

Scilab Manual for  
Communication and Signal Processing  
by Dr A. Rajeswari  
Electronics Engineering  
Coimbatore Institute of Tehnology<sup>1</sup>

Solutions provided by  
Prof Pinkesh Patel  
Electronics Engineering  
Dharmsinh Desai University, Nadiad-387001

June 14, 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 Analog Modulation and Demodulation Schemes	8
2 Digital Modulation and Demodulation Schemes	13
3 Performance analysis of Digital Communication System	17
4 Generation of Waveforms	20
5 Properties of Signals and Systems	26
6 Convolution	33
7 Correlation	36
8 Finding DFT and IDFT using FFT	39
9 Design of FIR filters	42
10 Design of IIR filters	45
11 Error control Codes	48
12 Basic Image Processing Operations	53
13 2D Convolution	64
14 Image Transforms -DFT,DCT and DWT	70



# List of Experiments

Solution 1.1	Perform the following Analog Modulation Scheme on Binary Data . . . . .	8
Solution 2.1	Perform the following Digital Modulation Scheme on Binary Data . . . . .	13
Solution 3.1	Measure the Performance of Communication System in term of Bit Error Rate . . . . .	17
Solution 4.1	To generate basic discrete signal used in Digital Signal Processing . . . . .	20
Solution 5.1	Property of Signal and System . . . . .	26
Solution 6.1	Perform Linear and Circular Convolution Operation on Two Discrete Sequences . . . . .	33
Solution 7.1	Perform Correlation Operation on Two Discrete Sequences . . . . .	36
Solution 8.1	Perform DFT and IDFT of discrete signal . . . . .	39
Solution 9.1	Design the following FIR filters with the given specification . . . . .	42
Solution 10.1	Design the following IIR filters with the given specification . . . . .	45
Solution 11.1	Perform the Linear Block Coding on binary Data . . . . .	48
Solution 12.1	Perform the following basic image processing operation on digital image . . . . .	53
Solution 13.1	Perform the 2D Convolution on digital image . . . . .	64
Solution 14.1	Perform the Following Transform on Gray Scale Image . . . . .	70
Solution 15.1	Perform the Various Edge Detection Methods on Gray Scale Image . . . . .	78
AP 1	Perform the Various Edge Detection Methods on Gray Scale Image . . . . .	88

# List of Figures

1.1	Perform the following Analog Modulation Scheme on Binary Data . . . . .	11
1.2	Perform the following Analog Modulation Scheme on Binary Data . . . . .	12
1.3	Perform the following Analog Modulation Scheme on Binary Data . . . . .	12
2.1	Perform the following Digital Modulation Scheme on Binary Data . . . . .	16
3.1	Measure the Performance of Communication System in term of Bit Error Rate . . . . .	19
4.1	To generate basic discrete signal used in Digital Signal Processing . . . . .	24
4.2	To generate basic discrete signal used in Digital Signal Processing . . . . .	25
4.3	To generate basic discrete signal used in Digital Signal Processing . . . . .	25
5.1	Property of Signal and System . . . . .	30
5.2	Property of Signal and System . . . . .	30
5.3	Property of Signal and System . . . . .	31
5.4	Property of Signal and System . . . . .	32
5.5	Property of Signal and System . . . . .	32
6.1	Perform Linear and Circular Convolution Operation on Two Discrete Sequences . . . . .	35
7.1	Perform Correlation Operation on Two Discrete Sequences . . . . .	38

8.1	Perform DFT and IDFT of discrete signal . . . . .	41
9.1	Design the following FIR filters with the given specification .	44
10.1	Design the following IIR filters with the given specification .	47
12.1	Perform the following basic image processing operation on digital image . . . . .	57
12.2	Perform the following basic image processing operation on digital image . . . . .	58
12.3	Perform the following basic image processing operation on digital image . . . . .	59
12.4	Perform the following basic image processing operation on digital image . . . . .	60
12.5	Perform the following basic image processing operation on digital image . . . . .	61
12.6	Perform the following basic image processing operation on digital image . . . . .	62
12.7	Perform the following basic image processing operation on digital image . . . . .	63
13.1	Perform the 2D Convolution on digital image . . . . .	66
13.2	Perform the 2D Convolution on digital image . . . . .	67
13.3	Perform the 2D Convolution on digital image . . . . .	68
13.4	Perform the 2D Convolution on digital image . . . . .	69
14.1	Perform the Following Transform on Gray Scale Image . . .	73
14.2	Perform the Following Transform on Gray Scale Image . . .	74
14.3	Perform the Following Transform on Gray Scale Image . . .	75
14.4	Perform the Following Transform on Gray Scale Image . . .	76
14.5	Perform the Following Transform on Gray Scale Image . . .	76
14.6	Perform the Following Transform on Gray Scale Image . . .	77
15.1	Perform the Various Edge Detection Methods on Gray Scale Image . . . . .	81
15.2	Perform the Various Edge Detection Methods on Gray Scale Image . . . . .	82
15.3	Perform the Various Edge Detection Methods on Gray Scale Image . . . . .	83

15.4 Perform the Various Edge Detection Methods on Gray Scale Image . . . . .	84
15.5 Perform the Various Edge Detection Methods on Gray Scale Image . . . . .	85
15.6 Perform the Various Edge Detection Methods on Gray Scale Image . . . . .	86

# Experiment: 1

## Analog Modulation and Demodulation Schemes

**Scilab code Solution 1.1** Perform the following Analog Modulation Scheme on Binary Data

```
1 // LAB:1 – Perform the following Analog Modulation
  Scheme on Binary Data
2 //           (I) Amplitude Modulation (II) Frequency
  Modulation (III) Phase Modulation
3 // Version : Scilab 5.4.1
4 // Operating System : Window-xp, Window-7
5
6 clc;
7 clear;
8 xdel(winsid());
9 f_signal=1; // Signal Frequencies
10 f_carrier=8; // Carrier Frequencies
11
12 t=0:0.001:5;
13
14 ////////////////////////////////// Amplitude Modulation
  //////////////////////////////////
15 information_signal=2*sin(2*%pi*f_signal*t); //
```

```

    Information Signal
16 carrier_signal=15*sin(2*pi*f_carrier*t); // Carrier
    Signal
17 Modulated_signal=(5+information_signal(:,:)).*sin(2*
    %pi*f_carrier*t);
18 subplot(3,1,1);plot(information_signal,'LineWidth'
    ,1.5); //plot of Information Signal
19 xgrid;
20 title('Information Signal','color','Red','fontsize'
    ,3); //title the Graph
21 subplot(3,1,2);plot(carrier_signal,'LineWidth',1.5);
    //plot of Carrier signal
22 xgrid;
23 title('Carrier Signal','color','Red','fontsize',3);
    //title of plot
24 subplot(3,1,3);plot(Modulated_signal,'LineWidth'
    ,1.5); //plot of modulated signal
25 xgrid;
26 title('Amplitude Modulation Signal','color','Red','
    fontsize',3); //title of plot
27
28 //////////////// Frequency Modulation
    ////////////////
29 Deviation_facotr=5.5
30 information_signal=2*sin(2*pi*f_signal*t); //
    Information Signal
31 carrier_signal=15*sin(2*pi*f_carrier*t); // Carrier
    Signal
32 Modulated_signal=15.*cos((2*pi*f_carrier*t)+(
    Deviation_facotr.*sin(2*pi*f_signal*t)));
33 figure;
34 subplot(3,1,1);plot(information_signal,'LineWidth'
    ,1.5); //plot of Information Signal
35 xgrid;
36 title('Information Signal','color','Red','fontsize'
    ,3); //title the Graph
37 subplot(3,1,2);plot(carrier_signal,'LineWidth',1.5);
    //plot of Carrier signal

```

```

38 xgrid;
39 title('Carrier Signal','color','Red','fontsize',3);
    //title of plot
40 subplot(3,1,3);plot(Modulated_signal,'LineWidth'
    ,1.5);//plot of modulated signal
41 xgrid;
42 title('Frequency Modulation Signal','color','Red','
    fontsize',3);//title of plot
43
44
45 //////////////// Phase Modulation
    ////////////////
46 Phase_deviation_facotr=35
47 information_signal=2*sin(2*pi*f_signal*t); //
    Information Signal
48 carrier_signal=15*sin(2*pi*f_carrier*t);// Carrier
    Signal
49 Modulated_signal=15.*cos((2*pi*f_carrier*t)+(
    Phase_deviation_facotr.*sin(2*pi*f_signal*t)));
50 figure;
51 subplot(3,1,1);plot(information_signal,'LineWidth'
    ,1.5);//plot of Information Signal
52 xgrid;
53 title('Information Signal','color','Red','fontsize'
    ,3);//title the Graph
54 subplot(3,1,2);plot(carrier_signal,'LineWidth',1.5);
    //plot of Carrier signal
55 xgrid;
56 title('Carrier Signal','color','Red','fontsize',3);
    //title of plot
57 subplot(3,1,3);plot(Modulated_signal,'LineWidth'
    ,1.5);//plot of modulated signal
58 xgrid;
59 title('Phase Modulation Signal','color','Red','
    fontsize',3);//title of plot

