Achieving Code Synchronization In Direct-Sequence Spread Spectrum System Applications Using Recurrent Neural Networks

Vahid Forutan
Spread Spectrum Communication Research Laboratory
Department of Electrical Engineering
Polytechnic of Tehran, Iran

ABSTRACT

Code synchronization is the most important matter of concern in all applications involving direct sequence spread spectrum system, such as CDMA. Conventional synchronization techniques are primarily based on the auto/cross correlation properties of the spreading codes. The objective of the present paper is to propose a new method for fast and reliable code synchronization based on the abilities of recurrent neural networks in pattern association problems. One major benefit of this method is the independence of its performance on correlation characteristics of the spreading codes. The feasibility of the proposed scheme in achieving synchronization between transmitter and receiver in the presence of noise and interference are considered and the results are presented.

INTRODUCTION

Synchronization of spreading signal (code) between transmitter and receiver in any type of direct-sequence spread spectrum system plays the principal role in the successful extraction of the information being hidden in the overall transmitted signal. By synchronization we mean achieving the coincidence between the initialization point of the transmitted code and the locally generated one at the receiver.

For an spreading code of length N chips, there exist N distinct phases and in fact each phase can be viewed as a *pattern*. With this regard, the overall process of code synchronization becomes a classical pattern association problem and it is the objective of the present paper to develop a new method to solve it using recurrent neural networks.

In practical communication systems, because of the unknown time delay between transmitter and receiver, there exists uncertainty about the correct phase of the received code. The receiver should be capable of generating a code exactly in the same phase as the received one in the presence of channel noise (and also present interference from the other users e.g. in DS-CDMA) in order to extract the information (data signal).

Traditional code synchronization process is performed via *correlating* the received signal with the locally generated code at the receiver and then monitoring the output of the correlator until a peak is observed. This peak occurs when the correlator inputs are synchronized. With respect to the structure of the receiver, the synchronization process can be carried out through some fully developed techniques such as serial search, parallel search and the method of matched filtering[1].

In some papers the implementation of the synchronization process using artificial neural networks (ANN) has been reported[2]-[6]. For example in [3] a method called RANN which stands for "rapid acquisition using neural network" is proposed and is based on the training of ANN with all possible phases of the code and has been proven to have high synchronization probability and fast synchronization time. But its restriction is that it can be applied only in situations where the length of the code is short. In [2] another scheme based on multi-layer neural networks (MLNN) has been introduced which has better performance for codes with larger length and been compared with digitally implemented matched filters. At the receiver, the code of length N is passed through a tapped delay line with M taps ($M < N$) and tap spacing equal to code chip time and then the output of taps are entered into the neural network which is trained so that its output is 1 for a specific sequence of length M and is 0 for all other sequence of that length. This enables the receiver to be synchronized exactly with the transmitter when that special sequence of length M is passed through the receiver.

If a way could be found to enable the receiver to distinguish among several sequences of length M (instead of only one sequence) selected from different positions in the code, then the synchronization process will be accomplished faster and also more reliable. This is the objective of the present paper to develop a method to provide all necessities both theoretically and practically to design such a receiver.

A FEW BASICS OF RECURRENT NEURAL NETWORKS

The Hopfield neural network (HNN) which belongs to the category of recurrent networks, consists of a set of neurons and a corresponding set of unit delays, forming a multiple-loop feedback system (Figure 1). The number of feedback loops is equal to the number of neurons. Basically, the output of each neuron is fed back, via a unit delay element, to each of the other neurons in the network. The detailed description of HNN is beyond the scope of this text, here a brief description is mentioned.

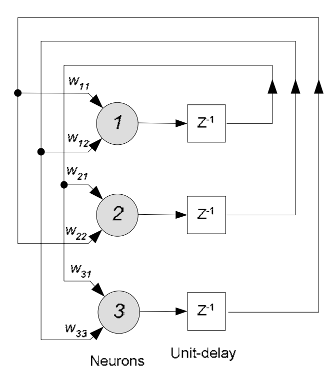

Fig. 1. A typical structure of recurrent network with $M = 3$ neurons

HNN has attracted a great deal of attention in the literature as a content-addressable memory (CAM). The primary function of a CAM is to retrieve a pattern stored in it in response to the presentation of an incomplete or noisy version of that pattern. An important property of a CAM is therefore the ability to retrieve a stored pattern, given a reasonable subset of information content of that pattern.

