Iterative Threshold Decoding Of High Rates Quasi-Cyclic OSMLD Codes

Karim Rkizat  
Mohammed V University in Rabat, ENSIAS  
Labo SIME, Team TSE  
Rabat, Morocco

Anouar Yatribi  
Mohammed V University in Rabat, ENSIAS  
Labo SIME, Team TSE  
Rabat, Morocco

Mohammed Lahmer  
Moulay Ismail University  
High School of Technologie  
Meknes, Morocco

Mostafa Belkasmri  
Mohammed V University in Rabat, ENSIAS  
Labo SIME, Team TSE  
Rabat, Morocco

Abstract—Majority logic decoding (MLD) codes are very powerful thanks to the simplicity of the decoder. Nevertheless, to find constructive families of these codes has been recognized to be a hard job. Also, the majority of known MLD codes are cyclic which are limited in the range of the rates. In this paper a new adaptation of the Iterative threshold decoding algorithm is considered, for decoding Quasi-Cyclic One Step Majority logic codes (QC-OSMLD) codes of high rates. We present the construction of QC-OSMLD codes based on Singer difference sets of rate 1/2, and codes of high rates based on Steiner triple system which allows to have a large choice of codes with different lengths and rates. The performances of this algorithm for decoding these codes on both Additive White Gaussian Noise (AWGN) channel and Rayleigh fading channel, to check its applicability in wireless environment, is investigated.

Keywords—Iterative threshold decoding; Quasi-Cyclic codes; OSMLD codes; Majority logic decoding; Steiner Triple System; BIBD

I. INTRODUCTION

Today LDPC codes [1] are present in most Telecom standards like DVB-S2 and WiMAX [2]. However, the decoding of these codes remain algorithmically complex and in situations such as the DVB-S2 [3] are often concatenated with codes such as Reed Solomon to improve performances. In our point of view, the MLD codes are better competitors for LDPC codes after describing the iterative threshold decoding algorithm and explaining the modification made for the Rayleigh fading channel. Finally, the last part presents the simulations results and analyses the ITD algorithm for decoding the constructed codes on both Rayleigh fading channel and AWGN channel.

II. QUASI-CYCLIC OSMLD CODES

A. OSMLD Codes

Consider an (n, k) linear code C with parity-check matrix H. The row space of H is an (n, n-k) code, denoted by C⊥, which is the dual code of C or the null space of C. For any vector v in C and any vector w in C⊥, the inner product of v and w is zero [8]. Now let consider that a codeword in C is transmitted over a binary symmetric channel. Taking into consideration that e(ε₁, ε₂,..., εₙ) and r(r₁, r₂,..., rₙ) are the error vector and the received vector, respectively. Then r = v + e. The construction of the below linear sum of the received
vector for any vector \( w \) in the dual code \( C^\perp \):

\[
A = \sum_{p=1}^{n} r_p w_p
\]

(1)

Which is called a parity-check sum. Using the fact that \( \langle w, v \rangle = 0 \), the following relationship between the parity-check sum \( A \) and error digits in \( e \) is obtained:

\[
A = \sum_{p=1}^{n} e_p w_p
\]

(2)

Suppose that there exist \( J \) vectors in the dual code \( C^\perp \), which have the following properties:

1) The \( j^{th} \) component of each vector \( w_i \) is a 1.
2) For \( i \neq j \) there is at most one vector whose \( i^{th} \) component is a 1.

These \( J \) vectors are said to be orthogonal on the \( j^{th} \) digit position. They are called orthogonal vectors. Now, let us form \( J \) parity-check sums from these \( J \) orthogonal vectors. For each \( i \) in \( 1, \ldots, J \)

\[
A_i = \sum_{p=1}^{n} e_p + e_j
\]

The error digit \( e_j \) is checked by all the check sums above. Because of the second property of the orthogonal vectors, any error digit other than \( e_j \) is checked by at most one check sum. These \( J \) check sums are said to be orthogonal on the error digit \( e_j \). If all the error digits in the sum \( A_i \) are zero for \( i \neq j \), the value of \( A_i \) is equal to \( e_j \). Based on this fact, the parity-check sums orthogonal on \( e_j \) can be used to estimate \( e_i \), or to decode the received digit \( r_i \).

**B. Majority logic decoding principle**

The error digit \( e_j \) is decoded as 1 if at least one-half of the check sums orthogonal on \( e_j \) are equal to 1; otherwise, \( e_j \) is decoded as 0 like majority rule [8]. When \( C \) is a cyclic code, each \( e_i \) can be decoded simply by cyclically permuting the received word \( r \) into the buffer store.

**Example 1:**

Let us consider the (7,3) code, which is the short code in difference set codes class. This code is specified by the perfect difference set \( P = 0, 2, 3 \) of order 21. From this perfect set, the following three check sums orthogonal on \( e_7 \) could be formed:

\[
A_1 = e_4 + e_5 + e_7
\]

\[
A_2 = e_2 + e_6 + e_7
\]

\[
A_3 = e_1 + e_3 + e_7
\]

If a simple error \( e = (000001) \) occurs, then \( A_1 = A_2 = A_3 = 1 \). If a double error occurs; for example, \( e_7 = 1 \) and one value of \( e_1, \ldots, e_6 \) is equal to 1, then two values of \( A_i \) are 1. So we can say that:

1) \( e_7 = 1 \) if only and if at least 2 values of \( A_i \) are 1
2) \( e_7 = 0 \), otherwise

**C. Quasi-cyclic Codes**

A code is said to be quasi-cyclic if every cyclic shift of a codeword by \( p \) positions results in another codeword [9]. Therefore, a QC codes are a generalization of cyclic codes with \( p = 1 \). A QC code \( (m n_0, m k_0) \) with a minimum distance \( d \) based on difference set can be specified with \( k_0 \) disjoint difference sets \( \{D_1, D_2, \ldots, D_{k_0}\} \) such that \( D_i (d_{i0}, d_{i1}, d_{i2}, \ldots, d_{i(S-1)}) \) of order \( S \), chosen from the set \( \{0,1,2,\ldots, mk_0\} \) [5]. The parity check matrix \( H \) in the systematic form of such code is completely defined as follows:

\[
H = [P_1 I_2 \ldots P_{k_0} I_{n-k}]
\]

(3)

The circulant matrix \( P_i \) is deducted from the difference set \( D_i \); the elements of \( D_i \) can specify the position in the matrix header \( P_i \) with one, while \( d_{ij} \) represents one in the position \( j \), the others rows are obtained by a cyclic shift of the header. Where \( I \) represents the identity matrix.

