Scilab Manual for Communication and Signal Processing

by Dr A. Rajeswari
Electronics Engineering
Coimbatore Institute of Tehnology[1]

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

May 18, 2024

[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 Experiments

List of Figures

Experiment: 1

Analog Modulation and Demodulation Schemes

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

```
// LAB:1 – Perform the followinng Analog Modulation Scheme on Binary Data
//          (I)Amplitude Modulation (II)Frequency Modulation (III)Phase Modulation
// Version : Scilab 5.4.1
// Operating System : Window–xp, Window–7

clc;
clear;
xdel(winsid());
f_signal=1;//Signal Frequencies
f_carrier=8;//Carrier Frequencies

t=0:0.001:5;

//////////////// Amplitude Modulation /////////////////////////
information_signal=2*sin(2*%pi*f_signal*t); //
```

```
Information Signal
carrier_signal=15*sin(2*%pi*f_carrier*t);// Carrier Signal
Modulated_signal=(5+information_signal(:,:)).*sin(2*%pi*f_carrier*t);
subplot(3,1,1);plot(information_signal,'LineWidth',1.5);//plot of Information Signal
xgrid;
title('Information Signal','color','Red','fontsize',3);//title the Graph
subplot(3,1,2);plot(carrier_signal,'LineWidth',1.5);//plot of Carrier signal
xgrid;
title('Carrier Signal','color','Red','fontsize',3);//title of plot
subplot(3,1,3);plot(Modulated_signal,'LineWidth',1.5);//plot of modulated signal
xgrid;
title('Amplitude Modulation Signal','color','Red','fontsize',3);//title of plot

//////////////// Frequency Modulation ////////////////////////
Deviation_facotr=5.5
information_signal=2*sin(2*%pi*f_signal*t); // Information Signal
carrier_signal=15*sin(2*%pi*f_carrier*t);// Carrier Signal
Modulated_signal=15.*cos((2*%pi*f_carrier*t)+(Deviation_facotr.*sin(2*%pi*f_signal*t)));
figure;
subplot(3,1,1);plot(information_signal,'LineWidth',1.5);//plot of Information Signal
xgrid;
title('Information Signal','color','Red','fontsize',3);//title the Graph
subplot(3,1,2);plot(carrier_signal,'LineWidth',1.5);//plot of Carrier signal
```

```
xgrid;
title('Carrier Signal','color','Red','fontsize',3);//title of plot
subplot(3,1,3);plot(Modulated_signal,'LineWidth',1.5);//plot of modulated signal
xgrid;
title('Frequency Modulation Signal','color','Red','fontsize',3);//title of plot


//////////////// Phase Modulation ////////////////////////
Phase_deviation_facotr=35
information_signal=2*sin(2*%pi*f_signal*t); // Information Signal
carrier_signal=15*sin(2*%pi*f_carrier*t);// Carrier Signal
Modulated_signal=15.*cos((2*%pi*f_carrier*t)+(Phase_deviation_facotr.*sin(2*%pi*f_signal*t)));
figure;
subplot(3,1,1);plot(information_signal,'LineWidth',1.5);//plot of Information Signal
xgrid;
title('Information Signal','color','Red','fontsize',3);//title the Graph
subplot(3,1,2);plot(carrier_signal,'LineWidth',1.5);//plot of Carrier signal
xgrid;
title('Carrier Signal','color','Red','fontsize',3);//title of plot
subplot(3,1,3);plot(Modulated_signal,'LineWidth',1.5);//plot of modulated signal
xgrid;
title('Phase Modulation Signal','color','Red','fontsize',3);//title of plot
```

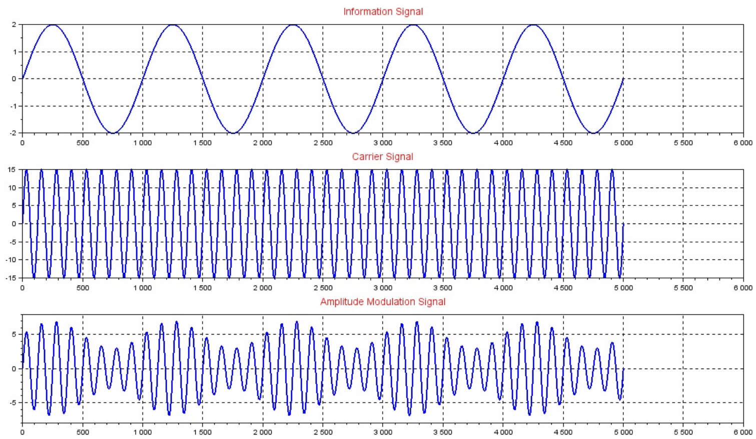


Figure 1.1: Perform the followinng Analog Modulation Scheme on Binary Data

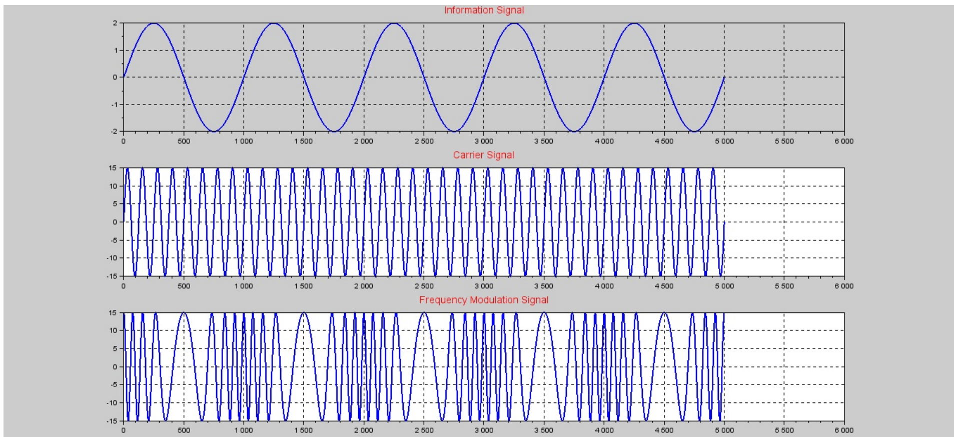


Figure 1.2: Perform the followinng Analog Modulation Scheme on Binary Data

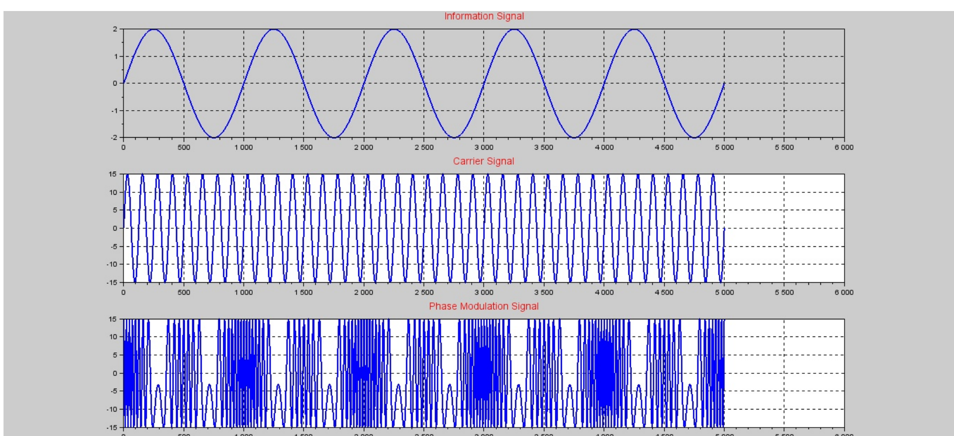


Figure 1.3: Perform the followinng Analog Modulation Scheme on Binary Data

Experiment: 2

Digital Modulation and Demodulation Schemes

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

```
// LAB:2 - Perform the followinng Digital Modulation Scheme on Binary Data
//          (I)Amplitude Shift Keying (II)Frequency Shift Keying (III)Phase Shift keying

// Version : Scilab 5.4.1
// Operating System : Window-xp, Window-7

clc;
clear;
xdel(winsid());
g=[1 0 1 0 0 1 1 0 1 0 ]//binary data
f1=4;f2=8;//Carrier Frequencies
t=0:2*%pi/99:2*%pi;// Time
//ASK
cp=[];bit=[];mod_ask=[];mod_fsk=[];mod_psk=[];cp1=[];cp2=[];
for n=1:length(g);//ASK modulation
```

```
    if g(n)==0;
        die=zeros(1,100);
    else g(n)==1;
        die=ones(1,100);
    end
    c_ask=sin(f1*t); // Signal with Frequency f1
    cp=[cp die];
    mod_ask=[mod_ask c_ask];
    end
ask=cp.*mod_ask;//ASK modulated signal

//////////// Frequency Shift Keying ////////////
for n=1:length(g);
    if g(n)==0;
        die=ones(1,100);
        c_fsk=sin(f1*t); // Signal with Frequency f1
    else g(n)==1;
        die=ones(1,100);
        c_fsk=sin(f2*t); // Signal with Frequency f2
    end
    cp1=[cp1 die];
    mod_fsk=[mod_fsk c_fsk];
end
fsk=cp1.*mod_fsk;//FSK molated signal

//PSK
for n=1:length(g);
    if g(n)==0;
        die=ones(1,100);
        c_psk=sin(f1*t);
    else g(n)==1;
        die=ones(1,100);
        c_psk=-sin(f1*t);
    end
    cp2=[cp2 die];
    mod_psk=[mod_psk c_psk];
end
psk=cp2.*mod_psk;//PSK modulated signal
```

```
subplot(4,1,1);plot(cp,'LineWidth',1.5);//plot binary signal
xgrid;
title('Binary Signal');//title the Graph
mtlb_axis([0 100*length(g) -2.5 2.5]); //axis range
subplot(4,1,2);plot(ask,'LineWidth',1.5);//plot of ASK modulated signal
xgrid;
title('ASK modulation');//title of plot
mtlb_axis([0 100*length(g) -2.5 2.5]);//axis range
subplot(4,1,3);plot(fsk,'LineWidth',1.5);//plot of FSK modulated signal
xgrid;
title('FSK modulation');//title of plot
mtlb_axis([0 100*length(g) -2.5 2.5]);//axis range
subplot(4,1,4);plot(psk,'LineWidth',1.5);//plot of PSK modulated signal
xgrid;
title('PSK modulation');//title of plot
mtlb_axis([0 100*length(g) -2.5 2.5]);//range of axis
```

