# 第四章 图像频域增强

学号 17042127　　姓名 陶逸群

## 一、编程部分

**编程语言：**

- Matlab（推荐）
- Python（可能需要使用 OpenCV）

**实现以下操作：**

1. 图像傅里叶变换：

(1) 原始图像：Lena 图或自选其他图像；

(2) 对图像进行傅里叶变换，并显示频谱和相位谱；

(3) 进行傅里叶逆变换，计算恢复后图像与原始图像之间的 L2 距离（即 MSE）。

代码：

```
clc;
close all;
clear all;

%读取图像
img = imread('cameraman.tif');
figure('name','第一题');
subplot(221);
imshow(img);
title('原图');
% 原图像傅里叶变换
f_img = fft2(img);                %对图像进行傅里叶变换
fs_img = fftshift(f_img);         %将零频率的分量移到频谱的中心
scope = log(abs(fs_img)+1);           %%图像的幅度谱
subplot(223);
imshow(scope,[]);
title('频谱');
phase = log(abs(angle(fs_img)*180/pi));  %%图像的相位谱
subplot(224);
imshow(phase,[]);
title('相位谱');
% 原图像傅里叶逆变换
r_img = ifft2(f_img);
```

```
img = double(img);
cha = img - r_img;
MSE = mse(cha);
subplot(222);
imshow(r_img,[]);
title('逆变换后的图像');
title({'逆变换后的图像';['MSE 为',num2str(MSE)]});
```

实验效果图：


结果分析：

傅里叶逆变换的图像与原图像看上去几乎一模一样，两者的均方误差(MSE)为$4.3116 \times 10^{-28}$基本为 0，图像经过傅里叶变换再逆变换回去几乎不会产生损失。

2. 傅里叶变换定理（可选）：

(1) 分别对图像做平移、旋转、尺度变换，计算变换后图像的傅里叶变换

(2) 显示其频谱和相位谱，并与原始图像的结果进行比较。

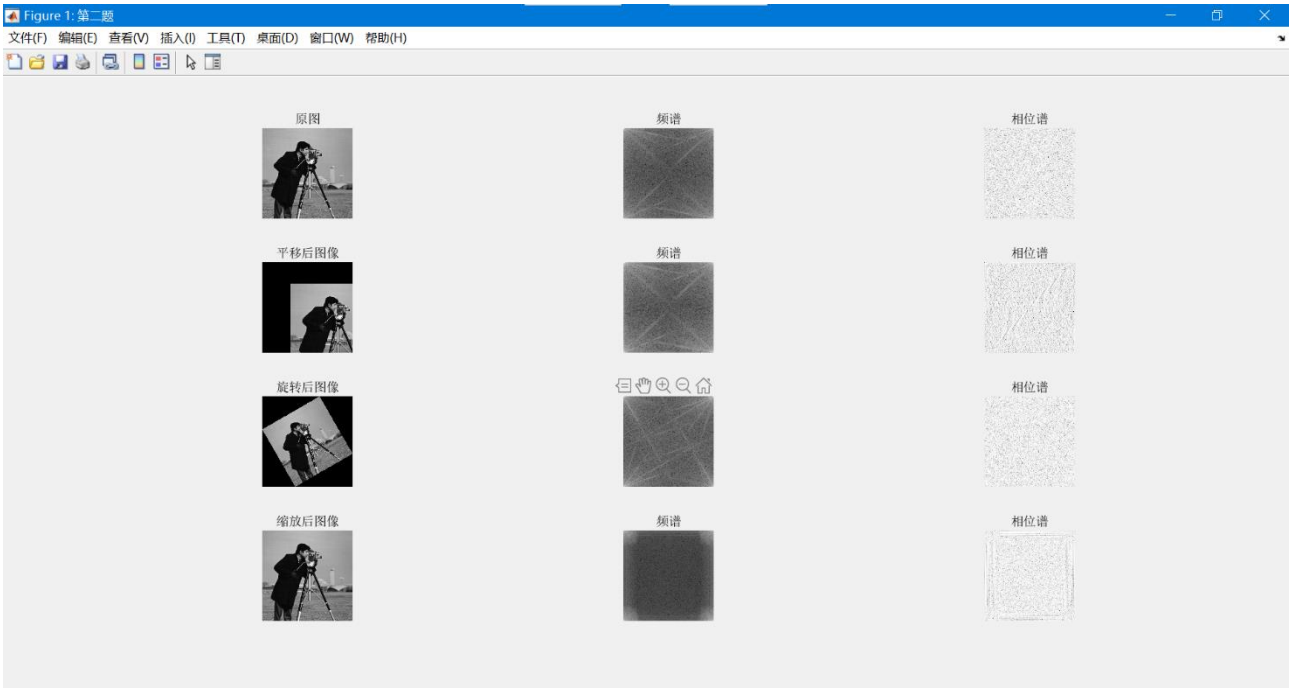
代码：

```
clc;
close all;
clear all;

%读取图像
img = imread('cameraman.tif');
figure('name','第二题');
subplot(431);
imshow(img);
```

```
title('原图');

% 原图像傅里叶变换
f_img = fft2(img);              %对图像进行傅里叶变换
scope = log(abs(f_img)+1);          %%图像的幅度谱
subplot(432);
imshow(scope,[]);
title('频谱');
phase = log(abs(angle(f_img)*180/pi));  %%图像的相位谱
subplot(433);
imshow(phase,[]);
title('相位谱');

%图像平移的傅里叶变换
se = translate(strel(1),[60,80]);
p_img = imdilate(img,se);               %对图像进行平移
subplot(434);
imshow(p_img);
title('平移后图像');
f_img = fft2(p_img);            %对图像进行傅里叶变换
scope = log(abs(f_img)+1);          %%图像的幅度谱
subplot(435);
imshow(scope,[]);
title('频谱');
phase = log(abs(angle(f_img)*180/pi));  %%图像的相位谱
subplot(436);
imshow(phase,[]);
title('相位谱');

%图像旋转并进行傅里叶变换
p_img = imrotate(img,30,'nearest','loose');                %对图像进行旋转
subplot(437);
imshow(p_img);
title('旋转后图像');
f_img = fft2(p_img);            %对图像进行傅里叶变换
scope = log(abs(f_img)+1);          %%图像的幅度谱
subplot(438);
imshow(scope,[]);
title('频谱');
phase = log(abs(angle(f_img)*180/pi));  %%图像的相位谱
subplot(439);
imshow(phase,[]);
title('相位谱');
%图像缩放并进行傅里叶变换
```

```
p_img =imresize(img,5);                %对图像进行缩放
subplot(4,3,10);
imshow(p_img);
title('缩放后图像');
f_img = fft2(p_img);                %对图像进行傅里叶变换
scope = log(abs(f_img)+1);          %%图像的幅度谱
subplot(4,3,11);
imshow(scope,[]);
title('频谱');
phase = log(abs(angle(f_img)*180/pi));  %%图像的相位谱
subplot(4,3,12);
imshow(phase,[]);
title('相位谱');
```