```

---

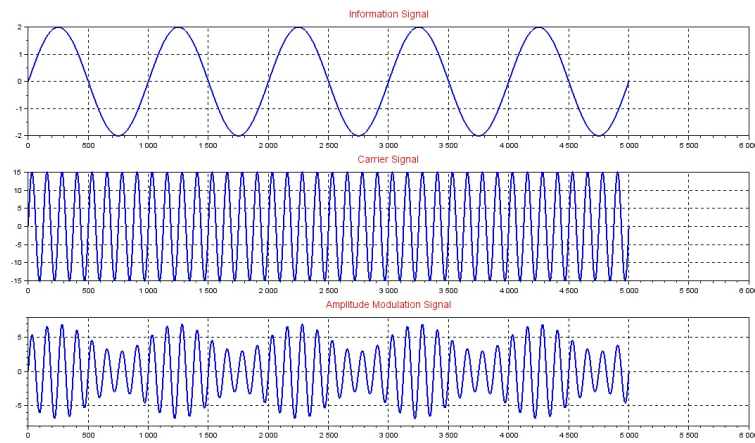


Figure 1.1: Perform the following Analog Modulation Scheme on Binary Data

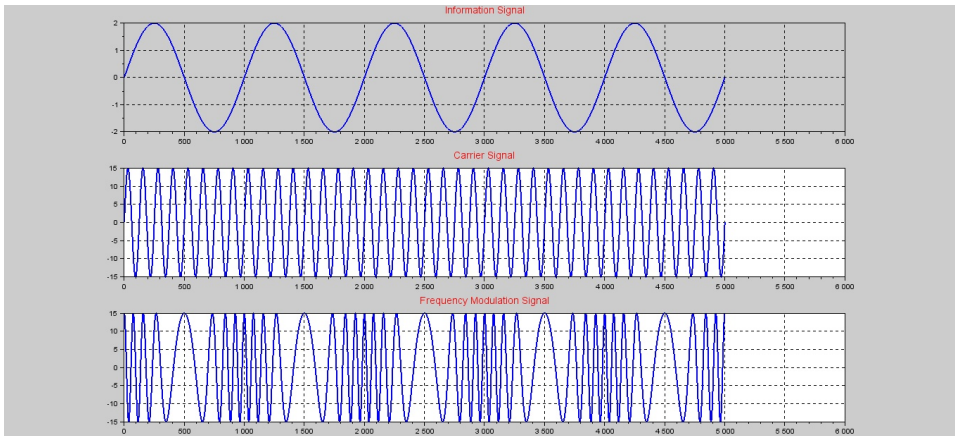


Figure 1.2: Perform the following Analog Modulation Scheme on Binary Data

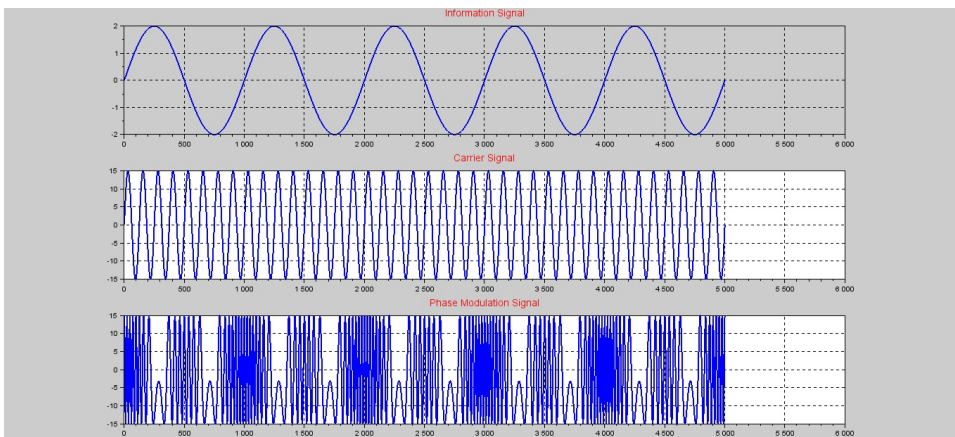


Figure 1.3: Perform the following Analog Modulation Scheme on Binary Data

## Experiment: 2

# Digital Modulation and Demodulation Schemes

**Scilab code Solution 2.1** Perform the following Digital Modulation Scheme on Binary Data

```
1 // LAB:2 – Perform the following Digital Modulation
  Scheme on Binary Data
2 //           (I) Amplitude Shift Keying (II) Frequency
  Shift Keying (III) Phase Shift keying
3
4 // Version : Scilab 5.4.1
5 // Operating System : Window-xp, Window-7
6
7 clc;
8 clear;
9 xdel(winsid());
10 g=[1 0 1 0 0 1 1 0 1 0] //binary data
11 f1=4; f2=8; // Carrier Frequencies
12 t=0:2*%pi/99:2*%pi; // Time
13 //ASK
14 cp=[]; bit=[]; mod_ask=[]; mod_fsk=[]; mod_psk=[]; cp1
  =[]; cp2=[];
15 for n=1:length(g); //ASK modulation
```

```

16     if g(n)==0;
17         die=zeros(1,100);
18     else g(n)==1;
19         die=ones(1,100);
20     end
21     c_ask=sin(f1*t); // Signal with Frequency f1
22     cp=[cp die];
23     mod_ask=[mod_ask c_ask];
24     end
25 ask=cp.*mod_ask;//ASK modulated signal
26
27 //////////////// Frequency Shift Keying ////////////////
28 for n=1:length(g);
29     if g(n)==0;
30         die=ones(1,100);
31         c_fsk=sin(f1*t); // Signal with Frequency f1
32     else g(n)==1;
33         die=ones(1,100);
34         c_fsk=sin(f2*t); // Signal with Frequency f2
35     end
36     cp1=[cp1 die];
37     mod_fsk=[mod_fsk c_fsk];
38 end
39 fsk=cp1.*mod_fsk;//FSK modulated signal
40
41 //PSK
42 for n=1:length(g);
43     if g(n)==0;
44         die=ones(1,100);
45         c_psk=sin(f1*t);
46     else g(n)==1;
47         die=ones(1,100);
48         c_psk=-sin(f1*t);
49     end
50     cp2=[cp2 die];
51     mod_psk=[mod_psk c_psk];
52 end
53 psk=cp2.*mod_psk;//PSK modulated signal

```

```

54 subplot(4,1,1);plot(cp,'LineWidth',1.5);//plot
    binary signal
55 xgrid;
56 title('Binary Signal');//title the Graph
57 mtlb_axis([0 100*length(g) -2.5 2.5]); //axis range
58 subplot(4,1,2);plot(ask,'LineWidth',1.5);//plot of
    ASK modulated signal
59 xgrid;
60 title('ASK modulation');//title of plot
61 mtlb_axis([0 100*length(g) -2.5 2.5]); //axis range
62 subplot(4,1,3);plot(fsk,'LineWidth',1.5);//plot of
    FSK modulated signal
63 xgrid;
64 title('FSK modulation');//title of plot
65 mtlb_axis([0 100*length(g) -2.5 2.5]); //axis range
66 subplot(4,1,4);plot(psk,'LineWidth',1.5);//plot of
    PSK modulated signal
67 xgrid;
68 title('PSK modulation');//title of plot
69 mtlb_axis([0 100*length(g) -2.5 2.5]); //range of
    axis

```

---

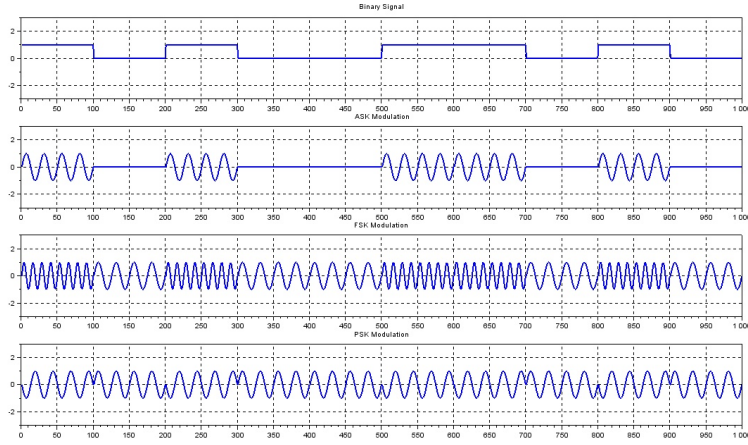


Figure 2.1: Perform the following Digital Modulation Scheme on Binary Data

## Experiment: 3

# Performance analysis of Digital Communication System

**Scilab code Solution 3.1** Measure the Performance of Communication System in term of Bit Error Rate

```
1 // LAB:3 – Measure the Performance of Communication
   System in term of Bit Error Rate(BER).
2 // Version : Scilab 5.4.1
3 // Operating System : Window–xp, Window–7
4
5 clc;
6 clear;
7 close;
8
9 temp=[];
10 temp2=[];
11 for snr=0:0.5:8
12
13     x=rand(1,100000,"uniform"); // Random Data
       Generation with uniform distribution
14     y=ones(1,100000);
15     out=ones(1,100000);
16     [r0 c0]=find(x<0.5);
```

```

17     [r1 c1]=find(x>=0.5);
18     y(r0,c0)=-1;
19
20     var=(1/(10^(snr/10)))/2;
21     noise=var*rand(1,100000,"normal");
22     sig_noise=y+noise; //Noise added to Signal
23     r0=[];
24     c0=[];
25     [r0,c0]=find(sig_noise<0.2228154); //0.2228154
26     out(r0,c0)=-1;
27     total_error=0;
28     for i=1:100000
29         if(y(i)~=out(i))
30             total_error=total_error+1;
31         end
32     end
33     q=erfc(sqrt(2/var));
34
35     temp=[temp (total_error/100000)];
36     temp2=[temp2 q];
37
38     end
39
40     figure;
41     plot(0:0.5:8,temp,0:0.5:8,temp2,'r');
42     xtitle("SNR Vs BER","SNR","BER");
43     legend(['BER Practical';'BER Theoretical']);

```

---

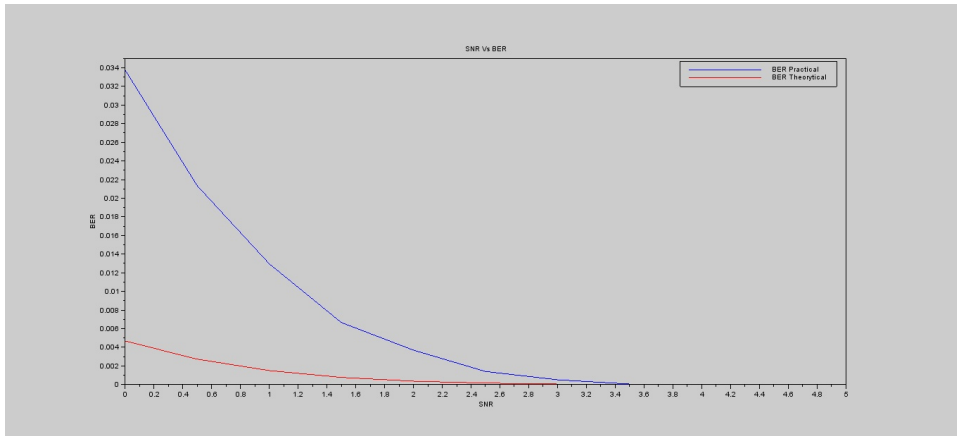


Figure 3.1: Measure the Performance of Communication System in term of Bit Error Rate

# Experiment: 4

## Generation of Waveforms

**Scilab code Solution 4.1** To generate basic discrete signal used in Digital Signal Processing

```
1 // LAB:4 To generate basic discrete signal used in
   Digital Signal Processing
2
3 // Version : Scilab 5.4.1
4 // Operating System : Window-xp, Window-7
5
6 clc;
7 clear;
8 xdel(winsid());
9 t=0:0.1:20;
10 f=0.2;
11 pi=3.14;
12
13 ////////////////////////////////////// SINEWAVE
   //////////////////////////////////////
14 x1=sin(2*pi*f*t);
15 //scf();
16 subplot(231);
17 plot2d3(t,x1);
18 title('Sinewave','color','red','fontsize',2);
```

```

19 xlabel("Index", "fontsize", 2, "color", "blue");
20 ylabel("Amplitude", "fontsize", 2, "color", "blue");
21
22 //////////////////////////////////////// Cosine Wave
23 ///////////////////////////////////////////////////
24 x2=cos(2*pi*f*t);
25 //scf();
26 subplot(232);
27 plot2d3(t,x2);
28 title('Cosinewave', 'color', 'red', 'fontsize', 2);
29 xlabel("Index", "fontsize", 2, "color", "blue");
30 ylabel("Amplitude", "fontsize", 2, "color", "blue");
31
32 //////////////////////////////////////// Impulse Wave
33 ///////////////////////////////////////////////////
34 t1=-10:10;
35 x3=[zeros(1,10) 1 zeros(1,10)];
36 //scf();
37 subplot(233);
38 plot2d3(t1,x3);
39 title('Impulse', 'color', 'red', 'fontsize', 2);
40 xlabel("Index", "fontsize", 2, "color", "blue");
41 ylabel("Amplitude", "fontsize", 2, "color", "blue");
42
43 //////////////////////////////////////// Ramp Wave
44 ///////////////////////////////////////////////////
45 t4=0:10;
46 x4=t4;
47 //scf();
48 subplot(234);
49 plot2d3(t4,x4);
50 title('Ramp Wave', 'color', 'red', 'fontsize', 2);
51 xlabel("Index", "fontsize", 2, "color", "blue");
52 ylabel("Amplitude", "fontsize", 2, "color", "blue");
53
54 //////////////////////////////////////// Exponetial Wave

```

```

        ////////////////
54 t5=0:10;
55 x5=exp(t5);
56 //scf();
57 subplot(235);
58 plot2d3(t5,x5);
59 title('Exponetial Wave','color','red','fontsize',2);
60 xlabel("Index", "fontsize", 2,"color", "blue");
61 ylabel("Amplitude", "fontsize", 2, "color", "blue");
62
63
64 //////////////// Random Wave
        ////////////////
65
66 x6=rand(1,100);
67 //scf();
68 subplot(236);
69 plot2d3(1:length(x6),x6);
70 title('Random Wave','color','red','fontsize',2);
71 xlabel("Index", "fontsize", 2,"color", "blue");
72 ylabel("Amplitude", "fontsize", 2, "color", "blue");
73
74
75
76 ////////////////Impulse Sequence ////////////////
77 n1=1,n0=50,n2=100;
78 if((n0<n1)|(n0>n2)|(n1>n2))
79     error('arugument incorrect');
80 end
81 n=[n1:n2];
82 x7=[(n-n0)==0,1];
83 scf()
84 subplot(221);
85 plot2d3(n,x7(n1:n2));
86 title('Impulse Sequence','color','red','fontsize',2)
    ;
87 xlabel("Index", "fontsize", 2,"color", "blue");
88 ylabel("Amplitude", "fontsize", 2, "color", "blue");

```

```

89
90
91 ////////////////////////////////////////////////// Step Sequence ///////////////////////////////////
92 n1=1,n0=50,n2=100;
93 if((n0<n1)|(n0>n2)|(n1>n2))
94     error('arugument incorrect');
95 end
96 n=[n1:n2];
97 x8=[(n-n0)>=0,1];
98 subplot(222);
99 plot2d3(n,x8(n1:n2));
100 title('Step Sequence','color','red','fontsize',2);
101 xlabel("Index","fontsize",2,"color","blue");
102 ylabel("Amplitude","fontsize",2,"color","blue");
103
104
105 ////////////////////////////////////////////////// RECTANGULAR FUNCTION
    ///////////////////////////////////
106 t=-5:0.1:5
107 y=[zeros(1,45) ones(1,11) zeros(1,45)]
108 subplot(223)
109 plot2d3(t,y)
110 title('Rectangular Function','color','red','fontsize',2);
111 xlabel("Index","fontsize",2,"color","blue");
112 ylabel("Amplitude","fontsize",2,"color","blue");
113
114 ////////////////////////////////////////////////// TRIANGULAR FUNCTION
    ///////////////////////////////////
115 t=0:1:10
116 y=t;
117 z=11-t;
118 b=[y z y z y z];
119 subplot(224)
120 plot2d3(b)
121 title('Triangular Function','color','red','fontsize',2);
122 xlabel("Index","fontsize",2,"color","blue");

