**Experiment-1.2**

**Aim of the Experiment :** Image enhancement/denoising using spatial domain and frequency domain filters

1. Image enhancement/denoising using spatial domain filter
2. Image enhancement/denoising using frequency domain filter
3. **IMAGE ENHANCEMENT/DENOISING USING SPATIAL DOMAIN FILTER**

Problem Description :

Spatial Filtering technique is used directly on pixels of an image. Mask is usually considered to be added in size so that it has specific center pixel. This mask is moved on the image such that the center of the mask traverses all image pixels.

There are two types:

1. Mean Filter:  
   Linear spatial filter is simply the average of the pixels contained in the neighborhood of the filter mask. The idea is replacing the value of every pixel in an image by the average of the grey levels in the neighborhood define by the filter mask.
2. Order Statistics Filter:  
   It is based on the ordering the pixels contained in the image area encompassed by the filter. It replaces the value of the center pixel with the value determined by the ranking result. Edges are better preserved in this filtering.

Sharpening Spatial Filter: It is also known as derivative filter. The purpose of the sharpening spatial filter is just the opposite of the smoothing spatial filter. Its main focus in on the removal of blurring and highlight the edges. It is based on the first and second order derivative.

Laplacian Filter: Enhances edges by emphasizing regions of rapid intensity change. It is often combined with the original image to enhance details.

Sobel and Prewitt Filters: These filters emphasize vertical and horizontal edges, respectively.

Both Laplacian and Sobel Filters comes under Edge Enhancement Filters

Histogram Equalization: Adjusts the intensity values of an image to enhance the overall contrast. It redistributes the pixel values to cover the entire intensity range.

Code (Image Complement and Pixel Mapping):

f = imread('C:/Users/milan\OneDrive/Desktop\MAI - 1/Sem 2/Computer Vision Lab/Images/lena\_gray.png');

subplot(2,3,1), imshow(f)

title("PNG Image")

% Convert format of image

imwrite(f,'lena\_gr.jpg');

g = imread('lena\_gr.jpg');

subplot(2,3,2), imshow(g);

title("JPG Image");

% Complement of image (255-value)

%h1 = imcomplement(g);

h1 = 255-g;

subplot(2,3,3), imshow(h1);

title("Complement")

% Convert pixels into 0 to 1 for specific operation

%h2 = imadjust(g, [0 1], [1 0]);

min\_input = 0 \* 255;

max\_input = 1 \* 255;

min\_output = 1 \* 255;

max\_output = 0 \* 255;

h2 = (double(g) - min\_input) \* ((max\_output - min\_output) / (max\_input - min\_input)) + min\_output;

h2 = uint8(h2);

subplot(2,3,4), imshow(h2);

title("Adjusted Image 1")

% Mapping of pixels

%h3 = imadjust(g, [0.3 0.7], [0.2 1.0]);

min\_input = 0.3 \* 255;

max\_input = 0.7 \* 255;

min\_output = 0.2 \* 255;

max\_output = 1.0 \* 255;

h3 = (double(g) - min\_input) \* ((max\_output - min\_output) / (max\_input - min\_input)) + min\_output;

h3 = uint8(h3);

subplot(2,3,5), imshow(h3);

title("Adjusted Image 1")

% Convert pixel back into 0 to 255

m = 0.5; E = 0.5;

ff = 1./(1+(m./(double(g)+eps)).^E);

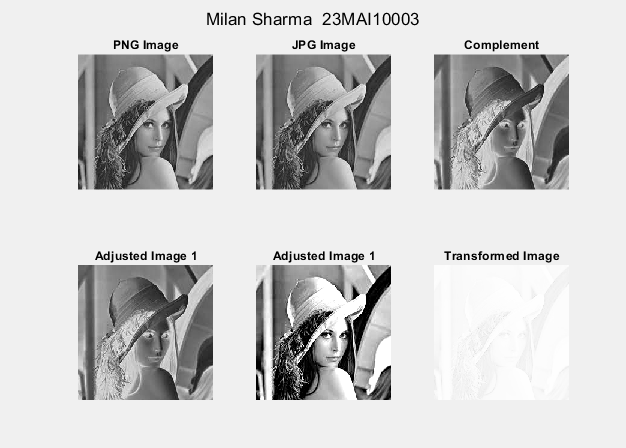
gg = im2uint8(mat2gray(ff));

subplot(2,3,6),imshow(gg);

title("Transformed Image")

sgtitle('Milan Sharma 23MAI10003')

