Digital Image Processing

Practical File

17.11.2022
Anushka Mathur (20020570007)
BSc Hons Computer Science

Table Of Contents

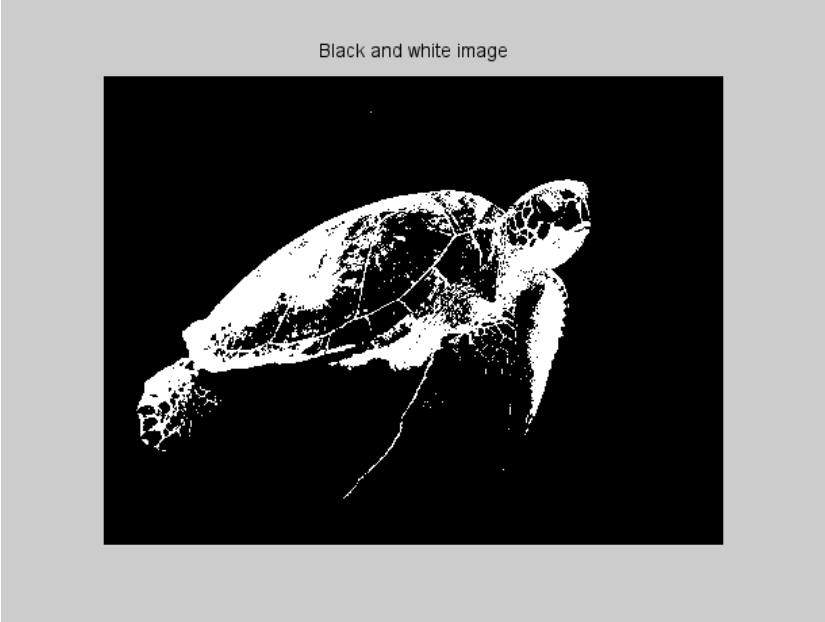
Question 1

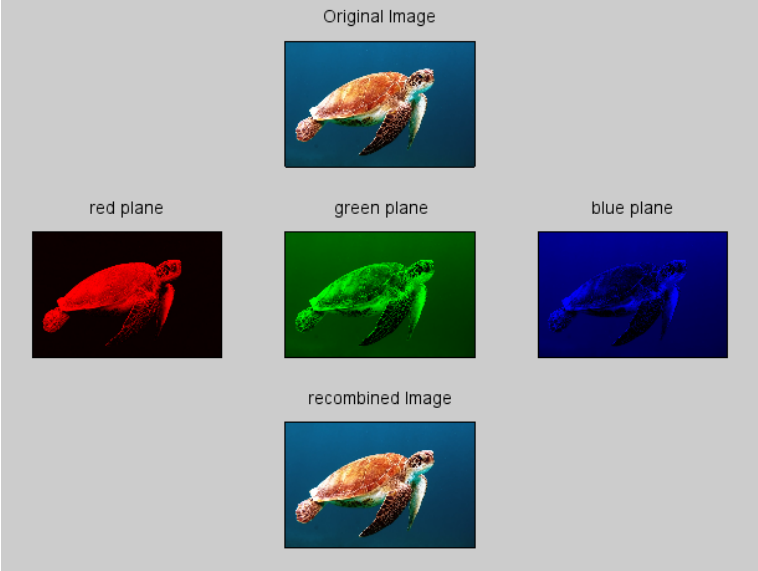
1. Write program to read and display digital image using MATLAB or SCILAB
   a. Become familiar with SCILAB/MATLAB Basic commands
   b. Read and display image in SCILAB/MATLAB
   c. Resize given image
   d. Convert given color image into gray-scale image
   e. Convert given color/gray-scale image into 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

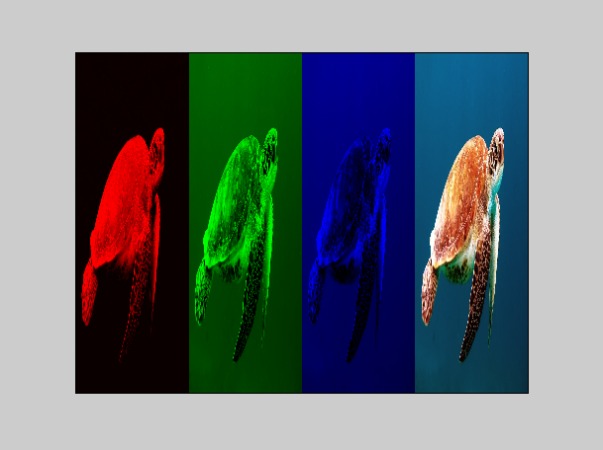
```
// QUESTION 1

//part(a),(b)
Originalimg = imread("C:\Users\RMC-1\Desktop\turtle.jpg");
figure
imshow(Originalimg);

//part(c)
ResizedImg = imresize(Originalimg,0.1);
figure
imshow(ResizedImg);
title("Resized Img");
figure
imshow(Originalimg);
title("Original Img");

// part(d)
RGBtoGray = rgb2gray(Originalimg);
figure
imshow(RGBtoGray)
title("Gray scale image");
```

```
// part(e)
bw = im2bw(Originalimg, 0.5);
figure
imshow(bw);
title("Black and white image");

// part (g),(f),
red = Originalimg(:, :, 1);
green = Originalimg(:, :, 2);
blue = Originalimg(:, :, 3);
AllBlack = zeros(size(Originalimg,1), size(Originalimg,2), 'uint8');
JustRed = cat(3, red, AllBlack, AllBlack);
JustGreen = cat(3, AllBlack, green, AllBlack);
JustBlue = cat(3, AllBlack, AllBlack, blue);
NewOriginalimg = cat(3, red, green, blue);
figure
img = [JustRed JustGreen JustBlue NewOriginalimg];
imshow(img);

// part (h)
recombinedRGBImage = cat(3,red, green, blue);
figure
subplot(3,3,2),imshow(Originalimg);
title("Original Image");
subplot(3,3,8),imshow(recombinedRGBImage);
title("recombined Image");
subplot(3,3,4);
imshow(JustRed);
title("red plane");
subplot(3,3,5);
imshow(JustGreen);
title("green plane");
subplot(3,3,6);
imshow(JustBlue);
title("blue plane");


// part (i) -> flow control demonstration
// program to determine if number is even or odd
num = input("Enter a number: ");
if modulo(num,2) == 0
    disp("The number is even");
elseif modulo(num,2) ~= 0
```

```
    disp("The number is odd");
else
    disp("The input is invalid");
end
// part (i) -> loop control not known

// part (j)
arr = [1,5,7,9; 11,23,60,34; 44,78,88,32];
image(arr)
colorbar
```


