HOME WORK-8

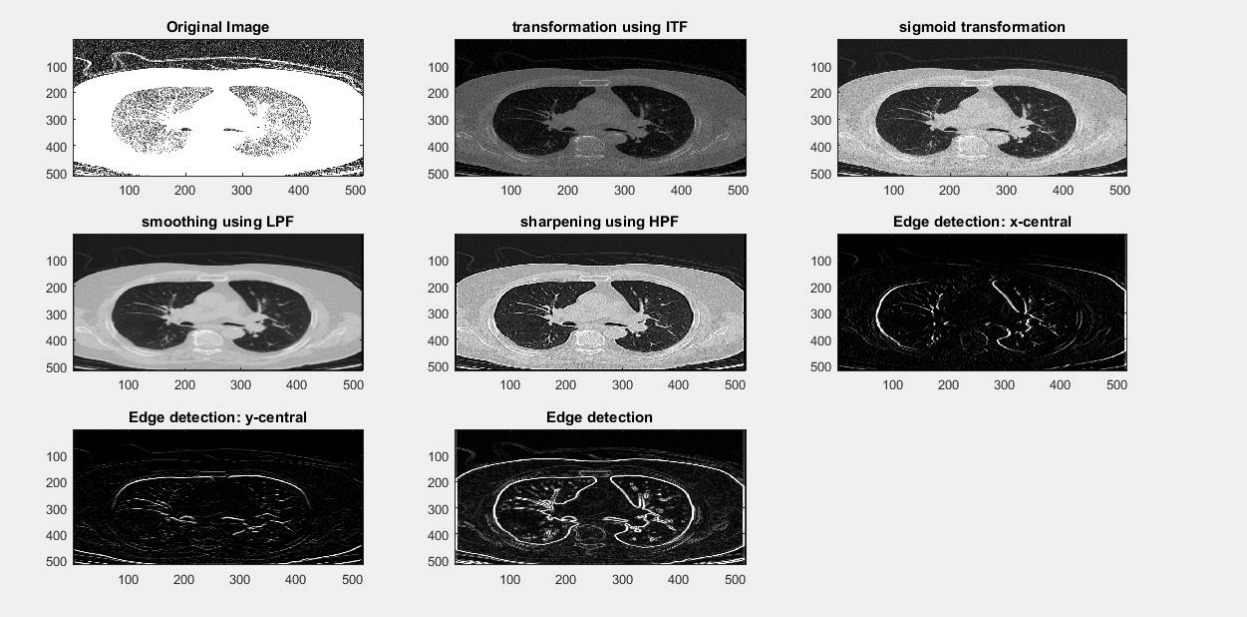
SOLUTIONS

1. Download the image file named "LowDoseCT.tif" from Chap. 5 folder; use appropriate processing tools to generate similar images given on the right. Please make a note on what processing procedures/filters you use for smoothing, sharpening, and edge detection.

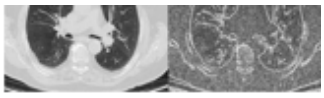
Solution:

```
clc

clear all
close all
original=double(imread('lowDoseCT.tif'));
colormap(gray(256));
subplot(3,3,1)
image(original),title('Original Image')
%transformation using ITF
trans=256.*((original-min(original(:))./(max(original(:))-min(original(:))));
subplot(3,3,2)
image(trans),title('transformation using ITF')
%sigmoid transformation
omega=67;
sigma=30;
sigmoid=256./(1+exp(-(trans-omega)./sigma));
subplot(3,3,3)
image(sigmoid),title('sigmoid transformation')
% LPF using gaussian smoothing
ksmooth1=1/256*([1,4,6,4,1;4,16,24,16,4;6,24,36,24,6;4,16,24,16,4;1,4,6,4,1 ]);
Asmooth1=conv2(ksmooth1,sigmoid);
subplot(3,3,4)
image(Asmooth1),title('smoothing using LPF')
%HPF using sharping filter
ksharp1=([-1,-1,-1;-1,9,-1;-1,-1,-1]);
Asharp1=conv2(ksharp1,Asmooth1);
subplot(3,3,5)
image(Asharp1),title('sharpening using HPF')
%Edge detection
kx=([-1,0,1;-2,0,2;-1,0,1]);
Akx=conv2(kx,Asmooth1);
subplot(3,3,6)
image(Akx),title('Edge detection: x-central')
ky=([-1,-2,-1;0,0,0;1,2,1]);
Aky=conv2(ky,Asmooth1);
subplot(3,3,7)
```

```
image(Aky),title('Edge detection: y-central')
edge=sqrt(Akx.^2+Aky.^2);
subplot(3,3,8)
image(edge),title('Edge detection')
```