The majority logic decoding algorithm for QC codes is the same as cyclic codes. However, there is a little bit difference between them. Hence, in cyclic codes each error digit \( e_i \) can be decoded by cyclically permuting the received word \( r \), but in QC codes in systematic form, shift is done cyclically by one position of each \( (n-k) \) bits simultaneously.

**Example 2:**

Let consider the QC code \( C(6,3,3) \). This code is of the rate \( 1/2 \) and based on the Singer difference set \( DS\{0,1\} \) of order 2.

The parity check matrix \( H \) in systematic form is \([P I_3]\)

\[
H = \begin{pmatrix}
1 & 1 & 0 & 1 & 0 & 0 \\
0 & 1 & 1 & 0 & 1 & 0 \\
1 & 0 & 1 & 0 & 0 & 1 \\
\end{pmatrix}
\]

The parity-check sum orthogonal on \( e_3 \) is obtained from the parity check matrix \( H \):

\[
A_1 = e_2 + e_3 + e_5
\]

\[
A_2 = e_1 + e_3 + e_6
\]

**III. CONSTRUCTION BASED ON SINGER DIFFERENCE SET**

**A. Singer Difference Set**

A difference set [10] of order \( S \) and modulo \( m \geq S(S - 1)+1 \) is defined as a collection of \( S \) integer specified from the set \{0,1,2,\ldots,m-1\} such that no two of the \( S(S - 1) \) ordered differences modulo \( m \) are identical. If \( m=S(S - 1)+1 \), then for any non-zero integer \( n < m \), there is exactly one pair of elements in the difference set such that their difference is congruent to \( n \) modulo \( m \). Such a set is called a perfect difference set.

Singer [11] has demonstrated how to construct such sets when \( S = p^m + 1 \) and \( p \) is prime. Points and lines in the projective geometry \( PG(m - 1, q) \) form a difference set of parameters \( \{q,m\} \). The construction of Singer difference sets with parameters \( \{q,m\} \) is straightforward if a primitive polynomial of degree \( m \) in \( F_q \) is known. The following is the basis of Singer algorithm:

1) Choose a primitive polynomial of degree \( m \) in \( F_q 

\[
f(x) = x^m + \sum_{i=1}^{m} a_i x^{m-i}
\]

(4)
2) Choose the start value \( \lambda_0 = 0, \lambda_1, \lambda_2, \ldots, \lambda_{m-1} \)

3) Calculate the recurrence relation

\[
\lambda_n = -\sum_{i=1}^{m} a_i \lambda_{n-i} \quad (5)
\]

4) The set of integers \( \{0 \leq i < \frac{p^n-1}{q-1} : \lambda_i = 0\} \) is a Perfect Difference Set

Example 3:

1) Taking primitive polynomial \( x^4 + x^3 + 2 \) of degree 4 on \( \mathbb{F}_5 \)
2) Choosing the start sequence \( \lambda_0 = 0, \lambda_1 = 0, \lambda_2 = 1 \)
3) Then, calculating the recurrence relation as follows: \( \lambda_n = 2\lambda_{n-1} + \lambda_{n-4} \) yields the sequence

\[
100121222011222021120102212022002000
\]

4) The Positions where \( \lambda_i = 0 \) are:

\[
\{1, 2, 3, 9, 17, 19, 24, 26, 29, 30, 35, 38, 39\}
\]

that form a Perfect Difference Set.

B. Construction based on Singer Difference Set

The Singer construction allows to have codes with rate \( \frac{1}{2} \).

It’s possible to construct a single perfect difference set of order \( v=2^m, 1 \) and modulo \( m=2p^u+1 \) where \( p \) is a prime number, and \( u \) is a positive integer. As a result, it’s always possible to construct codes with minimum distance \( p^u+2 \) and length \( 2(p^{2u}+p^u+1) \).

All constructed codes are OSMLD and completely orthogonalizable since the orthogonal parity-check equations number \( J \) is always equal to \( d_{mn-1} \), where \( d \) is the minimum distance of the code.

In [5], Townsend and Weldon has constructed a rate \( \frac{1}{2} \) codes of small length up to \( n=366 \). With the help of Magma [12], many perfect difference sets have been constructed, which allows to construct a large number of QC-OSMLD codes of large lengths up to \( n = 2.10^8 \).

Table 1 shows a part of constructed codes. The parameters in this table are:

- \( P^n \): \( P \) is prime, and \( u \) an integer
- \( m \): is the modulo, \( m = 2p^u+p^u+1 \)
- \( n, k, d \): \( n \) is the length and \( k \) is the dimension of the code, \( d \) is the minimum distance of the code

Difference Sets : Represent the constructed Singer difference set

Density : is the density of the parity check matrix \( H \)

LDPC : specifies if the code is Low Density Parity Check (LDPC) code, for that it’s obligatory to have density \( \leq \log_2(n) \)

From this table, it’s clear that the majority of QC-OSMLD codes are LDPC codes which allow us to decode them with LDPC decoder like Sum-Product, Belief Propagation ...

IV. Construction based on Steiner Triple System

Historically, Smith [13] presented in 1968 an application of incomplete block designs to the construction of several families of error-correcting codes which may be decoded using a relatively simple majority logic decoding procedure. However, he didn’t give any explicit construction for such designs. Special cases of these codes are equivalent to the Self-orthogonal Quasi-cyclic codes based on Perfect Difference Sets discussed by Townsend and Weldon [5] (1967).

Chen Zhi and al stated an [6] explicit constructions of many classes of difference families considered as base blocks for Steiner designs. He presented a construction of infinite optimal self-orthogonal quasi-cyclic codes with high rates.

This section describes briefly the different construction methods of QC-OSMLD codes based on block design. Also, a part of constructed codes generated automatically by Matlab programmes is represented.

A. Balanced Incomplete Block Design

A Balanced Incomplete Block design (BIBD) [10] is a pair \((V, B)\) where \( V \) is a set and \( B \) is a collection of \( b \) \( k \)-subsets of \( V \) (blocks) such that each element of \( V \) is contained in exactly \( r \) blocks, and any \( 2 \)-subset of \( V \) is contained in exactly \( \lambda \) blocks. The numbers \( v, b, r, k, \) and \( \lambda \) are said parameters of the BIBD.

