MAP Decoding of Interleaved Multilevel Codes

Ivan Stanojević, Tomaš Vušurović, Branislava Grbić, Vojin Šenk

Abstract— **A way of decoding multilevel codes using MAP (Maximum A Posteriori) algorithm is presented. A bit error rate (BER) performance comparison between the MAP decoder and the standard multistage Viterbi decoder is made.**

Keywords— **Multilevel codes, MAP algorithm, Interleavers, Iterative decoding.**

I. Introduction

In the last several years, multilevel codes have attracted significant attention due to the flexibility of construction and low complexity of decoding algorithms. Coding systems consisting of adjustable rate component convolutional encoders with appropriate modulation symbol mappings have found practical application in various band limited channel transmission systems.

The standard way of decoding multilevel codes, denoted multistage decoding (widely described in the literature, e.g. [1]), employs standard Viterbi decoding for each level of the system and hard information interchange (sequences of 0's and 1's) between different levels. The structure of multilevel coding systems can be observed as concatenation of component codes through joint mapping into output modulation symbols. When the outputs of component convolutional encoders are interleaved, this is similar to parallel concatenated codes (PCCs or turbo-codes) [2], where the input information sequence is the same for all the component codes, as well as to serial concatenated codes (SCCs) [3], where the output sequence of the "outer" encoder is the input sequence of the "inner" encoder.

The analogy with PCCs and SCCs can be extended to the decoding algorithm. The MAP algorithm, as described in [5], can be applied to each component code, and soft infomation can be interchanged between levels. As with PCCs and SCCs, multiple decoding iterations can be used to improve BER performance.

II. System Description

The structure of the analyzed multilevel coding system is presented in Fig. 1. A block of k user information bits, $u_0, \ldots, u_{k-1}$, enters the system input, where k is determined as $k = k_0 + \cdots + k_{r-1}$, and r is the number of encoder levels. These information bits are partitioned into r blocks $x_{s,0}, \ldots, x_{s,k_s-1}$, $s \in \{0, \ldots, r-1\}$. Each of these blocks is fed into a particular convolutional encoder C_s, which outputs a block of encoded bits, $v_{s,0}, \ldots, v_{s,l-1}$. All the encoder output bit blocks have the same length l, which means that the code rates applied are $R_s = k_s/l$. The overall bit code rate of the system is

$$R = \frac{k}{rl} = \frac{\sum_{s=0}^{r-1} k_s}{rl} = \frac{1}{r}\sum_{s=0}^{r-1} R_s. \qquad (1)$$

Each block of encoded bits, $v_{s,0}, \ldots, v_{s,l-1}$, is fed into the corresponding interleaver, I_s, which outputs a block of bits $y_{s,0}, \ldots, y_{s,l-1}$, according to the relation

$$y_{s,j} = v_{s,I_s(j)}, \qquad (2)$$

where $I_s(\cdot)$ is the permutation function realized by the interleaver.

r-tuples of bits, $(y_{0,j}, \ldots, y_{r-1,j})$, $j \in \{0, \ldots, l-1\}$, enter the QAM mapper, and complex modulation symbols, $z_j = M(y_{0,j}, \ldots, y_{r-1,j})$, are generated.

Modulation symbols are transmitted through the channel with additive white Gaussian noise n_j, so complex values at the decoder input, w_j, are given as

$$w_j = z_j + n_j. \qquad (3)$$

III. Decoding Process Description

In the following text, random variables of all quantities in the system will be denoted by upper case letters and their realizations by lower case letters. Continuous and discrete random variables will be treated in the same way, i.e. $P[D = d]$ will denote the probability that a discrete random variable D takes the value d, whereas $P[C = c]$ will denote the value of the probability density function of some continuous random variable C in the point c. In accordance with this, notation of type $P[D = d, C = c]$ will be used for mixed discrete-continuous probability functions.

