**Practical – 1**

**AIM:** Explore the image read and display functionality of MATLAB and Python. Observe the different components of images.

**CODE:**

clear all;

close all;

clc;

img = imread('cameraman.tif');

imshow(img),title("Image");

**OUTPUT:**

****

**CONCLUSION:**

From this practical, we learnt to read and display an image in MATLAB.

**Practical – 2**

**AIM:** Write an MATLAB and Python program to demonstrate following operations on your own passport size image. Observe the effect of sampling and quantization on original image.

* 1. Reduce the spatial resolution of the image by ½.

**CODE:**

clear all;

close all;

clc;

img = imread('cameraman.tif');

imshow(img),title("Image");

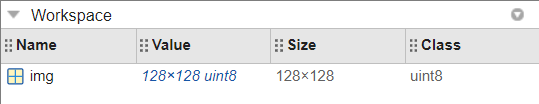
img=imresize(img,0.5);

figure;

imshow(img),title('Resized Image');

**OUTPUT:**

****

****

* 1. Reduce the spatial resolution of the image by ¼.

**CODE:**

clear all;

close all;

clc;

img = imread('cameraman.tif');

imshow(img),title("Image");

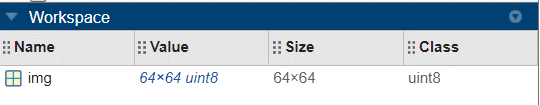
img=imresize(img,0.25);

figure;

imshow(img),title('Resized Image');

**OUTPUT:**

****

****

* 1. Double the spatial resolution of image.

**CODE:**

clear all;

close all;

clc;

img = imread('cameraman.tif');

imshow(img),title("Image");

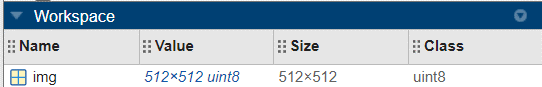
img=imresize(img,2);

figure;

imshow(img),title('Resized Image');

**OUTPUT:**

****

****

* 1. Increase image resolution to 4 times.

**CODE:**

clear all;

close all;

clc;

img = imread('cameraman.tif');

imshow(img),title("Image");

img=imresize(img,4);

figure;

imshow(img),title('Resized Image');

**OUTPUT:**

****

****

* 1. Set gray level resolution of image to 2.

**CODE:**

clear;

clc;

close all;

img=imread('cameraman.tif');

[r,c]=size(img);

% 2 level

for i=1:r

for j=1:c

if(img(i,j)>127)

img(i,j)=255;

else

img(i,j)=0;

end

end

end

figure;

imshow(img);

**OUTPUT:**

****

* 1. Set gray level resolution of image to 3.

**CODE:**

clear;

clc;

close all;

img=imread('cameraman.tif');

[r,c]=size(img);

for i=1:r

for j=1:c

if img(i,j)<100

img(i,j)=100;

elseif img(i,j)>= 100 && img(i,j)<=200

img(i,j)=150;

else

img(i,j)=200;

end

end

end

figure;

imshow(img);

**OUTPUT:**

****

* 1. Set gray level resolution of image to 10.

**CODE:**

clear;

clc;

close all;

img=imread('cameraman.tif');

[r,c]=size(img);

for i=1:r

for j=1:c

if img(i,j)<120

img(i,j)=120;

elseif img(i,j)>= 120 && img(i,j)<=130

img(i,j)=img(i,j);

else

img(i,j)=130;

end

end

end

figure;

imshow(img);

**OUTPUT:**

****

* 1. Set gray level resolution of image to 50.

**CODE:**

clear;

clc;

close all;

img=imread('cameraman.tif');

[r,c]=size(img);

for i=1:r

for j=1:c

if img(i,j)<150

img(i,j)=150;

elseif img(i,j)>= 150 && img(i,j)<=200

img(i,j)=img(i,j);

else

img(i,j)=200;

end

end

end

figure;

imshow(img);

**OUTPUT:**

****

* 1. Set gray level resolution of image to 100.

**CODE:**

clear;

clc;

close all;

img=imread('cameraman.tif');

[r,c]=size(img);

for i=1:r

for j=1:c

if img(i,j)<100

img(i,j)=100;

elseif img(i,j)>= 100 && img(i,j)<=200

img(i,j)=img(i,j);

else

img(i,j)=200;

end

end

end

figure;

imshow(img);

**OUTPUT:**

****

**CONCLUSION:**

From this practical, we learnt to set spatial resolution and set gray level resolution of an image in MATLAB.

**Practical – 3**

**AIM:** Using MATLAB and Python implement and analyze effect of below given spatial domain techniques on specified sample images

* 1. Image negation.

