Digital Image Processing Lab

Name - Abhishek Maheshwari

Section - E

Roll No - 13

University Roll No - 191500030

Submitted To - Pooja Mam

Enhancement using Arithmetic and Logic Operation

CODE:

```
import cv2

import numpy as np

# Load original image

img = cv2.imread('D:/downloads/forest.jpg')

# Create list to store noisy images

images = []

# Generate noisy images using cv2.randn. Can use your own mean and std.

for _ in range(20):

    img1 = img.copy()

    cv2.randn(img1,(0,0,0),(50,50,50))

    images.append(img+img1)

# For averaging, create an empty array, then add images to this array.

img_avg=np.zeros((img.shape[0],img.shape[1],img.shape[2]),np.float32)

for im in images:

    img_avg=img_avg+im/20

# Round the float values. Always specify the dtype

img_avg=np.array(np.round(img_avg),dtype=np.uint8)

# Display the images

cv2.imshow('average_image',img_avg)

cv2.imshow('original_image',img)

cv2.imshow('noise_image',images[1])
```

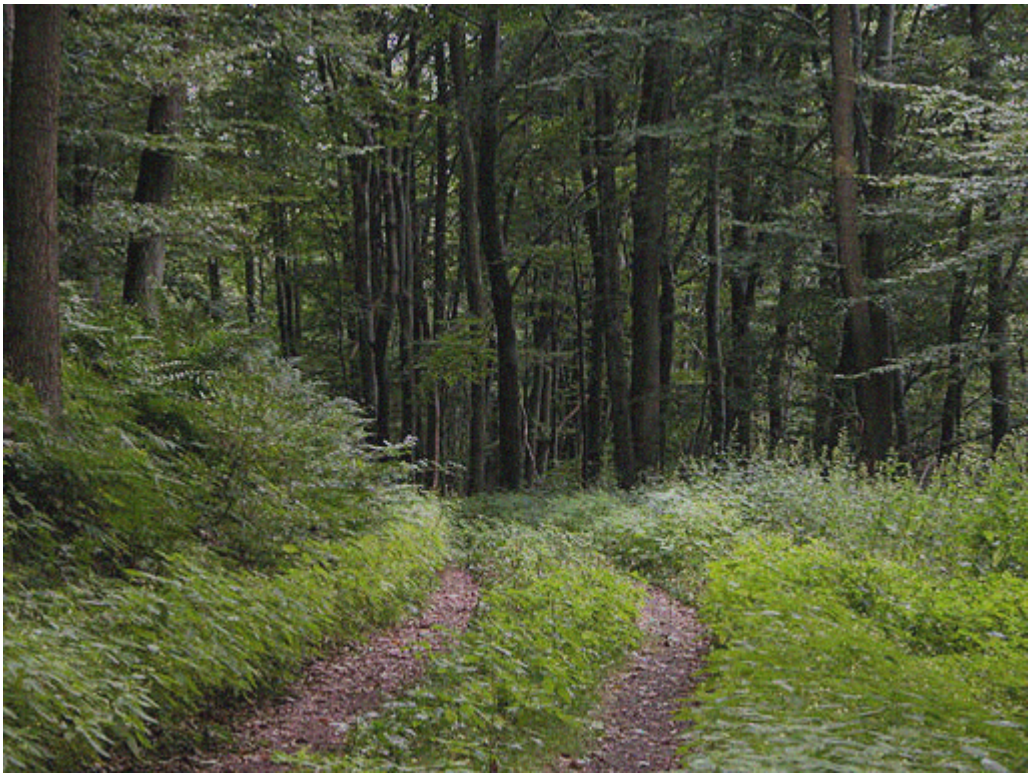
```
cv2.waitKey(0)
```

```
e',images[1])
cv2.waitKey(0)
```

OUTPUT:

Noisy

Averaged



Digital Image Processing Lab

Name - Abhishek Maheshwari

Section - E

Roll No - 13

University Roll No - 191500030

Submitted To - Pooja Mam

Write a MATLAB code to perform the following grey level transformation and display original image and resultant image.

a. Identity image
b. Image negative
c. Log transformation
d. Power law transformation

CODE:

```
clear all;
close all;
clc;
a=imread('cameraman.tif');
```

```
for i=1:256
    for j=1:256
             t(i,j)=a(i,j);
    end
end

for i=1:256
    for j=1:256
             n(i,j)=255-a(i,j);
    end
end

d=im2double(a);
l=d;
for i=1:256
    for j=1:256
             l(i,j)=log10(1+d(i,j));
    end
end

for i=1:256
    for j=1:256
             p(i,j)=power(a(i,j),2);
    end
end
subplot(2,3,1);
imshow(a);
title('image of cameramen','color','r');
subplot(2,3,2);
imshow(t);
title('image after identity transformation','color','m');
subplot(2,3,3);
imshow(n);
title('image after negative transformation','color','r');
subplot(2,3,4);
imshow(l);
title('image after log transformation','color','m');
subplot(2,3,5);
imshow(p);
title('image after power law transformation','color','m');
```

OUTPUT:

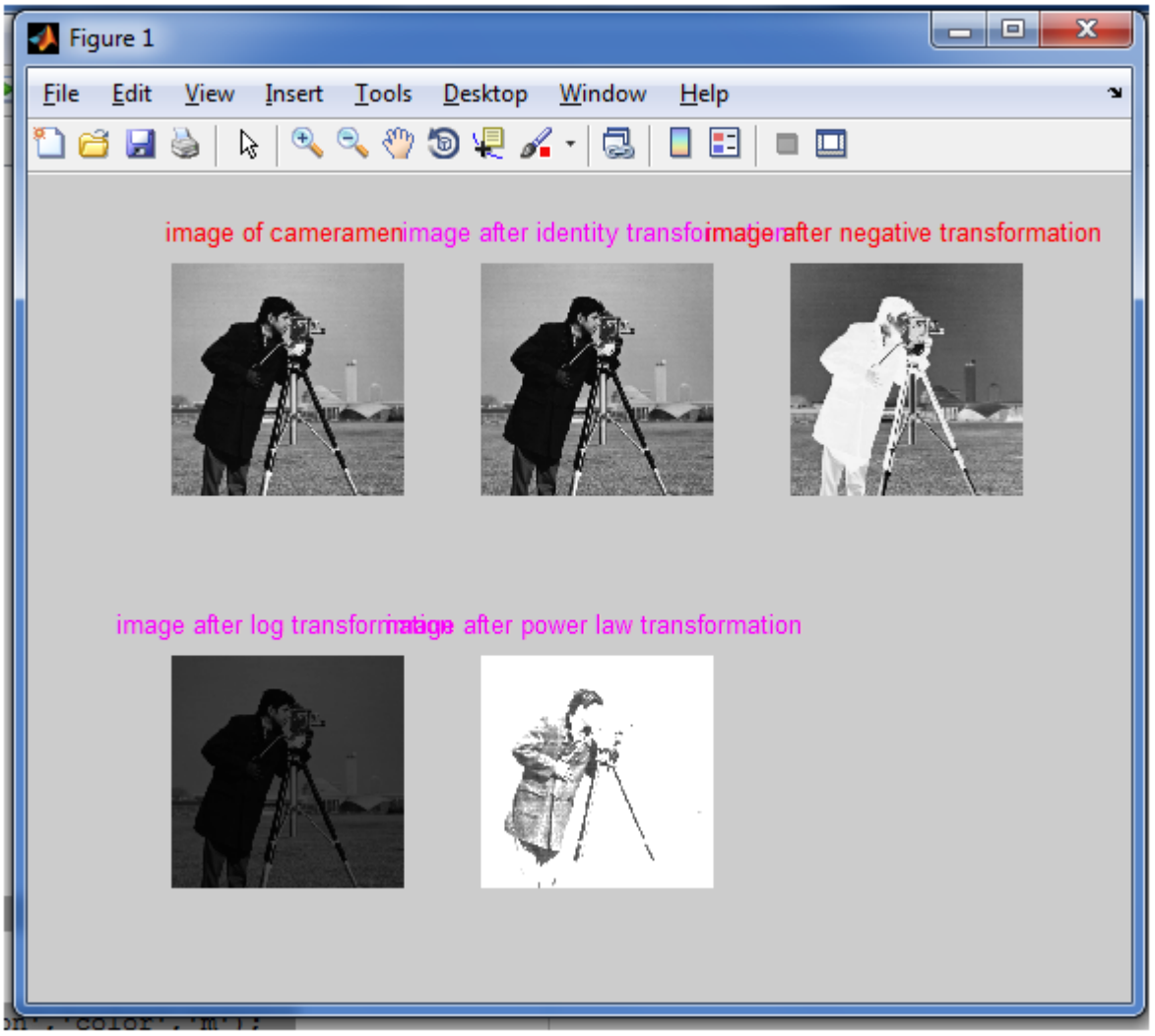


Figure 1
File Edit View Insert Tools Desktop Window Help
image of cameramen
image after identity transfo
image after negative transformation
image after log transformation
image after power law transformation

Digital Image Processing Lab

Name - Abhishek Maheshwari

Section - E

Roll No - 13

University Roll No - 191500030

Submitted To - Pooja Mam

LAB 1

```
>> 4+5^2
ans =
 29

>> 1+12+123
ans =
 136

>> [4:32:2:6]
ans =
 1×0 empty double row vector

>> [2:3:4:5]
ans =
 2 3 4 5

>> [3,6,9,12]
ans =
 3 6 9 12

>> 4:12
ans =
 Columns 1 through 8
```

```
4 5 6 7 8 9 10 11
Column 9
12

>> 2:6
ans =
2 3 4 5 6

>> 2:6+1
ans =
2 3 4 5 6 7

>> 1:3:9
ans =
1 4 7

>> 5:-2:1
ans =
5 3 1

>> [1 2 3;6 7 8;4 5 6]
ans =
1 2 3
6 7 8
4 5 6

>> [[65;43;23] [45;56;7]]
ans =
65 45
43 56
23 7

>> [[12;43;54] [18;22;73]]
ans =
12 18
43 22
54 73

>> [[1 2;4 3;5 4] [18;22;73]]
ans =
1 2 18
4 3 22
5 4 73

>> zeros
ans =
0

>> ones
ans =
1

>> eye
ans =
1

>> rand
ans =
0.8147
```

```
>> [rand rand rand;rand rand rand; rand rand rand]
ans =
 0.9058 0.1270 0.9134
 0.6324 0.0975 0.2785
 0.5469 0.9575 0.9649

>> x=[1 2 3;4 5 6;7 8 9]

x =

 1 2 3
 4 5 6
 7 8 9
>> diag(x)
ans =
 1
 5
 9

>> diag(x,1)
ans =
 2
 6

>> size(x)
ans =
 3 3

>> length(x)
ans =
 3

>> numel(x)
ans =
 9

>> det(x)
ans =
 -9.5162e-16

>> cond(x)
ans =
 5.0523e+16

>> inv(x)
Warning: Matrix is close to singular or badly
scaled. Results may be inaccurate. RCOND
= 2.202823e-18.

ans =
 1.0e+16 *

 0.3153 -0.6305 0.3153
 -0.6305 1.2610 -0.6305
 0.3153 -0.6305 0.3153

>> y=[1 0 ;3 6 ;3 7]

y =
```

```
 1 0
 3 6
 3 7
>> inv(y)
Error using inv
Matrix must be square.

>> eig(x)
ans =
 16.1168
 -1.1168
 -0.0000

>> [v,d]=eig(x)

v =

   -0.2320   -0.7858   0.4082
-0.5253    -0.0868    -0.8165
-0.8187 0.6123 0.4082


d =

   16.1168    0    0       0
-1.1168 0  0 0 -0.0000

>> inf
ans =
 Inf

>> inf-inf
ans =
 NaN

>> 1-inf
ans =
 -Inf

>> inf + nan
ans =
 NaN

>> x

x =

 1 2 3
 4 5 6
 7 8 9
>> find(x,5)
ans =
 1
 2
 3
 4
 5

>> x(2,3)
```

```
ans =
 6

>> x(:,2)
ans =
 2
 5
 8

>> x(2,:)
ans =
 4 5 6

>> x(:,:)
ans =
 1 2 3
 4 5 6
 7 8 9

>> x([2,3,5])
ans =
 4 7 5

>> x([1,3,5])
ans =
 1 7 5

>> x([1,3],2:end)
ans =
 2 3
 8 9

>> y=[rand rand rand;rand rand rand;rand rand rand]

y =

 0.1576 0.9706 0.9572
 0.4854 0.8003 0.1419
 0.4218 0.9157 0.7922
>> C = x.*y
C =
   0.1576    1.9412    2.8715
1.9415     4.0014     0.8513
2.9523 7.3259 7.1299

>> C = x./y
C =
 6.3447 2.0606 3.1342
8.2410 6.2478 42.2874
16.5971 8.7361 11.3607

>> c = x.^3
c =
 1 8 27
 64 125 216
 343 512 729

>> transpose(x)
ans =
 1 4 7
```

```
 2 5 8
 3 6 9

>> abs(x)
ans =
 1 2 3
 4 5 6
 7 8 9

>> n=-6
n =
 -6

>> sign(n)
ans =
 -1

Did you mean:
>> asin(45)
ans =
 1.5708 - 4.4997i

>> acos(45)
ans =
 0.0000 + 4.4997i

>> sin(45)
ans =
 0.8509
>> cos(45)
ans =
 0.5253

>> exp(56)
ans =
 2.0917e+24

>> log(10)
ans =
 2.3026

>> ceil(10.11)
ans =
 11

>> floor(10.11)
ans =
 10

>> round(11.56)
ans =
 12

>> real(23)
ans =
 23

>> real(x)ans = 1 2 3  4 5 6
 7 8 9
```

```
>> imag(x)
ans =
 0 0 0
 0 0 0
 0 0 0

>> sort(x)
ans =
 1 2 3
 4 5 6
 7 8 9
```

