Extended Kalman Filter for Turbo-synchronization Application to DVB-RCS

Frédéric Cérou

▶ To cite this version:

Frédéric Cérou. Extended Kalman Filter for Turbo-synchronization Application to DVB-RCS. [Research Report] RR-5443, INRIA. 2004, pp.16. inria-00070564

HAL Id: inria-00070564

https://hal.inria.fr/inria-00070564

Submitted on 19 May 2006

Extended Kalman Filter for Turbo-synchronization Application to DVB-RCS

Frédéric Cérou

N° 5443

December 2004

Thème NUM

ISSN 0249-6399 ISRN INRIA/RR--5443--FR+ENG

Extended Kalman Filter for Turbo-synchronization Application to DVB-RCS

Frédéric Cérou [*]

Thème NUM — Systèmes numériques
Projet Aspi

Rapport de recherche n° 5443 — December 2004 — 16 pages

Abstract: We give in this paper a new algorithm combining the so called Extended Kalman Filter, a technique coming from stochastic filtering, within a turbo synchronization loop. Simulated numerical results are presented in the last section.

Key-words: Extended Kalman Filter, Turbo-synchronization

This work was partially supported by Alcatel Space, Toulouse

[*] Frederic.Cerou@inria.fr

Filtre de Kalman Etendu pour la Turbo-synchronisation Application à DVB-RCS

Résumé : Nous décrivons ici un nouvel algorithme qui combine le Filtre de Kalman Etendu, une technique qui vient du filtrage stochastique, à l'intérieur d'une boucle de turbo synchronisation. Des résultats numériques obtenus par simulation sont donnés dans la dernière partie.

Mots-clés : Filtre de Kalman Etendu, Turbo-synchronisation

1 Generalities about Extended Kalman Filter

We recall here the basic equations of EKF in a simple setting (see [6] for details). Let us consider the following model. X is a Markov chain, taking its values in $\mathbb{R}^d$, and with the following dynamics:

$$X_{k+1} = f(X_k) + \sigma w_k, \tag{1}$$

where the w_k are independent and identically distributed (i.i.d.), whose law is known. Are known as well the f function and the law of X_0, denoted P_0. On this hidden process we get measures of the form:

$$Z_k = h(X_k) + v_k, \tag{2}$$

where the w_k are i.i.d., often Gaussian, and independent of X. Then we are interested in estimating the trajectory of X,using the measurements Z_k.

We write here the EKF equations in the framework of equations (1) and (2). Note that this method is often used in practice, for example to track moving objects.

Assume that $\hat{X}_{k-1}$ is the previous estimate of the state, and P_{k-1} a corresponding pseudo-variance (a kind of "very rough" estimate of the variance). We first linearize the state equation:

$$X_k \simeq \hat{X}_k^- + A_k(X_{k-1} - \hat{X}_{k-1}) + \sigma w_{k-1},$$

with

$$A_k = \frac{\partial f}{\partial X}(\hat{X}_{k-1}) \quad \text{and} \quad \hat{X}_k^- = f(\hat{X}_{k-1}).$$

We do the same with the observation:

$$Z_k \simeq h(\hat{X}_k^-) + H_k(X_k - \hat{X}_k^-) + v_k,$$

with

$$H_k = \frac{\partial h}{\partial X}(\hat{X}_k^-).$$

Using Kalman-Bucy filter equations (which completely solves the problem the in linear Gaussian case [7]) we can obtain a new state estimate $\hat{X}_k$ and a new pseudo-variance P_k in two steps. Let us denote $Q = \sigma\sigma^T$ the state equation variance, and R the observation noise variance.

Prediction

$$\hat{X}_k^- = f(\hat{X}_{k-1}) \text{ and } P_k^- = A_k P_{k-1} A_k^T + Q.$$

Correction

$$\hat{X}_k = \hat{X}_k^- + K_k(Z_k - h(\hat{X}_k^-)) \text{ and } P_k = (I - K_k H_k) P_k^-,$$

where K_k is the gain :

$$K_k = P_k^- H^T (H P_k^- H^T + R)^{-1}.$$

Note that even if it comes from a linear algorithm, the EKF is itself non linear.

