Low Density Parity Check Convolutional Codes Derived from Quasi-Cyclic Block Codes *

Daniel J. Costello Jr., Arvind Sridharan, and Deepak Sridhara

Department of Electrical Engineering
University of Notre Dame
Notre Dame, IN 46556, U.S.A.
email:{*costello.2, asridhar, dsridhar*} @nd.edu

R. Michael Tanner

Department of Computer Science
University of California, Santa Cruz
Santa Cruz, CA 95064, U.S.A.
email:*tanner*@cse.ucsc.edu

Dedicated to Professor Ian Blake on the occasion of his 60th birthday
Victoria, British Columbia

June 7-8, 2001

Abstract

Using algebraic techniques, one of the co-authors has designed a [155,64,20] low density parity check code based on permutation matrices. This code is quasi-cyclic by construction and hence admits a convolutional representation. A set of low density parity check convolutional codes is derived from this quasi-cyclic code and its generalizations. We investigate the performance of these convolutional codes when decoded using belief propagation on their corresponding graph representations.

*This work was supported in part by NSF Grant CCR00-75514 and NASA Grant NAG5-10503.

1 Introduction

In the past few years, excellent performance has been achieved for binary transmission over noisy channels using *turbo coding* and *low density parity check (LDPC)* codes. These codes have an inherent block structure that requires splitting the data to be transmitted into frames, which can be a disadvantage in some applications. Convolutional codes, on the other hand, have no such requirement, and hence are well suited for continuous transmission. Inspired by the pioneering work of Blake [1], on algebraic coding, we derive a class of algebraically constructed convolutional codes from a class of LDPC codes that are based on an algebraic construction given by Tanner [2] [3], and investigate the performance of these codes.

At the Recent Results Session of ISIT 2000, Tanner presented a [155,64,20] LDPC code that was designed using algebraic techniques [2]. The parity check matrix of this code is made up of a block of circulant matrices, which makes the code *quasi-cyclic*. Quasi-cyclic codes provide a link between the theories of block and convolutional codes [4] [5]. Thus, these constructions, apart from providing an algebraic approach to constructing LDPC convolutional codes, also serve to bridge the gap between the theories of LDPC block and convolutional codes. The resulting convolutional codes can be represented by a semi-infinite low density parity check matrix that retains the structure of the original parity check matrix of the block code. The low density structure of the block code, that is replicated in the convolutional code, allows for graph based iterative decoding on the convolutional code's constraint graph. An added advantage of converting the block codes into convolutional codes is the ease of encoding.

In the following section we describe the construction of such quasi-cyclic LDPC codes and show how LDPC convolutional codes are derived from them. The performance of the convolutional codes, when decoded with a continuously operating decoder, is then examined.

2 Code Construction

2.1 The Block Codes [3]

Let Z_m denote the integers from 0 to $m-1$, where m is a prime number. If addition and multiplication are defined modulo m, then Z_m is the Galois field $GF(m)$ of size m. We can then construct a (j,k) regular LDPC code, where j and k are among the prime factors of $m-1$. Let a and b denote two non-zero elements in $GF(m)$ with multiplicative orders k and j, respectively. Then the $j \times k$ matrix P (shown below) is made up of integers and its $(s,t)^{th}$ element is given by $P_{s,t} = b^{(s-1)}a^{(t-1)}$ modulo m. (Here, $1 \leq s \leq j$ and $1 \leq t \leq k$.) An LDPC code is constructed by specifying its parity check matrix H (shown below). Specifically, H is made up of a $j \times k$ block of permutation matrices – each permutation matrix being an $m \times m$ matrix obtained by circularly shifting the $m \times m$ identity

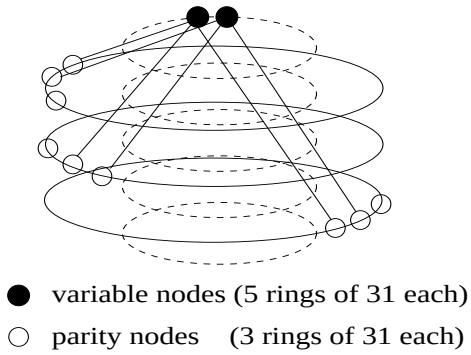

Figure 1: Tanner graph for Tanner's [155,64,20] quasi-cyclic code

matrix. The permutation matrix in position (s,t) within H is denoted as $I_{P_{s,t}}$, the permutation matrix obtained by circularly shifting the identity matrix by $P_{s,t}$ places.