实验效果图：




结果分析：

从实验效果图可知：

a) 对图片进行平移后进行傅里叶变换， 频谱不发生改变，相位谱发生了改变。

b) 对图片进行旋转后进行傅里叶变换， 频谱发生改变，相位谱也发生了改变。

c) 对图片进行缩放后进行傅里叶变换， 频谱发生改变，相位谱也发生了改变。

3. 低通滤波：

(1) 在傅里叶变换的基础上进行低通滤波；

(2) 采用理想低通、巴特沃斯低通；

(3) 对滤波后结果进行逆变换，得到滤波后的图像；

(4) 计算滤波后图像与原始图像之间的信噪比，并分析实验结果；

代码：

理想低通滤波函数：

```
%理想低、高通滤波函数，flag 为 1 则是高通。
function res = dream_filter(img,d,flag)
fs_img = fftshift(fft2(img));%傅里叶变换
[i,j] = size(fs_img);
[u,v] = meshgrid(-j/2:(j/2-1),-i/2:(i/2-1));%产生离散数据
distance = sqrt(u.^2+v.^2);
res = zeros(i,j);
h = double(distance<=d);
if(flag == 1)
    h = 1-h;
end
res = real(ifft2(h.*fs_img));
```

巴特沃斯低通滤波函数：

```
%巴特沃斯低、高通滤波函数，flag 为 1，为高通滤波。
function res = butterworth_filter(img,d,n,flag)
fs_img = fftshift(fft2(img));%傅里叶变换
[i,j] = size(fs_img);
[u,v] = meshgrid(-j/2:(j/2-1),-i/2:(i/2-1));%产生离散数据
distance = sqrt(u.^2+v.^2);
res = zeros(i,j);
h = 1./(1+(distance/d).^(2*n));
if(flag == 1)
    h = 1-h;
end
res = real(ifft2(h.*fs_img));
```

