SWAMI KESHVANAND INSTITUTE OF TECHNOLOGY, MANAGEMENT & GRAMOTHAN, JAIPUR

SESSION 2021-22

Digital Image Processing Lab(6CS4-21) Learning Material

Computer Science Engineering

Prepared By:
Dr. Vinay Kanungo
Dr. Niketa Sharma
Mrs. Neha Mathur

RAJASTHAN TECHNICAL UNIVERSITY

Detailed Syllabus B. Tech. CS VI Sem 2021-2022

6CS4-21: Digital Image Processing Lab

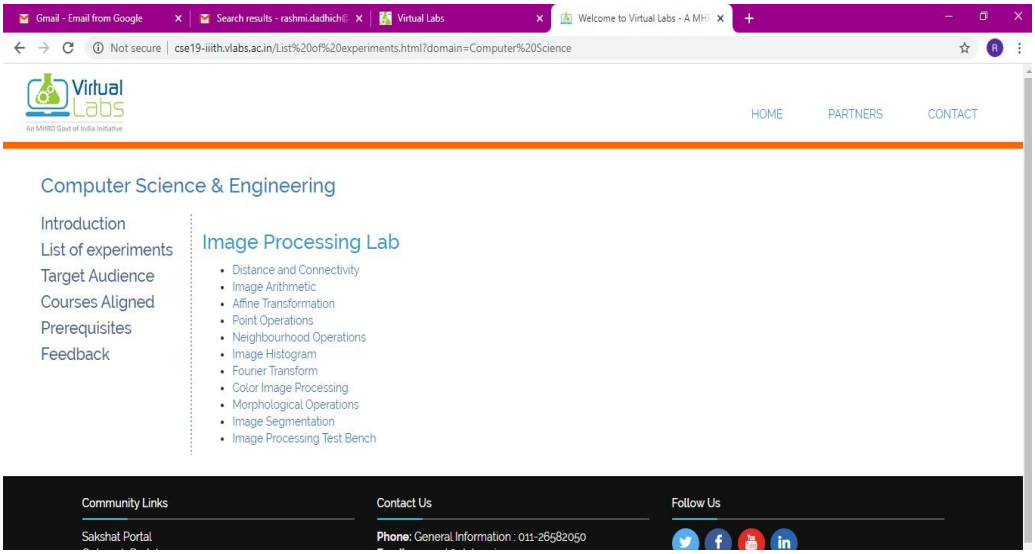
Credit: 1.5 **Max. Marks: 75(IA:45, ETE:30)**
0L+0T+3P **End Term Exam: 2 Hours**

SN	List of Experiments
1	Point-to-point transformation. This laboratory experiment provides for thresholding an image and the evaluation of its histogram. Histogram equalization. This experiment illustrates the relationship among the intensities (gray levels) of an image and its histogram.
2	Geometric transformations. This experiment shows image rotation, scaling, and translation. Two-dimensional Fourier transform
3	Linear filtering using convolution. Highly selective filters.
4	Ideal filters in the frequency domain. Non Linear filtering using convolutional masks. Edge detection. This experiment enables students to understand the concept of edge detectors and their operation in noisy images.
5	Morphological operations: This experiment is intended so students can appreciate the effect of morphological operations using a small structuring element on simple binary images. The operations that can be performed are erosion, dilation, opening, closing, open-close, close-open.

Virtual Lab Contents

Students are advised to go though all the experiments listed below for better understanding of the course. Follow these Steps:

1. Go to the browser and use the following link to access the experiments
   http://cse19-iiith.vlabs.ac.in/List%20of%20experiments.html?domain=Computer%20Science



2. From the above list of experiments study the given experiments as mapped with RTU Syllabus

   a. Affine Transformation
   b. Point Operations
   c. Neighborhood Operations
   d. Image Histogram
   e. Fourier Transform
   f. Color Image Processing
   g. Morphological Operations
   h. Image Segmentation

EXPEREMENT-1

OBJECTIVE: Point-to-point transformation. This laboratory experiment provides for thresholding an image and the evaluation of its histogram. Histogram equalization. This experiment illustrates the relationship among the intensities (gray levels) of an image and its histogram.

1.1) To perform image enhancement

SOFTWARE: MATLAB 7.0.4.

THEORY:

In the MATLAB workspace, most images are represented as two-dimensional arrays (matrices), in which each element of the matrix corresponds to a single pixel in the displayed image. For example, an image composed of 200 rows and 300 columns of different colored dots stored as a 200-by-300 matrix. Some images, such as RGB, require a three-dimensional array, where the first plane in the third dimension represents the red pixel intensities, the second plane represents the green pixel intensities, and the third plane represents the blue pixel intensities.

This convention makes working with graphics file format images similar to working with any other type of matrix data. For example, you can select a single pixel from an image matrix using normal matrix subscripting:

I(2,15)

This command returns the value of the pixel at row 2, column 15 of the image I.

