**Practical – 1**

**Aim:** Read, Display and Analyse different type of image in MATLAB and implement and analyse effect of below given spatial domain techniques. (Without using inbuilt function)

1. Image zooming
2. Image shrinking
3. Change Gray level resolution
4. Image negation
5. Log transformation
6. Power low transformation
7. Gray level slicing (two approach)
8. Plot image histogram
9. Histogram equalization
10. **Aim:** Image zooming

**Code:**

clc

clear all

% read input image using imread() function

img = imread('Images\Jaydip.jpg');

% convert rgb image to gray scale image

img = rgb2gray(img);

%find number of rows and column of matrix of image

[row,col] = size(img);

% Take Zooming factor from user

Z = input('Enter the Zooming Factor = ');

% Apply image zooming using image replication method

for i=1:row

for j = 1:col

for k = 1:Z

outImg((i-1)\*Z+k,(j-1)\*Z+k) = img(i,j);

end

end

end

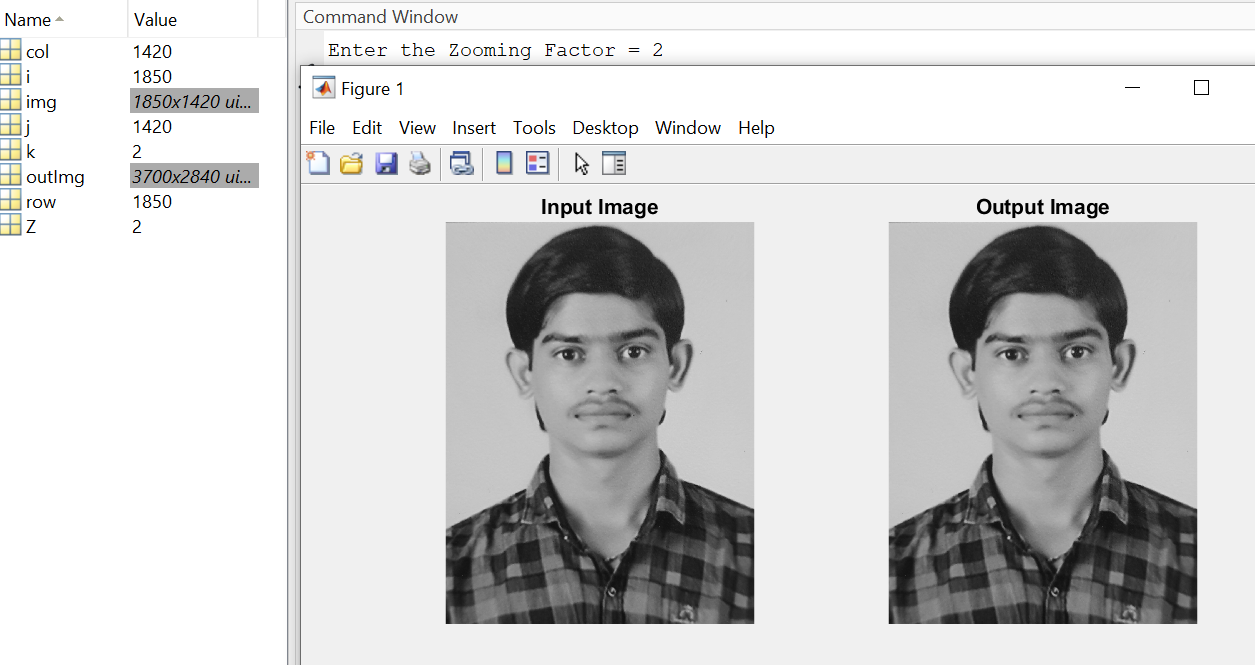
%Display input and output images in plot

subplot(1,2,1);imshow(img);title('Input Image')

