An Adaptive Matrix Embedding Technique for Binary Hiding With an Efficient LIAE Algorithm

JYUN-JIE WANG, HOUSHOU CHEN
Department of Electrical Engineering,
National Chung Hsing University
Taichung 402, Taiwan, ROC
fractals_jie@pchome.com.tw; d9464108@mail.nchu.edu.tw

and CHI-YUAN LIN*
Department of Computer Science and Information Engineering
National Chin-Yi University of Technology
Taichung 411, Taiwan, ROC
chiyuan@ncut.edu.tw
*Corresponding Author

Abstract: - Researchers have developed a great number of embedding techniques in steganography. Matrix embedding, otherwise called the binning scheme, is one such technique that has been proven to be an efficient algorithm. Unlike conventional matrix embedding, which requires a maximum likelihood decoding algorithm to find the coset leader, this study proposes an adaptive algorithm called the linear independent approximation embedding (LIAE) algorithm. There are numerous concerns with the cover location selection, such as less significant cover to be modified, alterable part of the cover and forced the cover to be modified, when embedding a secret message into the cover. The LIAE algorithm has the ability to perform data embedding at an arbitrarily specified cover location. Therefore, the embedded message can be identified at the receiver without incurring any damage to the associated cover location. The simulation results show that the LIAE embedding algorithm has superior efficiency and adaptability compared with other suboptimal embedding algorithms. Moreover, the experimental results also demonstrate the trade-off between embedding efficiency and computational complexity.

Key-Words: - Steganography, matrix embedding, ML decoding, coset leader, embedding efficiency, linear block code

1 Introduction

Steganography is a crucial approach for issues of secure communication [1]-[3]. For security concerns, steganography must meet the requirement of being statistical undetectability. Embedding capacity and embedding distortion are two critical subjects in steganography, and are involved in an effective steganographic technique referred to as the binning scheme. Although the number of approaches to hide data is considerable [4], [5], this study focuses on the steganography technique.

Data classification is reached using a parity check matrix in a binning method, also referred to as matrix embedding (ME) [3], [6]. Matrix embedding using linear block codes, also called syndrome codes [7] or coset codes [8]. [3], [6], [7], [8] embed and extract a message by using the parity check matrix of linear block codes. [9] generalized the concept of matrix embedding and defined the codes with the parity check matrix H as steganographic codes, so called stego codes. Some special cases, involve constructive and fast embedding algorithms [10]-[13]. [10] and [11] proposed two schemes, called tree-based parity check (TBPC) and block-overlapping parity check (BOPC), to reduce distortion on a cover object based on some special structures. TBPC and BOPC are both very efficient issue for steganography. [14] utilized a majority vote strategy to further improve the computational complexity of TBPC. [12] presented a steganographic scheme capable of concealing a large amount of data in a binary image. The scheme uses a secret key and a weight matrix to increase the security, it uses a weight matrix to increase the embedding rate, and it uses an XOR operator to decrease the time complexity. Moreover, [13] proposed a high embedding efficiency scheme,

which was an ME-based embedding technique for large payloads, demonstrated by simple codes. [13] resulted in superior steganographic security for large payload. In addition, [13] used structured simple codes, that is, decoding by using fast and efficient Hadamard decoding that is suitable for large code length, to achieve the efficient ME codes and to approach the embedding bound for large payload. However, the schemes proposed by [10]-[13] are either less efficient or computationally expensive. Generally, embedding data using a parity check matrix results in an average embedding distortion that is superior to using methods [10]-[12] that do not require such a structured matrix. This performance difference is due to the nature of linear block codes. Moreover, the receiver can simply extract the secret message using multiplication operations between the parity check matrix and received sequence. In principle, ME methods can be evaluated with respect to theoretically achievable bounds [6].

Because of rapid developments in multimedia, the requirements for multimedia messages have become increasingly demanding. For instance, regarding messages of significance that are not permitted to be altered, when performing embedding through ME, message blocks must remain intact. In another case, with the cover in the transmitter, a superior distortion metric is obtained if the secure message is embedded into the specific region of the cover. Accordingly, this study proposes an adaptive ME algorithm to assign certain blocks for embedding with messages that cannot be altered. Although an adaptive ME algorithm has inferior embedding capacity, it has significant importance in quantization, filtering, lossy compression, dither, sampling, and other situations. The chief concern with this signal processing is not the integrity of the original cover signal; the adaptive ME algorithm combines secure messages with signal processing. For example, with JPEG compression, the adaptive ME algorithm can embed some information into coefficients defined in the frequency domain when handling the truncation error inevitably encountered during the quantization of DCT conversion coefficients.

For matrix embedding, finding the stego with minimum distortion is difficult. A search for the toggle, that is, the altered sequence corresponding to the cover, of the minimum Hamming distance is equivalent to linear block codes decoding problem, or a binning scheme problem. This study proposes an adaptive algorithm to reduce the complexity of ME and to improve the embedding efficiency of ME. Moreover, the proposed algorithm is also suitable for various applications.

The rest of this paper is organized as follows. Section 2 provides a brief description of coding theory and embedding efficiency for binary data hiding. Section 3 introduces the issue of adaptive matrix embedding. Section 4 presents the proposed suboptimal embedding algorithm. Section 5 shows the experimental results and a constructive discussion, with an analysis of the performance of various suboptimal algorithms; and finally, Section 6 offers a conclusions.

2 The bound for matrix embedding

This section presents a discussion on coding theory-related knowledge. Binary data hiding refers to a situation where the average distortion d of an embedding strategy can be determined using a (n,k) linear block code at an embedding rate $R_m = (n-k)/n$. The lower bound δ is thus estimated using the rate-distortion function $R_m = h(\delta)$ of a binary symmetric source. Thus, $\delta(R_m) = h^{-1}(R_m)$ is used to generate a bound of embedding efficiency η.