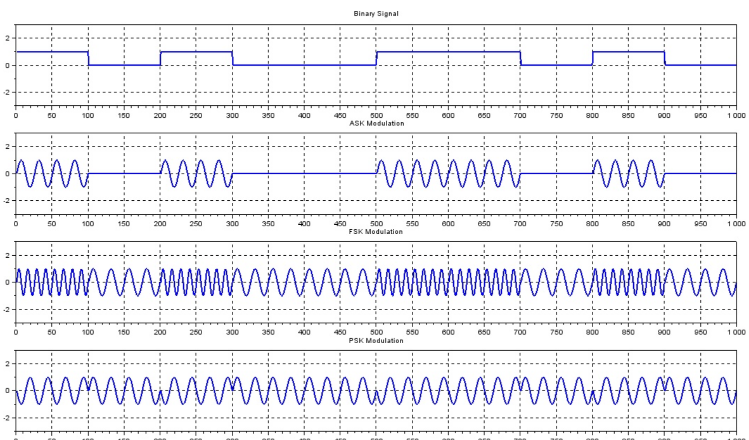


Figure 2.1: Perform the followinng Digital Modulation Scheme on Binary Data

Experiment: 3

Performance analysis of Digital Communication System

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

```
// LAB:3 — Measure the Performance of Communnication
    System in term of Bit Error Rate(BER).
// Version : Scilab 5.4.1
// Operating System : Window—xp, Window—7

clc;
clear;
close;

temp=[];
temp2=[];
for snr=0:0.5:8

    x=rand(1,100000,"uniform"); // Random Data
       Generation with uniform distribution
    y=ones(1,100000);
    out=ones(1,100000);
    [r0 c0]=find(x<0.5);
```

```
    [r1 c1]=find(x>=0.5);
    y(r0,c0)=-1;

    var=(1/(10^(snr/10)))/2;
    noise=var*rand(1,100000,"normal");
    sig_noise=y+noise; //Noise added to Signal
    r0=[];
    c0=[];
    [r0,c0]=find(sig_noise<0.2228154);  //0.2228154
    out(r0,c0)=-1;
    total_error=0;
    for i=1:100000
        if(y(i)~=out(i))
        total_error=total_error+1;
        end
end
q=erfc(sqrt(2/var));

temp=[temp (total_error/100000)];
temp2=[temp2 q];

end

figure;
plot(0:0.5:8,temp,0:0.5:8,temp2,'r');
xtitle("SNR Vs BER","SNR","BER");
legend(['BER Practical';'BER Theoryticai']);
```

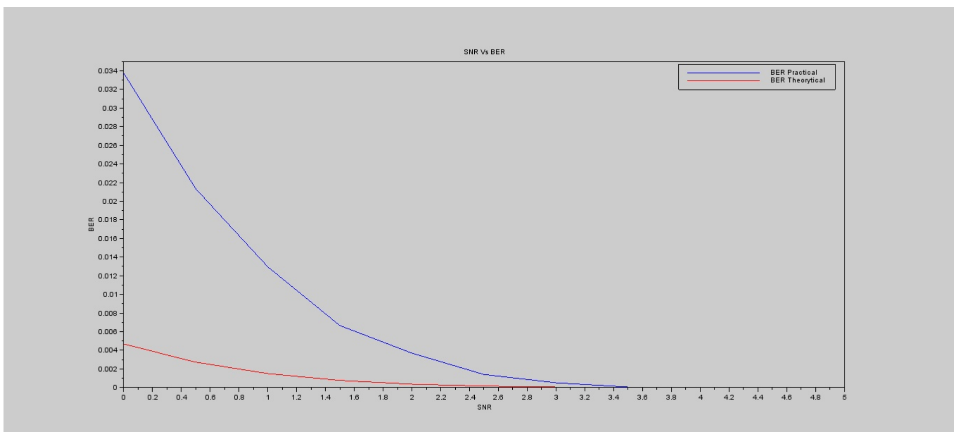


Figure 3.1: Measure the Performance of Communnication System in term of Bit Error Rate

Experiment: 4

Generation of Waveforms

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

```
// LAB:4 To generate basic discrete signal used in
   Digital Signal Processing

// Version : Scilab 5.4.1
// Operating System : Window-xp, Window-7

clc;
clear;
xdel(winsid());
t=0:0.1:20;
f=0.2;
pi=3.14;

////////////////////////      SINEWAVE
   /////////////////
x1=sin(2*pi*f*t);
//scf();
subplot(231);
plot2d3(t,x1);
title('Sinewave','color','red','fontsize', 2);
```

```
xlabel("Index", "fontsize", 2,"color", "blue");
ylabel("Amplitude", "fontsize", 2, "color", "blue");

////////////////////////    Cosine Wave    /////////////////
x2=cos(2*pi*f*t);
//scf();
subplot(232);
plot2d3(t,x2);
title('Cosinewave','color','red','fontsize', 2);
xlabel("Index", "fontsize", 2,"color", "blue");
ylabel("Amplitude", "fontsize", 2, "color", "blue");


////////////////////////    Impulse Wave    /////////////////
t1=-10:10;
x3=[zeros(1,10) 1 zeros(1,10)];
//scf();
subplot(233);
plot2d3(t1,x3);
title('Impulse','color','red','fontsize', 2);
xlabel("Index", "fontsize", 2,"color", "blue");
ylabel("Amplitude", "fontsize", 2, "color", "blue");


////////////////////////    Ramp Wave    /////////////////
t4=0:10;
x4=t4;
//scf();
subplot(234);
plot2d3(t4,x4);
title('Ramp Wave','color','red','fontsize',2);
xlabel("Index", "fontsize", 2,"color", "blue");
ylabel("Amplitude", "fontsize", 2, "color", "blue");

////////////////////////    Exponetial Wave
```

```
/////////////////
t5=0:10;
x5=exp(t5);
//scf();
subplot(235);
plot2d3(t5,x5);
title('Exponetial Wave','color','red','fontsize',2);
xlabel("Index", "fontsize", 2,"color", "blue");
ylabel("Amplitude", "fontsize", 2, "color", "blue");


////////////////////////    Random Wave /////////////////

x6=rand(1,100);
//scf();
subplot(236);
plot2d3(1:length(x6),x6);
title('Random Wave','color','red','fontsize',2);
xlabel("Index", "fontsize", 2,"color", "blue");
ylabel("Amplitude", "fontsize", 2, "color", "blue");



/////////////Impulse Sequence  /////////////////
n1=1,n0=50,n2=100;
if((n0<n1)|(n0>n2)|(n1>n2))
    error('arugument incorrect');
end
n=[n1:n2];
x7=[(n-n0)==0,1];
scf()
subplot(221);
plot2d3(n,x7(n1:n2));
title('Impulse Sequence','color','red','fontsize',2);
xlabel("Index", "fontsize", 2,"color", "blue");
ylabel("Amplitude", "fontsize", 2, "color", "blue");
```

```


////////////////// Step Sequence  ////////////////
n1=1,n0=50,n2=100;
if((n0<n1)|(n0>n2)|(n1>n2))
    error('arugument incorrect');
end
n=[n1:n2];
x8=[(n-n0)>=0,1];
subplot(222);
plot2d3(n,x8(n1:n2));
title('Step Sequence','color','red','fontsize', 2);
xlabel("Index", "fontsize", 2,"color", "blue");
ylabel("Amplitude", "fontsize", 2, "color", "blue");


////////////////// RECTANGULAR FUNCTION ////////////////
t=-5:0.1:5
y=[zeros(1,45) ones(1,11) zeros(1,45)]
subplot(223)
plot2d3(t,y)
title('Rectangular Function','color','red','fontsize',2);
xlabel("Index", "fontsize", 2,"color", "blue");
ylabel("Amplitude", "fontsize", 2, "color", "blue");

////////////////// TRIANGULAR FUNCTION ////////////////
t=0:1:10
y=t;
z=11-t;
b=[y z y z y z];
subplot(224)
plot2d3(b)
title('Triangular Function','color','red','fontsize',2);
xlabel("Index", "fontsize", 2,"color", "blue");
```

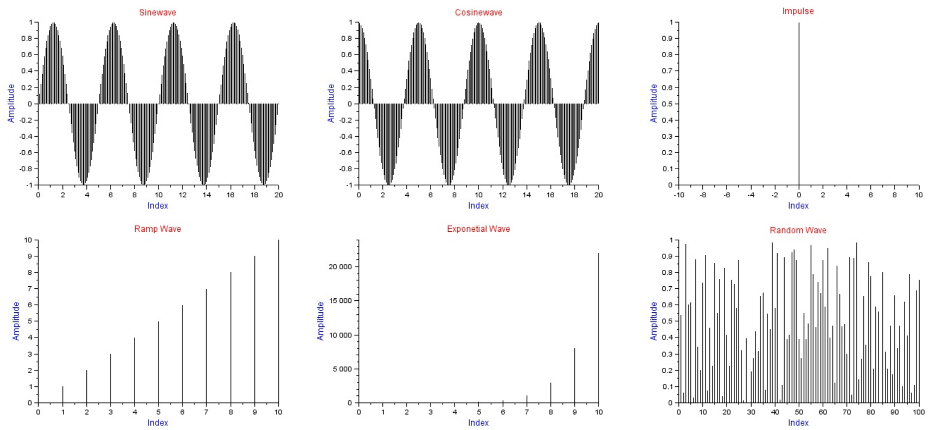


Figure 4.1: To generate basic discrete signal used in Digital Signal Processing

```
ylabel("Amplitude", "fontsize", 2, "color", "blue");


////////////////// SINC FUNCTION  ////////////////
x=linspace(-10,10,3000);
figure;
plot2d3(x,sinc(x))
title('SINC Function','color','red','fontsize',2);
xlabel("Index", "fontsize", 2,"color", "blue");
ylabel("Amplitude", "fontsize", 2, "color", "blue");
```