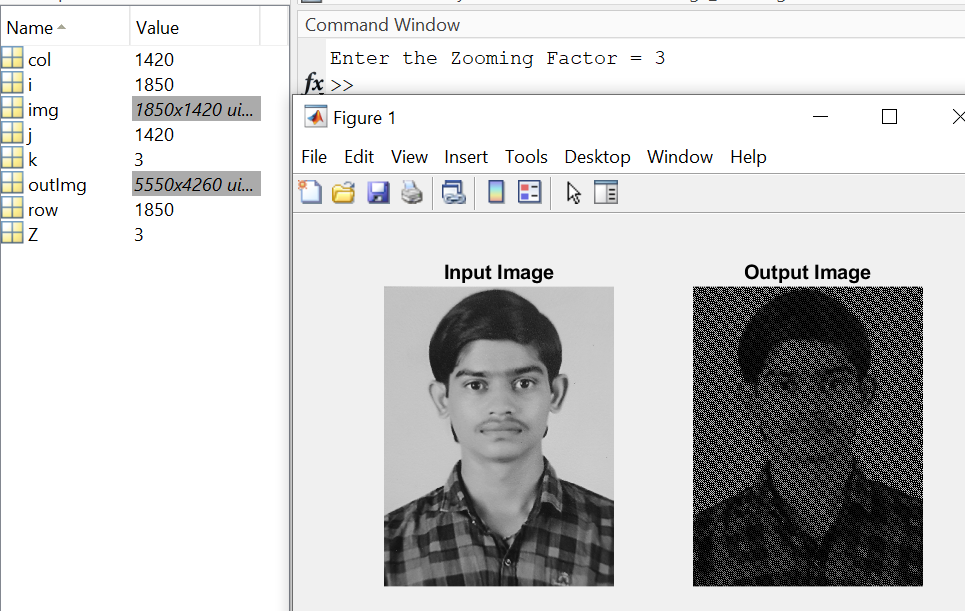
subplot(1,2,2);imshow(outImg);title('Output Image')

**Input - Output:**

* For zooming factor = 2 you can see changes in dimensions also.

****

* For zooming factor = 3

****

**Observation:**

Here we can see that according the zooming factor image dimension is multiplied by the zooming factor. Means we add the rows and columns of image in image zooming.

1. **Aim:** Image shrinking

**Code:**

clc

clear all

% read input image using imread() function

img = imread('Images\Jaydip.jpg');

% convert rgb image to gray scale image

img = rgb2gray(img);

%find number of rows and column of matrix of image

[row,col] = size(img);

% Take shrinking factor from user

Z = input('Enter the Shrinking Factor = ');

% first we zoom image

for i=1:row

for j = 1:col

for k = 1:Z

zoomedImg((i-1)\*Z+k,(j-1)\*Z+k) = img(i,j);

end

end

end

% Now we shrink the zoomed image

for i=1:row

for j = 1:col

for k = 1:Z

shrinkedImg(i,j) = zoomedImg((i-1)\*Z+k,(j-1)\*Z+k);

end

end

end

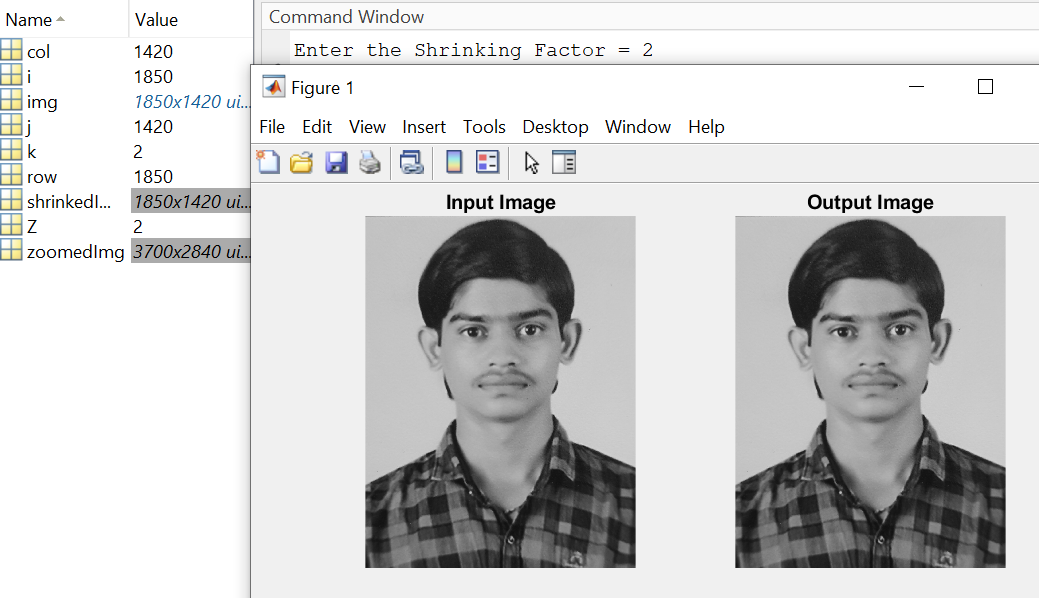
%Display input and output images in plot

subplot(1,2,1);imshow(zoomedImg);title('Input Image')