主 m 函数：

```
clc
clear all
close all

d = 60; %阈值
img = imread('cameraman.tif');
img = im2double(img);
flag = 0;
figure('name','第三题');
subplot(131);
```

```
imshow(img);
title('原图');
dr_img = dream_filter(img,d,0);
dr_snr = roundn(snr(img,dr_img-img),-2);
bf_img = butterworth_filter(img,d,1,0);
bf_snr = roundn(snr(img,bf_img-img),-2);
subplot(132);
imshow(dr_img);
title({'理想低通滤波';['信噪比为',num2str(dr_snr),'dB']});
subplot(133);
imshow(bf_img);
title({'n 为 1 的巴特沃斯低通滤波';['信噪比为',num2str(bf_snr),'dB']});
```

实验效果图：


结果分析：

设置阈值为 60 对图像进行低通滤波，像素值低于 60 的点将保留，而高于 60 的点被过滤。设置巴斯沃特低通滤波器的 n 为 1。与理想低通滤波器相比，巴特沃斯低通滤波获得的结果更好，信噪比也更高。

4. 高通滤波：

(1) 在傅里叶变换的基础上进行高通滤波；

(2) 分别采用理想高通、巴特沃斯高通；

(3) 对滤波后结果进行逆变换，得到滤波后的图像；

(4) 计算滤波后图像与原始图像之间的信噪比，并分析实验结果；

代码：

理想高通滤波函数：

```
%理想低、高通滤波函数，flag 为 1 则是高通。
function res = dream_filter(img,d,flag)
fs_img = fftshift(fft2(img));%傅里叶变换
[i,j] = size(fs_img);
[u,v] = meshgrid(-j/2:(j/2-1),-i/2:(i/2-1));%产生离散数据
distance = sqrt(u.^2+v.^2);
res = zeros(i,j);
h = double(distance<=d);
if(flag == 1)
    h = 1-h;
end
res = real(ifft2(h.*fs_img));
```

巴特沃斯高通滤波函数：

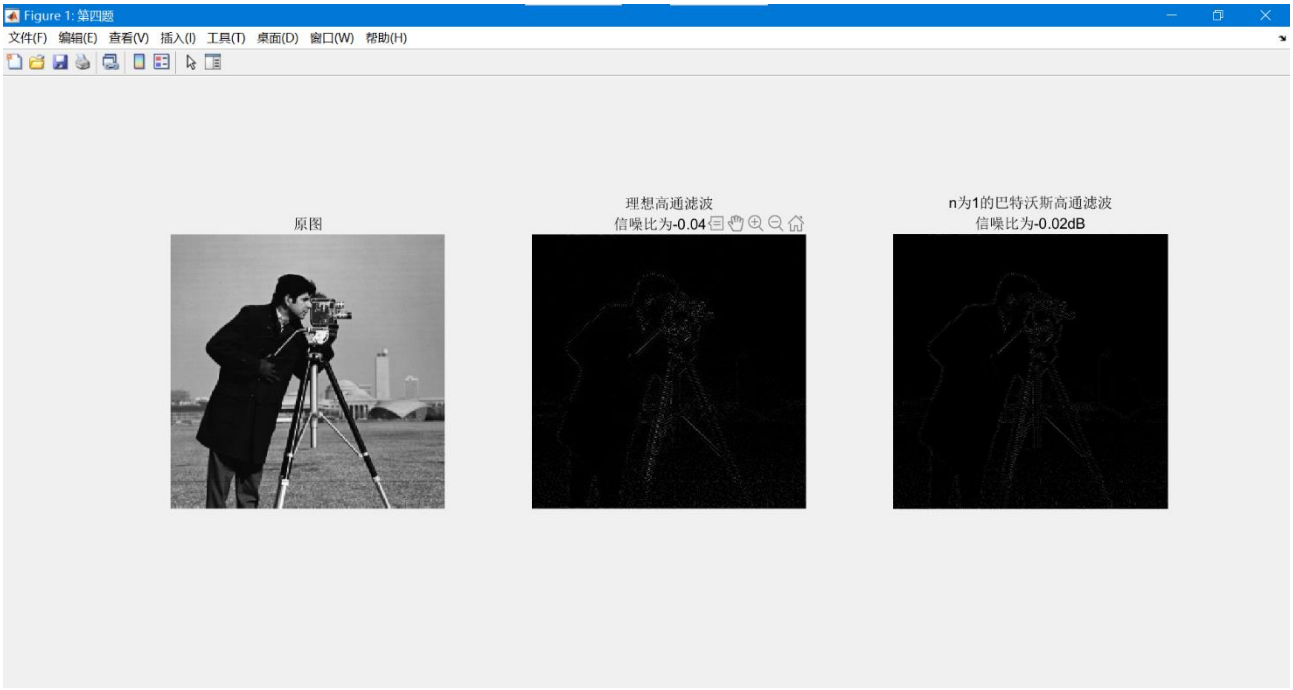
```
%巴特沃斯低、高通滤波函数，flag 为 1，为高通滤波。
function res = butterworth_filter(img,d,n,flag)
fs_img = fftshift(fft2(img));%傅里叶变换
[i,j] = size(fs_img);
[u,v] = meshgrid(-j/2:(j/2-1),-i/2:(i/2-1));%产生离散数据
distance = sqrt(u.^2+v.^2);
res = zeros(i,j);
h = 1./(1+(distance/d).^(2*n));
if(flag == 1)
    h = 1-h;
end
res = real(ifft2(h.*fs_img));
```