**CODE:**

clear all;

close all;

clc;

img = imread('e7.tif');

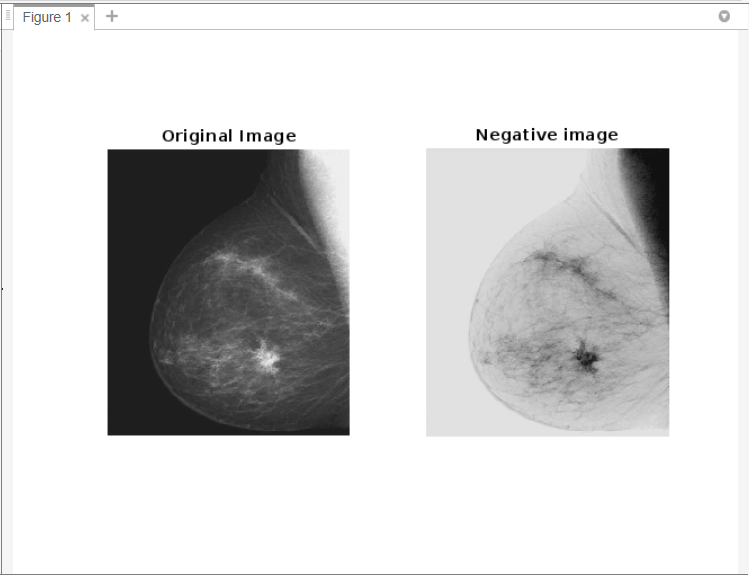
n\_img=255-img;

figure;

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

subplot(1,2,2); imshow(n\_img); title('Negative image');

**OUTPUT:**

****

* 1. Log Transformation.

**CODE:**

clear;

close;

close all;

% Load the image

img = imread('e9.tif');

% Convert the image to double precision

img = im2double(img);

% Apply log transformation

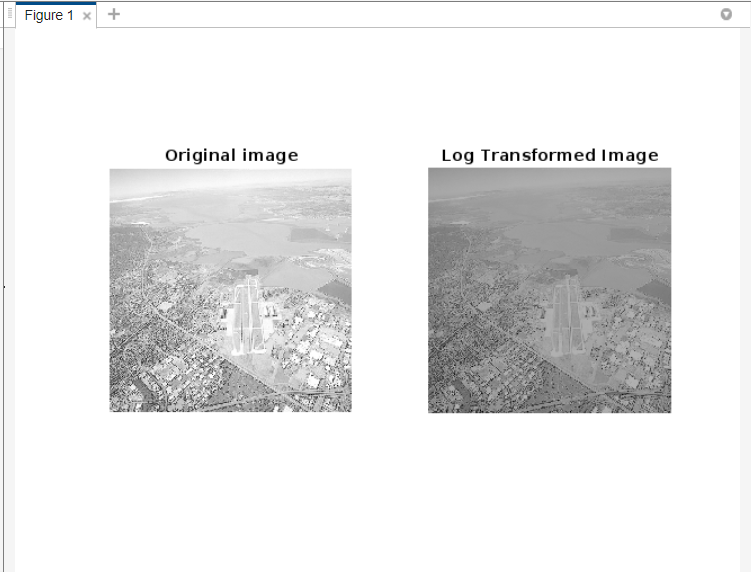
log\_img = log(1 + img);

figure;

subplot(1,2,1); imshow(img); title('Original image');

subplot(1,2,2); imshow(log\_img); title('Log Transformed Image');

**OUTPUT:**

****

* 1. Inverse Log Transformation.

**CODE:**

clear;

close;

close all;

% Load the image

img = imread('e9.tif');

% Convert the image to double precision

img = im2double(img);

% Apply log transformation

log\_img = log(1 + img);

figure;

imshow(log\_img),title('Log Transformation');

% Apply inverse log transformation

inv\_log\_img = exp(log\_img) - 1;

% Display the results

figure;

subplot(1,2,1); imshow(img); title('Original image');

subplot(1,2,2); imshow(inv\_log\_img); title('Inverse log transformed image');

**OUTPUT:**

****

* 1. Power Low Transformation.

**CODE:**

clc;

clear all;

close all;

img=imread('e9.tif');

imshow(img);

[r,c]=size(img);

img2=size(img);

img3=size(img);

img2=uint8(img2);

img3=uint8(img3);

for i=1:r

for j=1:c

img2(i,j)=1\*(double(img(i,j))^(1/1.5));

end

end

for i=1:r

for j=1:c

img3(i,j)=1\*(double(img2(i,j))^1.5);