Trivial necessary conditions for the existence of a BIBD\((v, b, r, k, \lambda)\) are :

1) \( vr = bk \)
2) \( r(k-1) = \lambda(v-1) \)

The incidence matrix of a BIBD \((V, B)\) with parameters \( v, b, r, k, \lambda \) is a \( v \times b \) matrix \( A = (a_{ij}) \), in which \( a_{ij} = 1 \) when the \( i \)-th element of \( V \) occurs in the \( j \)-th block of \( B \), and \( a_{ij} = 0 \) otherwise.

B. Construction methods

1) QC-OSMLD codes based on STS \((v = 6t+1)\) : This construction [6][14] is applicable for \( k = 3 \) and \( v \) is a power of a prime of the form \( v \equiv 1(mod 6) \). Considering for each \( v \), \( GF(v = p^f) \), the Galois field of order \( v \). Let \( \omega \) be a primitive root. Then the STS difference family with parameters \( v = 6t+1, b = 6t^2+t, r = 3t, k = 3, \lambda = 1 \), is described by base blocks typically given by the form :

\[
(\omega^0, \omega^{2t}, \omega^{4t}), (\omega^t, \omega^{2t+i}, \omega^{4t+i}), (\omega^{t-1}, \omega, \omega^{5t-1})
\]

Another construction proposed by Rosa [15][10], for which the knowledge of a primitive root is not required, may be applicable for these designs using Skolem Sequences.

From [14] the block designs given above, an infinite class of optimal QC-OSMLD \([t+1](6t+1), t(6t+1), 4\) codes with the basic block length \( n_0 = (t+1)p^{e-1} \) is specified.
TABLE I: Rate $\frac{1}{7}$ QC-OSMLD codes based on Perfect Difference sets

<table>
<thead>
<tr>
<th>P</th>
<th>m</th>
<th>(n,k)</th>
<th>d</th>
<th>Difference Sets</th>
<th>Density</th>
<th>log₂(n)</th>
<th>LDPC</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>(6,3)</td>
<td>3</td>
<td>0 1</td>
<td>66.66</td>
<td>2.5849</td>
<td>No</td>
</tr>
<tr>
<td>2</td>
<td>7</td>
<td>(14,7)</td>
<td>4</td>
<td>0 1 3</td>
<td>28.57</td>
<td>3.8073</td>
<td>No</td>
</tr>
<tr>
<td>3</td>
<td>13</td>
<td>(26,13)</td>
<td>5</td>
<td>0 1 3</td>
<td>19.23</td>
<td>4.7004</td>
<td>No</td>
</tr>
<tr>
<td>2²</td>
<td>21</td>
<td>(42,21)</td>
<td>6</td>
<td>0 1 4 14 16</td>
<td>14.28</td>
<td>5.9392</td>
<td>No</td>
</tr>
<tr>
<td>5</td>
<td>31</td>
<td>(62,31)</td>
<td>7</td>
<td>0 1 10 14 26</td>
<td>11.29</td>
<td>5.9541</td>
<td>No</td>
</tr>
<tr>
<td>7</td>
<td>57</td>
<td>(114,57)</td>
<td>9</td>
<td>0 1 6 15 22 26 45 55</td>
<td>8.33</td>
<td>6.5849</td>
<td>No</td>
</tr>
<tr>
<td>2³</td>
<td>73</td>
<td>(146,73)</td>
<td>10</td>
<td>0 1 12 20 26 30 33 35 57</td>
<td>7.89</td>
<td>6.8328</td>
<td>No</td>
</tr>
<tr>
<td>3²</td>
<td>91</td>
<td>(182,91)</td>
<td>11</td>
<td>0 1 37 59 58 66 82 86</td>
<td>6.04</td>
<td>7.5077</td>
<td>Yes</td>
</tr>
<tr>
<td>13</td>
<td>113</td>
<td>(266,133)</td>
<td>13</td>
<td>0 1 13 17 21 58 65 78 100 101 115 124</td>
<td>4.88</td>
<td>8.0552</td>
<td>Yes</td>
</tr>
<tr>
<td>183</td>
<td>183</td>
<td>(366,183)</td>
<td>15</td>
<td>0 1 3 24 41 52 57 60 70 92 149 164 176</td>
<td>4.09</td>
<td>8.5156</td>
<td>Yes</td>
</tr>
<tr>
<td>2³</td>
<td>273</td>
<td>(546,273)</td>
<td>18</td>
<td>0 1 22 33 83 122 135 141 159 175 200 226 229 231 238 246</td>
<td>3.09</td>
<td>9.09</td>
<td>Yes</td>
</tr>
<tr>
<td>17</td>
<td>307</td>
<td>(614,307)</td>
<td>19</td>
<td>0 1 30 37 50 55 76 98 117 129 133 157 189 199 222 293 299</td>
<td>2.75</td>
<td>9.26</td>
<td>Yes</td>
</tr>
<tr>
<td>19</td>
<td>381</td>
<td>(762,381)</td>
<td>21</td>
<td>0 1 13 38 52 65 80 104 112 145 201 212 217 241 261 307 339 375</td>
<td>2.26</td>
<td>9.57</td>
<td>Yes</td>
</tr>
<tr>
<td>23</td>
<td>533</td>
<td>(1106,533)</td>
<td>25</td>
<td>0 1 13 41 60 64 109 146 151 185 265 286 306 312 331 337 375 454 460 479 486 501 544</td>
<td>1.77</td>
<td>10.11</td>
<td>Yes</td>
</tr>
<tr>
<td>29</td>
<td>871</td>
<td>(1742,871)</td>
<td>31</td>
<td>0 1 23 30 41 88 97 132 165 186 201 213 235 306 319 345 425 431 542 547</td>
<td>1.66</td>
<td>10.76</td>
<td>Yes</td>
</tr>
<tr>
<td>31</td>
<td>993</td>
<td>(1986,993)</td>
<td>33</td>
<td>0 1 13 101 127 154 169 204 210 225 259 289 297 317 356 434 474 478 495</td>
<td>1.37</td>
<td>10.95</td>
<td>Yes</td>
</tr>
<tr>
<td>37</td>
<td>1407</td>
<td>(2841,1407)</td>
<td>39</td>
<td>0 1 3 25 32 82 99 208 313 410 453 479 487 557 621 649 709 736 742 827 827</td>
<td>1.24</td>
<td>11.47</td>
<td>Yes</td>
</tr>
<tr>
<td>47</td>
<td>2257</td>
<td>(4514,2257)</td>
<td>49</td>
<td>0 1 13 138 143 296 377 381 457 546 550 690 712 773 802 891 905 973 979 996</td>
<td>1.06</td>
<td>12.14</td>
<td>Yes</td>
</tr>
<tr>
<td>95</td>
<td>957</td>
<td>(19014,9570)</td>
<td>99</td>
<td>0 1 3 7 25 192 308 332 433 914 919 984 1093 1155 1231 1238 1600</td>
<td>0.49</td>
<td>21.42</td>
<td>Yes</td>
</tr>
<tr>
<td>181</td>
<td>32943</td>
<td>(65886,32943)</td>
<td>183</td>
<td>0 1 129 145 211 306 460 514 543 747 781 800 894 1044 1101 1152 1573 1579 1579</td>
<td>0.27</td>
<td>20.80</td>
<td>Yes</td>
</tr>
</tbody>
</table>