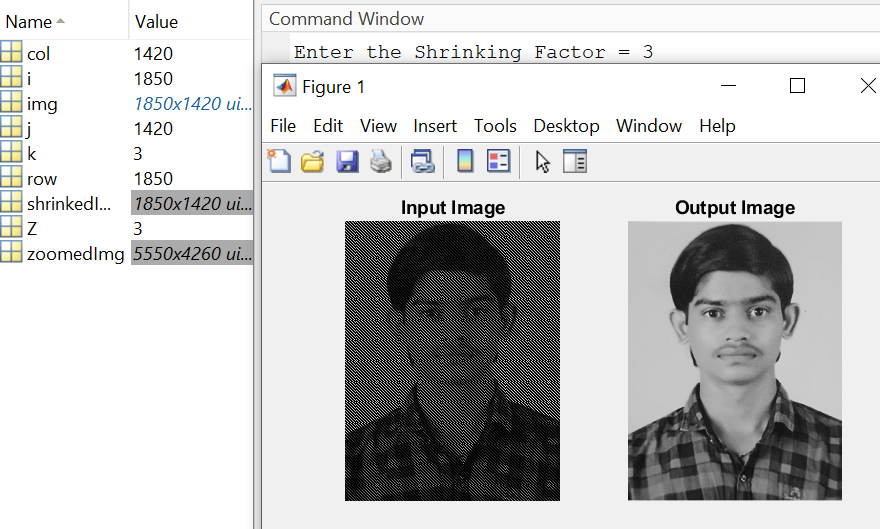
subplot(1,2,2);imshow(shrinkedImg);title('Output Image')

**Input - Output:**

* For shrinking factor = 2 you can see changes in dimensions also.



* For shrinking factor = 3

****

**Observation:**

Here we can see that according the shrinking factor image dimension is divided by the shrinking factor. Means we delete the rows and columns of image in image shrinking.

1. **Aim:** Change Gray level resolution

**Code:**

clc

clear all

% read input image using imread() function

inImg = imread('Images\Jaydip.jpg');

%convert image int to gray scale

grayImg = rgb2gray(inImg);

%Get row,col and plane number

[row,col,plane] = size(grayImg);

max = 0;

bit = [ 1 2 3 4 ]; %Divide 8 bit image by different levels

% traverse loop to get maximum pixel intensity of image

for x=1:row

for y=1:col

if inImg(x,y) > max

max = inImg(x,y);

end

end

end

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

figure

for i = 1:length(bit)

d = 2^i; %Total number of bits equal to 2^i

outImg = round(grayImg/d); %Divide each pixel by 2^i;

subplot(2,3,i+1);

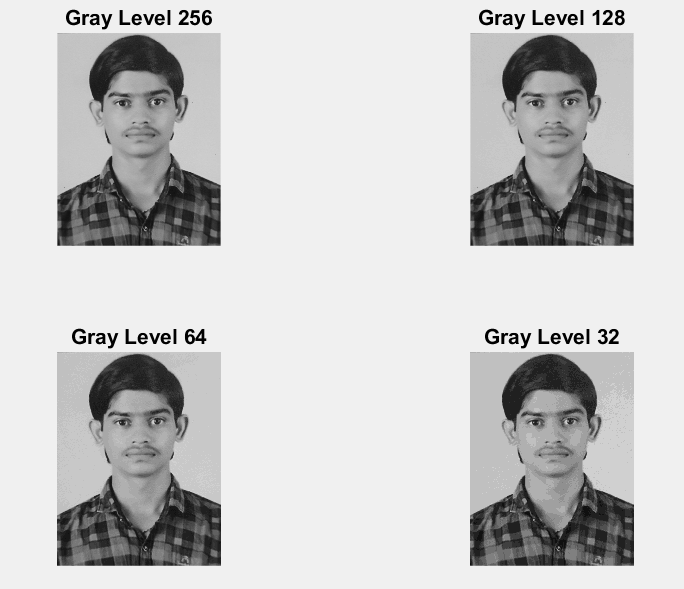
title(['Gray Level ',num2str(256/(2^(i-1)))]);

hold on

imshow(outImg \* d);

end

**Input - Output:**

****

**Observation:**

Here we can clearly see that the intensity values of pixels are changed according it’s Gray level. We divide the intensity values of pixels to reduces the Gray level.

