

Forward Error Correction Technique using Convolution Encoder & Viterbi Decoder

Awantika Vishwakarma¹, Pankaj Gulhane²

Dept. of VLSI & Embeded System, Electronics & tele Communication, Disha Institute of Management & Technology^{1,2}
awantika.vish@gmail.com¹, gulhanep@gmail.com²

Abstract- All communication channel are heart to Additive White Gaussian Noise (AWGN) around the environment. Error Correction technique enhances the capacity by adding redundant(unneeded) information for the data transmission. During the transmission of message, the data can corrupted due to plenty of disturbances in the communication channel. So it is necessary for the decoder tool to also have a function of correcting the error which may occur. So basically the main aim of any communication scheme is to provide error-free data communication. Error control coding is a method to detect and possibly correct errors by introducing redundancy to the bunch of bits which is to be sent to the channel. This coding has the utility that it allows us to boost the rate at which information may be transmitted over a channel while maintaining a fixed error rate. Convolutional encoding is a forward error correction technique which is used for correction of errors at the receiver side. Convolutional codes protect information by adding redundant bits to the binary data. Viterbi decoding is the technique for decoding the Convolutional codes. The Viterbi algorithm estimates the maximum likelihood path through a trellis based on received symbols. In this paper we are concluding about soft decision viterbi decoding for code rate of $\frac{1}{2}$ & $\frac{1}{3}$ and $m=2$ to 6 as well as we are finding the unbound viterbi decoding ,both for 4-bit input.

Index term : Convolution encoder, Viterbi decoder, Trellis structure, code rate, constraint length.

1. INTRODUCTION

Communication system transmits data from source to transmitter through a channel or medium such as wired or wireless. The reliability of received data depends on the channel medium and external noise and this noise creates interference to the signal and introduces errors in transmitted data. Shannon through his coding theorem showed that reliable transmission could be achieved only if data rate is less than that of channel capacity. The theorem shows that a sequence of codes of rate less than the channel capacity have the capability as the code length goes to infinity [1]. Error detection and correction can be achieved by adding redundant symbols to the original data called as error correction and correction codes (ECCs). Without ECCs data need to retransmitted if it could detect there is an error in the received data. ECC are also called as for error correction (FEC) as we can correct bits without retransmission. Retransmission adds delay, cost and wastes system throughput. ECCs are really helpful for long distance one way communications such as deep space communication or satellite communication. They also have application in wireless communication and storage devices[2]. The basic digital communication system is shown in figure.[3]

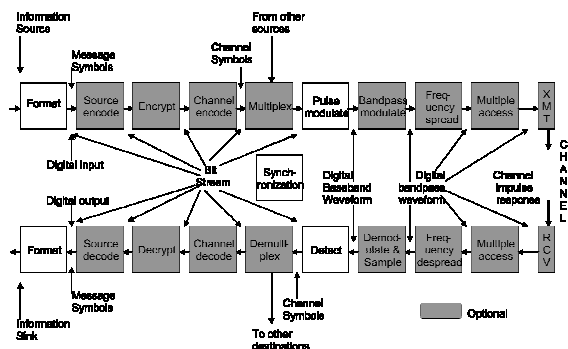


Figure 1: Representation of Digital Communication System

2. FORWARD ERROR CORRECTION

There are several ways of classifying the forward error correction codes as per different characteristics [1].

1. Linear vs. Nonlinear- Linear codes are those in which the sum of any two valid code words is also a valid code word. In case of nonlinear code the above statement is not always true.

2. Cyclic vs. Non-Cyclic - Cyclic code word are those in which shifting of any valid code word is also a valid code word. In case of non-circular code word the above statement is not always true.

3. Systematic vs. Nonsystematic- Systematic codes are those in which the actual information appears unaltered in the encoded data and redundant bits are added for detection and correction of error. In non-systematic code the actual message does not appear in its original form in the code rather there exists one mapping method from the data word to code word and vice versa.

4. Block vs. convolutional -The block codes are those in which one block of message is transformed into on block of code. In this case no memory is required. In case of convolutional code a sequence of message is converted into a sequence of code. Hence encoder requires memory as present code is combination of present and past message.