2) QC-OSMLD codes based on STS (v = 6t + 3): The construction of such designs using the Extended Skolem Sequences is proposed. A Skolem sequence of order n is a sequence $S = (s_1, s_2, ..., s_{2n})$ of 2n integers satisfying the conditions:

1) for every $k \in \{1, 2, ..., n\}$ there exists exactly two elements $s_i, s_j \in S$ such that $s_i = s_j = k$

2) if $s_i = s_j = k$ with $i < j$, then $j - i = k$.

Skolem sequences are also written as collections of ordered pairs $\{(a_i, b_i) : 1 \leq i \leq n, b_i - a_i = i\}$ with $\cup_{i=1}^{n} \{a_i, b_i\} = \{1, 2, ..., 2n\}$.

Example 4:
A Skolem sequence of order $S : S = (1, 1, 3, 4, 5, 3, 2, 4, 2, 5)$ or, equivalently, the collection $\{(1, 2), (2, 3), (3, 6), (4, 8), (5, 10)\}$.

An extended Skolem sequence of order $n$ is a sequence $ES = (s_1, s_2, ..., s_{2n})$ of $2n + 1$ integers satisfying conditions 1 and 2 of the previous definition and:

3) there is exactly one $s_i \in ES$ such that $s_i = 0$.

The element $s_i = 0$ is called the hook or zero of the sequence.

3) Construction By A. Rosa: Suppose $\{1, ..., 3n+1\} \setminus \{2n+1\}$ can be partitioned into $m$ triples $\{a, b, c\}$ such that $a + b + c = 0$ or $a + b + c = \pm (mod 6n + 3)$. (This problem is called the second Heffter difference problem). Then the set of all triples $\{0, a, b\}$, together with “short block” $\{0, 2n+1, 4n+2\}$, is a $(6n + 3, 3, 1)$ cyclic partial difference family; the base blocks for a STS$(6n + 3)$. Heffter’s second difference problem is solved using extended Skolem sequences of order $n$ with a hook in the $n$th position. Therefore, such as the sequence, the pairs $(b, a_r)$ is constructed such that $b - a_r = r$, for $1 \leq r \leq n$. Then the set of all triples $\{r, a_r + n, b_r\}$ is taken, for $1 \leq r \leq n$. Below an explicit construction for the required Skolem sequences (as ordered pairs):

$$(r, 4s - r + 1) : r = 1, ..., s - 1$$

$$(s + r - 1, 3s - r) : r = 1, ..., s - 1$$

$$(4s + r + 1, 8s - r + 1) : r = 1, ..., s - 1$$

$$(5s + r + 1, 7s - r + 1) : r = 1, ..., s - 1$$

$$(2s - 1, 2s), (3s, 5s), (3s, 1 + 7s), (6s, 1 + 8s)$$

www.ijacsa.thesai.org
When $n = 2$, take the sequence : 

\[ \{ (1, 2), (4, 6) \} \]

When $n = 5$, take : 

\[ (1, 5), (2, 7), (3, 4), (8, 10), (9, 12) \]

When $n = 1$, the sequence does not exist. The construction above gives $STS(6n + 3)$ with parameters $(v, b, r, k, \lambda) = (6t + 3, (3t + 1)(2t + 1), 3t + 1, 3, 1)$.

4) **QC-OSMLD codes based on STS** ($v = 12t + 7$): This construction [10] is available for $v$ a prime power in the form $v = 12t + 7$. Let $\omega$ be a primitive root of the Galois field $GF(12t + 7)$. Then, the base blocks of a design with parameters $v = 12t + 7, b = (2t + 1)(12t + 7), r = 3(2t + 1), k = 3, \lambda = 1$ are given in the form $(\omega^i, \omega^{2i+2t}, \omega^{4i+4})$ such that $i = 0, \ldots, t - 1$. Block designs [6] given above specify an infinite class of optimal QC-OSMLD $(n, k, d_{min}) = ((t + 1)(12t + 1), t(12t + 1), 5)$ codes with basic block length $n_0 = (t + 1)p^{r-1}$.

5) **QC-OSMLD codes based on STS** ($v = 12t + 1$): This construction [6] is applicable for $v$ a prime power in the form $v = p^r = 12t + 1$. Let $\omega$ be the primitive root of $GF(p^r)$ such that $\omega^{4t + 1} = \omega^9$ where $q$ is odd. Then, the base blocks of a design with parameters $(v = 12t + 1, b = t(12t + 1), r = 3(12t + 1), k = 4, \lambda = 1)$ are given in the form : 

\[ (0, \omega^0, \omega^{4t}, \omega^{8t}), (0, \omega^{2i}, \omega^{4i+2t}, \omega^0, \omega^{4t+2}, \omega^6), (0, \omega^{2i+2}, \omega^{6i+2}, \omega^{10t+2}), \]

such that $i = 0, \ldots, t - 1$. Block designs [6] given above specify an infinite class of optimal QC-OSMLD $(n, k, d_{min}) = ((t + 1)(12t + 1), t(12t + 1), 5)$ codes with basic block length $n_0 = (t + 1)p^{r-1}$.

