Design of Convolutional Encoder and Viterbi Decoder using MATLAB

Kanchana Katta

Dept. of Electronics and Communication Engineering, Don Bosco College of Engineering and Technology, Assam Don Bosco University, Guwahati, India

kanch414@gmail.com

ABSTRACT

The main aim of any communication schemes is to provide error-free data transmission .Error control coding is a method to detect and possibly correct errors by introducing redundancy to the stream of bits to be sent to the channel. This coding has the usefulness that it allows us to increase the rate at which information may be transmitted over a channel while maintaining a fixed error rate. The Convolution (Channel) Encoder will add bits to the message bits to be transmitted systematically. After passing through the channel, at the receiver end the original message sequence is obtained from the received data using Viterbi decoder. It implements Viterbi Algorithm, which is a maximum likelihood algorithm, based on the minimum cumulative hamming distance it decides the optimal trellis path that is most likely followed at the encoder. Convolutional encoding and Viterbi decoding are error correction techniques widely used in Communication systems to improve the bit error rate (BER) performance. In this paper, I present the convolution encoder and Viterbi decoder for constraint length 4 and bit rate 1/2. In addition, a Viterbi decoder is developed in MATLAB.

Keywords — Convolution Encoder, Viterbi decoder, trellis structure

1. INTRODUCTION

Coding theory deals with transmission of data over noisy channels by adopting various source and channel coding/decoding schemes. The convolution codes are widely used as forward error correction codes. The main decoding strategy for convolutional codes is based on the widely used Viterbi algorithm.

Convolutional encoding with Viterbi decoding is a powerful Forward Error Correction technique that is particularly suited to a channel in which the transmitted signal is corrupted mainly by Additive White Gaussian Noise (AWGN). The purpose of forward error correction (FEC) is to improve the capacity of a channel by adding some carefully designed redundant information to the data being transmitted through the channel [1]. The Viterbi algorithm essentially performs maximum likelihood decoding to correct the errors in received data, which are caused by the channel noise. Hence, minimize the bit error rate (BER) to improve the performance.

Viterbi decoding has the advantage that it has a fixed decoding time and it is well suited to hardware decoder implementation. The requirements for the Viterbi decoder, which is a processor that implements the Viterbi algorithm, depend on the application in which it is used. Viterbi algorithm is the most resource consuming, and efficient [2].

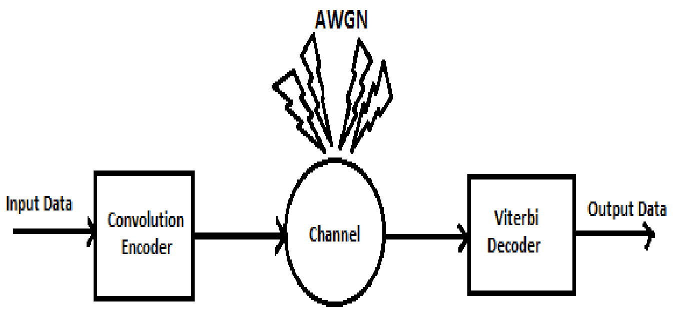


Fig-1: Block diagram of digital communication system using convolution encoder and Viterbi Decoder

2. CONVOLUTION ENCODER

Convolutional coding is a bit-level encoding technique. Convolutional codes are used in applications that require good performance with low implementation cost. Using convolutional codes a continuous sequence of information bits is mapped into a continuous sequence of encoder output bits. The encoded bits depend not only on current k input bits but also on past input bits. This mapping is highly systematic so that decoding is possible. As compared with the block codes, convolutional codes have a larger coding gain.

Encoding of convolutional codes can be accomplished using simple registers. The convolutional codes are denoted by (n,k,L),where n is number of output bits(coded),k is the number of input bits (uncoded), and L is the code memory depth, which represents the number of register stages. The number of the registers used in the encoding process is called the constraint length and it is indicated with C= (L+1).The efficiency or data rate of a convolutional code is measured by the ratio of the number of bits in the input (k) and the number of bits in the output (n), therefore Bit Rate: r = k/ n.