1. **Aim:** Image negation

**Code:**

clc

clear all

% read input image using imread() function

Img = imread('Images\e7.tif');

%find number of rows and column of matrix of image

[row,col] = size(Img);

max = 0;

% s = (L-1)-r

% Here s = output image

% L-1 = Maximum pixel intensity of image

% r = input image

% traverse loop to get maximum pixel intensity of image

for x=1:row

for y=1:col

if Img(x,y) > max

max = Img(x,y);

end

end

end

% Apply formula

% here, r = Img and L-1 = max

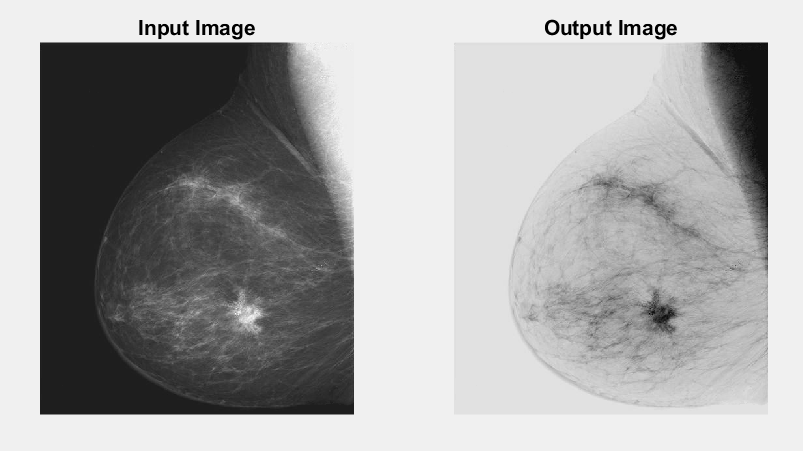
s = max - Img;

%Display input and output images in plot

subplot(1,2,1);imshow(Img);title('Input Image')

subplot(1,2,2);imshow(s);title('Input Image')

**Input - Output:**

****

**Observation:**

Here we can clearly see that after applying negative transformation in image all the dark pixel become white and vies-versa.

1. **Aim:** Log transformation

**Code:**

clc

clear all

% read input image using imread() function

inImg = imread('Images\e18.tif');

% s = c\*log(1+r)

% Here s = output image

% c = constant value

% r = input image

%Apply formula for log transformation

% For 8 bit image we have to multiplied with 256

s1 = (1\*log(1+double(inImg)))\*256;

s2 = (2\*log(1+double(inImg)))\*256;

s3 = (3\*log(1+double(inImg)))\*256;

%Display input and output images in plot

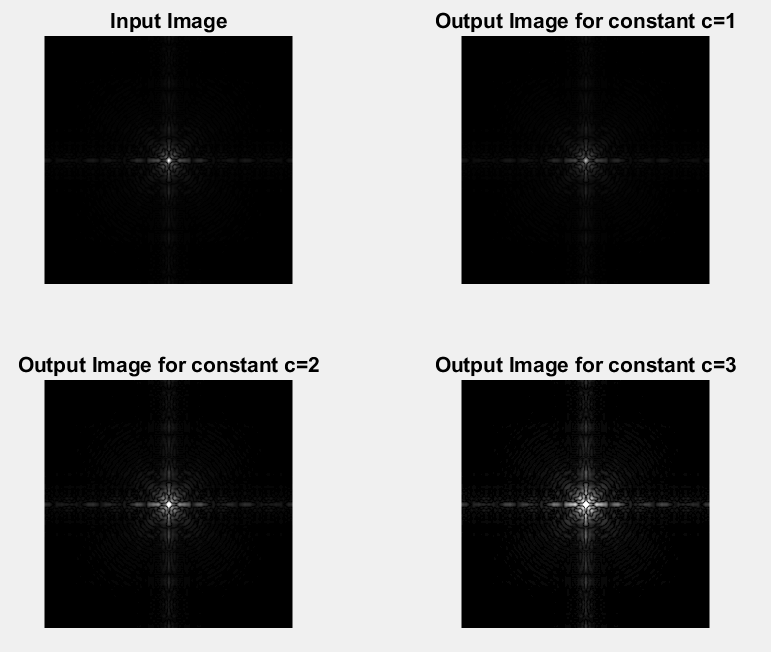
subplot(2,2,1);imshow(inImg);title('Input Image')

