

Scilab Manual for  
Image Processing  
by Mr Gautam Pal  
Computer Engineering  
Tripura Institute of Technology<sup>1</sup>

Solutions provided by  
Mr R.Senthilkumar- Assistant Professor  
Electronics Engineering  
Institute of Road and Transport Technology

April 22, 2026

<sup>1</sup>Funded by a grant from the National Mission on Education through ICT, <http://spoken-tutorial.org/NMEICT-Intro>. This Scilab Manual and Scilab codes written in it can be downloaded from the "Migrated Labs" section at the website <http://scilab.in>



# Contents

List of Scilab Solutions	4
1 Distance and Connectivity: To understand the notion of connectivity and neighborhood defined for a point in an image.	6
2 Image Arithmetic - To learn to use arithmetic operations to combine images.	8
3 Image Arithmetic –To study the effect of these operations on the dynamic range of the output image.	11
4 Image Arithmetic –To study methods to enforce closure forces the output image to also be an 8 bit image.	15
5 Affine Transformation - To learn basic image transformation i) Translation ii) Rotation iii) Scaling	19
6 Affine Transformation –To learn the role of interpolation operation i) Bi-linear ii) Bi-cubic iii) nearest neighbor	21
7 Affine Transformation –To learn the effect of multiple transformations i) Significance of order in which one carried out	23
8 Point Operations - To learn image enhancement through point transformation-i)Linear transformation ii) Non-linear transformation	25
9 Neighborhood Operations - To learn about neighborhood	

operations and use them for i) Linear filtering ii) Non-linear filtering	28
10 Neighborhood Operations –To study the effect of the size of neighborhood on the result of processing	30
11 Image Histogram - To understand how frequency distribution can be used to represent an image.	32
12 Image Histogram –To study the correlation between the visual quality of an image with its histogram.	35
13 Fourier Transform: To understand some of the fundamental properties of the Fourier transform.	38
14 Colour Image Processing: To learn colour images are handled and processed i)Models for representing colour ii) Methods of proces	40
15 Morphological Operations: To understand the basics of morphological operations which are used in analyzing the form and shape de	43

# List of Experiments

Solution 1.1	Exp1 . . . . .	6
Solution 2.2	Exp2 . . . . .	8
Solution 3.3	Exp3 . . . . .	11
Solution 4.4	Exp4 . . . . .	15
Solution 5.5	Exp5 . . . . .	19
Solution 6.6	Exp6 . . . . .	21
Solution 7.7	Exp7 . . . . .	23
Solution 8.8	Exp8 . . . . .	25
Solution 9.9	Exp9 . . . . .	28
Solution 10.10	Exp10 . . . . .	30
Solution 11.11	Exp11 . . . . .	32
Solution 12.12	Exp12 . . . . .	35
Solution 13.13	Exp13 . . . . .	38
Solution 14.14	Exp14 . . . . .	40
Solution 15.15	Exp15 . . . . .	43
AP 1	2D Fast Fourier Transform . . . . .	46
AP 2	2D Inverse Fast Fourier Transform . . . . .	47
AP 3	Histogram Equalization . . . . .	48
AP 4	Gray Pixel value to Binary value . . . . .	48
AP 5	Total number of pixels in an image . . . . .	48
AP 6	Pad Array . . . . .	49

# List of Figures

3.1	Exp3	13
3.2	Exp3	14
4.1	Exp4	17
4.2	Exp4	18
8.1	Exp8	27
11.1	Exp11	34
14.1	Exp14	42
15.1	Exp15	45

# Experiment: 1

**Distance and Connectivity: To understand the notion of connectivity and neighborhood defined for a point in an image.**

Scilab code Solution 1.1 Exp1

```
1 //Prog1.Image Arithmetic – To learn to use
  arithmetic operations to combine images.
2 //Software version
3 //OS Windows7
4 //Scilab5.4.1
5 //Image Processing Design Toolbox 8.3.1–1
6 //Scilab Image and Video Processing toolbox
  0.5.3.1–2
7 clc;
8 clear;
9 close;
10 I = imread('C:\Users\senthilkumar\Desktop\
  Gautam_PAL_Lab\DIP_Lab2\cameraman.jpeg'); //SIVP
  toolbox
11 J = imread('C:\Users\senthilkumar\Desktop\
```

```
    Gautam_PAL_Lab\DIP_Lab2\rice.png'); //SIVP toolbox
12 IMA = imadd(I,J); //SIVP toolbox
13 figure
14 ShowImage(IMA, 'Image Addition '); //IPD toolbox
15 IMS = imabsdiff(I,J); //SIVP toolbox
16 figure
17 ShowImage(IMS, 'Image Subtraction '); //IPD toolbox
18 IMD = imdivide(I,J); //SIVP toolbox
19 IMD = imdivide(IMD,0.01); //SIVP toolbox
20 figure
21 ShowImage(uint8(IMD), 'Image Division '); //IPD toolbox
22 IMM = immultiply(I,I); //SIVP toolbox
23 figure
24 ShowImage(uint8(IMM), 'Image Multiply '); //IPD toolbox
```

