Lab Sheet 1

IMPLEMENT DSSS AND LINE CODING IN MATLAB OR EQUIV. TOOL

INTRODUCTION TO MATLAB:

MATLAB is an interactive system whose basic data element is an array that does not require dimensioning. It stands for Matrix Laboratory. It is a software package for high-performance mathematical computation, visualization, and programming environment. It provides an interactive environment with hundreds of built-in functions for technical computing, graphics, and animations. It provides vast library of mathematical functions for linear algebra, statistics, Fourier analysis, filtering, optimization, numerical integration and solving ordinary differential equations.

MATLAB provides built-in graphics for visualizing data and tools for creating custom plots. MATLAB's programming interface gives development tools for improving code quality maintainability and maximizing performance. It provides tools for building applications with custom graphical interfaces. MATLAB is a modern programming language environment, and it has refined data structures, includes built-in editing and debugging tools, and supports object-oriented programming. MATLAB is Multi-paradigm. So, it can work with multiple types of programming approaches, such as Functional, Object-Oriented, and Visual.

Some of the basic commands used in MATLAB are listed below:

clc	Clears command window.
clear	Removes variables from memory.
exist	Checks for existence of file or variable.
global	Declares variables to be global.
help	Searches for a help topic.
lookfor	Searches help entries for a keyword.
quit	Stops MATLAB.
who	Lists current variables.
cd	Changes current directory.
date	Displays current date.
delete	Deletes a file.
disp	Displays contents of an array or string.

fscanf	Read formatted data from a file.
fprintf	Performs formatted writes to screen or file.
input	Displays prompts and waits for input.
axis	Sets axis limits.
fplot	Intelligent plotting of functions.
grid	Displays gridlines.
plot	Generates xy plot.
print	Prints plot or saves plot to a file.
title	Puts text at top of plot.
xlabel	Adds text label to x-axis.
ylabel	Adds text label to y-axis.
cat	Concatenates arrays.
find	Finds indices of nonzero elements.
length	Computes number of elements.
max	Returns largest element.
min	Returns smallest element.
size	Computes array size.
sort	Sorts each column.
sum	Sums each column.
%s	Format as a string.
%d	Format as an integer.
%f	Format as a floating point value.
%e	Format as a floating point value in scientific notation.
%g	Format in the most compact form: %f or %e.
\n	Insert a new line in the output string.
\t	Insert a tab in the output string.
;	Suppresses screen printing.

DIRECT SEQUENCE SPREAD SPECTRUM (DSSS)

THEORY:

The Direct Sequence Spread Spectrum (DSSS) is a spread-spectrum modulation technique primarily used to reduce overall signal interference in telecommunication. The Direct Sequence Spread Spectrum modulation makes the transmitted signal wider in bandwidth than the information bandwidth. In DSSS, the message bits are modulated by a bit sequencing process known as a spreading sequence. This spreading-sequence bit is known as a chip. It has a much shorter duration (larger bandwidth) than the original message bits. DSSS increases transmit signal bandwidth to remove intersymbol interference (ISI) and narrowband interference. It provides security of transmission because receiver must know pseudo-random sequence to get transmitted data signal. Otherwise, receiver can't detect original message signal, and it only sees transmitted signal as a noise. This property of DSSS technique make it suitable to use for military communication systems. DSSS also enables to usage of a bandwidth by multiple users because each user multiply its message signal by different pseudo-random sequence, and a receiver gets its message signal if it has transmitter pseudo-random sequence. The process of DSSS is shown below.

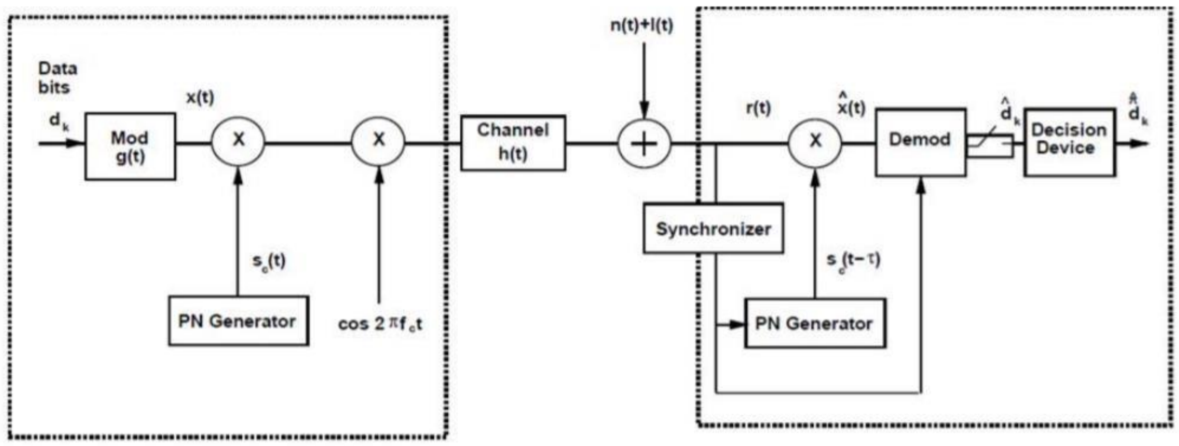


Figure : *Block diagram of Spread Spectrum System*

CODE IMPLEMENTATION:

```
  clear all;
  %% parameters
  Fs = 1000;
  fc = 100;
  fp = 4;
  bit_t = 0.1;
%% message generation with BPSK
  m = [0 0 1 1 1 1 0 0];
  for bit = 1:length(m)
     if(m(bit)==0)
          m(bit) = -1;
```

```
   end
end
message =  repmat(m,fp,1);
message =  reshape(message,1,[]);

%% PN generation and multiply with message
pn_code = randi([0,1],1,length(m)*fp);
for bit = 1:length(pn_code)
   if(pn_code(bit)==0)
        pn_code(bit) = -1;
   end
end
DSSS = message.*pn_code;

%% create carrier and multipy with encoded sequence
t = 0:1/Fs:(bit_t-1/Fs);
s0 = -1*cos(2*pi*fc*t);
s1 = cos(2*pi*fc*t);
carrier = [];
BPSK = [];
for i = 1:length(DSSS)
    if (DSSS(i) == 1)
        BPSK = [BPSK s1];
    elseif (DSSS(i) == -1)
        BPSK = [BPSK s0];
    end
    carrier = [carrier s1];

end


%% demodulation
rx =[];
for i = 1:length(pn_code)
    if(pn_code(i)==1)
        rx = [rx BPSK((((i-1)*length(t))+1):i*length(t))];
    else
        rx = [rx (-1)*BPSK((((i-1)*length(t))+1):i*length(t))];
    end
end
demod = rx.*carrier;
result = [];
for i = 1:length(m)
   x = length(t)*fp;
   cx = sum(carrier(((i-1)*x)+1:i*x).*demod(((i-1)*x)+1:i*x));
   if(cx>0)
       result = [result 1];
```

```
  else
      result = [result -1];
  end
end
pn_codeWrong = randi([0,1],1,length(m)*fp);
resultWrong = [];
rx2 =[];
for i = 1:length(pn_code)
   if(pn_codeWrong(i)==1)
       rx2 = [rx2 BPSK((((i-1)*length(t))+1):i*length(t))];
   else
       rx2 = [rx2 (-1)*BPSK((((i-1)*length(t))+1):i*length(t))];
   end
end
demod2 = rx2.*carrier;
for i = 1:length(m)
  x = length(t)*fp;
  cx = sum(carrier(((i-1)*x)+1:i*x).*demod2(((i-1)*x)+1:i*x));
  if(cx>0)
      resultWrong = [resultWrong 1];
  else
      resultWrong = [resultWrong -1];
  end
end
message1 =  repmat(result,fp,1);
message1 =  reshape(message1,1,[]);
message2 =  repmat(resultWrong,fp,1);
message2 =  reshape(message2,1,[]);

%% Draw original message, PN code , encoded sequence on time domain
pn_size = length(pn_code);
tpn = linspace(0,length(m)*bit_t-bit_t/fp,pn_size);
tm = 0:bit_t/fp:length(m)*bit_t-bit_t/fp;
figure
subplot(311)
stairs(tm,message,'linewidth',2)
title('Message bit sequence')
axis([0 length(m)*bit_t -1 1]);
subplot(312)
stairs(tpn,pn_code,'linewidth',2)
title('Pseudo-random code');
axis([0 length(m)*bit_t -1 1]);
subplot(313)
stairs(tpn,DSSS,'linewidth',2)
title('Modulated signal');
axis([0 length(m)*bit_t -1 1]);
```

```
figure
subplot(311)
stairs(tm,message,'linewidth',2)
title('Message bit sequence')
axis([0 length(m)*bit_t -1 1]);
subplot(312)
stairs(tm,message1,'linewidth',2)
title('Received message using true pseudo-random code')
axis([0 length(m)*bit_t -1 1]);
subplot(313)
stairs(tm,message2,'linewidth',2)
title('Received message using wrong pseudo-random code')
axis([0 length(m)*bit_t -1 1]);

%% Draw original message, PN code , encoded sequence on frequency domain
f = linspace(-Fs/2,Fs/2,1024);
figure
subplot(311)
plot(f,abs(fftshift(fft(message,1024))),'linewidth',2);
title('Message spectrum')
subplot(312)
plot(f,abs(fftshift(fft(pn_code,1024))),'linewidth',2);
title('Pseudo-random code spectrum');
subplot(313)
plot(f,abs(fftshift(fft(DSSS,1024))),'linewidth',2);
title('Modulated signal spectrum');
figure;
subplot(311)
plot(f,abs(fftshift(fft(BPSK,1024))),'linewidth',2);
title('Transmitted signal spectrum');
subplot(312)
plot(f,abs(fftshift(fft(rx,1024))),'linewidth',2);
title('Received signal multiplied by pseudo code');
subplot(313)
plot(f,abs(fftshift(fft(demod,1024))),'linewidth',2);
title('Demodulated signal spectrum before decision device ');
```

OUTPUT:

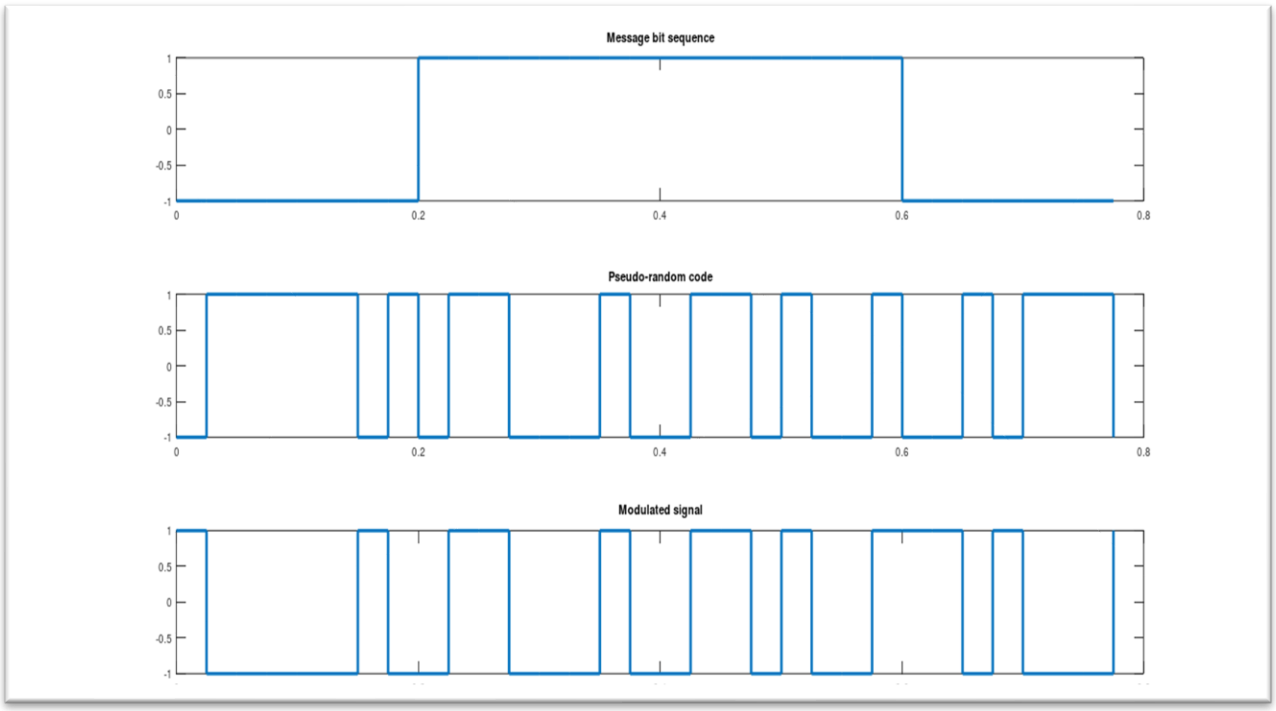


Figure 1: Message bit sequence, Pseudo random code and Modulated signal

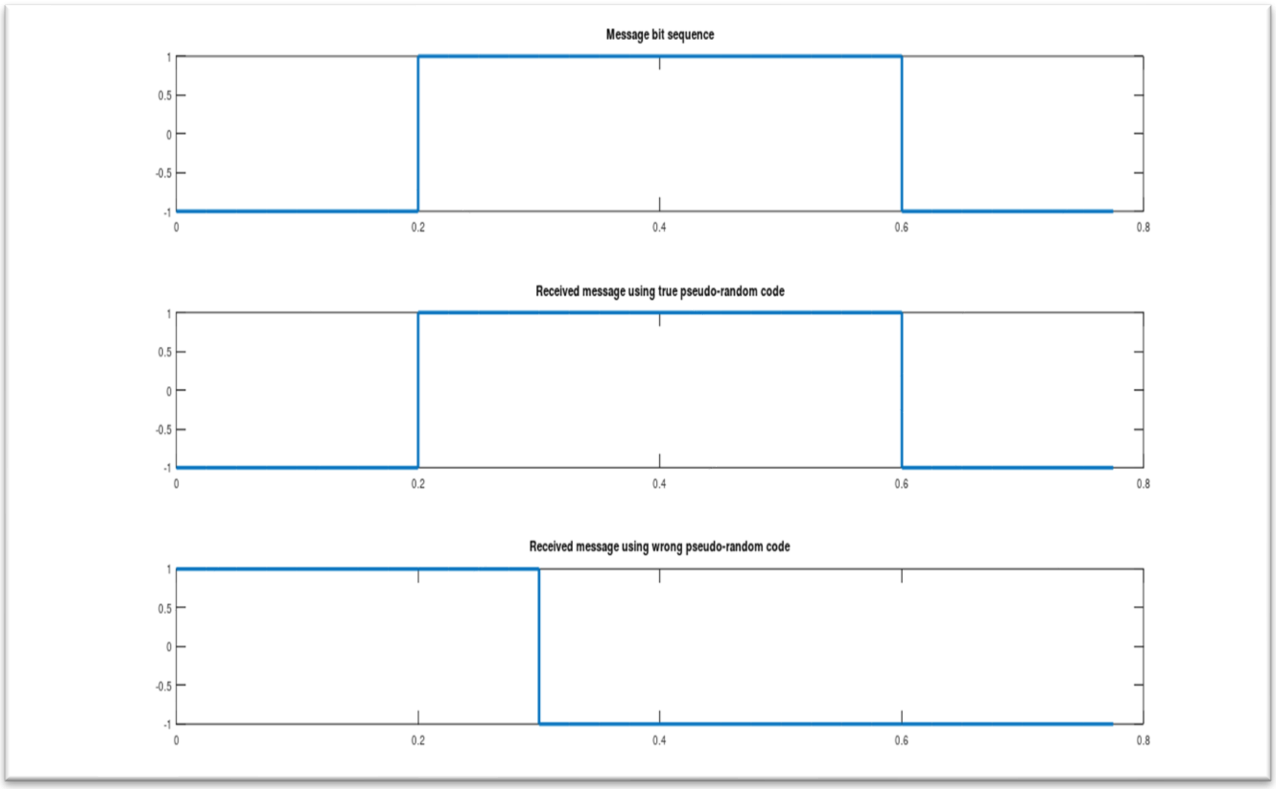


Figure 2: Original Message and results with true and wrong key at receiver

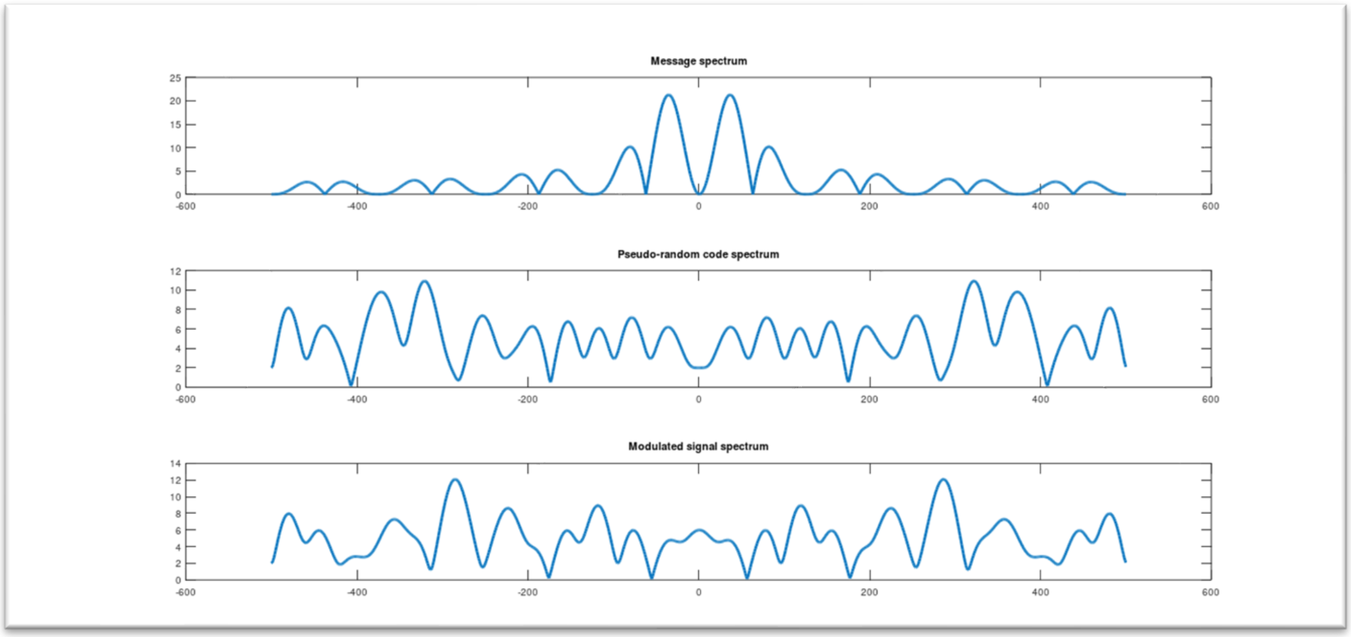


Figure3: Message spectrum, Pseudo random code spectrum and Modulated signal spectrum

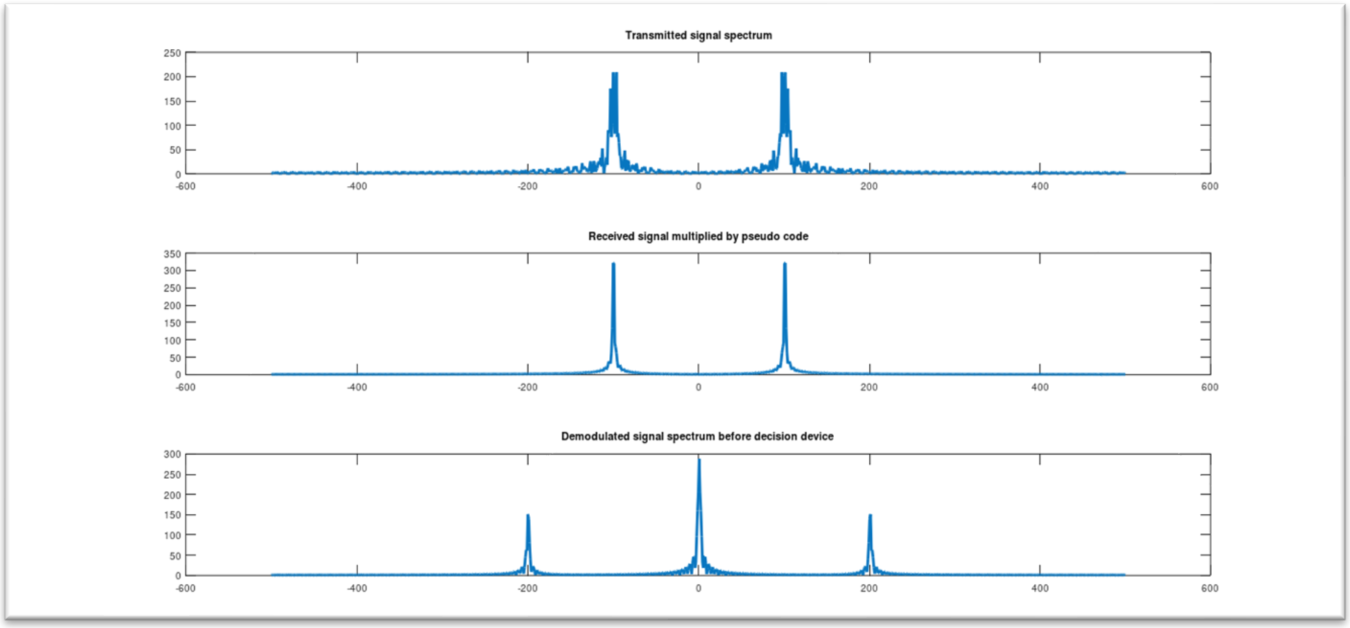


Figure 4: Draw original message, PN code , encoded sequence on frequency domain

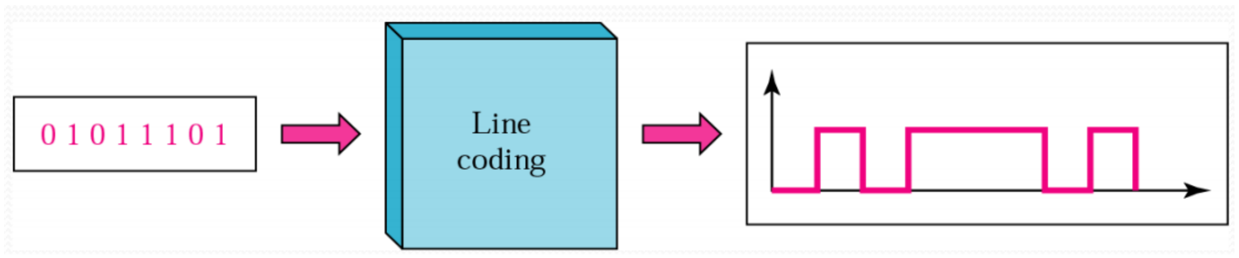
CONCLUSION:

Thus, in this way, Direct Sequence Spread Spectrum was implemented successfully.