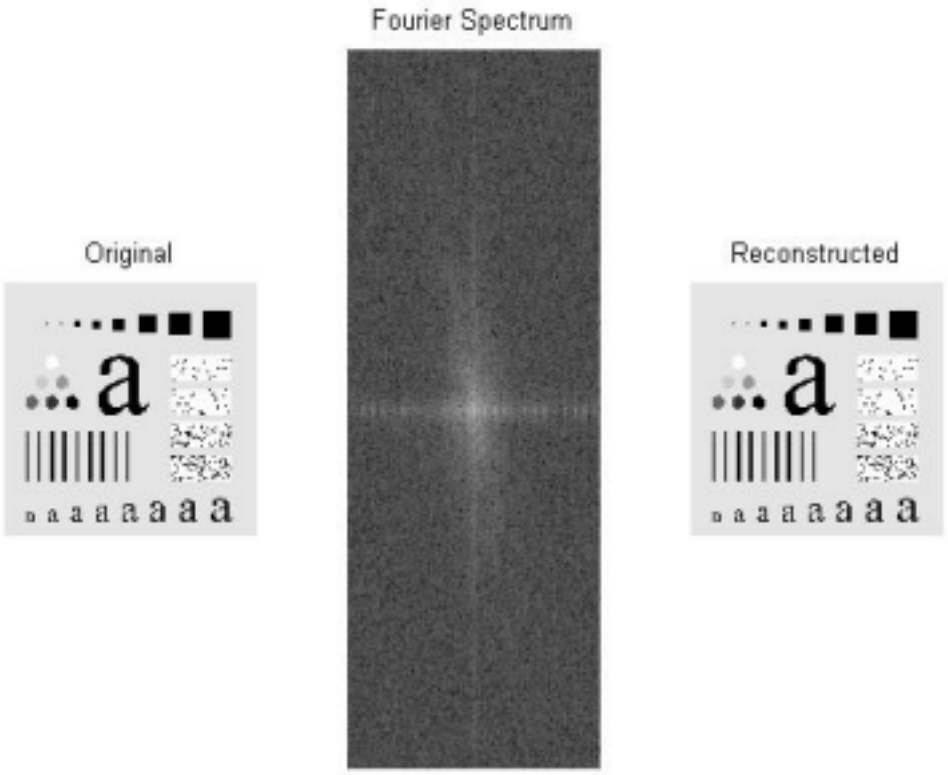
Name: Abhishek Maheshwari
University-Roll-No: 191500030
Roll-No: 13
Section: E

3. Exercises

Exercise1: Apply FFT and IFFT.

```
%ex1.m
close all
clear
clc
%====================================
% 1) Displaying the Fourier Spectrum:
%====================================
I=imread('Lab8_1.jpg');
I=im2double(I);
FI=fft2(I); %(DFT) get the frequency for the image
FI_S=abs(fftshift(FI));%Shift zero-frequency component to center
of  img_spectrum.
I1=ifft2(FI);
I2=real(I1);
subplot(131),imshow(I),title('Original'),
subplot(132),imagesc(0.5*log(1+FI_S)),title('Fourier Spectrum'),axis
off subplot(133),imshow(I2),title('Reconstructed')
%imagesc: the data is scaled to use the full colormap.
```


Output:

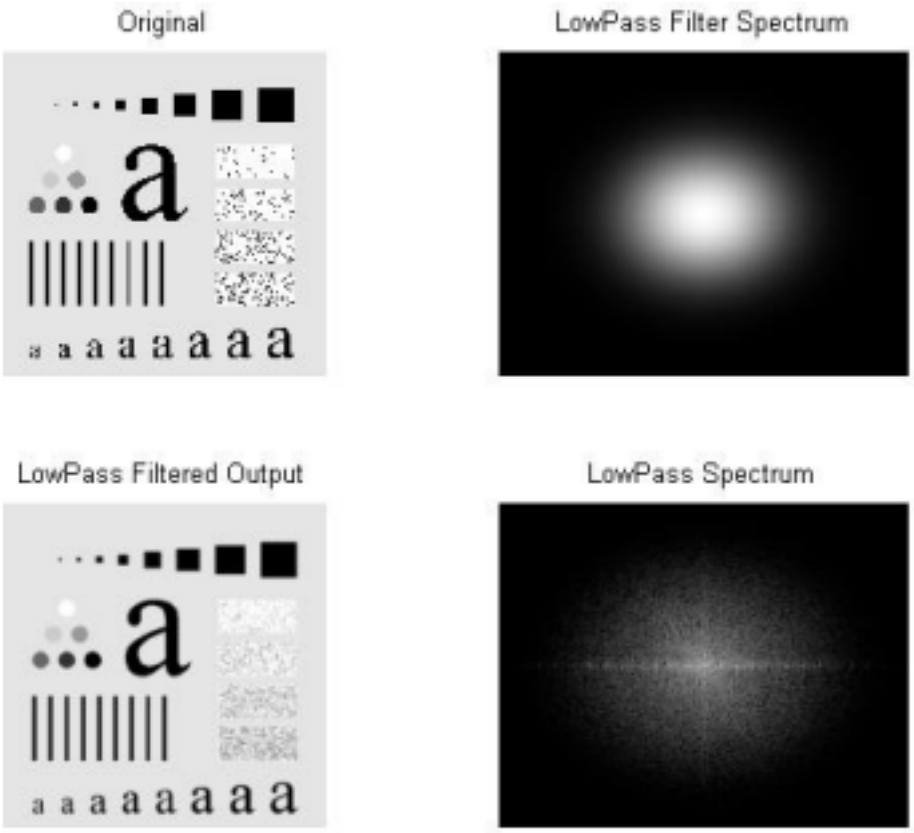


Exercise2: Apply lowpass filter.

```
%ex2.m
close all
clear
clc
%============================
% 2) Low-Pass Gaussian Filter:
%============================
I=imread('Lab8_1.jpg');
I=im2double(I);
FI=fft2(I); %1.Obtain the Fourier transform
LP=fspecial('gaussian',[11 11],1.3); %2.Generate a Low-Pass
filter  FLP=fft2(LP,size(I,1),size(I,2)); %3. Filter padding
LP_OUT=FLP.*FI; %4.Multiply the transform by the
filter I_OUT_LP=ifft2(LP_OUT); %5.inverse DFT
I_OUT_LP=real(I_OUT_LP); %6.Obtain the real part(Output)

%%%%spectrum%%%%
FLP_S=abs(fftshift(FLP));%Filter spectrum
LP_OUT_S=abs(fftshift(LP_OUT));%output spectrum

subplot(221),imshow(I),title('Original'),
subplot(222),imagesc(0.5*log(1+FLP_S)),title('LowPass
Filter  Spectrum'),axis off
subplot(223),imshow(I_OUT_LP),title('LowPass Filtered Output')
```

```
subplot(224),imagesc(0.5*log(1+LP_OUT_S)),title('LowPas
s  Spectrum'),axis off
```