2 Phase tracking

2.1 Phase noise and modulation

The problem under investigation here is motivated by satellite communication. We assume a simple channel model with no Inter Symbol Interference (ISI), and the only problem is to eliminate the residual phase. This is very important as we assume that the transmitted message is Quadratic Phase Shift Keying (QPSK) modulated. All this means that the observations are of the form:

$$Z_k = a_k e^{i\theta_k} + W_k,$$

where the transmitted symbols a_k are in $\{1, i, -1, -i\}$, θ_k is a random (unknown) phase, and W_k an i.i.d. sequence of complex Gaussian random variables. The bits to symbols mapping is done as follows: if the bits (with values 0 or 1) to be transmitted are $b_1, \ldots, b_l, \ldots, b_N$, we group them in pairs $(b_1, b_2), \ldots, (b_{2*k-1}, b_{2k}), \ldots, (b_{N-1}, b_N)$, then for the k^{th} pair the corresponding symbol is given by:

- if $(b_{2*k-1}, b_{2k}) = (0,0)$ then $a_k = 1$,
- if $(b_{2*k-1}, b_{2k}) = (1,0)$ then $a_k = i$,

- if $(b_{2*k-1}, b_{2k}) = (1,1)$ then $a_k = -1$,
- if $(b_{2*k-1}, b_{2k}) = (0,1)$ then $a_k = -i$.

Usually the unknown phase is estimated through a Phase Lock Loop (PLL). Note that PLLs can be viewed as an Extended Kalman Filter with a constant gain.

The dynamical model for the random phase is the same as in [2]:

$$\theta_{k+1} = \theta_k + \delta\theta + \sigma_\theta B_k, \tag{3}$$

where $\delta\theta$ is constant but unknown, σ_θ may be known or not, and the B_k's are i.i.d standard Gaussian random variables.

In [2] the non linear filtering approach was already applied to phase tracking, using the particle filter implementation. The authors considered only uncoded data. The overall conclusion was that in this case this approach performed significantly better than the standard PLLs mainly at the beginning (smaller acquisition time). From some of our experiments not presented in this paper, the reason seems to be that PLLs have constant gains, which are chosen to have good asymptotic (and not transient) properties. If we take a varying gain given by the EKF, then in most case there is not much difference between the particle filter and the EKF. Also note that the idea to use EKF for phase tracking is not new, as it was already proposed in [1], but it was considered too complex by that time to be implemented in a receiver.

2.2 Coding

We consider in this paper the DVB-RCS norm (see [5]), and more precisely one of its Turbo-Code implementations. Turbo-Codes, first introduced by [4] are very robust to low SNRs (under 3dB) but they are also known to be very sensitive to phase errors, so we need an efficient algorithm to estimate the random phase error which works well at very low SNR.

The Turbo-Code in DVB-RCS is depicted in figure 1. It uses a double Circular Recursive Systematic Convolutional (CRSC) code. The first bit of the message to be coded is assigned to A, the second one to B, and so on until the end of the sequence. The polynomials characterizing the connections are:

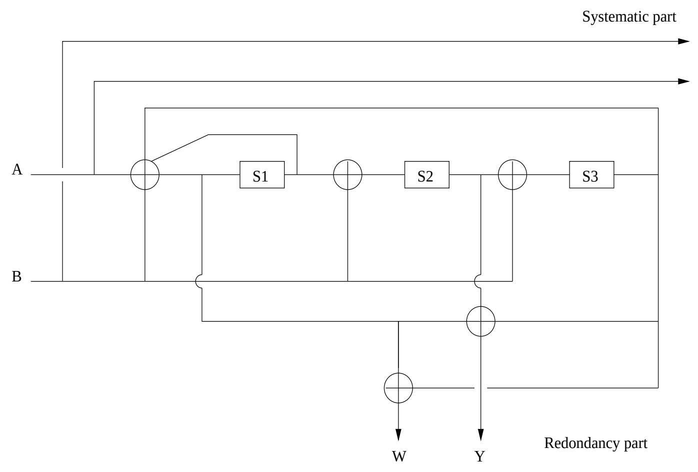

Figure 1: Encoder block diagram

feedback branch: $1 + D + D^3$,

Y parity bits: $1 + D^2 + D^3$,

W parity bits: $1 + D^3$.