For HNN to be used as a CAM, let $\boldsymbol{p}_1, \boldsymbol{p}_2 \ldots, \boldsymbol{p}_P$ denote a known set of M-dimensional patterns, the synaptic weights of the network will be computed from[7]:

$$w_{ji} = \begin{cases} \dfrac{1}{M}\displaystyle\sum_{k=1}^{P} p_{k,j}\, p_{k,i}, & j \neq i \\ 0 & j = i \end{cases}$$

Once the synaptic weights are computed, they will be kept fixed. This is called *learning step* or *storage phase*. After that, let $\boldsymbol{p}_{probe}$ denote an unknown (and noisy) M-dimensional input vector presented to the network, the CAM algorithm is initialized by setting $x_j(0) = p_{j,probe}$ where $x_j(0)$ is the state of neuron j at time $n = 0$, Updating the elements of M-dimensional *state vector* $\boldsymbol{x}(n)$ randomly and one at a time according to the rule:

$$x_j(n+1) = \operatorname{sgn}\left[\sum_{i=1}^{M} w_{ji} x_i(n) + b_j\right],\ j = 1,2,\ldots,M$$

and repeating the iteration until the state vector $\boldsymbol{x}$ will remain unchanged. Letting $\boldsymbol{x}_{fixed}$ denote the fixedpoint (stable state) computed at the end of previous step, the resulting network output will be $\boldsymbol{y} = \boldsymbol{x}_{fixed}$. For an HNN possessing the total number of M neurons, the number of patterns P which can be stored and retrieved successfully is obtained from[7]:

$$P = \frac{M}{2\log_e M}$$

STRUCTURE OF THE PROPOSED SYNCHRONIZATION SCHEME

We assume that the received signal $r(t)$ be written in the general baseband form (i.e. asynchronous case in CDMA) as [8]:

$$r(t) = \sum_{k=1}^{K}\sum_{i=-M}^{M} A_k b_k[i] s_k(t - iT - \tau_k) + \sigma n(t)$$

with observation interval to be $t \in [-MT, MT]$. In the above equations, it is assumed that the total number of K users are present and the received signal $r(t)$ in each *epoch* (one bit time T) consists of desired signal $A_1 b_1 s_1(t-\tau_1)$ plus interference term $\sum_{k=2}^{K} A_k b_k s_k(t-\tau_k)$ and the gaussian noise term $n(t)$ with power σ. A_k , b_k and $s_k(t)$ are the k^{th} user signal amplitude, data bit and spreading code respectively and $b_k \in \{-1,1\}$. T is the bit time and is equal to NT_c (T_c is the code chip time). $b_k[i]$ is the k^{th} user data bit in the i^{th} epoch and τ_k's represent the random time offsets that model the lack of alignment of the bit epochs at the receiver[8].

In the non-CDMA case (i.e. a general spread spectrum communication system with only one authorized user), the above equation are still valid by assuming that the interference term is ommitted ($K = 1$) and only (strong) noise is present. The input signal is sampled every T_c seconds and samples will be processed by HNN.

In the case of a code with short length, the patterns to be stored in the network (i.e. $\boldsymbol{p}_1, \ldots, \boldsymbol{p}_P$ introduced in the previous section) are the P distinct phases of the desired user code chosen from all N possible phases of it (i.e. $M = N$, with M and N to be the number of the neurons and the length of the code respectively). To be specific, $\boldsymbol{p}_1$ stands for the first phase of the code starting from the first chip of the it and continuing N–1 chips ahead, $\boldsymbol{p}_2$ stands for the second phase of the code starting from the second chip of it, continuing $N-1$ chips ahead and so on until obtaining $\boldsymbol{p}_P$. This leads to the total number of P patterns each with length N.