Output:



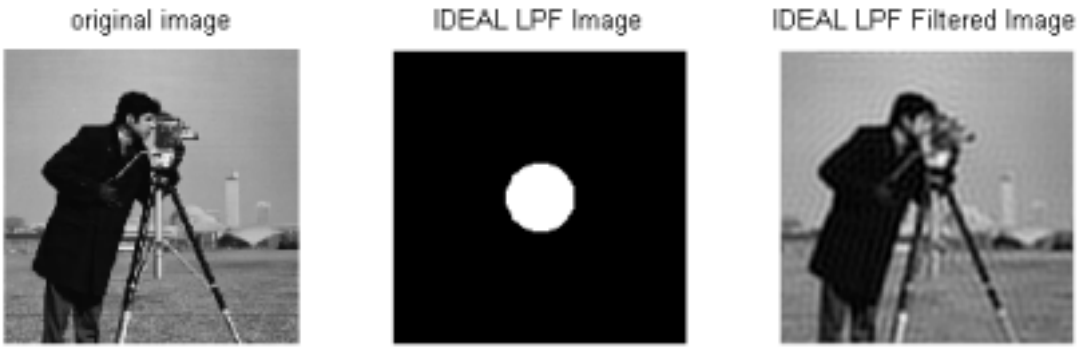
Exercise3: Apply Ideal lowpass filter.

```
%ex3.m
close all
clear
clc
a=imread('Lab8_2.tif');
 [M N]=size(a);
a=im2double(a);
F1=fft2(a); %1.Obtain the Fourier transform

% Set up range of variables.
u = 0:(M-1); %0-255
v = 0:(N-1);%0-255
 % center (u,v) = (M/2,N/2)
% Compute the indices for use in meshgrid
idx = find(u > M/2);% indices 130-255
u(idx) = u(idx) - M;
idy = find(v > N/2);
v(idy) = v(idy) - N;
```

```
%set up the meshgrid arrays needed for
% computing the required distances.
 [U, V] = meshgrid(u, v);

% Compute the distances D(U, V).
D = sqrt(U.^2 + V.^2);

disp('IDEAL LOW PASS FILTERING IN FREQUENCY DOMAIN');

D0=input('Enter the cutoff distance==>');
% Begin filter computations.
H = double(D <= D0);

G=H.*F1; %Multiply
G=ifft2(G);
G=real(G);
ff=abs(fftshift(H));
subplot(131),imshow(a),title('original image')
subplot(132),imshow(ff),title('IDEAL LPF Image')
subplot(133),imshow(G),title('IDEAL LPF Filtered
Image') figure, mesh(ff),axis off,grid off
```

Output:

IDEAL LOW PASS FILTERING IN FREQUENCY DOMAIN
Enter the cutoff distance==>30



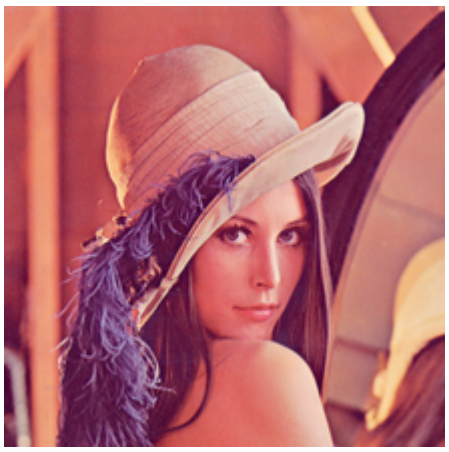
Name: Abhishek Maheshwari
University-Roll-No: 191500030
Roll-No: 13
Section: E

% MATLAB code for Dilation

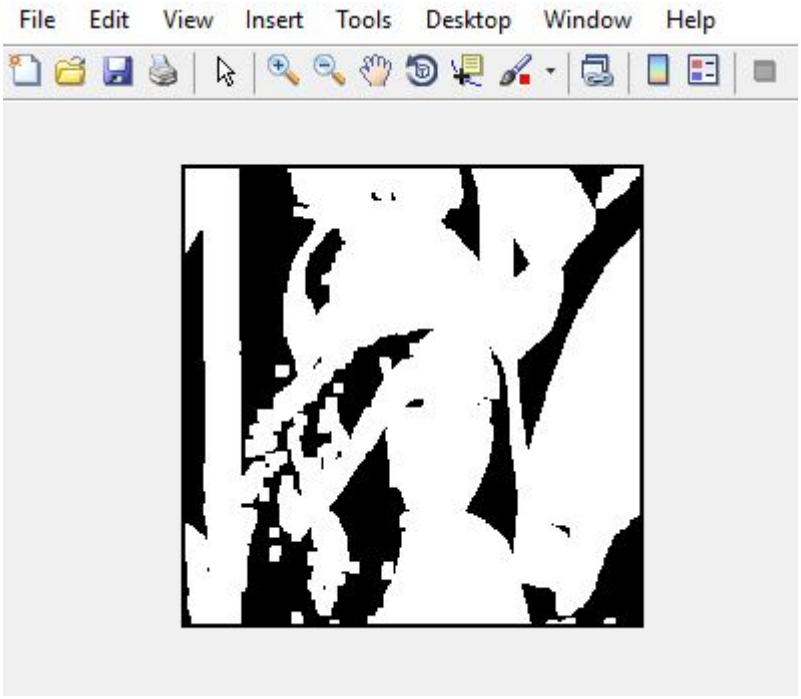
```
% read image

I=imread('lenna.png');

% convert to binary
I=im2bw(I);

% create structuring element
se=ones(5, 5);

% store number of rows in P and
% number of columns in Q.
[P, Q]=size(se);

% create a zero matrix of size I.
In=zeros(size(I, 1), size(I, 2));

for i=ceil(P/2):size(I, 1)-floor(P/2)
    for j=ceil(Q/2):size(I, 2)-floor(Q/2)

        % take all the neighborhoods.
        on=I(i-floor(P/2):i+floor(P/2), j-floor(Q/2):j+floor(Q/2));

        % take logical se
        nh=on(logical(se));
```

```
        % compare and take minimum value of the neighbor
        % and set the pixel value to that minimum value.
        In(i, j)=max(nh(:));
    end
end

imshow(In);
```