Output :



Code (Histogram):

A=imread('C:/Users/milan\OneDrive/Desktop\MAI - 1/Sem 2/Computer Vision Lab/Images/lena\_gray.png');

Eq\_A =histeq(A);

%H\_A = imhist (A, 256);

[row1,column1] = size(A);

H\_A= (zeros(1,256));

intensity = 0;

while( intensity <256 )

count = 0;

for i = 1:row1

for j = 1:column1

if A(i,j) == intensity

count = count+1;

end

end

end

H\_A(1,intensity+1) = count;

intensity = intensity+1;

end

%H\_Eq\_A= imhist (Eq\_A, 256);

[row2,column2] = size(Eq\_A);

H\_Eq\_A= (zeros(1,256));

intensity = 0;

while( intensity <256 )

count = 0;

for i = 1:row2

for j = 1:column2

if Eq\_A(i,j) == intensity

count = count+1;

end

end

end

H\_Eq\_A(1,intensity+1) = count;

intensity = intensity+1;

end

figure

subplot(2,2,1); imshow(A)

title("Gray Image")

subplot(2,2,2);bar(H\_A);

title("Histogram")

subplot(2,2,3); imshow(Eq\_A)

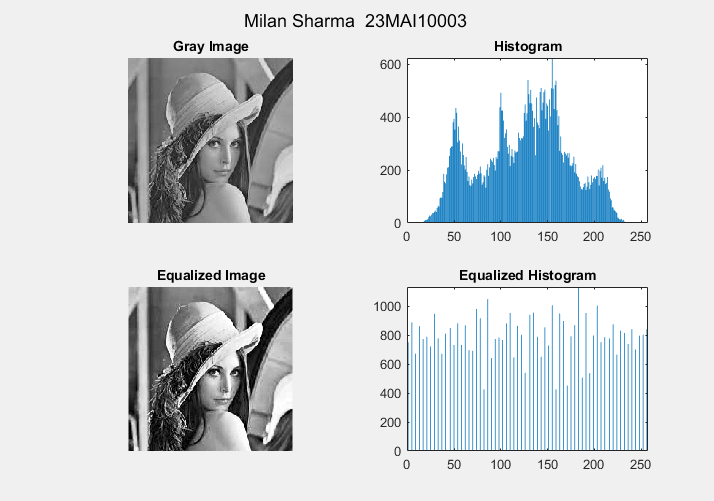
title("Equalized Image")

subplot(2,2,4); bar(H\_Eq\_A);

title("Equalized Histogram")

sgtitle("Milan Sharma 23MAI10003")

Output:



Code (Sobel Filter and Laplace Filter, Motion, Blurred and Sharpened Image):

%load in lunar north pole image

f=imread('C:/Users/milan\OneDrive/Desktop\MAI - 1/Sem 2/Computer Vision Lab/Images/moonB.png');

% creates 3x3 laplacian, alpha=0 [0:1]

w4=fspecial('laplacian',0);

% create a Laplacian that fspecial can’t

w8=[1 1 1;1 -8 1;1 1 1];

% output same as input unit8 so negative values are truncated.

% Convert to double to keep negative values.

f=im2double(f);

% filter using default values

g4=f-imfilter(f,w4,'replicate');

% filter using default values

g8=f-imfilter(f,w8,'replicate');

% Sobel Filter

h = fspecial('sobel');

sfi = imfilter(double(f),h, 0, 'conv');

figure

% display original image

subplot(2,4,1); imshow(f);

title('Original Image');

subplot(2,4,2),imshow(sfi, []);

title("Sobel Filter")

% display g4 processed image

subplot(2,4,3); imshow(g4);

title('Laplace Filter 1');

% display g8 processed image

subplot(2,4,4); imshow(g8);

title('Laplace Filter 2');

I = imread('C:/Users/milan\OneDrive/Desktop\MAI - 1/Sem 2/Computer Vision Lab/Images/moonB.png');

subplot(2,4,5);imshow(I);title('Original Image');

H = fspecial('motion',50,45);

