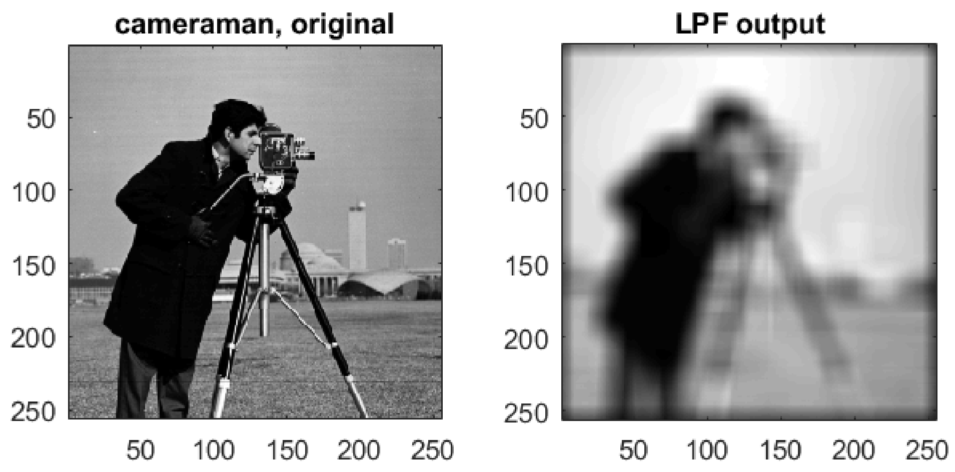
Table of Contents

Image Processing: filtering, watermarking and compression

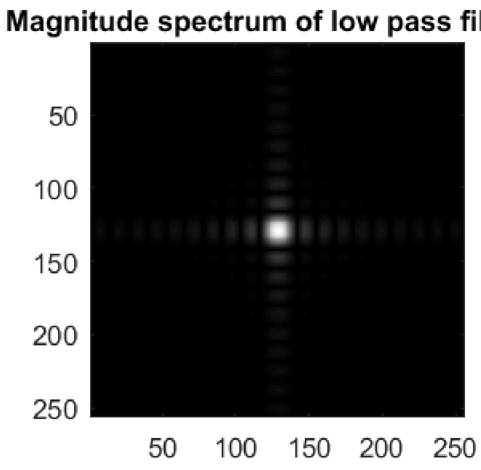
(C) 2017 by Yu Hen Hu created: April 26, 2017

```
clear all; close all;

load cameraman.mat
%x=imread('cameraman.bmp');
figure,
subplot(1,2,1),imagesc(x); colormap('gray'); axis square
title('cameraman, original')

% filtering (low pass filter)
h=(1/4)*ones(20);
xLPF=conv2(x,h,'same');
subplot(1,2,2),imagesc(xLPF), colormap('gray'); axis square
title('LPF output')
```



filtering using DFT

```
hpad0=zeros(size(x));
[mh,nh]=size(h); % size of the low pass filter used in last section
hpad0(1:mh,1:nh)=h;
HLP=fft2(hpad0);
% display the frequency response of low pass filter, after fftshift
% Note that HLP is a complex-valued matrix.
figure, subplot(1,2,1),
imagesc(fftshift(abs(HLP))); colormap('gray'), axis square
title('Magnitude spectrum of low pass filter h')
% now do the frequency domain filtering
X=fft2(x);  % X is complex valued
XLPfre=X.*HLP;
xLPfre=ifft2(XLPfre);
subplot(1,2,2),imagesc(xLPfre), colormap('gray'); axis square
title('LPF output using DFT')
```



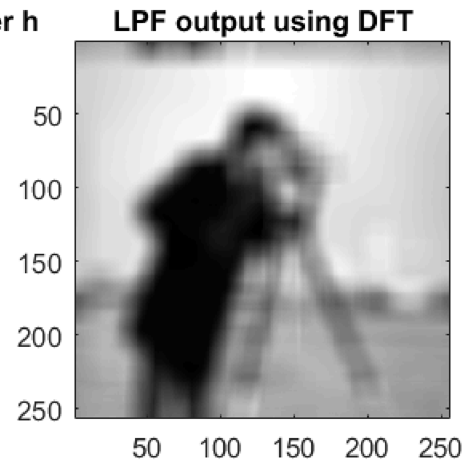


Image water marking

display image bit plane each image pixel is an unsigned 8-bit integer. we may participate a given image into 8 bit-planes. Each bit plane is a binary image the Most significant bit equals to 1 if the pixel value x(i,j) > 127=2^7-1. So bitplane{1}=[x > 127]; to see the second bit plane, compute xm1=x-128*bitplane{1}; bitplane{2}=[xm1>63 = 2^6-1]; and so on

```
xm=double(x); % xm will be a temporary variable
for i=1:7
    bitplane{i}=double([xm>2^(8-i)]);
    xm=xm-2^(8-i)*bitplane{i};
    disp(['i = ' int2str(i)])
end
bitplane{8}=xm;
figure,
for i=1:8
    % eval(['subplot(2,4,' int2str(i) ')'])
    subplot(2, 4, i)
    imagesc(bitplane{i}),colormap('gray'),axis square
    title(['Bit plane ' int2str(i)])
end
```

```
i = 1
i = 2
i = 3
i = 4
```

```
i = 5
i = 6
i = 7
```