---

## Experiment: 2

# Image Arithmetic - To learn to use arithmetic operations to combine images.

Scilab code Solution 2.2 Exp2

```
1 //Prog2.Distance and Connectivity: To understand the
   notion of connectivity
2 //and neighborhood defined for a point in an image.
3 //Software version
4 //OS Windows7
5 //Scilab5.4.1
6 //Image Processing Design Toolbox 8.3.1-1
7 //Scilab Image and Video Processing toolbox
   0.5.3.1-2
8 clc;
9 clear;
10 close;
11 //[1].Euclidean Distance between images and their
   histograms
12 I = imread('C:\Users\senthilkumar\Desktop\
   Gautam_PAL_Lab\DIP_Lab2\lenna.jpg');
13 J = imread('C:\Users\senthilkumar\Desktop\
```

```

    Gautam_PAL_Lab\DIP_Lab2\cameraman.jpeg')
14 h_I = CreateHistogram(I); //IPD toolbox
15 h_J = CreateHistogram(J); //IPD toolbox
16 I = double(I);
17 J = double(J);
18 E_dist_Hist = sqrt(sum((h_I-h_J).^2)); //Euclidean
    Distance between histograms of two images
19 E_dist_images = sqrt(sum((I(:)-J(:)).^2)); //
    Euclidean Distance between two images
20 disp(E_dist_images, 'Euclidean Distance between two
    images');
21 disp(E_dist_Hist, 'Euclidean Distance between
    histograms of two images')
22 // [2]. Connectivity - 8 connected to the background
23 exec('C:\Users\senthilkumar\Desktop\Gautam_PAL_Lab\
    gray2bin.sci')
24 Ibin = gray2bin(I);
25 Jbin = gray2bin(J);
26 //conversion of gray image into binary image
27 conn = [1,1,1;1,1,1;1,1,1]; //8-connectivity
28 exec('C:\Users\senthilkumar\Desktop\Gautam_PAL_Lab\
    numdims.sci')
29 num_dims = numdims(I);
30 exec('C:\Users\senthilkumar\Desktop\Gautam_PAL_Lab\
    padarray.sci')
31 B = padarray(Ibin);
32 global FILTER_ERODE;
33 StructureElement = CreateStructureElement('square',
    3);
34 B_eroded = MorphologicalFilter(B, FILTER_ERODE,
    StructureElement.Data); //IPD toolbox
35 //note: StructureElement.Data and conn both are same
    values
36 //except that StructureElement.Data is boolean
    either true or false
37 p = B & ~B_eroded;
38 [m,n] = size(p);
39 for i = num_dims:m+num_dims-2

```

```
40     for j = num_dims:n+num_dims-2
41         pout(i-1,j-1) = p(i,j);
42     end
43 end
44 figure
45 ShowImage(uint8(I), 'Gray Lenna Image')
46 figure
47 ShowImage(Ibin, 'Binary Lenna Image')
48 figure
49 ShowImage(pout, '8 neighbourhood connectivy in Lenna
    Image')
```

---

check Appendix [AP 4](#) for dependency:

gray2bin.sci

check Appendix [AP 5](#) for dependency:

numdims.sci

check Appendix [AP 6](#) for dependency:

padarray.sci

## Experiment: 3

**Image Arithmetic –To study the effect of these operations on the dynamic range of the output image.**

Scilab code Solution 3.3 Exp3

```
1 //Program 3:Image Arithmetic —To study the effect
  of these operations on the dynamic range of the
  output image.
2 //Software version
3 //OS Windows7
4 //Scilab5.4.1
5 //Image Processing Design Toolbox 8.3.1–1
6 //Scilab Image and Video Processing toolbox
  0.5.3.1–2
7 clc;
8 clear;
9 close;
10 I = imread('C:\Users\senthilkumar\Desktop\
  Gautam_PAL_Lab\DIP_Lab2\redrose.jpg');
11 J = imread('C:\Users\senthilkumar\Desktop\
```

```

    Gautam_PAL_Lab\DIP_Lab2\mistymorning.jpg');
12 I = imresize(I,[300,300], 'bicubic');
13 J = imresize(J,[300,300], 'bicubic');
14 K = imadd(I,J)
15 ShowColorImage(I,'Red Rose Color Image')
16 figure
17 ShowColorImage(J,'Misty Morning Color Image')
18 figure
19 ShowColorImage(K,'Color Images addition result image
    ')
20 I_gray = rgb2gray(I);
21 J_gray = rgb2gray(J);
22 figure
23 ShowImage(I_gray,'Red Rose Gray Image')
24 figure
25 ShowImage(J_gray,'Misty Morning Gray Image')
26 Imean = mean2(I_gray);
27 Jmean = mean2(J_gray);
28 Ithreshold = double(Imean)/double(max(I_gray(:)));
29 Jthreshold = double(Jmean)/double(max(J_gray(:)));
30 I_bw = im2bw(I,Ithreshold);
31 J_bw = im2bw(J,Jthreshold);
32 figure
33 ShowImage(I_bw,'Red Rose Binary Image')
34 figure
35 ShowImage(J_bw,'Misty Morning Binary Image')

