Product Perfect $\mathbb{Z}_2\mathbb{Z}_4$-linear codes in Steganography

Josep Rifà and Lorena Ronquillo
Department of Information and Communications Engineering,
Universitat Autònoma de Barcelona,
08193-Cerdanyola del Vallès, Spain.
Email: Josep.Rifa@autonoma.edu, Lorena.Ronquillo@autonoma.edu

***Abstract*—Product perfect codes have been proven to enhance the performance of the $F5$ steganographic method, whereas perfect $\mathbb{Z}_2\mathbb{Z}_4$-linear codes have been recently introduced as an efficient way to embed data, conforming to the ± 1-steganography. In this paper, we present two steganographic methods. On the one hand, a generalization of product perfect codes is made. On the other hand, this generalization is applied to perfect $\mathbb{Z}_2\mathbb{Z}_4$-linear codes. Finally, the performance of the proposed methods is evaluated and compared with those of the aforementioned schemes.**

I. Introduction

Digital steganography is an information hiding application which consists of hiding data within seemingly innocuous host media, by distorting it in such a way that unintended recipients are not only unable to detect the presence of embedded data, but also given no reason for suspecting that anything is unusual. This is the main difference from encryption, which only prevents the adversary from decoding the message and not from suspecting that a secret message is being sent.

Many steganographic schemes have been developed and proposed. One well known scheme is *matrix encoding*, first introduced by Crandall [5] and later analyzed by Bierbrauer et al. [2], which requires the sender and the recipient to agree in advance on a parity check matrix H, and the secret message is then extracted by the recipient as the syndrome (with respect to H) of the received cover object. This method was made popular by Westfeld [10], who incorporated a specific implementation using Hamming codes. The resulting method is known as the $F5$ algorithm, and it can embed t bits of message in $2^t - 1$ cover symbols by changing, at most, one of them.

As Willems et al. in [11], we will also assume that a discrete source produces a sequence $\mathbf{x} = (x_1, \ldots, x_N)$, where N is the block length, $x_i \in \aleph = \{0, 1, \ldots, 2^B - 1\}$, and $B \in \{8, 12, 16\}$ depends on the kind of source (digital image, CD audio, etc). Let $\mathbf{s} \in \{1, \ldots, M\}$ be the message we want to hide into a host sequence $\mathbf{x}$, which produces a composite sequence $\mathbf{y} = f(\mathbf{x}, \mathbf{s})$, for $\mathbf{y} = (y_1, \ldots y_N)$ and $y_i \in \aleph$. The sequence $\mathbf{y}$ is obtained from distorting $\mathbf{x}$, and that distortion will be assumed to be of squared-error type (see [11]), that is $|\mathbf{y} - \mathbf{x}|^2 = \sum_{i=1}^{N} |y_i - x_i|^2$. In these conditions, information can be carried by the least significant bit (LSB) or by the two least significant bits of each x_i. An appropriate solution for the first case comes from applying the $F5$ algorithm [10], which has been improved in [7] by using the Kronecker product of the corresponding generator matrices of two binary perfect codes. The latter case is known as "± 1-steganography" and the magnitude of changes is limited to 1, that is, $y_i = x_i + c$, where $c \in \{0, +1, -1\}$. This case has usually involved the use of ternary codes [6], [11] until the results from [8], which introduces a method based on perfect $\mathbb{Z}_2\mathbb{Z}_4$-linear codes. This kind of codes are not linear but have a representation using a parity check matrix that makes them as efficient as the Hamming codes. The steganographic method therein presented not only outperforms the one obtained by direct sum of ternary Hamming codes, but it also deals better with the boundary grayscale values problem. That is, the problem we may have when the steganographic method requires adding one unit to a grayscale value which already has the maximum allowed value $2^B - 1$, or substracting one unit from a grayscale of value 0. Schemes from [6], [11] suggest, in these cases, performing changes of magnitude greater than one, while the magnitude of changes performed by the method in [8] is never greater than one.

The following two parameters are often used to evaluate the performance of a steganographic method over a cover message of N symbols: the *average distortion* $D = \frac{R_a}{N}$, where R_a is the expected number of changes over uniformly distributed messages; and the *embedding rate* $E = \frac{t}{N}$, which is the amount of bits that can be hidden in a cover message. Given two methods with the same embedding rate, the one with smaller average distortion is better. Following the terminology used by Fridrich et al. [6], the tuple (D, E) will be called *CI-rate*. However, when plotting the performance results of a steganographic scheme we will use what we have called the *normalized embedding rate* e, instead of the embedding rate E.

Let $H_q(x) = \frac{1}{\log_2(q)}(H(x) + x\log_2(q-1))$ be the q-ary entropy function [1] on the interval $[0, (q-1)/q]$, where $H(x) = -x\log_2(x) - (1-x)\log_2(1-x)$ is the usual binary entropy function on the interval $[0, 1/2]$. The normalized embedding rate is the ratio $e = \frac{{H_q}^{-1}(E)}{D}$, where ${H_q}^{-1}(\cdot)$ is the inverse of the q-ary entropy function $H_q(x)$. In the binary case, e will be computed by considering the binary entropy function $H_2(x)$, whereas in the ± 1-steganography the

This work was partially supported by the Spanish MICINN Grants MTM2009-08435, PCI2006-A7-0616, and also by the *Comissionat per a Universitats i Recerca del DIUE de la Generalitat de Catalunya* and the *European Social Fund* with Grants 2009SGR1224 and FI-DGR.

ternary entropy function $H_3(x)$ is used. One of the purposes of steganographic methods is to approach the upper bound on the normalized embedding rate e subject to the constraint of an average distortion D. This upper bound on e for a fixed D is $e \leq 1$, and it has the advantage of being the same for any kind of steganography, either binary or ± 1-steganography.

In this paper we propose a technique based on products of perfect $\mathbb{Z}_2\mathbb{Z}_4$-linear codes and compare its performance with that of product binary perfect codes [7] and perfect $\mathbb{Z}_2\mathbb{Z}_4$-linear codes [8].

The current paper has been organized as follows. Some basic concepts on perfect $\mathbb{Z}_2\mathbb{Z}_4$-linear codes, as well as the steganographic method based on these codes [8], are reviewed in Section II. Then, Section III reviews the product perfect codes method [7] and presents a generalization that enhances its performance. In Section IV, this generalization is used for perfect $\mathbb{Z}_2\mathbb{Z}_4$-linear codes in ± 1-steganography. Finally, the paper is concluded in Section V.

II. Perfect $\mathbb{Z}_2\mathbb{Z}_4$-linear codes and steganography

Any non-empty subgroup $\mathcal{C}$ of $\mathbb{Z}_2^\alpha \times \mathbb{Z}_4^\beta$ is a $\mathbb{Z}_2\mathbb{Z}_4$-*additive code*, where $\mathbb{Z}_2^\alpha$ denotes the set of all binary vectors of length α and $\mathbb{Z}_4^\beta$ is the set of all quaternary vectors of length β. Let ϕ be the usual *Gray map* from $\mathbb{Z}_4$ onto $\mathbb{Z}_2^2$, where $\phi(0) = (0,0)$, $\phi(1) = (0,1)$, $\phi(2) = (1,1)$, and $\phi(3) = (1,0)$; and let $\Phi : \mathbb{Z}_2^\alpha \times \mathbb{Z}_4^\beta \longrightarrow \mathbb{Z}_2^n$ be the *extended Gray map* given by

$$\Phi(u_1, \ldots, u_\alpha | v_1, \ldots, v_\beta) = (u_1, \ldots, u_\alpha | \phi(v_1), \ldots, \phi(v_\beta)).$$

A $\mathbb{Z}_2\mathbb{Z}_4$-additive code $\mathcal{C}$ is isomorphic to an abelian structure like $\mathbb{Z}_2^\gamma \times \mathbb{Z}_4^\delta$. Therefore, $\mathcal{C}$ has $|\mathcal{C}| = 2^\gamma 4^\delta$ codewords, where $2^{\gamma+\delta}$ of them are of order two. We call such code $\mathcal{C}$ a $\mathbb{Z}_2\mathbb{Z}_4$-*additive code of type* $(\alpha, \beta; \gamma, \delta)$ and its binary image $C = \Phi(\mathcal{C})$ is a $\mathbb{Z}_2\mathbb{Z}_4$-*linear code of type* $(\alpha, \beta; \gamma, \delta)$. Note that the Lee distance of a $\mathbb{Z}_2\mathbb{Z}_4$-additive code $\mathcal{C}$ coincides with the Hamming distance of the $\mathbb{Z}_2\mathbb{Z}_4$-linear code C, and that the binary code C may not be linear.

The $\mathbb{Z}_2\mathbb{Z}_4$-*additive dual code* of $\mathcal{C}$, denoted by $\mathcal{C}^\perp$, is defined as the set of vectors in $\mathbb{Z}_2^\alpha \times \mathbb{Z}_4^\beta$ that are orthogonal to every codeword in $\mathcal{C}$, being the definition of inner product in $\mathbb{Z}_2^\alpha \times \mathbb{Z}_4^\beta$ the following (see [3]):

$$\langle u, v \rangle = 2(\sum_{i=1}^{\alpha} u_i v_i) + \sum_{j=\alpha+1}^{\alpha+\beta} u_j v_j \in \mathbb{Z}_4, \qquad (1)$$

where $u, v \in \mathbb{Z}_2^\alpha \times \mathbb{Z}_4^\beta$ and computations are made considering the zeros and ones in the α binary coordinates as quaternary zeros and ones, respectively.

The binary code $C_\perp = \Phi(\mathcal{C}^\perp)$, of length $n = \alpha + 2\beta$, is called the $\mathbb{Z}_2\mathbb{Z}_4$-*dual code* of C.

A $\mathbb{Z}_2\mathbb{Z}_4$-additive code $\mathcal{C}$ is said to be *perfect* if code $C = \Phi(\mathcal{C})$ is a perfect $\mathbb{Z}_2\mathbb{Z}_4$-linear code, that is all vectors in $\mathbb{Z}_2^n$ are within distance one from a codeword and the distance between two codewords is, at least, 3.

It is well known [4] that for any $m \geq 2$ and each $\delta \in \{0, \ldots, \lfloor \frac{m}{2} \rfloor\}$ there exists a perfect $\mathbb{Z}_2\mathbb{Z}_4$-linear code C of binary length $n = 2^m - 1$, such that its $\mathbb{Z}_2\mathbb{Z}_4$-dual code is of type $(\alpha, \beta; \gamma, \delta)$, where $\alpha = 2^{m-\delta} - 1$, $\beta = 2^{m-1} - 2^{m-\delta-1}$ and $\gamma = m - 2\delta$ (note that the binary length can be computed as $n = \alpha + 2\beta$). This allows us to write the parity check matrix $\mathcal{H}_\mathcal{C}$ of any $\mathbb{Z}_2\mathbb{Z}_4$-additive perfect code $\mathcal{C}$ for a given value of δ. Matrix $\mathcal{H}_\mathcal{C}$ can be represented by taking as columns all possible vectors in $\mathbb{Z}_2^\gamma \times \mathbb{Z}_4^\delta$, up to sign changes. In this representation, there are α columns which correspond to the binary part of codewords in $\mathcal{C}$, and β columns of order four which correspond to the quaternary part. We agree on a representation of the α binary coordinates as coordinates in $\{0, 2\} \in \mathbb{Z}_4$. Let $\mathbf{h}_i$, for $i \in \{1, \ldots, \alpha + \beta\}$, denote the i-th column vector of $\mathcal{H}_\mathcal{C}$.

Now we proceed to review how a perfect $\mathbb{Z}_2\mathbb{Z}_4$-linear code $C = \Phi(\mathcal{C})$ can be used in steganography. Considering its $\mathbb{Z}_2\mathbb{Z}_4$-dual code, of type $(\alpha, \beta; \gamma, \delta)$, we have a parity check matrix $\mathcal{H}_\mathcal{C}$ with γ rows of order two and δ rows of order four.

Take $N = 2^{m-1}$ and let $\mathbf{x} = (x_1, \ldots, x_N)$ be a source of grayscale symbols such that $x_i \in \aleph = \{0, 1, \ldots, 2^B - 1\}$, where, for instance, $B = 8$ for grayscale images.

We assume each grayscale symbol x_i is represented as a binary vector $(v_{(B-1)i}, \ldots, v_{1i}, v_{0i})$, obtained by first representing x_i in base 4 and then applying the Gray map ϕ to every quaternary symbol in the base 4 representation. For example, the grayscale value 239 is represented as the quaternary vector (3233), which then gives rise to the binary vector (10111010) after applying the Gray map ϕ.

The N-length packet $\mathbf{x}$ of symbols is translated into a vector $\mathbf{w}$ of α binary and β quaternary coordinates. The binary coordinates come from taking the least significant bit of the binary representation of x_1, that is v_{01}, along with the two least significant bits v_{1i}, v_{0i} of the following $(\alpha + 1)/2$ grayscale symbols x_i. The quaternary coordinates of $\mathbf{w}$ come from taking the two least significant bits of the last β symbols x_i and interpreting them as integer numbers $\phi^{-1}(v_{1i}, v_{0i})$ in $\mathbb{Z}_4$.

The obtained vector $\mathbf{w} \in \mathbb{Z}_2^\alpha \times \mathbb{Z}_4^\beta$ is then distorted according to the matrix encoding method [5], [10] in such a way that $\mathcal{H}_\mathcal{C}\mathbf{w}^T + \epsilon\mathbf{h}_i = \mathbf{s}$ holds, where $\mathbf{s} \in \mathbb{Z}_2^\gamma \times \mathbb{Z}_4^\delta$ is the secret message we want to embed in the source $\mathbf{x}$, the value of ϵ can be $\{0, 1, 3\}$, the syndrome vector of $\mathbf{w}$ is $\mathcal{H}_\mathcal{C}\mathbf{w}^T$, and $\mathbf{h}_i$ is a column vector in $\mathcal{H}_\mathcal{C}$. This method also deals with boundary grayscale values in a rather efficient way: instead of distorting a symbol x_i having a boundary value (0 or $2^B - 1$) in a way that would lead its value out of the range defined by $\aleph$, two other symbols are changed one magnitude. One of these symbols is always x_1. Since we are using a perfect $\mathbb{Z}_2\mathbb{Z}_4$-linear code, with this scheme we might need to modify one coordinate or two coordinates (to deal with boundary values problems) from $\mathbf{w}$, and this distortion will involve having to add or substract one unit to/from the corresponding grayscale value. Note that the magnitude of change is never greater than 1. Thus considering the squared-error distortion, the distortion caused by this scheme is lower than that of the schemes in [6], [11]. It is easy to see that this method has

CI-rate $(D_m, E_m) = \left(\frac{2N-1+\frac{N-1}{2^{B-2}}}{2N^2}, \frac{1+\log(N)}{N}\right)$. We refer the reader to [8] for further details and examples on the steganographic scheme based on perfect $\mathbb{Z}_2\mathbb{Z}_4$-linear codes.

III. Product of perfect codes and steganography

Let $\mathbb{F}_q$ be a finite field of q elements, where q is a prime power. Let C be a Hamming code over $\mathbb{F}_q$ of length $n = \frac{q^m-1}{q-1}$ and dimension $n-m$. Let G_C, H_C be, respectively, a generator matrix and a parity check matrix for C.

Definition 1: The Kronecker product of two matrices $A = [a_{r,t}]$ and $K = [k_{i,j}]$ over $\mathbb{F}_q$ is a new matrix $A \otimes K$ obtained by changing every element $a_{r,t}$ in A by the matrix $a_{r,t}K$.

The q-ary code $C^{\perp}_{H_C \otimes H_C}$, that is the dual of the code constructed by taking $H_C \otimes H_C$ as generator matrix, is a $[n^2, n^2 - m^2]$ code with covering radius $\rho_{C^{\perp}_{H_C \otimes H_C}} = m$ (see [9]). Codewords of $C^{\perp}_{H_C \otimes H_C}$ can be seen as $n \times n$ matrices whose rows or columns are codewords in the q-ary Hamming code C.

The q-ary code $C_{G_C \otimes G_C}$, constructed from the generator matrix $G_C \otimes G_C$ is a $[n^2, (n-m)^2]$ code with covering radius $\rho_{C_{G_C \otimes G_C}} = n + 1 + 2(n-m-1)$ [9], whose codewords can be seen as $n \times n$ matrices where both rows and columns are codewords in the q-ary Hamming code C. We will refer to code $C_{G_C \otimes G_C}$ as *product code*.

There is an efficient steganographic method [7] which uses the above defined code $C_{G_C \otimes G_C}$, for $q = 2$, to embed data. Given two binary Hamming codes of the same length $n = 2^m - 1$, their product gives a linear code of length $n = (2^m - 1)^2$, dimension $(n-m)^2$ and whose codewords can be seen as $(2^m-1) \times (2^m-1)$ matrices where every row and every column are codewords in the binary Hamming code. The embedding scheme therefore consists of first taking blocks in the cover source of size $(2^m - 1) \times (2^m - 1)$, and then applying the $F5$ algorithm to every row and to the first c columns, for $1 \leq c \leq 2^{m-1} - 1$. As proved in [7], the performance one can obtain with this method is better than the one obtained by just using the conventional $F5$ algorithm on the corresponding codes with the same average distortion. We refer to [7] for further details on this method.

Now, in this paper we will proceed with a generalization of the above procedure, by taking the product of more than two q-ary Hamming codes.

As defined at the beginning of this section, let C be a q-ary Hamming code of length $n = \frac{q^m-1}{q-1}$, dimension $n-m$, with generator matrix G_C and parity check matrix H_C. Take the code C' of all the $n \times n$ matrices such that all their rows, as well as their first column, are codewords in C. Code C' is a $[n^2, n(n-m)-m]$ code, $C_{G_C \otimes G_C} \subset C' \subset C^{\perp}_{H_C \otimes H_C}$ with covering radius $\rho_{C'} = n + 1$.

For the sake of a well understanding, the following reasoning will be limited to the binary case. However, a generalization to the q-ary case is straightforward.

Just as the method based on the product of two Hamming codes from [7], this procedure consists of a row embedding and a column embedding steps. We will take the LSB bit of every grayscale symbol in the cover source and form blocks of size $n \times n$, where $n = 2^m - 1$. Let $c_{i,j}$ be the coordinate in the i-th row and j-th column of these blocks, where $i, j \in \{1, \ldots, n\}$.

1) Rows Embedding:

   The matrix encoding standard procedure [5], [10] applied to every row lets us embed $\frac{m}{n}$ bits with an average distortion of $\frac{1}{n+1}$ coordinates, thus giving a CI-rate of $(\frac{1}{n+1}, \frac{m}{n})$.

2) Column Embedding:

   After processing all rows, we can embed $\frac{m}{n}$ additional bits with an average distortion of $\frac{1}{n+1}$ by applying the same standard procedure to the first column.
   However, note that the following situations can happen when processing this column:

   - No coordinate needs to be changed in $\frac{1}{n+1}$ cases because the first column may already have, by chance, the desired value.
   - We may need to change a coordinate $c_{i,1}$ in $\frac{n}{n+1}$ cases. In this case, the i-th row may have been already modified in the corresponding row embedding with a probability of $\frac{n}{n+1}$, while it may have not been modified with a probability of $\frac{1}{n+1}$.
     Let us consider the i-th row was modified in the j-th coordinate, $c_{i,j}$, for $j > 1$. In this case, we will also have to restore the original value of $c_{i,j}$ and distort another appropiate coordinate $c_{i,k}$, for $k \in \{2, \ldots, n\}$ and $k \neq j$, such that the distortion being introduced now by the column embedding is compensated and does not affect the embedding in the i-th row (see Lemma 2 from [7]). Note that this situation is also including the case in which the coordinate that was modified during the i-th row embedding is precisely the coordinate $c_{i,1}$ we now need to change to embed data in the column. In summary, if the i-th row was modified, no matter in which coordinate, the column embedding step will introduce one distortion besides the ones introduced by the row embedding step.

     Otherwise, if during the column embedding we need to distort a coordinate $c_{i,1}$ and the i-th row was not modified, then we will also need to distort two more coordinates within the same row, $c_{i,j}$ and $c_{i,k}$, for $j, k \in \{2, \ldots, n\}$ and $j \neq k$, to make up for this distortion. Hence, the column embedding step will be now introducing three changes.

In short, we can leave invariant the average distortion of the row embedding step, but $\frac{(n+3)/(n+1)}{n+1}$ should be added for the embedding in the first column. Note that this is only a tight upper bound on the average distortion, as we will later show.

By the method just described we can embed m bits into the first column and also in every row of the matrix; therefore, we

embed $(n+1)m$ bits in n^2 coordinates. The average distortion is upper bounded by $\frac{(n+3)/(n+1)}{n+1}$ for the coordinates in the first column and $\frac{1}{n+1}$ in each of the n rows. Summing this up, the average distortion is bounded by $\frac{n\frac{(n+3)/(n+1)}{n+1}+n^2\frac{1}{n+1}}{n^2} = \frac{1}{n+1}\left(1+\frac{(n+3)/(n+1)}{n}\right)$.

The method we propose in the present paper consists of repeating over and over the same procedure we have just described. Hence, we can generalize the computations of the average distortion and the embedding rate by using ${G_C}^l = G_C \otimes (G_C \otimes \cdots \otimes G_C)$. In each step ${G_C}^l = G_C \otimes {G_C}^{l-1}$, only the first column in the first component G_C will be used to embed information.

Let D_l be the average distortion at the l-th step. As computed before, we have $D_1 = \frac{1}{n+1}$ and $D_2 = \frac{1}{n+1}(1 + \frac{(n+3)/(n+1)}{n})$. In the general case we have:

$$D_l = \frac{1}{n+1} + \xi D_{l-1},$$

where $\xi = \frac{n+3}{n(n+1)}$.

Now, the overall average distortion can be computed as $\frac{1}{n+1}\left(1+\xi+\cdots+\xi^{l-1}\right)$, which converges asymptotically very fast to

$$\frac{1}{n+1}\left(\frac{\xi^l-1}{\xi-1}\right) \rightarrow \frac{1}{n+1}\left(\frac{1}{1-\xi}\right) = \frac{1}{n+1}\left(\frac{n(n+1)}{n^2-3}\right).$$

As for the embedding rate, it can be computed as $\frac{(1+n+n^2+\cdots+n^{l-1})m}{n^l}$, which converges to

$$\frac{(1+n+n^2+\cdots+n^{l-1})m}{n^l} = m\frac{\frac{n^l-1}{n-1}}{n^l} \rightarrow \frac{m}{n-1}.$$

Finally, we obtain the asymptotical CI-rate $\left(\frac{n}{n^2-3}, \frac{m}{n-1}\right)$.

Note that we are not able to generate an embedding scheme for any CI-rate but only for natural values of m. However, given any non-allowable parameter D for the average distortion, we can always take two codes with CI-rates (D_1, E_1) and (D_2, E_2), where $D_1 < D < D_2$, such that their direct sum gives rise to a new CI-rate (D, E), with $D = \lambda D_1 + (1-\lambda)D_2$ and $E = \lambda E_1 + (1-\lambda)E_2$.

A comparison of the normalized embedding rate $e = \frac{{H_q}^{-1}(E)}{D}$, where $q = 2$, as a function of the average distortion D for the introduced Kronecker product technique (KP-technique) and the standard matrix encoding procedures [5], [10] is shown in Fig. 1. As explained before, this plot has been made by first computing the allowable points (D, e), and then applying the direct sum between the codes corresponding to two contiguous points (D_1, e_1) and (D_2, e_2), where $D_1 < D_2$.

For the sake of simplicity, some particular cases which may produce a lower distortion have been omitted in the computation of the distortion in the above CI-rate. For this reason, the distortion D in that CI-rate is an upper bound. As an example of one of these cases, recall that the column embedding step of our procedure is introducing three distortions when we need to change the $c_{i,1}$ coordinate and the i-th row was not modified in the row embedding step. Note, however, that there is one particular case in which we may need to introduce two distortions instead of three. This happens when there exist two other coordinates in the same column, $c_{j,1}$ and $c_{k,1}$, for $j, k \in \{1, \ldots, n\}$ and $j \neq k$, whose distortion is equivalent to distort only $c_{i,1}$, and both j-th and k-th rows were modified in the row embedding step. It is easy to see that we can distort coordinates $c_{j,1}$ and $c_{k,1}$ instead of $c_{i,1}$, and perform afterwards the appropiate changes to compensate these distortions in the j-th and k-th rows, respectively. Therefore, the column embedding step will be introducing two distortions besides the ones introduced in the row embedding step, and not three, as we previously stated. However, if no two other coordinates, $c_{j,1}$ and $c_{k,1}$ can be found such that both j-th and k-th rows were modified, then the column embedding step does actually introduce three distortions. We have implemented and executed a simulation of the embedding procedure described in this section which considers, among others, this particular case. For this reason the experimental results of the Kronecker product technique ("KP-technique (simulation)" in Fig. 1) have lower average distortion than the results obtained from the above CI-rate (plotted as "KP-technique" in Fig. 1).

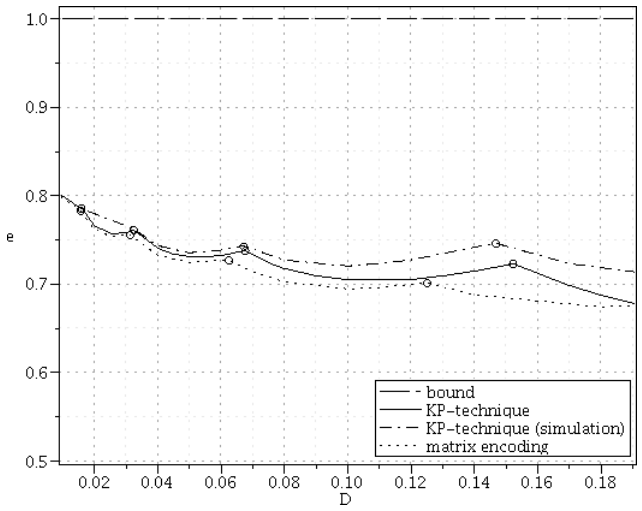

Fig. 1. Normalized embedding rate e as a function of the average distortion D of steganographic methods based on the matrix encoding procedure [5], [10], and on the Kronecker product technique, using an upper bound on the average distortion ("KP-technique") and using the experimental results ("KP-technique (simulation)").

IV. Product of perfect $\mathbb{Z}_2\mathbb{Z}_4$-linear codes

The previous procedure deals with Hamming codes. Now, we will apply it to perfect $\mathbb{Z}_2\mathbb{Z}_4$-linear codes. Let $\mathcal{C}$ be a $\mathbb{Z}_2\mathbb{Z}_4$-additive perfect code of type $(\alpha, \beta; \gamma, \delta)$ and binary length $n = 2^m - 1$, for $m \geq 2$, and let $\mathcal{H}_\mathcal{C}$ be its parity check matrix. Take the code $\mathcal{C}'$ whose codewords are all the $n \times N$ matrices, where $N = 2^{m-1}$, such that all rows are codewords in $\mathcal{C}$, and so is the first column after applying the inverse of the extended Gray map Φ.

Take blocks of $n \times N$ grayscale symbols in the source,

$$\begin{matrix} x_{1,1}, & \ldots, & x_{1,N} \\ \vdots & \vdots & \vdots \\ x_{n,1}, & \ldots, & x_{n,N} \end{matrix}$$

where $N = 2^{m-1}$, and translate them into n vectors of α binary and β quaternary coordinates, as reviewed in Section II and explained in depth in [8]. At the same time, the first coordinate of those n vectors is making up a binary vector of length n which can also be seen as a vector of α binary and β quaternary coordinates by means of the inverse of the extended Gray map Φ. Note that by considering the n rows and the first column, we end up having $n+1$ different vectors of binary length n.

The embedding procedure we will apply here is very similar to the KP-technique described in Section III. Once the $n+1$ vectors have been translated into $n+1$ vectors in $\mathbb{Z}_2^\alpha \times \mathbb{Z}_4^\beta$, we will proceed by steps: first, we will apply the embedding scheme from [8] to every row, and then we will apply it to the first column. Each distortion in the rows will involve adding or subtracting one unit to/from a grayscale symbol, and this requires considering the possibility of having extreme grayscale values problems. Recall that, unlike vectors in rows, the vector in the first column is only made up of the least significant bit of n grayscale symbols and not of their two least significant bits. This means that any distortion over a coordinate in this vector will involve a flip in the least significant bit of a grayscale symbol $x_{i,1}$, for $i \in \{1, \ldots, n\}$, which leads us to conclude that, unlike the rows embedding step, no extreme grayscale values problem will ever crop up during the column embedding step.

Furthermore, as in the KP-technique from Section III, during the column embedding step we have to consider different situations. The embedding method may require modifying a certain coordinate in the first column, and this coordinate may correspond to a row which was (or was not) modified during the row embedding step, or to a row that contains two distorted grayscale symbols, probably to deal with an extreme grayscale value problem. In any of these cases the action to be taken may vary, but still the aim is performing the appropriate changes in the affected row so that the distortion being introduced now by the column embedding step does not affect the embedding in the row. Take any two column vectors $\mathbf{h}_j, \mathbf{h}_k$ of order four in matrix $\mathcal{H}_\mathcal{C}$, such that one is the complementary of the other, that is $\mathbf{h}_j = \mathbf{h}_k + \mathbf{2}$, where $\mathbf{2}$ is the all-twos vector. The changes above mentioned will consist of considering that any distortion in coordinate $x_{i,1}$, for any $i \in \{1, \ldots, n\}$, can be compensated either by doing $x_{i,j-(\alpha+1)/2} + 1$ and $x_{i,j-(\alpha+1)/2} + 1$ or by doing $x_{i,j-(\alpha+1)/2} - 1$ and $x_{i,j-(\alpha+1)/2} - 1$. Note that, whenever possible, we will avoid modifying those grayscale symbols associated with column vectors in $\mathcal{H}_\mathcal{C}$ that are complementary of themselves, because in these cases we would have to distort the associated symbol in two units instead of one, which would not conform to ± 1-steganography.

By means of this method we can embed m bits into the first column $x_{1,1}, \ldots, x_{n,1}$ and also in every row of the block. Since the first column is made up of n grayscale symbols and each row is made up of N symbols, we are actually embedding $(n+1)m$ bits in $nN = n(n+1)/2$ symbols. It is easy to see that an upper bound of the average distortion for the symbols in the first column is $\frac{(n+3)/(n+1)}{(n+1)}$. As for the symbols in each row, the average distortion is given by $\frac{2N-1+\frac{N-1}{2^{B-2}}}{2N^2} = \frac{2n+\frac{n-1}{2^{B-2}}}{(n+1)^2}$ (see Section II). Summing this up, an upper bound for the average distortion is

$$\frac{n\frac{(n+3)/(n+1)}{(n+1)} + \frac{n(n+1)}{2}\frac{2n+\frac{n-1}{2^{B-2}}}{(n+1)^2}}{n(n+1)/2} =$$

$$\frac{2n+\frac{n-1}{2^{B-2}}}{(n+1)^2}\left(\frac{n+3}{(n+\frac{n-1}{2^{B-1}})(n+1)}+1\right).$$

In a similar way as we did in Section III, we can repeat this method over and over and generalize the computations of the average distortion and the embedding rate by taking the code whose codewords are all the l-dimensional matrices, where $l = n \times (n \times \cdots \times n \times N)$, such that their rows and the first component of every dimension are codewords in the $\mathbb{Z}_2\mathbb{Z}_4$-additive perfect code $\mathcal{C}$.

Let D_l be the average distortion at the l-th step. For the first steps we have $D_1 = \frac{2n+\frac{n-1}{2^{B-2}}}{(n+1)^2}$ and $D_2 = \frac{2n+\frac{n-1}{2^{B-2}}}{(n+1)^2}\left(\frac{n+3}{(n+\frac{n-1}{2^{B-1}})(n+1)}+1\right)$. In the general case we have:

$$D_l = \frac{2n+\frac{n-1}{2^{B-2}}}{(n+1)^2} + \xi D_{l-1},$$

where $\xi = \frac{n+3}{(n+\frac{n-1}{2^{B-1}})(n+1)}$.

Now, the overall average distortion can be computed as $\frac{2n+\frac{n-1}{2^{B-2}}}{(n+1)^2}\left(1+\xi+\cdots+\xi^{l-1}\right)$, which converges asymptotically very fast to

$$\frac{2n+\frac{n-1}{2^{B-2}}}{(n+1)^2}\left(\frac{1}{1-\xi}\right).$$

As for the embedding rate, it can be computed as $\frac{(1+n+n^2+\cdots+n^{l-1})m}{Nn^{l-1}}$, which converges to $\frac{mn}{N(n-1)}$.

Finally, we obtain a CI-rate of $\left(\frac{2n+\frac{n-1}{2^{B-2}}}{(n+1)^2}(\frac{1}{1-\xi}), \frac{mn}{N(n-1)}\right)$.

Fig. 2 shows a comparison of the normalized embedding rate $e = \frac{{H_q}^{-1}(E)}{D}$, for $q = 3$, as a function of the average distortion D for the steganographic method based on perfect $\mathbb{Z}_2\mathbb{Z}_4$-linear codes [8], the one based on ternary Hamming codes [6], [11] and the new method based on the product of perfect $\mathbb{Z}_2\mathbb{Z}_4$-linear codes. Recall that the distortion we have computed in the above CI-rate is an upper bound on the average distortion, meaning that lower distortion can be achieved in some particular cases, as it happened in the simulation results from Section III.

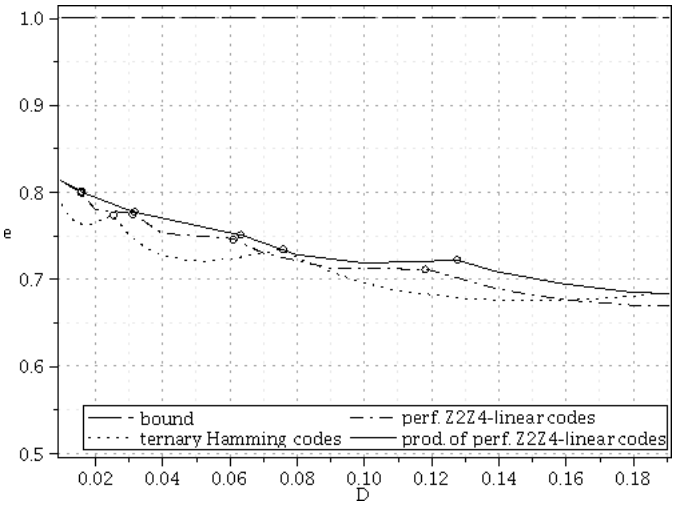

Fig. 2. Normalized embedding rate e as a function of the average distortion D, of steganographic methods based on perfect $\mathbb{Z}_2\mathbb{Z}_4$-linear codes [8] ("perf. Z2Z4-linear codes"), on ternary Hamming codes [6], [11] and on the product of perfect $\mathbb{Z}_2\mathbb{Z}_4$-linear codes ("prod. of perf. Z2Z4-linear codes").

V. Conclusions

The use of perfect $\mathbb{Z}_2\mathbb{Z}_4$-linear codes in ± 1-steganography was first proposed in [8]. This method has a better performance compared to those based on the direct sum of ternary Hamming codes from [6] and [11], and also deals with boundary grayscale values more efficiently, because unlike methods in [6] and [11], no changes of magnitude greater than one are ever made.

In this paper we have presented a technique based on products of these perfect $\mathbb{Z}_2\mathbb{Z}_4$-linear codes. Therefore, the proposed method has all the advantages related to the performance and the processing of extreme grayscale values compared to the techniques based on the direct sum of ternary Hamming codes. Furthermore, we have shown that it performs better than the method in [8].

References

[1] J. Bierbrauer, "Crandall's problem" (Available from http://www.ws.binghamton.edu/fridrich/covcodes.pdf), 1998.

[2] J. Bierbrauer and J. Fridrich, "Constructing good covering codes for applications in steganography", in Transactions on Data Hiding and Multimedia Security III, vol. 4920 of Lecture Notes in Computer Science, pp. 1-22, Springer, Berlin, Germany, 2008.

[3] J. Borges, C. Fernández, J. Pujol, J. Rifà and M. Villanueva, "Z2Z4-linear codes: generator matrices and duality", Desings, Codes and Cryptography, vol. 54(2), pp. 167-179, January, 2010.

[4] J. Borges and J. Rifà, "A characterization of 1-perfect additive codes", IEEE Trans. Information Theory, vol. 45(5), pp. 1688-1697, 1999.

[5] R. Crandall, "Some notes on steganography", (Available from http://os.inf.tu-dresden.de/∼westfeld/crandall.pdf), 1998.

[6] J. Fridrich and P. Lisoněk, "Grid colorings in steganography", IEEE Trans. Information Theory, vol. 53(4), pp. 1547-1549, April, 2007.

[7] H. Rifà-Pous and J. Rifà, "Product perfect codes and steganography", Digital Signal Processing, vol. 19(4), pp. 764-769, July, 2009.

[8] H. Rifà-Pous, J. Rifà and L. Ronquillo, "Perfect $\mathbb{Z}_2\mathbb{Z}_4$-linear codes in steganography", (Available from http://arxiv.org/abs/1002.0026v2), February, 2010.

[9] J. Rifà and V. Zinoviev, "New completely regular q-ary codes based on Kronecker products", IEEE Trans. Information Theory, vol. 56(1), pp. 266-272, January, 2010.

[10] A. Westfeld, "High capacity despite better steganalysis (F5 - A steganographic algorithm)", vol. 2137 of Lecture Notes in Computer Science, pp. 289-302, Springer-Verlag, 2001.

[11] F.M.J. Willems and M. van Dijk, "Capacity and codes for embedding information in grayscale signals", IEEE Trans. Information Theory, vol. 51(3), pp. 1209-1214, March, 2005.