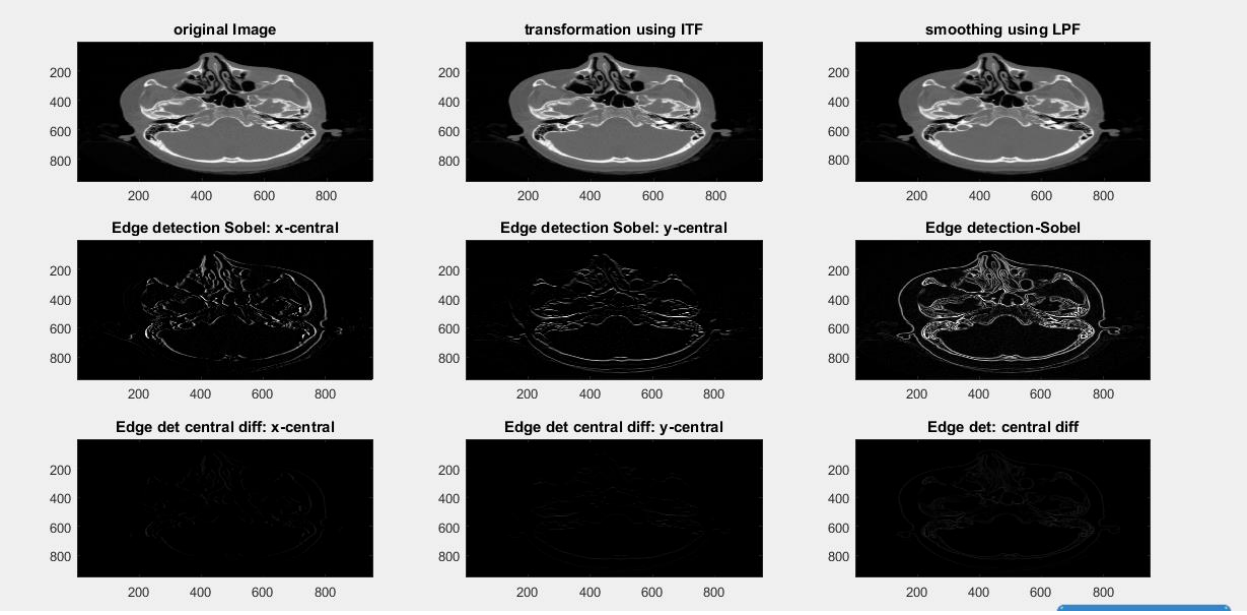
2. Understand the meaning of the **Sobel Kernel** filter by comparing it to **Central difference kernel or operation.** Then apply to one of the given images in Chap. 5 LessonData folder to perform edge detection.



