UNIVERSITY OF MUMBAI

PRACTICAL ON

IMAGE PROCESSING

# SUBMITTED BY

NEHA RAMESH SAKHARE

Application Id:- 171330



D.T.S.S. College of Commerce

Subject Teacher name

# Mr. Chirag Deora

# MCA. I [2023-24]



UNIVERSITY OF MUMBAI

INSTITUTE OF DISTANCE AND OPEN LEARNING (IDOL)

CERTIFICATE

THE EXPERIMENTS DULY SIGNED IN THIS PROJECT REPORT REPRESENT THE BONAFIDE

WORK BY MISS. NEHA RAMESH SAKHARE APPLICATION ID / SEAT NO. 171330 IN

SEMESTER 1 OF FIRST YEAR OF MASTER OF COMPUTER APPLICATION (FYMCA

1YRS) OF PCP CENTER DTSS COLLEGE MALAD (EAST) FOR IMAGE PROCESSING PRACTICAL DURING THE ACADEMIC YEAR 2021-2022.

LECTURE IN CHARGE HEAD OF DEPARTMENT COURSE IN CHARGE

Examiner \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Date: 23 -01-2024 DTSS College Seal

INDEX

|  |  |  |  |
| --- | --- | --- | --- |
| MODULE NO. | PG. NO. | TOPIC | REMARK |
| 1 | 04 | Median Filter in MATLAB to remove Salt & Pepper noise. |  |
|  |  |  |
| 2 | 06 | MATLAB program for Deblur Images Using a Wiener  Filter |
|  |  |  |
| 3 | 07 | MATLAB program for Image Negation. |
|  |  |  |
| 4 | 08 | Edge Detection using Sobel, Prewitt and Roberts Operators. |
|  |  |  |
| 5 | 09 | MATLAB program for morphological operations on binary images. |
|  |  |  |
| 6 | 11 | Image Smoothening and Sharpening |
|  |  |  |
| 7 | 12 | MATLAB program for Scaling & Rotation Scaling (Resize). |
| 8 | 13 | MATLAB program for edge detection, gray level Thresholding in Image Segmentation. |
| 9 | 15 | Write a program on Discrete Cosine Transform. |

# PRACTICAL NO: -1

AIM: Median Filter in MATLAB to remove Salt & Pepper noise.

CODE: % MATLAB Code for removal of Salt and

% Pepper noise from image.

k=imread("gfglogo.png");

RemoveSaltAndPepperNoise(k);

function RemoveSaltAndPepperNoise(k)

% Convert to grayscale if not.

[M,N,D]=size(k);

if(D==3)

    k=rgb2gray(k);

end

% Add noise to image.

kn=imnoise(k,'salt & pepper',0.03);

% Display original and noisy image.

imtool(k,[]);

imtool(kn,[]);

% Denoising.

dn=medfilt2(kn,[5,5]);

% Display denoised image.

imtool(dn,[]);

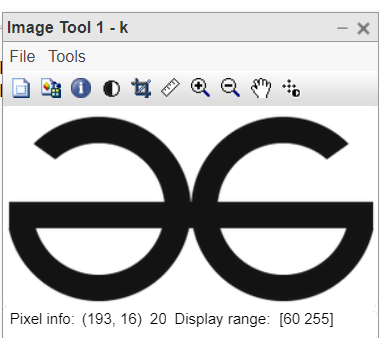
% Pause and close img windows.

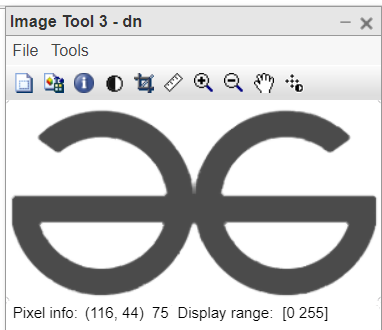
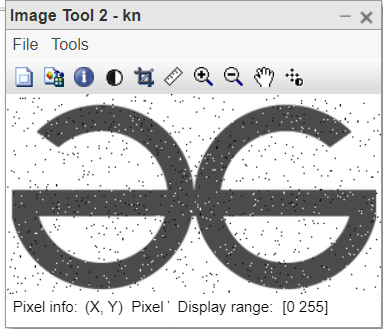
pause(10);

imtool close all;

end

OUTPUT:





PRACTICAL NO: -2

AIM: MATLAB program for Deblur Images Using a Wiener Filter.