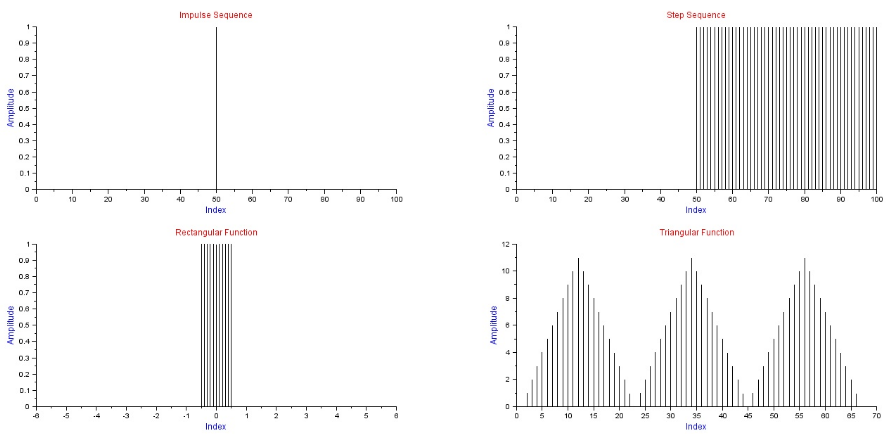


Figure 4.2: To generate basic discrete signal used in Digital Signal Processing

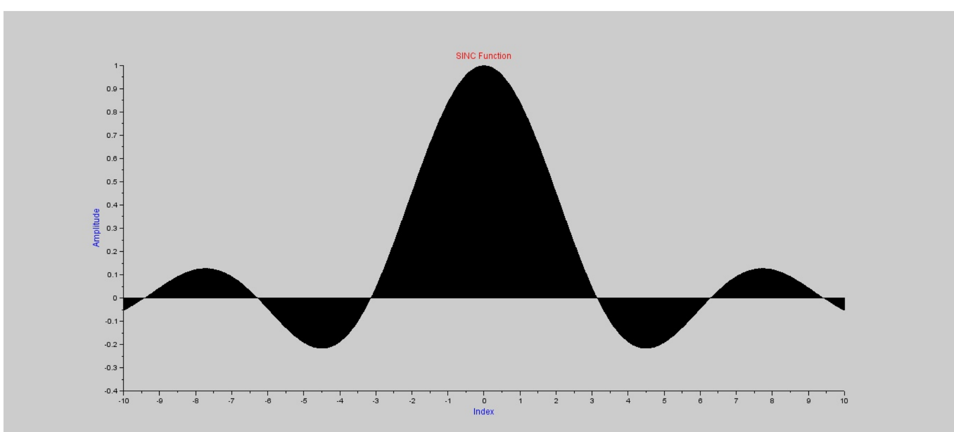


Figure 4.3: To generate basic discrete signal used in Digital Signal Processing

Experiment: 5

Properties of Signals and Systems

Scilab code Solution 5.1 Property of Signal and System

```
// LAB:5 − Property of Signal and System.
// Version : Scilab 5.4.1
// Operating System : Window−xp, Window−7

clc;
close;
clear;
xdel(winsid());
//////////// LINEAR PROPERTY //////////////////
x1=[0 0 ones(1,10)]
x2=[ones(1,12)]
t=0:1:11
y1=t.*x1
y2=t.*x2
x3=[2*x1+3*x2].*t
y3=2*y1+3*y2
figure;
subplot(4,1,1)
plot(t,x1)
```

```
title('Signal 1','color','red','fontsize',2,'
    position',[0.3 0.8]);
xlabel("Index", "fontsize", 2,"color", "blue");
ylabel("Amplitude", "fontsize", 2, "color", "blue");
subplot(4,1,2)
plot(t,x2)
title('Signal 2','color','red','fontsize',2,'
    position',[0.3 1.5]);
xlabel("Index", "fontsize", 2,"color", "blue");
ylabel("Amplitude", "fontsize", 2, "color", "blue");
subplot(4,1,3)
plot(t,y1)
title('Y1=t*Signal 1','color','red','fontsize',2,'
    position',[0.3 7]);
xlabel("Index", "fontsize", 2,"color", "blue");
ylabel("Amplitude", "fontsize", 2, "color", "blue");
subplot(4,1,4)
plot(t,y2)
title('Y2=t*Signal 2','color','red','fontsize',2,'
    position',[0.3 7]);
xlabel("Index", "fontsize", 2,"color", "blue");
ylabel("Amplitude", "fontsize", 2, "color", "blue");
figure
subplot(2,1,1)
plot(t,x3)
title('x3=[2*Signal1+3*Signal2].*t','color','red','
    fontsize',2);
xlabel("Index", "fontsize", 2,"color", "blue");
ylabel("Amplitude", "fontsize", 2, "color", "blue");
subplot(2,1,2)
plot(t,y3)
title('y3=2*y1+3*y2','color','red','fontsize',2);
xlabel("Index", "fontsize", 2,"color", "blue");
ylabel("Amplitude", "fontsize", 2, "color", "blue");

/////////////////////////// TIME VARIANT
    ////////////////////
x1=[0,0,ones(1,10)]
```

```
t=0:1:11
y1=t.*x1
t2=t+5
figure
subplot(2,1,1)
plot(t2,x1)
title('Time Variant Property','color','red','fontsize',2);
xlabel("Index", "fontsize", 2,"color", "blue");
ylabel("Amplitude", "fontsize", 2, "color", "blue");
subplot(2,1,2)
plot(t2,y1)
xlabel("Index", "fontsize", 2,"color", "blue");
ylabel("Amplitude", "fontsize", 2, "color", "blue");


/////////////////// CAUSAL & NON CAUSAL System //////////////
x=[1,4,2,8,0,4,3]
t=0:1:6
t1=t*2
t2=t/2;
figure;
subplot(3,1,1)
plot(t,x)
title('CAUSAL & NON CAUSAL Property','color','red','fontsize',2);
xlabel("Index", "fontsize", 2,"color", "blue");
ylabel("Amplitude", "fontsize", 2, "color", "blue");
subplot(3,1,2)
plot(t1,x)
title('CAUSAL Property','color','red','fontsize',2,'position',[0.3 6]);
xlabel("Index", "fontsize", 2,"color", "blue");
ylabel("Amplitude", "fontsize", 2, "color", "blue");
subplot(3,1,3)
plot(t2,x)
title('NON CAUSAL Property','color','red','fontsize'
```

```
    ,2,'position',[0.3 6]);
xlabel("Index", "fontsize", 2,"color", "blue");
ylabel("Amplitude", "fontsize", 2, "color", "blue");


/////////////////// STATIC & DYNAMIC
    ////////////////////////
x1=[4,6,3,2,9,5]
t=0:1:5;
y1=x1.^2;
t1=t-5;
figure;
subplot(3,1,1)
plot(t,x1)
title('STATIC & DYNAMIC Property','color','red','
    fontsize',2);
xlabel("Index", "fontsize", 2,"color", "blue");
ylabel("Amplitude", "fontsize", 2, "color", "blue");
subplot(3,1,2)
plot(t,y1)
title('STATIC Property','color','red','fontsize',2,'
    position',[0.3 60]);
xlabel("Index", "fontsize", 2,"color", "blue");
ylabel("Amplitude", "fontsize", 2, "color", "blue");
subplot(3,1,3)
plot(t1,y1)
title('DYNAMIC Property','color','red','fontsize',2,
    'position',[-4.8 60]);
xlabel("Index", "fontsize", 2,"color", "blue");
ylabel("Amplitude", "fontsize", 2, "color", "blue");
```