6) **QC-OSMLD codes based on STS** ($v = 20t + 1$): This construction [6] is applicable for $v$ a prime power in the form $v = p^r = 20t + 1$. Let $\omega$ be the primitive root of $GF(p^r)$ such that $\omega^{4t + 1} = \omega^9$ where $q$ is odd. Then, the base blocks of a design with parameters $(v = 20t + 1, b = t(20t + 1), r = 5t, k = 5, \lambda = 1)$ are given in the form : 

\[ (\omega^{2i}, \omega^{4i+2}, \omega^{6i+2}, \omega^{8i+2}, \omega^{12i+2}, \omega^{16i+2}), \]

such that $i = 0, \ldots, t - 1$. Block designs [6] given above specify an infinite class of optimal QC-OSMLD $(n, k, d_{min}) = ((t + 1)(20t + 1), t(20t + 1), 6)$ codes with basic block length $n_0 = (t + 1)p^{r-1}$.

**C. New constructed codes**

The following tables represent a small part of many constructed QC-OSMLD codes based on the methods described above. The length $n$, the dimension $k$, the minimum distance $d$, and the rate, and also the base blocks which represent the headers of the circulant matrix of the parity-check matrix $H$, are represented. Due to the significant number of base blocks in high rate, aren’t represented in the tables.

**TABLE II: QC-OSMLD codes based on STS $v = 12t + 7$**

<table>
<thead>
<tr>
<th>n</th>
<th>k</th>
<th>d</th>
<th>Rate</th>
<th>Base blocks</th>
</tr>
</thead>
<tbody>
<tr>
<td>76</td>
<td>57</td>
<td>4</td>
<td>0.75</td>
<td>1 4 16</td>
</tr>
<tr>
<td>186</td>
<td>155</td>
<td>4</td>
<td>0.833</td>
<td>1 19 20 9 16 29</td>
</tr>
<tr>
<td>344</td>
<td>301</td>
<td>4</td>
<td>0.87</td>
<td>1 4 41 9 12 25 10 36 38</td>
</tr>
<tr>
<td>804</td>
<td>737</td>
<td>4</td>
<td>0.92</td>
<td>1 19 25 4 9 52 16 36 37 7 10 64 14 40 55</td>
</tr>
<tr>
<td>4564</td>
<td>4401</td>
<td>4</td>
<td>0.96</td>
<td></td>
</tr>
</tbody>
</table>

**TABLE III: QC-OSMLD codes based on STS $v = 6t + 3$**

<table>
<thead>
<tr>
<th>n</th>
<th>k</th>
<th>d</th>
<th>Rate</th>
<th>Base blocks</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>35</td>
<td>4</td>
<td>0.7</td>
<td>1 3 4</td>
</tr>
<tr>
<td>91</td>
<td>70</td>
<td>4</td>
<td>0.77</td>
<td>1 4 6 2 6 8</td>
</tr>
<tr>
<td>144</td>
<td>117</td>
<td>4</td>
<td>0.81</td>
<td>2 7 10 3 8 12 4 11 13</td>
</tr>
<tr>
<td>206</td>
<td>176</td>
<td>4</td>
<td>0.85</td>
<td>1 6 10 2 7 12 3 8 9 4 13 15 5 14 17</td>
</tr>
<tr>
<td>286</td>
<td>247</td>
<td>4</td>
<td>0.86</td>
<td>1 7 12 2 8 11 3 9 15 4 10 14 5 14 16 6 16 17</td>
</tr>
<tr>
<td>851</td>
<td>762</td>
<td>4</td>
<td>0.919</td>
<td></td>
</tr>
<tr>
<td>1000</td>
<td>925</td>
<td>4</td>
<td>0.925</td>
<td></td>
</tr>
<tr>
<td>10000</td>
<td>10485</td>
<td>4</td>
<td>0.981</td>
<td></td>
</tr>
</tbody>
</table>
TABLE IV: QC-OSMLD codes based on STS \( v = 6t + 1 \)

<table>
<thead>
<tr>
<th>n</th>
<th>k</th>
<th>d</th>
<th>Rate</th>
<th>Base blocks</th>
</tr>
</thead>
<tbody>
<tr>
<td>39</td>
<td>26</td>
<td>4</td>
<td>0.667</td>
<td>1 3 9 2 5 6</td>
</tr>
<tr>
<td>76</td>
<td>57</td>
<td>4</td>
<td>0.75</td>
<td>1 7 11 3 14 4 6 9</td>
</tr>
<tr>
<td>125</td>
<td>100</td>
<td>4</td>
<td>0.80</td>
<td>1 6 11 2 12 22 4 19 24 8 13 23</td>
</tr>
<tr>
<td>186</td>
<td>155</td>
<td>4</td>
<td>0.833</td>
<td>1 5 25 3 13 15 8 9 14 11 24 27 2 10 19</td>
</tr>
<tr>
<td>344</td>
<td>301</td>
<td>4</td>
<td>0.875</td>
<td>1 6 36 3 18 22 9 11 23 26 27 33 13 35 38 19 28 39 14 30 41</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1649 1552 4 0.941 -</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2071 1982 4 0.947 -</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>64898 64021 4 0.99 -</td>
</tr>
</tbody>
</table>

TABLE V: QC-OSMLD codes based on STS \( v = 12t + 1 \)

<table>
<thead>
<tr>
<th>n</th>
<th>k</th>
<th>d</th>
<th>Rate</th>
<th>Base blocks</th>
</tr>
</thead>
<tbody>
<tr>
<td>75</td>
<td>50</td>
<td>5</td>
<td>0.666</td>
<td>0 1 8 11 2 4 19 24</td>
</tr>
<tr>
<td>148</td>
<td>111</td>
<td>5</td>
<td>0.75</td>
<td>0 1 10 26 3 4 30 9 12 16</td>
</tr>
<tr>
<td>245</td>
<td>196</td>
<td>5</td>
<td>0.80</td>
<td>0 1 25 37 9 29 38 16 23 32 0 32 43 46</td>
</tr>
<tr>
<td>366</td>
<td>305</td>
<td>5</td>
<td>0.833</td>
<td>0 1 25 37 9 29 38 16 23 32 0 32 43 46</td>
</tr>
<tr>
<td>511</td>
<td>438</td>
<td>5</td>
<td>0.86</td>
<td>-</td>
</tr>
<tr>
<td>873</td>
<td>776</td>
<td>5</td>
<td>0.89</td>
<td>-</td>
</tr>
<tr>
<td>1331</td>
<td>1210</td>
<td>5</td>
<td>0.91</td>
<td>-</td>
</tr>
<tr>
<td>10470</td>
<td>10121</td>
<td>5</td>
<td>0.966</td>
<td>-</td>
</tr>
</tbody>
</table>

