**east west university**

**Lab Report - 03**

**Department:** **Computer Science and Engineering**

**Course Title:** Digital Image Processing

**Course Code:** CSE438

**Section No:** 02

**Submitted To**:

Dr. Ahmed Wasif Reza

Professor, Department of CSE

Additional Director, Institutional Quality Assurance

Cell (IQAC)

**Submitted By**:

Name: Apurba Roy Ajay

ID: 2018-3-60-063

**1. Sharpen the following image by applying the following and find out which one is better:**

**A.Unsharp Masking**

**Code:**

image\_1 = imread('tumor.jpg');

H=ones(3)/9;

B = imfilter(image\_1,H,'replicate');

mask = imsubtract(image\_1,B);

imshowpair(image\_1,mask,'montage');

**Output:**

A close-up of a brain scan

Description automatically generated

**B. High Boost Filtering**

**Code:**

I1 = imread('tumor.jpg');

H = [0,-1,0;-1,5,-1;0,-1,0];

I2 = imfilter(I1,H,'replicate');

imshowpair(I1,I2,'montage');

**Output:**

A close-up of a brain scan

Description automatically generated

**2. Sharpen the following image using the concept of Laplacian Filtering.**

**Code:**

I2 = imread('Spine\_CT.jpg');

I2 = rgb2gray(I2);

mask = [0 1 0; 1 -4 1; 0 1 0];

res = uint8(filter2(mask,I2,"same"));

shape = imsubtract(I2,res);

subplot(1,2,1);imshow(I2);title('Orginal Image');

subplot(1,2,2);imshow(shape);title('Sharp Image');

**Output:**

Close-up of spine x-ray

Description automatically generated

**3. Use Roberts-cross, Sobel, and Prewitt operators to detect the edge of the following**

**image.**

**Code:**

**Use Roberts-cross-**

image = imread('Head\_CT\_Scan.jpg');

Gray = rgb2gray(image);

Detect = edge(Gray,'roberts');

subplot(1,3,1);imshow(image);title('Original Image');

subplot(1,3,2);imshow(Gray);title('Gray Image');

subplot(1,3,3);imshow(Detect);title('Roberts-cross');

**Output:**

A close-up of a brain scan

Description automatically generated

**Code:**

**Sobel-**

image = imread('Head\_CT\_Scan.jpg');

Gray = rgb2gray(image);

Detect = edge(Gray,'sobel');

subplot(1,3,1);imshow(image);title('Original Image');

subplot(1,3,2);imshow(Gray);title('Gray Image');

subplot(1,3,3);imshow(Detect);title('Sobel Detector');

**Output:**

 **Code:**

**Prewitt-**

image = imread('Head\_CT\_Scan.jpg');

Gray = rgb2gray(image);

Detect = edge(Gray,'prewitt');

subplot(1,3,1);imshow(image);title('Original Image');

subplot(1,3,2);imshow(Gray);title('Gray Image');

subplot(1,3,3);imshow(Detect);title('Prewitt Detector');

**Output:**

A close-up of a brain scan

Description automatically generated

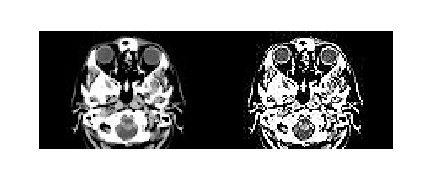
**4. Show performance comparison among High Boost, Unsharp, Laplacian Roberts-cross,**

**Sobel, Prewitt, and Canny filtering for edge detection – find out which one is better for**

**the given image.**

**Answer:**

**1.high boost –**



**2. Unsharp-**

A close-up of a scan of a brain

Description automatically generated

**3. Laplacian Roberts-cross-**

A close-up of a brain scan

Description automatically generated

**4.Sobel-**

A close-up of a brain scan

Description automatically generated

**5.Prewitt-**

A close-up of a brain scan

Description automatically generated

**Here, for the given image High boost filter is given better result.**

**5.Apply Fourier transform to transform an image (Figure 1) from the spatial domain to the**

**frequency domain. Apply inverse Fourier transform to transform the image from the**

**frequency domain to the spatial domain.**

**Fourier transform and Inverse Fourier transform:**

**Code:**

img = imread('Brain MRI.png');

img2 = double(mat2gray(img));

