Scilab Manual for
Digital Signal and Image Processing
by Ms chhaya
Computer Engineering
Xavier Institute Of Engineering[1]

Solutions provided by
Mr R Senthilkumar
Electronics Engineering
Institute of Road and Transport Technology

November 22, 2026

[1]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 3

1 To find Linear Convolution, Circular Convolution 5

2 To find output of Digital FIR filter using convolution principle 10

3 To find DFT/FFT forward and Inverse Transform of Image 15

4 To find DWT forward and Inverse Transform of Image 19

5 To enhance image using Histogram Equalization 22

6 To enhance image using Contrast Stretching 27

7 To find edges using LOG and DOG 32

8 To find Edges using Prewit/ Sobel/ Fri-chen / Robert operators 37

List of Experiments

Solution 1.1	Linear Convolution	5
Solution 1.2	Circular Convolution	7
Solution 2.1	To design and test the FIR Filter output	10
Solution 3.1	DFT and IDFT of 2D Image	15
Solution 4.1	DWT and IDWT of 2D Image	19
Solution 5.1	Image Enhancement using Histogram Equalization	22
Solution 6.1	Image Enhancement using Contrast Stretching	27
Solution 7.1	Edge Detection using LoG and DoG	32
Solution 8.1	Edge Detection using Prewit and Sobel	37
AP 1	Lena Gray Image	42
AP 2	Lena Image	43
AP 3	pout gray image	44
AP 4	pout gray image	45
AP 5	Cameraman Gray Image	46
AP 6	Lena Image	47

List of Figures

1.1 Linear Convolution 6

2.1 To design and test the FIR Filter output 13
2.2 To design and test the FIR Filter output 14

3.1 DFT and IDFT of 2D Image 16
3.2 DFT and IDFT of 2D Image 17

4.1 DWT and IDWT of 2D Image 21
4.2 DWT and IDWT of 2D Image 21

5.1 Image Enhancement using Histogram Equalization 23
5.2 Image Enhancement using Histogram Equalization 24

6.1 Image Enhancement using Contrast Stretching 28
6.2 Image Enhancement using Contrast Stretching 29

7.1 Edge Detection using LoG and DoG 33
7.2 Edge Detection using LoG and DoG 34

8.1 Edge Detection using Prewit and Sobel 38
8.2 Edge Detection using Prewit and Sobel 39

Experiment: 1

To find Linear Convolution, Circular Convolution

Scilab code Solution 1.1 Linear Convolution

```
1 //Caption: Program to find the Convolution Sum [
    Linear Convolution]
2 //of two discrete sequences
3 //Software version
4 //For: OS Windows7 and 8
5 //Scilab5.4.1 and above
6 //Image Processing Design Toolbox 8.3.1 and above
7 //Scilab Image and Video Proccessing toolbox
    0.5.3.1-2 and above
8 //For: OS Ubuntu Linux 14.04 and above
9 //Scilab5.5.1
10 //Image Processing Design Toolbox 8.3.1 and above
11 //Scilab Image and Video Proccessing toolbox
    0.5.3.1-2 and above
12 clear all;
13 close;
14 clc;
```

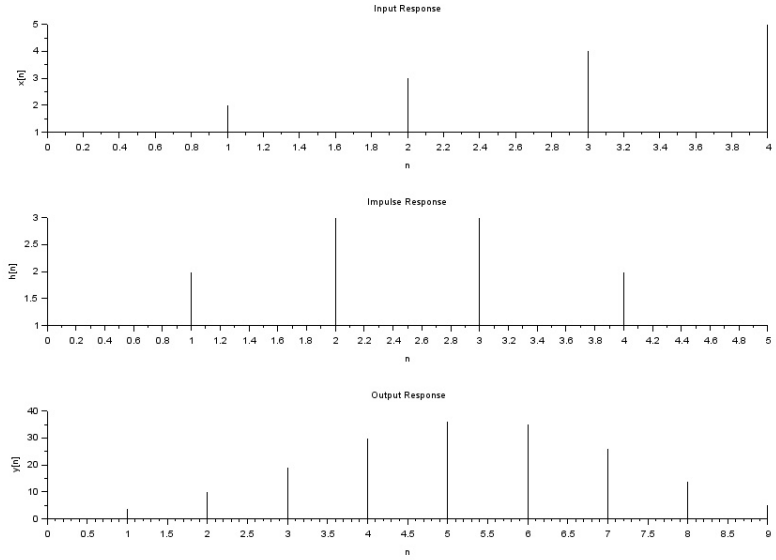


Figure 1.1: Linear Convolution

```
15 h = input('Enter the input sequence1=')
16 //h = [1,2,3,3,2,1];
17 x = input('Enter the input sequence2=')
18 //x = [1,2,3,4,5];
19 y = convol(x,h);
20 for i = 1:length(y)
21   if (y(i)<=0.0001)
22     y(i)=0;
23   end
24 end
25 disp(y,'Linear Convolution Output y[n]=')
26 subplot(3,1,2)
27 a=gca();
28 a.thickness = 2;
29 plot2d3('gnn',[0:length(h)-1],h)
30 xtitle('Impulse Response','n','h[n]');
31 subplot(3,1,1)
32 a=gca();
33 a.thickness = 2;
```

```
34 plot2d3('gnn',[0:length(x)-1],x)
35 xtitle('Input Response','n','x[n]');
36 subplot(3,1,3)
37 a=gca();
38 a.thickness = 2;
39 plot2d3('gnn',[0:length(y)-1],y)
40 xtitle('Output Response','n','y[n]');
41 //Result
42 //Enter the input sequence1=[1,2,3,3,2,1]
43 //Enter the input sequence2=[1,2,3,4,5]
44 //
45 // Linear Convolution Output y[n]=
46 //
47 //
48 //          column 1 to 8
49 //
50 //    1.    4.    10.    19.    30.    36.    35.
       26.
51 //
52 //          column  9 to 10
53 //
54 //    14.    5.
55 //
```

