Experiment-1.2

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

a) Image enhancement/denoising using spatial domain filters.

b) Image enhancement/denoising using frequency domain filters.

1) IMAGE ENHANCEMENT/DENOISING USING SPATIAL DOMAIN FILTERS :

Problem Description :

Spatial filtering refers to the process of manipulating an image's pixel values based on their spatial relationships. Spatial filters are commonly used for tasks such as image enhancement, noise reduction, and feature extraction. Spatial filters operate on the pixel values of an image using a convolution operation. The basic idea is to slide a small matrix, known as a kernel or filter, over the image and perform a mathematical operation at each position. The kernel defines how the values of neighboring pixels contribute to the new value of the current pixel.

There are different types of spatial filters, and they serve various purposes:

1. Smoothing Filters (Low-pass filters): These filters are used to reduce noise and blur an image. Example is Gaussian filter and the averaging filter.

2. Sharpening Filters (High-pass filters): These filters enhance the edges and fine details in an image. Example is Laplacian filter and the Sobel filter.

3. Edge Detection Filters: These filters are designed to highlight edges and boundaries in an image. Example is Sobel, Prewitt, and Roberts operators.

4. Median Filters: Used for noise reduction, especially in removing salt-and-pepper noise. It replaces each pixel value with the median value of its neighboring pixels.

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.

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('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 2");

% 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('Ashish Kumar, 23MAI10008')
```

Course Name: Master of Engineering - AIML Course Code: AI-301

Output :

Code (Histogram):

```
A=imread('lena_gr.jpg');
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));
```

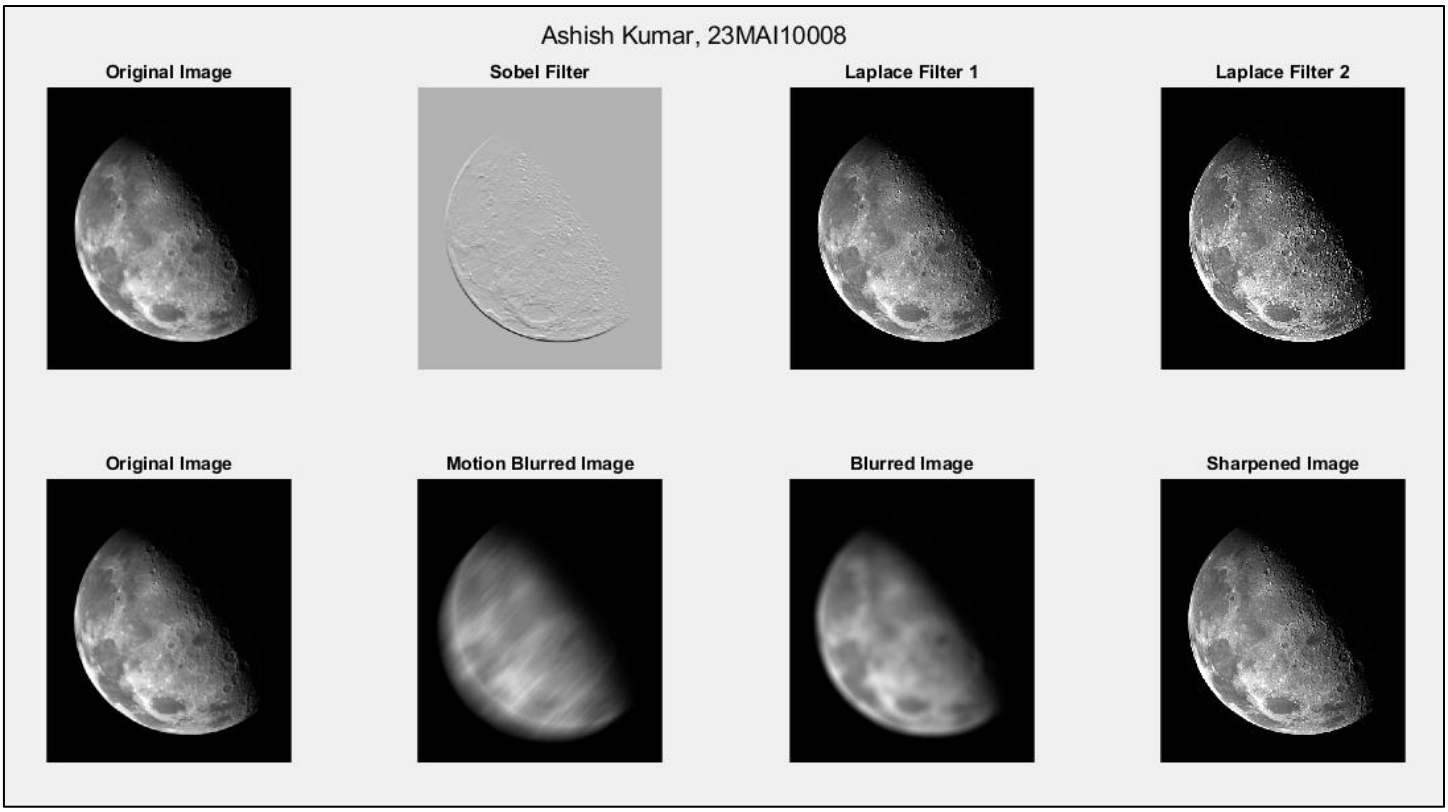
```
intensity = 0;
while( intensity <256 )
       count = 0;
       for i = 1:row2
              for j = 1:column2
                     if Eq_A(i,j) == intensity
                     count = count+1;
                     end
              end
       end
       H_Eq_A(1,intensity+1) = count;
       intensity = 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('Ashish Kumar, 23MAI10008')
```

Output:

Code (Laplacian, Sobel, Blur and Sharp Filter):

