Adaptive SOVA for 3GPP-LTE Receivers

Authors: Francisco Blanquez-Casado, F. Javier Martin-Vega, David Morales-Jimenez, Gerardo Gomez, and J. Tomás Entrambasaguas

Corresponding author: F. Blanquez-Casado (fbc@ic.uma.es)

Note: This is a pre-print of a paper published at IEEE Communications Letters

Abstract— Turbo codes are extensively used in wireless communication systems and have been adopted as the channel coding scheme for next generation cellular standards. The complexity associated with the decoding process is an important issue. In this paper, we propose an adaptive Soft Output Viterbi Algorithm (adSOVA) based turbo decoder which is proved to reduce complexity without experiencing a degradation in error correction performance. The proposed adSOVA is evaluated in a long term evolution (LTE) downlink scenario and compared with the SOVA and the Max-Log-MAP algorithms. Simulation results show that a complexity reduction of 40% can be achieved with respect to the conventional SOVA with no BLock Error Rate (BLER) degradation.

I INTRODUCTION

CHANNEL coding is a crucial part of current wireless and mobile communication systems. Through a good error correction method, transmission rates close to channel capacity can be achieved [1]. Due to their high error correction capabilities, turbo codes have been chosen as the physical layer coding scheme in new cellular standards like Long Term Evolution (LTE) [2]. Classical turbo decoding is an iterative process in which two Soft Input Soft Output (SISO) decoders exchange the so called extrinsic information in form of Log Likelihood Ratio (LLR). Two well-known SISO algorithms are currently available in the literature: the Maximum A Posteriori (MAP) algorithm [3] and the Soft Output Viterbi Algorithm (SOVA) [4], [5]. The MAP algorithm and its sub-optimal forms, the Log-MAP and theMax-Log-MAP algorithm [6], provide the best performance of the decoding process, but the highest complexity. For that reason the SOVA decoder is frequently used in practical implementations of mobile communications systems, where a high processing speed is required [7].

To reduce the complexity of the SOVA, a truncation of the decoding process [5] is usually performed. This truncation can achieve a high complexity reduction, although it also might cause a degradation in its error correction capability, especially when high coding rates are used [8]. In standards like 3GPPLTE, different Modulation and Coding Schemes (MCSs) are used due to the Link Adaptation (LA) process [9]. As it will be shown, truncating the SOVA algorithm can bring important BLock Error Rate (BLER) degradation, especially when using less robust MCSs.

We propose an adaptive SOVA (adSOVA) turbo decoder for 3GPP-LTE receivers which adaptively selects the minimum truncation length that fulfills the best error correction performance. Thus, this adSOVA can reduce complexity, which is defined in this paper as CPU cycles consumption, while achieving the same BLER than the conventional SOVA.

II. SOVA

The SOVA [4], [5] builds a Trellis tree [10] of the received sequence **y**. Transmitting a systematic bit xt = 0 or xt = 1 causes different transitions from an encoder state s_0 or s_1, respectively, to a state s. Each transition is associated with a parity bit zt and an accumulated metric $Mt(s\_, s)$, which is obtained as

$$M_t(s', s) = M_{t-1}(s', s) + x_t \cdot L_c \cdot y_{t,1} + z_t \cdot L_c \cdot y_{t,2} + x_t \cdot L_t \quad (1)$$

where *Lc* is the channel reliability; *yt,*1 is the received systematic bit; *yt,*2 is the received parity bit; and *Lt* is the *a priori* reliability obtained through the previous SOVA. The transition with the highest accumulated metric produces the decoded bit *ut* and a branch of the *survivor* path, while the opposite transition produces a branch of the *competing* path. The rest of branches of these paths are obtained by linking earlier branches with higher accumulated metrics. For the branch with the highest accumulated metric, a *reliability* of the last decoded bit of the *survivor path* is defined as

$$\Delta_t^0 = \frac{1}{2} \left| M_t\left(s'_0, s\right) - M_t\left(s'_1, s\right) \right|. \quad (2)$$

This reliability and its associated bit of the *survivor* path are stored. Then an *updating* process is done as follows. Branches of the *survivor* and *competing* paths are traced back and, when the binary decision is different in both paths at the memory level *MEM*, the reliability is updated as

$$\left|\Delta_t^{MEM} = \min_{k=0,\ldots MEM} \left\{\Delta_t^k\right\}. \quad (3)$$