To convolutionally encode the data, start with L memory elements shift register, each element holding one input bit. The encoder has modulo-2 adders, and n generator polynomials one for each adder. The Fig-2 illustrates a (2, 1) convolutional encoder with constraint length C=4.

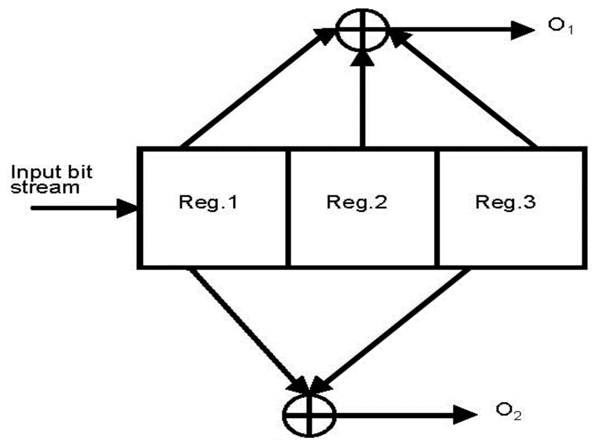


Fig-2: Block diagram of convolution encoder

The selection of bits is to be added (uses XOR operation) to produce the output bits is called generator polynomial. The generator polynomials O_1 and O_2 are of (111, 101) i.e. O_1 = mod2 (Reg.1 + Reg.2+ Reg.3), O_2 = mod2 (Reg.1 + Reg.3).If the initial contents of the register are all zeros, for the input bit stream m=1 1 0 1 1, the output code word sequence obtained is 11 01 01 00 01 01 11.

TABLE- I

STATE TABLE FOR THE CONVOLUTIONALLY ENCODING A MESSAGE SEQUENCE

Input bit m	Register contents	State at time t_i	State at time t_{i+1}	Output code word	
				O_1	O_2
	000	00	00		
1	100	00	10	1	1
1	110	10	11	0	1
0	011	11	01	0	1
1	101	01	10	0	0
1	110	10	11	0	1
0	011	11	01	0	1
0	001	01	00	1	1

Convolution encoding can be done using the following methods

a. State diagram
b. Trellis diagram

a) State Diagram:

One way to represent simple encoders is with a state diagram. The state of an encoder is defined as its shift register contents. Each new input bit results in a new state. Therefore for one bit entering the encoder there are 2 possible branches for every state. To make easy for tracking the transition two different types of line are used. Solid line (——) represents the transition when the input bit is '1' and dotted line (------) represents the transition when the input bit is '0'. The diagram shown in Fig-3 illustrates all the state transitions that are possible for the encoder.

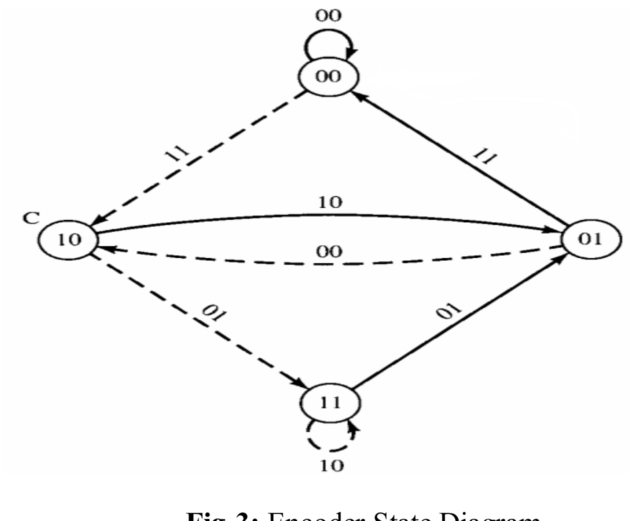


Fig-3: Encoder State Diagram

b) Trellis Diagram:

A trellis diagram is a kind of state diagram and mainly it can be used for decoding of convolutional codes. The detection of the original stream can be described as finding the most probable path through the trellis.

