Two-Dimensional Error-Pattern-Correcting Codes

Sung Whan Yoon, *Student Member, IEEE*, and Jaekyun Moon, *Fellow, IEEE*

***Abstract*—Two-dimensional (2D) cyclic codes are presented which correct any single occurrence of known 2D error patterns within a 2D array of bits. Applications for this type of codes include storage and display devices. The code construction begins with a generation of distinct syndrome sets for all targeted 2D error patterns. A method to refine the syndrome sets is then presented for making each syndrome set to contain distinct members, thereby guaranteeing full correction capability for the given list of known error patterns. Using an example construction, the effectiveness of the proposed coding approach is demonstrated versus the maximum-distance-separable (MDS) random-error-correcting code and known 2D burst-correcting codes for a 2D intersymbol interference (ISI) channel that yields a few dominant, but relatively large error patterns.**

***Index Terms*—Error correction codes, intersymbol interference, multidimensional signal processing, magnetic memory.**

I. Introduction

In storage systems, 2D intersymbol interference (ISI) arises as the physical size of individual bit cells becomes increasingly smaller to meet the ever increasing demand for higher storage density. 2D ISI causes a new class of error events in the form of 2D error patterns [1], [2]. For traditional one-dimensional (1D) storage channels with ISI, a few well-characterized error patterns often dominate the error rate performance, and 1D cyclic codes can be designed to correct such dominant error patterns effectively, providing improved code rate efficiencies over conventional error correction codes targeting random or burst errors [3], [4]. 2D error events caused by 2D ISI have been characterized in [5]. It has been shown that the minimum-distance events in the realistic 2D ISI channel that dominate the error probability performance are fairly large [5], suggesting that the type of codes that attack a few specific dominant error patterns may also yield performance advantage in such 2D ISI channels upon traditional random-error or burst correcting codes.

In [6], the present authors have discussed 2D error-pattern-correcting codes by establishing a mathematical condition for detecting any single occurrence of known error patterns at any position in a codeword, driven by the same motivation for developing 1D error-pattern-correcting code [3], [7]–[9]. The 1D error-pattern-correcting code (EPCC) design, however, cannot be extended easily to the 2D case, and new theoretical properties and techniques need to be explored and developed to enable efficient 2D error-pattern correcting code design. Whereas the theories of the 1D EPCC of [3] were based on the well-established general theory of conventional (1D) cyclic codes, the present 2D EPCC theories are built upon the theory of basic 2D cyclic codes developed in [10], [11]. Our theoretical contributions include establishing a mathematical foundation for designing 2D cyclic codes that can both detect and correct any single occurrence of predetermined 2D error patterns anywhere in a 2D array of bits. A general algorithm for constructing such 2D EPCCs is also given.

At a more specific level, to achieve detection capability for single occurrence of predetermined 2D error patterns, the corresponding syndrome sets for the list of known error patterns should be all distinct without any common syndrome elements. We present a theorem that provides a set of zeros producing distinct syndrome sets for predetermined 2D error patterns. For perfect correction capability, each syndrome within a set should indicate the exact position where a 2D error pattern has occurred. For this, each syndrome should be unique in its syndrome set. Another theorem presented reveals the necessary and sufficient conditions for maintaining unique syndrome elements within each syndrome set. Overall, the pattern type of the occurred error event is specified by the syndrome set while the exact position of the error event is uniquely determined by examining its syndrome.

The 2D code construction procedure first begins with generation of a set of zeros for a minimum parity code with error pattern detection capability. By exchanging and adding certain zeros, the procedure eventually identifies sets of zeros of 2D cyclic codes with correction capability for any single occurrence of a given list of predetermined 2D error patterns.

While there have been no previous attempts to the best of our knowledge to correct specific lists of 2D error pattern types as proposed here, a natural question arises as to how well this type of specific-error-pattern-correcting code works in comparison with existing 2D random and burst error correcting codes. In fact, there have been prior research efforts to target 2D errors of some general shape. For example, an early version of 2D codes correcting a rectangular-burst error of size $(b_1 \times b_2)$ has been constructed in [12]. The work of [13] also addresses correcting a single $(b_1 \times b_2)$-size 2D burst by using 1D component codes along vertical and horizontal directions. In both [12] and [13], however, the shape of the 2D error bursts is restricted to a rectangular form. In [14], a theoretical lower bound on the amount of redundancy for correcting a single arbitrary shaped 2D burst has been established without actual encoder/decoder construction. The authors of [15], [16] utilize multidimensional

Manuscript received November 18, 2014; revised April 3, 2015; accepted May 18, 2015. Date of publication June 1, 2015; date of current version August 7, 2015. This work was supported by NRF of Korea under Grant 2011-0029854. This paper was presented in part at IEEE Globecom 2012. The associate editor coordinating the review of this paper and approving it for publication was L. Dolecek.

The authors are with the Department of Electrical Engineering, Korea Advanced Institute of Science and Technology, Daejeon 305-701, Korea (e-mail: shyoon8@kaist.ac.kr; jmoon@kaist.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCOMM.2015.2438869

interleaving schemes to correct multidimensional bursts using 1D components codes. More recently, 2D codes correcting a single 2D error cluster with certain specified shapes have been constructed in [17] using a direct 2D algebraic coding approach. Burst-correcting codes for a 2D or multidimensional array with the same purposes have also been suggested in [18] but based on 1D component codes.

In contrast to all the above-mentioned works with the exception of [17], our code is based on *a direct 2D* design (versus designs based on some forms of concatenation of 1D codes), resulting in a higher code rate efficiency. Compared to the 2D codes of [17] which correct error bursts of various shapes under certain 2D error model constraints, our code can target a list of *any* arbitrary error patterns provided they are known *a priori*.

This paper is organized as follows. Section II starts with a quick review of 2D cyclic code theory of Imai [10], [11]. Also, a proof for the theorem in [6] for establishing detection capability is provided and some useful properties for the syndromes in 2D cyclic codes are presented. In Section III, another theorem which describes a necessary and sufficient condition for a given code to possess "full period" is presented. Having a full period means having complete position information for the occurred error event and thus implies perfect correction capability. Section IV describes a general algorithm to design full period codes. Next, in Section V, direct performance comparisons are made versus general random error correcting codes and existing 2D burst-correcting codes using a 2D ISI channel with a known list of minimum distance error events. Finally, conclusions are drawn in Section VI.

II. Construction of Error-Pattern-Detecting 2D Cyclic Codes

A. General Description of Two-Dimensional Cyclic Codes

The following description of general 2D cyclic codes is based on Imai's work [10], [11]. Our description here will aim at providing just enough background and establish notations for developing the present idea. This subsection is divided into two parts. One is about 'set of zeros of 2D cyclic codes' which is the most important concept of the codes and the other one is about details of encoding/decoding process. The reader is referred to [10], [11] for more details and general theory.

1) Set of Zeros of 2D Cyclic Codes: A 2D cyclic codeword is basically a bit array with N_x rows and N_y columns. The size of this 2D cyclic code, say, C, is said to be $(N_x \times N_y)$. Define the set

$$\Omega = \left\{(i,j) | 0 \le i \le N_x - 1, 0 \le j \le N_y - 1\right\}. \tag{1}$$

Then, a 2D cyclic codeword of size $(N_x \times N_y)$ can be represented by a polynomial:

$$c(x,y) = \sum_{(i,j)\in\Omega} c_{i,j} x^i y^j \tag{2}$$

where $c_{i,j}$ are binary coefficients taking the value 1 or 0. Like the 1D cyclic codes, cyclic shifting (in the 2D sense) along x or y direction of a 2D codeword $c(x,y)$ in code C results in another valid 2D codeword in the same code.

Recall that a 1D cyclic code is completely specified by its generator polynomial. Moreover, the zeros of the generator polynomial of 1D cyclic code C also becomes zeros of all codewords of the code C. For a 2D cyclic code, the "set of zeros" plays the role of the generator polynomial. A 2D cyclic code is completely specified by the set of zeros. With C denoting a 2D cyclic code of size $(N_x \times N_y)$, the set of zeros, denoted V_c, is defined as

$$V_c = \left\{(\alpha^i, \beta^j) | c(\alpha^i, \beta^j) = 0 \; \forall \text{ codewords } c(x,y)\right\} \tag{3}$$

where α and β are the N_x^{th} and N_y^{th} roots of the equations $x^{N_x} - 1 = 0$ and $y^{N_y} - 1 = 0$, respectively. For 1D cyclic codes, the encoder and decoder are constructed using a feedback shift-register with the feedback coefficients reflecting the given generator polynomial. For 2D cyclic codes, the shift-register setting for encoding and syndrome computation is specified by the corresponding set of zeros V_c.

To understand the shift-register operation for 2D cyclic codes, we need to first talk about the parity check bits. Recall that a 1D cyclic code of length n has $n - k$ bit positions designated for the parity bits. The message bits occupy the remaining k bit positions. Likewise, a 2D cyclic code has a designated area $\Pi \in \Omega$ for the parity check bits. Naturally, the message bits occupy the remaining area $\Omega - \Pi$. The set of zeros V_c determines the parity area Π and the corresponding feedback connections. In the encoding of a 1D cyclic code, the contents of the feedback shift-register become the parity bits once the entire message bits feed through the register. Likewise, the contents of the 2D feedback shift-register eventually become the parity bits in 2D codes. The 2D register allows shifting of their contents in two directions. Like the 1D case, the contents of the register that are being pushed out of the parity check area Π are fed back to the 2D register in Π.

Let U denote the set of zeros of a 2D polynomial over $GF(2)$. Then for any element (ξ, η) in U, (ξ^{2^i}, η^{2^i}) for $i = 1, 2, \cdots, n-1$ are also members of U. Here n is the least positive integer such that $(\xi, \eta) = (\xi^{2^n}, \eta^{2^n})$. The group of all such (ξ^{2^i}, η^{2^i}) is called the conjugate point set of (ξ, η). The conjugate set is completely represented by (ξ, η). From this point on, we assume that a set of zeros is the collection of representative points for the conjugate point sets. Generally a set of zeros can be written as

$$\begin{aligned} V_c = \big\{ & (\xi_1, \eta_{1,1}), (\xi_1, \eta_{1,2}), \cdots, (\xi_1, \eta_{1,t_1}) \\ & (\xi_2, \eta_{2,1}), (\xi_2, \eta_{2,2}), \cdots, (\xi_2, \eta_{2,t_2}) \cdots \\ & (\xi_s, \eta_{s,1}), (\xi_s, \eta_{s,2}), \cdots, (\xi_s, \eta_{s,t_s}) \big\}. \end{aligned} \tag{4}$$

We can then construct a row vector for each parity bit position $(k,l) \in \Pi$:

$$\begin{aligned} \mathbf{h}_{k,l} = \Big[& \left(\xi_1^k \eta_{1,1}^l\right) \left(\xi_1^k \eta_{1,2}^l\right) \cdots \left(\xi_1^k \eta_{1,t_1}^l\right) \\ & \left(\xi_2^k \eta_{2,1}^l\right) \left(\xi_2^k \eta_{2,2}^l\right) \cdots \left(\xi_2^k \eta_{2,t_2}^l\right) \cdots \\ & \left(\xi_s^k \eta_{s,1}^l\right) \left(\xi_s^k \eta_{s,2}^l\right) \cdots \left(\xi_s^k \eta_{s,t_s}^l\right) \Big] \end{aligned} \tag{5}$$

where each element $(\xi_i^k \eta_{i,j}^l)$, $1 \le i \le s$ and $1 \le j \le t_i$, is a $m_i n_{i,j}$-tuple. Here m_i is the degree of the minimal polynomial of ξ_i and $n_{i,j}$ is the degree of the monic minimal polynomial of $\eta_{i,j}$ over $GF(2^{m_i})$. For example, let us consider a set of zeros $V_c = \{(\gamma^1, \gamma^3), (\gamma^1, \gamma^1), (\gamma^1, \gamma^0), (\gamma^5, \gamma^5)\}$ where γ is the 15th root of the equation $x^{15} - 1 = 0$. For the first zero (γ^1, γ^3), it is obvious that the degree of the minimal polynomial for γ^1 is $m_1 = 4$. Then over $GF(2^4)$, the monic minimal polynomial for γ^3 is $(x + \gamma^3)$, whose degree is $n_{1,1} = 1$. This means that for parity bit position (k, l), the first element of (5) is $4\,(= m_1 n_{1,1})$-bit representation of $\gamma^{(k+3l)}$ on $GF(2^4)$. For position (k, l), other two zeros (γ^1, γ^1) and (γ^1, γ^0) construct two elements $\gamma^{(k+l)}$ and γ^k, respectively, which are also 4-tuple representations on $GF(2^4)$. For the zero (γ^5, γ^5), $m_2 = 2$ and $n_{2,1} = 1$ so that the corresponding element $\gamma^{(5k+5l)}$ is 2-tuple. For position $(k, l) = (3, 2)$, as an example, the corresponding row vector is:

$$\begin{aligned}\mathbf{h}_{3,2} &= \left[(\gamma^9)(\gamma^5)(\gamma^3)(\gamma^{25})\right] \\ &= [(0101)(0110)(0001)(11)] \qquad (6)\end{aligned}$$

All row vectors in Π can be constructed in the same manner. Each row vector is a 14-tuple. Moreover, Π has a size of 14 (has 14 elements), as will be clear shortly. Accordingly, 14 row vectors are constructed for the area Π and they constitute a set of basis vectors of length 14.