A (n,k) linear block code C is characterized using a parity check matrix $H \in \{0,1\}^{m\times n}$, where $m = n-k$ Assuming that the coding rate is $R = k/n$, the code C is of size $|C| = 2^{nR}$ .With a binary symmetric source (BSS) and an n bit source sequence $u \in \{0,1\}^n$, the average distortion per a bit is defined as

$$d = \frac{E\left[d(\hat{u},u)\right]}{n} = \frac{D}{n}, \qquad (1)$$

where $\hat{u}$ represents a quantized codeword existing in code C , and D is the average hamming distortion between $\hat{u}$ and u per each block. For a good (n,k) linear block code, the equation is defined approximately as follows:

$$h(\delta) \approx 1 - \frac{k}{n} = 1 - R_s = \frac{m}{n}, \qquad (2)$$

where $h(\delta) = \delta\log_2(1/\delta) + (1-\delta)\log(1/(1-\delta))$ denotes a binary entropy function, that is, the optimal embedding rate R_m at the low bound δ of the distortion. Therefore, whether δ can be reached by m/n is a well-posed problem. The binning technique can solve this problem. All the binary sequences within 2^n dimensional space can be

partitioned into $\binom{n}{\delta_n}$ bins, approximated as $2^{nh(\delta)}$ using Stirling's approximation. Considering binary source coding, each bin is of $2^{n[1-h(\delta)]}$ sequences corresponding to a coding rate k/n. Producing a parity check matrix with a well-structured (n,k) linear block code and a coding rate $R_s = k/n$ remains of significant concern. Furthermore, with an embedding rate requested in such a linear block code as C, this equation can be rewritten as $h(\delta) \approx 1 - k/n = m/n$. On account of $m \approx nh(\delta)$, 2^m cosets are employed as an approach to reach R_m. For an (n,k) linear block code, the minimum average distortion is up to

$$\delta = h^{-1}(m/n) = h^{-1}(R_m), \tag{3}$$

where $h^{-1}(*)$ is the inverse function of the binary entropy function h. Equation (3) is referred to as the rate-distortion function. The low bound δ of average distortion for each bit in a code block is $\delta \le d = D/n$. When performing binary embedding to a cover sequence, the embedding efficiency is defined as

$$\eta = \frac{R_m}{d} = \frac{m}{D}. \tag{4}$$

The asymptotic upper bound is obtained using (3) and (4), as follows:

$$\eta_\delta = \frac{m}{n\delta} = \frac{m/n}{h^{-1}(R_m)} = \frac{R_m}{h^{-1}(R_m)}. \tag{5}$$

In the end, with an (n,k) linear block code C with an embedding rate $R_m = m/n$, a lower bound δ of average distortion exists, with a constraint of $0 \le \delta \le 1/2$ imposed. For the linear block code C, the embedding efficiency between both the optimal (i.e., the maximum likelihood decoding) and the suboptimal algorithms can be related as

$$\frac{m}{nh^{-1}(R_m)} \ge \frac{m}{D_{out}} \ge \frac{m}{D_{sub}}, \tag{6}$$

where D_{opt} and D_{sub} represent the average distortion estimated for each block in the optimal decoding and the suboptimal decoding, respectively. Equation (6) can be expressed in an alternative form as $\eta_\delta \ge \eta_{opt} \ge \eta_{sub}$. Thus, as the measure of efficiency, interval measure parameters are defined as

$$\varepsilon_{sub} = \eta_\delta - \eta_{sub}. \tag{7}$$

For a good suboptimal embedding code, the value ε_{sub} should be as small as possible.

Assuming that the distortion of an adaptive matrix embedding is d_{apt} for an n bits cover sequence u the adaptive matrix embedding technique exhibits an inferior distortion relative to the original embedding version. This is because the alterable part contains
part of the subset of u. The average distortion $D_{apt} = nd_{apt}$ of each block for an adaptive matrix embedding is larger than that of the original embedding version. However, the embedding algorithm achieves optimal embedding distortion D_{opt} or suboptimal embedding distortion D_{sub}.

3 The solution for adaptive matrix embedding

Although the purpose of adaptive matrix embedding is dissimilar to that of conventional matrix embedding, we can provide an identical algebraic description. A binary matrix embedding scheme can be considered a problem in which the embedder must quantize a binary symmetric source to render the expected quantization error as small as possible. We can employ linear block codes to solve the quantization problem. This section proposes a solution for binary matrix embedding and shows that this solution can be classified as an optimal solution or suboptimal solution.

3.1 Maximum likelihood decoding

This study demonstrates binary data embedding using a standard array as follows: With an (n,k) linear block code C, we built a standard array with a size of $2^{n-k} \times 2^k$, as shown in Fig. 1.

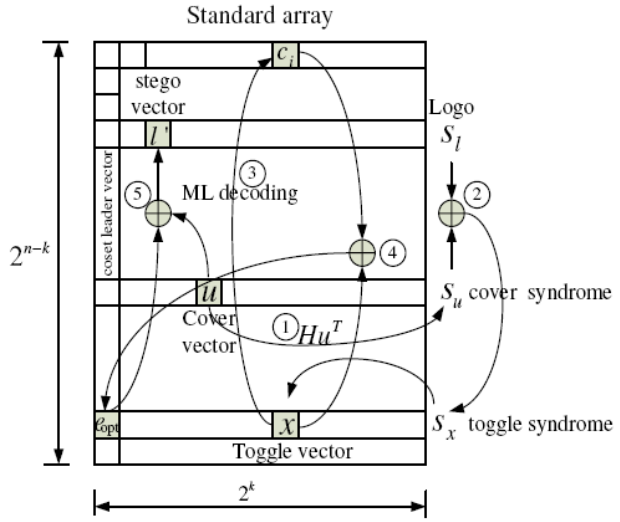

Fig. 1. Embedding procedure.

Alternatively, the required coset leader can be found precisely to perform binary data embedding or optimal embedding. A (n,k) linear block code C can be characterized with a parity check matrix H of size $(n-k)\times n$ as

$$C=\{r \mid Hr^T=0\}, \tag{8}$$

where the sequence is $r\in F_2^n$. The basic concept of the standard array is that a space F_2^n is partitioned into 2^k disjoint subsets based on the linear block code $C=\{c_1,\cdots,c_{2^k}\}$. Based on (8), the syndrome s of the sequence r is defined as $s=Hr^T$. Furthermore, the set composed of all the sequences r corresponding to the identical s is referred to as the coset of code C, and is defined as

$$C^s=\{r \mid Hr^T=s\}=\{c+e \mid c\in C\}, \tag{9}$$

where e denotes the coset leader in the standard array. The term s can be derived through H from an arbitrary sequence r, and e can be expressed by an ML decoding function as

$$e=f(Hr^T)=f(s), \tag{10}$$

where $f(\cdot)$ represents the decoding function of the linear block codes. Determined by ML decoding, the coset leader e is added to r to recover the codeword c, which is closest to the sequence r.