F = fftshift(fft2(img2));

I = log(1+abs(F));

imginv = abs(ifft2(F));

subplot(2,2,1);imshow(img);title('orginal image')

subplot(2,2,2);imshow(I);title('Fourier transform')

subplot(2,2,3);bar(I);title('frequency bar graph')

subplot(2,2,4);imshow(imginv);title('inverse Fourier transform')

**Output:**

A close-up of a brain scan

Description automatically generated

**6. Compress the image (Figure 1) using Discrete Cosine Transform (DCT), Haar**

**Transform, and DCT-Haar, and find out which one is better in terms of compression ratio**

**and PSNR for the given image.**

**Code:**

img = im2gray(img);

compress = dct2(img);

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

Compress(abs(Compress)<10) = 0;

K = idct2(Compress);

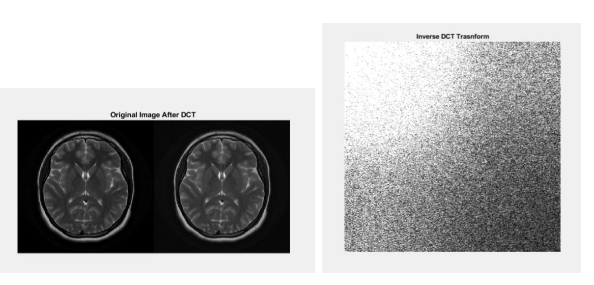
K = rescale(K);

figure,montage({img,K});title ('Original Image After DCT');

InverseDct = abs(idct2(img));

figure,imshow(inverseDct);title ('Inverse DCT Transformation');

**Output:**



**Compress the image (Figure 1) using Discrete Cosine Transform.**

img = imread('Brain MRI.png');

imgage = im2gray(img);

compress = dct2(imgage);

Compress(abs(Compress) < 10) = 0;

K = idct2(Compress);

K = rescale(K);

subplot(221);imshow(img);title ('Original Image');

subplot(222);imshow(K);title ('DCT Compress Image');

**Output:**

A black rectangular object with a white background

Description automatically generated

**Compress the image using Haar Transform.**

**Code:**

I = 'Brain MRI.png';

delta = 0.0001;

disp(delta);

1.0000e-04

if(delta>1 || delta <0)

error('harr\_wt: Delta value range is between 0 to 1');

end

H1=[0.5 0 0 0 0.5 0 0 0;0.5 0 0 0 -0.5 0 0 0;0 0.5 0 0 0 0.5 0 0 ;0 0.5 0 0 0 -0.5 0 0; 0 0 0.5 0 0 0 0.5 0;0 0 0.5 0 0 0 -0.5 0;0 0 0 0.5 0 0 0 0.5;0 0 0 0.5 0 0 0 -0.5;];

H2=[0.5 0 0.5 0 0 0 0 0;0.5 0 -0.5 0 0 0 0 0;0 0.5 0 0.5 0 0 0 0;0 0.5 0 -0.5 0 0 0 0; 0 0 0 0 1 0 0 0;0 0 0 0 0 1 0 0;0 0 0 0 0 0 1 0;0 0 0 0 0 0 0 1;];

H3=[0.5 0.5 0 0 0 0 0 0;0.5 -0.5 0 0 0 0 0 0;0 0 1 0 0 0 0 0;0 0 0 1 0 0 0 0;0 0 0 0 1 0 0 0; 0 0 0 0 0 1 0 0;0 0 0 0 0 0 1 0;0 0 0 0 0 0 0 1;];

H1o = (H1.\*(2^0.5));

H2o = (H2.\*(2^0.5));

H3o = (H3.\*(2^0.5));

Ho=normc(H1o\*H2o\*H3o);