Black and white image


Original Image
red plane
green plane
blue plane
recombined Image

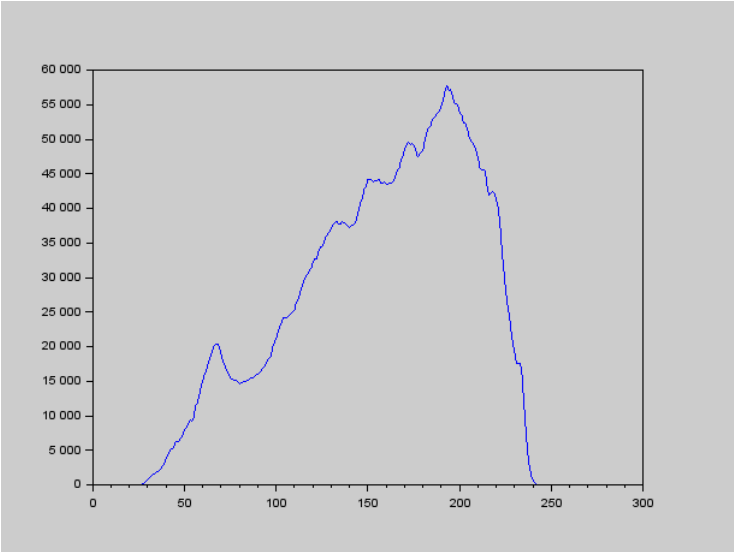


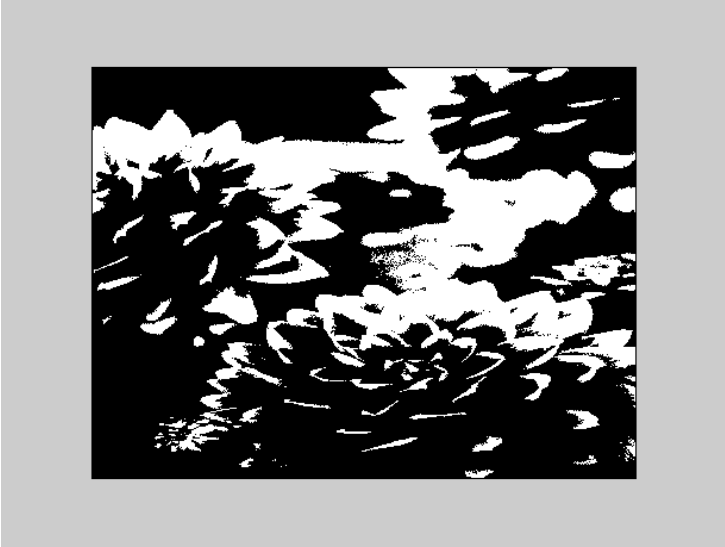
Question 2

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

```
// QUESTION 2

// Part (a)
OriginalImg = imread("C:\Users\RMC-1\Desktop\flower.jpg");
figure
imshow(OriginalImg);
title("Original image");
NegativeImg = 255 - OriginalImg;
figure
imshow(NegativeImg);
title("Negative image");

// Part (b)
FlippedImg = flipdim(OriginalImg, 1);
imshow(FlippedImg);
title("Flipped image");

// Part (c)
// step 1 -> convert image to grey scale
GrayImg = rgb2gray(OriginalImg);
// step 2 -> Create histogram
Histogram = imhist(GrayImg);
figure
plot(0:255, Histogram);
// step 3 -> select threshold value from histogram
figure
imshow(GrayImg>190);

// Part (d)
```

```
a = 0 // lower limit on the output intensity we wish to have
b = 255 // upper limit on the output intensity we wish to have
c = min(GrayImg) // min present intensity in the original image
d = max(GrayImg) // max present intensity in the original image
stretchedImg = (GrayImg - c).*((b-a)/(d-c))+a
figure
imshow(stretchedImg);
title("Stretched Image")
```


60 000
55 000
50 000
45 000
40 000
35 000
30 000
25 000
20 000
15 000
10 000
5 000
0
0
50
100
150
200
250
300