Scilab code Solution 1.2 Circular Convolution

```
1 //Caption: Program to find the Cicrcular Convolution
     of given discrete sequences
2 //Software version
3 //For: OS Windows7 and 8
4 //Scilab5.4.1 and above
5 //Image Processing Design Toolbox 8.3.1 and above
6 //Scilab Image and Video Proccessing toolbox
    0.5.3.1-2 and above
7 //For: OS Ubuntu Linux 14.04 and above
```

```
8  //Scilab5.5.1
9  //Image Processing Design Toolbox 8.3.1 and above
10 //Scilab Image and Video Proccessing toolbox
       0.5.3.1-2 and above
11 clear all;
12 clc;
13 x1 = input('Enter the first sequence x1[n]=')
14 x2 = input('Enter the second sequence x2[n]=')
15 //x1 = [1,3,5,7];
16 //x2 = [2,4,6,8];
17 m = length(x1)
18 n = length(x2)
19 a = zeros(1,max(m,n))
20 if (m >n)
21     for i = n+1:m
22       x2(i) = 0;
23     end
24 elseif (n>m)
25   for i = m+1:n
26     x1(i) = 0;
27   end
28 end
29 N = length(x1)
30 x3 = zeros(1,N);
31 a(1) = x2(1);
32 for j = 2:N
33   a(j) = x2(N-j+2);
34 end
35 for i =1:N
36   x3(1) = x3(1)+x1(i)*a(i);
37 end
38 for k = 2:N
39  for j =2:N
40    x2(j) = a(j-1);
41  end
42    x2(1) = a(N);
43    x2
44  for i = 1:N
```

```
45      a(i) = x2(i);
46      x3(k) = x3(k)+x1(i)*a(i);
47   end
48 end
49 disp(x3,'Circular Convolution Result x3 = ')
50 //RESULT
51 //Input Sequence x1 =
52 //
53 //     1.     3.     5.     7.
54 //
55 // Input Sequence x2 =
56 //
57 //     2.     4.     6.     8.
58 //
59 // Circular Convolution Result x3 =
60 //
61 //     84.     92.     84.     60.
```

Experiment: 2

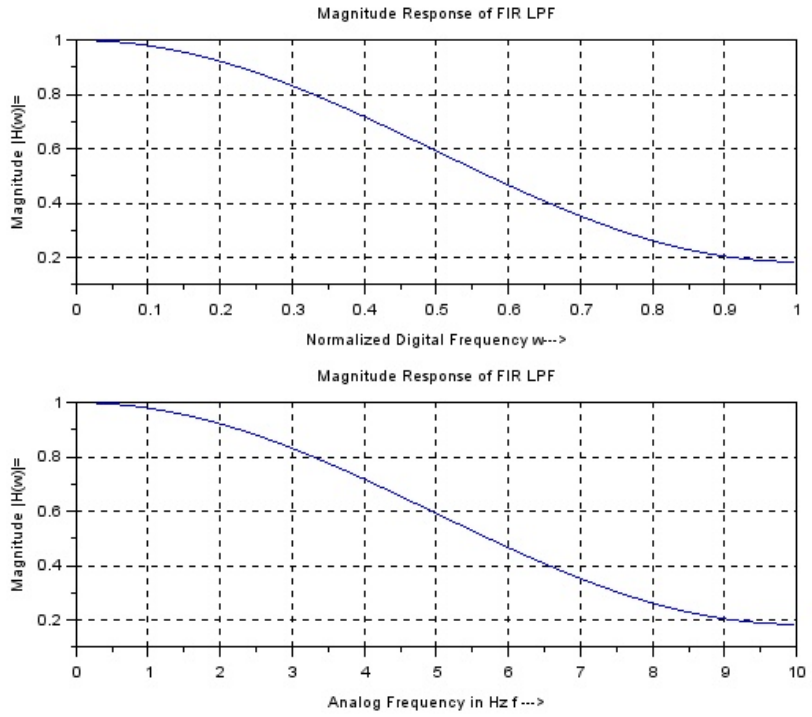
To find output of Digital FIR filter using convolution principle

