课程实践 2： 图像变换

实验目的：给定图像（lena.png 512x512x256）,分别编程实现 DFT 和 DCT 变换。显示其频谱图像，相位图, 实部，虚部图像, 以及特定频谱的还原。通过实验了解图像空间和频率之间的关系。

实验要求：

1. 显示 DFT,DCT 频谱图 S(u,v)=DFT(f(x,y))，S(u,v)=DCT(f(x,y))，显示实部，虚部，相位图。
2. 分别用不同频率的信号从原图像。

   低频重构： 窗口大小 8x8, 16x16,32x32,64x64

   中频重构：

   高频频重构：挖去窗口大小 256x256, 128x128,64x64,32x32

实验环境：MATLAB R2018a

原理和方法：

①二维离散 Fourier 变换(DFT)：

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

谱： $|F(u,v)|=[R^2(u,v)+I^2(u,v)]^{-1/2},R=\mathrm{Re}(F),I=\mathrm{Im}(F)$

相位角： $\phi(u,v)=\arctan[I(u,v)/R(u,v)],R=\mathrm{Re}(F),I=\mathrm{Im}(F)$

但事实上，若直接采用上述 DFT 公式进行计算效率会非常低，一种高效的办法是转化为矩阵进行运算：

矩阵表达式： $F=PfQ$， $F(u,v)=\sum_{x=0}^{N-1}\sum_{y=0}^{N-1}P(x,u)f(x,y)Q(y,v)$

$P(x,u)=\exp[-j2\pi ux/N]$， $Q(y,v)=\exp[-j2\pi vy/N]$， $u,v=0,1,2,...,N-1$

查资料发现 DFT 公式的系数有三种版本，$\frac{1}{N^2}$、$\frac{1}{N}$ 和 1。不影响逆变换结果。

MATLAB 中已有函数 fft2()可完成 DFT，查阅 doc 知用的是系数 1。这里自编函数 DFT() 也取系数为 1，并将结果与之对比。

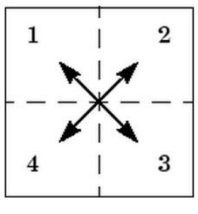
逆变换： $f=P^{-1}FQ^{-1}$， $f(x,y)=\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}F(u,v)e^{j2\pi(ux/M+vy/N)}$

平移变换：对 Cameraman 图像进行二维傅里叶变换，模的最大值必为低频值，除左上角外，大值多位于四个角落。一般研究频域图像是把低频部分，也就是变换后的边缘部分移到图像中心点。

MATLAB 提供 fftshift()函数完成平移，这里自编函数 DFTshift()与之对比。

平移的思路有两个：

（1）通过 Fourier 变换平移定理先把原始图像做变换再做 FFT



（2）先做 FFT 后再依据频域图像的对称性做对称变换

查阅 MATLAB 文档发现它是用另外一种方法（如右图所示），对图像做子矩阵交换。

低维重构：只保留平移后频谱图的中心部分（低频），然后反 DFT 还原图像。

高维重构：只保留平移后频谱图的边缘部分（高频），然后反 DFT 还原图像。

②二维离散余弦变换(DCT)：

代数表达式：$C(u,v)=a(u)a(v)\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}f(x,y)\cos\left[\frac{(2x+1)u\pi}{2M}\right]\cos\left[\frac{(2y+1)v\pi}{2N}\right]$

同样可以写成矩阵形式：$C=PfQ$，$C(u,v)=\frac{1}{N}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}P(x,u)f(x,y)Q(y,v)$

$$P(u,v)=a(u)\cos\left[\frac{(2x+1)u\pi}{2M}\right],\quad Q(u,v)=a(v)\cos\left[\frac{(2y+1)v\pi}{2N}\right],\quad u,v=0,1,2,\ldots,N-1$$