```

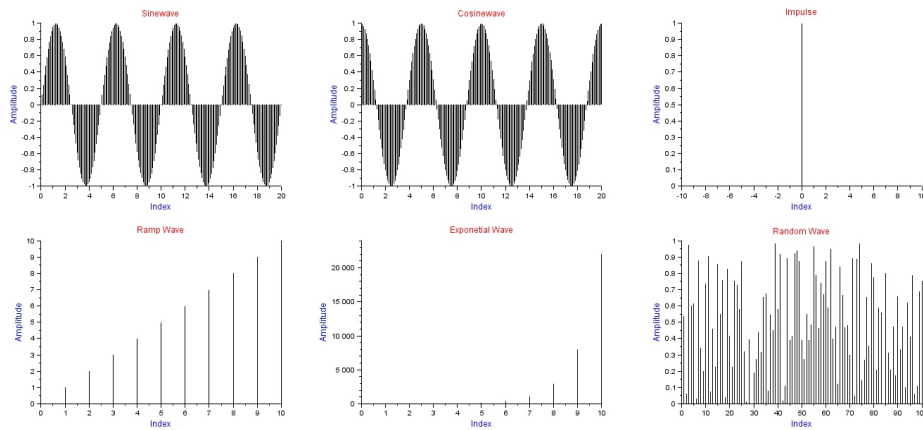


Figure 4.1: To generate basic discrete signal used in Digital Signal Processing

```

123 ylabel(" Amplitude", " fontsize", 2, " color", " blue");
124
125
126 ////////////////////////////////////////////////// SINC FUNCTION //////////////////////////////////////
127 x=linspace(-10,10,3000);
128 figure;
129 plot2d3(x,sinc(x))
130 title('SINC Function','color','red','fontsize',2);
131 xlabel("Index", " fontsize", 2," color", " blue");
132 ylabel(" Amplitude", " fontsize", 2, " color", " blue");

```

---

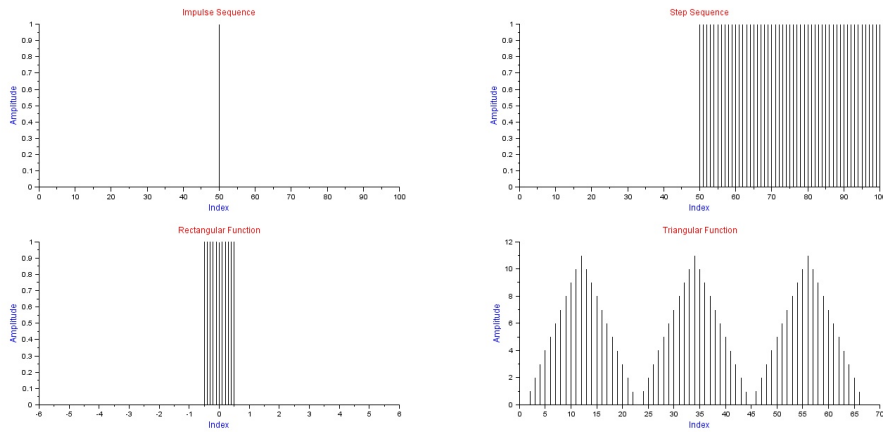


Figure 4.2: To generate basic discrete signal used in Digital Signal Processing

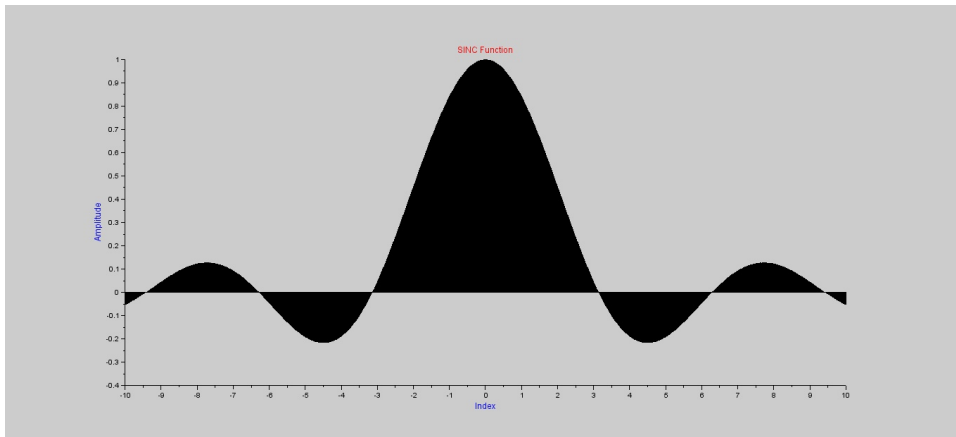


Figure 4.3: To generate basic discrete signal used in Digital Signal Processing

# Experiment: 5

## Properties of Signals and Systems

Scilab code Solution 5.1 Property of Signal and System

```
1 // LAB:5 – Property of Signal and System.
2 // Version : Scilab 5.4.1
3 // Operating System : Window–xp, Window–7
4
5 clc;
6 close;
7 clear;
8 xdel(winsid());
9 /////////////////////////////////////////////////// LINEAR PROPERTY ///////////////////////////////////
10 x1=[0 0 ones(1,10)]
11 x2=[ones(1,12)]
12 t=0:1:11
13 y1=t.*x1
14 y2=t.*x2
15 x3=[2*x1+3*x2].*t
16 y3=2*y1+3*y2
17 figure;
18 subplot(4,1,1)
19 plot(t,x1)
```

```

20 title('Signal 1','color','red','fontsize',2,'
    position',[0.3 0.8]);
21 xlabel("Index", "fontsize", 2,"color", "blue");
22 ylabel("Amplitude", "fontsize", 2, "color", "blue");
23 subplot(4,1,2)
24 plot(t,x2)
25 title('Signal 2','color','red','fontsize',2,'
    position',[0.3 1.5]);
26 xlabel("Index", "fontsize", 2,"color", "blue");
27 ylabel("Amplitude", "fontsize", 2, "color", "blue");
28 subplot(4,1,3)
29 plot(t,y1)
30 title('Y1=t*Signal 1','color','red','fontsize',2,'
    position',[0.3 7]);
31 xlabel("Index", "fontsize", 2,"color", "blue");
32 ylabel("Amplitude", "fontsize", 2, "color", "blue");
33 subplot(4,1,4)
34 plot(t,y2)
35 title('Y2=t*Signal 2','color','red','fontsize',2,'
    position',[0.3 7]);
36 xlabel("Index", "fontsize", 2,"color", "blue");
37 ylabel("Amplitude", "fontsize", 2, "color", "blue");
38 figure
39 subplot(2,1,1)
40 plot(t,x3)
41 title('x3=[2*Signal1+3*Signal2].*t','color','red','
    fontsize',2);
42 xlabel("Index", "fontsize", 2,"color", "blue");
43 ylabel("Amplitude", "fontsize", 2, "color", "blue");
44 subplot(2,1,2)
45 plot(t,y3)
46 title('y3=2*y1+3*y2','color','red','fontsize',2);
47 xlabel("Index", "fontsize", 2,"color", "blue");
48 ylabel("Amplitude", "fontsize", 2, "color", "blue");
49
50 //////////////////////////////////////////////////// TIME VARIANT
    //////////////////////////////////////
51 x1=[0,0,ones(1,10)]

```

```

52 t=0:1:11
53 y1=t.*x1
54 t2=t+5
55 figure
56 subplot(2,1,1)
57 plot(t2,x1)
58 title('Time Variant Property','color','red','
    fontsize',2);
59 xlabel("Index", "fontsize", 2,"color", "blue");
60 ylabel("Amplitude", "fontsize", 2, "color", "blue");
61 subplot(2,1,2)
62 plot(t2,y1)
63 xlabel("Index", "fontsize", 2,"color", "blue");
64 ylabel("Amplitude", "fontsize", 2, "color", "blue");
65
66
67 ////////////////////////////////////////////////// CAUSAL & NON CAUSAL System
    //////////////////////////////////////
68 x=[1,4,2,8,0,4,3]
69 t=0:1:6
70 t1=t*2
71 t2=t/2;
72 figure;
73 subplot(3,1,1)
74 plot(t,x)
75 title('CAUSAL & NON CAUSAL Property','color','red','
    fontsize',2);
76 xlabel("Index", "fontsize", 2,"color", "blue");
77 ylabel("Amplitude", "fontsize", 2, "color", "blue");
78 subplot(3,1,2)
79 plot(t1,x)
80 title('CAUSAL Property','color','red','fontsize',2,'
    position',[0.3 6]);
81 xlabel("Index", "fontsize", 2,"color", "blue");
82 ylabel("Amplitude", "fontsize", 2, "color", "blue");
83 subplot(3,1,3)
84 plot(t2,x)
85 title('NON CAUSAL Property','color','red','fontsize'

```

```

    ,2, 'position ', [0.3 6]);
86 xlabel("Index", "fontsize", 2, "color", "blue");
87 ylabel("Amplitude", "fontsize", 2, "color", "blue");
88
89
90 /////////////////////////////////////////////////// STATIC & DYNAMIC
   ///////////////////////////////////////////////////
91 x1=[4,6,3,2,9,5]
92 t=0:1:5;
93 y1=x1.^2;
94 t1=t-5;
95 figure;
96 subplot(3,1,1)
97 plot(t,x1)
98 title('STATIC & DYNAMIC Property', 'color', 'red', '
    fontsize', 2);
99 xlabel("Index", "fontsize", 2, "color", "blue");
100 ylabel("Amplitude", "fontsize", 2, "color", "blue");
101 subplot(3,1,2)
102 plot(t,y1)
103 title('STATIC Property', 'color', 'red', 'fontsize', 2, '
    position ', [0.3 60]);
104 xlabel("Index", "fontsize", 2, "color", "blue");
105 ylabel("Amplitude", "fontsize", 2, "color", "blue");
106 subplot(3,1,3)
107 plot(t1,y1)
108 title('DYNAMIC Property', 'color', 'red', 'fontsize', 2, '
    position ', [-4.8 60]);
109 xlabel("Index", "fontsize", 2, "color", "blue");
110 ylabel("Amplitude", "fontsize", 2, "color", "blue");

