Reduced-Complexity Syndrome-Based TTCM Decoding

Zunaira Babar, Soon Xin Ng, *Senior Member, IEEE*, and Lajos Hanzo, *Fellow, IEEE*

***Abstract*—The iterative decoder of Turbo Trellis Coded Modulation (TTCM) exchanges extrinsic information between the constituent TCM decoders, which imposes a high computational complexity at the receiver. Therefore we conceive the syndrome-based block decoding of TTCM, which is capable of reducing the decoding complexity by disabling the decoder, when syndrome becomes zero. Quantitatively, we demonstrate that a decoding complexity reduction of at least 17% is attained at high SNRs, with at least 20% and 45% reduction in the 5^{th} and 6^{th} iterations, respectively.**

***Index Terms*—Syndrome decoding, TTCM, error trellis, block syndrome decoding, iterative decoding.**

I. Introduction

TURBO Trellis-Coded Modulation (TTCM) [1] constitutes a bandwidth-efficient near-capacity joint modulation/coding solution, which relies on the classic turbo coding architecture, but involves the bandwidth-efficient Trellis-Coded Modulation (TCM) [2] instead of the constituent convolutional codes. More explicitly, the constituent TCM codes, which can be optimally designed using EXtrinsic Information Transfer (EXIT) charts [3], are concatenated in a parallel fashion and iterative decoding is invoked at the receiver for exchanging extrinsic information between the pair of TCM decoders. In order to reduce its decoding complexity, we propose to reduce the effective number of decoding iterations by appropriately adapting the syndrome-based block decoding approach of [4], [5] for TTCM.

In contrast to the trellis of the conventional convolutional decoder which uses the Generator Matrix (GM), $\mathbf{G}$, for generating its trellis, the trellis used for syndrome decoding is based on the syndrome former $\mathbf{H^T}$ [6], [7]. The idea of syndrome decoding was first conceived in [6] for the efficient hard decoding of convolutional codes using an error trellis[1]. Later, soft-decision syndrome decoding approaches were presented in [8] and [9], which were based on the error and codeword trellis, respectively.

In the error trellis of a syndrome decoder, the state probabilities are a function of the channel errors rather than of the coded sequence. Consequently, at high SNRs, the syndrome decoder is more likely to encounter a zero-state due to the predominant error-free transmissions. This underlying property of syndrome decoding has been exploited in [4] for developing a Block Syndrome Decoder (BSD) for convolutional codes, which divides the received sequence into erroneous and error-free parts based on the syndrome. More specifically, the BSD only decodes the erroneous blocks, with the initial and final states of the trellis initialized to zero. Therefore, the decoding complexity is substantially reduced at higher SNRs. This approach was then further extended to turbo codes in [5], where a pre-correction sequence[2] was also computed at each iteration to correct the errors. Thus, the decoding complexity was reduced not only at higher SNRs, but also for the higher-indexed iterations. Furthermore, a syndrome-based MAP decoder was proposed in [10] for designing an adaptive low complexity decoding approach for turbo equalization.

Against this background, our novel contribution is that we have extended the application of the syndrome-based MAP decoder of [10] together with the BSD of [5] to TTCM for the sake of reducing its decoding complexity. The proposed BSD-TTCM achieved an overall complexity reduction of at least 17% as compared to the conventional TTCM decoder. More specifically, at least 20% and 45% complexity reduction was achieved for the 5^{th} and 6^{th} iterations, respectively.

This letter is organized as follows. In Section II, the block-based syndrome decoder designed for TTCM will be presented. Our results will be discussed in Section III and our conclusions are offered in Section IV.

II. Block-based Syndrome Decoder for TTCM

Fig. 1 shows the schematic of one of the two constituent decoders of our proposed Block Syndrome Decoder conceived for TTCM (BSD-TTCM). The received symbol sequence y_k is demapped onto the nearest point x_i in the corresponding 2^{n}-ary constellation diagram, yielding the hard-demapped symbols $\hat{y}_k$, i.e.

$$\hat{y}_k = \arg\min_i (y_k - x_i), \tag{1}$$

for $i \in \{0, \ldots, 2^{\mathrm{n}} - 1\}$ and,

$$y_k = x_k + n_k. \tag{2}$$

Here, x_k is the complex-valued phasor corresponding to the n-bit transmitted codeword c_k, which is obtained using the 2^{n}-PSK mapper μ as follows:

$$x_k = \mu\left(c_k\right), \tag{3}$$

Manuscript received January 23, 2013. The associate editor coordinating the review of this letter and approving it for publication was M. Lentmaier.

