Analysis of three-dimensional turbo codes

Dhouha Kbaier, Catherine Douillard, Sylvie Kerouedan

Analysis of 3-Dimensional Turbo Codes

D. Kbaier Ben Ismail, C. Douillard and S. Kerouédan

Institut Telecom, Telecom Bretagne. UMR CNRS 3192 Lab-STICC. Technopôle Brest iroise CS 83818 29238 Brest Cedex 3

Université européenne de Bretagne, France

Email: {dhouha.kbaier, catherine.douillard, sylvie.kerouedan}@telecom-bretagne.eu

***Abstract*— Our paper presents a detailed study of the 3-dimensional turbo code (3D TC). This code which combines both parallel and serial concatenation is derived from the classical TC by concatenating a rate-1 post-encoder at its output. The 3D TC provides very low error rates for a wide range of block lengths and coding rates, at the expense of an increase in complexity and a loss in convergence. This paper deals with the performance improvement of the 3D TC. First, we optimize the distance spectrum of the 3D TC by means of the adoption of a non regular post-encoding pattern. This allows us to increase the Minimum Hamming Distance (MHD) and thereby to improve the performance at very low error rates. Then, we propose a time varying (TV) construction of the post-encoded parity in order to reduce the observable loss of convergence at high error rates. Performance comparisons are made between the 3GPP2 standardized TC and the corresponding 3D code. The different improvement stages are illustrated with simulation results, asymptotical bounds and EXIT charts.**

***Keywords*— turbo code; iterative decoding; 3-dimensional turbo code; 3GPP2 code; convergence threshold; time varying trellis.**

I. INTRODUCTION

TCs [1] have been adopted in various communication standards [2-5] due to their near-capacity performance and low decoding complexity. But they suffer from a flattening around 10^{-5} of Frame Error

Rate (FER). In future system generations, lower error rates will be required to open the way to real time and more demanding applications, such as TV broadcasting or videoconferencing.

In [6,7], a 3D TC was introduced, combining both parallel and serial concatenation. It is simply derived from the classical TC by concatenating a rate-1 post-encoder at its output, which encodes only a fraction λ of the parity bits from the upper and lower constituent encoders. The fraction $1-\lambda$ of parity bits which is not re-encoded is directly sent to the channel or punctured to achieve the desired code rate. The 3D TC improves performance in the error floor compared to the TC, at the expense of a loss in convergence and an increase in complexity.

This paper is organised as follows. In section II, we present the 3D TC structure. The decoding process is also briefly discussed. Performance of the 3D TC depends on the interleaving laws, the post-encoder and the permeability rate. In order to optimize these key parameters, a thorough analysis is carried out in the same section. Furthermore, a detailed study of the complexity increase of the 3D TC is available in section III. Then, in section IV, we introduce a method to optimize the 3D TC in order to increase even more the MHD. Several upper bounds on the minimum distance of binary 3D TCs with 8-state upper and lower constituent encoders and 3GPP2 interleavers are presented. In section V, we discuss convergence issues and we introduce a time varying (TV) post-encoder as an alternative to reduce the observable loss of convergence. Finally, section VI draws some conclusions.

II. CODING SCHEME

A. *Encoding Structure*

A block diagram of the 3D turbo encoder is depicted in Fig. 1. In our work, we focused on the 3GPP2 code, an 8-state binary TC, used in the third generation (3G) mobile phone communication systems [5]. The 3GPP2 TC is built from the parallel concatenation of two 8-state Recursive Systematic Convolutional (RSC) codes, with generator polynomials 13 (recursivity) and 15 (redundancy). The overall code rate before puncturing is 1/3.

A fraction λ of the parity bits from the upper and lower constituent encoders are grouped by a parallel / serial (P/S) multiplexer, permuted by a permutation π', and encoded by an encoder of unity rate. In [2,3], λ is referred to as the permeability rate. Usually, very simple regular permeability patterns are applied. For instance, if $\lambda = \frac{1}{8}$ the bits to be post-encoded are chosen in a regular basis {10000000} for both the upper and the lower encoders. Note that the permeability rate has an effect on the performance of the 3D TC similar to the doping ratio concept of [8].

B. *Choice of the Post-Encoder*

The choice of the post-encoder influences the performance in both the waterfall and error floor regions. In general, the post-encoder must be simple to limit the complexity increase of the corresponding decoder, and must not exhibit too much error amplification (see [6,7] for details), to prevent from a high loss in convergence. Low memory RSC codes satisfy this requirement. Three linear RSC codes having memory 2 are given in Fig. 2. Besides, the convolutional code is made tail-biting [9] to prevent from any side effects as the initial state and the final state of the post-encoder are identical. This requirement is important for real–time and demanding applications, such as TV broadcasting or videoconferencing, where very low error rates are sought for. To complete the analysis in [2,3], the choice of the post-encoder is justified by means of EXtrinsic Information Transfer (EXIT) [10] analysis.

In Fig. 3 we report the EXIT curves for the three linear post-encoders of Fig. 2. When no *a priori* information is available at the input of the pre-decoder (*i.e.* first iteration) the Mutual Information (MI) at its output is higher for post-encoder (a). In fact, code (a) has a corresponding decoder which only doubles the number of errors of its input at the first step of the iterative process, while code (b) will roughly triple the number of errors at the first step. The worst case occurs with code (c) because its decoder causes a mistake once every two bits in its entry.

Let us assume that a post-encoder such as code (c), where the MI at its output is zero when there is no MI at its input, has been selected. The worst case occurs when all the parity bits are post-encoded, which corresponds to high coding rates such as code rate $R = \frac{2}{3}$ for $\lambda = \frac{1}{4}$ or code rate $R = \frac{4}{5}$ for $\lambda = \frac{1}{8}$. In this case, the error rate at the output of the corresponding pre-decoder at the first iteration will be 0.5. And the turbo decoder will have no parity to decode with at the first step of the iterative process. It will just be something catastrophic as the performance will not be improved through the iterative process! Therefore, the EXIT analysis is a very important tool to select a post-encoder convenient at low but also at high coding rates.

In Fig. 4, we report the FER performance of the 3GPP2 3D TC to compare it with that of the 3GPP2 TC for the block size 570 bits, at coding rate $R = \frac{1}{3}$ and $\lambda = \frac{1}{4}$. We observe a loss of convergence in the waterfall region when the post-encoder of Fig. 2(a) is used. As expected, this loss of convergence increases when the post-encoder of Fig. 2(b) is used. The largest loss of convergence is observed when the code of Fig. 2(c) is used. Similar simulations at code rate $R = \frac{2}{3}$ for $\lambda = \frac{1}{4}$, not represented in the figure, confirm that the 3D TC does not converge when the code of Fig. 2(c) is selected to be the post-encoder. Therefore, due to its better convergence, code (a) with generator polynomial 5 (recursivity) and 4 (redundancy) has been selected to be the post-encoder in different simulations of the 3D TC. However, the main drawback is that code (a) cannot ensure tail-biting encoding in order to properly deal with blocks of data. In other words, we can not ensure that the initial state and the final state of the post-encoder are identical. Thus, state mapping encoding has been introduced in [11]. The problem can easily be resolved by an exchange of metrics at the end of the forward and backward recursions.

C. *Choice of the Key Parameter λ*

If we choose a large value of λ, the minimum distance is significantly increased. However, the performance in the waterfall region is degraded. For example at coding rate $R = \frac{1}{2}$, we observe a small loss of convergence in the waterfall region of 0.13 dB when the permeability pattern is $\lambda = \frac{1}{8}$, which corresponds to the loss for the convergence threshold of the 3D TC compared with the original code. This loss of convergence increases with λ. It is about 0.22 dB for $\lambda = \frac{1}{4}$. On the other hand, in terms of MHD ($d_{\min}$), the behavior is the opposite, *i*.e. larger minimum distances are obtained by increasing λ. For instance, the use of 3GPP2 3D TC for blocks of 762 bits at coding rate $R = \frac{1}{2}$ results in an increase in $d_{\min}$ by more 63% with $\lambda = \frac{1}{8}$ (from $d_{\min} = 11$ to $d_{\min} = 18$). When the permeability pattern is $\lambda = \frac{1}{4}$, the new value of $d_{\min}$ (*i*.e. 23) exceeds the double of the minimum distance of the classical TC.

Thus, there is a trade-off between performance in the waterfall and error floor regions. In fact, the more redundancies are post-encoded the less redundant information at the first iteration the decoder will have, then causing more errors at its output. In our simulations, $\lambda = \frac{1}{4}$ and $\lambda = \frac{1}{8}$ are considered, since they represent a good trade-off between convergence and MHD.