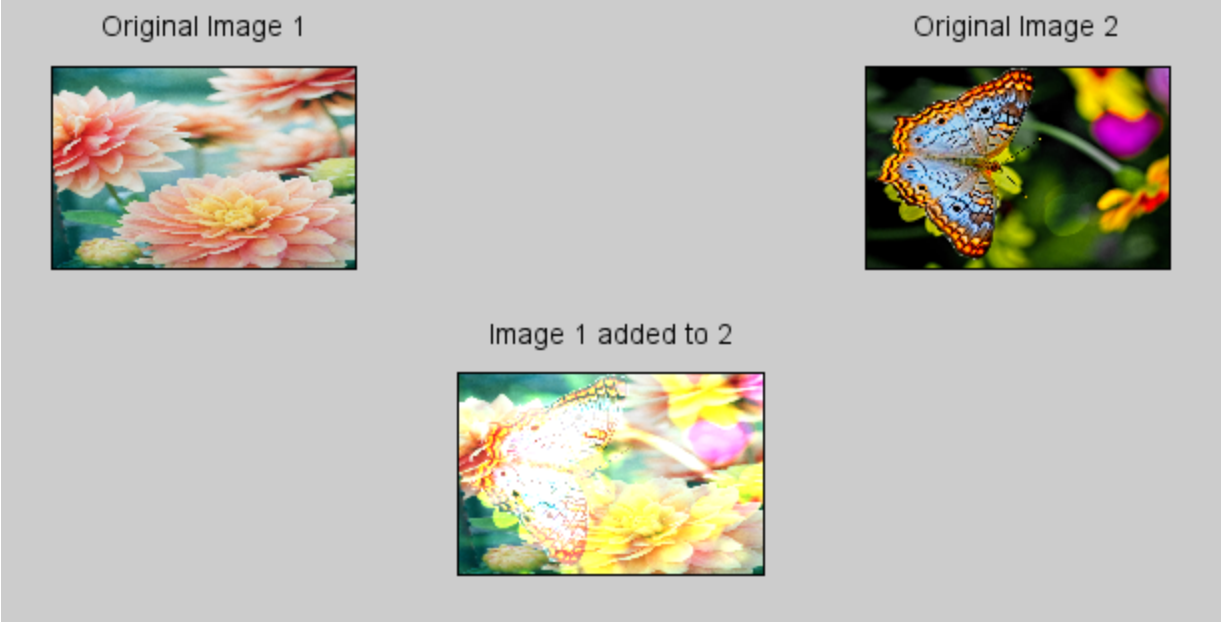


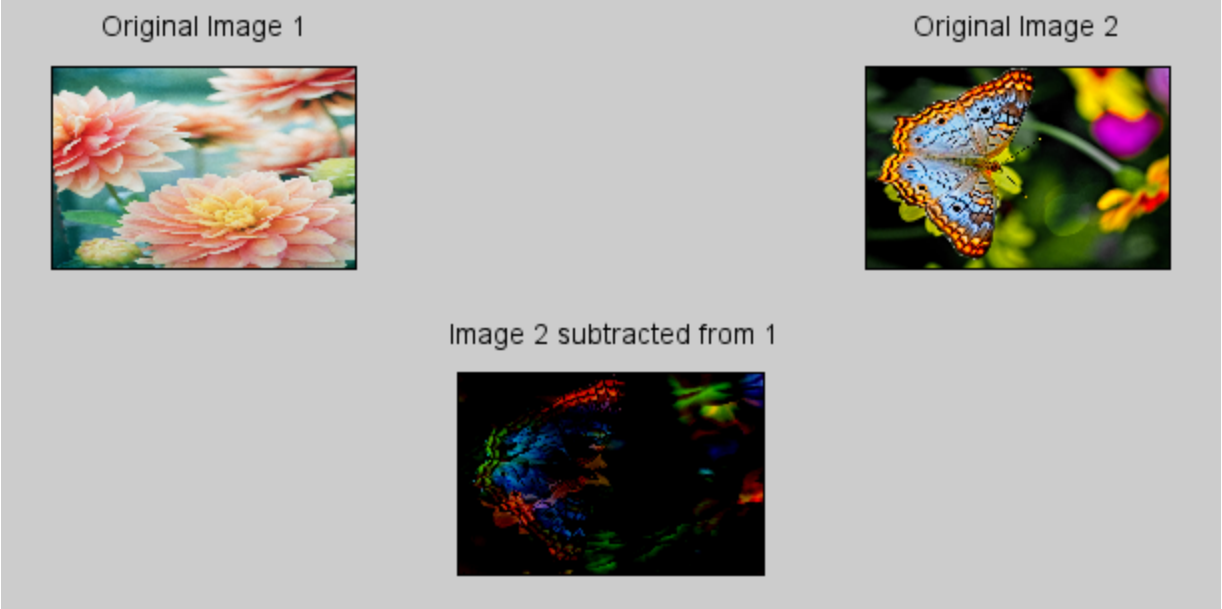

Stretched Image

Question 3

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

```
// QUESTION 3

// Part a
img = imread("C:\Users\RMC-1\Desktop\flower.jpg");
img1 = imresize(img, [3648, 4607])
img2 = imread("C:\Users\RMC-1\Desktop\butterfly.jpg");
addedImg = imadd(img1, img2)
figure
subplot(3,3,1),imshow(img1);
title("Original Image 1");
subplot(3,3,3),imshow(img2);
title("Original Image 2");
subplot(3,3,5);
imshow(addedImg);
title("Image 1 added to 2");

// Part b
SubtractImg = imsubtract(img2,img1);
figure
subplot(3,3,1),imshow(img1);
title("Original Image 1");
subplot(3,3,3),imshow(img2);
title("Original Image 2");
subplot(3,3,5);
imshow(SubtractImg);
title("Image 2 subtracted from 1");

// part (c)
MeanVal = mean2(img1(:));
disp(MeanVal);
```






```
--> exec('C:\Users\RMC-1\Desktop\Anushka_Scilab_q3.sce', -1)

   160.95789
```

Question 4

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)