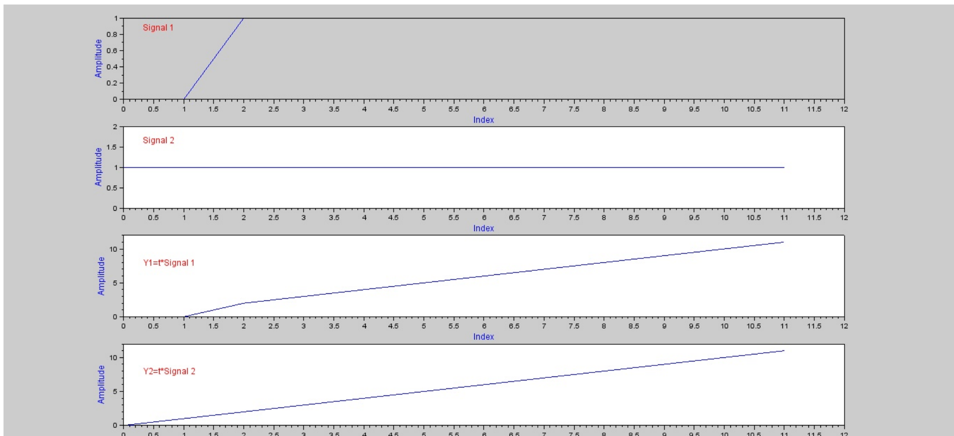


Figure 5.1: Property of Signal and System

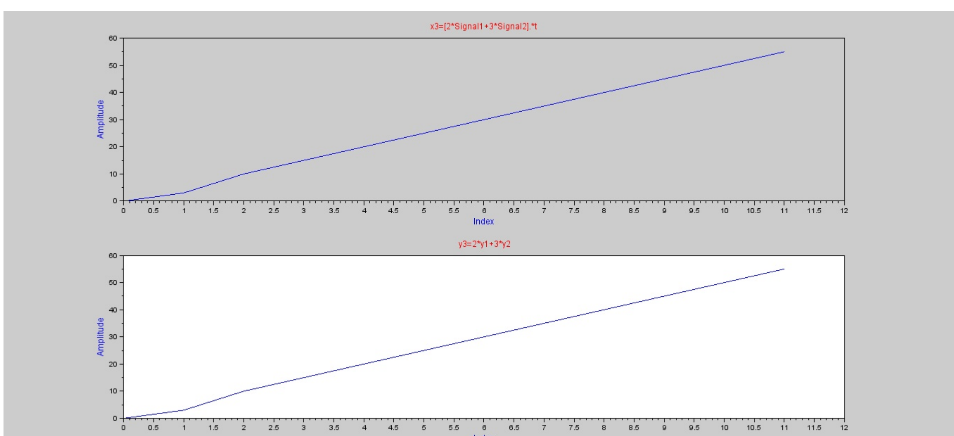


Figure 5.2: Property of Signal and System

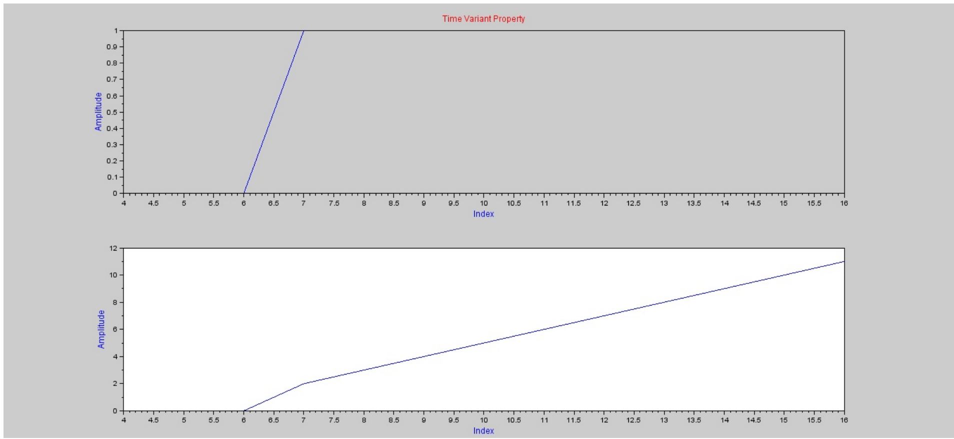


Figure 5.3: Property of Signal and System

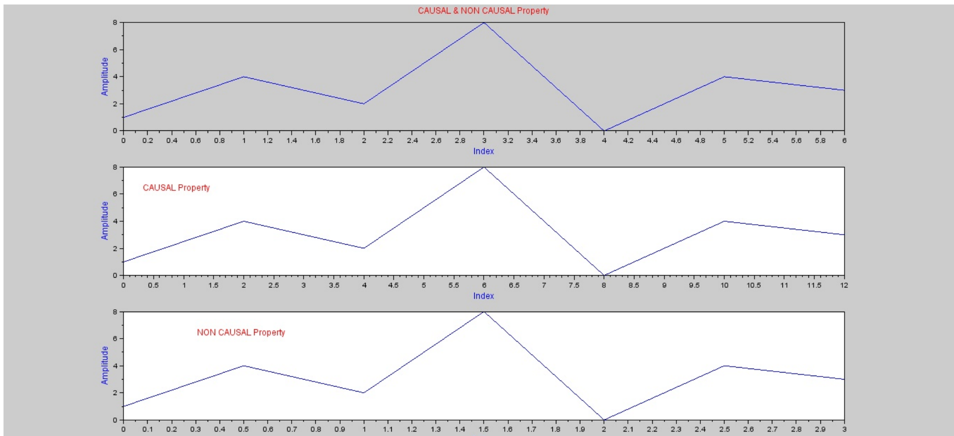


Figure 5.4: Property of Signal and System

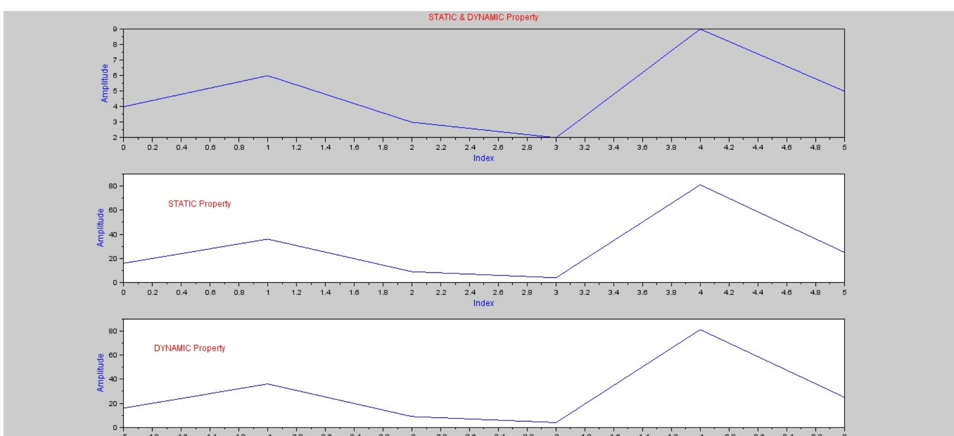


Figure 5.5: Property of Signal and System

Experiment: 6

Convolution

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

```
// LAB:6 Perform Linear and Circular Convolution Operation on Two Discrete Sequences

// Version : Scilab 5.4.1
// Operating System : Window-xp, Window-7


clc;
clear;
xdel(winsid());

/////////////   LINEAR CONVOLUTION ////////////////////
disp('LINEAR CONVOLUTION OPERATION');
x1=input("Enter the Sequence_1 :");    //  [1 2 3 4 5];
x2=input("Enter the Sequence_2 :");    //  [5 4 8];
n = length(x1);
m = length(x2);
for k = 1:(m+n-1)
```

```
    w(k) = 0;
        for j =max(1,k+1-m) : min(k,n)
            w(k)= w(k)+(x1(j)*x2(k+1-j));
        end
end
disp('Convoled Sequence:');
disp(w);
scf();
subplot(3,1,1);
bar(x1,0.05,'red');
title('Sequence_1','color','red','fontsize', 4);
xlabel("Index", "fontsize", 2,"color", "blue",'position',[0.6 0.3]);
ylabel("Amplitude", "fontsize", 2, "color", "blue");

subplot(3,1,2);
bar(x2,0.05,'yellow');
title('Sequence_2','color','red','fontsize', 4);
xlabel("Index", "fontsize", 2,"color", "blue",'position',[0.6 0.3]);
ylabel("Amplitude", "fontsize", 2, "color", "blue");

subplot(3,1,3);
bar(w,0.05,'green');
title('Convoled Sequence','color','green','fontsize', 4);
xlabel("Index", "fontsize", 2,"color", "blue",'position',[0.3 0.3]);
ylabel("Amplitude", "fontsize", 2, "color", "blue");

/////////////   CIRCULAR CONVOLUTION ////////////////////
disp('CIRCULAR CONVOLUTION OPERATION');
x1=input('Enter First Sequence : ');    //  [1 2 3 4 5];
x2=input('Enter Second Sequence : ');    //  [1 2 3 4 5];
l1=length(x1);
```

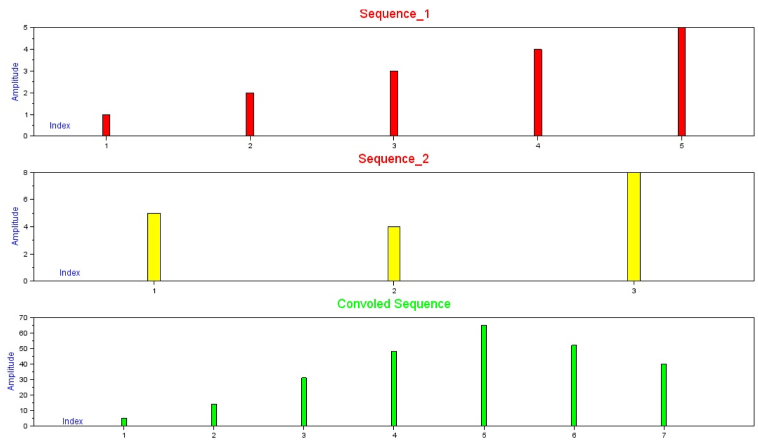


Figure 6.1: Perform Linear and Circular Convolution Operation on Two Discrete Sequences

```
l2=length(x2);
if(l1==l2)
    a=[x1(1) x1(l1:-1:2)];
    b=a;
    for i=2:l1
        a=[a(l1) a(1:l1-1)];
        b=[b;a];
    end
    c=x2';
    d=b*c;
    y=d';
    disp('Circular Convolution Output : ')
    disp(y);
else
    disp('Circular Convolution is not possible.')
end
```

Experiment: 7

Correlation

Scilab code Solution 7.1 Perform Correlation Operation on Two Discrete Sequences

```
// LAB:7 Perform Correlation Operation on Two
   Discrete Sequences

// Version : Scilab 5.4.1
// Operating System : Window-xp, Window-7

clc;
clear;
xdel(winsid());
x=[2 4 5 6];
y=[2 4 5];

m=length(x);
n=length(y);

for k=1:m+n-1
           w(k)=0;
        for j=max(1,k+1-n):min(k,m)
                w(k)=w(k)+(x(j)*y(n-k+j));
        end
```

```
end
disp('Sequence 1:');
disp(x);
disp('Sequence 2:');
disp(y);
disp('Correlation Out Put:');
disp(w);
////////// Graphical Display /////////////////
scf();
subplot(3,1,1);
bar(x,0.05,'red');
title('Sequence_1','color','red','fontsize', 4);
xlabel("Index", "fontsize", 2,"color", "blue",'
    position',[0.6 0.3]);
ylabel("Amplitude", "fontsize", 2, "color", "blue");

subplot(3,1,2);
bar(y,0.05,'yellow');
title('Sequence_2','color','red','fontsize', 4);
xlabel("Index", "fontsize", 2,"color", "blue",'
    position',[0.6 0.3]);
ylabel("Amplitude", "fontsize", 2, "color", "blue");

subplot(3,1,3);
bar(w,0.05,'green');
title('Correlation of Sequences','color','green','
    fontsize', 4);
xlabel("Index", "fontsize", 2,"color", "blue",'
    position',[0.3 0.3]);
ylabel("Amplitude", "fontsize", 2, "color", "blue");
```

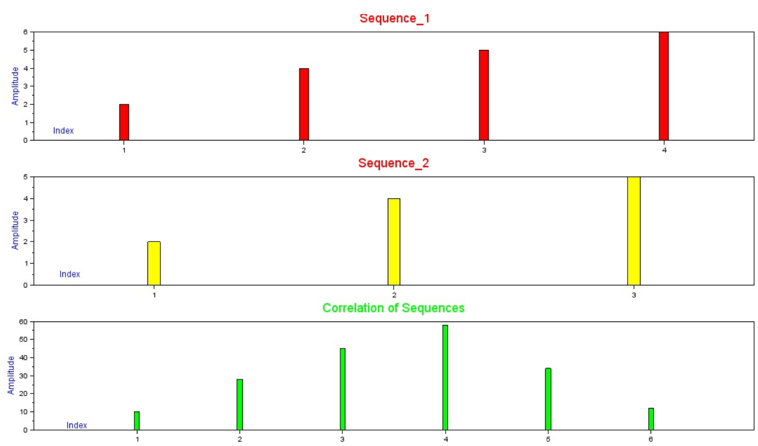


Figure 7.1: Perform Correlation Operation on Two Discrete Sequences

Experiment: 8

Finding DFT and IDFT using FFT

Scilab code Solution 8.1 Perform DFT and IDFT of discrete signal

```
// LAB:8 Perform DFT and IDFT of discrete signal.
// Version : Scilab 5.4.1
// Operating System : Window-xp, Window-7


clc;
close;
clear;
x=[1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0]; // Step Sequence
N=length(x); // Number of Point DFT
for k=1:2*N
    y(k)=0;
    for n=1:N
        y(k)=y(k)+(x(n)*exp((-2*%pi*(k-1)*(n-1)*%i)/N));
    end
end
subplot(311);
```

```
plot2d3(x);
title('Step Sequence','color','Red','fontsize',3);
xlabel('Time Index');
ylabel('Amplitude');

subplot(312);
plot2d3(abs(y));
title('DFT','color','Red','fontsize',3);
xlabel('Frequency Scale');
ylabel('Amplitude');

/////////////// IDFT ////////////////
for n=1:N
p(n)=0;
for  k=1:N
p(n)=p(n)+((y(k)*exp((%i*2*%pi*(k-1)*(n-1))/N))/N);
end
end
subplot(313);
plot2d3(abs(p));
title('IDFT','color','Red','fontsize',3);
xlabel('Time Index');
ylabel('Amplitude');
```

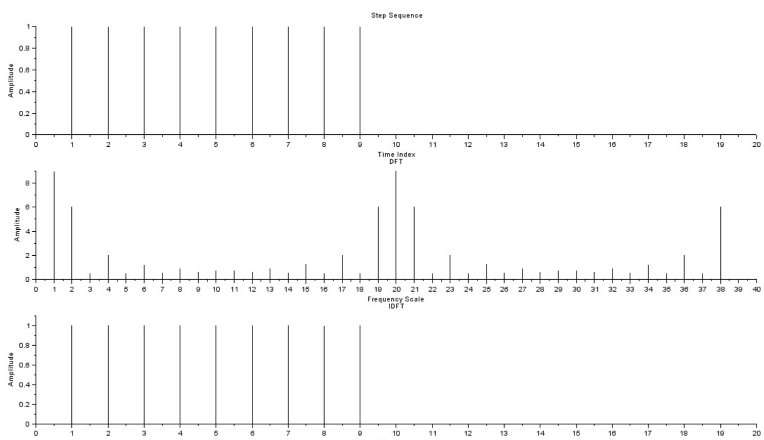


Figure 8.1: Perform DFT and IDFT of discrete signal

Experiment: 9

Design of FIR filters

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

```
// LAB:09  Design the following FIR filters with the given specification.
//

// ///////////////////////////////////////////////////////////////////////
//     Example:
//                  filter type ('lp','hp','sb','bp')
//                  Filter order (pos integer)(odd for ftype='hp' or 'sb')
//                  cfreq=2-vector of cutoff frequencies (0<cfreq(1),cfreq(2)<.5) only cfreq(1) is used when ftype='lp' or 'hp'
//                  wtype= Window type ('re','tr','hm','hn','kr','ch')
//                  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
```

```
//                  wft=time domain filter coefficients
//                  wfm=frequency domain filter response on the grid fr
//                  fr=Frequency grid
// ///////////////////////////////////////////////////////////////

//Evaluate magnitude response of the filter

// Version : Scilab 5.4.1
// Operating System : Window-xp, Window-7

clc;
close;
clear;

ftype='bp';
forder=33;
fs=8000;
cfreq=[(450/fs) (500/fs)];
wtype='kr';
fpar=[0.8 0];


[wft1,wfm1,fr1]=wfir(ftype,forder,cfreq,'re',fpar);
[wft2,wfm2,fr2]=wfir(ftype,forder,cfreq,'hm',fpar);
[wft3,wfm3,fr3]=wfir(ftype,forder,cfreq,'hn',fpar);
[wft4,wfm4,fr4]=wfir(ftype,forder,cfreq,'kr',fpar);
[wft5,wfm5,fr5]=wfir(ftype,forder,cfreq,'tr',fpar);
//[wft6,wfm6,fr6]=wfir(ftype,forder,cfreq,'ch',fpar);

clf();
plot(fr1,wfm1,fr2,wfm2,fr3,wfm3,fr4,wfm4,fr5,wfm5);
legend('rectangal Window','Hamming Window','Hanning Window','Kaiser Window','Triagle Window');
```