We chose the rate $1/2$ so the puncturing pattern keeps only the Y parity bits. The bits are modulated and transmitted in couples, in natural order, that is the couples (A, B) first – systematic part –, and then the couples (Y_1, Y_2) of the two redundant sequences – Y_1 by coding the original sequence, and Y_2 by coding the sequence after the permutation.

2.3 Turbo Synchronization

At low SNR, PLLs do not provide good efficiency to be used in conjunction with powerful Turbo-Codes. The main problem is that both part of the unknown – bits and phase – are estimated independently of each other. From a statistical point of view, this is far from optimality. The right approach should be to estimate both bits and phase jointly. Unfortunately this seems still intractable with a reasonable computational burden.

A first idea to overcome this difficulty is to use the intermediate output of the decoder at each Turbo iteration to re-estimate the phase. This is the basic idea of what is presented in the literature as Turbo-Synchronization, and it is a kind of Expectation Maximization (EM) algorithm (see [9]). As the Turbo iterations are performed, the bits are better estimated, and so are – hopefully – the phase. This approach may be found in several papers, e.g. [10, 12, 9]. This algorithm gives very good results but only for a constant phase. When the phase is varying, the estimate of the first Turbo iteration may be too far away from the truth to estimate correctly the phase, and thus the method may not converge.

Going one step further towards the joint estimation of all the unknowns we can re-estimate the phase in the decoder, at each inside step – that is at every new measurement. Note that from a stochastic filtering point of view, this may require a particle filter to estimate the phase, each particle having attached a decoder to jointly estimate the bits. Unfortunately this is still a bit unrealistic.

At this stage it is useful to note that the two components – bits and phase – have very different dynamics: the bits are changing much faster than the phase. This means that we can use a past estimate on the phase to estimate the current bit, and then use this to re-estimate the phase.

This idea is used in [8] under the term Tentative Decisions (TD), which was first introduced for phase estimation in [11]. This is also the idea we use in the sequel, with the difference that we consider a different setup, and instead of a PLL to estimate the phase, we use an EKF. Inside the decoder, we also used BCJR algorithm instead of Soft Output Viterbi. We estimate dynamically the phase only in the forward part of the decoding algorithm, and use it unchanged for the backward part.

We did not designed the EKF to deal with $\delta\theta$, as it is a constant that can be only estimated after several Turbo iterations. Instead we set up the EKF with a large value of σ_θ and estimate the linear trend after several Turbo iterations. An example of phase estimate along one single packet after the algorithm have converged is shown in figure 2 (SNR is 1.6 dB). The dashed line shows the linear trend as estimated.

Note also that at each Turbo iteration it is necessary to re-estimate the phase for the symbols corresponding to the systematic part using the bit probabilities from the last Turbo iteration, and then to use these phase estimates to recompute the probabilities of the systematic part.