The authors are with the School of Electronics and Computer Science, University of Southampton, SO17 1BJ, United Kingdom (e-mail: {zb2g10, sxn, lh}@ecs.soton.ac.uk).

The financial support of the European Union's Seventh Framework Programme (FP7/2007-2013) under the auspices of the CONCERTO project (grant agreement no 288502), as well as that of the European Research Council under its Advanced Fellow grant is gratefully acknowledged.

Digital Object Identifier 10.1109/LCOMM.2013.13.130182

[1]Syndrome decoding can be based either on the error trellis (syndrome $\neq 0$) or codeword trellis (syndrome $= 0$).

[2]Pre-correction sequence is an estimated/predicted error sequence, which is used to correct errors in the received information.

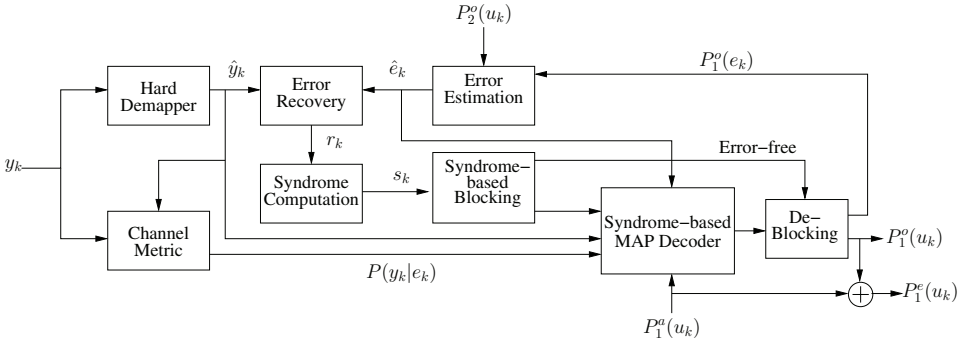

Fig. 1. Schematic of the proposed BSD-TTCM Decoder. *Only one constituent decoder is shown here.* $P_i^a[.]$, $P_i^e[.]$ *and* $P_i^o[.]$ *are the a-priori, extrinsic and a-posteriori probabilities related to the* i^{th} *decoder;* e_k *is the channel error on the transmitted symbol and* u_k *is the information part of* e_k.

and n_k is the noise experienced by the k^{th} symbol in an AWGN channel.

Recall that in TTCM, the odd and even symbols are punctured for the upper and lower TCM decoders respectively, while the parity bits of the corresponding hard-demapped symbols are set to zero [1]. Then, a pre-correction sequence $\hat{e}_k$, which is predicted by the error estimation module, is used for correcting any predicted errors. This sequence is initialized to zero for the first iteration. The syndrome $\mathbf{s}$ is computed for the corrected symbol stream $\mathbf{r}$ using the syndrome former $\mathbf{H^T}$ as follows:

$$\mathbf{s} = \mathbf{r}\mathbf{H}^{\mathbf{T}}, \tag{4}$$

where, the j^{th} bit of $\mathbf{r}_k$ is related to that of $\hat{\mathbf{y}}_k$ and $\hat{\mathbf{e}}_k$, for $j \in \{0, \ldots, \mathrm{n}-1\}$, as follows:

$$r_{k,j} = \hat{y}_{k,j} \oplus \hat{e}_{k,j}, \tag{5}$$

with $\mathbf{r}_k = [r_{k,0}, \ldots, r_{k,j}, \ldots, r_{k,\mathrm{n}}]$, $\hat{\mathbf{y}}_k = [\hat{y}_{k,0}, \ldots, \hat{y}_{k,j}, \ldots, \hat{y}_{k,\mathrm{n}}]$, and $\hat{\mathbf{e}}_k = [\hat{e}_{k,0}, \ldots, \hat{e}_{k,j}, \ldots, \hat{e}_{k,\mathrm{n}}]$.

