International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249-8958 (Online), Volume-2 Issue-2, December 2012

My Viterbi vs MATLAB Viterbi

Vikas Gupta, Chanderkant Verma

Abstarct- The importance of convolutional codes is well established. They are widely used to encode digital data before transmission through noisy or error-prone communication channels to reduce occurrence of errors. To decode these convolutional code viterbi decoder is best choice. In this paper selection of viterbi decoder over conventional decoder is justified and a viterbi decoder is developed in MATLAB. This decoder is named My Viterbi and compared and analysed with the MATLAB viterbi decoder.

Key Words: Convolutional Encoder, My Viterbi, Viterbi Encoder, Packet Loss.

I. INTRODUCTION

According to Shannon, "the fundamental problem of communication is that of reproducing at one point either exactly or approximately a message selected at another point"[8]. In order to achieve this goal, channel coding tries to protect information from disturbances in a channel used as transportation medium. Due to these disturbances, transmitted and received signal are not the same. In order to restore the original information, the decoder exploits the received signal's redundancy, which was added by the channel code. There are several coding schemes and decoding algorithms that re-establish the original information with almost any desired accuracy. The achievable accuracy depends on the complexity of the coding scheme; the more complex, the more accurate the estimates. Clearly, the ultimate goal is to exactly reproduce the sent information, which is theoretically possible for communication systems that operate below the capacity bound defined by Shannon. Viterbi decoders are used to decode convolutional coding, which has been used in deep space communications as well as wireless communications. The Viterbi algorithm is to find a maximum likelihood sequence of state transitions, equivalently a path, in a trellis by assigning a transition metric to possible state transitions. A transition metric is called a branch metric, and the cumulative branch metrics along the path from the initial state to a given state is called the path metric of the state. When two or more paths end at the same state, the path with the smallest (or largest) path metric is selected as the most likely path. The survivor path obtained by back tracing in time corresponds to the decoded output. The hardware complexity of a Viterbi decoder is proportional to the number of states in the trellis, which is equal to the number of states of the corresponding convolutional encoder [1].

Manuscript published on 30 December 2012.
***** Correspondence Author (s)
Vikas Gupta, Ph. D Research Scholar, Pacific Academy of Higher Education And Research University, Udaipur Rajasthan, India
Dr. Chanderkant Verma, Associate Professor, Department of Computer Applications Kurushetra University, Kurushetra Haryana, India

II. SELECTION OF VITERBI DECODER OVER CONVENTIONAL DECODER

In the conventional ½ rate encoder, each input bit is XOR'ed with 2 different combinations of flip flops (FF1, FF2,FF 3 and FF6 for lower bits and FF2, FF3, FF5 and FF6 for upper bits) to produce two output bits. These bits are then interleaved and transmitted. This is the structure described in Figure 1. At the receiver, XORing alternate arriving bits with the corresponding set of flip flops (FF1, FF2, FF3 and FF6 for lower bits and FF2, FF3, FF5 and FF6 for upper bits), gives back the original message bit.

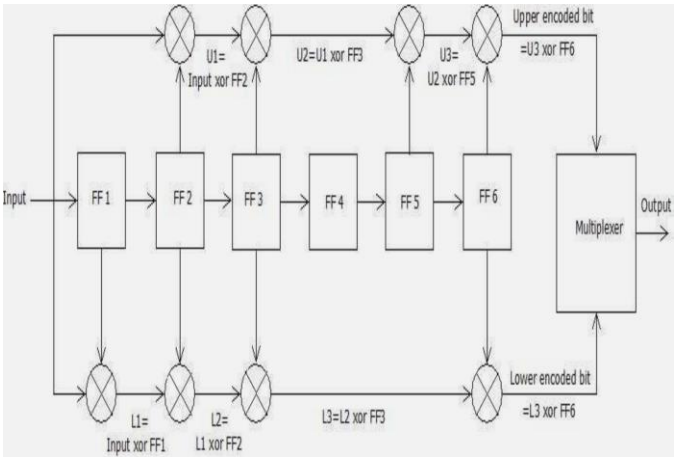

Figure 1: Conventional ½ Encoder

It is also understood that each upper received bit and each corresponding lower received bit were produced, at the transmitter, by the same original information bits. Assume that a correct copy of all previous information bits is available at the receiver. This assumption is bound to be correct at the beginning of a packet transmission, since all previous bits, at both the transmitter and receiver, are assumed to be zero. XORing the upper received bit with the 'appropriate' correct copies (as held at the receiver) of the previous information bits should produce the current original information bit. The term 'appropriate' refers to the information bits that were taken into account in the upper part of the convolutional encoder i.e. the 2nd, 3rd, 5th and 6th bit in this example. Similarly, XORing the lower received bit with the 'appropriate' correct receiver copies of the previous information bits should also produce the same correct information bit [2]. If there is no bit-error the same correct value for both the upper and lower decoded bits will be obtained at the receiver. Clearly, in this case, the process can then continue with the next received upper and lower bits. The diagram for the proposed design is provided in Figure 2.