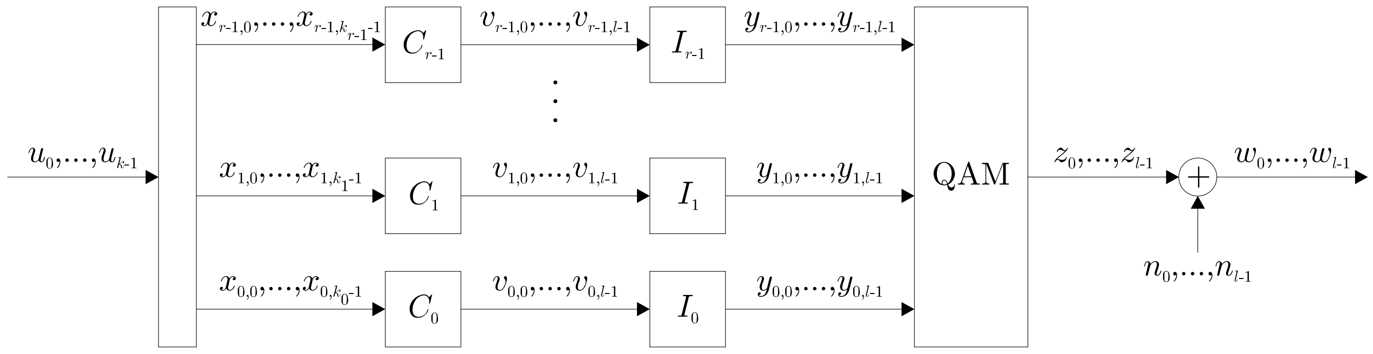
Before each decoding pass (using the trellis of a single encoder C_s), probabilities $P[Y_{s,j} = b | W_j = w_j] = P[Y_{s,j} = b, W_j = w_j]/P[W_j = w_j]$ are calculated. We have

$$\begin{aligned} P[Y_{s,j} = b, W_j = w_j] &= P[Y_{s,j} = b, Z_j + N_j = w_j] = \\ &= \sum_{z_j} P[N_j = w_j - z_j, Y_{s,j} = b, Z_j = z_j] = \\ &= \sum_{z_j} P[N_j = w_j - z_j] P[Y_{s,j} = b, Z_j = z_j]. \end{aligned} \qquad (4)$$

The last equality is a consequence of the independence of noise from the transmitted sequence. In the previous sums, z_j takes values from the set of all alphabet symbols. The noise is complex white Gaussian with zero mean and component variance σ, so that

$$P[N_j = w_j - z_j] = \frac{1}{2\pi\sigma^2} e^{-\frac{|w_j - z_j|^2}{2\sigma^2}}. \qquad (5)$$

The authors are with the Communications Division of The Institute of Power Engineering, Electronics and Communications, Faculty of Engineering, University of Novi Sad, Yugoslavia. E-mail: cet_ivan@uns.ns.ac.yu



$x_{r-1,0},\dots,x_{r-1,k_{r-1}-1}$
C_{r-1}
$v_{r-1,0},\dots,v_{r-1,l-1}$
I_{r-1}
$y_{r-1,0},\dots,y_{r-1,l-1}$
$u_0,\dots,u_{k-1}$
$x_{1,0},\dots,x_{1,k_1-1}$
C_1
$v_{1,0},\dots,v_{1,l-1}$
I_1
$y_{1,0},\dots,y_{1,l-1}$
QAM
$z_0,\dots,z_{l-1}$
$w_0,\dots,w_{l-1}$
$x_{0,0},\dots,x_{0,k_0-1}$
C_0
$v_{0,0},\dots,v_{0,l-1}$
I_0
$y_{0,0},\dots,y_{0,l-1}$
$n_0,\dots,n_{l-1}$

Fig. 1. Multilevel encoder and additive white Gaussian noise channel

On the other hand,

$$\begin{aligned} &P[Y_{s,j}=b, Z_j=z_j]= \\ &\quad = P[Y_{s,j}=b, \\ &\qquad M(Y_{0,j},\dots,Y_{r-1,j}) = M(y_{0,j},\dots,y_{r-1,j})] = \\ &\quad = P[Y_{s,j}=b, Y_{0,j}=y_{0,j},\dots,Y_{r-1,j}=y_{r-1,j}] = \\ &\quad = P[Y_{s,j}=b, Y_{s,j}=y_{s,j}] \prod_{\substack{t=0\\t\neq s}}^{r-1} P[Y_{t,j}=y_{t,j}]. \end{aligned} \tag{6}$$

The equality before the last follows from the bijectiveness of the mapper function $M(\cdot)$, and the last is valid due to the independence of encoded bits from different blocks. It is obvious that

$$P[Y_{s,j}=b, Y_{s,j}=y_{s,j}] = \begin{cases} P[Y_{s,j}=b], & y_{s,j}=b \\ 0, & y_{s,j}\neq b \end{cases}, \tag{7}$$

so due to the fact that z_j is uniquely determined by the values $y_{0,j},\dots,y_{r-1,j}$, we finally obtain (8) and $P[Y_{s,j}=b|W_j=w_j]$ can be computed, as well.

