Joint Synchronization and Decoding Exploiting the Turbo Principle

John M. Walsh and C. Richard Johnson, Jr.
School of E.C.E.
Cornell University
Email: jmw56@cornell.edu,
johnson@ece.cornell.edu

Phillip A. Regalia
Dept. Comm., Image and Information Processing
Inst. National des Télécommunications/GET
91011 Evry cedex France

***Abstract*— This paper investigates turbo methods for joint synchronization and decoding in pulse amplitude modulated (PAM) systems. We begin with a brief review of the turbo principle before applying it to the synchronization problem. Specifically, we propose and investigate a baud timing offset synchronization algorithm for error control coded systems. Unlike many previously proposed algorithms, which feed estimates for the coded symbols to the decoder, our algorithm calculates and exchanges extrinsic information values for the coded bits, as is consistent with the turbo principle. Simulations prove the algorithm's utility and verify that it has performance near to that of an ideal synchronizer achieving the Cramer Rao lower bound on timing error estimation variance. We investigate both systems employing regular convolutional error correcting codes as well as serially concatenated convolutional turbo codes.**

I. Introduction

Much of past research on synchronization of (turbo) coded systems, [1]–[6], has focused on the use of maximum likelihood estimation theory to determine the various synchronization parameters (e.g. carrier phase, timing phase, and frame synchronization). While this research has addressed the parameter estimation problem rigorously (see, for example, the EM algorithm justification in [7]), it has focused on using the estimated synchronization parameters to create a synchronized signal which is passed to the decoder. In systems with baud timing offsets that use pulse shapes whose bandwidths exceed the symbol rate this can result in a loss of sufficient statistics for the transmitted symbols, [8], [9]. An optimal (minimum BER) receiver allows the decoder to determine the a posteriori probabilities (APPs) for the coded bits.

In this paper, we investigate using the estimated synchronization parameters in a manner motivated by the general (serially concatenated) turbo principle. Specifically, we consider the synchronization unit to be another decoder, which we use in a turbo scheme. Instead of passing a synchronized signal, the synchronizer passes extrinsic information (EI) for each of the coded bits to the decoder (which may or may not be a turbo decoder). The decoder then generates EI for the coded bits using the EI from the synchronizer as a priori information, as is consistent with the turbo principle. Next, the EI values from the decoder are passed as a priori probabilities to the synchronizer, which uses them to update its synchronization parameter estimates and its EI, and the structure iterates.

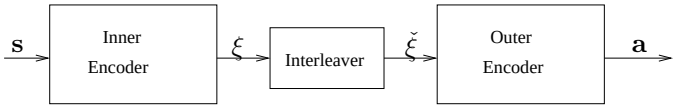

Fig. 1. The encoder in a serially concatenated turbo system.

While [10] also considers the possibility of using turbo methods for synchronization, there are a number of significant differences between that development and the one here. First of all, we consider fractionally spaced (twice oversampled) receivers, and give our description for an arbitrary pulse shape, choosing a specific pulse shape only for the simulations. Second of all, we do not quantize the timing offset, and, more significantly, we consider the timing offset to be fixed over the length of one block. Furthermore, we use the EM algorithm to determine the timing offset instead of a PLL. Also, we consider only the synchronization problem here, not joint synchronization and equalization. We compare the BER performance of the scheme with that of other synchronizers proposed in the literature by verifying its performance relative to an optimal Cramer Rao lower bound achieving estimator.

To provide a justification for our technique, we will remind the reader of the general turbo principle in Section II. After introducing our notation and the system model in Section III, we explain our scheme and how it relates to previously investigated synchronization techniques in Section IV. Section V discusses simulations which show how the scheme performs relative to other strategies.

II. The Turbo Principle for Serial Concatenation

A heuristic explanation of turbo principle for serial concatenation is as follows. To encode data to protect its reliability, we can use serially concatenated codes separated by interleavers. We use the word "code" loosely here, in that it can refer to both error control codes, such as error correction convolutional codes, as well as the convolutional structure of the model of an asynchronous pulse shaped, multipath, or markovian channel. Due to the interleavers in between the codes, the relationship between the original transmitted data and the received data is a complex one with a long memory, and an optimal receiver algorithm (e.g., ML or MAP by brute force) would