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)

If the upper and lower received bits are found to be different at any stage, it can be concluded that a single bit-error has just occurred, either in the upper transmission or in the lower one.

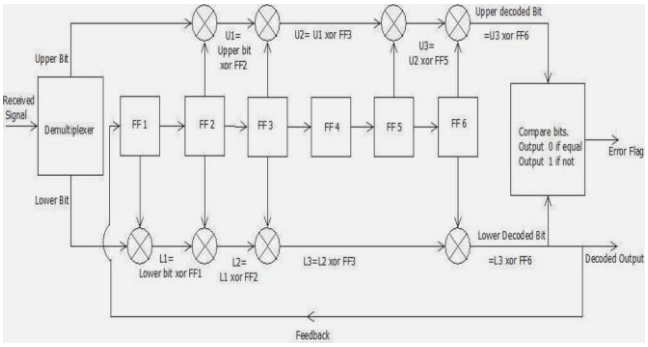

Figure 2: Conventional Deocder

On the other hand, if they are found to be equal, it cannot be concluded that there was no bit-error introduced in transmission. There may be two bit-errors, one in the upper transmission and one in the lower transmission. Therefore the occurrence of multiple bit-errors may not be detected straight away, and it is indeed possible for some bit-error patterns to pass completely undetected. As will be shown in the following chapters, it is expected that in these rare cases not even

III. DESIGN

3.1 The Transmitter Block

The transmitter block is designed with the following components

3.1.1 A Data Generating Source

Random binary data is generated using the 'randsrc' function. Six '0' bits are appended to the randomly generated data. These act as zero buffers. Since there are 6 shift registers in the convolutional encoder, it is necessary to run the encoder for an additional 6 time slots after the last data input for the last data bit to appear at the output of the encoder. For this reason, zero buffers are appended to the end of the data bits.

3.1.2. A Convolutional Encoder

A ½ K=7 (171 133) convolutional encoder is used. The six shift registers are initialized to 0. At each time slot a new data bit is accepted and XOR'ed with values of the corresponding shift registers as was shown in the Figure 1[11]. These connections represent $(171)_8$ and $(133)_8$ in binary form. In this way the value for the upper encoded bit and lower encoded bit is determined. The values of all registers are then shifted to the register on the right. The 2:1 multiplexer outputs the upper encoded bit and lower encoded bit in alternating sequences [3][4][5].

3.1.3 A QPSK Modulator

Before transmitting the signal it is modulated into QPSK signals with Gray encoding. This is implemented via predefined functions available in MATLAB®. An oversampling rate of 4 is used which results in 4 pulses for each data bit.

3.2 The Communications Channel

In order to simulate the effects of the communication channel Additive White Gaussian noise is added to the transmitted signal. The signal to noise ratio is reduced by 10×log10 (4) in to account for oversampling. It is further reduced by 10×log10 (1/code rate) so that the noise power is scaled to match coded symbol rate. The symbol to noise ratio is varied over different iterations.

3.3 The Receiver Block

The receiver block contains the following components

3.3.1 A QPSK Demodulator

The demodulator accepts the received signals and demodulates them. The demodulated signals are then passed to the Simple Decoder [12].

3.3. 2. My Viterbi Decoder

A traceback Viterbi decoder with a traceback depth of 35, i.e. five times the constraint length is used. Instead of setting the accumulated error for the first state to 0 as is the convention, the accumulated error for 7th previous state of the Simple Decoder is set to 0 for the first time slot. Once data for 35 time slots have been built up using Block 1 and Block 2 as described in Section 3, traceback operations can begin. When this traceback begins, a counter is also maintained. This counts the number of consecutive time slots for which the accumulated path metric of the global winner has remained constant. If this path metric has remained constant for 7 consecutive slots it is fairly certain that bit errors have stopped occurring. The Viterbi decoder is then stopped and the last traceback state passed to the Simple Decoder. The Simple Decoder can then resume operations accurately. On switching from the Viterbi decoder to the Simple Decoder (when errors stop occurring), the initial state of the flip flops is set to the binary value representation of the traceback state that gave the last decoded bit. This ensures that the initial state of the Simple Decoder is correct and therefore it gives correct outputs [6][7][10][13]. A check is also maintained on whether less than 35 time slots are remaining for end of data. If this condition is satisfied no switch is made to the Simple Decoder even if errors have stopped occurring. This ensures that if errors do occur shortly afterwards, a sufficient traceback depth still exists to accurately decode remaining data. However this check can be performed only if data length is known beforehand.

