Steganography Combining Data Decomposition Mechanism and Stego-coding Method

Xinpeng Zhang, Shuozhong Wang and Weiming Zhang
School of Communication and Information Engineering, Shanghai University, China
E-mail: {xzhang,shuowang,weimingzhang}@shu.edu.cn

Keywords: steganography, data decomposition, embedding efficiency

Received: September 1, 2008

A novel steganographic scheme based on data decomposition and stego-coding mechanisms is proposed. In this scheme, a secret message is represented as a sequence of digits in a notational system with a prime base. Each digit block is decomposed into a number of shares. By using stego-coding technique, these shares are then embedded in different cover images respectively. In each cover, a share is carried by a group of cover pixels and, at most, only one pixel in the group is increased or decreased by a small magnitude. That implies a high embedding efficiency, and therefore distortion introduced to the covers is low, leading to enhanced imperceptibility of the secret message. A further advantage of the scheme is that, even a part of stego-images are lost during transmission, the receiver can still extract embedded messages from the surviving covers.

Povzetek: Predstavljena je nova steganografska metoda.

1 Introduction

Steganography is a branch of information hiding that aims to send secret messages under the cover of a carrier signal. While many steganographic methods have been proposed for various types of cover media in recent years, techniques of steganalysis have also rapidly developed to detect the presence of secret messages based on statistical abnormality caused by data hiding [1, 2]. Generally speaking, the more the embedded data, the more vulnerable the system will be to the steganalytic attempts. When a multimedia product is under suspicion, the channel warden may refuse to transmit it, and the source of the message can be tracked. As a countermeasure, the data-hider always tries to improve statistical imperceptibility of the hidden message.

An important technique to improve imperceptibility is to reduce the amount of alterations to be introduced into the cover for hiding the same quantity of data, in other words, to improve embedding efficiency. For example, Matrix encoding uses less than one change of the least significant bit (LSB) in average to embed l bits into $2^l–1$ pixels [3]. In this way, distortion is significantly lowered compared to a plain LSB technique in which secret bits simply replace the LSB. Further, some effective encoding methods derived from the cyclic coding have been described [4], and the matrix encoding can be viewed as a special case. In [5], two methods based on random linear codes and simplex codes are developed for large payloads. Another method, termed running coding, can also be performed on a data stream derived from the host in a dynamically running manner [6]. All the above-mentioned stego-coding techniques are independent of any particular cover-bit-modification approaches. For example, if a stego-coding method is used in the LSB plane of an image, adding 1 to a pixel is equivalent to subtracting 1 from the pixel to flip its LSB for carrying the secret message. In addition, we [7] and Fridrich et al. [8] independently presented a same method with better performance, termed respectively exploiting modification direction (EMD) and grid coloring (GC for short). Using this method, $\log_2(2q+1)$ secret bits are embedded into q cover pixels and, at most, only one pixel is increased or decreased by 1. In [8], a data-hiding approach incorporating GC with Hamming-derived steganographic encoding technique is also studied, which in fact is a special case of GC. We also applied the wet paper codes to steganography to further increase embedding efficiency [9, 10 11].

Since stego-covers may be lost due to an active warden or poor channel conditions, a steganographic system capable of resisting interference is also desired to the data-hider. This paper proposes a novel steganographic scheme by introducing a data decomposition mechanism together with stego-coding techniques, such as running coding and EMD embedding methods. In this way, the secret message is inserted into a number of cover images with high embedding efficiency. Even a part of stego-images are missing, one can still extract the hidden message from the remaining covers.

The rest of this paper is organized as follows. Section 2 introduces the related stego-coding methods. The proposed scheme is described in Section 3 and 4. Then, the experimental results are shown in Section 5. Finally, we conclude in Section 6.

2 Related Stego-coding methods

In stego-coding methods, a number of patterns of cover data are used to represent a type of secret data, and the data-hider modifies the original cover data to the nearest pattern mapping the secret data to be hidden. This way, by changing a small part of cover data, a fairly large amount of secret data can be embedded. In this section, we briefly review the related techniques including running coding and EMD embedding methods.

2.1 Running coding

With running coding method [6], each secret bit is represented by a series of consecutive cover bits, and each available cover bit also relates to several consecutive secret bits. In other words, the secret message is embedded as a data stream, and each cover-bit-alteration is used to embed several consecutive secret bits.