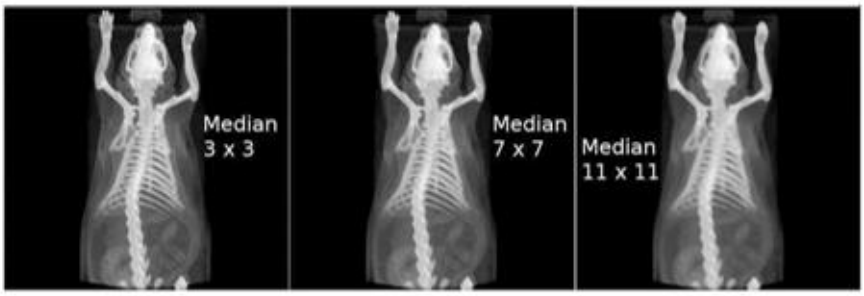
Solution:

```
clc

clear all
close all
original=double(imread('SKULLBASE_Original.jpg'));
%original=double(imread('MouseCT.jpg'));
subplot(3,3,1),colormap(gray(256))
image(original),title('original Image')
%transformation using ITF
trans=256.*((original-min(original(:)))./(max(original(:))-min(original(:))));
subplot(3,3,2)
image(trans),title('transformation using ITF')
% LPF using gaussian smoothing
```

```
ksmooth1=1/256*([1,4,6,4,1;4,16,24,16,4;6,24,36,24,6;4,16,24,16,4;1,4,6,4,1 ]);
Asmooth1=conv2(ksmooth1,trans);
subplot(3,3,3)
image(Asmooth1),title('smoothing using LPF')
%Edge detection using sobel
skx=([-1,0,1;-2,0,2;-1,0,1]);
Askx=conv2(skx,Asmooth1);
subplot(3,3,4)
image(Askx),title('Edge detection Sobel: x-central')
sky=([-1,-2,-1;0,0,0;1,2,1]);
Asky=conv2(sky,Asmooth1);
subplot(3,3,5)
image(Asky),title('Edge detection Sobel: y-central')
edge1=sqrt(Askx.^2+Asky.^2);
subplot(3,3,6)
image(edge1),title('Edge detection-Sobel')
%edge detection using Central difference
kx=(1/2)*([0,0,0;-1,0,1;0,0,0]);
Akx=conv2(kx,Asmooth1);
subplot(3,3,7)
image(Akx),title('Edge det central diff: x-central')
ky=(1/2)*([0,-1,0;0,0,0;0,1,0]);
Aky=conv2(ky,Asmooth1);
subplot(3,3,8)
image(Aky),title('Edge det central diff: y-central')
edge2=sqrt(Akx.^2+Aky.^2);
subplot(3,3,9)
image(edge2),title('Edge det: central diff')
```



3. Use the median filter to generate the following images.
The image file named “MouseCT.jpg” is in Chap.5 LessonData folder.




```
clc

clear all
close all
original=double(imread('MouseCT.jpg'));
subplot(3,3,1),colormap(gray(256))
image(original),title('original Image')
%transformation using ITF
trans=256.*((original-min(original(:))./(max(original(:))-min(original(:))));
subplot(3,3,2)
image(trans),title('transformation using ITF')

%padding matrix with zeroes
trans1=zeros(size(trans)+2);
trans2=zeros(size(trans));

%original matrix with padded matrix
        for x=1:size(trans,1)
            for y=1:size(trans,2)
                trans1(x+1,y+1)=trans(x,y);
            end
        end
      %storing 3 by 3 as an array and sorting it

for i= 1:size(trans1,1)-2
    for j=1:size(trans2,2)-2
        window=zeros(9,1);
        k=1;
        for x=1:3
            for y=1:3
                window(k)=trans1(i+x-1,j+y-1);
                k=k+1;
            end
        end

        medi=sort(window);
        %placing the median
        trans2(i,j)=medi(5);
```

```
    end
end
%converting output
trans2=uint8(trans2);
figure,imshow(trans2)
title('image after med filter')
```