3.2 Optimal solution for adaptive matrix embedding

This subsection presents a description of a linear block code and explains the use of the standard array for binary embedding. Using the standard array can describe the embedding issue with a logo message s_l and a cover sequence u, and a stego l' close to the u and corresponding to the logo message s_l. In Fig. 1, with an n bits cover sequence $u\in C^u$, an n bits stego sequence $l'\in C^l$ with a logo message $s_l\in F_2^m$ is to be found (Fig. 1). Assume that there a toggle sequence $x=l'+u$ is the distance between sequences u and l', and the toggle sequence with minimum weight $x=e_{opt}\in C^x$. In other words, the cover sequence u and the stego sequence l' are of a minimal weight sequence e_{opt}, i.e., the coset leader in C^x. With the constraint $E\lfloor w_H(e_{opt})\rfloor\le n\delta$, where the average distortion $0\le\delta\le 0.5$, this remains an issue that is referred to as the binning problem. Suppose that the toggle sequence $x\in C^x$ exists, and C^x represents a coset in F_2^n. Seek x with the minimal weight, i.e., $x=e_{opt}$. From the decoding viewpoint, the coset leader e_{opt} can be discovered through decoding function, expressed as

$$\begin{aligned} e_{opt}&=f_{opt}(s_u+s_l)\\ &=f_{opt}(s_x). \end{aligned} \tag{11}$$

To obtain the optimal x, i.e., e_{opt}, the following maximum likelihood decoding is used to achieve

$$\begin{aligned} e_{opt}&=\arg\min_{x\in C^x} w_H(x)\\ &=\arg\min_{c\in C} d_H(c,x)+x. \end{aligned} \tag{12}$$

Once discovered, the coset leader e_{opt} is added to the cover sequence u as $l'=u+e_{opt}$. Essentially, l' is the stego sequence closest to the cover sequence u within F_2^n dimensional space, and contains the logo message s_l. The procedure of finding the coset leader e_{opt} in the toggle coset C^x is the optimal solution for adaptive matrix embedding.

Because of the constraint imposed on the location selection in an adaptive embedding algorithm, the optimal solution may not be the least weight sequence $x = e_{opt}$ in the coset C^x, whereas the intended toggle sequence is located using the ML algorithm. Suppose that a toggle sequence $x = (x_1, \cdots, x_n)$, an index set $S \subseteq \{1, 2, \cdots, n\}$, and the alterable cover locations in u are confined to u_i, $i \in S$ then e_{opt} is no loner the optimal modification sequence, but is instead defined as

$$e_{opt} = \arg\min_{x \in C^x} w_H(x), \tag{13}$$

where $\{x_i = 0 | i \notin S\}$. The determination of e_{apt} is dependent on the location of S, i.e., e_{apt} may not exist. In other words, a search is conducted within the confined region C^x for the intended toggle sequence x. The adaptive optimal embedding to embed a binary logo message s_l is as follows:

Algorithm ML algorithm for Adaptive ME

1. With alterable cover locations $S \subseteq \{1, 2, \cdots, n\}$, the n bits cover sequence u and the m bits logo message s_l.
2. Calculate the syndrome
$$s_u = Hu^T.$$
3. That which is derived from s_l is added to s_u to obtain s_x.
4. The sequence $x = (x_1, \cdots, x_n)$ corresponding to s_x is then decoded by applying the ML algorithm to a codeword c as follows:
$$e_{apt} = x + \arg\min_{c \in C} d_H(c_i, x),$$
where $x_i = 0$ and $i \notin S$.
5. Modify the cover sequence u to obtain stego l' so that
$$l' = u + e_{apt}.$$
6. In the receiver, extract the logo message using
$$s_l = Hl'^T.$$

Once e_{apt} is known, the optimal adaptive stego sequence l' can be discovered. However, finding e_{apt} is difficult in a (n, k) linear block code C with a sufficiently large length because the complexity of ML decoding increases as 2^k. The following section proposes a suboptimal embedding algorithm to supplant the ML algorithm and resolve the issue.

4 Adaptive suboptimal embedding algorithm

The section presents an efficient algorithm to perform the binary data embedding. As shown in section 3, building a standard array that corresponds to an arbitrarily large linear block code is unrealistic because the embedding algorithm cannot enable embedding using an exhaustive search. Thus, a fast and efficient suboptimal embedding algorithm must be developed.

4.1 Suboptimal embedding algorithm

The proposed algorithm locates a toggle sequence by using a different method than that of the conventional ML decoding algorithm; it uses a simple technique to search for a low-weight toggle sequence. This algorithm is designed to locate a suboptimal toggle sequence e_{sub}, where $w_H(e_{sub}) \geq w_H(e_{opt})$ within the toggle coset C^x. However, $w_H(e_{sub})$ must be as close to $w_H(e_{opt})$ as possible. Note that e_{sub} is a sequence defined in C^x. Finally, the stego sequence l'' obtained by adding e_{sub} to the cover sequence u cannot be ensured as the optimal stego sequence. Fig. 2 shows the geometric interpretation of a suboptimal embedding algorithm.

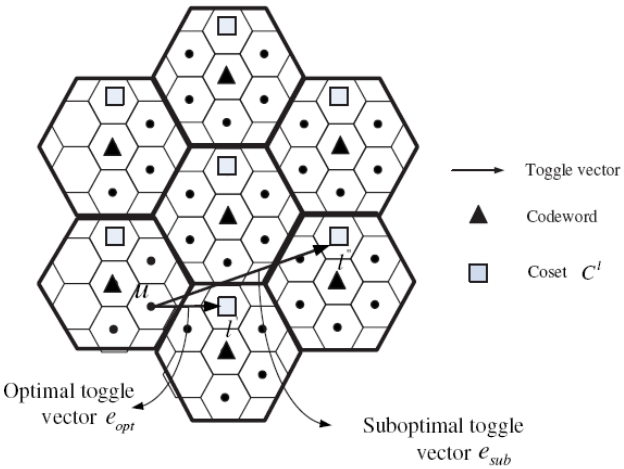

Fig. 2. Geometric interpretation of suboptimal embedding algorithm.