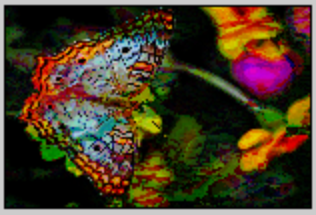
```
// QUESTION 4

// Part (a)
img = imread("C:\Users\RMC-1\Desktop\flower.jpg");
img1 = imresize(img, [3648, 4607])
img2 = imread("C:\Users\RMC-1\Desktop\butterfly.jpg");
andOperation = bitand(img1, img2)
figure
subplot(3,3,1),imshow(img1);
title("Original Image 1");
subplot(3,3,3),imshow(img2);
title("Original Image 2");
subplot(3,3,5);
imshow(andOperation);
title("Image 1 AND Image 2");

// Part (b)
img = imread("C:\Users\RMC-1\Desktop\flower.jpg");
img1 = imresize(img, [3648, 4607])
img2 = imread("C:\Users\RMC-1\Desktop\butterfly.jpg");
andOperation = bitor(img1, img2)
figure
subplot(3,3,1),imshow(img1);
title("Original Image 1");
subplot(3,3,3),imshow(img2);
title("Original Image 2");
subplot(3,3,5);
imshow(andOperation);
title("Image 1 OR Image 2");
```

```
// Part (c)
BWimg1 = rgb2gray(img1);
BWimg2 = rgb2gray(img2);
IntersectOperation = bitand(BWimg1, BWimg2);
figure
subplot(3,3,1),imshow(img1);
title("Original Image 1");
subplot(3,3,3),imshow(img2);
title("Original Image 2");
subplot(3,3,5);
imshow(IntersectOperation);
title("Image 1 INTERSECTION Image 2");

// Part (d)
NotOperation = bitcmp(img1);
figure
subplot(3,3,1),imshow(img1);
title("Original Image 1");
subplot(3,3,3),imshow(NotOperation);
title("Not Operation");
```

Original Image 1

Original Image 2

Image 1 AND Image 2



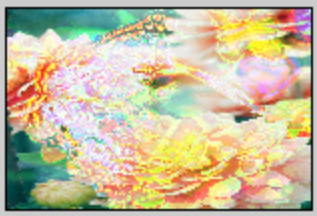
Original Image 1

Not Operation

Original Image 1

Original Image 2

Image 1 OR Image 2



Original Image 1

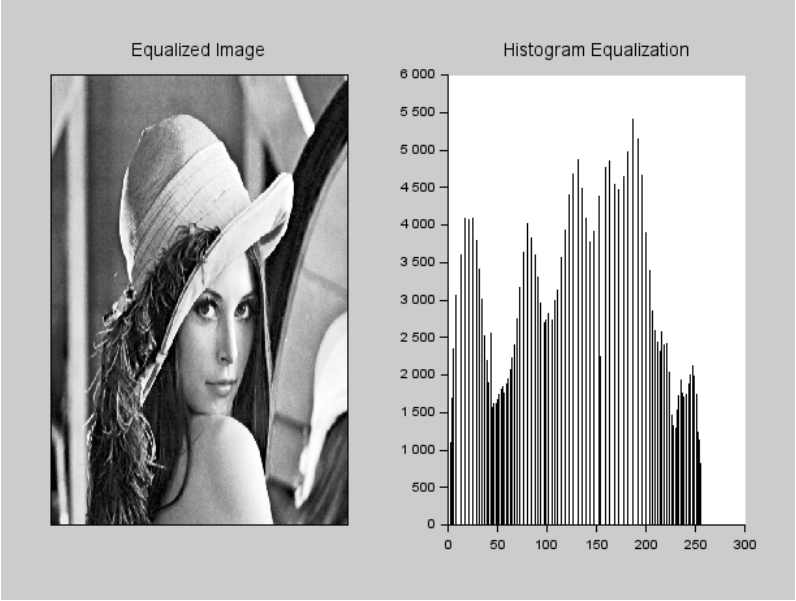
Original Image 2

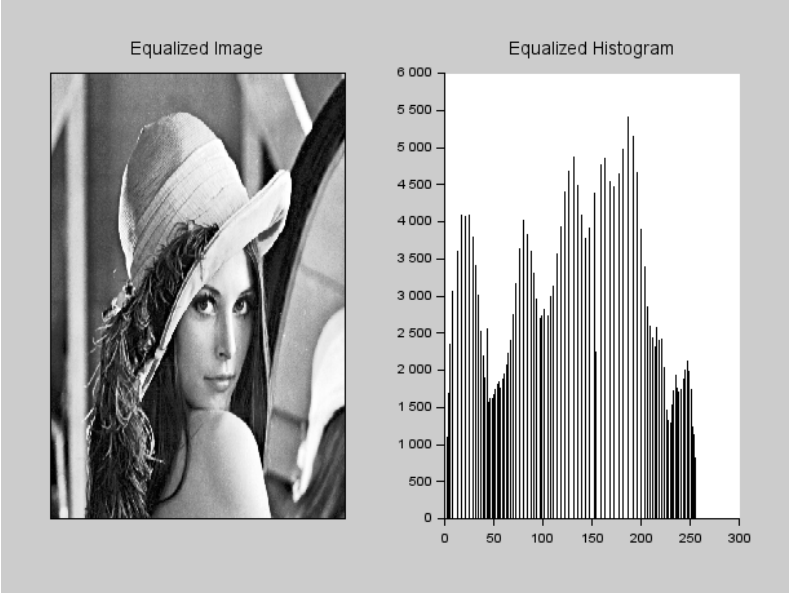
Image 1 INTERSECTION Image 2

Question 5

5. To write a program for histogram calculation and equalization using
   a. Standard MATLAB function
   b. Program without using standard MATLAB functions

