BLIND FRAME SYNCHRONIZATION ON GAUSSIAN CHANNEL

Rodrigue Imad, Sebastien Houcke and Catherine Douillard

GET/ENST-Bretagne - PRACOM
Technopole Brest Iroise - CS 83818 - 29238 Brest Cedex 3, France
phone: + (33)229001381, fax: + (33)229001012, email: {firstname}.{lastname}@enst-bretagne.fr

ABSTRACT

Being initially proposed for a Binary Symmetric Channel, we adapt in this paper a blind frame synchronization method to a Gaussian channel. This new method is based on the computation of the Logarithmic Likelihood Ratio (LLR) of the syndromes elements. A comparison between the performance of these two methods is done by synchronizing some Low Density Parity Check (LDPC) codes and Convolutional codes. The simulated results are also compared to the theoretical ones by finding the probabilities of false synchronization for the two methods. The theoretical results may help us to define a synchronization threshold so that the calculation time can be decreased.

1. INTRODUCTION

In digital communication systems, one important step is Channel Coding. At the reception, we should be able to find the beginning of each codeword, so that the decoder can work properly. This is what we call Frame Synchronization.
Classical frame synchronization is realized by adding to the transmitted sequence, some synchronization bits known by the receiver.
The actual codes are very powerful and can decode sequences having several numbers of errors. To keep their performance, traditional synchronization methods should increase the length of the inserted training sequence, which reduces the spectral efficiency of the transmission. For this reason, we are interested in developing blind methods of frame synchronization. Note that when we say blind, we mean that no additional sequence is added to the coded one.
We generalize in this paper a simple method of blind frame synchronization, initially developed for a Binary Symmetric Channel (Hard data), so it can work now on a Gaussian Channel (Soft data).
The remaining part of this paper is organized as follows. Section 2 presents the Hard and Soft techniques of our blind frame synchronization method.
For each technique, we find the theoretical probability of false synchronization, which is compared to our simulated results in the third section. A comparison between the performance of the two methods will also be done by synchronizing some LDPC and convolutional codes. Finally, section 4 concludes the work.

2. BLIND FRAME SYNCHRONIZATION METHOD

In this paper, we consider that the transmitter is sending a binary sequence of codewords. Suppose that a Binary Phase Shift Keying (BPSK) modulation is applied to codewords bits and that the propagation channel is an Additive White Gaussian Noise (AWGN) channel.
We will see later that our synchronization techniques are based on the use of the parity check matrix of the code.
For a code of rate

$$R = \frac{n_c - n_r}{n_c},$$

it is often possible to find the corresponding parity check matrix H of size $n_r \times n_c$, where n_c represents the length of a codeword and n_r the number of parity relations.
Let $s(r)$ be the syndrome of a received codeword r:

$$s(r) = rH^T = eH^T,$$

where e is the corresponding error vector.
At the synchronization position and in a noise free channel, it is clear that the syndrome of a given codeword is equal to zero. This is not the case when we are not synchronized. We will see in the next sections that our synchronization techniques are based on this simple idea.
Let us define now the i^{th} received sample by:

$$x(i) = b(i - t_0) + w(i), \tag{1}$$

where t_0 is a delay introduced by the propagation channel, $b(i) = \pm 1$ a modulated coded bit and $w(i)$ a white gaussian noise.
The received sequence of N samples can be written as:

$$X = [x(1), \ldots, x(N)].$$

The main target of frame synchronization is to find the position of a codeword in the received sequence. In other words, we have to estimate the delay t_0. Without loss of generality, we assume that t_0 is always lower than n_c.