D. Permutations Π and Π'

The 3D TC is characterized by two permutations denoted by π and π', as shown in Fig. 1. In theory, both permutations should be jointly optimized. However, π is the internal permutation of the TC, and we keep π unchanged with regard to the original code for reasons of backward compatibility. π' is used to spread a fraction λ of the parity bits before feeding them to the post-encoder. In other words, π' is used to spread $P = 2 \times \lambda \times k$ parity bits at the output of the TC before post-encoding. The main role of the permutation π' is to avoid that the pre-decoder returns packages of errors to the entry of the main decoder.

To optimize π', different types of interleavers were tested starting from random permutations to more structured permutations such as the regular interleaver. It was observed through the different simulations that the important property is the spread. In fact, performance of an interleaver is degraded by low values of spread. And the regular permutation is an interleaver achieving a spread of $\sqrt{2P}$ [12], where P is the size of the frame to be post-encoded. So it performs better than a random interleaver in terms of MHD and convergence.

Fig. 5 shows the simulated performance of the 3GPP2 3D TC with random and regular interleavers π' for code rate $R = \frac{1}{2}$, $\lambda = \frac{1}{8}$ and $k = 762$ bits. The 3GPP2 3D TC using a random permutation π' does not perform well in terms of MHD, but also in terms of convergence. However, the use of a regular permutation π' results in an increase in the MHD. For example, an increase by more than 60% is observed in Fig. 5; which provides a gain of more than two decades in the error floor. These simulation results were confirmed with the asymptotical bounds as shown in Fig. 5. In fact, for transmission over the Gaussian channel, the FER can be upperbounded by the union bound:

$$FER \leq \frac{1}{2} \sum_{d \geq d_{\min}} n(d)\ \mathrm{erfc}(R\ d\ \frac{E_b}{N_0})$$

where *n(d)* is the code multiplicity (number of codewords with weight *d*), and erfc(x) is the complementary error function.

E. Decoding Process

The classical turbo principle is used to decode the 3D TC. We have three decoders corresponding to the three constituent encoders and all of them exchange extrinsic information. First, the 4-state Soft Input/Soft Output (SISO) pre-decoder is activated to feed the two 8-state decoders with extrinsic information about the post-encoded parity bits. The two 8-state decoders exchange extrinsic information about the systematic bits, as for the classical turbo procedure. They also provide the pre-decoder with extrinsic information about the

post-encoded parity bits. The decoding process continues iteratively until all constituent decoders have converged, or a maximum number of iterations have been performed.

Compared to a classical turbo decoder, the additional complexity is mainly due to the implementation of the binary 4-state decoder but also to the calculation of the extrinsic information about the post-encoded parity bits. A thorough analysis of the 3D turbo decoder complexity is carried out in section III. Information is also available in [13].

III. Complexity Analysis of 3-Dimensional turbo decoders

In [3], the complexity increase was estimated to be less than 10% with respect to classical 2-dimensional TC. In this section, we propose a more detailed analysis of the complexity of a 3D TC. In fact, compared to a classical turbo decoder, the additional complexity of the 3D turbo decoder is mainly due to the implementation of the binary 4-state decoder but also to the calculation of the extrinsic information about the post-encoded parity bits.

A. 3D Turbo Decoder Architecture

The typical overall turbo decoder architecture is composed of three modules, represented in Fig. 6. First, the input module receives the input frames and transmits them to the decoder module. It requires a double input buffer, in order to receive the next frame while decoding the current one. The input buffer is divided into as many memory banks (MB) as the number of processors placed in parallel (i.e. P). This parallelism allows having different throughputs according to the application. Then, the decoder module performs I iterations on the frame stored in the input module and writes the decoded codeword into the output module. This module contains P SISO processors and an extrinsic memory decomposed into as many memory banks as the number of physical processors (not represented in the figure). A finite state machine (not represented) controls the processors. For each iteration, the set of P processors has to perform the decoding of the component codes. The output module stores the hard decisions produced by the decoder module and sends them to the output of the decoder. In the case of 3D TC, since the pre-decoder has much

less data to process than the main SISO decoders (only $\lambda = \frac{1}{8}$ or $\lambda = \frac{1}{4}$ of the parity bits are re-encoded by the post-encoder), no parallelism is considered for the pre-decoder.

B. Max-Log-MAP Decoder Complexity Analysis

To analyse the complexity of 3D TC, let us consider a RSC code with the following parameters: ν is the memory length of the code, n is the number of coded bits provided by the encoder at each trellis stage when no puncturing is performed, and k is the trellis length. For a classical binary TC, k is also the length of information sequence, in terms of binary bits. This section details the different steps of the decoding process and the associated decoding complexity, in terms of arithmetic and logical operations. We assume a transmission over an AWGN channel with noise variance σ^2, using BPSK modulation. The following description of the algorithm is also valid when high order modulations are considered; in the case where a Bit-Interleaved Coded Modulation (BICM) [14] approach is adopted.

- Computation of branch metric $met_t(s', s)$:

At time step *t*, the metric associated with trellis branch or transition *(s',s)* is defined as

$$met_t(s', s) = \pm x_t \pm y_{t,1} \ldots \pm y_{t,n-1} + z_t \qquad (1)$$

where x_t is the received systematic data, $y_{t,1} \ldots y_{t,n-1}$ are the $n-1$ received redundant data, and z_k is the *a priori* incoming information. The computation of the 2^n different values $\pm x_t \pm y_{t,1} \ldots \pm y_{t,n-1}$ requires $2^{n+1} - 4$ additions/subtractions. The addition of the *a priori* term requires two extra additions. We assume that the computation of the branch metrics is performed twice, once for the forward recursion and once for the backward recursion.

- Computation of forward and backward state metrics for each trellis stage *s*:

The state metrics are computed recursively using the following relations:

Forward recursion (for $t = 1, \ldots k$):

$$M_t^F(s) = \min_{s' \in \{0,\ldots 2^\nu - 1\}} \left(M_{t-1}^F(s') + met_{t-1}(s', s) \right) \qquad (2)$$

Backward recursion (for $t = k - 1,\ldots 0$):

$$M_t^B(s) = \min_{s' \in \{0,\ldots 2^\nu - 1\}} \left(M_{t+1}^B(s') + met_t(s, s') \right) \qquad (3)$$

According to the equations above, the update of one forward state metric, involves the comparison and selection of two concurrent paths that can be performed using 2 additions and one comparison-selection operation, implementing a tree structure. The update of backward state metrics requires the same number of operations.

- Computation of soft decisions and hard decisions:

If we denote by $\lambda_t(\delta)$ the soft information defined as

$$\lambda_t(\delta) = \min_{(s',s)} \left(M_t^F(s') + met_t(s', s) + M_{t+1}^B(s) \right) \qquad (4)$$

where $\delta \in \{0,1\}$, the *a posteriori* log-likelihood related to data δ at time step *t* is computed as

$$L_t(\delta) = \frac{1}{2} \left(\lambda_t(\delta) - \min_{\delta' \in \{0,1\}} \lambda_t(\delta') \right) \qquad (5)$$

Term $\min_{\delta' \in \{0,1\}} \lambda_t(\delta')$ is a normalization term.

The hard decision provided by the decoder corresponds to the binary representation of δ that minimizes $\lambda_t(\delta)$ and makes $L_t(\delta)$ equal to zero.

$$\hat{\delta} = \arg\min_{\delta \in \{0,1\}} (L_t(\delta)) = \arg\min_{\delta \in \{0,1\}} (\lambda_t(\delta)) \qquad (6)$$

The computation of two *a posteriori* LLRs requires the computation of two values of $\lambda_t(\delta)$, $\delta \in \{0,1\}$. Relation (4) involves two additions for each transition in the trellis. This complexity can be reduced to one

addition by observing those partial terms $M_t^F(s') + met_t(s', s)$ or $met_t(s', s) + M_{t+1}^B(s)$ are already available through the forward or backward recursion.

For each value of δ, the minimum value of 2^ν terms $M_t^F(s') + met_t(s', s) + M_{t+1}^B(s)$ has to be computed, resulting in $2^\nu - 1$ compare-select operations, using a tree structure. Consequently, the computation of two values for $\lambda_t(\delta)$ requires $2^{\nu+1}$ additions and $2\left(2^\nu - 1\right)$ comparisons and selections.