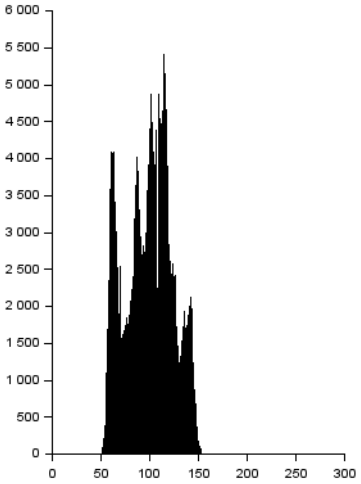
```
// QUESTION 5

// Part a
lena_img = imread(fullpath(getIPCVpath() + "/images/Lena_dark.png"));
subplot(1,2,1),title("Original Image "),imshow(lena_img);
subplot(1,2,2),title("Original Histogram"),imhist(lena_img,[],1);
h_img = imhistequal(lena_img);
figure();
subplot(1,2,1),title("Equalized Image "),imshow(h_img);
subplot(1,2,2),title("Equalized Histogram"),imhist(h_img,[],1);

// Part b
function eq_img=histeq(lena_img)
 [freq, bins] = imhist(lena_img,256);
 bins = 255;
 [row, col] = size(lena_img);
 freq = cumsum(freq);
 npixels = prod(size(lena_img));
 output = round(bins.*(freq./npixels));
 // Creating Equalized Image
 for i = 1:row
 for j = 1:col
 eq_img(i,j) = output(lena_img(i,j) + 1);
 end
 end
endfunction
he_img = uint8(histeq(lena_img));
figure
subplot(1,2,1),title("Equalized Image "),imshow(he_img);
subplot(1,2,2),title("Histogram Equalization"),imhist(he_img, [], 1);
```


Equalized Image
Histogram Equalization
6 000
5 500
5 000
4 500
4 000
3 500
3 000
2 500
2 000
1 500
1 000
500
0
0
50
100
150
200
250
300


Equalized Image
Equalized Histogram
6 000
5 500
5 000
4 500
4 000
3 500
3 000
2 500
2 000
1 500
1 000
500
0
0
50
100
150
200
250
300

Original Image

Original Histogram

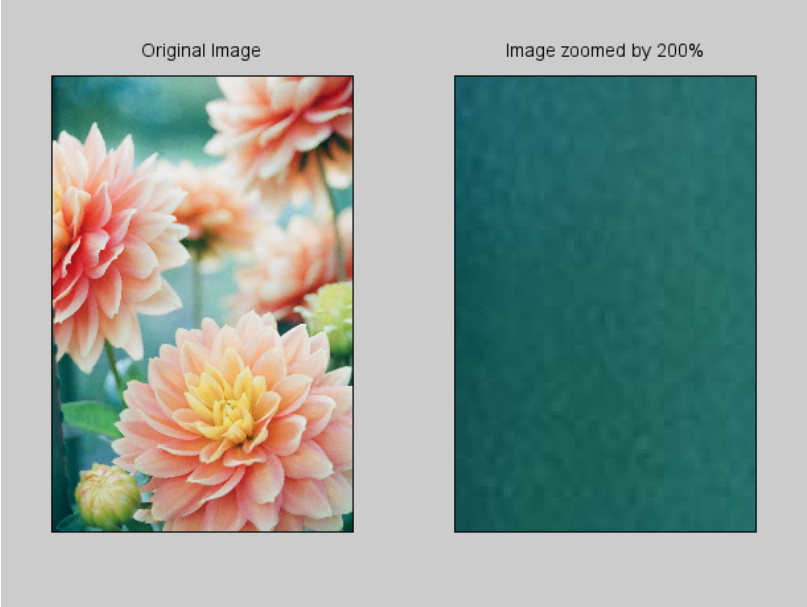


Question 6

6. To write and execute program for geometric transformation of image
   a. Translation
   b. Scaling
   c. Rotation
   d. Shrinking
   e. Zooming

```
// QUESTION 6

img = imread("C:\Users\RMC-1\Desktop\flower.jpg");
imshow(size(img));
// Part a
// Translation for (20,20)
mat = [   1   0 0;...
          0   1 0;...
        160 160 1];
transformedImg = imtransform(img, mat, 'affine');

// Part b
// Scaling Factor (2,4)
width = size(img,'c');
height = size(img,'r');
w = 2;
h = 4;
mat2 = [  w   0;...
          0   h;...
          0   0];

ScaledImg = imtransform(img, mat2, 'affine', width*w, height*h);

// Part c
// Rotating for 45 degrees
rotateFactor = 45;
RotatedImg = imrotate(img, rotateFactor, 'bilinear');

// Part d
/*[row col] = size(img);
matrix = zeros(row, col, 'uint8');
ShrunkImg = rgb2gray(imresize(img, 0.5));
```

```
matrix(48:143, 40:120) = ShrunkImg;
figure
subplot(1, 2, 1), title('Original Image'), imshow(img);
subplot(1, 2, 2), title('Image Shrinked by 50%'), imshow(matrix);*/

// Part e
im2 = imresize(img, 2);
figure
subplot(121), title('Original Image'), imshow(img);
subplot(122), title('Image zoomed by 200%'), imshow(im2(96:287, 81:241, :));

figure
subplot(2,3,1), imshow(img);
title('Original Image');
subplot(2,3,2), imshow(transformedImg);
title('Translation for (20,20)');
subplot(2,3,4), imshow(RotatedImg);
title('Rotating for 45 degrees');
subplot(2,3,5), imshow(ShrunkImg);
title('Shrinking by factor of 2');
figure
imshow(ScaledImg);
title('Scaling for (2,4)');
```


