On the Robustness of Average-Entropy Stopping Criterion in Turbo Iterative Decoding

Roslina Mohamad[1,2], Harlisya Harun[2,3], Makhfudzah Mokhtar[4], Wan Azizun Wan Adnan[4], Kaharudin Dimyati[5]
[1]Faculty of Electrical Engineering, Universiti Teknologi Mara, Shah Alam, Selangor, Malaysia.
[2]Aerospace Engineering Department, Faculty of Engineering, Universiti Putra Malaysia, Serdang, Selangor, Malaysia.
[3]Avionics Section, UniKL MIAT, Dengkil, Selangor, Malaysia.
[4]Computer and Communication System Engineering Department, Faculty of Engineering, Universiti Putra Malaysia, Serdang, Selangor, Malaysia.
[5]Electrical and Electronic Engineering Department, Universiti Pertahanan Nasional Malaysia, Kuala Lumpur, Malaysia.
harlisya@unikl.edu.my

***Abstract*—Average-entropy (AE) is an efficient early stopping criterion that is able to terminate early in a varying signal-to-noise ratio (SNR) environment while suffering very little performance degradation. However, the performance of AE is only based on two code structures and depends on correct SNR; hence, the robustness of AE is still unknown, especially for other code structures and in incorrect SNR estimation. Thus, this paper presents the robustness test and analysis of AE based on the following parameters: frame size, code structure, channel reliability, and code rate. From the analysis, AE is capable to achieve good average iteration number performance for SNR ≤ 0.5 dB while maintaining the BER performance in correct SNR estimation. AE can also operate well in constant estimated SNR= 0 dB. However, AE suffers BER degradation for SNR > 0.5 dB both in correct SNR estimation and constant estimated SNR= 1 dB.**

***Index Terms*—Turbo Codes; Iterative Decoding; Average-Entropy; Stopping Criterion.**

I. INTRODUCTION

The superior performance of turbo codes is largely due to the iterative nature of the decoding algorithm [1]. The turbo-decoding principle requires an iterative algorithm involving two soft-input soft-output (SISO) decoders switching information with the aim of increasing the error correction performance with the decoding iterations [2]. Two suitable decoders are the maximum a posteriori (MAP) algorithm and the soft-output Viterbi algorithm (SOVA) [3]. The simplification of the decoding algorithm from MAP algorithm to log-MAP and its hybrid solutions can reduce the per-iteration complexity; however, the complexity of turbo codes remains substantial due to their iterative nature [4].

Hence, the optimisation of iterative decoding can be achieved by developing early stopping criteria. The stopping criteria can curtail the unnecessary iterations by terminating the convergence iterations earlier [1] while also maintaining reasonable degradation in bit error rate (BER) performances [5]. This can minimise the complexity of turbo codes by reducing the iteration numbers and decreasing the delays caused by iterative decoding. Based on the entropy concept, authors in [6] proposed a metric called average-entropy (AE) stopping criterion to measure the average uncertainty of the estimated bits of each iteration. The advantage of AE is it achieves better average iteration number (AIN) performance in various signal-to-noise ratio (SNR) compared with existing stopping criteria, e.g. cross-entropy criterion [7], sign-change ratio [8], sum reliability [9], and zero-entropy detection [10]. The AE also shows the reasonable degradation in BER performance similar to that of the existing stopping criteria. However, the research on robustness of the AE in additive white Gaussian noise (AWGN) channel is insufficient, especially for the various code structures, code rates, frame sizes, and channel reliability.

In addition, the research in [6] assumed correct channel SNR information or correct channel reliability (CCR) available at the receiver since AE requires three thresholds based on SNR information. However, operational environments can vary according to frame sizes, code structures, and channel reliability (CR) [11, 12], further aggravating the correct real-time threshold determination [11] and convergence or non-convergence detection [13-15]. Incorrect thresholds can cause too early a cessation of the iterative decoding by the stopping criteria or else could contribute to a need for extra iterations even though the decoder output is already converged. This situation will increase the error probability, additional computational complexity, and delay rather than the actual performance of the stopping criteria with the correct threshold.