subplot(2,2,2);imshow(uint8(s1));title('Output Image for constant c=1')

subplot(2,2,3);imshow(uint8(s2));title('Output Image for constant c=2')

subplot(2,2,4);imshow(uint8(s3));title('Output Image for constant c=3')

**Input - Output:**



**Observation:**

Here we can observe that after applying log transformation on image the darker pixel gets brighter whereas the brighter pixels don’t change much.

1. **Aim:** Power low transformation

**Code:**

clc

clear all

% read input image using imread() function

inImg = imread('Images\e9.tif');

% s = c\*r^G

% Here s = output image

% G = gamma value

% r = input image

%For c=1

%Apply formula for power law transformation

s1 = double(inImg).^1.1;

s2 = double(inImg).^1;

s3 = double(inImg).^0.9;

%Display input and output images in plot

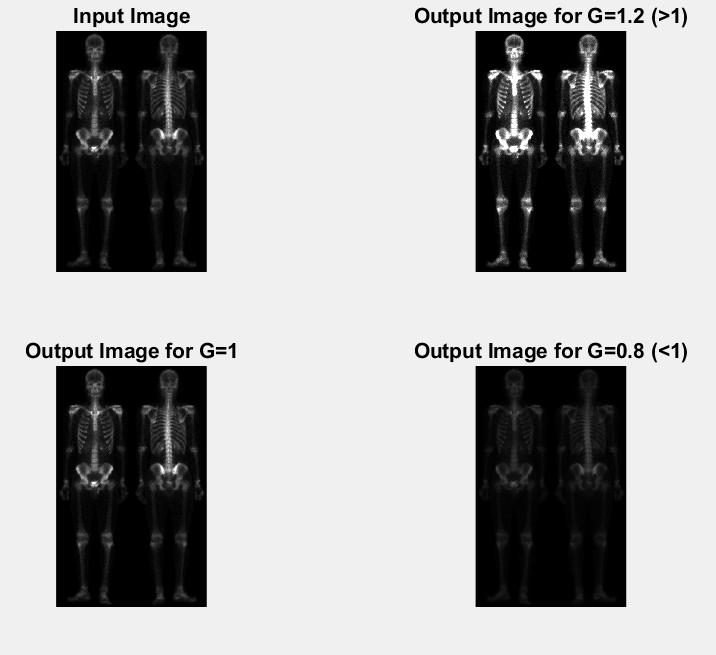
subplot(2,2,1);imshow(inImg);title('Input Image')

subplot(2,2,2);imshow(uint8(s1));title('Output Image for G=1.1 (>1)')

subplot(2,2,3);imshow(uint8(s2));title('Output Image for G=1')

subplot(2,2,4);imshow(uint8(s3));title('Output Image for G=0.9 (<1)')

**Input - Output:**



**Observation:**

Here you can see that the for G >1 you can see the dark pixel become brighter and for G < 1 you can clearly see that the bright pixel become darker. And for G = 1 is image is same as input image.