Original Image
Image zoomed by 200%

Original Image

Translation for (20,20)

Rotating for 45 degrees

Scaling for (2,4)

Question 7

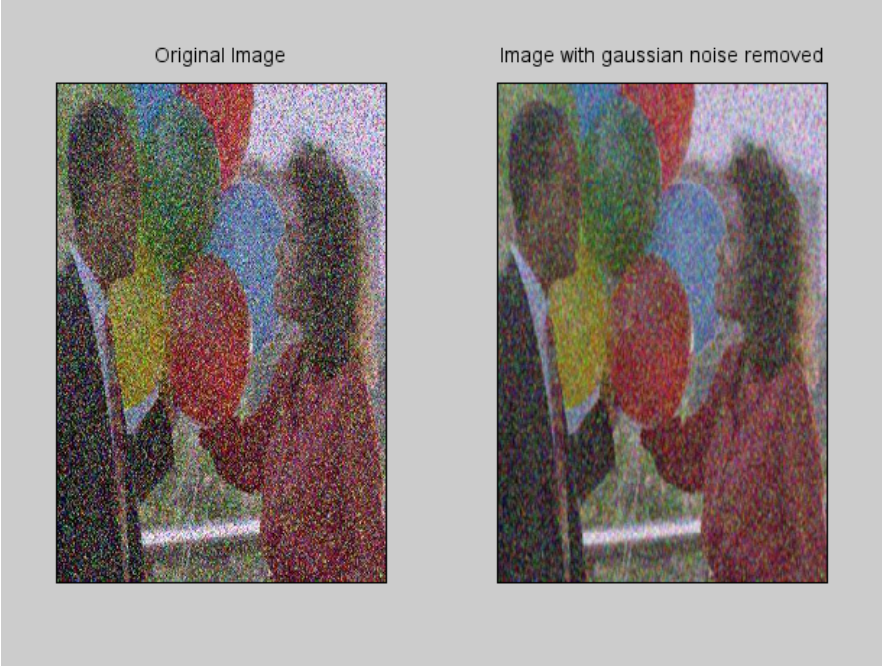
7. To understand various image noise models and to write programs for
   a. image restoration
   b. Remove Salt and Pepper Noise
   c. Minimize Gaussian noise
   d. Median filter

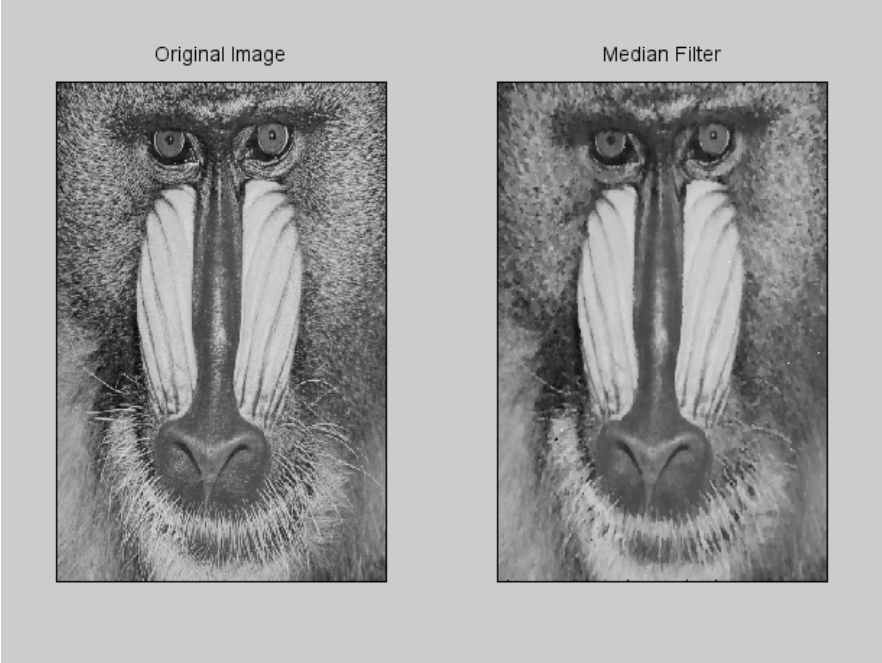
```
// QUESTION 7

img = imread("C:\Users\RMC-1\Desktop\noisyImg.jpg");

// Part a
filter = fspecial('gaussian', [8,8], 2);
filteredImg = imfilter(img, filter);
figure
subplot(1, 2, 1), title('Noisy Image'), imshow(img);
subplot(1, 2, 2), title('Filtered Image'), imshow(filteredImg);

// Part b
saltAndPepperFiltered = immedian(img, 5)
figure
subplot(1, 2, 1), title('Original Image'), imshow(img);
subplot(1, 2, 2), title('Image with salt and pepper removed'),
imshow(saltAndPepperFiltered);

// Part c
filter2 = fspecial('average', 3);
filteredImg2 = imfilter(img, filter2);
figure
subplot(1, 2, 1), title('Original Image'), imshow(img);
subplot(1, 2, 2), title('Image with gaussian noise removed'),
imshow(filteredImg2);

// Part d
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(121), title('Original Image'), imshow(im2);
subplot(122), title("Median Filter"), imshow(img1(2:r+1, 2:c+1));
```