Thus, this paper will investigate and test the robustness of AE by analysing and comparing its performance with Genie stopping criterion in terms of AIN and BER. The rest of this paper is organised as follows. The stopping rules of AE and its algorithm are given in Section 2. Then, Section 3 describes the details of the parameters for robustness test and present the simulation results in Section 4 to compare the AE performance with Genie. Finally, the concluding remarks are given in Section 5.

II. AE STOPPING CRITERION

In [6], a metric was proposed for measurement of the AE of the soft decoder output for each iteration. This method was based on the a posteriori AE that had been applied in ZED [10]. The AE metric was used to describe the decoder's uncertainty for the estimated bits; the smaller this metric was, the higher the probability that the estimated bits were correct. This metric has a close relation to the BER estimator, as the trend of the AE variation graph was consistent with the BER variation graph. A long offline stage was needed to obtain the AE and BER variations according to different noise levels.

Three thresholds were determined according to low, turning point (TP) or medium and high SNR regions from the AE and BER variation graphs, as shown in Table 1. The measurement for TP and the thresholds were based on turbo codes of N = 400, g = (7, 5) K = 3 and R = 1/3; this code was tested on the AWGN channel. In the online stage, the specific threshold values for different SNR ranges were used with AE stopping criterion to halt iteration in various SNRs.

Table 1
The low, TP and high SNR values with the respective thresholds

SNR region	SNR value (dB)	Threshold
Low	SNR < 0.8	Th1 = 10-3
TP/Medium	0.8 ≤ SNR < 1.2	Th2 = 1.5 ×10-3
High	SNR ≥ 1.2	Th3 = 2.5 ×10-3

The AE algorithm and stopping process are discussed as follows:

During the online stage, entropy of the LLR output of the I-the iteration is computed as:

$$\begin{aligned} H^{(i)}(\boldsymbol{z}|\boldsymbol{s}) &= -\sum_{\boldsymbol{z}} p(\boldsymbol{z}|\boldsymbol{s})\log p(\boldsymbol{z}|\boldsymbol{s}) \\ &= \sum_{k=1}^{N}\Big[p^{(i)}\left(z_k=0|s\right)\log p^{(i)}\left(z_k=0|s\right)\ldots \\ &\quad + p^{(i)}\left(z_k=1|s\right)\log p^{(i)}\left(z_k=1|s\right)\Big] \end{aligned} \tag{1}$$

where $p^{(i)}(z_k|\boldsymbol{s})$ is the belief in the estimated bits z_k and the derivation of $p^{(i)}(z_k|\boldsymbol{s})$, as given in (2).

$$p^{(i)}(z_k=d|\boldsymbol{s}) = \begin{cases} \dfrac{e^{L^{(i)}_{llr(2)}(z_k)}}{1+e^{L^{(i)}_{llr(2)}(z_k)}}, & for \;\; z_k=1 \\ \dfrac{e^{-L^{(i)}_{llr(2)}(z_k)}}{1+e^{-L^{(i)}_{llr(2)}(z_k)}}, & for \;\; z_k=0 \end{cases} \tag{2}$$

The AE per bit is then computed, as follows:

$$\begin{aligned} H^{(i)}_{av}(\boldsymbol{z}|\boldsymbol{s}) = -\frac{1}{N}\sum_{k=1}^{N}\Big[p^{(i)}(z_k=0|\boldsymbol{s})\log p^{(i)}(z_k=0|\boldsymbol{s})\ldots \\ + p^{(i)}(z_k=1|\boldsymbol{s})\log p^{(i)}(z_k=1|\boldsymbol{s})\Big] \end{aligned} \tag{3}$$