Assume that the secret message to be hidden contains K bits: [x_1, x_2, …, x_K], and the available LSB for carrying the secret message are [$b_{1,1}$, $b_{1,2}$, …, $b_{1,T}$; $b_{2,1}$, $b_{2,2}$, …, $b_{2,T}$; …; $b_{K,1}$, $b_{K,2}$, …, $b_{K,T}$], where T is an integer power of 2 ($T = 2^t$). A binary generating matrix **G** sized $(t+1) \times T$ is first constructed. Denote the elements in **G** as $g(i, j)$, where $1 \le i \le t+1$ and $1 \le j \le T$. Assign all the elements in the first row as '1' and make all the 2^t columns in **G** mutually different. For example, the generating matrix of the 4th running coding is

$$\mathbf{G} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \end{bmatrix} \tag{1}$$

According to the original host data and the generating matrix **G**, calculate

$$y_v = \sum_{i=1}^{t+1} \sum_{j=1}^{T} \left[g(i,j) \cdot b_{v-i+1,j} \right] \bmod 2, \quad v = 1, 2, \ldots, K \tag{2}$$

where $b_{v-i+1,j} = 0$ if $v-i+1 \le 0$. The data-hider can use a small number of alterations in these host bits to make the values of y_vs equal to the secret bits x_vs. Let

$$z_v = \begin{cases} 0, & \text{if } x_v = y_v \\ 1, & \text{if } x_v \ne y_v \end{cases} \tag{3}$$

Orderly arrange all z_vs to form a vector $\mathbf{Z} = [z_1, z_2, \ldots, z_K]^{\mathrm{T}}$, and divide **Z** into a set of sub-vectors in the following way:

1. Scan the vector **Z** from the beginning to the end;
2. If the encountered bit is '0', define this '0' as a sub-vector containing only one element;
3. If the bit is '1', define this '1' together with the following t bits as a sub-vector with a length (t+1). Obviously, the sub-vector in this case must be identical to one of the columns in **G**.

That means **Z** is segmented into a sequence of sub-vectors, each being either a column of the generating matrix **G** or a single zero. According to (2), flipping the value of host bit $b_{v,j}$ will change the value of y_{v+i-1} if $g(i, j) = 1$ ($1 \le i \le t+1$, $1 \le j \le T$). Thus, we can modify only one host bit to change the values of several y_vs. Assume that a sub-vector $[z_v, z_{v+1}, \ldots, z_{v+t}]^{\mathrm{T}}$ is same as the j-th column of **G**. By flipping the value of host bit $b_{v,j}$, the data-hider may make [y'_v, y'_{v+1}, …, y'_{v+t}] identical to [x_v, x_{v+1}, …, x_{v+t}], where [y'_v, y'_{v+1}, …, y'_{v+t}] are obtained from the modified host bits according to (2). This way, the secret data can be embedding using a small number of bit-alterations.

2.2 EMD embedding

EMD embedding [7] is an alternative method for inserting secret data into a certain cover image with a high embedding efficiency. Using this method, each symbol in notational system with an odd base will be carried by a group of pixels, and, at most, only one pixel is increased or decreased by 1.

Denote a secret symbol in notational system with an odd base ($2q$+1) as s, and the gray values of pixels in a group as g_1, g_2, …, g_q. Calculate the extraction function f as a weighted sum modulus ($2q$+1)

$$f(g_1,\ g_2, \ldots, g_q) = \sum_{i=1}^{q} (g_i \cdot i) \bmod (2q+1) \tag{4}$$

Consider the vector [g_1, g_2, …, g_q] as a hyper-cube in q-dimensional space. The extraction function must have the following two properties: 1) values of the extraction function on all hyper-cubes fall in the interval [0 $2q$], and 2) the values of f on any hyper-cube and its $2q$ neighbors are mutually different. This implies that a symbol in the ($2q$+1)-ary notational system can be carried by a pixel-group, and, at most, only one pixel will be increased or decreased by 1. If the symbol s equals the extraction function of the original corresponding pixel-group, no modification is needed. When $s \ne f$, calculate $u = s - f \bmod p$. If u is no more than q, increase the value of g_u by 1, otherwise, decrease the value of g_{p-u} by 1.

For example, considering an original pixel-group [137 139 141 140] with q = 4, f = 3 and a corresponding symbol 4 in 9-ary notational system, a data-hider can calculate u = 1, so he can increase the gray value of the first pixel by 1 to produce the stego-pixels [138 139 141 140]. If the symbol to be hidden is 0, u = 8 can be calculated and the gray value of the forth pixel will be decrease by 1 to yield [137 139 141 139].