Finally, when the updating process is done for all entry bits, soft output values are obtained as

$$L(u_t) = \Delta_t \cdot u_t. \quad (4)$$

III. MAX-LOG-MAP ALGORITHM

The MAP algorithm [3] estimates the *a posteriori* probability of each decoded bit *ut* from the received sequence **y** in LLR form

$$L(u_t|\mathbf{y}) = \log\left(\frac{P(u_t = +1|\mathbf{y})}{P(u_t = -1|\mathbf{y})}\right). \quad (5)$$

To compute this *a posteriori* probability, forward (*At*(*s*)) and backward (*Bt*−1(*s*_)) metrics are defined for each encoder state *s* in a recursive manner

$$A_t(s) = \max_{s'}^{*} \left[A_{t-1}(s') + M_t(s', s)\right] \quad (6)$$

$$B_{t-1}(s') = \max_{s}^{*} \left[B_t(s) + M_t(s', s)\right] \quad (7)$$

being *s*_ in (6) the states connected to *s* in a forward direction and *s* in (7) the states connected to *s*_ in a backward direction. *Mt* is computed as in (1). The max∗ operator is the Jacobian logarithm, which is

approximated as max(*a, b*)[6] for the Max-Log-MAP algorithm. Finally, after computing the forward and backward metric, the *a posteriori* probabilities of decoded bits are calculated as

$$L(u_t|\mathbf{y}) = \max^*_{R_0}\left([A_{t-1}(s') + M_t(s', s) + B_t(s)\right) - \max^*_{R_0}\left([A_{t-1}(s') + M_t(s', s) + B_t(s)\right) \quad (8)$$

where *R*0 and *R*1 are the possible transitions from states *s*_ tostates *s* when a bit 0 or a bit 1 is transmitted, respectively.

IV. PROPOSED ADSOVA

To reduce complexity, a limitation on the number of trace back steps of the updating process is carried out in the SOVA. Typically, a value of 2*K* or 3*K* (where *K* is the constraint length of the turbo encoder) is recommended [5]. However, this limitation may degrade the performance of the decoding process depending on the scenario. For instance, [8] evaluates the need of a deeper updating window for high coding rates to avoid Bit Error Rate (BER) degradation. Thus, a short updating window could lead to a degradation of the error correction performance for the highest coding rates, while a deep updating window might unnecessarily increase the complexity of the decoding process for the most robust coding rates.

Moreover, the channel model has a significant impact on the SOVA performance too [11]. Hence, the *a priori* selection of a value for the depth of the updating process could reduce the efficiency of the turbo decoder.

We propose an adSOVA which stops the updating process as soon as no further trace back steps are needed. Hence, the proposed algorithm achieves the same error correction performance as the SOVA while reduces complexity by avoiding unnecessary processing. To illustrate this, Fig. 1 shows an example of the *competing* and the *survivor* paths at the instant $t-1$ and t for a four-states turbo code. At memory level 1, the *survivor* path at t matchs the *survivor* path at $t-1$. The same coincidence happens with the *competing* path at memory level 3. Thus, if any of the *competing* or the *survivor* paths reaches a node at t which had been reached at $t-1$, the rest of the trace back steps coincide.

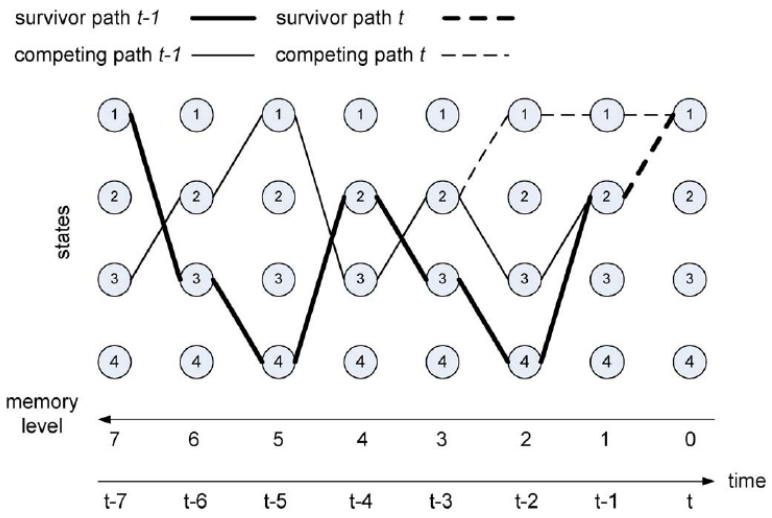

Fig. 1. Trellis tree example.