The process of synchronization at the receiver starts with applying the first received N samples to HNN. Due to the unknown time delay between transmitter and receiver, the process of sampling of the received signal is indeed a *blind* process in which these N samples can belong to every N possible distinct phases of the code and this is the task of HNN to distinguish the correct phase in which these N samples belong to. Once the response of the network to the input N samples was computed (pattern retrieval), this retrieved pattern is applied to the receiver code generator which outputs a phase-aligned code with respect to the received signal. When the network fails to respond correctly, the next received N samples are applied to it and this process is repeated until the correct phase is detected. The structure of the receiver is depicted in figure 2.

In the case of asynchronous CDMA, it is possible for the the receiver to start to sample the received signal in a manner that nearly one half of the samples belong to one data bit $b[i]$ (i^{th} epoch) and the other half belong to $b[i+1]$ ($i+1^{\text{th}}$ epoch) which is not necessarily in sign agreement with $b[i]$. It is clear that in this case the resulting N samples do not resemble in any respect with those patterns stored in HNN, hence the neural network fails to associate the correct phase with the input N samples presented to it and outputs a wrong response. In order to enhance the ability of HNN to cope with such a problem, it is possible to consider these N samples as another pattern to be stored in the network.

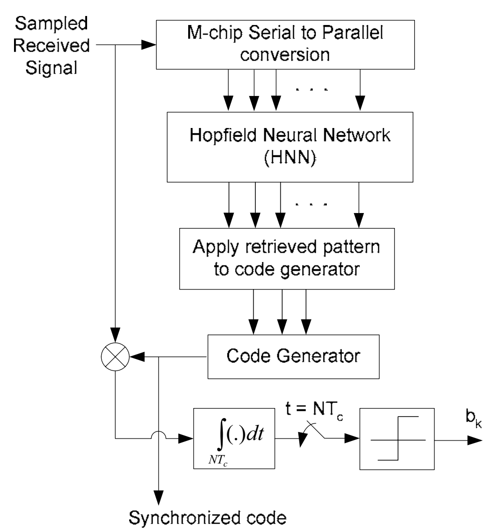

Fig. 2. Structure of the receiver.

As the length of the code grows exponentially, which is common in practical spread spectrum systems, implementing the receiver based on the procedure previously described becomes practically formidable with respect to both mean synchronization time and the number of neurons required to construct the neural network, leading to a computationally extravagant burden on the receiver. Hence another consistent strategy suitable for applications involving longer codes is proposed.

Considering again the scenario described previously where the full length versions of the code phases was chosen as the patterns to be stored in the network ($\boldsymbol{p}_1$, $\boldsymbol{p}_2$, …), but now the code with length N is subdivided into (much) smaller sequences each of length M and then a number of only P subdivisions selected from different positions in the code are applied as the patterns to the network in the storage phase. the appropriate choices for M and P are also dependent on the restrictions dictated by the receiver hardware limitations and also the time required to perform the synchronization process. Selecting larger values for M (hence for P) leads to more probable occurrence of synchronization but longer time for the network to respond. With this regard, in the selection of the value for M, trade off must be considered between more reliable occurrence of synchronization and the synchronization time. The structure of the receiver in this case is the same as shown in figure 2.

In the case of a code with considerably large length, it is reasonable to choose M-chip sequences which possess larger Hamming distances with respect to the others. A method to achieve this goal is described in [3].

RESULTS AND SIMULATION

To simulate the proposed code synchronization scheme, two situations are considered;

1) A code with length $N = 2^6 - 1 = 63$ belonging to the family of Gold codes (used extensively in CDMA system) and with the total number of $K = 16$ and $K = 32$ active users.
2) An m-sequence code with the length of $N = 2^{10} - 1 = 1023$ generated from a 10-stage linear feedback shift register (LFSR).

In both situations, the codes were multiplied by binary data signal obtained from a random source and then corrupted by gaussian noise and interference with variable powers.

For the purpose of evaluating the performance of the receiver, several criteria such as probability of correct phase detection P_d, probability of false alarm P_{fa} and the mean value for synchronization time T_s and also the bit error rate (BER) for the case of CDMA, were obtained via simulation.

In the case of code with length $N = 63$, the value of M (number of neurons) was chosen to be 63 (i.e. full length patterns) and by considering the mentioned rule for the number of patterns that can be stored in the network, the total number of $P = 8$ patterns were selected. In other words, in the learning step of the network, a number of 8 sequences each with length 63 were selected from $N = 63$ possible code phases and stored in the network as the patterns.

