**LAB#07**

**Object:** Introduction to image processing.

**TASK #01:**

Become familiar with matlab basic commands & write a short note on them.

.imread .imwrite .im2bw

.imshow .imresize . rgb2gray

**NOTES:**

1. **.imread:**

imread Read image from graphics file.

A = imread(FILENAME,FMT) reads a grayscale or color image from the file

specified by the string FILENAME. If the file is not in the current

directory, or in a directory on the MATLAB path, specify the full

pathname.

1. **.imwrite:**

imwrite Write image to graphics file.

imwrite(A,FILENAME,FMT) writes the image A to the file specified by

FILENAME in the format specified by FMT.

1. **.imshow:**

imshow Display image in Handle Graphics figure.

imshow(I) displays the grayscale image I.

imshow(I,[LOW HIGH]) displays the grayscale image I, specifying the display

range for I in [LOW HIGH]. The value LOW (and any value less than LOW)

displays as black, the value HIGH (and any value greater than HIGH) displays

as white. Values in between are displayed as intermediate shades of gray,

using the default number of gray levels. If you use an empty matrix ([]) for

[LOW HIGH], imshow uses [min(I(:)) max(I(:))]; that is, the minimum value in

I is displayed as black, and the maximum value is displayed as white.

imshow(RGB) displays the truecolor image RGB.

imshow(BW) displays the binary image BW. imshow displays pixels with the

value 0 (zero) as black and pixels with the value 1 as white.

1. **.imresize:**

imresize Resize image.

B = imresize(A, SCALE) returns an image that is SCALE times the

size of A, which is a grayscale, RGB, or binary image.

B = imresize(A, [NUMROWS NUMCOLS]) resizes the image so that it has

the specified number of rows and columns. Either NUMROWS or NUMCOLS

may be NaN, in which case imresize computes the number of rows or

columns automatically in order to preserve the image aspect ratio.

1. **.rgb2gray:**

rgb2gray Convert RGB image or colormap to grayscale.

rgb2gray converts RGB images to grayscale by eliminating the

hue and saturation information while retaining the

luminance.

1. **.im2bw:**

im2bw Convert image to binary image by thresholding.

im2bw produces binary images from indexed, intensity, or RGB images. To do

this, it converts the input image to grayscale format (if it is not already

an intensity image), and then converts this grayscale image to binary by

thresholding. The output binary image BW has values of 1 (white) for all

pixels in the input image with luminance greater than LEVEL and 0 (black)

for all other pixels. (Note that you specify LEVEL in the range [0,1],

regardless of the class of the input image.)

**TASK #02:** Write a code to import RGB image from indemos library & display the images.

**CODING:**

C=imread(‘peppers.png’)

Imshow(c)

**FIGURE:**

**TASK #03:** Resize given image to 256 x 256.

**CODING:**

c=imread('peppers.png')

subplot(1,2,1)

imshow(c)

title('Original')

d=imresize(c,[256 256])

subplot(1,2,2)

imshow(d)

title('Resize')

**FIGURE:**

**TASK #04:**

1. Convert given image to gray scale
2. Convert gray image to black & white image.

**CODING:**

1. Convert given image to gray

c=imread('peppers.png')

subplot(2,1,1)

imshow(c)

d=rgb2gray(c)

subplot(2,1,2)

imshow(d)

**FIGURE:**



1. Convert gray image to black & white image.

c=imread('peppers.png')

imshow(c)

d=rgb2gray(c)

subplot(2,1,1)

imshow(d)

e=im2bw(d)

subplot(2,1,2)

imshow(e)

**FIGURE:**

**TASK #05:** Separate color image to R,G & B planes & plot them using Subplots.

**CODING:**

c=imread('peppers.png')

subplot(4,1,1)

imshow(c)

title('original')

d=c(:,:,1)

subplot(4,1,2)

imshow(d)

title('RED')

e=c(:,:,2)

subplot(4,1,3)

imshow(e)

title('GREEN')

g=c(:,:,3)

subplot(4,1,4)

imshow(g)

title('BLUE')

**FIGURE:**

**TASK #06:** Generate coding of the following image.

**CODING:**

r=130;

c=130;

for i=1:r;

for j=1:c;

if i==j

b(i,j)=0;

elseif abs(i-j)==1;

b(i,j)=0.5;

else

b(i,j)=1;

end

end

end

imshow(b)

**FIGURE:**



**TASK #07:**  Generate coding of the following image.

**CODING:**

r=130;

c=130;

for i=r:1;

for j=c:1;

if i==j

b(i,j)=0;

elseif abs(j-i)==1;

b(i,j)=0.5;

else

b(i,j)=1;

end

end

end

e=fliplr(b)

imshow(e)

**FIGURE:**



**TASK #08:**  Generate coding of the following image.

**CODING:**

r=130;

c=130;

for i=1:r;

for j=1:c;

m=mod(i,4)

if m==0

b(i,j)=0;

else

b(i,j)=1;

end

end

end

imshow(b)

**FIGURE:**

**TASK #09:**  Generate coding of the following image.



**CODING:**

r=130;

c=130;

for i=1:r;

for j=1:c;

m=mod(i,4)

if m==0

b(i,j)=0;

else

b(i,j)=1;

end

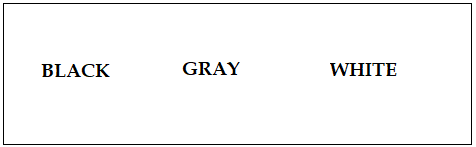
end

end

e=rot90(b)

imshow(e)

**FIGURE:**

**TASK #10:**  Generate coding of the following image.

**CODING:**

for m=1:100;

for k=0;

for n=1:1000

q(m,n)=k;

k=k+0.0010;

end

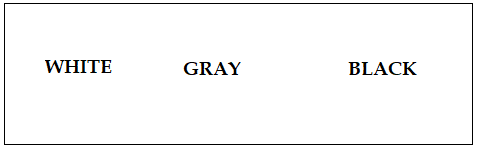
end

end

imshow(q)

**FIGURE:**



**TASK #11:**  Generate coding of the following image.

**CODING:**

for m=1:100;

for k=1;

for n=1:1000

q(m,n)=k;

k=k-0.0010;

end

end

end

imshow(q)

**FIGURE:**

