# Consider the torn picture with line on it as illustrated in Figure 1. Can we get rid of line?



**Figure 1**

Aim:

Remove the line in the picture.

**Method 1 [Pixel manipulation]**

**Procedure:**

1. Read the image using imread.
2. Convert the image from RGB to Gray scale using rgb2gray.
3. A matrix of 256\*256 dimension is obtained.
4. Every intensity bit is checked through using a 2 variable FOR loop.
5. Every intensity bit less than 60 is replaced by a value of 200.
6. Gaussian filter of order 0.8 is added to the image to achieve equality and smooth transition between the pixels.
7. Image is displayed using imshow.

**Code:**

img = imread('Fig1.jpg');

im = rgb2gray(img);

figure(1)

subplot(1,3,1);

%Original image

imshow(im);

count = 0;

for i = 1:255

for j = 1:255

if im(i,j) < 60

count = count + 1;

im(i,j) = 200;

end

end

end

subplot(1,3,2)

%Manipulating intensity values

imshow(im);

blur = imgaussfilt(im,0.8);

subplot(1,3,3)

%Applying Gaussian filter of order 0.8

imshow(blur)

**Output:**

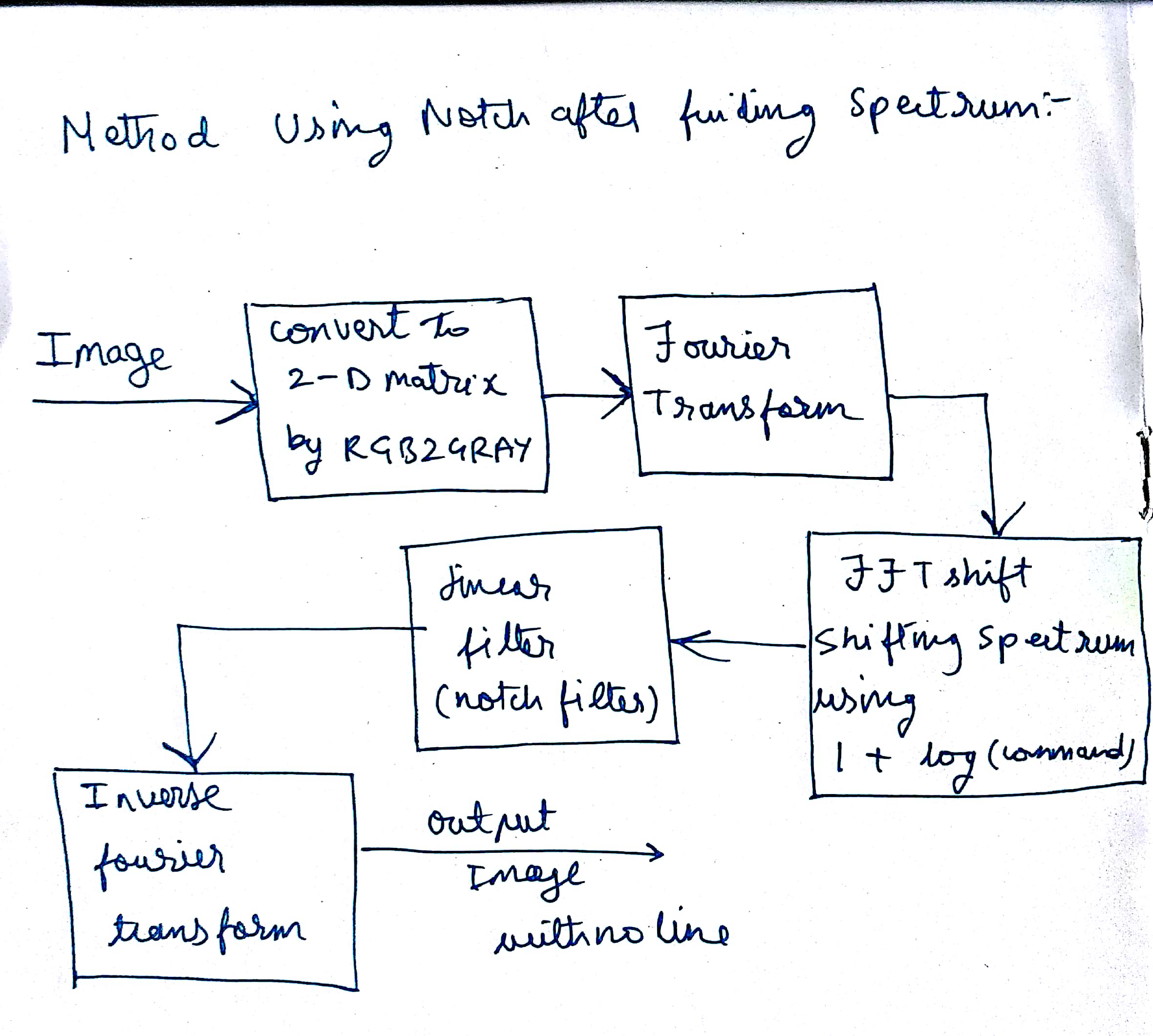
Conclusion:

Though an easy and efficient method, this method leads to some substantial loss of information. It can be avoided with further extensive pixel studying by monitoring the intensity of every pixel individually and not grouping them in chunks of less than 50 or greater than 200. But the latter is a redundant and time taking procedure.

**Method 2 [FFT + Notch Filter]**

**Procedure:**

1. Read the image using imread.
2. Convert the image from RGB to Gray scale using rgb2gray.
3. A matrix of 256\*256 dimension is obtained.
4. Obtain 2D FFT of the image and shift the spectrum to obtain ideal graph.
5. Apply notch filter and filter out only the required frequencies.
6. Obtain the IDFFT of the filtered spectrum.
7. Display the obtained image.



**Code:**

clc

clf

img = imread('Fig1.jpg');

im = rgb2gray(img);

figure(1)

subplot(1,3,1);

%Original image

imshow(im);

ft = fft2(im);

ft = log(1+abs(ft));

ft = fftshift(ft);

figure(2)

subplot(2,1,1)

%DFT of the given Picture

imshow(ft,[]);

%Notch Filter Apply

%IDFT

ift = ifft2(ft);

subplot(2,1,2)

imshow(ift,[])

Output:



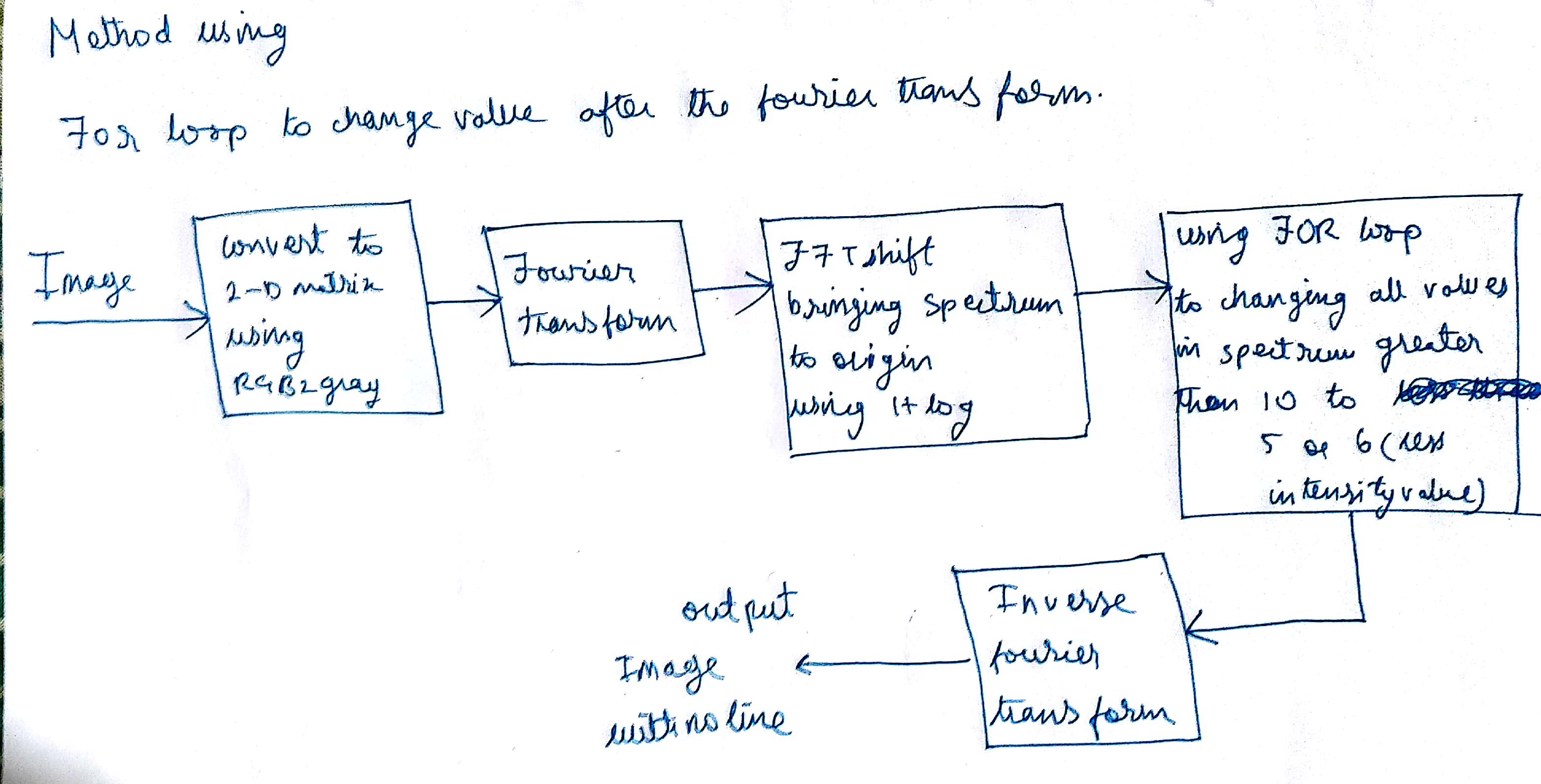
**Problems faced:**

Couldn’t apply the notch filter for a 2D image and the IDFFT keeps giving us a 100% black output image.