LINE CODING

THEORY:

The line coding is defined as the process of converting binary data, a sequence of bits to a digital signal. The digital data such as text, numbers, graphical images, audio and video are stored in computer memory in the form of sequences of bits. Data as well as signals that represents data can either be digital or analog. Line coding is the process of converting digital data to digital signals. Whenever we transmit data it is in the form of digital signals, so with the help of line coding, we can convert a sequence to bits (or encoding) into a digital signal which then again converted into bits by the receiver (or can be said as decoded by the receiver). A line code is the code used for data transmission of a digital signal over a transmission line. This process of coding is chosen so as to avoid overlap and distortion of signal such as inter-symbol interference.



CODE IMPLEMENTATION:

```
N=10
n=randi([0,1],1,N)
for m=1:N
  if n(m)==1
    nn(m)=1;
  else
    nn(m)=0;
  endif
endfor
i=1
t=0:0.01:length(n);
for j=1:length(t)
  if t(j)<=i
    y(j)=nn(i);
  else
    i=i+1
    y(j)=nn(i);
  endif
 endfor
 plot(t,y);
 axis([1,N,-2,2])
```

OUTPUT:

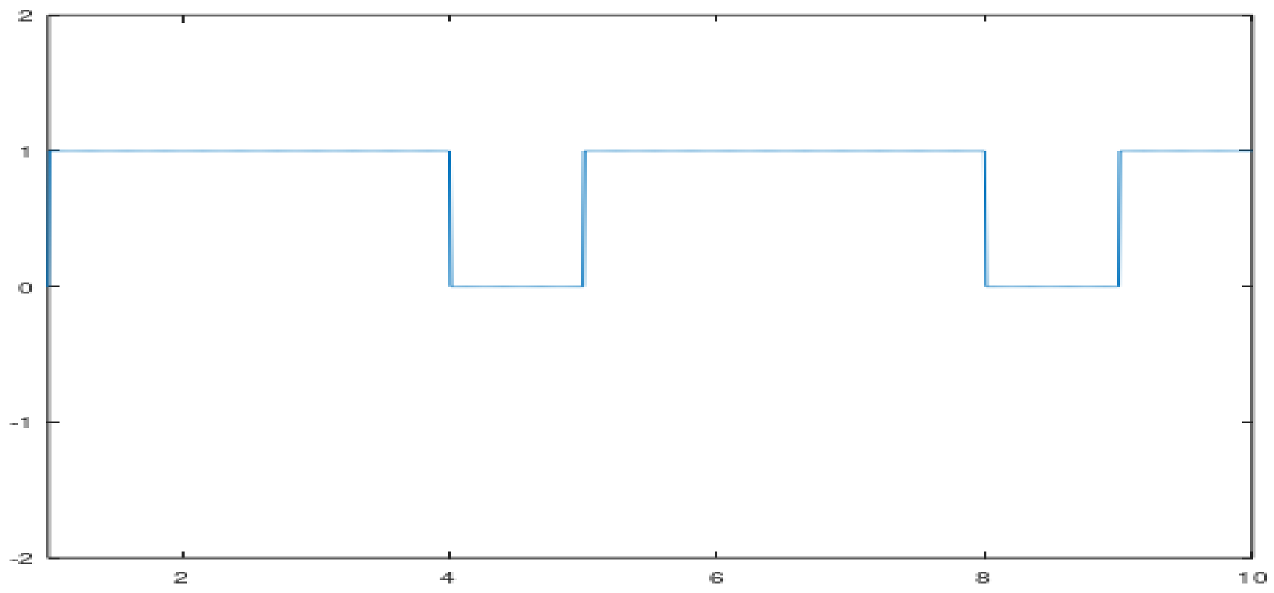


Figure: Line coding

CONCLUSION:

Thus, in this way, line coding was implemented successfully.

Lab Sheet 2

ANALYZE THE PERFORMANCE OF WIFI NETWORK USING NETSIM OR EQUIV. TOOL.

INTRODUCTION TO NETSIM (NETWORK SIMULATOR)

NetSim is an end-to-end, full stack, packet level network simulator and emulator. It provides network engineers with a technology development environment for protocol modeling, network R&D and military communications. The behavior and performance of new protocols and devices can be investigated in an virtual network within NetSim at significantly lower cost and in less time than with hardware prototypes. It is software that predicts the behavior of a computer network. Since communication networks have become too complex for traditional analytical methods to provide an accurate understanding of system behavior, network simulators are used. In simulators, the computer network is modeled with devices, links, applications etc. and the network performance is reported. Simulators come with support for the most popular technologies and networks in use today such as 5G, Internet of Things (IoT), Wireless LANs, mobile ad hoc networks, wireless sensor networks, vehicular ad hoc networks, cognitive radio networks, LTE etc.

WIFI (WIRELESS FIDELITY)

THEORY:

Wi-Fi is a family of wireless network protocols, based on the IEEE 802.11 family of standards, which are commonly used for local area networking of devices and Internet access. The WiFi standards define a fixed channel bandwidth of 25 MHz for 802.11b and 20 MHz for either 802.11a or g networks. primary channel access mechanism is CSMA/CA Wi-Fi is a trademark of the non-profit Wi-Fi Alliance, which restricts the use of the term Wi-Fi Certified to products that successfully complete interoperability certification testing.

Wi-Fi uses multiple parts of the IEEE 802 protocol family and is designed to interwork seamlessly with its wired sibling Ethernet. Compatible devices can network through wireless access points to each other as well as to wired devices and the Internet. The different versions of Wi-Fi are specified by various IEEE 802.11 protocol standards, with the different radio technologies determining radio bands, and the maximum ranges, and speeds that may be achieved. Wi-Fi most commonly uses the 2.4 gigahertz (120 mm) UHF and 5 gigahertz (60 mm) SHF ISM radio bands; these bands are subdivided into multiple channels. Channels can be shared between networks but only one transmitter can locally transmit on a channel at any moment in time.Performance of a WiFi link and system depends on signal quality, interference and so on.

Signal Quality:

Signal qualtity is the function of transmitted power, path loss, noise, shadowing, etc. Poor signal quality leads to packet errors and losses. Quality is how much interference there is between the satellite and your decoder. Signal quality is how good the information you are recieving actually is All signals hold reflections/ghosting or interference that we call noise. Noise will cause the loss of data. The noise will be low if the quality is high or good. If the the signal is suffering due to reflections or interfernce from electrical installations switches/motors etc then the quality will be poor as these problems will cause the loss of data in the signal as the effect of them will block/overpower the signal and your box won't be able to "read" the true data. Without any of these problems the your quality will be high.

Interference:

Interference is that which modifies a signal in a disruptive manner, as it travels along a communication channel between its source and receiver. The term is often used to refer to the addition of unwanted signals to a useful signal. Interference occurs when unwanted signals disrupt wireless communication.Interference may prevent reception altogether, may cause only a temporary loss of a signal, or may affect the quality of the audio or video produced by your equipment. Interference is the primitive nature of the wireless communication system. It is the case in which multiple transmission occurs many times simultaneously over a common communication medium. Interference can come from WiFi and non WiFi radios waves.It leads to poor channel utilization, packet errors and losses.

IMPLEMENTATION:

Network Setup: An IEEE 802.11n link between an access point and a client downlink UDP traffic between a server and the client was configured as shown in figure below:

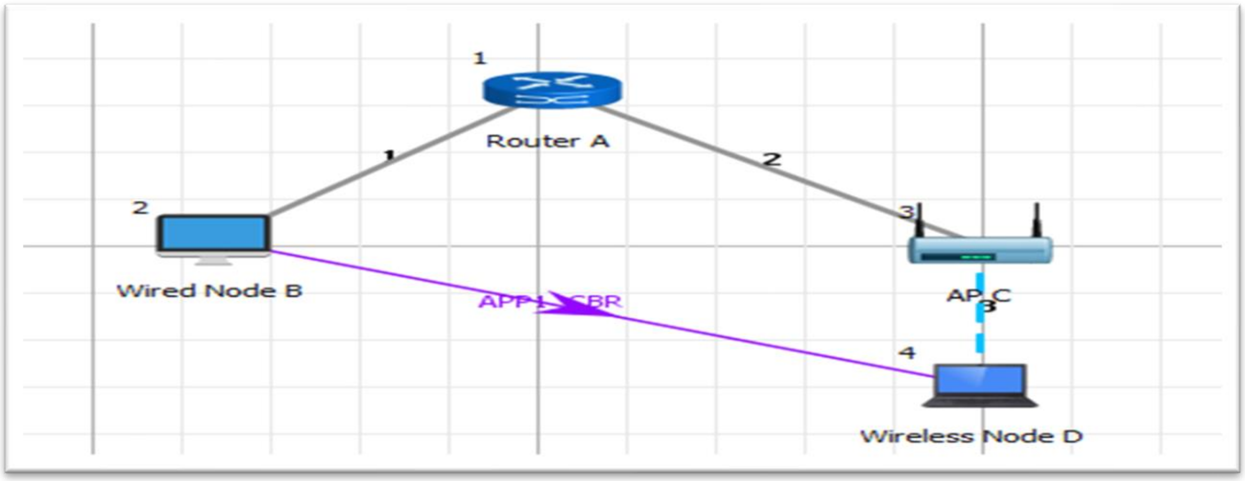


Figure: IEEE 802.11n network topology

IEEE 802.11n supports a fixed set of data rates among which we had used BPSK and QPSK both having coding rate of ½.