The computation of two *a posteriori* LLRs in (5) requires a compare and select tree to compute the min term in (5), that is one compare and select operation, two subtractions and two divisions by 2. Actually the subtraction in the case of $\lambda_t\left(\hat{\delta}\right)$ can be avoided, since $L_t\left(\hat{\delta}\right) = 0$ and the number of subtractions can be reduced to one. Divisions by 2 are not taken into account in the operator calculation, since they only come to remove the least significant bit. The hard decision $\hat{\delta}$ can be directly inferred from the compare and select tree allowing the minimum value of $\lambda_t(\delta)$ to be computed.

- Computation of extrinsic information $L_t^e(\delta)$ related to information symbols:

The extrinsic information computation is similar to the *a posteriori* log-likelihood $L_t(\delta)$, using the extrinsic branch metrics. We compute the extrinsic soft information $\lambda_t^e(\delta)$ defined as

$$\lambda_t^e(\delta) = \min_{(s',s)}\left(M_t^F(s') \pm y_{t,1} \ldots \pm y_{t,n-1} + M_{t+1}^B(s)\right) \qquad (7)$$

where the *min* operation is performed among 2^ν transitions corresponding to data value δ. Then, the extrinsic log-likelihood $L_t^e(\delta)$ is computed as follows

$$L_t^e(\delta) = \frac{1}{2}\left(\lambda_t^e(\delta) - \lambda_t^e\left(\hat{\delta}\right)\right) \qquad (8)$$

The term subtracted to $\lambda_t^e(\delta)$ is the extrinsic value corresponding to the hard decision $\hat{\delta}$.

If we assume that terms $\pm x_t + z_t$ have already been made available during the branch metrics computation step (equation (1)), each piece of extrinsic information is obtained from the *a posteriori* LLR with two subtractions. The total extrinsic information computation is then performed using 2^2 subtractions.

- Computation of extrinsic LLRs related to redundancy bits:

In the case of the 3D TCs, additional extrinsics related to re-encoded redundancy bits have to be computed by the main SISO decoders. Each additional extrinsic LLR is computed from the following relation:

$$L_t^y = \frac{1}{2}\left(\min_{(s',s)/\, y_t=0} \left(M_t^F(s') + met_t(s', s) + M_{t+1}^B(s)\right) - \min_{(s',s)/\, y_t=1} \left(M_t^F(s') + met_t(s', s) + M_{t+1}^B(s)\right) \right) - y_t \qquad (9)$$

where *y* is the considered redundancy bit.

Observing that terms $M_t^F(s') + met_t(s', s) + M_{t+1}^B(s)$ are already available, we only have to compute the minimum value of these terms for value redundancy 0 and for redundancy 1. Thus two minimum values have to be computed among $2^{\nu+2}$ terms, resulting in using two tree structures requiring $2^{\nu} - 1$ compare-select operations each. Then, the extrinsic LLR is computed by subtracting these two values, dividing by 2 and subtracting the received redundancy bit. Consequently, the computation of each additional extrinsic redundancy value requires 2 subtractions and $2\left(2^{\nu} - 1\right)$ compare-select operations.

Table I summarizes the resulting complexity for the process of a trellis stage, or equivalently of an information bit. In order to compare the complexity of the different families of decoders, it is assumed that addition/subtraction and compare-select operators have similar hardware complexity. This complexity assessment does not take the size of the operators into account.

C. Memory Requirements for the 3D Turbo Decoder

The memory requirements for the turbo decoder are the amount of both RAM and ROM memory. A very small amount of ROM memory is required to store the TC permutation parameters. This amount of memory is the same for all coding schemes under consideration. But for the RAM memory, two input

buffers are necessary for each data sequence, including systematic and parity bits, stemming from the transmission channel. Thus, if k is the length of the information sequence, $\frac{2k}{R}$ input samples, quantized on q_x bits, have to be stored at the decoder input. In addition, $2k$ extrinsics (dual-port RAM) need to be stored (quantized on $q_x + 1$ bits). For a 3D TC, additional extrinsics ($2 \times \lambda \times k$) related to re-encoded redundancy bits need to be stored. Then, the hardware decision at the decoder output requires k memory bits (single-port RAM). Inside the SISO decoding processors, state metrics have to be stored at each iteration. The straightforward application of the Max-Log-MAP algorithm requires storing $k.2^{\nu}$ state metrics (either forward or backward). This can represent an unaffordable amount of memory for large k. In order to overcome this limitation, sliding window [15] processing can be implemented. The decoding length is then limited to a given truncated length (TL) rather than to the frame length k. This allows the overall decoding delay and the memory requirement to be reduced. Only $TL.2^{\nu}$ state metrics have to be stored. In practice, a window size equal to $TL = 32$ represents a good trade-off between complexity and performance.

D. Summary

Table II compares the hardware complexity of the 3GPP2 turbo decoder and the corresponding 3D decoder when $\lambda = \frac{1}{8}$ is used: $k = 6138$ bits and $R = \frac{1}{2}$, for the worst case in terms of memory size. Table II provides the complexity of the overall hardware dedicated to SISO decoding with the Max-Log-MAP algorithm in terms of add / compare-select operators; and the amount of RAM memory required for the implementation, in terms of equivalent single-port RAM bit (we assume that one dual-port RAM bit is equivalent to two single-port RAM bits). The number of SISO decoders placed in parallel, P, depends both on the required data throughput and on the hardware implementation technology. Table II presents complexity figures for $P = 1$, $P = 2$ and $P = 4$. This complexity assessment does not take the size of

the SISO internal operands into account. The implementation of the control part (state machines) and interleavers is not taken into account either. Note that the complexity of the state machines does not differ a lot between the different families of decoders.

The authors in [16] provide a detailed comparison on the 3D decoder's complexity for a duo binary TC. They consider the implementation complexity on FPGA and in 65nm ASIC technology. According to their approach, an additional complexity between 20% and 40%, depending on the implemented technique, is required for the 3D configuration compared to the classical turbo decoder. Their results are coherent with what we have obtained. This brings a complementary view of our analysis, dealing with computational complexity and memory requirements.

To conclude, the first estimation of the complexity in [3] was optimistic. And Table II shows that the more important the degree of parallelism, the less the impact in terms of relative additional complexity of using a 3D TC.

IV. Increasing the Minimum Hamming Distance of the 3-Dimensional Turbo Code

A. *Performance of the 3-Dimensional Turbo Code without Optimization*

We have investigated the effect of adding a third dimension to the 3GPP2 TC on the distance gain and on the convergence threshold for different block sizes, coding rates and permeability rates. Similarly to the case of double-binary codes in [2,3], we have observed that the addition of the post-encoder improves the asymptotical behavior of the 3GPP2 TC in many cases. Table III presents examples of MHD values obtained with this code for the block size $k = 762$ bits and different coding rates, using the all-zero iterative decoding algorithm [17]. We can observe that the direct application of the third coding dimension to the existing code leads to an increase of its minimum distance, except in the case of high coding rates.

The FER performance of the 3GPP2 3D TC has been simulated with $\lambda = \frac{1}{8}$ and $\lambda = \frac{1}{4}$. Then one bit out of eight (respectively one bit out of four) is regularly picked from each of the parity streams starting

with the first bit from each stream. Fig. 7 shows that the use of 3D TC results in an increase in the MHD by more than 28 % for code rate $R = \frac{1}{2}$, $\lambda = \frac{1}{8}$ and k = 1530 bits (where k is the number of data bits), compared to the standardized 3GPP2 TC. For $k = 1146$ bits, code rate $R = \frac{2}{3}$ and $\lambda = \frac{1}{4}$, the increase in d_{min} is even larger, more than 70%, (not represented in the figure). It is possible to increase even more this gain in distance by using an irregular pattern of permeability as explained in the following paragraph.

B. Optimization Method

To obtain the distance spectrum of the 3GPP2 3D TC, we apply the all-zero iterative decoding algorithm [17]: this technique is based on the transmission of an all-zero sequence corrupted by an impulsive noise. It allows us to determine low weight codewords and to estimate their multiplicity.

We have observed in the distance spectrum of the 3GPP2 3D TC that the first terms have a low multiplicity. The idea is to eliminate the corresponding codewords in order to increase the minimum distance d_{min}. Therefore, we have modified the pattern of post-encoding, which is no more regular, to generate more ones in the codeword with the lowest weight. The following algorithm illustrates the principle of the method:

1. Consider the codeword with the lowest weight.
2. Extract the addresses where the systematic bit x, the parity bit y or the post-encoded parity bit w is equal to one.
   - ✓ If the systematic bit x is one and the corresponding parity bit y does not benefit from the regular post-encoding, include this address in the new post-encoding pattern.
   - ✓ If the systematic bit x is zero, but the parity y is one, check whether the address of the parity bit benefits from the regular post-encoding. If not, include this address also in the new post-encoding pattern.