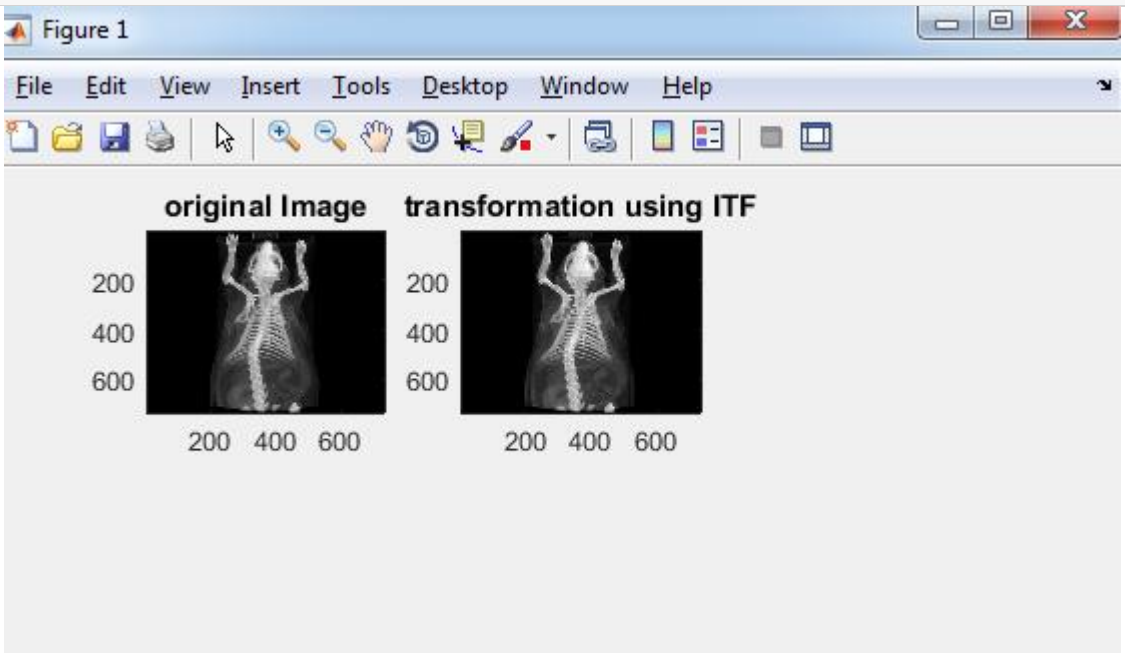


image after med filter

4. Given the following functions:

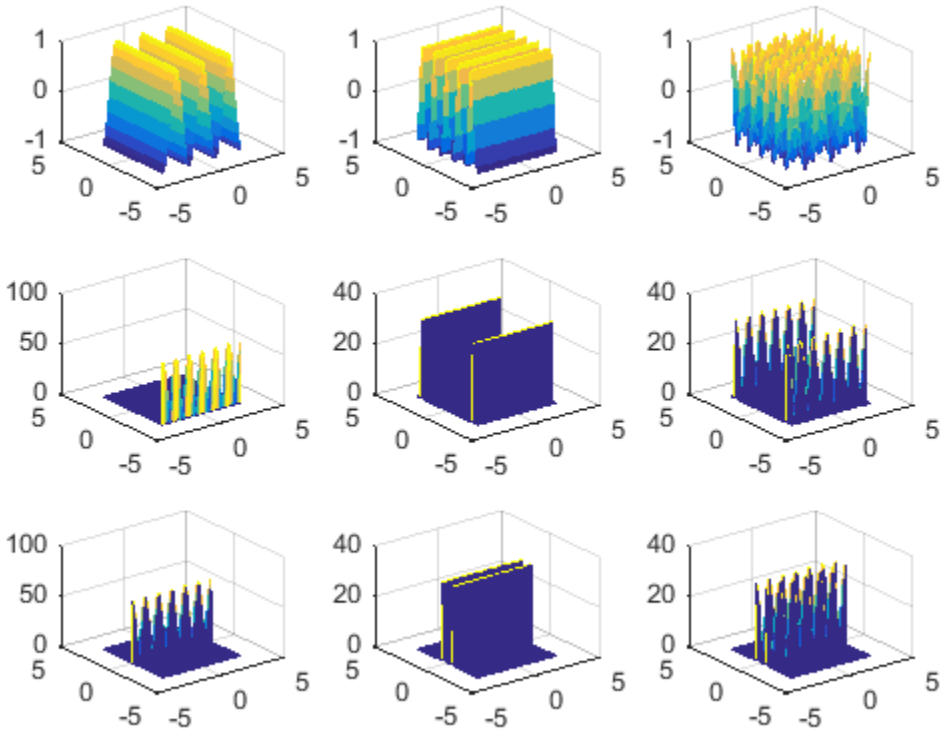
   (1) $f_1(x, y) = \cos(3x)$; (2) $f_2(x, y) = \cos(5y)$; (3) $f_3(x, y) = \cos(3x)\cos(5y)$;

   Please plot 3D plots/images for each of the functions. Please show your steps/ derivations to explain why you obtain those plots/figures.