$$P = \begin{bmatrix} 1 & a & a^2 & \dots & a^{k-1} \\ b & ab & a^2b & \dots & a^{k-1}b \\ \dots & \dots & \dots & \dots & \dots \\ b^{j-1} & ab^{j-1} & a^2b^{j-1} & \dots & a^{k-1}b^{j-1} \end{bmatrix}_{(j\times k)}$$

$$H = \begin{bmatrix} I_1 & I_a & I_{a^2} & \dots & I_{a^{k-1}} \\ I_b & I_{ab} & I_{a^2b} & \dots & I_{a^{k-1}b} \\ \dots & \dots & \dots & \dots & \dots \\ I_{b^{j-1}} & I_{ab^{j-1}} & I_{a^2b^{j-1}} & \dots & I_{a^{k-1}b^{j-1}} \end{bmatrix}_{(jm\times km)}$$

The code thus constructed has a binary parity check matrix that is $jm \times km$, which means the code has a rate $R \geq 1-(j/k)$. (The rate may be greater than $1-(j/k)$ because there may be a linear dependence among the rows of H thus constructed; indeed, it is easy to see that there are in fact at least $j-1$ dependent rows). By construction, it is clear that every column of H contains exactly j ones and every row contains exactly k ones, i.e., H represents a (j,k) regular LDPC code [6].

Tanner's example: A [155,64,20] quasi-cyclic code [2]:
Elements $a=2$, $b=5$ are chosen from $GF(31)$:

$$o(a) = 5, o(b) = 3.$$

The LDPC matrix

$$H = \begin{bmatrix} I_1 & I_2 & I_4 & I_8 & I_{16} \\ I_5 & I_{10} & I_{20} & I_9 & I_{18} \\ I_{25} & I_{19} & I_7 & I_{14} & I_{28} \end{bmatrix}_{(93\times 155)}$$

is thus obtained. The length of the shortest cycle (or, in other words, the girth) of the Tanner graph corresponding to this matrix is 8. A sparse Tanner graph representation having such a large girth

makes the code ideal for graph based belief propagation decoding. The elegant graph structure for this code is shown in Figure 1. The associated code has a minimum distance (d_{min}) of 20. All parity check matrices constructed in the above fashion have Tanner graphs with similar properties, making them well suited for graph based decoding. In fact, for short to moderate block lengths (typically between 1000 and 10,000 depending on the values of j and k), they outperform other regular random constructions with comparable rates and block lengths [3].

2.2 The Convolutional Codes

The LDPC code constructed above is quasi-cyclic or, in other words, a tail biting convolutional code; hence, a convolutional representation of this code can be obtained by simply unwrapping the quasi-cyclic code [4] [5]. Every circulant block of the $jm \times km$ binary parity check matrix of the regular LDPC code constructed above is expressed in polynomial form to obtain the following $j \times k$ matrix

$$H(D) = \begin{bmatrix} 1 & D^{a-1} & D^{a^2-1} & \dots & D^{a^{k-1}-1} \\ D^{b-1} & D^{ab-1} & D^{a^2b-1} & \dots & D^{a^{k-1}b-1} \\ \dots & \dots & \dots & \dots & \dots \\ D^{b^{j-1}-1} & D^{ab^{j-1}-1} & D^{a^2b^{j-1}-1} & \dots & D^{a^{k-1}b^{j-1}-1} \end{bmatrix}_{(j \times k)}.$$

This is the parity check matrix of the corresponding convolutional code in polynomial form. The convolutional code has a design rate $r = 1 - (j/k)$.

Thus, from Tanner's [155, 64, 20] code mentioned above, we obtain a rate 2/5 convolutional code with parity check and generator matrices given by

$$H(D) = \begin{bmatrix} 1 & D & D^3 & D^7 & D^{15} \\ D^4 & D^9 & D^{19} & D^8 & D^{17} \\ D^{24} & D^{18} & D^6 & D^{13} & D^{27} \end{bmatrix}_{(3 \times 5)}$$

$$G(D) = \begin{bmatrix} \frac{a_1(D)}{\Delta(D)} & \frac{a_2(D)}{\Delta(D)} & \frac{a_3(D)}{\Delta(D)} & 1 & 0 \\ \frac{b_1(D)}{\Delta(D)} & \frac{b_2(D)}{\Delta(D)} & \frac{b_3(D}{\Delta(D)} & 0 & 1 \end{bmatrix}_{(2 \times 5)},$$