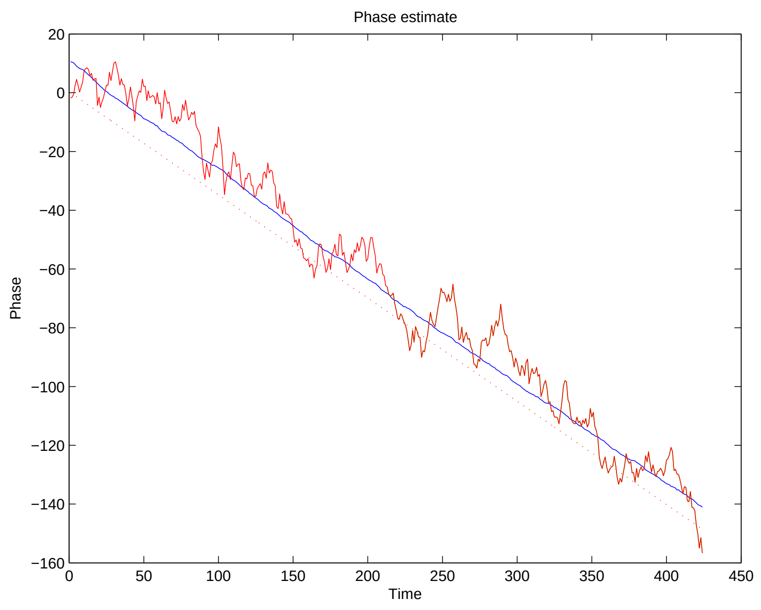

Figure 2: Example of phase estimate

3 Implementation details

In this section we give more details of the practical implementation. First we write the EKF for synchronization.

3.1 EKF equations for phase tracking

Assume first that the transmitted symbols are known, and that $\delta\theta = 0$. Multiplying the observations by a_k^{-1}, the dynamical model for the phase may be written as:

$$\begin{aligned} \theta_k &= \theta_{k-1} + \sigma_\theta B_k, \\ Z_k &= \begin{bmatrix} \cos(\theta_k) \\ \sin(\theta_k) \end{bmatrix} + \sqrt{R} W_k, \end{aligned}$$

with B and W resp. 1 and 2 dimensional standard Gaussian white noises independent of each other. In this setting, EKF equations are:

Prediction

$$\hat{\theta}_k^- = \hat{\theta}_{k-1}$$

$$P_k^- = P_{k-1} + \sigma_\theta^2.$$

Correction

$$\hat{\theta}_k = \hat{\theta}_k^- + K_k \left(Z_k - \left[\begin{array}{c} \cos(\hat{\theta_k}^-) \\ \sin(\hat{\theta_k}^-) \end{array} \right] \right)$$

$$P_k = \left(1 - K_k \left[\begin{array}{c} -\sin(\hat{\theta_k}^-) \\ \cos(\hat{\theta_k}^-) \end{array} \right] \right) P_k^-,$$

where K_k is the gain :

$$K_k = P_k^- \left[\begin{array}{cc} -\sin(\hat{\theta_k}^-) & \cos(\hat{\theta_k}^-) \end{array} \right] \left(P_k^- \left[\begin{array}{cc} \sin^2(\hat{\theta_k}^-) & -\sin(\hat{\theta_k}^-)\cos(\hat{\theta_k}^-) \\ -\sin(\hat{\theta_k}^-)\cos(\hat{\theta_k}^-) & \cos^2(\hat{\theta_k}^-) \end{array} \right] + \left[\begin{array}{cc} R & 0 \\ 0 & R \end{array} \right] \right)^{-1}.$$

3.2 Overall algorithm

The overall algorithm is depicted in details in figure 3. Like any turbo synchronization procedure, the only information passed from one decoder to the other is the extrinsic information. Note that here it is necessary to pass the extrinsic information to the synchronization of the systematic part, to use it to improve the phase estimate for this part. In the picture, π denotes the permutation used for the turbo code.