5. Binary vs. Non binary -Binary codes are those in which error detection and correction is done on binary information i.e. on bits. Hence after the error is located, correction means only flipping the bit found in error. In Non-binary code error detection and corrections are done on symbols, symbols may be binary though. Hence both the error location and magnitude is required to correct the symbol in error.

3. CONVOLUTION CODING

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 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.[6]

4. CONVOLUTION ENCODER

The convolutional encoder maps a continuous information bit stream into a continuous bit stream of encoder output. The convolutional encoder is a finite state machine, which is a machine having memory of past inputs and also having a finite number of different states. The number of output

bits depends on the number of modulo 2-adders used with the shift registers.[5]

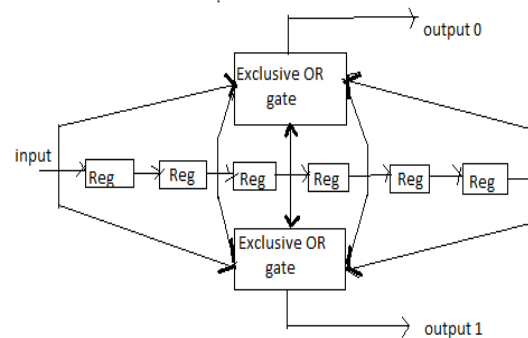


Figure 2: A convolution Encoder

4.1 Convolution Encoder parameters

Convolutional codes are commonly specified by the three parameters (n, k, m) , where n = number of output bits, k = number of input bits and, m = number of shift registers.[5]

Commonly k and n parameters range from 1 to 8, m from 2 to 10, and the code rate from 1/8 to 7/8 except for deep space applications where code rates as low as 1/100 or even longer can be employed.[4] The convolutional codes discussed here will be referred as (n, k, m) codes. [1]

Passing the information sequence to be transmitted through a linear finite shift register generates a Convolutional code. The shift register consists of k bit stages and n linear algebraic function generators. The contents of shift register are multiplied by respective term in generator matrix and are then XORed together to generate respective generator Polynomials.[8]

4.2 Generator polynomial

Generator Polynomial is defined by-

$$g^{(i)}(D) = g_0^{(i)} + g_1^{(i)}(D) + g_2^{(i)}(D^2) + \dots + g_m^{(i)}(D^m)$$
 Where, D = unit delay variable

m = number of stages of shift registers.

The encoder connections are characterized by the term generator polynomial (g). For producing the output bits the selection of which bits (in the memory registers) are to be added (using modulo-2 adders) is called the generator polynomial for that output bit. Various choices are available for

polynomials for any m order code. It is again a task to find good polynomials which are normally found by trial and error method using computer simulations.[1][5]

4.3 Example : (2,1,3) Convolution encoder

For understanding the working of a convolutional encoder and the forward error correction technique, we have taken the following assumptions.

- (a) We are using a (2, 1, 3) convolutional encoder.
- (b) A 3-bit input sequence [1 0 1] is specified bits.
- (c) 2 generator polynomials [1 1 1] and [1 0 1] are used.

Figure 3 is a (2, 1, 3) convolutional encoder. This encoder is going to be used to encode the 3-bit input sequence [1 0 1] with the two generator polynomials specified by the bits [1 1 1] and [1 0 1]. u_1 represents the input bit, and v_1 and v_2 represent the output bits 1 and 2 respectively. u_0 and u_{-1} represent the initial state of the memory registers which are initially set to zero.[1][4][2]

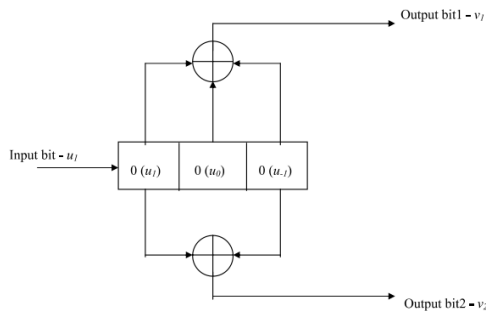


Figure 3: A (2,1,3) convolution encoder

4.4 State representation of convolution encoder

The convolutional encoder can use a look-up table, otherwise called the state transition table to do the encoding. The state transition table consists of four items:[6][7]

- a) The input bit.
- b) The state of the encoder, which is one of the 4 possible states (00 01 10 11) for the (2, 1, 3) convolutional encoder.
- c) The output bits, which for the (2, 1, 3) convolutional encoder are: 00 01 10 11, since only two bits are output.
- d) The output state which is the input state for the next bit.