A number of researchers have implemented binary data embedding by using the suboptimal embedding algorithm [10], [11], [12]. The following discussion presents an example that does not require a parity check matrix. As proposed by [12], the binary embedding algorithm is of an embedding capacity that is dependent on the partitioned cover sequence of size $m \times n$ into which the $\log_2(mn+1)$ number of 1s can be embedded, and a maximum of 2 bits can be altered. As Oscar, et al. [10] proposed in 2007, the tree-based parity check (TBPC) is a binary data embedding algorithm with an embedding capacity of up to $2^{m-1}/(2^m-1)$, where m is an integer larger than 2. The TBPC provides high embedding capacity and fast computational time, up to approximately half the cover sequence length, with a maximal embedding distortion of up to 0.36 change/ bit. In 2007 Oscar et al. proposed a second version of the binary image embedding algorithm, called block-overlapping parity check BOPC [11]. The BOPC algorithm uses the properties of block-overlapping parity check to reduce toggling required in [11]. Although approaches for suboptimal embedding are numerous, such as those proposed by [10], [11] and [12], previous algorithms have been unable to embed a message into a number of adaptive locations. The following subsection details a suboptimal and adaptive matrix embedding technique.

4.2 Suboptimal solution and LIAE algorithm

This subsection presents a description of the proposed adaptive suboptimal algorithm and demonstrates the algorithm by using an example. For optimal embedding, a $(2^m-1, 2^m-1-m, 3)$ Hamming code employs a parity check matrix H to hide the logo message s_l. Using H to estimate the cover syndrome $s_u = Hu^T$, a Hamming code is used to discovers the difference $s_x = s_l + s_u$ between s_u and the logo message s_l, which is intended for embedding. Finally, a column vector that is identical to s_x of H is located, and the corresponding cover position is altered accordingly. Nevertheless, the altered sequence x, i.e., the toggle sequence, is not unique. We may select an arbitrary sequence x within coset C^x as the toggle sequence. In other words, the stego sequence can be expressed as $l' = f(s_x) + u$ in the sequence domain corresponding to the syndrome domain, where $f(\cdot)$ denotes a decoding function. For $s_l \neq s_u$, the Hamming code merely requests the corresponding parity check matrix to alter a maximum of 1 bit to embed the logo messages. If $s_l = s_u$, i.e., the syndrome of the cover is exactly the same as the logo message s_l, altering any bit location in the cover is unnecessary. The receiver extracts the logo messages by evaluating the equation $H(l')^T = s_l$. Considering an adaptive location problem, with a set $S \subseteq \{1, 2, \cdots, n\}$ as the alterable bit location index, the equation $s_l + Hu^T$ is solved for a legitimate $x = (x_1, \cdots, x_n)$, where $x_i = 0$ and $i \notin S$ by applying the Gaussian elimination method and searching for the minimum distortion solution. For example, adaptive data embedding is illustrated using a $(7, 4)$ Hamming code with a parity check matrix expressed as

$$H = \begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} \phi_1 & \phi_2 & \phi_3 & \phi_4 & \phi_5 & \phi_6 & \phi_7 \end{bmatrix}, \quad (14)$$

where h_i denotes the column vector of H. The embedding algorithm by the parity check matrix H embeds 3 bits of logo message s_l into the 7-bit cover sequence u. The ME algorithm first identifies the difference s_x between the syndrome of the cover sequence u and the logo message s_l intended for embedding, and then the coset leader corresponding to s_x. For instance, for the cover sequence $u = (1100100)$ and the logo message $s_l = (110)^T$, the syndrome of u is $s_u = Hu^T = (011)^T$, and the difference is $s_u = (011)^T$ and $s_l = (110)^T$; hence, $s_x = (101)^T$. The equation

$$Hx^T = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$$

is solved for the least weight sequence x. If all the bits within the cover sequence u are permitted to be

altered, x is then discovered as (0000100), and that the 5th bit is modified to embed s_l. Essentially, the operation is tantamount to adding the 5th column vector within H to s_u. Assuming that the alterable bit location set within $\{u_i | i \in S\}$ and $S = \{2, 3, 6, 7\}$, i.e., the bits 1, 4, and 5 are those disallowed to be altered, that is the toggle sequence $x = \{x_1, \cdots, x_7\}$, where $\{x_i = 0 \,|\, i = 1, 4\}$, determining whether a linear combination exists between column vectors 2, 3, 6, and 7 is necessary to form the toggle syndrome $s_x = (101)^T$, and the difference between $s_u = (011)^T$ and $s_l = (110)^T$. Note that, $\phi_5 = \phi_3 + \phi_6 = s_u + s_l$, i.e., $(110)^T + (011)^T = (101)^T$, in this case is an approach to embed s_l into the cover sequence u. However, in most cases, yielding a column vector of H as a linear combination of others in the same manner is unlikely. Considering n bits of the cover sequence u, an index set S corresponds to the column vectors for the selection of an $m \times n$ matrix H. For $|S| \geq m$, the size of the set $\phi = \{\phi_i | i \in S_\phi, S_\phi \subseteq S\}$ composed of linearly independent column vectors within S, the required toggle sequence is a linear combination of the linearly independent set ϕ, with m bits of coefficient λ. Subsequently, this study proposes an effective approach to meet the requirement of the adaptive ME algorithm. During data embedding, the equation $Hx^T = s_x$ is derived to search for the least weight solution x. The equation can be solved using the LIAE algorithm as follows: Suppose that $(n-k)$ linearly independent row vectors are within the parity check matrix H, i.e., $Rank(H) = n - k$ of an (n,k) linear block code, $S \subseteq \{1, 2, \cdots, n\}$, $|S| \geq m$, $S_\phi \subseteq S$, and $|S_\phi| = m$. Randomly selected from H and verified as linearly independent, the m column vector $\phi = \{\phi_i | i \in S_\phi\}$, where $|\phi| = m$, is employed as a basis for representing an arbitrary m bit toggle s_x. Assuming that an m bit syndrome $s_x = (s_{x,1}, \cdots, s_{x,m})$ corresponding to H exists, it can be expressed as a linear combination of ϕ.

