I I I T Bhubaneswar

Imagine, Innovate, Inspire

(A University established by Government of Odisha)

Fundamentals of Image Processing
Lab Record

Name- D CHANDANA REDDY

ID- B219021

Branch-ETC

CONTENTS

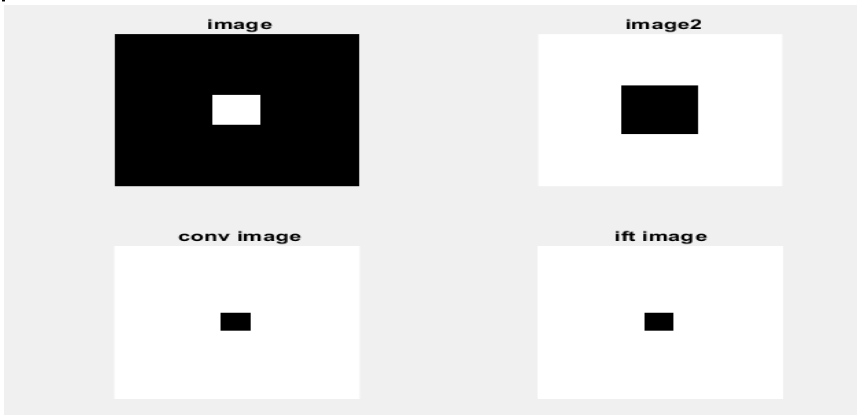
EXPERIMENT-5

a)Generate a synthetic 256 * 256 image with black background and a white square of size 50 * 50 in the center. Generate another synthetic 256 * 256 image with white background and a black square of size 80 * 80 in the center. WAP to prove the convolution property using DFT.

CODE:

```
close all;
clc;
image = zeros(256,256);
image2 = ones(256,256);
for i = 103:153
    for j = 103:153
        image(i,j) = 1;
    end
end
for i = 88:168
    for j = 88:168
        image2(i,j) = 0;
    end
end
conv_image = conv2(image,image2,"same");
ft_image = fft2(image);
ft_image2 = fft2(image2);
final_image = ft_image .* ft_image2;
ift_image = fftshift(ifft2(final_image));
subplot(2,2,1);
imshow(image);
title(" image");
subplot(2,2,2);
imshow(image2);
title(" image2");
subplot(2,2,3);
imshow(conv_image);
title(" conv image");
subplot(2,2,4);
imshow(real(ift_image));
title("ift image");
```

Output:

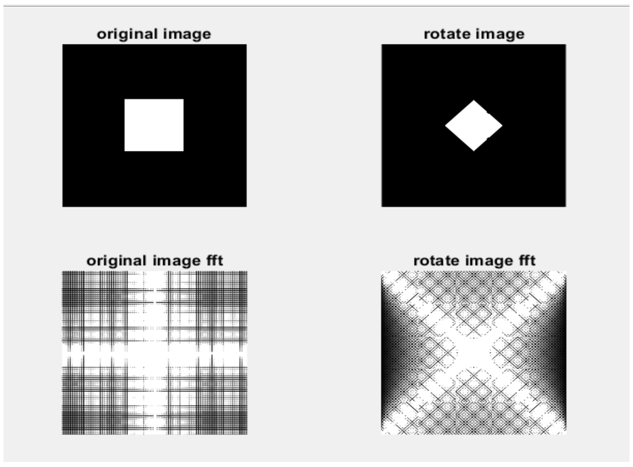


b) Generate another synthetic 256 * 256 image with black background and a white square of size 80 * 80 in the center. WAP to prove the rotation property of DFT.

CODE:

```
clear all;
close all;
clc;

image = zeros(256,256);
for i = 88:168
    for j = 88:168
        image(i,j) = 1;
    end
end
subplot(2,2,1);
imshow(image);
title("original image")
i = imrotate(image,45);
subplot(2,2,2);
imshow(i);
title("rotate image");
ft_image1 = abs(fftshift(fft2(image)));
ft_rotimage = fft2(i);
subplot(2,2,3);
imshow(ft_image1);
title("original image fft")
subplot(2,2,4);
ft_image2 = abs(fftshift(ft_rotimage));
imshow(ft_image2);
title("rotate image fft");
```

Output:



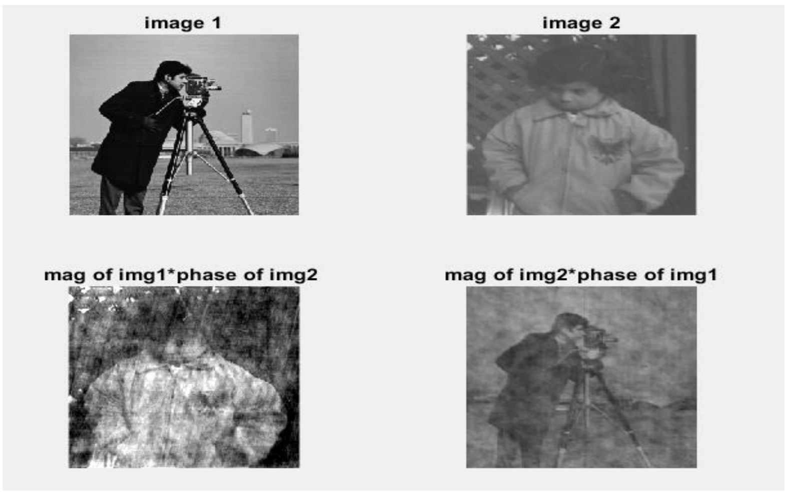
C) WAP to interchange phase between two images. Analyse the result.