```

---



Figure 3.1: Exp3  
13



Figure 3.2: Exp3

## Experiment: 4

**Image Arithmetic –To study methods to enforce closure forces the output image to also be an 8 bit image.**

Scilab code Solution 4.4 Exp4

```
1 //Prog4.Image Arithmetic —To study methods to
   enforce closure forces the output image to also
   be an 8 bit image.
2 //Software version
3 //OS Windows7
4 //Scilab5.4.1
5 //Image Processing Design Toolbox 8.3.1–1
6 //Scilab Image and Video Processing toolbox
   0.5.3.1–2
7 clc;
8 clear;
9 close;
10 I = imread('C:\Users\senthilkumar\Desktop\
   Gautam_PAL_Lab\DIP_Lab2\cameraman.jpeg');
11 J = imread('C:\Users\senthilkumar\Desktop\
```

```

    Gautam_PAL_Lab\DIP_Lab2\lenna.jpg ');
12 K = imabsdiff(I,J);
13 ShowImage(I, 'Cameraman Image')
14 figure
15 ShowImage(J, 'Lenna Image')
16 figure
17 ShowImage(K, 'Absolute Difference Between cameraman
    and Lenna Image')
18 L = imcomplement(K);
19 figure
20 ShowImage(L, 'Complement of difference Image K ')
21 rect = [20,30,200,200];
22 I_subimage = imcrop(I,rect);
23 J_subimage = imcrop(J,rect);
24 figure
25 ShowImage(I_subimage, 'Sub Image of Cameraman Image')
26 figure
27 ShowImage(J_subimage, 'Sub Image of Lenna Image')
28 a=2;
29 b =0.5;
30 M = imlincomb(a,I,b,J);
31 figure
32 ShowImage(M, 'Linear Combination of cameraman and
    Lenna Image')
33 N= imlincomb(b,I,a,J);
34 figure
35 ShowImage(N, 'Linear Combination of cameraman and
    Lenna Image')

```

---



Figure 4.1: Exp4



Figure 4.2: Exp4

## Experiment: 5

# Affine Transformation - To learn basic image transformation i) Translation ii) Rotation iii) Scaling

Scilab code Solution 5.5 Exp5

```
1 //Prog5. Affine Transformation – To learn basic
   image transformation
2 // i) Translation ii) Rotation iii) Scaling
3 //Software version
4 //OS Windows7
5 //Scilab5.4.1
6 //Image Processing Design Toolbox 8.3.1–1
7 //Scilab Image and Video Processing toolbox
   0.5.3.1–2
8 clc;
9 clear;
10 clc;
11 I = imread('C:\Users\senthilkumar\Desktop\
   Gautam_PAL_Lab\DIP_Lab2\lenna.jpg'); //size 256
   x256
```

```

12 [m,n] = size(I);
13 for i = 1:m
14     for j =1:n
15         //Scaling
16         J(2*i,2*j) = I(i,j);
17         //Rotation
18         p = i*cos(%pi/2)+j*sin(%pi/2);
19         q = -i*sin(%pi/2)+j*cos(%pi/2);
20         p = ceil(abs(p)+0.0001);
21         q = ceil(abs(q)+0.0001);
22         K(p,q)= I(i,j);
23         //shear transformation
24         u = i+0.2*j;
25         v = j;
26         L(u,v)= I(i,j);
27     end
28 end
29 figure
30 ShowImage(I,'original Image');
31 figure
32 ShowImage(J,'Scaled Image');
33 figure
34 ShowImage(K,'Rotated Image');
35 figure
36 ShowImage(L,'Shear transformed (x direction)Image');

```

---

## Experiment: 6

### Affine Transformation –To learn the role of interpolation operation i) Bi-linear ii) Bi-cubic iii) nearest neighbor

Scilab code Solution 6.6 Exp6

```
1 //Prog6. Affine Transformation —To learn the role
  of interpolation operation
2 // i) Bi-linear ii) Bi-cubic iii) nearest neighbor
3 //Software version
4 //OS Windows7
5 //Scilab5.4.1
6 //Image Processing Design Toolbox 8.3.1-1
7 //Scilab Image and Video Processing toolbox
  0.5.3.1-2
8 clc;
9 clear;
10 close;
11 I = imread('C:\Users\senthilkumar\Desktop\
  Gautam_PAL_Lab\DIP_Lab2\lenna.jpg'); //size 256
  x256
```

```

12 [m,n] = size(I);
13 for i = 1:m
14     for j =1:n
15         //Scaling
16         J(1.5*i,1.5*j) = I(i,j); //512x512 Image
17     end
18 end
19 I_nearest = imresize(J,[256,256]); // 'nearest' -
    nearest-neighbor interpolation
20 I_bilinear = imresize(J,[256,256], 'bilinear'); // '
    bilinear' - bilinear interpolation
21 I_bicubic = imresize(J,[256,256], 'bicubic'); // '
    bicubic' - bicubic interpolation
22 figure
23 ShowImage(uint8(I_nearest), 'nearest-neighbor
    interpolation');
24 figure
25 ShowImage(uint8(I_bilinear), 'bilinear - bilinear
    interpolation');
26 figure
27 ShowImage(uint8(I_bicubic), 'bicubic - bicubic
    interpolation');