$$s_x = \sum_{i \in S_\phi} \lambda_i \phi_i = \phi \lambda^T = \left(s_{x,1}, \cdots, s_{x,m}\right)^T. \quad (15)$$

With ϕ, and therefore, s_x, the coordinates $\lambda = \{\lambda_i\}$ corresponding to the basis $\phi = \{\phi_i\}$ can be evaluated as

$$\lambda^T = \phi^{-1} s_x. \quad (16)$$

Assume that an n bit sequence $x = (x_1, \cdots, x_n)$ is expressed as

$$x_i = \begin{cases} \lambda_i & , i \in S_\phi \\ 0 & , i \notin S_\phi \end{cases} \quad (17)$$

Therefore,

$$Hx^T = \phi \lambda^T = s_x. \quad (18)$$

The original equation $Hx^T = s_x$ can also be resolved for x with a linear combination of column vectors from H. The next task is to discover the least weight $w_H(x)$ corresponding to the minimum embedding distortion. Therefore, the least weight coordinate vector λ, associated with a randomly selected basis ϕ, is described as follows:

Algorithm Adaptive LIAE algorithm

Encoder: Given a (n,k) linear block code with $H = [\phi_1 \cdots \phi_i \cdots \phi_n]$, $Rank(H) = m$, cover sequence u, m bits logo message s_l, alterable location index S, basis index $S_\phi \subseteq S$, and constant B. First, calculate the s_x syndrome using $s_x = s_l + Hu^T$ and $s_x = Hx^T = \phi \lambda^T$

1. Let $j = 1$, and randomly select m column vectors from H according to index S as $\phi^{(j)} = \left\{\phi_i^{(j)} \middle| i \in S_\phi^{(j)}\right\}$.
2. Determine whether $\phi^{(j)}$ is linearly independent as follows: $\det\left(\phi^{(j)}\right) \neq 0 \rightarrow s_l = \sum_{i \in S_\phi^{(j)}} \lambda_i^{(j)} \phi_i^{(j)}$ and find $\lambda^{(j)} = \left\{\lambda_i^{(j)}\right\}$. For $\det\left(\phi^{(j)}\right) = 0$, i.e., nonexistent solvability, return to Step 1.
3. In the event that $j = B$, then $\lambda = \left\{\lambda^{(1)}, \lambda^{(2)}, \cdots, \lambda^{(B)}\right\}$ Proceed to Step 4. Otherwise, $j = j + 1$, and return to Step 1.

4. Select a minimum coefficient vector from the candidate set λ as
$$\lambda' = \arg\min_{\lambda^{(l)} \in \lambda} w_H\left(\lambda^{(l)}\right).$$
5. Set a toggle sequence as $x = (x_1, \cdots, x_n)$, where
$$x_i = \begin{cases} \lambda'_i & , i \in S_\phi \\ 0 & , i \notin S_\phi. \end{cases}$$
6. Find the adaptive stego sequence as $l' = x + u$

Decoder: Recover the logo message s_l by y and H.
$$\begin{aligned} \hat{s} &= Hl'^T \\ &= H(x+u)^T \\ &= \phi\lambda^T + Hu^T \\ &= s_l \end{aligned}$$
7. The embedded data are then extracted by performing
$$s_l = Hl'^T.$$

5 Simulation Results

The experimental results in this study demonstrate the performance of various embedding algorithms and the application of the LIAE algorithm. The algorithms were simulated to test their embedding efficiency. The programs were developed using MATLAB-R2007, and executed in CPU-Intel E8300 2.83G on a computer with 2G DRAM. In the experiment, the cover U and logo message s_l were uniformly and randomly selected. The $n \times N$ cover U was divided into the N non-overlapping blocks, where each block u is $1 \times n$ in size, and each block u is embedded according to the logo message s_l of m bits.

1) Computational complexity

Table 1 shows the speed and operation of embedding for various suboptimal embedding algorithms with fixed 10^5 random logo messages. The LIAE algorithm incurs constant complexity for the number B of LI bases. By contrast, the complexity cost of the maximum likelihood embedding algorithm plays an important role in evaluating optimization when the algorithm is performed to locate the optimal toggle sequence or the coset leader. Table 1 shows that, compared to using the embedding efficiency of the ML embedding algorithm, the performance of the LIAE algorithm is poor, less than 0.3 for random code cases. Although the embedding efficiency of the LIAE algorithm suffers a loss to a certain degree, its computational complexity is superior to that of the ML embedding algorithm, according to the number B of LI bases. By introducing the LIAE algorithm, we efficiently reduce the time complexity as shown in Table 1. Obviously, the proposed LIAE algorithm outperforms these methods in [12], [13], but provides fewer embedding efficiency.

TABLE 1
THIS TABLE COMPARES THE PERFORMANCE AND TIME OF VARIOUS EMBEDDING ALGORITHMS.

m	R_m	η	sec
[10]	0.242	2.7957	0.38
[11]	0.5161	3.6056	9.41
[12]	0.125	2.6244	74.33
[13], $k=14$	0.9991	2.0585	2405
[13], $k=16$	0.9998	2.0361	8691
[13], $k=18$	0.9999	2.0185	34100
[13], $k=20$	0.9999	2.0104	135095
ML Random(24,12)	0.5	3.3904	1013
ML Random(24,18)	0.25	3.7714	74754
LIAE Random(24,12), B=10	0.5	3.1352	99
LIAE Random(24,18), B=10	0.25	3.5568	52
LIAE Random(24,12), B=100	0.5	3.3334	917
LIAE Random(24,18), B=100	0.25	3.5762	541

2) Solving probability

Solutions using the LIAE algorithm for the random binary matrix yield a high probability. Consider the issue of solvability of the random matrix H with an embedding rate $R_m = 1/2$, and determine the probability of solvability. The term H is a binary $m \times n$ random matrix consisting of n columns. The term H can be deleted from a number of columns to form an $(n, k-i)$ random block embedding code, where i is the number of deleted columns. For $m = 16$, 14, 12, 10, and 8, the parity check matrix H of the LIAE algorithm has a solution of approximately 0.3 in Fig. 3. This probability of solutions decreases when increasing the likelihood

for randomly deleting arbitrary columns of H.