TABLE VI: QC-OSMLD codes based on STS \( v = 20t + 1 \)

<table>
<thead>
<tr>
<th>n</th>
<th>k</th>
<th>d</th>
<th>Rate</th>
<th>Base blocks</th>
</tr>
</thead>
<tbody>
<tr>
<td>244</td>
<td>183</td>
<td>6</td>
<td>0.75</td>
<td>1 9 20 34 58 4 14 19 36 49 13 15 16 22 56</td>
</tr>
<tr>
<td>405</td>
<td>324</td>
<td>6</td>
<td>0.80</td>
<td>1 7 19 49 52 4 28 34 46 76 16 22 31 55 61 17 43 58 64</td>
</tr>
<tr>
<td>606</td>
<td>505</td>
<td>6</td>
<td>0.83</td>
<td>-</td>
</tr>
<tr>
<td>847</td>
<td>726</td>
<td>6</td>
<td>0.86</td>
<td>-</td>
</tr>
<tr>
<td>1810</td>
<td>1629</td>
<td>6</td>
<td>0.90</td>
<td>-</td>
</tr>
<tr>
<td>4215</td>
<td>3934</td>
<td>6</td>
<td>0.93</td>
<td>-</td>
</tr>
<tr>
<td>6859</td>
<td>6498</td>
<td>6</td>
<td>0.95</td>
<td>-</td>
</tr>
</tbody>
</table>

V. ITERATIVE THRESHOLD DECODING

A. Encoding

In the case of QC codes, the encoding can be achieved with simple shift registers while the complexity is linear [9][16]. Because the quasi-cyclic code is not in systematic form, an additional k-stage register is required by this encoder to store the information symbols of the next block until encoding is completed. This difficulty can be avoided by using an equivalent systematic code. In this case the codes constructed are in systematic form.

QC codes could be encoded by either generator matrix or polynomial multiplication. These codes are defined by the parity check matrix \( H \). The generator matrix \( G \) is obtained by the following transformation:

\[
H = [PI_{n-k}] \Leftrightarrow G = [I_kP^T]
\] (6)

The encoding algorithm consists of multiplying the message \( i \) by the generator matrix \( G \) to get the codeword \( v \).

\[
v = i \ast G
\] (7)

The Quasi-cyclic codes have a polynomial form. Consider C(n,k) a systematic quasi-cyclic code with rate \( \frac{1}{2} \), and let \( P \) which defines the code be the circulant matrix. The information vector to be encoded is denoted by \( i \), then

\[
v = iG = [i, iP]
\] (8)

Let \( i(x) \) and \( p(x) \) represent the information vector and the header of the circulant matrix \( P \) in polynomial form, respectively. Obviously, the remaining rows of \( P \) are:

\[\begin{align*}
&= \left[ \begin{array}{c}
1 & x & x^2 & \ldots & x^{k-1} \end{array} \right] \mod x^k - 1
\end{align*}\] (9)

The algebra of polynomials modulo \( x^m - 1 \) is equivalent to the algebra of \( m \times m \) circulant matrices; besides, the polynomial product \( i(x)p(x) \mod x^k - 1 \) is similar to multiplying the vector \( i \) by the circulant matrix \( P \). Hence,

\[
v(x) = [i(x), i(x)p(x)]
\] (10)

Example 5: Let consider the same code as in the example 2. Now, to transmit the message \( i=101 \).

\( i = 101 \) then \( i(x) = 1+x^2 \)

And \( p = 101 \) then \( p(x) = 1+x^2 \)

Then the codeword is:

\[
v(x) = [i(x), i(x)p(x)]=[1+x^2, (1+x^2)*(1+x^2)]=[1+x^2, 1+x^2]
\]

\[\Rightarrow v = 101110.
\]

In the case of QC codes with rate of the form \( \frac{n_o - 1}{2} \), the encoding like codes with rate \( \frac{1}{2} \) can be realised by either generator matrix or polynomial multiplication [9].

These codes are defined by the parity check matrix \( H \) of the form:

\[
H = [P_1P_2\ldots P_k I_{n-k}]
\] (11)

The generator matrix \( G \) is obtained by the following transformation:

\[
H = [P_1P_2\ldots P_k I_{n-k}] \Leftrightarrow G = \begin{bmatrix} P_1^T & \\ & \ddots & \\ & & P_k^T \end{bmatrix}
\]
In the case of codes with rate $\frac{n-1}{n}$, there are many circulant matrix $(P_1, P_2, \ldots, P_{n-1})$, then for encoding the information vector $i$, it must be divided into $k_0$ subgroup $(i_1, i_2, \ldots, i_{k_0})$ then based on the equation (10), the following equation is obtained:

$$v(x) = \sum_{j=1}^{k_0} i_j(x) p_j(x)$$

To clear this, let us consider the following example.

**Example 6:**

Let consider the OSMLD QC(15,10) code, with the minimum distance is $d=3$. The disjoint difference sets of order $s=2$ which define the parity matrix $H$ of this code are $D_1\{0,1\}$ and $D_2\{0,2\}$.