Code: second.m

clc; clear all;close all;

Ioriginal = imread('flowers.jpg');subplot(1,3,1); imshow(Ioriginal); title('Original Image');

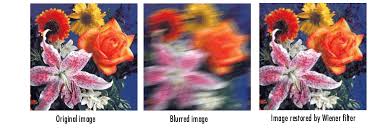
PSF = fspecial('motion',21,11);

Idouble = im2double(Ioriginal); blurred = imfilter(Idouble,PSF,'conv','circular'); subplot(1,3,2); imshow(blurred); title('Blurred Image');

wnr1 = deconvwnr(blurred,PSF);subplot(1,3,3); imshow(wnr1);

title('Restored Blurred Image');

OUTPUT: -



PRACTICAL NO: -3

AIM: MATLAB program for Image Negation.

Code: % reading the RGB file into the Matlab environment

skI = imread("sakura.jpg");

subplot(1, 2, 1),

% displaying the RGB image

imshow(skI);

title("Original image");

% levels of the 8-bit image

L = 2 ^ 8;

% finding the negative

neg = (L - 1) - skI;

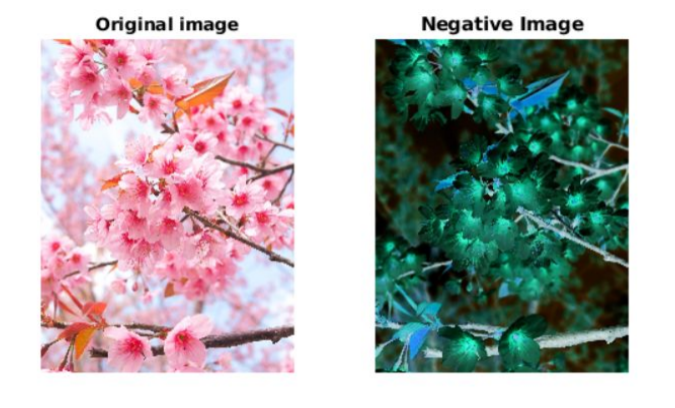
subplot(1, 2, 2),

% displaying the negative image

imshow(neg);

title("Negative Image")

Output:-

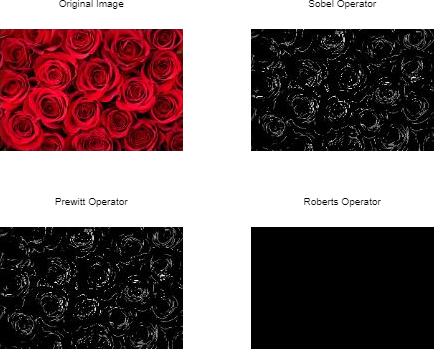
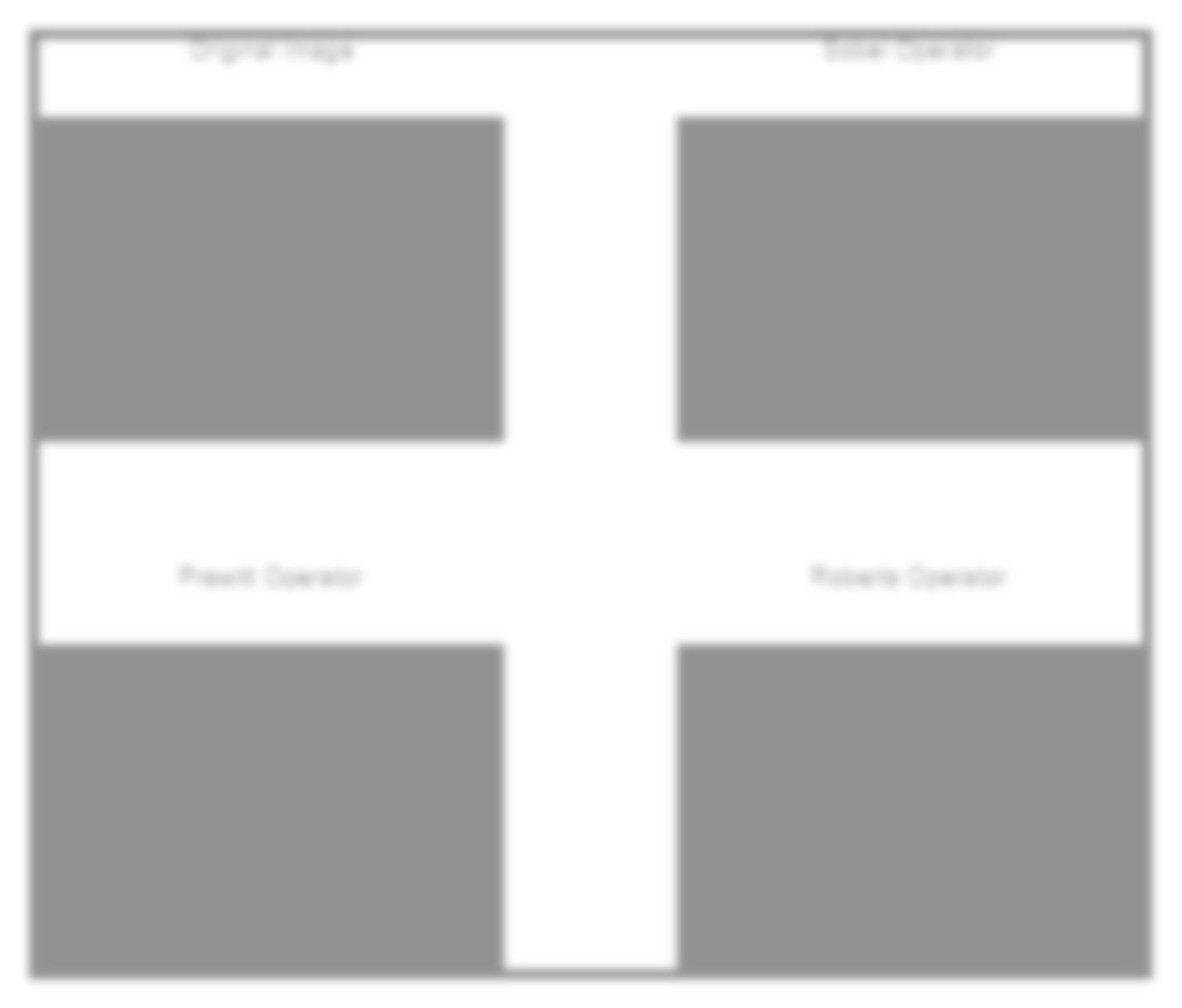