2.1 Hard Synchronization

The method initially proposed in [2] requires a binary sequence to work. Therefore, we should take hard decisions before applying it.
Let

$$Z = [z(1), \ldots, z(N)]$$

with

$$z(k) = \frac{\text{sign}(x(k)) + 1}{2},$$

denote the hard decision values of the bits taken from the received sequence X. And let W_d be an extracted sequence of Z:

$$W_d = [z(d), \ldots, z(d + Kn_c - 1)]^T.$$

W_d is the synchronization window: It is a sliding window of length Kn_c bits, where d represents its position on the received sequence. W_d can also be divided into K blocks, each

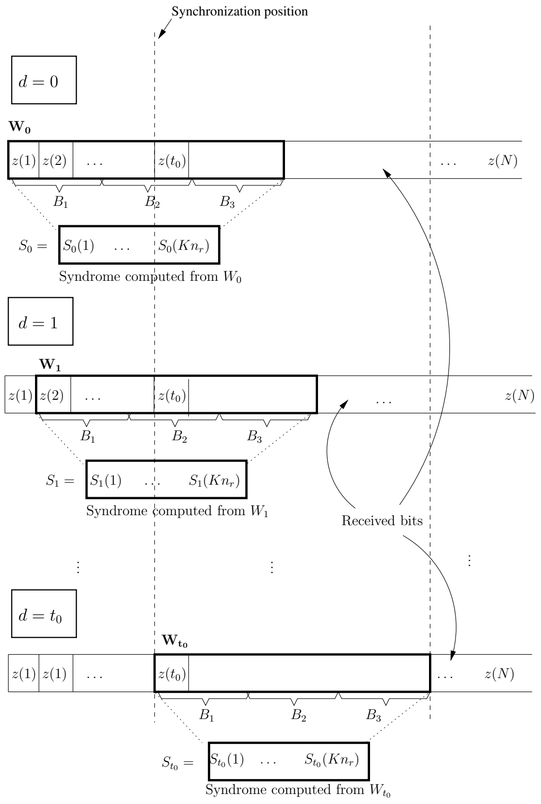

Figure 1: Blind frame synchronization principle

one of length n_c.
For each block, let us calculate its corresponding syndrome, then we form the vector of syndromes S_d:

$$S_d = [S_d(1), \dots, S_d(Kn_r)]^T.$$

Figure 1 represents three different sequences of W_d corresponding for $d = 0$, 1 and t_0, where the size of W_d is fixed to $K = 3$ blocks.
Having S_d, we evaluate $\phi(d)$, the sum of non-zero elements in S_d:

$$\phi(d) = \sum_{k=1}^{Kn_r} S_d(k).$$

By repeating this procedure for n_c consecutive values of d, the frame synchronization instant is estimated as the value of d minimizing ϕ:

$$\hat{t}_0 = \underset{d=0,\dots,n_c-1}{\operatorname{argmin}} \ \phi(d).$$

2.1.1 *Theoretical probability of false synchronization*

Let P_F be the probability of false synchronization:

$$P_F = Pr\big[\phi(t_0) > \min_{d=0,\dots,n_c-1, d\neq t_0} \phi(d)\big].$$

To be able to calculate this probability, we have to find the probability laws followed by $\phi(t_0)$ and $\phi(d)$.
At non-synchronized positions, the k^{th} parity check equation has a probability $1/2$ to be verified and therefore $S_d(k)$ is a Bernoulli random variable of parameter $1/2$.
On the other hand, at the synchronized position $S_{t_0}(k)$ follows a Bernoulli law of parameter p_k where p_k is the probability that $S_{t_0}(k)$ is equal to 1. It is the probability of having an odd number of errors in the u_k bits verifying the k^{th} parity equation.

$$\begin{aligned} p_k &= Pr(S_{t_0}(k) = 1) \\ &= \sum_{l=0}^{\lfloor \frac{u_k-1}{2} \rfloor} \binom{u_k}{2l+1} p_e^{2l+1}(1-p_e)^{u_k-2l-1}, \end{aligned} \qquad (2)$$

where u_k is the number of ones in the k^{th} line of H, p_e the error probability of the equivalent binary symmetric channel, $\binom{n}{k}$ the number of combinations of k elements from n and $\lfloor n \rfloor$ the nearest integer to n towards minus infinity.
Let us assume now that the elements of the syndrome are independent. By assuming that u_k is constant for all the rows of H, we have $p_k = p \ \forall k$ and therefore ϕ is a Binomial random variable:

$$\begin{aligned} \phi(d) &\propto \mathscr{B}(Kn_r, 1/2) \qquad d \neq t_0 \\ \text{and} \quad \phi(t_0) &\propto \mathscr{B}(Kn_r, p). \end{aligned}$$