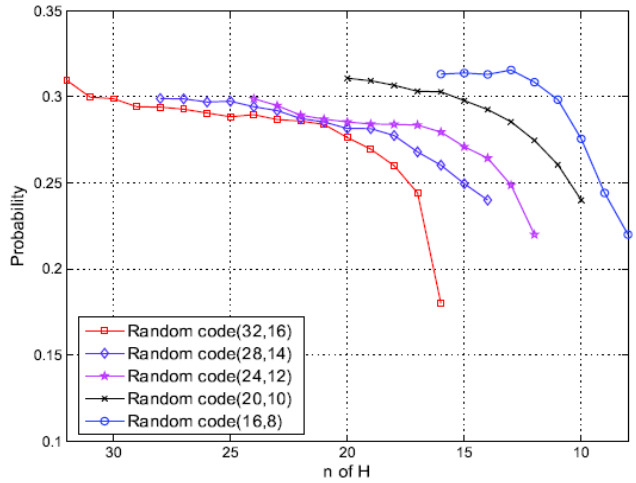

Fig. 3. Solving probability for performing LIAE algorithm.

3) The embedding efficiency η vs. the number B of candidate basis for LIAE algorithm

The experiments in this study involved using a $(16, 8)$ random embedding code to performing the LIAE algorithm. Each data point shown in Fig. 4 is the average embedding efficiency of $N = 10^5$ randomly cover.

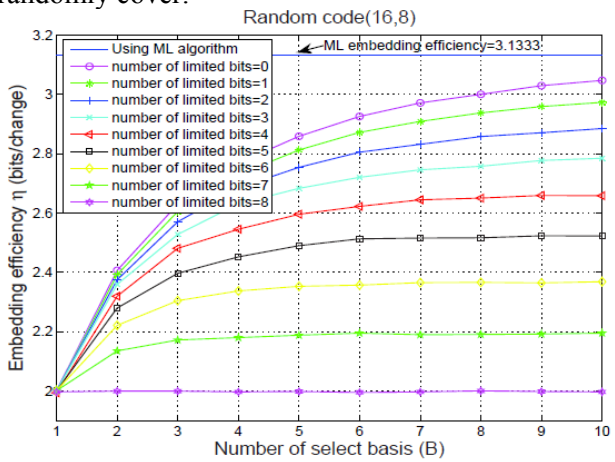

Fig. 4. The embedding efficiency of the LIAE algorithm with respect to the number B of L.I. sets.

This figure demonstrates two objectives. The first objective is to show the embedding efficiency η for different numbers B of the candidate basis for the LIAE algorithm, with B ranging from 1 to 10. The embedding efficiency η increases in conjunction with the number B. The results show that a large number B has an insignificant effect on the embedding efficiency η. The second objective is to show that the forbidden altering location is from 0 to 8. Whereas the forbidden altering location is large, the embedding efficiency η decreases with the number of locations.

4) The embedding efficiency for suboptimal algorithm

Figs. 5 and 6 show a comparison of the embedding efficiency of various suboptimal algorithms for different inverse embedding rate n/m. Each point was obtained by performing these suboptimal algorithms with an $N = 10^5$ random cover block. The LIAE and adaptive LIAE algorithms are proposed to embed the m bits secret message for each code block from exiting codes. The forbidden altering random locations for the adaptive LIAE algorithm is 20% of the length n. Figs. 5 and 6 show that the LIAE algorithm is more efficient than that proposed in [10], [11], [12], and [13]. Even when the LIAE algorithm is adaptive, its embedding efficiency remains superior to that of the other suboptimal algorithms. Table 2 shows a comparison of the performance of various embedding algorithms regarding on the differing efficiency ε_{sub} between the optimal ML embedding algorithm and various suboptimal embedding algorithms. For a good suboptimal embedding code, the value ε_{sub} should be as small as possible. Table 2 shows a further investigation of the sensitivity of the various codes, independent bases, and forbidden altering locations to the ε_{sub}. The results in Table 2 demonstrate that the value ε_{sub} of the LIAE and adaptive LIAE algorithms are also superior compared to those proposed in [10], [11], [12], and [13] at the same embedding rate.

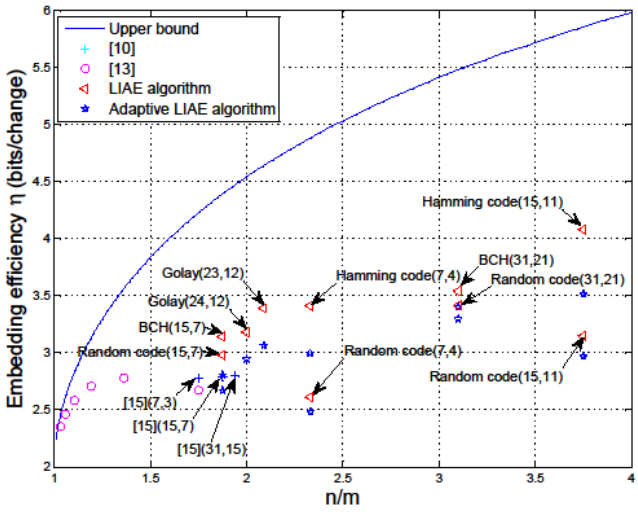

Fig. 5. Embedding efficiency versus inverse embedding rate.

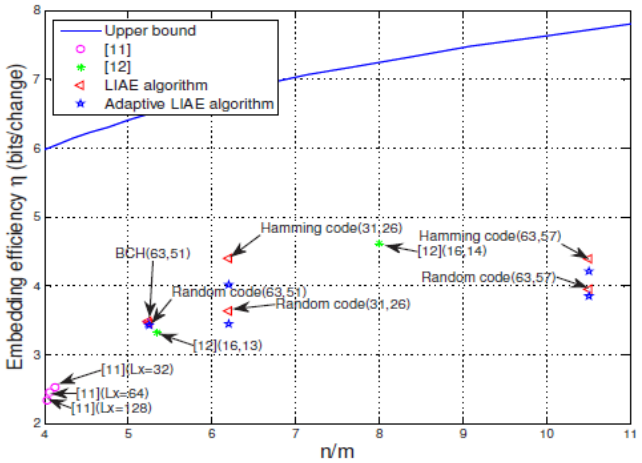

Fig. 6. Embedding efficiency versus inverse embedding rate.