Input Bit	Input State	Output Bits	Output State
0	00	00	00
1	00	11	10
0	01	11	00
1	01	00	10
0	10	10	01
1	10	01	11
0	11	01	01
1	11	10	11

Table 1: State transition table for the (2, 1, 3) convolutional encoder

4.5 State Diagram Representation

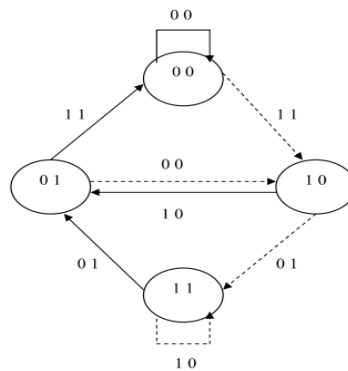


Figure 4 : State diagram for the (2, 1, 3) convolutional encoder [1]

5. VITERBI DECODER

Viterbi decoders work on Viterbi algorithm to decode the encoded data. The Viterbi decoding algorithm was discovered and analyzed by Viterbi in 1967. The Viterbi algorithm essentially performs maximum likelihood decoding; however, it reduces the computational load by taking advantage of the special structure in the code trellis [8].

5.1. Viterbi Decoding Technique

The Viterbi decoder examines an entire received sequence of a given length. The decoder computes a metric for each path and makes a decision based on this metric. All paths are followed until two paths

converge on one node. When two paths enter the same state, the one having the best metric is chosen; this path is called the surviving path. The early rejection of the unlikely paths reduces the decoding Complexity[4][7][8]

5.2. Block Diagram of Viterbi Decoder

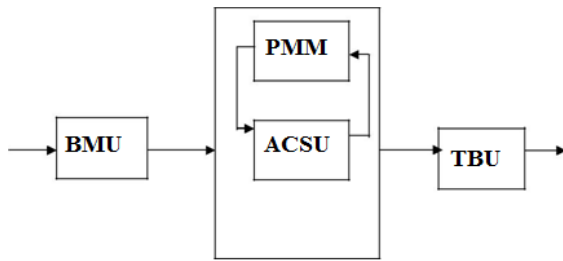


Figure 5 : Block diagram of Viterbi Decoder

1) Branch metric unit (BMU): From the encoder output through the channel the BMU receives input data and computes a metric for each state and each input bit. BMU compares the received data bits are compared with the expected or idel outputs and counts the number of differing bits [8].

2) Path Metric Unit : The path metric unit includes the Add compare and select unit. The partial path metrics are compared by the comparator and branch metric is selected by the selector. That means the selector selects the smaller value.[9]

3) Add-Compare Select unit (ACSU):The Add-Compare Select Unit (ACSU) adds the Branch Metrics (BM) to the partial Path Metrics (PM) to obtain new path metric. When two paths enter the same state, it compares the new PMs and the one having minimum metric is chosen , this path is called survivor path. The selection for survivor path is done for all states. It then stores the selected PMs in the

Path Metric Memory (PMM). The PM of the survivor path of each state is updated and stored back into the PMM [8].

6. SOFT DECISION VITERBI DECODING

Also referred to as the soft input Viterbi decoding technique, this uses a path metric called the Euclidean Distance metric, to determine the survivor paths as we move through the trellis. The soft decision Viterbi decoder discussed in this report uses a 3-bit quantizer to quantize the received channel data stream. A Viterbi decoder with soft decision data inputs quantized to three or four bits of precision can perform about 2 dB better than one working with hard decision inputs.

6.1 Trellis explanation

Based on the example considered, of the encoded 3-bit input stream [1 0 1] trellis are shown in the figures[1].