- ✓ If the post-encoded parity *w* is equal to one, do not modify the pattern for the corresponding address.
- ✓ If there are several low weight codewords, go to step 2.

3. Finally, adapt the pattern of post-encoding in order to take into account the previous constraints. The addresses which will not any more benefit from the post-encoding are randomly selected. However, it is preferable to spread the modifications on all the length of the frame, not to discriminate a given region.

Note that each change of an address in the post-encoding pattern involves four exchanges of parity bits. Let us assume that N_{tot} is the total number of modifications necessary to generate the irregular pattern of post-encoding. Then the distance of the code may be increased or in the worst case reduced by $4N_{tot}$. The following subsections show the application of this optimization in two particular cases of the 3GPP2 3D code.

C. Optimization Results for k = 1530, R = 1/2 and λ =1/8

Table IV provides the first terms of the distance spectrum of the 3GPP2 3D TC obtained for $k = 1530$ bits, $R = \frac{1}{2}$ and $\lambda = \frac{1}{8}$. In this case, the post-encoding occurs regularly for the bits which address modulo 8 is equal to 1. Table IV shows that there is only one codeword with weight 18. We have noticed that there are ones in the systematic part for the systematic bits at addresses {498, 512, 610, and 624}. There are also ones in the redundant part for the parity bits at addresses {350, 352, 356, 499,501, 507, 511, 611, 613, 619, 623, 782, 784, and 788}. However, all the post-encoded bits are equal to zero in the codeword with weight 18. Therefore, we have changed the pattern of post-encoding to generate more ones and to eliminate this codeword with the lowest weight. In fact, we have introduced an irregularity in the previous pattern of post-encoding in order to postcode the bits at addresses {499, 501, 507, 511, 611, 613, 619 and 623} instead of {9, 81, 241, 401, 785, 961, 1121 and 1361}.

The codeword with weight 18 was eliminated. However, it was not possible to eliminate the codeword with weight 20, since there are many low weight codewords in the distance spectrum. Indeed, it was noticed that there are many common addresses containing ones in the different codewords of this distance spectrum. In other words, one codeword differ from the others by only few addresses. It makes it difficult to find an irregular pattern of post-encoding that leads to a huge increase in the minimum distance. So, the minimum distance was finally increased by 2, and the new distance of the optimized 3GPP2 3D TC is 20.

Now compared to the standardized 3GPP2 TC, the optimization of the post-encoding pattern resulted in an increase in d_{min} by more than 42 % for code rate $R = \frac{1}{2}$, $\lambda = \frac{1}{8}$ and $k = 1530$ bits, which provides a gain of 2.5 decades in the error floor as shown in Fig. 7 with the asymptotical bounds.

D. Optimization Results for k = 1146, R = 2/3 and λ = 1/4

The same kind of optimization was performed with another frame length $k = 1146$, $R = \frac{2}{3}$ and $\lambda = \frac{1}{4}$. In this case, there is only one codeword with weight 12, three codewords with weight 15, and all the other codewords are with very high weight. In the codeword with the lowest weight (*i*.e. 12), there are only ones in the systematic part for the systematic bits at addresses {586, 587, 591, 650, 651,655, 763, 764, 768, 1019, 1020, 1024}, and the corresponding parity bits do not benefit from the post encoding which occurs regularly for the bits which address modulo 4 is equal to 1. To optimize the 3D TC, we have slightly modified the permeability pattern in order to postcode the bits at addresses {585, 587, 650, 651, 763 and 764} instead of {9, 101, 581, 925, 1029 and 1133}. Also, we have chosen to spread our modifications on all the length of the frame, not to discriminate a given region.

The idea was easier to implement, compared with $k = 1530$ bits, since there were few low weight codewords in the spectrum. Consequently, we have succeeded in eliminating the codewords with lower weights ((*i.e.*, 12 and 15) at once. The new minimum distance of the optimized 3D turbo code is 33. This value has to be compared to 7 which is the distance of the standardized 3GPP2 TC. The use of optimized

permeability patterns resulted in a huge increase in d_{min} for code rate $R = \frac{2}{3}$, $\lambda = \frac{1}{4}$ and $k = 1146$ bits. The spectrum has changed: some codewords disappeared and other codewords appeared with new distances, but all of them are largely greater than 12 or 15.

These results are optimistic as they encourage implementing the optimization method for high coding rates, in order to increase even more the MHD of the 3D TC in this case.

V. Reducing the convergence loss of 3-Dimensional Turbo Codes

The use of the 3D TC significantly increases the MHD, at the expense of an increase in complexity and a loss of convergence at high error rates. In this section, we analyse the convergence threshold of 3D TCs using the EXIT chart, introduced by Stephan ten Brink [10], and we propose a method to reduce this loss of convergence.

A. Determination of the Convergence Threshold of 3-Dimensional Turbo Codes

In the case of 3D TC, the two 8-state SISO decoders exchange extrinsic information about the systematic part of the received codeword, like for classical turbo decoding. But both of them exchange also extrinsic information about the post-encoded parity bits with the 4-state SISO pre-decoder and we have to take into account in the EXIT chart that the extrinsic information about these parity bits is changing from an iteration to the other. Consequently, the curves of mutual information exchange between the two decoders change every iteration.

For example, the convergence threshold of a 3D TC at code rate $R = \frac{2}{3}$ and $\lambda = \frac{1}{8}$ is 1.55 dB. In fact, for signal to noise ratios (SNRs) lower than $E_b/N_0 = 1.55$, the EXIT curves intersect for input and output mutual information values less than 1 and the iterative process cannot converge. Thus, the exchange of the extrinsic information between the two decoders about the systematic part of the received codeword is

blocked. Whereas for $E_b/N_0 = 1.55$ dB, the tunnel between the EXIT curves is open, and the exchange of the extrinsic information continues along the iterations until we reach the intersection point (1,1). These results are confirmed by the simulations of the code. On the other hand, we computed the convergence threshold for binary 8-state TCs at code rate $R = \frac{2}{3}$, equal to 1.49 dB. The comparison of the previous results shows a loss of 0.06 dB (1.55 dB – 1.49 dB) for the convergence threshold of the 3D TC compared with the original code, at code rate $R = \frac{2}{3}$ and $\lambda = \frac{1}{8}$. The curves of the error rate performance are available in [18].

If we use a larger permeability rate λ, the loss of convergence threshold is more important. For instance, the observable loss of convergence for code rate $R = \frac{2}{3}$ and $\lambda = \frac{1}{4}$ is 0.18 dB. Even more, this loss of convergence increases when the code rate decreases. For example, the observable loss of convergence for code rate $R = \frac{1}{3}$ and $\lambda = \frac{1}{4}$ is 0.26 dB.

B. Time Varying 3-Dimensional Turbo Codes

In order to reduce the loss of convergence of 3D TCs, we propose the adoption of a time varying (TV) encoder as a rate-1 post-encoder.

1) *Time Varying Encoding*: To simulate 3D TC, we have used the RSC code (5,4) as a post-encoder (details are available in [18]). In order to obtain a TV trellis, two redundancies $W_1 = 4$ and $W_2 = 7$ can be alternated in time instead of having only one. The recursivity polynomial remains unchanged, that is 5. We denote by (5,4:7) the polynomials of this new code. This idea was first introduced in [19]. If we look at the trellis of this code, we can easily identify two different paths corresponding to the all–zero sequence. So the decoder will not be able to distinguish between them and the distance of the code is only 2, compared with 3 for the RSC code (5,4).

In order to get closer to the code (5,7) which distance is 5, the idea is to replace periodically some redundancies W_1. The replacement period is denoted by L. Then the challenge involves finding a value of L which improves the performance of the 3D TC, without loosing a lot in convergence at the first iteration. The code (5,7) has a corresponding decoder which triples the number of errors of its input at the first step of the iterative process. The code (5,4) has a corresponding decoder which only doubles the number of errors of its input at the first iteration:

$$BER_{out} = 2 \times BER_{in}$$

where BER_{in} is the channel error rate. Using a TV post encoder increases the BER at its output. The BER at the first step is then expressed in the following way:

$$BER_{out,TV} = 2 \times (BER_{in} + \xi)$$

where ξ is an additive error rate at the output of the pre-decoder at the first step of the iterative process.