3 Data embedding procedure

In this proposed scheme, a secret message is firstly represented as a series of shares according to a data decomposition mechanism and various indices, and the shares corresponding to different indices are respectively inserted into different cover images. Then, a generalized running coding or EMD method is employed to keep stego-induced distortion at a low level, and redundancy in the shares ensures that one can recover the original secret message from a part of stego-covers.

3.1 Data decomposition

At the beginning, a data-hider converts a secret message into a digit sequence in a notational system with an odd and prime base p, such as 3, 5, 7, 11, etc. If the secret message is a binary stream, it can be segmented into many pieces, each having L_1 bits, and the decimal value of each secret piece is represented by L_2 digits in a p-ary notational system, where

$$L_1 = \lfloor L_2 \cdot \log_2 p \rfloor \tag{5}$$

For example, the binary message (1001 1101 0110) can be rewritten as (14 23 11) in 5-ary notational system when $L_1 = 4$ and $L_2 = 2$. Thus, the rate of redundancy in the digit sequence

$$R_\mathrm{R} = 1 - \frac{L_1}{L_2 \cdot \log_2 p} < \frac{1}{L_1 + 1} \tag{6}$$

With large L_1 and L_2, R_R is very close to 0, therefore can be ignored. So, the secret message is regarded as a digit sequence in p-ary notational system in the following discussion.

Then, the data-hider segments the secret digit sequence into a series of blocks, each of which contains m digits. Denote the number of blocks as K, and the block as $\{d_{k,1}, d_{k,2}, \ldots, d_{k,m}\}$ ($k = 1, 2, \ldots, K$). Inspired by [12], decompose each secret block into n shares, $\{s_{k,1}, s_{k,2}, \ldots, s_{k,n}\}$, in the following way,

$$[s_{k,1} \quad s_{k,2} \quad \cdots \quad s_{k,n}] = [d_{k,1} \quad d_{k,2} \quad \cdots \quad d_{k,m}] \cdot \mathbf{A} \tag{7}$$

where $m \le n \le p$,

$$\mathbf{A} = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ a_1 & a_2 & a_3 & \cdots & a_n \\ a_1^{\ 2} & a_2^{\ 2} & a_3^{\ 2} & \cdots & a_n^{\ 2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_1^{\ m-1} & a_2^{\ m-1} & a_3^{\ m-1} & \cdots & a_n^{\ m-1} \end{bmatrix} \tag{8}$$

and the symbol "·" in (7) is a multiplication operator with a modulus p. We call a_1, a_2, …, and a_n as indices. All indices lie between [0 p–1] and are mutually different. For example, assuming p = 5, n = 4, m = 3, and

$$\mathbf{A} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 4 & 0 & 1 \\ 4 & 1 & 0 & 1 \end{bmatrix} \tag{9}$$

a digit block {2, 4, 1} can be represented as 4 shares: 4, 4, 2, and 2. Note that the shares are also within the p-ary notational system.

Collect all shares and divide them into n sets $\{s_{1,1}, s_{2,1}, \ldots, s_{K,1}\}$, $\{s_{1,2}, s_{2,2}, \ldots, s_{K,2}\}$, …, $\{s_{1,n}, s_{2,n}, \ldots, s_{K,n}\}$, each of which contains K shares. Then, the n share-sets and their corresponding indices will be embedded into n cover images, respectively. Since the indices are within [0 p–1], they can also be regarded as symbols in the p-ary notational system. In other words, each cover image will be used to conceal (K+1) symbols in the p-ary notational system, $s_{1,t}$, $s_{2,t}$, …, $s_{K,t}$ and a_t (t = 1, 2, …, n).

3.2 Generalized running coding

In order to improve steganographic imperceptibility, we use stego-coding technique to lower the distortion caused by data embedding. As mentioned above, running coding in [6] is only suitable for binary data-hiding system. This subsection generalizes the running coding method, so that the secret symbols in the p-ary notational system can be carried by a sequence of gray-pixel-value of cover image. Actually, for each cover image, either generalized running coding or EMD embedding can be employed to embed the shares and index.

In the generalized running coding method, each secret symbol in the p-ary notational system is represented by a series of consecutive cover values, and each cover value also relates to several consecutive secret symbols. Thus, a data-hider can modify a selected cover value to embed several secret symbols, so that the distortion introduced into the cover signal is significantly reduced, which also means the data-hiding efficiency is increased.