The corrupted, and quantized data bit stream at the input of the soft decision Viterbi decoder is assumed as [3-4 -43 33 -43 -4-4].

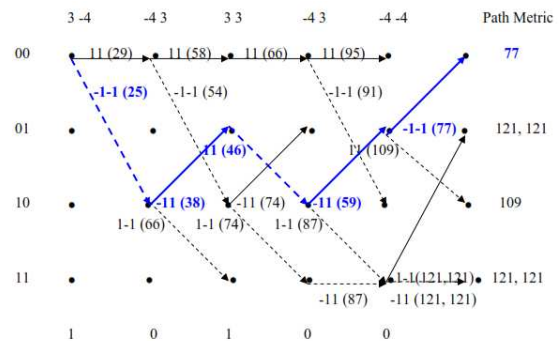


Figure 6 : Decoded sequence 10100 for the noisy encoded bit stream 3-4 -43 33 -43 -4 -4.

6.2 Soft decision decoding model

Soft-Decision Decoding

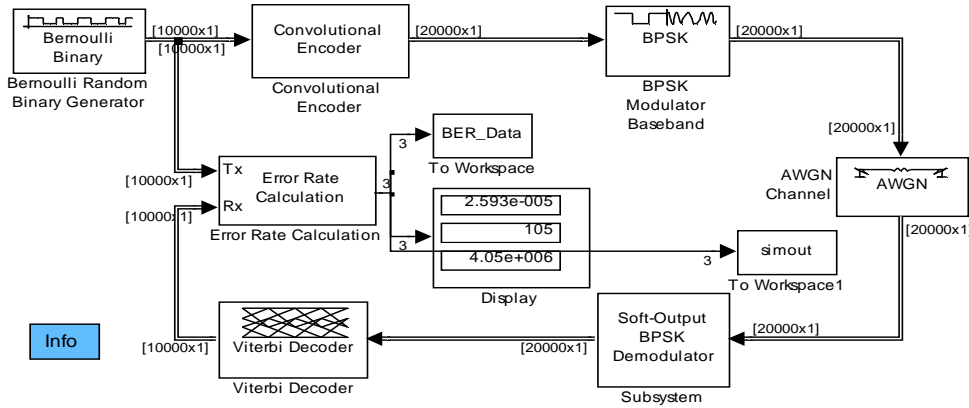


Figure 7 : Soft decision viterbi decoding model [9]

7. 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 a 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.

8. ACKNOWLEDGEMENT

The pleasure, the achievement, the glory, the satisfaction and the construction of my paper cannot be thought of without the few, who apart from their regular schedule spared their valuable time, I owe a debt of gratitude to my project guide Asst. Prof. PankajGulhane, Assistant Prof. and of Electronics and Telecommunication Engineering Department for providing me with an opportunity to start this project. With his timely advice, constructive criticism and supervision he was a real source of inspiration for me.

9. RESULTS

We have calculated BER for $E_b/N_0 = 2$ to 7 & found simulated result using MATLAB14a. We have taken 4 input bits 1000 & generator polynomials

[1,0,0,1,1,1,1;1,1,0,1,1,0,1]. We took code rate of $1/2$ & $m=6$.

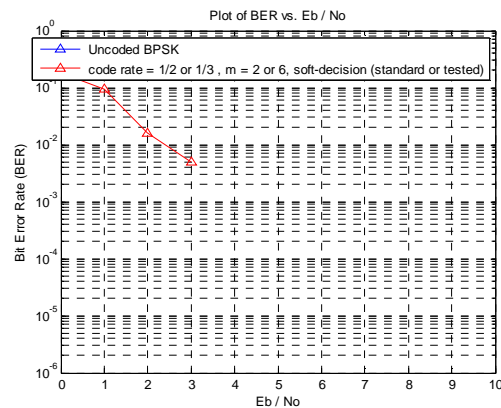


Figure 8: BER vs E_b/N_0 (SNR) for soft decision decoding for rate $1/2$ & $m=6$

We calculated BER for the unbound result for viterbi decoding for SNR 1 to 12 & input bits 1000.