The following sections describe the different data and image types, and give details about how to read, write, work with, and display graphics images; how to alter the display properties and aspect ratio of an image during display; how to print an image; and how to convert the data type or graphics format of an image.

Data Types

MATLAB math supports three different numeric classes for image display:

- double-precision floating-point (double)

- 16-bit unsigned integer (uint16)
- 8-bit unsigned integer (uint8)

The image display commands interpret data values differently depending on the numeric class the data is stored in. 8-Bit and 16-Bit Images includes details on the inner workings of the storage for 8- and 16-bit images.

By default, most data occupy arrays of class double. The data in these arrays is stored as double-precision (64-bit) floating-point numbers. All MATLAB functions and capabilities work with these arrays.

For images stored in one of the graphics file formats supported by MATLAB functions, however, this data representation is not always ideal. The number of pixels in such an image can be very large; for example, a 1000-by-1000 image has a million pixels. Since at least one array element represents each pixel , this image requires about 8 megabytes of memory if it is stored as class double.

To reduce memory requirements, you can store image data in arrays of class uint8 and uint16. The data in these arrays is stored as 8-bit or 16-bit unsigned integers. These arrays require one-eighth or one-fourth as much memory as data in double arrays.

Bit Depth

MATLAB input functions read the most commonly used bit depths (bits per pixel) of any of the supported graphics file formats. When the data is in memory, it can be stored as uint8, uint16, ordouble. For details on which bit depths are appropriate for each supported format, see imread and imwrite.

CODE:

```
//for grey and color images
 clc;
 close all;
 clear all;
 rgb=imread('C:\Documents and Settings\All Users\Documents\My Pictures\Sample
 Pictures\Water lilies.jpg');
 figure(1)
 imshow(rgb)
```

```
gray=rgb2gray(rgb)
figure(2)
imshow(gray)
```

OUTPUT:

Fig: RGB Image

Fig: Gray Scale Image

RESULT: Gray/Colour images of different formats has been designed and implemented. Simulation results have been obtained in the form of images.

1.2) OBJECTIVE: To perform enhancement manually

SOFTWARE: MATLAB 7.0.4.

THEORY: The histogram in the context of image processing is the operation by which the occurrences of each intensity value in the image is shown. Normally, the histogram is a graph showing the number of pixels in an image at each different intensity value found in that image. For an 8-bit grayscale image there are 256 different possible intensities, and so the histogram will graphically display 256 numbers showing the distribution of pixels amongst those grayscale values. Histogram modification is a classical method for image enhancement, especially histogram equalization. Histogram equalization method is a self-acting process since it does not request any information, just only the probability of each intensity level of image. However, the enhanced image is obtained by the global area histogram equalization will cause an effect of intensity saturation in some areas.

Histogram Equalization

Histogram equalization is the technique by which the dynamic range of the histogram of an image is increased. Histogram equalization assigns the intensity values of pixels in the input image such that the output image contains a uniform distribution of intensities. It improves contrast and the goal of histogram equalization is to obtain a uniform histogram. This technique can be used on a whole image or just on a part of an image. Histogram equalization redistributes intensity distributions. If the histogram of any image has many peaks and valleys, it will still have peaks and valley after equalization, but peaks and valley will be shifted. Because of this, "spreading" is a better term than "flattening" to describe histogram equalization. In histogram equalization, each pixel is assigned a new intensity value based on its previous intensity level.

CODE:

```
% Plot histogram
clc
clear all;
close all;
i= imread('C:\Documents and Settings\All Users\Documents\My Pictures\Sample Pictures\Blue hills.jpg')
figure(1)
subplot(1,1,1);
imshow(i);

x=rgb2gray(i)
figure(2)
subplot(2,2,1);
imshow(x)
title('Gray Scale image');
subplot(2,2,2);
imhist(x);
title('Histrogram of gray scale image');
im3= histeq(x);
subplot(2,2,3);
imshow(im3);
title('Equalized image');
subplot(2,2,4);
imhist(im3);
title('Histrogram of Equalized image');
```

OUTPUT:

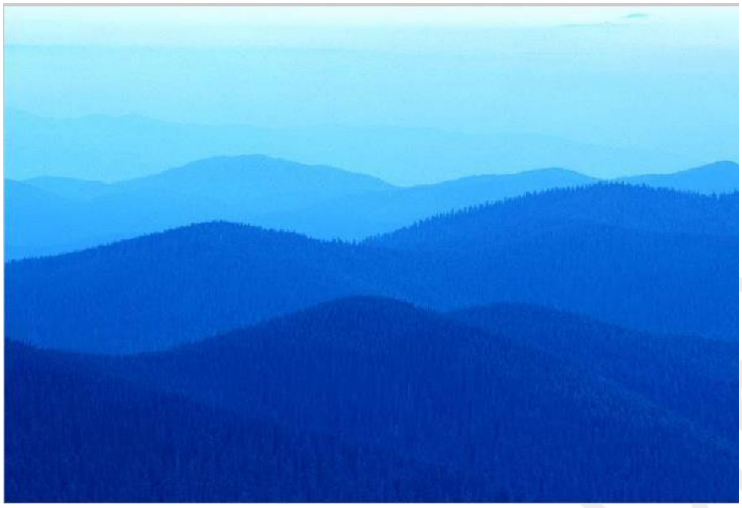


Fig: Original Image

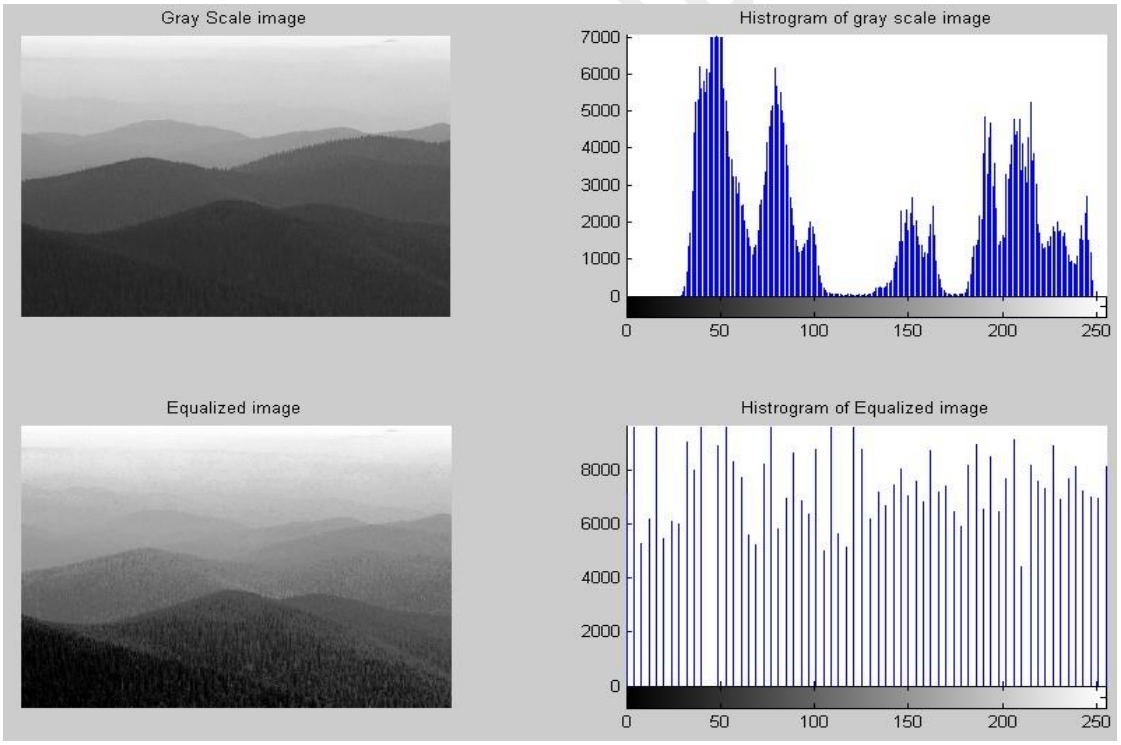


Fig: Gray Scale Image and Histogram Equalized Image

RESULT: Image Histogram and histogram equalization has been designed and implemented. Simulation results have been obtained in the form of images.

EXPERIMENT- 2

OBJECTIVE: Geometric transformations. This experiment shows image rotation, scaling, and translation. Two-dimensional Fourier transform.

Program 1:

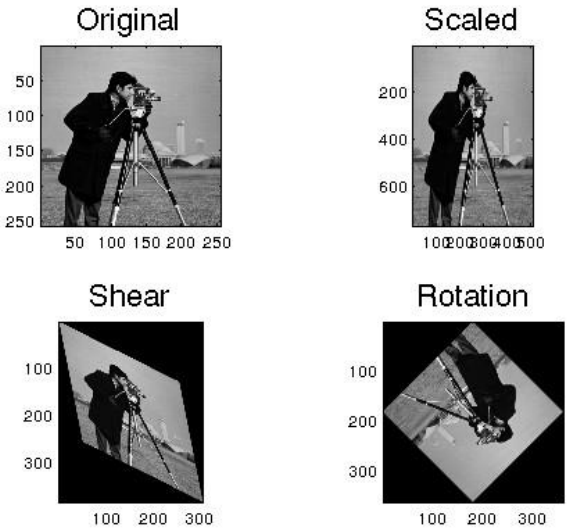
Geometric operations (resize, rotate, crop)

```
i=imread('C:\Users\student\Desktop\image24.jpg');

m=imresize (i, [600 600]);

subplot (3,3,1);
imshow(m);
title ('Original resized image');

c=imcrop(m);
subplot(3,3,2);
imshow(c);
title ('Cropped image by user');

d=imcrop (m, [150,200,250,350]);
subplot (3,3,3);
imshow(d);
title ('Cropped image by parameter');

e=imrotate(m, -120,'Bilinear','crop');
subplot (3,3,4);
imshow(e);
title ('rotated image');
```

