Digital Image Processing

Practical file

**Submitted by,**Divya A.   
College Roll No: 20201457  
Exam Roll No: 20016570005  
BSc (Hons) Computer Science

Ramanujan College

**Submitted to,**Ms. Bhavya Ahuja Grover

Assistant Professor  
Dept. of Computer Science

Ramanujan College

University of Delhi

**INDEX**

|  |  |  |
| --- | --- | --- |
| S. NO. | PROLOGUE | PAGE |
| 1. | Q1. Write program to read and display digital image using MATLAB or SCILAB a. Become familiar with SCILAB/MATLAB Basic commands b. Read and display image in SCILAB/MATLAB c. Resize given image d. Convert given color image into gray-scale image e. Convert given color/gray-scale image into black & white image f. Draw image profile g. Separate color image in three R G & B planes h. Create color image using R, G and B three separate planes i. Flow control and LOOP in SCILAB j. Write given 2-D data in image file | 1-7 |
| 2. | Q2. To write and execute image processing programs using point processing method a. Obtain Negative image b. Obtain Flip image c. Thresholding d. Contrast stretching | 8-11 |
| 3. | Q3. To write and execute programs for image arithmetic operations a. Addition of two images b. Subtract one image from other image c. Calculate mean value of image | 12-13 |
| 4. | Q4. To write and execute programs for image logical operations a. AND operation between two images b. OR operation between two images c. Calculate intersection of two images d. NOT operation (Negative image) | 14-16 |
| 5. | Q5. To write a program for histogram calculation and equalization using a. Standard MATLAB function b. Program without using standard MATLAB functions | 16-20 |
| 6. | Q6. To write and execute program for geometric transformation of image a. Translation b. Scaling c. Rotation d. Shrinking e. Zooming | 20-24 |
| 7. | Q7. To understand various image noise models and to write programs for a. image restoration b. Remove Salt and Pepper Noise c. Minimize Gaussian noise d. Median filter | 25-28 |
| 8. | Q8. Write and execute programs to use spatial low pass and high pass filters | 29-30 |
| 9 | Q9. Write and execute programs for image frequency domain filtering a. Apply FFT on given image b. Perform low pass and high pass filtering in frequency domain c. Apply IFFT to reconstruct image | 31-33 |
| 10 | Q10. Write a program in C and MATLAB/SCILAB for edge detection using different edge detection mask | 33-35 |
| 11 | Q11. Write and execute program for image morphological operations erosion and dilation. | 35-38 |

**Q1. Write program to read and display digital image using MATLAB or SCILAB**

**a. Become familiar with SCILAB/MATLAB Basic commands**

disp('2 \* 3 - 4 + 8 / 3 \ 9 =', 2\*3-4+8/3\9);

x = linspace(0,8,100);

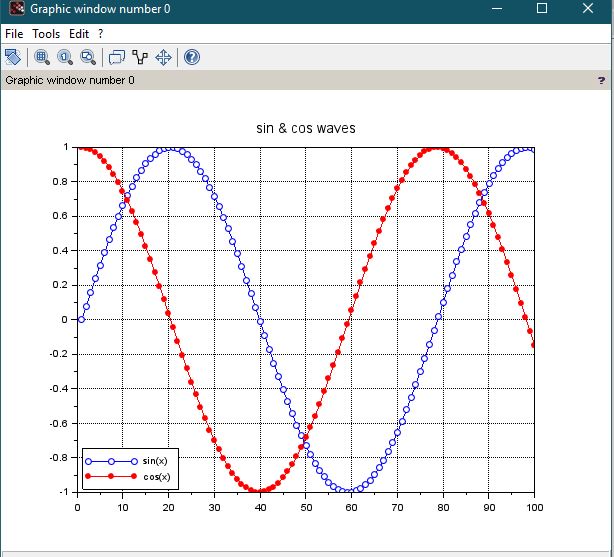
plot(sin(x),'o-');

plot(cos(x),'r.-');

xtitle('sin & cos waves');

legend('sin(x)','cos(x)', 3);

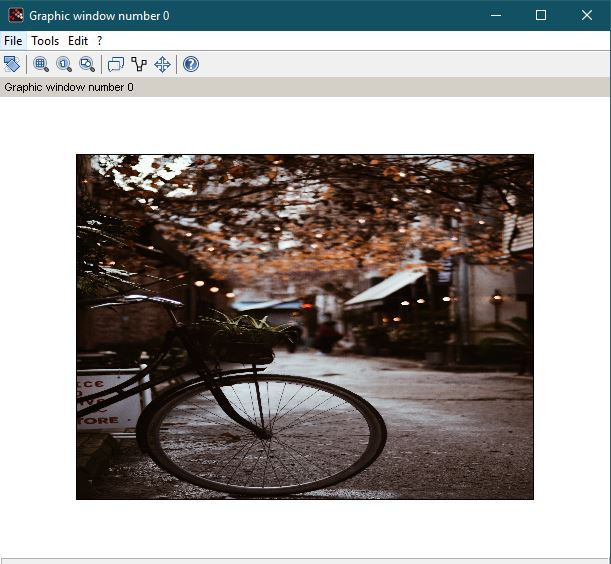
xgrid(0,1,7);



**b. Read and display image in SCILAB/MATLAB**

image=imread("F:\CS\sem 5\Digital Image Processing\Pracicals\Pictures\cycle.jpg");

imshow(image);

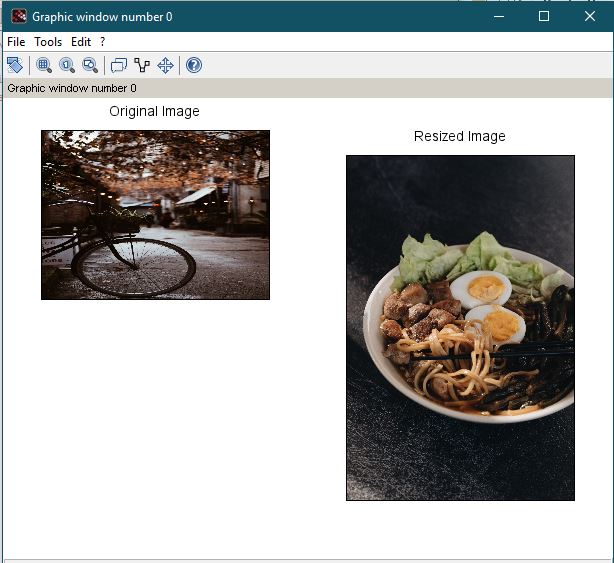


**c. Resize given image**

resize = imresize(img,2);

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

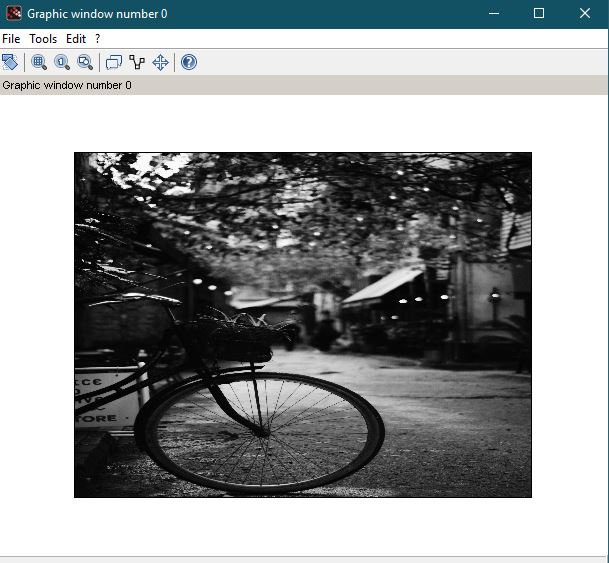
subplot(1,2,2), title('Resized Image'), imshow(resize);



**d. Convert given color image into gray-scale image**

gray=rgb2gray(image);

imshow(gray);



**e. Convert given color/gray-scale image into black & white image**

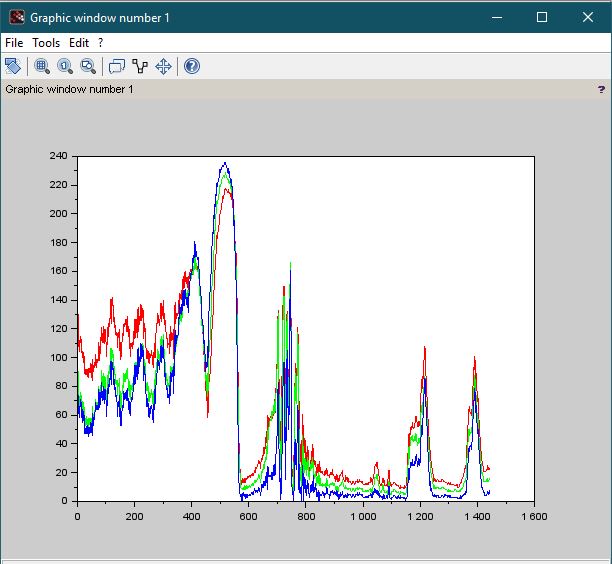
bw=im2bw(gray,0.5);

imshow(bw);



