HOMEWORK 3

M10907324 吳俊逸

1. Implementation of 2D-DCT and its inverse transform

Required Submission: Code, Image Output in Spatial and Frequency Domain

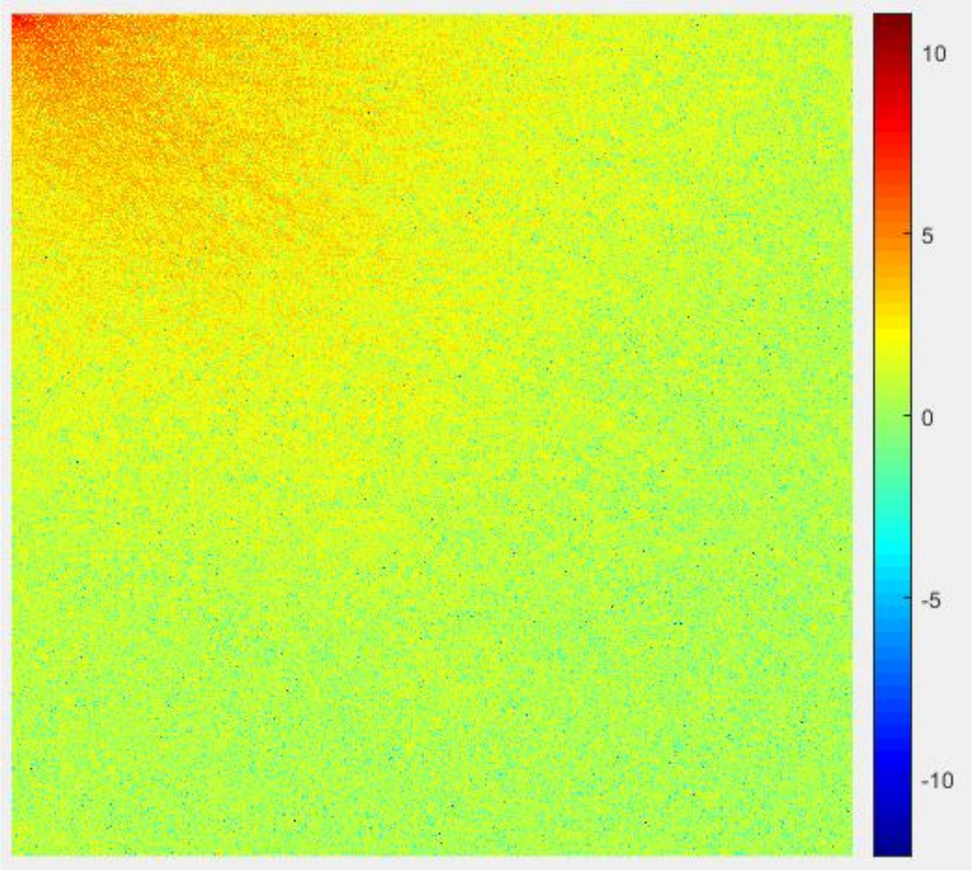
Original image：

Frequency Domain

Code：

```
clc;clear all;close all;

input=imread('lena.bmp');

J=dct2(input);

figure

imshow(log(abs(J)),[])

colormap(gca,jet(64))

colorbar
```

Result：



Spatial Domain

DCT Use formula：

$$F(i,j)=\frac{2}{N}C(i)C(j)\sum_{x=0}^{N-1}\sum_{y=0}^{N-1}f(x,y)cos\left[\frac{(2x+1)i\pi}{2N}\right]cos\left[\frac{(2y+1)j\pi}{2N}\right]$$

IDCT Use formula：

$$f(x,y)=\frac{2}{N}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}C(i)C(j)F(i,j)cos\left[\frac{(2x+1)i\pi}{2N}\right]cos\left[\frac{(2y+1)j\pi}{2N}\right]$$