MotionBlur = imfilter(I,H);

subplot(2,4,6);imshow(MotionBlur);title('Motion Blurred Image');

H = fspecial('disk',10);

blurred = imfilter(I,H);

subplot(2,4,7);imshow(blurred);title('Blurred Image');

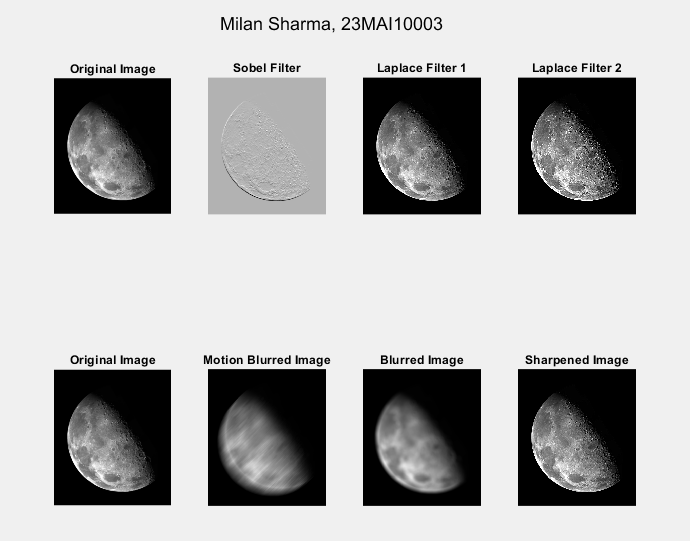
H = fspecial('unsharp');

sharpened = imfilter(I,H);

subplot(2,4,8);imshow(sharpened);title('Sharpened Image');

sgtitle('Milan Sharma 23MAI10003')

Output :



1. **IMAGE ENHANCEMENT/DENOISING USING FREQUENCY DOMAIN FILTER**

**Problem Description :**

Frequency Domain Filters are used for smoothing and sharpening of image by removal of high or low frequency components. Sometimes it is possible of removal of very high and very low frequency. Frequency domain filters are different from spatial domain filters as it basically focuses on the frequency of the images. It is basically done for two basic operation i.e., Smoothing and Sharpening.

1. Low pass filter :  
   Low pass filter removes the high frequency components that means it keeps low frequency components. It is used for smoothing the image. It is used to smoothen the image by attenuating high frequency components and preserving low frequency components.

Mechanism of low pass filtering in frequency domain is given by:

G(u, v) = H(u, v) . F(u, v)

where F(u, v) is the Fourier Transform of original image

and H(u, v) is the Fourier Transform of filtering mask

1. High pass filter :  
   High pass filter removes the low frequency components that means it keeps high frequency components. It is used for sharpening the image. It is used to sharpen the image by attenuating low frequency components and preserving high frequency components.

Mechanism of high pass filtering in frequency domain is given by:

G(u, v) = H(u, v) . F(u, v)

H(u, v) = 1 - H'(u, v)

where H(u, v) is the Fourier Transform of high pass filtering

and H'(u, v) is the Fourier Transform of low pass filtering

1. Band pass filter :  
   Band pass filter removes the very low frequency and very high frequency components that means it keeps the moderate range band of frequencies. Band pass filtering is used to enhance edges while reducing the noise at the same time.

Code (Filters using Fourier Transformation):

% Loading Image

im=double(imread('lena\_gr.jpg'));

F\_u\_v=fft2(im);

F\_u\_v=(fftshift(F\_u\_v));

subplot(2,3,1); imshow(uint8(im));

title('Original Image');

temp=log(1+abs(F\_u\_v));

subplot(2,3,4); imshow(temp,[]);

title('Fourier Spectra');

% Idle Lowpass filter

% Creating Transfer function

[M,N]=size(im);

% Set up range of variables.

u = 0:(M - 1);

v = 0:(N - 1);

% Compute the indices for use in meshgrid.

idx = find(u > M/2);

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

idy = find(v > N/2);

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

% Compute the meshgrid arrays.

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

% Compute the distances D(U, V).

D0=40;

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

H =ifftshift( double(D <=D0));

%Applying the transfer function

g=real(ifft2(H.\*F\_u\_v));

%Crop to original size.