end

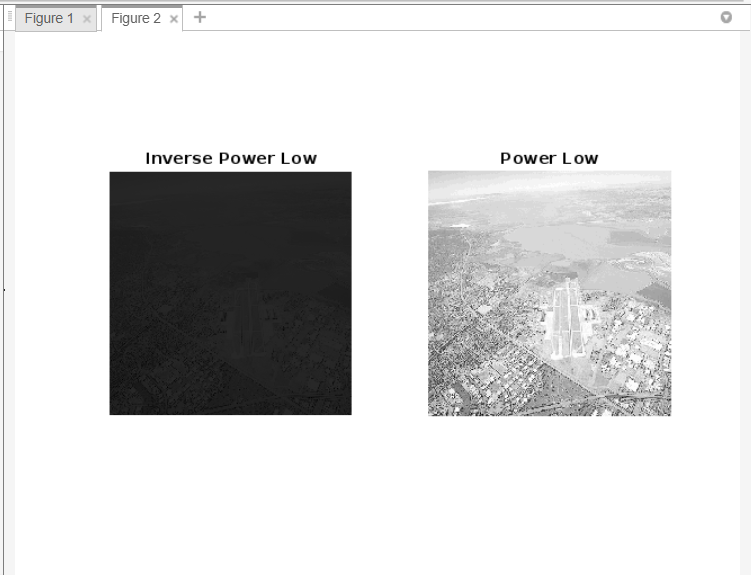
end

figure;

subplot(1,2,1), imshow(img2),title('Inverse Power Low');

subplot(1,2,2), imshow(img3),title('Power Low');

**OUTPUT:**

****

* 1. Gray level slicing

**CODE:**

clear;

close;

close all;

img1 = imread('cameraman.tif');

img2 = img1;

T1=input('Enter the Lowest threshold value:');

T2=input('Enter the Highest threshold value:');

[r,c] = size(img2);

for i = 1:r

for j = 1:c

if((img2(i,j)>T1) && (img2(i,j)<T2))

img2(i,j)=img1(i,j);

else

img2(i,j)=255;

end

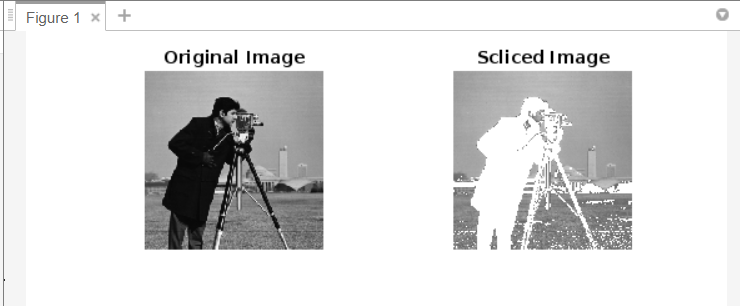
end

end

subplot(2,2,1), imshow(img1), title('Original Image');

subplot(2,2,2), imshow(img2), title('Scliced Image');

**OUTPUT:**

****

* 1. Contrast stretching.

**CODE:**

clear;

close;

close all;

img1 = imread('e10.tif');

img2 = img1;

fmax = max(max(img1));

fmin = min(min(img1));

[r,c] = size(img1);

for i = 1:r

for j = 1:c

img2(i,j) = 255 \* ((img1(i,j) - fmin) / (fmax - fmin));

end

end

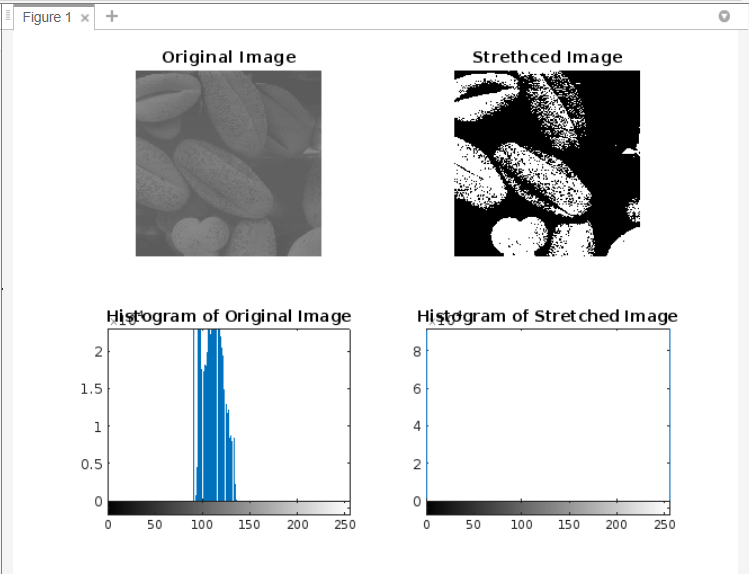
subplot(2,2,1), imshow(img1), title('Original Image');

subplot(2,2,2), imshow(img2), title('Strethced Image');

subplot(2,2,3), imhist(img1), title('Histogram of Original Image');

subplot(2,2,4), imhist(img2), title('Histogram of Stretched Image');

**OUTPUT:**

****

* 1. Thresholding

**CODE:**

clear;

close;

close all;

img = imread('e10.tif');

img1=img;