1. **Aim:** Gray level slicing (two approach)

**Code:**

clc

clear all

% read input iname usin imread() function

inImg = imread('Images\Jaydip.jpg');

inImg=rgb2gray(inImg);

subplot(2,2,1);imshow(inImg);title('Input Image')

[x,y]=size(inImg);

r1=input('enter value r1: ');

r2=input('enter value r2: ');

for i=1:1:x

for j=1:1:y

if inImg(i,j)>=r1 && inImg(i,j)<=r2

outImg1(i,j)=255;

outImg2(i,j)=255;

else

outImg1(i,j)=inImg(i,j);

outImg2(i,j)=0;

end

end

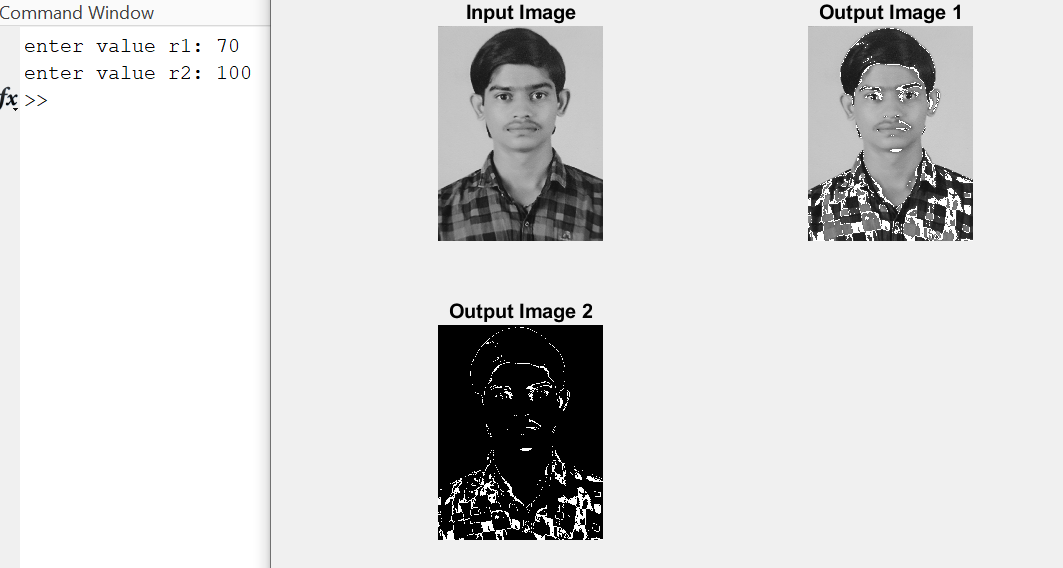
end

% s1=uint8(s);

subplot(2,2,2);imshow(outImg1);title('Output Image 1')

subplot(2,2,3);imshow(outImg2);title('Output Image 2')

**Input - Output:**

****

**Observation:**

Here we can see that using Gray level slicing we can see the image like x-rays images or CT scan images.

1. **Aim:** Plot image histogram

**Code:**

clc

clear all

% read input image using imread() function

inImg = imread('Images\Jaydip.jpg');

[rows, columns, numberOfColorChannels] = size(inImg);

counts = zeros(1, 256);

for col = 1 : columns

for row = 1 : rows

% Get the gray level.

grayLevel = inImg(row, col);

% Add 1 because graylevel zero goes into index 1 and so on.

counts(grayLevel+ 1) = counts(grayLevel+1) + 1;

end

end

% Plot the histogram.

subplot(2,2,1);imshow(inImg);title('Input Image')

grayLevels = 0 : 255;

bar(grayLevels, counts, 'BarWidth', 1, 'FaceColor', 'b');

xlabel('Gray Level', 'FontSize', 20);

ylabel('Pixel Count', 'FontSize', 20);

title('Histogram', 'FontSize', 20);

grid on;

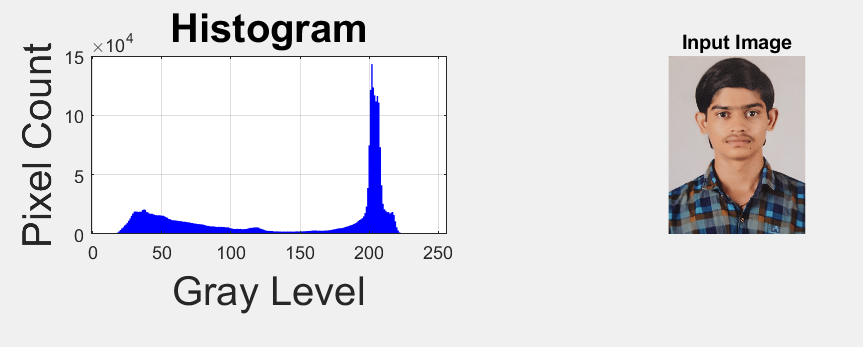
subplot(2,2,2)

imshow(inImg)