For convenience, we denote the (K+1) symbols in the p-ary notational system to be embedded into a certain cover as $[x_1, x_2, \ldots, x_{K+1}]$. Pseudo-randomly select $(K+1)\cdot T$ pixels in cover image according to a secret key, and denote the gray-levels of them as $[h_{1,1}, h_{1,2}, \ldots, h_{1,k+1}; h_{2,1}, h_{2,2}, \ldots, h_{2,k+1}; \ldots; h_{T,1}, h_{T,2}, \ldots, h_{T,k+1}]$. That means the number of host values is T times of that of secret symbols.

3.2.1 The case of $T = p^t$

Firstly, we discuss the case that T is an integer power of p ($T = p^t$). Inspired from [6], construct a generating matrix **G** sized $(t+1) \times T$. Denote the elements in **G** as $g(i, j)$, and assign them according to the following principle,

1. All elements are integers within [0, p–1].
2. All elements in the first row are 1.
3. All p^t columns in **G** are different.

For example, when p=3 and k=9,

$$\mathbf{G} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 2 & 0 & 1 & 2 & 0 & 1 & 2 \\ 0 & 0 & 0 & 1 & 1 & 1 & 2 & 2 & 2 \end{bmatrix} \qquad (10)$$

From the original host data and the generating matrix **G**, calculate

$$y_v = \sum_{i=1}^{t+1}\sum_{j=1}^{T}\left[g(i,j)\cdot h_{v-i+1,j}\right] \bmod p, \qquad v = 1, 2, \ldots, K+1 \qquad (11)$$

where $h_{v-i+1,j} = 0$ if $v-i+1 \le 0$. That means the value of y_v is determined by $(t+1) \times T$ host values $h_{v-t,1}$, $h_{v-t,2}$, …, $h_{v-t,T}$, $h_{v-t+1,1}$, $h_{v-t+1,2}$, …, $h_{v-t+1,T}$, …, $h_{v,1}$, $h_{v,2}$, …, $h_{v,T}$. Similarly, we will modify a small number of host values to make each y_v equal to the corresponding secret x_v. Let

$$z_v = x_v - y_v \bmod p, \quad v = 1, 2, \ldots, K+1 \qquad (12)$$

Arrange all the z_v to form a vector $\mathbf{Z} = [z_1, z_2, \ldots, z_{K+1}]^{\mathrm{T}}$, and then divide **Z** into a set of sub-vectors in the following way:

1. Scan the vector **Z** from the beginning to the end;

2. If the encountered digit is ‘0’, define this ‘0’ as a sub-vector containing only one element;

3. If the encountered digit z_v is not ‘0’, define this digit together with the following t digits as a sub-vector with a length $(t+1)$. Because all the elements in the first row of **G** are 1 and p is prime, the sub-vector in this case must be equal to product of z_v and one of the columns in **G** with modulus p.

Equation (11) indicates that any change on host value $h_{v,j}$ will affect the values of y_v, y_{v+1}, …, y_{v+t}. Thus, we can modify only one host value but embed several secret symbols. Assume that z_v is not 0 and the sub-vector $[z_v, z_{v+1}, \ldots, z_{v+t}]^{\mathrm{T}}$ equals the product of z_v and the j-th column of **G** with modulus p. Either increasing the value of $h_{v,j}$ by z_v or decreasing the value of $h_{v,j}$ by $p-z_v$ will make $[y'_v, y'_{v+1}, \ldots, y'_{v+t}]$ identical to $[x_v, x_{v+1}, \ldots, x_{v+t}]$, where $[y'_v, y'_{v+1}, \ldots, y'_{v+t}]$ are obtained from the modified host values according to (11). In this way, all secret symbols can be embedded by performing the similar operation for all sub-vectors.

Consider that, for example, a host value sequence with length 21 for carrying secret message is [40 187 99, 93 231 19, 82 78 33, 11 176 134, 56 27 121, 31 249 83, 90 111 24], and 7 secret digits in ternary system, implying $p = 3$, [2110100]. Because $T = 21/7 = 3^1$, construct a generating matrix **G**.

$$\mathbf{G} = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 2 \end{bmatrix} \qquad (13)$$