[r,c] = size(img1);

for i = 1:r

for j = 1:c

if(img1(i,j)<127)

img1(i,j)=0;

else

img1(i,j)=255;

end

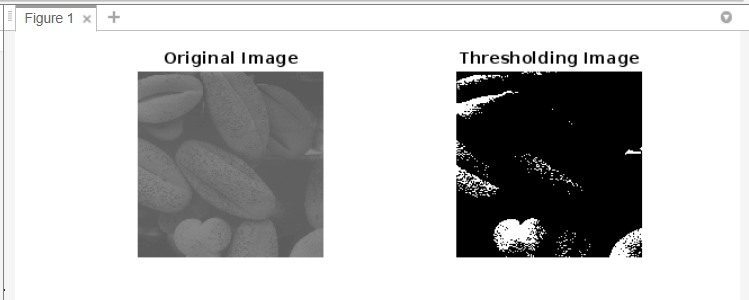
end

end

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

subplot(2,2,2), imshow(img1), title('Thresholding Image');

**OUTPUT:**

****

* 1. Histogram equalization.

**CODE:**

clc;

clear;

close all;

img=imread('e10.tif');

imghist=imhist(img);

% PMF calculation start %

pmf=zeros(size(imghist));

[r,c]=size(pmf);

[ro,co]=size(img);

for i=1:r

for j=1:c

pmf(i,j)=imghist(i,j)/(ro\*co);

end

end

% PMF calculation end %

% CDF calculation start %

cdf=zeros(size(imghist));

[r,c]=size(cdf);

maxinten=max(max(img));

cdf(1,1)=pmf(1,1)\*maxinten;

for i=2:r

for j=1:c

cdf(i,j)=cdf(i-1,j)+pmf(i,j)\*maxinten;

end

end

% CDF calculation end %

% Equalization calculation start %

img2=zeros(size(img));

for k=1:ro

for i=1:ro

for j=1:co

if((k-1)==img(i,j))

img2(i,j)=cdf(k,1);

end

end

end

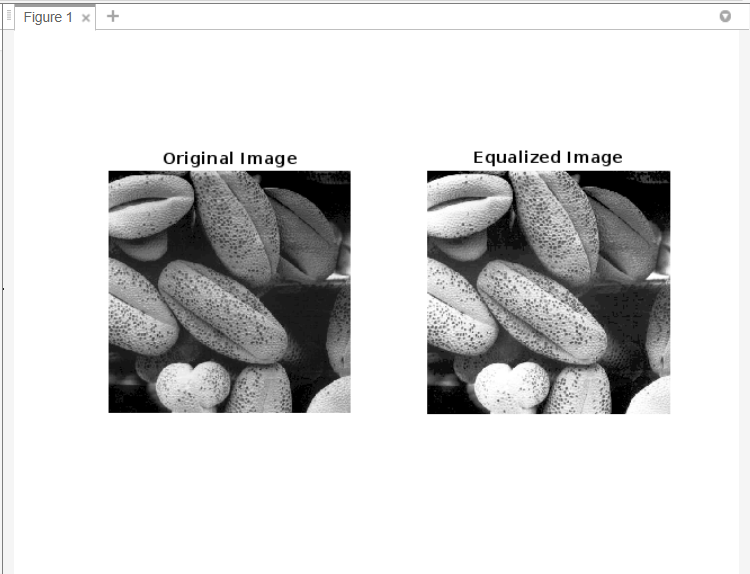
end

% Equalization calculation start %

subplot(1,2,1), imshow(img,[]),title('Original Image');

subplot(1,2,2), imshow(img2,[]),title('Equalized Image');

**OUTPUT:**

****

* 1. Histogram matching.

**CODE:**

Clc;

clear All;

%Histogram Matching

img7 = imread('e3.jpg');

target\_histogram = imhist(imread('e11.tif'));

matched\_img = histeq(img7, target\_histogram);

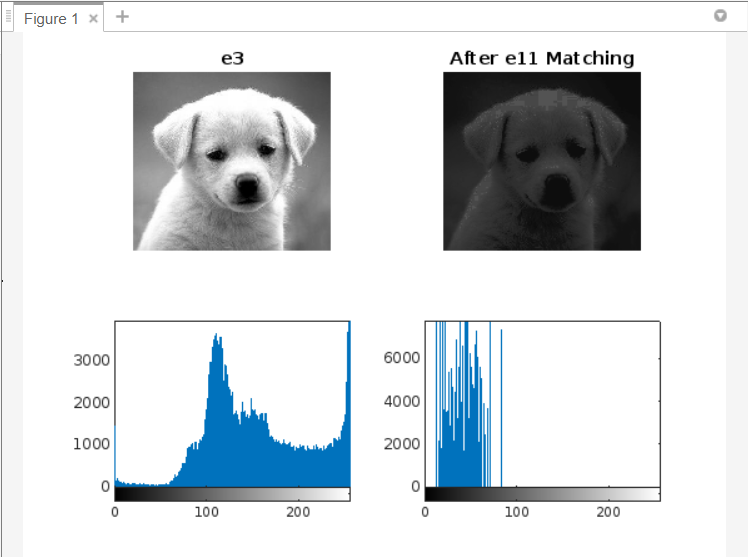
subplot(2,2,1);imshow(img7);title('e3');

subplot(2,2,2);imshow(matched\_img);title('After e11 Matching');

subplot(2,2,3);imhist(img7);

subplot(2,2,4);imhist(matched\_img);

**OUTPUT:**

****

**CONCLUSION:**

From this practical, we learnt to implement different spatial domain techniques on an image in MATLAB.