Noisy Image
Filtered Image


Original Image
Image with salt and pepper removed


Original Image
Image with gaussian noise removed


Original Image
Median Filter

Question 8

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

```
// QUESTION 8

// Low pass filter
img = imread("C:\Users\RMC-1\Desktop\lena.png");

// filter 1
gaussianFilter = fspecial('gaussian');
lowPass1 = imfilter(img, gaussianFilter);

// filter 2
gaussianFilter2 = fspecial('gaussian', [10,10], 10);
lowPass2 = imfilter(img, gaussianFilter2);

// filter 3
gaussianFilter3 = fspecial('gaussian',[25,25], 31);
lowPass3 = imfilter(img, gaussianFilter3);

figure
subplot(2,2,1), title('Original Image'), imshow(img);
subplot(2,2,2), title('Default Gaussian kernel'), imshow(lowPass1);
subplot(2,2,3), title('Gaussian kernel with 10 * 10 with sigma =
10'), imshow(lowPass2);
subplot(2,2,4), title('Gaussian kernel with 25 * 25 with sigma =
31'), imshow(lowPass3);

// High pass filter
Sharpeningfilter = fspecial('laplacian');
highPass1 = imfilter(img, Sharpeningfilter);
figure
subplot(1,2,1), title('Original Image'), imshow(img);
subplot(1,2,2), title('Laplacian Filter'), imshow(highPass1);
```


Original Image
Laplacian Filter


Original Image
Default Gaussian kernel
Gaussian kernel with 10 * 10 with sigma = 10
Gaussian kernel with 25 * 25 with sigma = 31

Question 9

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

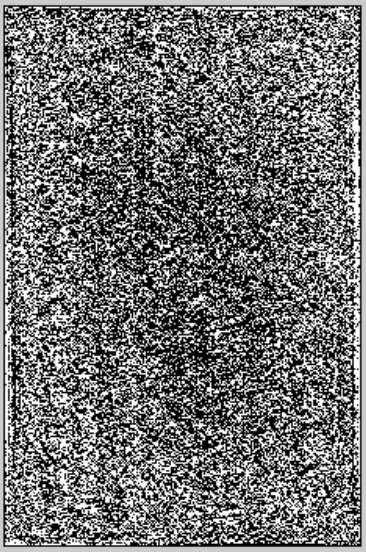
```
// QUESTION 9

// Part a -> discrete fourier transform

img = imread("C:\Users\RMC-1\Desktop\lena.png");
ft_img = fft(double(img));
figure
subplot(1,2,1), title('Original Image'), imshow(img);
subplot(1,2,2), title('Direct Fourier Transformed
Image'),imshow(ft_img);

// Part b -> using butterworth low pass and high pass filters
lowPass = mkfftfilter(img, 'butterworth1', 0.3);
highPass = 1 - lowPass;
figure
subplot(121), title('DFT Butterworth Low Pass Image'),
imshow(lowPass);
subplot(122), title('DFT Butterworth High Pass
Image'),imshow(highPass);

// Part c -> reconstructing the image
reconstructedImg1 = ft_img .* fftshift(lowPass);
butterworthLow = uint8(ifft(reconstructedImg1));
reconstructedImg2 = ft_img .* fftshift(highPass);
butterworthHigh = uint8(ifft(reconstructedImg2));
figure
subplot(1, 2, 1), title('DFT Butterworth Low Pass Image'),
imshow(butterworthLow);
subplot(1, 2, 2), title('DFT Butterworth High Pass
Image'),imshow(butterworthHigh);
```