DCT 的平移变换和高低维重构与 DFT 完全相同。

MATLAB 没有 dctshift()函数，但可以直接使用 fftshift()。

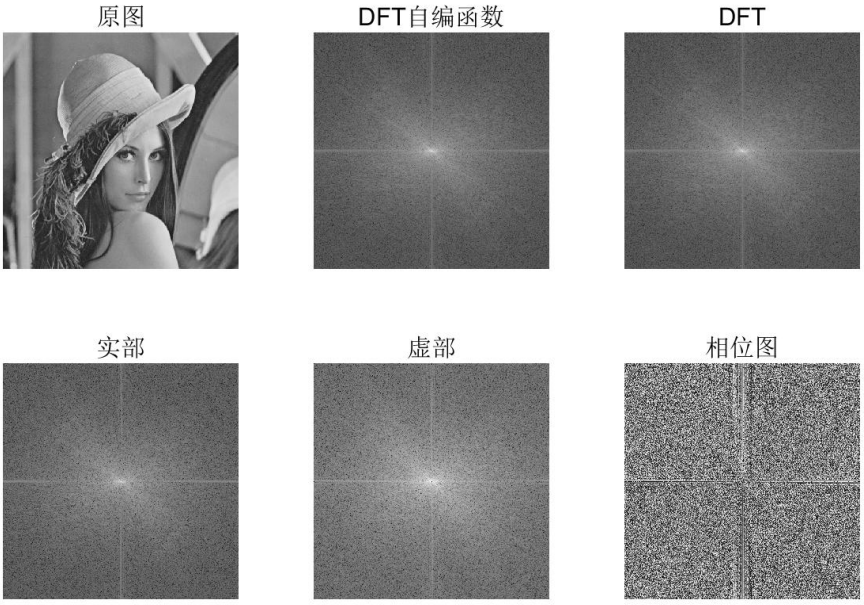
实验代码和结果：

①二维离散 Fourier 变换(DFT)：

```
clear all
A=imread('lena.png');
[F,P,Q] = DFT(A);
F_A1 = DFTshift(F);
F_A2 = fftshift(fft2(A));
FP_A = atan2(imag(F_A1),real(F_A1)); % 或FP_A = angle(F_A1)
figure(1)
subplot(2,3,1),imshow(A),title('原图')
subplot(2,3,2),imshow(log(1+abs(F_A1)),[]),title('DFT自编函数')
subplot(2,3,3),imshow(log(1+abs(F_A2)),[]),title('DFT')
subplot(2,3,4),imshow(log(1+abs(real(F_A1))),[]),title('实部')
subplot(2,3,5),imshow(log(1+abs(imag(F_A1))),[]),title('虚部')
subplot(2,3,6),imshow(FP_A,[]),title('相位图')
```

```
function [F,P,Q] = DFT(f)
f=double(f);
[M N] = size(f);
if M~=N
    error('图像必须为方阵！');
end
x = 0:M-1; v = x;
y = (0:N-1)'; u = y;
F = zeros(size(f));
P = exp(-i*2*pi*(u*x)/M);
Q = exp(-i*2*pi*(v'*y')/N);
F = P*f*Q;
end
```

```
function A_shift = DFTshift(A)
A=double(A);
M = size(A,1);
N = size(A,2);
if log2(M)~=fix(log2(M)) || log2(N)~=fix(log2(N))
    error('图像尺寸必须为2的幂！')
end
A_shift = zeros(size(A));
A_shift(1:M/2,1:N/2) = A(M/2+1:M,N/2+1:N);
A_shift(1:M/2,N/2+1:N) = A(M/2+1:M,1:N/2);
A_shift(M/2+1:M,1:N/2) = A(1:M/2,N/2+1:N);
A_shift(M/2+1:M,N/2+1:N) = A(1:M/2,1:N/2);
end
```



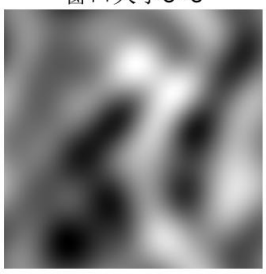
从图中可以看出，自编函数实现 DFT 的效果与 MATLAB 自带函数的效果一致，也证明了 fft()使用的公式前的系数确实为 1。且由于实部和虚部均有图像，说明 DFT 后的图像值为复数。

低频重构

```
w1 = [8 16 32 64];
[M N] = size(F);
T1 = zeros(size(F));
figure(2)
for i=1:length(w1)
    t = M/2-w1(i)/2+1:M/2+w1(i)/2;
    T1(t,t) = F_A1(t,t);
    T_A = inv(P)*T1*inv(Q);
    subplot(2,2,i),imshow(abs(T_A),[]),title(['窗口大小',num2str(w1(i)),'×',num2str(w1(i))])
end
suptitle('低频重构')
```