At the termination stage, the SNR values from the SNR estimator are compared. If the SNR value is within the low SNR region (see Table 1), the difference between AE for two consecutive iterations $(\Delta H^{(i)}_{av}(\boldsymbol{z}|\boldsymbol{s}))$ is computed as in (4). The iterative decoding is stopped when $\Delta H^{(i)}_{av}(\boldsymbol{z}|\boldsymbol{s})$ is equal to or smaller than a *Th*1, as given by (5):

$$\Delta H^{(i)}_{av}(\boldsymbol{z}|\boldsymbol{s}) = H^{(i-1)}_{av}(\boldsymbol{z}|\boldsymbol{s}) - H^{(i)}_{av}(\boldsymbol{z}|\boldsymbol{s}) \tag{4}$$

$$\Delta H^{(i)}_{av}(\boldsymbol{z}|\boldsymbol{s}) \le Th_1 \tag{5}$$

If the SNR value is under TP or within the high SNR region, the iterative decoding is terminated when $H^{(i)}_{av}(\boldsymbol{z}|\boldsymbol{s})$ is equal to or less than *Th*2 and *Th*3, respectively. This is shown by (6) and (7) for TP and high SNR values, respectively.

$$H^{(i)}_{av}(\boldsymbol{z}|\boldsymbol{s}) \le Th_2 \tag{6}$$

$$H^{(i)}_{av}(\boldsymbol{z}|\boldsymbol{s}) \le Th_3 \tag{7}$$

III. Robustness Test Parameters

In order to validate the performance of a stopping criterion, usually, a benchmark stopping criterion such as Genie or fixed iteration stopping criterion was used. Genie criterion is useful for establishing an unbeatable performance benchmark against which the other stopping criteria are measured. In the Genie stopping technique, the decoder is required to know all the transmitted bits and stops the decoding process when all the bits are correctly decoded. The stopping criterion is only suitable for implementation in offline-stage simulations and cannot be implemented for real-time applications because it is impossible for the receiver to know the exact value of the transmitted bits.

Table 2 shows the four parameters for robustness test, which include the frame size, CR, code structure (consisting of constraint lengths and generator polynomials), and code rate. The frame sizes were divided into three categories: small, medium, and large frames, representing the sizes of N = 100, 1000, and 10000, respectively. A correct channel reliability (CCR) and constant estimated channel reliability (CECR) were assumed throughout the robustness test. The channel reliability value (L_c) was set to 2/σ^2 for CCR, L_c = 2 (equal to SNR = 0 dB) and L_c = 2.52 (SNR = 1 dB) for CECR [16-19]. Three types of generator polynomials were used: g_1 = (7, 5), K_1 = 3; g_2 = (15, 17), K_2 = 4; and g_3 = (37, 21), K_3 = 5, which are optimum for maximising the minimum free distance of the component codes [13, 20]. Both codes were punctured (R_1 = 1/2) and unpunctured (R_2 = 1/3) to test the capability of stopping criteria on the code rate factors.

One million random binary data were generated and passed to the turbo encoder. The encoder output was modulated by binary phase shift keying (BPSK) and passed through the AWGN channel. At receiver, the BPSK data was demodulated and decoded by log-MAP decoder with maximum iteration, i_{max} = 7. The robustness test began by selecting the robustness factors and the parameters involved (e.g: fixed and tested parameters for frame size). The same steps were repeated for other parameters and robustness factors.

Table 2
Parameters for robustness test

Robustness factor	Fixed parameter	Tested parameter
Frame size	(7, 5, 3); R=1/2; CCR	100; 1000; 10000
CR	(7, 5, 3); R=1/2, N=1000	CCR; CECR
Code structure	CCR; R =1/2; N=1000	(7, 5, 3)); (15, 17, 4); (37, 21, 5)
Code rate	CCR; (7, 5, 3); N=1000	1/2; 1/3

IV. Results and Discussion

This section presents and evaluates the effects of robustness factor on the performance of AE based on the AIN and BER. The performance of AE is compared with the Genie stopping criterion in terms of its proximity between the two. The four robustness factors effect was discussed in its sub-section.

A. The Effect of Frame Size on AE

The effects of frame size on AIN of AE performances are plotted in Figure 1. AE with N = 100, 1000 and 10000 are able to terminate early at low SNRs with minimum AIN = 2, 2.2 and 2.5, respectively. Meanwhile, the AIN for AE with N = 1000 and 10000 are seen outperforming the AIN of the respective Genie at high SNRs. This led the lower AIN to be even lower than the benchmark, while N = 100 requires an addition of at least 0.8 AIN as compared to the respective Genie.

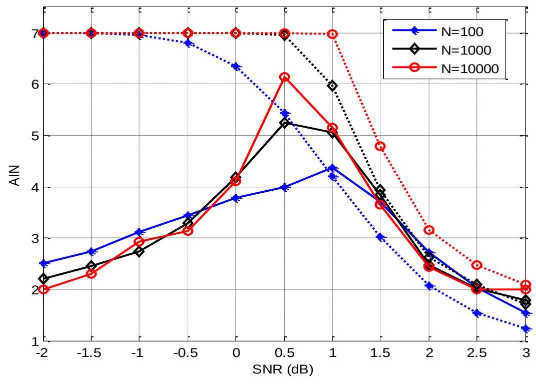

Figure 1: The effect of frame size on AIN of AE. [Dotted line: Genie, solid line: AE]

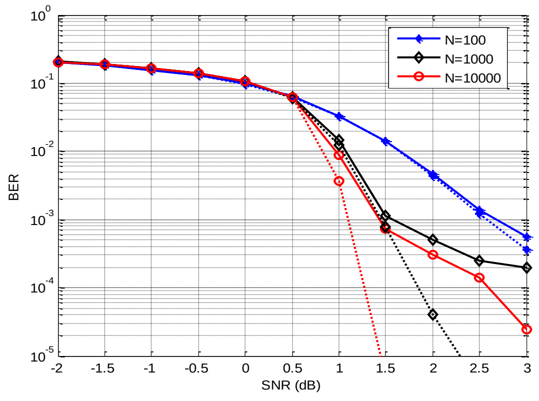

Figure 2: The effect of frame size on BER of AE. [Dotted line: Genie, solid line: AE]

The effects of frame size on BER of AE performances are depicted in Figure 2. At low SNRs, all the frame sizes are able to maintain the BER performances. However, the insufficient AIN at high SNRs causes a penalty in BER performance for N = 1000 and 10000. Furthermore, at this stage, the difference with a BER of AE is quite large, approaching to 1.2 dB at BER = 2×10^{-4}. This indicates that the termination in AE is only suitable for a small frame size (N = 100) whereas for N = 1000 and 10000, AE is unable to detect the correct convergence output. Due to this, AE makes the turbo decoder experience a false-alarm situation and causes the decoder to stop too early even though the decoder output is still reliable. From the result at low SNRs, AE terminates early at low SNRs while maintaining the BER performance for all frame sizes. When referred to the results at high SNRs, AE is more likely to be robust to small frame sizes for various SNRs whereas it appears to be experiencing a false alarm and generating a penalty in BER performance for medium and large frame sizes.

B. The Effect of CR on AE

The effects of the CR on AIN of AE are shown in Figure 3. At low SNRs, CCR and CECR with the value of L_c = 2 for AE are able to terminate early, but the CECR with L_c value of 2.52 reaches the maximum iteration number, as well as the Genie. However, AE with the CECR, L_c = 2.52, shows a good performance at high SNRs compared to others. In fact, AE with L_c = 2 shows the worst AIN reading even when compared with Genie. Meanwhile, AE with CCR performs slightly better than Genie.

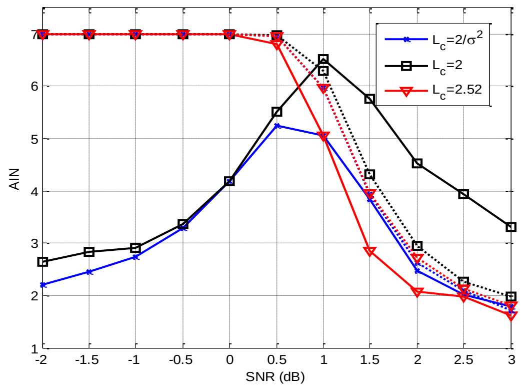

Figure 3: The effect of CR on AIN of AE. [Dotted line: Genie, solid line: AE]

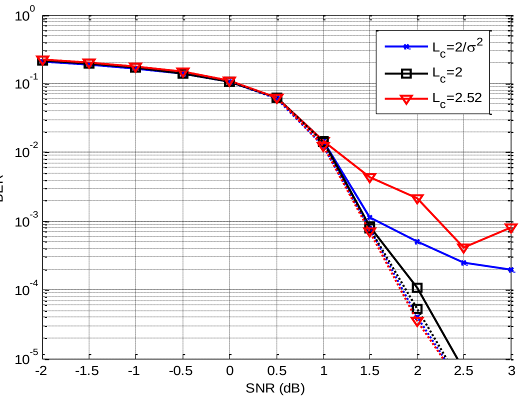

Figure 4: The effect of CR on BER of AE. [Dotted line: Genie, solid line: AE]

The AE is able to maintain the BER performance from a low SNR to SNR = 1.0 dB for all CR as shown in Figure 4. At SNR > 1.5 dB, however, the extra AIN required by AE of CECR with L_c = 2 has the effect of making the BER performance better than AE with the other CR requirements. In fact, its performance is close to the Genie with all CR requirements. Though CCR and CECR with L_c = 2.52 give a small AIN, the poor BER performance occurred at high SNRs where the BER tends to flatten due to error floor. This incident is caused by the threshold values in AE, where it only calculated the BER variation until 1.5 dB. Therefore, it makes the threshold only adopted between the specific ranges. The problem can be solved by measuring the threshold for higher SNR values. The BER of AE of CCR with L_c = 2 seems a better choice at high SNRs even though it requires a slightly higher AIN than the AE with the other CR requirements.

C. The Effect of Code Structure (g and K) on AE

In general, the effects of code structures to the AIN of the AE most likely are the same for different parameters, as depicted in Figure 5. At low SNRs, AEs are able to terminate early as compared to maximum iteration in Genie. Elsewhere, AE performed according to the Genie at high SNRs and with slightly reduced AIN. In comparison to the respective code structures, (g_1, K_1) requires more AIN, followed by (g_2, K_2) and (g_3, K_3) for all SNR environments.

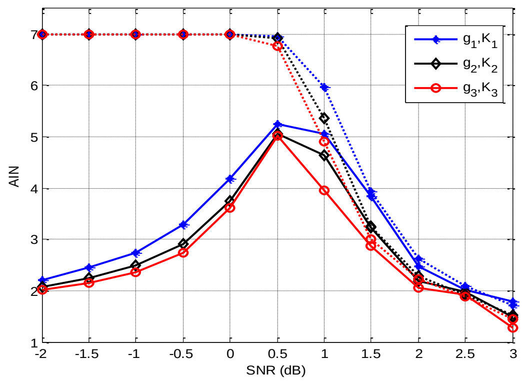

Figure 5: The effect of code structure (g and K) on AIN of AE. [Dotted line: Genie, solid line: AE]

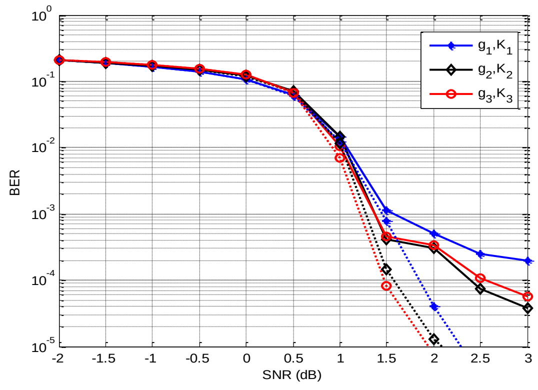

Figure 6: The effect of code structure (g and K) on BER of AE. [Dotted line: Genie, solid line: AE]

Figure 6 depicts the effect of code structure on the BER of AE. At low SNRs, all of the code structures are able to maintain the BER performance; however, this is not the case at high SNRs. The BER performance degrades significantly as compared to Genie, which exceeds 1 dB for SNR ≥ 1.5 dB. The result suggests that AE can terminate early at low SNRs while maintaining the BER performance for all code structures. In contrast, AE generates a penalty in BER performance for various code structures for SNR ≥ 1.5 dB.