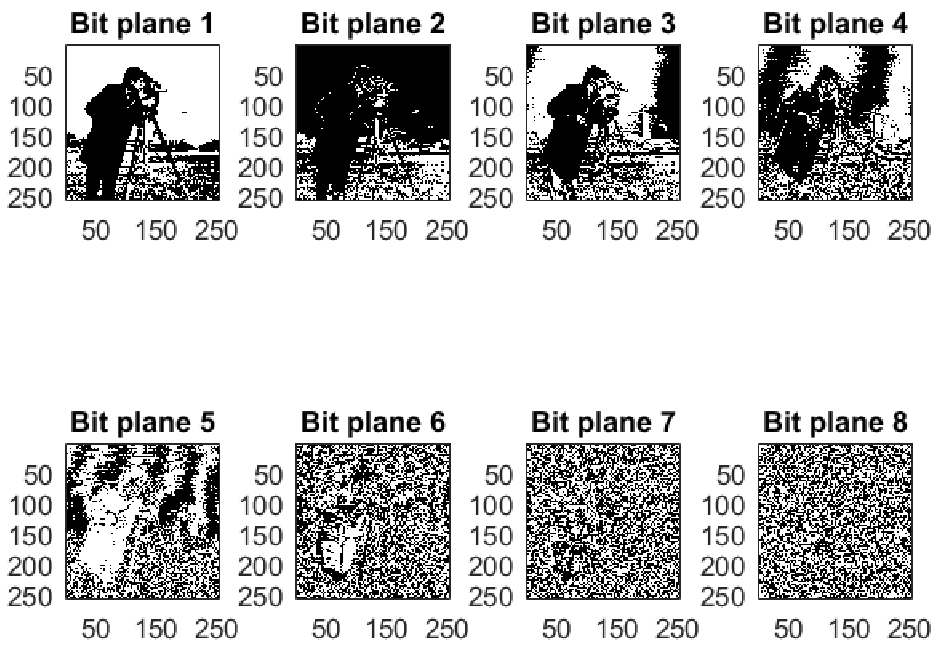
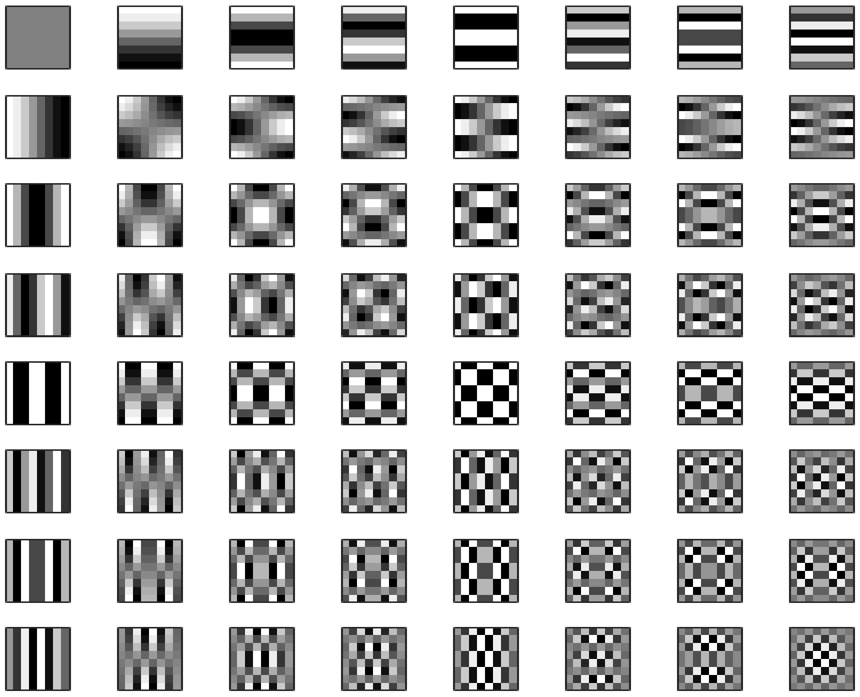