Program 2:

```
I=imread('cameraman.tif');
I=double(I);

s=[2,3];
tform1 = maketform('affine',[s(1) 0 0; 0 s(2) 0; 0 0 1]);       % scaling
I1 = imtransform(I,tform1);

sh=[0.5 0.2];
tform2 = maketform('affine',[1 sh(1) 0; sh(2) 1 0; 0 0 1]);     % shear
I2 = imtransform(I,tform2);

theta=3*pi/4;                                          % rotation
A=[cos(theta) sin(theta) 0; -sin(theta) cos(theta) 0; 0 0 1];
```

```
tform3 = maketform('affine',A);
I3 = imtransform(I,tform3);

figure
subplot(2,2,1),imagesc(I),axis image
title('Original','FontSize',18)
subplot(2,2,2),imagesc(I1),axis image
title('Scaled','FontSize',18)
subplot(2,2,3),imagesc(I2),axis image
title('Shear','FontSize',18)
subplot(2,2,4),imagesc(I3),axis image
title('Rotation','FontSize',18)
colormap(gray)
```



EXPERIMENT- 3

OBJECTIVE: Linear filtering using convolution. Highly selective filters.

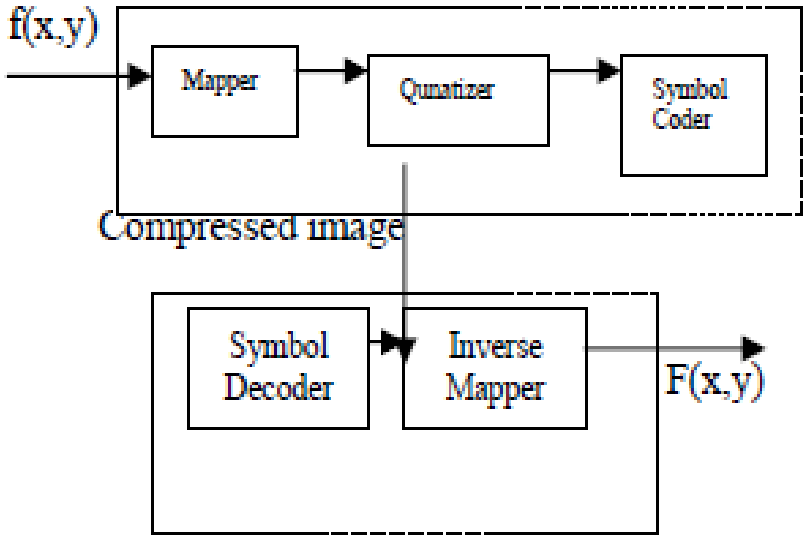
IMAGE COMPRESSION

Image compression addresses the problem of reducing the amount of data required to represent a digital image. It is a process intended to yield a compact representation of an image, thereby reducing the image storage/transmission requirements. Compression is achieved by the removal of one or more of the three basic data redundancies:

1. Coding Redundancy
2. Interpixel Redundancy
3. Psychovisual Redundancy

Coding redundancy is present when less than optimal code words are used. Interpixel redundancy results from correlations between the pixels of an image. Psychovisual redundancy is due to data that is ignored by the human visual system (i.e. visually non essential information). Image compression techniques reduce the number of bits required to represent an image by taking advantage of these redundancies. An inverse process called decompression (decoding) is applied to the compressed data to get the reconstructed image. The objective of compression is to reduce the number of bits as much as possible, while keeping

the resolution and the visual quality of the reconstructed image as close to the original image as possible. Image compression systems are composed of two distinct structural blocks : an encoder and a decoder.



Figure(1) Block diagram of image Encoder and Decoder

Image f(x,y) is fed into the encoder, which creates a set of symbols form the input data and uses them to represent the image. If we let n1 and n2 denote the number of information carrying units(usually bits) in the original and encoded images respectively, the compression that is achieved can be quantified numerically via the compression ratio,

$$CR = n1 / n2$$

The encoder is responsible for reducing the coding, interpixel and psychovisual Redundancies of input image. In first stage, the mapper transforms the input image into a format designed to reduce interpixel redundancies. The second stage, qunatizer block reduces the accuracy of mapper's output in accordance with a predefined criterion. In third and final stage, a symbol decoder creates a code for quantizer output and maps the output in accordance with the code. These blocks perform, in reverse order, the inverse operations of the encoder's symbol coder and mapper block. As quantization is irreversible, an inverse quantization is not included.