where

$a_1(D) = D^4(1 + D^7 + D^{10} + D^{14} + D^{18} + D^{29})$
$a_2(D) = D^3(1 + D^3 + D^6 + D^{18} + D^{21} + D^{36})$
$a_3(D) = D^7(1 + D^4 + D^8 + D^{11} + D^{15} + D^{22})$
$b_1(D) = D^{13}(1 + D^6 + D^{14} + D^{15} + D^{23} + D^{28})$
$b_2(D) = D^{12}(1 + D^2 + D^{11} + D^{21} + D^{23} + D^{35})$
$b_3(D) = D^{21}(1 + D^3 + D^4 + D^5 + D^{10} + D^{16})$
$\Delta(D) = 1 + D^4 + D^{14} + D^{25} + D^{26} + D^{33}$.

The minimal-basic generator matrix [7] has the minimum overall constraint length among all equivalent rational and polynomial generator matrices and is thus of interest. The minimal-basic form of $G(D)$ is shown below.

$$G_{min}(D) = \begin{bmatrix} a_1(D) & a_2(D) & a_3(D) & a_4(D) & a_5(D) \\ b_1(D) & b_2(D) & b_3(D) & b_4(D) & b_5(D) \end{bmatrix}_{(2\times 5)} ,$$

where

$a_1(D) = D^6 + D^{10} + D^{14} + D^{17} + D^{18}$

$a_2(D) = D^5 + D^8 + D^9 + D^{11} + D^{13} + D^{14} + D^{15} + D^{19} + D^{21} + D^{22} + D^{23}$

$a_3(D) = D^9$

$a_4(D) = D^2 + D^{10} + D^{13} + D^{15} + D^{16} + D^{18} + D^{19} + D^{20} + D^{21} + D^{23}$

$a_5(D) = 1 + D + D^3 + D^9 + D^{10} + D^{11} + D^{12} + D^{13} + D^{15}$

$b_1(D) = D^4 + D^5 + D^6 + D^8 + D^{10} + D^{11} + D^{12} + D^{13} + D^{14} + D^{16} + D^{17}$

$b_2(D) = D^3 + D^4 + D^5 + D^6 + D^8 + D^{11} + D^{12} + D^{13} + D^{14} + D^{16} + D^{17} + D^{18} + D^{19} + D^{20}$

$b_3(D) = D^7 + D^8 + D^9 + D^{12}$

$b_4(D) = 1 + D + D^2 + D^5 + D^8 + D^9 + D^{10} + D^{11} + D^{13} + D^{14} + D^{15} + D^{16} + D^{21} + D^{22}$

$b_5(D) = 1 + D^2 + D^4 + D^7 + D^8 + D^{13} + D^{14}.$

Several quasi-cyclic LDPC block codes and their corresponding convolutional representations have been obtained following the above procedure. A few of them are listed in Table 1. The free distance (d_{free}) of the convolutional code is lower bounded by the minimum distance (d_{min}) of the quasi-cyclic code (i.e., $d_{free} \geq d_{min}$) [4] [5]. Thus, in the example above, d_{free} of the convolutional code is at least 20. We conjecture that the above convolutional code has, in fact, a d_{free} of 24. Choosing one of the inputs to the encoder $G(D)$ as the all zero sequence and the other input as the denominator polynomial, i.e., $1 + D^4 + D^{14} + D^{25} + D^{26} + D^{33}$, produces a codeword of weight 24. This, strictly speaking, provides an upper bound on the d_{free} of this convolutional code. Interestingly, this is the same as the upper bound obtained by Mackay and Davey [8] for LDPC matrices constructed by using permutation matrices, as is the case here[1]. The graph structure of the block code carries over to the convolutional representation. This, as we shall see, allows for decoding the convolutional codes efficiently using belief propagation decoding.

[1]Mackay and Davey show that (j, k) regular LDPCs constructed from permutation matrices (that commute) have a minimum distance of at most $(j + 1)!$ (which is 24 for these (3,5) LDPCs). The assertion also holds for the convolutional codes that are derived from them.