Finally, the probability of false synchronization is

$$P_F = 1 - \left[\sum_{k=0}^{Kn_r-1} \left(\binom{Kn_r}{k} p^k (1-p)^{Kn_r-k} \sum_{j=k+1}^{Kn_r} \left(\frac{\binom{Kn_r}{j}}{2^{Kn_r}} \right) \right) \right]^{n_c-1}.$$

Let us adapt now this method for a Gaussian Channel.

2.2 Soft Synchronization

The use of hard decision on the received sequence involves a loss of information. Therefore, it is logical to try to benefit from this information and the common way to do it is to calculate the LLR of the received bits. In our case, we are interested in the LLR of $S_d(k)$ [3]. To simplify the notations, let us assume that the length of the synchronization window is n_c bits.
Each element of the syndrome being the sum (modulo 2) of u_k bits:

$$S_d(k) = W_d h_k^T = \sum_{i=1}^{u_k} z(d + k_i), \qquad (3)$$

where k_i is the position of the i^{th} non zero element in the k^{th} line of H.
The LLR of each element of the sum is:

$$L(z(d+k_i)) = \frac{2}{\sigma^2} x(d+k_i), \qquad (4)$$

where σ^2 is the variance of the channel noise. An approximation of the LLR of a syndrome's element is given by [1]:

$$L(S_d(k)) = (-1)^{u_k+1} \Big(\prod_{j=1}^{u_k} sign(L(z(d+k_j))) \Big) \beta \qquad (5)$$

with

$$\beta = \min_{j=1,\ldots,u_k} |L(z(d+k_j))|. \quad (6)$$

The criterion that we consider now is:

$$\phi_L(d) = \sum_{k=1}^{Kn_r} L(S_d(k))$$

and the synchronization position is estimated by:

$$\hat{t}_0 = \underset{d=0,\ldots,n_c-1}{\operatorname{argmin}} \phi_L(d).$$

Notice that $L(z(k))$ and $x(k)$ are proportional. By assuming that the variance of the noise is constant during a transmission, we can replace $L(z(d+k_j))$ by $x(d+k_j)$ in (5) and (6), so that we do not need to know the variance of the noise to make our calculations.
$L(S_d(k))$ is estimated by:

$$\hat{L}(S_d(k)) = (-1)^{u_k+1}\Big(\prod_{j=1}^{u_k} sign(x(d+k_j))\Big) \min_{j=1,\ldots,u_k} |x(d+k_j)|. \quad (7)$$

Note that the term

$$(-1)^{u_k+1}\Big(\prod_{j=1}^{u_k} sign(x(d+k_j))\Big)$$

involved in (7) is the modulated value of $S_d(k)$. An estimation of $\phi_L(d)$ is also obtained by:

$$\hat{\phi}_L(d) = \sum_{k=1}^{Kn_r} \hat{L}(S_d(k)) = \sum_{k=1}^{Kn_r} (2S_d(k)-1)Y_d(k),$$

where $Y_d(k) = \min_{j=1,\ldots,u_k} |x(d+k_j)|$ represents the reliability of $S_d(k)$.