Then the syndrome is analyzed for sake of dividing the received block into error-free and erroneous sub-blocks. The error-free sub-blocks are then subjected to a hard-decision and only the erroneous sub-blocks are passed to the MAP decoder. Like in the conventional TTCM decoder, both constituent decoders have a similar structure and iterative decoding is invoked for exchanging extrinsic information between the two.

A. Syndrome-Based MAP Decoder

We have invoked the syndrome-based MAP decoder of [10] in the proposed BSD-TTCM of Fig. 1. In contrast to the conventional MAP decoder, which operates on the basis of the codeword trellis, its syndrome-based MAP counterpart relies on the error trellis constructed using the syndrome former $\mathbf{H^T}$ [6], [7]. More explicitly, each trellis path of a codeword trellis represents a legitimate codeword. By contrast, each path of an error trellis specifies the hypothetical error sequence causing a departure from a specific legitimate codeword trellis path. Furthermore, both trellises have the same complexity and every error path in the error trellis uniquely corresponds to a codeword path in the codeword trellis [7]. The classic MAP algorithm [11] computes the *A-Posteriori* Probability (APP) $P^o(u_k)$ for every M-ary transmitted information symbol u_k given by $P^o(u_k) = P(u_k = m|y_k)$ for $m \in \{0, 1, \ldots, M-1\}$, where $M = 2^{\mathrm{n}-1}$, $(\mathrm{n}-1)$ is the number of bits in an information symbol and $R = \frac{\mathrm{n}-1}{\mathrm{n}}$ is the coding rate. However, the syndrome-based MAP computes the APP for every M-ary channel error experienced by the information symbol. In other words, u_k is the transmitted information symbol in the codeword trellis, whereas, it is the M-ary channel error experienced by the information symbol in the error trellis. Therefore, the channel information $P(y_k|x_k)$ for the transmitted codeword x_k, is modified to $P(y_k|e_k)$ for the channel error e_k, which is formulated as:

$$P(y_k|e_k) = \frac{1}{2\pi\sigma^2} \,.\, e^{-\frac{|y_k - \hat{x}_k|^2}{2\sigma^2}}, \tag{6}$$

where σ^2 is the noise variance per dimension and $\hat{x}_k$ is given by:

$$\hat{x}_k = \mu\left(\hat{c}_k\right), \tag{7}$$

for,

$$\hat{c}_{k,j} = \hat{y}_{k,j} \oplus e_{k,j}. \tag{8}$$

Here, we have $\hat{\mathbf{c}}_k = [\hat{c}_{k,0}, \ldots, \hat{c}_{k,j}, \ldots, \hat{c}_{k,\mathrm{n}}]$ and $\mathbf{e}_k = [e_{k,0}, \ldots, e_{k,j}, \ldots, e_{k,\mathrm{n}}]$. The APP of u_k can be calculated in terms of the forward-backward recursive coefficients α_k and β_k as follows:

$$P^o(u_k) = \sum_{\substack{(\hat{\tau},\tau) \Rightarrow \\ u_k = m}} \gamma_k(\hat{\tau}, \tau) \,.\, \alpha_{k-1}(\hat{\tau}) \,.\, \beta_k(\tau), \tag{9}$$

where the summation implies adding all the probabilities associated with those transitions (from state $\hat{\tau}$ to τ) of the error trellis for which $u_k = m$. Furthermore, we have:

$$\begin{aligned} \gamma_k(\hat{\tau}, \tau) &= P^a(u_k) \,.\, P(y_k|e_k), \\ \alpha_k(\tau) &= \sum_{\text{all } \hat{\tau}} \gamma_k(\hat{\tau}, \tau) \,.\, \alpha_{k-1}(\hat{\tau}), \\ \beta_{k-1}(\hat{\tau}) &= \sum_{\text{all } \tau} \gamma_k(\hat{\tau}, \tau) \,.\, \beta_k(\tau), \end{aligned} \tag{10}$$

where $P^a(u_k)$ is the *a-priori* probability of the information part of the error e_k, i.e. u_k. At the first iteration, no *a-priori* information is available; hence, it is initialized to be equiprobable, i.e. $P^a(u_k) = 1/M$.

B. Error Estimation

Similar to the bit-wise pre-correction sequence proposed in [5] for turbo codes, we make an estimate of the 2^{n}-ary symbol error in each iteration to ensure that the Hamming weight of the syndrome decreases with ongoing iterations.