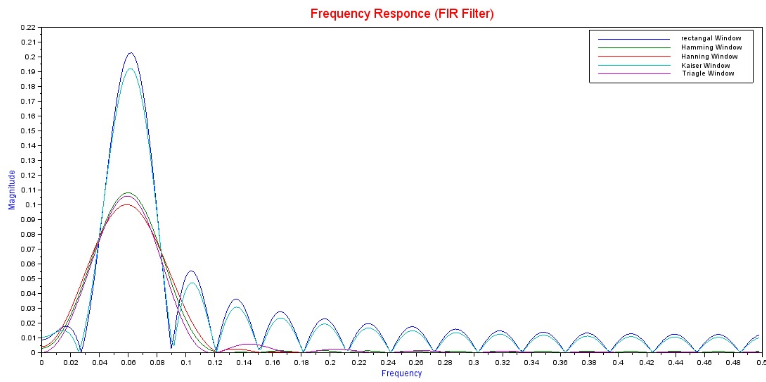


Figure 9.1: Design the following FIR filters with the given specification

```
title('Frequency Responce','color','red','fontsize',
    4);
xlabel("Frequency", "fontsize", 2,"color", "blue");
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

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

// ///////////////////////////////////////////////////////////////
//    Example:
//                  filter type ('lp','hp','sb','bp')
//                  design approximation ('butt','cheb1','cheb2','ellip')
//                  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.
//                  0<= deltap <=1
//                  0<= deltas <=1
// ///////////////////////////////////////////////////////////////
```

```
//Evaluate magnitude response of the filter

// Version : Scilab 5.4.1
// Operating System : Window-xp, Window-7

clc;
close;
clear;

ftype='bp'; // Type of Filter
approx='ellip'; //Design Approximation
om=[.15 .25]; // Cut off Frequency
deltap=0.08;
deltas=0.03;

hz_ellip=iir(3,ftype,approx,om,[deltap deltas]); // Band Pass Filter with Elliptic
[hzm1,fr1]=frmag(hz_ellip,256); //Frequency Magnitude
hz_butt=iir(3,ftype,'butt',om,[deltap deltas]); // Band Pass Filter with Butterworth
[hzm2,fr2]=frmag(hz_butt,256); //Frequency Magnitude
hz_cheby1=iir(3,ftype,'cheb1',om,[deltap deltas]); // Band Pass Filter with Chebysev 1
[hzm3,fr3]=frmag(hz_cheby1,256); //Frequency Magnitude
hz_cheby2=iir(3,ftype,'cheb2',om,[deltap deltas]); // Band Pass Filter with Chebysev 1
[hzm4,fr4]=frmag(hz_cheby2,256); //Frequency Magnitude

plot(fr1',hzm1',fr2',hzm2',fr3',hzm3',fr4',hzm4');
xtitle('Discrete IIR filter band pass  0.15 < fr < 0.25 ',' ',' ');
xlabel('Frequency Scale');
ylabel('Magnitude');
h=legend(['Ellip';'Butter';'Chaby1';'Cheby2']);
```

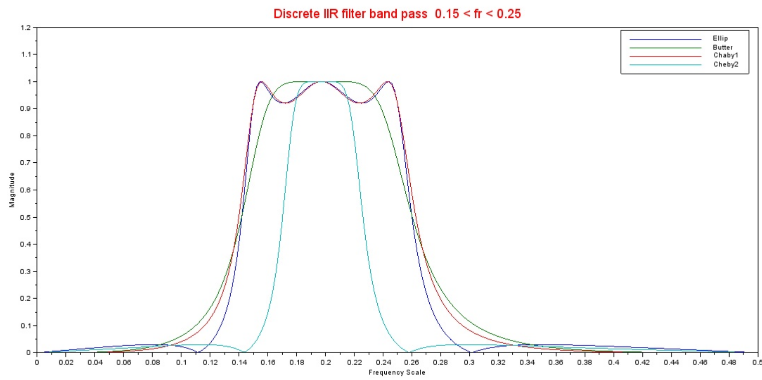


Figure 10.1: Design the following IIR filters with the given specification

Experiment: 11

Error control Codes

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

```
// LAB:11 - Perform the Linear Block Coding on binary Data.
// Version : Scilab 5.4.1
// Operating System : Window-xp, Window-7

clc;
close;
clear;
xdel(winsid());

global P n k;

n=7;
k=4;
P=[1 1 0; 0 1 1; 1 0 1;1 1 1]; //% parity  matrix of size k*(n-k) to be
//                              % selected so that the systematic generator
//                              % matrix is linearly independent or full rank
```

```
//                                    % matrix


//%This is an linear block encoding function file%

function y1=linblkcode(x);
global P n k;
G=[eye(k,k) P];

y1=zeros(1,n);
for i=1:k
    var(i,:)=x(1,i) & G(i,:);
    var(i,:)=bool2s(var(i,:));
    y1(1,:)=bitxor(var(i,:),y1(1,:));
end

endfunction


//%This is a linear block syndrome decoding function file%

function x1=linblkdecoder(y)
//% here y is recieved vector 7 bits long(7,4) linear block code

global P n k;


//H=[ ]; //% PARITY CHECK MATRIX

H=[P' eye((n-k),(n-k))];
Ht=H'; //%transpose of H

S=zeros(1,n-k); //%syndrome of recieved vector x
for i=1:n-k
    S(i)=y(1) & Ht(1,i);
    S(i)=bool2s(S(i));
```

```
    for j=2:n

        S(i)=bitxor(S(i), bool2s((y(j) & Ht(j,i))));
    end
end
//%%****SYNDROME LOOK UP TABLE************
if S==[0 0 0]
   e=[0 0 0 0 0 0 0];
    z=bitxor(y,e);
end

if S==[0 0 1]
   e=[0 0 0 0 0 0 1];
    z=bitxor(y,e);
end
if S==[0 1 0]
   e=[0 0 0 0 0 1 0];
    z=bitxor(y,e);
end
if S==[1 0 0]
   e=[0 0 0 0 1 0 0];
    z=bitxor(y,e);
end
if S==[1 1 1]
   e=[0 0 0 1 0 0 0];
    z=bitxor(y,e);
end
if S==[1 0 1]
   e=[0 0 1 0 0 0 0];
    z=bitxor(y,e);
end
if S==[0 1 1]
   e=[0 1 0 0 0 0 0];
    z=bitxor(y,e);
end
if S==[1 1 0]
   e=[1 0 0 0 0 0 0];
    z=bitxor(y,e);
```

```
end

x1=z(1,1:k);
endfunction

///////////////////// Main Programm ///////////////////////
x=[0 1 0 1];   //                % input bits to the encoder of size 1* k
y1=linblkcode(x);//           //   % y1 is the output of linear block encoder

e1=[1 0 0 0 0 0 0];            // % intentionally error introduced after
//                             % encoding and before sending to decoder (in
  //                           % this case pls introduce only one bit error)
y=bitxor(y1,e1);//             % input that will be made available to linear
                  //           % block decoder

x1=linblkdecoder(y) //         % x1 is the output of the linear block decoder
                      //       % which will be same as x provided that here
                        //     % have introduced only one bit error

disp('The information signal=')
disp(x)
disp('The transmitted encoded signal=')
disp(y1)
disp('The recieved signal=')
disp(y)
disp('The decoded signal=')
disp(x1)
```

```
////////////////////////// Input and Output Values
    /////////////////////////
//The information signal(x)=[0 1 0 1];
//The transmitted encoded signal(y1)=[0 1 0 1 1 0
    0];
//The recieved signal(y)=[1 1 0 1 1 0 0];
//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 Performm the following basic image processing operation on digital image

```
// LAB:12 Performm the following basic image
   processing operation on digital image.
//(I)RGB to Gray (II) Image Information (III) Image
   Resizing (IV) Image Cropping (V) Image Negative (
   VI) Gamma Intensity transformation
// Version : Scilab 5.4.1
// Operating System : Window-xp, Window-7
//Toolbox: Image Processing Design 8.3.1-1
//Toolbox: SIVP 0.5.3.1-2

clc;
close;
clear;
```

```
xdel(winsid())//to close all currently open figure(s).

path=getSIVPpath();
Image=imread(path+"images\lena.png");
//figure,ShowColorImage(Image,'RGB Image');
//title('RGB Image','color','blue','fontsize',4);
imwrite(Image,'LAB12_1.jpg');
Gray_Image=rgb2gray(Image); // Convert RGB Image to Gray Scale Image
//figure,ShowImage(Gray_Image,'Gray Scale Image');
//title('Gray Scale Image','color','blue','fontsize',4);
imwrite(Gray_Image,'LAB12_2.jpg');
//imfinfo(SIVP_Path + 'images\lena.png');// Display Image Information
Crop_Image = imcrop(Gray_Image, [100, 30, 300, 300]); // Crop the Image form the Specific Location
//figure,ShowImage(Crop_Image,'Image Cropping');
//title('Croped Image','color','blue','fontsize',4);
imwrite(Crop_Image,'LAB12_3.jpg');
Resize_Image = imresize(Gray_Image,0.5); //Resize the Image with Factor 0.5
//figure,ShowImage(Resize_Image,'Image Resizing');
//title('Resized Image','color','blue','fontsize',4);
imwrite(Resize_Image,'LAB12_4.jpg');

///////////////// Negative Intensity Transformation ///////
[r p]=size(Gray_Image);
for i=1:r
    for j=1:p
        Negative_Image(i,j)=255-Gray_Image(i,j);
    end
end
//figure,ShowImage(Negative_Image,'Image Negative');
```

```
//title('Nagative Image','color','blue','fontsize
    ',4);
imwrite(Negative_Image,'LAB12_5.jpg');

/////////////// Image Fliping (Left to Right)
    ///////////////
[r p]=size(Gray_Image);
for i=1:r
    for j=1:p
        Fliped_Image(i,j)=Gray_Image(i,p-j+1);
    end
end
//figure,ShowImage(Fliped_Image,'Image Flipng (Left
    to Right)');
//title('Fliped Image (Left to Right)','color','blue
    ','fontsize',4);
imwrite(Fliped_Image,'LAB12_6.jpg');

//////////// Gamma Intensity transformation
    //////////////
gray=imread("gray.tif");
gray=imresize(gray,0.5);
gray1=im2double(gray);
//figure,ShowImage(gray,'Gray Image');
//title('Original Image','color','blue','fontsize
    ',4);
[M,N]=size(gray);
temp=[]
temp=[temp gray1];
c=1;
gamma=[0.6 0.4 0.3];
for i=1:length(gamma)
    b=c.*(gray).^gamma(i); //Gamma transformation
    b=mat2gray(b);
    temp=[temp ones(M,20) b]; // Padding Data for
        Displaying the Image
end
//figure,ShowImage(temp,'Gray Image');
```

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

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

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

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

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

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

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

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

Experiment: 13

2D Convolution

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

```
// LAB:13 Performm the 2D Convolution on digital image (Filtering Operation).
// Version : Scilab 5.4.1
// Operating System : Window-xp, Window-7
//Toolbox: Image Processing Design 8.3.1-1
//Toolbox: SIVP 0.5.3.1-2

clc;
clear;
xdel(winsid());
path=getSIVPpath();
Image=imread(path+"images\lena.png");
gray=rgb2gray(Image);
//figure;
//ShowImage(gray,"gray image");
//title("grayscale",'fontsize',3 );
imwrite(gray,'LAB13_1.jpg');
[row,column]=size(gray);

filter=fspecial('average',3); // Average Filter
imfilt=imfilter(gray,filter); // 2D Convolution
```