2.2.1 *Theoretical probability of false synchronization*

This time we have continuous variables instead of discrete ones. By applying the Central Limit theorem, $\hat{\phi}_L(d)$ follows a Gaussian law:

$$\hat{\phi}_L(d) \propto \mathcal{N}(Kn_r m_d, Kn_r\sigma_d^2),$$

where m_d and σ_d^2 represent respectively the mean and variance of $\hat{L}(S_d(k))$.
For $d \neq t_0$, we have:

$$Pr(S_d(k)=0) = Pr(S_d(k)=1) = 1/2,$$

whether we have noise or not and therefore $S_d(k)$ and $Y_d(k)$ are independent. This leads to:

$$\begin{aligned} m_d &= 0 \\ \text{and} \quad \sigma_d^2 &= E[Y_d(k)^2]. \end{aligned}$$

Note that σ_d^2 is estimated by simulations.

On the other hand, when $d = t_0$, $S_{t_0}(k)$ and $Y_{t_0}(k)$ are not independent anymore because in this case, the sign of a syndrome depends only on the presence or not of an error. Therefore, m_{t_0} can be written as:

$$\begin{aligned} m_{t_0} &= Pr(S_{t_0}(k)=0)E\big[\hat{L}(S_{t_0}(k))_{/S_{t_0}(k)=0}\big] \\ &\quad + Pr(S_{t_0}(k)=1)E\big[\hat{L}(S_{t_0}(k))_{/S_{t_0}(k)=1}\big] \end{aligned}$$

where $Pr\big(S_{t_0}(k)=1\big) = p$ is the same one computed in (2).
Let us find now

$$E\big[\hat{L}(S_{t_0}(k))_{/S_{t_0}(k)=1}\big] = E\big[Y_{t_0}(k)_{/S_{t_0}(k)=1}\big].$$

Recall that a BPSK modulation is used (cf. (1)):

$$x(i) = b(i) + w(i)$$

where $b(i) = \pm 1$ is a modulated bit and $w(i)$ the noise.
Saying $S_{t_0}(k) = 1$ means that there is an odd number of errors in the u_k bits $z(t_0+k_i)$ involved in (3). And because

$$z(k) = \frac{\text{sign}\big(b(k)+w(k)\big)+1}{2},$$

we can say that among the u_k elements, there is an odd number of elements $w(k)$ that, once added to b(k), changes its sign.
Notice that an error occurs when:

$$\begin{aligned} & b(k) = +1 \quad \text{and} \quad w(k) < -1 \\ \text{or} \quad & b(k) = -1 \quad \text{and} \quad w(k) > 1. \end{aligned}$$

Furthermore, as

$$Pr(b(k)=1) = Pr(b(k)=-1) = 1/2$$

and w(k) is a Gaussian variable of zero mean, we can say that

$$E[Y_{t_0}(k)_{/S_{t_0}(k)=1}] = E\big[\min_{j=1,\ldots,u_k} |1+w(t_0+k_j)|_{/B_1}\big],$$

where B_1 is the event: $\{$Among u_k noise samples, an odd number of these elements have them values lower than $-1\}$.

Doing the same procedure for the case $S_{t_0}(k) = 0$, we have:

$$E\big[\hat{L}(S_{t_0}(k))_{/S_{t_0}(k)=0}\big] = -E\big[Y_{t_0}(k)_{/S_{t_0}(k)=0}\big]$$

and

$$E[Y_{t_0}(k)_{/S_{t_0}(k)=0}] = E\big[\min_{j=1,\ldots,u_k} |1+w(t_0+k_j)|_{/B_2}\big],$$

where B_2 is the event: $\{$Among u_k noise samples, none or an even number of these elements have them values lower than $-1\}$.