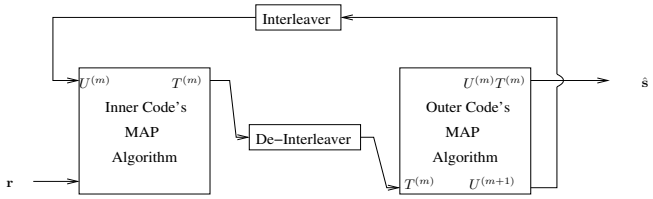

Fig. 2. A turbo decoder.

be computationally prohibitive. However, it is this complex structure which makes the overall "effective" code between the user's data and the received data robust against errors due to channel noise. Thus, we settle on a sub-optimal receiver, which implements locally optimum symbol by symbol MAP decoders for each of the constituent codes. Because the locally optimum decoders operate on a symbol by symbol basis, to handle the complex interactions caused by the interleavers we can just interleave and de-interleave the individual symbol probabilities/ extrinsic information values. The decoders exchange EI for their individual input and output symbols, each one using appropriately interleaved/deinterleaved EI values from the other decoders as prior probabilities in determining their own pseudo APPs and EI values. The structure iterates, and although a general convergence proof is yet to be established, practice and simulations suggest that it converges to good estimates of the transmitted signal.

Figure 2 illustrates this idea for the case of two serially concatenated codes connected as shown in Figure 1. For the following discussion, we will parallel a development in [11]. Assume we observe the outputs of this encoder through an additive white gaussian noise channel

$$\mathbf{r} = \mathbf{a} + \mathbf{n}$$

where $\mathbf{r}$ is a vector of our real valued observations, $\mathbf{a}$ is the antipodal (BPSK) output of the innner encoder, and $\mathbf{n}$ is a vector of independently and identically distributed zero mean gaussian random variables with variance N_0. The inner decoder determines pseudo-APPs for each of the symbols of $\check{\xi}$ using the interleaved EI values from the outer decoder as priors in a MAP algorithm for the inner code. To clarify what this means, suppose that there were just the inner encoder. Also suppose that the user of the communication system had picked the elements in the vector $\check{\xi}$ independently from symbol to symbol, so that the probability density $P(\check{\xi}) = P(\xi_1)\cdots P(\xi_N)$, where ξ_i denotes the ith element in the vector ξ. Then the locally optimal MAP detector for the inner encoder would calculate the probability ratios

$$\frac{P(\xi_i = 1|\mathbf{r})}{P(\xi_i = 0|\mathbf{r})} = \frac{\sum_{\xi|\xi_i=1} P(\mathbf{r}|\xi)P(\xi)}{\sum_{\xi|\xi_i=0} P(\mathbf{r}|\xi)P(\xi)}$$

factoring out $P(\xi_i = 1)$ in the numerator and $P(\xi_i = 0)$ in the denominator gives

$$\frac{P(\xi_i = 1|\mathbf{r})}{P(\xi_i = 0|\mathbf{r})} = \underbrace{\frac{P(\xi_i = 1)}{P(\xi_i = 0)}}_{\frac{U_i(1)}{U_i(0)}} \underbrace{\frac{\sum_{\xi|\xi_i=1} P(\mathbf{r}|\xi)\prod_{j\neq i} P(\xi_j)}{\sum_{\xi|\xi_i=0} P(\mathbf{r}|\xi)\prod_{j\neq i} P(\xi_j)}}_{\frac{T_i(1)}{T_i(0)}} \quad (1)$$

We call the first term $U_i(1)/U_i(0)$ the intrinsic information and pseudo prior probability ratio for this code interchangeably. The second term $T_i(1)/T_i(0)$ is called the extrinsic information EI. Similarly, suppose there was only the inner decoder, and we were given some prior probabilities for a sequence ξ. Suppose we learned that ξ had come from the output of the outer encoder. Call this event $\mathcal{B}$. Then, we can calculate a posteriori probabilities given this new information, namely

$$\frac{P(\xi_i = 1|\mathcal{B})}{P(\xi_i = 0|\mathcal{B})} = \frac{\sum_{\xi:\xi_i=1} P(\mathcal{B}|\xi)P(\xi)}{\sum_{\xi:\xi_i=0} P(\mathcal{B}|\xi)P(\xi)} \quad (2)$$