Figures 7, 8, 9, and 10 show the average embedding efficiency of the LIAE algorithm in various numbers of linearly independent bases employing both Hamming codes and BCH codes. For $\mathrm{BCH}(15,7,5)$, $\mathrm{BCH}(15,11,3)$, $\mathrm{Ham}(63,51,35)$, and $\mathrm{Ham}(63,57,3)$, the embedding efficiency of the various codes is close to the ML embedding algorithm, under 100 number of LI bases B. For decoding concerns, the complexity of the LIAE algorithm does not require the degree of power that is necessary for the ML embedding algorithm. A significant number of LI bases B leads to superior embedding efficiency; however, the results in the high complexity of the LIAE algorithm are produced.

TABLE 2
THE TABLE COMPARES THE PERFORMANCE OF VARIOUS EMBEDDING ALGORITHMS, WHERE [10], [11], [12], AND [13] WITH VARIOUS EMBEDDING RATES AND THE LIAE ALOGRITHM ARE APPLIED TO VARIOUS LINEAR BLOCK CODES (''P''=FORBID ALTERING LOCATION IN PERCENTAGE OF n AND ''B''=NUMBER OF THE CANDIDATE BASS FOR LIAE ALGORITHM).

code (n,k)	m/n	n	ε_{sub}
[13], $k=14$	0.9991	16383	0.013
[13], $k=16$	0.9998	65535	0.0086
[13], $k=18$	0.9999	262143	0.0052
[13], $k=20$	0.9999	1048575	0.0031
[12](16,14)	0.125	16	4.616
[12](64,62)	0.031	64	6.708
[12](16,13)	0.187	16	3.327
[12](64,61)	0..46	64	5.443
[11](Lx=32)	0.2424	4225	2.526
[11](Lx=64)	0.2461	16641	2.455
[11](Lx=128)	0.2481	66049	2.337
[10](7,3)	0.571	7	1.436
[10](15,7)	0.533	15	1.607
[10](31,15)	0.516	31	1.68
Hamming(7,4) (P=0%,B=10)	0.4286	7	1.4676
Hamming(15,11) (P=0%,B=10)	0.2667	15	1.7794
Hamming(31,26) (P=0%,B=10)	0.1613	31	2.4067
Hamming(63,57) (P=0%,B=10)	0.0952	63	3.3495
Hamming(7,4) (P=20%,B=10)	0.4286	7	1.8822
Hamming(15,11) (P=20%,B=10)	0.2667	15	2.3446
Hamming(31,26) (P=20%,B=10)	0.1613	31	2.789
Hamming(63,57) (P=20%,B=10)	0.0952	63	3.5302
BCH(15,7) (P=0%,B=10)	0.5333	15	1.2472
BCH(31,21) (P=0%,B=10)	0.3226	31	1.9365
BCH(63,51) (P=0%,B=10)	0.1905	63	3.0125
BCH(15,7) (P=20%,B=10)	0.5333	15	1.5829
BCH(31,21) (P=20%,B=10)	0.3226	31	2.0729
BCH(63,51) (P=20%,B=10)	0.1905	63	3.0484
Random code(7,4) (P=0%,B=10)	0.4286	7	2.27
Random code(15,11) (P=0%,B=10)	0.2667	15	2.7117
Random code(15,7) (P=0%,B=10)	0.5333	15	1.4092
Random code(31,26) (P=0%,B=10)	0.1613	31	3.1648
Random code(31,21) (P=0%,B=10)	0.3226	31	2.0619
Random code(63,57) (P=0%,B=10)	0.0952	63	3.7918
Random code(63,51) (P=0%,B=10)	0.1905	63	3.041
Random code(7,4) (P=20%,B=10)	0.4286	7	2.3969
Random code(15,11) (P=20%,B=10)	0.2667	15	2.8892
Random code(15,7) (P=20%,B=10)	0.5333	15	1.7191
Random code(31,26) (P=20%,B=10)	0.1613	31	3.3515

Random code(31,21) (P=20%,B=10)	0.3226	31	2.1776
Random code(63,57) (P=20%,B=10)	0.0952	63	3.8916
Random code(63,51) (P=20%,B=10)	0.1905	63	3.0717
Golay(23,12) (P=0%,B=10)	0.4783	23	0.7804
Golay(24,12) (P=0%,B=10)	0.5	24	0.9716
Golay(23,12) (P=20%,B=10)	0.4783	23	1.5765
Golay(24,12) (P=20%,B=10)	0.5	24	1.5958

6 Conclusions

This study proposed a suboptimal adaptive matrix embedding algorithm with low operation complexity, and applied it to an arbitrary selection of cover locations. Although the proposed scheme has decreased embedding efficiency, it can function as an adaptive matrix embedding method for various applications. The proposed method also provides a fast and efficient embedding method by using the LIAE algorithm for arbitrary cover. Based on the LIAE algorithm, the number of candidate toggle sequences is unnecessary 2^k as the maximum likelihood algorithm. For a sufficiently large linear block code, the LIAE algorithm is also more efficient compared to the conventional matrix embedding using maximum likelihood algorithm. In the receiver, the proposed embedding method can extract the embedded logo message more easily than the previous methods proposed in [10], [11], and [12]. The performance difference is due to the use of a parity check matrix. Experimental results confirm that the LIAE algorithm has higher embedding efficiency than those proposed in [10], [11], and [12] at various embedding rates. Moreover, the approaches in [10], [11], [12], and [13] are incapable of selecting the location for the intended embedding.

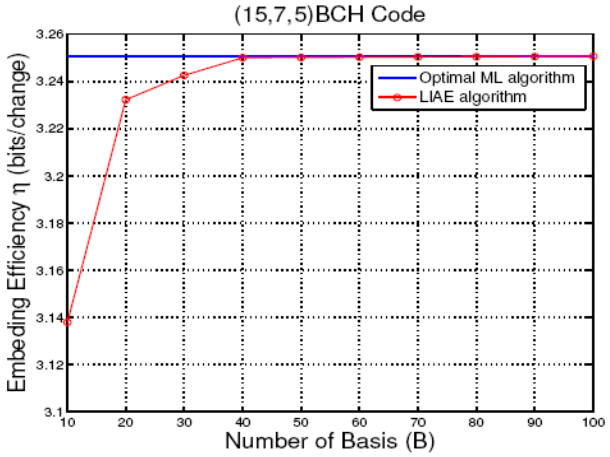

Fig. 7. Embedding effciency versus the number of LI set.