Code：

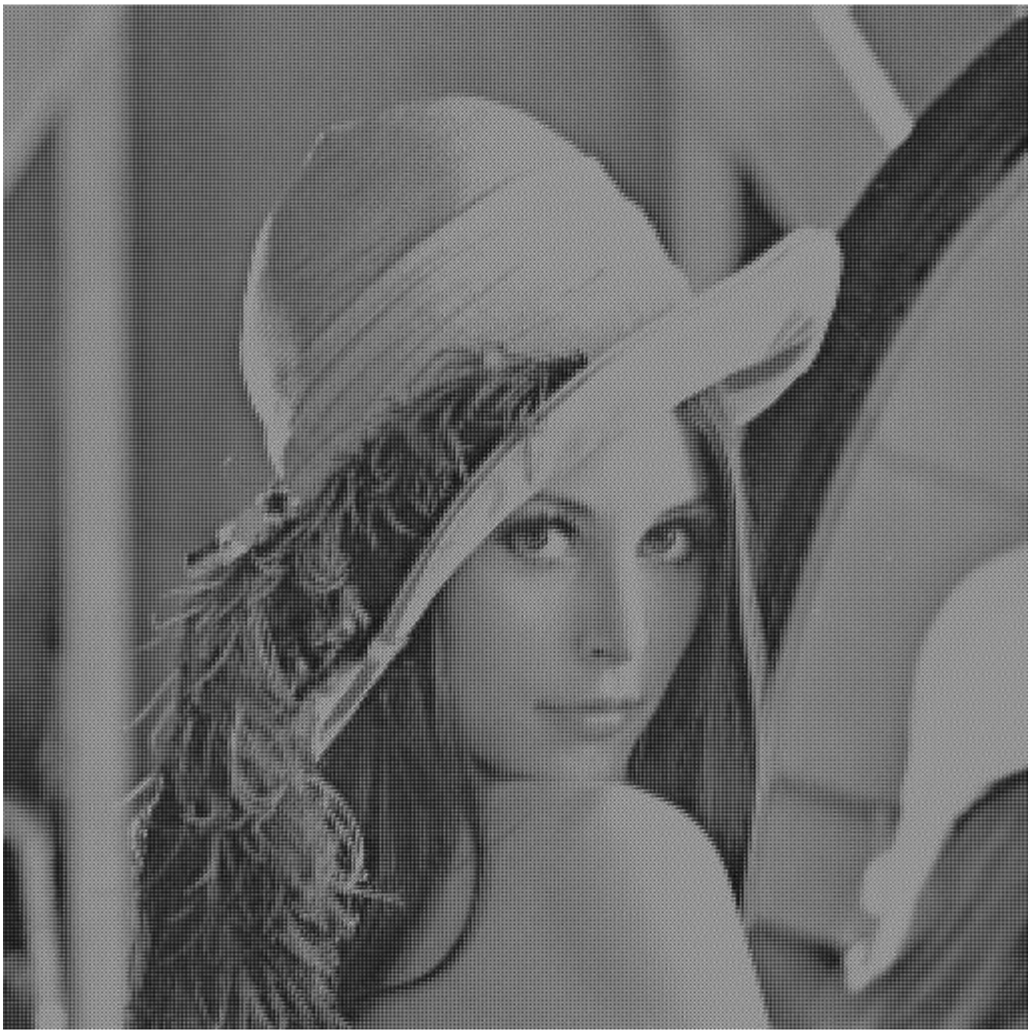
```
clc;clear all;close all;

n=2;

% a=[1 2 3 4;5 6 7 8;9 10 11 12;13 14 15

16];

a=imread('lena.bmp');

[A,B] = size(a);

%DCT

for i=1:2:A

    for j=1:2:B

        %00

        Ans_DCT(i,j)=2/n*(2^-0.5)*(2^-

0.5)*[a(i,j)*cos(0)*cos(0)+a(i,j+1)*cos(0)*
cos(0)+a(i+1,j)*cos(0)*cos(0)+a(i+1,j+1)*co
s(0)*cos(0)];
        %01
        Ans_DCT(i,j+1)=2/n*(2^-
0.5)*1*[a(i,j)*cos(0)*cos(0.25*pi)+a(i,j+1)
*cos(0)*cos(0.75*pi)+a(i+1,j)*cos(0)*cos(0.
25*pi)+a(i+1,j+1)*cos(0)*cos(0.75*pi)];
        %10
        Ans_DCT(i+1,j)=2/n*1*(2^-
0.5)*[a(i,j)*cos(0.25*pi)*cos(0)+a(i,j+1)*c
os(0.25*pi)*cos(0)+a(i+1,j)*cos(0.75*pi)*co
s(0)+a(i+1,j+1)*cos(0.75*pi)*cos(0)];
        %11

Ans_DCT(i+1,j+1)=2/n*1*1*[a(i,j)*cos(0.25*p
i)*cos(0.25*pi)+a(i,j+1)*cos(0.25*pi)*cos(0
.75*pi)+a(i+1,j)*cos(0.75*pi)*cos(0.25*pi)+
a(i+1,j+1)*cos(0.75*pi)*cos(0.75*pi)];
    end
```

```
end
%IDCT
for i=1:2:A
    for j=1:2:B
        %00
        Ans_IDCT(i,j)=2/n*[(2^-0.5)*(2^-
0.5)*Ans_DCT(i,j)*cos(0)*cos(0)+(2^-
0.5)*1*Ans_DCT(i,j+1)*cos(0)*cos(0.25*pi)+1
*(2^-
0.5)*Ans_DCT(i+1,j)*cos(0.25*pi)*cos(0)+1*1
*Ans_DCT(i+1,j+1)*cos(0.25*pi)*cos(0.25*pi)
];
        %01
        Ans_IDCT(i,j+1)=2/n*[(2^-0.5)*(2^-
0.5)*Ans_DCT(i,j)*cos(0)*cos(0)+(2^-
0.5)*1*Ans_DCT(i,j+1)*cos(0)*cos(0.75*pi)+1
*(2^-
0.5)*Ans_DCT(i+1,j)*cos(0.25*pi)*cos(0)+1*1
*Ans_DCT(i+1,j+1)*cos(0.25*pi)*cos(0.75*pi)
];
        %10
        Ans_IDCT(i+1,j)=2/n*[(2^-0.5)*(2^-
0.5)*Ans_DCT(i,j)*cos(0)*cos(0)+(2^-
0.5)*1*Ans_DCT(i,j+1)*cos(0)*cos(0.25*pi)+1
*(2^-
0.5)*Ans_DCT(i+1,j)*cos(0.75*pi)*cos(0)+1*1
*Ans_DCT(i+1,j+1)*cos(0.75*pi)*cos(0.25*pi)
];
        %11
        Ans_IDCT(i+1,j+1)=2/n*[(2^-0.5)*(2^-
0.5)*Ans_DCT(i,j)*cos(0)*cos(0)+(2^-
0.5)*1*Ans_DCT(i,j+1)*cos(0)*cos(0.75*pi)+1
*(2^-
0.5)*Ans_DCT(i+1,j)*cos(0.75*pi)*cos(0)+1*1
*Ans_DCT(i+1,j+1)*cos(0.75*pi)*cos(0.75*pi)
];
    end
end
```