Original Image:



Output image:



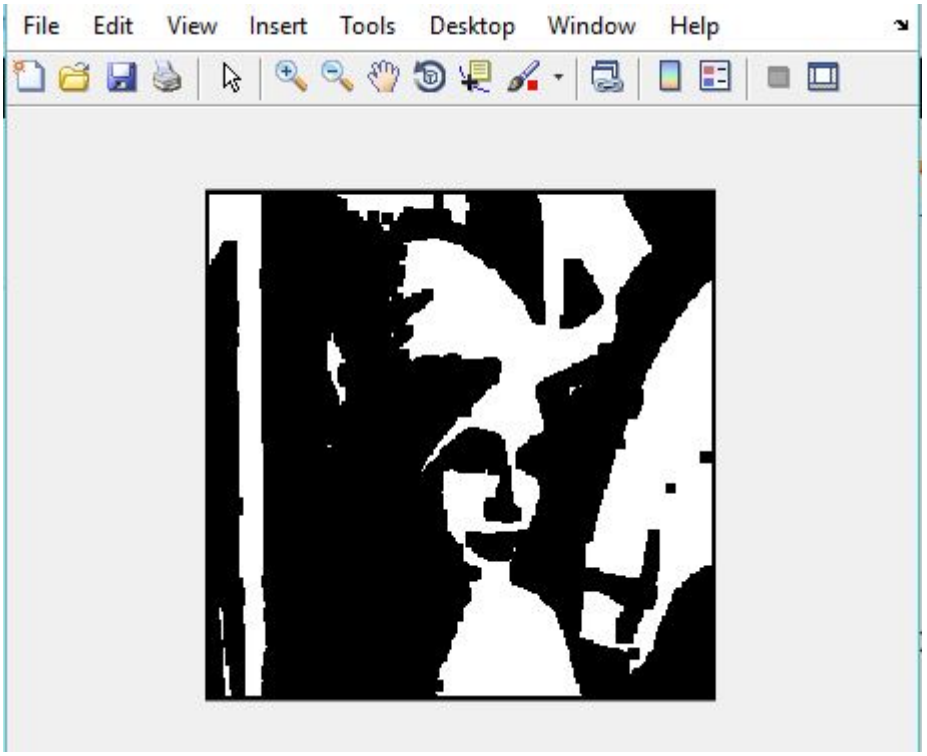
% Matlab code for Erosion

```
% read image
I=imread('lenna.png');

% convert to binary
I=im2bw(I);

% create structuring element
se=ones(5, 5);

% store number of rows
% in P and number of columns in Q.
[P, Q]=size(se);

% create a zero matrix of size I.
In=zeros(size(I, 1), size(I, 2));

for i=ceil(P/2):size(I, 1)-floor(P/2)
    for j=ceil(Q/2):size(I, 2)-floor(Q/2)
```

```
        % take all the neighbourhoods.
        on=I(i-floor(P/2):i+floor(P/2), j-floor(Q/2):j+floor(Q/2));

        % take logical se
        nh=on(logical(se));

        % compare and take minimum value of the neighbor
        % and set the pixel value to that minimum value.
        In(i, j)=min(nh(:));
    end
end

imshow(In);
```

Original Image:

Output image:


File Edit View Insert Tools Desktop Window Help

Digital Image Processing Lab

Name - Abhishek Maheshwari

Section - E

Roll No - 13

University Roll No - 191500030

Submitted To - Pooja Mam

How to Normalize a Histogram in MATLAB?

CODE:

```
% MATLAB code for
% Histogram normalisation.
% Read the image.
k=imread("lincoln.jfif");

% Convert into grayscale
```

```
k1=rgb2gray(k);

% Display the image and histogram.
imtool(k1,[]);
imhist(k1);

% Set the minimum and maximum
% Values from histogram.
min=45;
max=180;

% Convert image into double.
k2=double(k1);

% Apply the formula.
k3=(k2-min)./(max-min);

% Multiply with maximum possible value.
k4=k3.*255;

% Convert the final result into uint8.
k5=uint8(k4);

% Display the enhanced image and histogram.
imtool(k5,[]);
imhist(k5);
```

OUTPUT:

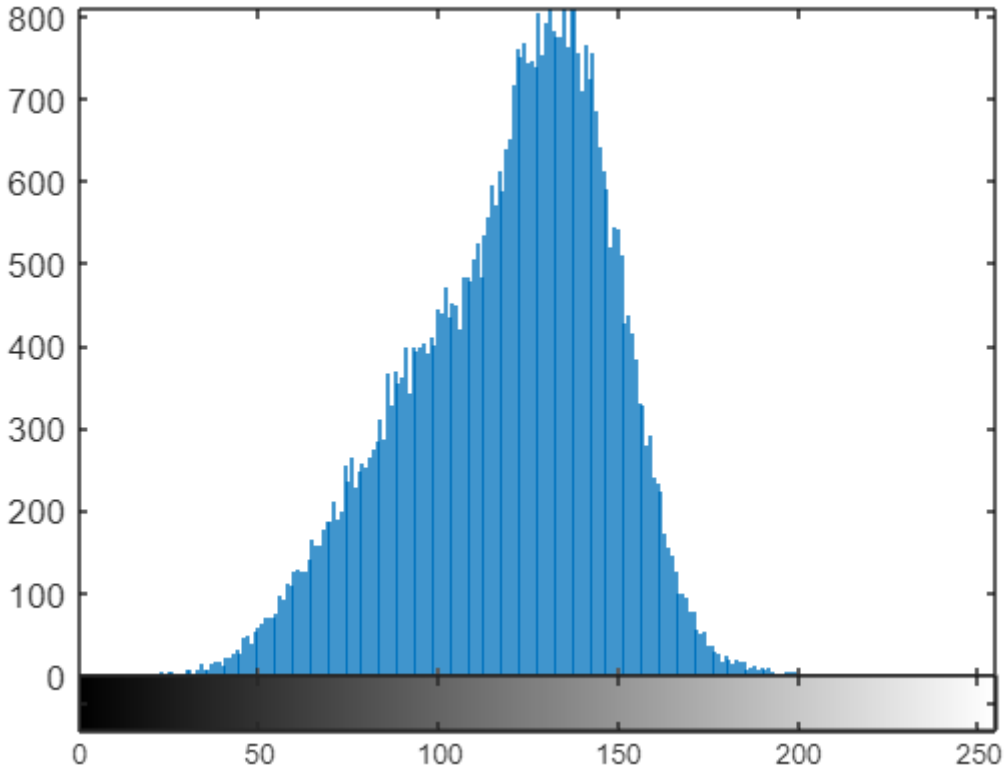


Figure: Original histogram

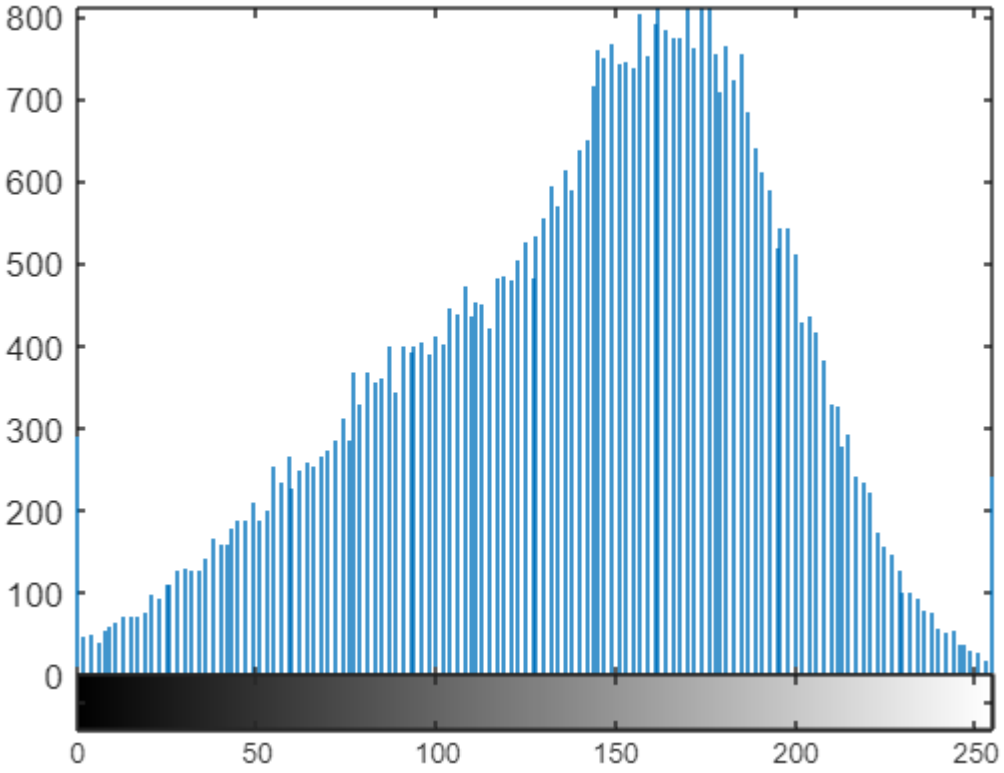


Figure: Normalized histogram

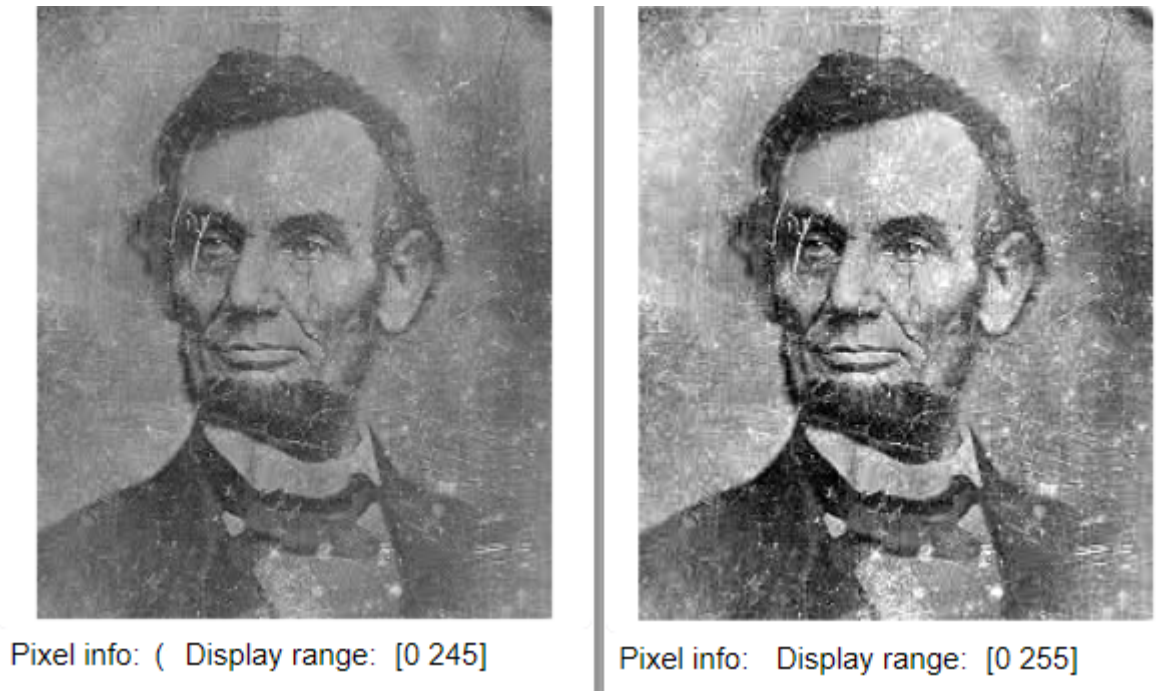


Figure: Images before and after normalization

Matlab code: Histogram equalization without using histeq function

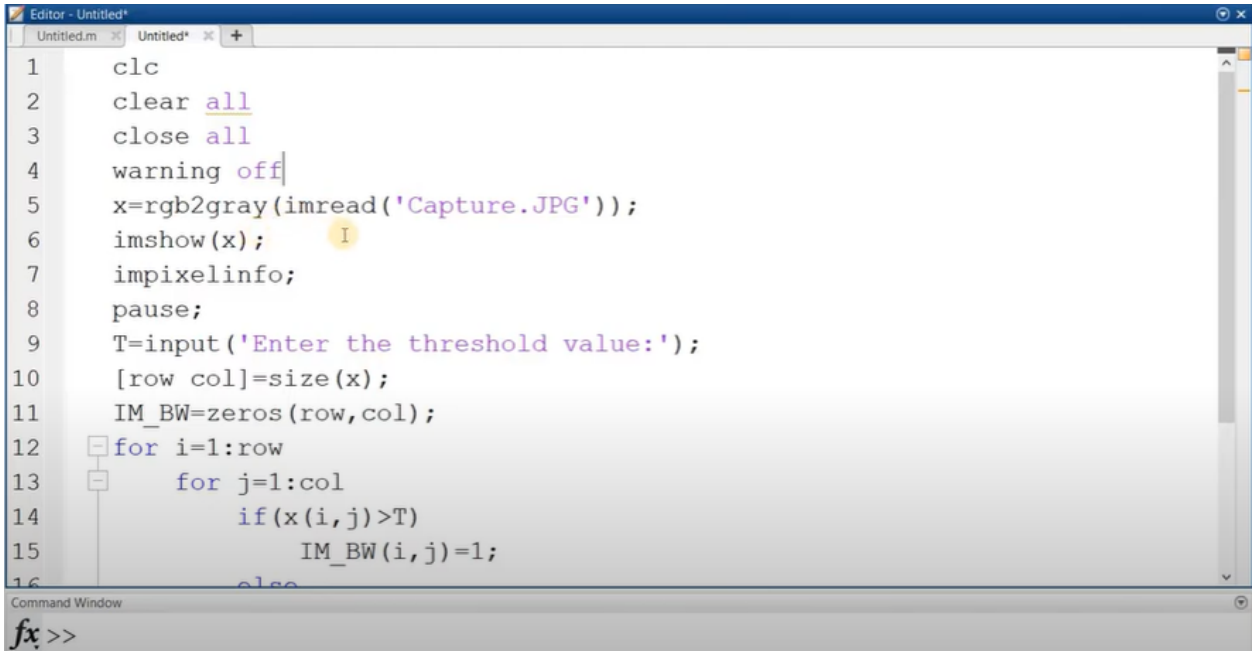
```
GIm=imread('tire.tif');

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

figure,imshow(GIm);

title('Original Image');
```