The probabilities $P[Y_{s,j}=y_{s,j}]$ are not known at the receiver side and they are one of the results the decoder should give. In the iterative decoding process, at the beginning it is assumed that $P[Y_{s,j}=1]=P[Y_{s,j}=0]=\frac{1}{2}$ and these values are being memorized and updated in each iteration.

Introducing ordinary, conditional and joint log-likelihood ratios (LLRs) of a binary random variable B [6], defined as

$$\Lambda[B] \overset{\triangle}{=} \ln\frac{P[B=1]}{P[B=0]}, \tag{9}$$

$$\Lambda[B|A] \overset{\triangle}{=} \ln\frac{P[B=1|A]}{P[B=0|A]}, \tag{10}$$

$$\Lambda[B,A] \overset{\triangle}{=} \ln\frac{P[B=1,A]}{P[B=0,A]}, \tag{11}$$

where A is some event, and taking into account that $\Lambda[B|A]=\Lambda[B,A]$ and $\ln(P[B=b]/P[B=0])=\Lambda[B]b$, we obtain (12).

If for a set of real numbers $\{p_1,\dots,p_m\}$, we define [6]

$$\max^*\{p_1,\dots,p_m\} \overset{\triangle}{=} \ln\sum_{t=1}^{m} e^{p_t}, \tag{13}$$

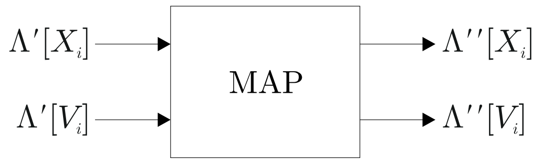


Fig. 2. MAP algorithm module

(12) can be rewritten as (14).

The MAP algorithm module, which is the basic block used for iterative decoding, is shown in Fig. 2. The inputs of the MAP algorithm are a priori probabilities of information and encoded bits, given by the corresponding LLRs, $\Lambda'[X_i]$ and $\Lambda'[V_i]$, respectively. Under the limitations imposed by the encoder trellis, a posteriori probabilities of information and encoded bits are given at the output, by the corresponding LLRs, $\Lambda''[X_i]$ and $\Lambda''[V_i]$, respectively. Detailed MAP algorithm descriptions are given in [4] and [5].

The process of decoding consists of multiple iterations. Before the first iteration, all encoded bit LLRs, $\Lambda[Y_{s,j}]$, are set to 0. Depending on the order of LLR updating, there are two types of iterations: serial and parallel.

One decoding iteration with serial LLR update is as follows:

1. $s=0$.
2. Using the sequence of received channel symbols, w_j, and current encoded bit LLRs, $\Lambda[Y_{s,j}]$, calculate
   $\Lambda[Y_{s,j}|W_j=w_j]$,
   for $j\in\{0,\dots,l-1\}$.
3. Initialize a priori LLRs by deinterleaving
   $\Lambda'[V_i]=\Lambda[Y_{s,I_s^{-1}(i)}|W_{I_s^{-1}(i)}=w_{I_s^{-1}(i)}]$,
   for $i\in\{0,\dots,l-1\}$.
4. Perform a MAP algorithm pass for the trellis of C_s, assuming that $\Lambda'[X_i]=0$, for $i\in\{0,\dots,k_s-1\}$. The outputs of the algorithm are the values $\Lambda''[X_i]$ and $\Lambda''[V_i]$.
5. If this is the last iteration, estimate the part of the user information sequence which is the input of C_s,

$$P[Y_{s,j}=b,W_j=w_j]=\frac{P[Y_{s,j}=b]}{2\pi\sigma^2}\sum_{y_{0,j}}\cdots\sum_{y_{s-1,j}}\sum_{y_{s+1,j}}\cdots\sum_{y_{r-1,j}}e^{-\frac{|w_j-M(y_{0,j},\ldots,y_{s-1,j},b,y_{s+1,j},\ldots,y_{r-1,j})|^2}{2\sigma^2}}\prod_{\substack{t=0\\t\neq s}}^{r-1}P[Y_{t,j}=y_{t,j}] \quad (8)$$