BENEFITS OF COMPRESSION

- It provides a potential cost savings associated with sending less data over switched telephone network where cost of call is really usually based upon its duration.
- It not only reduces storage requirements but also overall execution time.

- It also reduces the probability of transmission errors since fewer bits are transferred.
- It also provides a level of security against illicit monitoring.

IMAGE COMPRESSION TECHNIQUES

The image compression techniques are broadly classified into two categories Depending whether or not an exact replica of the original image could be Reconstructed using the compressed image.

These are:

1. Lossless technique
2. Lossy technique

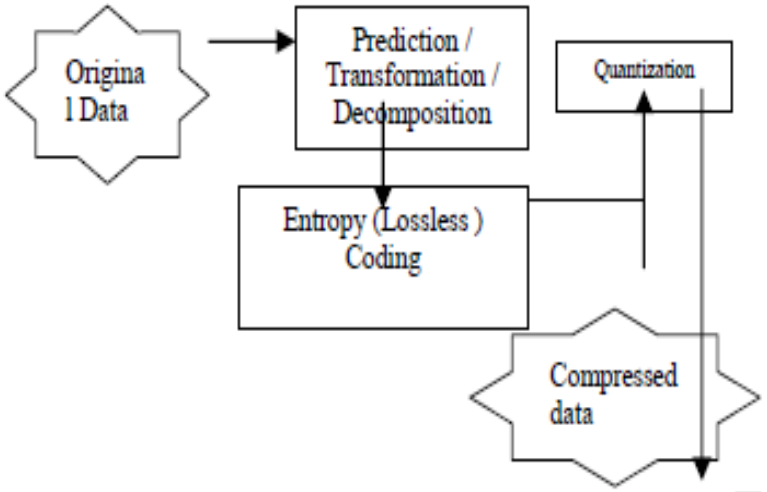
Lossless compression technique

In lossless compression techniques, the original image can be perfectly recovered from the compressed (encoded) image. These are also called noiseless since they do not add noise to the signal (image).It is also known as entropy coding since it use statistics/decomposition techniques to eliminate/minimize redundancy. Lossless compression is used only for a few applications with stringent requirements such as medical imaging. Following techniques are included in lossless compression:

1. Run length encoding
2. Huffman encoding
3. LZW coding
4. Area coding

Lossy compression technique

Lossy schemes provide much higher compression ratios than lossless schemes. Lossy scheme s are widely used since the quality of the reconstructed images is adequate for most applications .By this scheme, the decompressed image is not identical to the original image, but reasonably close to it.



Figure(2) Outline of lossy image compression

In this prediction – transformation – decomposition process is completely reversible .The quantization process results in loss of information. The entropy coding after the quantization step, however, is lossless. The decoding is a reverse process.

Firstly, entropy decoding is applied to compressed data to get the quantized data. Secondly, dequantization is applied to it. It not only reduces storage requirements but also reverse process. Firstly, entropy decoding is applied to compressed data to get the quantized data. Secondly, dequantization is applied to it & finally the inverse transformation to get the reconstructed image.

Major performance considerations of a lossy compression scheme include:

1. Compression ratio
2. Signal to noise ratio
3. Speed of encoding & decoding.

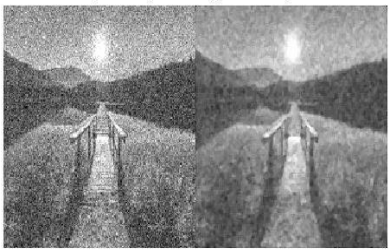
Program:

```
m1=imread('C:\Users\student\Desktop\pho.jpeg');
m2=imresize(m1,[180,180]);
subplot(2,2,1);
imshow(m2);
title('original');
m3=rgb2gray(m2);
subplot(2,2,2);
```

```
imshow(m3);
title('gray image');
n1=imnoise(m3,'gaussian');
subplot(2,2,3);
imshow(n1);
title('gaussain noise img');
n2=medfilt2(n1);
subplot(2,2,4);
imshowpair(n1,n2,'montage');
title('paired image of filter img and noise img');
figure(2);
n3=imnoise(m3,'gaussian',0.3);
subplot(2,2,1);
imshow(n3);
title('gaussian noise with parameter');
n5=medfilt2(n3);
subplot(2,2,2);
imshowpair(n3,n5,'montage');
title('paired image of filter img and noise img');
n8=imnoise(m3,'poisson');
subplot(2,2,3);
imshow(n8);
title('poisson noise');
n9=medfilt2(n8);
subplot(2,2,4);
imshowpair(n8,n9,'montage');
title('paired image of filter img and noise img');
figure(3);
n6=imnoise(m3,'salt & pepper');
subplot(2,2,1);
imshow(n6);
title('salt and pepper img');
n7=medfilt2(n6);
subplot(2,2,2);
imshowpair(n6,n7,'montage');
title('paired image of filter img and noise img');
n10=imnoise(m3,'salt & pepper',0.3);
subplot(2,2,3);
imshow(n10);
title('salt and pepper with parameter');
n11=medfilt2(n10);
subplot(2,2,4);
imshowpair(n10,n11,'montage');
title('paired image of filter img and noise img');
```