```
between Image and Filter
//figure,ShowImage(imfilt,'filtered image'); // Display Gary Scale Image in Window
//title('averaging filter, order = '+string(i)); // Title on Displayed Image
imwrite(imfilt,'LAB13_2.jpg');

filter=fspecial('average',7); // Average Filter
imfilt2=imfilter(gray,filter); // 2D Convolution between Image and Filter
//figure,ShowImage(imfilt,'filtered image'); // Display Gary Scale Image in Window
//title('averaging filter, order = '+string(i)); // Title on Displayed Image
imwrite(imfilt2,'LAB13_3.jpg');

filter=fspecial('average',11); // Average Filter
imfilt3=imfilter(gray,filter); // 2D Convolution between Image and Filter
//figure,ShowImage(imfilt,'filtered image'); // Display Gary Scale Image in Window
//title('averaging filter, order = '+string(i)); // Title on Displayed Image
imwrite(imfilt3,'LAB13_4.jpg');
```

Figure 13.1: Perforrm the 2D Convolution on digital image

Figure 13.2: Performm the 2D Convolution on digital image

Figure 13.3: Performm the 2D Convolution on digital image

Figure 13.4: Performm the 2D Convolution on digital image

Experiment: 14

Image Transforms -DFT,DCT and DWT

Scilab code Solution 14.1 Perforrm the Following Transform on Gray Scale Image

```
// LAB:14 Perforrm the Following Transform on Gray Scale Image.
//(I)DFT (II)DCT (III)DWT
// Version : Scilab 5.4.1
// Operating System : Window-xp, Window-7
//Toolbox: Image Processing Design 8.3.1-1
//Toolbox: SIVP 0.5.3.1-2
//Toolbox: Wavelet


clc;
clear;
xdel(winsid());
path=getSIVPpath();
Image=imread(path+"images\lena.png");
gray=im2double(rgb2gray(Image));
//ShowImage(gray,"gray image");
//title("grayscale",'fontsize',3 );
```

```
imwrite(gray,'LAB14_1.jpg');

///////////// DFT /////////////////////////
h1=fft2(gray);//fft2() is used to find 2-Dimensional
    Fast Fourier Transform of an matrix
i1=log(1+abs(h1));
in1=fftshift(i1);//fftshift() is used to rearrange
   the fft output, moving the zero frequency
inm1=mat2gray(in1);
//figure;
//ShowImage(inm1,'Frequency Spectrum of Original
   Image');
//title("Frequency Spectrum",'fontsize',3 );
imwrite(inm1,'LAB14_2.jpg');

/////////////// DCT ///////////////////////
h2=dct(gray,1); // DCT Transfom
//figure;
//ShowImage(h2,'Frequency Spectrum of Original Image
   ');
//title("DCT Coefficient",'fontsize',3 );
imwrite(h2,'LAB14_3.jpg');
h3=idct(h2); // IDCT Transfom
h4=mat2gray(h3);
//figure;
//ShowImage(h4,'Frequency Spectrum of Original Image
   ');
//title("Recovered Image using DCT Coefficient",'
   fontsize',3 );
imwrite(h4,'LAB14_4.jpg');

/////////////// DWT ////////////////////////
[CA CH CV CD]=dwt2(gray,'db2','mode','asymh');
[M N]=size(CA);
temp=[CA ones(M,10) CH ones(M,10) CV ones(M,10) CD];
    //Padding Data for Displayng DWT Coefficients
//figure;ShowImage(temp,"CA CH CV CD DWT Coefficient
   ");
```

```
//title("CA CH CV CD DWT Coefficient",'fontsize',3 )
    ;
imwrite(temp,'LAB14_5.jpg');

x1=size(gray);
X = idwt2(CA,CH,CV,CD,'db2',x1); // Inverse DWT
//figure;ShowImage(X,'Recovered Image');
//title('inverse dwt','fontsize',3);
imwrite(X,'LAB14_6.jpg');
```

Figure 14.1: Performm the Following Transform on Gray Scale Image

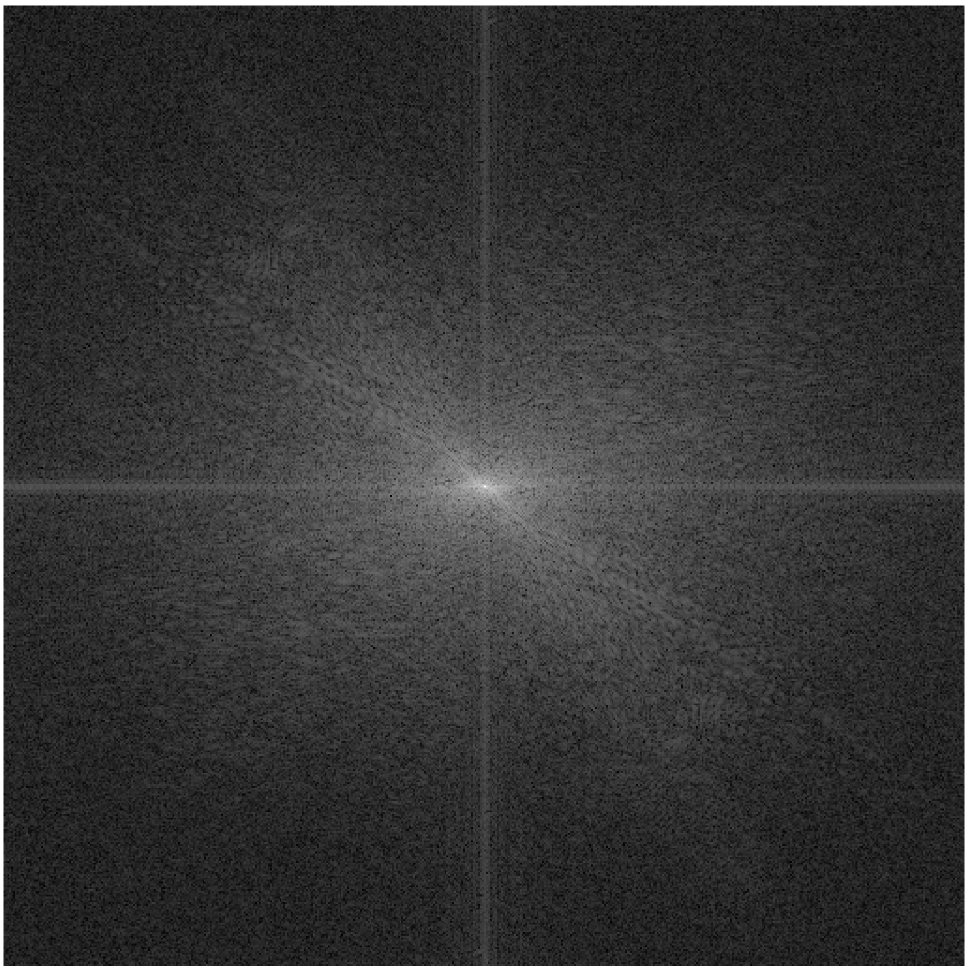


Figure 14.2: Performm the Following Transform on Gray Scale Image

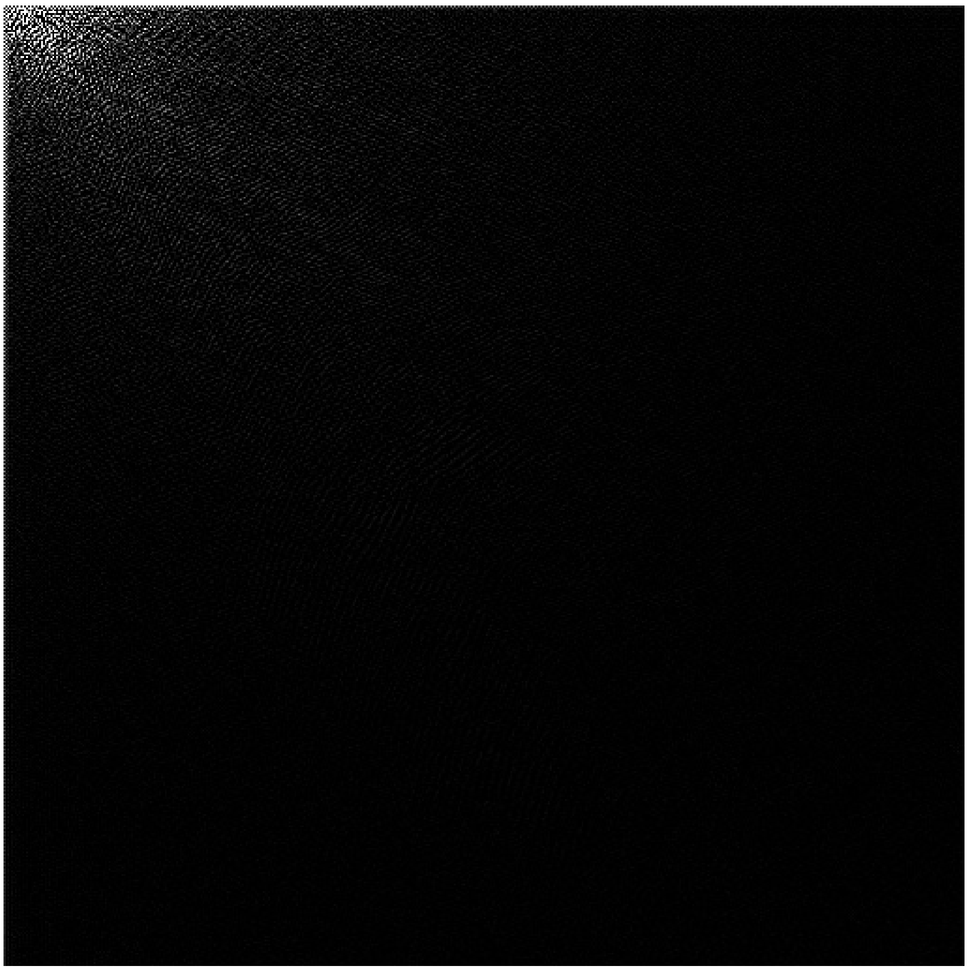


Figure 14.3: Performm the Following Transform on Gray Scale Image

Figure 14.4: Performm the Following Transform on Gray Scale Image

Figure 14.5: Performm the Following Transform on Gray Scale Image

Figure 14.6: Performm 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 Performm the Various Edge Detection Methods on Gray Scale Image

```
// LAB:15 Performm the Various Edge Detection
    Methods on Gray Scale Image.
// Version : Scilab 5.4.1
// Operating System : Window-xp, Window-7
//Toolbox: Image Processing Design 8.3.1-1
//Toolbox: SIVP 0.5.3.1-2

clc;
clear all;
xdel(winsid());
a=imread('LAB15_1.jpg');
a=rgb2gray(a);
//figure,ShowImage(a,'horizontal');
//title('Original Image','color','red','fontsize',
    4);
imwrite(a,'LAB15_1.jpg');
```