3.3 Systematic part

For the systematic part there is no actual information in the bits which may be used to synchronize. So we first compute the phase, using the equations of section 3.1, where the transmitted symbol is estimated using the extrinsic

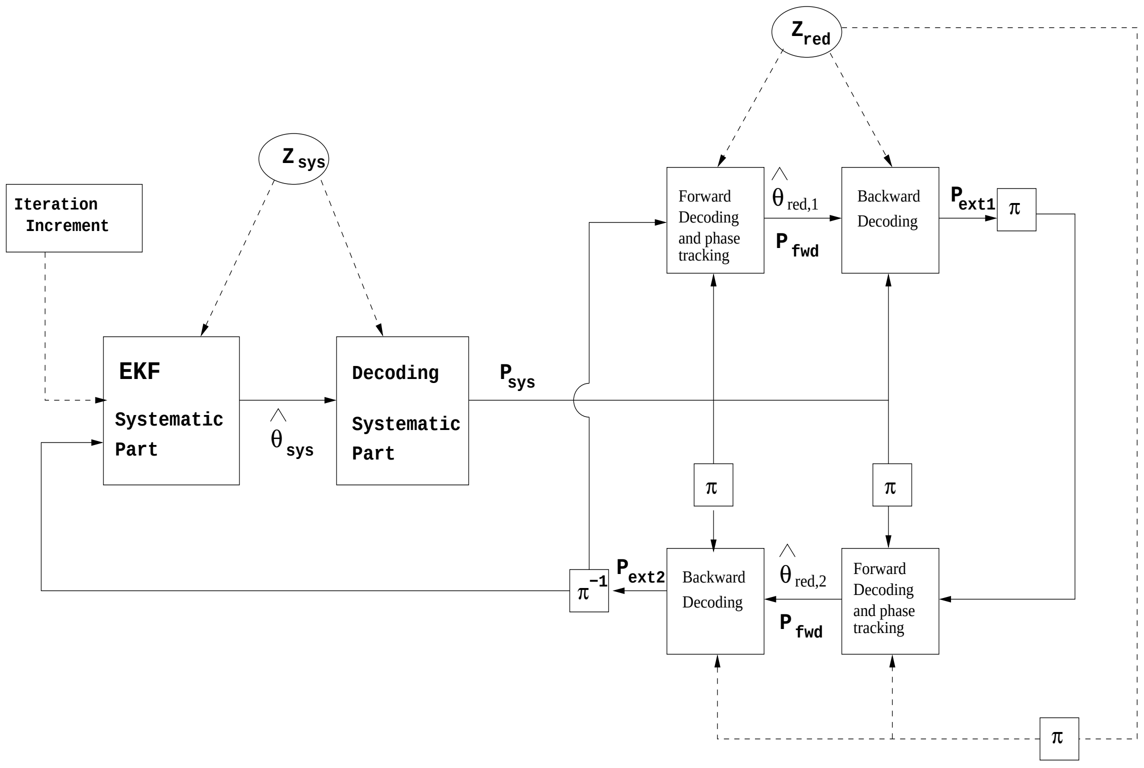

Figure 3: Diagram of joint turbo synchronization and decoding

information of the last decoder iteration (or only the observations at the fist iteration), and then this estimated phase is used to demodulate the observations, giving the systematic probabilities P_{sys}. To estimate the symbol, we simply take the maximum of the likelihood based only on the extrinsic information and the current measurement Z.

3.4 Redundant part

Like in a standard turbo decoder, there are two decoders dealing respectively with the data and the permuted data, and sending to each other their extrinsic information. Each decoder is divided in a forward pass and a backward pass. The backward pass is standard, and only the forward pass deals with the phase estimation.

The forward processing is depicted in figure 4. It is the most tricky part of the whole method.

Assume first that we have an estimate $\hat{a}$ of the current symbol. Then assuming the current symbol is actually $\hat{a}$, and using section 3.1 we compute a new estimate for the phase by applying one step of the EKF. This gives an estimate $\hat{\theta}$ of the current phase, which is used to demodulate the current observation. This is done by the right hand part of the picture.

To have an estimate of the symbol, we chose compute an approximate conditional probability of the four possible symbols, and take the one with the maximum probability (i.e. the idea of a Maximum A Posteriori – MAP– estimate). Practically, we compute the law of the current symbol, conditionally to all the past observations, the extrinsic information, and the systematic part, and moreover assuming that the past trajectory of the phase is that given by the EKF. But to compute this, we need to compute one decoder step of *each* redundant part, because each symbol is the result of the redundant bits (Y_1, Y_2) of the two redundant sequences (see section 2.2). Having the probabilities of both Y_1 and Y_2 we can easily get the probability of the possible symbols, and then take the most likely $\hat{a}$.