**Method 3 [FFT + Spectral manipulation]**

**Procedure:**

1. Read the image using imread.
2. Convert the image from RGB to Gray scale using rgb2gray.
3. A matrix of 256\*256 dimension is obtained.
4. Obtain 2D FFT of the image and shift the spectrum to obtain ideal graph.
5. Manipulate the Frequency. The Black line intensity pixels have lesser frequency compared to the other intensity pixels. So we run a 2 variable FOR loop and manipulate each frequency greater 10 to equal to 7.
6. Obtain the IDFFT of the filtered spectrum.
7. Display the obtained image.



**Code:**

img = imread('Fig1.jpg');

im = rgb2gray(img);

figure(1)

subplot(1,3,1);

%Original image

imshow(im);

ft = fft2(im);

ft = log(1+abs(ft));

ft = fftshift(ft);

figure(2)

subplot(2,1,1)

%DFT of the given Picture

imshow(ft,[]);

%Filter

for i=1:256

for j=1:256

if ft(i,j)>10.0

ft(i,j)=7;

end

end

end

subplot(1,3,2)

imshow(ft,[])

%IDFT

ift = ifft2(ft);

subplot(1,3,3)

imshow(ift,[])

**Output:**

****

**Problems faced:**

1. **We obtain a noise in the middle of the filtered spectrum.**
2. **IDFFT still gives us a 100% black output though the workspace shows us different intensities.**