低频重构

窗口大小8×8



窗口大小16×16

窗口大小32×32

窗口大小64×64

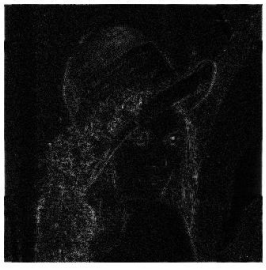
可以看出，窗口保留大小越大，图像低频重构的还原效果越好，说明图像的信息绝大部分位于低频段。

高频重构

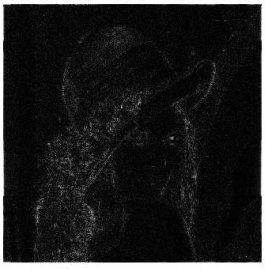
```
w2 = [256 128 64 32];
[M N] = size(F);
T2 = F_A1;
figure(3)
for i=1:length(w2)
    t = M/2-w2(i)/2+1:M/2+w2(i)/2;
    T2(t,t) = 0;
    T_A = inv(P)*T2*inv(Q);
    subplot(2,2,i),imshow(abs(T_A),[]),title(['挖去窗口大小',num2str(w2(i)),'×',num2str(w2(i))])
end
suptitle('高频重构')
```

高频重构

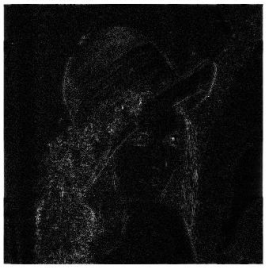
挖去窗口大小256×256



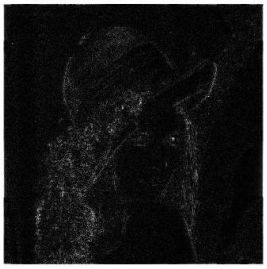
挖去窗口大小128×128



挖去窗口大小64×64



挖去窗口大小32×32



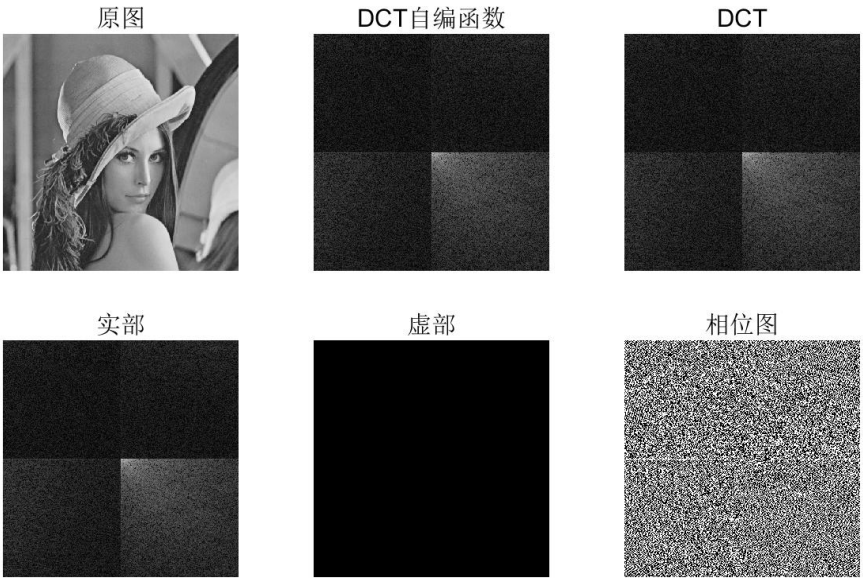
可以看出，挖去窗口大小对高频重构的结果没有太大影响，说明原图像只有很少的信息位于高频段。