For the code with length $N = 1023$, two values for M were chosen; namely 100 and 150, leading to $P = 11$ and $P = 15$ respectively (i.e. partial length patterns). These patterns were arbitrarily selected from different positions in a code of length N. Choosing larger values for M (and hence for P) leads to better performance of the network with respect to increasing P_d and decreasing P_{fa}. In one respect, choosing larger values for P leads to faster synchronization process, since in this case it becomes more likely for the input to HNN to be similar to one of the stored patterns in it, but this in fact needs more neurons to be added to HNN, leading to more time for calculations to be undertaken. Applying parallel computing can improve the speed of calculations and hence overcome this limitation.

The simulation results are depicted in figures 3 to 9. Figures 3 to 5 correspond to the case where the code length N is 1023 and it is assumed that only strong gaussian noise has corrupted data (i.e. $K = 1$) and SNR ranges from -9 dB to 9 dB. In each plot the results are shown for $M = 100$ and 150. Figures 6 to 9 correspond to the asynchronous CDMA case with a short code with length $N = 63$. In this case it is assumed that the interference is dominant relative to the gaussian noise and SIR (signal to interference ratio) ranges from the worst case -3 dB up to 15 dB. In addition to the plots of P_d, P_{fa} and T_s , the probability of bit error is also sketched and shown in figure 9.

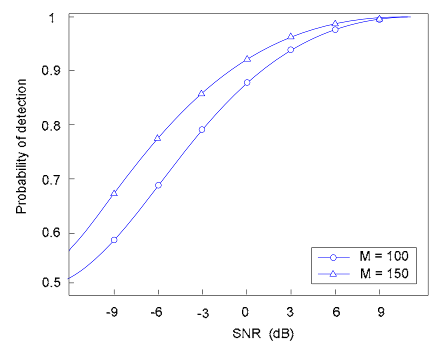

Fig. 3. Probability of Dtection P_d versus SNR, Code length $N = 1023$, M is the number of neurons, The number of users is $K = 1$.

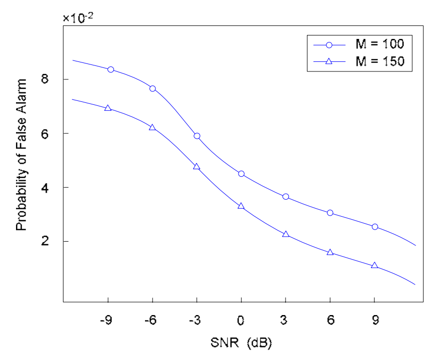

Fig. 4. Probability of false alarm P_{fa} versus SNR for $N = 1023$, M is the number of neurons, $K = 1$.

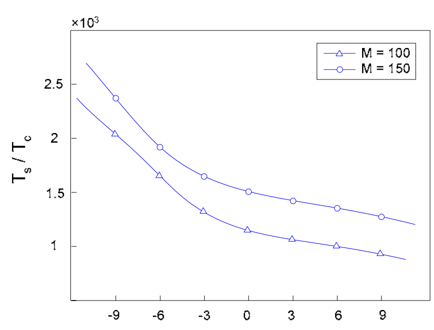

Fig. 5. Mean synchronization time T_s versus SNR for $N = 1023$, M is the number of neurons, $K = 1$.

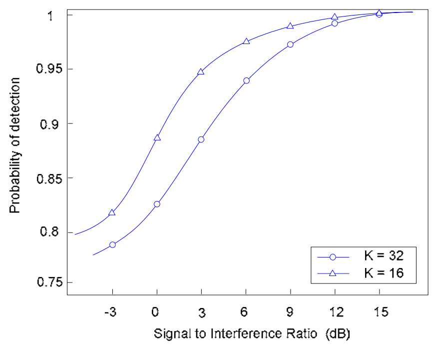

Fig. 6. P_d for Asynchronous CDMA, K is the number of users, the number of Neurons is $M = 63$ and $N = 63$.