In the trellis diagram, each node specifies an individual state at a given time and indicates a possible pattern of recently received data bits. Each branch indicates the transition to a new state at the next timing cycle. When two paths enter the same state, the one having the best metric is chosen: this path is called the surviving path. In the decoder the selection of surviving paths are performed by all the states. Based on the received coded bits we can choose the more likely path and ignore the least likely paths. The decoding complexity can reduce by ignoring the minimum likely paths.

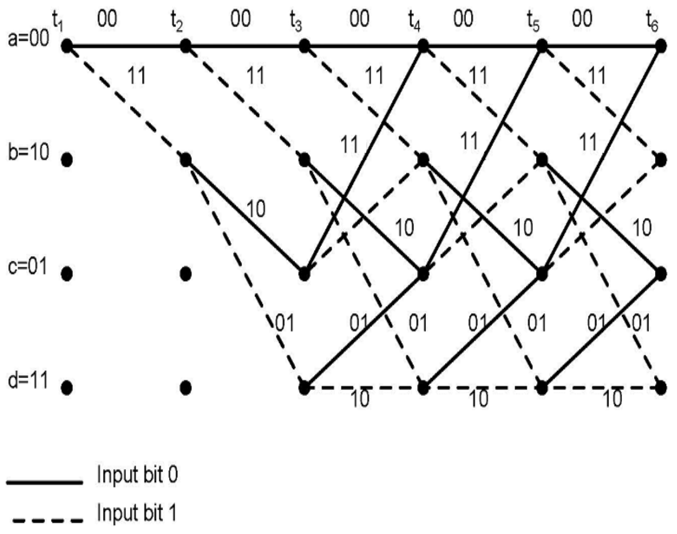


Fig-4: Encoder trellis structure for C=4 and r=1/2

Fig-4 shows the encoder trellis, the branch word seen on the encoder trellis branches characterize the convolutional encoder. These encoder branch words are the code symbols that would expected to come from the encoder output as a result of each of the state transitions.

As the code symbols are received, each branch of the decoder trellis is labeled with a metric of similarity between the received code symbol and each of the branches for that time interval. Here we can see how the decoding of the surviving branch is facilitated by having drawn the trellis branches with solid lines for the input 0's and dashed lines for the input 1's.

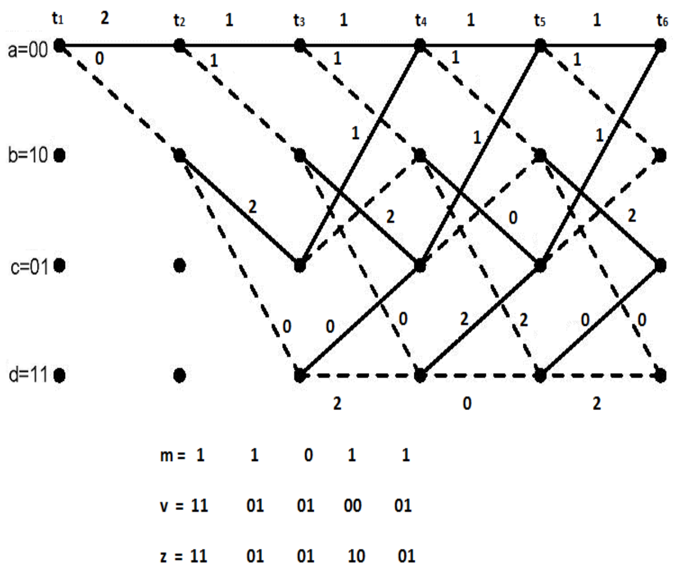


Fig-5: Decoder trellis structure for C=4 and r=1/2

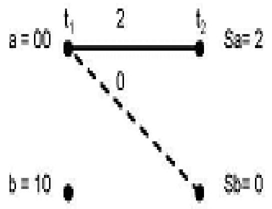
Labeling procedure is done like this. From the received sequence Z, we see that the code symbol received at time t_1 is 11.In order to label the decoder branches at time t_i with the appropriate hamming distance metric shown in fig.5 encoder trellis. Here we see that a state 00 $\rightarrow$ 00 transition yields an output branch word of 00 , but we received 11.Therefore ,on the decoder trellis we label the stat 00 $\rightarrow$ 00 transition with Hamming distance between them, namely 2.Looking at the encoder trellis again ,we see that a state 00 $\rightarrow$ 10 transition yields an output branch word of 11,which corresponds exactly with the code symbols we received at time t_1.Therefore,on the decoder trellis ,we label the stat 00 $\rightarrow$ 10 transition with Hamming distance 0 and so on.