CODE:

```
clear all;
close all;
clc;
img1 = imread("cameraman.tif");
```

```
img2 = imread("pout.tif");
img3 = imresize(img2,[256 256]);
fft_image1 = fft2(img1);
mag1 = abs(fft_image1);
phase1 = angle(fft_image1);
fft_image2 = fft2(img3);
mag2 = abs(fft_image2);
phase2 = angle(fft_image2);
final_image1 = mag1.*exp(1i*phase2);
final_image2 = mag2.*exp(1i*phase1);
cov_image1 = ifft2(final_image1);
cov_image2 = ifft2(final_image2);
subplot(2,2,1);
imshow(img1);
title("image 1");
subplot(2,2,2);
imshow(img3);
title("image 2");
subplot(2,2,3);
imshow(uint8(cov_image1));
title("mag of img1*phase of img2 ");
subplot(2,2,4);
imshow(uint8(cov_image2));
title("mag of img2*phase of img1");
```

Output:



Observation:
While processing an image we mainly consider the magnitude the image and neglect the phase of the image but the above experiment clearly shows that phase is more important than the magnitude as after changing the phase and magnitude of two images the resultant image looks alike the image whose phase it contains.

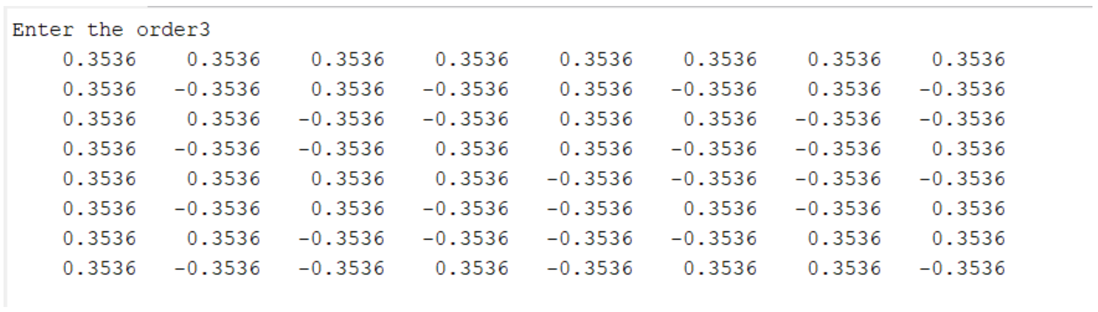
5.d) WAP to generate a Hadamard Transformation matrix of specified order. Input the size of transformation matrix from user.

MATLAB CODE:

```
clear all;
close all;
clc;
h = [1 1 ; 1 -1];
n = input("Enter the order");
```

```
val = 2^(-.5);
for i = 2:n
    m = [h h ; h -h];
    h = m;
end
 h = h * val ^ n;
disp(h);
```

Output:



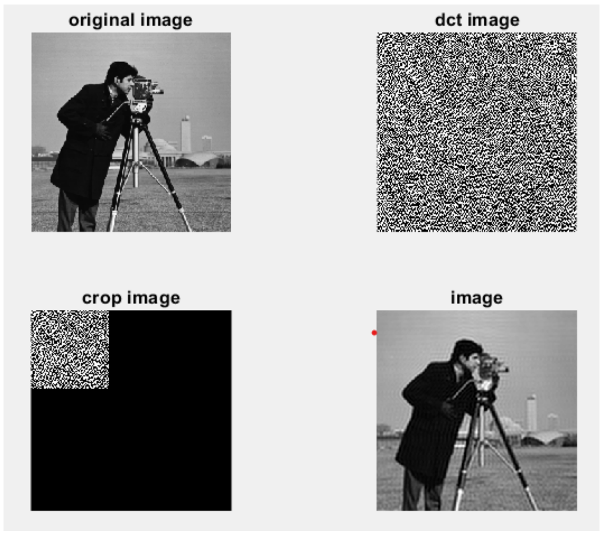
```
Enter the order3
    0.3536    0.3536    0.3536    0.3536    0.3536    0.3536    0.3536    0.3536
    0.3536   -0.3536    0.3536   -0.3536    0.3536   -0.3536    0.3536   -0.3536
    0.3536    0.3536   -0.3536   -0.3536    0.3536    0.3536   -0.3536   -0.3536
    0.3536   -0.3536   -0.3536    0.3536    0.3536   -0.3536   -0.3536    0.3536
    0.3536    0.3536    0.3536    0.3536   -0.3536   -0.3536   -0.3536   -0.3536
    0.3536   -0.3536    0.3536   -0.3536   -0.3536    0.3536   -0.3536    0.3536
    0.3536    0.3536   -0.3536   -0.3536   -0.3536   -0.3536    0.3536    0.3536
    0.3536   -0.3536   -0.3536    0.3536   -0.3536    0.3536    0.3536   -0.3536
```

e**) WAP to perform DCT and IDCT on 'camearman' image. Crop the DCT of the image to 100 * 100 and find IDCT. Analyse your result.**

MATLAB CODE:

```
clear all;
close all;
clc;
img1 = imread("cameraman.tif");
img2 = dct2(img1);
img3 = img2;
for i = 1 : 256
    for j = 1:256
        if i > 100 || j > 100
            img3(i,j) = 0;
        end
    end
end
img4 = uint8(idct2(img3));
subplot(221);
imshow(img1);

subplot(222);
imshow(img2);

subplot(223);
imshow(img3);

subplot(224);
imshow(img4);
```

Output:



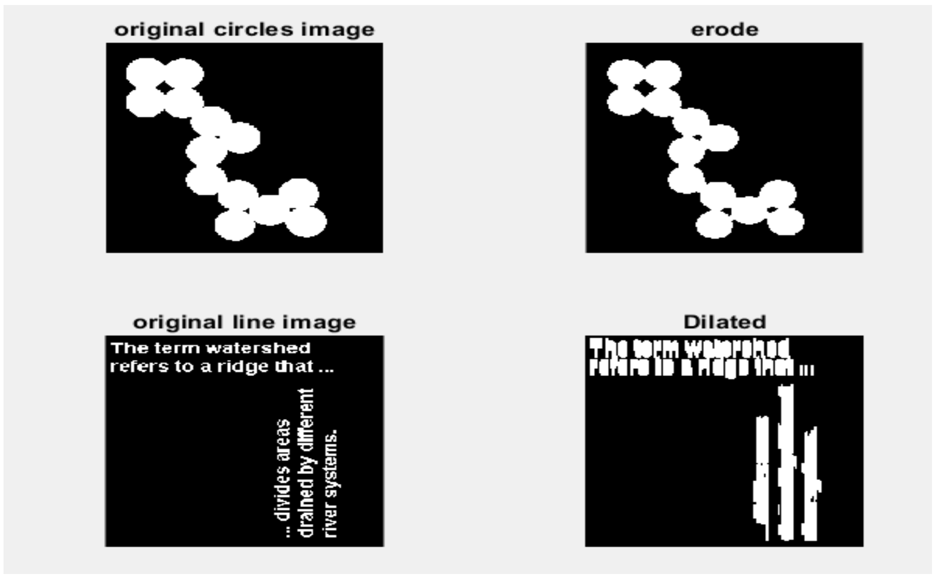
Comment:
In DCT the energy of the image is stored toward the centre. So the entire image can be reconstructed from the centre part of the image. In the above example we reconstructed the original image from 100*100 pixel part of the DCT image.

EXPERIMENT-6:

a) **WAP to erode and dilate with structuring elements as circle and line on different images. [Use imerode(), imdilate(), strel()].**

CODE:
```
clc;
clear all;
close all;
img1 = imread("circles.png");
se1 = strel('disk',2);
erodBw=imerode(img1,se1);
subplot(221)
imshow(img1);
title('original circles image');
subplot(222)
imshow(erodBw);
title('erode');
img2 = imread("text.png");
se = strel('line',11,90);
BW2 = imdilate(img2,se);
subplot(223);
imshow(img2);
title('original line image');
subplot(224);
imshow(BW2);
title('Dilated');
```

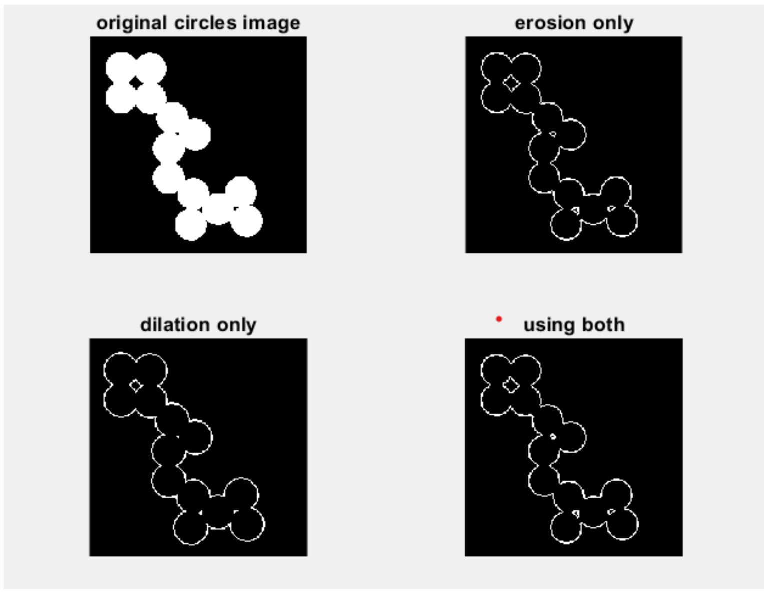
Output:



b) WAP to find edges in an image using morphological operators i.e. erosion only, dilation only and using both. Analyze the result.

CODE:
```
clc;
clear all;
close all;
img = imread("circles.png");
se1 = strel('disk',2);
erodBw=imerode(img,se1);
subplot(221)
imshow(img);
title('original circles image');
img1=img-erodBw;
subplot(222)
imshow(img1);
title('erosion only');
BW2 = imdilate(img,se1);
img2=BW2-img;
subplot(223);
imshow(img2);
title('dilation only');
img3=img1-img2;
subplot(224);
imshow(img3);
title('using both');
```

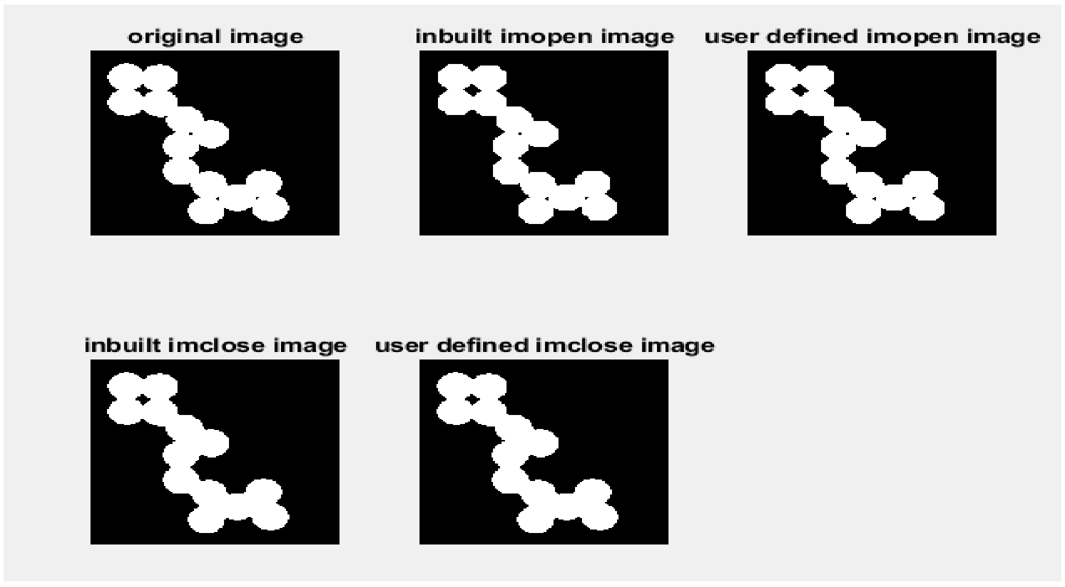
Output:



c) WAP for opening and closing operation of an image using in-built function imopen(), and imclose(). Also perform same operations using dilation and erosion.

```
clc;
clear all;
close all;
img = imread("circles.png");
se1 = strel('disk',12);
se2 = strel('disk',2);
subplot(231)
imshow(img);
title("original image")
afterOpening = imopen(img,se1);
subplot(232);
imshow(afterOpening);
title("inbuilt imopen image")
ero=imerode(img,se1);
img3=imdilate(ero,se1);
subplot(233);
imshow(img3);
title("user defined imopen image")
afterclose = imclose(img,se2);
subplot(234);
imshow(afterclose);
title("inbuilt imclose image")
img3=imdilate(img,se2);
ero=imerode(img3,se2);
subplot(235);
imshow(ero);
title("user defined imclose image");
```

Output:

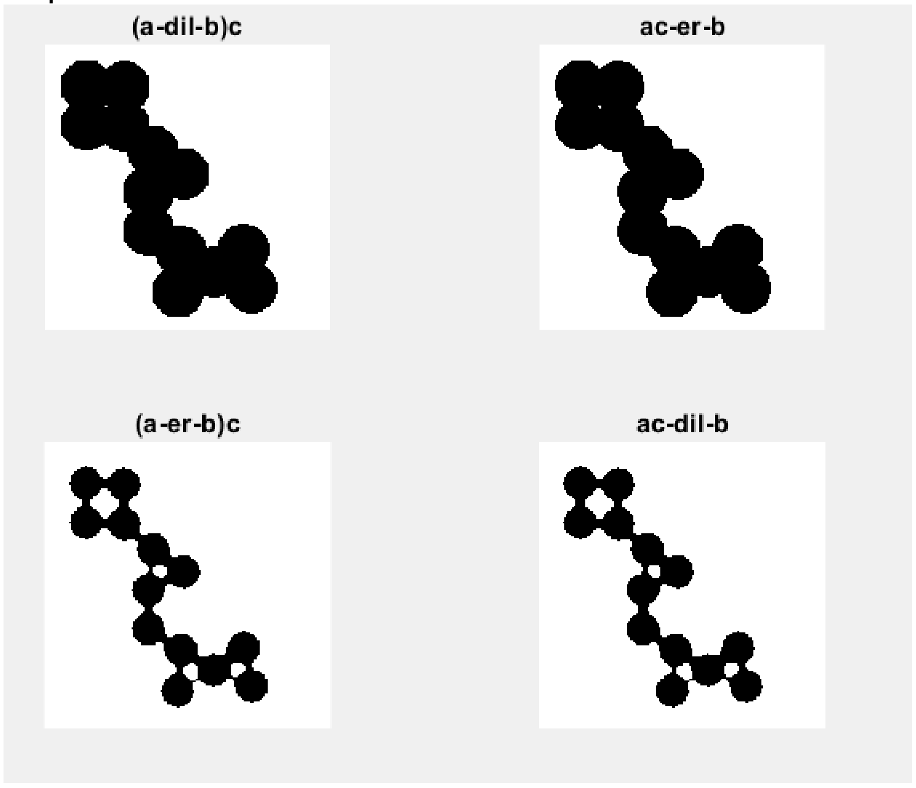


d) Prove the duality theorem of erosion and dilation considering an image A and structuring element B.

CODE:

```
clc;
clear all;
close all;
img = imread("circles.png");
img_comp = imcomplement(img);
se = strel('disk', 5);
a_dil_b = imdilate(img,se);
a_dil_b_cp = imcomplement(a_dil_b);
a_cp_er_b = imerode(img_comp,se);
a_er_b = imerode(img,se);
a_er_b_cp = imcomplement(a_er_b);
a_cp_dil_b = imdilate(img_comp,se);
subplot(221);
imshow(a_dil_b_cp);
title("(a-dil-b)c");
subplot(222);
imshow(a_cp_er_b);
title("ac-er-b");
subplot(223);
imshow(a_er_b_cp);
title("(a-er-b)c");
subplot(224);
imshow(a_cp_dil_b);
title("ac-dil-b");
```

Output:

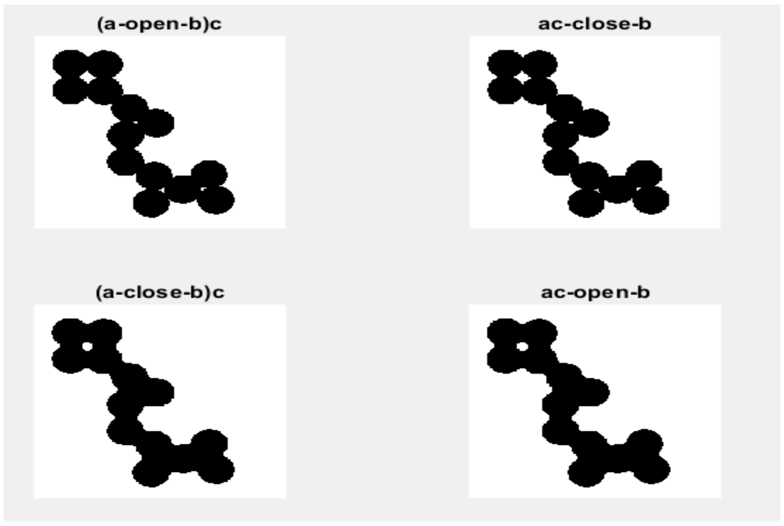


e) Prove the duality theorem of opening and closing considering an image A and structuring element B.

CODE:

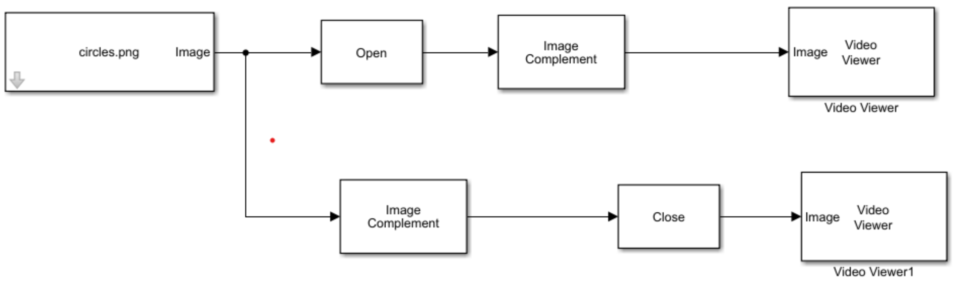
```
clc;
clear all;
close all;
img = imread("circles.png");
img_comp = imcomplement(img);
se = strel('disk', 5);
a_op_b = imopen(img,se);
a_op_b_cp = imcomplement(a_op_b);
a_cp_cl_b = imclose(img_comp,se);
a_cl_b = imclose(img,se);
a_cl_b_cp = imcomplement(a_cl_b);
a_cp_op_b = imopen(img_comp,se);
subplot(221);
imshow(a_op_b_cp);
title("(a-open-b)c");
subplot(222);
imshow(a_cp_cl_b);
title("ac-close-b");
subplot(223);
imshow(a_cl_b_cp);
title("(a-close-b)c");
subplot(224);
imshow(a_cp_op_b);
```

```
title("ac-open-b");
```
Output:

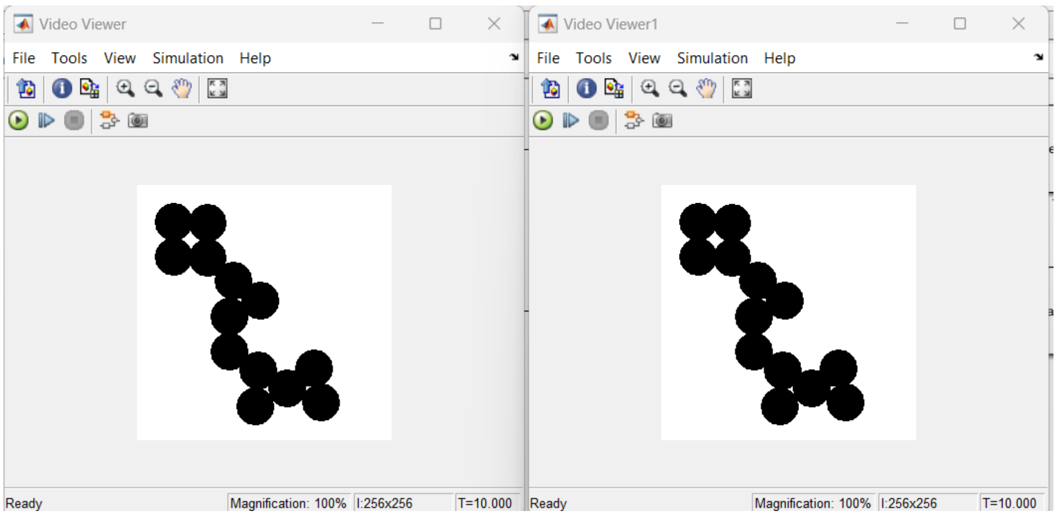


f) Repeat (d) using Simulink

MATLAB SIMULINK:

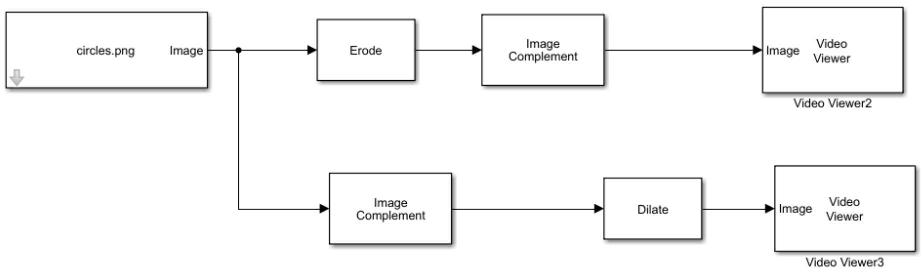


Output:

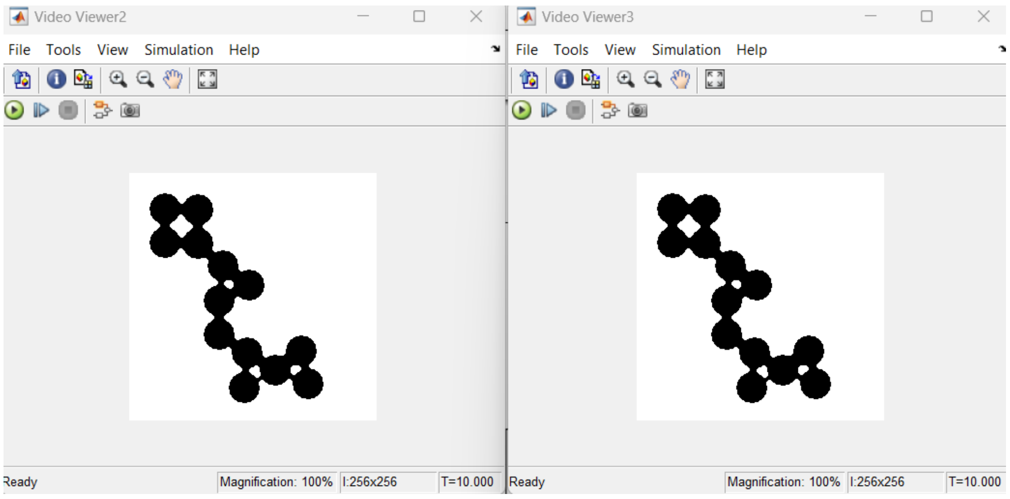


g) Repeat (e) using Simulink

MATLAB SIMULINK:



Output:



EXPERIMENT-7

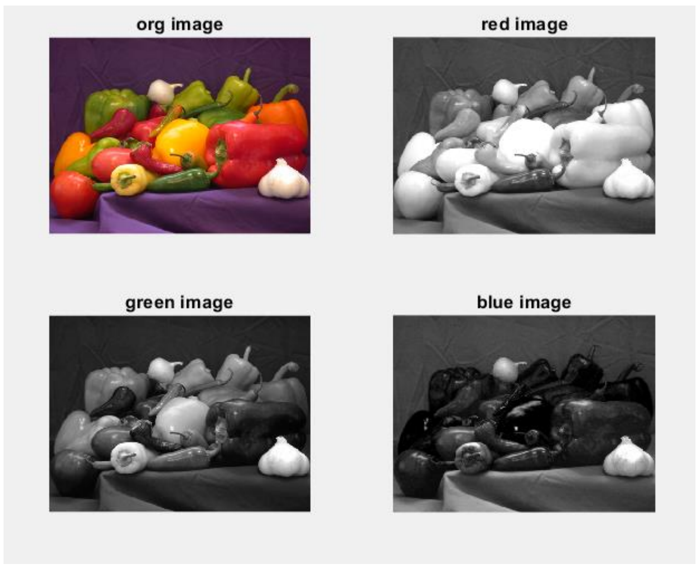
a) Read a RGB image and extract each colour component and display it.

CODE:

```
clc
close all;
clear all;
img=imread('peppers.png');
subplot(221);
imshow(img);
title('org image')

[x y z]=imsplit(img);
subplot(222);
imshow(x);
title('red image')
```

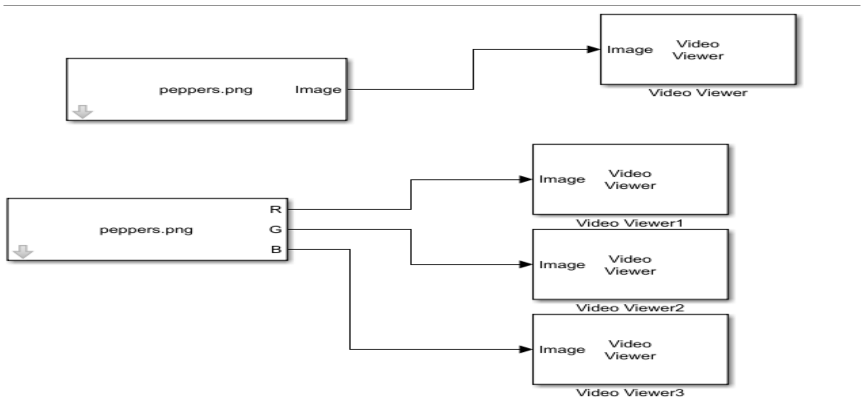
```
subplot(223);
imshow(y);
title('green image')

subplot(224);
imshow(z);
title('blue image');
```

Output:

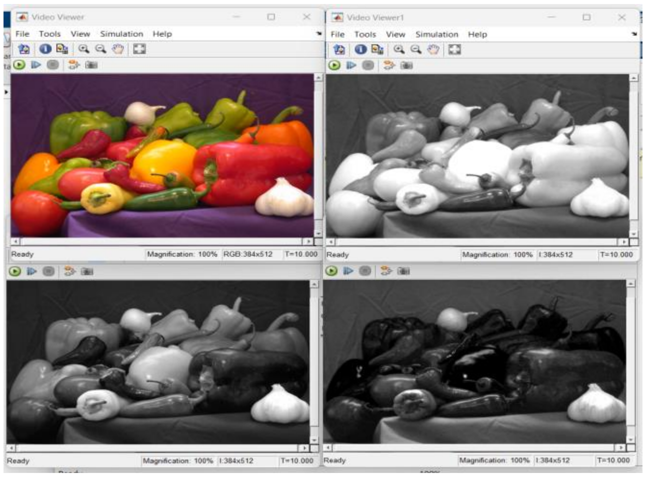


a) Repeat (a) using Simulink

Simulink:



Output:

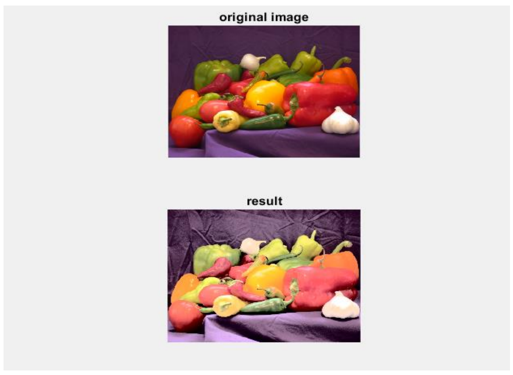


b) Read a RGB image. Convert it into YIQ format. [Use rgb2ntsc()]. Apply histogram equalization on Y component only. Convert it to RGB and display the result.