H = H1\*H2\*H3;

x=double(imread(I));

len=length(size(x));

if len~=2

error('harr wt: Input image must be a gray image, use "harr\_wt\_rgb" function to compress RGB image ');

end

yo = zeros(size(x));

y = zeros(size(x));

[r,c]=size(x);

for i=0:8:r-8

for j=0:8:c-8

p=i+1;

q=j+1;

yo(p:p+7,q:q+7)=(Ho')\*x(p:p+7,q:q+7)\*Ho;

y(p:p+7,q:q+7)=(H')\*x(p:p+7,q:q+7)\*H;

end

end

figure;

imshow(x/255);

n1=nnz(y);

zo=yo;

m=max(max(yo));

yo=yo/m;

yo(abs(yo)<delta)=0;

yo=yo\*m;

z=y;

y=y/m;

y(abs(y)<delta)=0;

y=y\*m;

n2=nnz(y);

for i=0:8:r-8

for j=0:8:c-8

p=i+1;

q=j+1;

zo(p:p+7,q:q+7)=Ho\*yo(p:p+7,q:q+7)\*Ho;

z(p:p+7,q:q+7)=inv(H')\*y(p:p+7,q:q+7)\*inv(H);

end

end

figure;

subplot(121);

imshow(x/255);

title("Original Image");

subplot(122)

imshow(z/255);

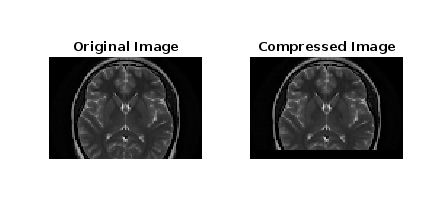
title("Compressed Image");

imwrite(x/255,'original.tif');

imwrite(z/255,'compossed.tif');

compression\_ratio = n2/n1;

**Output:**



**7.Apply three types of high pass filtering in the frequency domain in Figure 1 and find out**

**which one is better to produce the enhanced image (sharpen) for the given image (output**

**must show all steps as shown in Figure 2).**

**i. Ideal high pass filter (IHPF)-**

**Code:**

Fig\_1=imread('Brain MRI.png');

[M,N] = size(Fig\_1);

FT\_Img = fft2(double(Fig\_1));

D0 = 10;

u=0:(M-1);

idx=find(u>M/2);

u(idx)=u(idx)-M;

v=0:(N-1);

idy=find(v>N/2);

v(idy)=v(idy)-N;

[V,U]=meshgrid(v,u);

D=sqrt(U.^2+V.^2);

H=double(D>D0);

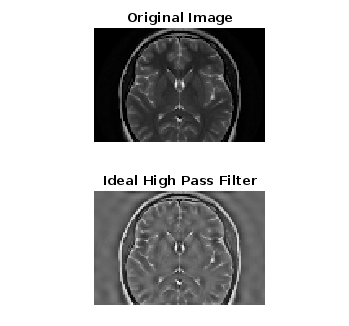
G=H.\*FT\_Img;

output\_image= real(ifft2(double(G)));

subplot(2, 1, 1), imshow(Fig\_1),title('Original Image');

subplot(2, 1, 2), imshow(output\_image, [ ]);title('Ideal High Pass Filter');

**Output:**



**ii. Butterworth high pass filter (BHPF)**

**Code:**

Fig\_1=imread('Brain MRI.png');

[M,N] = size(Fig\_1);

FT\_Img = fft2(double(Fig\_1));

D0 = 10;

u=0:(M-1);

idx=find(u>M/2);

u(idx)=u(idx)-M;

v=0:(N-1);

idy=find(v>N/2);

v(idy)=v(idy)-N;

[V,U]=meshgrid(v,u);

D=sqrt(U.^2+V.^2);

n=2;

H = 1./(1 + (D0./D).^(2\*n));

G = H.\*FT\_Img;

output\_image = real(ifft2(double(G)));

subplot(2, 1, 1), imshow(Fig\_1),title('Original Image');

subplot(2, 1, 2), imshow(output\_image, [ ]);title('Butterworth High Pass Filter');