```
im=im2double(a);
f1=[-1 -2 -1;0 0 0;1 2 1] //Horizontal Edge Detection Mask
b=imfilter(im,f1); // 2D Convoluation between Image and Filter Mask
//figure,ShowImage(b,'horizontal');
//title('Horizontal Edge Detected Image','color','red','fontsize', 4);
imwrite(b,'LAB15_2.jpg');


f2=[-1 0 1;-2 0 2;-1 0 1] //Vertical Edge Detection Mask
c=imfilter(im,f2); // 2D Convoluation between Image and Filter Mask
//figure,ShowImage(c,'vertical');
//title('Vertical Edge Detected Image','color','red','fontsize', 4);
imwrite(c,'LAB15_3.jpg');

f3=[0 -1 -2;1 0 -1;2 1 0] //+45 Diagonal Edge Detection Mask
d=imfilter(im,f3); // 2D Convoluation between Image and Filter Mask
//figure,ShowImage(d,'+45 degree');
//title('+45 Diagonal Edge Detected Image','color','red','fontsize', 4);
imwrite(d,'LAB15_4.jpg');

f4=[-2 -1 0;-1 0 1;0 1 2] //-45 Diagonal Edge Detection Mask
e=imfilter(im,f4); // 2D Convoluation between Image and Filter Mask
//figure,ShowImage(e,'-45 degree');