On the other side, we have by definition:

$$BER_{out,TV} = \frac{\text{Number of erroneous bits with TV}}{\text{Number of blocks} \times k}$$

$$BER_{out,TV} = \frac{nb + 3 \times R_p}{N_{blocks} \times k}$$

where nb is the number of erroneous bits without TV, N_{blocks} is the number of frames of k bits and R_p is the number of replacements. The term $3 \times Rp$ results from the three input values which cannot be inferred from parity for each replacement [19]. Therefore:

$$BER_{out,TV} = BER_{out\ without\ TV} + \frac{3 \times R_p}{N_{blocks} \times k}$$

$$= 2 \times BER_{in} + \frac{3 \times R_p}{N_{blocks} \times k}$$

If we compare the last equation to the first one, we can identify:

$$2 \times \xi = \frac{3 \times R_p}{N_{blocks} \times k}$$

Finally, we obtain:

$$\xi = \frac{3}{2} \frac{\text{Number of replacements per bloc}}{k}$$

The number of replacements per block is equal to $ceil\left[\frac{k-3}{L}\right] + 1$. The term $(k-3)$ is due to the first replacement which occurs in the beginning for the third parity (*i.e.* instead of alternating W_1, W_2, $\boldsymbol{W_1}$, W_2, W_1…we will have W_1, W_2, $\boldsymbol{W_2}$, W_2, W_1…). At the end, the evolution of ξ according to L results from the formula:

$$\xi = \frac{3}{2} \frac{ceil\left[\frac{k-3}{L}\right] + 1}{k}$$

The TV theoretical analysis shows that the loss of convergence is more significant at the first iteration for L small, as the number of the redundancies W_2 exceeds that of the redundancies W_1. In this case, the TV encoder is closer to the code (5,7) which decoder triples the number of errors of its input. From the convergence point of view, it is preferable to choose L high. However, from the asymptotic performance point of view, it is better to choose L small because the code (5,7) has the higher distance, that is 5. Thus, the optimal value of L is a convergence/distance trade-off. More information about this topic is also available in [18].

2) <u>EXIT chart analysis</u>: Fig. 8 shows the EXIT chart of a TV 3D TC for $E_b/N_0 = 1.58$ dB at code rate $R = 2/3$ and $\lambda = 1/4$. This signal-to-noise value is actually the convergence threshold of the TV 3D TC. We can observe that the transfer characteristics of the two decoders are no more symmetric. In fact, for the post encoder, two redundancies $W_1 = 4$ and $W_2 = 7$ are alternated in time, but W_1 is periodically replaced by W_2 with period $L = 30$. This replacement generates the asymmetry between the transfer

characteristics of the two 8-state SISO decoders. Note that, after the seventh iteration, the transfer characteristics remain almost unchanged.

For SNRs lower than 1.58 dB, the curves have intersection points different from the point (1,1). Then, the iterative process starting with zero average MI in entry cannot end in a perfect determination of the message. E_b/N_0 equal to 1.58 dB is the minimum SNR value where the tunnel between the EXIT curves opens (see Fig. 8). These results were confirmed by simulations of the code. On the other hand, the convergence threshold of the TC at code rate $R = 2/3$ is estimated around 1.49 dB and that of the 3D TC is 1.67 dB. As a conclusion, the use of TV post-encoder reduced the loss of convergence by 50% from 0.18 dB (1.67 -1.49) to 0.09 dB (1.58- 1.49) at code rate $R = 2/3$ and $\lambda = 1/4$. And among the simulated cases it was observed that the TV parity construction reduces the observable loss of convergence by 10% to 50% of the value expressed in dB. In [13], we have shown that when the code is associated with high order modulations, there is no need to use a TV trellis and a specific mapping allows obtaining even a gain in the waterfall region. Therefore, the 3D TC is adapted to be used in high spectral efficiency transmission schemes.

VI. CONCLUSIONS

The 3D TC recently introduced by Berrou *et al.*, calls for both parallel and serial concatenation and increases the minimum distance with respect to the classical TCs. In this paper we discussed how to choose a post-encoder by means of an EXIT analysis. Then, a complexity study of 3D TCs was presented to estimate the additional complexity. When high throughputs are required for a given application, several processors can be placed in parallel; which decreases the relative additional complexity of the 3D coding scheme.

Afterwards, we focused on the improvement of this hybrid concatenated structure. In fact, in the case of the 3GPP2 code, a specific optimization of the permeability pattern allows to improve even more performance in the error floor region. Finally, the use of a TV post-encoder reduces the loss of convergence of the 3D TC without degrading its asymptotic performance. So, it is possible to build 3D TCs which have good performance in both the waterfall error floor regions. And this code structure is expected to reach a performance/complexity trade-off never yet attained.

REFERENCES

[1] C. Berrou, A. Glavieux, and P. Thitimajshima, “Near Shannon limit error-correcting coding and decoding: turbo-codes,” IEEE International Conference on Communications. Geneva, Switzerland, May 1993, pp. 1064–1070.

[2] Third Generation Partnership Project (3GPP) Technical Specification Group, “Multiplexing and channel coding (FDD),” June 1999, TS 25.212, v2.0.0.

[3] DVB,”Interaction channel for satellite distribution systems,“ December 2000, ETSI EN 301 790, v. 1.2.2.

[4] DVB, “Interaction channel for digital terrestrial television,” Mars 2001, ETSI EN 301 958, v. 1.1.1.

[5] Third Generation Partnership Project 2 (3GPP2), “Physical layer standard for cdma2000 spread spectrum systems, Release D,” Feb. 2004, 3GPP2 C.S0002-D, Version 1.0.

[6] C. Berrou, A. Graell i Amat, Y. Ould-Cheikh-Mouhamedou, C. Douillard, and Y. Saouter, “Adding a rate-1 third dimension to turbo codes,” in Proc. IEEE Inform. Theory Workshop, Lake Tahoe, CA, Sep. 2007, pp. 156–161.

[7] C. Berrou, A. Graell i Amat, Y. Ould-Cheikh-Mouhamedou, and Y. Saouter, “Improving the distance properties of turbo codes using a third component code: 3D turbo codes,” IEEE Trans. Commun., vol. 57, no. 9, Sep. 2009, pp. 2505–2509.

[8] S. ten Brink, “Code doping for triggering iterative decoding convergence,” in Proc. IEEE Int. Symp. Inf. Theory, Washington, DC, June 2001, p. 235.

[9] C. Weiß, C. Bettsteter, and S. Riedel, “Code construction and decoding of parallel concatenated tail biting codes,” IEEE Trans. Inform. Theory,vol. 47, Jan. 2001, pp. 366-386.

[10] S. Ten Brink, “Convergence behaviour of iteratively decoded parallel concatenated codes,” IEEE Trans. Commun., vol. 49, no. 10, Oct. 2001, pp. 1727–1737.

[11] J. Sun and O. Y. Takeshita, “Extended tail-biting schemes for turbo codes,” IEEE Commun. Letters, vol. 9, Mar. 2005, pp. 252–254.

[12] E. Boutillon and D. Gnaeding, “Maximum spread of D-dimensional multiple turbo codes,” IEEE Trans. Commun., vol. 53, no. 8, Aug. 2005, pp. 1237–1242.

[13] D. Kbaier Ben Ismail, C. Douillard, and S. Kerouédan, “Reducing the convergence loss of 3-dimensional turbo codes,” 6th International Symposium on Turbo Codes and Related Topics. Sep. 2010, pp. 146-150.

[14] G. Caire, G. Taricco and E. Biglieri, “Bit-interleaved coded modulation,” IEEE Trans. Inform. Theory, vol. 44, no. 3, 1998, pp. 927-946.

[15] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, “Soft-output decoding algorithms in iterative decoding of turbo codes”, Jet Propulsion Laboratory TDA Progress Report 42-124, Feb. 1996, pp. 63-87.

[16] T. Lehnigk-Emden, M. Alles, N. Wehn, “3D duo binary turbo decoder hardware implementation,” In Proc. ICT Mobile and Wireless Communications Summit (ICT-MobileSummit 2009), Santander, Spain, June 2009.

[17] R. Garello and A. Casado, “The All-Zero Iterative Decoding Algorithm for Turbo Code Minimum Distance Computation,” IEEE International Conference on Communications, Paris, France, June 2004, pp. 361–364.

[18] D. Kbaier Ben Ismail, C. Douillard, and S. Kerouédan, “Improving 3-dimensional turbo codes using 3GPP2 interleavers,” ComNet'09: 1st International Conference on Communications and Networking, Hammamet, Tunisia, Nov. 2009.

