩阵
    ones = (-1).^(X+Y);
    center_fp = ones.*fp;         %为了使傅里叶变换后的图像的中心移动到图像的中心(P/2,Q/2)
end
```

运行结果

原图像



高斯低通滤波器



结果图像

检验谱平面随着图像旋转而旋转的性质

检验谱平面随着图像旋转而旋转的性质

解决思路

使用极坐标

$$x = rcos\theta, y = rsin\theta, u = \omega cos\varphi, v = \omega sin\varphi$$

可以得到

$$f(r, \theta + \theta_0) \Leftrightarrow F(\omega, \varphi + \theta_0)$$

将图像f(x, y)旋转θ角度，可以观察到其谱平面也旋转θ角度

代码展示

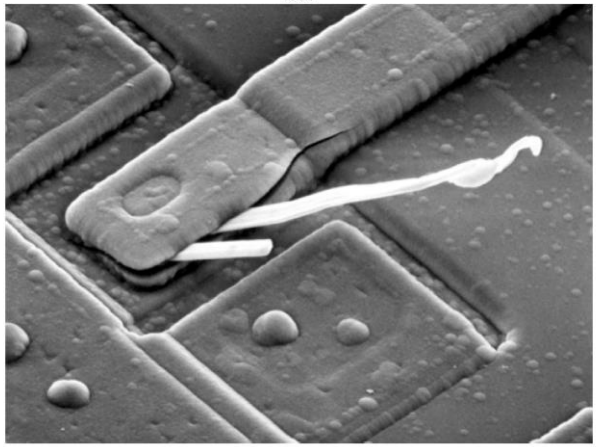
```
f = imread('Fig044.tif');
figure;imshow(uint8(f)); title('原图');

sp = rotation_fft(f,0);
figure;imshow(uint8(log(1+sp)),[]);title('旋转0度傅里叶频谱');

sp = rotation_fft(f,45);
figure;imshow(uint8(log(1+sp)),[]);title('旋转45度傅里叶频谱');