%g=g(1:size(F\_u\_v,1),1:size(F\_u\_v,2));

subplot(2,3,5); imshow(uint8(abs(g)));

title('Filtered Image (Low Pass)');

subplot(2,3,2); imshow(H,[]);

title('Idle Lowpass Filter');

% Idle Highpass filter

D0=15;

H =ifftshift( double(D <=D0));

Hp=1-H;

% Applying Highpass filter

g=real(ifft2(Hp.\*F\_u\_v));

subplot(2,3,6); imshow(uint8(abs(g)));

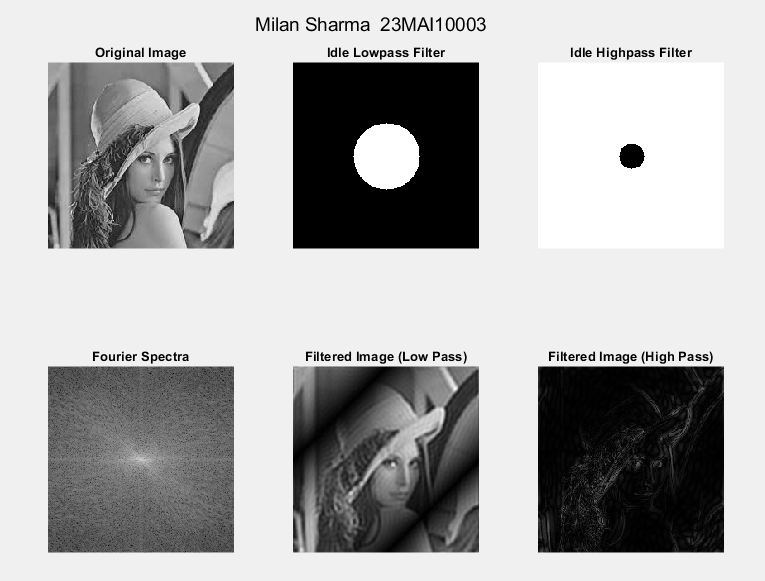
title('Filtered Image (High Pass)');

subplot(2,3,3); imshow(Hp,[]);

title('Idle Highpass Filter');

sgtitle("Milan Sharma 23MAI10003")

Output :



Code (Convolution using fft2 filter) :

%Create a black 30x30 image

f=zeros(30,30);

subplot(2,3,1), imshow(f,'InitialMagnification', 'fit')

% With a white rectangle in it.

f(5:24,13:17)=1;

subplot(2,3,2), imshow(f,'InitialMagnification', 'fit')

%Calculate the DFT.

F=fft2(f);

%F2=abs(F);

subplot(2,3,3), imshow(log(abs(fftshift(F))),[], 'InitialMagnification','fit')

F=fft2(f, 256, 256);

F2=abs(F);

subplot(2,3,4), imshow(F2, [])

%The zero-frequency coefficient is displayed in the

%upper left hand corner. To display it in the center,

%you can use the function fftshift.

F2=fftshift(F);

F2=abs(F2);

subplot(2,3,5),imshow(F2,[])

%In Fourier transforms, high peaks are so high they

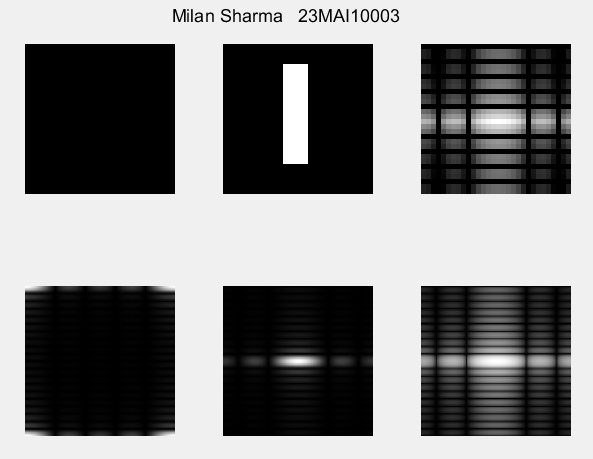
%hide details. Reduce contrast with the log function.