Now denote the set of all possible codewords output from the outer coder by $\mathcal{C}$. Then

$$P(\mathcal{B}|\xi) = \begin{cases} 1 & \xi \in \mathcal{C} \\ 0 & \text{otherwise} \end{cases} := \phi(\xi)$$

then we can rewrite (2) as

$$\frac{P(\xi_i = 1|\mathcal{B})}{P(\xi_i = 0|\mathcal{B})} = \frac{\sum_{\xi:\xi_i=1} \phi(\xi)P(\xi)}{\sum_{\xi:\xi_i=0} \phi(\xi)P(\xi)} \quad (3)$$

A turbo receiver uses the extrinsic information values for the symbols from the inner decoder as prior probabilities for the outer decoder. That is, we replace $P(\xi)$ in 3 with $T(\xi) = T_1(\xi_1)\cdots T_N(\xi_N)$ from 1 to get

$$\frac{P(\xi_i = 1|\mathcal{B})}{P(\xi_i = 0|\mathcal{B})} = \frac{T_i(1)}{T_i(0)} \underbrace{\frac{\sum_{\xi:\xi_i=1} \phi(\xi)\prod_{j\neq i} T_j(\xi_j)}{\sum_{\xi:\xi_i=0} \phi(\xi)\prod_{j\neq i} T_j(\xi_j)}}_{\rightarrow \frac{U_i(1)}{U_i(0)}} \quad (4)$$

Where we replace the previous value of U_i in the first decoder with the underbraced term in (4). The structure repeats this iteration, multiplying the intrinsic information gives the pseudo APP at every step. Letting a superscript (m) indicate the current iteration number, we have

$$\frac{\sum_{\xi|\xi_i=1} P(\mathbf{r}|\xi)\prod_j P(\xi_j)}{\sum_{\xi|\xi_i=0} P(\mathbf{r}|\xi)\prod_j P(\xi_j)} = \frac{T_i^{(m)}(1)}{T_i^{(m)}(0)}\frac{U_i^{(m+1)}(1)}{U_i^{(m+1)}(0)}$$

$$\frac{\sum_{\xi:\xi_i=1} \phi(\xi)\prod_j T_j(\xi_j)}{\sum_{\xi:\xi_i=0} \phi(\xi)\prod_j T_j(\xi_j)} = \frac{T_i^{(m)}(1)}{T_i^{(m)}(0)}\frac{U_i^{(m+1)}(1)}{U_i^{(m+1)}(0)}$$

After these iterations have converged to a constant $T^{(m)}(\xi)U^{(m)}(\xi)$, we calculate the pseudo APPs for the user's transmitted signal $\mathbf{s}$ with

$$\frac{P(s_i = 1|\mathbf{r})}{P(s_i = 1|\mathbf{r})} \approx \frac{\sum_{\mathbf{s}:s_i=1} T^{(m)}(\xi(\mathbf{s}))U^{(m)}(\xi(\mathbf{s}))}{\sum_{\mathbf{s}:s_i=0} T^{(m)}(\xi(\mathbf{s}))U^{(m)}(\xi(\mathbf{s}))}$$

A key component of the turbo structure is the computationally efficient way of implementing these equations, described in [12].

When the encoder used at the transmitter is a normal convolutional code (ie, not a turbo code) the turbo synchronizer to be discussed in this paper will take the form described above, with the outer encoder being the error control code used at the transmitter, and the inner code the intersymbol interference channel caused by the delayed pulse-shape in between the transmitter and receiver. If a turbo error control code is used to encode the sequence, we will have two of the decoders described above. We will discuss this case in more detail in the section on turbo synchronization.

III. System Model

Our model for the communications system is shown in Figure 3. For our examples, we are focusing on a sample timing synchronizer. Binary data to be transmitted, $\mathbf{s} \in \{\pm 1\}^n$, is coded with a convolutional code to create the coded sequence $\mathbf{a}$, which is interleaved using a random interleaver. The interleaved sequence is then antipodally modulated on a square root raised cosine pulse with roll off parameter $.25$, and passed through a noisy channel with a delay and sampled to create the received signal $\mathbf{r}$. If the sampling is done at the baud rate, this yields an effective overall discrete time equivalent model

$$\mathbf{r} = \mathbf{HGa} + \mathbf{n}$$

where

$$\mathbf{H} = \begin{pmatrix} h(\tau) & & \\ \vdots & \ddots & \\ h(L_h T + \tau) & & h(\tau) \\ & \ddots & \vdots \\ & & h(L_h T + \tau) \end{pmatrix}$$