```

---

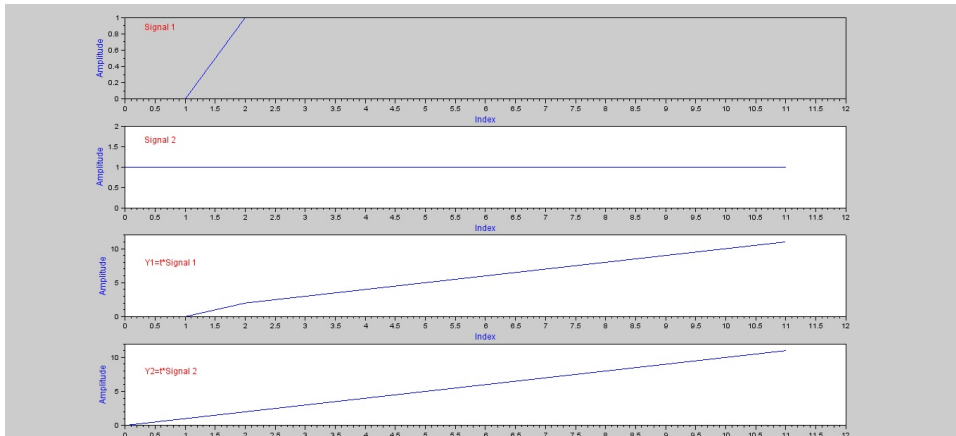


Figure 5.1: Property of Signal and System

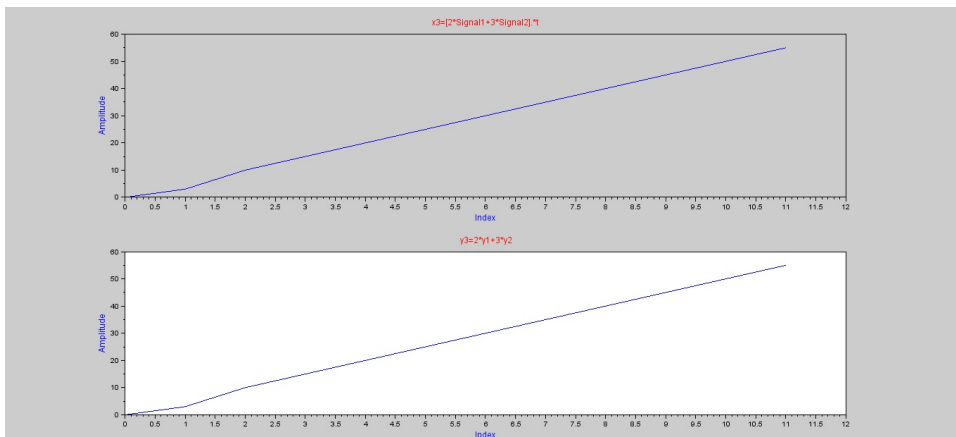


Figure 5.2: Property of Signal and System

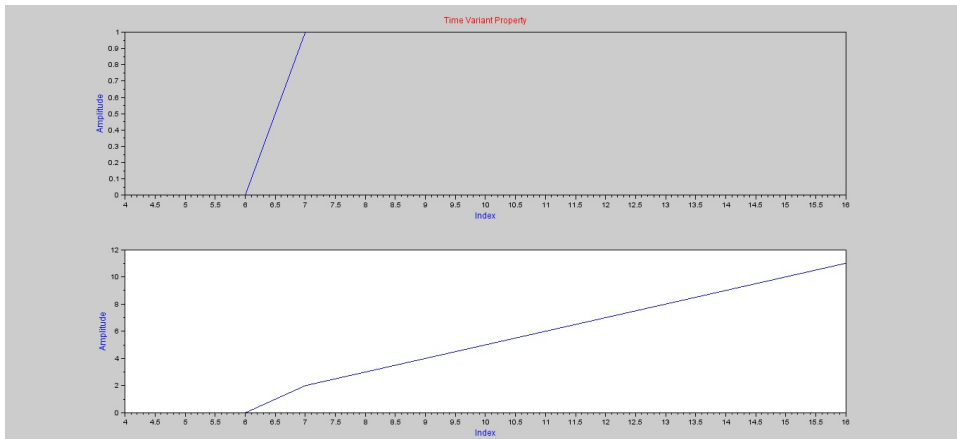


Figure 5.3: Property of Signal and System

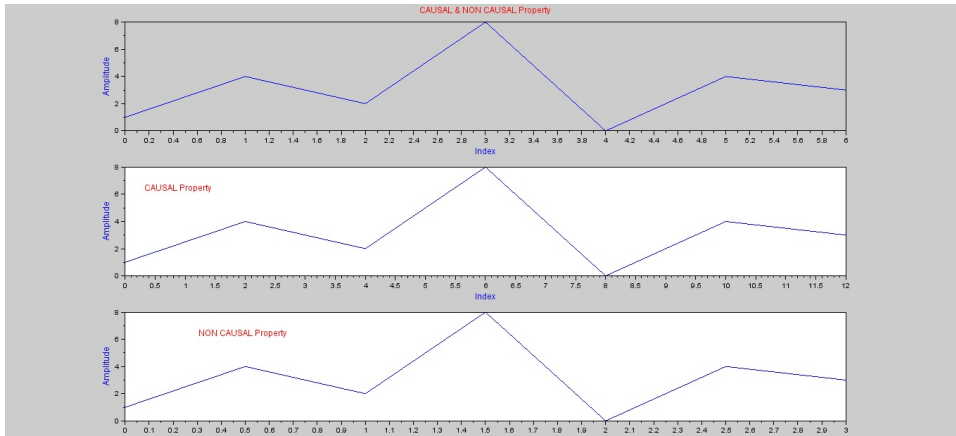


Figure 5.4: Property of Signal and System

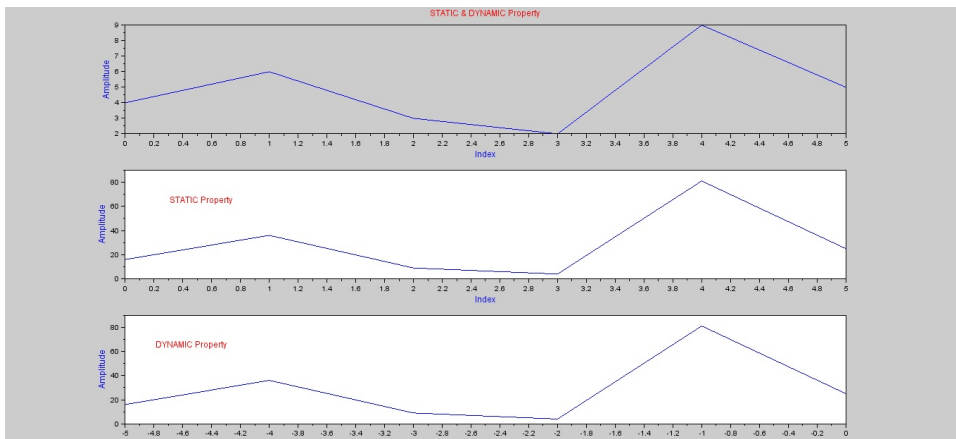


Figure 5.5: Property of Signal and System

# Experiment: 6

## Convolution

**Scilab code Solution 6.1** Perform Linear and Circular Convolution Operation on Two Discrete Sequences

```
1 // LAB:6 Perform Linear and Circular Convolution
   Operation on Two Discrete Sequences
2
3 // Version : Scilab 5.4.1
4 // Operating System : Window-xp, Window-7
5
6
7 clc;
8 clear;
9 xdel(winsid());
10
11 //////////////////////////////////// LINEAR CONVOLUTION
   ////////////////////////////////////
12 disp('LINEAR CONVOLUTION OPERATION');
13 x1=input("Enter the Sequence_1 :"); // [1 2 3 4
   5];
14 x2=input("Enter the Sequence_2 :"); // [5 4 8];
15 n = length(x1);
16 m = length(x2);
17 for k = 1:(m+n-1)
```

```

18     w(k) = 0;
19         for j =max(1,k+1-m) : min(k,n)
20             w(k)= w(k)+(x1(j)*x2(k+1-j));
21         end
22 end
23 disp('Convoled Sequence: ');
24 disp(w);
25 scf();
26 subplot(3,1,1);
27 bar(x1,0.05,'red');
28 title('Sequence_1','color','red','fontsize',4);
29 xlabel("Index", "fontsize", 2,"color", "blue", '
    position',[0.6 0.3]);
30 ylabel("Amplitude", "fontsize", 2, "color", "blue");
31
32 subplot(3,1,2);
33 bar(x2,0.05,'yellow');
34 title('Sequence_2','color','red','fontsize',4);
35 xlabel("Index", "fontsize", 2,"color", "blue", '
    position',[0.6 0.3]);
36 ylabel("Amplitude", "fontsize", 2, "color", "blue");
37
38 subplot(3,1,3);
39 bar(w,0.05,'green');
40 title('Convoled Sequence','color','green','fontsize',
    4);
41 xlabel("Index", "fontsize", 2,"color", "blue", '
    position',[0.3 0.3]);
42 ylabel("Amplitude", "fontsize", 2, "color", "blue");
43
44 //////////////// CIRCULAR CONVOLUTION
    ////////////////
45 disp('CIRCULAR CONVOLUTION OPERATION');
46 x1=input('Enter First Sequence : '); // [1 2 3 4
    5];
47 x2=input('Enter Second Sequence : '); // [1 2 3 4
    5];
48 l1=length(x1);

```

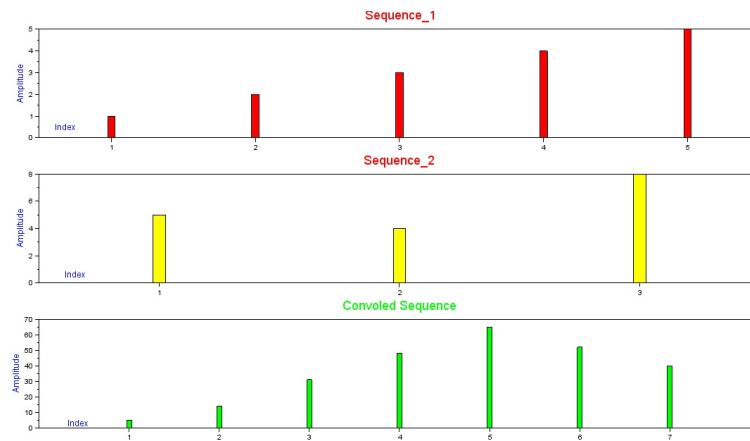


Figure 6.1: Perform Linear and Circular Convolution Operation on Two Discrete Sequences

```

49 l2=length(x2);
50 if(l1==l2)
51     a=[x1(1) x1(l1:-1:2)];
52     b=a;
53     for i=2:l1
54         a=[a(l1) a(1:l1-1)];
55         b=[b;a];
56     end
57     c=x2';
58     d=b*c;
59     y=d';
60     disp('Circular Convolution Output : ')
61     disp(y);
62 else
63     disp('Circular Convolution is not possible.')
64 end

```

---

# Experiment: 7

## Correlation

**Scilab code Solution 7.1** Perform Correlation Operation on Two Discrete Sequences

```
1 // LAB:7 Perform Correlation Operation on Two
   Discrete Sequences
2
3 // Version : Scilab 5.4.1
4 // Operating System : Window-xp, Window-7
5
6 clc;
7 clear;
8 xdel(winsid());
9 x=[2 4 5 6];
10 y=[2 4 5];
11
12 m=length(x);
13 n=length(y);
14
15 for k=1:m+n-1
16     w(k)=0;
17     for j=max(1,k+1-n):min(k,m)
18         w(k)=w(k)+(x(j)*y(n-k+j));
19     end
```

```

20 end
21 disp('Sequence 1:');
22 disp(x);
23 disp('Sequence 2:');
24 disp(y);
25 disp('Correlation Out Put:');
26 disp(w);
27 ////////////// Graphical Display //////////////
28 scf();
29 subplot(3,1,1);
30 bar(x,0.05,'red');
31 title('Sequence_1','color','red','fontsize',4);
32 xlabel("Index", "fontsize", 2,"color", "blue", '
    position',[0.6 0.3]);
33 ylabel("Amplitude", "fontsize", 2, "color", "blue");
34
35 subplot(3,1,2);
36 bar(y,0.05,'yellow');
37 title('Sequence_2','color','red','fontsize',4);
38 xlabel("Index", "fontsize", 2,"color", "blue", '
    position',[0.6 0.3]);
39 ylabel("Amplitude", "fontsize", 2, "color", "blue");
40
41 subplot(3,1,3);
42 bar(w,0.05,'green');
43 title('Correlation of Sequences','color','green', '
    fontsize', 4);
44 xlabel("Index", "fontsize", 2,"color", "blue", '
    position',[0.3 0.3]);
45 ylabel("Amplitude", "fontsize", 2, "color", "blue");

```

---

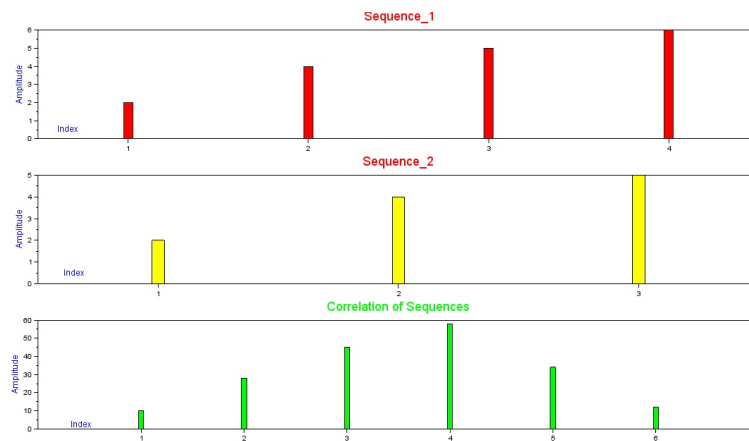


Figure 7.1: Perform Correlation Operation on Two Discrete Sequences

# Experiment: 8

## Finding DFT and IDFT using FFT

Scilab code Solution 8.1 Perform DFT and IDFT of discrete signal

```
1 // LAB:8 Perform DFT and IDFT of discrete signal.
2 // Version : Scilab 5.4.1
3 // Operating System : Window-xp, Window-7
4
5
6 clc;
7 close;
8 clear;
9 x=[1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0]; // Step
    Sequence
10 N=length(x); // Number of Point DFT
11 for k=1:2*N
12     y(k)=0;
13     for n=1:N
14         y(k)=y(k)+(x(n)*exp((-2*%pi*(k-1)*(n-1)*%i)/
            N));
15     end
16 end
17 subplot(311);
```

```

18 plot2d3(x);
19 title('Step Sequence','color','Red','fontsize',3);
20 xlabel('Time Index');
21 ylabel('Amplitude');
22
23 subplot(312);
24 plot2d3(abs(y));
25 title('DFT','color','Red','fontsize',3);
26 xlabel('Frequency Scale');
27 ylabel('Amplitude');
28
29 //////////////// IDFT ////////////////
30 for n=1:N
31 p(n)=0;
32 for k=1:N
33 p(n)=p(n)+((y(k)*exp((%i*2*pi*(k-1)*(n-1))/N))/N);
34 end
35 end
36 subplot(313);
37 plot2d3(abs(p));
38 title('IDFT','color','Red','fontsize',3);
39 xlabel('Time Index');
40 ylabel('Amplitude');