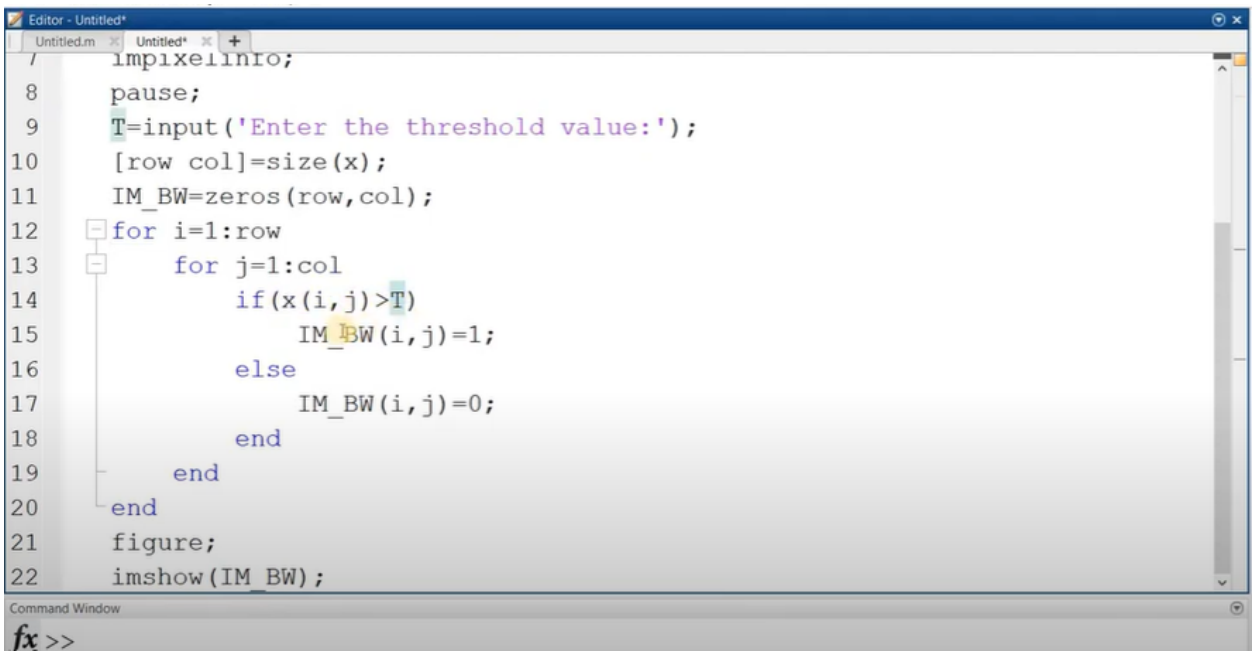

```
HIm=uint8(zeros(size(GIm,1),size(GIm,2)));

freq=zeros(256,1);

probf=zeros(256,1);

probc=zeros(256,1);

cum=zeros(256,1);

output=zeros(256,1);

%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

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
```

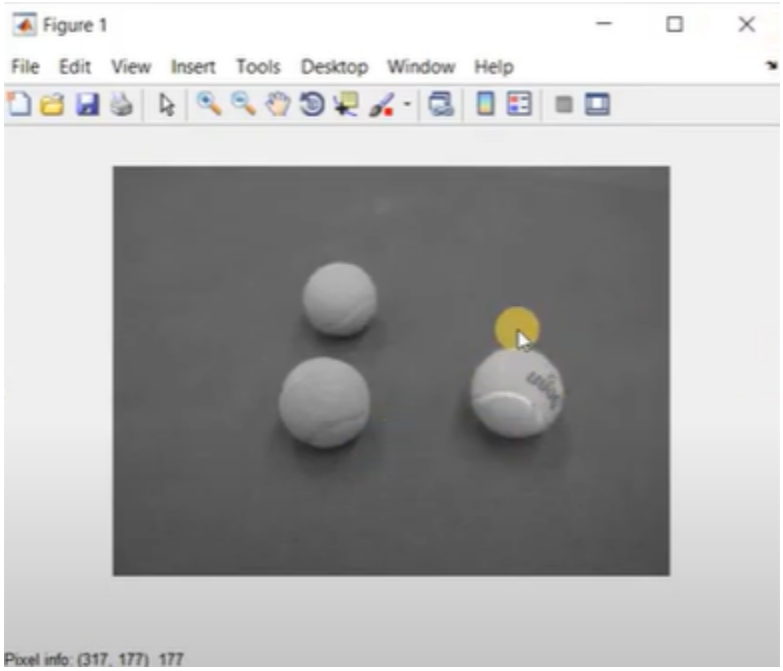
```
for i=1:size(GIm,1)

    for j=1:size(GIm,2)

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

    end

end

figure,imshow(HIm);

title('Histogram equalization');
```