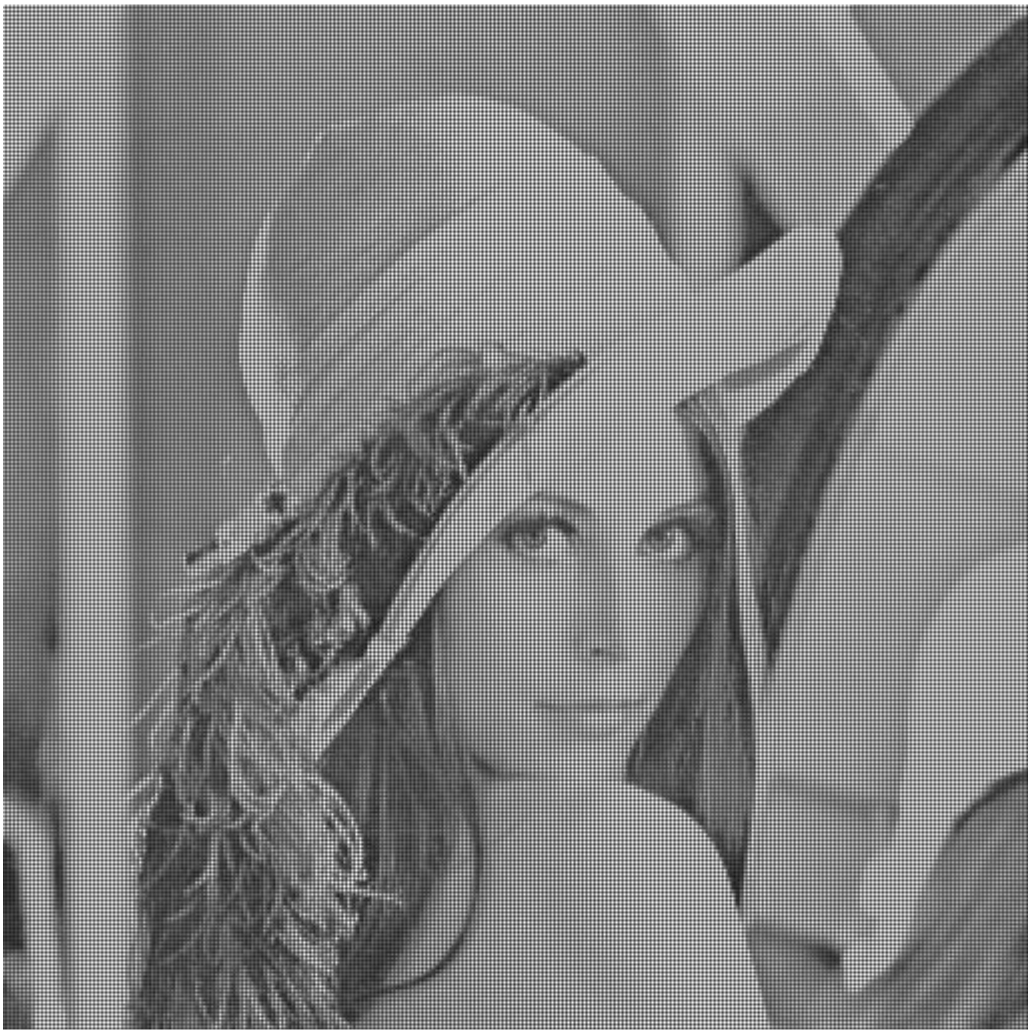
```
% Ans_DCT

imshow(Ans_IDCT)

imwrite(Ans_DCT,'lena_DCT.png')

imwrite(Ans_IDCT,'lena_IDCT.png')
```

