**\*RAND FUNCTION**

**1. 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**

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

**a. Obtain Negative image**

**b. Obtain Flip image**

**c. Thresholding**

**d. Contrast stretching**

**3. 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**

**4. 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)**

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

**a. Standard MATLAB function**

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

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

**a. Translation**

**b. Scaling**

**c. Rotation**

**d. Shrinking**

**e. Zooming**

**7. 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**

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

**9. 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**

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

**detection mask**

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

**Rand function**

subplot(x,y,positionofpag)

subplot(x,y,6);

x=rand(100,1);

y=rand(100,1);

plot(x,y,'b');

Title('first pag');

subplot(2,3,1);

x=rand(100,1);

y=rand(100,1);

plot(x,y,'b');

Title('first pag');

subplot(2,3,2);

x=rand(100,1);

y=rand(100,1);

plot(x,y,'b');

Title('first pag');

subplot(2,3,3);

x=rand(100,1);

y=rand(100,1);

plot(x,y,'b');

Title('first pag');

subplot(2,3,4);

x=rand(100,1);

y=rand(100,1);

plot(x,y,'b');

Title('first pag');

Q1

Rand function

subplot(x,y,positionofpag)

subplot(x,y,6);

x=rand(100,1);

y=rand(100,1);

plot(x,y,'b');

Title('first pag');

subplot(2,3,1);

x=rand(100,1);

y=rand(100,1);

plot(x,y,'b');

Title('first pag');

subplot(2,3,2);

x=rand(100,1);

y=rand(100,1);

plot(x,y,'b');

Title('first pag');

subplot(2,3,3);

x=rand(100,1);

y=rand(100,1);

plot(x,y,'b');

Title('first pag');

subplot(2,3,4);

x=rand(100,1);

y=rand(100,1);

plot(x,y,'b');

Title('first pag');

Q2

Q-2 To write and execute image processing programs using point processing method

a. Obtain Negative image b.Obtain Flip image

c. Thresholding

d. Contrast stretching

Code:

clear all;

pkg load image;

%Displaying Original Image% a=imread('car.jpg'); subplot(3,3,1);

imshow(a); title('ORIGINAL IMAGE');

grid on; axis on;

%Displaying negative Image% a\_neg=255-a;

subplot(3,3,2); imshow(a\_neg); title('NEGATIVE IMAGE');

grid on; axis on;

%Displaying VERTICAL FLIPPED Image% flipped\_vertical = flipud(a);

subplot(3,3,3); imshow(flipped\_vertical); title('VERTICAL FLIPPED Image'); grid on; axis on;

%Displaying HORIZONTAL FLIPPED Image% flipped\_horizontal = fliplr(a);

subplot(3,3,4); imshow(flipped\_horizontal); title('HORIZONTAL FLIPPED Image');

grid on; axis on;

**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**

clc;

clear all;

close all;

pkg load image;

img1 = imread('rk.jpg');

img2 = imread('rkg.jpg');

subplot(3,2,1);

imshow(img1);

title('Original 1');

subplot(3,2,2);

imshow(img2);

title('Original 1');

#a add two images

imres1 = imresize(img1, [200,400]);

imres2 = imresize(img2, [200,400]);

addim = imadd(imres1,imres2);

subplot(3,2,3);

imshow(addim);

title('Addition');

#b subtraction two images

subim = imsubtract(imres1, imres2);

subplot(3,2,4);

imshow(subim);

title('Subtraction');

#c Mean

Mean = mean2(img1);

**Question 4:**

**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)**

clc;

clear all;

close all;

#Reading Image

pkg load image;

img1 = imread("ho.jpg");

img2 = imread("ver.jpg");

im1 = im2bw(imresize(img1,[200,400]));

im2 = im2bw(imresize(img2,[200,400]));

#plotting image 1

subplot(3,2,1)

imshow(im1)

title("Image 1")

#plotting image 2

subplot(3,2,2)

imshow(im2)

title("Image 2")

#plotting and image

andimg = and(im1,im2)

subplot(3,2,3)

imshow(andimg)

title("And Image")

#plotting or image

orimg = bitor(im1,im2)

subplot(3,2,4)

imshow(orimg)

title("Or Image")

#plotting intersection image

intimg = im1-im2

subplot(3,2,5)

imshow(intimg)

title("Intersection Image")

#plotting not image

notimg = not(im1)

subplot(3,2,6)

imshow(notimg)

title("Not Image")