```

---

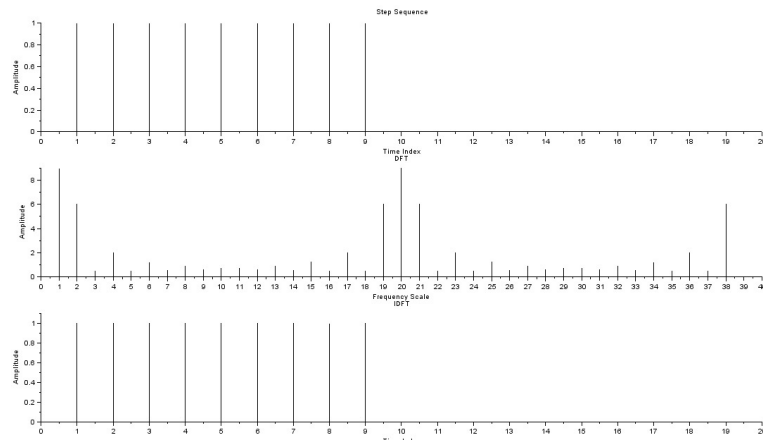


Figure 8.1: Perform DFT and IDFT of discrete signal

# Experiment: 9

## Design of FIR filters

**Scilab code Solution 9.1** Design the following FIR filters with the given specification

```
1 // LAB:09  Design the following FIR filters with the
    given specification .
2 //
3 //
4 //
    //////////////////////////////////////
5 //      Example :
6 //          filter type ('lp', 'hp', 'sb', 'bp')
7 //          Filter order (pos integer)(odd
    for ftype='hp' or 'sb')
8 //          cfreq=2-vector of cutoff
    frequencies (0<cfreq(1),cfreq(2)<.5) only cfreq
    (1) is used when ftype='lp' or 'hp'
9 //          wtype= Window type ('re', 'tr', 'hm
    ', 'hn', 'kr', 'ch')
10 //          fpar=2-vector of window
    parameters. Kaiser window fpar(1)>0 fpar(2)=0.
    Chebyshev window fpar(1)>0, fpar(2)<0 or fpar(1)
    <0, 0<fpar(2)<.5
```

```

11 //          wft=time domain filter
    coefficients
12 //          wfm=frequency domain filter
    response on the grid fr
13 //          fr=Frequency grid
14 //
    //////////////////////////////////////
15
16 //Evaluate magnitude response of the filter
17
18 // Version : Scilab 5.4.1
19 // Operating System : Window-xp, Window-7
20
21 clc;
22 close;
23 clear;
24
25 ftype='bp';
26 forder=33;
27 fs=8000;
28 cfreq=[(450/fs) (500/fs)];
29 wtype='kr';
30 fpar=[0.8 0];
31
32
33 [wft1,wfm1,fr1]=wfir(ftype,forder,cfreq,'re',fpar);
34 [wft2,wfm2,fr2]=wfir(ftype,forder,cfreq,'hm',fpar);
35 [wft3,wfm3,fr3]=wfir(ftype,forder,cfreq,'hn',fpar);
36 [wft4,wfm4,fr4]=wfir(ftype,forder,cfreq,'kr',fpar);
37 [wft5,wfm5,fr5]=wfir(ftype,forder,cfreq,'tr',fpar);
38 // [wft6,wfm6,fr6]=wfir(ftype,forder,cfreq,'ch',fpar)
    ;
39
40 clf();
41 plot(fr1,wfm1,fr2,wfm2,fr3,wfm3,fr4,wfm4,fr5,wfm5);
42 legend('rectangal Window','Hamming Window','Hanning
    Window','Kaiser Window','Triagle Window');

```

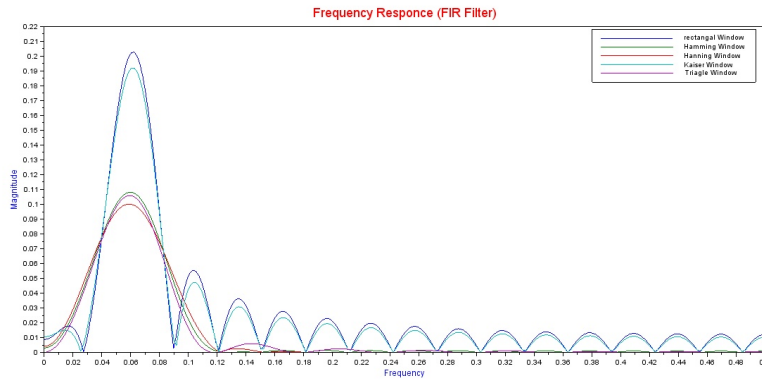


Figure 9.1: Design the following FIR filters with the given specification

```

43 title('Frequency Responce', 'color', 'red', 'fontsize',
44       4);
44 xlabel("Frequency", "fontsize", 2, "color", "blue");
45 ylabel("Magnitude", "fontsize", 2, "color", "blue");

```

---

# Experiment: 10

## Design of IIR filters

**Scilab code Solution 10.1** Design the following IIR filters with the given specification

```
1 // LAB:10 : Design the following IIR filters with
  the given specification.
2 //           (1) Butter Worth   (2) Chebyshev-I   (3)
  Chebyshev-II   (4) Elliptical
3
4 //
  ///////////////////////////////////////////////////////////////////

5 //   Example :
6 //           filter type ('lp', 'hp', 'sb', 'bp')
7 //           design approximation ('butt', '
  cheb1', 'cheb2', 'ellip')
8 //           om=[om1,om2,om3,om4], 0 <= om1 <=
  om2 <= om3 <= om4 <= pi .When ftype='lp' or 'hp
  ', om3 and om4 are not used and may be set to 0.
9 //           0<= deltap <=1
10 //          0<= deltas <=1
11 //
  ///////////////////////////////////////////////////////////////////
```

```

12
13 //Evaluate magnitude response of the filter
14
15 // Version : Scilab 5.4.1
16 // Operating System : Window-xp, Window-7
17
18 clc;
19 close;
20 clear;
21
22 ftype='bp'; // Type of Filter
23 approx='ellip'; //Design Approximation
24 om=[.15 .25]; // Cut off Frequency
25 deltap=0.08;
26 deltas=0.03;
27
28 hz_ellip=iir(3,ftype,approx,om,[deltap deltas]); //
    Band Pass Filter with Elliptic
29 [hzm1,fr1]=frmag(hz_ellip,256); //Frequency
    Magnitude
30 hz_butt=iir(3,ftype,'butt',om,[deltap deltas]); //
    Band Pass Filter with Butterworth
31 [hzm2,fr2]=frmag(hz_butt,256); //Frequency Magnitude
32 hz_cheby1=iir(3,ftype,'cheb1',om,[deltap deltas]);
    // Band Pass Filter with Chebysev 1
33 [hzm3,fr3]=frmag(hz_cheby1,256); //Frequency
    Magnitude
34 hz_cheby2=iir(3,ftype,'cheb2',om,[deltap deltas]);
    // Band Pass Filter with Chebysev 1
35 [hzm4,fr4]=frmag(hz_cheby2,256); //Frequency
    Magnitude
36
37 plot(fr1',hzm1',fr2',hzm2',fr3',hzm3',fr4',hzm4');
38 xtitle('Discrete IIR filter band pass 0.15 < fr <
    0.25 ',' ',' ');
39 xlabel('Frequency Scale');
40 ylabel('Magnitude');
41 h=legend(['Ellip';'Butter';'Chaby1';'Cheby2']);

```

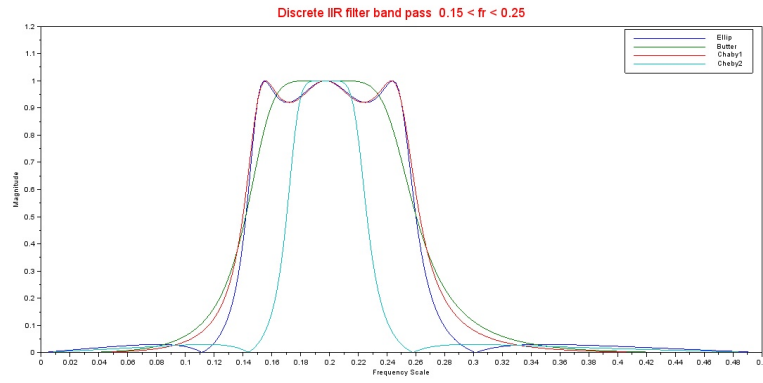


Figure 10.1: Design the following IIR filters with the given specification

# Experiment: 11

## Error control Codes

**Scilab code Solution 11.1** Perform the Linear Block Coding on binary Data

```
1 // LAB:11 – Perform the Linear Block Coding on
  binary Data.
2 // Version : Scilab 5.4.1
3 // Operating System : Window–xp, Window–7
4
5 clc;
6 close;
7 clear;
8 xdel(winsid());
9
10 global P n k;
11
12 n=7;
13 k=4;
14 P=[1 1 0; 0 1 1; 1 0 1;1 1 1]; //% parity matrix of
  size k*(n–k) to be
15 //                               % selected so that
  the systematic generator
16 //                               % matrix is
  linearly independent or full rank
```

```

17 //                                     % matrix
18
19
20 //%This is an linear block encoding function file%
21
22 function y1=linblkcode(x);
23 global P n k;
24 G=[eye(k,k) P];
25
26 y1=zeros(1,n);
27 for i=1:k
28     var(i,:)=x(1,i) & G(i,:);
29     var(i,:)=bool2s(var(i,:));
30     y1(1,:)=bitxor(var(i,:),y1(1,:));
31 end
32
33 endfunction
34
35
36 //%This is a linear block syndrome decoding function
    file%
37
38 function x1=linblkdecoder(y)
39 //% here y is recieved vector 7 bits long(7,4)
    linear block code
40
41 global P n k;
42
43
44 //H=[ ]; //% PARITY CHECK MATRIX
45
46 H=[P' eye((n-k),(n-k))];
47 Ht=H'; //%transpose of H
48
49 S=zeros(1,n-k); //%syndrome of recieved vector x
50 for i=1:n-k
51     S(i)=y(1) & Ht(1,i);
52     S(i)=bool2s(S(i));

```

```

53     for j=2:n
54
55         S(i)=bitxor(S(i), bool2s((y(j) & Ht(j,i))));
56     end
57 end
58 //%%***SYNDROME LOOK UP TABLE*****
59 if S==[0 0 0]
60     e=[0 0 0 0 0 0 0];
61     z=bitxor(y,e);
62 end
63
64 if S==[0 0 1]
65     e=[0 0 0 0 0 0 1];
66     z=bitxor(y,e);
67 end
68 if S==[0 1 0]
69     e=[0 0 0 0 0 1 0];
70     z=bitxor(y,e);
71 end
72 if S==[1 0 0]
73     e=[0 0 0 0 1 0 0];
74     z=bitxor(y,e);
75 end
76 if S==[1 1 1]
77     e=[0 0 0 1 0 0 0];
78     z=bitxor(y,e);
79 end
80 if S==[1 0 1]
81     e=[0 0 1 0 0 0 0];
82     z=bitxor(y,e);
83 end
84 if S==[0 1 1]
85     e=[0 1 0 0 0 0 0];
86     z=bitxor(y,e);
87 end
88 if S==[1 1 0]
89     e=[1 0 0 0 0 0 0];
90     z=bitxor(y,e);

```

```

91 end
92
93 x1=z(1,1:k);
94 endfunction
95
96 ////////////////////////////////// Main Programm
97 //////////////////////////////////
97 x=[0 1 0 1]; // % input bits to the
encoder of size 1* k
98 y1=linblkcode(x);// // % y1 is the
output of linear block encoder
99
100 e1=[1 0 0 0 0 0 0]; // % intentionally
error introduced after
101 // % encoding and
before sending to decoder (in
102 // % this case pls
introduce only one bit error)
103 y=bitxor(y1,e1);// % input that will
be made available to linear
104 // % block decoder
105
106 x1=linblkdecoder(y) // % x1 is the output
of the linear block decoder
107 // % which will be
same as x provided that
here
108 // % have introduced
only one bit error

109
110 disp('The information signal=')
111 disp(x)
112 disp('The transmitted encoded signal=')
113 disp(y1)
114 disp('The recieved signal=')
115 disp(y)
116 disp('The decoded signal=')
117 disp(x1)

```

```
118
119
120 //////////////////////////////////////////////////////////////////// Input and Output Values
    ////////////////////////////////////////////////////////////////////
121 //The information signal(x)=[0 1 0 1];
122 //The transmitted encoded signal(y1)=[0 1 0 1 1 0
    0];
123 //The recieved signal(y)=[1 1 0 1 1 0 0];
124 //The decoded signal(x1)=[0 1 0 1];
```

---

# Experiment: 12

## Basic Image Processing Operations

check Appendix ?? for dependency:

```
gray.tif
```

**Scilab code Solution 12.1** Perform the following basic image processing operation on digital image

```
1 // LAB:12 Perform the following basic image
   // processing operation on digital image.
2 //(I)RGB to Gray (II) Image Information (III) Image
   // Resizing (IV) Image Cropping (V) Image Negative (
   // VI) Gamma Intensity transformation
3 // Version : Scilab 5.4.1
4 // Operating System : Window-xp, Window-7
5 //Toolbox: Image Processing Design 8.3.1-1
6 //Toolbox: SIVP 0.5.3.1-2
7
8 clc;
9 close;
10 clear;
```

```

11 xdel(winsid())//to close all currently open figure(s
    ).
12
13 path=getSIVPpath();
14 Image=imread(path+"images\lena.png");
15 //figure ,ShowColorImage(Image,'RGB Image');
16 //title('RGB Image','color','blue','fontsize',4);
17 imwrite(Image,'LAB12.1.jpg');
18 Gray_Image=rgb2gray(Image); // Convert RGB Image to
    Gray Scale Image
19 //figure ,ShowImage(Gray_Image,'Gray Scale Image');
20 //title('Gray Scale Image','color','blue','fontsize
    ',4);
21 imwrite(Gray_Image,'LAB12.2.jpg');
22 //iminfo(SIVP_Path + 'images\lena.png');// Display
    Image Information
23 Crop_Image = imcrop(Gray_Image, [100, 30, 300, 300])
    ; // Crop the Image form the Specific Location
24 //figure ,ShowImage(Crop_Image,'Image Cropping');
25 //title('Cropped Image','color','blue','fontsize',4);
26 imwrite(Crop_Image,'LAB12.3.jpg');
27 Resize_Image = imresize(Gray_Image,0.5); //Resize
    the Image with Factor 0.5
28 //figure ,ShowImage(Resize_Image,'Image Resizing');
29 //title('Resized Image','color','blue','fontsize',4)
    ;
30 imwrite(Resize_Image,'LAB12.4.jpg');
31
32 //////////////// Negative Intensity Transformation
    ////////////
33 [r p]=size(Gray_Image);
34 for i=1:r
35     for j=1:p
36         Negative_Image(i,j)=255-Gray_Image(i,j);
37     end
38 end
39 //figure ,ShowImage(Negative_Image,'Image Negative');