$$\begin{aligned}\Lambda[Y_{s,j}|W_j=w_j]=\Lambda[Y_{s,j}]&+\ln\left(\sum_{y_{0,j}}\cdots\sum_{y_{s-1,j}}\sum_{y_{s+1,j}}\cdots\sum_{y_{r-1,j}}e^{-\frac{|w_j-M(y_{0,j},\ldots,y_{s-1,j},1,y_{s+1,j},\ldots,y_{r-1,j})|^2}{2\sigma^2}+\sum_{\substack{t=0\\t\neq s}}^{r-1}\Lambda[Y_{t,j}]y_{t,j}}\right)\\&-\ln\left(\sum_{y_{0,j}}\cdots\sum_{y_{s-1,j}}\sum_{y_{s+1,j}}\cdots\sum_{y_{r-1,j}}e^{-\frac{|w_j-M(y_{0,j},\ldots,y_{s-1,j},0,y_{s+1,j},\ldots,y_{r-1,j})|^2}{2\sigma^2}+\sum_{\substack{t=0\\t\neq s}}^{r-1}\Lambda[Y_{t,j}]y_{t,j}}\right)\end{aligned} \quad (12)$$

$$\begin{aligned}&\Lambda[Y_{s,j}|W_j=w_j]=\\&=\Lambda[Y_{s,j}]+\max^*_{(y_{0,j},\ldots,y_{s-1,j},y_{s+1,j},\ldots,y_{r-1,j})}\left\{-\frac{|w_j-M(y_{0,j},\ldots,y_{s-1,j},1,y_{s+1,j},\ldots,y_{r-1,j})|^2}{2\sigma^2}+\sum_{\substack{t=0\\t\neq s}}^{r-1}\Lambda[Y_{t,j}]y_{t,j}\right\}\\&\quad-\max^*_{(y_{0,j},\ldots,y_{s-1,j},y_{s+1,j},\ldots,y_{r-1,j})}\left\{-\frac{|w_j-M(y_{0,j},\ldots,y_{s-1,j},0,y_{s+1,j},\ldots,y_{r-1,j})|^2}{2\sigma^2}+\sum_{\substack{t=0\\t\neq s}}^{r-1}\Lambda[Y_{t,j}]y_{t,j}\right\}\end{aligned} \quad (14)$$

$$\hat{X}_{s,i}=\begin{cases}1, & \Lambda''[X_i]\geq 0\\0, & \Lambda''[X_i]<0\end{cases},$$

for $i\in\{0,\ldots,k_s-1\}$.

6. Update encoded bit LLRs by interleaving

   $\Lambda[Y_{s,j}]=\Lambda''[V_{I_s(j)}]$,

   for $j\in\{0,\ldots,l-1\}$.
7. $s=s+1$.
   If $s<r$ go to step 2.

One decoding iteration with parallel LLR update differs in the last three steps:

6. Remember the values $\Lambda''[V_i]$ in an auxiliary array

   $\lambda_{s,i}=\Lambda''[V_i]$,

   for $i\in\{0,\ldots,l-1\}$.
7. $s=s+1$.
   If $s<r$ go to step 2.
8. Update encoded bit LLRs by interleaving, using the auxiliary arrays

   $\Lambda[Y_{s,j}]=\lambda_{s,I_s(j)}$,

   for $s\in\{0,\ldots,r-1\}$ and $j\in\{0,\ldots,l-1\}$.

After one iteration, updated LLRs are used for subsequent iterations. In serial update iterations, they are immediately used for subsequent levels, as well. The number of iterations is determined as a compromise between BER improvement and the number of operations.

A comparison with multistage Viterbi decoding can be made here. Multistage Viterbi decoding can be viewed as a modified version of a single serial LLR update iteration. The idea is that after decoding the s-th system level, decoded bits $y_{s,j}$ are fixed and used as such in subsequent levels, each time reducing the set of possible modulation symbols. This is equivalent to setting

$$\Lambda[Y_{s,j}]=\begin{cases}\infty, & \Lambda''[V_{I_s(j)}]\geq 0\\-\infty, & \Lambda''[V_{I_s(j)}]<0\end{cases} \quad (15)$$

in the 6. step of the serial LLR update iteration.