gaussian noise with parameter

paired image of filter img and noise img



poisson noise

paired image of filter img and noise img

original

gray image

gaussain noise img

paired image of filter img and noise img

salt and pepper img

paired image of filter img and noise img

salt and pepper with parameter

paired image of filter img and noise img

EXPERIMENT- 4

OBJECTIVE: Ideal filters in the frequency domain. Non-Linear filtering using convolutional masks. Edge detection. This experiment enables students to understand the concept of edge detectors and their operation in noisy images.

4.1) To perform segmentation of an image

SOFTWARE - MATLAB 7.0.4.

THEORY:

The RGB color model is an additive system in which each color is defined by the amount of red, green, and blue light emitted. In the RGB scheme, colors are represented numerically with a set of three numbers, each of which ranges from 0 to 255. White has the highest RGB value of (255, 255, 255) while black has the lowest value of (0, 0, 0). This is consistent with the additive nature of the RGB system, since white light is the presence of all colors of light, and black is the absense of all light.

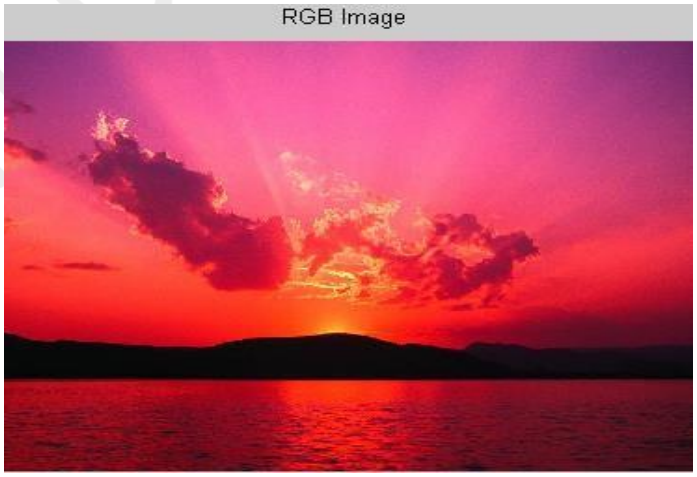
There are other three-parameter representations of colors. One such system is the HSI color model, which encodes colors according to their **H**ue, **S**aturation, and **I**ntensity. The HSI model is used by some graphics programs and color monitors as an alternative to, or alongside the RGB representation.

In the HSI system, the hue of a color is its angle measure on a color wheel. Pure red hues are 0°, pure green hues are 120°, and pure blues are 240°. (Neutral colors--white, gray, and black--are set to 0° for convenience.) Intensity is the overall lightness or brightness of the color, defined numerically as the average of the equivalent RGB values.

CODE:

```
clc;
close all;
clear all;
 F=imread('C:\Documents and Settings\All Users\Documents\My Pictures\Sample
 Pictures\Sunset.jpg')
 F=im2double(F);
```

```
r=F(:,:,1);
g=F(:,:,2);
b=F(:,:,3);
th=acos((0.5*((r-g)+(r-b)))./((sqrt((r-g).^2+(r-b).*(g-b)))+eps));
H=th;
H(b>g)=2*pi-H(b>g);
H=H/(2*pi);
S=1-3.*(min(min(r,g),b))./(r+g+b+eps);
I=(r+g+b)/3;
hsi=cat(3,H,S,I);
figure(1)
imshow(F)
title('RGB Image');
figure(2)
imshow(hsi)
title('HSI Image');
```



RESULT: Conversion for an RGB to HSI has been designed and implemented. Simulation results have been obtained in the form of images.