Scilab code Solution 2.1 To design and test the FIR Filter output

```
1
2 //Caption: To Design an Low Pass FIR Filter and find
      out the output for sine wave
3 //input
4 //Filter Length =5, Order = 4
5 //Window = Hamming Window
6 //Software version
7 //For: OS Windows7 and 8
8 //Scilab5.4.1 and above
9 //Image Processing Design Toolbox 8.3.1 and above
10 //Scilab Image and Video Proccessing toolbox
     0.5.3.1-2 and above
11 //For: OS Ubuntu Linux 14.04 and above
12 //Scilab5.5.1
13 //Image Processing Design Toolbox 8.3.1 and above
14 //Scilab Image and Video Proccessing toolbox
     0.5.3.1-2 and above
```

```
clc;
clear;
xdel(winsid());
fc = input("Enter Analog cutoff freq. in Hz=")
fs = input("Enter Analog sampling freq. in Hz=")
M = input("Enter order of filter =")
w = (2*%pi)*(fc/fs);
disp(w,'Digital cutoff frequency in radians.cycles/
   samples');
wc = w/%pi;
disp(wc,'Normalized digital cutoff frequency in
   cycles/samples');
[wft,wfm,fr]=wfir('lp',M+1,[wc/2,0],'hm',[0,0]);
disp(wft,'Impulse Response of LPF FIR Filter:h[n]=')
   ;
//Plotting the Magnitude Response of LPF FIR Filter
subplot(2,1,1)
plot(2*fr,abs(wfm)/max(abs(wfm)))
xlabel('Normalized Digital Frequency w-->')
ylabel('Magnitude |H(w)|=')
title('Magnitude Response of FIR LPF')
xgrid(1)
subplot(2,1,2)
plot(fr*fs,abs(wfm)/max(abs(wfm)))
xlabel('Analog Frequency in Hz f -->')
ylabel('Magnitude |H(w)|=')
title('Magnitude Response of FIR LPF')
xgrid(1)
t = 0:0.01:1;
f = fc;//maximum frequency of sine wave
A = 5; //Amplitude of sine wave
X = A*sin(2*%pi*f*t);
N = rand(X)
N = N/max(N)
X_N = X+N;
Y = convol(X_N,wft)
figure
subplot(3,1,1)
```

```
plot(t*f,X)
title('FIR Filter input sine wave')
xgrid(1)
subplot(3,1,2)
plot(t,X_N)
title('Random Noise+Sine Wave')
xgrid(1)
subplot(3,1,3)
t = 0:1/(length(Y)-1):1
plot(t*f,Y)
title('FIR Filter output ')
xgrid(1)
//Example
//Note: Use Low cutoff freq inorderto
//clearly visible the output waveform
//Enter Analog cutoff freq. in Hz=5
//Enter Analog sampling freq. in Hz=20
//Enter order of filter =4
//Digital cutoff frequency in radians.cycles/samples
//    1.5707963
//Normalized digital cutoff frequency in cycles/
   samples        0.5
//Impulse Response of LPF FIR Filter:h[n]=
//column 1 to 4
// 1.559D-18    0.1718873    0.5    0.1718873
//column 5
// 1.559D-18
```


Magnitude Response of FIR LPF
Magnitude |H(w)|=
1
0.8
0.6
0.4
0.2
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Normalized Digital Frequency w--->
Magnitude Response of FIR LPF
Magnitude |H(w)|=
1
0.8
0.6
0.4
0.2
0
1
2
3
4
5
6
7
8
9
10
Analog Frequency in Hz f--->
Figure 2.1: To design and test the FIR Filter output