```
%load in lunar north pole image
f=imread('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('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('Ashish Kumar, 23MAI10008')
```

Output:



2) IMAGE ENHANCEMENT/DENOISING USING FREQUENCY DOMAIN FILTERS:

Problem Description :

Frequency filtering is a signal processing technique used to manipulate the frequency content of a signal. It involves selectively attenuating or enhancing certain frequency components of a signal while leaving others unchanged.

There are several methods for frequency filtering, including:

1. Low-pass filtering: This allows frequencies below a certain cutoff frequency to pass through while attenuating frequencies above that cutoff. It's commonly used for smoothing or removing high-frequency noise from a signal. Mechanism of low pass filtering is given by:

$$G(u, v) = H(u, v) \cdot F(u, v)$$

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

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

2. High-pass filtering: This allows frequencies above a certain cutoff frequency to pass through while attenuating frequencies below that cutoff. Mechanism of high pass filtering 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

3. Band-pass filtering: This selectively allows a certain range of frequencies (bandwidth) to pass through while attenuating frequencies outside that range. It's useful for isolating specific frequency components of a signal.

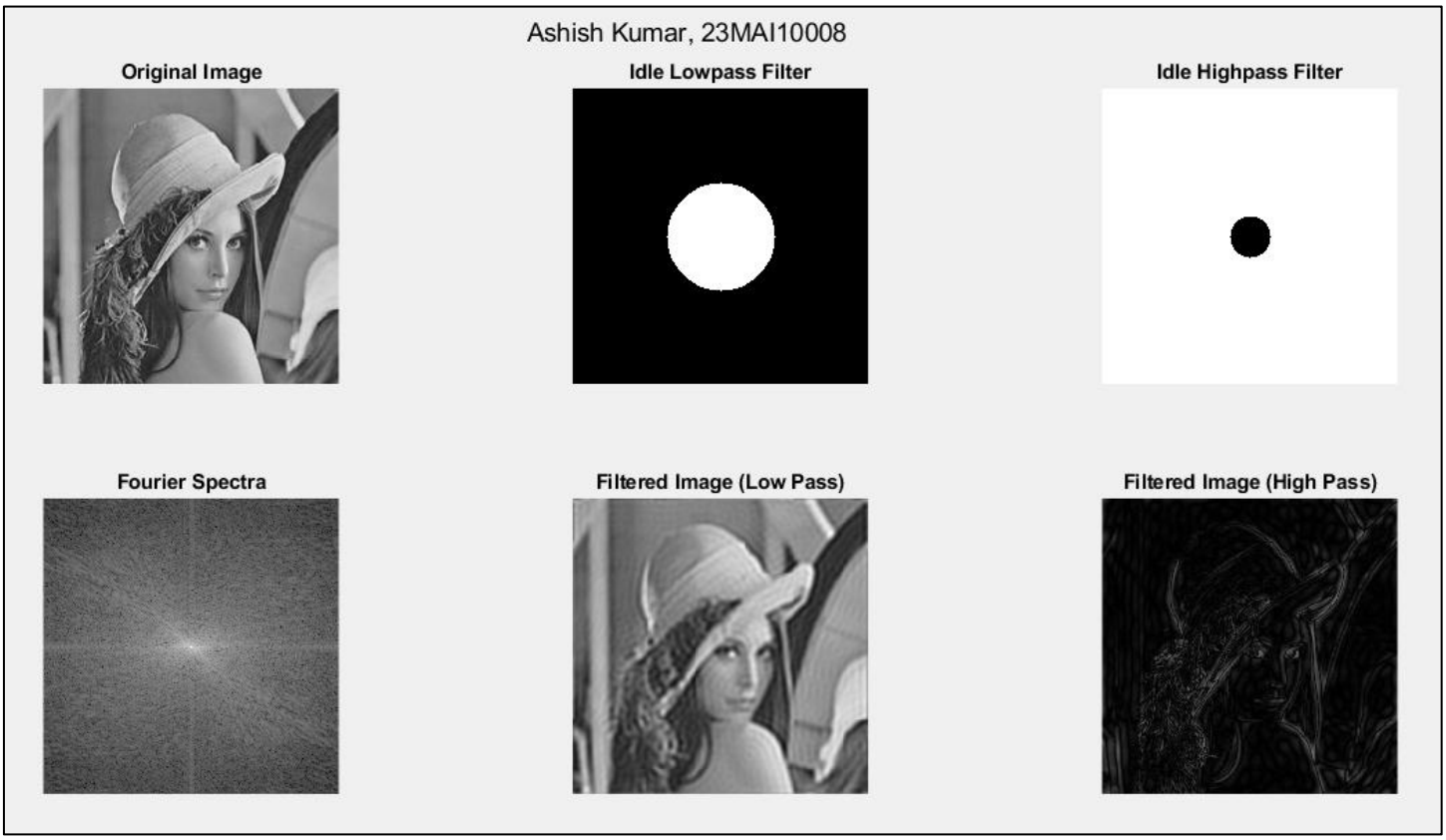
4. Band-stop filtering: This selectively attenuates a certain range of frequencies while allowing frequencies outside that range to pass through. It's commonly used for removing interference or unwanted frequency components from a signal.

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(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("Ashish Kumar, 23MAI10008")
```