Then assuming that the current symbol *is* $\hat{a}$, we compute a new phase estimate using one step EKF (see section 3.1). And then iterate to the next time step.

Note that as usual in Turbo decoding, the only information passed from one iteration to the next is the extrinsic information, and nothing about the phase estimation.

3.5 Linear trend

The algorithm presented so far do not take into account the (unknown) constant $\delta\theta$ in equation 3. If we use it with $\delta\theta \neq 0$, the decoder may fail to lock on the true values of the transmitted bits. To improve the performances in this case, we have implemented the following. After a "warm up" of several iterations (we chose 6 in our simulations), we estimate the linear trend of the phase estimate (the dashed strait line on figure 2). Then every several iterations (say 3), we compute the mean $\hat{m}$ of the estimated linear trends, and multiply each observation Z_k by $\exp[-i\,\hat{m}\,t_k]$.

3.6 Stopping test

After the phase estimate stabilizes, i.e. the maximum difference between two consecutive iterations is less than some small ε, we go on for 8 more iterations to make the turbo decoding converge.

If the phase estimate does not stabilizes, or does too late, we chose to stop the algorithm after 40 iterations to keep the computation time reasonable.

4 Simulation results

We show in this section some simulation results. The choice of the different parameters is the following. For the random phase dynamics, we decided to tune the algorithm to deal with severe conditions, that is $\delta\theta = 0.0063$ rad which corresponds to a 2π phase shift every 1000 symbols. We made the experiments on packets of 53 bytes length (ATM cells). Note that synchronization on small sequences is quite difficult. We assume that the initial phase is known up to an error of 0.175 rad (10 degrees). With the same settings, we ran the algorithm for different values of $\delta\theta = 0.0063, 0.0025, 0$. The results (Frame error rate and Bit error rate) are presented on figures 5 and 6. We show also on these figures the results when the phase is perfectly known as a lower bound, and