Result：

lena_DCT.png



lena_IDCT.png



Additional Bonus: Fast DCT Algorithm

Spatial Domain

FDCT Use formula：

$$
\begin{aligned}
&f(u, v) \\
&= \frac{1}{4} C(u) C(v) \sum_{i=0}^{7} \sum_{j=0}^{7} s(i, j) \cos\left(\frac{(2i+1) u\pi}{16}\right) \cos\left(\frac{(2j+1) v\pi}{16}\right).
\end{aligned}
$$

FIDCT Use formula：

$$
\begin{aligned}
&s'(i, j) \\
&= \frac{1}{4} \sum_{u=0}^{7} \sum_{v=0}^{7} C(u) C(v) f(u, v) \cos\left(\frac{(2i+1) u\pi}{16}\right) \cos\left(\frac{(2j+1) v\pi}{16}\right),
\end{aligned}
$$

Code：

```
clc;clear all;close all;
in_image=imread('lena.bmp');

% The array of variables needed for "u", "v", "i", "j" in the FDCT
formula
cosines_DCT = [1.0000  1.0000  1.0000  1.0000  1.0000  1.0000  1.0000
1.0000
                0.9808  0.8315  0.5556  0.1951 -0.1951 -0.5556 -0.8315 -
0.9808
                0.9239  0.3827 -0.3827 -0.9239 -0.9239 -0.3827  0.3827
0.9239
                0.8315 -0.1951 -0.9808 -0.5556  0.5556  0.9808  0.1951 -
0.8315
                0.7071 -0.7071 -0.7071  0.7071  0.7071 -0.7071 -0.7071
0.7071
                0.5556 -0.9808  0.1951  0.8315 -0.8315 -0.1951  0.9808 -
0.5556
                0.3827 -0.9239  0.9239 -0.3827 -0.3827  0.9239 -0.9239
0.3827
```