A vector for position $(i, j) \in \Omega - \Pi$ can also be represented using the form of (5) with $(k, l) \in \Pi$ replaced by $(i, j) \in \Omega - \Pi$. Furthermore, a linear combination of the basis vectors obtained for area Π can also represent each vector in $\Omega - \Pi$:

$$\mathbf{h}_{i,j} = \sum_{(k,l)\in\Pi} h_{k,l}^{(i,j)} \mathbf{h}_{k,l} \qquad (7)$$

where $(i, j) \in \Omega - \Pi$. Since $\mathbf{h}_{i,j}$ and $\mathbf{h}_{k,l}$ are all completely specified once V_c is given, the coefficients $h_{k,l}^{(i,j)}$, which represent the 2D feedback shift-register coefficients, can be obtained by solving a linear system of equations.

Let $[k]_x$ and $[l]_y$ be the short-hand notations for $k_{\mathrm{mod}N_x}$ and $l_{\mathrm{mod}N_y}$, respectively. All zeros $(\xi, \eta) \in V_c$ are such that $\xi^k = \xi^{[k]_x}$ and $\eta^l = \eta^{[l]_y}$ because $\xi^{N_x} = 1$ and $\eta^{N_y} = 1$. Consequently, for any integer pair (k, l), the corresponding row vector can be written as

$$\mathbf{h}_{k,l} = \mathbf{h}_{[k]_x, [l]_y}. \qquad (8)$$

2) Encoding/Decoding of 2D Cyclic Codes: We now specify the feedback connections in the 2D shift-register. First define the border areas between the parity region and the message region:

$$\Pi_{\partial x} = \{(i, j) \in \Pi | (i + 1, j) \in \Omega - \Pi\} \qquad (9)$$

$$\Pi_{\partial y} = \{(i, j) \in \Pi | (i, j + 1) \in \Omega - \Pi\} \qquad (10)$$

See Fig. 1. The gray area of a peculiar shape (consisting of a number of m_i by n_i subarrays) in the upper-left corner of the 2D array represents the parity region Π. View the 2D array also as a 2D shift-register with feedback paths given only to the positions in the parity region. Imagine the message bits entering the 2D array from the upper-left corner. The shift register is initially cleared. The entered bits initially fill the first column. The bits that have filled the first column then gets shifted to the right to the second column. New bits enter the first column again. Once filled, the bits in the first and second columns get shifted to the right by one column to make a room for new bits. This process continues until a 2D encoder buffer containing the message bits as well as $\sum_{i=1}^{s} m_i n_i$ extra zero bits completely empty itself to fill the array.

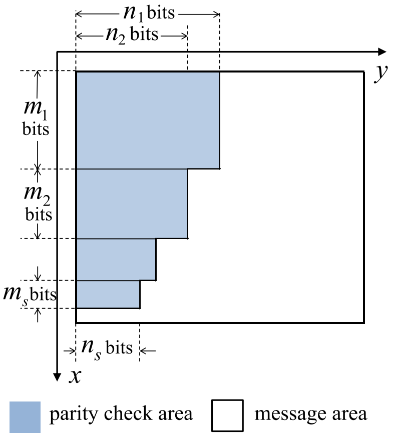

Fig. 1. Encoding for 2D cyclic code.

Let the contents of the 2D shift-register corresponding to the parity region at some point in the encoding process be written as

$$\sigma(x, y) = \sum_{(k,l)\in\Pi} \sigma_{k,l} x^k y^l. \qquad (11)$$

Assuming the message bits that are being entered have already arrived at the left (or the upper) side of the border line between the parity and message areas, a further shift in y (or x) direction will trigger feedback connection. Specifically, the new content $\sigma'_{k,l}$ for position (k, l) is determined by

$$\sigma'_{k,l} = \sigma_{k-1,l} + \sum_{(i,j)\in\Pi_{\partial x}} h_{k,l}^{(i+1,j)} \sigma_{i,j} \qquad (12)$$

after a step along x-direction. For shifting one step along y-direction, the new content $\sigma'_{k,l}$ is similarly determined by

$$\sigma'_{k,l} = \sigma_{k,l-1} + \sum_{(i,j)\in\Pi_{\partial y}} h_{k,l}^{(i,j+1)} \sigma_{i,j}. \qquad (13)$$

As the last tail zero bit enters the array, the message area is now filled with the message bits and the parity check area is occupied by the final parity bits.

Let us consider an illustrative example for feedback connection in the 2D cyclic code encoder. For a (15 × 15)-bit array code with the set of zeros $V_c = \{(\gamma^1, \gamma^3), (\gamma^1, \gamma^1), (\gamma^1, \gamma^0), (\gamma^5, \gamma^5)\}$, there exist 14 parity positions and 14 corresponding basis vectors. See Fig. 2. 225 2D shift registers are deployed in the form of a rectangular array. Assume that during a horizontal shift, the content of the register at position (3,2) is

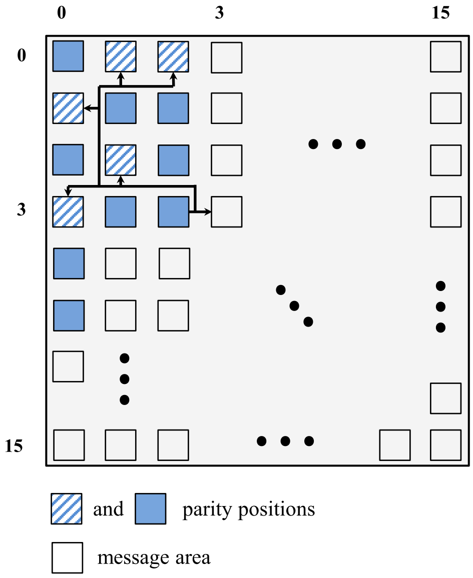

Fig. 2. Feedback in action during encoding: horizontal shifting.

shifted to the adjacent register at position (3,3). Simultaneously, the content is also fed back to certain 2D registers in the parity area. The feedback connections are uniquely determined by the coefficients $h_{k,l}^{(3,3)}$ where $(k,l) \in \Pi$. Using the definition (7), $h_{0,1}^{(3,3)} = h_{0,2}^{(3,3)} = h_{1,0}^{(3,3)} = h_{2,1}^{(3,3)} = h_{3,0}^{(3,3)} = 1$ and the coefficients corresponding to other positions $(k,l) \in \Pi$ are zero. This means that the content of the shift register at (3,2) is only fed back to the five registers at positions (0, 1), (0, 2), (1, 0), (2, 1) and (3, 0), as shown.

Consider two operators T_x and T_y such that $T_x\sigma(x,y)$ and $T_y\sigma(x,y)$ represent the contents of the shift register corresponding to one-step shifting of $\sigma(x,y)$ along the x and y directions, respectively. Then, after shifting $\sigma(x,y)$ by k positions in the x direction and by l positions in the y direction, we get $T_x^kT_y^l\sigma(x,y)$. Further defining $m(T_x,T_y)$ as the final content of the parity check area at the end of the encoding process, we can write the corresponding codeword as $c(x,y) = m(T_x,T_y) + m(x,y)$. The syndrome polynomial for an error polynomial $e(x,y)$ can be written as $e(T_x,T_y) = \sigma^{(0,0)}(x,y)$. The syndrome $e(T_x,T_y)$ can be viewed as the content of the 2D feedback shift-register corresponding to the parity check area after the $N_x \times N_y$ word $e(x,y)$ completely enters the 2D array.

B. Condition for Error-Pattern-Detecting 2D Cyclic Codes

Consider the 2D error patterns listed in Table I in polynomial forms. These are dominant error patterns in the 2D Partial Response 1 channel with additive white Gaussian noise [2]. For instance, an error pattern $e_5(x,y) = 1 + x + y + xy$ is a group of four erroneous bits occupying a 2 × 2 bit array. A geometric representation of these eight dominant error patterns occurring in some arbitrary positions in a 9 × 16 bit array is given in Fig. 3.

Let us establish some general notations. L 2D dominant error patterns can be represented using polynomials $e_i(x,y)$, $0 \le i \le L-1$. The corresponding syndrome polynomial for error pattern $e_i(x,y)$ is denoted as $e_i(T_x,T_y) = \sigma_i^{(0,0)}(x,y)$. Assume that the ith error pattern has occurred at position (k,l). The error can be represented as $x^ky^le_i(x,y)$. The resulting syndrome polynomial is $T_x^kT_y^le_i(T_x,T_y)$ and we can write: $T_x^kT_y^le_i(T_x,T_y) = \sigma_i^{(k,l)}(x,y)$. Considering all possible positions on a $N_x \times N_y$ bit array, there exist N_xN_y syndrome polynomials. Thus, for each error patten, the N_xN_y possible syndrome polynomials can be collected to make a syndrome set S_i.

$$S_i = \left\{\sigma_i^{(k,l)}(x,y) | (k,l) \in \Omega\right\}. \tag{14}$$

This is the definition of the syndrome set S_i for a predetermined 2D error pattern $e_i(x,y)$. For detecting any single occurrence of the L predetermined 2D error patterns, all L syndrome sets S_i should be completely separated without any common syndrome polynomials to one another. A mathematical condition established in this section supports the detection capability for our 2D cyclic codes. Before delving into the details, it is useful to establish the following properties (the proofs are given in the Appendix):

Property 1 (for Syndrome Polynomials): Let $e(x,y) = \sum_{(i,j)\in\Omega} e_{i,j}x^iy^j$ represent an error polynomial and $\sigma(x,y) = \sum_{(i,j)\in\Pi}\sigma_{i,j}x^iy^j$ the corresponding syndrome polynomial. Then the following equality holds based on the vector definition of (5):

$$\sum_{(i,j)\in\Omega} e_{i,j}\mathbf{h}_{i,j} = \sum_{(i,j)\in\Pi} \sigma_{i,j}\mathbf{h}_{i,j}. \tag{15}$$

Property 2 (for Syndrome Polynomials): For the same error and syndrome polynomials,

$$\sigma(x,y) = \sum_{(i,j)\in\Pi} e_{i,j}x^iy^j + \sum_{(k,l)\in\Pi}\left(\sum_{(i,j)\in\Omega-\Pi} h_{k,l}^{(i,j)}e_{i,j}\right)x^ky^l. \tag{16}$$

Property 3: For a zero (ξ,γ) in V_c,

$$\xi^i\gamma^j = \sum_{(k,l)\in\Pi} h_{k,l}^{(i,j)}(\xi^k\gamma^l) \tag{17}$$

Distinct syndrome sets guarantee detection capability, and the sufficient condition for the detection capability is supported by the following theorem [6] (for which we present a concrete proof for the first time in this paper):

Theorem 1: Let V_c be the set of zeros of a 2D cyclic code whose size is $(N_x \times N_y)$. Define the set $E_i = \{(\alpha^k,\beta^l)|e_i(\alpha^k,\beta^l) = 0, (k,l) \in \Omega\}$, $0 \le i \le L-1$, where α and β are the N_x^{th} and N_y^{th} roots of unity. Further let C_i be the intersection of E_i and V_c. If $C_i \ne C_j$ for all $i \ne j$, $0 \le i,j \le L-1$, then the syndrome sets S_i and S_j of the respective error polynomials $e_i(x,y)$ and $e_j(x,y)$ are distinct and do not share any member.

The proof is based on the following two lemmas.

Lemma 1.1: Let E be the collection of the zeros of an error polynomial $e(x,y)$. If a point (ξ,γ) is in the intersection set $C(= E \cap V_c)$, then this point is also a zero of the corresponding syndrome polynomial $e(T_x,T_y) = \sigma^{(0,0)}(x,y)$. Moreover, a zero (ξ_a,γ_a) in V_c but outside C cannot be a zero of $\sigma^{(0,0)}(x,y)$.

TABLE I
ERROR EVENTS FOR 2D PR1 CHANNEL IN POLYNOMIAL FORM

pattern no. i	$e_i(x,y)$	pattern no. i	$e_i(x,y)$	pattern no. i	$e_i(x,y)$	pattern no. i	$e_i(x,y)$
0	1	2	$1+x$	4	$1+x+x^2$	6	$1+xy$
1	$1+y$	3	$1+y+y^2$	5	$1+x+y+xy$	7	$x+y$

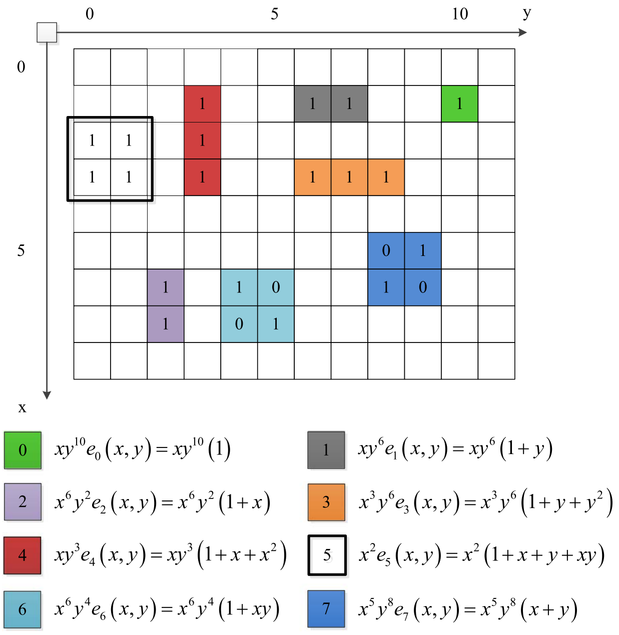
0
5
10
y
0
5
x
0 $xy^{10}e_0(x,y)=xy^{10}(1)$
1 $xy^6e_1(x,y)=xy^6(1+y)$
2 $x^6y^2e_2(x,y)=x^6y^2(1+x)$
3 $x^3y^6e_3(x,y)=x^3y^6(1+y+y^2)$
4 $xy^3e_4(x,y)=xy^3(1+x+x^2)$
5 $x^2e_5(x,y)=x^2(1+x+y+xy)$
6 $x^6y^4e_6(x,y)=x^6y^4(1+xy)$
7 $x^5y^8e_7(x,y)=x^5y^8(x+y)$

Fig. 3. Examples of 2D error events.

Lemma 1.2: Let $A^{(k,l)}$ be the collection of zeros for shifted syndrome polynomial $T_x^kT_y^l\sigma(x,y)=T_x^kT_y^l\sum_{(i,j)\in\Pi}\sigma_{i,j}x^iy^j$. For any $(k,l)\in\Omega$, the corresponding set $A^{(k,l)}$ remains fixed regardless of the shift (k,l), i.e., $A^{(k,l)}=A^{(0,0)}$.

The proofs for these two lemmas are provided in the Appendix. The two lemmas lead to the following proof for Theorem 1.

Proof (for Theorem 1): From Lemmas 1.1, the inequality $C_i\neq C_j$ guarantees that the sets of zeros for the corresponding initial syndromes $\sigma_i(x,y)$ and $\sigma_j(x,y)$ are differ by at least one element which, in turn, implies $\sigma_i(x,y)\neq\sigma_j(x,y)$. Moreover, from Lemma 1.2, all syndrome polynomials corresponding to a given syndrome set have a common collection of zeros. Therefore, the condition $C_i\neq C_j$ guarantees that the syndrome sets associated with $e_i(x,y)$ and $e_j(x,y)$ are completely separated from each other without any common syndromes. □

C. Example of Error-Pattern-Detecting 2D Cyclic Codes

We now construct a specific code. Let us target the dominant eight error patterns of Table I. Consider a 2D cyclic code with size $(2^6-1,2^6-1)=(63,63)$. Let α be the primitive 63^{th} root of unity, i.e., $\alpha^{63}=1$. For a particular error polynomial $e_i(x,y)$, its zeros are given by

$$E_i=\left\{(\alpha^k,\alpha^l)|e_i(\alpha^k,\alpha^l)=0,\,(k,l)\in\Omega\right\}. \tag{18}$$

Fig. 4 shows all eight collections of zeros E_i, $0\leq i\leq 7$. In this figure the label (21, 0) is short for (α^{21},α^0), for instance. Moreover, different colors are used to visually distinguish the sets E_i. The additional superscript i on the left-upper corner of each box also identifies the set it belongs to. Note that some zeros like $(\alpha^{21},\alpha^{21})$ belong to multiple sets.

First, to achieve the designed error detecting capability, the members of the code's set of zeros, V_c, should be carefully selected to satisfy the condition of Theorem 1. When we select a zero, only one zero needs be selected from one conjugate set because zeros from the same conjugate set show the same effect in the view of Theorem 1.

Among multiple choices for V_c satisfying Theorem 1, it makes sense to seek zeros with a minimum number of parity bits. Because we should distinguish the eight sets $C_i=E_i\cap V_c$, V_c should have at least eight subsets. This means that V_c should have at least three members. To distinguish all eight subsets with a three-member set, say, $A=\{a_1,a_2,a_3\}$, every one of the three members should appear in four different subsets. However (α^0,α^0) is the only zero that appears in more than four of the eight sets E_i. Therefore, in this case, at least four zeros are needed to construct V_c in the view of Theorem 1. We can easily see that (α^0,α^0) costs only one bit for inclusion in V_c. Moreover, each of the eight zeros, (α^{21},α^0), (α^0,α^{21}), $(\alpha^{21},\alpha^{21})$, $(\alpha^{21},\alpha^{42})$, (α^{42},α^0), (α^0,α^{42}), $(\alpha^{42},\alpha^{21})$ and $(\alpha^{42},\alpha^{42})$, costs two bits for inclusion in V_c. The zero (α^0,α^0) forms a conjugate point set by itself. On the other hand, the other eight zeros requiring two parity bits each are collected from the following conjugate point sets:

$$\begin{aligned}U_0&=\left\{(\alpha^{21},\alpha^0),(\alpha^{42},\alpha^0)\right\}\\U_1&=\left\{(\alpha^0,\alpha^{21}),(\alpha^0,\alpha^{42})\right\}\\U_2&=\left\{(\alpha^{21},\alpha^{21}),(\alpha^{42},\alpha^{42})\right\}\\U_3&=\left\{(\alpha^{21},\alpha^{42}),(\alpha^{42},\alpha^{21})\right\}\end{aligned} \tag{19}$$

For the three conjugate point sets U_0, U_1 and U_2, (α^{21},α^0), (α^0,α^{21}) and $(\alpha^{21},\alpha^{21})$ should be the representative points based on a complexity consideration. Let us define the computation factor of a zero (α^a,α^b) as $J\{(\alpha^a,\alpha^b)\}=a+b$, which is the computational complexity measure of the zero being included in V_c. For the conjugate point set U_3, either member can be chosen. Therefore, selecting the zero (α^0,α^0) and additional three zeros from the four representative points of U_i, $0\leq i\leq 3$, would give a 2D cyclic code with the smallest number of parity bits, which is exactly 7. Among all valid combinations,

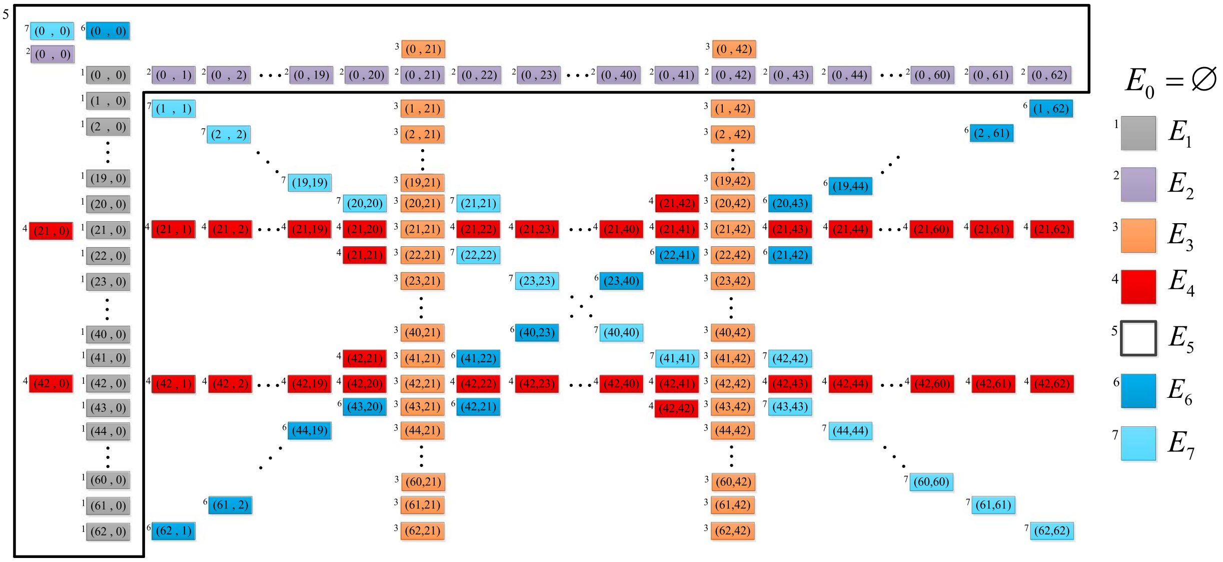

Fig. 4. Collection of sets of zeros E_i, $0 \le i \le 7$.

only the following three combinations satisfy the condition of Theorem 1:

$$V_{c_0} = \left\{(\alpha^0, \alpha^0), (\alpha^{21}, \alpha^0), (\alpha^{21}, \alpha^{42}), (\alpha^0, \alpha^{21})\right\}$$

$$V_{c_1} = \left\{(\alpha^0, \alpha^0), (\alpha^{21}, \alpha^0), (\alpha^{42}, \alpha^{21}), (\alpha^0, \alpha^{21})\right\}$$

$$V_{c_2} = \left\{(\alpha^0, \alpha^0), (\alpha^{21}, \alpha^0), (\alpha^{21}, \alpha^{21}), (\alpha^0, \alpha^{21})\right\} \quad (20)$$

Implementing a code with V_{c_0} or V_{c_1} is more complex than with V_{c_2} due to the presence of α^{42}. Thus, V_{c_2} is preferred over V_{c_0} and V_{c_1}. It can easily be confirmed that V_{c_2} satisfies the condition of Theorem 1. The corresponding intersection sets $C_i's$ are:

$$\begin{aligned} C_0 &= \varnothing, \\ C_1 &= \left\{(\alpha^0, \alpha^0), (\alpha^{21}, \alpha^0)\right\}, \\ C_2 &= \left\{(\alpha^0, \alpha^0), (\alpha^0, \alpha^{21})\right\}, \\ C_3 &= \left\{(\alpha^0, \alpha^{21}), (\alpha^{21}, \alpha^{21})\right\}, \\ C_4 &= \left\{(\alpha^{21}, \alpha^0), (\alpha^{21}, \alpha^{21})\right\}, \\ C_5 &= \left\{(\alpha^0, \alpha^0), (\alpha^{21}, \alpha^0), (\alpha^0, \alpha^{21})\right\}, \\ C_6 &= \left\{(\alpha^0, \alpha^0)\right\}, C_7 = \left\{(\alpha^0, \alpha^0), (\alpha^{21}, \alpha^{21})\right\} \end{aligned} \quad (21)$$

All sets are distinct and the condition of Theorem 1 is satisfied. The code requires 7 parity bits among a total of 63 × 63 = 3969 bits, yielding a very high code rate of approximately 0.998.

With this example code, any single occurrence of the error patterns in Table I can be detected by computing the syndrome. However, we also observe that within a given syndrome set, some syndrome polynomials appear at multiple positions. For example, take the syndrome set corresponding to the error polynomial $e_0(x, y) = 1$. This syndrome set exhibits a pattern of repetition. The same syndrome values reappear as +3 steps are taken along the y direction for example, as can be seen from the fact that $\sigma_0^{(0,0)} = 1$, $\sigma_0^{(0,3)} = 1$, $\sigma_0^{(0,6)} = 1$ and so forth. Additionally, all syndrome values reappear with shifts of +3 along the x direction as well. We shall say that the syndrome set for $e_0(x, y) = 1$ has a *period* of (3, 3). This means that while computing the syndrome would correctly identify the error pattern, it will not be able to determine its position of occurrence with full accuracy.

Likewise, other error patterns, $e_1(x, y) = 1 + y$, $e_2(x, y) = 1 + x$, $e_6(x, y) = 1 + xy$ and $e_7(x, y) = x + y$, also have their syndrome sets with a periodicity of (3, 3). For the error pattern $e_3(x, y) = 1 + y + y^2$, shifting 3 along the x direction gives the same syndrome value while shifting along the y direction does not change the syndrome value for any amount of shift. Thus, the syndrome set for $e_3(x, y)$ is said to be periodic with period (3, 1). Along the same line, $e_4(x, y) = 1 + x + x^2$ has its syndrome set with a (1, 3) period. For $e_5(x, y) = 1 + x + y + xy$, the syndrome values reappear repeatedly at positions $(3k - (l_{\mathrm{mod}3}), l)$ for $1 \le k \le 21$ and $0 \le l \le 62$.

From the above discussions, it is clear that to attain perfect error correction capability of the prescribed error patterns, all syndrome sets must have a period equal to the size of the given code array, namely, (N_x, N_y). We shall say that in this case, the syndrome sets have *full* periods. This condition would allow one to pin down on the exact position of the identified error pattern. Next, we discuss this additional condition related to the full periodicity of the syndrome sets.

III. Construction of Error-Pattern-Correcting 2D Cyclic Codes

A. Condition for Error-Pattern-Correcting 2D Cyclic Codes

Definition: Let Ω' be Ω excluding the single point (0,0). A 'period set' PS_i is defined as

$$PS_i = \left\{(P_x, P_y) \in \Omega' | \sigma_i^{(0,0)}(x, y) = \sigma_i^{(P_x, P_y)}(x, y)\right\} \quad (22)$$

Theorem 2: Consider a 2D cyclic code having V_c as its set of zeros. Let E_i be the collection of zeros for a predetermined error pattern polynomial $e_i(x, y)$. Then, $(P_x, P_y) \in \Omega'$ belongs to PS_i, if and only if $\xi^{P_x}\gamma^{P_y} = 1$ for all zeros (ξ, γ) in $V_c \cap E_i^c$.

This theorem basically says that the full position information on the occurred error event can be obtained by designing V_c such that the period set corresponding to the given error pattern is empty, i.e., the code with empty period sets is the "full-period code" for which all the syndrome sets have full periods. To facilitate the proof, consider the following lemmas.

Lemma 2.1: Let $\sigma_i(x, y) = \sum_{(k,l)\in\Pi} \sigma_{i,k,l}x^k y^l$ be the syndrome polynomial corresponding to a predetermined error pattern $e_i(x, y) = \sum_{(k,l)\in\Omega} e_{i,k,l}x^k y^l$. Then, for an arbitrary shift $(P_x, P_y) \in \Omega'$, we have

$$\sum_{(k,l)\in\Omega} e_{i,k,l}\mathbf{h}_{[k+P_x]_x,[l+P_y]_y} = \sum_{(k,l)\in\Pi} \sigma_{i,k,l}\mathbf{h}_{[k+P_x]_x,[l+P_y]_y}. \quad (23)$$

Lemma 2.2: A certain shift $(P_x, P_y) \in \Omega'$ belongs to the period set PS_i if and only if

$$\sum_{(k,l)\in\Omega} e_{i,k,l}\mathbf{h}_{k,l} = \sum_{(k,l)\in\Omega} e_{i,k,l}\mathbf{h}_{[k+P_x]_x,[l+P_y]_y}. \quad (24)$$

The proofs of the two lemmas are given in the Appendix. Using the above two lemmas, we are now ready to prove Theorem 2.

Proof (for Theorem 2): From the two lemmas given above, we know that (P_x, P_y) is in PS_i if and only if

$$\sum_{(k,l)\in\Omega} e_{i,k,l}\mathbf{h}_{k,l} = \sum_{(k,l)\in\Omega} e_{i,k,l}\mathbf{h}_{[k+P_x]_x,[l+P_y]_y}. \quad (25)$$

Remind the definition (5) for row vectors $\mathbf{h}_{k,l}$ in Π and its extension to the message positions in $\Omega - \Pi$. By focusing on a zero (ξ, γ) in $V_c \cap E_i^c$ and the corresponding element $(\xi^k \gamma^l)$ in the row vector $\mathbf{h}_{k,l}$, (25) becomes

$$\sum_{(k,l)\in\Omega} e_{i,k,l}(\xi^k \gamma^l) = \sum_{(k,l)\in\Omega} e_{i,k,l}(\xi^{[k+P_x]_x}\gamma^{[l+P_y]_y}) \quad (26)$$

which holds if and only if $\xi^{P_x}\gamma^{P_y}$ equals one. Summarizing, we must have

$$\forall(\xi, \gamma) \in V_c \cap E_i^c, \xi^{P_x}\gamma^{P_y} = 1 \quad (27)$$

in order to satisfy (26). All zeros in $V_c \cap E_i$ naturally satisfy (26) and thus (25). Therefore, a certain shift $(P_x, P_y) \in \Omega'$ is in the period set PS_i if and only if for all zeros (ξ, γ) in $V_c \cap E_i^c$, $\xi^{P_x}\gamma^{P_y} = 1$. □

Theorem 2 directly gives rise to the following corollary:

Corollary 1: The given 2D cyclic code can correct all single occurrences of known error patterns $e_i(x, y)$, $0 \leq i \leq L - 1$, at any position, if and only if for every error pattern $e_i(x, y)$, PS_i is empty.

B. Example of Error-Pattern-Correcting 2D Cyclic Codes

Before constructing an example error-pattern-correcting 2D cyclic code, it is instructive to examine the periods of the code already designed in the previous section, in light of the periodic properties established by Theorem 2. Recall that the code detects the predetermined eight error patterns in Table I. The set of zeros of the constructed 2D cyclic code is

$$V_c = \left\{(\alpha^0, \alpha^0), (\alpha^{21}, \alpha^0), (\alpha^{21}, \alpha^{21}), (\alpha^0, \alpha^{21})\right\}. \quad (28)$$

For the first syndrome set for the error pattern $e_0(x, y) = 1$, the intersection set between E_0^c and V_c is

$$V_c \cap E_0^c = \left\{(\alpha^0, \alpha^0), (\alpha^{21}, \alpha^0), (\alpha^{21}, \alpha^{21}), (\alpha^0, \alpha^{21})\right\}. \quad (29)$$

The first element, (α^0, α^0), is shown to satisfy the equality $\alpha^{0\times P_x}\alpha^{0\times P_y} = 1$ for any shift (P_x, P_y). For the second one, (α^{21}, α^0), the equality $\alpha^{21\times P_x}\alpha^{0\times P_y} = 1$ holds for any shift along the y direction but for only the shifts in multiples of 3 steps along the x direction. The next member $(\alpha^{21}, \alpha^{21})$ satisfies $\alpha^{21\times P_x}\alpha^{21\times P_y} = 1$ only for the shifts such that $P_x + P_y$ is a multiple of 3. For the last zero, (α^0, α^{21}), the result is similar to the second zero except for switching the x direction to y. Overall, only the shifts by multiples of 3 steps along x or y direction satisfies the condition of Theorem 2, meaning that the syndrome set for E_0 has a period set consisting only of $(P_x = 3k, P_y = 3l)$, $1 \leq k, l < 21$.

This code is now refined to get correction capability. By adding (α^1, α^{62}) to V_c and exchanging $\{(\alpha^0, \alpha^0), (\alpha^{21}, \alpha^0)\}$ with $\{(\alpha^1, \alpha^1), (\alpha^1, \alpha^0)\}$, we obtain a refined set of zeros V_f:

$$V_f = \left\{(\alpha^1, \alpha^{62}), (\alpha^1, \alpha^0), (\alpha^1, \alpha^1), (\alpha^{21}, \alpha^{21}), (\alpha^0, \alpha^{21})\right\} \quad (30)$$

which can easily be confirmed to have full correction capability for any single occurrence of the target error patterns via Corollary 1. This conclusion can be easily confirmed by investigating the intersections $E_i^c \cap V_f$, $0 \leq i \leq 7$.

$$\begin{aligned}
E_0^c \cap V_f &= V_f\\
E_1^c \cap V_f &= \left\{(\alpha^1, \alpha^{62}), (\alpha^1, \alpha^1), (\alpha^{21}, \alpha^{21}), (\alpha^0, \alpha^{21})\right\}\\
E_2^c \cap V_f &= \left\{(\alpha^1, \alpha^{62}), (\alpha^1, \alpha^0), (\alpha^1, \alpha^1), (\alpha^{21}, \alpha^{21})\right\}\\
E_3^c \cap V_f &= \left\{(\alpha^1, \alpha^{62}), (\alpha^1, \alpha^0), (\alpha^1, \alpha^1)\right\}\\
E_4^c \cap V_f &= \left\{(\alpha^1, \alpha^{62}), (\alpha^1, \alpha^0), (\alpha^1, \alpha^1), (\alpha^0, \alpha^{21})\right\}\\
E_5^c \cap V_f &= \left\{(\alpha^1, \alpha^{62}), (\alpha^1, \alpha^1), (\alpha^{21}, \alpha^{21})\right\}\\
E_6^c \cap V_f &= \left\{(\alpha^1, \alpha^0), (\alpha^1, \alpha^1), (\alpha^{21}, \alpha^{21}), (\alpha^0, \alpha^{21})\right\}\\
E_7^c \cap V_f &= \left\{(\alpha^1, \alpha^{62}), (\alpha^1, \alpha^0), (\alpha^0, \alpha^{21})\right\}. \quad (31)
\end{aligned}$$

Taking the error pattern $e_0(x, y) = 1$ for example, the period set PS_0 corresponding to the refined intersection set $E_0^c \cap V_f = V_f$ is now empty. This means that the syndrome set S_0 contains $63 \times 63 = 3969$ distinct syndromes matched to all possible error positions. Therefore, any single occurrence of the error pattern $e_0(x, y) = 1$ can be corrected by the code. Similar arguments apply to all other syndrome sets and it can be easily shown that they all have the full period.

IV. General Algorithm for Constructing Cyclic Codes With Correction Capability for Targeted Error Patterns

We present a general algorithm for constructing a code that satisfies the full-periodicity condition while maintaining computational as well as code rate efficiency. The approach is basically an iterative algorithm that starts with the construction of distinct intersection sets (as stated as the condition of Theorem 1) under the minimum parity constraint. In achieving the full syndrome periodicity, the suggested procedures basically refine the set of zeros V_c by eliminating and replacing the zeros one at a time. The iterative procedure eventually yields a set of zeros that guarantees the full periodicity of all the syndrome sets.

We first establish several guidelines as well as notations. The guidelines are for dropping or adding zeros from the given V_c.

Guideline 1- An Essential Set

At the kth iteration, starting with an intermediate set of zeros V_{i_k}, suppose we are to drop one of its element for refinement. This choice should be made carefully because the absence of a certain zero $(\xi, \gamma) \in V_{i_k}$ would make it impossible to satisfy the condition of Theorem 1 even with any additional zeros. Moreover, zeros which have been newly introduced during previous refining iterations also should be protected. Such zeros are called *essential* and should not be dropped during the kth iteration. Denote such essential sets as $ES(V_{i_k})$. We know that $ES(V_{i_k}) \subseteq V_{i_k}$. For the case $ES(V_{i_k}) = V_{i_k}$, we forgo dropping a zero and add the simplest candidate zero to V_{i_k}. This type of additional zeros without the dropping step also be in $ES(V_{i_l})$ for $l = k+1, k+2, \cdots$.

Guideline 2 - Period Set

Definition: For a given point (ξ, γ), define the period set $PS_{(\xi,\gamma)}$ associated with it:

$$PS_{(\xi,\gamma)} = \left\{(k, l) \in \Omega' | \xi^k \gamma^l = 1\right\},$$

where Ω' is as defined in the Section III-A.

A zero in V_{i_k} having a large period set is not desirable as it gives rise to significant error position ambiguity initially. Accordingly, zeros with large period sets are the first candidates for elimination. By the same token, a zero with a small period set would be a preferred choice as replacement.

Guideline 3 - Minimizing the Number of Additional Zeros

Let D_{i_k} denote the zero (ξ, γ) chosen for elimination from V_{i_k}. To achieve a higher code rate, the preferred choice for D_{i_k} should be the one that would require a minimum number of additional zeros for satisfying the condition of Theorem 1. Moreover, the conjugate points of zeros in V_{i_k} or the conjugate points of already dropped zeros, $D_{i_0}, \cdots, D_{i_{k-1}}$, should not be considered as additional zeros. Denote the possible sets of additional zeros as $AS\{D_{i_k}, V_{i_k}\}_n$ where the index $n = 1, 2, \cdots$ corresponds to a particular choice. We can always think of such sets as long as $D_{i_k} \notin ES(V_{i_k})$. Let $|AS\{D_{i_k}, V_{i_k}\}|$ denote the minimum number of additional zeros needed to satisfy the Theorem 1's condition when we dropped the zero $D_{i_k} = \{(\xi, \gamma)\}$ from V_{i_k}. Minimizing $|AS\{D_{i_k}, V_{i_k}\}|$ is a secondary criterion.

Guideline 4 - Minimizing the Computation Factor

A zero with a large computation factor should be replaced by one with a lower computation factor. As we drop a zero and add one, we should try to minimize the difference in the computation factor

$$\Delta J = \left(\sum J_a\right) - J_e \tag{32}$$

where $\sum J_a$ is the total computation factor for all additional zeros and J_e is for the eliminated zero. For $AS\{D_{i_k}, V_{i_k}\}_n$, the corresponding delta is written as

$$\Delta J\left(AS\{D_{i_k}, V_{i_k}\}_n\right) = J\left(AS\{D_{i_k}, V_{i_k}\}_n\right) - J(D_{i_k}). \tag{33}$$

Based on these definitions and guidelines, Algorithm 1 has been constructed as listed below.

Algorithm 1 General Algorithm for Constructing Full-Period Codes

```
Input: $V_c$ of the minimum parity code satisfying the condition of Theorem 1
Output: $V_f$ of full-period codes
Initialization: Set $k = 0$ and $V_{i_0} = V_c$.
while $V_{i_k}$ is not of a full-period code do
    Step 1: If $ES(V_{i_k}) = V_{i_k}$, go to Step 6. Otherwise, among zeros in $V_{i_k} \cap ES(V_{i_k})^c$, find ones with the largest period set. If there is only one such zero, go to Step 5, otherwise proceed to Step 2.
    Step 2: Among the qualified candidates from Step 1, collect ones which need the least number of additional zeros for satisfying Theorem 1's condition when eliminated. If there is only one such zero, go to Step 5, otherwise go to Step 3.
    Step 3: Among candidates from Step 2, collect ones for which, if eliminated, the replaced additional zeros result in the least sum of the period set sizes. If there is only one such zero, go to Step 5, otherwise go to Step 4.
    Step 4: Among candidates from Step 3, collect ones which minimize the computation factor when eliminated and the minimum number of additional zeros are chosen. Among such zeros, choose one, label it as $(\xi, \gamma)$ and go to Step 5.
    Step 5: Given the zero $(\xi, \gamma)$ to drop, first find the choices with the minimum number of additional zeros for
```

```
        satisfying Theorem 1's condition. Among those
        choices, find ones with the least additional period
        set sizes. Among these qualified choices, find ones
        with the minimum additional computation factor.
        Among the final choices, take one choice and add
        the additional zeros to $V_{i_k}$ and drop the zero $(\xi, \gamma)$.
        Set $k = k + 1$ and skip Step 6.
    Step 6: Find the error pattern $l$ which shows the largest
        value of $|PS_l|$. Among the zeros in $E_l^c$ that are not
        conjugates of any zeros of $V_{i_k}$, collect ones with
        the minimum period set size and, if more than one,
        select any with the minimum computation factor.
        Add the selected zero to $V_{i_k}$. Set $k = k + 1$.
end while
Return $V_f = V_{i_k}$
```

As in any iterative search algorithm, convergence is a critical issue here and needs to be analyzed carefully. Fortunately, the convergence property of Algorithm 1 is easily confirmed when the procedures are examined carefully. In each iteration, as long as the updated set of zeros V_{i_k} is not completely filled with essential zeros, refinement to V_{i_k} takes place from **Step 2** through **Step 5**, i.e., the steps of selecting a zero to discard and choosing new additional zeros for inclusion in V_{i_k}. In almost all cases, the algorithm terminates after enough iterations with a set of zeros guaranteeing full-periodicity. In rare cases it does not, the resulting V_{i_k} will always be filled with essential zeros. At this point, the process moves to **Step 6** for adding a new zero without dropping any. In this step, the additional zero should be chosen only from set E_l^c for which $e_l(x, y)$ gives the maximum syndrome period set size $|PS_l|$. This criterion always increase the size of $E_l^c \cap V_{i_k}$ so that the period set size $|PS_l|$ of syndromes for $e_l(x, y)$ is monotonically non-increasing. After repeated runs of **Step 6**, one of two things happens. Either the full-periodicity solution is attained or all zeros will have been exhausted without yielding V_f with full periodicity (meaning not all targeted error patterns can be corrected for the given code array dimension). Since each execution of **Step 6** adds a new zero without dropping any and there exist a finite number of zeros available, it is clear that eventually all available zeros will be inspected.

For illustrative purposes, consider an example in which actual code is constructed using Algorithm 1. Start with the initial set of zeros V_c obtained in Section II-C satisfying the minimum parity requirement, which we reproduce here:

$$V_c = \left\{(\alpha^0, \alpha^0), (\alpha^{21}, \alpha^0), (\alpha^{21}, \alpha^{21}), (\alpha^0, \alpha^{21})\right\}. \tag{34}$$

Initialization: Set $k = 0$ and $V_{i_0} = V_c$ for initialization.

Loop condition: Obviously, V_{i_0} is not of full-period.

Step 1: $ES(V_{i_0}) = \emptyset$.

Only (α^0, α^0) shows the maximum period set size of 3969. Go to **Step 5**.

Step 5: With the elimination of (α^0, α^0), the resulting three-member set gives rise to an unacceptable situation:

$$C_0 = C_6 = \emptyset. \tag{35}$$

To resolve this conflict, we should consider a new zero from E_6. Among all zeros in E_6, the following zeros are associated with the minimum period set size of 63:

$$(\alpha^l, \alpha^{63-l}) \tag{36}$$

where $l = 1, 2, 4, 5, 8, 10, 11, 13, 16, 17, 19, 20, 22, 23, 25, 26, 29, 31, 32, 34, 37, 38, 40, 41, 43, 44, 46, 47, 50, 52, 53, 55, 58, 59, 61$ and 62. These candidate zeros all have the same computation factor of 63. Make an arbitrary choice for (α^1, α^{62}).

Drop (α^0, α^0) from V_{i_0} and add (α^1, α^{62}) for V_{i_1}:

$$V_{i_1} = \left\{(\alpha^1, \alpha^{62}), (\alpha^{21}, \alpha^0), (\alpha^{21}, \alpha^{21}), (\alpha^0, \alpha^{21})\right\}. \tag{37}$$

Set $k = 1$ and skip **Step 6**.

Loop condition: Inspect the period of the syndrome sets from V_{i_1}. All error patterns cannot be perfectly corrected. V_{i_1} cannot yield a full-period code yet.

For the second iteration,

Step 1: $ES(V_{i_1}) = \{(\alpha^1, \alpha^{62})\}$.

All three members in $V_{i_1} \cap ES(V_{i_1})^c$ show the same period set size of 1323. Go to **Step 2**.

Step 2:

Scenario 1 - Case of eliminating (α^{21}, α^0):

The following three conflicts occur.

$$\begin{aligned} C_0 &= C_1 = \emptyset \\ C_2 &= C_5 = \left\{(\alpha^0, \alpha^{21})\right\} \\ C_4 &= C_7 = \left\{(\alpha^{21}, \alpha^{21})\right\} \end{aligned} \tag{38}$$

For resolving the first conflict, we need one of the zeros from E_1 regardless of the other conflicts, because E_0 is an empty set. Moreover, with any zero from E_1, the second conflict can also be resolved because E_1 is a subset of E_5. Therefore, with a new zero for overcoming the third conflict, we need at least two additional zeros.

Scenario 2 - Case of eliminating (α^0, α^{21}):

The following three conflicts occur.

$$\begin{aligned} C_0 &= C_2 = \emptyset \\ C_1 &= C_5 = \left\{(\alpha^{21}, \alpha^0)\right\} \\ C_3 &= C_7 = \left\{(\alpha^{21}, \alpha^{21})\right\} \end{aligned} \tag{39}$$

For resolving the first conflict, we need one of the zeros from E_2 regardless of the other conflicts, because E_0 is an empty set. Moreover with any zero from E_2, the second conflict can also be resolved because E_2 is a subset of E_5. Therefore, with a new zero for overcoming the third conflict, we need at least two additional zeros.

Scenario 3 - Case of eliminating $(\alpha^{21}, \alpha^{21})$:
The following three conflicts occur.

$$C_0 = C_7 = \emptyset$$
$$C_1 = C_4 = \left\{(\alpha^{21}, \alpha^0)\right\}$$
$$C_2 = C_3 = \left\{(\alpha^0, \alpha^{21})\right\} \tag{40}$$

For resolving the first conflict, we need one of the zeros from E_7 regardless of the other conflicts, because E_0 is an empty set. For the new zero, we cannot select $(\alpha^{42}, \alpha^{42})$ because it is a conjugate point of $(\alpha^{21}, \alpha^{21})$, which has already been eliminated. Therefore, we can easily see that all remaining candidates in E_7 do nothing to help resolving other conflicts. Let us consider the second conflict. When we select a zero in E_1, we need one more zero for resolving the third conflict because E_1 does not share any elements with E_2 and E_3 except for the eliminated zero (α^0, α^0). On the other hands, for selecting a zero in E_4, if we select $(\alpha^{21}, \alpha^{42})$ or $(\alpha^{42}, \alpha^{21})$ then the zeros resolve the third conflict also.

Considering above points, all the candidates should select two additional zeros. Go to **Step 3**.

Step 3: (incorporating **Step 4:**)

Scenario 1 - Case of eliminating (α^{21}, α^0):

As we already mentioned in **Step 2**, we should select a zero of E_1 with the smallest period set size of 63. Among all candidates, (α^1, α^0) shows the smallest computation factor of 1. Consider one additional zero for resolving the conflict $C_4 = C_7$. If we select a new zero from E_4, (α^{21}, α^1) gives the minimum period set size as well as computation factor. On the other hand, if we select a new zero from E_7, (α^1, α^1) should be selected. Therefore, it is better to select a new zero from E_7. Overall, we should select (α^1, α^0) and (α^1, α^1) as two additional zeros.

Scenario 2 - Case of eliminating (α^0, α^{21}):

Following the same argument as in Scenario 1 above, we should select two additional zeros (α^0, α^1) and (α^1, α^1) for satisfying Theorem 1.

Scenario 3 - Case of eliminating $(\alpha^{21}, \alpha^{21})$:

As we already mentioned in **Step 3**, we should select a zero from E_7 first. The zero with the smallest period set size and computation factor is (α^1, α^1). Next, one of $(\alpha^{21}, \alpha^{42})$ and $(\alpha^{42}, \alpha^{21})$ should be selected. However, the period set size for them is too large so that eliminating $(\alpha^{21}, \alpha^{21})$ do not offer advantage over the previous two candidates.

Consequently, Scenarios 1 and 2 give the same total period set size for the additional zeros, namely, $63 + 63 = 126$ as well as the same computation factor of -18. Therefore, let us arbitrary choose one of (α^{21}, α^0) and (α^0, α^{21}) as the zero to be dropped. Choose (α^{21}, α^0) and go to **Step 5**.

Step 5:

From the previous investigation, we can conclude that the zero to be dropped from V_{i_1} is (α^{21}, α^0) and two additional zeros $\{(\alpha^1, \alpha^0), (\alpha^1, \alpha^1)\}$ should be included in V_{i_2}. Then

$$V_{i_2} = \left\{(\alpha^1, \alpha^{62}), (\alpha^1, \alpha^0), (\alpha^1, \alpha^1), (\alpha^{21}, \alpha^{21}), (\alpha^0, \alpha^{21})\right\}. \tag{41}$$

Set $k = 2$ and skip the **Step 6**.

Loop condition: Check the periods of the syndrome set of code from V_{i_2}. It is shown to have full-period syndrome sets so that all seven target error patterns can be corrected at any position on the 63 × 63 bit array.

Return: This step terminates the loop and returns $V_f = V_{i_2}$.

$$V_f = V_{i_2} = \left\{(\alpha^1, \alpha^{62}), (\alpha^1, \alpha^0), (\alpha^1, \alpha^1), (\alpha^{21}, \alpha^{21}), (\alpha^0, \alpha^{21})\right\} \tag{42}$$

In fact, all sets of zeros with the following forms

$$V_c = \left\{(\alpha^k, \alpha^{63-k}), (\alpha^1, \alpha^0), (\alpha^1, \alpha^1), (\alpha^{21}, \alpha^{21}), (\alpha^0, \alpha^{21})\right\} \tag{43}$$

$$V_c = \left\{(\alpha^k, \alpha^{63-k}), (\alpha^0, \alpha^1), (\alpha^1, \alpha^1), (\alpha^{21}, \alpha^{21}), (\alpha^{21}, \alpha^0)\right\}, \tag{44}$$

where $k = 1, 2, 4, 5, 8, 10, 11, 13, 16, 17, 19, 20, 22, 23, 25, 26, 29, 31, 32, 34, 37, 38, 40, 41, 43, 44, 46, 47, 50, 52, 53, 55, 58, 59, 61$ and 62, yield full-period codes. The sets of zeros (44) is the result for the choice of (α^0, α^{21}) rather than (α^{21}, α^0) as the dropped zero in the second iteration. They all require 22 parity bits and achieve a code rate of 0.994. The resulting set of zeros V_f for full-period code is exactly the same as the set of zeros (30) which is already confirmed to have a full-period code.

Before ending this section, we briefly comment on the complexity of Algorithm 1. The algorithm complexity grows with N_x and N_y obviously, although we have not explored exact functional dependence. As the size of code array increases, more candidates for the set of zeros will have to be scanned. Nevertheless, the search process takes minimal computer time for any practically meaningful values of N_x and N_y and does not create any difficulties in the code design. Also, note that the proposed coding scheme corrects only a single error event in a given array, so to avoid multiple error events, it is expected that a large data array will be broken into a number of smaller subarrays to each of which coding is applied separately. As for the encoding/decoding complexity, given that the proposed codes are cyclic codes, implementation complexity is not expected to be an issue.

V. Performance Comparison With Other Error Correcting Codes

In this section, we compare the performance of our codes with other error correcting codes. When the error rate performance is dominated by a few large error events, our error-pattern-correcting code performs better than general MDS random error correcting codes and existing 2D burst-correcting codes.

As an example of the channel environments where dominant error events exist, take the high density perpendicular recording system with the channel response $h(D) = 5 + 6D - D^3$ [3].

TABLE II
2D Error Patterns With the Minimum Distance for a Multi-Head Detection Scheme With $h(D) = 5 + 6D - D^3$

Form of the error pattern	Error Polynomial
$\pm\begin{bmatrix}+2 & -2 & +2\\ -2 & +2 & -2\end{bmatrix}$	$(1+x)(1+y+y^2)$
$\pm\begin{bmatrix}+2 & -2 & +2 & -2\\ -2 & +2 & -2 & +2\end{bmatrix}$	$(1+x)(1+y+y^2+y^3)$
$\pm\begin{bmatrix}+2 & -2 & +2 & -2+2\\ -2 & +2 & -2 & +2-2\end{bmatrix}$	$(1+x)(1+y+y^2+y^3+y^4)$
$\vdots$	$\vdots$

This is a traditional 1D ISI channel with dominant input error patterns taking the form:

$$\pm\,[+2, -2, +2, \cdots] \tag{45}$$

assuming each input bit takes the value $+1$ or -1. This type of alternating error patterns gives rise to minimum distance error events at the output of the maximum-likelihood channel sequence detector.

When using a multi-track sensor, 2D ISI arises in magnetic recording. It has been shown that minimum-distance 2D error events can be obtained from 1D minimum-distance events [5]. Let $\mathbf{s}_l = F(\mathbf{a}_l)$ denote the 1D ISI channel output corresponding to the input symbol sequence $\mathbf{a}_l$, $1 \le l \le N$, for the l^{th} track. Then, for a multi-head detection scheme, the channel output suffers from inter-track-interference (ITI) in addition to ISI and can be expressed as

$$\mathbf{y}_k = \alpha\mathbf{s}_{k-1} + \mathbf{s}_k + \alpha\mathbf{s}_{k+1} + \mathbf{w}_k \tag{46}$$

using some constant $0 \le \alpha \le 1$ and $1 \le k \le N$. The AWGN vector noise is captured in $\mathbf{w}_k$. The results of [5] show that for $1 - 1/\sqrt{2} \le \alpha \le \gamma_1 (\approx 0.389)$, the error patterns for the minimum distance output are strongly correlated over a pair of parallel tracks. Let e_1 and e_2 be 1D minimum distance error events for two adjacent tracks. Then, the minimum distance event arises for the 2D multi-head channel when

$$F(e_2) = -F(e_1). \tag{47}$$

Accordingly, for the 2D multi-detection model based on a 1D ISI channel of $h(D) = 5 + 6D - D^3$, 2D error patterns with minimum distance can be specified. The resulting 2D input error patterns take the form shown in Table II.

The error patterns in Table II can be quite large. Attempting to correct these error patterns using general random error correcting codes may not be efficient. We show that the present error-pattern correcting codes perform excellently for this type of channel. First, assume a 15 × 15 bit array. Let us attempt to correct error patterns of the form in Table II up to length 7 bits, i.e., 5 patterns with the horizontal length of 3 to 7, expressed in polynomial form as $e_i(x, y) = (1 + x)(1 + y + \cdots + y^{(i+2)})$, $0 \le i \le 4$. We note that imposing an upper limit on the length of repetitive error patterns along a given track is quite common in storage by means of modulation coding without incurring significant rate penalty [19], [20].

Let γ be the primitive 15th root of unity, i.e., $\gamma^{15} = 1$. Defining the sets

$$\begin{aligned} Zx_0 &= \left\{(\gamma^k, \gamma^l)|(k, l) \in \Omega, k = 0\right\}\\ Zy_0 &= \left\{(\gamma^k, \gamma^l)|(k, l) \in \Omega, l = 0\right\}\\ Zy_3 &= \left\{(\gamma^k, \gamma^l)|(k, l) \in \Omega, l \in (3, 6, 9, 12)\right\}\\ Zy_5 &= \left\{(\gamma^k, \gamma^l)|(k, l) \in \Omega, l \in (5, 10)\right\} \end{aligned} \tag{48}$$

where $\Omega = \{(k, l)|0 \le k \le 14, 0 \le l \le 14\}$, the zeros for the five sets E_i are expressed as

$$\begin{aligned} &E_0 = Zx_0 \cup Zy_5, E_1 = Zx_0 \cup Zy_0, E_2 = Zx_0 \cup Zy_3\\ &E_3 = Zx_0 \cup Zy_0 \cup Zy_5, \text{ and } E_4 = Zx_0. \end{aligned} \tag{49}$$

Common to all five sets E_i is the set Zx_0, which is identical to E_4. Therefore, the zeros in E_4 do nothing for distinguishing the five intersection sets, $C_i = E_i \cap V_c$, in view of Theorem 1. Thus, the zeros of E_4 should not be considered when searching the simplest set of zeros with minimum parity bits. Moreover, at least three zeros are needed for V_c, to separate all five intersection sets. This is because the number of subsets of V_c needs be more than 5. So, then what is the simplest set of zeros with minimum parity bits, also satisfying the condition of Theorem 1?

In searching for viable candidates of zeros, we notice that the eight zeros in four conjugate point sets, $\{(\gamma^0, \gamma^5), (\gamma^0, \gamma^{10})\}$, $\{(\gamma^5, \gamma^0), (\gamma^{10}, \gamma^0)\}$, $\{(\gamma^5, \gamma^5), (\gamma^{10}, \gamma^{10})\}$ and $\{(\gamma^5, \gamma^{10}), (\gamma^{10}, \gamma^5)\}$, all require 2 parity bits each, a minimum number. However, any combination of three of these zeros cannot satisfy the condition of Theorem 1.

The next simplest choice is then to select two zeros from the above list of eight candidates each costing two parity bits and one zero requiring four parity bits. Because the zeros on E_4 cannot be selected, knowing $E_0 \cup E_1 = E_3$, $E_3 \cap E_2 = E_4$ and $E_0 \cap E_1 = E_4$, we select a zero from E_2, E_0 and E_1 each. For E_2, all zeros equally cost four parity bits each, but the computation factor is minimized with (γ^1, γ^3). Moreover, for E_0 and E_1, the zeros costing two parity bits should be considered first. Among such zeros, $(\gamma^5, \gamma^5) \in E_0$ and $(\gamma^5, \gamma^0) \in E_1$ are shown to have the minimum computation factor. Overall, the following set of zeros gives the minimum number of parity bits and the smallest computation factor among all sets satisfying the condition of Theorem 1:

$$V_c = \left\{(\gamma^1, \gamma^3), (\gamma^5, \gamma^5), (\gamma^5, \gamma^0)\right\} \tag{50}$$

The resulting code detects any single occurrence of the predetermined five 2D error patterns at any position, and we now refine this particular V_c to meet the condition of Theorem 2 for achieving full periodicity of the syndrome sets.

First, it is easily seen that V_c must have at least four elements to meet the condition of Theorem 2, because the five sets $V_c \cap E_i^c$, which determine the periods of the syndrome sets S_i, should have at least two elements each to satisfy the condition of Theorem 2. Therefore, we first set out to add a zero.

For selecting a proper zero for this purpose, consider the sets $V_c \cap E_i^c$. Because only the set $V_c \cap E_3^c = \{(\gamma^1, \gamma^3)\}$ has one element so that unless a new zero comes from E_3^c, the corresponding error pattern $e_3(x, y)$ never shows full-periodicity. So select a zero with the minimum parity bits and the least computation factor in E_3^c. The zero (γ^1, γ^1) turns out to be such a zero; adding it, we now apply the general algorithm to the set:

$$V_c = \left\{(\gamma^1, \gamma^3), (\gamma^1, \gamma^1), (\gamma^5, \gamma^5), (\gamma^5, \gamma^0)\right\} \tag{51}$$

After just one iteration of our Algorithm 1, the zero (γ^5, γ^0) is deleted and an additional zero (γ^1, γ^0) is selected. The resulting set

$$V_c = \left\{(\gamma^1, \gamma^3), (\gamma^1, \gamma^1), (\gamma^5, \gamma^5), (\gamma^1, \gamma^0)\right\} \tag{52}$$

satisfies the condition of Theorem 2 for perfect correction of all single occurrences of the five predetermined error patterns anywhere on the (15 × 15) array.

The corresponding intersection sets are

$$\begin{aligned} C_0 &= \left\{(\gamma^5, \gamma^5)\right\}, C_1 = \left\{(\gamma^1, \gamma^0)\right\}, \\ C_2 &= \left\{(\gamma^1, \gamma^3)\right\}, C_3 = \left\{(\gamma^5, \gamma^5), (\gamma^1, \gamma^0)\right\}, C_4 = \emptyset. \end{aligned} \tag{53}$$

By inspecting the sets

$$\begin{aligned} V_c \cap E_0^c &= \left\{(\gamma^1, \gamma^3), (\gamma^1, \gamma^1), (\gamma^1, \gamma^0)\right\}, \\ V_c \cap E_1^c &= \left\{(\gamma^1, \gamma^3), (\gamma^1, \gamma^1), (\gamma^5, \gamma^5)\right\}, \\ V_c \cap E_2^c &= \left\{(\gamma^1, \gamma^1), (\gamma^5, \gamma^5), (\gamma^1, \gamma^0)\right\}, \\ V_c \cap E_3^c &= \left\{(\gamma^1, \gamma^3), (\gamma^1, \gamma^1)\right\}, \\ V_c \cap E_4^c &= \left\{(\gamma^1, \gamma^3), (\gamma^1, \gamma^1), (\gamma^5, \gamma^5), (\gamma^1, \gamma^0)\right\} \end{aligned} \tag{54}$$

we see that

$$\begin{aligned} &|PS_0| = 0, |PS_1| = 0, \\ &|PS_2| = 0, |PS_3| = 0, |PS_4| = 0, \end{aligned} \tag{55}$$

confirming full-periodicity. The resulting code requires 14 parity bits among 225 bits, giving a code rate of 0.938.

For comparing our 2D code to general random correcting codes like MDS codes and other 2D burst-correcting codes, a channel model for evaluating frame error rate (FER) performance is constructed. We take the 2D ISI channel giving rise to minimum-distance error patterns of Table II and, under the assumption of an ideal 2D maximum-likelihood equalizer, construct an equivalent channel that produces an error event with probability p. We assume that given that an error event has occurred, the probability of a particular event of Table II depends only on its *a priori* probability, a reasonable assumption since the error events in the table are all characterized by the same Euclidean distance at the 2D ISI channel output. This is to say that the conditional probability of an error event of horizontal length l in Table II is set proportional to 2^{-2l}. We assume the probability of any non-minimum-distance error events is negligible.

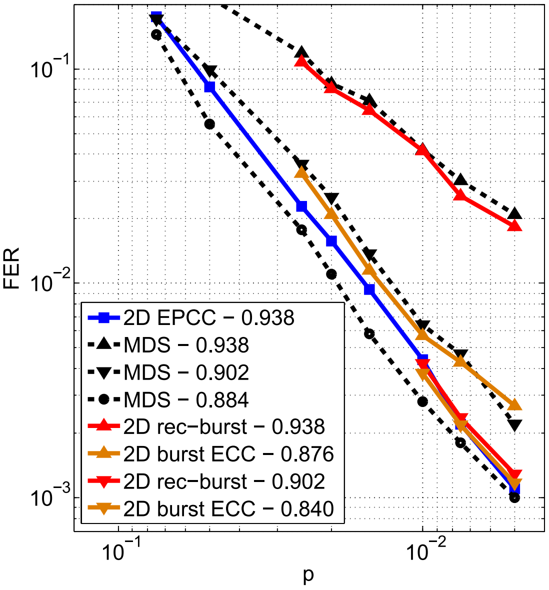

Fig. 5. FER of designed 2D pattern-correcting code, MDS codes and 2D burst (rectangular/arbitrary-shaped) codrrecting codes of various rates.

We divide a large 2D array or frame into many 2D subblocks in which only one error event may arise independently. The proposed code is assumed to have failed when there occur two or more error events in the whole frame, unless the error events occurring in two adjacent subblocks jointly form one of the targeted patterns. Specifically, a (15 × 15) bit array is broken into five (3 × 7)-bit blocks and five (3 × 8)-bit blocks. The designed code, as given in the example above, can correct 5 error patterns of horizontal dimensions 3 to 7 with 14 parity bits. The equivalent channel is such that once an error event occurs, it is always one of the minimum distance error events of length 3 to 7.

For comparison, we consider an MDS code that can correct any errors up to one half of the number of redundant bits used. We also consider two of the known 2D burst-correcting codes [12], [18]. The first code corrects rectangular-shaped 2D bursts [12] while the second code corrects arbitrary-shaped 2D bursts using 1D component codes. For the first code a known relationship exists between the number of redundant bits and the size of the correctable bursts [12], whereas in the second code the lower bound on the amount of redundancy presented in [18] is utilized for generating FER curves (which would represent an optimistic performance for the code).

Fig. 5 shows FER curves as function of p. For the competing codes, different code rates are considered as well. It can be shown that the proposed 2D EPCC code with rate 0.938 performs as well as the rate-0.902 2D rectangular-burst- correcting code of [12], and the rate-0.84 2D code of [18] and nearly as well as the rate-0.884 MDS code in the low FER region. It can be seen that the competing codes all degrade significantly when their rates match that of the proposed code. Overall, it is confirmed that designing codes specifically tailored to known error patterns can provide substantially stronger error correction capability for the same code rate.

VI. Conclusion

In this paper, 2D cyclic codes have been proposed and designed. The codes correct any single occurrence of a given list of known error patterns anywhere in a 2D array of bits. The 2D code is completely characterized by its set of zeros. Theorem 1 provides a condition for constructing the set of zeros such that all syndrome sets corresponding to the target error patterns are distinct with no shared elements. Theorem 2 specifies another condition based on which refinement to the set of zeros obtained from Theorem 1 can be made so that each syndrome set has distinct members, guaranteeing full correction capability. A general algorithm for the code construction is described. A number of example codes have been constructed. As a practical example, a 2D ISI channel that arises from magnetic recording has been considered; for this type of channel where a few dominant error patterns containing a relatively large number of bit errors dominate the error rate performance, the suggested 2D code performs error correction using a fewer number of parity bits than the MDS codes and known 2D burst-correcting codes tested.

Appendix

A. Proof for Property 1 (for Syndrome Polynomials)

Note that $e_{i,j}$, $(i,j) \in \Omega$, in (15) represent the content of the shift register when an N_x by N_y word $e(x,y)$ fills array with the feedback connection turned off (feedback disabled) whereas $\sigma_{i,j}$, $(i,j) \in \Pi$, corresponds to the content formed with the feedback connections on (feedback enabled) when the same word completely has entered the array.

To prove (15), we first assume that at some point in the middle of the process of filling the array with $e(x,y)$, the content with the feedback on is identical to that with the feedback off. We will then just have to show that at the next point (after an x-shift or y-shift) the contents for two cases are again identical to each other.

Let $I_1(x,y) = \sum_{(i,j)\in\Pi} I_{1,i,j} x^i y^j$ and $I_2(x,y) = \sum_{(i,j)\in\Omega} I_{2,i,j} x^i y^j$ correspond to the contents of the register at some point with and without the feedback on, respectively. So let us assume

$$\sum_{(i,j)\in\Pi} I_{1,i,j}\mathbf{h}_{i,j} = \sum_{(i,j)\in\Omega} I_{2,i,j}\mathbf{h}_{i,j}. \tag{56}$$

Now, imagine that a new input bit, denoted a, enters the upper-left corner of the shift register. The content with the feedback on would be

$$\sum_{(i,j)\in\Pi-\Pi_{\partial x}} I_{1,i,j} x^{[i+1]_x} y^j + \sum_{(i,j)\in\Pi_{\partial x}} I_{1,i,j}\left(\sum_{(k,l)\in\Pi} h_{k,l}^{(i+1,j)} x^k y^l\right) + a. \tag{57}$$

for which the row vector sum representation is

$$\sum_{(i,j)\in\Pi-\Pi_{\partial x}} I_{1,i,j}\mathbf{h}_{i+1,j} + \sum_{(i,j)\in\Pi_{\partial x}} I_{1,i,j}\left(\sum_{(k,l)\in\Pi} h_{k,l}^{(i+1,j)}\mathbf{h}_{k,l}\right) + a\mathbf{h}_{0,0}. \tag{58}$$

As for the shift-register content with feedback disconnected, we have

$$\sum_{(i,j)\in\Omega} I_{2,i,j} x^{[i+1]_x} y^j + a \tag{59}$$

or

$$\sum_{(i,j)\in\Omega} I_{2,i,j}\mathbf{h}_{i+1,j} + a\mathbf{h}_{0,0}. \tag{60}$$

We shall show that (58) and (60) are identical. Based on the definition (5), for all zeros $(\xi_p, \eta_{p,q}) \in V_c$, $1 \le p \le s$ and $1 \le q \le t_p$, the assumption (56) is equivalent to:

$$\sum_{(i,j)\in\Pi} I_{1,i,j}\left(\xi_p^i \eta_{p,q}^j\right) = \sum_{(i,j)\in\Omega} I_{2,i,j}\left(\xi_p^i \eta_{p,q}^j\right) \tag{61}$$

which, because $\xi_p \neq 0$, is also equivalent to:

$$\sum_{(i,j)\in\Pi} I_{1,i,j}\left(\xi_p^{i+1} \eta_{p,q}^j\right) = \sum_{(i,j)\in\Omega} I_{2,i,j}\left(\xi_p^{i+1} \eta_{p,q}^j\right) \tag{62}$$

or

$$\sum_{(i,j)\in\Pi} I_{1,i,j}\mathbf{h}_{i+1,j} = \sum_{(i,j)\in\Omega} I_{2,i,j}\mathbf{h}_{i+1,j}. \tag{63}$$

Adding $a\mathbf{h}_{0,0}$, it follows that

$$\begin{aligned}\sum_{(i,j)\in\Omega} I_{2,i,j}\mathbf{h}_{i+1,j} + a\mathbf{h}_{0,0} &= \sum_{(i,j)\in\Pi} I_{1,i,j}\mathbf{h}_{i+1,j} + a\mathbf{h}_{0,0}\\ &= \sum_{(i,j)\in\Pi-\Pi_{\partial x}} I_{1,i,j}\mathbf{h}_{i+1,j}\\ &\quad + \sum_{(i,j)\in\Pi_{\partial x}} I_{1,i,j}\mathbf{h}_{i+1,j} + a\mathbf{h}_{0,0}\\ &= \sum_{(i,j)\in\Pi-\Pi_{\partial x}} I_{1,i,j}\mathbf{h}_{i+1,j}\\ &\quad + \sum_{(i,j)\in\Pi_{\partial x}} I_{1,i,j}\left(\sum_{(k,l)\in\Pi} h_{k,l}^{(i+1,j)}\mathbf{h}_{k,l}\right)\\ &\quad + a\mathbf{h}_{0,0}\end{aligned} \tag{64}$$

which shows (58) equals (60).

For shifting along the y-direction, we also see that the contents with and without feedback connections continue to be identical. In particular, we have

$$\begin{aligned}\sum_{(i,j)\in\Omega} I_{2,i,j}\mathbf{h}_{i,j+1} &= \sum_{(i,j)\in\Pi} I_{1,i,j}\mathbf{h}_{i,j+1}\\ &= \sum_{(i,j)\in\Pi-\Pi_{\partial_y}} I_{1,i,j}\mathbf{h}_{i,j+1} + \sum_{(i,j)\in\Pi_{\partial_y}} I_{1,i,j}\mathbf{h}_{i,j+1}\\ &= \sum_{(i,j)\in\Pi-\Pi_{\partial_y}} I_{1,i,j}\mathbf{h}_{i,j+1}\\ &\quad + \sum_{(i,j)\in\Pi_{\partial_y}} I_{1,i,j}\left(\sum_{(k,l)\in\Pi} h_{k,l}^{(i,j+1)}\mathbf{h}_{k,l}\right) \quad (65)\end{aligned}$$

Notice that there is no input entering the array during the y-direction shift.

B. Proof for Property 2 (for Syndrome Polynomials)

The left side of (15) can be written as

$$\begin{aligned}\sum_{(i,j)\in\Omega} e_{i,j}\mathbf{h}_{i,j} &= \sum_{(i,j)\in\Pi} e_{i,j}\mathbf{h}_{i,j} + \sum_{(i,j)\in\Omega-\Pi} e_{i,j}\mathbf{h}_{i,j}\\ &= \sum_{(i,j)\in\Pi} e_{i,j}\mathbf{h}_{i,j} + \sum_{(k,l)\in\Pi}\left(\sum_{(i,j)\in\Omega-\Pi} h_{k,l}^{(i,j)} e_{i,j}\right)\mathbf{h}_{k,l}\\ &= \sum_{(k,l)\in\Pi}\left(e_{k,l} + \sum_{(i,j)\in\Omega-\Pi} h_{k,l}^{(i,j)} e_{i,j}\right)\mathbf{h}_{k,l}. \quad (66)\end{aligned}$$

Using (15), we can write

$$\sum_{(k,l)\in\Pi}\sigma_{k,l}\mathbf{h}_{k,l} = \sum_{(k,l)\in\Pi}\left(e_{k,l} + \sum_{(i,j)\in\Omega-\Pi} h_{k,l}^{(i,j)} e_{i,j}\right)\mathbf{h}_{k,l}. \quad (67)$$

Because the basis vectors $\mathbf{h}_{k,l}$ are linearly independent, the equality (67) holds if and only if for all positions $(k, l) \in \Pi$,

$$\sigma_{k,l} = e_{k,l} + \sum_{(i,j)\in\Omega-\Pi} h_{k,l}^{(i,j)} e_{i,j}. \quad (68)$$

Thus, we finally have

$$\begin{aligned}\sigma(x, y) &= \sum_{(k,l)\in\Pi}\sigma_{k,l}x^k y^l\\ &= \sum_{(k,l)\in\Pi}\left(e_{k,l} + \sum_{(i,j)\in\Omega-\Pi} h_{k,l}^{(i,j)} e_{i,j}\right)x^k y^l\\ &= \sum_{(i,j)\in\Pi} e_{i,j}x^i y^j + \sum_{(k,l)\in\Pi}\left(\sum_{(i,j)\in\Omega-\Pi} h_{k,l}^{(i,j)} e_{i,j}\right)x^k y^l\end{aligned} \quad (69)$$

C. Proof for Property 3

We reproduce (7) here for convenience:

$$\mathbf{h}_{i,j} = \sum_{(k,l)\in\Pi} h_{k,l}^{(i,j)}\mathbf{h}_{k,l}$$

for any $(i, j) \in \Omega - \Pi$, which states that any row vector in $\Omega - \Pi$ can be uniquely represented by a sum of the basis row vectors in Π. From the definition of a row vector in (5), it is clear that (7) holds if and only if

$$\xi_p^i \eta_{p,q}^j = \sum_{(k,l)\in\Pi} h_{k,l}^{(i,j)}\left(\xi_p^k \eta_{p,q}^l\right) \quad (70)$$

for all zeros $(\xi_p, \eta_{p,q}) \in V_c$, $1 \le p \le s$ and $1 \le q \le t_p$. This statement is identical to equality (17).

D. Proof for Lemma 1.1

For a zero $(\xi, \gamma) \in C(= E \cap V_c)$, we know that

$$\begin{aligned}e(\xi, \gamma) &= \sum_{(i,j)\in\Omega} e_{i,j}\xi^i\gamma^j\\ &= \sum_{(i,j)\in\Pi} e_{i,j}\xi^i\gamma^j + \sum_{(i,j)\in\Omega-\Pi} e_{i,j}\xi^i\gamma^j = 0. \quad (71)\end{aligned}$$

Using *Property 3*, we further write

$$\begin{aligned}e(\xi, \gamma) &= \sum_{(i,j)\in\Pi} e_{i,j}\xi^i\gamma^j + \sum_{(i,j)\in\Omega-\Pi} e_{i,j}\left(\sum_{(k,l)\in\Pi} h_{k,l}^{(i,j)}\xi^k\gamma^l\right)\\ &= \sum_{(i,j)\in\Pi} e_{i,j}\xi^i\gamma^j + \sum_{(k,l)\in\Pi}\left(\sum_{(i,j)\in\Omega-\Pi} h_{k,l}^{(i,j)} e_{i,j}\right)\xi^k\gamma^l = 0.\end{aligned} \quad (72)$$

Now from *Property 2* the syndrome polynomial for $e(x, y)$ is given by

$$\begin{aligned}e(T_x, T_y) &= \sigma^{(0,0)}(x, y)\\ &= \sum_{(i,j)\in\Pi} e_{i,j}x^i y^j + \sum_{(k,l)\in\Pi}\left(\sum_{(i,j)\in\Omega-\Pi} h_{k,l}^{(i,j)} e_{i,j}\right)x^k y^l,\end{aligned} \quad (73)$$

which leads to

$$\begin{aligned}\left\{e(T_x, T_y)\right\}_{(x,y)=(\xi,\gamma)} &= \sigma^{(0,0)}(\xi, \gamma)\\ &= \sum_{(i,j)\in\Pi} e_{i,j}\xi^i\gamma^j\\ &\quad + \sum_{(k,l)\in\Pi}\left(\sum_{(i,j)\in\Omega-\Pi} h_{k,l}^{(i,j)} e_{i,j}\right)\xi^k\gamma^l\end{aligned} \quad (74)$$

which, from (72), is equal to zero. This completes the proof of the first part of Lemma 1.1.

For proving the second part, take a point (ξ_a, γ_a) that is in V_c but not in C. Since $(\xi_a, \gamma_a) \in V_c$, $\xi_a^i\gamma_a^j = \sum_{(k,l)\in\Pi} h_{k,l}^{(i,j)}\xi_a^k\gamma_a^l$ for

$(i,j) \in \Omega - \Pi$. Also, since $(\xi_a, \gamma_a) \notin E$, we know $e(\xi_a, \gamma_a) \neq 0$. But, following (72), we have

$$e(\xi_a, \gamma_a) = \sum_{(i,j)\in\Pi} e_{i,j}\xi_a^i\gamma_a^j + \sum_{(k,l)\in\Pi}\left(\sum_{(i,j)\in\Omega-\Pi} h_{k,l}^{(i,j)} e_{i,j}\right)\xi_a^k\gamma_a^l, \quad (75)$$

which is seen to be the same as $\{e(T_x, T_y)\}_{(x,y)=(\xi_a,\gamma_a)}$. Thus, we have established that

$$\left\{e(T_x, T_y)\right\}_{(x,y)=(\xi_a,\gamma_a)} \neq 0$$

or that the point (ξ_a, γ_a) cannot be a zero of $e(T_x, T_y) = \sigma^{(0,0)}(x, y)$. This completes the proof.

E. Proof for Lemma 1.2

When we shift the given polynomial $\sigma(x, y) = \sum_{(i,j)\in\Pi} \sigma_{i,j}x^iy^j$ along the x direction, the resulting content of each 2D shift register at position $(k, l) \in \Pi$ is

$$\sigma_{k-1,l} + \sum_{(i,j)\in\Pi_{\partial x}} h_{k,l}^{i+1,j}\sigma_{i,j} \quad (76)$$

(the first term exists only when the shift register at position (k, l) is connected to the adjacent register at position $(k-1, l)$). Let us consider an arbitrary zero $(\xi, \gamma) \in A^{(0,0)}$. Then

$$\sigma(\xi, \gamma) = \sum_{(i,j)\in\Pi} \sigma_{i,j}\xi^i\gamma^j = 0 \quad (77)$$

Therefore,

$$\xi \sum_{(i,j)\in\Pi} \sigma_{i,j}\xi^i\gamma^j = \sum_{(i,j)\in\Pi} \sigma_{i,j}\xi^{i+1}\gamma^j = 0. \quad (78)$$

Partitioning Π into $\Pi_{\partial x}$ and $\Pi - \Pi_{\partial x}$, we can write

$$\sum_{(i,j)\in\Pi-\Pi_{\partial x}} \sigma_{i,j}\xi^{i+1}\gamma^j + \sum_{(i,j)\in\Pi_{\partial x}} \sigma_{i,j}\xi^{i+1}\gamma^j = 0. \quad (79)$$

The second term on the left side can be written as

$$\begin{aligned}\sum_{(i,j)\in\Pi_{\partial x}} \sigma_{i,j}\xi^{i+1}\gamma^j &= \sum_{(i,j)\in\Pi_{\partial x}} \sigma_{i,j}\left(\sum_{(k,l)\in\Pi} h_{k,l}^{(i+1,j)}\xi^k\gamma^l\right)\\ &= \sum_{(k,l)\in\Pi}\left(\sum_{(i,j)\in\Pi_{\partial x}} h_{k,l}^{(i+1,j)}\sigma_{i,j}\right)\xi^k\gamma^l. \quad (80)\end{aligned}$$

Now consider the positions $(k, l) \in \Pi$ such that $k = i + 1$ and $l = j$ with $(i, j) \in \Pi - \Pi_{\partial x}$. Let B be the collection of all such positions (k, l). When we shift a syndrome polynomial in the x direction, only the shift registers at positions $(k, l) \in B$ receive input values from adjacent registers at positions $(k-1, l)$. Therefore, the first term on the left side of (79) can be written as

$$\sum_{(i,j)\in\Pi-\Pi_{\partial x}} \sigma_{i,j}\xi^{i+1}\gamma^j = \sum_{(k,l)\in B} \sigma_{k-1,l}\xi^k\gamma^l. \quad (81)$$

From (79)–(81), we have

$$\sum_{(k,l)\in B} \sigma_{k-1,l}\xi^k\gamma^l + \sum_{(k,l)\in\Pi}\left(\sum_{(i,j)\in\Pi_{\partial x}} h_{k,l}^{(i+1,j)}\sigma_{i,j}\right)\xi^k\gamma^l = 0. \quad (82)$$

Note that (76) defines the coefficients of a syndrome polynomial shifted in the x direction. Comparing (76) and (82), we see that substituting $(\xi, \eta) \in A^{(0,0)}$ into the syndrome polynomial shifted along the x direction yields a zero. This proves that a zero of $\sigma(x, y)$ is also a zero of $T_x\sigma(x, y)$. The proof for $T_y\sigma(x, y)$ follows in the same manner. Consequently, for any shift $(k, l) \in \Omega'$, $A^{(0,0)} \subseteq A^{(k,l)}$.

When we shift the syndrome polynomial by N_x and N_y steps along x and y directions corresponding to the size of the code, the resulting polynomial turns to the original syndrome polynomial, meaning $A^{(0,0)} = A^{(N_x,N_y)}$. Therefore,

$$A \subseteq \cdots \subseteq A^{(k,l)} \subseteq \cdots \subseteq A^{(N_x,N_y)} = A^{(0,0)}. \quad (83)$$

Consequently, for all $(k, l) \in \Omega'$,

$$A^{(0,0)} = A^{(k,l)}. \quad (84)$$

F. Proof for Lemma 2.1

Equation (23) follows directly from Property 1 by utilizing the fact that each term $\xi_p^k\eta_{p,q}^l$ in the vector vector representation of $\mathbf{h}_{k,l}$ (see (5)) can be written as $\xi_p^{[k]_x}\eta_{p,q}^{[l]_y}$.

G. Proof for Lemma 2.2

From the definition of period set PS_i, a certain shift (P_x, P_y) is an element of PS_i if and only if

$$\sigma_i^{(0,0)}(x, y) = \sigma_i^{(P_x,P_y)}(x, y). \quad (85)$$

The equivalent expression using the basis vectors in Π is:

$$\sum_{(k,l)\in\Pi} \sigma_{i,k,l}\mathbf{h}_{k,l} = \sum_{(k,l)\in\Pi} \sigma_{i,k,l}^{(P_x,P_y)}\mathbf{h}_{k,l}. \quad (86)$$

Now from Property 1 and the fact that $\xi_p^k\eta_{p,q}^l = \xi_p^{[k]_x}\eta_{p,q}^{[l]_y}$, (86) becomes

$$\sum_{(k,l)\in\Omega} e_{i,k,l}\mathbf{h}_{k,l} = \sum_{(k,l)\in\Omega} e_{i,k,l}\mathbf{h}_{[k+P_x]_x,[y+P_y]_y}. \quad (87)$$

This completes the proof. Note that (87) is equivalent to

$$\sum_{(k,l)\in\Pi} \sigma_{i,k,l}\mathbf{h}_{k,l} = \sum_{(k,l)\in\Pi} \sigma_{i,k,l}\mathbf{h}_{[k+P_x]_x,[l+P_y]_y} \quad (88)$$

due to Property 1.

REFERENCES

[1] M. Marrow, "Detection and modelling of 2-dimensional signals," Ph.D. dissertation, Elect. Eng., Univ. California, San Diego, CA, USA, 2004.

[2] I. Demirkan, P. H. Siegel and J. K. Wolf, "Error event characterization on 2-D ISI channels," *IEEE Trans. Inf. Theory*, vol. 55, no. 3, pp. 1146–1152, Mar. 2009.

[3] J. Park and J. Moon, "Error-pattern-correcting cyclic codes tailored to a prescribed set of error cluster patterns," *IEEE Trans. Inf. Theory*, vol. 55, no. 4, pp. 1747–1765, Apr. 2009.

[4] H. Alhussien, J. Park, and J. Moon, "Iterative decoding based on error pattern correction," *IEEE Trans. Magn.*, vol. 44, no. 1, pp. 181–186, Jan. 2008.

[5] F. Rusek, E. K. S. Au, J. B. Anderson, and W. H. Mow, "Minimum distance analysis of a certain class of 2-D ISI channels," *IEEE Trans. Inf. Theory*, vol. 58, no. 2, Feb. 2012.

[6] S. W. Yoon and J. Moon, "Two-dimensional cyclic codes correcting known error patterns," in *Proc. IEEE GLOBECOM*, Dec. 2012, pp. 3231–3236.

[7] J. Park and J. Moon, "Cyclic codes tailored to a known set of error patterns," in *Proc. IEEE GLOBECOM*, Nov. 2006, pp. 1–5.

[8] J. Park and J. Moon, "High-rate error-correction codes targeting dominant error patterns," *IEEE Trans. Magn.*, vol. 42, no. 10, pp. 2573–2575, Oct. 2006.

[9] J. Park and J. Moon, "A new class of error-pattern-correcting codes capable of handling multiple error occurrences," *IEEE Trans. Magn.*, vol. 43, no. 6, pp. 2268–2270, Jun. 2007.

[10] H. Imai, "Two-dimensional fire codes," *IEEE Trans. Inf. Theory*, vol. 19, no. 6, pp. 796–806, Nov. 1973.

[11] H. Imai, "A theory of two-dimensional cyclic codes," *Inf. control*, vol. 34, no. 1, pp. 1–21, May 1977.

[12] K. A. S. Abdel-Ghaffar, "An information-and coding-theoretic study of bursty channels with applications to computer memories," Ph.D. dissertation, Division Eng. Appl. Sci., Calif. Inst. Technol., Pasadena, CA, USA, 1986.

[13] M. Breitbach, M. Bossert, V. Zyablov, and V. Sidorenko, "Array codes correcting a two-dimensional cluster of errors," *IEEE Trans. Inf. Theory*, vol. 44, no. 5, pp. 2025–2031, Sep. 1998.

[14] M. Bossert and V. Sidorenko, "Singleton-type bounds for blot-correcting codes," *IEEE Trans. Inf. Theory*, vol. 42, no. 3, pp. 1021–1023, May 1996.

[15] M. Blaum, J. Bruck, and A. Vardy, "Interleaving schemes for multidimensional cluster errors," *IEEE Trans. Inf. Theory*, vol. 44, no. 2, pp. 730–743, Mar. 1998.

[16] T. Etzion and A. Vardy, "Two-dimensional interleaving schemes with repetitions: constructions and bounds," *IEEE Trans. Inf. Theory*, vol. 48, no. 2, pp. 428–457, Feb. 2002.

[17] M. Schwartz and T. Etzion, "Two-dimensional cluster-correcting codes," *IEEE Trans. Inf. Theory*, vol. 51, no. 6, pp. 2121–2132, Jun. 2005.

[18] T. Etzion and E. Yaakobi, "Error-correction of multidimensional bursts," *IEEE Trans. Inf. Theory*, vol. 55, no. 3, pp. 961–976, Mar. 2009.

[19] J. Moon and B. Brickner, "Maximum transition run codes for data storage systems," *IEEE Trans. Magn.*, vol. 32, no. 5, pp. 3992–3994, Sep. 1996.

[20] R. D. Cideciyan, E. Eleftheriou, B. H. Marcus, and D. S. Modha, "Maximum transition run codes for generalized partial response channels," *IEEE J. Sel. Areas Commun.*, vol. 19, no. 4, pp. 619–634, Apr. 2001.

Sung Whan Yoon received the B.S. and M.S. degrees in electrical engineering from the Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea, in 2011 and 2013, respectively. He is currently pursuing the Ph.D. degree in KAIST. His research interests include error-pattern-correcting codes and massive MIMO effects on wireless multi cellular system.

Jaekyun Moon (M'90–SM'97–F'05) received the Ph.D. degree in electrical and computer engineering at Carnegie Mellon University, Pittsburgh, PA, USA. He is currently a Professor of electrical engineering at KAIST. From 1990 through early 2009, he was with the faculty of the Department of Electrical and Computer Engineering at the University of Minnesota, Twin Cities. He consulted as Chief Scientist for DSPG, Inc. from 2004 to 2007. He also worked as Chief Technology Officier at Link-A-Media Devices Corporation. His research interests are in the area of channel characterization, signal processing and coding for data storage and digital communication. Prof. Moon received the McKnight Land-Grant Professorship from the University of Minnesota. He received the IBM Faculty Development Awards as well as the IBM Partnership Awards. He was awarded the National Storage Industry Consortium (NSIC) Technical Achievement Award for the invention of the maximum transition run (MTR) code, a widely used error-control/modulation code in commercial storage systems. He served as Program Chair for the 1997 IEEE Magnetic Recording Conference. He is also Past Chair of the Signal Processing for Storage Technical Committee of the IEEE Communications Society, In 2001, he cofounded Bermai, Inc., a fabless semiconductor start-up, and served as founding President and CTO. He served as a guest editor for the 2001 IEEE JSAC issue on Signal Processing for High Density Recording. He also served as an Editor for IEEE TRANSACTIONS ON MAGNETICS in the area of signal processing and coding for 2001–2006.