PRACTICAL NO: -4

AIM: Edge Detection using Sobel, Prewitt and Roberts Operators. Code : edge.m

clc; clear all; close all; a = imread('rose.jpg'); b = rgb2gray(a); subplot(2,2,1); imshow(a); title('Original Image'); c1 = edge(b,'sobel'); subplot(2,2,2); imshow(c1); title('Sobel Operator'); c2 = edge(b,'prewitt'); subplot(2,2,3); imshow(c2); title('Prewitt Operator'); c3 = edge(b,'roberts'); subplot(2,2,4); imshow(c3); title('Roberts Operator');

Output:



PRACTICAL NO: -5

AIM: MATLAB program for morphological operations on binary images.

Code: # Importing the image

I = imread("flowers.jpg");

subplot(2, 3, 1),

imshow(I);

title("Original image");

% Dilated Image

se = strel("line", 7, 7);

dilate = imdilate(I, se);

subplot(2, 3, 2),

imshow(dilate);

title("Dilated image");

% Eroded image

erode = imerode(I, se);

subplot(2, 3, 3),

imshow(erode);

title("Eroded image");

% Opened image

open = imopen(I, se);

subplot(2, 3, 4),

imshow(open);

title("Opened image");

% Closed image

close = imclose(I, se);

subplot(2, 3, 5),

imshow(close);

title("Closed image");