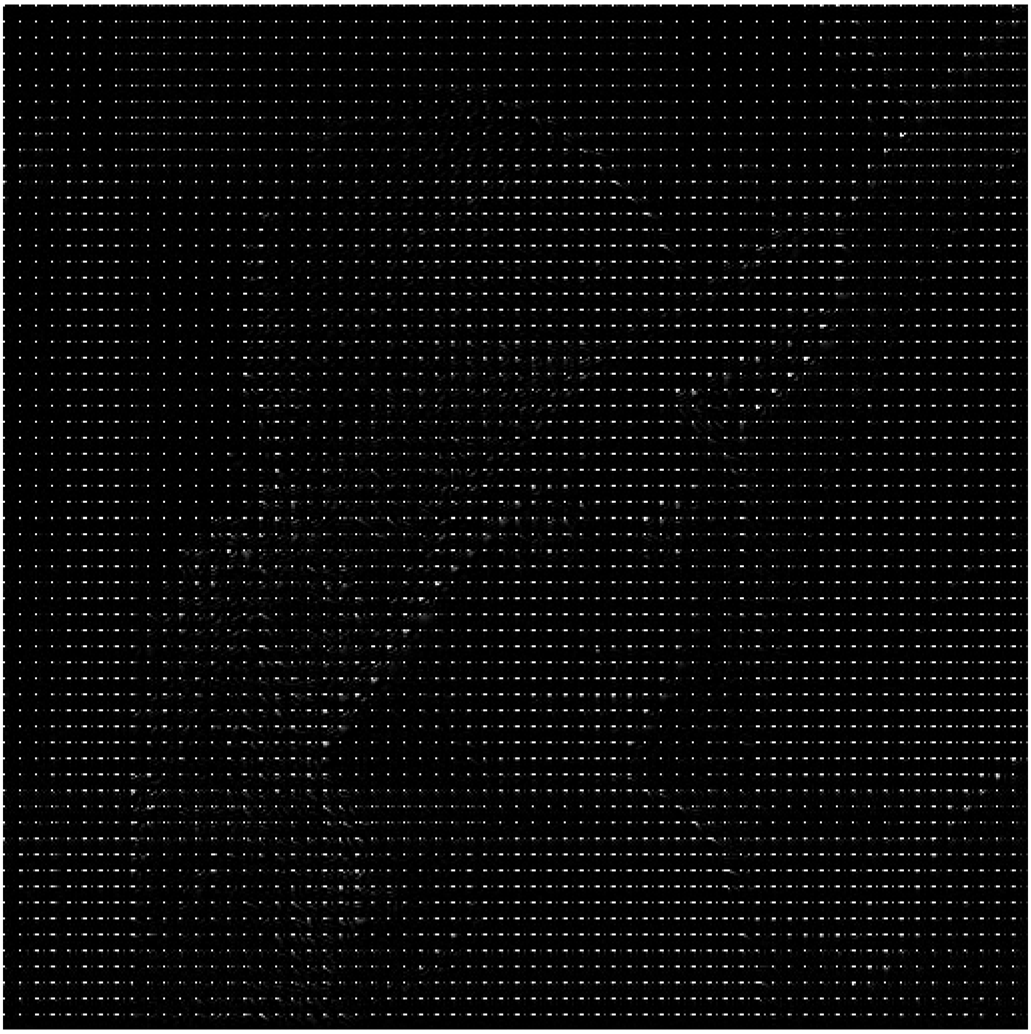
```
             0.1951 -0.5556  0.8315 -0.9808  0.9808 -0.8315  0.5556 -
0.1951];

alpha_DCT = [0.1250  0.1768  0.1768  0.1768  0.1768  0.1768  0.1768
0.1768
            0.1768  0.2500  0.2500  0.2500  0.2500  0.2500  0.2500
0.2500
            0.1768  0.2500  0.2500  0.2500  0.2500  0.2500  0.2500
0.2500
            0.1768  0.2500  0.2500  0.2500  0.2500  0.2500  0.2500
0.2500
            0.1768  0.2500  0.2500  0.2500  0.2500  0.2500  0.2500
0.2500
            0.1768  0.2500  0.2500  0.2500  0.2500  0.2500  0.2500
0.2500
            0.1768  0.2500  0.2500  0.2500  0.2500  0.2500  0.2500
0.2500
            0.1768  0.2500  0.2500  0.2500  0.2500  0.2500  0.2500
0.2500];

% The array of variables needed for "u", "v", "i", "j" in the FIDCT
formula
cosines_IDCT = [1.0000  1.0000  1.0000  1.0000  1.0000  1.0000  1.0000
1.0000
              0.9808  0.8315  0.5556  0.1951 -0.1951 -0.5556 -0.8315 -
0.9808
              0.9239  0.3827 -0.3827 -0.9239 -0.9239 -0.3827  0.3827
0.9239
              0.8315 -0.1951 -0.9808 -0.5556  0.5556  0.9808  0.1951 -
0.8315
              0.7071 -0.7071 -0.7071  0.7071  0.7071 -0.7071 -0.7071
0.7071
              0.5556 -0.9808  0.1951  0.8315 -0.8315 -0.1951  0.9808 -
0.5556
              0.3827 -0.9239  0.9239 -0.3827 -0.3827  0.9239 -0.9239
0.3827
              0.1951 -0.5556  0.8315 -0.9808  0.9808 -0.8315  0.5556 -
0.1951];
```

```
alpha_IDCT = [0.1250  0.1768  0.1768  0.1768  0.1768  0.1768  0.1768  
0.1768
             0.1768  0.2500  0.2500  0.2500  0.2500  0.2500  0.2500  
0.2500
             0.1768  0.2500  0.2500  0.2500  0.2500  0.2500  0.2500  
0.2500
             0.1768  0.2500  0.2500  0.2500  0.2500  0.2500  0.2500  
0.2500
             0.1768  0.2500  0.2500  0.2500  0.2500  0.2500  0.2500  
0.2500
             0.1768  0.2500  0.2500  0.2500  0.2500  0.2500  0.2500  
0.2500
             0.1768  0.2500  0.2500  0.2500  0.2500  0.2500  0.2500  
0.2500
             0.1768  0.2500  0.2500  0.2500  0.2500  0.2500  0.2500  
0.2500];
O_DCT = double(zeros(8,8));
O_IDCT = double(zeros(8,8));

[A,B] = size(in_image);
for i=1:8:A
   for j=1:8:B
%        Create an 8*8 array with the input image divided into 8*8 
values
      I=[in_image(i,j) in_image(i,j+1) in_image(i,j+2) 
in_image(i,j+3) in_image(i,j+4) in_image(i,5) in_image(i,6) 
in_image(i,7);
         in_image(i+1,j) in_image(i+1,j+1) in_image(i+1,j+2) 