Moreover, if the reliability value obtained for the memory level 0 at the instant $t(\Delta 0t)$ is the lowest value at this time, this value will be used during the *updating* process in the rest of memory levels when binary decisions are different. Otherwise, the reliability value used at each memory level will be the minimum value found until this level, and it will match the minimum value found until this level at $t-1$. Thus, if both the *survivor* and the *competing* paths coincide at the instant t and $t-1$, it is not necessary to complete the updating process from these nodes. Based on this fact, an *adaptive updating* algorithm is proposed for the adSOVA.

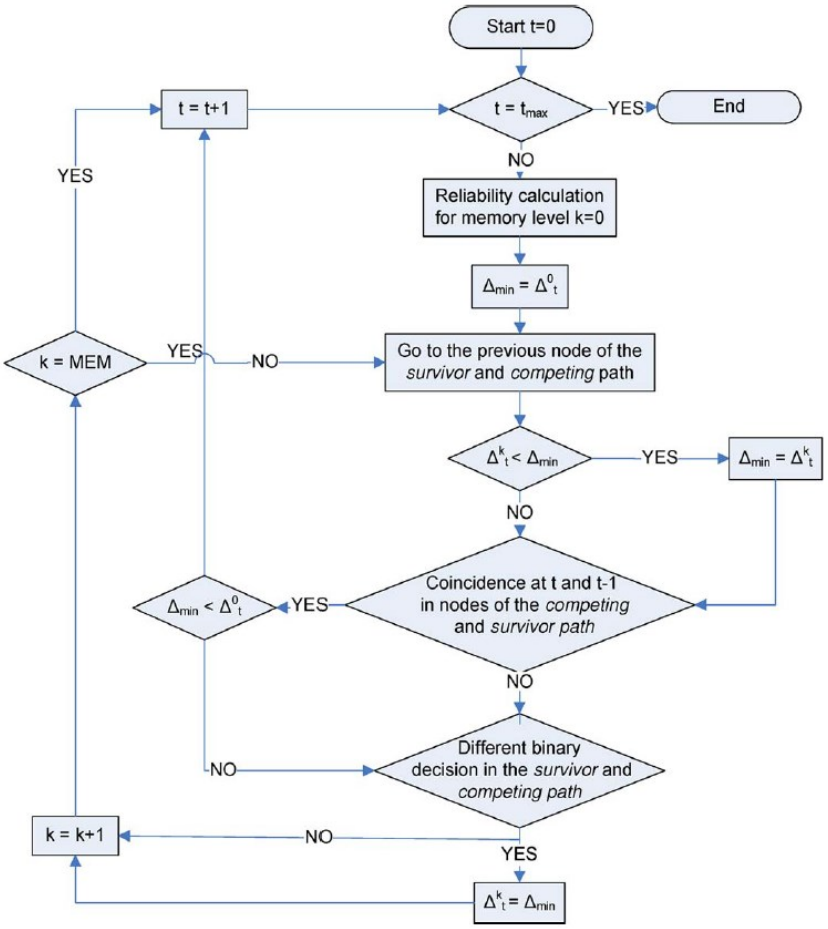

Fig. 2. Flow diagram of the proposed adSOVA.

To define the *adaptive updating* process at time t, consider the nodes of the *survivor* and *competing* path of the instant $t-1$, as well as the minimum reliability value found during the trace back process. The proposed algorithm is given by the following steps (see Fig. 2):

1) Calculate the reliability value for thememory level $k = 0$. This value is taken as the minimum reliability value ($\Delta min = \Delta 0t$). Then the trace back process through the survivor and competing path begins.

2) Check if the reliability value stored for each memory level is the minimum value found. In that case the minimum reliability value is updated ($\Delta min = \Delta kt$).

3) Check if both the *survivor* and *competing* path for each node at instant t belong to the *survivor* and the *competing* path of the instant $t-1$.

4) If step 3 is satisfied and reliability value of the memory level $k = 0$ is not the minimum reliability value, the *updating* process can be finished since no new updating in the rest of the trace back process will be done. Otherwise, go back to step 2 until the trace back process is terminated, or stopping conditions are satisfied.