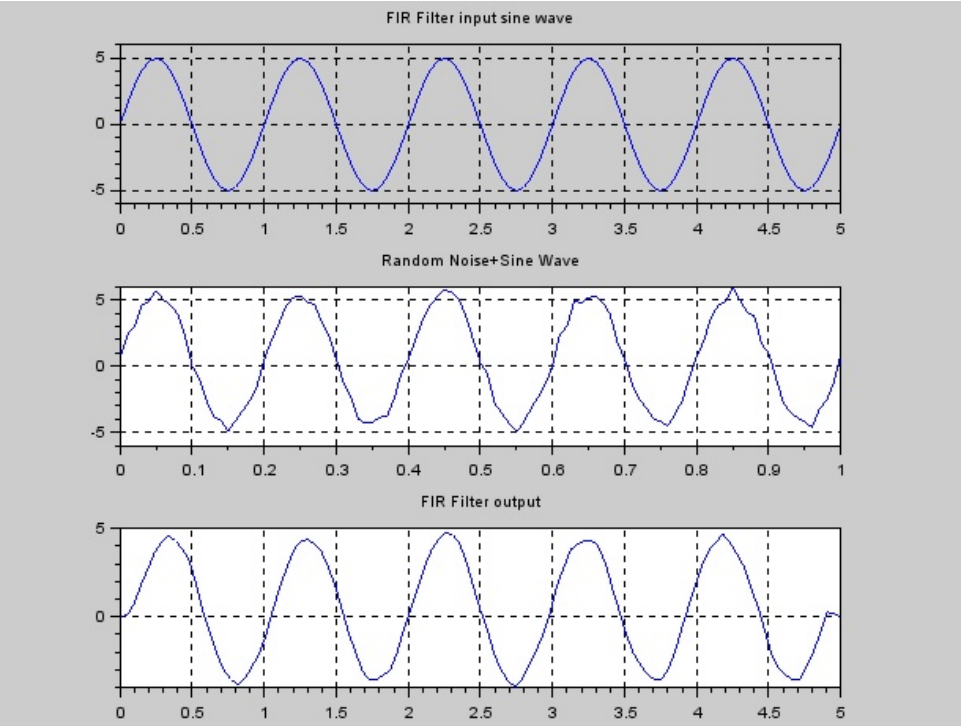


Figure 2.2: To design and test the FIR Filter output

Experiment: 3

To find DFT/FFT forward and Inverse Transform of Image

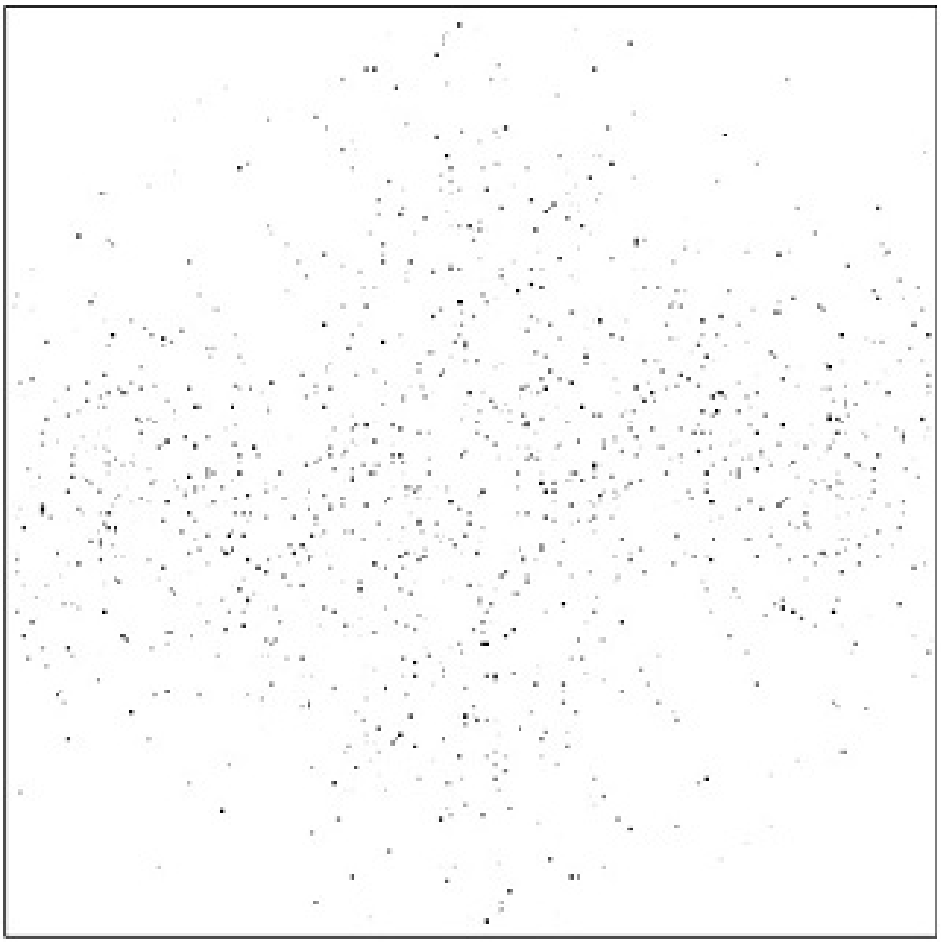
check Appendix AP 6 for dependency:

`Exp3lena.jpg`

Scilab code Solution 3.1 DFT and IDFT of 2D Image

```
1 //Caption: 2D-Discrete Fourier Transform using
    inbuilt fast fourier transform
2 //function fft2()
3 //Software version
4 //For: OS Windows7 and 8
5 //Scilab5.4.1 and above
6 //Image Processing Design Toolbox 8.3.1 and above
7 //Scilab Image and Video Proccessing toolbox
    0.5.3.1-2 and above
8 //For: OS Ubuntu Linux 14.04 and above
9 //Scilab5.5.1
```

2D-Discrete Fourier Transform of Lena Image



2D DFT-fftshifted

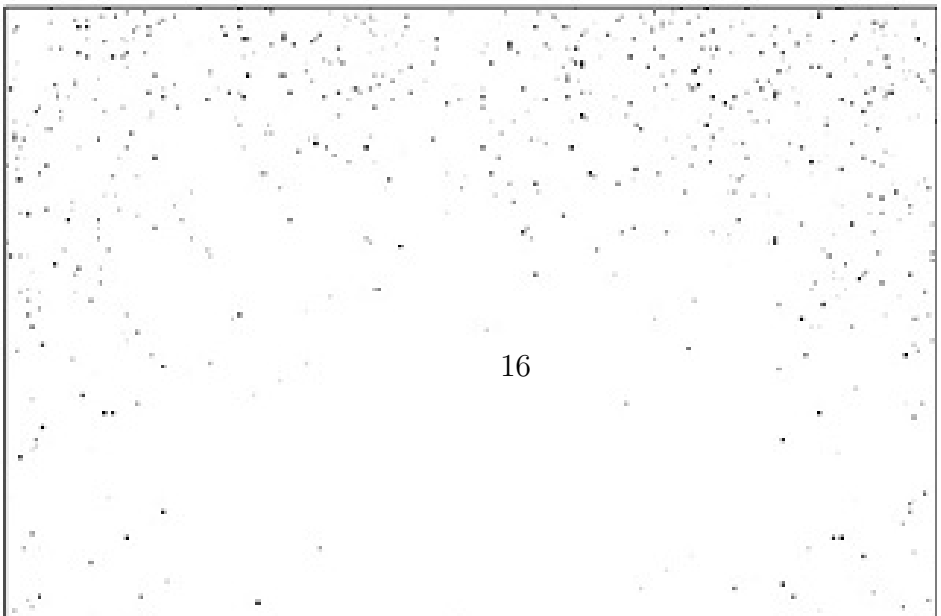




Figure 3.2: DFT and IDFT of 2D Image

```
10 //Image Processing Design Toolbox 8.3.1 and above
11 //Scilab Image and Video Proccessing toolbox
      0.5.3.1-2 and above
12 //To read and write in the correct folder
13 //change the current directory in scilab console
14 clc;
15 clear;
16 close;
17 a = ReadImage('Exp3lena.jpg');
18 [m,n]= size(a);
19 ShowImage(a,'Original lenna Image');
20 //2D-DFT using FFT
21 A = fft2(double(a));
22 figure(1)
23 ShowImage(abs(A),'2D-Discrete Fourier Transform of
      Lena Image');
24 //fftshifted image
25 B = fftshift(A)
26 figure(2)
```

```
27 ShowImage(abs(B),'2D-Discrete Fourier Transform of
      Lena Image-fftshifted');
28 //2D-IDFT using FFT
29 a_inv = fft2(A')
30 a_inv = a_inv'/(m*n);
31 figure(3)
32 ShowImage(uint8(abs(a_inv)),'2D-Inverse Discrete
      Fourier Transform ');
```

Experiment: 4

To find DWT forward and Inverse Transform of Image

check Appendix AP 5 for dependency:

`Exp4cameraman.jpg`

Scilab code Solution 4.1 DWT and IDWT of 2D Image

```
1 //Caption: Scilab code to implement Discrete Wavelet
    Transform
2 //Software version
3 //For: OS Windows7 and 8
4 //Scilab5.4.1 and above
5 //Image Processing Design Toolbox 8.3.1 and above
6 //Scilab Image and Video Proccessing toolbox
    0.5.3.1-2 and above
7 //For: OS Ubuntu Linux 14.04 and above
8 //Scilab5.5.1
9 //Image Processing Design Toolbox 8.3.1 and above
10 //Scilab Image and Video Proccessing toolbox
    0.5.3.1-2 and above
11 //To read and write in the correct folder
12 //change the current directory in scilab console
```

```
clc;
clear;
close;
//Original Image
img = imread('Exp4cameraman.jpg');
figure(1)
ShowImage(img,'Original Image')
title('Original Image');
[p q] = size(img);
[CA,CH,CV,CD] = dwt2(double(img),'db1');//Compute 2D
    wavelet transform -daubcheis wavelet
figure(2)
ShowImage(uint8(CA),'LPF-LPF output')
title('LPF-LPF output')
figure(3)
ShowImage(uint8(CH),'LPF-HPF output')
title('LPF-HPF output')
figure(4)
ShowImage(uint8(CV),'HPF-LPF output')
title('HPF-LPF output')
figure(5)
ShowImage(uint8(CD),'HPF-HPF output')
title('HPF-HPF output')
img_inv = idwt2(CA,CH,CV,CD,'db1');
img_inv = uint8(img_inv);
figure(6)
ShowImage(img_inv,'Reconstructed Image')
title('Reconstructed Image')
```

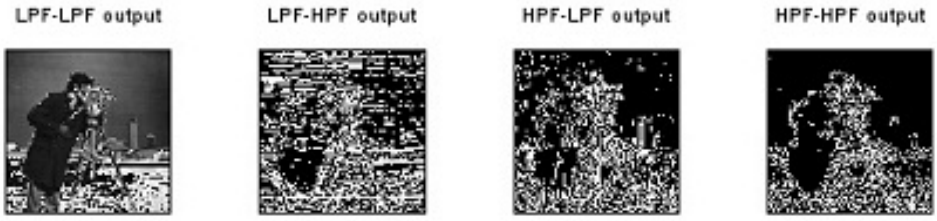


Figure 4.1: DWT and IDWT of 2D Image



Figure 4.2: DWT and IDWT of 2D Image

Experiment: 5

To enhance image using Histogram Equalization

check Appendix AP 4 for dependency:

`Exp5pout.jpg`

Scilab code Solution 5.1 Image Enhancement using Histogram Equalization

```
1 //Program Title: Image Enhancement - Image Histogram
      Equalisation
2 //For Gray Image
3 //Software version
4 //For: OS Windows7 and 8
5 //Scilab5.4.1 and above
6 //Image Processing Design Toolbox 8.3.1 and above
7 //Scilab Image and Video Proccessing toolbox
     0.5.3.1-2 and above
8 //For: OS Ubuntu Linux 14.04 and above
```