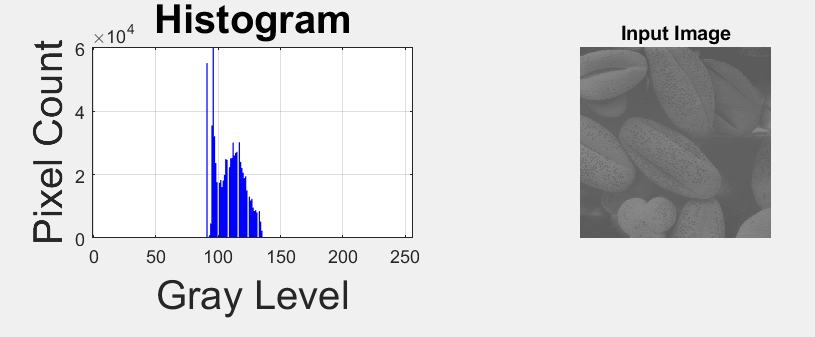
title('Input Image')

**Input - Output:**

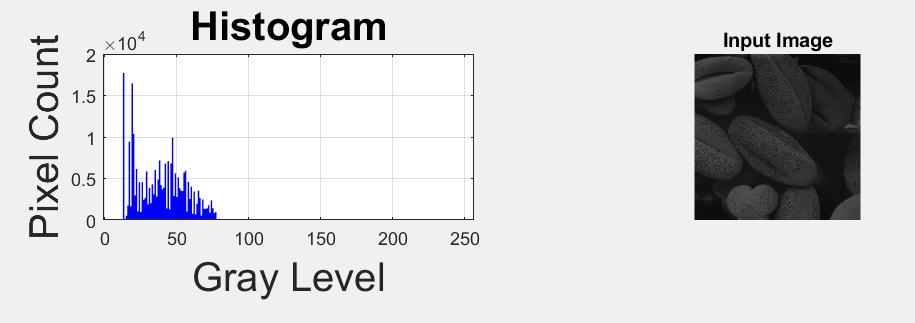
1. **My Passport Size Photo:**

****

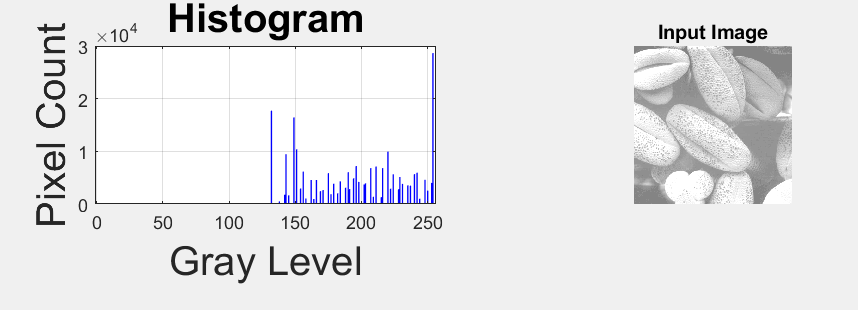
1. **Image no e10:**

****

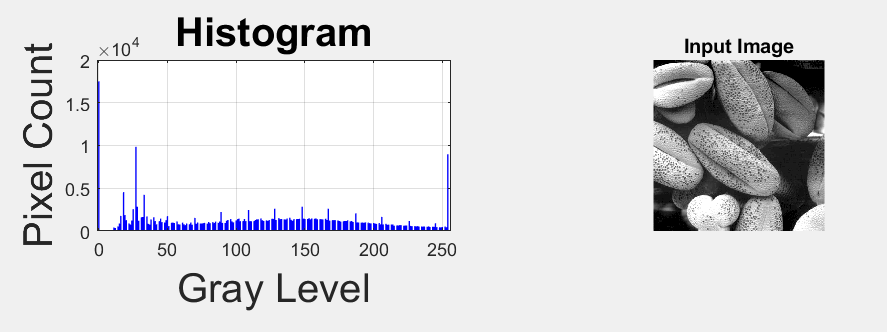
1. **Image no e11:**

****

1. **Image no e12:**

****

1. **Image no e13:**

****

**Observation:**

Through Plot image histogram we can see whether the image is high contrast image, dark image, bright image, or low contrast image.