From (11), the vector **Y** is [2200021], so that $\mathbf{Z} = [0210112]^{\mathrm{T}}$. Append ‘0’ to the end of **Z** and segment it into 5 sub-vectors: [0], $[21]^{\mathrm{T}}$, [0], $[11]^{\mathrm{T}}$, and $[20]^{\mathrm{T}}$. Note that appending ‘1’ or ‘2’ is also allowable. Since the sub-vector $[21]^{\mathrm{T}}$ is a product of 2 and the 3rd column of **G** with modulus 3, the data-hider should increase $h_{2,3}$ by 2 or decrease $h_{2,3}$ by 1. To lower the distortion, the value of $h_{2,3}$ is decreased by 1. Similarly, $h_{5,2}$ should be increased by 1, and $h_{7,1}$ decreased by 1. So, the stego-sequence [40 187 99, 93 231 18, 82 78 33, 11 176 134, 56 28 121, 31 249 83, 89 111 24] are produced. In this way, 7 symbols in ternary system are embedded by adding/subtracting 1 to/from three pixels. On the receiving side, a simple calculation of (11) can recover the embedded data, when the receiver knows the values of p, T and **G**.

A ratio between the number of embedded bits and the distortion energy caused by data hiding, E, is used to indicate the embedding efficiency. As mentioned above, a sub-vector must be ‘0’, or contains $(t+1)$ elements and begins with a non-zero digit. Since the values of z are also uniformly distributed within $[0, p-1]$, the probability of the former case is $1/p$, while that of the later case is $(p-1)/p$. In the former case, the secret symbol has been represented and any modification is needless, while in the later case, a modification on one host value is made to embed $(t+1)$ digits. In average, $(p\cdot t-t+p)/p$ secret symbols are embedded by modifying $(p-1)/p$ host values. As p is odd, the modifications on host values are within $[(1-p)/2, (p-1)/2]$, thus,

$$E = \frac{\left[\left(p\cdot t - t + p\right)/p\right]\cdot \log_2 p}{\frac{p-1}{p}\cdot\frac{2}{p-1}\sum_{u=1}^{(p-1)/2} u^2} = \frac{\left(p\cdot t - t + p\right)\cdot \log_2 p}{2\cdot\sum_{u=1}^{(p-1)/2} u^2} \qquad (14)$$

which is significantly larger than 2, the embedding efficiency of plain LSB replacement/matching method.

3.2.2 The case of $p^t < T < p^{t+1}$

If T is not an integer power of p, i.e., $p^t < T < p^{t+1}$, a generating matrix **G** sized $(t+2) \times T$ can also be constructed as follows:

1. All elements are integers within $[0, p-1]$.

2. All elements in the first row are 1.

3. All $g(t+2, j)$ are 0 where $1 \le j \le p^t$.

4. The columns in **G** are different.

For instance, when $p=5$ and $T=8$,

$$\mathbf{G} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 2 & 3 & 4 & 0 & 1 & 2 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix} \qquad (15)$$

Similarly, y_s and z_s can be computed from (11) and (12). Vector **Z** can be segmented into sub-vectors in the following way:

1. Scan the vector **Z** from the beginning to the end;

2. If the encountered digit is 0, designate this digit as a sub-vector and denote this type of sub-vector as SV_0;

3. If the encountered digit z_v is not 0, and the vector containing z_v and the following (t+1) digits is equal to a product of z_v and one of the columns in **G** with modulus p, designate the (t+2) digits as a sub-vector, and denote this type of sub-vector as SV_1;

4. If the encountered digit z_v is not 0, and the vector containing z_v and the following (t+1) digits is not the same as the product of z_v and any column in **G** with modulus p, designate z_v and the following t digits as a sub-vector with length (t+1), and denote this type of sub-vector as SV_2. In this case, the next sub-vector must start with a non-zero digit.

Denoting the up-left sub-matrix of **G** sized $(t+1) \times p^t$ as **G'**, an SV_2 sub-vector must be equal to a product of z_v and one of the columns in **G'** with modulus p. For an SV_0 sub-vector, no modification is needed. But for an SV_1 sub-vector equal to a product of its first element z_v and the j-th column in **G** with modulus p or an SV_2 sub-vector equal to a product of its first element z_v and the j-th column in **G'** with modulus p, the data-hider should increase the value of $h_{v,j}$ by z_v or decrease it by $p–z_v$.

For example, consider 80 host values available for carrying secret message and 10 secret symbols in 5-ary notational system [2314023343]. Because T = 80/10 = 8, we can construct a generating matrix **G** as in (14). Assuming the vector **Y** = [2130204131] can be calculated according to (11), thus **Z** = $[0234324212]^T$. Append a ‘0’ to the end of **Z** and segment it into 5 sub-vectors [0], $[23]^T$, $[43]^T$, $[242]^T$, and $[120]^T$. Following the rule of modification as described above, the data-hider should increase $h_{2,5}$ by 2, decrease $h_{4,3}$ by 1, increase $h_{6,8}$ by 2, and increase $h_{9,3}$ by 1 so as to embed the secret data.