sp = rotation_fft(f,90);
figure;imshow(uint8(log(1+sp)),[]);title('旋转90度傅里叶频谱');
```

代码展示

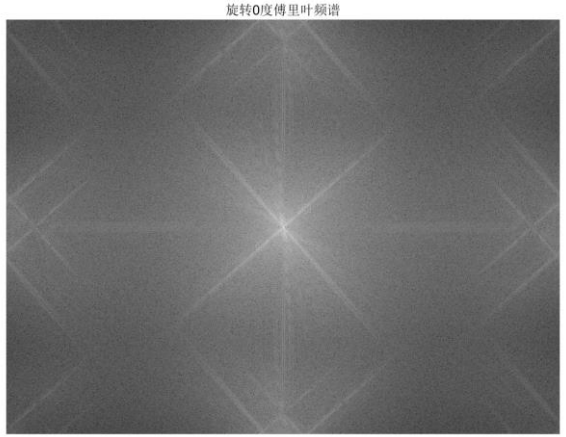
```
function sp = rotation_fft(f,degree)   %对图像进行逆时针旋转degree度之后，进行傅里叶变换，并得到频谱
    fi = imrotate(f,degree);           %将图像旋转degree
    fi = double(fi);
    [M,N] = size(fi);
    [Y,X] = meshgrid(1:N,1:M);         %X,Y均为M*N为矩阵，X(x,y)+Y(x,y)=x+y
    ones = (-1).^(X+Y);
    fi = ones.*fi;                      %中心化
    FLY = fft2(fi);
    sp = abs(FLY);%可能超过255
end
```

运行结果

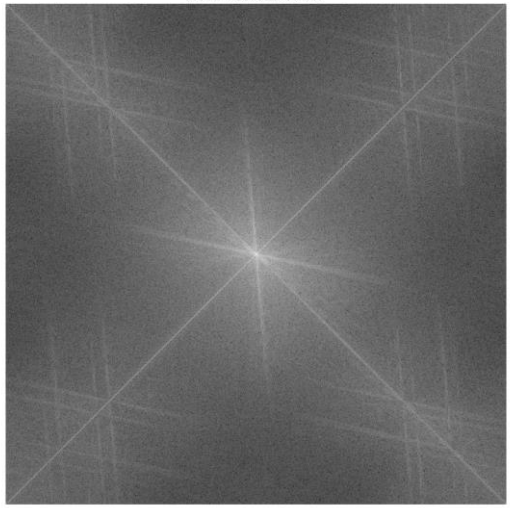
原图



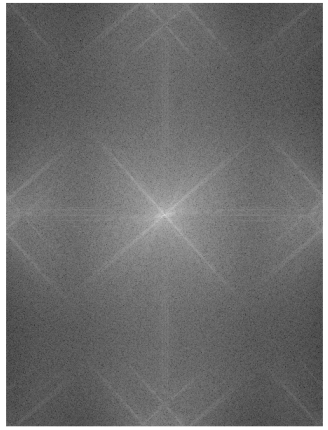
旋转0度傅里叶频谱



旋转45度傅里叶频谱



旋转90度傅里叶频谱

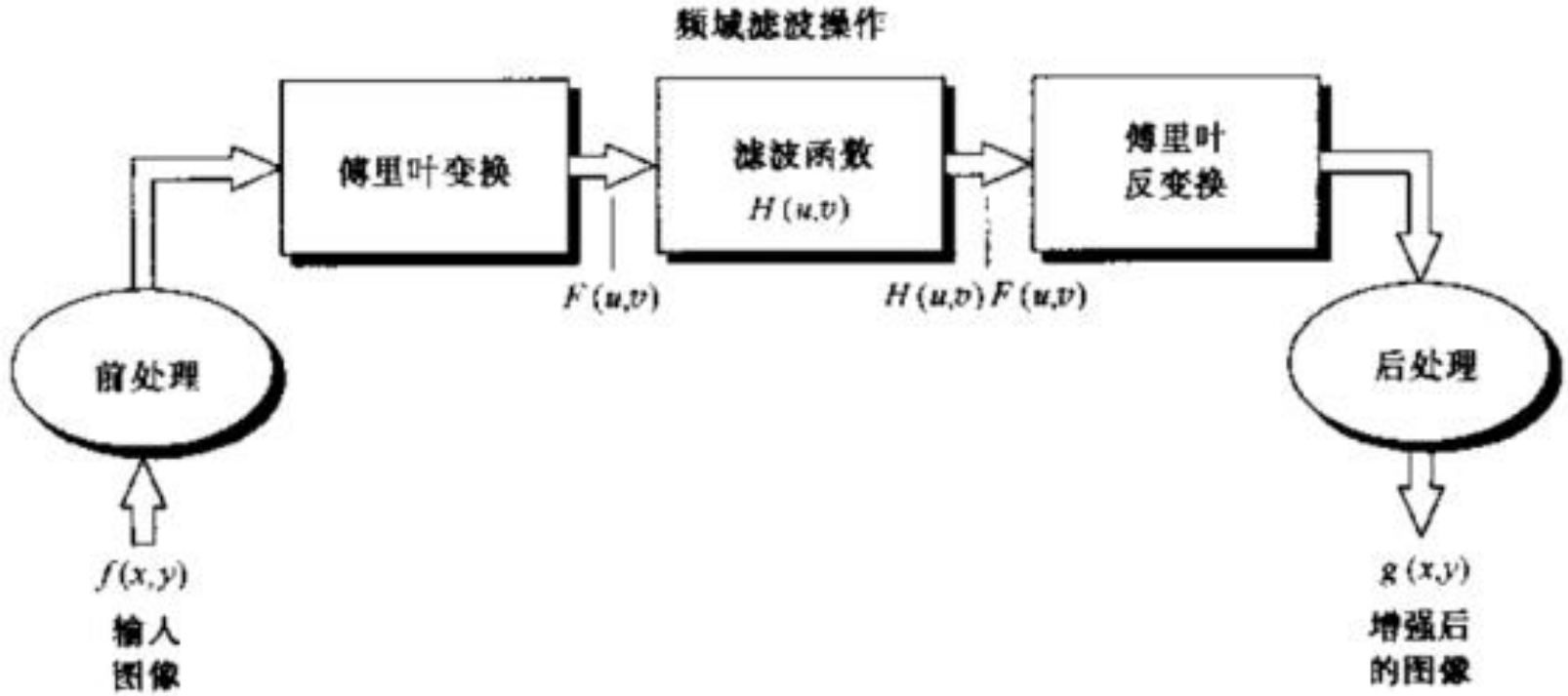


Project 04-04

高通滤波器

Project04-04 高通滤波器

频率域滤波的流程如下图所示。注意这里要进行图像的**延拓**，以确保傅里叶变换的正确性。



原理

二维高斯滤波器由下式给出

$$H(u,v) = \mathrm{e}^{-D^2(u,v)/2\sigma^2}$$

其中 $D(u,v)$ 为距傅里叶变换原点的距离，σ 为高斯曲线扩展的程度。

利用下式可以得到高通滤波的图像

$$f_{hp}(x,y) = f(x,y) - f_{lp}(x,y)$$

其中 f_{lp} 即为 PROJECT 04-03 通过低通滤波后的图像。

代码