Table of codes - A few examples of convolutional codes derived from quasi-cyclic LDPC codes

Block length	design parameters		rate (conv. code)	rate (blk. code)	field size
N	j	k	r_{conv}	r_{blk}	m
155	3	5	0.4000	0.4129	31
1055	3	5	0.4000	0.4018	211
1477	5	7	0.2857	0.2884	211
1967	5	7	0.2857	0.2878	281
2105	3	5	0.4000	0.4009	421
2947	5	7	0.2857	0.2870	421
7357	3	5	0.2857	0.2863	1051

3 Decoding

The convolutional codes constructed in this fashion are found to have very large constraint lengths, which makes Viterbi or BCJR decoding impractical. Sequential decoding, although close to maximum likelihood, is computationally practical only for rates below the cut-off rate. We therefore investigate decoding these codes using belief propagation based on the graph obtained from the constraints imposed by the parity check entries [4] [9]. We now illustrate, through a simple example, the methodology used for obtaining the constraint graph of the convolutional code from its parity check matrix and the strong resemblance it bears to the corresponding graph of the original quasi-cyclic block code. Consider a $[15, 5]$ quasi-cyclic code [5] with the following parity check matrix:

$$H_{QC} = \left[\begin{array}{ccccc|ccccc|ccccc} 1&1&0&0&0&1&0&0&0&0&0&1&0&0&0 \\ 0&1&1&0&0&0&1&0&0&0&0&0&1&0&0 \\ 0&0&1&1&0&0&0&1&0&0&0&0&0&1&0 \\ 0&0&0&1&1&0&0&0&1&0&0&0&0&0&1 \\ 1&0&0&0&1&0&0&0&0&1&1&0&0&0&0 \\ \hline 0&0&1&0&0&1&1&0&0&0&1&0&0&0&0 \\ 0&0&0&1&0&0&1&1&0&0&0&1&0&0&0 \\ 0&0&0&0&1&0&0&1&1&0&0&0&1&0&0 \\ 1&0&0&0&0&0&0&0&1&1&0&0&0&1&0 \\ 0&1&0&0&0&1&0&0&0&1&0&0&0&0&1 \end{array}\right]$$

The corresponding Tanner graph for this code is shown in Figure 2. As described above, we can derive a convolutional code from this quasi-cyclic code. Here we obtain a rate 1/3 convolutional code

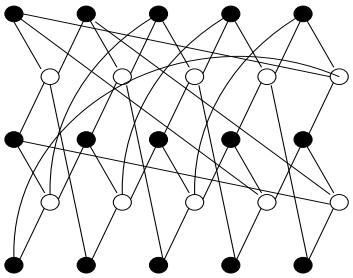

Figure 2: Tanner graph for a (15,5) quasi-cyclic code

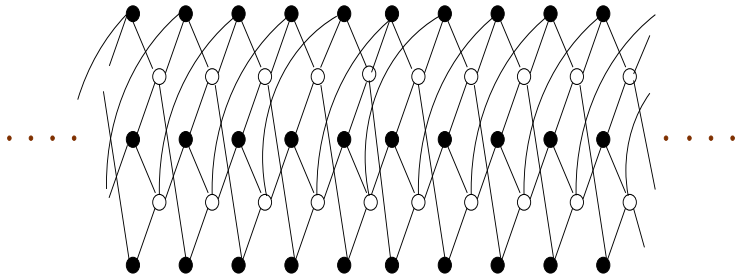

Figure 3: Tanner graph for the convolutional code derived from a (15,5) quasi-cyclic code

(The dark circles are the codebits and the empty circles are the constraints (parity checks).)

from the above quasi-cyclic block code with parity check and generator matrices given by

$$H_C(D) = \begin{bmatrix} 1+D & 1 & D \\ D^2 & 1+D & 1 \end{bmatrix}, \quad G_C(D) = \begin{bmatrix} 1+D+D^2 & 1+D+D^3 & 1 \end{bmatrix}.$$

This code is described by the graph shown in Figure 3. There is a striking similarity between the graph structures for the quasi-cyclic block code and its corresponding convolutional representation. In fact, the graph for the convolutional code is obtained by unwrapping that of the block code.

The primary reason for obtaining a convolutional representation of the block codes is the fact that we can decode these codes in a continuous fashion without any significant loss in performance. To this end, a decoder (which we refer to as the continuous decoder) that operates over a window that slides across the code-constraint graph of the convolutional code was implemented, following the approach in [10]. This allows the decoding results to be output continuously, after some initial delay. The other alternative would be to wait for the entire sequence of transmitted symbols and then decode the entire frame as an LDPC block code of that length. We refer to this kind of decoding as one-shot decoding, since decoding is performed only once after the whole frame has been received. We were able to decode using the one-shot decoder for very long frame lengths (close to a million bits) quite easily. This is because of the simple structure of the Tanner graphs of the convolutional codes, which just carries over from the block codes.

4 Results

The performance obtained with these convolutional codes for rates 2/5 and 2/7 when decoded using both continuous and one-shot decoding is shown in Figures 4 and 5. The performance of the corresponding quasi-cyclic LDPC block code is also shown for comparison. We find that there is little loss in performance by decoding continuously compared to decoding after the whole frame has been received and that the rate 2/5 codes achieve good performance beyond the cut-off rate. Interestingly,

the convolutional codes seem to perform better than the block codes despite requiring fewer iterations to converge. However, the performance of these convolutional codes is not very close to capacity. There are two reasons for this - one, the construction of quasi-cyclic codes described in section 2 is good primarily for short to moderate block lengths. This implicitly bounds the performance of the convolutional codes that are derived from these codes; and two, the regular nature of the Tanner graphs representing these convolutional codes further limits the performance of belief propagation decoding on such graphs (it is been shown in [11] how only carefully designed irregular graphs approach the capacity limit). In fact, the lower rate 2/7 codes perform worse than the rate 2/5 codes! This can be explained in terms of the threshold phenomenon of iterative decoders based on belief propagation [11]. (The rate 2/5 convolutional codes[2] resemble (3,5) LDPCs whose threshold is 0.965 dB, while the rate 2/7 codes resemble (5,7) LDPCs whose threshold is 2.65 dB for a binary input AWGN channel.) This opens an interesting avenue for further work: can we optimize the degree profile of the nodes in the constraint graph and thereby improve the performance of such convolutional codes?

5 Conclusions

A set of convolutional codes with a simple graph structure has been constructed from their corresponding quasi-cyclic block versions. Good performance beyond the cut-off rate limit has been obtained with belief propagation decoding for the rate 2/5 codes. Recently, for short to moderate block lengths, modifications have been proposed to the belief propagation algorithm to achieve performance closer to maximum likelihood [12]. We are currently testing the performance of the convolutional codes using similar techniques. It is of interest to note that the performance of the convolutional codes is better than the associated block codes; this suggests that with good block LDPC codes that are also quasi-cyclic codes, it may be possible to obtain convolutional versions that perform even better. Also, modifications of the above construction may result in convolutional codes with irregular graph structures that can improve further upon the current results.

[2]More precisely, we mean the constraint graphs representing these codes.

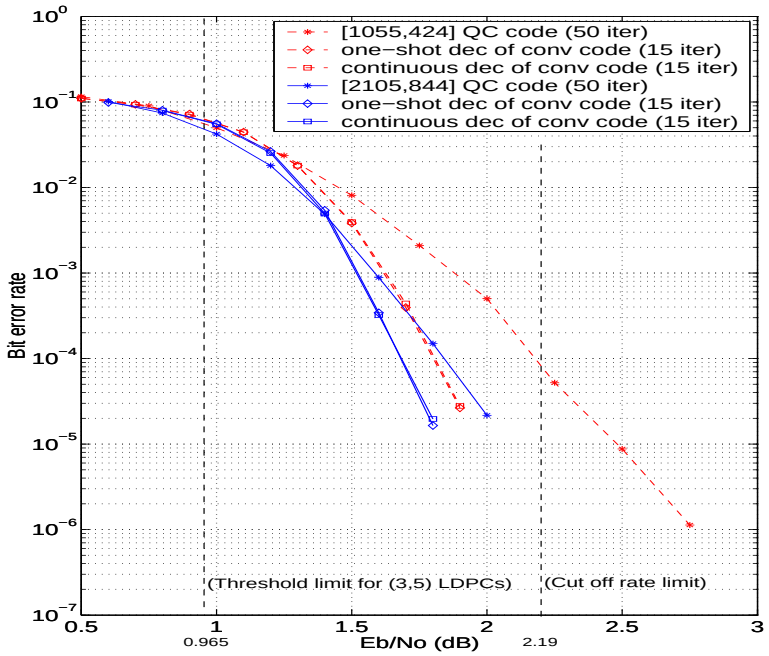

Figure 4: Performance of rate 2/5 convolutional codes derived from quasi-cyclic (QC) codes. The dotted lines correspond to codes derived from a block length 1055 LDPC code and the solid lines correspond to codes derived from a block length 2105 LDPC code.