**f. Draw image profile**

improfile(image);



**g. Separate color image in three R, G & B planes**

[r,c] = size(image); //

all\_black = zeros(r,c,'uint8');

red\_img = cat(3, image(:,:,1), all\_black, all\_black);

green\_img = cat(3, all\_black, image(:,:,2), all\_black);

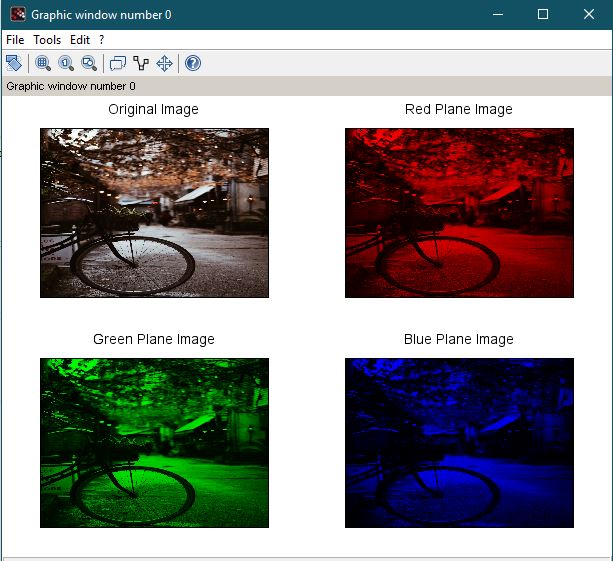
blue\_img = cat(3, all\_black, all\_black, image(:,:,3));

subplot(2,2,1),title("Original Image"),imshow(image);

subplot(2,2,2),title("Red Plane Image"),imshow(red\_img);

subplot(2,2,3),title("Green Plane Image"),imshow(green\_img);

subplot(2,2,4),title("Blue Plane Image"),imshow(blue\_img);



**h. Create color image using R, G and B three separate planes**

merged = red\_img + green\_img + blue\_img;

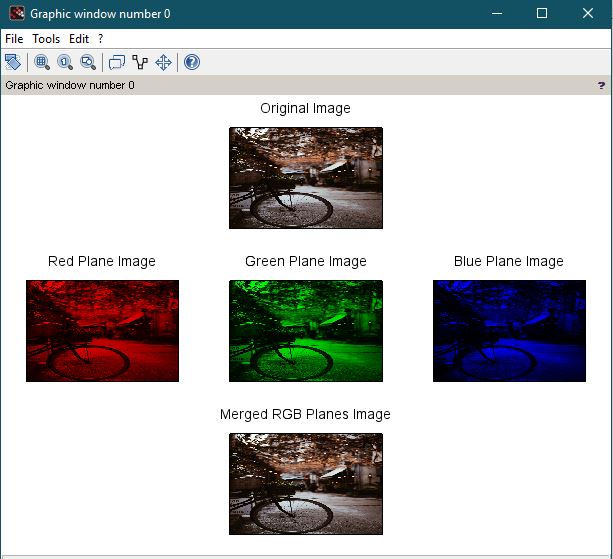
subplot(3,3,2),title("Original Image"), imshow(image);

subplot(3,3,4),title("Red Plane Image"), imshow(red\_img);

subplot(3,3,5),title("Green Plane Image"), imshow(green\_img);

subplot(3,3,6),title("Blue Plane Image"), imshow(blue\_img);

subplot(3,3,8),title("Merged RGB Planes Image"), imshow(merged);



**i. Flow control and LOOP in SCILAB**

for i=0:10

disp(i)

end



**j. Write given 2-D data in image file**

mat = zeros(20,20,'uint8');

mat(5:10, 5:10) = 1; // box

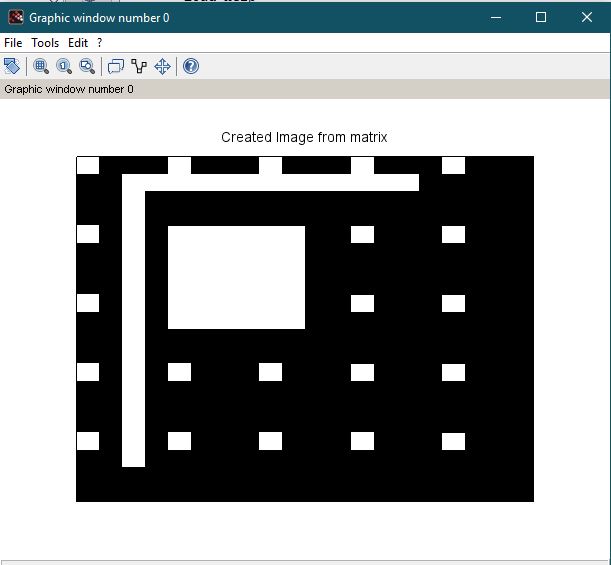
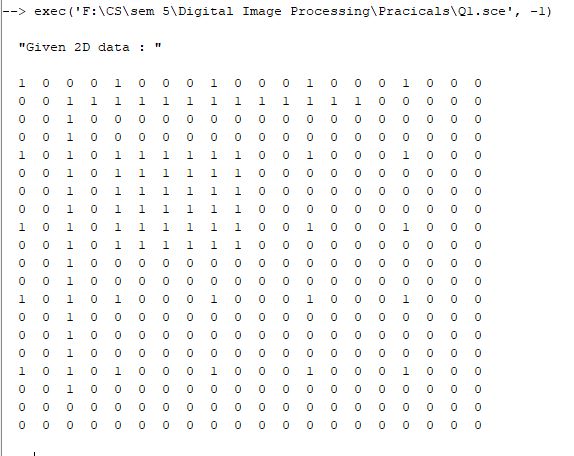
mat(2, 3:15) = 1; //vertical line

mat(2:18, 3) = 1; //horizontal line

mat(1:4:20,1:4:20) = 1; // linear points

disp('Given 2D data : ', mat);

title('Created Image from matrix'), imshow(mat2gray(mat));



**Q2. To write and execute image processing programs using point processing method**

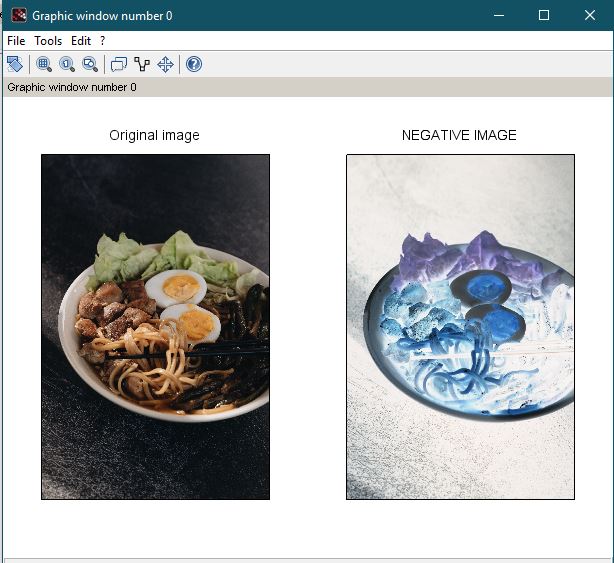
**a. Obtain Negative image**

img=imread("F:\CS\sem 5\Digital Image Processing\Pracicals\Pictures\Ramen.jpg");

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

neg=255-img;

subplot(1,2,2),title("NEGATIVE IMAGE"), imshow(neg);



**b. Obtain Flip image**

img=imread("F:\CS\sem 5\Digital Image Processing\Pracicals\Pictures\Kitten.jpg");