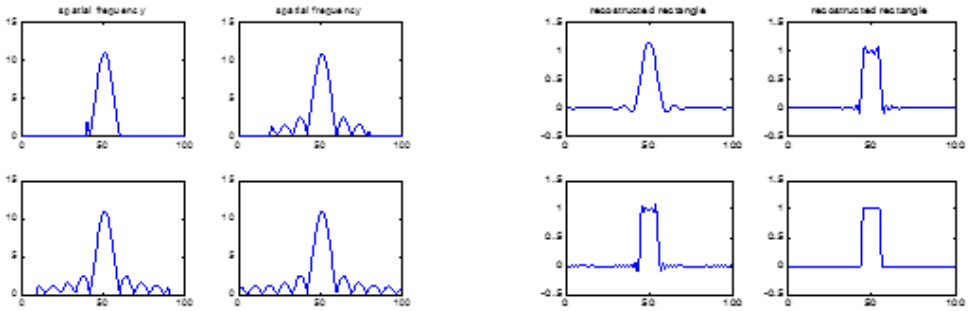
Solution:

```
clc

clear all
close all
[x,y]=meshgrid(-pi:.1:pi);
f1=cos(3.*x);
f2=cos(5.*y);
f3=f1.*f2;
subplot(3,3,1)
```

```
mesh(x,y,f1)
subplot(3,3,2)
mesh(x,y,f2)
subplot(3,3,3)
mesh(x,y,f3)
fou1=fft(f1);
subplot(3,3,4)
mesh(x,y,abs(fou1))
fou2=fft(f2);
subplot(3,3,5)
mesh(x,y,abs(fou2))
fou3=fft(f3);
subplot(3,3,6)
mesh(x,y,abs(fou3))
shif1=fftshift(fou1);
subplot(3,3,7)
mesh(x,y,abs(shif1))
shif2=fftshift(fou2);
subplot(3,3,8)
mesh(x,y,abs(shif2))
shif3=fftshift(fou3);
subplot(3,3,9)
mesh(x,y,abs(shif3))
```



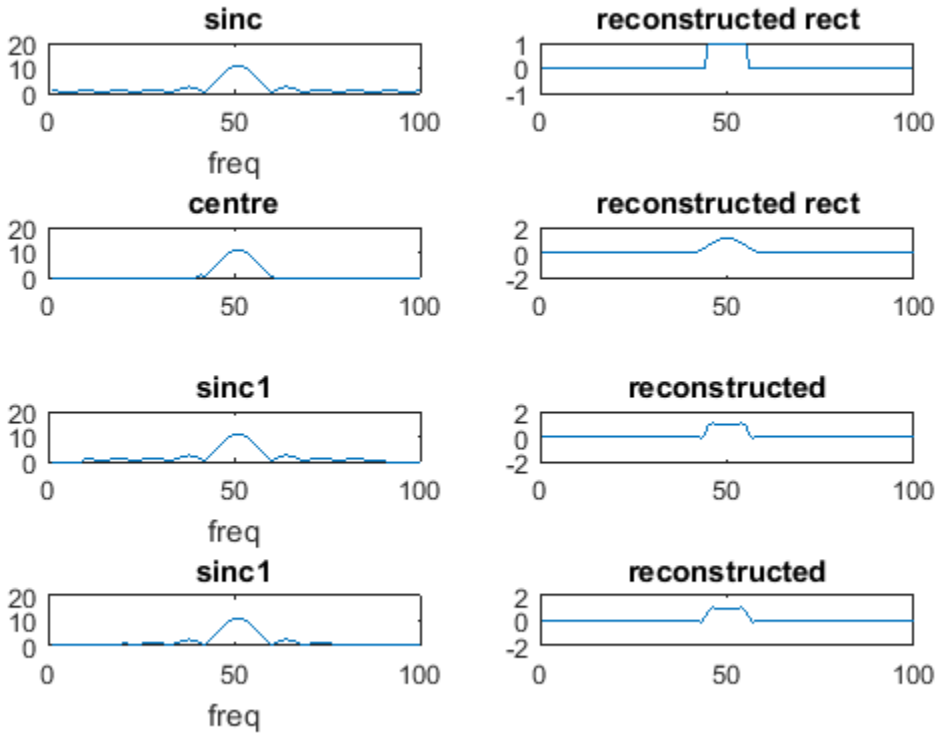
5. For a 1D rectangular function (as given in the lecture notes in Chap. 5), please use appropriate operations to generate the following figures and give explanations.