```

---

## Experiment: 7

# Affine Transformation –To learn the effect of multiple transformations i) Significance of order in which one carried out

Scilab code Solution 7.7 Exp7

```
1 //Prog7.Affine Transformation —To learn the effect
   of multiple transformations i) Significance of
   order in which one carried out
2 //Software version
3 //OS Windows7
4 //Scilab5.4.1
5 //Image Processing Design Toolbox 8.3.1–1
6 //Scilab Image and Video Processing toolbox
   0.5.3.1–2
7 clc;
8 clear;
9 close
10 I = imread('C:\Users\senthilkumar\Desktop\
```

```

    Gautam_PAL_Lab\DIP_Lab2\lenna.jpg ');
11 [m,n] = size(I);
12 for i = 1:m
13     for j =1:n
14         //shear transformation and rotation
15         u = i+0.2*j;
16         v = 0.3*i+j;
17         M(u,v) = I(i,j);
18         //shear transformation, rotation and scaling
19         N(u*1.5,v*1.5)= I(i,j);
20     end
21 end
22 figure
23 ShowImage(I,'original Lenna Image')
24 figure
25 ShowImage(M,'Shear transformed+rotated Lenna Image')
26 figure
27 ShowImage(N,'Shear Transformed+rotated+scaled Lenna
    Image')

```

---

## Experiment: 8

# Point Operations - To learn image enhancement through point transformation-i) Linear transformation ii) Non-linear transformation

Scilab code Solution 8.8 Exp8

```
1 //Prog8.Point Operations – To learn image
  enhancement through point transformation
2 //i) Linear transformation ii) Non-linear
  transformation
3 //Software version
4 //OS Windows7
5 //Scilab5.4.1
6 //Image Processing Design Toolbox 8.3.1-1
7 //Scilab Image and Video Processing toolbox
  0.5.3.1-2
8 clc;
9 clear;
10 close;
```

```
11 I = imread('C:\Users\senthilkumar\Desktop\  
    Gautam_PAL_Lab\DIP_Lab2\rice.png');  
12 //(i). Linear Transformation  
13 //IMAGE NEGATIVE  
14 I = double(I);  
15 J = 255-I;  
16 figure  
17 ShowImage(I, 'Original Image')  
18 figure  
19 ShowImage(J, 'Linear Transformation-IMAGE NEGATIVE')  
20 //(ii) Non-linear transformation  
21 //GAMMA TRANSFORMATION  
22 GAMMA = 0.9;  
23 K = I.^GAMMA;  
24 figure  
25 ShowImage(K, 'Non-linear transformation-GAMMA  
    TRANSFORMATION')
```