Now we calculate the embedding efficiency. As mentioned, a sub-vector following an SV_2 must be SV_1 or SV_2. Therefore, any sequence of sub-vectors between the end of an SV_1 and the end of the next SV_1 must be in the form of {0, 0, …, 0, SV_2, SV_2, …, SV_2, SV_1}. Denote the numbers of consecutive 0s and SV_2 sub-vectors as l_0 and l_2 (l_0, l_2 = 0, 1, 2, …), respectively. In the above example, the pattern of the first 4 sub-vectors is {0, SV_2, SV_2, SV_1} (l_0 = 1, l_2 = 2). Denoting

$$\eta = T / p^{t+1} \tag{16}$$

the probability of a sub-vector sequence with l_0 ‘0’s, l_2 SV_2 sub-vectors, and an SV_1 is

$$P(l_0, l_2) = \frac{p-1}{p^{l_0+1}} \cdot (1-\eta)^{l_2} \cdot \eta \tag{17}$$

For the sub-vector sequence, a total of $[l_0 + l_2 \cdot (t+1) + t + 2]$ secret digits are embedded by modifying (l_2+1) host values. Thus,

$$\begin{aligned} E &= \frac{[(p \cdot t - t + p)/p] \cdot \log_2 p}{\frac{p-1}{p} \cdot \frac{2}{p-1} \sum_{u=1}^{(p-1)/2} u^2} \\ &= \frac{(p \cdot t - t + p) \cdot \log_2 p}{2 \cdot \sum_{u=1}^{(p-1)/2} u^2} \end{aligned} \tag{18}$$

3.3 Application of EMD embedding

When using EMD embedding for concealing (K+1) p-ary symbols, including K shares and an index, into a cover image, pseudo-randomly select $(K+1) \cdot q$ pixels according to a secret key, and divide them into (K+1) pixel-groups, each of which contains q pixels. Here,

$$q = \frac{p-1}{2} \tag{19}$$

Then, we map the (K+1) symbols to the pixel-groups in a one-by-one manner. Using EMD embedding method, each symbol in the p-ary notational system is carried by a group of pixels, and, at most, only one pixel is increased or decreased by 1. As analysed in [7], the embedding efficiency is

$$E = \frac{p \cdot \log_2 p}{p-1} \tag{20}$$

which is also significantly larger than 2, the embedding efficiency of plain LSB replacement/matching method.

Note that both generalized running coding method and EMD embedding method can be used to gain a high embedding efficiency, and the stego-coding techniques used in different covers may be different. So, an additional bit that labels the stego-coding technique used in a certain cover, e.g., ‘0’ for generalized running coding and ‘1’ for EMD embedding, as well as the values of p and K, should be embedded into the cover image itself. If running coding is executed, the parameter T should be also hidden in the corresponding stego-image. Actually, LSB replacement method can be used to embed the additional secret information into cover images, and the embedding positions may be determined by the secret key.

4 Data extracting procedure

As mentioned in the previous section, the secret message is embedded into n cover images, and all the n stego-images are sent through a poor channel. Assume the stego-images may be lost in the channel. If the number of received stego-images is no less than m, one can still recover the original secret message using m arbitrary stego-images.

For each received stego-image, the receiver first extracts the embedded label-bit of stego-coding technique and the values of parameter p, K and T. If generalized running coding is used in the cover, the receiver selects $(K+1)\cdot T$ pixels according to the same secret key, and calculates the K embedded shares, $s_{1,t}$, $s_{2,t}$, …, $s_{K,t}$, and the embedded index a_t (t = 1, 2, …, n) using (11). If EMD method is used, the receiver selects $(K+1)\cdot q$ pixels according to the same secret key, and divides them into (K+1) pixel-groups. Then, he calculates the extraction function of stego-pixel-groups to obtain the (K+1) embedded symbols. This way, the receiver may extract a total of $K\cdot m$ shares and m indices from m received stego-images. Denote the extracted indices as $a_{t_1}, a_{t_2}, \dots, a_{t_m}$. For each digit block, Equation (7) can be reformulated as

$$\begin{bmatrix} s_{k,t_1} & s_{k,t_2} & \dots & s_{k,t_m} \end{bmatrix} = \begin{bmatrix} d_{k,1} & d_{k,2} & \dots & d_{k,m} \end{bmatrix} \cdot \mathbf{A}_t \tag{21}$$

The left side is m shares extracted from different stego-images, and $\mathbf{A}_t$ is an $m\times m$ matrix made up of m columns of $\mathbf{A}$ corresponding to the extracted indices

$$\mathbf{A}_t = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ a_{t_1} & a_{t_2} & a_{t_3} & \cdots & a_{t_m} \\ a_{t_1}^{\ 2} & a_{t_2}^{\ 2} & a_{t_3}^{\ 2} & \cdots & a_{t_k}^{\ 2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{t_1}^{\ m-1} & a_{t_2}^{\ m-1} & a_{t_3}^{\ m-1} & \cdots & a_{t_k}^{\ m-1} \end{bmatrix} \tag{22}$$

As well known, $\mathbf{A}_t$ is a Vandermonde matrix, and its determinant is

$$|\mathbf{A}_t| = \prod_{i>j} \left(a_{t_i} - a_{t_j}\right) \tag{23}$$

Since all a_t are mutually different, $|\mathbf{A}_t|$ can not be zero. That means $\mathbf{A}_t$ must have an inverse with the modulus p,

$$\mathbf{A}_t^{-1} = \frac{\mathbf{A}_t^*}{|\mathbf{A}_t|} \tag{24}$$

where $\mathbf{A}_t^*$ is the adjoint matrix of $\mathbf{A}_t$. So, the secret digit block can be restored by using m extracted shares and the inverse matrix of $\mathbf{A}_t$

$$\begin{bmatrix} d_{k,1} & d_{k,2} & \dots & d_{k,m} \end{bmatrix} = \begin{bmatrix} s_{k,t_1} & s_{k,t_2} & \dots & s_{k,t_m} \end{bmatrix} \cdot \mathbf{A}_t^{-1} \tag{25}$$

For example, for the matrix $\mathbf{A}$ in Equation (9), the indices extracted from three stego-images are 2, 4, and 1, the receiver can obtain

$$\mathbf{A}_t = \begin{bmatrix} 1 & 1 & 1 \\ 2 & 4 & 1 \\ 4 & 1 & 1 \end{bmatrix} \tag{26}$$

and its inverse

$$\mathbf{A}_t^{-1} = \begin{bmatrix} 3 & 0 & 2 \\ 2 & 2 & 1 \\ 1 & 3 & 2 \end{bmatrix} \tag{27}$$

If the three extracted shares are respectively 4, 4, and 2, the digit block is then calculated

$$\begin{bmatrix} 4 & 2 & 2 \end{bmatrix} \cdot \mathbf{A}_t^{-1} = \begin{bmatrix} 2 & 4 & 1 \end{bmatrix} \tag{28}$$

After calculating all the digit blocks, the receiver can concatenate them to retrieve the secret message.

5 Experiment results

In the experiment, a secret message with 3.6×10^5 bits was first converted into 1.3×10^5 digits in 7-ary notational system. After segmenting the secret digit sequence into a series of blocks with length 4, each digit block was decomposed into 6 shares using Equation (7). That means p = 7, n = 6, and m = 4. Then, the 6 share-sets and their corresponding indices were embedded into 6 cover images sized 512×512. In other words, each cover image was used to conceal 3.2×10^4 symbols in 7-ary notational system. Then, we produced 6 stego-images, three of them produced by using generalized running coding with T = 8, and the rest three by using EMD method. Figure 1 shows 4 stego-images among them. In each stego-image produced by generalized running coding, the number of changed pixels was 1.5×10^4 with the modifications within [–3, 3], and the value of PSNR due to data hiding is 53.9 dB, indicating the visual imperceptibility. In each stego-image produced by EMD method, there were 2.8×10^4 pixels increased/decreased by 1, and the value of PSNR is 57.8 dB. Since only a small part of cover pixels were increased or decreased by small magnitudes, it is difficult to detect the presence of secret message. Actually, if one receives no less than four stego-imges among all the six, he can always recover the secret message using the data extracted from the received stego-images.

We also attempted to conceal the same secret message using various steganographic methods, respectively. With a plain LSB embedding method, two cover images with a size of 512×512 were required to provide sufficient LSBs for accommodating the secret data. In this case, the values of PSNR of the stego-images are 52.1 dB. When employing the original running coding and assigning the parameter T = 2, the secret message were carried by three 512×512 cover images with PSNR 52.9 dB. Alternatively, after converting the secret message into a series of 7-ary symbols (q = 3), we exploited EMD embedding method to conceal them into four cover images. Here, PSNR due to data-hiding is 57.8 dB. When the three methods are used, all stego-images are necessary for data extraction at receiver side. Table 1 shows the performance comparison between the three methods and the proposed scheme. Note that, although the proposed scheme exploits more cover images, the steganographic distortion is lower and the secret message can be transmitted through a severe channel.

(a) (b) (c) (d)

Figure 1: Four stego-images. While (a) and (b) are produced by running coding with PSNR 53.9 dB, (c) and (d) are produced by EMD embedding with PSNR 57.8 dB

Table 1. Performance comparison between various steganographic techniques

Steganographic technique	Number of cover images	PSNR due to data embedding	Condition for data extraction
LSB method	2	52.1 dB	All stego-images must be received
Running coding (T = 2)	3	52.9 dB	
EMD embedding (q = 3)	4	57.8 dB	
Proposed scheme (Data decomposition & Stego-coding)	6	53.9 dB and 57.8 dB	Any four stego-imges among all the six are received

6 Conclusion

In the proposed steganographic scheme, a data decomposition mechanism is introduced to represent the secret message as a number of share sets, and both generalized running coding and EMD embedding methods can be employed to embed the shares into different cover images with high efficiency. This way, even though a part of stego-images are lost in a severe channel, one can still recover the hidden message from the received covers.

Two aspects deserve further study in the future. One is error correcting capability of the proposed scheme. In many applications, the receiver may obtain most stego-images with channel noise. Since there is redundancy between the share-sets embedded into different covers, the receiver can still restore the secret message when the noise is not too serious. On the other hand, since it is necessary to distribute the payloads into cover images according to their various sizes, a technique for decomposing the secret message into share-sets with different amounts is desired, i.e., a generalization of the data-decomposing mechanism should be developed.

Acknowledgement

This work was supported by the Natural Science Foundation of China (60872116, 60773079, 60502039), the High-Tech Research and Development Program of China (2007AA01Z477), and the China Postdoctoral Science Foundation Funded Project (20070420096).

References

[1] H. Wang, and S. Wang, “Cyber Warfare: Steganography vs. Steganalysis,” *Communication of ACM*, **47**(10), pp.76-82, 2004.

[2] J. Fridrich, and M. Goljan, “Practical Steganalysis of Digital Images — State of the Art,” in *Security and Watermarking of Multimedia Contents IV, Proceedings of SPIE*, **4675**, San Jose, USA, Jan. 2002, pp.1–13.

[3] A. Westfeld, “F5 — A Steganographic Algorithm,” in *4th International Workshop on Information Hiding, Lecture Notes in Computer Science*, **2137**, Springer-Verlag, 2001, pp. 289–302.

[4] M. Dijk, and F. Willems, “Embedding Information in Grayscale Images,” in *Proc. 22nd Symp. Inform. Theory in the Benelux*, The Netherlands, 2001, pp. 147-154.

[5] J. Fridrich, and D. Soukal, “Matrix Embedding for Large Payloads,” in *Security, Steganography, and Watermarking of Multimedia Contents VIII, Proceeding of SPIE-IS&T*, **6072**, 2006, pp. 60721W1–12.

[6] X. Zhang and S. Wang, “Dynamically Running Coding in Digital Steganography,” *IEEE Signal Processing Letters*, **13**(3), pp. 165–168, 2006.

[7] X. Zhang and S. Wang, “Efficient Steganographic Embedding by Exploiting Modification Direction,” *IEEE Communications Letters*, **10**(11), pp.781-783, 2006.

[8] J. Fridrich, and P. Lisonek, “Grid Colorings in Steganography,” *IEEE Trans. Information Theory*, **53**(4), pp. 1547-1549, 2007.

[9] X. Zhang, W. Zhang and S. Wang, “Efficient Double-Layered Steganographic Embedding,” *Electronics Letters*, **43**(8), pp. 482–483, 2007.

[10] W. Zhang, X. Zhang, and S. Wang, “A Double Layered ‘Plus-Minus One’ Data Embedding Scheme,” *IEEE Signal Processing Letters*, **14**(11), pp. 848–851, 2007.

[11] X. Zhang, W. Zhang, and S. Wang, “Integrated Encoding with High Efficiency for Digital Steganography,” *Electronics Letters*, **43**(22), pp. 1191–1192, 2007.

[12] A. Shamir, “How to Share a Secret,” *Communication of ACM*, **22**(11), pp.612-613, 1979.