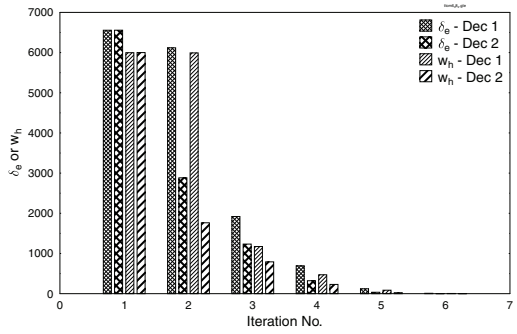

Fig. 2. Variation in the number of differences (δ_e) between the actual and estimated error and hamming weight (w_h) of the syndrome with increasing iterations at $E_b/N_0 = 3.8$ dB.

While the extrinsic information was used in [5] for the estimating the pre-correction sequence, we have improved the estimation by using the APP. This proceeds as follows:

- The information part of the pre-correction sequence $\hat{e}_k$ is set to the hard decision of the APP of the information symbol $(P^o(u_k))$ computed by the other decoder.
- The parity part of $\hat{e}_k$ is set to the hard decision value of the APP of the codeword $(P^o(e_k))$ gleaned from the previous iteration of the same decoder, which yields the same information symbol as that computed in the first step.

Fig. 2 verifies the accuracy of our pre-correction sequence. Here the average number of differences δ_e, between the actual and estimated error, is plotted against the number of iterations at an SNR per bit of $E_b/N_0 = 3.8$ dB, for 1000 frames of 12000 TTCM-8PSK symbols transmitted over an AWGN channel. Both constituent decoders are characterized separately, which are referred to as *Dec 1* and *Dec 2* in Fig. 2. Observe that the differences decrease at each successive iteration, eventually reaching zero at the 6^{th} iteration. Furthermore, the Hamming weight w_h of the syndrome closely follows the same trend.

C. Syndrome-based Blocking

The Hamming weight of the syndrome sequence of Eq. (4) decreases at higher SNRs, since only a few errors are encountered. It also decreases with each successive iteration as seen in Fig. 2. This is because the errors are estimated at each iteration and the corresponding correction is applied to the received symbols. In other words, upon increasing the number of iterations or SNR, the syndrome exhibits longer sequences of zeros, which indicates error-free transmission. This fact can be exploited to partition the received block into error-free and erroneous segments, as proposed in [4], [5]. This is achieved by heuristically choosing a design parameter, $L_{\min} = (L_{\text{start}} + L_{\text{end}} + 1)$, which is the minimum number of consecutive zero syndromes after which the sub-block is deemed to be error-free. Furthermore, L_{start} and L_{end} define the start and end of the next and previous sub-blocks, respectively. If L_0 is the length of the sub-block having at least $L_{\min}$ consecutive zero syndromes, then the initial $L_{\text{end}} = (L_{\min} - 1)/2$ symbols of this sub-block are appended to the previous erroneous block and the last $L_{\text{start}} = (L_{\min} - 1)/2$ symbols are appended to the following erroneous block [4], [5]. Only the remaining $(L_0 - L_{\min} + 1)$ symbols are considered error-free. This ensures that the trellis of the erroneous sub-blocks starts from and terminates at the zero state. The hypothetical error-free blocks do not undergo further decoding and the corresponding APPs of the error-free trellis segment are set to 1. On the other hand, the erroneous blocks are fed to a MAP decoder with the initial and final states of the decoding trellis set to zero.

TABLE I
OPTIMUM $L_{\min}$ FOR VARYING E_b/N_0.

SNR Range	$L_{\min}$
$E_b/N_0 \leq 3.5$ dB	51
$E_b/N_0 = 3.6$ dB	111
$E_b/N_0 = 3.7$ dB	401
$E_b/N_0 = 3.8$ dB	3001
$E_b/N_0 = 3.9$ dB	5001

It must be mentioned here that the design parameter $L_{\min}$ strikes a trade-off between the BER performance attained and the complexity imposed. A lower value of $L_{\min}$ will result in more error-free blocks, thereby reducing the complexity imposed. However, it will degrade the BER performance of the system. On the other hand, a higher value of $L_{\min}$ will give a better BER performance but at the expense of an increased decoding complexity.