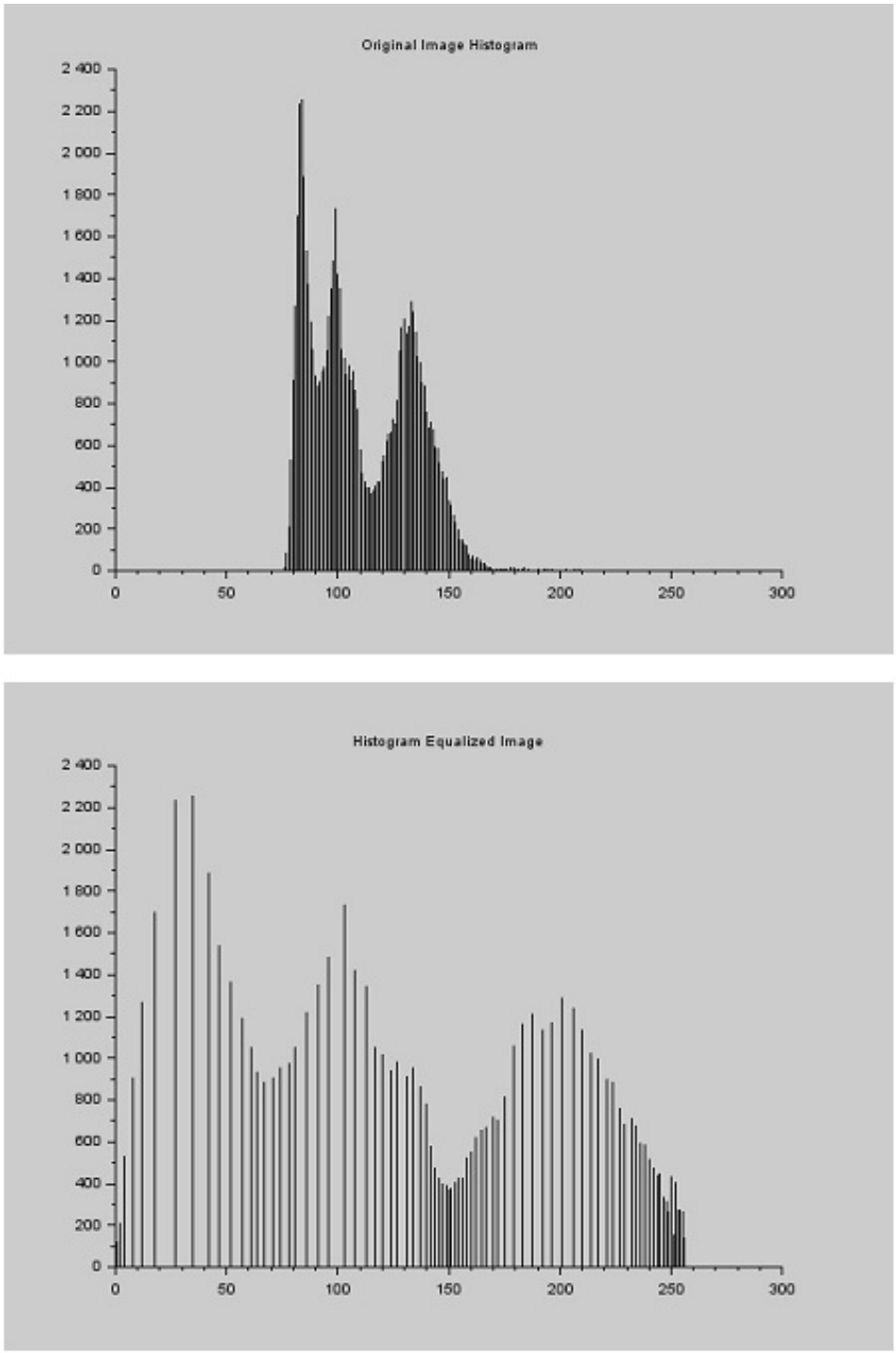


Figure 5.1: Image Enhancement using Histogram Equalization



Figure 5.2: Image Enhancement using Histogram Equalization

```
9  //Scilab5.5.1
10 //Image Processing Design Toolbox 8.3.1 and above
11 //Scilab Image and Video Proccessing toolbox
      0.5.3.1-2 and above
12 //To read and write in the correct folder
13 //change the current directory in scilab console
14
15 clc;
16 clear all;
17
18 f=ReadImage('Exp5pout.jpg');// Get image (Keep image
      file in same folder)
19 OrigSize=size(f);//Get size of original image
20 OrigRow=OrigSize(1);//Get number of rows in original
      image
21 OrigCol=OrigSize(2);//Get number of colomns in
      original image
22
23 figure
24 ShowImage(f,'Original Image Pout')
25 WriteImage(f,'Exp6OriginalImagePout.jpg') //Comment
      out if you do not want o/p file to be written on
      disk
26
27 OrigHist=CreateHistogram(uint8(f));//Create
      Histogram
28 figure
29 plot2d3(OrigHist)//Show Histogram
30 title('Original Image Histogram')
31
32 P=OrigHist/(OrigRow*OrigCol);//get average pixel
      value
33
34 C=zeros(1,256);
35
36 for i=2:256
37     C(1,i)=C(1,i-1)+P(1,i);
38 end
```

```
39
40 Cdash = round(C*255);
41
42 EqImage=[]//Empty matrix for output image
43 for i=1:OrigRow
44     for j=1:OrigCol
45         EqImage(i,j)=Cdash(1,f(i,j)+1);
46     end
47 end
48
49 figure
50 ShowImage(uint8(EqImage),'Image Histogram
      Equalisation')
51 WriteImage(uint8(EqImage),'
      Exp6ImageHistogramEqualisation.jpg') //Comment
      out if you do not want o/p file to be written on
      disk
52
53 EqHist=CreateHistogram(uint8(EqImage));//Create
      Histogram
54 figure
55 plot2d3(EqHist)//Show Histogram
56 title('Histogram Equalized Image')
```

Experiment: 6

To enhance image using Contrast Stretching

check Appendix AP 3 for dependency:

`Exp6Pout.jpg`

Scilab code Solution 6.1 Image Enhancement using Contrast Stretching

```
1 //Program Title: Image Enhancement - Image Contrast
    Stretching
2 //For Gray Image
3 //Sample Input:-
4 //Set value r1 (0<r1<255):75
5 //Set value r2 (r1<r2<255:175
6 //Set value s1 (0<s1<255):10
7 //Set value s2 (s1<s2<255:245
8 //Software version
9 //For: OS Windows7 and 8
10 //Scilab5.4.1 and above
```