主 m 函数：

```
clc
clear all
close all

d = 60; %阈值
img = imread('cameraman.tif');
img = im2double(img);
flag = 0;
figure('name','第四题');
subplot(131);
imshow(img);
title('原图');
dr_img = dream_filter(img,d,1);
dr_snr = roundn(snr(img,dr_img-img),-2);
```

```
bf_img = butterworth_filter(img,d,1,1);
bf_snr = roundn(snr(img,bf_img-img),-2);
subplot(132);
imshow(dr_img);
title({'理想高通滤波';['信噪比为',num2str(dr_snr),'dB']});
subplot(133);
imshow(bf_img);
title({'n 为 1 的巴特沃斯高通滤波';['信噪比为',num2str(bf_snr),'dB']});
```

实验效果图：




结果分析：

设置阈值为 60 对图像进行高通滤波，像素值高于 60 的点将保留，而低于 60 的点被过滤。设置巴斯沃特低通滤波器的 n 为 1。与理想低通滤波器相比，巴特沃斯低通滤波获得的结果更好，信噪比也更高。

5. 高通增强滤波：

(1) 基于上述高通滤波，对原始图像（或模糊后的图像）进行增强；

(2) 显示图像，并计算信噪比。

代码:

理想高通滤波函数：

```
%理想低、高通滤波函数，flag 为 1 则是高通。
function res = dream_filter(img,d,flag)
fs_img = fftshift(fft2(img));%傅里叶变换
[i,j] = size(fs_img);
[u,v] = meshgrid(-j/2:(j/2-1),-i/2:(i/2-1));%产生离散数据
distance = sqrt(u.^2+v.^2);
```

```
res = zeros(i,j);
h = double(distance<=d);
if(flag == 1)
    h = 1-h;
end
res = real(ifft2(h.*fs_img));
```

巴特沃斯高通滤波函数：

```
%巴特沃斯低、高通滤波函数，flag 为 1，为高通滤波。
function res = butterworth_filter(img,d,n,flag)
fs_img = fftshift(fft2(img));%傅里叶变换
[i,j] = size(fs_img);
[u,v] = meshgrid(-j/2:(j/2-1),-i/2:(i/2-1));%产生离散数据
distance = sqrt(u.^2+v.^2);
res = zeros(i,j);
h = 1./(1+(distance/d).^(2*n));
if(flag == 1)
    h = 1-h;
end
res = real(ifft2(h.*fs_img));
```

主 m 函数：

```
clc
clear all
close all

d = 60; %阈值
img = imread('cameraman.tif');
img = im2double(img);
flag = 0;
figure('name','第五题');
subplot(131);
imshow(img);
title('原图');
dri_img = dream_filter(img,d,1)+img;
dri_snr = roundn(snr(img,dri_img-img),-2);
bfi_img = butterworth_filter(img,d,1,1)+img;
bfi_snr = roundn(snr(img,bfi_img-img),-2);
subplot(132);
imshow(dri_img);
title({'理想高通滤波增强';['信噪比为',num2str(dri_snr),'dB']});
subplot(133);
imshow(bfi_img);
```

```
title({'n 为 1 的巴特沃斯高通滤波增强';['信噪比为',num2str(bfi_snr),'dB']});
```

实验效果图：


结果分析：

从结果图来看，采用巴斯沃特高通滤波图像增强的效果要好于采用理想高通滤波的图像增强，经过巴斯沃特高通滤波图像增强的图片的信噪比也高于经过理想高通滤波图像增强的图片的信噪比。

## 二、理论部分

1. 已知 $N\times N$ 的 $f(x,y)$ 的傅里叶变换为 $F(u,v)$，写出 $f(x,y)\exp[j\pi(x+y)]$ 的傅里叶变换

解：由二维离散傅里叶变换的平移特性

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

则

$$e^{j\pi(x+y)} = e^{j2\pi(u_0Nx/2N+v_0Ny/2N)}$$

所以

$$f(x,y)e^{j\pi(x+y)} = f(x,y)e^{j2\pi(u_0Nx/2N+v_0Ny/2N)} = F(u-\frac{u_0}{2N}+v-\frac{v_0}{2N})$$