---

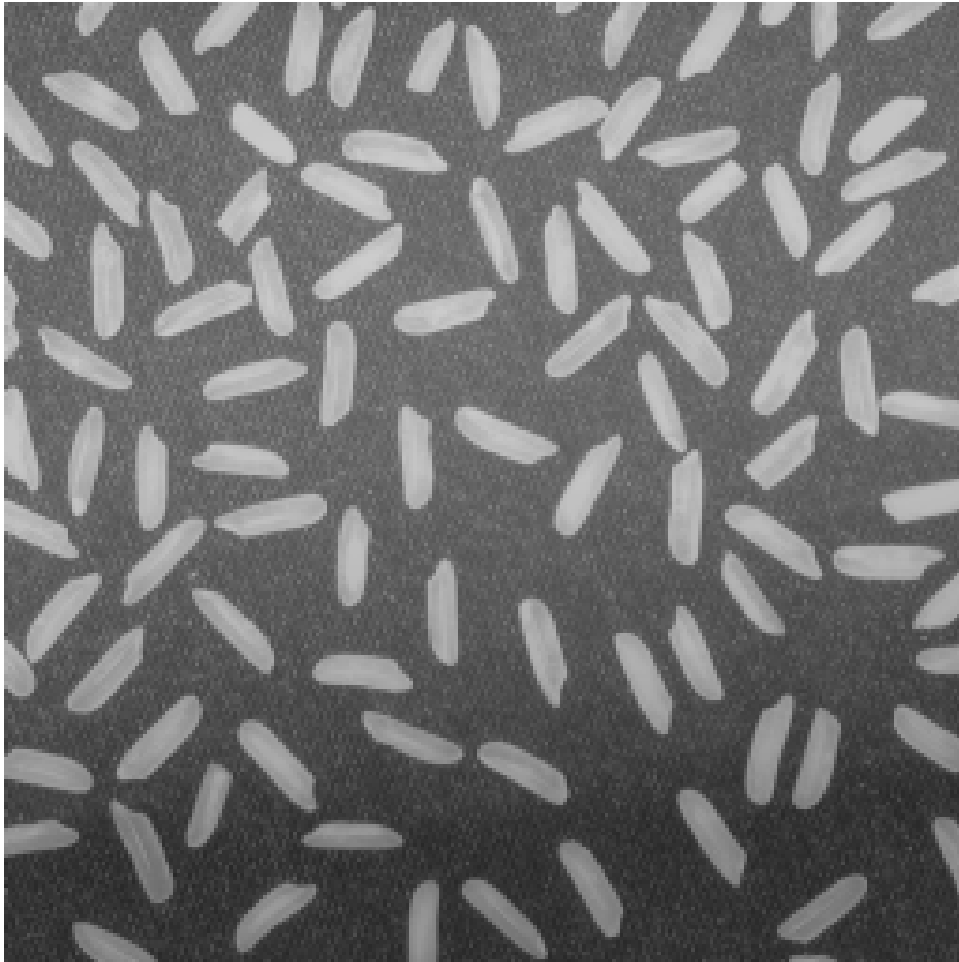


Figure 8.1: Exp8

## Experiment: 9

# Neighborhood Operations - To learn about neighborhood operations and use them for i) Linear filtering ii) Non-linear filtering

Scilab code Solution 9.9 Exp9

```
1 //Prog9.Neighborhood Operations – To learn about
   neighborhood operations and use them for
2 //i) Linear filtering ii) Non-linear filtering
3 //Software version
4 //OS Windows7
5 //Scilab5.4.1
6 //Image Processing Design Toolbox 8.3.1-1
7 //Scilab Image and Video Processing toolbox
   0.5.3.1-2
8 clc;
9 clear;
10 close;
11 I = imread('C:\Users\senthilkumar\Desktop\
```

```

    Gautam_PAL_Lab\DIP_Lab2\lenna.jpg');
12 I_noise = imnoise(I, 'salt & pepper');
13 figure
14 ShowImage(I, 'Original Lenna Image')
15 figure
16 ShowImage(I_noise, 'Noisy Lenna Image')
17 //Case 1: Linear Filtering
18 //(i).Linear Filtering –Example 1: Average Filter
19 F_linear1 = 1/25*ones(5,5); //5x5 mask
20 I_linear1 = imfilter(I_noise, F_linear1); //linear
    filtering –Average Filter
21 figure
22 ShowImage(I_linear1, 'Linear Average Filtered Noisy
    Lenna Image')
23 //(ii).Linear Filtering – Example 2: Gaussing
    filter
24 hsize = [5,5];
25 sigma = 1;
26 F_linear2 = fspecial('gaussian', hsize, sigma); //
    Linear filtering – gaussian Filter
27 I_linear2 = imfilter(I_noise, F_linear2);
28 figure
29 ShowImage(I_linear2, 'Linear Gaussian Filtered Noisy
    Lenna Image')
30 //Case 2: Non-Linear Filtering
31 //(i).Median Filtering
32 F_NonLinear = [3,3];
33 I_NonLinear = MedianFilter(I_noise, F_NonLinear); //
    Median Filter 3x3
34 figure
35 ShowImage(I_NonLinear, 'Median Filtered (Non-Linear)
    Noisy Lenna Image')

```

---

## Experiment: 10

# Neighborhood Operations –To study the effect of the size of neighborhood on the result of processing

Scilab code Solution 10.10 Exp10

```
1 //Prog10.Neighborhood Operations —To study the
   effect of the size of neighborhood on the result
   of processing
2 //Software version
3 //OS Windows7
4 //Scilab5.4.1
5 //Image Processing Design Toolbox 8.3.1–1
6 //Scilab Image and Video Processing toolbox
   0.5.3.1–2
7 clc;
8 clear;
9 close;
10 I = imread('C:\Users\senthilkumar\Desktop\
   Gautam_PAL_Lab\DIP_Lab2\lenna.jpg');
11 I_noise = imnoise(I, 'salt & pepper');
```

```
12 FilterSize = [3 3]; //filter size 3x3
13 I_3x3 = MedianFilter(I_noise,FilterSize);
14 I_5x5 = MedianFilter(I_noise,[5 5]);
15 I_7x7 = MedianFilter(I_noise,[7 7]);
16 I_9x9 = MedianFilter(I_noise,[9 9]);
17 figure
18 ShowImage(I,'Original Lenna Image')
19 figure
20 ShowImage(I_noise,'Original Lenna Image')
21 figure
22 ShowImage(I_3x3,'Filtered Lenna Image-Filter size 3
    x3')
23 figure
24 ShowImage(I_5x5,'Filtered Lenna Image-Filter size 5
    x5')
25 figure
26 ShowImage(I_7x7,'Filtered Lenna Image-Filter size 7
    x7')
27 figure
28 ShowImage(I_9x9,'Filtered Lenna Image-Filter size 9
    x9')
```