D. The Effect of Code Rate on AE

Figure 7 shows the AIN with different code rates, where $R_1 > R_2$. In general, the AE with higher code rate, R_1 is able to terminate early with smaller AIN at low SNRs as compared to Genie. At high SNRs, the AIN for AE and Genie are the same with respect to the same code rate. The AIN of AE, interestingly, is higher for the low code rate at low SNRs while it is higher for the high code rate at high SNRs.

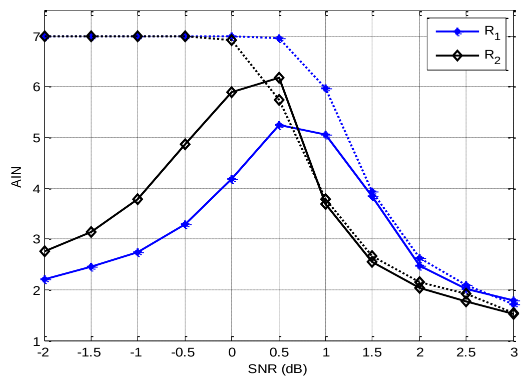

Figure 7: The effect of code rate on AIN of AE. [Dotted line: Genie, solid line: AE]

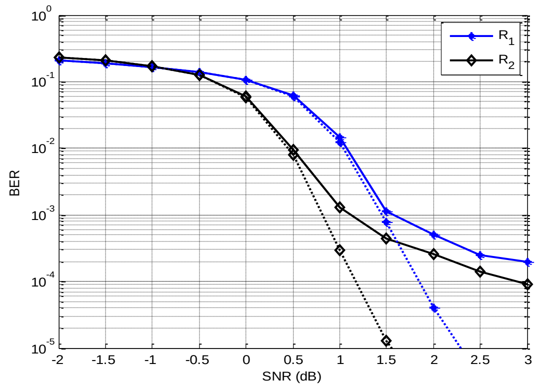

Figure 8: The effect of code rate on BER of AE. [Dotted line: Genie, solid line: AE]

However, the comparable AIN results for AE to the Genie does not retain for the BER performance. The AE with both R values, failed to maintain the BER performance at the level of the respective Genie at high SNRs compared to low SNRs, as shown in Figure 8. The BER degradation for AE for both R values increases with an increase in SNR values and approaching to 1.25 dB at BER = 2×10^{-4}. From the result, it can be seen that AE is able to terminate early at low SNRs while maintaining the BER performances for different code rates. However, there is a decrease in BER performance at high SNRs values for various code rates.

V. Conclusions

This paper presented the performance analysis of AE in determining its robustness in turbo iterative decoding. AE exhibits a good AIN performance at low SNRs and comparable performance to Genie at high SNRs. However, the AIN savings at high SNRs cannot maintain the BER performance. There is a trade-off when it comes to using AE at high SNRs that makes the turbo decoder experience false-alarm situation and causes the decoder to stop too early even though the decoder output is still reliable. Thus, the degradation of BER performances must be weighed against the excellent saving in AIN performance for all robustness factors. However, AE can be an alternative stopping criterion for SNR ≤ 0.5 dB in a varying SNR environment.

ACKNOWLEDGEMENT

The authors would like to thank Universiti Teknologi Mara (UiTM) and Ministry of Higher Education, Malaysia (MOHE) for supporting this research through Ph.D sponsorship.

REFERENCES

[1] Guerrieri, L., Veronesi, D., and Bisaglia, P. 2008. Stopping Rules For Duo-Binary Turbo Codes And Application To Homeplug AV. *IEEE Global Telecommunications Conference (GLOBECOM)*. 1-5.