**Practical – 4**

**AIM:** Create user define defined function to implement below given spatial domain smoothening filtering. Function should provide support for filtering with or without padding.

* 1. Box Filter

**CODE:**

clear all;

close all;

clc;

img = imread('e15.tif');

imshow(img);

nimg=img;

[r,c]=size(nimg);

temp=zeros(r+2,c+2);

for i=1:r

for j=1:c

temp(i,j)=nimg(i,j);

end

end

B=zeros(3,3);

sum=0;

for i=1:r

for j=1:c

for k=1:3 %3\*3 averaging filter

for m=1:3

B(k,m)=temp(i+k-1,j+m-1);

end

end

for k=1:3 %3\*3 averaging filter

for m=1:3

sum=sum+B(k,m);

end

end

sum=sum/9;

nimg(i,j)=sum;

end

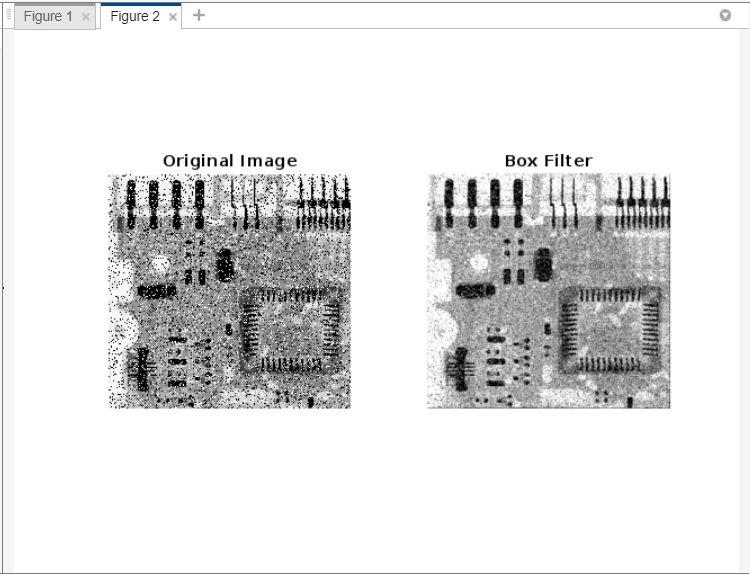
end

figure;

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

subplot(1,2,2); imshow(nimg); title('Box Filter');

**OUTPUT:**

****

* 1. Weighted average filter

**CODE:**

clear all;

close all;

clc;

img=imread('e15.tif');

imshow(img),title('Original Image');

nimg=img;

mask=[2,2,2;2,2,2;2,2,2];

[r,c]=size(nimg);

temp=zeros(r+2,c+2);

for i=1:r

for j=1:c

temp(i,j)=nimg(i,j);

end

end

B=zeros(3,3);

sum=0;

for i=1:r

for j=1:c

for k=1:3 %extracting mask

for m=1:3

B(k,m)=temp(i+k-1,j+m-1);

end

end

sum=0;

for k=1:3 %3\*3 averaging filter

for m=1:3

sum=sum+(mask(k,m)\*B(k,m));

end

end

sum=sum/18;

nimg(i,j)=sum;

end

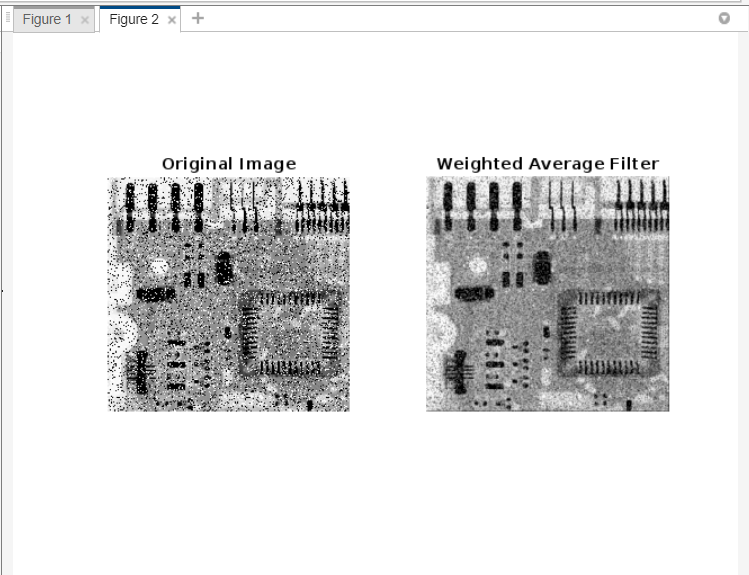
end

figure;