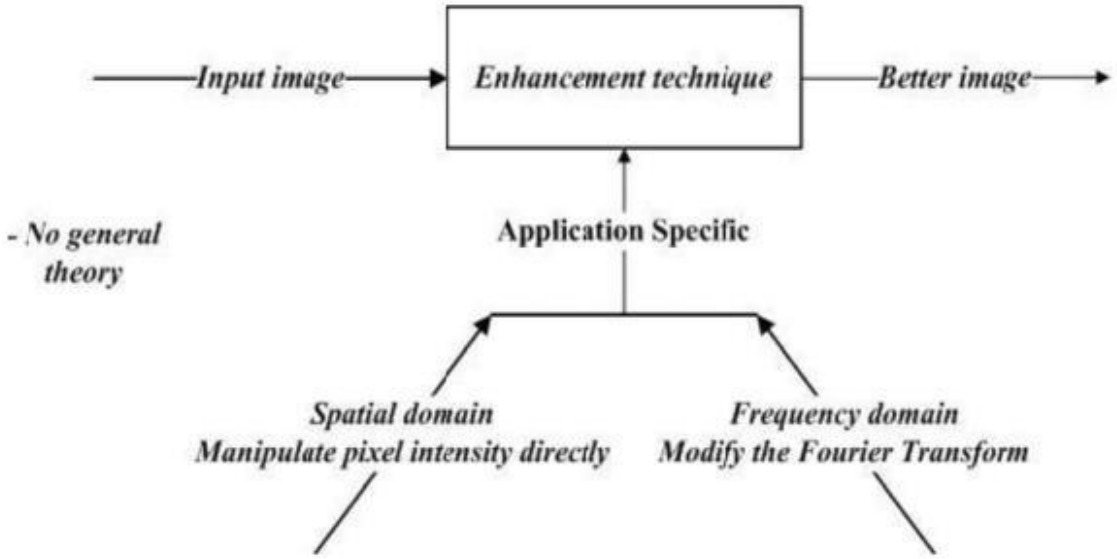
4.2) To Perform reading and displaying Gray/ Colour images of different formats

SOFTWARE - MATLAB 7.0.4.

THEORY:

Image Enhancement

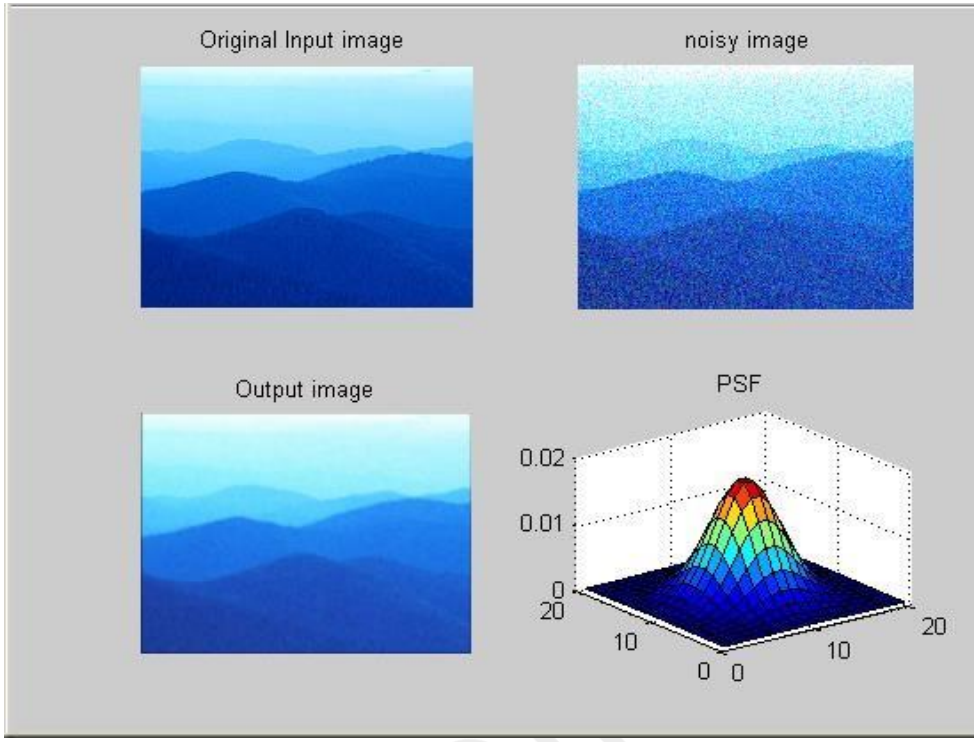
The main definition of enhancing is to make something greater in value, desirability or attractiveness. The term of enhancement implies a process to improve the visual quality of the image. Image Enhancement transforms images to provide better representation of the subtle details. The principal objective of enhancement is to process an image so that the result is more suitable than the original image for a specific application. Image enhancement processes consist of a collection of techniques that seek to improve the visual appearance of an image or to convert the image to a form better suited for analysis by a human or a machine. In an image enhancement system, there is no conscious effort to improve the fidelity of a reproduced image with regard to some ideal form of the image, as is done in image restoration.