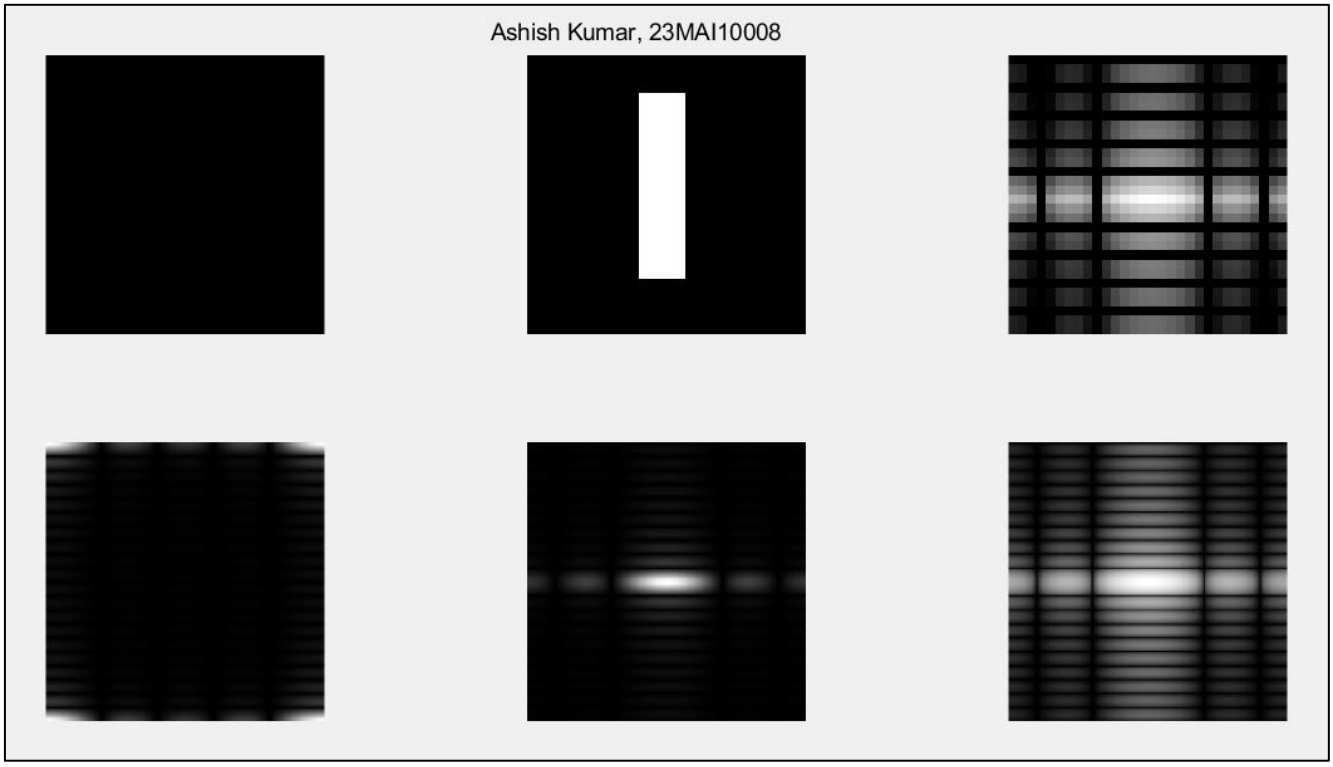
Output :