The parity check matrix $H$ in systematic form is $[P_1, P_2, I_5]$

$$
\begin{pmatrix}
1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0
\\
0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0
\\
0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0
\\
0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0
\\
1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1
\end{pmatrix}
$$

After transformation of the parity check matrix $H$, the generator matrix $G$ is obtained

$$
\begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0
\\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0
\\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1
\\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1
\\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0
\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0
\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1
\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix}
$$

Now transmitting the message $i=100101011$. To know the codeword $v$ corresponding to the message $i$, it’s possible to use the classical method by using the generator matrix $G$ $v = i^*G = 111011010110101$. For using the polynomial method, the message must be divided into $k_0=2$ vectors

$$i_1=10010 \Rightarrow i_1(x)=1+0x^3$$

$$i_2=01011 \Rightarrow i_2(x)=x+x^3+x^4$$

From the generator matrix $G$, headers of the two circulant matrix $P_1$ and $P_2$ is obtained:

$\begin{align*}
& c_1 = 10001 \Rightarrow c_1(x) = 1+x^4 \\
& c_2 = 10010 \Rightarrow c_2(x) = 1+x^3
\end{align*}$

Then, the equation (9) must be calculated:

$$v(x)=\frac{1}{2}[i(x), \frac{c_1(x)}{c_2(x)}]+(i_2(x)c_2(x)]$$

$$=[1+x^3+x^6+x^8+x^{12}, \frac{x^4(1+x^4)(1+x^4)}{2}+x^4(1+x^4)(1+x^4)]$$

$$=[1+x^3+x^6+x^8+x^{12}, (x^{2n+1}+x^{n+1})+x^4(1+x^4)]$$

Then, the codeword to transmit is:

$$v = 10100101110001$$

Which is the same as the codeword obtained by using generator matrix $G$.

**B. ITD**

Threshold decoding is simply the logical extension to soft decisions of majority decoding described above. In Massey’s original work [17], he considered two different variations of the decoding algorithm. Considering here the method which uses the Bi equations that are obtained from $A_i$ by a simple transformation [18].

Thanks to its speed and simplicity, the Majority Logic (ML) decoding of Quasi-Cyclic codes is significant. Therefore, it is worth investigating which Quasi-Cyclic codes can be decoded using ML decoder. Majority logic decoding is well described in [8,19]. It consists of cyclic shift register, XOR matrix, majority gate and XOR for correcting the codeword bit under decoding.

The ITD algorithm which is based on SISO extension of Massey threshold decoding algorithm [17] was developed mainly for decoding Parallel Concatenated Block Codes [20] and product codes constructed from OSMLD Codes [21], later it was enhanced for decoding not only Generalized Parallel Concatenated OSM LD Codes [22] but also OSM LD block codes [4]. ITD algorithm has an improvement in error correcting related to standard ML decoding.

Considering the transmission of the codeword $C(c_1, c_2, c_n)$ over an Additive White Gaussian Noise channel (AWGN), using BPSK modulation. The soft decision, which is the Log Likelihood Ratio (LLR), on the $j^{th}$ bit of the received word $R(r_1, r_2, \ldots, r_n)$ can be calculated as follows:

$$LLR_j = ln \frac{p(c_j = 1/R)}{p(c_j = 0/R)}$$

The hard decision vector corresponding to the received vector $R$ is denoted by $H(h_1, h_2, \ldots, h_n)$. Where $c_j$ is the $j^{th}$ bit of the transmitted codeword. For a code with $J$ orthogonal parity check equations; the equation (13) can be expressed as:

$$LLR_j = ln \frac{p(c_j = 1/B_j)}{p(c_j = 0/B_j)}$$

Where $B_i$, for $i$ in $\{1, \ldots, J\}$, are obtained from the orthogonal parity check equations on $c_j$ bit, as follows:

$$B_0 = h_j$$

and each $B_j$ with $i$ in $\{1, \ldots, J\}$, is calculated by eliminating the term $h_j$ from the $i^{th}$ orthogonal parity check equation. By applying BAYES rule, (14) becomes:

$$LLR_j = ln \frac{p(B_i/c_j=1)}{p(B_i/c_j=0)} \times \frac{p(c_j=1)}{p(c_j=0)}$$

Since the parity check equations are orthogonal on the $j^{th}$ position, so the individual probabilities $P(B_i/c_j=1$ or $0$) are all independent and (15) can be written as:

$$LLR_j = \sum_{i=0}^{J} \ln \left[ \frac{p(B_i/c_j=1)}{p(B_i/c_j=0)} \right] + \ln \left[ \frac{p(c_j=1)}{p(c_j=0)} \right]$$

Assume that the transmitted symbols are equally likely to be $+1$ or $-1$, and thus the last term in (16) is null. As a result, the equation (16) becomes:

$$LLR_j = \sum_{i=1}^{J} \ln \left[ \frac{p(B_i/c_j=1)}{p(B_i/c_j=0)} \right] + \ln \left[ \frac{p(B_0/c_j=1)}{p(B_0/c_j=0)} \right]$$

(17)
According to [18], (17) can be expressed as:

\[ LLR_j \simeq (1 - 2B_0)w_0 + \sum_{i=1}^{J} (1 - 2B_i)w_i \]  \hspace{1cm} (18)

Where the value of \(1-2B_i\) is equal to +1 or -1, and \(w_i\) is a weighting term proportional to the reliability of the \(i^{th}\) parity check equation. Then showing that:

\[ (1 - 2B_0)w_0 = 4 \frac{E_s}{N_0} r_j \] \hspace{1cm} (19)

Where \(E_s\) is the energy per symbol, and \(N_0\) is the noise spectral density.

\[ w_i = \ln \left[ 1 + \frac{k=n_j}{k=1,k\neq j} \tanh \left( \frac{L_{ik}}{2} \right) \right] - \frac{k=n_j}{k=1,k\neq j} \tanh \left( \frac{L_{ik}}{2} \right) \] \hspace{1cm} (20)

Where \(n_i\) is the total number of terms in the \(i^{th}\) orthogonal parity check equation without \(c_j\), \(ik\) represents the \(k^{th}\) element of the \(i^{th}\) parity check equation and with:

\[ L_{ik} = 4 \frac{E_s}{N_0} | r_{ik} | \] \hspace{1cm} (21)

Thus the soft output can be split into two terms, namely into a normalized version of the soft input \(r_j\) and an extrinsic information \(L_{E_j}\), representing the estimates made by the orthogonal bits on the current bit \(c_j\). Hence, (18) becomes

\[ LLR_j = 4 \frac{E_s}{N_0} r_j + L_{E_j} \] \hspace{1cm} (22)

Using the following notation:

\[ L_c = 4 \frac{E_s}{N_0} \] \hspace{1cm} (23)

Which is called the reliability value of the channel.