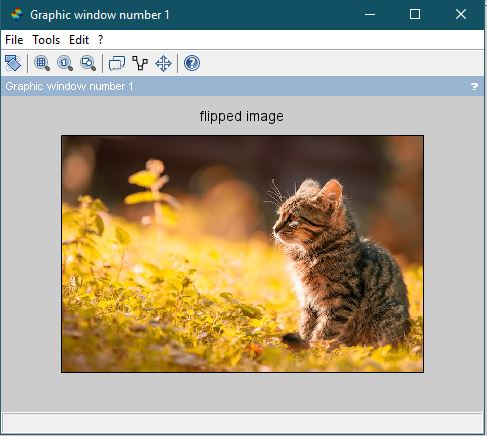
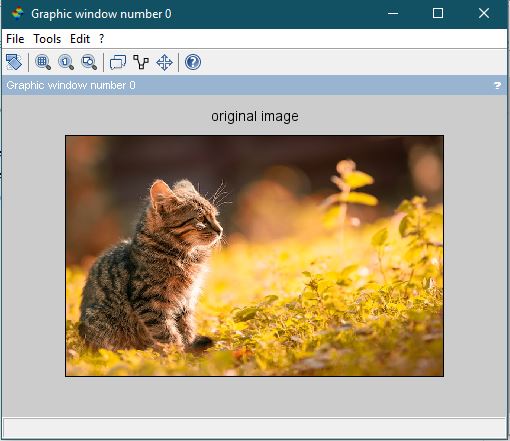
figure, imshow(img);

title("original image");

flip\_img=flipdim(img,2);

figure, imshow(flip\_img);

title("flipped image");



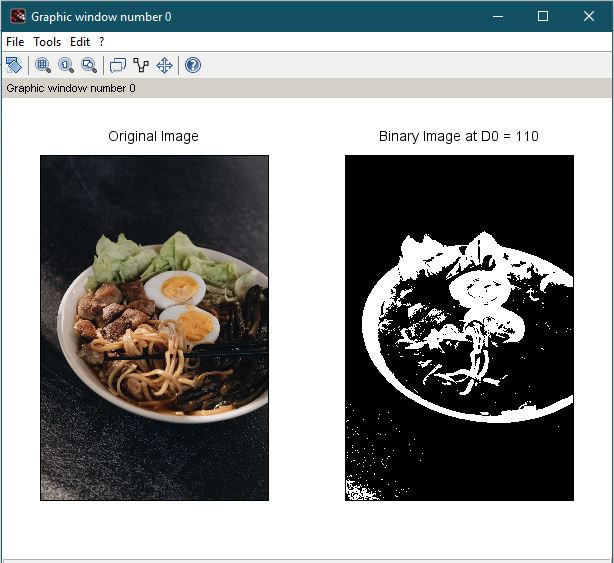
**c. Thresholding**

c2 = imread("F:\CS\sem 5\Digital Image Processing\Pracicals\Pictures\Ramen.jpg");

subplot(1,2,1), title('Original Image'), imshow(c2);

thres = im2bw(c2, 110/256)

subplot(1,2,2), title('Binary Image at D0 = 110'), imshow(thres);



**d. Contrast stretching**

c2 = imread("F:\CS\sem 5\Digital Image Processing\Pracicals\Pictures\Lena\_dark.png");

subplot(1,2,1), title('Original Image'), imshow(c2);

funcprot(0);

function image = contrast\_stretch(gray\_img)

a = min(gray\_img(:));

b = max(gray\_img(:));

image = (gray\_img - a)\*(255/(b-a));

endfunction;

c\_sam = contrast\_stretch(c2);

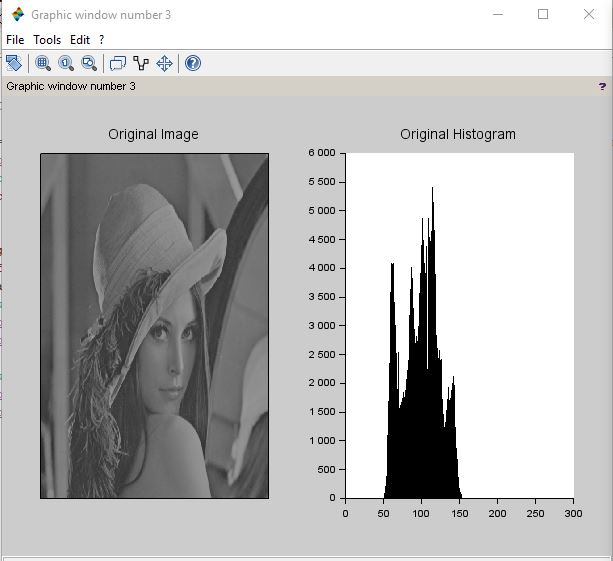
figure();

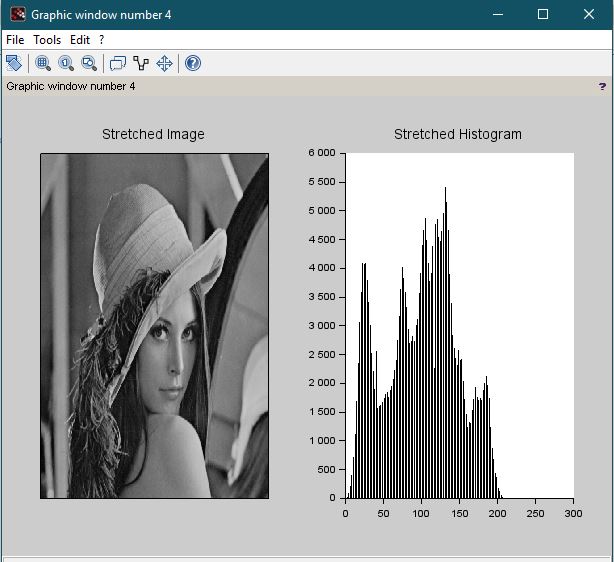
subplot(1,2,1),title("Original Image "),imshow(c2);

subplot(1,2,2),title("Original Histogram"),imhist(c2,[],1);

figure();

subplot(1,2,1),title("Stretched Image"),imshow(c\_sam);

subplot(1,2,2),title("Stretched Histogram"), plot2d3(imhist(c\_sam)); 



**3. To write and execute programs for image arithmetic operations**

**a. Addition of two images**

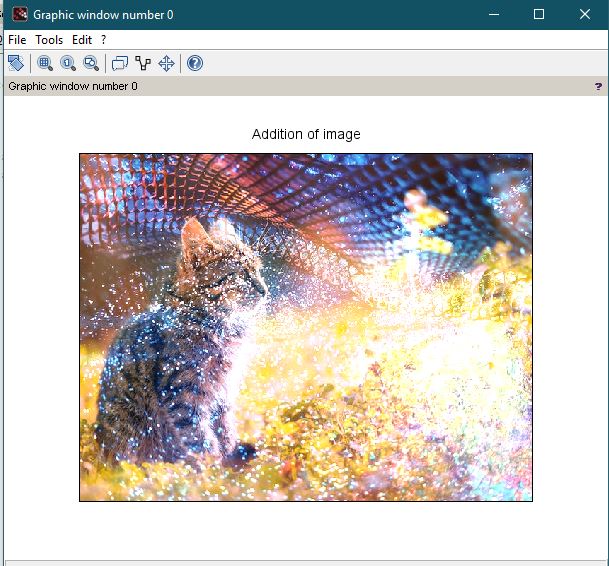
x=imread("F:\CS\sem 5\Digital Image Processing\Pracicals\Pictures\Kitten.jpg");

y=imread("F:\CS\sem 5\Digital Image Processing\Pracicals\Pictures\Glitter.jpg");

z=imadd(x,y);

title("Addition of image");

imshow(z);



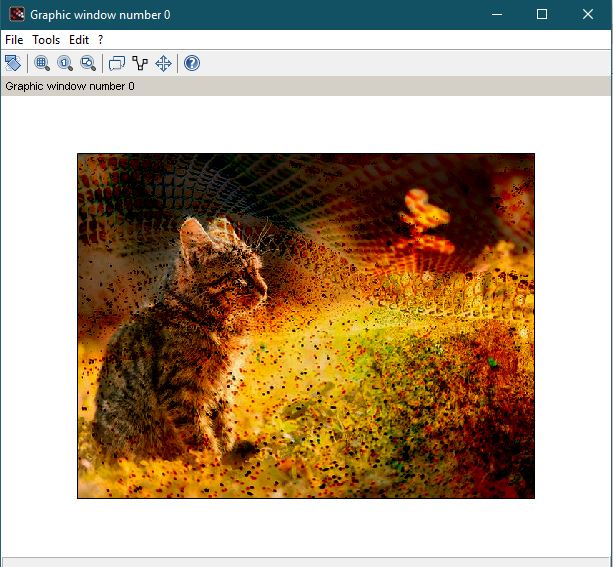
**b. Subtract one image from other image**

x=imread("F:\CS\sem 5\Digital Image Processing\Pracicals\Pictures\Kitten.jpg");

y=imread("F:\CS\sem 5\Digital Image Processing\Pracicals\Pictures\Glitter.jpg");

z=imsubtract(x,y);

imshow(z);