is a convolution matrix created from delayed samples of the pulse shape $h(t)$, $\mathbf{G}$ is an interleaving matrix (obtained by permuting the columns of an identity matrix), and $\mathbf{n}$ is i.i.d. Gaussian noise. It is also possible to oversample the signal, say by sampling it at twice the baud rate. This is often desirable because it gives sufficient statistics for the transmitted symbols while baud spaced samples do not when the pulse shape has bandwidth larger than the symbol rate, T. In this case of oversampling by two, we can write the received signal as

$$\mathbf{r} = \mathbf{H}_1\mathbf{Ga} + \mathbf{H}_2\mathbf{Ga} + \mathbf{n}$$

where

$$\mathbf{H}_1 = \begin{pmatrix} h(\tau) & & \\ \vdots & \ddots & \\ h(L_h T + \tau) & & h(\tau) \\ & \ddots & \vdots \\ & & h(L_h T + \tau) \end{pmatrix}$$

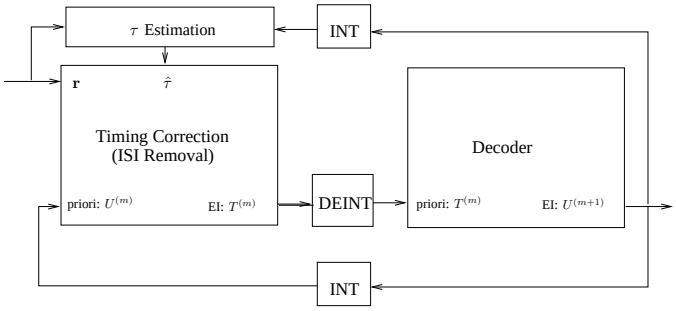

Fig. 4. A receiver employing the turbo principle to perform synchronization.

and

$$\mathbf{H}_2 = \begin{pmatrix} h(\tau + \frac{T}{2}) & & \\ \vdots & \ddots & \\ h(L_h T + \frac{T}{2} + \tau) & & h(\tau + \frac{T}{2}) \\ & \ddots & \vdots \\ & & h(L_h T + \frac{T}{2} + \tau) \end{pmatrix}$$

are convolution matrices obtained from the even and odd $T/2$ spaced samples of the delayed pulse shape respectively.

IV. Turbo Synchronization

A diagram of the proposed scheme for joint synchronization and decoding is shown in Figure 4. The receiver consists of three main blocks: an estimator for the baud timing offset, a timing correction mechanism, and a decoder.

The timing offset estimator uses the expectation maximization scheme described in [7] to form estimates, $\hat{\tau}^{(m)}$, of the baud fractional delay between the transmitter and receiver. It calculates

$$\hat{\tau}^{(m+1)} = \arg\max_{\tau} \mathcal{Q}\left(\tau, \hat{\tau}^{(m)}\right) \tag{5}$$

where

$$\begin{aligned} \mathcal{Q}\left(\tau, \hat{\tau}^{(m)}\right) &= \textstyle\int_{\mathbf{z}} p(\mathbf{a}|\mathbf{r}, \hat{\tau}^{(m)}) \ln p(\mathbf{r}|\mathbf{a}, \tau) \\ &\approx \textstyle\sum_{\mathbf{a}} T^{(m)}(\mathbf{a}) U^{(m)}(\mathbf{a}) \ln p(\mathbf{r}|\mathbf{a}, \tau) \end{aligned} \tag{6}$$

$$p(\mathbf{r}|\mathbf{a}, \tau) = f_{\mathbf{n}}\left(r - \mathbf{H}(\tau)\mathbf{a}\right) = \exp\left(-\frac{\|\mathbf{r} - \mathbf{H}(\tau)\mathbf{a}\|^2}{N_0}\right) \tag{7}$$

and $\mathbf{z} = [\mathbf{r}, \mathbf{a}]$. Here, $\mathbf{r}$ refers to the received signal model from Section III, and, likewise, $\mathbf{a}$ refers to the coded symbols.

As described in section II, the timing offset correction portion of the synchronizer calculates $T^{(m)}(\mathbf{a})$, the EI values for the coded bits, [13] [14], using the EI values, $U^{(m)}$ from the decoder's previous iteration as prior probabilites (or, for the first step, $m = 0$, an initialization of $U^{(0)}$). It does this using the turbo iteration:

$$\frac{T_i^{(m)}(1)}{T_i^{(m)}(0)} = \frac{\sum_{\mathbf{a}|a_i=1} p(\mathbf{r}|\mathbf{a}, \hat{\tau}^{(m)}) \prod_{j \neq i} U_j^{(m)}(a_j)}{\sum_{\mathbf{a}|a_i=0} p(\mathbf{r}|\mathbf{a}, \hat{\tau}^{(m)}) \prod_{j \neq i} U_j^{(m)}(a_j)} \tag{8}$$

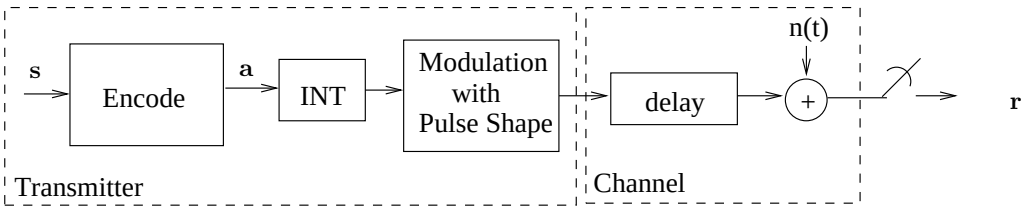

Fig. 3. The model for the received signal.

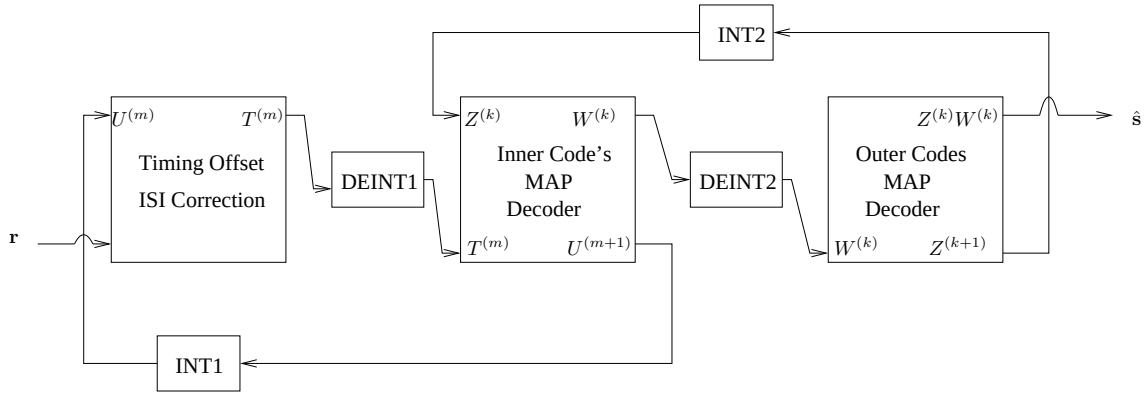

Fig. 5. A turbo synchronizer for a system using a turbo error control code.

The decoder then uses the EI values from the synchronizer as a priori probabilities for the coded bits. If the original code was not a turbo code, the decoder uses the following iteration:

$$\frac{U_i^{(m+1)}(1)}{U_i^{(m+1)}(0)} = \frac{\sum_{\mathbf{a}|a_i=1} \phi(\mathbf{a}) \prod_{j\neq i} T_j^{(m)}(a_j)}{\sum_{\mathbf{a}|a_i=0} \phi(\mathbf{a}) \prod_{j\neq i} T_j^{(m)}(a_j)} \tag{9}$$

where

$$\phi(\mathbf{a}) = \begin{cases} 1 & \mathbf{a} \in \mathcal{C} \\ 0 & \text{o.w.} \end{cases}$$

and $\mathcal{C}$ is the codebook (the set of all possible encoded sequences) used in the encoder.

If, on the other hand, the original code was a serially concatenated convolutional turbo code, the decoder is a turbo-decoder. Figure 5 depicts the turbo synchronization scheme for this case. Denoting the extrinsic information values of the input to the inner convolutional code, ξ, for the inner and outer convolutional error correcting code as W and Z, respectively, we have

$$\frac{W_i^{(k)}(1)}{W_i^{(k)}(0)} = \frac{\sum_{\xi|\xi_i=1} T^{(m)}(\mathbf{a}(\xi)) \prod_{j\neq i} Z_j^{(k)}(a_j)}{\sum_{\xi|\xi_i=0} T^{(m)}(\mathbf{a}(\xi)) \prod_{j\neq i} Z_j^{(k)}(a_j)} \tag{10}$$