OUTPUT: -

# 

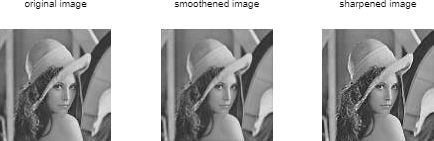
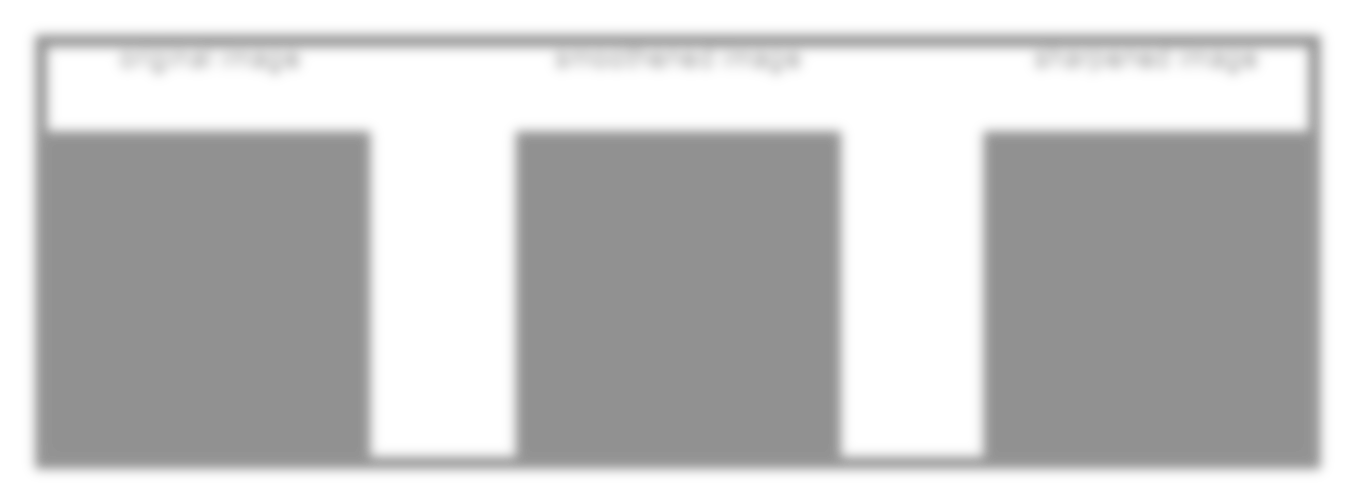
PRACTICAL NO: -6

# AIM: Image Smoothening and Sharpening

Code: Smoothening.m clc;

clear all; close all; a=imread('img.jpg'); subplot(1,3,1); imshow(a); title('original image'); h = fspecial('gaussian'); b = imfilter(a,h); subplot(1,3,2); imshow(b); title('smoothened image'); c = imsharpen(a); subplot(1,3,3); imshow(c); title('sharpened image');

OUTPUT:



# PRACTICAL NO: -7

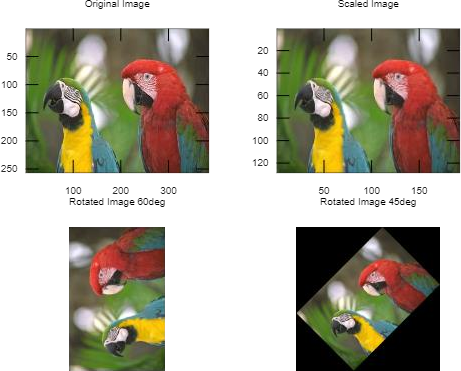
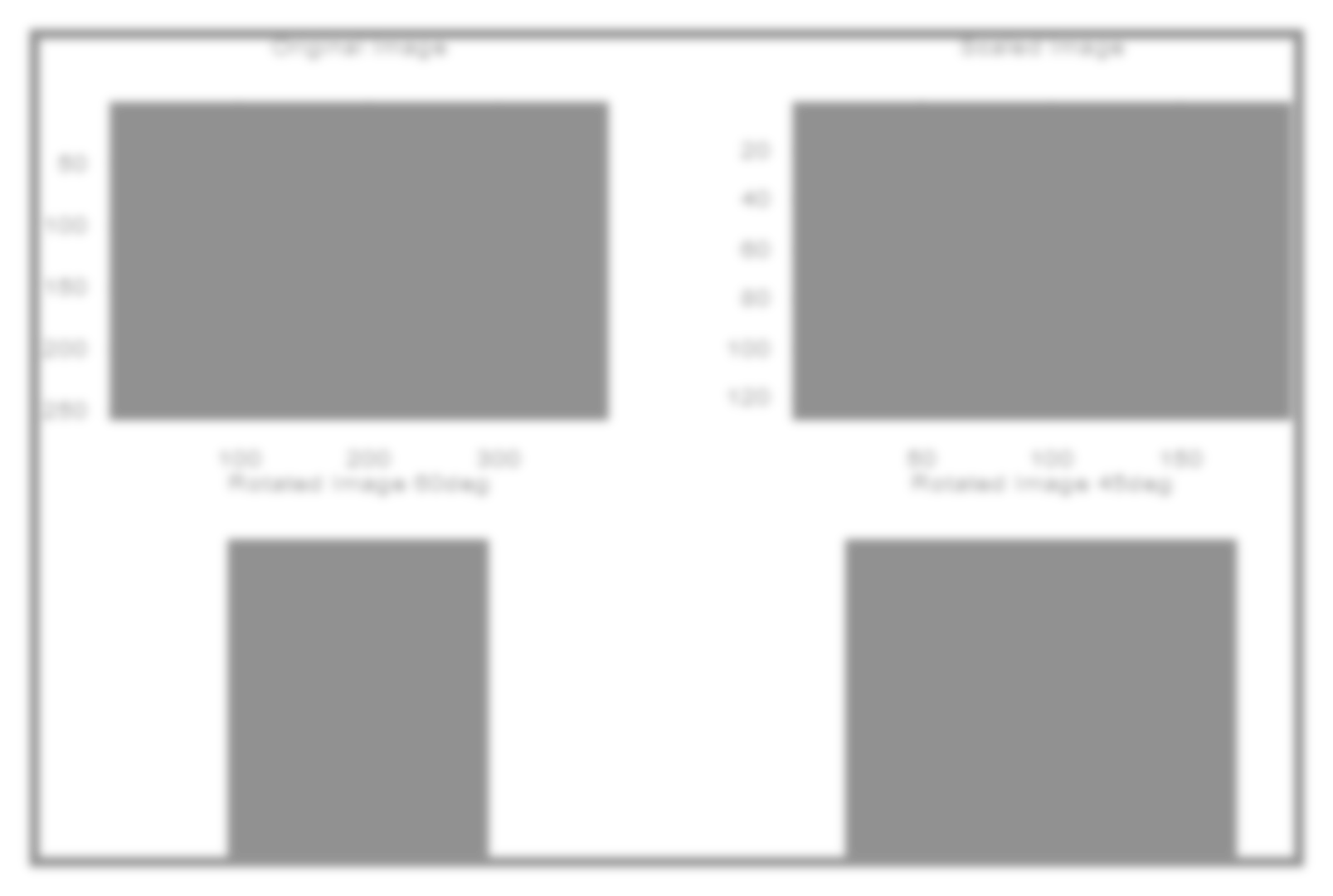
AIM: MATLAB program for Scaling & Rotation Scaling (Resize).