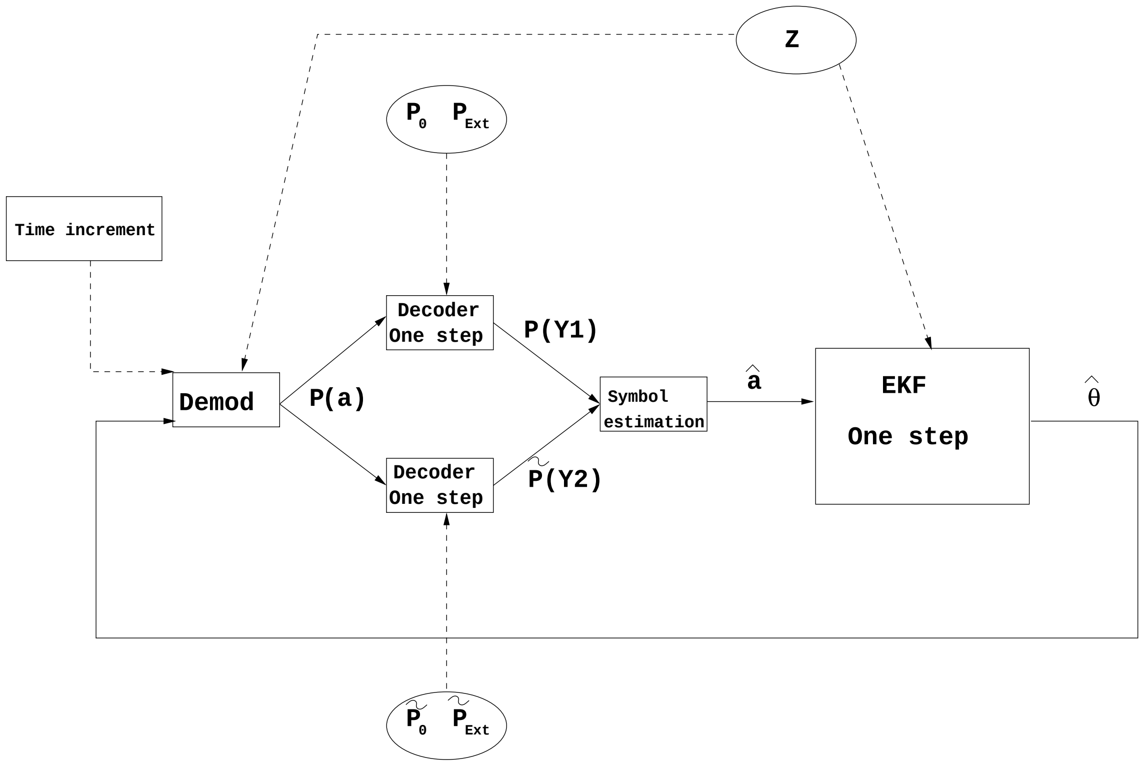

Figure 4: Diagram of joint phase tracking and forward decoding

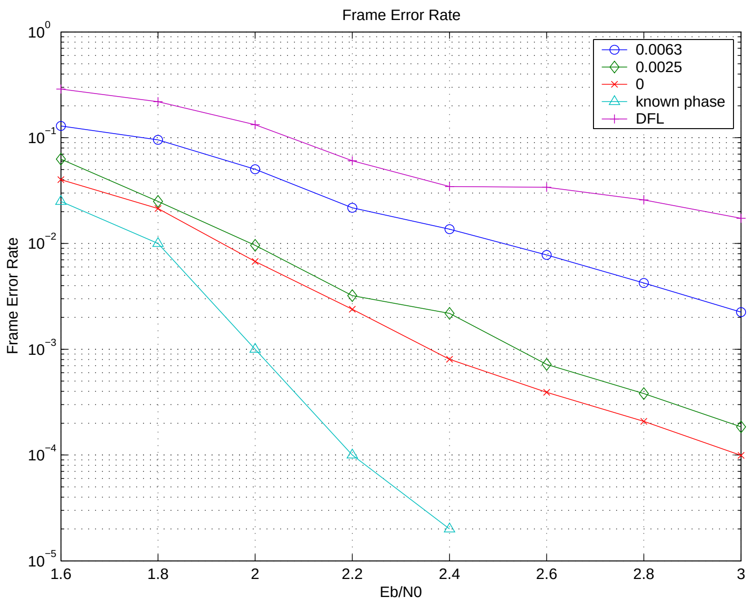

Figure 5: Frame error rate

the results when instead of an EKF we used a Decision Feedback Loop (DFL), i.e. with a constant gain.

Even in these severe conditions, we can achieve a Frame error rate lower than 10^{-2} for a SNR (Eb/N0) of 2.6 dB.

5 Conclusion

In this paper we recalled some results on (Turbo)-synchronization and showed how it can be implemented for the DVB-RCS norm for satellite communications, using Turbo iterations in conjunction with an EKF, which seems to perform better than a simpler DFL.

The proposed algorithm performed a frame error rate lower than 10^{-2} for a SNR (Eb/N0) of 2.6 dB and a 2π phase shift every 1000 symbols, on packets of 53 bytes length (ATM cells). Note that from each packet to the next, the phase estimation was restarted from scratch, which may be considered as severe synchronizing conditions.