***5 – To write a program for histogram calculation and equalization***

(a) Using standard MATLAB/OCTAVE function

#code#

***#histogram calculation and equalization - using standard function***

***pkg load image***

***x = imread('canon\_face.tif');***

***y = histeq(x);***

***subplot(2,2,1)***

***imshow(x)***

***title('Original Image')***

***grid on; axis on;***

***subplot(2,2,2)***

***imhist(x)***

***title('Histogram of Image')***

***subplot(2,2,3)***

***imshow(y)***

***grid on; axis on;***

***title('Equalized Histogram Image')***

***subplot(2,2,4)***

***imhist(y)***

***title('Equalized Histogram')***

(b) Program without using standard MATLAB/OCTAVE function

#code#

***#histogram calculation and equalization without using standard functions***

***pkg load image***

***#function for histogram calculation 'histcalculate'***

***function f = histcalculate(x)***

***[n,m] = size(x)***

***freq = 1:256 #for greyscale image***

***count = 0***

***for i = 1:256***

***for j = 1:n***

***for k = 1:m***

***if x(j,k) == i-1***

***count = count + 1;***

***endif***

***endfor***

***endfor***

***freq(i) = count***

***count = 0***

***endfor***

***f = freq***

***end***

***#function to get new intensity values 'histequalize'***

***function f = histequalize(mat)***

***pdf = mat/sum(mat)***

***cdf = cumsum(pdf)***

***f\_penult = cdf\*255***

***f = round(f\_penult)***

***end***

***#function to produce histogram equalized image***

***function newimg = equalizeimg(arr,x)***

***[n,m]=size(x)***

***newimg = x***

***for j = 1:n***

***for k = 1:m***

***newimg(j,k) = arr(1 + newimg(j,k));***

***endfor***

***endfor***

***end***

***#driver code***

***a = imread('canon\_face.tif');***

***x = histcalculate(a) #calling function to create histogram***

***y = histequalize(x) #calling function to get new intensity values***

***b = equalizeimg(y,a) #calling function to get histogram equalized image***

***n = 0:255***

***subplot(2,2,1);***

***imshow(a)***

***title('Original Image')***

***grid on; axis on;***

***subplot(2,2,2);***

***imshow(b)***

***title('Histogram Equalized Image')***

***grid on; axis on;***

***subplot(2,2,3);***

***stem(n,x);***

***xlabel('Frequency')***

***ylabel('Intensity Vals')***

***title('Histogram of Image')***

***subplot(2,2,4)***

***stem(y,x)***

***xlabel('Frequency')***

***ylabel('Intensity Vals')***

***title('Equalized Histogram')***

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

(a) Translation

#code#

***#geometric transforms: image translation***

***pkg load image***

***x = im2bw(imread('canon\_cameraman.tif'));***

***y = imtranslate(x,20,15,'crop')***

***subplot(2,1,1)***

***imshow(x)***

***title('Original Image')***

***grid on; axis on;***

***subplot(2,1,2)***

***imshow(y)***

***title('Translated Image')***

***grid on; axis on;***

**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**

**a = imread('pd.jpg');**

**subplot(3,3,1);**

**imshow(a);**

**title("Original image");**

**a\_img\_noise= imnoise(a,'salt & pepper',0.02);**

subplot(3,3,2);

imshow(a\_img\_noise);

title("Image with salt and pepper noise");

img\_gaussian\_noise = imnoise(a,'gaussian');

subplot(3,3,3);

imshow(img\_gaussian\_noise);

title("Image with gaussian noise");

#a

mefil=ones(3,3)/9;

img\_restoration = imfilter(a\_img\_noise,mefil) # using mean filter

subplot(3,3,4);

imshow(img\_restoration);

title("Image restoration using mean filter");

#b

sigma = 3 ;

cutoff = ceil(3\*sigma);

h=fspecial('gaussian', 2\*cutoff+1, sigma);

img\_noise\_removed=imfilter(a\_img\_noise,h); #using gaussian filter

subplot(3,3,5);

imshow(img\_noise\_removed);

title("Salt & pepper noise removed using gaussian filter");

#c

mefilf=ones(3,3)/9;

img\_gaussian\_noise\_removed = imfilter(img\_gaussian\_noise,mefilf) # using mean filter

subplot(3,3,6);

imshow(img\_gaussian\_noise\_removed);

title('Gaussian noise removed using harmonic mean filter');

#d

modifyJ=zeros(size(a\_img\_noise)+2);

B = zeros(size(a\_img\_noise));

for x=1:size(a\_img\_noise,1)

for y=1:size(a\_img\_noise,2)

modifyJ(x+1,y+1)=a\_img\_noise(x,y);

end

end

for i= 1:size(modifyJ,1)-2

for j=1:size(modifyJ,2)-2

window=zeros(9,1);

inc=1;

for x=1:3

for y=1:3

window(inc)=modifyJ(i+x-1,j+y-1);

inc=inc+1;

end

end

med=sort(window);

B(i,j)=med(5);

end

end

B = uint8(B);

subplot(3,3,7);

imshow(B);

title('Filtered Image with Median Filter')