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, [])
%fftshift display zero-frequency coefficient in center
F2=fftshift(F);
F2=abs(F2);
subplot(2,3,5),imshow(F2,[])

%Reduce contrast with the log function.
F2=log(1+F2);
subplot(2,3,6),imshow(F2,[])
sgtitle("Ashish Kumar, 23MAI10008")
```

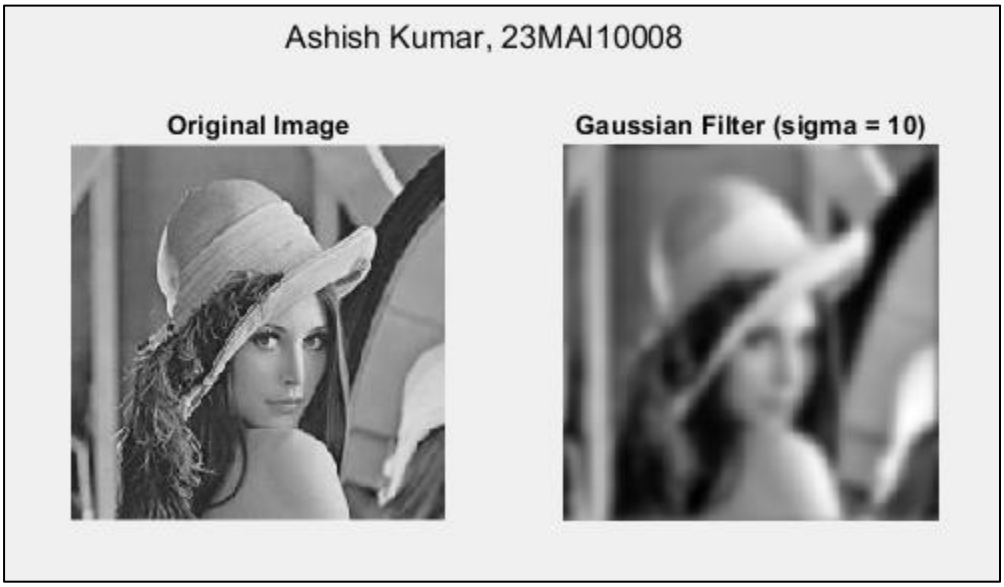
Result:



Code (Gaussian Filter with ifft2):

```
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("Ashish Kumar, 23MAI10008");