[19] C. Berrou, A. Graell i Amat and Y. Ould-Cheikh-Mouhamedou, “About rate-1 codes as inner codes,” 5th International Symposium on Turbo Codes and Related Topics, Lausanne, Switzerland, Sep. 2008.

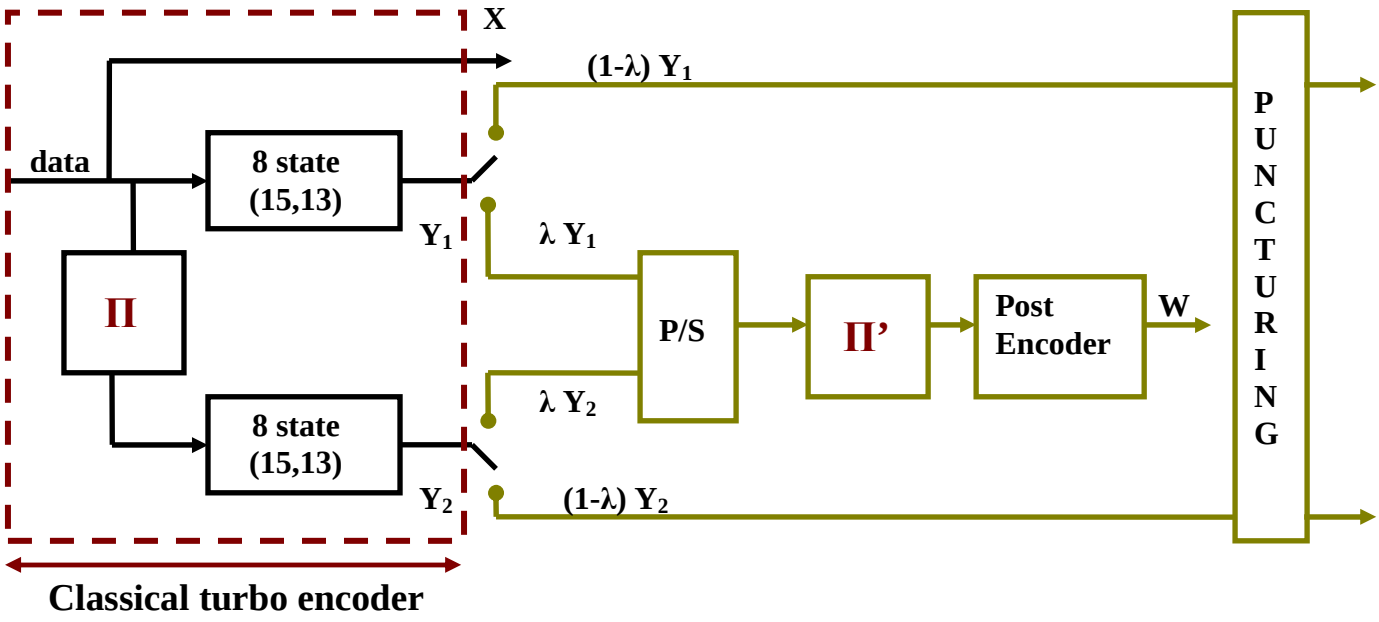

Figure 1. 3D turbo encoder structure. A fraction λ of the parity bits from both component encoders are grouped by a P/S multiplexer, permuted by the permutation Π', and encoded by a rate-1 post-encoder.

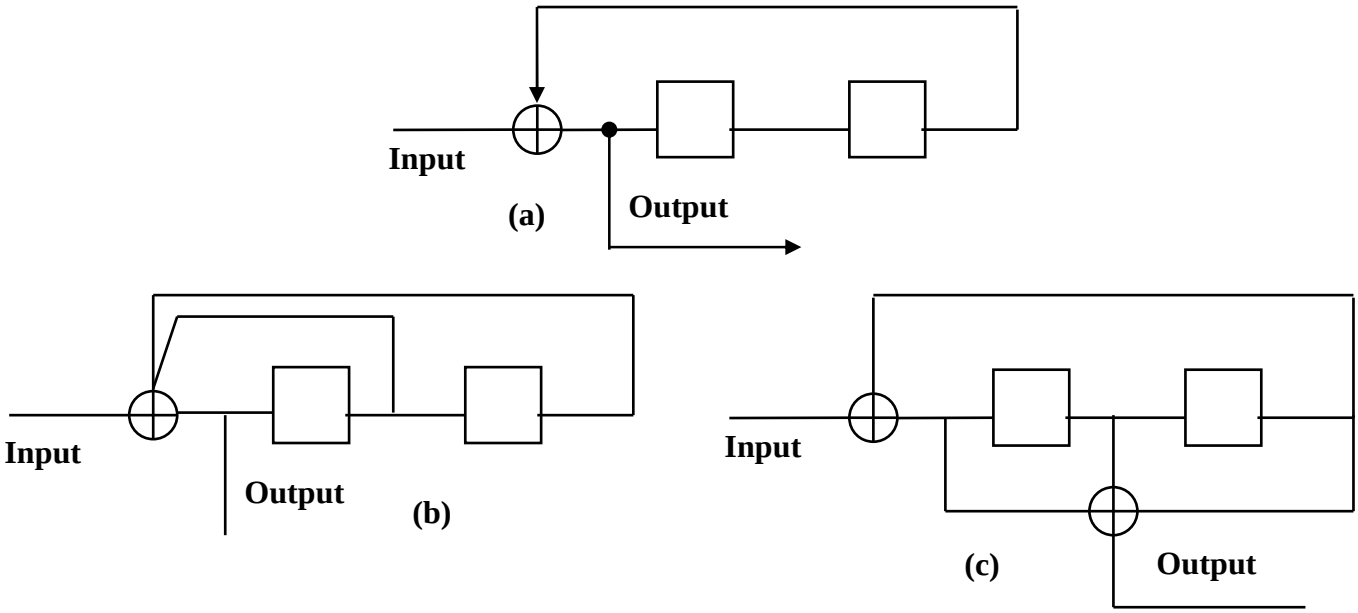

Figure 2. Possible linear post-encoder candidates with memory 2.

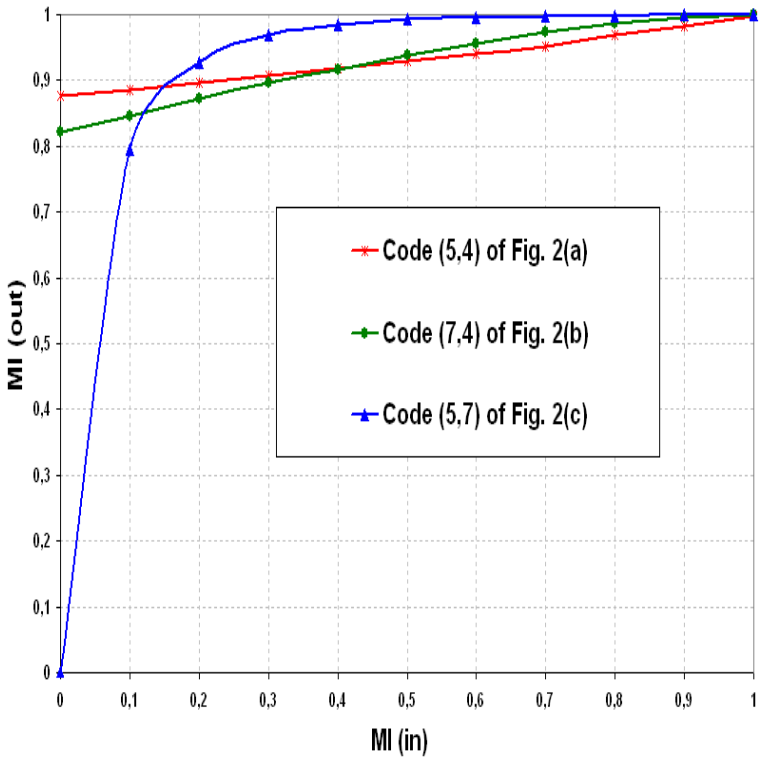

Figure 3. EXIT curves for different linear post-encoders.

