Available online at www.elixirpublishers.com (Elixir International Journal)

Network Engineering

Elixir Network Engg. 59 (2013) 15533-15537

ISSN: 2229-712X

Performance Analysis of MAP, Log MAP and Max Log MAP Turbo decoders in AWGN Channel

S.V.Viraktamath[1,*], Shweta S. Kulkarni[1] and Girish V. Attimarad[2]

[1]Department of Electronics and Communication Engineering, SDMCET, Dharwad, Karnataka, India.
[2]Department of Electronics and Communication Engineering, Dayanand Sagar College of Engineering, Bangalore, Karnataka, India

ARTICLE INFO

Article history:
Received: 21 April 2012;
Received in revised form:
2 June 2013;
Accepted: 6 June 2013;

Keywords
Error correcting code,
BER,
Turbo code,
MAP,
Log MAP,
Max Log MAP,
SNR.

ABSTRACT
Turbo codes are one of the most powerful error correcting codes. In this paper focus is made on Bit Error Rate (BER) performance analysis of MAP, Log MAP and Max Log MAP turbo decoding algorithms in wireless AWGN channels. The research is investigated in several aspects to evaluate the performance of turbo codes in AWGN channel with different Signal to Noise Ratio (SNR), different generator polynomials and for different constraint lengths. The simulation results give some demonstrations that are valuable for the applications of turbo codes in AWGN channel with different SNRs.

Introduction

Turbo codes, invented in 1993, have outstanding error correction performance, and have become one of the important research topics [1]. Hard decision and also Soft Output Viterbi Algorithms (SOVA) were also used for error correction in many applications. Among FEC schemes, convolutional encoding and Viterbi decoding [2] are the most popular because of their powerful coding-gain performances. A Turbo decoder gives a better performance when a MAP algorithm is employed. The computational complexity of the MAP algorithm was drastically reduced [3] by converting it from the probability domain to the logarithmic domain. The Log-MAP algorithm which is the derivative of the MAP (Maximum A Posteriori) algorithm is used in the implementation of turbo codes as it offers the best performance when compared to the Max-Log-MAP and the Viterbi algorithms. The Log-MAP algorithm is also superior with respect to the coding gain over the soft output Viterbi algorithm (SOVA) [4] that produces qualitatively inferior soft outputs. The storage of state metrics for the whole frame is mandatory before the likelihood decisions could be obtained, in a MAP decoder not using the sliding window technique requires large amount of memory.

The performance comparison of convolution code for IS -95 and turbo coded CDMA 2000 over Rayleigh fading channel is discussed in [5], Simulation is carried out in Rayleigh fading environment and it was observed that the turbo code performs better than convolution codes. Turbo code drastically improves the performance compared to convolutional codes even at low SNRs also. Turbo codes have three enhancements in the coding area. These include the use of recursive systematic convolutional (RSC) codes, interleaving and the separation of intrinsic and extrinsic information in the decoder for cooperation. The performance of turbo codes improves with increasing interleaver size due to the larger interleaver gain [6], [7]. However as the interleaver size increases it is difficult to use turbo codes in low transmission delay real time voice communications.

Turbo codes have shown very good performance in the AWGN channel. Specific guidelines for the optimal design of the constituent RSC codes have been proposed by Montorsi [8, 9]. It is shown that turbo codes also perform very well in fast fading channels [10, 11] and it is also shown that turbo codes perform poorly in slow fading channels [12, 13]. In quasi-static fading channels with considerable antenna diversity performance of turbo codes depends on the turbo codes parameters [14]. This paper is structured as follows. Section II describes the turbo encoder. Section III describes the turbo decoders. Simulation results are presented in Section IV. Section V concludes the work.

Turbo Encoder

The first turbo encoder was introduced in 1993 by Berrou et al. [15]. Since then several schemes have been proposed. A turbo encoder is formed from the parallel or series concatenation of convolutional codes separated by interleavers. Most of the turbo coding design follows the following characteristics. The encoders used are normally identical; the code is in a systematic form, i.e. the input bits also occur in the output. The interleaver reads the bits in a pseudo-random order.