Solution:

```
clc

clear all
close all
a=zeros(100,1);
a(45:55)=1;
% subplot(4,2,1)
% plot(a),title('rect function')
b=fft(a);
b=fftshift(b);
subplot(4,2,1)
plot(abs(b)),xlabel('freq'),title('sinc')

c=ifft(ifftshift(b));
subplot(4,2,2)
plot(real(c)),title('reconstructed rect')

b(1:40)=0;
b(61:100)=0;
subplot(4,2,3)
plot(abs(b)),title('centre')

c1=ifft(ifftshift(b));
subplot(4,2,4)
plot(real(c1)),title('reconstructed rect')

b1=fft(a);
b1=fftshift(b1);
a1=zeros(100,1);
a1(10:90)=b1(10:90);
subplot(4,2,5)
plot(abs(a1)),xlabel('freq'),title('sinc1')

c2=ifft(ifftshift(a1));
subplot(4,2,6)
```

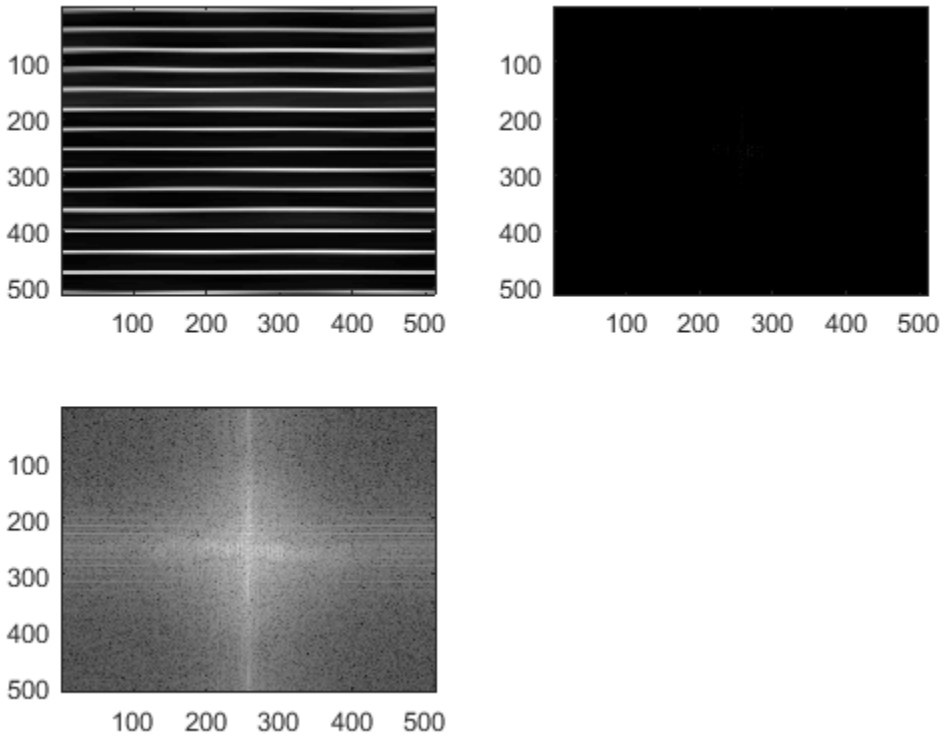
```
plot(real(c2)),title('reconstructed')

b1=fft(a);
b1=fftshift(b1);
a1=zeros(100,1);
a1(20:80)=b1(20:80);
subplot(4,2,7)
plot(abs(a1)),xlabel('freq'),title('sinc1')

c2=ifft(ifftshift(a1));
subplot(4,2,8)
plot(real(c2)),title('reconstructed')
```



6. You can find a 2D image file, 'brickwall.jpg', from the Chap.5 image data folder. We will talk or talked in the class about how to remove vertical lines from the image. Please
    a. understand and repeat similar procedures to remove the vertical lines,
    b. plot its 2D k-space images before after filtering (you may consider using log plots to show the contrast better),
    c. tell me and demonstrate if you can remove the horizontal lines, but keep the vertical lines. Give me reasons why or why not.