```

```

40 //title('Nagative Image','color','blue','fontsize
    ',4);
41 imwrite(Negative_Image,'LAB12_5.jpg');
42
43 //////////////// Image Flipping (Left to Right)
    ////////////////
44 [r p]=size(Gray_Image);
45 for i=1:r
46     for j=1:p
47         Fliped_Image(i,j)=Gray_Image(i,p-j+1);
48     end
49 end
50 //figure,ShowImage(Fliped_Image,'Image Flipng (Left
    to Right)');
51 //title('Fliped Image (Left to Right)','color','blue
    ','fontsize',4);
52 imwrite(Fliped_Image,'LAB12_6.jpg');
53
54 //////////////// Gamma Intensity transformation
    ////////////////
55 gray=imread("gray.tif");
56 gray=imresize(gray,0.5);
57 gray1=im2double(gray);
58 //figure,ShowImage(gray,'Gray Image');
59 //title('Original Image','color','blue','fontsize
    ',4);
60 [M,N]=size(gray);
61 temp=[]
62 temp=[temp gray1];
63 c=1;
64 gamma=[0.6 0.4 0.3];
65 for i=1:length(gamma)
66     b=c.*(gray).^gamma(i); //Gamma transformation
67     b=mat2gray(b);
68     temp=[temp ones(M,20) b]; // Padding Data for
        Displaying the Image
69 end
70 //figure,ShowImage(temp,'Gray Image');

```

```
71 // title('Original Image/Gamma Trasformed Images  
    (0.6,0.4,0.3)', 'color', 'blue', 'fontsize', 4);  
72 imwrite(temp, 'LAB12_7.jpg');
```

---



Figure 12.1: Perform the following basic image processing operation on digital image



Figure 12.2: Perform the following basic image processing operation on digital image



Figure 12.3: Perform the following basic image processing operation on digital image



Figure 12.4: Perform the following basic image processing operation on digital image



Figure 12.5: Perform the following basic image processing operation on digital image



Figure 12.6: Perform the following basic image processing operation on digital image



Figure 12.7: Perform the following basic image processing operation on digital image

# Experiment: 13

## 2D Convolution

Scilab code Solution 13.1 Perform the 2D Convolution on digital image

```
1 // LAB:13 Perform the 2D Convolution on digital
   image (Filtering Operation).
2 // Version : Scilab 5.4.1
3 // Operating System : Window-xp, Window-7
4 //Toolbox: Image Processing Design 8.3.1-1
5 //Toolbox: SIVP 0.5.3.1-2
6
7 clc;
8 clear;
9 xdel(winsid());
10 path=getSIVPpath();
11 Image=imread(path+"images\lena.png");
12 gray=rgb2gray(Image);
13 //figure;
14 //ShowImage(gray,"gray image");
15 //title("grayscale",'fontsize',3);
16 imwrite(gray,'LAB13_1.jpg');
17 [row,column]=size(gray);
18
19 filter=fspecial('average',3); // Average Filter
20 imfilt=imfilter(gray,filter); // 2D Convolution
```

```

    between Image and Filter
21 //figure,ShowImage(imfilt,'filtered image'); //
    Display Gary Scale Image in Window
22 //title('averaging filter , order = '+string(i)); //
    Title on Displayed Image
23 imwrite(imfilt,'LAB13.2.jpg');
24
25 filter=fspecial('average',7); // Average Filter
26 imfilt2=imfilter(gray,filter); // 2D Convolution
    between Image and Filter
27 //figure,ShowImage(imfilt,'filtered image'); //
    Display Gary Scale Image in Window
28 //title('averaging filter , order = '+string(i)); //
    Title on Displayed Image
29 imwrite(imfilt2,'LAB13.3.jpg');
30
31 filter=fspecial('average',11); // Average Filter
32 imfilt3=imfilter(gray,filter); // 2D Convolution
    between Image and Filter
33 //figure,ShowImage(imfilt,'filtered image'); //
    Display Gary Scale Image in Window
34 //title('averaging filter , order = '+string(i)); //
    Title on Displayed Image
35 imwrite(imfilt3,'LAB13.4.jpg');

```

---



Figure 13.1: Perform the 2D Convolution on digital image



Figure 13.2: Perform the 2D Convolution on digital image



Figure 13.3: Perform the 2D Convolution on digital image



Figure 13.4: Perform the 2D Convolution on digital image

# Experiment: 14

## Image Transforms -DFT,DCT and DWT

**Scilab code Solution 14.1** Perform the Following Transform on Gray Scale Image

```
1 // LAB:14 Perform the Following Transform on Gray
  Scale Image.
2 //(I)DFT (II)DCT (III)DWT
3 // Version : Scilab 5.4.1
4 // Operating System : Window-xp, Window-7
5 //Toolbox: Image Processing Design 8.3.1-1
6 //Toolbox: SIVP 0.5.3.1-2
7 //Toolbox: Wavelet
8
9
10 clc;
11 clear;
12 xdel(winsid());
13 path=getSIVPpath();
14 Image=imread(path+"images\lena.png");
15 gray=im2double(rgb2gray(Image));
16 //ShowImage(gray,"gray image");
17 //title("grayscale",'fontsize',3);
```

```

18 imwrite(gray, 'LAB14_1.jpg');
19
20 //////////////// DFT ////////////////
21 h1=fft2(gray); //fft2() is used to find 2-Dimensional
    Fast Fourier Transform of an matrix
22 i1=log(1+abs(h1));
23 in1=fftshift(i1); //fftshift() is used to rearrange
    the fft output, moving the zero frequency
24 inm1=mat2gray(in1);
25 //figure;
26 //ShowImage(inm1, 'Frequency Spectrum of Original
    Image');
27 //title("Frequency Spectrum", 'fontsize', 3);
28 imwrite(inm1, 'LAB14_2.jpg');
29
30 //////////////// DCT ////////////////
31 h2=dct(gray, 1); // DCT Transfom
32 //figure;
33 //ShowImage(h2, 'Frequency Spectrum of Original Image
    ');
34 //title("DCT Coefficient", 'fontsize', 3);
35 imwrite(h2, 'LAB14_3.jpg');
36 h3=idct(h2); // IDCT Transfom
37 h4=mat2gray(h3);
38 //figure;
39 //ShowImage(h4, 'Frequency Spectrum of Original Image
    ');
40 //title("Recovered Image using DCT Coefficient", '
    fontsize', 3);
41 imwrite(h4, 'LAB14_4.jpg');
42
43 //////////////// DWT ////////////////
44 [CA CH CV CD]=dwt2(gray, 'db2', 'mode', 'asymh');
45 [M N]=size(CA);
46 temp=[CA ones(M,10) CH ones(M,10) CV ones(M,10) CD];
    //Padding Data for Displayng DWT Coefficients
47 //figure; ShowImage(temp, "CA CH CV CD DWT Coefficient
    ");

```

```
48 //title("CA CH CV CD DWT Coefficient",'fontsize',3 )
    ;
49 imwrite(temp,'LAB14_5.jpg');
50
51 x1=size(gray);
52 X = idwt2(CA,CH,CV,CD,'db2',x1); // Inverse DWT
53 //figure;ShowImage(X,'Recovered Image');
54 //title('inverse dwt','fontsize',3);
55 imwrite(X,'LAB14_6.jpg');
```