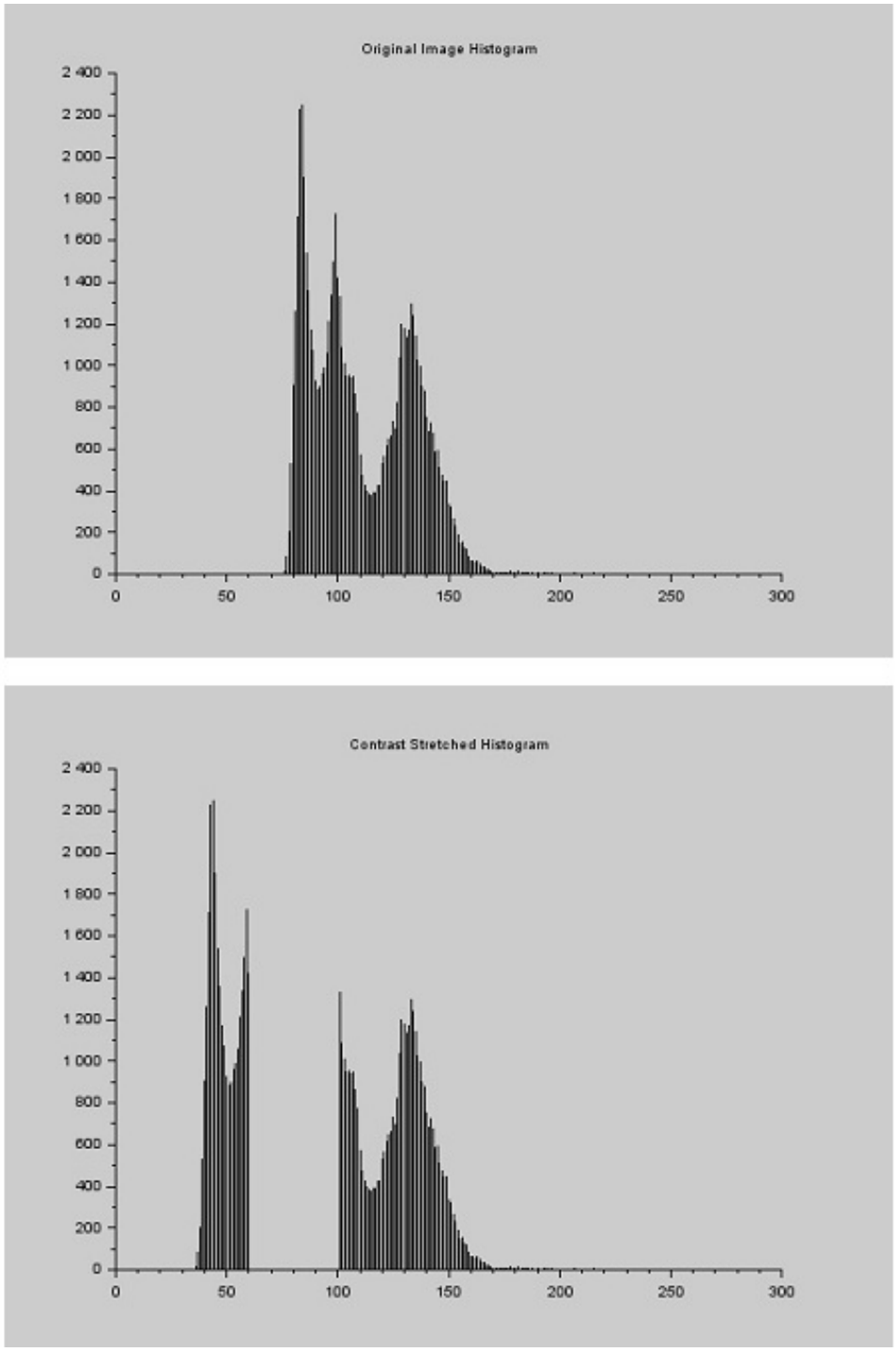


Figure 6.1: Image Enhancement using Contrast Stretching



Figure 6.2: Image Enhancement using Contrast Stretching

```
11 //Image Processing Design Toolbox 8.3.1 and above
12 //Scilab Image and Video Proccessing toolbox
      0.5.3.1-2 and above
13 //For: OS Ubuntu Linux 14.04 and above
14 //Scilab5.5.1
15 //Image Processing Design Toolbox 8.3.1 and above
16 //Scilab Image and Video Proccessing toolbox
      0.5.3.1-2 and above
17 //To read and write in the correct folder
18 //change the current directory in scilab console
19
20 clc;
21 clear all;
22
23 f=ReadImage('Exp6pout.jpg');// Get image as 16 bit
      unsigned integer
24 OrigSize=size(f);//Get size of original image
25 OrigRow=OrigSize(1);//Get number of rows in original
       image
26 OrigCol=OrigSize(2);//Get number of colomns in
      original image
27
28 figure
29 ShowImage(f,'Original Image Pout')
30 WriteImage(f,'OriginalImagePout.jpg') //Comment out
      if you do not want o/p file to be written on disk
31
32 OrigHist=CreateHistogram(uint8(f));//Create
      Histogram
33 figure
34 plot2d3(OrigHist)//Show Histogram
35 title('Original Image Histogram')
36
37 //Image Contrast Stretching
38
39 disp('Image Contrast Stretching')
40 r1=input('Set value r1 (0<r1<255):');
41 r2=input('Set value r2 (r1<r2<255:');
```

```
42 s1=input('Set value s1 (0<s1<255):');
43 s2=input('Set value s2 (s1<s2<255:');
44
45 Alpha=s1/r1;
46 Beta=(s2-s1)/(r2-r1);
47 Gamma=(255-s2)/(255-r2);
48
49 ContStr=[];//Empty matrix for output image
50
51 for x=1:OrigRow
52     for y=1:OrigCol
53         if f(x,y)<r1 then;
54             ContStr(x,y)=Alpha*f(x,y);
55         elseif (f(x,y)>r1)&(f(x,y)<r2);
56             ContStr(x,y)=Beta*(f(x,y)-r1)+s1;
57         else f(x,y)>r2;
58             ContStr(x,y)=Gamma*(f(x,y)-r2)+s2;
59         end
60     end
61 end
62
63 figure
64 ShowImage(uint8(ContStr),'Exp5 Image Contrast
     Stretching')
65 WriteImage(uint8(ContStr),'
     Exp5ImageContrastStretching.jpg')
66 ContStrHist=CreateHistogram(uint8(ContStr));//Create
      Histogram
67 figure
68 plot2d3(ContStrHist)//Show Histogram
69 title('Contrast Stretched Histogram')
70 //Result
71 //Image Contrast Stretching
72 //Set value r1 (0<r1<255):50
73 //Set value r2 (r1<r2<255: 100
74 //Set value s1 (0<s1<255):10
75 //Set value s2 (s1<s2<255: 100
```

Experiment: 7

To find edges using LOG and DOG

check Appendix AP 2 for dependency:

`Exp7lena.jpg`

Scilab code Solution 7.1 Edge Detection using LoG and DoG

```
1 //Caption: Scilab code for Edge Detection using
     Different Edge detectors
2 //[1]LoG  [2] DoG
3 //Software version
4 //For: OS Windows7 and 8
5 //Scilab5.4.1 and above
6 //Image Processing Design Toolbox 8.3.1 and above
7 //Scilab Image and Video Proccessing toolbox
     0.5.3.1-2 and above
8 //For: OS Ubuntu Linux 14.04 and above
9 //Scilab5.5.1
```