Solution:

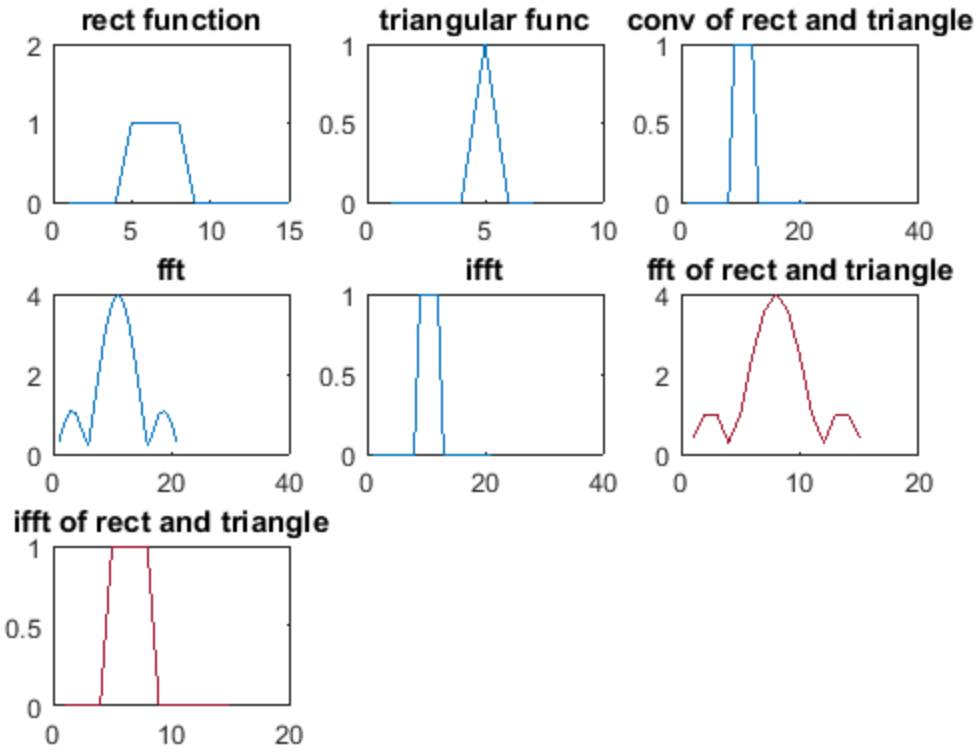
```
clc

clear all
close all
img = imread('brickwall.jpg');
fimg=fft2(img);
fimg= fftshift(fimg);

a=zeros(512, 512);
b=zeros(512, 512);
fimg2=complex(a,b);
fimg2(1:512, 255:257)=fimg(1:512, 255:257);

ft_img = real(ifft2(ifftshift(fimg2)));
subplot(2,2,1)
colormap(gray);
imagesc(ft_img)
subplot(2,2,2)
imagesc(abs(fimg))
subplot(2,2,3)
imagesc(log10(abs(fimg)))
```



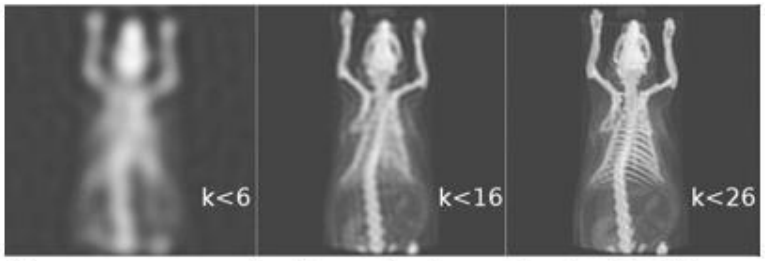
7. Given two functions below, please calculate the convolution between them, using (i) the Matlab function of 'Conv' and then (ii) fft and ifft to perform the operation/calculation. Please compare the results between them:
(1) a square function, and (2) a triangle function.

Solution:

```
clc

clear all
close all
rect=zeros(15,1);
rect(1:4)=0;
rect(5:8)=1;
rect(9:12)=0;
subplot(3,3,1)
plot(rect),xlim([0 15]),ylim([0 2])
title('rect function')
t=4:10;
y=(-1).^t;
y(1:4)=0;
y(6:7)=0;
subplot(3,3,2)
plot(y)
title('triangular func')
fina=conv(rect,y);
subplot(3,3,3)
plot(fina)
title('conv of rect and triangle')
fina1=fftshift(fft(fina));
subplot(3,3,4)
plot((abs(fina1)))
title('fft')
fina2=ifft(ifftshift(fina1));
subplot(3,3,5)
plot(real(fina2))
title('ifft')
f1=fft(rect);
f1s=fftshift(f1);
f2=fft(y);
f2s=fftshift(f2);
res1=f1s*f2s;
res2=ifft(ifftshift(res1));
subplot(3,3,6)
plot(abs(res1))
title('fft of rect and triangle')
subplot(3,3,7)
plot(abs(res2))
title('ifft of rect and triangle')
```