---



Figure 14.1: Perform the Following Transform on Gray Scale Image

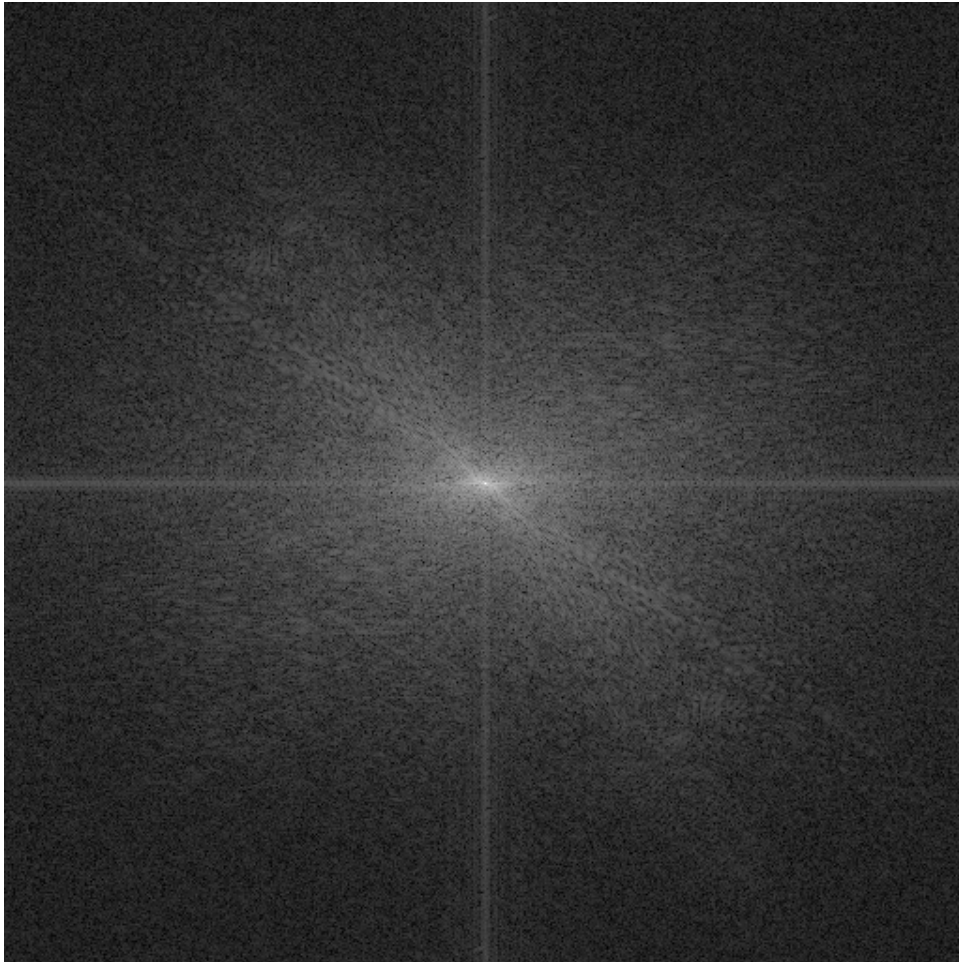


Figure 14.2: Perform the Following Transform on Gray Scale Image

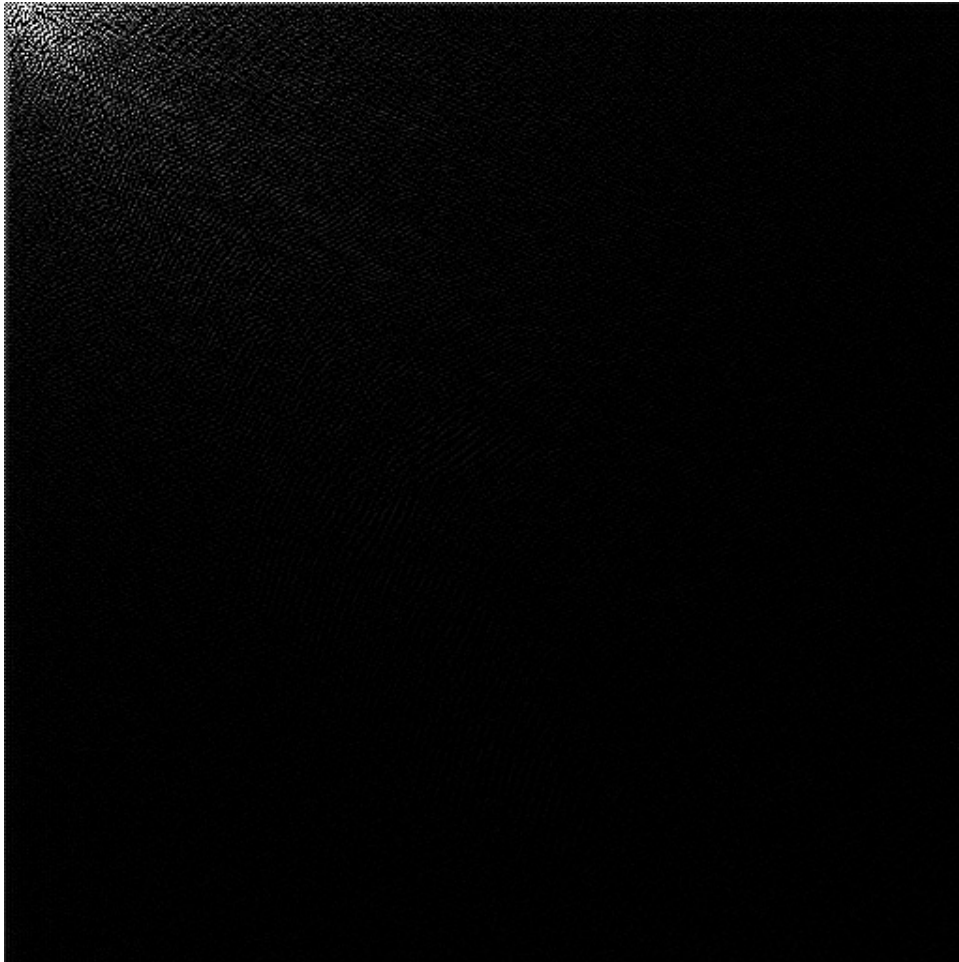


Figure 14.3: Perform the Following Transform on Gray Scale Image



Figure 14.4: Perform the Following Transform on Gray Scale Image

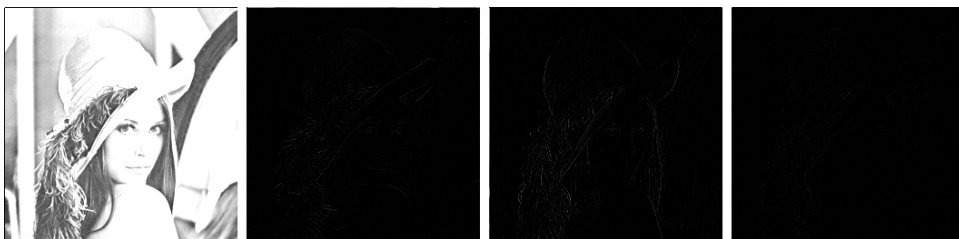


Figure 14.5: Perform the Following Transform on Gray Scale Image



Figure 14.6: Perform the Following Transform on Gray Scale Image

# Experiment: 15

## Edge Detection

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

LAB15\_1.jpg

**Scilab code Solution 15.1** Perform the Various Edge Detection Methods on Gray Scale Image

```
1 // LAB:15 Perform the Various Edge Detection
   Methods on Gray Scale Image.
2 // Version : Scilab 5.4.1
3 // Operating System : Window-xp, Window-7
4 //Toolbox: Image Processing Design 8.3.1-1
5 //Toolbox: SIVP 0.5.3.1-2
6
7 clc;
8 clear all;
9 xdel(winsid());
10 a=imread('LAB15_1.jpg');
11 a=rgb2gray(a);
12 //figure,ShowImage(a,'horizontal');
13 //title('Original Image','color','red','fontsize',
   4);
14 imwrite(a,'LAB15_1.jpg');
```

```

15 im=im2double(a);
16 f1=[-1 -2 -1;0 0 0;1 2 1] //Horizontal Edge
    Detection Mask
17 b=imfilter(im,f1); // 2D Convolution between Image
    and Filter Mask
18 //figure,ShowImage(b,'horizontal');
19 //title('Horizontal Edge Detected Image','color','
    red','fontsize',4);
20 imwrite(b,'LAB15_2.jpg');
21
22
23 f2=[-1 0 1;-2 0 2;-1 0 1] //Vertical Edge Detection
    Mask
24 c=imfilter(im,f2); // 2D Convolution between Image
    and Filter Mask
25 //figure,ShowImage(c,'vertical');
26 //title('Vertical Edge Detected Image','color','red
    ','fontsize',4);
27 imwrite(c,'LAB15_3.jpg');
28
29 f3=[0 -1 -2;1 0 -1;2 1 0] //+45 Diagonal Edge
    Detection Mask
30 d=imfilter(im,f3); // 2D Convolution between Image
    and Filter Mask
31 //figure,ShowImage(d,'+45 degree');
32 //title('+45 Diagonal Edge Detected Image','color','
    red','fontsize',4);
33 imwrite(d,'LAB15_4.jpg');
34
35 f4=[-2 -1 0;-1 0 1;0 1 2] //-45 Diagonal Edge
    Detection Mask
36 e=imfilter(im,f4); // 2D Convolution between Image
    and Filter Mask
37 //figure,ShowImage(e,'-45 degree');
38 //title('-45 Diagonal Edge Detected Image','color','
    red','fontsize',4);
39 imwrite(e,'LAB15_5.jpg');
40

```

```
41 f=edge(a, 'canny',0.5); //Canny Edge Detection Method
    for Edge Detection
42 //figure,ShowImage(f, 'Edge Detected Image');
43 //title('Canny Edge Detected Image', 'color', 'red', '
    fontsize', 4);
44 imwrite(f, 'LAB15_6.jpg');
```