TABLE I
COMPONENT CODE RATES AND PUNCTURING PATTERNS

System level	Code rate	Puncturing pattern
2	1/3	1110
1	2/3	1100, 1000
0	4/5	1100, 1000, 1000, 1000

IV. Simulation Results

The system for which simulations are done consists of $r=3$ levels, and its output alphabet is 64-QAM.

Component convolutional codes are obtained by puncturing the same mother code of rate 1/4 and memory $\mathcal{M}=6$, given in octal notation as $(133,171,145,133)$. Their rates and puncturing patterns are given in Table I. The overall bit code rate of the system is $R=0.6$ and the length of the input sequence is $k=21998$ bits.

Each output modulation symbol depends on two succesive triads of interleaved encoded bits,

$$z_j=M(y_{0,2j},y_{1,2j},y_{2,2j},y_{0,2j+1},y_{1,2j+1},y_{2,2j+1}), \quad (16)$$

which is somewhat different from the system described in the introduction. The conditional LLR derivations are valid for this system, as well, since the assumption of independent encoded bits used in (6) holds due to the interleaving. Modulation symbols and corresponding encoded

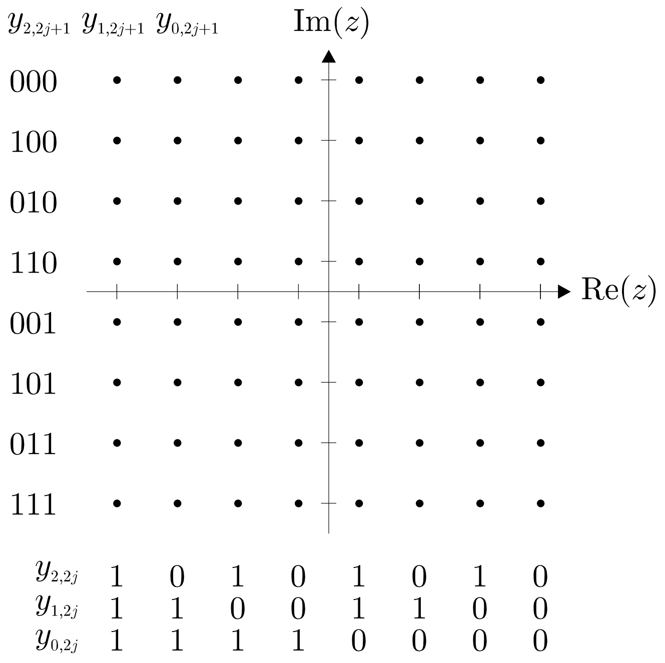

$y_{2,2j+1}$ $y_{1,2j+1}$ $y_{0,2j+1}$
Im(z)
000
100
010
110
Re(z)
001
101
011
111
$y_{2,2j}$ 1 0 1 0 1 0 1 0
$y_{1,2j}$ 1 1 0 0 1 1 0 0
$y_{0,2j}$ 1 1 1 1 0 0 0 0

Fig. 3. Modulation symbols and corresponding encoded bits

bits are shown in Fig. 3. In total, $l/2 = 6118$ modulation symbols are generated and the symbol code rate is $R^* = 3.6$ bit/symbol, since one modulation symbol is addressed by 6 encoded bits.

The interleavers I_0, I_1 and I_2 of appropriate lengths are constructed as pseudo-random.

The relationship between the noise component variance, σ, and the signal to noise ratio, E_b/N_0, is given by

$$\frac{E_b}{N_0} = \frac{E[|z|^2]}{2R^*\sigma^2}, \tag{17}$$

where $E[|z|^2]$ is the average energy of encoder outputs.

A BER performance comparison of MAP decoding with parallel LLR update, MAP decoding with serial LLR update and multistage Viterbi decoding is shown in Fig. 4. As can be seen for both parallel and serial update, more than two decoding iterations are not necessary. Viterbi multistage decoding can be used as a faster solution (Viterbi algorithm is simpler than MAP), but it is outperformed by serial update MAP decoding. Parallel update MAP decoding is both computationally inefficient and BER inferior.