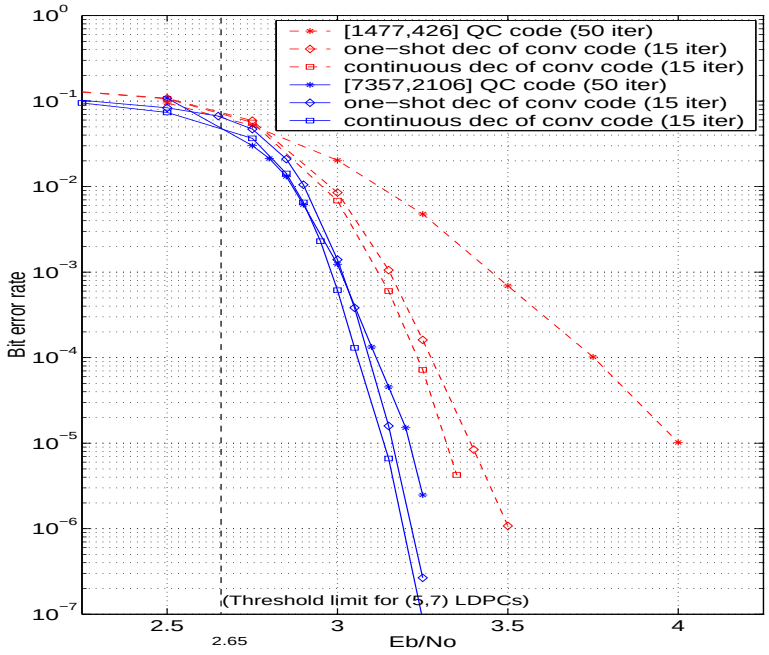

Figure 5: Performance of rate 2/7 convolutional codes derived from quasi-cyclic (QC) codes. The dotted lines correspond to codes derived from a block length 1477 LDPC code and the solid lines correspond to codes derived from a block length 7357 LDPC code.

References

[1] I. F. Blake and R. C. Mullin, *The mathematical theory of coding*. Academic Press, New York, 1975.

[2] R. M. Tanner, "A [155,64,20] sparse graph (LDPC) code." Presented at the recent results session, IEEE Intl. Symposium on Information Theory, Sorrento, Italy, June 2000.

[3] D. Sridhara, T. Fuja, and R. M. Tanner, "Low density parity check matrices from permutation matrices," in *Proceedings of 2001 Conference on Information Sciences and Systems*, p. 142, Johns Hopkins University, Baltimore, MD, March 2001.

[4] R. M. Tanner, "Convolutional codes from quasi-cyclic codes: A link between the theories of block and convolutional codes." Technical Report, Computer Research Laboratory, UC Santa Cruz, November 1987.

[5] Y. Levy and D. J. Costello, "An algebraic approach to constructing convolutional codes from quasi-cyclic codes," *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, vol. 14, pp. 189–198, 1993.

[6] R. Gallager, *Low-density parity-check codes*. M.I.T. Press, 1963.

[7] R. Johannesson and K. S. Zigangirov, *Fundamentals of Convolutional Coding*. IEEE Press, 1999.

[8] D. J. C. MacKay and M. C. Davey, "Evaluation of Gallager codes for short block length and high rate applications," in *Codes, Systems and Graphical Models*, vol. 123 of *IMA Volumes in Mathematics and its Applications*, pp. 113–130, New York: Springer-Verlag, 2000.

[9] R. M. Tanner, "Toward an algebraic theory for turbo codes," in *Proceedings of the Second International Symposium on Turbo Codes*, pp. 17–26, ENST-Bretagne, Brest, France, September 2000.

[10] A. J. Felstrom and K. S. Zigangirov, "Time-varying periodic convolutional codes with low-density parity-check matrix," *IEEE Transactions on Information Theory*, vol. 45, pp. 2181–2191, September 1999.

[11] T. Richardson, A. Shokrollahi, and R. Urbanke, "Design of capacity-approaching irregular low-density parity-check codes," *IEEE Transactions on Information Theory*, vol. 47, pp. 619–637, February 2001.

[12] M. Fossorier, "Iterative reliability-based decoding of low-density parity check codes," *IEEE Journal on Selected Areas in Communications*, vol. 27, pp. 533–547, May 2001.