The adSOVA is expected to reduce on average the number of trace back steps compared to the SOVA. Regarding the number of operations, three extra comparisons are done at each trace back step of the updating process.

V. PERFORMANCE OF ADSOVA IN 3GPP-LTE SYSTEMS

In this section the performance of the proposed adSOVA is evaluated and compared with the SOVA [5] and the Max-Log-MAP algorithm with sliding-window [12]. These algorithms have been included in a complete LTE downlink simulator [13], which includes a 3GPP-LTE turbo encoder [14]. Main simulation parameters are listed in Table I.

TABLE I
SIMULATION PARAMETERS

Parameter	Value
Carrier frequency	2 GHz
Sampling frequency	30.72 MHz
System bandwidth	20 MHz
Channel model	Extended Pedestrian A [15]
Mobile terminal speed	4 km/h
Source model	Full buffer
Uncoded block length	5000 bits
CQIs	4, 9, 14
Turbo decoding iterations	5

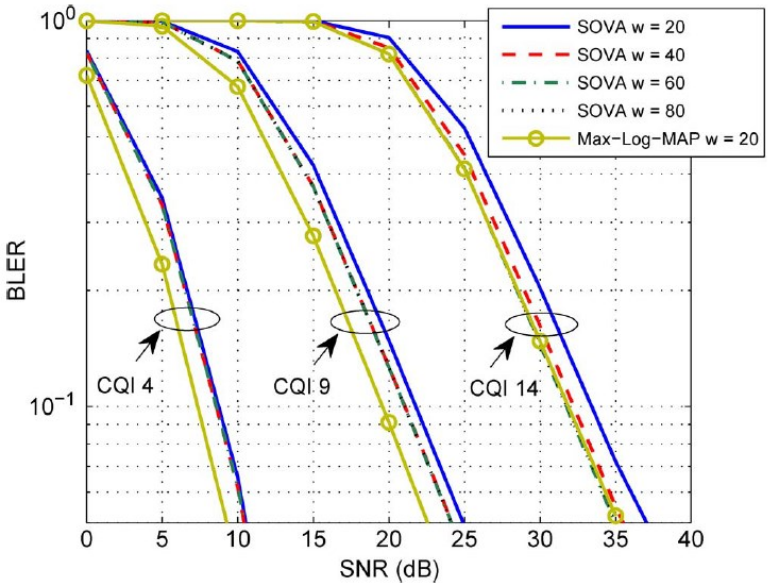

Fig. 3. BLER results for different coding rates.

In 3GPP-LTE standard, BLER is a more useful statistic than BER as it is used to perform the LA [9]. In Fig. 3 BLER results are shown for MCSs corresponding to the Channel Quality Indicators (CQIs) 4, 9 and

14 [9], for a SOVA decoder with different updating window depths w, and for the Max-Log-MAP decoder using a sliding-window1w of depth 20. Note that, as it was justified in Section IV, the error correction performance is identical for both the SOVA and the adSOVA for the same value of w. Regarding the SOVA, for the most robust MCS there are no major differences when the depth of the updating window increases. As the robustness of the MCS decreases, the use of a deeper updating window improves significantly the BLER performance. In particular, for a BLER of 0.1, a gain of 1.6 dB is achieved when using a depth of w = 80 steps instead of w = 20 steps. However, Fig. 3 shows that an updating window deeper than 60 hardly improves the BLER performance whatever the MCS. Concerning the Max-Log-MAP algorithm, it always improves the SOVA except for the least robust MCS, where results are very similar when a deep updating window is used for the SOVA, which means that a deeper sliding-window is necessary. Hence, these results reveal the impact of the depth of the updating window in the BLER performance of a SOVA based turbo decoder.