1. **Aim:** Histogram equalization

**Code:**

clc

clear all

% read input image using imread() function

inImg = imread('Images\e12.tif');

% histogram equalization function

outImg = histeq(inImg);

%Display input image, its histogram, equalize histogram and output image

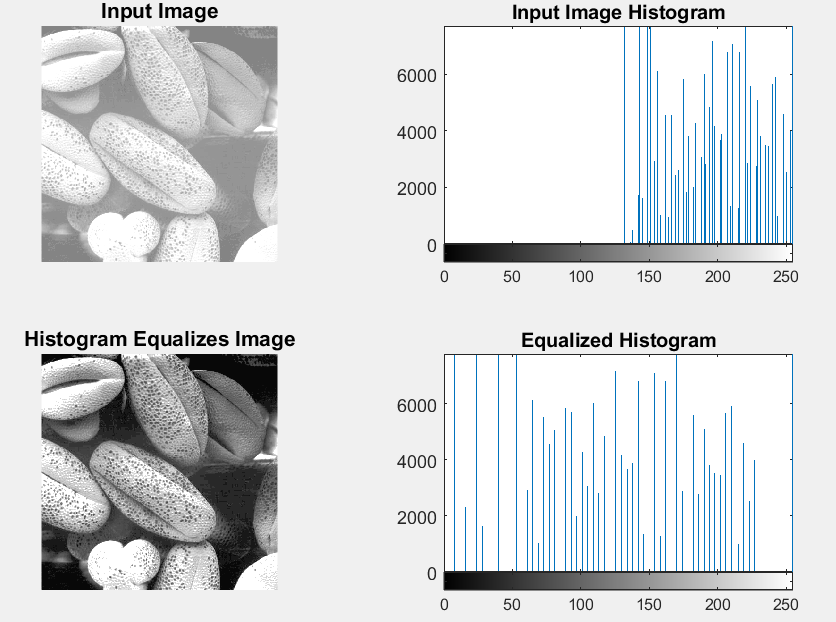
subplot(2,2,1);imshow(inImg);title('Input Image')

subplot(2,2,2);imhist(inImg );title('Input Image Histogram')

subplot(2,2,3);imshow(outImg );title('Histogram Equalizes Image')

subplot(2,2,4);imhist(outImg );title('Equalized Histogram')

**Input - Output:**

****

**Observation:**

Here we can see that the input brighter image becomes high contrast image after applying histogram equalization on image.

**Contrast starching:**

**X = imread('image.jpg');**%reading a grayscale image **figure(1);  
imshow(X);  
title('Original Image')  
a = min(X(:));**  %minimum pixel of image X **b = max(X(:));**%maximum pixel of imageX **X= (X-a).\*(255/(b-a));**%just using the formula above  
**figure(2);  
imshow(X);  
title('Contrast Streached Image')**

**Histogram equalization:**

close all;

clear all;

clc

GIm=imread('e:\p.hd works\test\_image\sailboat.bmp');

numofpixels=size(GIm,1)\*size(GIm,2);

figure,imshow(GIm);

title('Original Image');

HIm=uint8(zeros(size(GIm,1),size(GIm,2)));

freq=zeros(256,1);

probf=zeros(256,1);

probc=zeros(256,1);

cum=zeros(256,1);

output=zeros(256,1);

n=1:256;

%freq counts the occurrence of each pixel value.

%The probability of each occurrence is calculated by probf.

for i=1:size(GIm,1)

for j=1:size(GIm,2)

value=GIm(i,j);

freq(value+1)=freq(value+1)+1;

probf(value+1)=freq(value+1)/numofpixels;

end

end

figure,stem(n,probf)

title('Probability Distribution Function')

sum=0;

no\_bins=255;

%The cumulative distribution probability is calculated.

for i=1:size(probf)

sum=sum+freq(i);

cum(i)=sum;

probc(i)=cum(i)/numofpixels;

output(i)=round(probc(i)\*no\_bins);

end

cum

figure,stem(n,output)

for i=1:size(GIm,1)

for j=1:size(GIm,2)

HIm(i,j)=output(GIm(i,j)+1);

end

end

figure,imshow(HIm);

title('Histogram equalization');