$$\frac{Z_i^{(k+1)}(1)}{Z_i^{(k+1)}(0)} = \frac{\sum_{\xi|\xi_i=1} \psi(\xi) \prod_{j\neq i} W_j^{(k)}(\xi_j)}{\sum_{\xi|\xi_i=0} \psi(\xi) \prod_{j\neq i} W_j^{(k)}(\xi_j)} \tag{11}$$

where

$$\psi(\xi) = \begin{cases} 1 & \xi \in \mathcal{D} \\ 0 & \text{otherwise} \end{cases}$$

and $\mathcal{D}$ is the set of all possible codewords that could be output from the outer encoder. For each cycle of the turbo synchronizer, we iterate the turbo-decoder, (10) and (11), by incrementing k until its pseudo APPs converge at step $k = K$. Then, we use the converged APPs from (10) and (11), which are $U_i^{(m)} T_i^{(m)}$

$$\frac{U_i^{(m+1)}(1)}{U_i^{(m+1)}(0)} = \frac{\sum_{\xi|a_i=1} Z_i^{(K)}(\xi) W_i^{(K)}(\xi) \prod_{j\neq i} T_j(a_j(\xi))}{\sum_{\xi|a_i=1} Z_i^{(K)}(\xi) W_i^{(K)}(\xi) \prod_{j\neq i} T_j(a_j(\xi))} \tag{12}$$

to calculate the next set of extrinsic information values. These are then used by the timing offset estimation (5) and the timing offset correction (8) units, and the structure iterates (m increments).

As is consistent with the practical use of turbo methods, one uses a computationally efficient algorithm, [12], to compute (8)-(12). In fact, there are a number of variants of this algorithm, that have an even lower computational complexity. These considerations become important when implementing turbo devices in practice. Our focus here, though, is on the form of the algorithm, not its implementation. For a good development of several variants of the symbol by symbol MAP algorithm, and the tricks involved in implementing them in practice, one can look to [15].

V. Implementation and Simulations

To verify the performance of our turbo synchronization scheme, we simulated a communication system transmitting BPSK symbols with a square root raised cosine pulse shape with roll-off parameter .25 in MATLAB and C. Figures 6 and 7 show the results of simulations for a rate $1/2$ convolutional and rate $1/3$ turbo serially concatenated convolutional error control code, respectively, and Table I explains the abbreviations used in the figures.

TABLE I

ABBREVIATIONS USED IN THE FIGURES.

abbreviation	explanation
TE	Turbo synchronization scheme described in this paper.
IE	Performance of an ideal estimator acheiving the CRLB.
PS	Performance of a receiver with perfect synchronization
N	Length of original data sequence $\mathbf{s}$ per block.
σ	Variance of initial timing offset.
SNR	Signal to noise power ratio.
BER	Bit Error Rate.

Although it is known that one can design good interleavers for turbo codes (see [16] for example) we used a randomly chosen interleaver for ease of implementation. Thus, the performance shown is averaged over several possible interleaver choices.

We modelled the timing offset τ between the transmitter and receiver as a zero mean gaussian random variable with variance σ^2. We simulated two variances, $\sigma = .001$ and $\sigma = .01$, in order to verify that the algorithm worked just as well for small timing offsets as it did for large ones. Indeed, the comparable performance between the two subplots in Figures 6 and 7 suggest that this is the case.

To perform the minimization in (5), we used the golden section search routine outlined in [17] after initializing with the minimum bracketing routine found therein as well.

We wished to verify that the proposed turbo-synchronizer would perform well with respect to other timing offset estimation schemes. To do this, we simulated the performance of an estimator that achieves the Cramer Rao lower bound (as detailed in [2]). Furthermore, we wanted to determine what sort of loss of performance the scheme has with respect to a perfectly synchronized system. As can be determined from the closeness of the lines in Figures 6 and 7, all of these scenarios have very similar performance. Thus, turbo-estimation successfully achieves a performance near that of an ideal estimator, which achieves performance near that of a perfectly synchronized system.

We investigated using different block lengths for the turbo iteration. We expected that a longer block length would yield a higher performance. This extra performance comes at a price, however, since we have modelled the timing offset as being constant over a block. For a longer block length, then, the fastest timing offset change that the system can handle decreases. Figures 6 and 7 suggest that this is the case, although only slightly so, and for high SNR. Also, as expected, the system using the lower rate turbo code achieved a higher performance (lower BER) as the SNR increased than the system with higher rate convolutional code. The important thing to note, however, is that both systems attained performance close to that of perfect synchronization in all of the scenarios investigated.