Having this, the mean m_{t_0} of $\hat{L}_{t_0}(k)$ is then equal to:

$$\begin{aligned} m_{t_0} &= -(1-p)E\big[\min_{j=1,\ldots,u_k} |1+w(t_0+k_j)|_{/B_2}\big] \\ &\quad + pE\big[\min_{j=1,\ldots,u_k} |1+w(t_0+k_j)|_{/B_1}\big]. \end{aligned}$$

Concerning the variance, we can easily show that:

$$\sigma_{t_0}^2 = E[(\min_{j=1,\ldots,u_k} |x(t_0+k_j)|)^2] - m_{t_0}^2,$$

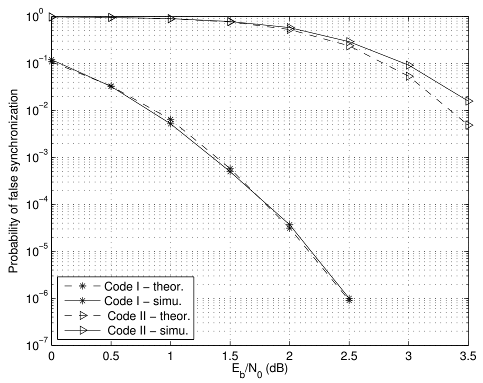

Figure 2: Hard synchronization of LDPC codes

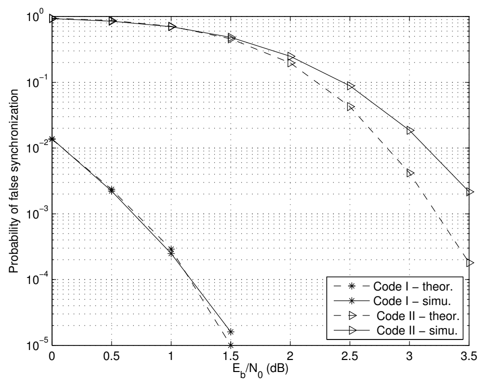

Figure 3: Soft synchronization of LDPC codes

which gives, assuming the independence of the syndromes elements:

$$\hat{\phi}_L(t_0) \propto \mathcal{N}(Kn_r m_{t_0}, Kn_r \sigma_{t_0}^2).$$

Finally, the probability of false synchronization is :

$$P_F = 1 - \left[1 - \frac{1}{2}\mathrm{erfc}\left(\frac{-Kn_r m_{t_0}}{\sqrt{2Kn_r(\sigma_{t_0}^2 + \sigma_d^2)}}\right)\right]^{n_c - 1}.$$

2.3 Simulation results

In order to estimate the performance of our synchronization algorithms, we estimated the probability of false synchronization. The evaluation of this probability is realized by Monte Carlo simulation: up to 1000000 realizations have been performed, where, for each realization, the noise, information bits and the delay of the channel were randomly chosen.

Consider two LDPC codes of length 511 bits and rate 0.7, that differ only by the composition of their parity check matrix: Code I (respectively II) has 4 (respectively 10) non zero elements on each line of its parity check matrix.

By using a synchronization window of size 1 block (511 bits), we applied the two synchronization methods to these codes. Fig. 2 shows the probability of false synchronization versus the Signal to Noise Ratio (E_b/N_0), for the Hard synchronization method.

We notice on this figure that our theoretical and simulated results are almost the same. For code II, the theoretical results are a little bit better than the simulated ones because having 10 non zero elements in each line of the parity check matrix of this code decreases the number of independent syndrome elements. However, in our theoretical computations we considered that all the elements of a syndrome are independent, which explains this difference in performance.

Let us compare now the plotted curves for codes I and II. We notice that the probability of false synchronization increases when the number of non zero elements in the parity check matrix increases.

Fig. 3 shows the theoretical and simulated results of Soft synchronization method applied to the previous LDPC codes. As in the case of Hard synchronization, the theoretical and simulated curves are very close to each other.

Note that for Code I, for E_b/N_0 greater than 1.5 dB, no error was found in the 1000000 realizations.

Let us compare now the differences in performance between Hard and Soft synchronizations: Comparing Fig. 2 and 3, it is clear that Soft synchronization has the best performance: For code I for example, there is a gap of around 0.62 dB between the two methods at a probability of false synchronization equals to 10^{-3}.