IV. MY VITERBI DECODER VS MATLAB DECODER

4.1. In order to test the performance of the Switching Decoder, the following measures were adopted. Random data was generated, encoded, modulated and transmitted. Uncoded data was also modulated and transmitted. Depending on the desired Eb/N0, the appropriate Additive White Gaussian Noise (AWGN) was added to the signal. At the receiver, data was demodulated. The encoded data was decoded by the three decoders, MATLAB® Viterbi decoder, 'My Viterbi' Decoder.

The following parameters are given to the MATLAB® Viterbi Decoder.

```
trellis = poly2trellis(7,[171 133]);
tblen = 35;
matdecodedHard = vitdec(Rx,trellis,tblen,'term','hard');
%Rx = Received Data
```


Published By:
Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)

The parameter 'hard' is used so that the Viterbi Decoder uses hard decisions in decoding. Now the MATLAB® Viterbi Hard Decision decoder can be compared with 'My Viterbi' decoder algorithm. The tests were conducted with a data length of 10,000 bits. Eb/N0 was varied from 0.5dB to 13 dB with an increment of 0.5 dB at each test. The tests were repeated 5 times and finally the results of the two decoders were compared and analyzed.

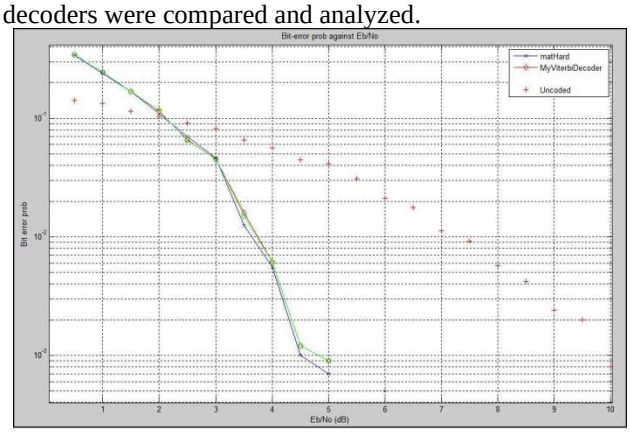

Figure 3: Performance of My Viterbi vs MATLAB Viterbi

Figure 3 shows that below 2 dB, the uncoded message performs better than the encoded messages. This is expected since the very high error rates cause the Viterbi decoder to follow an incorrect path. Above 2 dB, it is observed that the MATLAB® decoder and 'My Viterbi' Decoder both follow each other closely with only minor variations. As the SNR increases to 5 dB very few bit-errors occur, less than 10 in the 10,000 bits. This makes the result more unreliable at higher SNR [8][9]. Also, though the graph appears to show a larger difference in values at 5dB, this is not true. As the data moves to lower BEP, the log-scale increases the gap between two consecutive lines from 10^{-3} to 10^{-4}. This causes the gap between the two lines to appear larger even though difference in values remains the same.

4.2. Packet Loss Rate

In wireless communications most data is sent as packets. At the receiver, a check (usually a CRC check) is used to see whether the decoder was able to correct all bit errors in the packet. If there is even one bit-error in the packet, the packet is discarded. A new packet maybe requested. In this case, slight variations BEP performance will not matter. Whether the packet contained 1 bit-error or 10, the packet will still be discarded. Packet loss rate may differ from BEP depending on how close the bit-errors occur. Multiple bit-errors occurring within a single packet will result in only 1 packet loss. If these bit-errors are spread out into different packets, the packet loss rate increases considerably. In order to estimate the packet loss rate, 100 packets of 1000 data bits each were transmitted and the number of packets that were received without error after decoding using the three decoders was counted separately. Measurements were taken at each 0.25 dB going from 5 to 7.5 dB. The results are plotted in Figure 4.