Thus, the metric entered on a decoder trellis branch represents the difference between what was received and what should have been received, had the branch word been associated with that branch been transmitted. In effect, these metrics describe a correlation like measure between a received branch word and each of the candidate branch words. We continue labeling the decoder trellis branches in this way as the symbols are received at each time t_i. Thus the most likely (minimum distance) path through the trellis can be found that gives the decoded output.

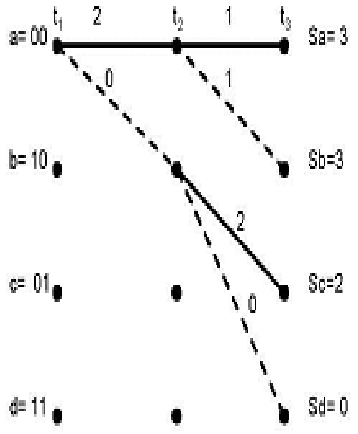
At each time t_i, there are 2^C-1 states in the trellis, where C is the constraint length, and each state can be entered by means of two paths. Viterbi decoding consists of computing the metrics for the two paths entering each state and eliminating one of them. This computation is done for each of the 2^C-1 states or nodes at time t_i, and then the decoder moves to time t_{i+1} and repeats the process. At a given time, the winning path metric for each state is designated as the state metric for that state at that time.

Fig.6 from (a-e) shows the state metric comparison at each state, the decoding process continues for all the state transitions and to make decisions on the input data bits by eliminating all paths but one. In the decoding process at time t_1 , the received code symbol are 11.From the state 00 the only possible transitions are to state 00 or 10.State 00 →00 transition has the branch metric 2; state 00 →10 transition has branch metric 0.

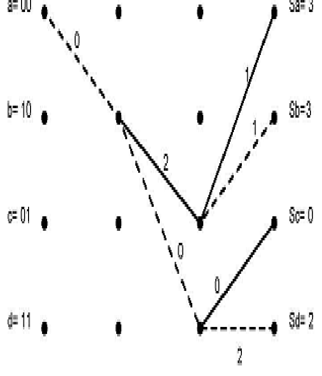
At time t_2 there are two possible branches leaving each state metric S_a=2 and S_b=0 as shown in Fig-6(a).Likewise for all other states it will continue the same process, we can see in Fig-6(e) the state metric S_d=1 is the maximum likelihood and the path metric branch word sequence is 11 01 01 10 01.



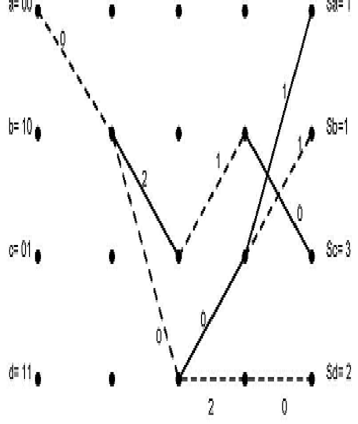
a. Survivors at t_2



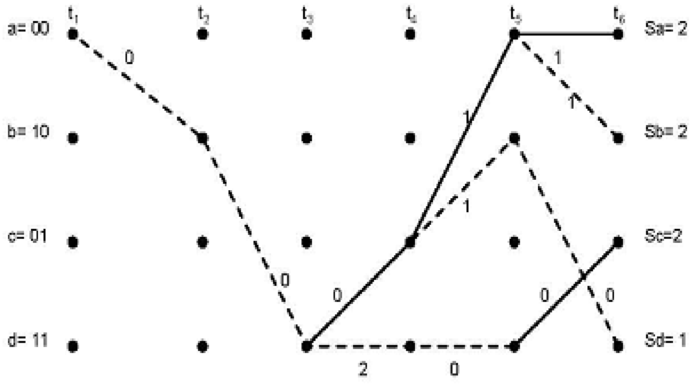
b. Survivors at t_3



c. Survivors at t_4



d. Survivors at t_5



e. Survivors at t_6

Fig-6: State Metric comparison at each state

3. VITERBI DECODER

In 1967 by Andrew Viterbi was proposed the Viterbi algorithm (VA) and is used to decoding a bit stream that has been encoded using FEC code [3]. Viterbi algorithm can be explained by a trellis diagram it requires which comprises of minimum path and minimum distance Calculation and retracing the path. Fig-7 shows the block diagram of the Viterbi decoder.