---

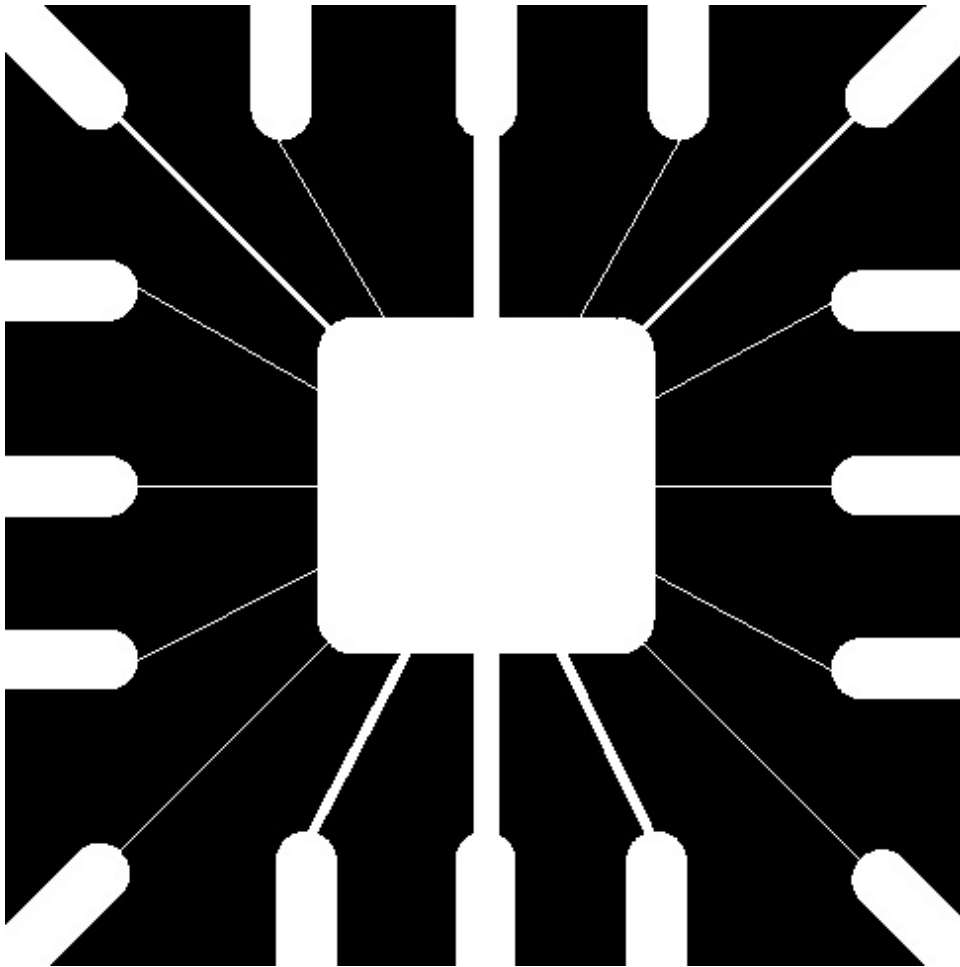


Figure 15.1: Perform the Various Edge Detection Methods on Gray Scale Image

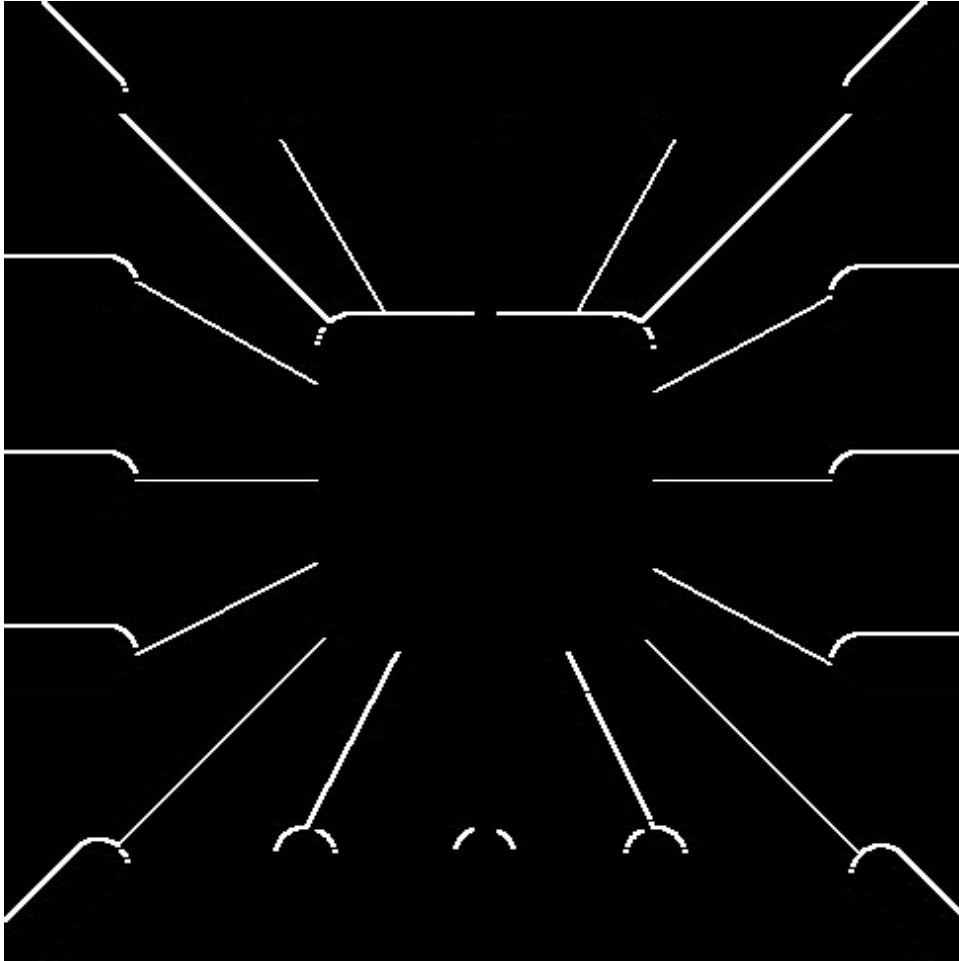


Figure 15.2: Perform the Various Edge Detection Methods on Gray Scale Image

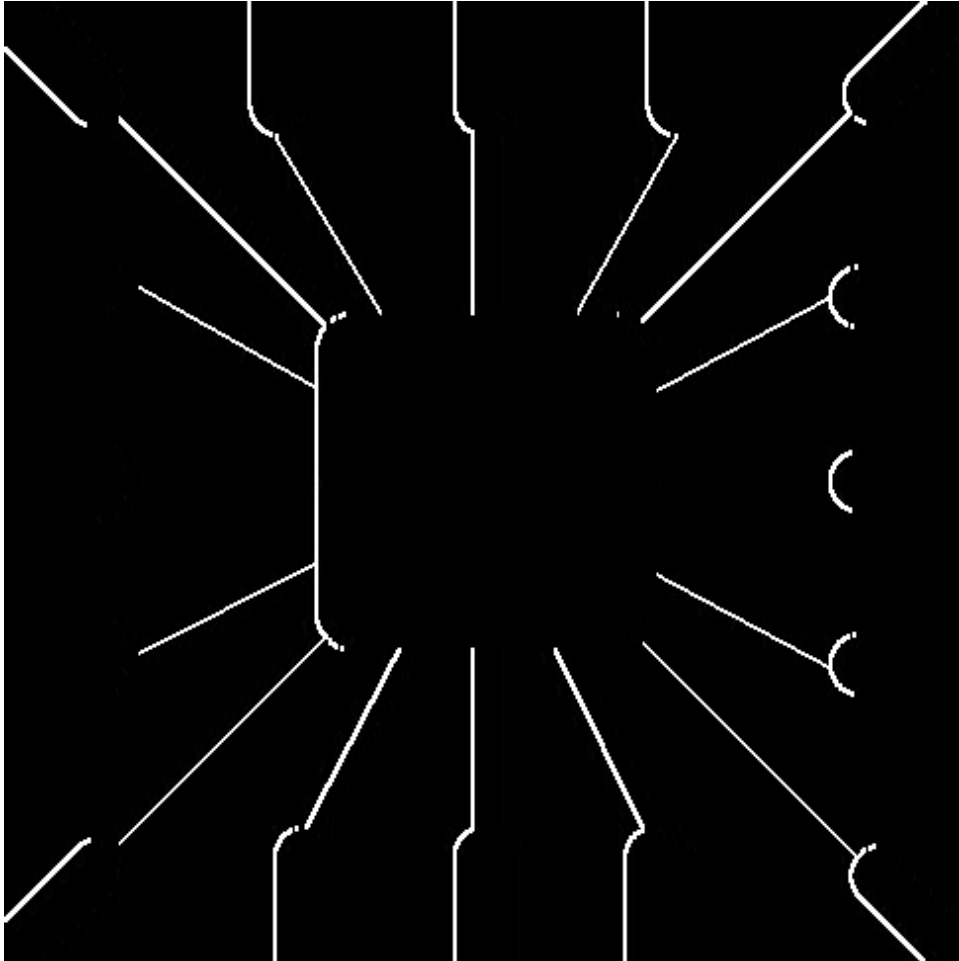


Figure 15.3: Perform the Various Edge Detection Methods on Gray Scale Image

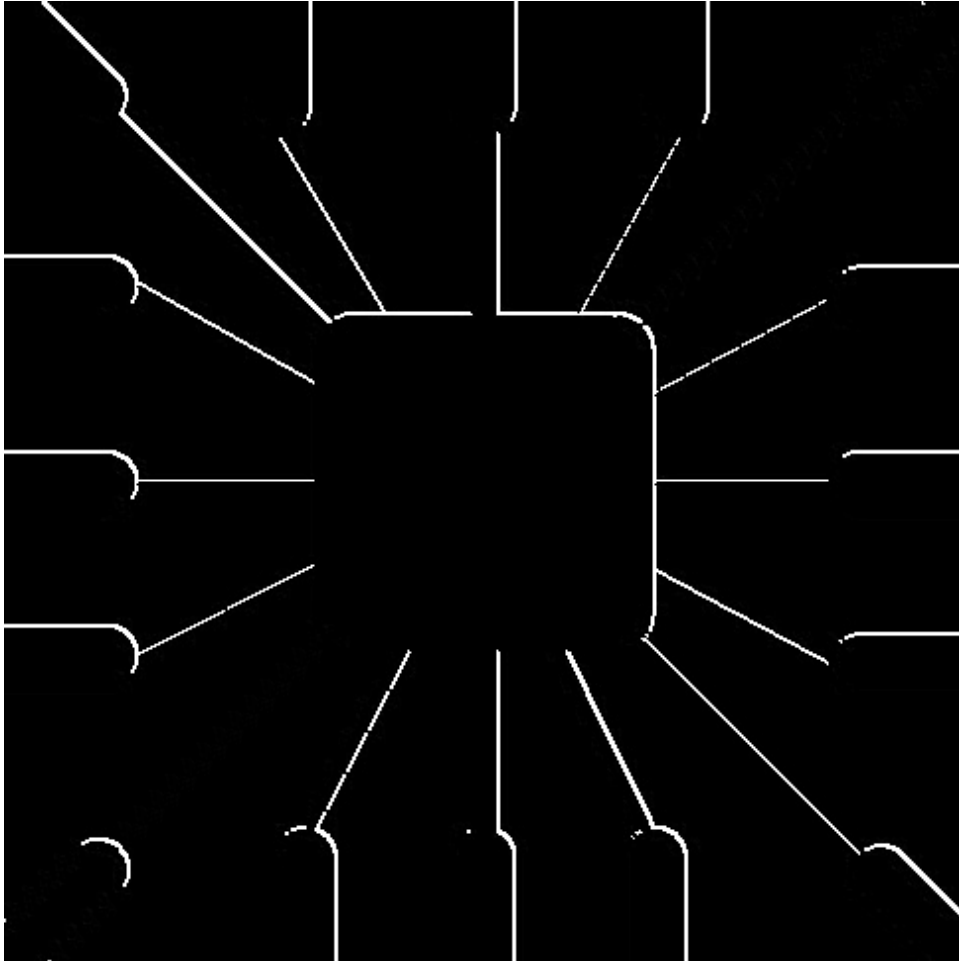


Figure 15.4: Perform the Various Edge Detection Methods on Gray Scale Image

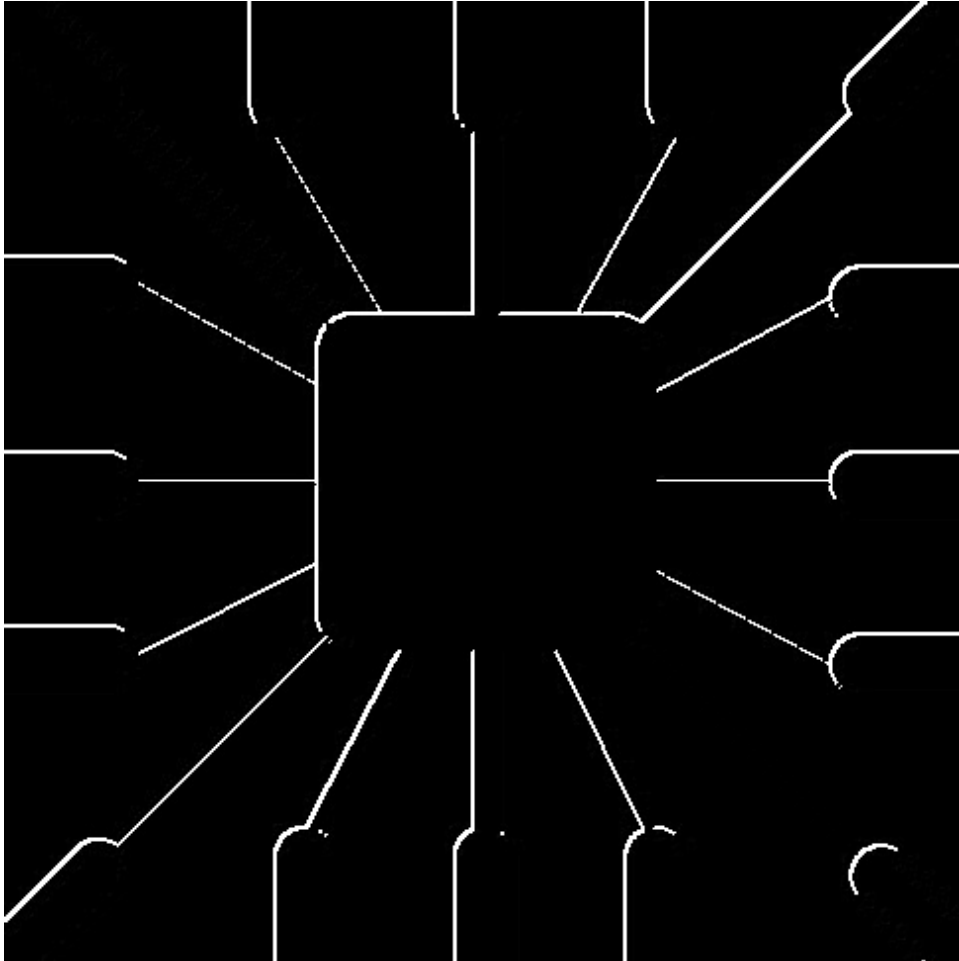


Figure 15.5: Perform the Various Edge Detection Methods on Gray Scale Image

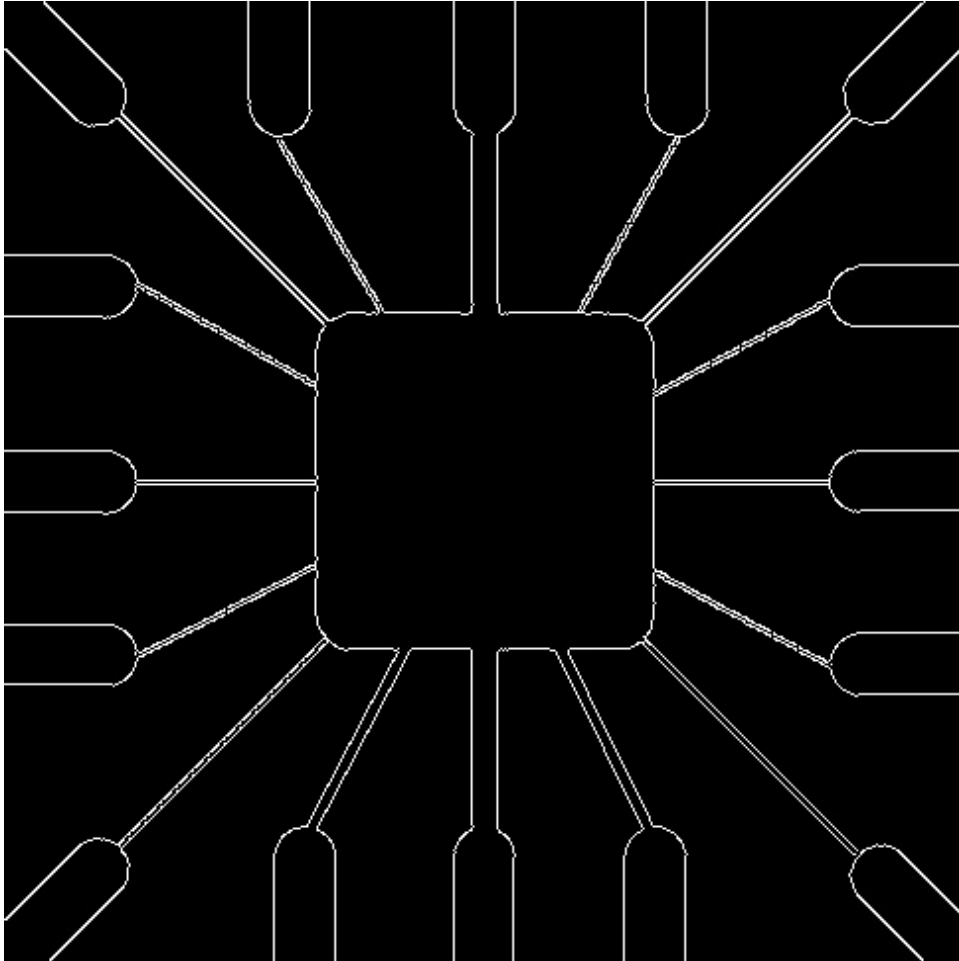
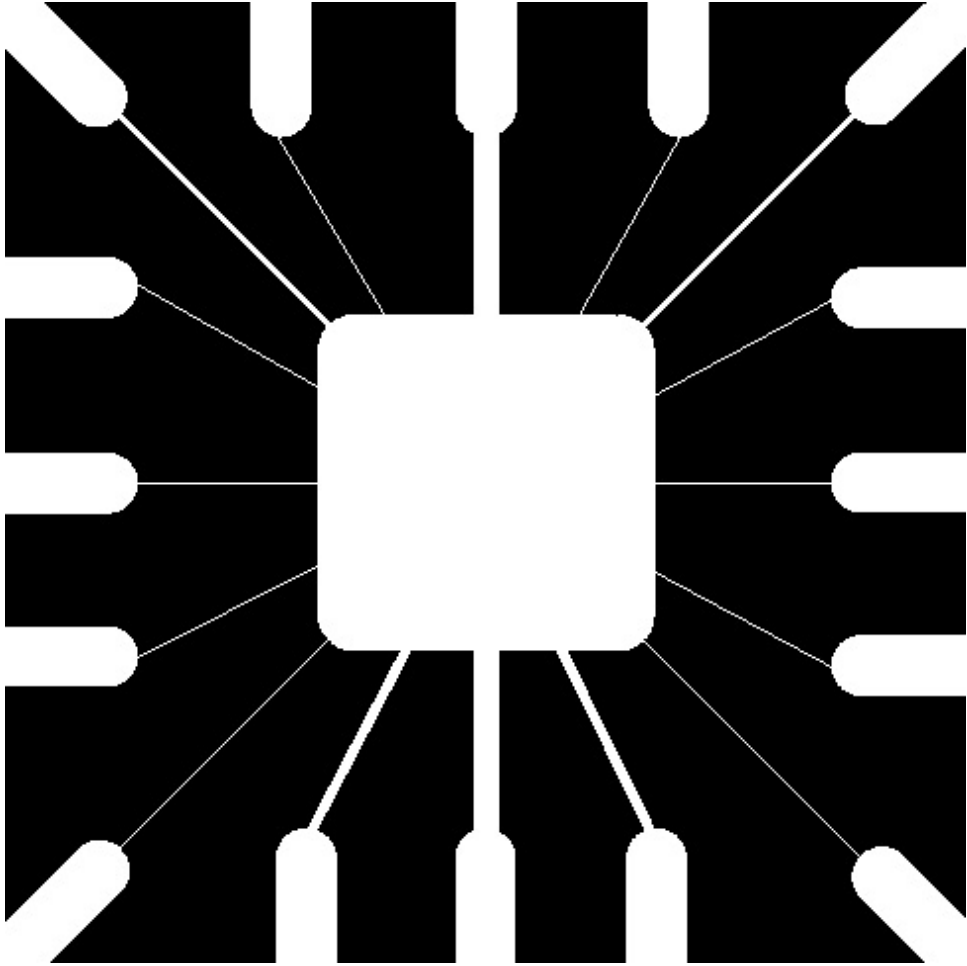


Figure 15.6: Perform the Various Edge Detection Methods on Gray Scale Image

# Appendix



Per-  
form the Various Edge Detection Methods on Gray Scale Image