MCS	Modulation Type	Coding Rate	Data Rate
0	BPSK	1/2	7.2
1	QPSK	1/2	14.4

Table: Modulation type and it's Data rate

Rate Adaptation: Experiment Configuration

WiFi Radio	
Standard	802.11n
Band	2.4 GHz
Channel	1
Bandwidth	20 MHz
Transmit Power	20 dBm and 10 dBm
Rate Adaptation	False

Wireless Channel and Link	
Channel Characteristics	Path Loss Only
Path Loss Model	Log Distance
Path Loss Exponent	3.5
Mobility Model	Constant
AP - Client Distance	Variable

Application Properties	
Application	CBR
Transport Protocol	UDP
Packet Size	1450 bytes
Interarrival Time	116 microseconds
TCP	Disabled

Miscellaneous	
Wired Link Capacity	1 Gbps
Wired Link Delay Simulation Time	10 microseconds 10 seconds

PERFORMANCE MEASURE:

1. **Data rate (from Packet Trace)**:

$$\text{DATARATE} = \frac{\text{PHY LAYER PAYLOAD} * 8}{(\text{PHY LAYER END TIME} - \text{PHY LAYER ARRIVAL TIME} - 40)}$$

2. **Average throughput (from Packet Trace):**

$$\text{AVERAGE THROUGHPUT} = \frac{\text{APPLICATION BYTES RECEIVED} * 8.0}{\text{SIMULATION TIME}}$$

OUTPUT:

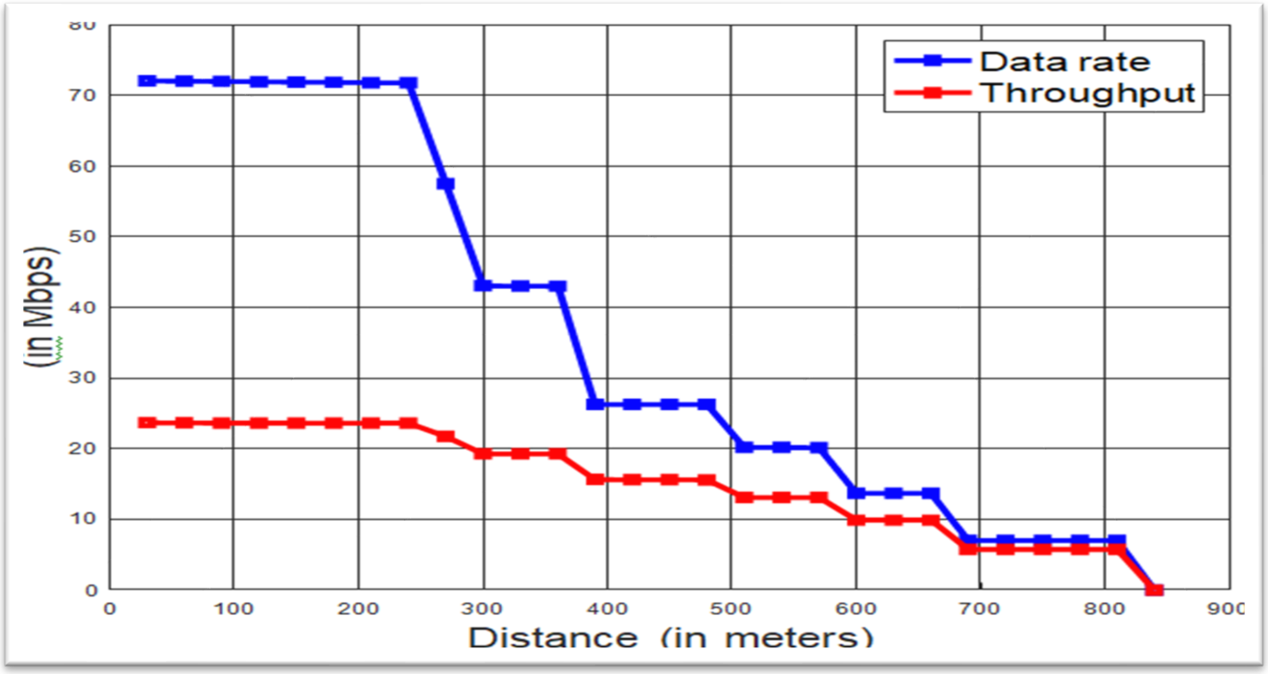


Figure: Relation between Data rate and throughput with distance

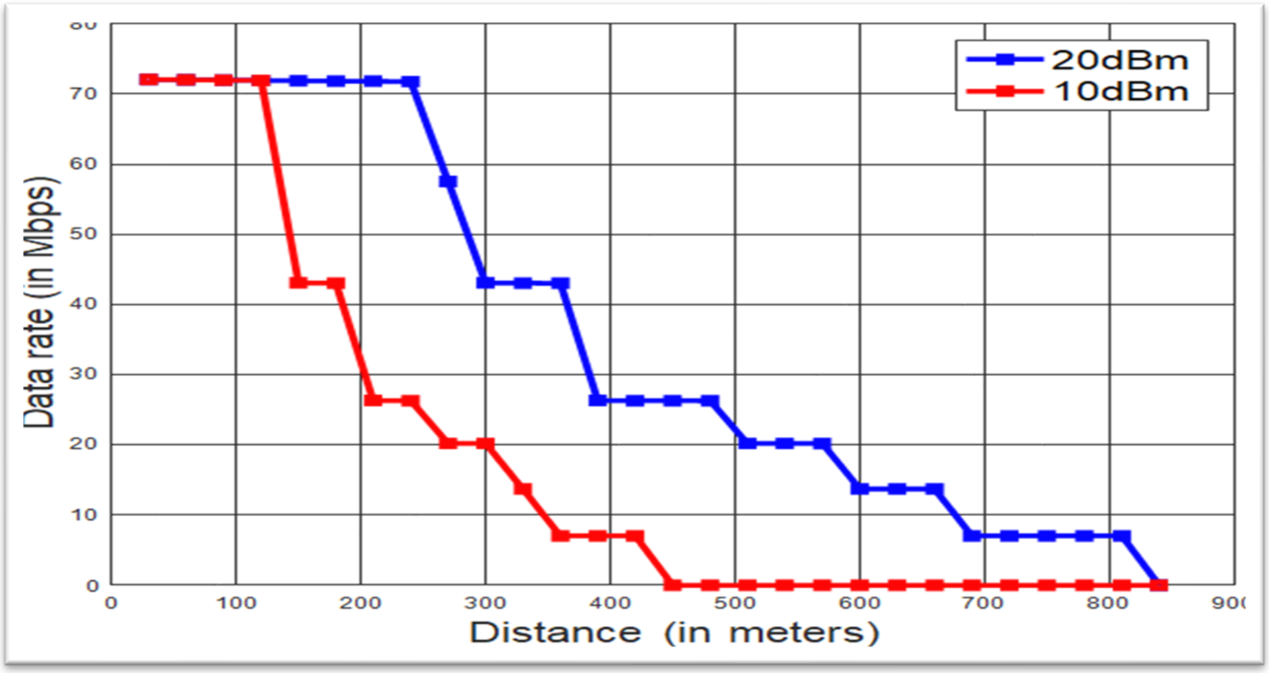


Figure: Relation between data rate with distance and transmit power

CONCLUSION:

Thus, data rate and throughput decreases with distance. Also data rate decreases as transmit power decreases and coverage (range) decreases as transmit power decreases.

Lab Sheet 3

DEVELOP QPSK DETECTOR AND UNDERSTAND THE RELATION BETWEEN BER AND SNR.

THEORY:

Modulation is the process of varying one or more properties of a periodic waveform, called the carrier signal, with a separate signal called the modulation signal that typically contains information to be transmitted Modulation is the process of changing the parameters of the carrier signal, in accordance with the instantaneous values of the modulating signal. The purpose of modulation is to impress the information on the carrier wave, which is used to carry the information to another location.Digital modulation methods can be considered as digital-to-analog conversion and the corresponding demodulation or detection as analog-to-digital conversion. The changes in the carrier signal are chosen from a finite number of M alternative symbols (the modulation alphabet). Digital Modulation provides more information capacity, high data security, quicker system availability with great quality communication.

QPSK is a modulation scheme that allows one symbol to transfer two bits of data. There are four possible two-bit numbers (00, 01, 10, 11), and consequently we need four phase offsets. Again, we want maximum separation between the phase options, which in this case is 90°. With four phases, QPSK can encode two bits per symbol, shown in the diagram with Gray coding to minimize the bit error rate (BER).Each adjacent symbol only differs by one bit.

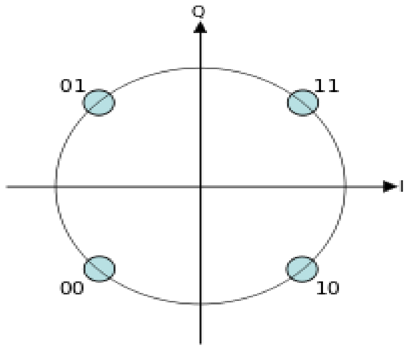


Figure: Constellation diagram for QPSK with Gray coding

SNR:

SNR is defined as the ratio of a signal power to noise power and it is normally expressed in decibel (dB). Signal-to-noise ratio (SNR or S/N) is a measure used in science and engineering that compares the level of a desired signal to the level of background.The mathematical expression of SNR is

$$SNR = 10\log_{10}\left(\frac{Signal\ \text{Power}}{Noise\ Power}\right)\text{dB}$$

If Vs = Vn, then S/N = 0. In this situation, the signal borders on unreadable, because the noise level severely competes with it. In digital communications, this will probably cause a reduction in data speed because of frequent errors that require the source (transmitting) computer or terminal to resend some packets of data.A ratio higher than 1:1 (greater than 0 dB) indicates more signal than noise.

BER:

BER is a performance measurement that specifies the number of bit corrupted or destroyed as they are transmitted from its source to its destination. Several factors that affect BER include bandwidth, SNR, transmission speed and transmission medium.Bit error rate, BER assesses the full end to end performance of a system including the transmitter, receiver and the medium between the two. In this way, bit error rate, BER enables the actual performance of a system in operation to be tested, rather than testing the component parts and hoping that they will operate satisfactorily when in place.

The number of bit errors is the number of received bits of a data stream over a communication channel that have been altered due to noise, interference, distortion or bit synchronization errors. The bit error rate (BER) is the number of bit errors per unit time. a bit error rate is defined as the rate at which errors occur in a transmission system. This can be directly translated into the number of errors that occur in a string of a stated number of bits. The definition of bit error rate can be translated into a simple formula:

BER=Errors/Total Number of Bits

If the medium between the transmitter and receiver is good and the signal to noise ratio is high, then the bit error rate will be very small - possibly insignificant and having no noticeable effect on the overall system However if noise can be detected, then there is chance that the bit error rate will need to be considered.

IMPLEMENTATION:

The implementation of QPSK indicates the implementation of higher-order PSK. Writing the symbols in the constellation diagram in terms of the sine and cosine waves used to transmit them:This yields the four phases π/4, 3π/4, 5π/4 and 7π/4 as needed.This results in a two-dimensional signal space with unit basis functions.The first basis function is used as the in-phase component of the signal and the second as the quadrature component of the signal.Hence, the signal constellation consists of the signal-space 4 points.The factors of 1/2 indicate that the total power is split equally between the two carriers.

Comparing these basis functions with that for BPSK shows clearly how QPSK can be viewed as two independent BPSK signals. Note that the signal-space points for BPSK do not need to split the symbol (bit) energy over the two carriers in the scheme shown in the BPSK constellation diagram. QPSK systems can be implemented in a number of ways.