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

subplot(1,2,2); imshow(nimg); title('Weighted Average Filter');

**OUTPUT:**

****

* 1. Max filter

**CODE:**

clear all;

close all;

clc;

img = imread('e15.tif');

imshow(img),title("Original Image");

nimg=img;

[r,c]=size(nimg);

temp=zeros(r+2,c+2);

for i=1:r

for j=1:c

temp(i+1,j+1)=nimg(i,j);

end

end

B=zeros(3,3);

for i=1:r

for j=1:c

for k=1:3 %3\*3 averaging filter

for m=1:3

B(k,m)=temp(i+k-1,j+m-1);

end

end

max=0;

for k=1:3 %3\*3 averaging filter

for m=1:3

if(B(k,m)>max)

max=B(k,m);

end

end

end

nimg(i,j)=max;

end

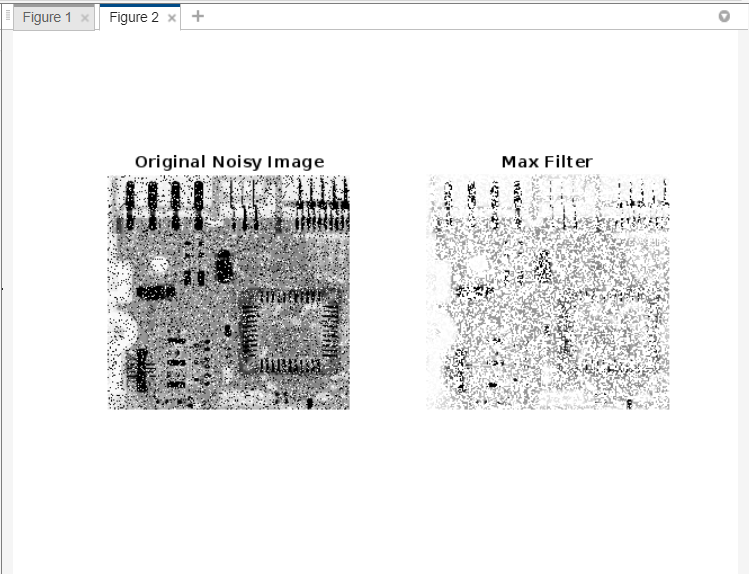
end

figure;

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

subplot(1,2,2); imshow(nimg); title('Max Filter');

**OUTPUT:**

****

* 1. Min filter

**CODE:**

clear all;

close all;

clc;

img = imread('e15.tif');

imshow(img),title("Original Image");

nimg=img;

[r,c]=size(nimg);

temp=zeros(r+2,c+2);

for i=1:r

for j=1:c

temp(i+1,j+1)=nimg(i,j);

end

end

B=zeros(3,3);

for i=1:r

for j=1:c

for k=1:3 %3\*3 averaging filter

for m=1:3

B(k,m)=temp(i+k-1,j+m-1);

end

end

max=256;

for k=1:3 %3\*3 averaging filter

for m=1:3

if(B(k,m)<max)

max=B(k,m);

end

end

end

nimg(i,j)=max;

end

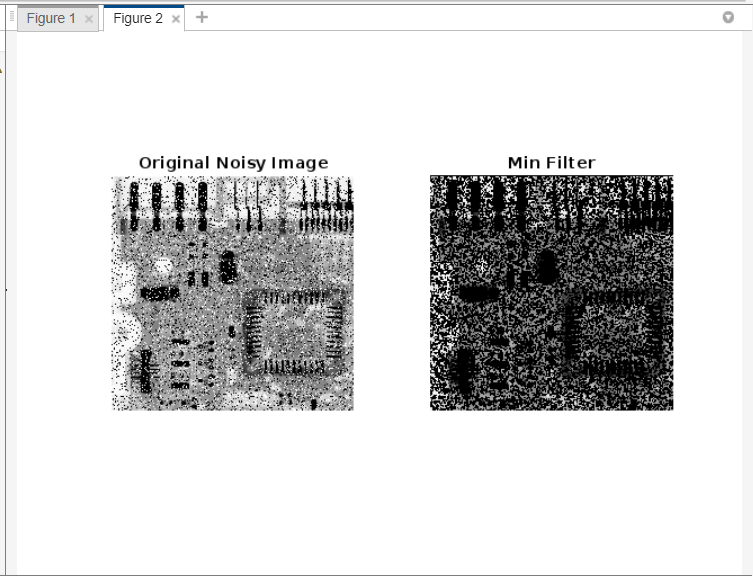
end

figure;

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

subplot(1,2,2); imshow(nimg); title('Min Filter');

**OUTPUT:**

****

* 1. Median filter.