```
close all;clc;clear all;
I = imread('Fig0418(a).tif');
I = double(I);
a = 20;
[h,L] = gauss(I,a,1);
I1 = I - h;
[I2,L] = gauss(I,a,0);
figure,
subplot(221),imshow(uint8(I));
title('Fig.4.18(a)原图')
subplot(222),imshow(L);
title('高斯高通滤波器')
subplot(223),imshow(uint8(I1));
title({'Fig.4.18(a)钝化模板';'(原图减低通)'})
subplot(224),imshow(uint8(I2));
title({'Fig.4.26(a)钝化模板';'(高通)'})
```

```
function [g, L] = gauss(img,a,lowpass_flag)
        [M,N] = size(img);
  % 延拓
        P = 2 * M;
  Q = 2 * N;
  % 添加0
  fp = zeros(P,Q);
        fp(1:M,1:N) = img(1:M,1:N);
  % 频谱中心化处理
  [Y,X] = meshgrid(1:Q,1:P);
  ones = (-1).^(X+Y);
        fp = ones.*fp;
  % 傅里叶变换
  FLY = fft2(fp);
  % 高斯滤波
  H = zeros(P,Q);
  L = zeros(P,Q);
  for i=1:P
     for j=1:Q
        H(i,j) = exp((-(i-M)^2-(j-N)^2)/(2*a^2));
        L(i,j) = 1 - H(i,j);
     end
  end
```

```
if lowpass_flag == 1
                G = H.*FLY;
        else
                G = L.*FLY;
  end
  % 傅里叶逆变换，恢复原图像
  g = real(ifft2(G));
  % 从左上角提取 M*N 大小区域
  g = ones.*g;
  g = g(1:M,1:N);
end
```

实验结果与分析

用原图像减去高斯低通滤波后图像的方法得到的高通图像与直接进行高斯高通滤波的方法产生的图像没有太大区别，事实上

$$F(u,v) - F(u,v)H(u,v) = F(u,v)(1 - H(u,v)) = F(u,v)G(u,v)$$

通过加大 σ 的值，可以使高斯低通滤波器作用的范围更大，进而剩余的高通分量更少，也即图像只剩部分细节可见，图像更加模糊。

实验结果与分析

Fig.4.18(a)原图

高斯高通滤波器

Fig.4.18(a)钝化模板
(原图减低通)

Fig.4.26(a)钝化模板
(高通)

图 2: 高斯高通滤波 $\sigma = 20$

Fig.4.18(a)原图

高斯高通滤波器

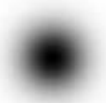


Fig.4.18(a)钝化模板
(原图减低通)

Fig.4.26(a)钝化模板
(高通)

图 3: 高斯高通滤波 $\sigma = 100$

Project 04-05

频率域的相关

Project04-05 频率域的相关

利用空间域的相关与频率域的乘积构成傅里叶变换对这一性质

$$f(x,y) \circ h(x,y) \iff F^*(u,v)H(u,v)$$

程序实现步骤:

1. 将两幅图像的大小都拓展 298×298(256+42)
2. 两幅图像的每个像素都乘以 $(-1)^{(x+y)}$，使其在频域位于中心位置
3. 做傅里叶变换，转换到频域
4. 在频域两幅图像，一个与另一个的共轭相乘计算相关函数
5. 作傅里叶逆变换转换回空间域
6. 乘以 $(-1)^{(x+y)}$，得到最终结果

代码

```
close all;clc;clear all;
I1 = imread('Fig0441(a).jpg');
I2 = imread('Fig0441(b).jpg');
[m1,n1] = size(I1);
[m2,n2] = size(I2);
P = 298;
Q = 298;
img1 = zeros(P,Q);
img2 = zeros(P,Q);
img1(1:m1,1:n1) = I1(1:m1,1:n1);
img2(1:m2,1:n2) = I2(1:m2,1:n2);
% 生成网格采样点，用(-1)^(x+y)乘以原图像
[x,y]=meshgrid(1:P,1:Q);
ones=(-1).^(x+y);
```

```
% 傅里叶变换
f1 = fft2(ones.*img1);
f2 = fft2(ones.*img2);
% 求共轭
rel = f2 .* conj(f1);
% 傅里叶逆变换
newI = ifft2(rel);
newI = ones.*newI;
% 画图
figure,
subplot(131),imshow(I1);
title('Fig.4.41(a)原图')
subplot(132),imshow(I2);
title('Fig.4.41(b)原图')
subplot(133),imshow(mat2gray(newI));
title('Fig.4.41图像相关')
% 二维相关函数中最大值位置的(x,y)坐标
max_value = max(max(newI));
[row,col] = find(newI == max_value);
disp(['max value is : ', num2str(max_value)]);
disp(['row: ', num2str(row), ' col: ', num2str(col)])
```

实验结果与分析

相关性结果如下图所示。

图 4: 频率域的相关

通过 Matlab 计算可以找出最大值出现在 $(191, 193)$ 的位置，符合预期。

感谢聆听！