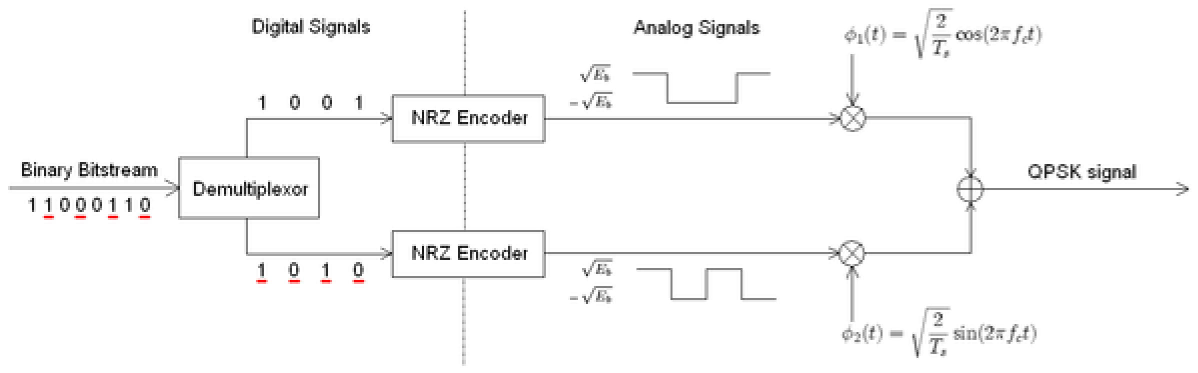


Figure:Conceptual transmitter structure for QPSK

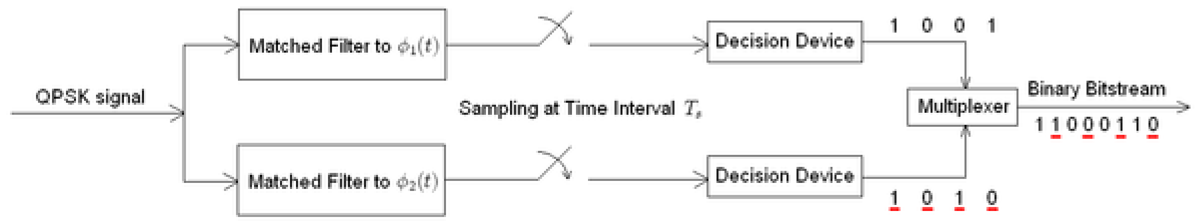


Figure: Receiver structure for QPSK. The matched filters can be replaced with correlators. Each detection device uses a reference threshold value to determine whether a 1 or 0 is detected.

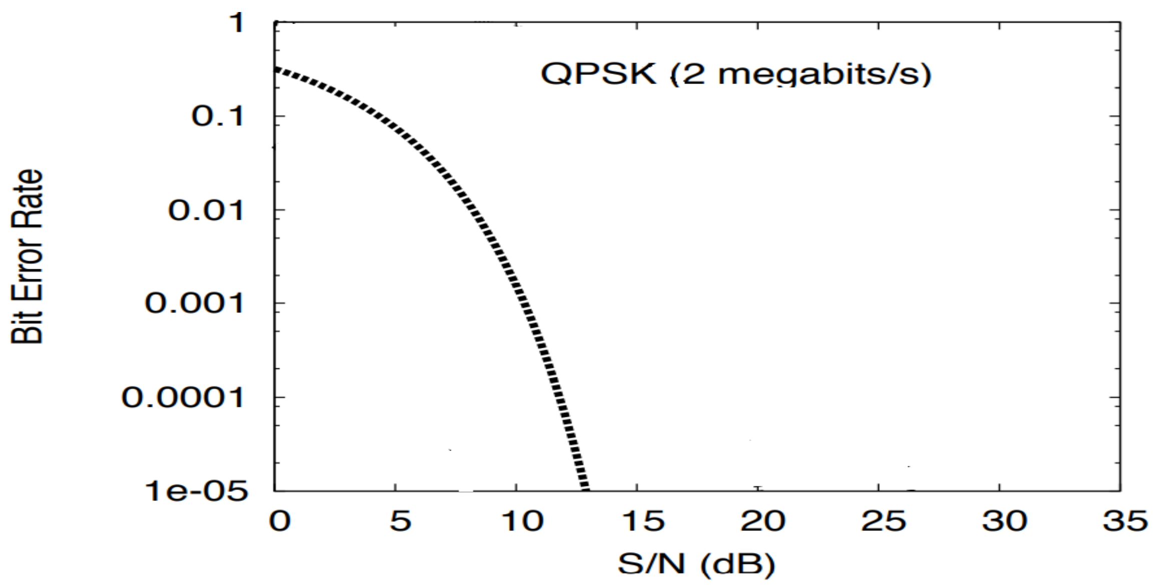


Figure: Graph between Bit Error Rate (BER) and Signal To Noise Ratio (SNR).

Table 1: Simulation result for evaluation on BER vs. SNR for ray tracing (also called 2-ray, one is LOS and other is reflected or NLOS) Additive white Gaussian noise (AWGN) channel for 1 user when the number of data is 200,000.

Signal-to-Noise Ratio (EbNo)	Number of Error	Bit Error rate (BER)
0	15615	7.807500e-002
1	11334	5.667000e-002
2	7520	3.760000e-002
3	4484	2.242000e-002
4	2489	1.244500e-002
5	1205	6.025000e-003
6	462	2.310000e-003
7	165	8.250000e-004
8	39	1.950000e-004
9	2	1.000000e-005
10	1	5.000000e-006

In this simulation, the BERs are obtained by varying the values of Eb/No in the range of 0 to 10. The iteration is done 1000 times where the total number of data transmitted is 200,000.

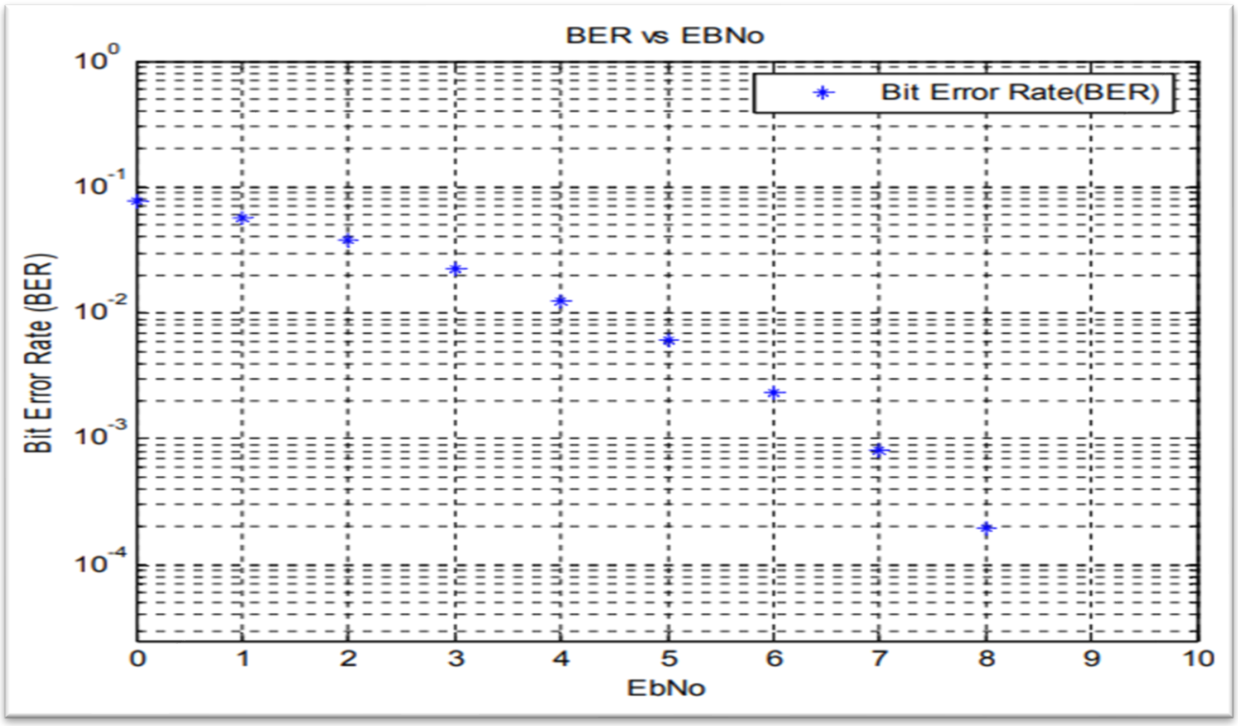


Figure 1: Performance of W-CDMA in ray-tracing model AWGN Channels for 1 user.

CONCLUSION:

Hence, there is inverse relation between SNR (Signal to Noise Ratio) and BER (Bit Error Rate). When BER is increases the SNR decreases and When BER decreases the SNR increases.

Lab Sheet 4

IMPLEMENT VARIOUS PULSE SHAPING FILTERS IMPLEMENTED IN WIRELESS COMMUNICATION.

THEORY:

Pulse shaping is a spectral processing technique by which fractional out of band power is reduced for low cost, reliable, power and spectrally efficient mobile radio communication systems. Pulse shaping is the process of changing the waveform of transmitted pulses. Its purpose is to make the transmitted signal better suited to its purpose or the communication channel, typically by limiting the effective bandwidth of the transmission. By filtering the transmitted pulses this way, the inter symbol interference caused by the channel can be kept in control. In RF communication, pulse shaping is essential for making the signal fit in its frequency band. It is clear that the pulse shaping filter not only reduces inter-symbol interference (ISI), but it also reduces adjacent channel interference. Typically pulse shaping occurs after line coding and modulation. Different Pulse Shapes are:

1) Rectangular Pulse
2) Raised Cosine Pulse
3) Square Root Raised Cosine Pulse
4) Gaussian Pulse
5) Flipped Exponential Pulse
6) Flipped Hyperbolic Secant Pulse
7) Flipped Inverse Hyperbolic Secant Pulse
Out of them, Rectangular pulse had been implemented

Rectangular Pulse: The most basic information unit in a digital transmission scheme is a rectangular pulse. It has a defined amplitude, A, and defined duration, T. Such a pulse is shown in Figure 7, where A = 1, T = To, with the pulse centered about the time origin at t = 0. Typically, a sequence of such pulses (each delayed by T seconds relative to the previous one) constitutes the transmission of information. The information, in this case, is encoded in the amplitude of the pulse. The simplest case is when a binary 0 is encoded as the absence of a pulse (A = 0) and a binary 1 is encoded as the presence of a pulse (A = constant). Since each pulse spans the period T, the maximum pulse rate is 1/T pulses per second, which leads to a data transmission rate of 1/T bits per second.

The pulses used to transmit symbols occupy a fixed time interval, T. Thus, the pulse rate is 1/T pulses per second, which leads to a symbol rate of 1/T symbols per second. The unit, symbols per second, is often referred to as baud. The data transmission rate in bits per second is the baud rate multiplied by the number of bits represented by each symbol.The logic 1 is represented by the presence of a pulse of unit amplitude and a logic 0 by the absence of a pulse (that is, zero amplitude).