**CODE:**

clear all;

close all;

clc;

img = imread('e15.tif');

imshow(img),title("Original Image");

nimg=img;

[r,c]=size(nimg);

temp=zeros(r+2,c+2);

for i=1:r

for j=1:c

temp(i+1,j+1)=nimg(i,j);

end

end

B=zeros(3,3);

for i=1:r

for j=1:c

for k=1:3 %3\*3 averaging filter

for m=1:3

B(k,m)=temp(i+k-1,j+m-1);

end

end

max=256;

for k=1:3 %3\*3 averaging filter

for m=1:3

B=sort(B);

end

end

nimg(i,j)=B(2,2);

end

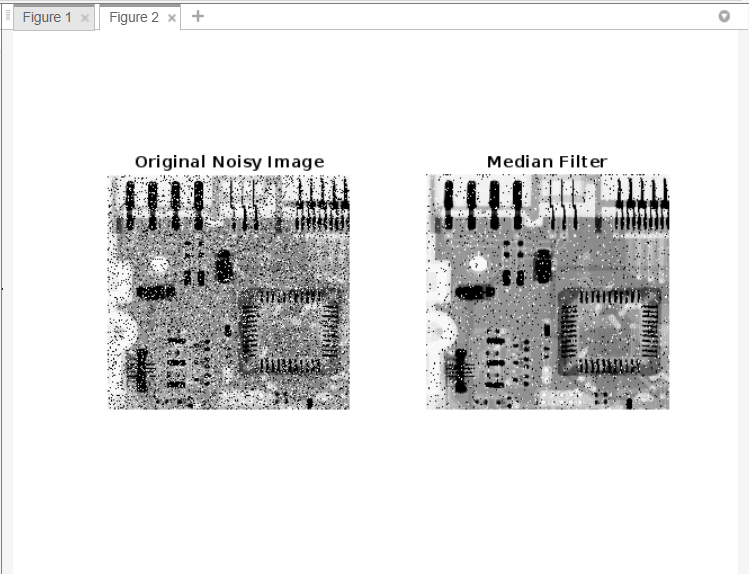
end

figure;

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

subplot(1,2,2); imshow(nimg); title('Median Filter');

**OUTPUT:**

****

**CONCLUSION:**

From this practical, we learnt to implement spatial domain smoothening filtering techniques on an image in MATLAB.

**Practical – 5**

**AIM:** Create user define defined function to implement below given spatial domain sharpening filtering. Function should provide support for filtering with or without padding.

* 1. Laplacian

**CODE:**

clear all;

close all;

clc;

img=imread('e16.tif');

imshow(img),title('Original Image');

nimg=img;

mask=[-1 -1 -1;-1 8 -1;-1 -1 -1];

[r,c]=size(nimg);

temp=zeros(r+2,c+2);

for i=1:r

for j=1:c

temp(i,j)=nimg(i,j);

end

end

B=zeros(3,3);

sum=0;

for i=1:r

for j=1:c

for k=1:3 %extracting mask

for m=1:3

B(k,m)=temp(i+k-1,j+m-1);

end

end

sum=0;

for k=1:3 %3\*3 averaging filter

for m=1:3

sum=sum+(mask(k,m)\*B(k,m));

end

end

nimg(i,j)=sum;

end

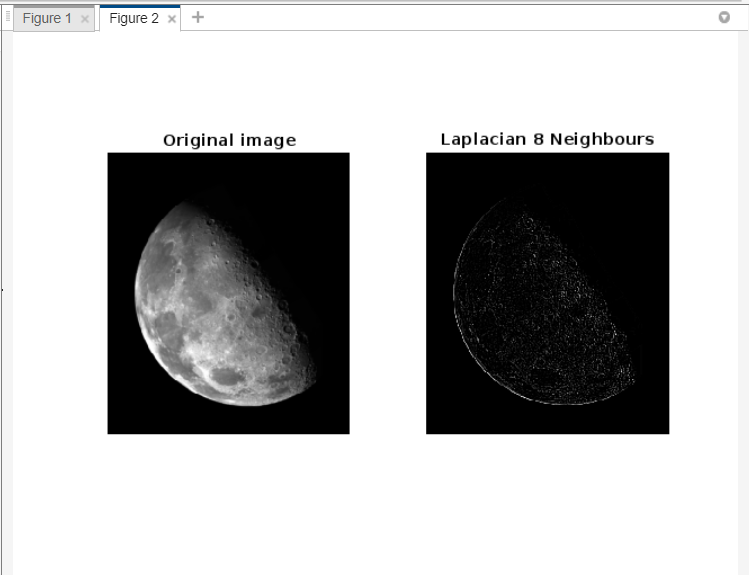
end

figure;

subplot(1,2,1); imshow(img); title('Original image');

subplot(1,2,2); imshow(nimg); title('Laplacian 8 Neighbours');