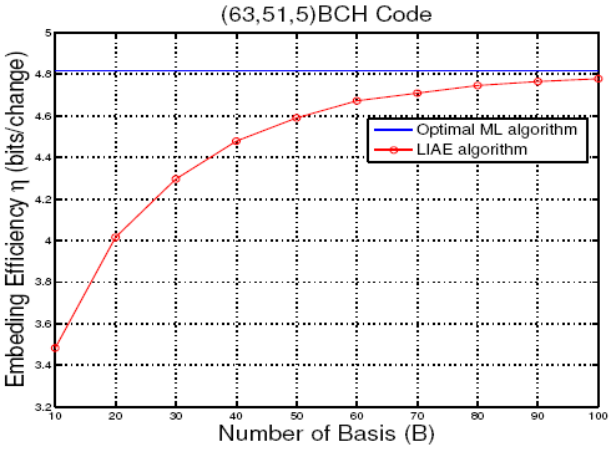

Fig. 8. Embedding effciency versus the number of LI set.

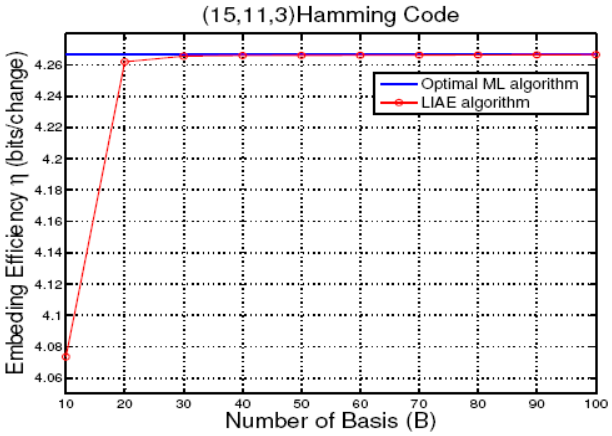

Fig. 9. Embedding effciency versus the number of LI set.

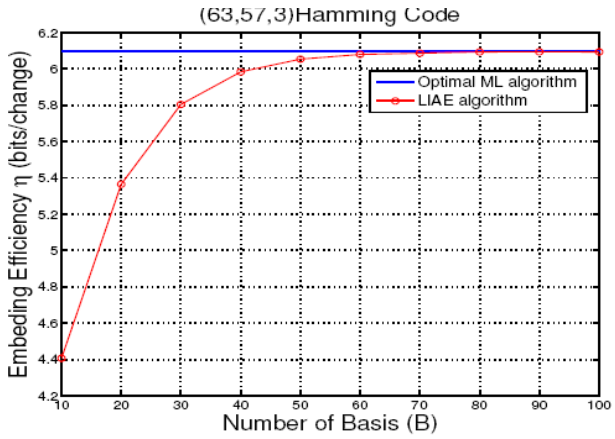

Fig. 10. Embedding effciency versus the number of LI set.

Acknowledgments:
This work was partly supported by the National Science Council of Taiwan, R.O.C. under grant NSC-96-2221-E-167-021 and NSC-100-2221-E-167-024.

References:

[1] G. Schott, Schola Steganographica: *In Classes Octo Distributa (Whipple Collection)*. Cambridge, U.K. Cambridge Univ.

[2] J. Reeds, Solved: The ciphers in book three of Trithemius' steganographic, *Cryptologia*, Nol. 2, No. 4 , Oct. 1998, pp. 291-317.

[3] R. Crandall, Some notes on steganography, Steganography Mailing List, available from *http://os.inf.tu-resden.de/westfeld/crandall.pdf*, 1998.

[4] F. A. P. Petitcolas, R. J. Anderson, and M. G. Kuhn, Information hiding a survey, *Proc. IEEE,* Vol. 87, No. 6, , Jul. 1999, pp. 1062-1078.

[5] P. Moulin and R. Koetter, Data-hiding codes, *Proc. IEEE*, Vol. 93, No. 12, Dec. 2005, pp. 2083-2126.

[6] J. Bierbrauer, *On Crandall's Problem*, unpublished, 1998.

[7] M. Khatirinejad and P. Lisonek, Linear codes for high payload steganography, *Discrete Applied Math.*, vol. 157, no. 5, 2009, pp. 971-981.

[8] G. Cohen, I. Honkala, S. Litsyn, and A. Lobstein, *Covering Codes*. Amsterdam, The Netherlands: North-Holland, 1997.

[9] W. Zhang and S. Li, A coding problem in steganography, Designs, Codes Cryptogr., vol. 46, no. 1, 2008, pp. 68-81.

[10] R. Y. M. Li, O. C. Au, K. K. Lai, C. K. Yuk, and S. Y. Lam, Data hiding with tree based parity check, in *Proc. IEEE Int. Conf. Multimedia and Expo (ICME 07)}*, 2007, pp. 635-638.

[11] R. Y. M. Li, O. C. Au, C. K. M. Yuk, S. K. Yip, and S. Y. Lam, Halftone Image Data Hiding with Block-Overlapping Parity Check, *Proc. IEEE*, Vol. 2, Apr. 2007, pp. 193-196.

[12] Y. C. Tseng, Y. Y. Chen, and H. K. pan, A secure data hiding scheme for binary images, *IEEE Trans. Commun.*, Vol. 50, No. 8, Aug. 2002, pp. 1227-231.

[13] J. Fridrich and D. Soukal, Matrix embedding for large payloads, *IEEE Trans. Inf. Forensics and Security,* Vol. 1, No. 3, 2006, pp. 390-394.

[14] C.-L. Hou, C. Lu, S.-C. Tsai, and W.-G. Tzeng, An Optimal Data Hiding Scheme With Tree-Based Parity Check, *IEEE Trans. Image Process*., Vol. 20, No. 3, March 2011, pp. 880-886.