②二维离散余弦变换(DCT)：

```
clear all
A=imread('lena.png');
[C,P,Q] = DCT(A);
C_A1 = DCTshift(C);
C_A2 = fftshift(dct2(A));
CP_A = atan2(imag(C_A1),real(C_A1)); % 或FP_A = angle(F_A1)
figure(1)
subplot(2,3,1),imshow(A),title('原图')
subplot(2,3,2),imshow(log(1+abs(C_A1)),[]),title('DCT自编函数')
subplot(2,3,3),imshow(log(1+abs(C_A2)),[]),title('DCT')
subplot(2,3,4),imshow(log(1+abs(real(C_A1))),[]),title('实部')
subplot(2,3,5),imshow(log(1+abs(imag(C_A1))),[]),title('虚部')
subplot(2,3,6),imshow(CP_A),title('相位图')
```

```
function [C,P,Q] = DCT(f)
f=double(f);
[M N] = size(f);
if M~=N
    error('图像必须为方阵！');
end
x = 0:M-1; v = x;
y = (0:N-1)'; u = y;
C = zeros(size(f));
a = sqrt(2/N)*eye(M); a(1,1)=sqrt(1/N);
P = a*cos(u*(2*x+1)*pi/2/M);
Q = a*cos((2*y+1)*v*pi/2/N);
C = P*f*Q;
end
```

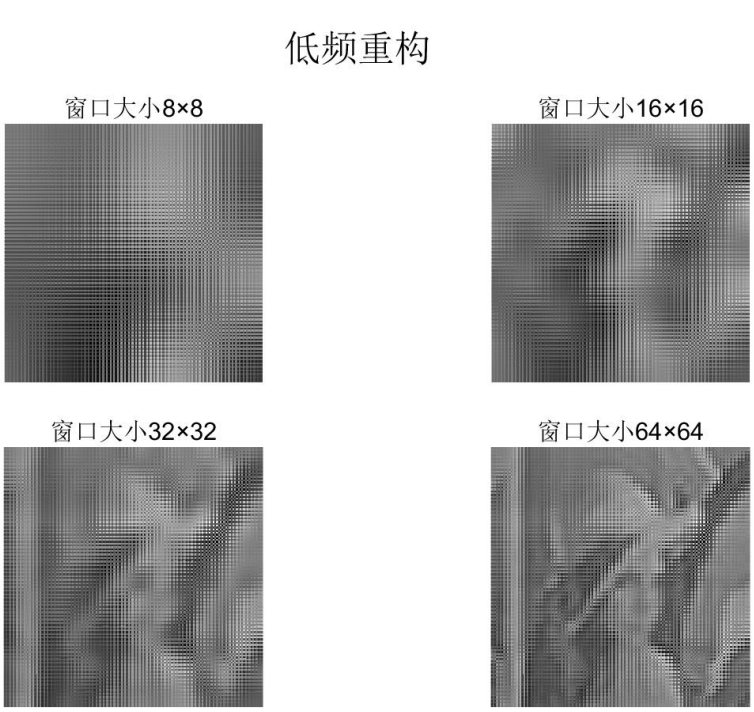
```
function A_shift = DCTshift(A)
A=double(A);
M = size(A,1);
N = size(A,2);
if log2(M)~=fix(log2(M)) || log2(N)~=fix(log2(N))
    error('图像尺寸必须为2的幂！')
end
A_shift = zeros(size(A));
A_shift(1:M/2,1:N/2) = A(M/2+1:M,N/2+1:N);
A_shift(1:M/2,N/2+1:N) = A(M/2+1:M,1:N/2);
A_shift(M/2+1:M,1:N/2) = A(1:M/2,N/2+1:N);
A_shift(M/2+1:M,N/2+1:N) = A(1:M/2,1:N/2);
end
```



由于虚部没有图像，说明 DCT 之后的图像值仍为实数。

低频重构

```
w1 = [8 16 32 64];
[M N] = size(C);
T1 = zeros(size(C));
figure(2)
for i=1:length(w1)
    t = M/2-w1(i)/2+1:M/2+w1(i)/2;
    T1(t,t) = C_A1(t,t);
    T_A = inv(P)*T1*inv(Q);
    subplot(2,2,i),imshow(abs(T_A),[]),title(['窗口大小',num2str(w1(i)),'×',num2str(w1(i))])
end
suptitle('低频重构')
```



高频重构

```
w2 = [256 128 64 32];
[M N] = size(C);
T2 = C_A1;
figure(3)
for i=1:length(w2)
    t = M/2-w2(i)/2+1:M/2+w2(i)/2;
    T2(t,t) = 0;
    T_A = inv(P)*T2*inv(Q);
    subplot(2,2,i),imshow(abs(T_A),[]),title(['挖去窗口大小',num2str(w2(i)),'×',num2str(w2(i))])
end
suptitle('高频重构')
```