**Output:**

A close-up of a brain scan

Description automatically generated

**iii. Gaussian high pass filter (GHPF)**

**Code:**

im = imread('Brain MRI.png');

fc = 100;

imf = fftshift(fft2(im));

[co,ro] = size(im);

cx = round(co/2);

cy = round(ro/2);

H = zeros(co,ro);

for i = 1:co

for j = 1:ro

d=(i-cx).^2+(j-cy).^2;

H(i,j) = exp(-d/2/fc/fc);

end

end

H = 1-H;

outf=imf.\*H;

out = abs(ifft2(outf));

subplot(2,2,1),imshow(im),title('Original Image');

subplot(2,2,2),imshow(out),title('Filtered Image');

subplot(2,2,3),imshow(H),title('2D View of H');

subplot(2,2,4),surf(H),title('3D View of H');

**Output:**

A close-up of a brain

Description automatically generated

**8. Apply three types of low pass filtering in the frequency domain in Figure 1 and find out**

**which one is better to produce the smoothen image for the given image (output must**

**show all steps as shown in Figure 2).**

**i.Ideal lowpass filter (ILPF)-**

Fig\_1=imread('Brain MRI.png');

[M,N] = size(Fig\_1);

FT\_Img = fft2(double(Fig\_1));

D0 = 30;

u=0:(M-1);

idx=find(u>M/2);

u(idx)=u(idx)-M;

v=0:(N-1);

idy=find(v>N/2);

v(idy)=v(idy)-N;

[V,U]=meshgrid(v,u);

D=sqrt(U.^2+V.^2);

H=double(D<=D0);

G=H.\*FT\_Img;

output\_image= real(ifft2(double(G)));

subplot(2, 1, 1), imshow(Fig\_1),title('Original Image');

subplot(2, 1, 2), imshow(output\_image, [ ]);title('Ideal Low Pass Filter');

**Output-**

A close-up of a brain scan

Description automatically generated

**ii. Butterworth low pass filter (BLPF)-**

Fig\_1=imread('Brain MRI.png');

[M,N] = size(Fig\_1);

FT\_Img = fft2(double(Fig\_1));

D0 = 20;

u=0:(M-1);

idx=find(u>M/2);

u(idx)=u(idx)-M;

v=0:(N-1);

idy=find(v>N/2);

v(idy)=v(idy)-N;

[V,U]=meshgrid(v,u);

D=sqrt(U.^2+V.^2);

n=2;

H = 1./(1 + (D0./D).^(2\*n));

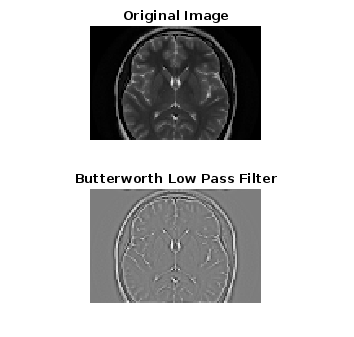
G = H.\*FT\_Img;

output\_image = real(ifft2(double(G)));

subplot(2, 1, 1), imshow(Fig\_1),title('Original Image');

subplot(2, 1, 2), imshow(output\_image, [ ]);title('Butterworth Low Pass Filter');

**output:**



**iii. Gaussian low pass filter (GLPF)**

im = imread('Brain MRI.png');

fc = 100;

imf = fftshift(fft2(im));

[co,ro] = size(im);

cx = round(co/2);

cy = round(ro/2);

H = zeros(co,ro);

for i = 1:co

for j = 1:ro

d=(i-cx).^2+(j-cy).^2;

H(i,j) = exp(-d/2/fc/fc);

end

end

outf=imf.\*H;

out = abs(ifft2(outf));

subplot(2,2,1),imshow(im),title('Original Image');

subplot(2,2,2),imshow(out),title('Filtered Image');

subplot(2,2,3),imshow(H),title('2D View of H');

subplot(2,2,4),surf(H),title('3D View of H');

Output:

