A Fast Timing Recovery Algorithm for Turbo-coded System

Nguyen Duc Long * and Hyuncheol Park*
* School of Engineering, Information and Communications University, Daejeon, 305-714, Korea
Tel : +82-042-866-6808 Fax : +82-042-866-6228 E-mail: {longnduc and hpark}@icu.ac.kr

Abstract: We proposed a new type of timing recovery scheme to work with a system that uses BCJR (Bahl-Cocke-Jelinek-Raviv) decoding algorithm and BPSK modulation. The unknown timing offset is estimated by the modified Mueller and Müller estimator with the aid of the decoder. Timing offset can be acquired as soon as the symbols are received and be updated symbol by symbol. The simulation results for turbo codes whose decoder uses BCJR algorithm show a satisfactory performance even in case of severe timing jitter.

Keywords: Turbo codes, BCJR algorithm, timing recovery.

1. INTRODUCTION

The turbo codes has been shown to be capable of approaching channel capacity limit on an AWGN channel, at least with perfect synchronization provided. The operational signal to noise ratio (SNR) of a conventional turbo coded system is extremely low (around 2 dB). Thus, symbol timing recovery becomes difficult, and demodulated signal itself may not provide enough information for synchronization.

One solution to synchronization problem in low SNR is the cooperation of estimator and decoder in an iterative manner. It means that the estimator does not work separately, but uses the outputs of the decoder as its inputs to estimate the timing offset. The received signals are sampled again with the new values of timing estimates and put to the decoder to get new values of data estimates. These data estimates will be used for the estimator, and the above procedures are repeated. Since the data estimates from the decoder are adequately correct, this solution provides better performance than that of conventional timing recovery algorithm, but an inherent drawback is very long delay [1]. To cope with this new challenge, tentative decisions, the values of data estimates output before complete decoding process, are used for synchronization [2]. The previously proposed algorithms for timing recovery have satisfactory performance but have high complexity, and a fast and simple algorithm is needed.

In this paper, we propose a timing recovery algorithm that uses the M&M (Mueller and Müller) timing updating equation. In order to aid the estimation process, a derivation to utilize the well-known decoding algorithm for turbo codes, BCJR (Bahl-Cocke-Jelinek-Raviv) decoding algorithm, is also proposed. In BCJR algorithm, the decoding process is carried out by calculating necessary parameters in both the forward direction, and backward direction on the trellis of the convolutional codes, so it takes long time. In our algorithm, we need only parameters in the forward direction to calculate the approximate expectations of data estimates, and use these values for M&M estimator. Since the parameters in the forward direction are calculated right after we receive the signal, the timing offset can be estimated during the data reception. After the whole block is received, we use the soft outputs of decoder to update the timing estimates to obtain a highly correct estimation.

The rest of the paper is structured as follows. In section 2, we review the M&M algorithm for timing recovery. The BCJR algorithm is briefly utilized to approximately calculate the expectation of the transmitted code word for M&M estimator and shown in section 3. The system model is presented in section 4. All of these results are used in section 5 for the description of the proposed algorithm. Numerical results are shown in section 6, and in section 7, we conclude the paper.

2. M&M ALGORITHM AND TIMING RECOVERY SCHEME

Let us consider the timing recovery scheme at a receiver in Fig. 1. The number controlled oscillator (NCO) control the sampler to sample the output of the matched filter achieving the digitalized received signal, $y(t_k)$, both for data detector and timing error detector (TED). The error signal (e_k) is put through the loop filter to generate the timing phase, $\hat{\tau}_k$.

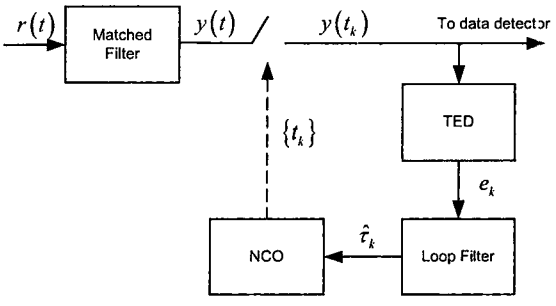

Fig. 1. General timing recovery scheme.

The timing phase is updated by the equation shown below

$$\hat{\tau}_{k+1} = \hat{\tau}_k + \gamma e_k , \quad (1)$$

where γ is step-size parameter.

Using two consecutive samples from the received signals, Mueller and Müller proposed the following equation to calculate error signal [3]

$$e_k = \hat{c}_{k-1} y(kT + \hat{\tau}_k) - \hat{c}_k y\left[(k-1)T + \hat{\tau}_{k-1}\right], \quad (2)$$

where T is symbol duration.

The estimator works well in uncoded system with high SNR but performs worse in a low SNR environment due to unreliable samples. To enhance the capability of this estimator, it is better to use the soft decision of the data instead of hard decision samples. One type of soft values is expectation of the transmitted symbol. We have modified the estimator as follow

$$e_k = E(c_{k-1}) y(kT + \hat{\tau}_k) - E(c_k) y\left[(k-1)T + \hat{\tau}_{k-1}\right], \quad (3)$$

where $E(\cdot)$ is the expectation operation.

In the next section, we will show how to calculate the expectation of the transmitted symbol in case of a soft output decoder (BCJR).

3. BCJR AND HOW TO ACHIEVE THE APPROXIMATE EXPECTATION OF THE TRANSMITTED CODE WORD

Using the following notations [4]

$$\alpha_k(m) = \Pr\{S_k = m; \mathbf{R}_1^k\}, \quad (4)$$

$$\gamma_k(m'; m) = \Pr\{S_k = m; R_k | S_{k-1} = m'\}, \quad (5)$$

where S_k is the state of the trellis at time k, and m, m' are states of the trellis, $\mathbf{R}_1^L$ is vector of transmitted symbols from the time 1 to L. The values of $\alpha_k(m)$ and $\gamma_k(m'; m)$ are calculated according to the BCJR algorithm [4].

For our algorithm, using an additional probability

$$w_k(m', m) = \Pr\{S_{k-1} = m'; S_k = m; \mathbf{R}_1^k\}. \quad (6)$$

This probability can be calculated using parameters given in the BCJR decoder in the forward direction on the trellis as following

$$w_k(m', m) = \Pr\{S_{k-1} = m'; \mathbf{R}_1^{k-1}\} \Pr\{S_k = m; R_k | S_{k-1} = m'\} = \alpha_{k-1}(m') \gamma_k(m'; m). \quad (7)$$

Let B_j be the set of all transitions S_{k-1} to S_k that give the output c_k equal to one of two BPSK symbol C_j (j = 0,1). Then

$$\Pr\{c_k = C_j; \mathbf{R}_1^k\} = \sum_{(m';m) \in B_j} w_k(m'; m). \quad (8)$$

The expectation of the codeword transmitted at time k is approximated as follows

$$E(c_k) = \sum_{j=0}^{1} \Pr\{c_k = C_j; \mathbf{R}_1^k\} \cdot C_j. \quad (9)$$

4. SYSTEM MODEL

The system model is considered as follows. The information bits b_k are grouped into blocks of length N and encoded with a turbo encoder, consisting of two parallel recursive systematic convolutional (RSC) encoders. The information bits and parity check bits (p_{1k}, p_{2k}) are multiplexed and punctured to achieve a code rate of ½. The outputs (d_k) are then modulated by binary phase shift keying (BPSK) modulator to get c_k. Before white Gaussian noise with a double-sided power spectral density of $N_o/2$ is added, the transmitted signals are put through the timing offset and pulse shaping modules as shown in Fig. 2.

At the receiver, the output of the matched filter is sampled with the current value of timing offset estimate, then de-multiplexed to get the information and parity parts of the received signals (x_k^s, x_k^{p1}, x_k^{p2}) (Fig. 4). These symbols are fed into the two decoders to calculate the parameters for BCJR algorithm. These parameter values will be used in conjunction with the received signals for the timing estimator to generate an update of the timing estimate.

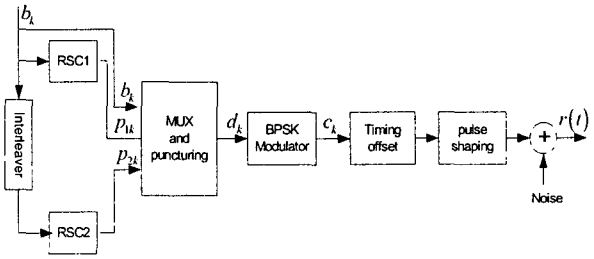

Fig. 2. Block diagram of the transmitter.

5. PROPOSED ALGORITHM

The iterative timing recovery process can be divided into two steps

- *Pre-iteration* Estimating the timing offset before receiving all the bits of a block.

First, initialize the carrier timing offset estimate with some value. When each symbol is received, sample it with the current estimate of timing offset, then put it into the decoder to calculate the values of parameters in forward direction on the trellis ($\alpha_k(m), \gamma_k(m', m)$) according to BCJR algorithm. Use the values of these parameters (of decoder 1 if turbo code is used) to calculate the expectation of transmitted information symbol (x_k^s) as in (9). This value is used in conjunction with the received signal for the M&M equation, as in (1), (3), to update the timing estimate. For the next symbol, the procedure is repeated. After receiving all the transmitted symbols of a block, go into iteration step. The block diagram of this step is shown in Fig. 3.

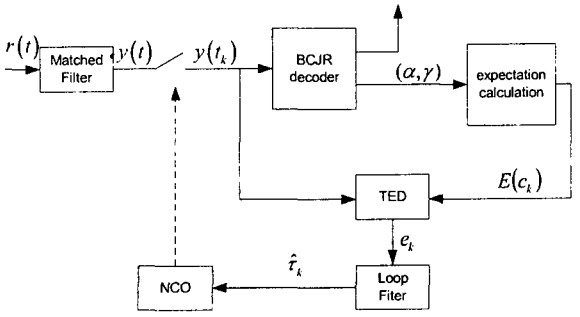

Fig. 3. Proposed algorithm

-*Iteration* At this time, all the received symbols are sampled with corresponding timing offset estimates and available at the input of the decoder. After decoding the block, the receiver uses the soft outputs to calculate the expectation of the transmitted symbol as in (10). This value is used for the timing estimator to update the timing estimate as in (1), (3). Note that the index k is the current

times of iterations. The received symbols are resampled by the new timing estimates, are then put into the decoder for a new iteration.

Fig. 4 shows the block diagram of proposed algorithm in case of a turbo-coded system. Switch 1 is closed when the estimator is in the pre-iteration step. After we receive the whole symbols of the block, switch 2 is closed to receive the expectation of the soft output from the decoder (10).

It is clear that the relation between the log-likelihood ratio and expectation of a bit is as follows

$$E(x_k^s) = \tanh\left(\frac{LLR(b_k)}{2}\right). \qquad (10)$$

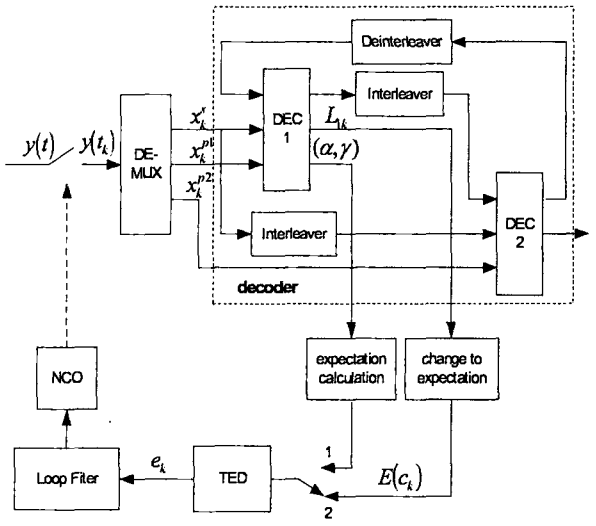

Fig. 4. Block diagram of proposed algorithm in the case of turbo coded system

6. SIMULATION RESULTS

For simulation, using the turbo encoder with two identical recursive systematic convolutional encoders (RSC), each has 8-state trellis and generators $g_1 = (1101)_2$ and $g_2 = (1011)_2$. The interleaver is pseudo-random with block length of 1008. The code rate is ½. The timing offset is modeled as a random walk $\tau_{k+1} = \tau_k + N(0,\sigma^2)$, where the variance σ^2 determines the severity of the timing jitter. The overall filter of both transmitter and receiver is a Nyquist filter.

Fig. 5 shows the BER performance as a function of E_b/N_0 for the different levels of timing jitter. The severity is ranged from $\sigma^2 = 0$ (constant timing) to $\sigma^2 = 5\cdot10^{-4}$ (very severe). In practice, σ^2 rarely rises up to 10^{-4}. We can see that at BER of 10^{-3} and $\sigma^2 = 10^{-4}$ the performance of the system is just 0.2 dB worse than that of ideal case.

The ability to track the large value of initial timing offset (the timing offset of the first symbol in the block) is shown in Fig. 6. An initial timing offset of up to 0.3 (normalized by symbol duration) is acquired with little degradation in performance of whole system. The performance gets worse when a large value of initial timing offset (near half of symbol duration) is applied. This demonstrates the limit of a blind algorithm.

Fig. 7 shows the acquisition time of proposed algorithm. In previously proposed algorithm, the acquisition time is not less than the length of a codeword (several hundreds to thousands of bit). Meanwhile the algorithm proposed in this paper can acquire the timing offset before receiving the whole block of the transmitted codeword. From the figure, we can see that when the timing offset is 0.3 (normalized by symbol duration), the estimator can acquire the timing in 100 symbols. In the worst case, the timing offset equals half of a symbol duration, the estimator can acquire the timing after nearly 250 symbols.

In case of large initial timing offset value, the results in Fig. 6 and Fig. 7 seem inconsistent, we still can acquire the timing offset but the whole performance is not attractive. This phenomenon can be explained as follows. This is a blind algorithm and the turbo codes work in very low SNR environment, this makes the tracking loop unstable with cycle slipping.

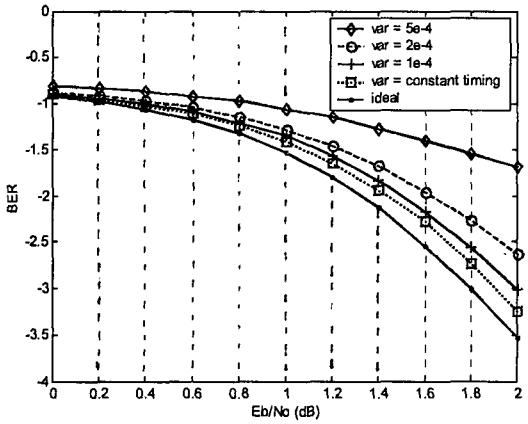

Fig. 5. BER performance vs E_b/N_0 for different severity levels, using 3 iterations

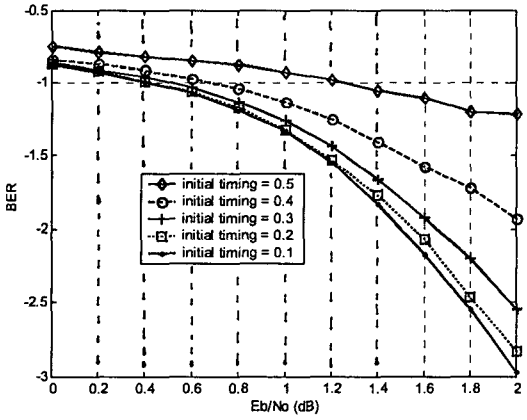

Fig. 6 BER performance vs E_b/N_0 for different values of initial timing offset, 3 iterations, $\sigma^2 = 10^{-4}$

7. CONCLUSIONS

We have proposed a fast and robust timing offset estimation algorithm with BER performance is nearly close to that in ideal synchronization case. This is also a low complexity estimator since there is no change inside the structure of the decoder and only little extra calculation is needed. This application of this algorithm is not only for turbo coded systems, but also for all kind of receivers using soft output decoders.

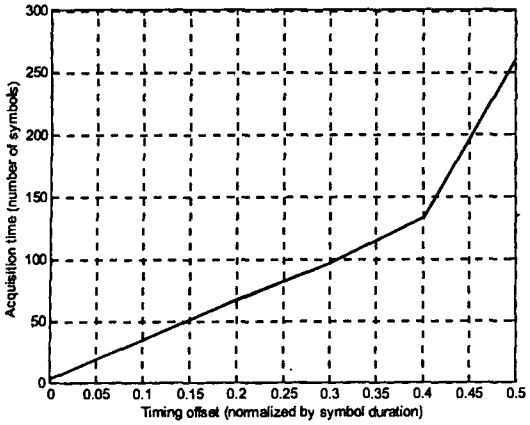

Fig. 7. Acquisition time characteristic, using pre-iteration step only, $\sigma^2 = 10^{-4}$.

References

[1] Z. Li, A. Burr, "APPA symbol timing recovery scheme for turbo codes", *IEEE International Symposium on Personal, Indoor and Mobile Radio Communications*, vol. 1, pp. 44-48, Sept. 2002.

[2] U. Mengali and A. N. D'Andrea, *Synchronization Techniques for Digital Receivers*, New York Plenum, 1997.

[3] K. Mueller, M. Muller, "Timing Recovery in Digital Synchronous Data Receivers", *IEEE Transactions on Communications*, vol. 24, no 5, pp. 516 – 531, May 1976.

[4] L. Bahl, J. Cocke, F. Jelinek and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate", *IEEE Transactions on Information Theory*, vol. 20, no 2, pp. 284–287, March 1974.