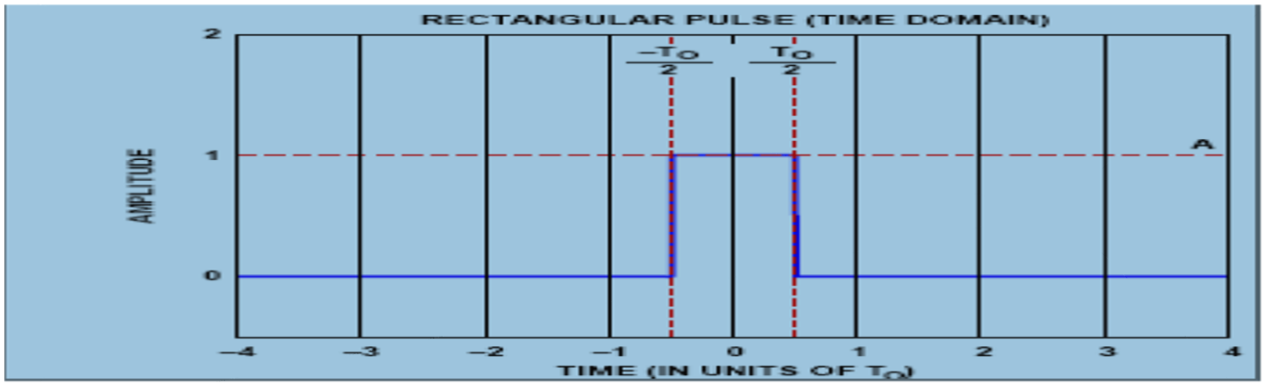


Figure 7. A Single Rectangular Pulse ($T=T_0$, $A=1$)

The rectangular pulse is defined as

$$\Pi_{LT}(t) = \begin{cases} A & -\frac{LT}{2} \le t \le \frac{LT}{2} \\ 0 & \text{otherwise} \end{cases}$$

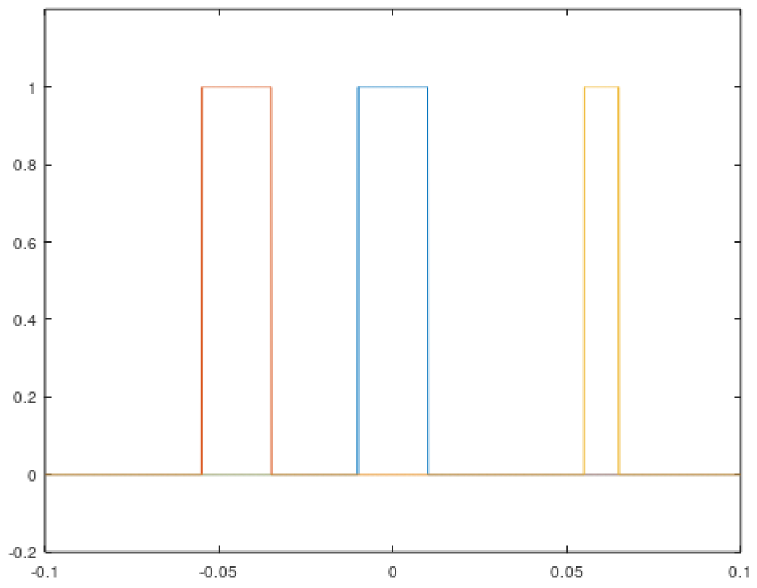
where A is the amplitude of the pulse and L is an integer. The frequency domain representation of the rectangular pulse is

$$F\{\Pi_{LT}(t)\} = A\int_{\frac{LT}{2}}^{\frac{LT}{2}} e^{-j2\pi ft}dt = ALT\frac{\sin(\pi LTf)}{\pi LTf}$$

CODE IMPLEMENTATION:

```
%Generate 200 ms of a rectangular pulse with a sample rate of 10 kHz and a width of 20 ms.
fs = 10e3;
t = -0.1:1/fs:0.1;
w = 20e-3;
x = rectpuls(t,w);
%Generate two copies of the same pulse:
%One displaced 45 ms into the past.
tpast = -45e-3;
xpast = rectpuls(t-tpast,w);
%One displaced 60 ms into the future and half as wide.
tfutr = 60e-3;
xfutr = rectpuls(t-tfutr,w/2);
%Plot the original pulse and the two copies on the same axes.
plot(t,x,t,xpast,t,xfutr)
ylim([-0.2 1.2])
```

OUTPUT



CONCLUSION:

Thus, the Fourier transform of the rectangular pulse is real and its spectrum, a sinc function, is unbounded. This is equivalent to an upsampled pulse-train of upsampling factor L. In real systems, rectangular pulses are spectrally bounded via filtering before transmission which results in pulses with finite rise and decay time.

Lab Sheet 5

IMPLEMENT WIRELESS ROUTING PROTOCOLS: DSDV & AODV

THEORY:

ADOV:

The Ad hoc On Demand Distance Vector (AODV) routing algorithm is a routing protocol designed for ad hoc mobile networks. AODV is capable of both unicast and multicast routing. It is an on-demand algorithm, meaning that it builds routes between nodes only as desired by source nodes. It maintains these routes as long as they are needed by the source. Ad hoc On-demand Distance Vector routing (AODV) protocol enables dynamic, self-starting, multi-hop routing between mobile nodes to establish the ad hoc network. The mobile nodes obtain routes only for those destinations that are in the active communication.

Link breakages are detected by the affected set of nodes and they invalidate the routes using the lost link. When a source node desires to establish a communication session, it initiates a path-discovery process to locate the other node. AODV builds routes using route requested and route reply mechanisms. In order to discover the path, a route request (RREQ) packet is broadcasted across the network to find the route to the destination. Nodes receiving this packet update their information for the source node and set up backwards pointers to the source node in the route tables. To find a path to the destination, the source a initiates Route Request (RREQ) packet across the network and it contains the source address, destination address, source sequence number, destination sequence number, the broadcast identifier and the time to live field. Nodes keep track of the RREQ's source IP address and broadcast ID. If they receive a RREQ which they have already processed, they discard the RREQ and do not forward it. When a node forwards a RREQ packet to its neighbors, it also records in its routing table the node from which the first copy came and it is required by the node to construct the reverse path for the RREP packet. Information about the preceding node from which the packet was received is recorded when a node receives a RREP packet, in turn to forward the data packets to this next node as the next hop toward the destination. Once the source node receives a RREP it can begin using the route to send data packets. Figure 1 shows the propagation of the RREQ across the network.

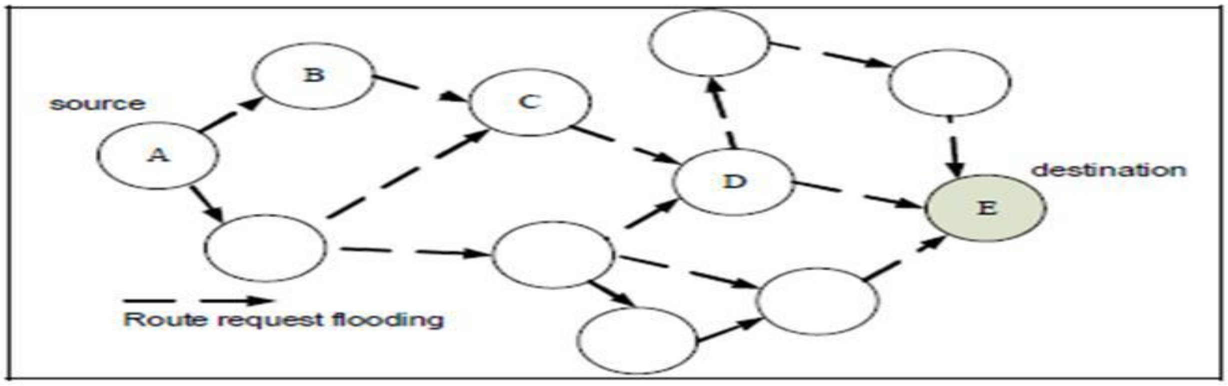


Figure 1. Route Request flooding

Once the RREQ reaches the destination or an intermediate node with a fresh enough route, this node responds by unicasting a route reply (RREP) packet back to the neighbor from which it received the RREQ packet. The path which follows the RREP message is shown in Figure 2.

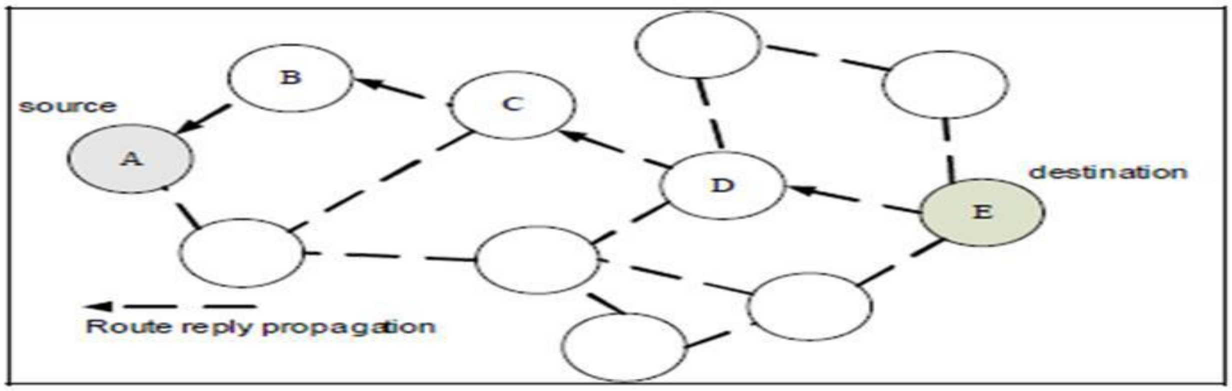
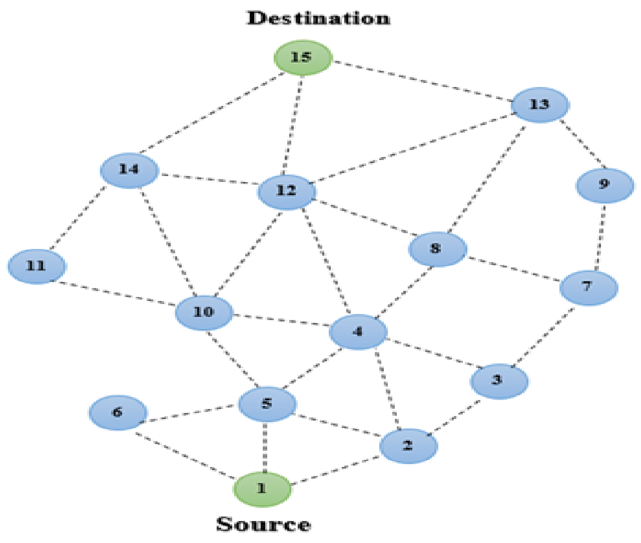
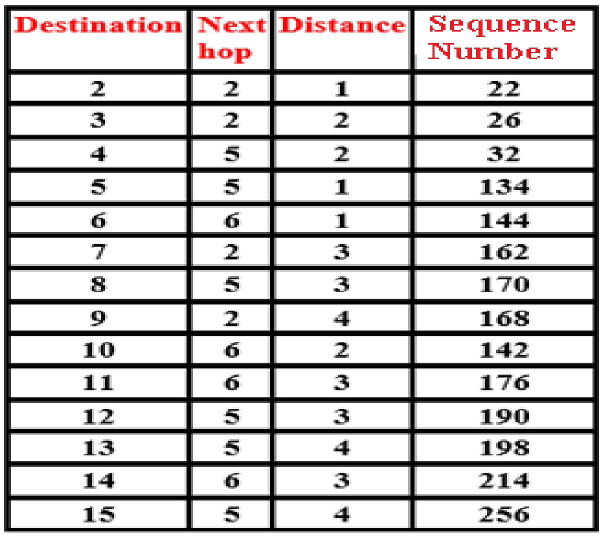


Figure 2. Route Reply propagation

DSDV:

DSDV (Destination Sequenced Distance Vector) is a table-driven algorithm based on the classical Bellman-Ford routing mechanism, but guaranteeing loop-freedom via sequence numbers. Every mobile node in the network maintains a routing table in which all of possible destination within the network and the number of hops to each destination are recorded. Each entry is marked with a sequence number assigned by the destination node. The sequence numbers enable the mobile nodes to distinguish stale routes from newer ones, avoiding the formation of routing loops. Update messages contain the address of the destination, the number of hops to reach the destination and the sequence number of the advertised route. Routes labelled with the most recent sequence number are always preferred. In the event that two updates have the same sequence number, the route with the smaller metric is used. Routes availability to all destinations implies that much less delay is involved in route setup process. The data broadcast by each node will contain its new sequence number, the destination's address, the number of hops count.