---

# Experiment: 11

## Image Histogram - To understand how frequency distribution can be used to represent an image.

Scilab code Solution 11.11 Exp11

```
1 //Prog11.Image Histogram – To understand how
   frequency distribution can be used to represent
   an image
2 //Software version
3 //OS Windows7
4 //Scilab5.4.1
5 //Image Processing Design Toolbox 8.3.1–1
6 //Scilab Image and Video Processing toolbox
   0.5.3.1–2
7 clc;
8 clear;
9 close;
10 I = imread('C:\Users\senthilkumar\Desktop\
   Gautam_PAL_Lab\DIP_Lab2\pout.png');
11 [count, cells]= imhist(I);
```

```
12 scf(0)
13 ShowImage(I, 'Original Image pout.png')
14 scf(1);
15 plot2d3('gnn', cells, count)
16 title('Histogram Plot of Original Image')
17 exec('C:\Users\senthilkumar\Desktop\Gautam_PAL_Lab\
    histeq.sci');
18 Iheq = histeq(I);
19 [count, cells]= imhist(Iheq);
20 scf(2)
21 ShowImage(Iheq, 'Histogram Equalized Image pout.png')
22 scf(3)
23 plot2d3('gnn', cells, count)
24 title('Histogram of Histogram Equalized Image')
```

---

check Appendix [AP 3](#) for dependency:

histeq.sci



Figure 11.1: Exp11

## Experiment: 12

**Image Histogram –To study the correlation between the visual quality of an image with its histogram.**

Scilab code Solution 12.12 Exp12

```
1 //Prog12.Image Histogram —To study the correlation
   between the visual quality of an image with its
   histogram.
2 //Software version
3 //OS Windows7
4 //Scilab5.4.1
5 //Image Processing Design Toolbox 8.3.1-1
6 //Scilab Image and Video Processing toolbox
   0.5.3.1-2
7 clc;
8 clear;
9 close;
10 I = imread('C:\Users\senthilkumar\Desktop\
   Gautam_PAL_Lab\DIP_Lab2\pout.png');
11 I = imresize(I,[256,256]);
```

```

12 [count, cells]= imhist(I);
13 exec('C:\Users\senthilkumar\Desktop\Gautam_PAL_Lab\
    DIP_Lab2\histeq.sci');
14 Iheq = histeq(I);
15 [count1, cells1]= imhist(Iheq);
16 Corr_Bet_Same_Images = corr2(I,Iheq);
17 disp(Corr_Bet_Same_Images, 'Correlation between
    original Image and Its Histogram equalized Image'
    )
18 J = imread('C:\Users\senthilkumar\Desktop\
    Gautam_PAL_Lab\cameraman.jpeg');
19 Corr_Bet_Diff_Images = corr2(Iheq,J);
20 disp(Corr_Bet_Diff_Images, 'Correlation between pout.
    png and cameraman.jpeg images')
21 x = xcorr(count, count); //correlation of histogram
    of the same
22 x1 = xcorr(count, count1); //correlation of histogram
    of original image and its histogram equalized
    image
23 scf(0)
24 plot2d3('ggn', 1:length(x), x, 5)
25 title('correlation between histograms of original
    image')
26 scf(1)
27 plot2d3('ggn', 1:length(x1), x1, 5)
28 title('correlation between histograms of original
    image and its histogram equalized image')
29 //RESULT
30 //Correlation between original Image and Its
    Histogram equalized Image
31 //
32 //      0.9784662
33 //
34 // Correlation between pout.png and cameraman.jpeg
    images
35 //
36 //      - 0.3204259
37 //

```

---

check Appendix [AP 3](#) for dependency:

`histeq.sci`

## Experiment: 13

# Fourier Transform: To understand some of the fundamental properties of the Fourier transform.

Scilab code Solution 13.13 Exp13

```
1 //Prog13.Fourier Transform: To understand some of
  the fundamental properties of the Fourier
  transform.
2 //Software version
3 //OS Windows7
4 //Scilab5.4.1
5 //Image Processing Design Toolbox 8.3.1-1
6 //Scilab Image and Video Processing toolbox
  0.5.3.1-2
7 clc;
8 clear;
9 close;
10 I = imread('C:\Users\senthilkumar\Desktop\
  Gautam_PAL_Lab\DIP_Lab2\lenna.jpg');
11 exec('C:\Users\senthilkumar\Desktop\Gautam_PAL_Lab\
```

```

    DIP_Lab2\fft2d.sci ');
12 exec('C:\Users\senthilkumar\Desktop\Gautam_PAL_Lab\
    DIP_Lab2\ifft2d.sci ');
13 //[1].2D-DFT and its Inverse 2D-DFT
14 I = double(I);
15 J = fft2d(I);
16 K = real(ifft2d(J));
17 figure
18 ShowImage(I,'Original Lenna Image')
19 figure
20 ShowImage(abs(J),'2D DFT (spectrum) of Lenna Image')
21 figure
22 ShowImage(K,'2d IDFT of Lenna Image')
23 //[2]. Two times fftshift results in original
    spectrum
24 L = fftshift(J);
25 M = fftshift(L);
26 figure
27 ShowImage(abs(L),'fftshited spectrum of Lenna Image'
    )
28 figure
29 ShowImage(abs(M),'two times fftshifted')