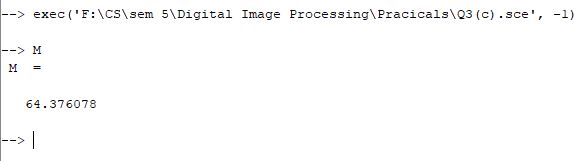


**c. Calculate mean value of image**

A=imread("F:\CS\sem 5\Digital Image Processing\Pracicals\Pictures\Cycle.jpg");

B=rgb2gray(A);

M=mean2(B);



**Q4. To write and execute programs for image logical operations**

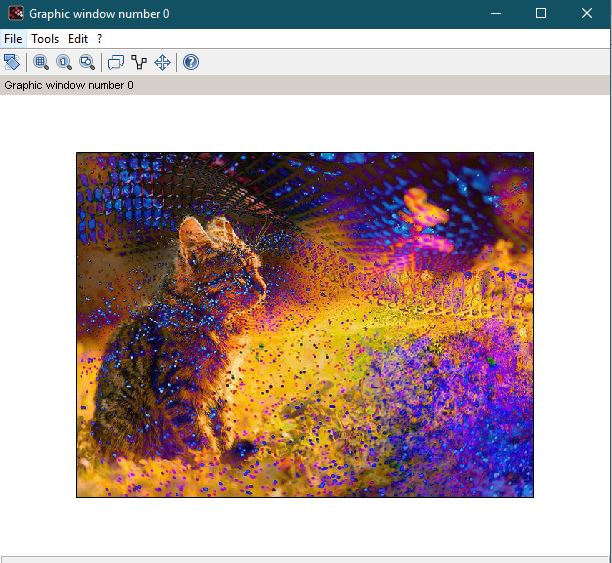
**a. AND operation between two images**

A=imread("F:\CS\sem 5\Digital Image Processing\Pracicals\Pictures\Kitten.jpg");

B=imread("F:\CS\sem 5\Digital Image Processing\Pracicals\Pictures\Glitter.jpg");

andimage=imabsdiff(A,B);

imshow(andimage);



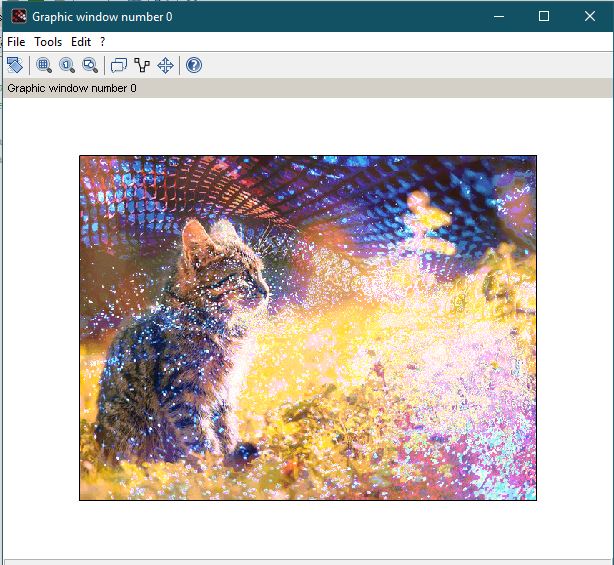
**b. OR operation between two images**

A=imread("F:\CS\sem 5\Digital Image Processing\Pracicals\Pictures\Kitten.jpg");

B=imread("F:\CS\sem 5\Digital Image Processing\Pracicals\Pictures\Glitter.jpg");

orimage=bitor(A,B);

imshow(orimage);



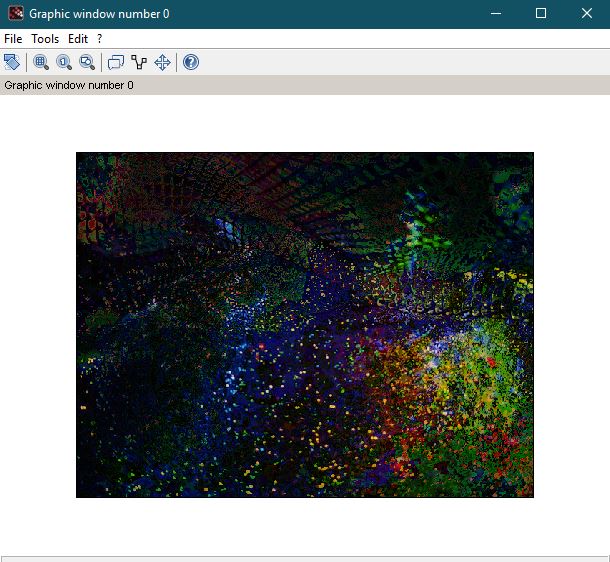
**c. Calculate intersection of two images**

A=imread("F:\CS\sem 5\Digital Image Processing\Pracicals\Pictures\Kitten.jpg");

B=imread("F:\CS\sem 5\Digital Image Processing\Pracicals\Pictures\Glitter.jpg");

intersection=bitand(A,B);

imshow(intersection);

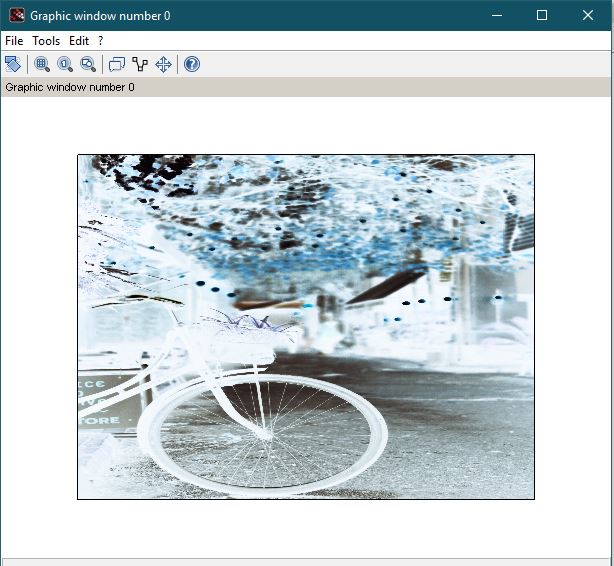


**d. NOT operation (Negative image)**

A=imread("F:\CS\sem 5\Digital Image Processing\Pracicals\Pictures\cycle.jpg");

not=~(A);

imshow(not);



**5. To write a program for histogram calculation and equalization using**

**a. Standard MATLAB function**

A=imread("F:\CS\sem 5\Digital Image Processing\Pracicals\Pictures\Ramen.jpg");

figure(), imshow(A), title("Original Image");

figure();

Agray=rgb2gray(A);

title("Grayscale Image"), imshow(Agray)

figure()

ori\_his=imhist(Agray, [], 1);

title("Original Histogram"), imshow(ori\_his);

figure();

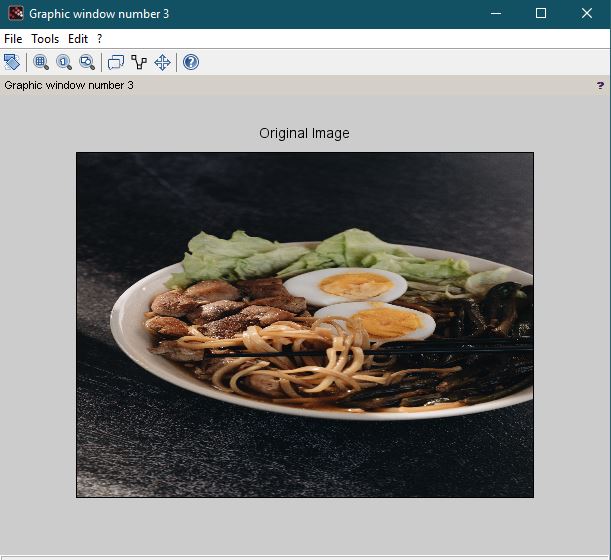
hist=imhistequal(Agray);