**OUTPUT:**

****

* 1. Gradient

**CODE:**

clear all;

close all;

clc;

img=imread('e16.tif');

imshow(img),title('Original Image');

img1=img;

mask=[-1 -2 -1;0 0 0;1 2 1];

mask1=[-1 0 1;-2 0 2;-1 0 1];

[r,c]=size(img);

temp=zeros(r+2,c+2);

for i=1:r

for j=1:c

temp(i,j)=img(i,j);

end

end

B=zeros(3,3);

sum=0;

for i=1:r

for j=1:c

for k=1:3 %extracting mask

for m=1:3

B(k,m)=temp(i+k-1,j+m-1);

end

end

sum=0;

for k=1:3

for m=1:3

sum=sum+(mask(k,m)\*B(k,m));

end

end

img(i,j)=sum;

end

end

[r,c]=size(img1);

temp1=zeros(r+2,c+2);

for i=1:r

for j=1:c

temp1(i,j)=img1(i,j);

end

end

C=zeros(3,3);

for i=1:r

for j=1:c

for k=1:3 %extracting mask

for m=1:3

C(k,m)=temp1(i+k-1,j+m-1);

end

end

sum=0;

for k=1:3

for m=1:3

sum=sum+(mask1(k,m)\*C(k,m));

end

end

img1(i,j)=sum;

end

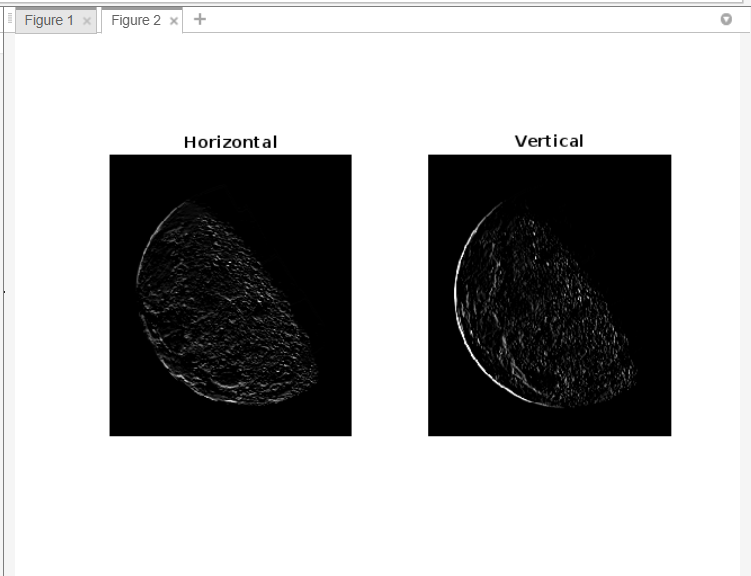
end

figure;

subplot(1,2,1); imshow(img); title('Horizontal');

subplot(1,2,2); imshow(img1); title('Vertical');

**OUTPUT:**

****

**CONCLUSION:**

From this practical, we learnt to implement different spatial domain sharpening filtering techniques on an image in MATLAB.

**Practical – 6**

**AIM:** Perform image compression using RLE algorithm. Analyse the effect on image size of e19 image.

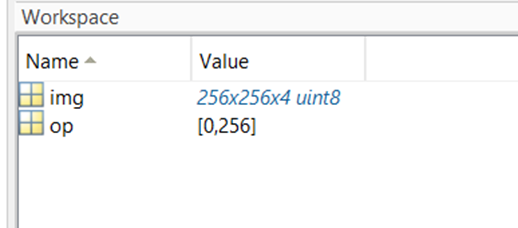
**CODE:**

img=imread("e19.tif");

op=rle(img);

imshow(op);

**OUTPUT:**

****

**CONCLUSION:**

From this practical, we learnt to implement image compression using RLE algorithm in MATLAB.

**Practical – 7**

**AIM:** After the uniform color of MS Dhoni and Ricky Ponting in e6 image

**CODE:**

% Load the RGB image

rgbImage = imread('e6.png');

% Convert RGB to HSV

hsvImage = rgb2hsv(rgbImage);

% Adjust the Hue value

hueShift = 0.58; % Shift the hue by 0.25

hsvImage(:,:,1) = mod(hsvImage(:,:,1) + hueShift, 1);

% Convert HSV back to RGB

newRgbImage = hsv2rgb(hsvImage);

% Display the new RGB image

subplot(1,2,1),imshow(rgbImage),title("Original");

subplot(1,2,2),imshow(newRgbImage),title("Converted");

**OUTPUT:**

****

**CONCLUSION:**

From this practical, we learnt to implement color changing techniques on an image in MATLAB.