```

---

check Appendix [AP 1](#) for dependency:

fft2d.sci

check Appendix [AP 2](#) for dependency:

ifft2d.sci

## Experiment: 14

# Colour Image Processing: To learn colour images are handled and processed i) Models for representing colour ii) Methods of proces

Scilab code Solution 14.14 Exp14

```
1 //Prog14. Colour Image Processing: To learn colour
   images are handled and processed
2 //i)Models for representing colour ii) Methods of
   proces
3 //Software version
4 //OS Windows7
5 //Scilab5.4.1
6 //Image Processing Design Toolbox 8.3.1-1
7 //Scilab Image and Video Processing toolbox
   0.5.3.1-2
8 clc;
9 clear;
10 close;
```

```
11 RGB = imread('C:\Users\senthilkumar\Desktop\  
    Gautam_PAL_Lab\DIP_lab2\football.jpg');  
12 figure  
13 ShowColorImage(RGB, 'RGB Color Image')  
14 YIQ = rgb2ntsc(RGB);  
15 figure  
16 ShowColorImage(YIQ, 'NTSC image YIQ')  
17 RGB = ntsc2rgb(YIQ);  
18 YCC = rgb2ycbcr(RGB);  
19 figure  
20 ShowColorImage(YCC, 'equivalent HSV image YCbCr')  
21 RGB = ycbcr2rgb(YCC);  
22 HSV = rgb2hsv(RGB);  
23 figure  
24 ShowColorImage(HSV, 'equivalent HSV image')  
25 RGB = hsv2rgb(HSV);  
26 R = RGB(:,:,1);  
27 G = RGB(:,:,2);  
28 B = RGB(:,:,3);  
29 figure  
30 ShowImage(R, 'Red Matrix')  
31 figure  
32 ShowImage(G, 'Green Matrix')  
33 figure  
34 ShowImage(B, 'Blue Matrix')
```

---

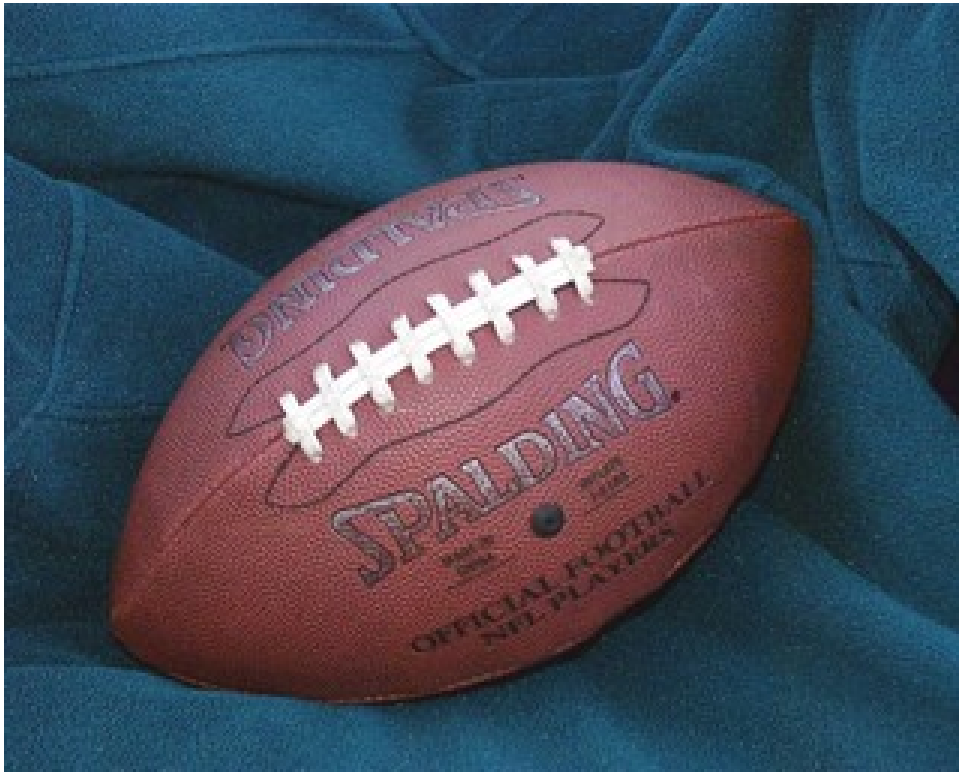


Figure 14.1: Exp14

## Experiment: 15

# Morphological Operations: To understand the basics of morphological operations which are used in analyzing the form and shape de