III. RESULTS AND DISCUSSIONS

In order to quantify the reduction in decoding complexity achieved using the proposed BSD-TTCM, we have analyzed the performance of the 8-state TTCM for 8PSK transmissions over an AWGN channel. Furthermore, a block length of 12000 TTCM-8PSK symbols and 6 iterations were used. We have first heuristically determined the optimum design parameter $L_{\min}$ while ensuring that the BSD-TTCM yields the same BER as the conventional TTCM decoder. Since the Hamming weight of the syndrome decreases with the SNR, the optimum $L_{\min}$ has to increase with the SNR to ensure that the performance is not compromised. We have particularly focussed our attention on the high-SNR region (i.e. $E_b/N_0 \geq 3.5$) and the $L_{\min}$ value was appropriately optimized for every 0.1 dB increment in E_b/N_0, as listed in Table I. It must be mentioned here that the optimum $L_{\min}$ for a particular value of E_b/N_0 will depend on the code parameters as well as on the channel type.

The BER performance of our BSD-TTCM based on the design parameter $L_{\min}$ of Table I is compared to that of the conventional TTCM decoder in Fig. 3. Both decoding schemes exhibit a similar performance.

The corresponding reduction in the decoding complexity is quantified in Fig. 4 in terms of:

- **Equivalent number of iterations** *(Right axis)*: Each iteration is weighted by the percentage of the symbols that had to be decoded, which quantified the equivalent (or effective) number of iterations.
- **Percentage of No-Decoding** *(Left axis)*: This quantifies the total number of symbols in the error-free sub-blocks as a percentage of the frame length (i.e. 12000).

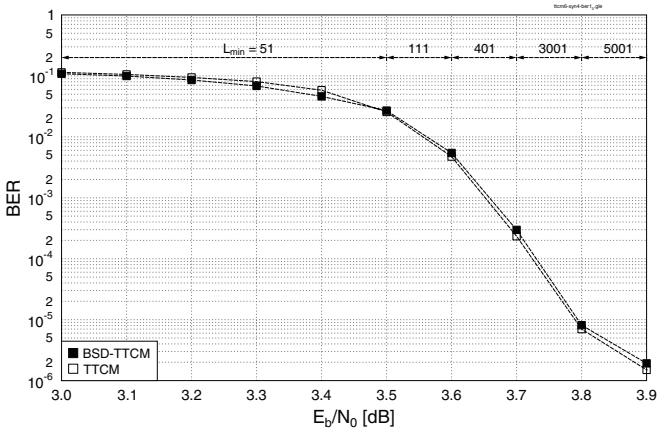

Fig. 3. Comparison of BER performance curve of BSD-TTCM with the conventional TTCM decoding.

In Fig. 4, as E_b/N_0 is increased from 3.0 dB to 3.5 dB for $L_{\min} = 51$, the number of effective iterations is reduced to a minimum of 4.8 at 3.5 dB. This is equivalent to a $(100 \times (6 - 4.8)/6) = 20\%$ reduction in the number of decoding iterations. Furthermore, the percentage of non-decoded symbols for each iteration also increases, reaching a maximum of 45% for the 6^{th} iteration at 3.5 dB. Then, when $L_{\min}$ is increased to 111 at 3.6 dB, the number of equivalent iterations increases to 5. This corresponds to a reduction of $(100 \times (6 - 5)/6) \approx 17\%$ compared to the maximum of 6 iterations and it is therefore still significant. Moreover, the percentage of non-decoded symbols in iterations 2 to 5 decreases, while that in the 6^{th} increases. This is because at this point there are two counter-acting forces:

1) An increased $L_{\min}$ would reduce the number of error-free blocks.
2) An increased E_b/N_0 would decrease the Hamming weight of the syndrome sequence and, therefore, increase the number of error-free blocks.

A similar trend is observed, when E_b/N_0 is increased further. Hence, the proposed scheme reduces the effective number of iterations by at least one, i.e. by 17%, for high SNRs. Furthermore, at least a 20% complexity reduction is achieved for the 5^{th} iteration and 45% for the 6^{th} iteration. We have also benchmarked the performance of our proposed BSD-TTCM decoder against the conventional hard-decision aided high-SNR Early Termination (ET) criterion of [5] in Fig. 4. Our proposed scheme outperforms ET by at least 0.5 iteration at high SNRs.