A simplified way of MAP decoding, denoted Max-MAP decoding, is compared to ordinary MAP decoding in Fig. 5. The simplification in Max-MAP decoding is realized by using the max$(\cdot)$ function instead of max*$(\cdot)$, both in the conditional LLR calculation and in the algorithm. If one of the elements in the argument set $\{p_1, \ldots, p_m\}$ is much greater than the others, $\max^*\{p_1, \ldots, p_m\} \approx \max\{p_1, \ldots, p_m\}$. An advantage of Max-MAP decoding is that estimation of noise component variance, σ, is unnnecessary, due to the homogeneity of max$(\cdot)$. The BER performance degradation is practically negligible, making Max-MAP decoding the preferred solution.

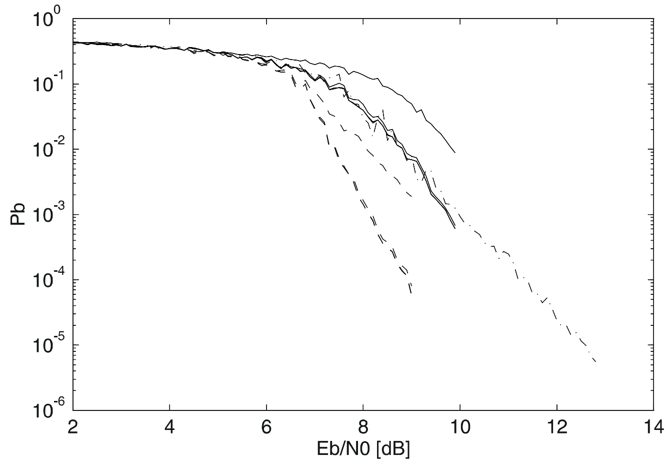


Fig. 4. BER performance of MAP decoding (up to 4 iterations) with parallel LLR update (full lines), serial LLR update (dashed lines) and multistage Viterbi decoding (dash-dot line)

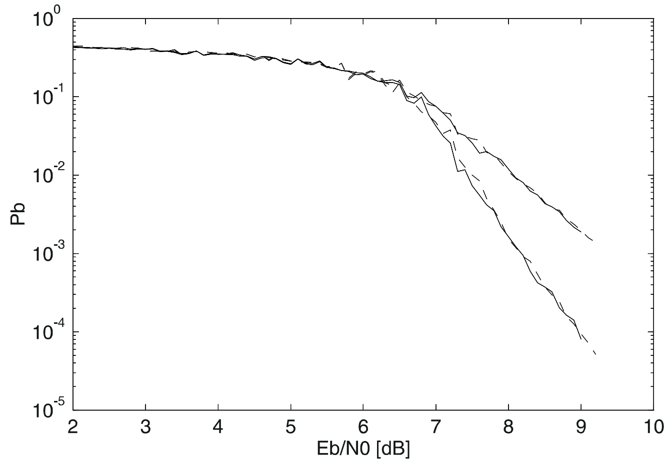


Fig. 5. BER performance of MAP decoding (full lines) and Max-MAP decoding (dashed lines), 2 serial LLR update iterations

References

[1] U. Wachsmann, F. H. Fischer, J. B. Huber, "Multilevel Codes: Theoretical Concepts and Practical Design Rules", *IEEE Transactions on Information Theory*, vol. IT-45, pp. 1361-1391, July 1999.

[2] C. Berrou, A. Glavieux, "Near Optimum Error Correcting, Coding and Decoding: Turbo-Codes", *IEEE Transactions on Communications*, vol. COM-44 pp. 1261-1271, October 1996.

[3] S. Benedetto, D. Divsalar, G. Montorsi, F. Pollara, "Serial Concatenation of Interleaved Codes: Performance Analysis, Design and Iterative Decoding", *IEEE Transactions on Information Theory*, vol. IT-44, pp. 909-926, May 1998.

[4] L. R. Bahl, J. Cocke, F. Jelinek, J. Raviv, "Optimal Decoding of Linear Codes for Minimizing Symbol Error Rate", *IEEE Transactions on Information Theory*, vol. IT-20, pp. 284-287, March 1974.

[5] S. Benedetto, G. Montorsi, "A Soft-Input Soft-Output Maximum A Posteriori (MAP) Module to Decode Parallel and Serial Concatenated Codes", *The Telecommunications and Data Acquisition Progress Report 42-127*, Jet Propulsion Laboratory, Pasadena, California, November 1996.

[6] A. J. Viterbi, "An Intuitive Justification and a Simplified Implementation of the MAP Decoder for Convolutional Codes", *IEEE Journal on Selected Areas in Communications*, February 1998., pp. 260-264