8. Using frequency filtering in Fourier space, please obtain similar figures given below.

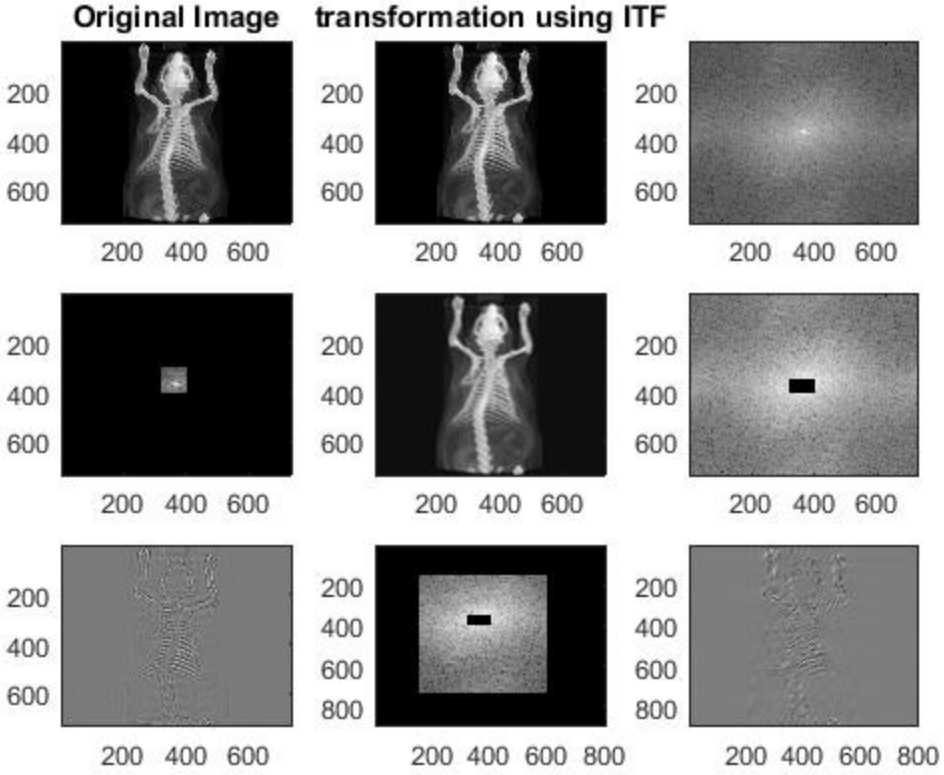


Following the same approach, please implement a simple high pass and band pass filter. A sample output can be found in Figure 5.37.

Solution:

```
clc


clear all
close all
original=double(imread('MouseCT.jpg'));
colormap(gray(256));
subplot(3,3,1)
```

```
image(original),title('Original Image')
%transformation using ITF
trans=256.*((original-min(original(:))./(max(original(:))-min(original(:))));
subplot(3,3,2)
image(trans),title('transformation using ITF')
%fft
a=fftshift(fft2(trans));
subplot(3,3,3)
imagesc(log10(abs(a)))
%low pass
y1=zeros(733,733);
y2=zeros(733,733);
y=complex(y1,y2);
y(300:400,320:400)=a(300:400,320:400);
subplot(3,3,4)
imagesc(log10(abs(y)))
b=ifft2(ifftshift(y));
subplot(3,3,5)
imagesc(real(b))
%high pass
a(350:400,320:400)=0;
subplot(3,3,6)
imagesc(log10(abs(a)))
c=ifft2(ifftshift(a));
subplot(3,3,7)
imagesc(real(c))
%band pass
a(350:400,320:400)=0;
a(1:900,1:150)=0;
a(1:900,600:800)=0;
a(1:150,100:600)=0;
subplot(3,3,8)
imagesc(log10(abs(a)))
d=ifft2(ifftshift(a));
subplot(3,3,9)
imagesc(real(d))
```


Original Image
transformation using ITF
200
400
600
800
200 400 600
200 400 600 800