Code:

I=imread('bird.jpg'); subplot(2,2,1); subimage(I); title('Original Image'); s=input('Enter Scaling Factor'); j=imresize(I,s); subplot(2,2,2); subimage(j); title('Scaled Image'); K=imrotate(I,90); subplot(2,2,3); imshow(K); title('Rotated Image 90deg'); R=imrotate(I,45); subplot(2,2,4); imshow(R);

title('Rotated Image 45deg');

OUTPUT:



# PRACTICAL NO: -8

AIM: MATLAB program for edge detection, gray level Thresholding in Image Segmentation.

Code: -

% Following MATLAB function will take a grayscale

% or an RGB image as input and will return a

% binary image as output

function [binary] = convert2binary(img)

     [x, y, z]=size(img);

     % if Read Image is an RGB Image then convert

     % it to a Gray Scale Image For an RGB image

     % the value of z will be 3 and for a Grayscale

     % Image the value of z will be 1

    if z==3

         img=rgb2gray(img);

    end

    % change the class of image

    % array from 'unit8' to 'double'

    img=double(img);

    % Calculate sum of all the gray level

    % pixel's value of the GrayScale Image

    sum=0;

    for i=1:x

         for j=1:y

        sum=sum+img(i, j);

     end

     end

    % Calculate Threshold value by dividing the

    % calculated sum by total number of pixels

    % total number of pixels = rows\*columns (i.e x\*y)

    threshold=sum/(x\*y);

    % Create a image array having same number

    % of rows and column as Original image

    % with all elements as 0 (Zero).

    binary=zeros(x, y);

    % iterate over all the pixels of Grayscale

    % Image and Assign 1 to binary(i, j), if gray

    % level value is >=  threshold value

    % else assign 0 to binary(i, j) .

    for i=1:x

     for j=1:y

        if img(i, j) >= threshold

                binary(i, j) = 1;

        else

            binary(i, j)=0;

        end

     end

    end

end

% driver function

% Read the target Image

img=imread('apple.png');

% Call convert2binary() function to convert

% Image to binary using thresholding

binary\_image=convert2binary(img);

% Display result

imshow(binary\_image);

OUTPUT: -