Scilab code Solution 15.15 Exp15

```
1 //Prog15.Morphological Operations: To understand the
   basics of morphological operations
2 //which are used in analyzing the form and shape
3 //Software version
4 //OS Windows7
5 //Scilab5.4.1
6 //Image Processing Design Toolbox 8.3.1-1
7 //Scilab Image and Video Processing toolbox
   0.5.3.1-2
8 clc;
9 clear;
10 close;
11 Image = imread('C:\Users\senthilkumar\Desktop\
```

```

    Gautam_PAL_Lab\DIP_Lab2\tire.jpeg ');
12 StructureElement = CreateStructureElement('square'
    ,3); // generate structuring element IPD atom
13 ResultImage1 = ErodeImage(Image,StructureElement);
    //IPD Atom
14 ResultImage2 = DilateImage(Image,StructureElement);
    //IPD Atom
15 ResultImage3 = BottomHat(Image,StructureElement);
    //IPD Atom
16 ResultImage4 = TopHat(Image,StructureElement); //
    IPD Atom
17 figure
18 ShowImage(Image,'Original Image')
19 figure
20 ShowImage(ResultImage1,'Eroded Image')
21 figure
22 ShowImage(ResultImage2,'Dilated Image')
23 figure
24 ShowImage(ResultImage3,'bottom hat filtered image')
25 figure
26 ShowImage(ResultImage4,'top hat filtered image')
27
28 ResultImage5 = imadd(ResultImage3,ResultImage4);
29 figure
30 ShowImage(ResultImage4,'top hat filtered image+
    bottom hat filtered image')

```

---



Figure 15.1: Exp15

# Appendix

```
Scilab code AP11 function [a2] = fft2d(a)
2 //a = any real or complex 2D matrix
3 //a2 = 2D-DFT of 2D matrix 'a'
4 m=size(a,1)
5 n=size(a,2)
6 // fourier transform along the rows
7 for i=1:n
8 a1(:,i)=exp(-2*i*pi*(0:m-1)'.*(0:m-1)/m)*a(:,i)
9 end
10 // fourier transform along the columns
11 for j=1:m
12 a2temp=exp(-2*i*pi*(0:n-1)'.*(0:n-1)/n)*(a1(j,:))
13 a2(j,:)=a2temp.'
14 end
15 for i = 1:m
16     for j = 1:n
17         if((abs(real(a2(i,j))))<0.0001)&(abs(imag(a2(
18             i,j))))<0.0001))
19             a2(i,j)=0;
20         elseif(abs(real(a2(i,j))))<0.0001)
21             a2(i,j)= 0+%i*imag(a2(i,j));
22         elseif(abs(imag(a2(i,j))))<0.0001)
23             a2(i,j)= real(a2(i,j))+0;
24         end
25     end
26 end
```

## 2D Fast Fourier Transform

---

```
Scilab code AP 12 function [a] =ifft2d(a2)
2 //a2 = 2D-DFT of any real or complex 2D matrix
3 //a = 2D-IDFT of a2
4 m=size(a2,1)
5 n=size(a2,2)
6 //Inverse Fourier transform along the rows
7 for i=1:n
8 a1(:,i)=exp(2*i*%pi*(0:m-1)'.*(0:m-1)/m)*a2(:,i)
9 end
10 //Inverse fourier transform along the columns
11 for j=1:m
12 atemp=exp(2*i*%pi*(0:n-1)'.*(0:n-1)/n)*(a1(j,:)).'
13 a(j,:)=atemp.'
14 end
15 a = a/(m*n)
16 a = real(a)
17 endfunction
```

## 2D Inverse Fast Fourier Transform

---

```
Scilab code AP 13 function [hea,b]=histeq(a)
2 //a- original image
3 //b- histogram
4 //hea- histogram equalized image
5 [m n]=size(a);
6 for i=1:256
7 b(i)=length(find(a==(i-1)));
8 end
9 pbb=b/(m*n);
10 pb(1)=pbb(1);
11 for i=2:256
12 pb(i)=pb(i-1)+pbb(i);
13 end
14
15 s=pb*255;
```

```

16     sb=uint8(round(s));
17     index =0;
18     for i=1:m
19         for j=1:n
20             index = double(a(i,j))+1;//convert it to
                double
21             //otherwise index = 255+1 =0
22             hea(i,j)= sb(index);//histogram
                equalization
23         end
24     end
25 endfunction

```

Histogram Equalization

---

**Scilab code AP 14** `function X = gray2bin(x)`

```

2     xmean = mean2(x);
3     [m,n]= size(x);
4     X = zeros(m,n);
5     for i = 1:m
6         for j = 1:n
7             if x(i,j)> xmean then
8                 X(i,j) = 1;
9             end
10        end
11    end
12 endfunction

```

Gray Pixel value to Binary value

---

**Scilab code AP 15** `function n = numdims(X)`

```

2     n = length(size(X));
3 endfunction

```

Total number of pixels in an image

---

**Scilab code AP 16** `function B = padarray(b)`

```
2     //pad zeros in columns and rows at both ends of
      an binary image
3 [m,n] = size(b);
4 num_dims = length(size(b));
5 B = zeros(m+num_dims,n+num_dims);
6 for i = num_dims:m+num_dims-1
7     for j = num_dims:m+num_dims-1
8         B(i,j) = b(i-1,j-1);
9     end
10 end
11 endfunction
```

Pad Array

---