Destination	Next hop	Distance	Sequence Number
2	2	1	22
3	2	2	26
4	5	2	32
5	5	1	134
6	6	1	144
7	2	3	162
8	5	3	170
9	2	4	168
10	6	2	142
11	6	3	176
12	5	3	190
13	5	4	198
14	6	3	214
15	5	4	256

Figure: DSDV routing table for above nodes

IMPLEMENTATION:

Network Topology: The following diagram shows a scenario with a topology of 100 nodes. Node 0 is the source which transmits data. Node 100 is the sink or the destination for the whole network.

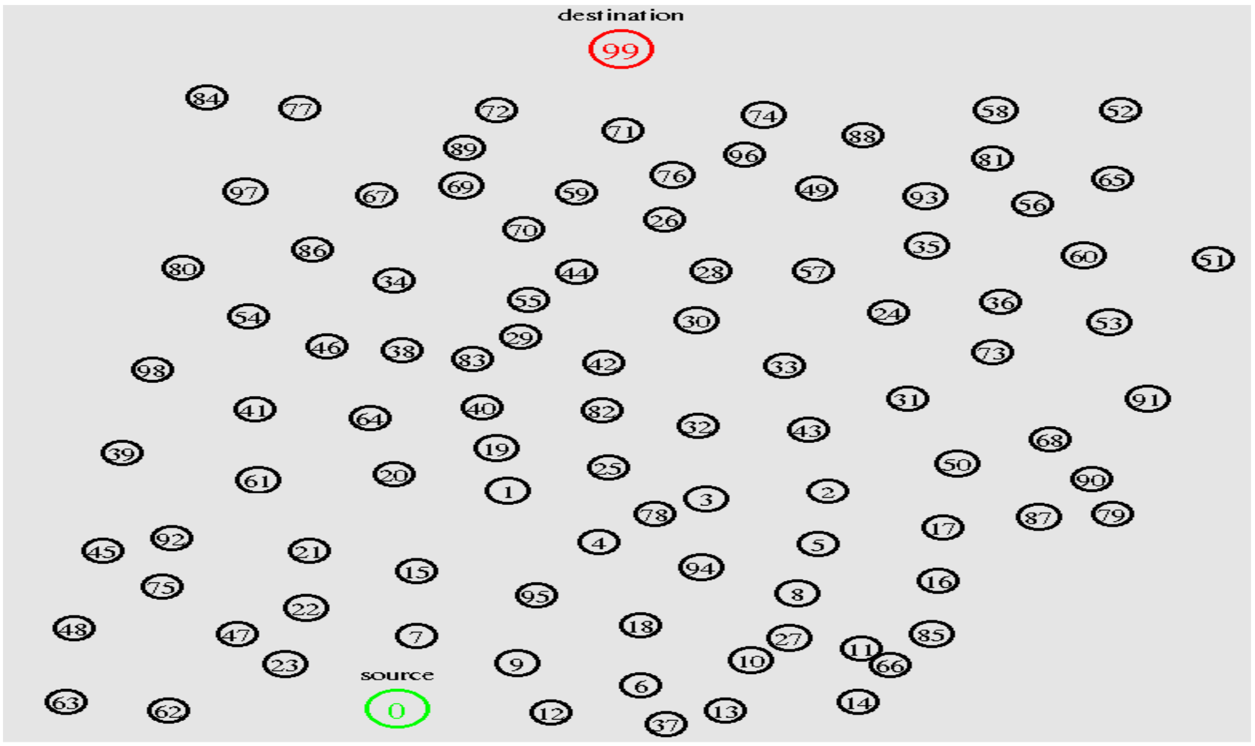


Figure: A WSN Topology with 100 nodes used in simulation

PERFORMANCE METRICS: Various Quality of Service parameters used for analysis routing protocols are defined as follows.

Packet Delivery Ratio (PDR): It is the ratio of deliver packet which is send by the source node and received by the destination node. When packet delivery ratio is high then performance is better.Mathematically, it can be written as in this equation:

$$PDR = \frac{\sum_{i=1}^{N} Total\ packets\ received\ by\ all\ node\ destination}{\sum_{i=1}^{N} Total\ packets\ send\ by\ all\ source}$$

PDR is calculated in % (percentage). Higher values of PDR carry better performance.

Average throughput: It is the ratio between the actual numbers of packets transmitted by the nodes in the system to the number of successfully delivered packets at the base station. The throughput is usually measured in bits per second (bit/sec), and sometimes in data packets per second or data packets per time slot. Higher throughput is always desirable in a communication system. The average throughput is given as follows:

$$\text{Average Throughput} = \frac{\text{recdvSize}}{\text{stopTime} - \text{startTime}} * \left(\frac{8}{1024}\right)$$

Where,

recdvSize = Store received packet ‘s size

Stop Time = Simulation stop time

startTime = Simulation start time

End to End delay: End-to-end delay refers to the time taken for a packet to be transmitted across a network from source to destination. A data packet may take longer time to reach to the destination due to queuing and different routing paths. It is derived in ms (mille second). Smaller values of End-to-end delay carries improved performance. The End-to-end delay is described as:

$$\text{EED} = \frac{\sum_{i=1}^{n}(\text{Tri} - \text{Tsi})}{\sum_{i=1}^{n} \text{Nb received packets}} * 1000 \text{ (ms)} \qquad (3)$$

Where:

I	=	packet identifier
Tri	=	Reception time
Tsi	=	Send time
N	=	Number of packets successfully delivered
NbreceivePackets	=	Number of received Packets

Packet Loss Ratio: Packet loss ratio is the number of packets that never reached the destination to the number of packets originated by the source. We aim to decrease the packet loss ratio. The packet loss ratio is given as:

$$PLR = \frac{\sum_{i=1}^{n} nSentPackets - nReceivedPackets}{\sum_{i=1}^{n} nsentPackets} * 100 \qquad (4)$$

Where

nSentPackets	=	Number of sent packets
nReceivedPackets	=	Number of received packets

SIMULATION RESULTS:

In this section, the performance analysis is carried out on DSDV as proactive candidate and AODV as reactive representative, Performance metrics like Throughput, Packet Delivery Ratio, Dropped Packet, End to End Delay are the four common measures used for the comparison of the performance of above protocols. In the scenario, the density of nodes varies from 10 to 100 nodes. In order to enable direct fair comparisons between the protocols, it was critical to challenge the protocols with identical loads and environmental conditions. Each run of the simulator accepts an input scenario file which describes the exact motion of each node. Since the chosen protocols were challenged with the same scenario file and during the same time (500 seconds), we can directly compare the performance results of all protocols carried out. Simulations were performed by using Network simulator 2 (NS 2.35) Performance metrics are calculated from trace file, with the help of AWK program and it is plotted with the help of Microsoft Excel 2007 tool. The analysis result helps the network designer to choose right protocol. Simulation results are shown in the following section in form of line graphs.

In this scenario, number of nodes connected in a network at a time is varied and thus varying the number of connections, through which the comparison graphs of AODV and DSDV protocols, is made. All nodes are fixed at one place. Table 1 shows the main characteristics used for scenario

Table 1: Various parameters for scenario

Parameter	**Value**
Routing Protocols	AODV, DSDV and ZRP
Number of nodes	10, 25, 50,7 5 and100
Simulation Time	500 seconds
Traffic Type/Network Protocol	CBR/UDP
Bandwidth	0.4 Mb
Packet size	1500 bytes
Mobility Model	Off: A Fixed Topology
Radio Propagation Model	Two Ray Ground
Channel Type	Wireless channel
Queue files	Queue/Drop Tail/Prique
Queue length	50
Mac layer	802.11
Antenna Type	Omni Antenna
Topology size	1200 x 1200

Table 2 and Table 3 give performances results of the routing protocols, with varying the number of nodes from within a margin of 10 to 100 nodes. Network traffic type is chosen as CBR (Constant Bit Rate). The routing protocols are set as AODV and DSDV to compare the simulation data. The performance metrics used for comparison are Average Throughput, End- to-End delay, Packet Loss Ratio, and Delivered Packet Ratio.

Table 2. DSDV Evaluation for scenario

Number of Nodes	**Number of Packet Loss**	**Delivered Packet Ratio**	**Packet Loss Ratio**	**Average throughput**	**End to End Delay**
10	1456	90.78	9.22	144.45	309.51
25	692	93.90	6.09	100.57	147.84
50	988	91.55	8.44	104.53	231.83
75	599	90.91	9.08	58	166.27
100	461	91.27	8.73	46	171.47

Table 3. AODV Evaluation for scenario

Number of Nodes	Number of Packet Loss	Delivered Packet Ratio	Packet Loss Ratio	Average throughput	End to End Delay
10	4701	77.85	22.15	182.48	301.27
25	1595	90.59	6.09	142.8	160.78
50	1881	86.91	8.44	120.68	138.84
75	1234	89.6	11.08	98.89	156.76
100	1175	88.92	8.73	46	157.31

All results are analyzed and briefed in form of graphs given below. These graphs are found very helpful in statistical analysis of these routing protocols. The required graphs were saved as the bitmap image for statistical analysis. In this figure, we estimate the average throughput for all three routing protocols namely AODV and DSDV