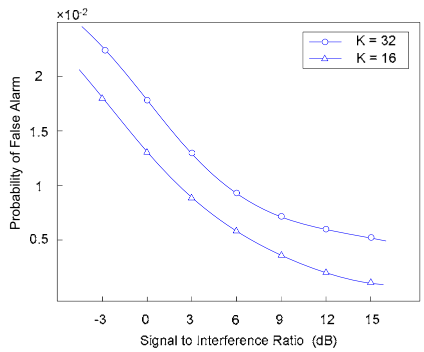

Fig. 7. P_{fa} for Asynchronous CDMA, K is the number of users, $M = 63$, $N = 63$.

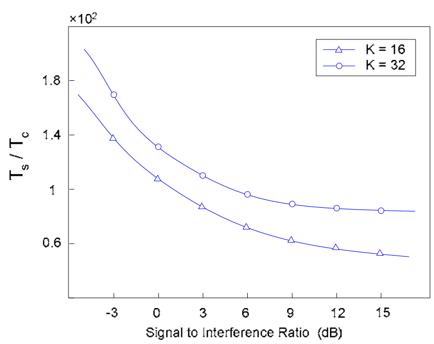

Fig. 8. T_s for Asynchronous CDMA, K is the number of users, $M = 63$, $N = 63$.

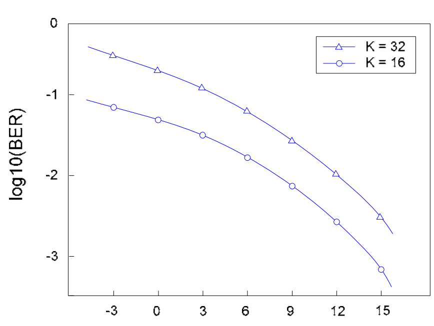

Fig. 9. Probability of Bir Error for Asynchronous CDMA, K is the number of users, $M = 63$, $N = 63$.

CONCLUSION

A new method for code synchronization in the applications involving direct sequence spread spectrum system is proposed in this text, which is based on the pattern association capability of the recurrent neural networks. Unlike the traditional code synchronization techniques such as the method of matched filtering in which the performance is dependent mainly on the auto/cross correlation characteristics of the spreading codes used, the suggested method in the present paper is completely independent of those charactersitics, hence in aplications such as DS-CDMA where the cross correlation between users codes (especially in asynchronous CDMA), makes the synchronization task difficult to achieve, the superiority of this method becomes more evident. The mean synchronization time, and also the probability of correct decision and false alarm are shown to be affected by modifying the number of patterns stored in the neural network (number of neurons). The storage phase for the receiver can be accomplished before the communication between transmitter and receiver takes place. The above factors could be the benefits of the synchronization scheme proposed in this paper at the expense of nearly more processing burden on the receiver when compared to other techniques such as matched filtering.

REFERENCES

[1] R. Peterson, R. Ziemer, D. Borth, "Introduction to Spread Spectrum Communication", Prentice Hall International Inc. 1995.

[2] S. H. Yoon, S. K. Park, "DS/CDMA PN Code Acquisition by Neural Network", IEEE 45th Vehicular Technology Conf., vol. 1, pp 276-270, July 1995.

[3] S. M. Lee, D.S. Hong, "Rapid Acquisition by Neural Networks in DS/SS Communication Systems", Proc. of 94'ICONIP (Intl. Conf. on Neural Information Processing), vol. 2, pp 993-998, Oct. 1994, Seoul Korea.

[4] J. Weng, G. Bi, "A Neural Network Based Adaptive Correlator In DS/SS Mobile Communication System", IEEE 45th Conf. on Vehicular Technology, vol. 2, pp 679-683, July 1995.

[5] S. E. El-Khamy, H. M. Abdou, "Neural Network Matched Filter Receiver for CDMA Systems", Proc. of IEEE 4th Intl. Symposium on Spread Spectrum Techniques and Applications. vol. 3, pp 1024-1028, July 1996.

[6] C. Bouder, G. Burel, "Spread spectrum Code Identification by Neural Networks", WSES publications in Systems and Control, Theory and Applications, pp. 257-262, 2000.

[7] S. Haykin, "Neural Networks, A Comprehensive Foundation", Prentice Hall International Inc. 1999.

[8] S. Verdu, "Multiuser Detection", Cambridge University press, 1998.