Figure 7.1: Edge Detection using LoG and DoG



Figure 7.2: Edge Detection using LoG and DoG

```
10 //Image Processing Design Toolbox 8.3.1 and above
11 //Scilab Image and Video Proccessing toolbox
     0.5.3.1-2 and above
12 //To read and write in the correct folder
13 //change the current directory in scilab console
14 close;
15 a = imread('Exp7lena.jpg');
16 //LAPLACIAN OF GAUSSIAN
     ////////////////////////////////////////////
17 [c,thresh] = edge(a,'log',0.2,'both',2);
18 //d = edge(a,'prewitt');
19 ShowImage(a,'Original Image')
20 title('Original Image')
21 figure
22 ShowImage(c,'Edge Detected Image-LoG')
23 //
     ////////////////////////////////////////////////////////////

24 //DERIVATIVE OF GAUSSIAN FUNCTION
25 function [mask]=DoG(op1,op2)
26     //If op1 and op2 are not specified -Default size
          3x3
27     if isempty(op1) then
28        siz = [3,3];
29     else
30        if length(op1)==1 then
31            siz = [op1, op1];
32        elseif length(op1)==2 then
33            siz = op1;
34        else
35            error("The second argument should have 1
                 or 2 elements for gaussian filter");
36        end
37     end
38      //set std for the filter
39     if isempty(op2) then
40        g_std = 0.5;
41     else
```

```
42          if length(op2)>1 then
43              error("The third argument should have
                    only 1 element for gaussian filter");
44          else
45              g_std = op2;
46          end
47      end
48       sizx = (siz(2)-1)/2;
49       sizy = (siz(1)-1)/2;
50       x2 = ones(siz(1),1) * ([-sizx:sizx]^2);
51       y2 = ([-sizy:sizy]^2)' * ones(1, siz(2));
52       r = sqrt(x2+y2);
53       sigma = g_std;
54       F = (1/(sigma^2))*(((r.*r)/sigma^2)-1).*exp(-r
            .*r/2*sigma^2);
55       F(F<%eps*max(F)) = 0;
56       sumF=sum(F);
57       if sumF~=0 then
58         F = F / sum(F);
59       end
60       mask = F;
61  endfunction
62  mask =  DoG([],[]);
63  mx=filter2(mask,a);
64  my=filter2(mask',a);
65  border=sqrt(mx.*mx + my.*my);
66  if thresh >=0 then
67     scale_thresh = min(border) * (1-thresh) + max(
          border)*thresh;
68     border=border > scale_thresh;
69  end
70  d = border;
71  //
       ////////////////////////////////////////////////////////////////

72  figure
73  ShowImage(d,'Edge Detected Image-DoG')
```

Experiment: 8

To find Edges using Prewit/ Sobel/ Fri-chen / Robert operators

check Appendix AP 1 for dependency:

`Exp8lena.jpg`

Scilab code Solution 8.1 Edge Detection using Prewit and Sobel

```
1 //Caption: Scilab code for Edge Detection using
    Different Edge detectors
2 //[1]. Sobel [2].Prewitt
3 //Roberts and Fr-Chen edge detection operators are
    not available in this atom
4 //Software version
5 //For: OS Windows7 and 8
6 //Scilab5.4.1 and above
7 //Image Processing Design Toolbox 8.3.1 and above
```



Figure 8.1: Edge Detection using Prewit and Sobel

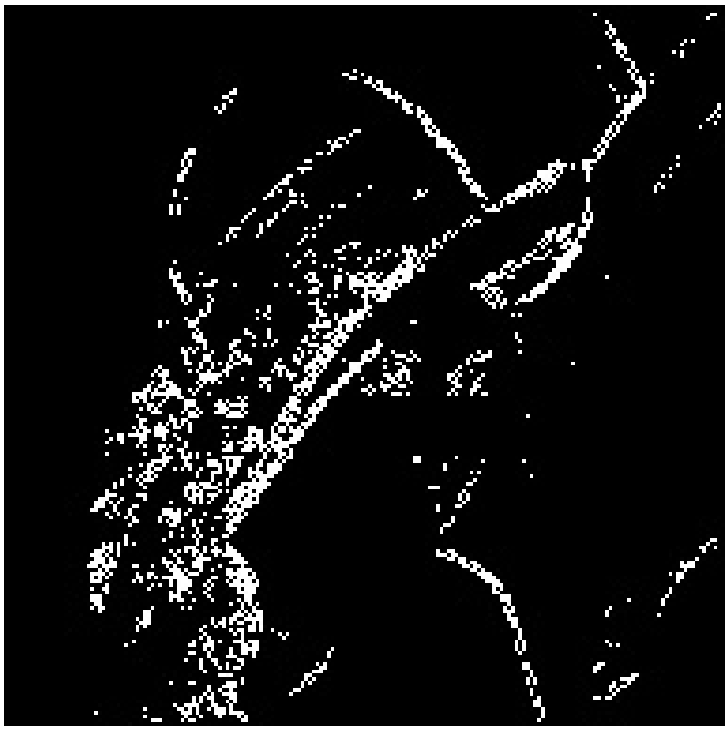


Figure 8.2: Edge Detection using Prewit and Sobel

```
8  //Scilab Image and Video Proccessing toolbox
       0.5.3.1-2 and above
9  //For: OS Ubuntu Linux 14.04 and above
10 //Scilab5.5.1
11 //Image Processing Design Toolbox 8.3.1 and above
12 //Scilab Image and Video Proccessing toolbox
       0.5.3.1-2 and above
13 ////To read and write in the correct folder
14 //change the current directory in scilab console
15 close;
16 a = imread('Exp8lena.jpg');
17 c = edge(a,'sobel');
18 d = edge(a,'prewitt');
19 ShowImage(a,'Original Image')
20 title('Original Image')
21 figure
22 ShowImage(c,'Edge Detected Image-Sobel')
23 figure
24 ShowImage(d,'Edge Detected Image-Prewitt')
```

Appendix


Lena Gray Image


Lena Image



pout
gray image



pout gray image



Cameraman Gray Image


Lena Image