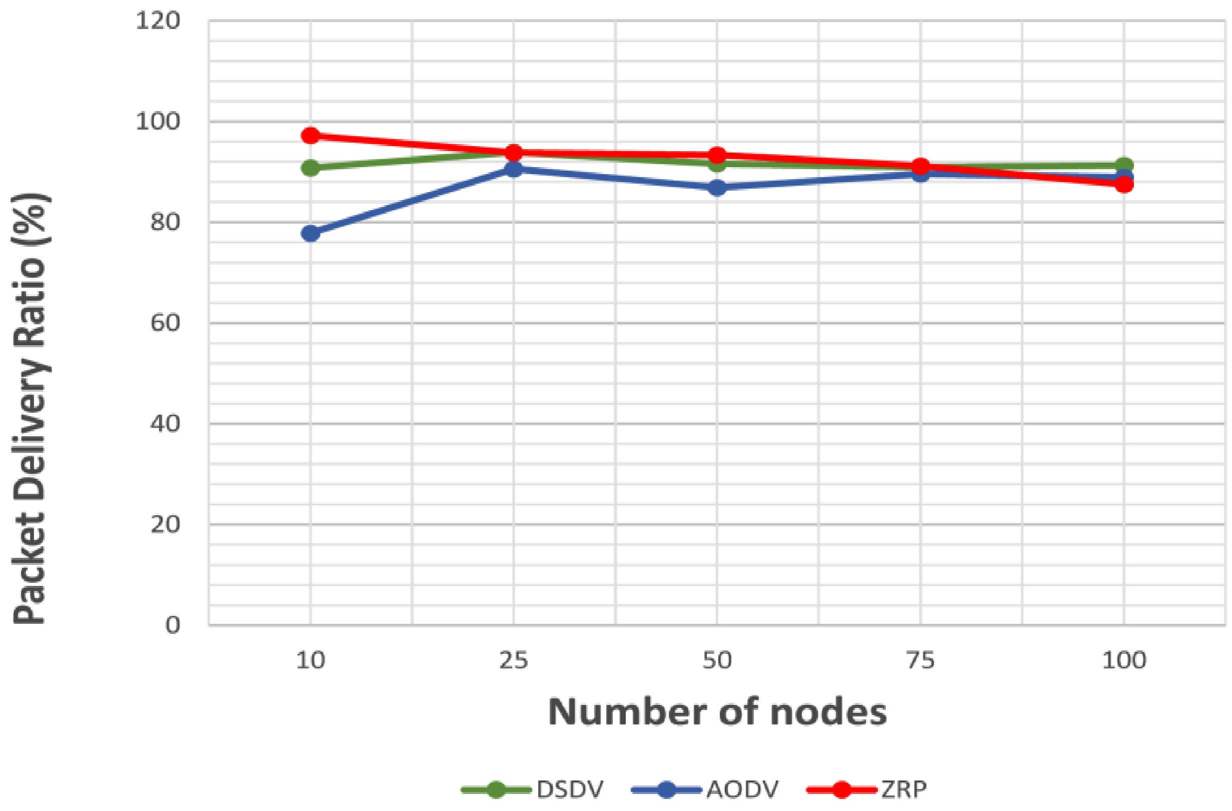


Figure A: Packet Delivery Ratio for DSDV, AODV

Thus, in Figure A we see the performance illustration of Packet Delivery Ratio depending on the number of nodes, we notice that the value of Packet Delivery Ratio in DSDV remains constant and AODV shows variation. AODV performance dropped as number of nodes increase because more packets dropped due to link breaks. When we analyse where these lost packets are in AODV, we notice that AODV has not only more packets in buffers waiting for a route; but also, more packets are lost because they were sent following old routes. So AODV, suffers in part from its lack of periodic update information but maintaining reasonably good delivery ratio.

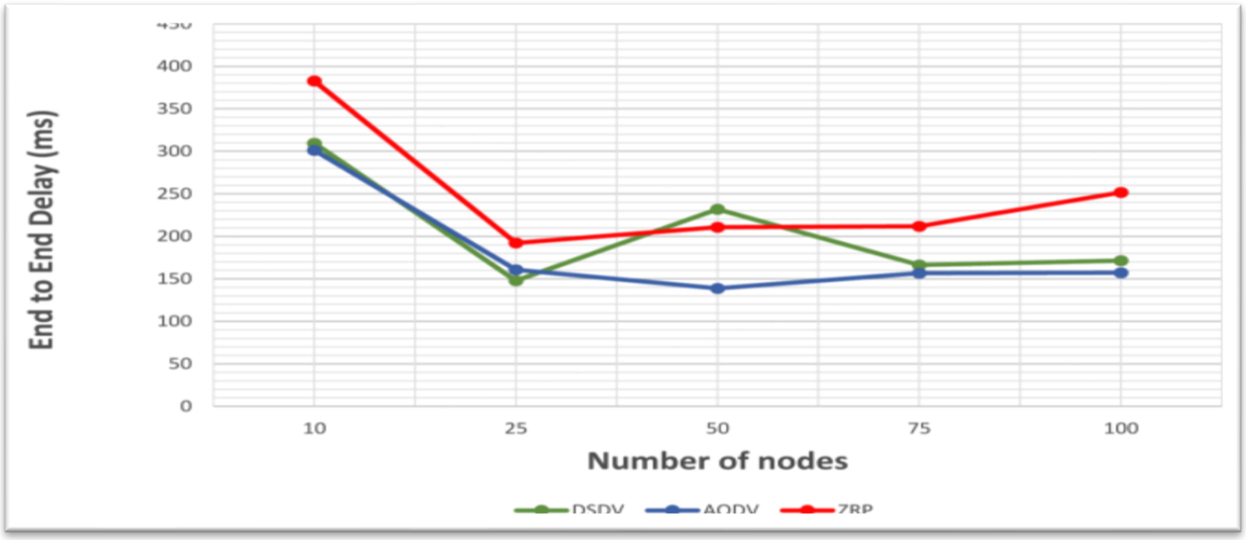


Figure B: End to End Delay for DSDV, AODV

This graph demonstrates the simulation results of End-to-End delay depending on the number of nodes. AODV didn't produce so much delay when the number of nodes increased. It performs better than the other protocol. In addition, it shows that, the AODV protocol improved the DSDV when the number of nodes is over 50. The End-to-End Delay of AODV is less because it has reduced routing overhead and queuing delay. This attribute can be explained by the fact that DSDV is a proactive routing protocol and in these types of protocols the path to a destination is immediately available. Furthermore, DSDV routing protocol tries to drop the packets, if it is not possible to deliver them which means less delay.

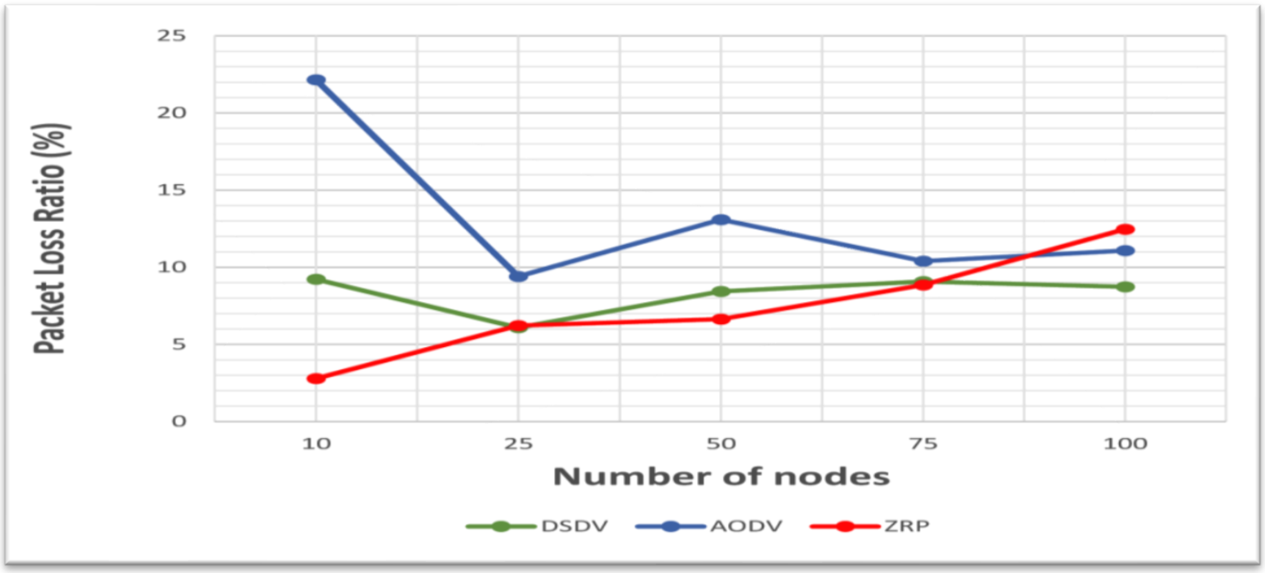


Figure C: Packet Loss Ratio for DSDV and AODV

With increasing number of sensor nodes AODV shows worst performance. AODV seems to be more sensitive to the effect of the density of nodes. Once more AODV suffers from not always up-to-date information. Since proactive routing maintains information that is immediately available, the Packet Loss Ratio before sending a packet is minimal in cost. So, overall, we can say that DSDV is the most preferred routing protocol for larger networks.

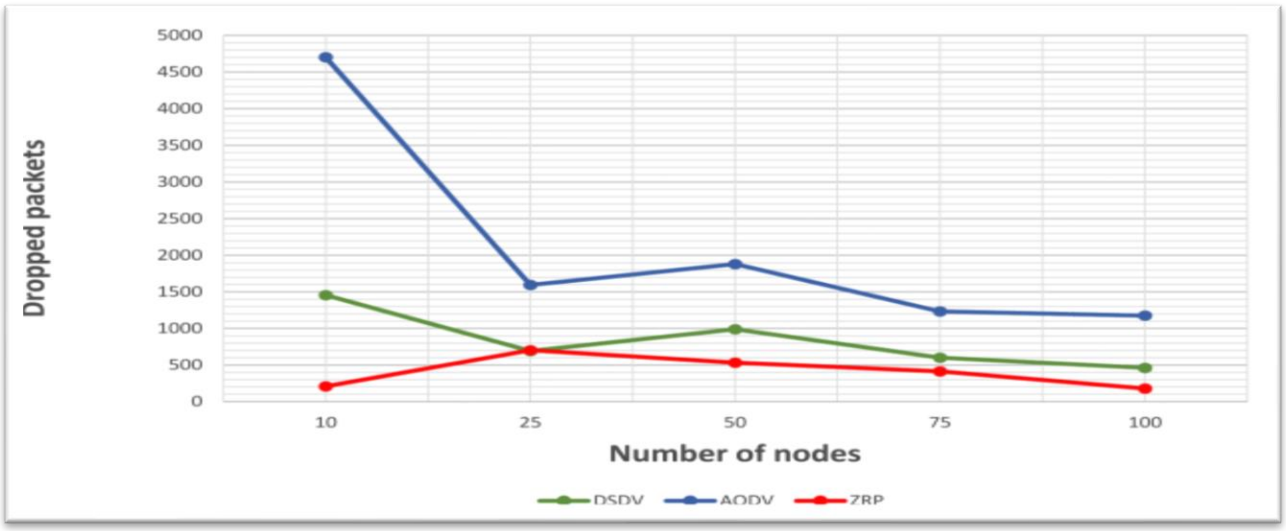


Figure D: Dropped Packets for DSDV and AODV

Here, we notice that as the number of nodes increases, the End-to-End delay becomes very high and increases with AODV. So overall, we can say that AODV is the most preferred routing protocol for larger networks in terms of End to End delay because its these parameters decrease more sharply than DSDV. Again, we conclude that with increasing number of sensor nodes AODV shows worst-performance. For 50 nodes AODV shows maximum dropped packet loss. Therefore, comparing to AODV and DSDV, they have equal choices/chances.

CONCLUSION:

Hence, wireless routing protocols AODV and DSDV were implemented successfully.