2. 用图(a)所示的模板与图像卷积可获得对图像低通滤波的效果，那用图(b)所示的模板与图像卷积可获得什么效果呢？试给出图(b)所示的模板在频域的等价滤波器 $H(u,v)$，并解释其功能。（提示：傅里叶变换的卷积定理）

| | | |
|---|---|---|
| | 1 | |

| | | |
|---|---|---|
| | 1 | |

| | | |
|---|---|---|
| 1 | 1 | 1 |
| | 1 | |

(a)

| | | |
|---|---|---|
| 1 | | 1 |
| | 1 | |

(b)

解： 用图(b)所示模板与图像卷积与使用图(a)所示模版与图像卷积获得的效果类似，同样可获得对图像低通滤波的效果，只不过使用图(a)所示模版与图像卷积获得的图片，保留细节更多，信噪比更高。

根据二维离散傅里叶变换(DFT)公式:

$$F(u,v)=\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}f(x,y)e^{-j2\pi(ux/M+vy/N)}$$

可以求得图(b)模板在频域的等价滤波器 H(u,v)

$$H(0,0)=\sum_{x=0}^{8}\sum_{x=0}^{8}f(x,y)e^{0}=\frac{1}{4}$$

$$H(0,1)=\sum_{x=0}^{8}\sum_{x=0}^{8}f(x,y)e^{-\frac{j2\pi y}{9}}=\frac{1}{4}+\frac{1}{2}e^{-\frac{j2\pi}{9}}+\frac{1}{4}e^{-\frac{j4\pi}{9}}$$

$$H(0,2)=\sum_{x=0}^{8}\sum_{x=0}^{8}f(x,y)e^{-\frac{j4\pi y}{9}}=\frac{1}{4}+\frac{1}{2}e^{-\frac{j4\pi}{9}}+\frac{1}{4}e^{-\frac{j8\pi}{9}}$$

$$H(1,0)=\sum_{x=0}^{8}\sum_{x=0}^{8}f(x,y)e^{-\frac{j2\pi x}{9}}=\frac{1}{4}+\frac{1}{2}e^{-\frac{j2\pi}{9}}+\frac{1}{4}e^{-\frac{j4\pi}{9}}$$

$$H(1,1)=\sum_{x=0}^{8}\sum_{x=0}^{8}f(x,y)e^{-\frac{j2\pi(x+y)}{9}}=\frac{1}{2}e^{-\frac{j2\pi}{9}}+\frac{1}{4}e^{-\frac{j6\pi}{9}}$$

$$H(1,2)=\sum_{x=0}^{8}\sum_{x=0}^{8}f(x,y)e^{-\frac{j2\pi(x+2y)}{9}}=\frac{1}{4}e^{-\frac{j2\pi}{9}}+\frac{1}{4}e^{-\frac{j4\pi}{9}}+\frac{1}{4}e^{-\frac{j8\pi}{9}}+\frac{1}{4}e^{-\frac{j10\pi}{9}}$$

$$H(2,0)=\sum_{x=0}^{8}\sum_{x=0}^{8}f(x,y)e^{-\frac{j4\pi y}{9}}=\frac{1}{4}+\frac{1}{2}e^{-\frac{j4\pi}{9}}+\frac{1}{4}e^{-\frac{j8\pi}{9}}$$

$$H(2,1)=\sum_{x=0}^{8}\sum_{x=0}^{8}f(x,y)e^{-\frac{j2\pi(2x+y)}{9}}=\frac{1}{4}e^{-\frac{j2\pi}{9}}+\frac{1}{4}e^{-\frac{j4\pi}{9}}+\frac{1}{4}e^{-\frac{j8\pi}{9}}+\frac{1}{4}e^{-\frac{j10\pi}{9}}$$

$$H(2,2)=\sum_{x=0}^{8}\sum_{x=0}^{8}f(x,y)e^{-\frac{j2\pi y}{9}}=\frac{1}{2}e^{-\frac{j8\pi}{9}}+\frac{1}{4}e^{-\frac{j12\pi}{9}}$$

$$H(u,v)=\begin{bmatrix}1+0i & -0.125-0.2165i & -0.125+0.2165i\\ -0.125-0.2165i & 0.25-0.4330i & -0.5+0i\\ -0.125+0.2165i & -0.5+0i & 0.25+0.433i\end{bmatrix}$$

图(b)模版的功能是，平滑图像，去除噪音。