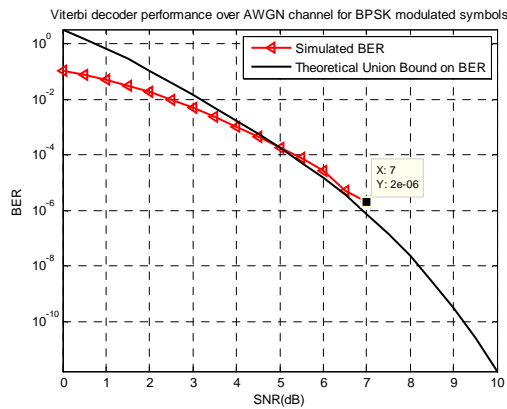


Figure 9 : BER vs Eb/No (SNR) for 4-bit input

REFERANCES

- [1] Haykin, Communication Systems, 3rd edition, John Wiley & Sons, New York, 1994.
- [2] J. G. Proakis, Digital Communications, 3rd edition, WCB/McGraw-Hill, Boston, Massachusetts, 1995.
- [3] B. Sklar, Digital Communications Fundamentals and Applications, 2nd edition, Prentice Hall, Upper Saddle River, New Jersey, 2001.
- [4] Himmat Kumawat, Sandhya Sharma, "An Implementation of a Forward Error Correction Technique using Convolution Encoding with Viterbi Decoding", International Journal of Soft Computing and Engineering (IJSCE), Volume-2, Issue-5, November 2012.
- [5] Sneha Bawane and V.V.Gohokar, "Simulation of convolution codes", IJRET: International Journal of Research in Engineering and Technology, Volume: 03 Issue: 03 Mar-2014.
- [6] Kanchana Katta, "Design of Convolutional Encoder and Viterbi Decoder using MATLAB", International Journal for Research in Emerging Science and Technology, Volume -, Issue -7, December-2014.
- [7] K. Padma Selvi and J. Julie Antony Roselin, "Design and Implementation of Convolution Encoder and Viterbi Decoder Using FPGA", International Journal of Innovative Research in Computer and Communication Engineering, Vol.3, Special Issue 1, February 2015.
- [8] Ashima Sood, Nagendra Sah, "Implementation of forward error correction technique using Convolutional Encoding with Viterbi Decoding", International Journal of Engineering and Technical Research (IJETR), Volume-2, Issue-5, May 2014.
- [9] David J.C. MacKay, "Block codes", Error correcting code, May, 1997.
- [10] Matlab 2010 help documents.
- [i] Benedetto, Sergio, and Guido Montorsi, "Performance of Continuous and Blockwise Decoded Turbo Codes," IEEE Communications Letters, Vol. 1, May 1997, pp.77-79.
- [ii] Benedetto, S., G. Montorsi, D. Divsalar, and F. Pollara, "A Soft-Input Soft-Output Maximum A Posterior (MAP) Module to Decode Parallel and Serial Concatenated Codes," JPL TDA Progress Report, Vol. 42-127, November 1996.
- [iii] Clark, George C., Jr., and J. Bibb Cain, Error-Correction Coding for Digital Communications, New York, Plenum Press, 1981.
- [iv] Frenger, P., P. Orten, and T. Ottosson, "Convolution Codes with Optimum Distance Spectrum," IEEE Communications Letters, Vol. 3, November 1999, pp. 317-319.
- [v] Gitlin, Richard D., Jeremiah F. Hayes, and Stephen B. Weinstein, Data Communications Principles, New York, Plenum, 1992.
- [vi] Heller, Jerrold A., and Irwin Mark Jacobs, "Viterbi Decoding for Satellite and Space Communication," IEEE Transactions on Communication Technology, Vol. COM-19, October 1971, pp. 835-848.
- [vii] Viterbi, Andrew J., "An Intuitive Justification and a Simplified Implementation of the MAP Decoder for Convolutional Codes," IEEE Journal on Selected Areas in Communications, Vol. 16, February 1998, pp. 260-264.
- [viii] C. Weiss, C. Bettstetter, S. Riedel, "Code Construction and Decoding of Parallel Concatenated Tail-Biting Codes," IEEE Transactions on Information Theory, Vol. 47, No. 1, Jan. 2001, pp. 366-386.