VI. CONCLUSION

After giving a brief description of the turbo principle for serially concatenated codes, we developed a joint synchronization and decoding algorithm motivated by a turbo framework. We determined the algorithm for both the case of a single convolutional error correction code, as well as a serially concatenated convolutional turbo code. Simulations indicated as suspected that the turbo coded system with the lower rate had better BER performance. By changing the variance of the actual timing offset that the receiver had to estimate, we were able to determine that the algorithm appears to be roughly equally capable to estimate all timing offsets. Furthermore, two different block lengths (and associated interleaver lengths) were tried, $N = 512$ and $N = 256$. The receiver appeared to perform only slightly better for the longer block length, and the two had very similar performance. The algorithm was shown via the simulations to have performance to both an ideal estimator attaining the Cramer Rao lower bound, and a system employing perfect synchronization.

ACKNOWLEDGEMENTS

This work was supported in part by Applied Signal Technology and NSF Grant CCR-0310023.

REFERENCES

[1] B. Mielczarek and A. Svensson, "Post-decoding timing synchronization of turbo codes on awgn channels," in *Proceedings VTC 2000*, vol. 2, May 2000, pp. 1265–1270.

[2] ——, "Joint synchronization and decoding of turbo codes on awgn channels," in *Proceedings VTC 1999*, vol. 3, May 1999, pp. 1886–1890.

[3] ——, "Improved map decoders for turbo codes with non-perfect timing and phase synchronization," in *Proceedings VTC 1999*, vol. 3, Sept. 1999, pp. 1590–1594.

[4] ——, "Timing error recovery in turbo-coded systems on awgn channels," *IEEE Trans. Commun.*, vol. 50, pp. 1584–1592, Oct. 2002.

[5] J. Sun and M. Valenti, "Synchronization of turbo codes based on online statistics," in *Proceedings ICC 2003*, vol. 4, June 2003, pp. 2733–2737.

[6] L. Lu and S. G. Wilson, "Synchronization of turbo coded modulation systems at low snr," in *Proceedings ICC 1998*, vol. 1, June 1998, pp. 428–432.

[7] N. Noels, C. Herzet, A. Dejonghe, V. Lottici, H. Steendam, M. Moeneclaey, M. Luise, and L. Vandendorpe, "Turbo synchronization: an em algorithm interpretation," in *Proceedings ICC 2003*, vol. 4, 2003, pp. 2933–2937.

[8] J. M. Wozencraft and I. M. Jacobs, *Principles of Communication Engineering*. Wiley, 1965.

[9] J. R. Barry, E. A. Lee, and D. G. Messerschmitt, *Digital Communication, 3rd. ed.* Kluwer Academic Publishers, 2004.

[10] J. R. Barry, A. Kavčić, S. W. McLaughlin, A. Nayak, and W. Zeng, "Iterative timing recovery," *IEEE Signal Processing Magazine*, pp. 89–102, January 2004.

[11] P. A. Regalia, "Blind turbo equalization using the constant modulus algorithm," in *13th IFAC Symp. System Identification (SYSID 2003)*, August 2003, pp. 584–589.

[12] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Inform. Theory*, pp. 284–287, Mar. 1974.

[13] S. Benedetto and G. Montorsi, "Iterative decoding of serially concatenated convolutional codes," *Electronics Letters*, vol. 32, pp. 1186–1187, June 1996.

[14] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near shannon limit error-correcting coding and decoding: Turbo-codes," in *Proceedings ICC 1993*, 1993, pp. 1064–1070.

[15] A. Giulietti, B. Bougard, and L. V. der Perre, *Turbo Codes. Desireable and Designable*. Kluwer Academic Publishers, 2004.

[16] C. Heegard and S. B. Wicker, *Turbo coding*. Kluwer Academic Publishers, 1999.

[17] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C++: The Art of Scientific Computing, 2nd. Ed.* Cambridge University Press, 2002.