IV. Conclusions

In this letter, we have conceived a syndrome-based block decoding approach for TTCM. The proposed BSD-TTCM only decodes the blocks deemed to be erroneous, which are identified using the syndrome sequence. Therefore, the decoding complexity is reduced. Furthermore, a pre-correction sequence is estimated at each iteration for reducing the decoding complexity of the forthcoming iterations. We have compared the BER performance of the proposed BSD-TTCM to that of the classic TTCM decoder and quantified the complexity reductions achieved in terms of the number of effective decoding iterations as well as the percentage of non-decoded blocks at each iteration.

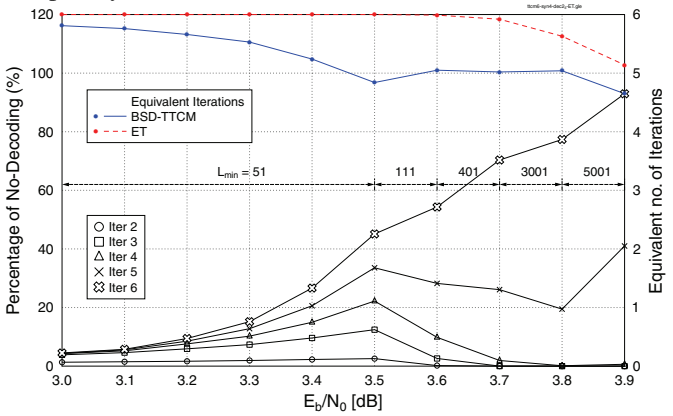

Fig. 4. Reduction in decoding complexity versus SNR per bit for the proposed BSD-TTCM.

References

[1] P. Robertson and T. Worz, "Bandwidth-efficient turbo trellis-coded modulation using punctured component codes," *IEEE J. Sel. Areas Commun.*, vol. 16, no. 2, pp. 206–218, Feb. 1998.

[2] G. Ungerboeck, "Channel coding with multilevel/phase signals," *IEEE Trans. Inf. Theory*, vol. 28, no. 1, pp. 55–67, Jan. 1982.

[3] H. Chen and A. Haimovich, "Exit charts for turbo trellis-coded modulation," *IEEE Commun. Lett.*, vol. 8, no. 11, pp. 668–670, Nov. 2004.

[4] J. Geldmacher, K. Hueske, and J. Gotze, "Syndrome based block decoding of convolutional codes," in *Proc. 2008 IEEE International Symposium on Wireless Communication Systems*, pp. 542–546.

[5] J. Geldmacher, K. Hueske, J. Gotze, and M. Kosakowski, "Low complexity syndrome based decoding of turbo codes," in *Proc. 2012 IEEE International Symposium on Information Theory*, pp. 2371–2375.

[6] J. Schalkwijk and A. Vinck, "Syndrome decoding of convolutional codes," *IEEE Trans. Commun.*, vol. 23, no. 7, pp. 789–792, Jul. 1975.

[7] M. Tajima, K. Shibata, and Z. Kawasaki, "Relation between encoder and syndrome former variables and symbol reliability estimation using a syndrome trellis," *IEEE Trans. Commun.*, vol. 51, no. 9, pp. 1474–1484, Sept. 2003.

[8] M. Ariel and J. Snyders, "Soft syndrome decoding of binary convolutional codes," *IEEE Trans. Commun.*, vol. 43, no. 234, pp. 288–297, Feb./Mar./Apr. 1995.

[9] V. Sidorenko and V. Zyablov, "Decoding of convolutional codes using a syndrome trellis," *IEEE Trans. Inf. Theory*, vol. 40, no. 5, pp. 1663–1666, Sep. 1994.

[10] J. Geldmacher, K. Hueske, J. Goetze, and S. Bialas, "Adaptive low complexity map decoding for turbo equalization," in *Proc. 2010 International Symposium on Turbo Codes and Iterative Information Processing*, pp. 63–67.

[11] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate (corresp.)," *IEEE Trans. Inf. Theory*, vol. 20, no. 2, pp. 284–287, Mar. 1974.