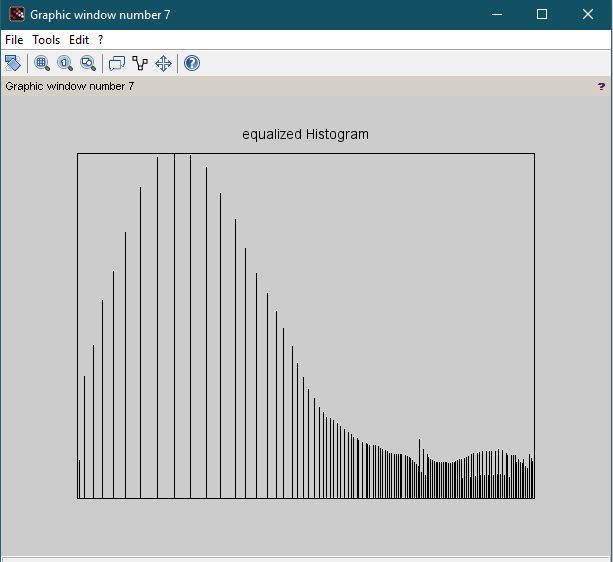
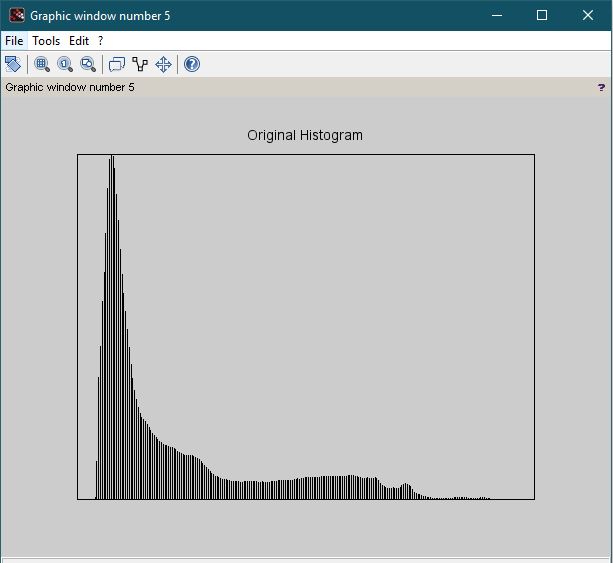
title("Equalised image"), imshow(hist);

figure()

equal\_his=imhist(hist, [], 1);

title("equalized Histogram"), imshow(equal\_his);





**b. Program without using standard MATLAB functions**

g\_img = rgb2gray(imread('C:\Users\RMC-1\Documents\DIP-Divya\Pictures\Kitten.jpg'));

figure(),

subplot(1,2,1),title("Original Image "),imshow(g\_img);

subplot(1,2,2),title("Original Histogram"),imhist(g\_img,[],1);

/\* Algorithm \*/

function eq\_img = histeq(g\_img)

[freq, bins] = imhist(g\_img,256);

bins = 255;

[mr, nc] = size(g\_img);

freq = cumsum(freq);

npixels = prod(size(g\_img));

output = round(bins.\*(freq./npixels));

// Creating Equalized Image

for i = 1:mr

for j = 1:nc

eq\_img(i,j) = output(g\_img(i,j) + 1);

end

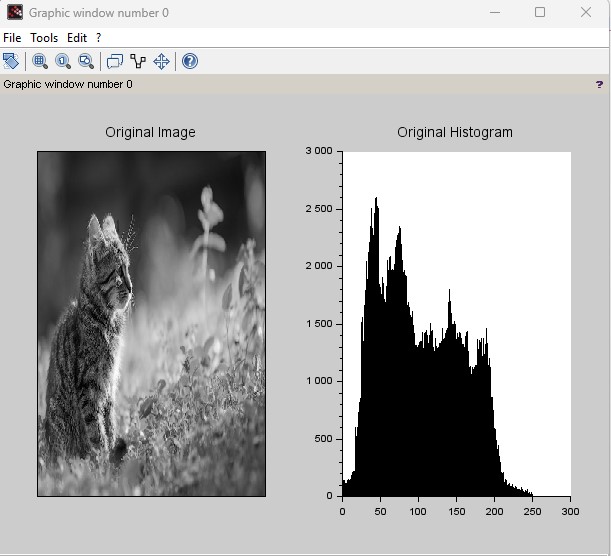
end

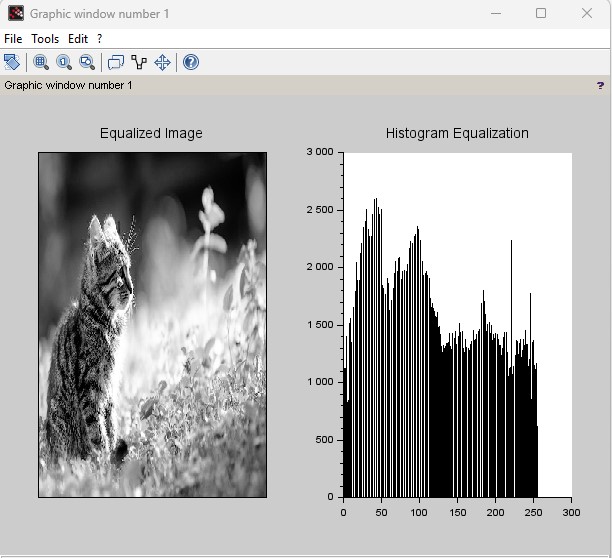
endfunction

he\_img = uint8(histeq(g\_img));

figure(),

subplot(1,2,1),title("Equalized Image "),imshow(he\_img);

subplot(1,2,2),title("Histogram Equalization"),imhist(he\_img, [], 1); 



**6. To write and execute program for geometric transformation of image**

**a. Translation**

S1 = imread('F:\CS\sem 5\Digital Image Processing\Pracicals\Pictures\kitten.jpg');

// Translation for x = 20

mat = [ 1 0 0; 0 1 0; 20 0 1];

S2 = imtransform(S1,mat,'affine');

// Translation for y = -20

mat(3, 1:2) = [0 -20];

S3 = imtransform(S1,mat,'affine');

// 'Translation for (-20, 30)'

mat(3, 1:2) = [-20 30];

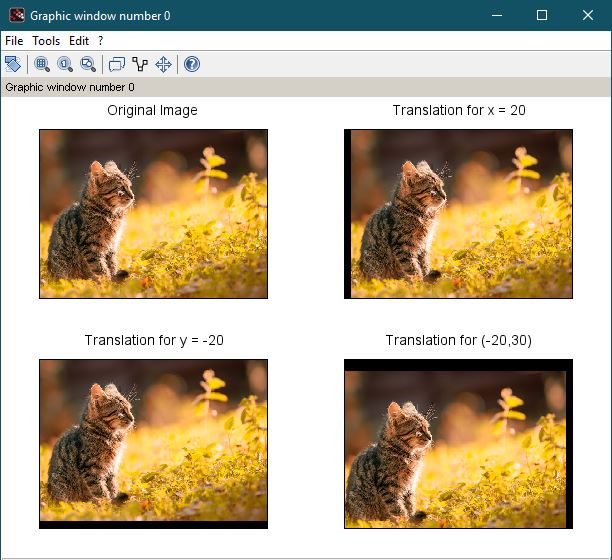
S4 = imtransform(S1,mat,'affine');

subplot(2,2,1), title('Original Image'), imshow(S1);

subplot(2,2,2), title('Translation for x = 20'), imshow(S2);

subplot(2,2,3), title('Translation for y = -20'), imshow(S3);

subplot(2,2,4), title('Translation for (-20,30)'), imshow(S4);



**b. Scaling**

s\_img = imread('F:\CS\sem 5\Digital Image Processing\Pracicals\Pictures\ramen.jpg');

width = size(s\_img, 'c'); // column pixels = width

height = size(s\_img, 'r'); // row pixels = height

// Scaling width by 2

mat = [ 2 0; 0 1; 0 0];

sc1 = imtransform(s\_img, mat, 'affine', width\*mat(1), height\*mat(5));

// Scaling height by 2

mat([1,5]) = [1 2];

sc2 = imtransform(s\_img, mat, 'affine', width\*mat(1), height\*mat(2));

// Scaling image by 2

mat([1,5]) = [2 2];

sc3 = imtransform(s\_img, mat, 'affine', width\*mat(1), height\*mat(2));

function s = str(img)

s = 'Size : ' + strcat(string(size(img)), ' \* ');