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

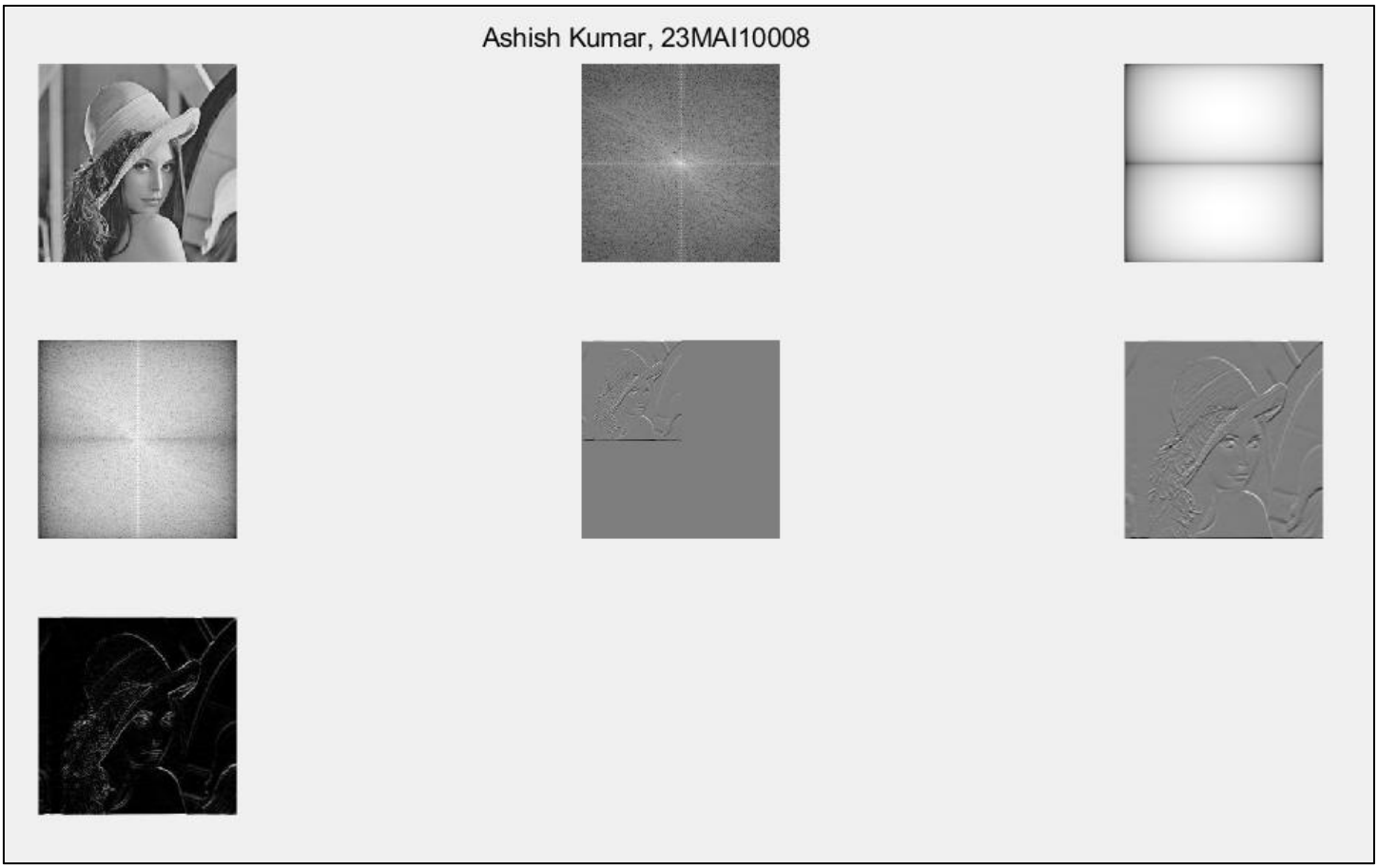
Result:



Code (Sobel Filter):

```
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("Ashish Kumar, 23MAI10008");


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
```

Result:



Learning outcomes (What I have learnt):

1. I learnt about spatial domain and frequency domain filters.
2. I learnt about Fourier Transformation in frequency domain.
3. I learnt about Low Pass and High Pass Filter.
4. I learnt about Histogram Equalization in spatial domain.
5. I learnt about convolution and padding in Image.