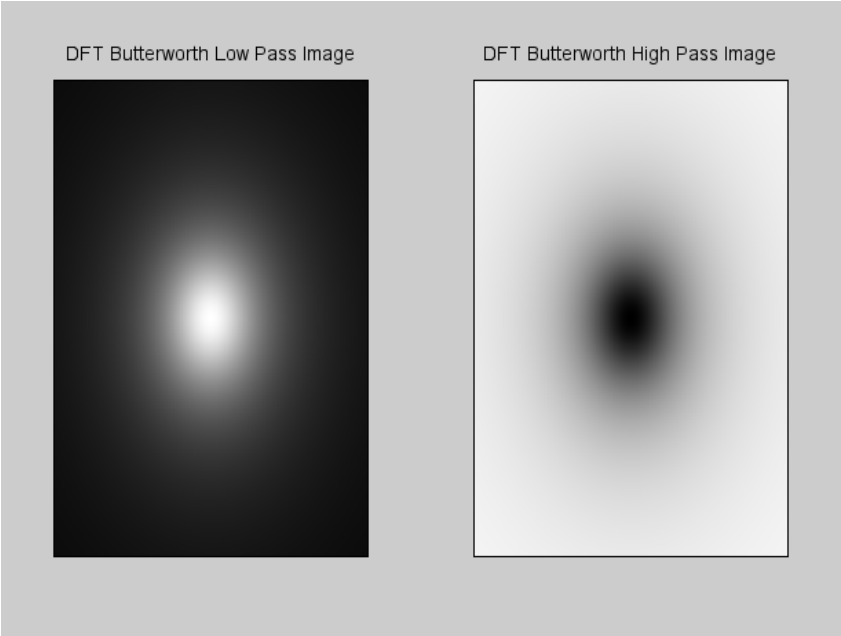
DFT Butterworth Low Pass Image

DFT Butterworth High Pass Image

Original Image

Direct Fourier Transformed Image




DFT Butterworth Low Pass Image
DFT Butterworth High Pass Image

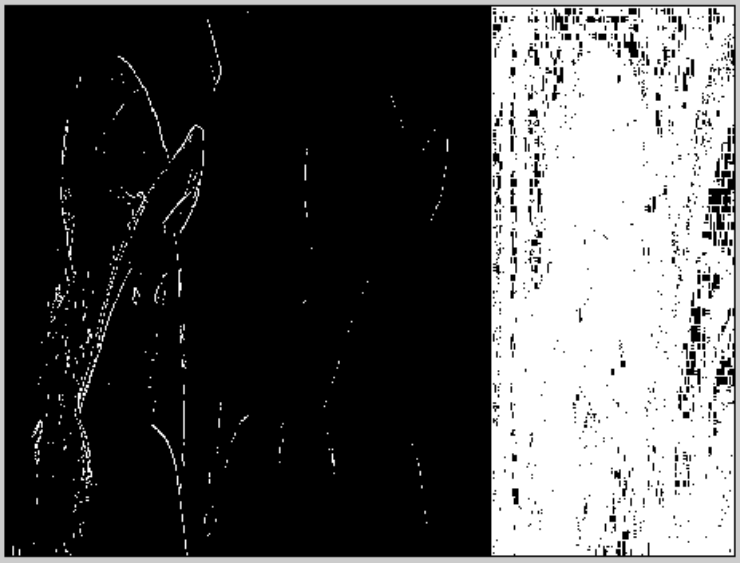
Question 10

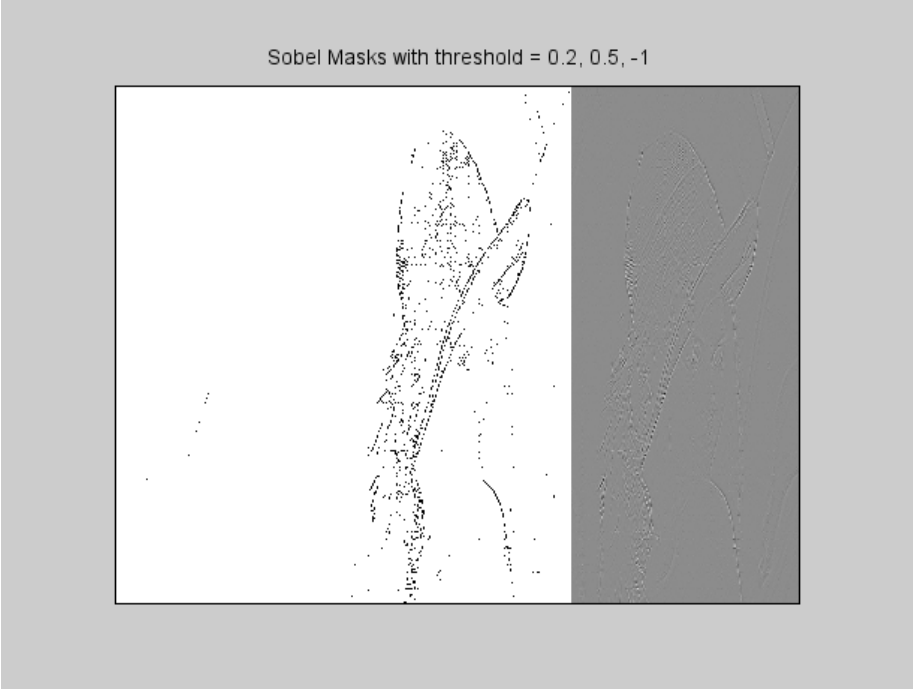
10. Write a program in C and MATLAB/SCILAB for edge detection using different edge detection mask

```
//  QUESTION 10

img = imread("C:\Users\RMC-1\Desktop\lena.png");
figure
title("Original Image");
imshow(img);

// Sobel operators
sobel = edge(img); // 0.2(Default)
sobel1 = edge(img, thresh = 0.5);
sobel2 = edge(img, thresh = -1);
montage = [sobel sobel1 sobel2];
figure
title("Sobel Masks with threshold = 0.2, 0.5, -1");
imshow(montage);

// Prewitt operators
prewitt = edge(img, 'prewitt');
prewitt1 = edge(img, 'prewitt', thresh = 0.5);
prewitt2 = edge(img, 'prewitt', thresh = -1);
montage1 = [prewitt prewitt1 prewitt2];
figure
title("prewitt Masks with threshold = 0.2, 0.5, -1");
imshow(montage1);
```

Original Image

prewitt Masks with threshold = 0.2, 0.5, -1




Sobel Masks with threshold = 0.2, 0.5, -1

Question 11

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

```
// QUESTION 11

img = imread("C:\Users\RMC-1\Desktop\shape.jpg");
imgBin = im2bw(img, 0.5);
structuringElement1 = imcreatese('cross',3, 3);
structuringElement2 = imcreatese('ellipse',3, 3);
structuringElement3 = imcreatese('rect',3, 3);

// erosion
e1 = imerode(imgBin, structuringElement1);
e2 = imerode(imgBin, structuringElement2);
e3 = imerode(imgBin, structuringElement3);

// Plotting
figure
subplot(2,2,3), title('Rectegular Erosion'), imshow(e1);
subplot(2,2,2), title('Ellipctical Erosion'), imshow(e2);
subplot(2,2,1), title('Cross Structure Erosion'), imshow(e3);

// dilation
d1 = imdilate(imgBin, structuringElement1);
d2 = imdilate(imgBin, structuringElement2);
d3 = imdilate(imgBin, structuringElement3);
```

```
// Plotting
figure
subplot(2,2,3), title('Rectegular Dilation'), imshow(d1);
subplot(2,2,2), title('Ellipctical Dilation'), imshow(d2);
subplot(2,2,1), title('Cross Structure Dilation'), imshow(d3);
```