F2=log(1+F2);

subplot(2,3,6),imshow(F2,[])

sgtitle("Milan Sharma 23MAI10003")

Output :



Code (Gaussian Filter using fft2):

f = imread('lena\_gr.jpg');

subplot(1,2,1); imshow(f);

title("Original Image")

f = double(f);

F = fftshift(fft2(f));

[m,n] = size(f);

sig = 10;

H = Gaussian(m, n, sig);

G = H.\*F;

g = abs(ifft2(G));

subplot(1,2,2); imshow(g,[]);

title("Gaussian Filter (sigma = 10)")

sgtitle("Milan Sharma 23MAI10003")

function H = Gaussian(m, n, sigma)

[X, Y] = meshgrid(-(n-1)/2:(n-1)/2, -(m-1)/2:(m-1)/2);

H = exp(-(X.^2 + Y.^2) / (2 \* sigma^2));

H = H / (2 \* pi \* sigma^2); % Normalize the filter

end

Output:



Code ():

f = imread('lena\_gr.jpg');

h = fspecial('sobel');

PQ = paddedsize(size(f));

F = fft2(double(f), PQ(1), PQ(2));

H = fft2(double(h), PQ(1), PQ(2));

F\_fH = H.\*F;

ff1 = ifft2(F\_fH);

ff2 = ff1(2:size(f,1)+1, 2:size(f,2)+1);

%Display results (show all values)

subplot(3,3,1), imshow(f);

subplot(3,3,2), imshow(log(abs(fftshift(F))),[]);

subplot(3,3,3), imshow(log(abs(fftshift(H))),[]);

subplot(3,3,4), imshow(log(abs(fftshift(F\_fH))),[]);

subplot(3,3,5), imshow(ff1,[]);

subplot(3,3,6), imshow(ff2,[]);

%The abs function gets correct magnitude

%when used on complex numbers

ffim = abs(ff2);

subplot(3,3,7), imshow(ffim,[]);

sgtitle("Milan Sharma 23MAI10003")

function PQ = paddedsize(AB, CD, PARAM)

if nargin == 1

PQ = 2\*AB;

elseif nargin == 2 && ~ischar(CD)

PQ = AB + CD - 1;

PQ = 2 \* ceil(PQ / 2);

elseif nargin == 2

m = max(AB); % Maximum dimension.

% Find power-of-2 at least twice m.

P = 2^nextpow2(2\*m);

PQ = [P, P];

elseif nargin == 3

m = max([AB CD]); %Maximum dimension.

P = 2^nextpow2(2\*m);

PQ = [P, P];

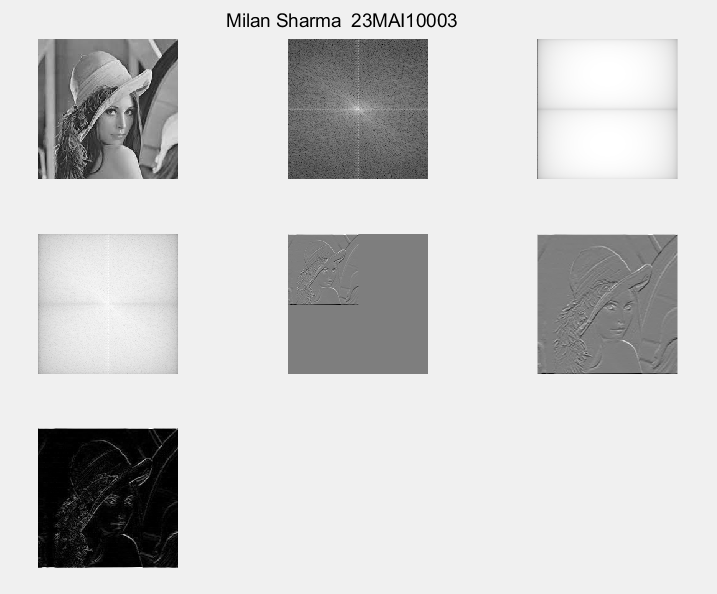
else

error('Wrong number of inputs.')

end

end

Output:



**Learning outcomes (What I have learnt):**

**1. Learnt about spatial and frequency domain.**

**2. Learnt about various filters in spatial and frequency domain.**

**3. Learnt about Sobel filters and Laplace filters and learnt how to use them in spatial and frequency domain.**

**4. Learnt about Corelation and Convolution using different filters and techniques.**

**5. Learnt about Fourier transformation and its properties.**