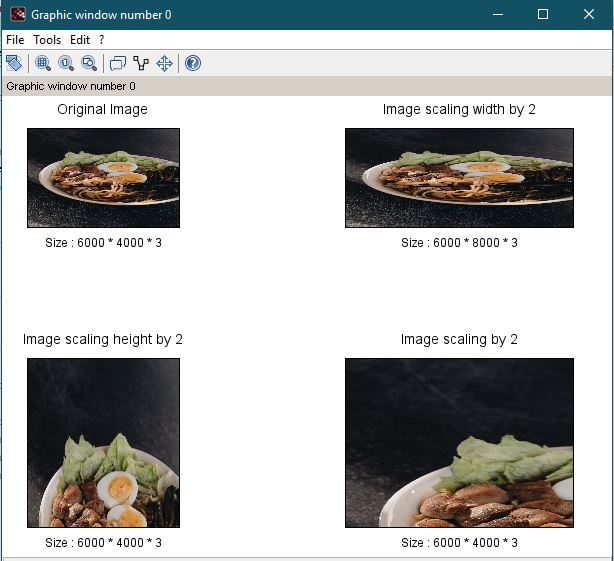
endfunction;

subplot(3,3,1), title('Original Image'), xlabel(str(s\_img)), imshow(s\_img);

subplot(3,2,2), title('Image scaling width by 2'),xlabel(str(sc1)), imshow(sc1);

subplot(2,3,4), title('Image scaling height by 2'),xlabel(str(sc2)), imshow(sc2);

subplot(2,2,4), title('Image scaling by 2'),xlabel(str(sc3)), imshow(sc3);



**c. Rotation**

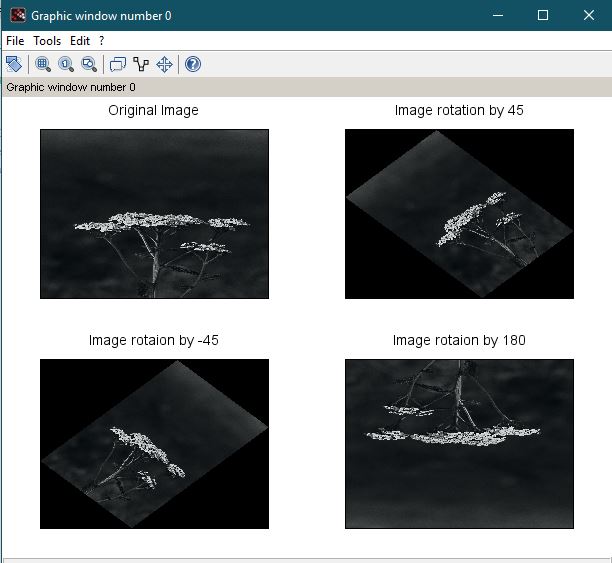
s\_img = imread('F:\CS\sem 5\Digital Image Processing\Pracicals\Pictures\Gray.jpg');

subplot(2,2,1), title('Original Image'), imshow(s\_img);

subplot(2,2,2), title('Image rotation by 45'), imshow(imrotate(s\_img, 45));

subplot(2,2,3), title('Image rotaion by -45'), imshow(imrotate(s\_img, -45));

subplot(2,2,4), title('Image rotaion by 180'), imshow(imrotate(s\_img, 180));



**d. Shrinking**

img=rgb2gray(imread("C:\Users\RMC-1\Documents\DIP-Divya\Pictures\Kitten.jpg"));

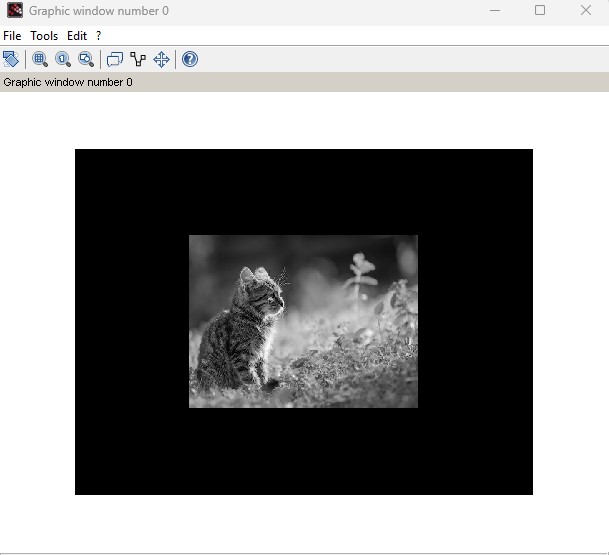
[r c]=size(img);

m=zeros(r,c,'uint8');

j=imresize(img, 0.5);

m(114:342,160:479)=j(:,:);

imshow(m)



**e. Zooming**

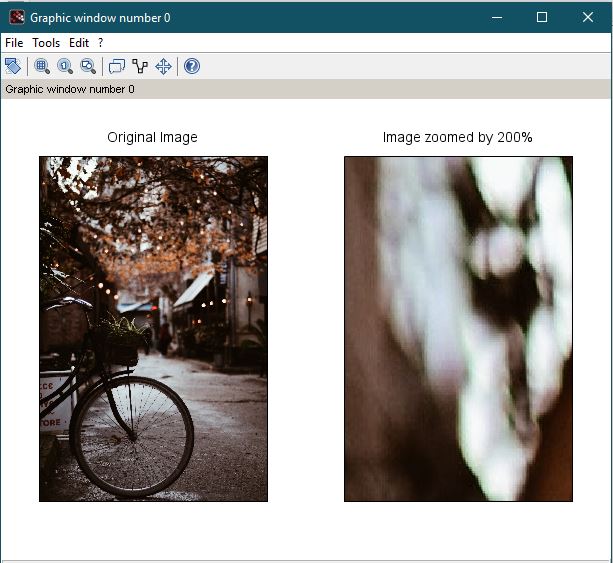
s\_img = imread('F:\CS\sem 5\Digital Image Processing\Pracicals\Pictures\cycle.jpg');

f = 2;

im2 = imresize(s\_img, f);

subplot(121), title('Original Image'), imshow(s\_img);

subplot(122), title('Image zoomed by 200%'), imshow(im2(96:287, 81:241, :));



**7. To understand various image noise models and to write programs for**

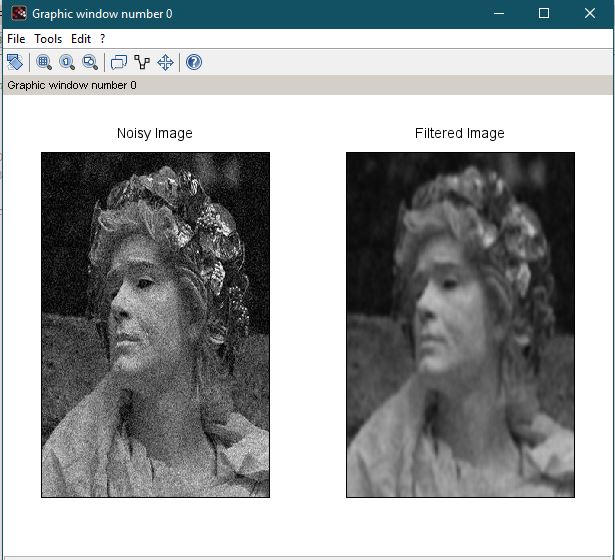
**a. image restoration**

im1 = imread('F:\CS\sem 5\Digital Image Processing\Pracicals\Pictures\noisy.jpg');

f = fspecial('gaussian', [8, 8], 2);

subplot(121), title('Noisy Image'), imshow(im1);

subplot(122), title('Filtered Image'), imshow(imfilter(im1, f));



**b. Remove Salt and Pepper Noise**

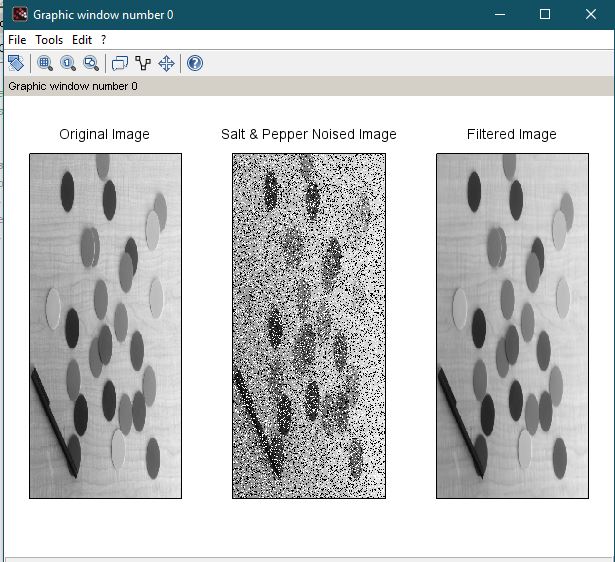
im2 = rgb2gray(imread('F:\CS\sem 5\Digital Image Processing\Pracicals\Pictures\coloredChips.png'));

im3 = imnoise(im2, 'salt & pepper', 0.3);

subplot(131), title('Original Image'), imshow(im2);

subplot(132), title('Salt & Pepper Noised Image'), imshow(im3);

subplot(133), title('Filtered Image'), imshow(immedian(im2,3));