in_image(i+1,j+3) in_image(i+1,j+4) in_image(i+1,5) in_image(i+1,6) 
in_image(i+1,7);
         in_image(i+2,j) in_image(i+2,j+1) in_image(i+2,j+2) 
in_image(i+2,j+3) in_image(i+2,j+4) in_image(i+2,5) in_image(i+2,6) 
in_image(i+2,7);
         in_image(i+3,j) in_image(i+3,j+1) in_image(i+3,j+2) 
in_image(i+3,j+3) in_image(i+3,j+4) in_image(i+3,5) in_image(i+3,6) 
in_image(i+3,7);
         in_image(i+4,j) in_image(i+4,j+1) in_image(i+4,j+2) 
in_image(i+4,j+3) in_image(i+4,j+4) in_image(i+4,5) in_image(i+4,6)
in_image(i+1,7);
        in_image(i+5,j) in_image(i+5,j+1) in_image(i+5,j+2)
in_image(i+5,j+3) in_image(i+5,j+4) in_image(i+5,5) in_image(i+5,6)
in_image(i+1,7);
        in_image(i+6,j) in_image(i+6,j+1) in_image(i+6,j+2)
in_image(i+6,j+3) in_image(i+6,j+4) in_image(i+6,5) in_image(i+6,6)
in_image(i+6,7);
        in_image(i+7,j) in_image(i+7,j+1) in_image(i+7,j+2)
in_image(i+7,j+3) in_image(i+7,j+4) in_image(i+7,5) in_image(i+7,6)
in_image(i+7,7)];

%      Start calculating FDCT, input is I array, output is O_DCT
array
       for p = 1 : 8
          for q = 1 : 8
             s_DCT = double(0);
             for m = 1 : 8
                for n = 1 : 8
%                    Make the sigma part behind the formula
                    s_DCT = s_DCT + (double(I(m,n)) *
cosines_DCT(p,m) * cosines_DCT(q,n));
                end
             end
%              Do the multiplication of C(u)C(v) in front of the
formula
             O_DCT(p,q) = alpha_DCT(p,q) * s_DCT;
          end
       end

%      Start calculating FIDCT, input is O_DCT array, output is
O_IDCT array
       for m = 1 : 8
          for n = 1 : 8
             s_IDCT = double(0);
             for p = 1 : 8
                for q = 1 : 8
%                    Do the sigma and multiplication part of the
```

```
formula
                    s_IDCT = s_IDCT + (alpha_IDCT(p,q) *
double(O_DCT(p,q)) * cosines_IDCT(p,m) * cosines_IDCT(q,n));
                 end
             end
%            Store results
             O_IDCT(m,n) = s_IDCT;
          end
       end

%      Combine the results of FIDCT conversion from multiple 8*8
arrays --> a whole image
       for aa=1:1:8
          for bb=1:1:8
             Ans_FIDCT(i+aa-1,j+bb-1)=uint8(O_IDCT(aa,bb));
          end
       end
  end
end

imshow(Ans_FIDCT)
imwrite(Ans_FIDCT,'lena_FIDCT.png')
```

Result：

lena_FDCT.png



lena_FIDCT.png