Let us apply now our methods to convolutional codes. For this type of codes, non zero elements in the parity check matrix are placed in a "staircase" form. For example, the parity check matrix of the convolutional code of generator polynomials (5,7) can be written as [4]:

$$H = \begin{pmatrix} 1 & 1 & & & & & & & & \\ 0 & 1 & 1 & 1 & & & & & & \\ 1 & 1 & 0 & 1 & 1 & 1 & & & & \\ & & 1 & 1 & 0 & 1 & 1 & 1 & & \\ & & & & 1 & 1 & 0 & 1 & & \\ & & & & & & 1 & 1 & & \\ & & & & & & & & & \\ & & & & & & & & \ddots & \ddots \end{pmatrix}$$

Therefore, for this code we might have:

$$S_d(k) = S_{d-2n}(k+n),$$

for $k > 2$ and n integer. This leads to a degradation in the performance of our synchronization methods. One simple solution to this problem is to interleave the bits in each codeword using a pseudo-random interleaver of size n_c bits.

The convolutional codes to which we applied our synchronization methods have respectively the generator polynomials $(5,7)$, $(23,35)$ and $(561,753)$. These three codes have the same length (512 bits), the same rate $(1/2)$ and constraint lengths equal to 3, 5 and 9 respectively.

Figure 4 shows a comparison between the performance of the two methods after being applied to those convolutional codes. The same behavior obtained with LDPC codes is found here: Soft synchronization is always the best.

By comparing the curves plotted for code $(5,7)$, for a probability of false synchronization equals to 2.10^{-2}, we can see a gap of 0.7 dB between the two methods.

Notice that when the constraint length of a code increases, the number of non zero elements in the parity check matrix increases also, which explains the degradation in the performance of our synchronization methods, as explained in [2].

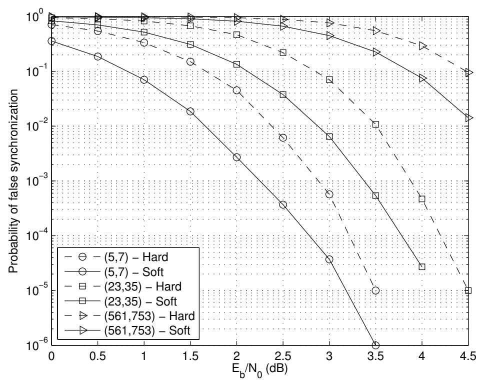

Figure 4: Hard and Soft synchronizations for convolutional codes

Note also that if those three convolutional codes were decoded using a Maximum A Posteriori (MAP) decoder, a frame error rate of around 10^{-2} would correspond to a E_b/N_0 between 5 and 6 dB. However, Fig. 4 shows that for E_b/N_0 greater than 4 dB, synchronizing codes (5,7) and (23,35) gave no error over the 1000000 realizations.
Thus, our synchronization method is well adapted for convolutional codes.

2.4 Conclusion

In this paper, we have described a new method of blind frame synchronization that is based on the calculation of the sum of the LLR of a syndrome elements. Compared with the Hard technique, our Soft approach gives the best results for any Signal to Noise Ratio.
We have also estimated the theoretical probabilities of false synchronization and the results were almost the same as the simulated ones.
After these promising results, our target will be to apply this blind frame synchronization method to other types of codes, in particular turbo codes.

REFERENCES

[1] J. Hagenauer. Iterative decoding of binary block and convolutional codes. *IEEE trans. on Information Theory*, vol. 42, no. 2, pp. 429-445, March 1996.

[2] S. Houcke and G. Sicot. Blind Frame Synchronization for Block Code. In *Proceedings of EUSIPCO, European Sig. Proc. Conf*, Florence, Italy, September 2006.

[3] S. Houcke, G. Sicot and M. Debbah. Blind Detection for block interleaved division multiple access. In *Proceedings of GLOBECOM*, San Francisco, USA, 2006. .

[4] R. Johannesson, K Sh. Zigangirov. Fundamentals of Convolutional Coding, IEEE series on Digital and Mobile Communication, IEEE Press, 1999.