CODE:

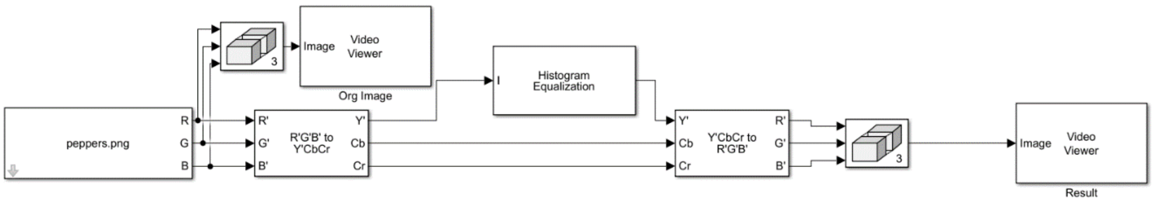
```
clc;
clear all
close all;
img=imread("peppers.png");
subplot(211);
imshow(img);
title("original image")
YIQ=rgb2ntsc(img);
[y iw qf]=imsplit(YIQ);
Y_EQH=histeq(y);
res=cat(3,Y_EQH,iw,qf);
rgb_res=ntsc2rgb(res)
subplot(212);
imshow(rgb_res);
title('result');
```

Output:

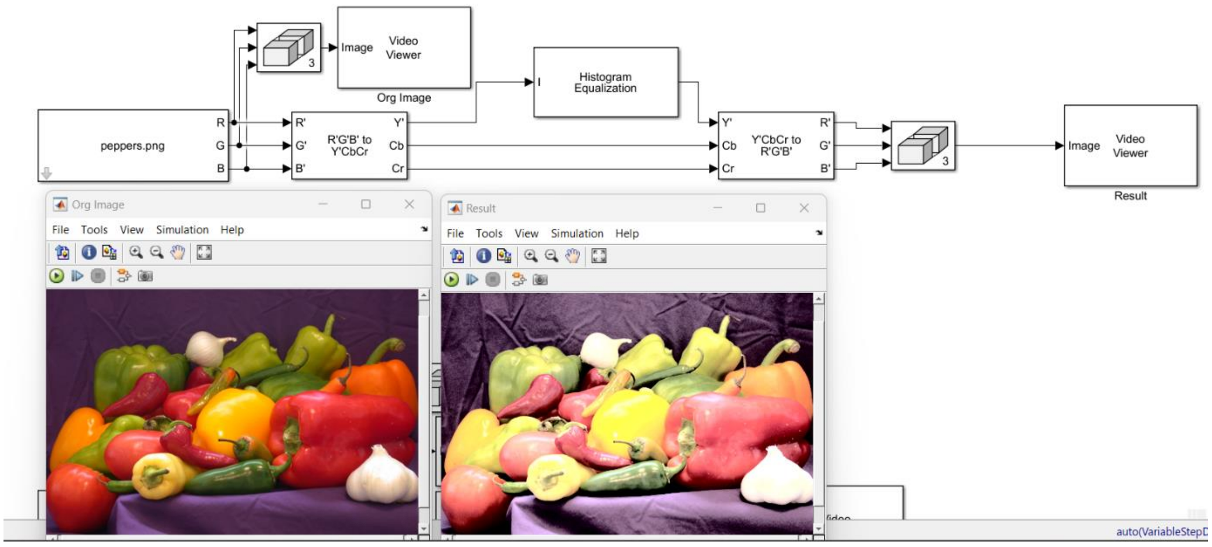


c) Repeat (c) using Simulink.

Simulink:



Output:

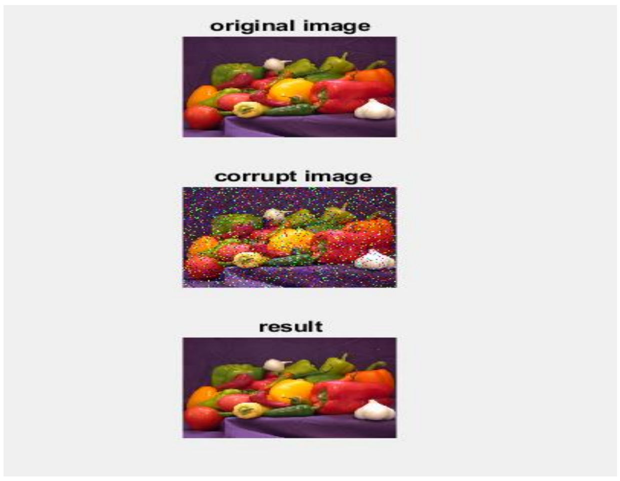


d) **Read a colour image. Corrupt the image by salt and pepper noise. Display the corrupted image. Apply median filter on each plane of the corrupted image and display the result. [Use medfilt2()]**

CODE:

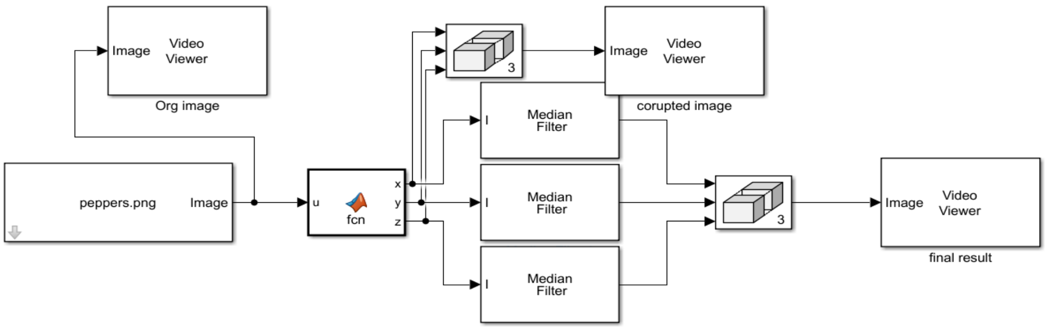
```
clc;
clear all
close all;
img=imread("peppers.png");
subplot(311);
imshow(img);
title("original image")
YIQ=imnoise(img,"salt & pepper",0.1);
subplot(312);
imshow(YIQ);
title(" corrupt image")
[y iw qf]=imsplit(YIQ);
Y_m=medfilt2(y);
IW_m=medfilt2(iw);
QF_m=medfilt2(qf);
res=cat(3,Y_m,IW_m,QF_m);
subplot(313);
imshow(res);
title('result');
```

Output:



e) Repeat (e) using Simulink.

Simulink:

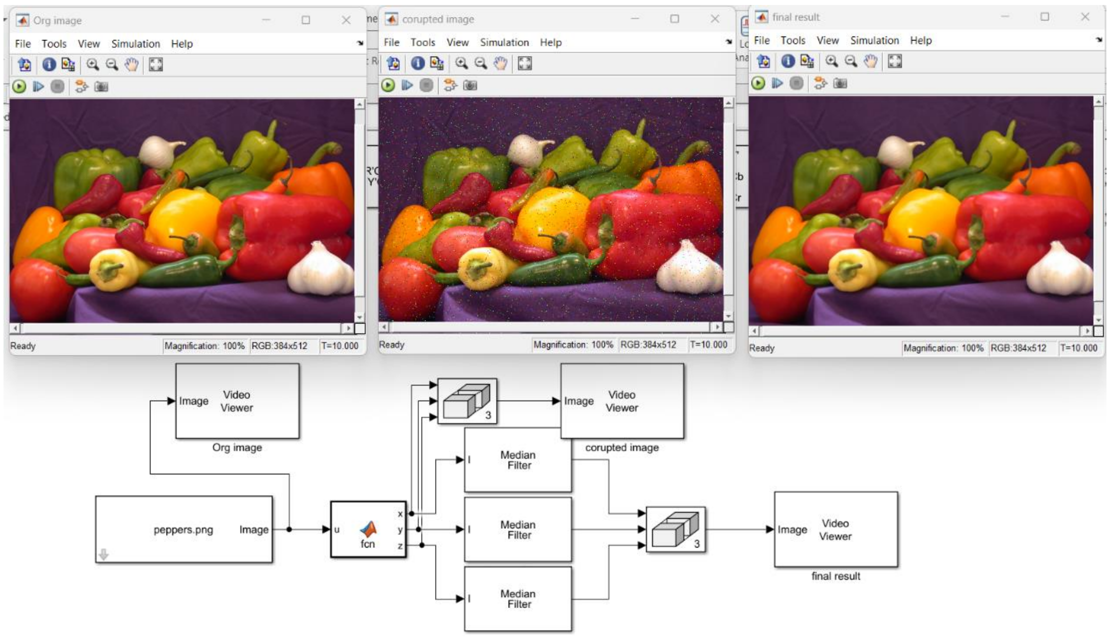


function [x,y,z]= fcn(u)

noise_img = imnoise(u,'salt & pepper',0.02);

[x, y, z] = imsplit(noise_img);

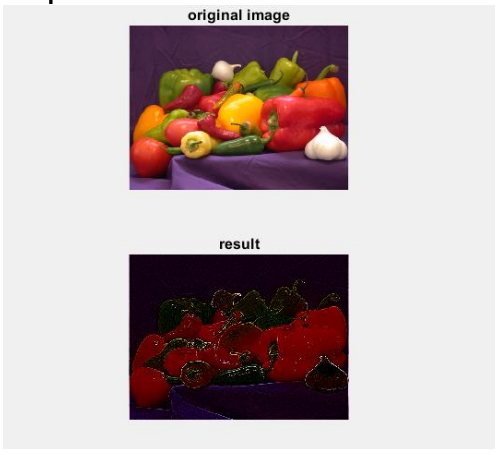
O/P:



f) **Read a RGB image. Convert it into YIQ format. Apply high pass filtering on Y component only. Convert it to RGB and display the result. (Apply filter h=[-1 -1 -1;-1 8 -1;-1 -1 -1]).** CODE:

```
clc;
clear all
close all;
img=imread("peppers.png");
subplot(211);
imshow(img);
title("original image");
h=[-1 -1 -1;-1 8 -1;-1 -1 -1];
YIQ=rgb2ntsc(img);
[y iw qf]=imsplit(YIQ);
Y_EQH=imfilter(y,h);
res=cat(3,Y_EQH,iw,qf);
rgb_res=ntsc2rgb(res);
subplot(212);
imshow(rgb_res);
title('result');
```

Output:


original image
result