The structure of a rate 1/3 turbo encoder is shown in Fig. 1. Two identical convolutional encoders are used. The information bits undergo a pseudo random interleaving (block I) before being fed into the second convolutional encoder. Accordingly, a turbo code is a block coding scheme where a block of K information bits are buffered prior to interleaving. Many types of interleavers can be used, provided they are sufficiently

Tele:
E-mail addresses: svvmath@gmail.com

random. Interleaver makes the two constituent encoders appear uncorrelated at the receiver.

Each convolutional encoder is based on a RSC code. Fig. 2 shows a simple 4-state example. The operation of the encoder can be summarized by the trellis diagram which shows the bit pairs output for each possible transition between successive states [16].

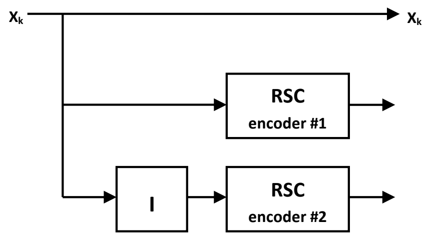

Fig. 1: Fundamental turbo code encoder

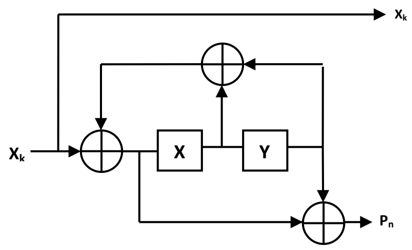

Fig.2: Recursive Systematic Convolutional RSC code

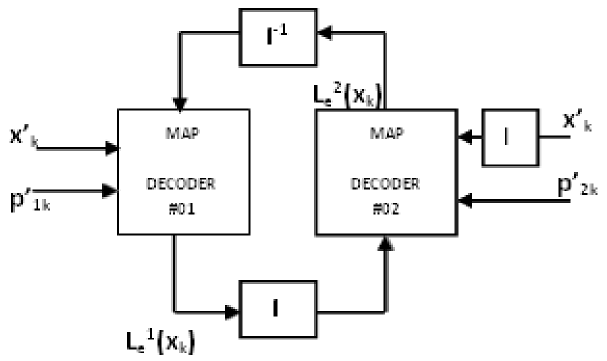

Fig 3: MAP Turbo Decoder

Turbo Decoder

The structure of a turbo decoder is shown in Fig.3. It consists of a pair of decoders which work cooperatively in order to refine and improve the estimate of the original information bits. The decoders are based on the MAP algorithm and output soft decision information learned from the noisy parity bits. Initially decoder #1 starts without initialization information (a priori estimates are set to zero). In subsequent iterations, the soft decision information of one decoder is used to initialize the other decoder. The decoder information is cycled around the loop until the soft decisions converge on a stable set of values. The latter soft decisions are then sliced to recover the original binary sequence [17]. The MAP algorithm minimizes the probability of bit error by using the entire received sequence to identify the most probable bit at each stage of the trellis. The MAP algorithm does not constrain the set of bit estimates to necessarily correspond to a valid path through the trellis. So the results can differ from those generated by a Viterbi decoder which identifies the most probable valid path through the trellis. For the very first iteration no extrinsic information is available from MAP decoder #2, so the apriori (L_a) information is set to zero. The branch probabilities can be calculated from the noisy observations according to Eq.1.

$$\gamma_k(s',s) = \exp\left(\frac{1}{2} x_k L_a(x_k) + x_k L_c x'_k + p_k L_c p'_k\right) \quad (1)$$

The term $\alpha_k(s')$ in Eq.2 is the probability of arriving at a branch in a particular state and the sequence of noisy observations y'1, $k_=$ $y'_1\ y'_2\ y'_3 \ldots\ldots\ldots\ y'_k$ which led up to that state. By summing over all paths leading into that state gives a forward recursion for calculating $\alpha_k(s')$ in terms of the values of $\gamma_k(s',s)$.

$$\alpha_k(s) = \sum_{s'} \Pr(s_{k-1} = s', y'_{1,k-1}) * \Pr(s_k = s, y'_k / s_{k-1} = s')$$

$$\alpha_k(s) = \sum_{s'} \alpha_{k-1}$$

$$\alpha_k(s) = \sum_{s'} \alpha_{k-1}(s')\gamma_k(s',s) \quad (2)$$

To begin the forward recursion, forward state probabilities must be initialized. In turbo coders all convolutional encoders are started in state 0. Thus the forward recursion can begin with:

$$\alpha_0(s) = \Pr(s = s_0) = 1 \quad s = 0$$
$$= 0 \quad s \neq 0 \quad (3)$$

The term $\beta_k(s)$ in Eq.4 is the probability of exiting a branch via a particular state s and the sequence of noisy observations $y'_{k+1,K=}$ $y'_{k+1}\ y'_{k+2}\ y'_{k+3\ldots\ldots\ldots}\ y'_K$ which finish off the trellis. By summing over all paths exiting that state backward recursion can be done for calculating $\beta_k(s)$ in terms of the values of $\gamma_k(s',s)$.

$$\beta_k(s') = \sum_{s} \Pr(s_{K+1} = s, y'_{k+1} \mid s_k = s') * \Pr(y'_{k+2.K} \mid s_{k-1} = s')$$

$$\beta_k(s') = \sum_{s} \beta_{k+1}(s)\gamma_{k+1}(s',s) \quad (4)$$

To begin the backward recursion we need to initialize the backward state probabilities. Convolutional encoder #1 is usually terminated at state 0. However, in general, the final state for convolutional encoder #2 is data dependent and unknown beforehand. A uniform distribution for the final state of encoder #2 may be assumed. A suitable initialization (where the number of states in the convolutional encoder is 2^v) is shown in Eq.5 and Eq.6 for decoder #1 and decoder #2 respectively.

MAP#1 $\beta_k(s) = \Pr(s_k = s) = 1 \quad s = 0 \quad = 0 \quad s \neq 0$ (5)

MAP#2 $\beta_k(s) = \Pr(s_k = s) = \dfrac{1}{2^v} \quad \forall\ s$ (6)

The MAP soft decisions are defined as the log likelihood ratio which is shown Eq.7 and Eq.8.

$$L_{map} = \ln\left[\frac{\Pr(x_k = +1 \mid y')}{\Pr(x_k = -1 \mid y')}\right] \quad (7)$$

$$L_{map}(x_k) = \ln\left[\frac{\sum_{(s',s)\in S^+} \alpha_{k-1}(s')\gamma_k(s',s)\beta_k(s)}{\sum_{(s',s)\in S^-} \alpha_{k-1}(s')\gamma_k(s',s)\beta_k(s)}\right] \quad (8)$$

MAP log likelihood is divided into three distinct components as given in Eq. 9.

$$L_{map}(x_k) = L_a(x_k) + L_c x_k' + L_e(x_k) \quad (9)$$

The first term $L_a(x_k)$ is the apriori information. This information is the initial estimate prior to running the MAP algorithm. The second term $L_c.x_k'$ is the information provided by that part of the noisy observation. The third term $L_e(xk)$is the 'extrinsic' information from the parity constraint.

Max-Log MAP Algorithm

The MAP algorithm is simplified to Max-Log MAP algorithm by transforming the equations of branch probability, forward recursion, backward recursion and L_{map} into the natural log arithmetic domain and then using the approximation given in Equation 10.

$$\ln(\sum e^{xi}) \approx \max_i(x_i) \qquad (10)$$

where $\max_{i(x_i)}$ means the maximum value of x_i .Then, branch probabilities *Γ*$\Gamma_k(s`,s)$, forward recursion $A_k(s)$ and backward recursion $B_k(s)$ and for Max-log-MAP algorithms are defined below. Branch probabilities of transition from previous state *s'* to present state *s* is given by Equation 11.

$$\Gamma_k(s',s) = \ln(\gamma_k(s',s))$$
$$= \ln(\exp[\frac{1}{2}(k_x L_a(x_k) + x_k L_c x'_k + p_k L_c p'_k)]$$
$$= [\frac{1}{2}(k_x L_a(x_k) + x_k L_c x'_k + p_k L_c p'_k)] \qquad (11)$$

Forward recursions are calculated as follows.

$$A_k(s) = \ln(\alpha_k(s))$$
$$\ln(\sum_{all\_s'} \alpha_{k-1}(s')\gamma_k(s',s)$$
$$= \ln(\sum_{all\_s'} \exp[A_{k-1}(s') + \Gamma_k(s',s)]$$
$$\approx \max_{s'}(A_{k-1}(s') + \Gamma_k(s',s)) \qquad (12)$$

Equation 12 implies that for each path (transition from the previous stage *s'* in the trellis to the present stage *s*), the algorithm adds a branch metric term $\Gamma_k(s',s)$ to the previous value of forward recursion $A_k(s')$ to find a new value $\tilde{A}_k(s)$ for that path. The new value of $A_k(s)$ according to Equation 12 is then the maximum of the values of the various paths reaching the state. This can be thought of as selecting one path as the "survivor" and discarding any other paths reaching the state.

Similar to the forward recursion, we can rewrite backward recursion as

$$B_{k-1}(s') = \ln(\beta_{k-1}(s'))$$
$$= \ln(\sum_{all\_s'} \beta_k(s)\gamma_k(s',s)$$
$$= \ln(\sum_{all\_s} \exp[B_k(s) + \Gamma_k(s',s)])$$
$$\approx \max_s(B_k(s) + \Gamma_k(s',s) \qquad (13)$$

Finally, the a-posteriori LLRs can be calculated by

$$L_{MAP}(x_k) = \ln\left[\frac{\sum_{(s',s)\in s} + \alpha_{k-1}(s')\gamma_k(s',s)\beta_k(s))}{\sum_{(s',s)\in s} - \alpha_{k-1}(s')\gamma_k(s',s)\beta_k(s))}\right]$$

$$= \ln\left[\frac{\sum_{(s',s)\in s} + \exp(A_{k-1}(s') + \Gamma_k(s',s) + B_k(s))}{\sum_{(s',s)\in s} - \exp(A_{k-1}(s') + \Gamma_k(s',s) + B_k(s))}\right]$$

$$= \max_{(s',s)\in s} + (A_{k-1}(s') + \Gamma_k(s',s) + B_k(s) - \max_{(s',s)\in s} - (A_{k-1}(s') + \Gamma_k(s',s) + B_k(s) \qquad (14)$$

This means that in the Max-Log-MAP algorithm for each bit x_k the a-posteriori LLR (L_{MAP}) is calculated by considering every transition from the trellis stage *s'* to the stage *s*. These transitions are grouped into those that might have occured if, x_k=+1 and those that might have occurred if x_k= -1. For both of these groups the transition giving the maximum value of [$A_{k-1}(s')+\Gamma_k(s',s)+B_k(s)$] is found, and the a-posteriori LLR is calculated based on only these two "best" transitions. For a binary trellis, there will be $(2*2^{K-1})$ transitions at each stage of the trellis, where *K* is the constraint length of the convolutional code. Therefore, there will be 2^{K-1} transitions to consider in each of the maximizations in Equation 14. The calculation of extrinsic information is same as for MAP algorithm.

Log MAP Algorithm

The Max-Log-MAP algorithm gives a slight degradation in performance compared to the MAP algorithm due to the approximation of Equation 10. When used for the iterative decoding of turbo codes, Robertson et al found this degradation to result in a drop in performance of about 0.35 dB. However, the approximation of Equation 4.1 can be made exact by using the Jacobian logarithm:

$$\ln(e^{x1} + e^{x2}) = \max(x_1, x_2) + \ln(1 + e^{-|x1-x2|})$$
$$= \max(x_1, x_2) + f_c(|x_1 - x_2|)$$
$$= g(x_1, x_2) \qquad (15)$$

where $f_c(x)$ can be called as the correction term. This forms the basis of the Log-MAP algorithm proposed by Robertson, Villebrun and Hoeher. Similar to the Max Log MAP algorithm, the forward recursions $A_k(s)$, backward recursions $B_k(s)$ and branch probabilities $\Gamma_k(s`,s)$ are calculated which are the natural logarithmic versions of $\alpha_k(s)$, $\beta_k(s)$ and $\gamma_k(s`,s)$ respectively. However, the maximization in Equations 12 and 13 is complemented by the correction term given in Equation 15. This means that the exact rather than approximate values of $A_k(s)$ and $B_k(s)$ are calculated. The Jacobian logarithm given by Equation 15 can be used to eliminate the approximation of the LLR (L_{MAP}) given in Equation 14. However, as explained earlier, there will be 2^{K-1} transitions to consider in each of the maximizations of Equation 14. Thus we must generalize Equation 15 in order to cope with more than two x_i terms. This is done by nesting the $g(x_1, x_2)$ operations as follows.

$$\ln(\sum_{i=1}^{l} e^{xi}) = g(x_l, g(x_{l-1}, ...g(x_3, g(x_2, x_1)))) \qquad (16)$$

The correction term $f_c(x)$ need not be computed for every value of *x*, but instead can be stored in a look-up table. This means that the Log-MAP algorithm is only slightly more complex than the Max-Log-MAP algorithm, but it gives better performance compared to the Max Log MAP algorithm. Therefore it is a very attractive algorithm to use in the component decoders of an iterative turbo decoder. The correction factor consideration can be as follows.

$$f_c(x) = 0.7 - \frac{x}{2} \quad 0 \le 0.51$$
$$= 0.57 - \frac{x}{4} \quad 0.51 \le 1.44$$
$$= 0.39 - \frac{x}{8} \quad 1.44 \le 2.88$$

$$= 0.33 \qquad 4 > x \qquad (17)$$

Simulation Results

The performance study of MAP, Log MAP and Max Log MAP algorithms for different constraint lengths, generator polynomials is presented. The performance measure parameter used for these algorithms is bit error rate. The original message is encoded for chosen generator polynomial and noise is introduced by AWGN channel for a given SNR. This is given as an input to the MAP, Log MAP and Max Log MAP decoders for decoding. BER is calculated for different SNR without varying generator polynomial. The process is repeated for different generator polynomials of same length. Varying generator polynomial length, different SNR vs BER plots are obtained. Processing time is also calculated for different input polynomials. Varying generator polynomial length processing time is calculated.

The performance analysis of MAP algorithm for the two different generator polynomials data 1 and data 2 keeping constraint length (CL) three is done. The performance is same for both the polynomials for the low SNRs. The block length considered for both the constraint lengths is ten. From these two simulations it was observed that as the generator polynomial changes the BER also changes. The Fig. 4 shows the variation in BER for different generator polynomials for the same set of received inputs. The block length considered is ten. It was observed that as the generator polynomial changes the BER also changes.

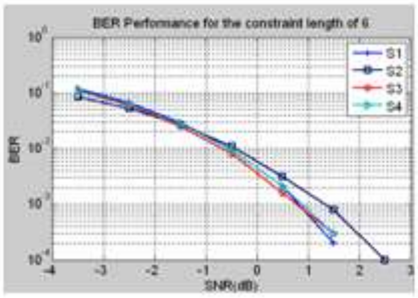

Fig 4: Performance Analysis of MAP for CL=6

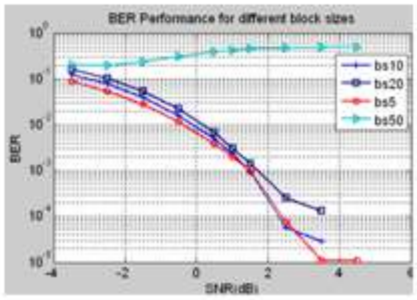

Fig 5: Performance Analysis of MAP for different block sizes

To study the impact of selection of block size on the performance of decoder, the block size was also varied. Fig.5 shows the impact of the variation in the block size on the BER for different SNR for the given constrain length. Four different block sizes are considered for the decoding are 5, 10, 20 and 50. It may be observed that as the block size changes the BER also changes. From the Fig. 5 it may be concluded that the BER is less for the block size of 5 and it is more for the block size of 50. As the block length becomes too much less overhead also become more. By plotting the BER performance for a particular application the block length can be decided.

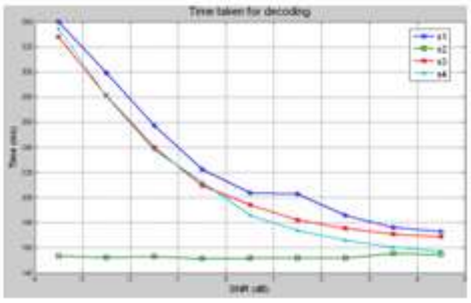

Fig 6: Time taken by the Decoder for different polynomials for the CL=6

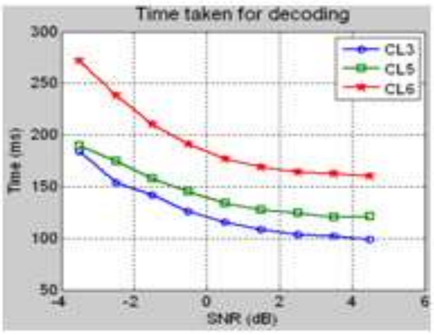

Fig 7: Time taken by the MAP decoder

Time taken by the MAP decoder for the different polynomials for the constraint length of 6 is plotted in Fig. 9. It may be concluded that time taken by the MAP decoder will vary for polynomial to polynomial. Fig. 7 shows the time taken by the turbo MAP decoding algorithm as the length of the generator polynomials changes. It may be observed that the time taken varies for the same set of received data as the constraint length changes; it increases as the constraint length increases.

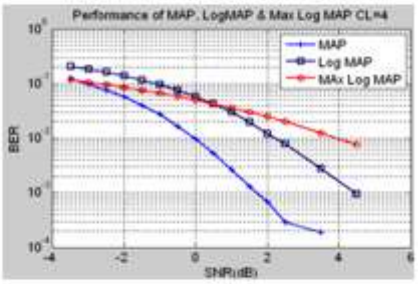

Fig 8: Performance comparison CL=4 for set 2

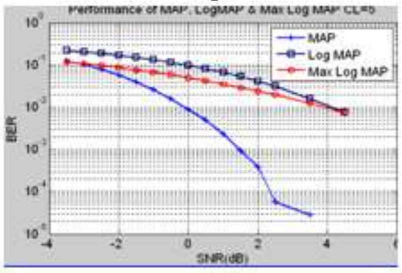

Fig 9: Performance comparison for CL=5 set 1

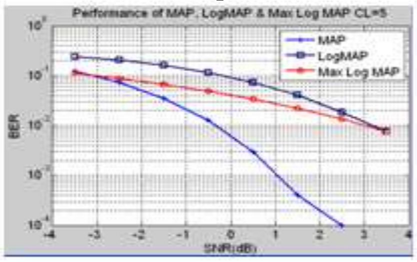

Fig 10: Performance comparison CL=5 - set 2

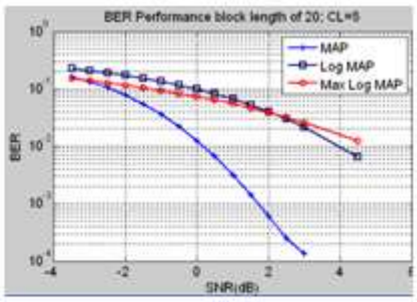

Fig 11: Performance comparison CL=5 set-1 Block length 20

Performance comparison of MAP, Log MAP and Max Log Map for the constraint length of three is done for two different set of polynomials with block size 10. It was observed that MAP performance is better compared to other two, and for the lower SNR Max Log Map performance is better compared to Log MAP.

Similarly the performance comparison of the three algorithms is made for the constraint lengths of four and five for two sets of inputs. Fig 10 shows the performance comparison of three algorithms for another set of input. Performance of comparison considering block length 20 is shown in Fig. 11.

Conclusion

Bit Error Rate (BER) performance analysis of MAP, Log MAP and Max Log MAP turbo decoding algorithms in wireless AWGN channels has been demonstrated in this paper. BER performance of MAP is done; from the results it may be concluded that performance changes as the generator polynomial changes as well as the constraint length changes. It is observed that as the constraint length increases the performance also improves. As the constraint length increases the time required to decode also increases. The performance of the MAP increases as the block length decreases.

In MAP, Log MAP and Max Log MAP turbo decoding algorithms in wireless AWGN channels; the performance of the MAP is better for any given constraint length and generator polynomial. The performance of Max Log MAP found to be better compared to Log MAP for lower SNR.

References

[1] LC. Berrou, A. Glavieux, and P.Thitimajshima, “Near Shannon Limit Error-Correcting Coding and Decoding: Turbo-Codes,” *IEEE International Conference on Communication (ICC’93)*, vol. 2, pp. 1064-1070, May.1993.

[2] A. J. Viterbi, “Convolutional codes and their performance in communication systems,” IEEE *Trans. Commun.*, vol. COM-19, pp. 751 -772, Oct., 1971.

[3] P. Robertson, E. Villerbrun, and P. H5her, "A Comparison of Optimal and Sub-Optimal MAP Decoding Algorithms Operating in the Log Domain," in Proceedings of the International Conference on Communications, (Seattle, USA), pp. 1009-10l3, June 1995.

[4] J. Hagenauer and P.Hoeher, "A Viterbi algorithm with soft-decision outputs and its applications," in Proc. Globecom, Dallas, TX, Nov. 1989, pp. 1680-1686.

[5] Dr. D. J. Shah, Prof. Vijay K. Patel, Prof. Himanshu A. Patel, “Performance Analysis of Turbo Code for CDMA 2000 with Convolutional Coded IS-95 System in Wireless Communication System”, 978-1-4244-7406-6/10/$26.00@ 2010 IEEE

[6] Y. Kim, J. Cho, W. Oh and K. Cheun, “Improving the performance of turbo codes by repetition and puncturing,” Project Report, Division of Electrical and Computer Engineering, Pohang University of Science and Technology.

[7] D. Divsalar, S. Dolinar, and F. Pollara, “Transfer Function Bounds on the Performance of Turbo Codes,” TDA Progress Report 42-122, Aug. 1995, Communications Systems and Research Section, R. J. McEliece California Institute of Technology, pp. 44-55.

[8] S. Benedetto and G. Montorsi, “Design of parallel concatenated convolutional codes,” IEEE Transactions on Communications, vol. 44, pp. 591-600, May 1996.

[9] J. P. Woodard and L. Hanzo, “Comparative study of turbo decoding techniques: An overview,” IEEE Transactions on Vehicular Technology, vol. 49, pp. 2208-2233, November 2000

[10] E. K. Hall and S. G. Wilson, “Design and analysis of turbo codes on Rayleigh fading channels,” IEEE Journal on Selected Areas in Communications, vol. 16, pp. 160-174, February

[11] R. Hoshyar, S. H. Jamali and A. R. S. Bahai, “Turbo coding performance in OFDM packet transmission,” Proceedings of the IEEE Vehicular Technology Conference-Spring, vol. 2, pp. 805-810, May 2000.

[12] L. Lin, L. J. Cimini and C. I. Chuang, “Comparison of convolutional and turbo codes for OFDM with antenna diversity in high-bit-rate wireless applications,” IEEE Communications Letters, vol.4, pp. 277- 279, September 2000.

[13] A. Stefanov and T. M. Duman, “Turbo-coded modulation for systems with transmit and receive antenna diversity over block fading channels: system model, decoding approaches, and practical considerations,” IEEE Journal on Selected Areas in Communications, vol. 19, pp. 958-968, May 2001.

[14] A. Stefanov and T. M. Duman, “Performance bounds for turbo-coded multiple antenna systems”, IEEE Journal on Selected Areas in Communications, vol.21, pp.374-381, April 2003.

[15] S. Benedetto and G. Montorsi, “Unveiling turbo codes: Some results on parallel concatenated coding schemes,” IEEE Transactions on Information Theory, vol. 42, pp. 409-429, March 1996.

[16] W.E. Ryan,"A Turbo Code Tutorial" New Mexico State University.

[17] M.C. Reed, S.S. Pietrobon, "Turbo-code termination schemes and a novel alternative for short frames" IEEE Int. Symp. on Personal, Indoor and Mobile Radio Commun., Taipei, Taiwan, pp. 354-358, Oct. 1996.