**c. Minimize Gaussian noise**

im1 = imread('F:\CS\sem 5\Digital Image Processing\Pracicals\Pictures\cycle.jpg');

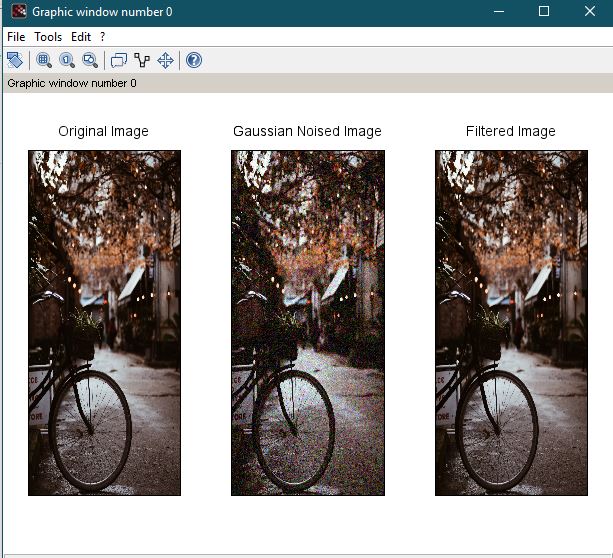
im2 = imnoise(im1, 'gaussian');

f = fspecial('average', 3);

subplot(131), title('Original Image'), imshow(im1);

subplot(132), title('Gaussian Noised Image'), imshow(im2);

subplot(133), title('Filtered Image'), imshow(imfilter(im1, f));



**d. Median filter**

im2 = rgb2gray(imread(fullpath(getIPCVpath() + 'images/baboon.png')));

d\_im = imnoise(im2, 'salt & pepper', 0.25);

[r c] = size(d\_im);

img1 = zeros(r+2, c+2, 'uint8');

img1(2:r+1, 2:c+1) = d\_im(:,:);

// border padded image

img1(1, 1) = d\_im(1, 1);

img1(r+2, 1) = d\_im(r, 1);

img1(1, c+2) = d\_im(1, c);

img1(r+2, c+2) = d\_im(r, c);

img1(2:r+1, 1) = d\_im(:,1);

img1(2:r+1, c+2) = d\_im(:,c);

img1(1, 2:c+1) = d\_im(1,:);

img1(r+2, 2:c+1) = d\_im(r,:);

for i = 2:r+1

for j = 2:c+1

img1(i,j) = gsort(img1(i-1:i+1, j-1:j+1))(5);

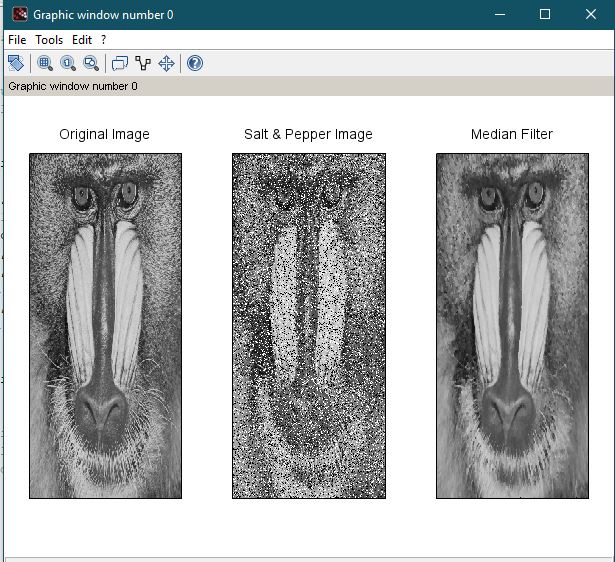
end

end

subplot(131), title('Original Image'), imshow(im2);

subplot(132), title('Salt & Pepper Image'), imshow(d\_im);

subplot(133), title("Median Filter"), imshow(img1(2:r+1, 2:c+1));



**8. Write and execute programs to use spatial low pass and high pass filters**

/\* Spatial Low Pass Filter \*/

i1 = imread('F:\CS\sem 5\Digital Image Processing\Pracicals\Pictures\einstein.jpg');

g\_filter = fspecial('gaussian');

i2 = imfilter(i1, g\_filter);

g\_filter2 = fspecial('gaussian', [8,8], 10);

i3 = imfilter(i1, g\_filter2);

g\_filter3 = fspecial('gaussian',[25,25], 31);

i4 = imfilter(i1, g\_filter3);

subplot(2,2,1), title('Original Image'), imshow(i1);

subplot(2,2,2), title('Default Gaussian kernel'), imshow(i2);

subplot(2,2,3), title('Gaussian kernel with 8 \* 8 with sigma = 10'), imshow(i3);

subplot(2,2,4), title('Gaussian kernel with 25 \* 25 with sigma = 31'), imshow(i4);

//highpass

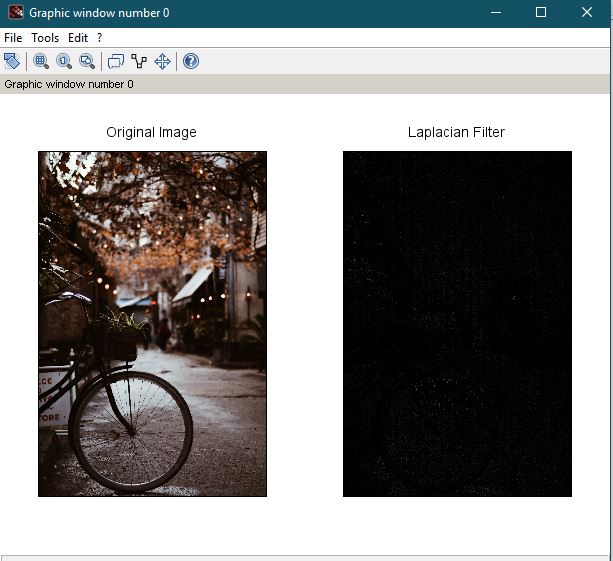
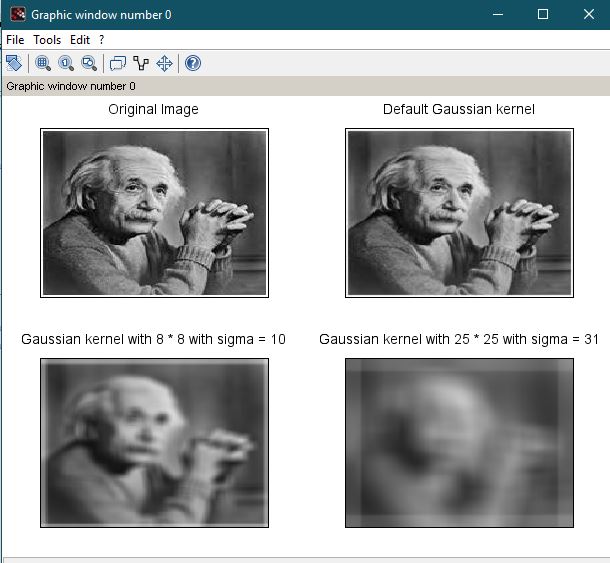
i1 = imread('F:\CS\sem 5\Digital Image Processing\Pracicals\Pictures\cycle.jpg');

l\_filter = fspecial('laplacian');

i2 = imfilter(i1, l\_filter);

subplot(1,2,1), title('Original Image'), imshow(i1);

subplot(1,2,2), title('Laplacian Filter'), imshow(i2);



**9. Write and execute programs for image frequency domain filtering**

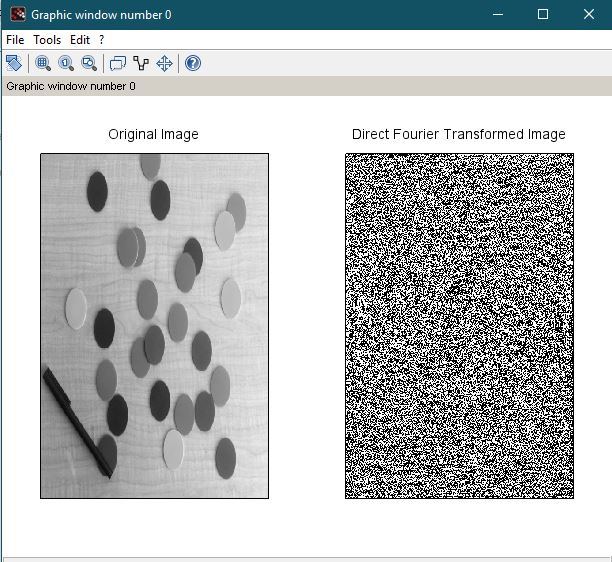
**a. Apply FFT on given image**

img = rgb2gray(imread('F:\CS\sem 5\Digital Image Processing\Pracicals\Pictures\coloredChips.png'));

ft\_img = fft(double(img));