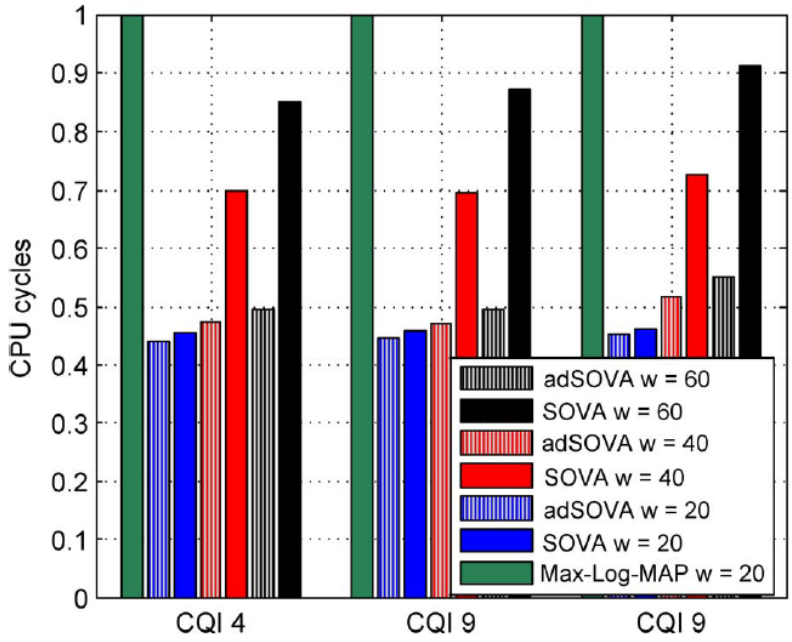

Fig. 4. Average CPU cycles consumption for different MCSs.

Regarding complexity, Fig. 4 presents an average CPU cycles consumption comparison between the adSOVA and the SOVA, for different updating window depths, and the Max-Log-MAP algorithm. For each MCS and each algorithm, it has been considered the average SNR for which a BLER of 0.1 is achieved (see Fig. 3), as this is the operating point of LTE [9]. These results have been normalized to the maximum value for each MCS, which corresponds to the Max-Log-MAP algorithm.

Fig. 4 shows that, for the adSOVA, the number of CPU cycles consumed is always lower than for the SOVA, for the same value of w. For instance, for the MCS corresponding to the CQI 14 and w = 60, a reduction of 40% is achieved. Furthermore, for the adSOVA, results when using a deep window are close to results of the SOVA when using the shortest window, i.e., w = 20. Thus, the proposed algorithm allows to use a higher value of w to improve BLER (especially for the least robust MCSs) with a complexity increase that is considerably lower than the experienced by the SOVA when the same value of w is used.

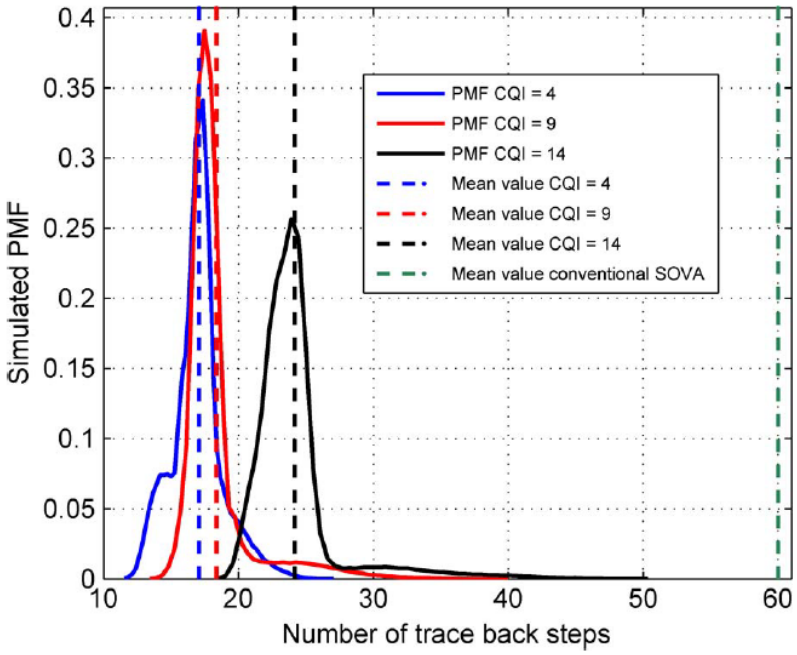

Fig. 5. Simulated PMF of trace back steps for adSOVA with $w = 60$.

Fig. 5 shows the simulated probability mass function (PMF) of the number of trace back steps for the adSOVA with w = 60, for the same MCSs and mean SNR values than used in Fig. 4. Also, the mean values of these numbers of steps are shown. These results show that despite the variability of the number of steps, results are almost always close to the mean value, which in all cases is far from the maximum value of 60. Then, the complexity reduction of the adSOVA will not have high variability for each decoded block. Thus, although the turbo decoder must be worst-case designed, i.e., to assume that the number of trace back steps of the adSOVA is always the maximum value w, as this case will hardly ever occur, a complexity reduction will be obtained in almost all decoded blocks. This reduction can be used, for instance, to increase the number of iterations of the turbo decoder until achieving the worst-case complexity, thus improving the error correction performance; or to stop earlier the turbo decoding process to save power [16].