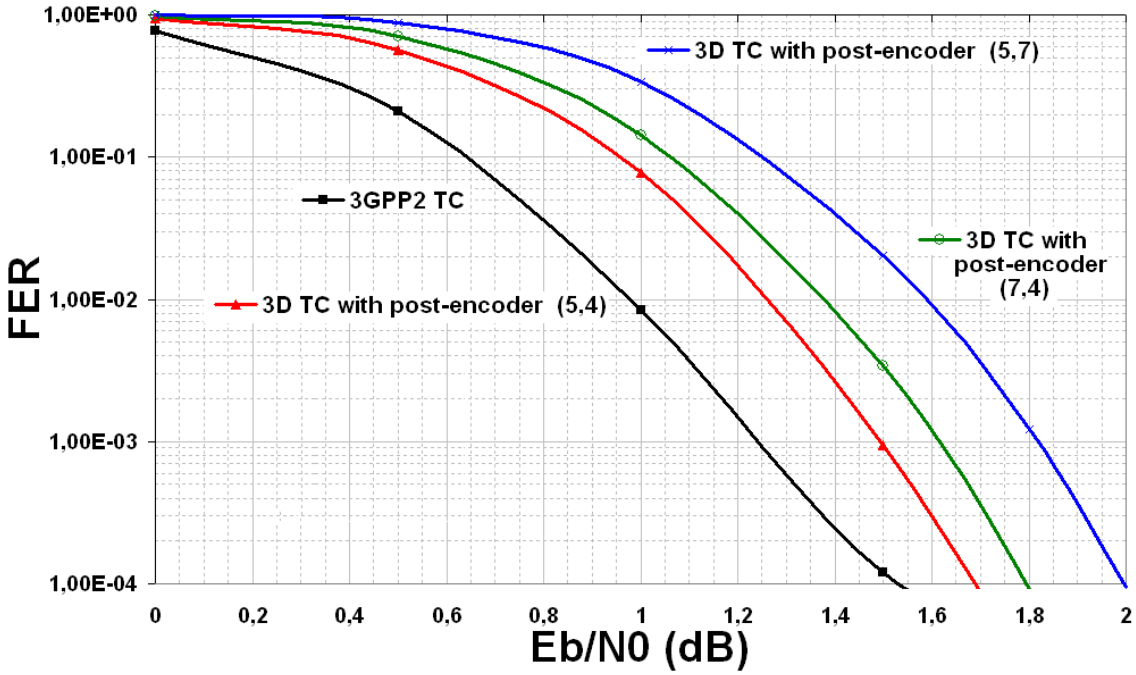

Figure 4. FER performance of the 3GPP2 3D TC with $\lambda = \frac{1}{4}$ for $k = 570$ bits, $R = \frac{1}{3}$ and comparison with the 3GPP2 TC. All simulations use the Max-Log-MAP algorithm with 10 decoding iterations.

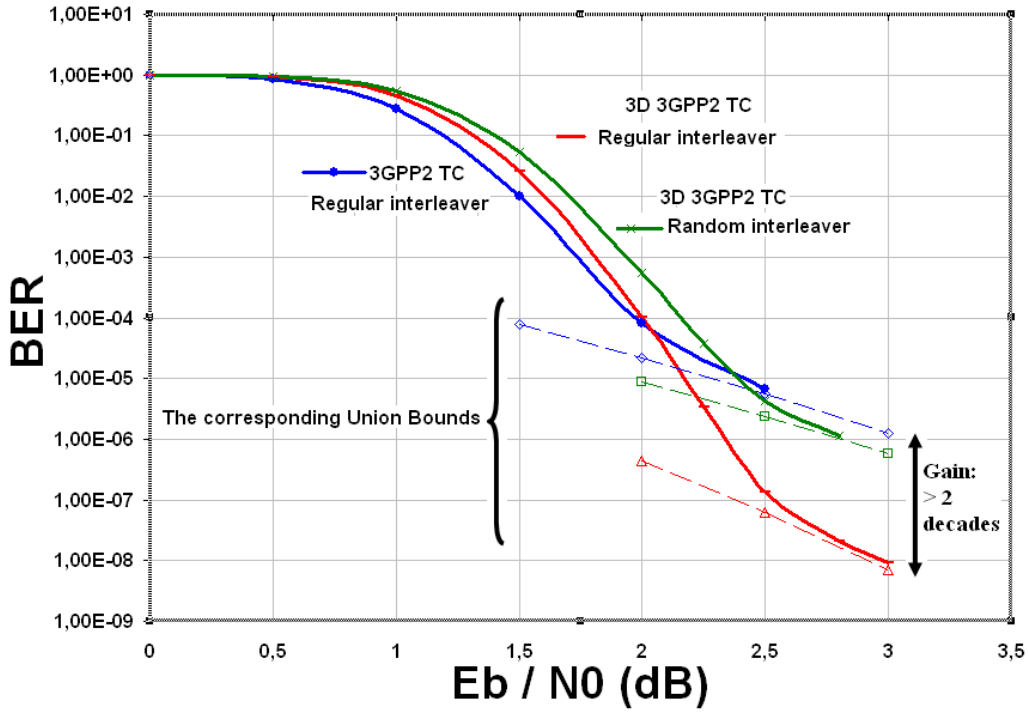

Figure 5. FER performance of the 3GPP2 3D TC with $\lambda = \frac{1}{8}$ for $k = 762$ bits, $R = \frac{1}{2}$ and comparison with the 3GPP2 TC. All simulations use the Max-Log-MAP algorithm with 10 decoding iterations.

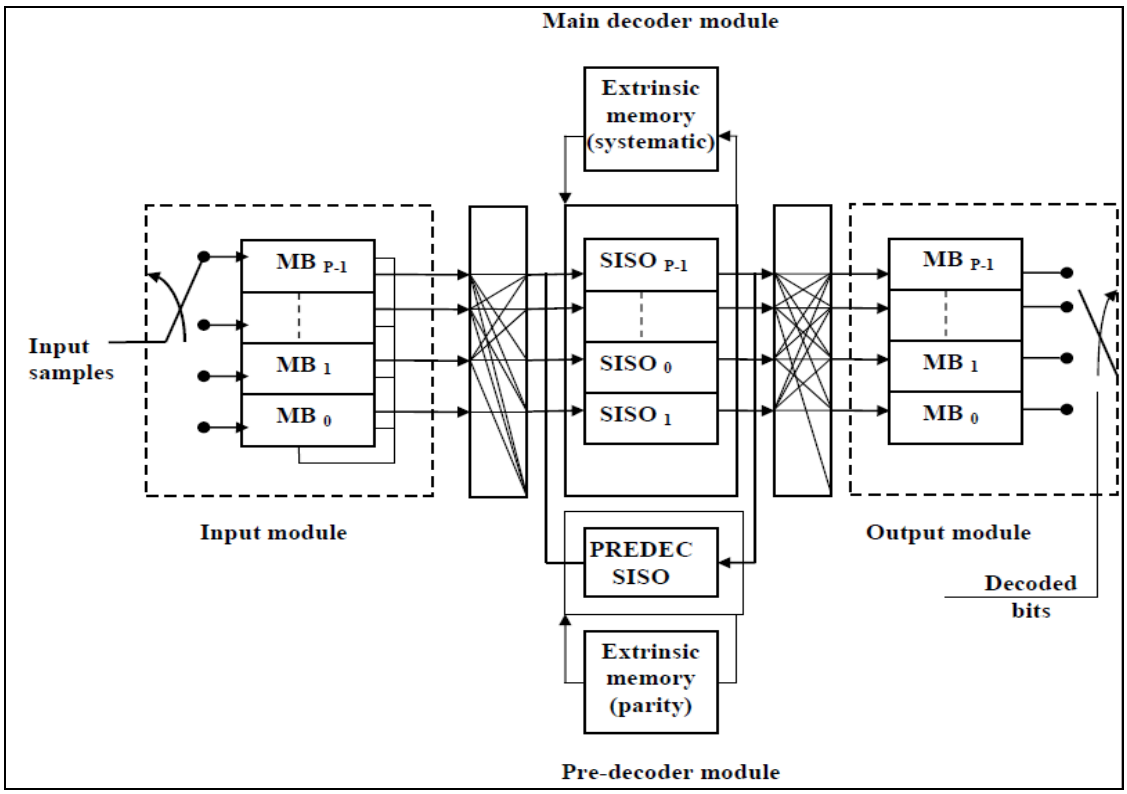

Figure 6. Generic 3D turbo decoder organization for *P* processors.

TABLE I. COMPUTATIONAL COMPLEXITY OF THE MAX-LOG-MAP ALGORITHM
N = TOTAL NUMBER OF ENCODED BITS AT THE DECODER INPUT, AT EACH TIME STEP AND ν = CODE MEMORY.