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

subplot(1,2,2), title('Direct Fourier Transformed Image'),imshow(ft\_img);



**b. Perform low pass and high pass filtering in frequency domain**

// Butterworth Filters

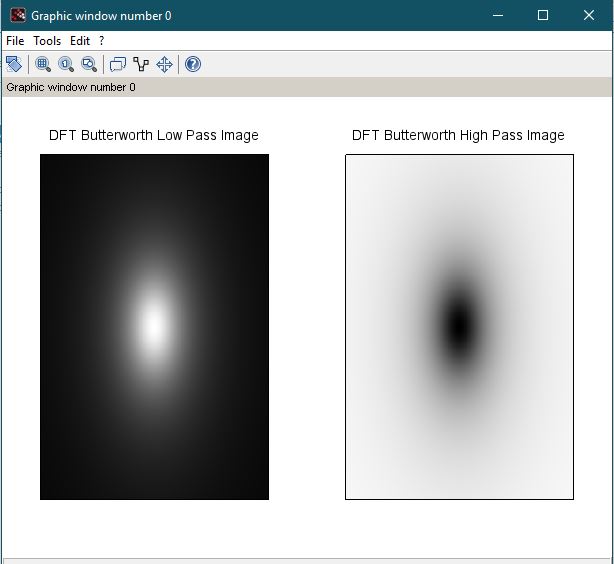
img = rgb2gray(imread('F:\CS\sem 5\Digital Image Processing\Pracicals\Pictures\coloredChips.png'));

G11 = mkfftfilter(img, 'butterworth1', 0.3);

H11 = 1 - G11;

subplot(121), title('DFT Butterworth Low Pass Image'), imshow(G11);

subplot(122), title('DFT Butterworth High Pass Image'),imshow(H11);



**c. Apply IFFT to reconstruct image**

img = rgb2gray(imread('F:\CS\sem 5\Digital Image Processing\Pracicals\Pictures\coloredChips.png'));

S2 = ft\_img .\* fftshift(G11);

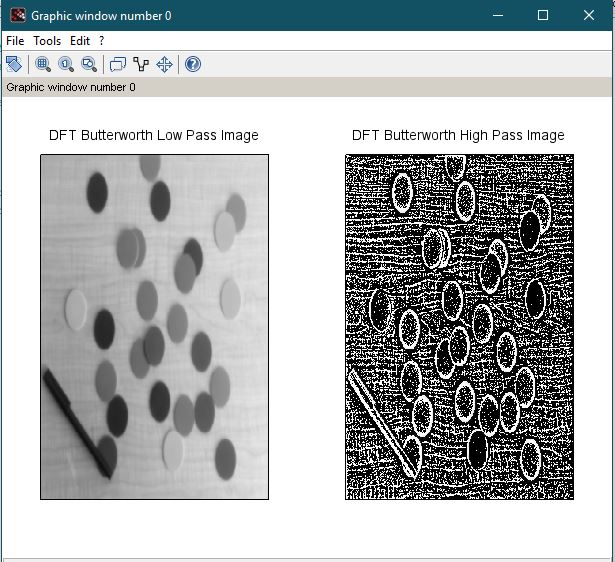
bwh\_l = uint8(ifft(S2));

S2 = ft\_img .\* fftshift(H11);

bwh\_h = uint8(ifft(S2));

subplot(121), title('DFT Butterworth Low Pass Image'), imshow(bwh\_l);

subplot(122), title('DFT Butterworth High Pass Image'),imshow(bwh\_h);



**10. Write a program in C and MATLAB/SCILAB for edge detection using different edge detection mask**

img = rgb2gray(imread("F:\CS\sem 5\Digital Image Processing\Pracicals\Pictures\einstein.jpg"));

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

sobel = edge(img); // 0.2(Default)

sobel1 = edge(img, thresh = 0.5);

sobel2 = edge(img, thresh = -1);

figure(), title("Sobel Masks with threshold = 0.2, 0.5, -1");

imshow([sobel sobel1 sobel2]);

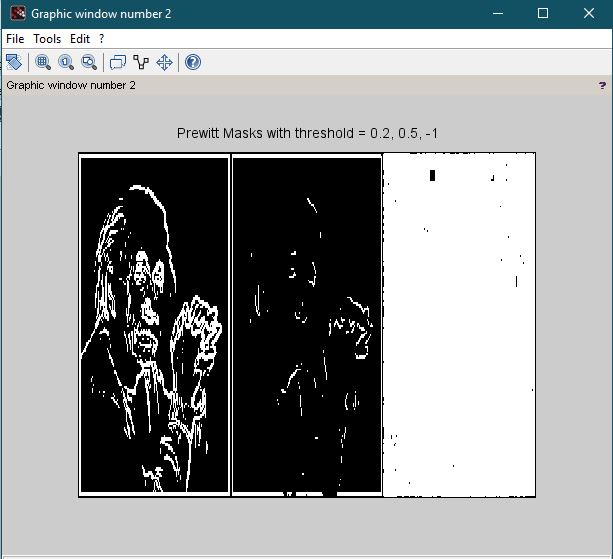
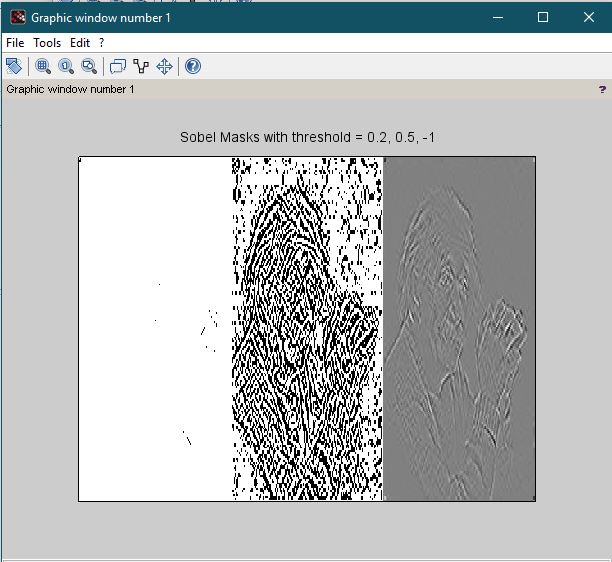
pre = edge(img, 'prewitt');

pre1 = edge(img, 'prewitt', thresh = 0.5);

pre2 = edge(img, 'prewitt', thresh = -1);

figure(), title("Prewitt Masks with threshold = 0.2, 0.5, -1");

imshow([pre pre1 pre2]);



**11. Write and execute program for image morphological operations erosion and dilation.**

function s = str(img)

s = 'Size : ' + strcat(string(size(img)), ' \* ');

endfunction;

// Image

c= im2bw(imread('F:\CS\sem 5\Digital Image Processing\Pracicals\Pictures\circlesBrightDark.png'), 0.5);

c = imcrop(c,[10,30,300,240]);

c(100:220, 130:250) = 1; // box

c(130:200, 150:240) = 0;

c(10:15:200, 30:250) = 1;

c(30:220, 30:10:200) = 1;

figure(), title('Original Image'), xlabel(str(c)), imshow(c);

// Structure element

s1 = imcreatese('rect', 3, 3);

s2 = imcreatese('ellipse', 5, 3);

s3 = imcreatese('cross', 3, 3);

// Plotting

figure();

subplot(1,3,1), title('Rectegular Element');

xlabel(str(s1)), imshow(mat2gray(s1));

subplot(1,3,2), title('Ellipctical Element');

xlabel(str(s2)), imshow(mat2gray(s2));

subplot(1,3,3), title('Cross Structure Element');

xlabel(str(s3)), imshow(mat2gray(s3))

// erosion

e1 = imerode(c, s1);

e2 = imerode(c, s2);

e3 = imerode(c, s3);

// Plotting

figure();

subplot(2,2,1), title('Rectegular Erosion'), imshow(e1);

subplot(2,2,2), title('Ellipctical Erosion'), imshow(e2);

subplot(2,2,3), title('Cross Structure Erosion'), imshow(e3);

// dilation

d1 = imdilate(c, s1);

d2 = imdilate(c, s2);

d3 = imdilate(c, s3);

// Plotting

figure();

subplot(2,2,1), title('Rectegular Dilation'), imshow(d1);

subplot(2,2,2), title('Ellipctical Dilation'), imshow(d2);

subplot(2,2,3), title('Cross Structure Dilation'), imshow(d3);