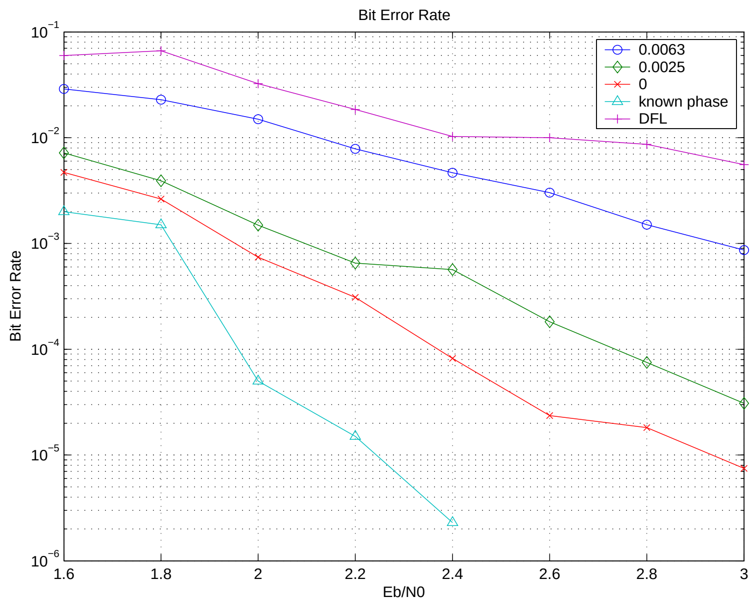

Figure 6: Bit error rate

The overall complexity of the algorithm is just several times that of simple turbo decoding, and should not be the main drawback.

References

[1] A. Aghamohammadi, H. Meyr, and G. Aschied. Adaptative synchronization and channel parameter estimation using an extended Kalman filter. *IEEE Transactions on Communications*, 37(11):1212–1219, nov 1989.

[2] P.O. Amblard, J.M. Brossier, and E. Moisan. Phase tracking: what do we gain from optimality? particle filtering versus phase-locked loops. *Signal Processing*, 83(1):151–167, 2003.

[3] R. Bahl, J. Cocke, F. Jelinek, and J. Raviv. Optimal decoding of linear codes for minimizing symbol error rate. *IEEE Trans. Inf. Th.*, 1974.

[4] C. Berrou, A. Glavieux, and P. Thitimajshima. Near shannon limit error-correcting coding and decoding: Turbo-codes. In *IEEE International Conference on Communications*, Geneva, 1993.

[5] European Telecommunication Institute (ETSI). *Digital Video Broadcasting (DVB); Interaction channel for satellite distribution systems.* EN 301 790.

[6] Arthur Gelb et al. *Applied Optimal Estimation*, volume 64 of *Mathematics in Science and Engineering.* The MIT Press, Cambridge, MA, 1974.

[7] Rudolph E. Kalman and Richard S. Bucy. New results in linear filtering and prediction theory. *Transactions of the ASME, Series D. Journal of Basic Engineering*, 83:95–108, 1961.

[8] C Langlais and M. Hélard. Phase carrier recovery for Turbo codes over a satellite link with the help of tentative decisions. In *2nd International Symposium on Turbo codes & Related Topics*, Brest, 2000.

[9] N. Noels, A. Dejonghe, V. Lottici, H. Steendam, M. Moeneclaey, M. Luise, and L. Vandendorpe. Turbo synchronization : an EM algorithm interpretation. In *IEEE International Conference on Communications*, Anchorage, 2003.

[10] Wangrok Oh and Kyungwhoon Cheun. Joint decoding and carrier phase recovery algorithm for turbo codes. *IEEE Communications Letters*, 5(9):375–377, sep 2001.

[11] Gottfried Ungerboeck. Channel coding with multilevel/phase signals. *IEEE Transactions on Information Theory*, 28(1):55–67, 1982.

[12] Li Zhang and A. Burr. A novel carrier phase recovery method for Turbo-coded QPSK system. In *European Wireless*, Florence, 2002.

Unité de recherche INRIA Rennes
IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)

Unité de recherche INRIA Futurs : Parc Club Orsay Université - ZAC des Vignes
4, rue Jacques Monod - 91893 ORSAY Cedex (France)
Unité de recherche INRIA Lorraine : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)
Unité de recherche INRIA Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)
Unité de recherche INRIA Sophia Antipolis : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399