[2] Boutillon, E., Douillard, C., and Montorsi, G. 2007. Iterative Decoding Of Concatenated Convolutional Codes: Implementation issues. *Proceedings of the IEEE*. 95(6): 1201-1227.

[3] Hagenauer, J. and Hoeher, P. 1989. A Viterbi algorithm With Soft-Decision Outputs And Its Applications. *IEEE Global Telecommunications Conference (GLOBECOM)*. 1680-1686.

[4] Hong, C., Maunder, R. G., and Hanzo, L. 2013. A Survey And Tutorial On Low-Complexity Turbo Coding Techniques And A Holistic Hybrid ARQ Design Example. *IEEE Communications Surveys and Tutorials*. 15(4): 1546-1566.

[5] Obiedat, E. A. and Cao, L. 2008. Turbo Decoder For Low-Power Ultrawideband Communication Systems. *International Journal of Digital Multimedia Broadcasting*. 2008(

[6] Chen, J. Y., Zhang, L., and Qin, J. 2008. Average-Entropy Variation In Iterative Decoding Of Turbo Codes And Its Application. *Electronics Letters*. 44(22): 1314-1315.

[7] Hagenauer, J., Offer, E., and Papke, L. 1996. Iterative Decoding Of Binary Block And Convolutional Codes. *IEEE Transactions on Information Theory*. 42(2): 429-445.

[8] Shao, R. Y., Shu, L., and Fossorier, M. P. C. 1999. Two simple Stopping Criteria for Turbo Decoding. *IEEE Transactions on Communications*. 47(8): 1117-1120.

[9] Gilbert, F., Kienle, F., and Wehn, N. 2003. Low Complexity Stopping Criteria for UMTS Turbo-decoders. *The 57th IEEE Semiannual Vehicular Technology Conference, (VTC-Spring)*. 2376-2380.

[10] Kocarev, L., Lehmann, F., Maggio, G. M., Scanavino, B., Tasev, Z., and Vardy, A. 2006. Nonlinear Dynamics of Iterative Decoding Systems: Analysis and applications. *IEEE Transactions on Information Theory*. 52(4): 1366-1384.

[11] Lei, H., Zhang, Q. T., and Cheng, L. L. 2011. Information Theoretic Criterion For Stopping Turbo Iteration. *IEEE Transactions on Signal Processing*. 59(2): 848-853.

[12] Cao, L. 2006. Bit-based SNR Insensitive Early Stopping For Turbo Decoding. *Electronics Letters*. 42(19): 1106-1107.

[13] Hanzo, L., Woodard, J. P., and Robertson, P. 2007. Turbo Decoding And Detection For Wireless Applications. *Proceedings of the IEEE*. 95(6): 1178-1200.

[14] Zhai, F. and Fair, I. J. 2003. Techniques for Early Stopping and Error Detection In Turbo Decoding. *IEEE Transactions on Communications*. 51(10): 1617-1623.

[15] Ten Brink, S. 2001. Convergence Behavior of Iteratively Decoded Parallel Concatenated Codes. *IEEE Transactions on Communications*. 49(10): 1727-1737.

[16] Lu, E.-H., Lin, Y.-N., and Hung, W.-W. 2009. Improvement of Turbo Decoding Using Cross-Entropy. *Computer Communications*. 32(6): 1034-1038.

[17] Sklar, B. 1997. A primer on turbo code concepts. *IEEE Communications Magazine*. 35(12): 94-102.

[18] Hanzo, L., Liew, T. H., Yeap, B. L., Tee, R. Y. S., and Ng, S. X. 2011. *Turbo Coding, Turbo Equalisation And Space–Time Coding*. John Wiley & Sons, Ltd.

[19] Hyundong, S. and Jae Hong, L. 2002. Channel Reliability Estimation For Turbo Decoding In Rayleigh Fading Channels With Imperfect Channel Estimates. *IEEE Communications Letters*. 6(11): 503-505.

[20] Proakis, J. G. 2008. *Digital Communications*. McGraw-Hill, New York.