	Add (or subtract)	Compare-select
Branch metrics (forward or backward recursion)	$2^{n+1} - 2$	
One step of recursion (forward or backward)	$2^{\nu+1}$	2^{ν}
A posteriori LLRs and hard decision	$2^{\nu+1} + 1$	$2^{\nu+1} - 1$
Extrinsic LLRs for information bits	4	
Extrinsic LLRs for redundancy bits (**only for 3D TC**)	2	$2^{\nu+1} - 2$
Total computational requirement per information bit for **classical TC**	$3 \times 2^{\nu+1} + 2^{n+2} + 1$	$2^{\nu+2} - 1$
Total computational requirement per information bit for **3D TC**	$3 \times 2^{\nu+1} + 2^{n+2} + 3$	$3 \times \left(2^{\nu+1} - 1\right)$

TABLE II. SUMMARY OF COMPLEXITY ANALYSIS FOR 3GPP2 AND 3D TURBO DECODERS FOR K = 6138 BITS, R = 1/2 AND Λ = 1/8.

	Overall SISO HW complexity (number of add / compare-select operators		
	$P = 1$	$P = 2$	$P = 4$
3GPP2	112 = C_1	224 = C_2	448 = C_4
3D TC based on 3GPP2	176 = C_1 + 57%	304 = C_2 + 36%	560 = C_4 + 25%

	RAM (equivalent single-port memory in bits) Input quantization: 6 bits		
	$P = 1$	$P = 2$	$P = 4$
3GPP2	101,510 = M_1	106,630 = M_2	126,251 = M_4
3D TC based on 3GPP2	= M_1 + 7.83 %	= M_2 + 7.73 %	= M_4 + 7.54 %

TABLE III. MINIMUM HAMMING DISTANCE VALUES FOR THE 3GPP2 AND THE 3D 3GPP2 TURBO CODES FOR BLOCKS OF K = 762 BITS AND DIFFERENT CODING RATES.

Code Rate	1 / 3	1 / 2	2 / 3	4 / 5
TC	19	11	6	4
3D TC (λ = 1/4)	39	23	9	
3D TC (λ = 1/8)	30	18	8	4

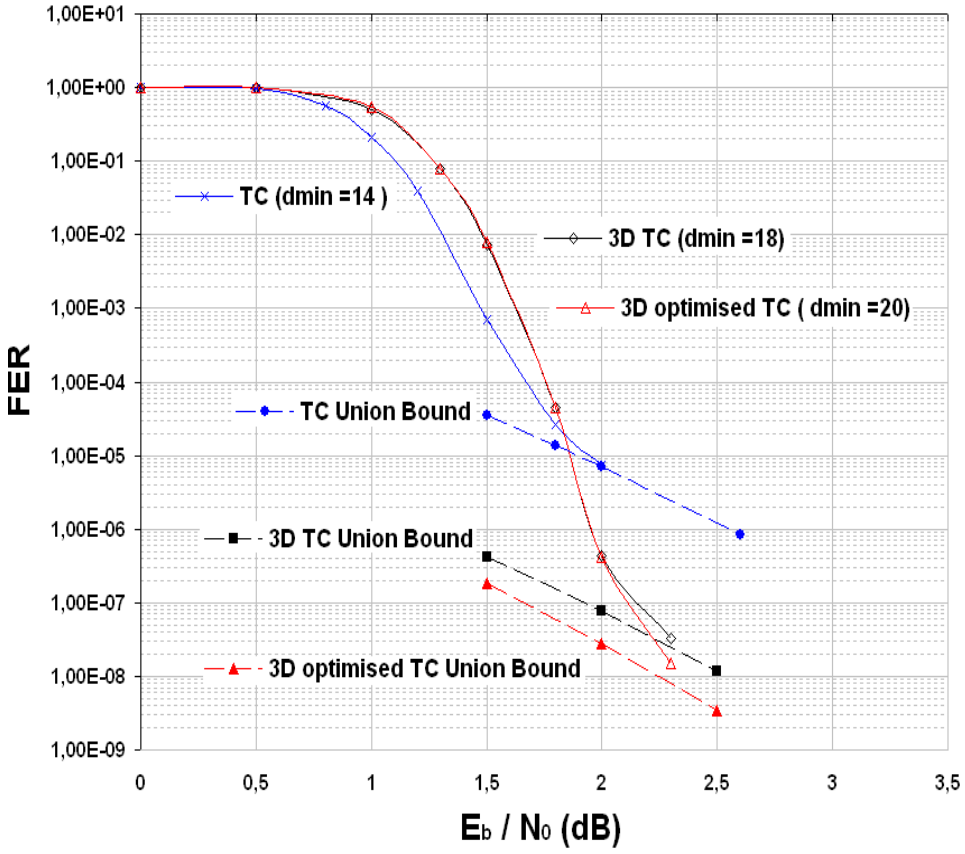

Figure 7. FER performance of the optimized 3D TC with $\lambda = \frac{1}{8}$ for $k = 1530$ bits, $R = \frac{1}{2}$ and comparison with the 3D TC and 3GPP2 standardized TC. All simulations use the Max-Log-MAP algorithm with 10 iterations.

TABLE IV. FIRST TERMS OF THE DISTANCE SPECTRUM FOR A 3D TC WHERE K = 1530 BITS, R = 1/2 AND Λ = 1/8.

Distance	18	20	21	22
Multiplicity	1	1	4	2
Distance	23	24	25	26
Multiplicity	6	1	2	4

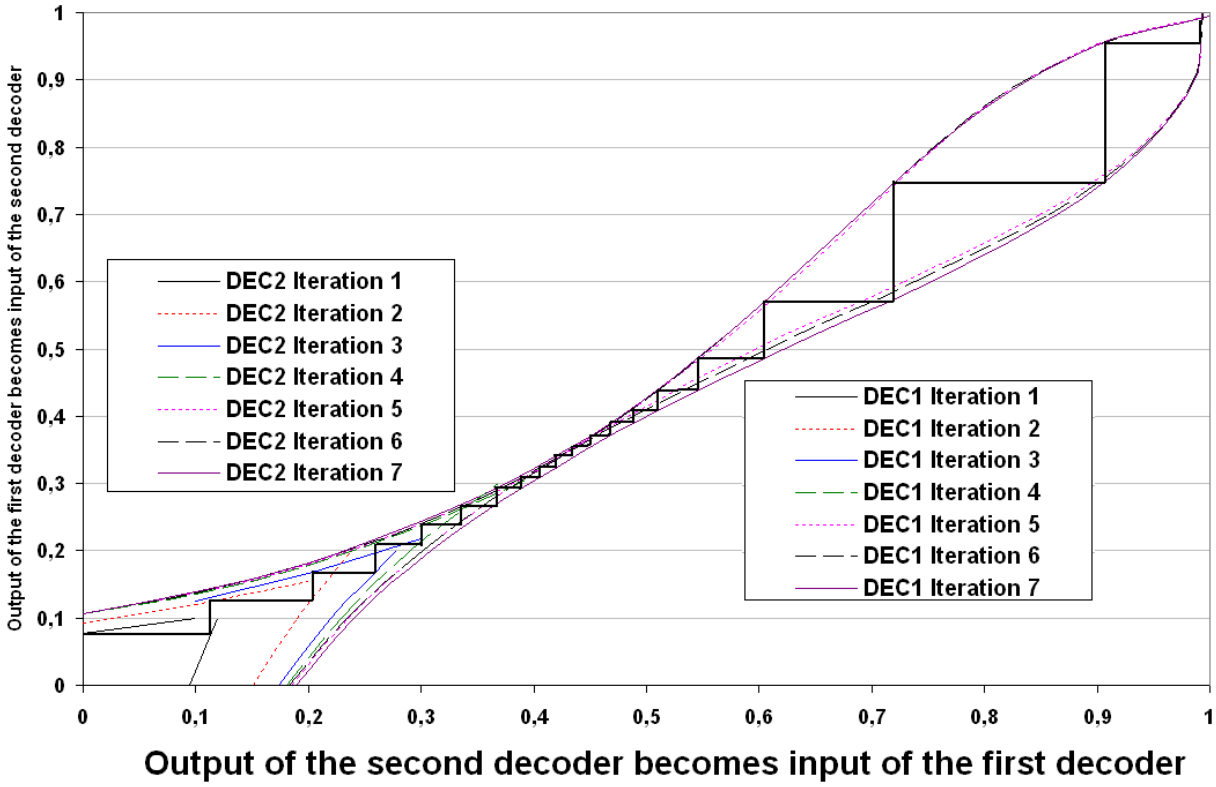

Figure 8. EXIT chart of a time varying 3D TC at code rate $R = \frac{2}{3}, \lambda = \frac{1}{4}$, and $E_b/N_0 = 1.58$ dB.