The algorithmic structure of the SISO threshold decoding can be summarized as follows:

For each \(j = 1,\ldots,n\):

- Compute the terms \(B_j\) and \(w_i\), \(i \in \{1,\ldots,J\}\)
- Calculate \(B_0\) and \(w_0\)
- Compute the extrinsic information \(L_{E_j}\)
- The soft-output is obtained by:

\[ LLR_j = L_c r_j + L_{E_j} \]

Iterative decoding process (see Figure 1) can be described as follows:

In the first iteration, the decoder only uses the channel output as input and generates extrinsic information for each symbol. In subsequent iterations, a combination of extrinsic information and channel output is used as input.

As shown in Figure 1, the soft input and the soft output of the \(q^{th}\) decoder is achieved through the following equations:

\[ R(q) = R + \alpha(q)L_{E}(q) \] \hspace{1cm} (24)

\[ LLR(q+1) = L_c R(q) + L_{E}(q+1) \] \hspace{1cm} (25)

Where \(R(q)\) represents lines (or columns) of the received data, and \(L_{E}(q)\) is the extrinsic information computed by the previous component decoder. In the proposed procedure, a fixed value \(1/J\) is used for the parameter \(\alpha(q)\) and this for all iterations. The value chosen for \(\alpha(q)\) reacts as an average of all \(J\) estimators which contribute in the computation of \(L_{E_j}\).

C. Modification of Rayleigh fading channel

In the channel model, each received bit \(r_j\) can be expressed as:

\[ r_j = a_j c_j + n_j \]

(26)

In this representation, \(c_j\) is a BPSK symbol associated to the transmitted bit \(c_j\), and \(n_j\) is an AWGN. The Rayleigh variable \(a_j\) is generated as:

\[ a_j = \sqrt{x_j^2 + y_j^2} \]

(27)

where \(x_j\) and \(y_j\) are zero mean statistically independent Gaussian random variables each having a variance \(\sigma^2\). Considering the power normalized to one as

\[ E\{a_j^2\} = \sigma^2 = 1 \]

(28)

Which gives a variance of 0.5 for Gaussian variables.

The main matter in determining the required modification for ITD algorithm is the availability of channel side information on the Rayleigh fading channel. The threshold decoding algorithm has to be modified slightly by changing equation (23) which defines the reliability value of the channel by

\[ L_c = 4 \frac{E_s}{N_0} a_j \]

(29)

With this modification, it’s possible to use the same decoder structure which was described in Figure 1.

VI. Simulation Results and Analysis

This section considers simulation results and analysis for some decoding QC-OSMLD codes of rate \(\frac{m_a-1}{n_0}\) and 1/2 with the Iterative Threshold Decoding algorithm. Some of our simulations are over AWGN channel, whereas others are over Rayleigh Fading channel; however, both of them are with modulation BPSK. Due to computational limitations, a minimal residual error of 200 have been used. In the simulations over Rayleigh fading channel, assuming an accurate fade estimate

![Fig. 1: Scheme of iterative threshold decoder](image-url)
at the receiving and an independent Rayleigh distribution of the fades.

The performance improves with each iteration in all simulation results presented. The following results represent the performance of decoding an QC-OSMLD code with the ITD algorithm and comparison with classic Threshold decoding algorithm.

A. AWGN

The Figure 2 depicts the performance of QC-OSMLD code (366,183,15). The improvement is great for the first iterations and is negligible after the 6th iteration. Figure 3 presents a comparison between three QC-OSMLD codes (366,183,15), (1106,553,25) and (4514,2257,49). Observing that smaller codes length has best performance at low SNR, whereas at SNR>5 for the code (4514,2257,49) the performance improves quickly from $10^{-1}$ to $10^{-5}$ between SNR=5 and SNR=6. The next comparison is between the code (182,91,11) decoded with ITD 10 iterations and the code LDPC WiMax(192,96,10), which is from [23], decoded with the Belief propagation decoder. These two codes are of the same rate $\frac{1}{2}$, and they have nearly the same dimension and minimum distance.

The Figure 4 shows that the LDPC WiMax code (192,96,10) decoded with BP algorithm outperforms the OSMLD-QC code (182,91,11) decoded with the ITD algorithm. However, the first decoder requires more iterations; on the other hand, the second one is less complex.

B. Rayleigh

The curves in the Figure 5 show the achieved bit error rates for the QC-OSMLD code (366,183,115) over Rayleigh fading channel. The number of iterations used is a 10, such that there is no significantly more to gain by more iterations.

Observing that the performance increases with each iteration, and the improvement is negligible after the 7th iteration.

Figure 6 presents a comparison between three QC-OSMLD codes (366,183,15), (1106,553,25) and (4514,2257,49) over Rayleigh fading channel. Observing that the same behaviour as the AWGN channel, but in high SNR.

The Figure 7 shows the performance of decoding the QC-
OSMLD code (366,183,15) on both Rayleigh and AWGN channels. As observed in the other simulations, the performance of this code in an independent Rayleigh channel is worse than that for the AWGN channel by approximately 5 dB. It is worth mentioning that the number of iterations needed is about the same for the both channels.

The figure 8 shows a comparison between the performance of decoding the QC-OSMLD codes of different rates. To simplify, two scenarios have been opted. In the first case, there are two codes (39, 26, 4) and (15, 10, 3) which have the same rate 2/3. From the graph above, it’s clear that as the code length increases, the performances rises, as well, which results in a gain of 1db at $10^{-5}$. In the second case, two codes with different rates has been compared (76,57,4) of rate 3/4 and (39,26,4) of rate 2/3 holding the same minimum distance which is 4. As a result, even if the code length is large, the code of rate 3/4 is outperformed by the code of rate 2/3 with difference of 3db at $10^{-5}$.
VII. CONCLUSION

In this paper, the construction of a class of QC-OSMLD codes based on Steiner triple system, and another class based on Singer difference sets has been investigated. The encoding methods has been presented for those codes. Also, the performances of decoding these codes with the ITD algorithm over AWGN channel and also over fading channel has been shown. The decoding algorithm used for AWGN channel is unchanged, and only the channel reliability factor needs to be redefined. The simulations results show that the constructed codes perform well when decoded with ITD algorithm. It is interesting to apply this iterative decoding algorithm on other channels models like Rice or Nakagami. Also as extension of this work we plan to investigate the performance of decoding rate $\frac{1}{n}$ QC-OSMLD codes with an adaptation of our ITD algorithm.

REFERENCES