VI. CONCLUSION

In this paper, it has been shown that a truncation of the updating process of the SOVA can cause an important degradation of the BLER performance, mainly for high coding rates. Thus, we proposed a modifed SOVA which adaptatively adjusts the depth of the updating window to the minimum value that achieves the best error correction. Simulations results show that for typical operating points of LTE the adSOVA reduces complexity compared with the conventional SOVA, but without BLER degradation. Then, a deeper updating window can be used to improve BLER without a signficant increase of complexity.

Furthermore, a comparison with Max-Log-MAP algorithm has been carried out, showing the adSOVA an important complexity reduction.

REFERENCES

[1] C. Berrou, A. Glavieux, and P. Thitimajshima, “Near shannon limit error-correcting coding and decoding: Turbo-codes,” in *Proc. IEEE ICC*, May 1993, vol. 2, pp. 1064–1070.

[2] G. Gómez, D. Morales-Jiménez, F. J. López-Martínez, J. J. Sánchez, and J. T. Entrambasaguas, *Long Term Evolution*, B. Furht and S. A. Ahson, Eds. New York, NY, USA: Auerbach, Apr. 2009.

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

[4] J. Hagenauer and P. Hoeher, "A Viterbi algorithm with soft – decision outputs and its applications," in *Proc. IEEE GLOBECOM*, Nov. 1989, vol. 3, pp. 1680–1686.

[5] C. Berrou, P. Adde, E. Angui, and S. Faudeil, "A low complexity softoutput Viterbi decoder architecture," in *Proc. IEEE ICC*, May 1993, vol. 2, pp. 737–740.

[6] P. Robertson, E. Villebrun, and P. Hoeher, "A comparison of optimal and sub-optimal map decoding algorithms operating in the log domain," in *Proc. Int. Conf. Communications*, Jun. 1995, vol. 2, pp. 1009–1013.

[7] L.-H. Ang, W.-G. Lim, and M. Kamuf, "Modification of SOVA-based algorithms for efficient hardware implementation," in *Proc. IEEE VTCSpring*, May 2010, pp. 1–5.

[8] O. J. Joeressen, M. Vaupel, and H. Meyr, "Soft-output Viterbi decoding: VLSI implementation issues," in *Proc. IEEE VTC*, 1993, pp. 941–944.

[9] *Evolved Universal Terrestrial Radio Access (E-UTRA); Physical Layer Procedures (Release 9)*, 3GPP TS 36.213, Oct. 2010.

[10] G. D. Forney, "The Viterbi algorithm," *Proc. IEEE*, vol. 61, no. 3, pp. 268–277, Mar. 1973.

[11] J. Ortin, P. Garcia, F. Gutierrez, and A. Valdovinos, "Performance analysis of turbo decoding algorithms in wireless OFDM systems," *IEEE Trans. Consum. Electron.*, vol. 55, no. 3, pp. 1149–1154, Aug. 2009.

[12] C.-M. Wu, M.-D. Shieh, C.-H. Wu, Y.-T. Hwang, and J.-H. Chen, "VLSI architectural design tradeoffs for sliding-window log-map decoders," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 13, no. 4, pp. 439–447, Apr. 2005.

[13] G. Gómez, D. Morales-Jiménez, J. J. Sánchez-Sánchez, and J. T. Entrambasaguas, "A next generation wireless simulator based on MIMO-OFDM: LTE case study," *EURASIP J. Wireless Commun. Netw.*, vol. 2010, pp. 15:1–15:12, Apr. 2010.

[14] *Evolved Universal Terrestrial Radio Access (E-UTRA); Multiplexing and channel coding (Release 9)*, 3GPP TS 36.212, Sep. 2011.

[15] *Evolved Universal Terrestrial Radio Access (E-UTRA); User Equipment (UE) Radio Transmission and Reception; (Release 8)*, 3GPP TS 36.803, Apr. 2008.

[16] O.-H. Leung, C.-Y. Tsui, and R.-K. Cheng, "Reducing power consumption of turbo-code decoder using adaptive iteration with variable supply voltage," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 9, no. 1, pp. 34–41, Feb. 2001.