Name: Abhishek Maheshwari
University-Roll-No: 191500030
Roll-No: 13
Section: E

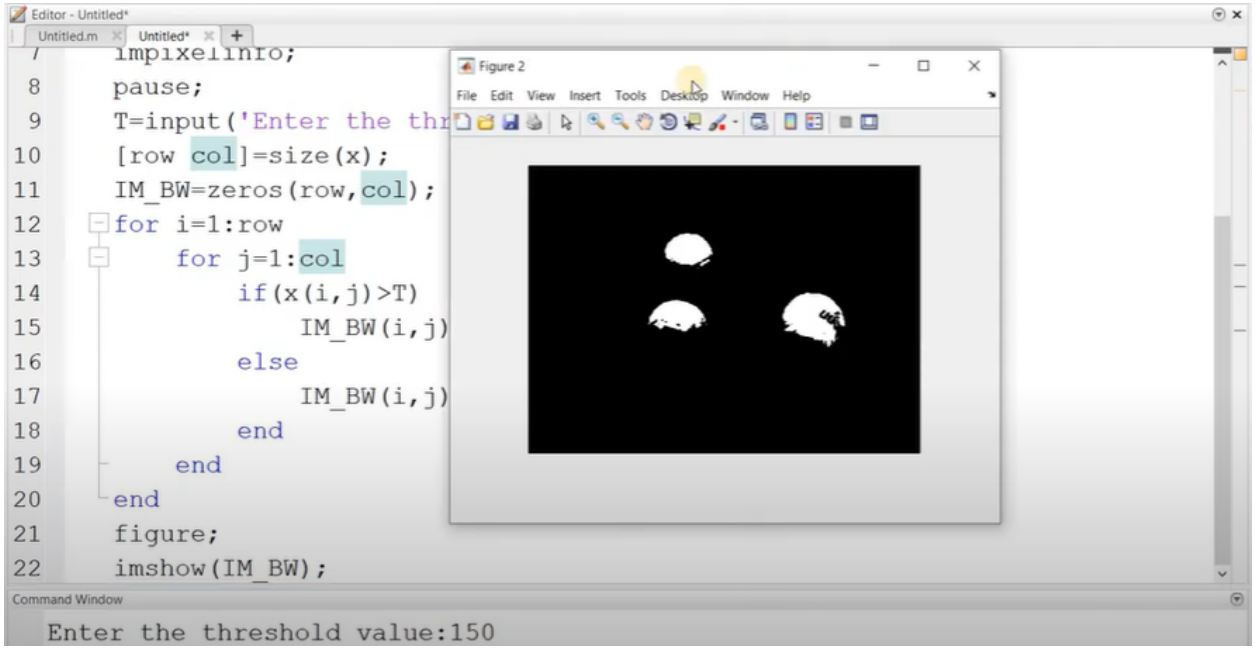




Original Image(before execution):



After the execution:



Digital Image Processing Lab

Name - Abhishek Maheshwari

Section - E

Roll No - 13

University Roll No - 191500030

Submitted To - Pooja Mam

Apply Gaussian Smoothing Filters to Images

```
I = imread('cameraman.tif');
```

```
Iblur1 = imgaussfilt(I,2);
Iblur2 = imgaussfilt(I,4);
Iblur3 = imgaussfilt(I,8);
```

Display the original image and all the filtered images.

```
figure
imshow(I)
title('Original image')
```


```
figure
imshow(Iblur1)
title('Smoothed image, \sigma = 2')
```


```
figure
imshow(Iblur2)
title('Smoothed image, \sigma = 4')
```


```
figure
imshow(Iblur3)
title('Smoothed image, \sigma = 8')
```


```
IblurX1 = imgaussfilt(I,[4 1]);
IblurX2 = imgaussfilt(I,[8 1]);
IblurY1 = imgaussfilt(I,[1 4]);
IblurY2 = imgaussfilt(I,[1 8]);
```

Display the filtered images.

```
figure
imshow(IblurX1)
title('Smoothed image, \sigma_x = 4, \sigma_y = 1')
```


```
figure
imshow(IblurX2)
title('Smoothed image, \sigma_x = 8, \sigma_y = 1')
```

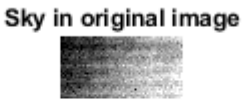

```
figure
imshow(IblurY1)
title('Smoothed image, \sigma_x = 1, \sigma_y = 4')
```


```
figure
imshow(IblurY2)
title('Smoothed image, \sigma_x = 1, \sigma_y = 8')
```


```
I_sky = imadjust(I(20:50,10:70));
IblurX1_sky = imadjust(IblurX1(20:50,10:70));
```

Display the original patch of sky with the filtered version.

```
figure
imshow(I_sky), title('Sky in original image')
```




```
figure
imshow(IblurX1_sky), title('Sky in filtered image')
```

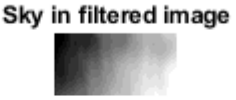


Image Sharpening Using Laplacian Filter

```
% MatLab program for edge sharpening.
% Read the image in variable 'a'
a=imread("cameraman.jpg");

% Defined the laplacian filter.
Lap=[0 1 0; 1 -4 1; 0 1 0];

% Convolve the image read
% in 'a' with Laplacian mask.
a1=conv2(a,Lap,'same');

% After convolution the intensity
% Values go beyond the range.
% Normalise the range of intensity.
a2=uint8(a1);

% Display the sharpened image.
imtool(abs(a-a2),[])

% Define strong laplacian filter
lap=[-1 -1 -1; -1 8 -1; -1 -1 -1];

% Apply filter on original image
a3=conv2(a,lap,'same');

% Normalise the resultant image.
a4=uint8(a3);
```

```
% Display the sharpened image.
imtool(abs(a+a4),[])
```

Output:


Image Tool 1 - (MATLAB Expres...
File Tools
Pixel info: (X Display range: [0 255]

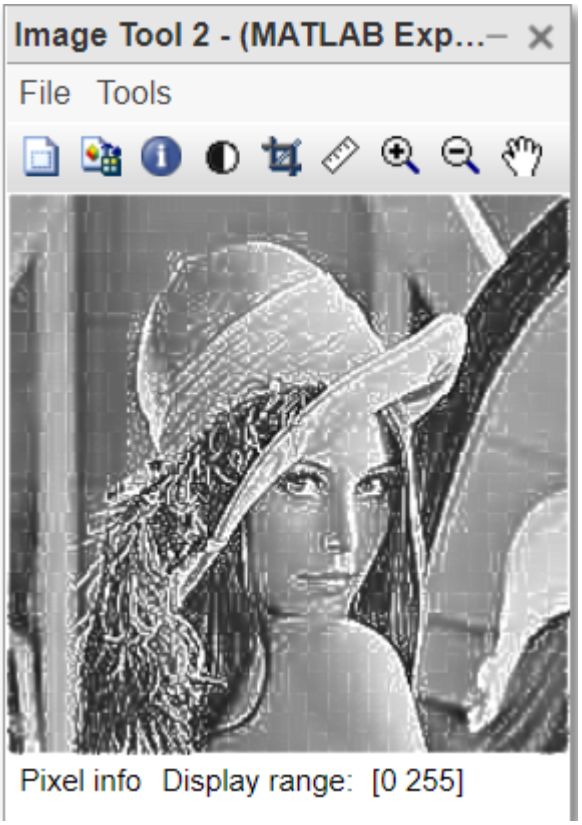


Image Tool 2 - (MATLAB Exp...
File Tools
Pixel info Display range: [0 255]