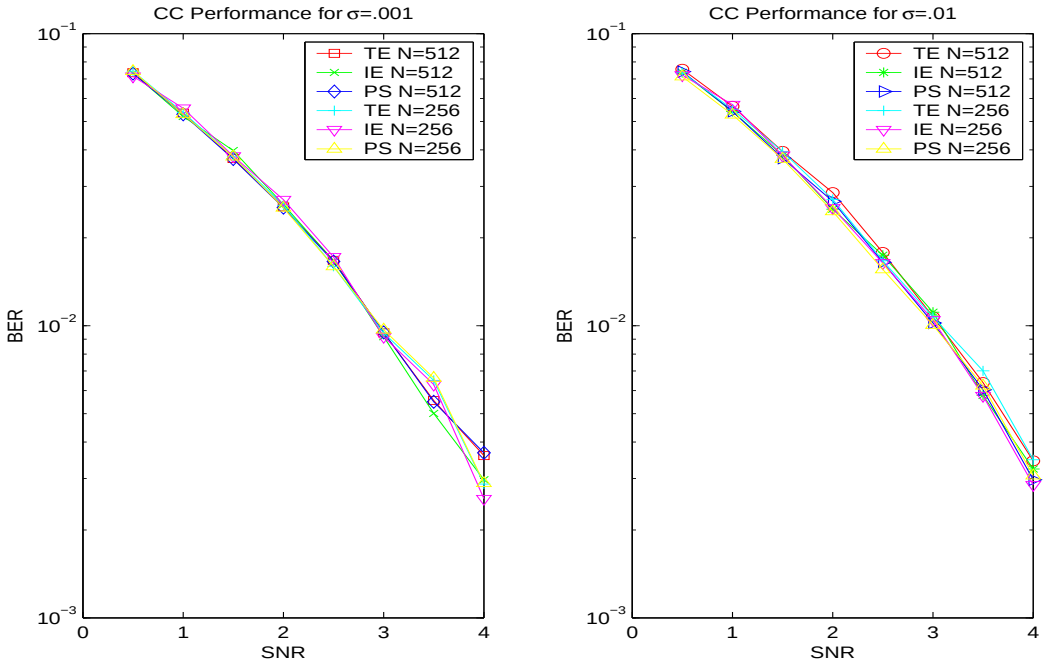

Fig. 6. Performance of turbo synchronization with a single rate 1/2 convolutional code.

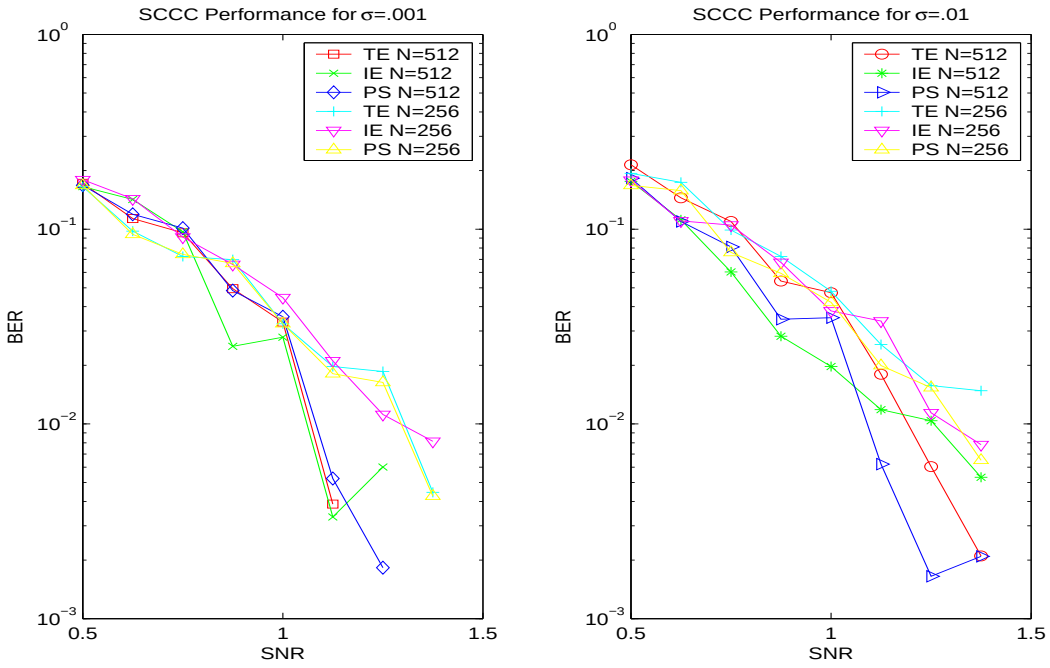

Fig. 7. Performance of turbo synchronization with a serially concatenated convolutional turbo code (rate 1/3). Note that the BER quickly drops out of the resolution of the simulations (200 monte carlo runs with the same range of SNRs as in Figure 6).