It consists of following blocks

a. Branch Metric Unit (BMU)
b. Path metric calculation
c. Add Compare and Select Unit (ACS)
d. Trace Back Unit (TBU)

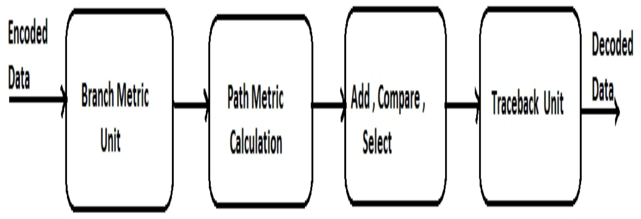


Fig-7: Block diagram of the Viterbi decoder

From the encoder output through the channel the BMU receives input data and computes a metric for each state and each input bit. There are two types of methods to calculate the metric. The metric which is used to calculate the

Hard-decision encoded data is the Hamming distance and the metric is the Euclidian distance for Soft-decision encoded data. The metric is calculated for the entire path. The ACS unit is based on minimum distance calculations that are obtained from the previous row values. The Trace-back unit restore maximum likelihood path from the decisions made by BMU. This is the final stage of the Viterbi decoder where the input that was transmitted by using the convolution encoder is once again retrieved [4].

When the convolutional code bits encoded from input bits are modulated in to respective waveforms for transmission over a medium that introduces attenuation, distortion, interference,noise,etc.the received waveforms become uncertain" in their shapes. One of the frequently applied criteria is the maximum-likelihood decoding (MLD) rule under which the probability of code word estimate error is minimized subject to an equiprobable prior on the transmitted code words. The Viterbi decoding flow chart is given in Fig-8.

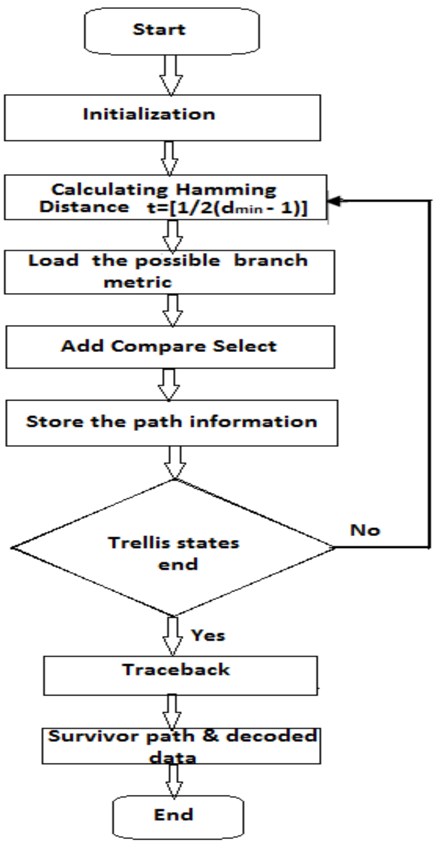

Start
Initialization
Calculating Hamming Distance t=[1/2(dmin - 1)]
Load the possible branch metric
Add Compare Select
Store the path information
Trellis states end
No
Yes
Traceback
Survivor path & decoded data
End

Fig-8: Flow chart of Viterbi Decoding

Soft Decisions versus Hard Decisions

The primary difference between the hard decision and soft decision Viterbi algorithm is that the soft decision algorithm cannot use hamming distance metric because of its limited resolution. A distance metric with needed resolution is the Euclidean distance.

To compare the performance of coded binary systems on a AWGN channel when the decoder performs either hard or soft decisions, the energy E_c per coded bit is fixed and $P_b(\varepsilon)$ is plotted versus ε of the hard decision BSC model where ε=1/2erfc√ (Ec/N0) .