//title('-45 Diagonal Edge Detected Image','color','red','fontsize', 4);
imwrite(e,'LAB15_5.jpg');

```

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

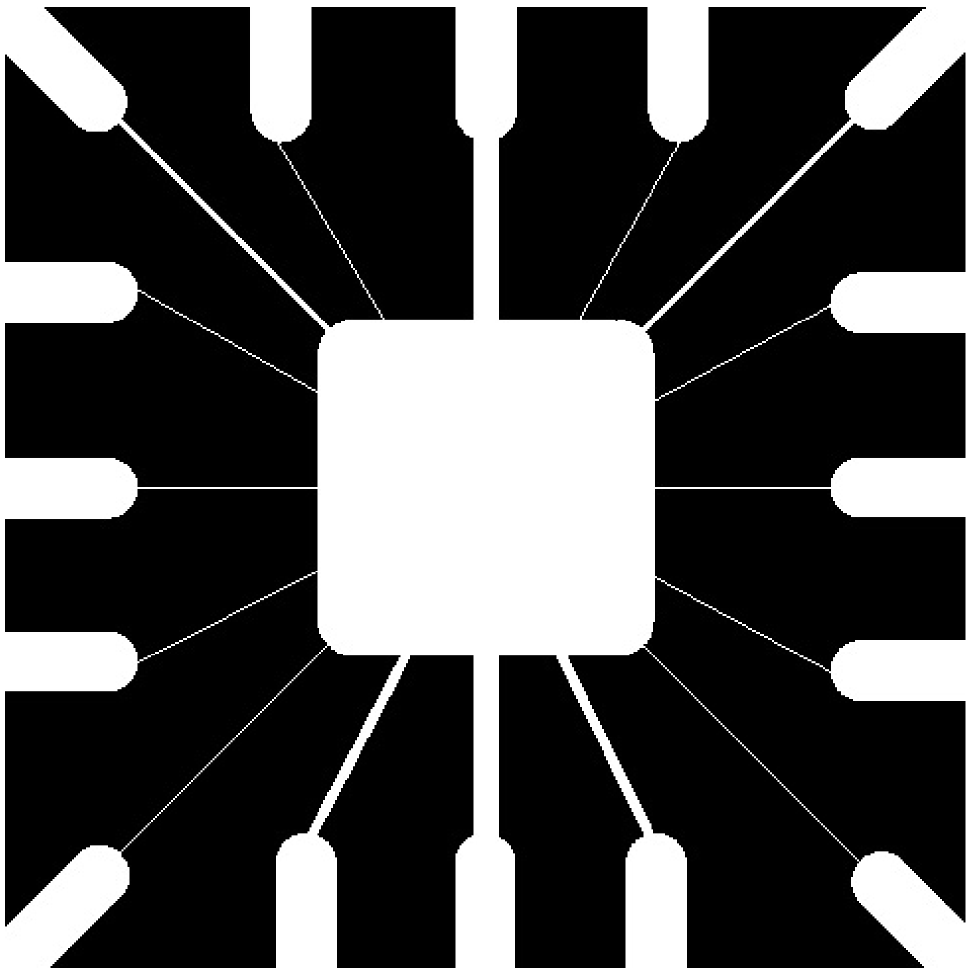


Figure 15.1: Performm the Various Edge Detection Methods on Gray Scale Image

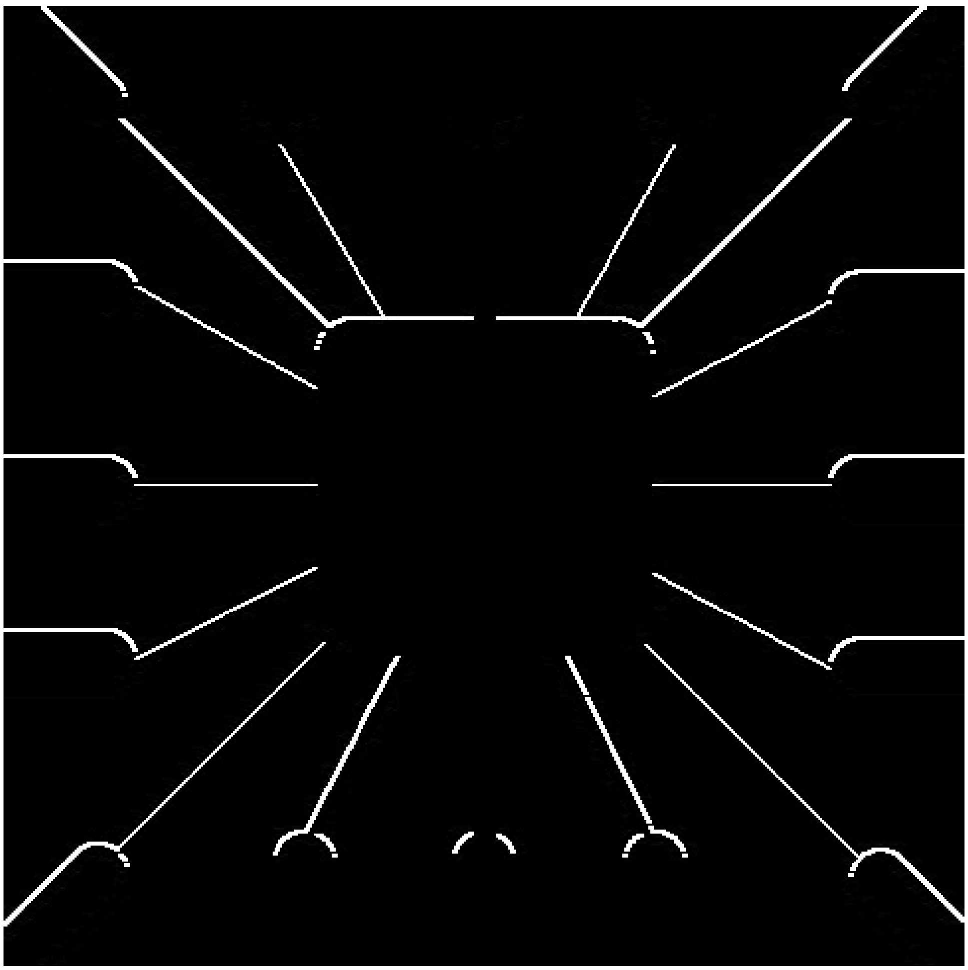


Figure 15.2: Performm the Various Edge Detection Methods on Gray Scale Image

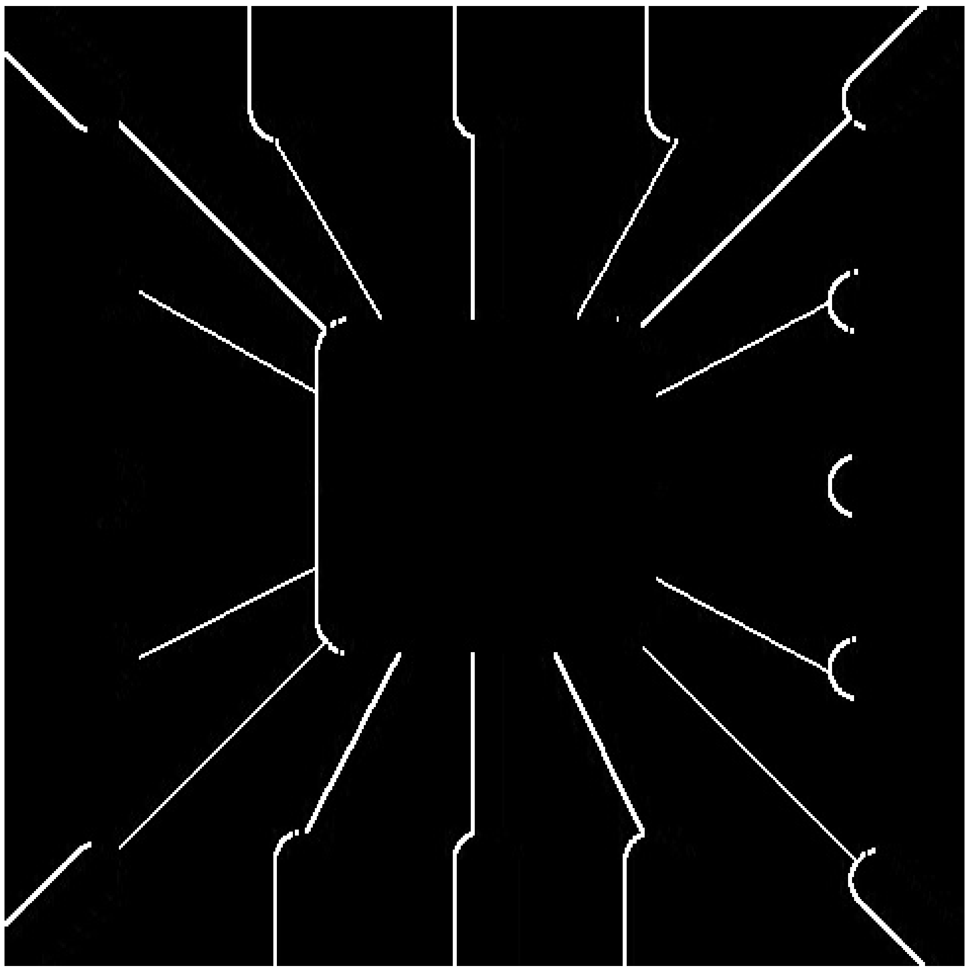


Figure 15.3: Performm the Various Edge Detection Methods on Gray Scale Image

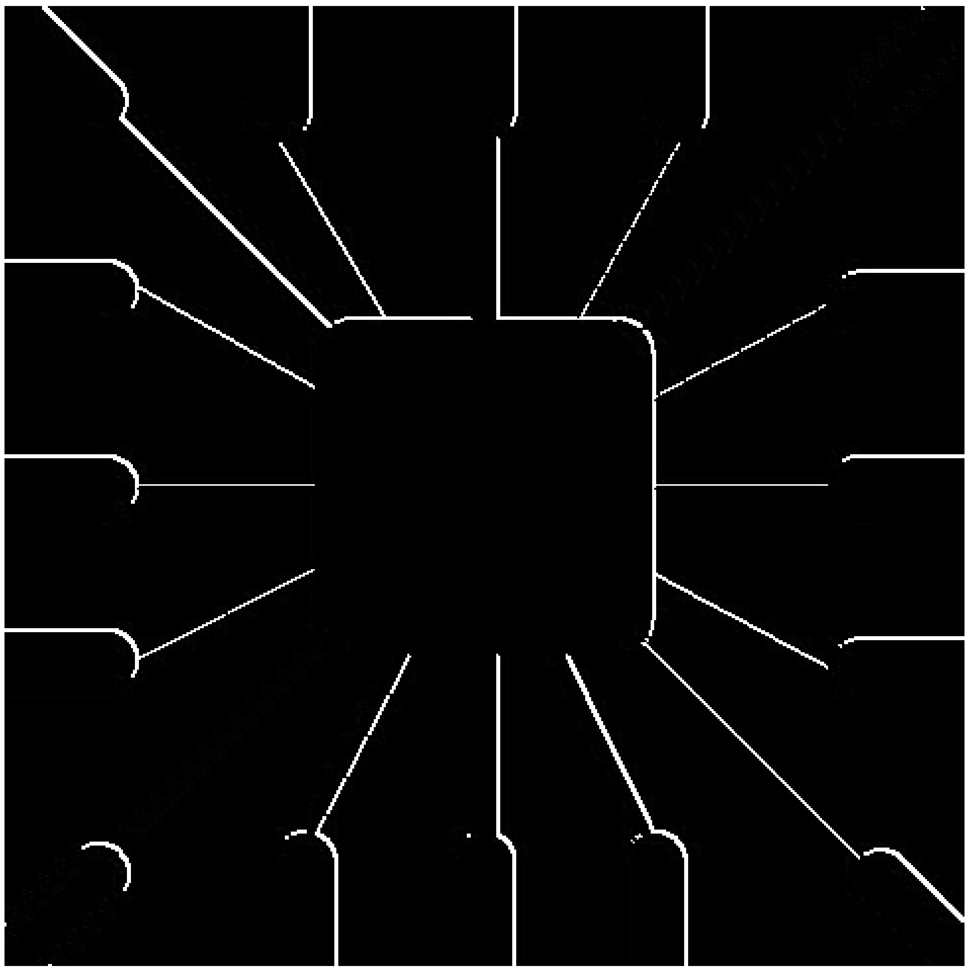


Figure 15.4: Performm the Various Edge Detection Methods on Gray Scale Image

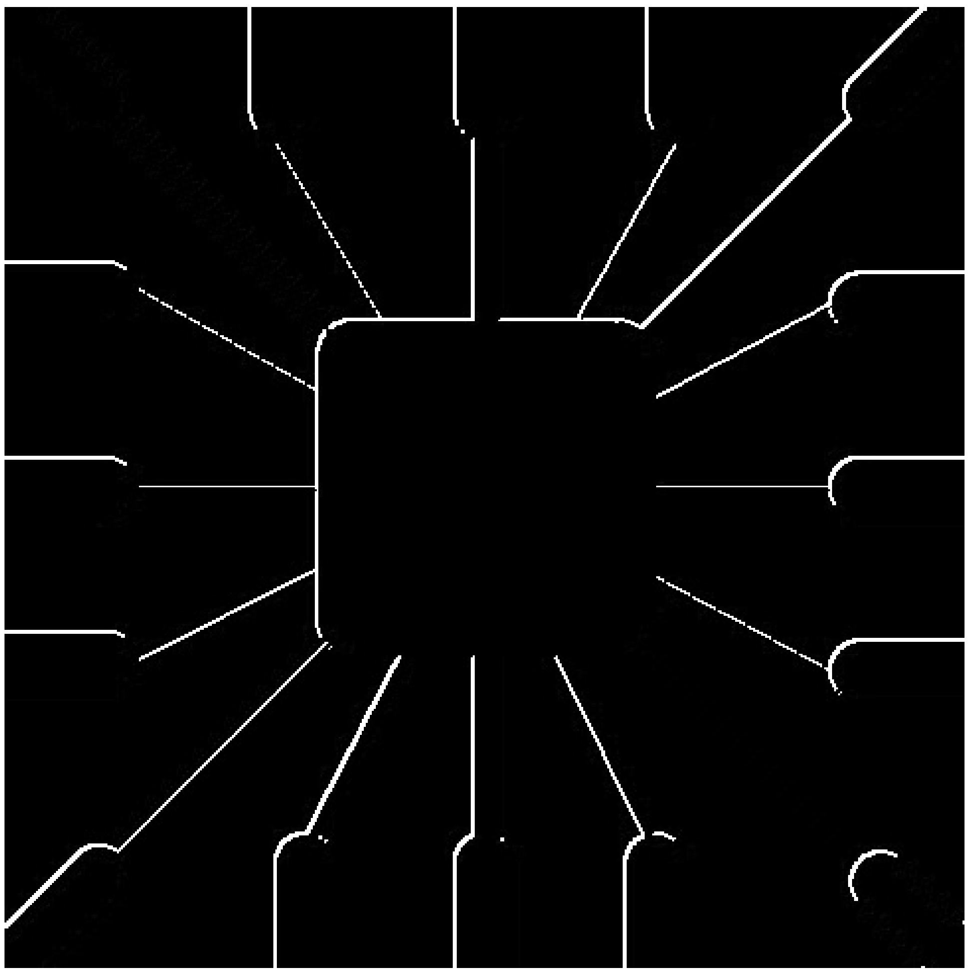


Figure 15.5: Performm the Various Edge Detection Methods on Gray Scale Image

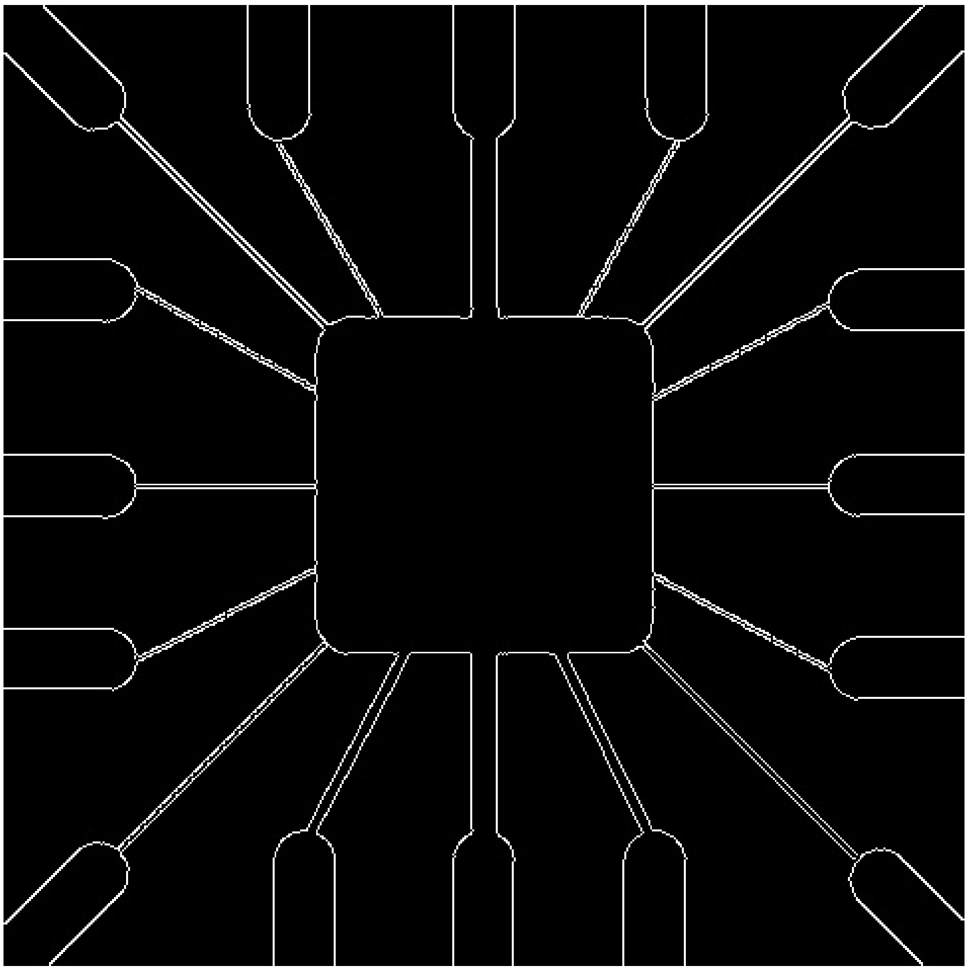
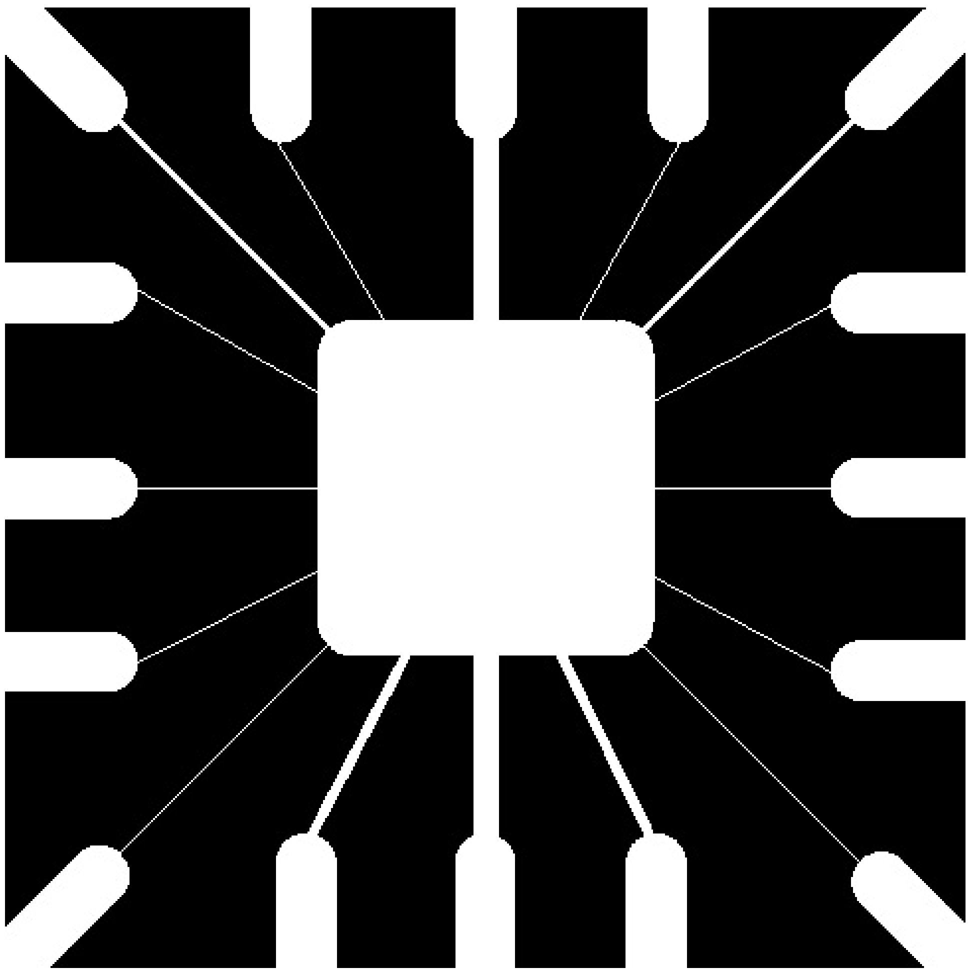


Figure 15.6: Performm the Various Edge Detection Methods on Gray Scale Image

Appendix



Perforrm the Various Edge Detection Methods on Gray Scale Image