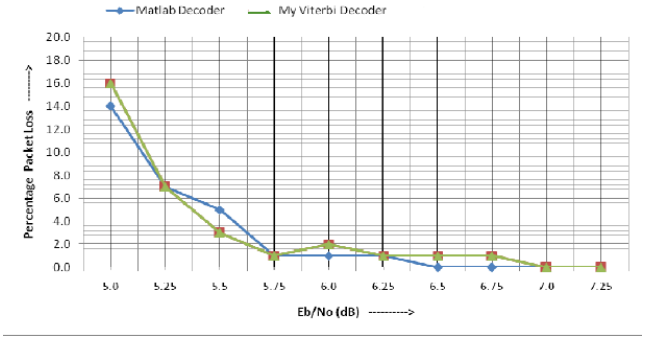

Figure 4: Packet Loss Performance of My Viterbi vs MATLAB Viterbi

On comparing with the MATLAB Viterbi decoder, there are slight variations in packet loss rate at some points though in several cases the packet loss rate is the same. According to the ITU Recommendations (ITU-R M.1079-2), a packet loss rate (PLR) of less than 3% is acceptable for real time audio communications. For video communications, PLR must be less than 1% and data communications require a PLR of 0%. From the graph in Figure 4, it is observed that above 5.75 dB packet loss rate is below 2%. When $E_b/N0$ drops below this value, packet loss rate increases rapidly.

V. CONCLUSION

Viterbi decoder is better than the conventional decoder because the errorcorrection capability of viterbi decoder is better than conventional. Viterbi decoder can find out the error even if two consecutive bits are changed. This is the main advantage of viterbi over conventional decoder. A viterbi decoder is designed and compared with the MATLAB decoder. My Viterbi decoder is checked on the basis of bit error rate and packet loss during decoding process against the MATLAB viterbi decoder. It has been found that My Viterbi is exactly matches to the MATLAB decoder and path of bit error rate coincides. Packet loss in My Viterbi also almost matches to MAYLAB viterbi decoder exactly from 5.75dB BER to 7.25dB.

REFERENCES

[1]. Kawokgy, M.; Salama, C.A.T. 2004. Low-power asynchronous Viterbi decoder for wireless applications. *IEEE Int. Symp. Low power Electronics and Design* (*ISLPED '04*), 9-11 Aug. 2004, pp. 286-289.

[2]. Kang, I. and Willson, A. N. Jr., 1998. Low-power Viterbi decoder for CDMA mobile terminals. *IEEE Journal Solid-State Circuits*, vol. 33 no.3, pp. 473 – 482.

[3]. C Arun and V Rajamani of Sri Venkateswara College of Engineering, Chennai, Design and VLSI architecture of non-polynomial based low probability of error Viterbi decoder, Journal of scientific and industrial research, vol. 68, February 2009, pages 97-106.

[4]. T Menakadevi and M Madheshwaran, Anna University, Coimbatore, Tamilnadu, design and implementation of high performance viterbi decoder for mobile communication data security, Computer intelligence in security for information systems, 2009, pages 69-76.

[5]. Hema.S, Suresh babu.V, Ramesh P, FPGA Implementation of Viterbi Decoder, Proceedings of the 6th WSEAS Int. Conf. on Electronics, Hardware, Wireless and Optical Communications, Corfu Island, Greece, 16-19 February 2007, pages 162-167.

[6]. K. S. Arunlal1 and Dr. S. A. Hariprasad," AN EFFICIENT VITERBI DECODER" International Journal of Advanced Information Technology (IJAIT) Vol. 2, No.1, February 2012

[7]. Ranpara, S.; Dong Sam Ha, 1999. A low-power Viterbi decoder design for wireless communications applications. *IEEE Proceedings of the Twelfth Annual IEEE International Int. ASIC Conference 1999*, Washington, DC, 15-18 Sept. 1999, pp. 377-381
[8]. Shannon, C.E., 1948. A Mathematical Theory of Communication. *Bell SystemTechnical Journal*, vol. 27, pp.379-423.
[9]. Sriram Swaminathan, Russel Tessier, Dennis Goeckel and Wayne Burleson, A Dynamically Reconfigurable Adaptive Viterbi Decoder, FPGA 02, Feb 2002, pages 24-26.
[10]. Y. Gang, A. T. Erdogan, and T. Arslan. An efficient pre-traceback architecture for the Viterbi decoder targeting wireless communication applications. *IEEE Transactions on Circuits and Systems—Part I: Regular Papers*, 53(9):1918–1927, Sept. 2006.
[11]. H.-L. Lou. Implementing the Viterbi algorithm. *IEEE Signal Processing Magazine*, 12(5):42–52, Sept. 1995.
[12]. P. Robertson and T. W¨orz. Bandwidth-efficient turbo trellis-coded modulation using punctured component codes. *IEEE Journal on Selected Areas in Communications*, 16(2):206–218, Feb. 1998.
[13]. Chip Fleming, "A Tutorial on Convolution Coding with Viterbi Decoding", Spectrum Applications, Nov 2006. Available :http://home.netcom.com/~chip.f/viterbi/tuto.