For soft decisions the expression Pw (ε) =1/2erfc√ (wE_c/N_0) is used for the probability that the ML encoder makes a detour with weight w from the correct path. The advantage of using soft decision decoding is to provide decoder with more information, which decoder then use for recovering the message sequence [5]. It provides better error performance than hard decision type Viterbi decoding.

4. RESULTS

Fig-9 shows that the bit error rate (BER) as a function of the signal to noise ratio (SNR), from this graph the performance improvement of soft decision Viterbi as compared with hard decision Viterbi is approximately 2db.

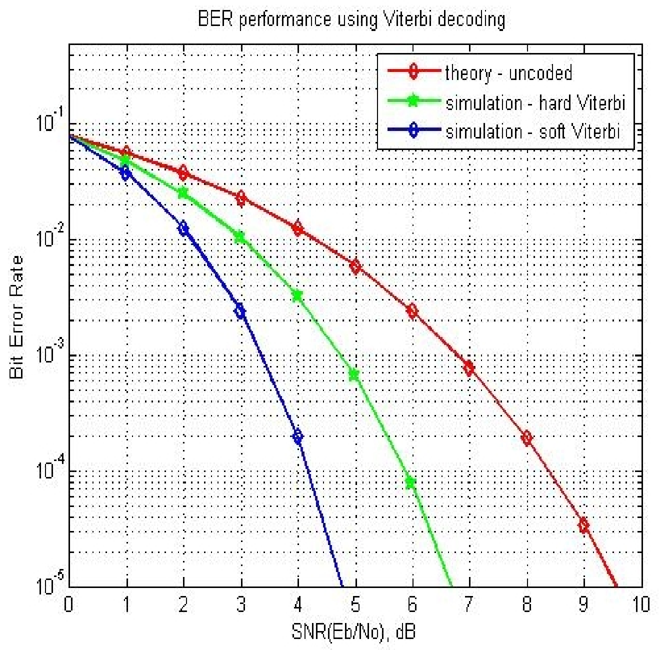


Fig-: 9 BER graph for hard decision (HD) and soft decision (SD) Viterbi decoder with C=4 and R=1/2.

5. CONCLUSION

The design of a convolutional encoder with a Viterbi decoder that can encode a bit stream of digital information and outputs a code word that has a capability to be transmitted to the destination and then decoded .The encoder was designed with constraint length 4 and rate 1/2.The Viterbi decoder design had been driven in such a way that it would calculate the decoding path with the minimum metric to be passed to the decoder output port.

Convolutional encoder and Viterbi decoder design has been successfully done using MATLAB and results obtained in terms of BER vs SNR.The motivation of this paper is to help the beginners to understand working of Viterbi algorithm those who wants to works on Viterbi decoder.

REFERENCES

[1] Jinjin H, Zhongfeng Wang, Zhiqiang Cui, LiLi, “Towards an Optimal Trade-off of Viterbi Decoder Design”, Circuits and Systems, ISCAS-2009. IEEE International Symposium, 24-27 May 2009 pp. 3030 – 3033.

[2] Swati Gupta, Rajesh Mehra,”FPGA Implementation of Viterbi Decoder using Trace back Architecture”, International Journal of Engineering Trends and Technology- May to June Issue 2011.

[3] Viterbi.A.J, "Convolution codes and their Performance in communication systems," IEEE Transaction on Communications, vol.com-19, pp. 751 to 771, October 1971.

[4] Mahe Jabeen, Salma Khan,” Design of Convolution Encoder and Reconfigurable Viterbi Decoder”, International Journal of Engineering and Science ISSN: 2278-4721, Vol. 1, Issue 3(Sept2012), PP 15-21.

[5] Hiral Pujara, Pankaj Prajapati,” RTL Implementation of ViterbiDecoder using VHDL” IOSR Journal of VLSI and Signal Processing (IOSR-JVSP)Volume 2, Issue 1(Mar. –Apr.2013), PP 65-71e-ISSN: 2319 –4200, p-ISSN No. : 2319 –4197