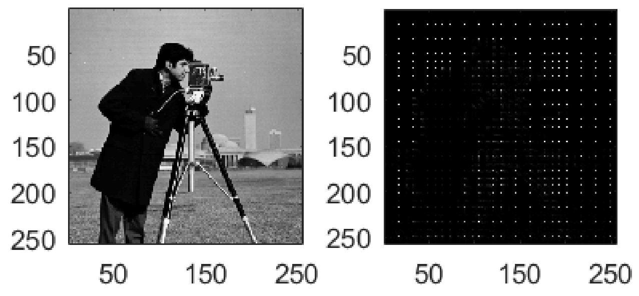
Image compression

```
% first, plot the DCT basis (from lab 8, section 4.3a)
N=8;
for m=1:N
    for n=1:N
        e=zeros(N,N);
        e(m,n)=1;
        b = idct2(e);
        subplot(8,8,m+(n-1)*N)
        imagesc(b)
        colormap(gray)
        set(gca,'Xtick',[])
        set(gca,'Ytick',[])
        axis('square')
    end
end
```



Step 1. 2D DCT on 8 x 8 blocks (Lab 8, section 4.3.b)

```
load cameraman.mat
%x=imread('cameraman.bmp');
[M,N] = size(x);
y=zeros(M,N); % initialize DCT coefficients
Mblocks = M/8;
Nblocks = N/8;
% compute DCT of 8x8 subimages
% y is the DCT coefficient of the entire image
for m = 1:Mblocks
    for n=1:Nblocks
        Mrange = (m-1)*8+1:(m-1)*8+8;
        Nrange = (n-1)*8+1:(n-1)*8+8;
        block = x(Mrange,Nrange);
        DCTblock = dct2(block);
        y(Mrange,Nrange) = DCTblock;
    end
end
figure,
subplot(1,3,1),imagesc(x),colormap('gray'),axis square
subplot(1,3,2),imagesc(abs(y)),colormap('gray'), axis square
```

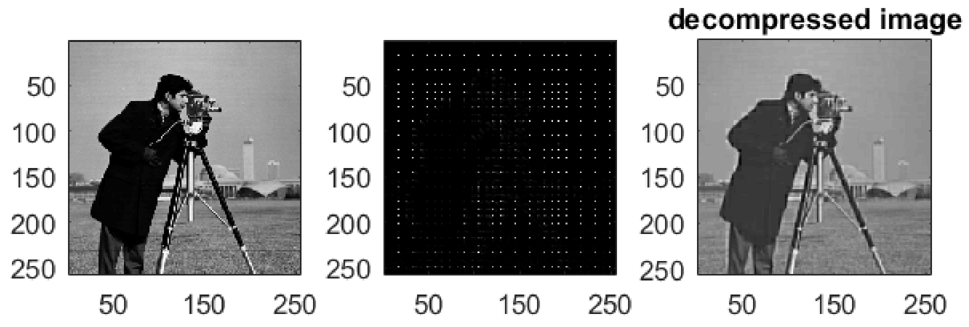


Quantization

```
% quantization matrix
Q = [16 11 10 16 24 40 51 61;
12 12 14 19 26 58 60 55;
14 13 16 24 40 57 69 56;
14 17 22 29 51 87 80 62;
18 22 37 56 68 109 103 77;
24 35 55 64 81 104 113 92;
49 64 78 87 103 121 120 101;
72 92 95 98 112 100 103 99];

yquantized=zeros(M,N); % initialize quanized DCT coefficients
% quantize each DCT coefficient
for m = 1:Mblocks
    for n=1:Nblocks
        Mrange = (m-1)*8+1:(m-1)*8+8;
        Nrange = (n-1)*8+1:(n-1)*8+8;
        quantized_DCTblock = round(y(Mrange,Nrange)./Q);
        yquantized(Mrange,Nrange)=quantized_DCTblock;
    end
end

% Now compute non-zero quantized DCT coefficients
```

```
nNZ=sum(sum(double(abs(yquantized >0)))); % number of non-zero
 entries.
CR = 100*nNZ/(M*N);
fprintf('Compression ratio = %g \n', CR)

xdecompressed=zeros(M,N);
% Finally reconstruct x from yquantized
for m = 1:Mblocks
    for n=1:Nblocks
        Mrange = (m-1)*8+1:(m-1)*8+8;
        Nrange = (n-1)*8+1:(n-1)*8+8;
        block = yquantized(Mrange,Nrange);
        IDCTblock = idct2(block);
        xdecompressed(Mrange,Nrange) = IDCTblock;
    end
end

subplot(1,3,3), imagesc(xdecompressed),colormap('gray'),axis square
title('decompressed image')
```

```
Compression ratio = 8.26263
```



Published with MATLAB® R2016a