**Question 8:**

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

Code:

clear all;

close all;

clc;

pkg load image;

img = imread('rkg.jpg');

subplot(2,2,1);

q= rgb2gray(img)

imshow((q));

title('Original image');

b=size(img);

if size(b,2)==3

 a = rgb2gray(img);

end

a = imnoise(img,'salt & pepper');

n=9;

n1=ceil(n/2);

a=double(a);

lpf=(1/n^2)\*ones(n);

hpf=-lpf;

hpf(n1,n1)=(n^2-1)/n^2;

c=0;

h=0;

for i=n1:b(1)-n1

 for j=n1:b(2)-n1

 p=1;

 for k=1:n

 for l=1:n

 c(p)=img(i-n1+k,j-n1+l);

 p=p+1;

 end

 end

 d(i,j)=median(c);

 c=0;

 end

end

e=uint8(d);

subplot(2,2,2);

imshow(e);

title('low pass image');

for i=n1:b(1)-n1

 for j=n1:b(2)-n1

 for k=1:n

 for l=1:n

 h=h+img(i-n1+k,j-n1+l)\*hpf(k,l);

 end

 end

 g(i,j)=h;

 h=0;

 end

end

f=uint8(g);

subplot(2,2,3);

imshow(f);

title('high pass image');

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

1. **Apply FFT on given Image**

#code#

***#Frequency domain filtering: FFT***

***pkg load image;***

***#reading an image***

***img1 = imread('canon\_cameraman.tif')***

***#using function fft2() to calculate fft img\_fft = fft2(img1)***

***% Shift zero frequency components to the center fft\_img\_shifted = fftshift(img\_fft);***

***#Compute the magnitude spectrum mag\_spec = log(1 + abs(fft\_img\_shifted));***

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

***subplot(2,1,2); imshow(mag\_spec, []); title('Magnitude Spectrum');***

1. Perform low pass and high pass filtering in frequency domain #code#

***#Frequency domain filtering: High pass and Low pass***

***pkg load image; pkg load signal;***

***img1 = imread('canon\_cameraman.tif')***

***#Using the same FFT conversion routine img\_fft = fft2(img1);***

***fft\_img\_shifted = fftshift(img\_fft);***

***#designing Butterworth highpass and lowpass filter #butter(order, cutoff\_frequency, filter\_type) [low\_a, low\_b] = butter(2, 0.15, 'low')***

***[high\_a, high\_b] = butter(2, 0.15, 'high')***

***#Applying filters to image***

***lowpass\_img = filter2(low\_b, low\_a, img1) highpass\_img = filter2(high\_b, high\_a, img1)***

***#Subplotting subplot(2,2,2) imshow(img1) title('Original Image')***

***subplot(2,2,1) imshow(highpass\_img) title('HighPass Image')***

***subplot(2,2,3) imshow(lowpass\_img) title('LowPass Image')***

1. Apply IFFT to reconstruct image #code#

***#Frequency domain filtering: IFFT***

***pkg load image;***

***#reading an image***

***img1 = imread('canon\_cameraman.tif')***

***#Applying FFT img\_fft = fft2(img1)***

***#Applying IFFT***

***img\_ifft = ifft2(img\_fft);***

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

***subplot(2,1,2); imshow(img\_ifft, []); title('Reconstructed Image');***

Q10

clear all;

close all;

clc;

pkg load image;

img = imread('rkg.jpg');

img= rgb2gray(img)

subplot(2,3,1);

imshow(img);

title("Original image");

img\_prewitt = edge(img,'Prewitt');

subplot(2,3,2);

imshow(img\_prewitt);

title("Edge detection using Prewitt mask");

img\_sobel = edge(img,'Sobel');

subplot(2,3,3);

imshow(img\_sobel);

title("Edge detection using Prewitt mask");

img\_robertcross = edge(img,'Sobel');

subplot(2,3,4);

imshow(img\_robertcross);

title("Edge detection using Robert cross mask");

img\_laplacian = edge(img,'log');

subplot(2,3,5);

imshow(img\_laplacian);

title("Edge detection using Robert cross mask");

**Q11. Write and execute program for image morphological operations erosion and dilation.**

clear all;

close all;

clc;

pkg load image;

I = imread('pd.jpg');

subplot(2,2,1);

imshow(I);

title('original image')

#erosion

I=im2bw(I);

se=ones(5, 5);

[P, Q]=size(se);

I\_erosion=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)

on=I(i-floor(P/2):i+floor(P/2), j-floor(Q/2):j+floor(Q/2));

nh=on(logical(se));

I\_erosion(i, j)=min(nh(:));

end

end

subplot(2,2,2);

imshow(I\_erosion);

title('erosion of image')

#dilation

se=ones(5, 5);

[P, Q]=size(se);

I\_dilation=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)

on=I(i-floor(P/2):i+floor(P/2), j-floor(Q/2):j+floor(Q/2));

nh=on(logical(se));

I\_dilation(i, j)=max(nh(:));

end

end

subplot(2,2,3);

imshow(I\_dilation);

title('dilation of image')