CODE:

```
clc;
close all;
clear all;
i= imread('C:\Documents and Settings\All Users\Documents\My Pictures\Sample Pictures\Blue hills.jpg');
figure(1), subplot(2,2,1); imshow(i); title('Original Input image')
a= imnoise (i, 'gaussian', .1,.01);
subplot(2,2,2); imshow(a);title(' noisy image')
h= fspecial('gaussian', [15,15], 3);
% h1=rgb2gray(h)
out= imfilter(a, h, 'conv');
subplot(2,2,3); imshow(out);title('Output image')
subplot(2,2,4);
surf(1:15, 1:15, h), title('PSF')
```

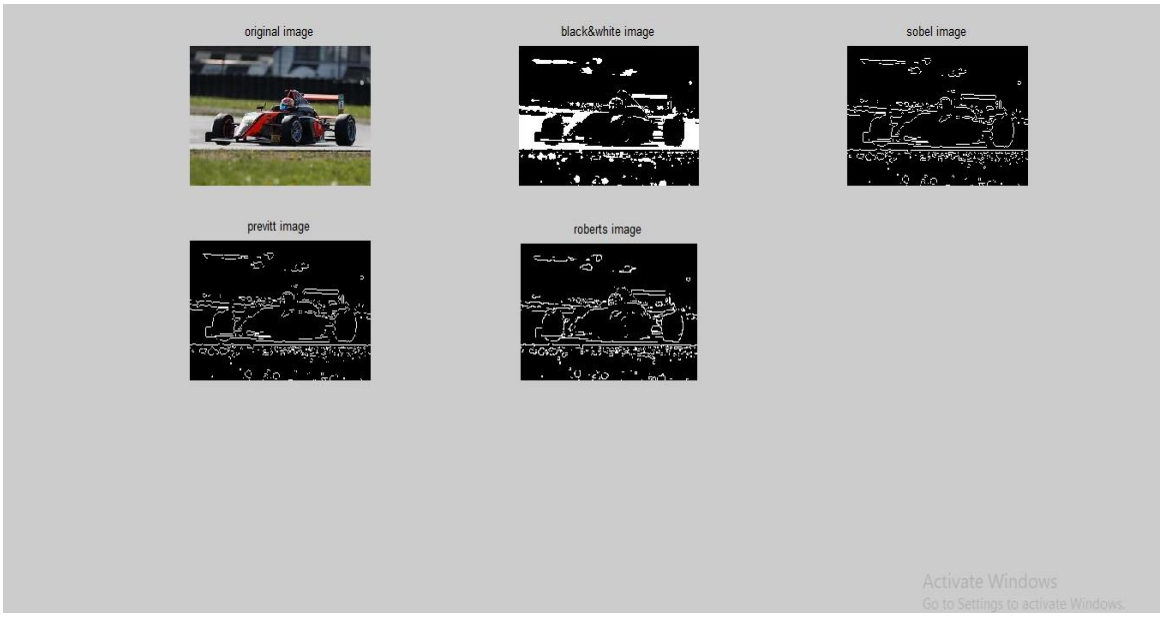
OUTPUT:



RESULT: Image filtering in Spatial and frequency domain has been designed and implemented. Simulation results have been obtained in the form of images.

4.3: Edge Detection Program:

```
clear;
clc;
i=imread('C:\Users\student\Desktop\oip2.jpg');
figure(1)
subplot(3,3,1)
imshow(i)
title('original image')
o=im2bw(i);
subplot(3,3,2)
imshow(o)
title('black&white image')
s=edge(o,'sobel');
subplot(3,3,3)
imshow(s)
title('sobel image')
s=edge(o,'prewitt');
subplot(3,3,4)
imshow(s)
title('previtt image')
s=edge(o,'roberts');
subplot(3,3,5)
imshow(s)
title('roberts image')
```



EXPERIMENT- 5

OBJECTIVE: Morphological operations : This experiment is intended so students can appreciate the effect of morphological operations using a small structuring element on simple binary images. The operations that can be performed are erosion , dilation , opening , closing , open – close, close – open.

```
m=imread('C:\Users\student\Desktop\ballsimg.jpg');
subplot(3,3,1);
imshow(m);
title('Original image');
g=rgb2gray(m);
subplot(3,3,2);
imshow(g);
title('Gray image');
bw=im2bw(g);
subplot(3,3,3);
imshow(bw);
title('Black and white image');
se=strel('disk',4);
mor=imopen(bw,se);
subplot(3,3,4);
imshow(mor);
title('imopen morphological operation');
se=strel('disk',4);
mor=imclose(bw,se);
subplot(3,3,5);
imshow(mor);
title(‘imclose morphological operation');

m=imread('C:\Users\student\Desktop\riceimg.jpg');
g=rgb2gray(m);
subplot(3,3,1);
imshow(g);
title('Gray image');
se=strel('disk',7);
mor=imopen(g,se);
subplot(3,3,3);
imshow(mor);
title('Background of image ');
se=strel('disk',5);
subplot(3,3,4);
mor2=g-mor;
imshow(mor2);
title('After removing Background of image');
```

```
I=mor2;
i=imadjust(I);
subplot(3,3,5);
imshow(i);
title('imadjust image');
bw=im2bw(i);
subplot(3,3,6);
imshow(bw);
title('Black and white image');
c=bwconncomp(bw,4);
disp(c);
disp(c.NumObjects);
f=false(size(bw));
f(c.PixelIdxList{50})=true;
subplot(3,3,7);
imshow(f);
title('Showing elements');
```

OUTPUT: