Dynamical Running Coding in Digital Steganography

Xinpeng Zhang and Shuozhong Wang

***Abstract*—A novel coding method for digital steganography is described, in which the amount of bit alterations introduced into a cover medium is significantly reduced, leading to less distortion and enhanced security against steganalysis. Unlike other block-based stego-coding approaches, the proposed method works in a running manner. In this way, each secret bit is represented by a series of consecutive cover bits, and flipping of one cover bit can be used to insert several secret bits. Theoretical analysis shows that, by reducing host alterations and inducing less distortion, the running coding has advantages over a previous technique.**

***Index Terms*—Encoding, information hiding, steganography.**

I. Introduction

PERFORMANCE of any steganographic embedding scheme is evaluated primarily by its ability to evade increasingly sophisticated steganalytic techniques, which aim at revealing the presence of hidden information by detecting the statistical abnormality of a stego-signal caused by data embedding [1], [2]. Generally speaking, the more the secret data are embedded, the more vulnerable to steganalytic attempts a steganographic system will be.

There are two approaches to improving steganographic security against statistical analysis. The first is to preserve, or compensate for the induced change in order to restore, the statistical property of the carrier medium [3], [4]. The other is to reduce the amount of alterations necessary to be introduced into the cover signal for data hiding when the number of secret bits is significantly less than that of available host samples. For example, the method proposed in [5] can conceal as many as $\lfloor \log_2(mn+1) \rfloor$ bits of data in a binary image block sized $m \times n$ by changing, at most, two bits in the block. Matrix encoding [6], on the other hand, uses less than one change of the least-significant bit (LSB) in average to embed l bits into $2^l - 1$ pixels. In this way, the introduced distortion is significantly lowered compared to a plain LSB technique in which secret bits are used to simply replace the LSB plane. Further, some effective encoding methods derived from the cyclic coding have been described [7], and the matrix encoding can be viewed as a special case. In all the above-mentioned techniques, data embedding and extraction are performed in a block-by-block manner, i.e., embedding/extraction operations in different cover blocks are mutually independent.

Manuscript received August 17, 2005; revised October 25, 2005. This work was supported in part by the National Natural Science Foundation of China under Grants 60372090 and 60502039, in part by the Key Project of Shanghai Municipality for Basic Research under Grant 04JC14037, and in part by Shanghai Leading Academic Discipline Project T0102. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Maja Bystrom.

The authors are with the School of Communication and Information Engineering, Shanghai University, 200072 Shanghai, China (e-mail: xzhang@staff.shu.edu.cn; shuowang@staff.shu.edu.cn).

Digital Object Identifier 10.1109/LSP.2005.862602

This letter proposes a different method of stego-coding that is performed on a data stream derived from the host in a dynamical running manner. In this way, insertion and extraction of each secret bit are carried out in a series of consecutive cover bits. Using this stego-coding scheme, the number of cover bit alterations for data hiding is significantly reduced, leading to less distortion, therefore enhanced security against steganalysis.

II. Dynamical Running Coding

In steganography, some designated space within the cover signal is generally used to accommodate the additional message to be conveyed in secrecy. For example, the entire LSB plane of a cover image may be replaced with the secret data. In this case, insertion of each bit causes a change of 1/2 LSB on average. We will show that, if the available space in the host signal is greater than that required by the secret bits, by using a dynamically varying organization of the data, less alterations are needed in embedding the same amount of secret bits; hence, less distortion is introduced. In this way, 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 short, the secret message is embedded as a data stream. This mechanism is very different from some previous approaches [5]–[7], in which a fraction of secret message is hidden into a block of host data. So, we call the proposed method *dynamical running coding*. It leads to improved security and, as a penalty, an increased amount of host-data consumption since more space in the host must be preserved, or "consumed," to ensure correct embedding and extraction. In other words, the actual amount of secret data bits N_s is less than the amount of available host data N_h preserved in the embedding process in which a large portion is in fact unaltered. N_h is the number of bytes in the case of LSB embedding. We refer to the ratio $k = N_h/N_s$ as the *transmission rate* and let k be a positive integer that is always greater than 1.

A. Case of $k = 2^t$

Assume that the secret message to be hidden contains L bits: $[x_1, x_2, \ldots, x_L]$, and the available host bits for carrying the secret message are $[b_{1,1}, b_{1,2}, \ldots, b_{1,k}; b_{2,1}, b_{2,2}, \ldots, b_{2,k}; \ldots; b_{L,1}, b_{L,2}, \ldots, b_{L,k}]$, where k is an integer power of 2 ($k = 2^t$).

A binary generating matrix $\mathbf{G}$ sized $(t+1) \times k$ is first constructed. Denote the elements in $\mathbf{G}$ as $g(i,j)$, assign all the elements in the first row as "1," and make all the 2^t columns in $\mathbf{G}$ different. For example

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

From the original host data and the generating matrix $\mathbf{G}$, calculate

$$y_s = \sum_{i=1}^{t+1}\sum_{j=1}^{k}[g(i,j)\cdot b_{s-i+1,j}] \bmod 2, \quad s=1,2,\dots,L \tag{2}$$

where $b_{s-i+1,j}=0$ if $s-i+1\le 0$. Equation (2) indicates that the value of y_s is determined by $(t+1)\times k$ available host bits $b_{s-t,1}, b_{s-t,2},\dots,b_{s-t,k}, b_{s-t+1,1}, b_{s-t+1,2},\dots,b_{s-t+1,k}$, $\dots, b_{s,1}, b_{s,2},\dots,b_{s,k}$. In the following, we will use a small number of alterations in these available host bits to make each y_s equal to the corresponding secret x_s. Let

$$z_s = \begin{cases} 0, & \text{if } x_s = y_s \\ 1, & \text{if } x_s \ne y_s \end{cases}, \quad s=1,2,\dots,L. \tag{3}$$

Arrange all the z_s to form a vector $\mathbf{Z}=[z_1,z_2,\dots,z_L]^{\mathrm{T}}$, and divide $\mathbf{Z}$ into a set of sub-vectors in the following way.

1) Scan the vector $\mathbf{Z}$ 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 $\mathbf{G}$.

In this way, $\mathbf{Z}$ is segmented into a sequence of sub-vectors, with each being either a column of the generating matrix $\mathbf{G}$ or a single zero.

According to (2), flipping the value of available host bit $b_{s,j}$ will change the value of y_{s+i-1} if $g(i,j)=1$ $(1\le i\le t+1, 1\le j\le k)$. For example, with a generating matrix $\mathbf{G}$ as given in (1), flipping of host bit $b_{s,2}$ from 0 to 1 or from 1 to 0 will change the values of y_s and y_{s+2}. Thus, we can modify only one available host bit to change the values of several y_s. Assume that a sub-vector $[z_s, z_{s+1},\dots,z_{s+t}]^{\mathrm{T}}$ is identical to the jth column of $\mathbf{G}$. This means that the vector $[y_s,y_{s+1},\dots,y_{s+t}]$ does not equal $[x_s,x_{s+1},\dots,x_{s+t}]$. Flip the value of available host bit $b_{s,j}$ to make $[y'_s,y'_{s+1},\dots,y'_{s+t}]$ identical to $[x_s,x_{s+1},\dots,x_{s+t}]$, where $[y'_s,y'_{s+1},\dots,y'_{s+t}]$ are obtained from the modified available host bits according to (2). Therefore, secret data can be embedded using the same operation for all sub-vectors that are the columns of $\mathbf{G}$.

Consider, for example, 40 available host bits for carrying secret message [0100, 0011, 1110, 0010, 1011, 1111, 1001, 0111, 0001, 1010] and ten secret bits [0 100 100 110] to be embedded. Because the transmission rate $k=40/10=4$, construct a generating matrix $\mathbf{G}$ as in (1). From (2), the vector $\mathbf{Y}$ made up of ten elements is [1 000 110 001]. Comparing $\mathbf{Y}$ and the secret data, a vector $\mathbf{Z}=[1\,100\,010\,111]^{\mathrm{T}}$ is obtained. Append a "0" to the end of $\mathbf{Z}$, and segment it into five sub-vectors: $[110]^{\mathrm{T}}$, [0], [0], $[101]^{\mathrm{T}}$, and $[110]^{\mathrm{T}}$. Note that appending a "1" is also allowable. Using the rule of modification described in the above, the values of $b_{1,3}$, $b_{6,2}$, and $b_{9,3}$ should be flipped, leading to the stego-data [0110, 0011, 1110, 0010, 1011, 1011, 1001, 0111, 0011, 1010]. In this way, 10 bits of secret data are embedded with alterations of only 3 bits. On the receiving side, a simple calculation of (2) can recover the embedded data.

The ratio between the numbers of flipped cover bits and of the embedded bits R, in other words, the average number of cover bit alterations required for embedding a secret bit, is used to indicate the performance of the proposed dynamical running coding method. A lower R means less distortion caused by data embedding and, therefore, better security of the hidden message against statistical detection. Let us find the expectation of the alteration rate. Since the data to be hidden $[x_1,x_2,\dots,x_L]$ are usually encrypted before embedding, they can be viewed as a random bit stream so that the values of z_s are also equally distributed between 0 and 1. Because a "1" always begins a sub-vector of a length $(t+1)$ that is a column of $\mathbf{G}$, this type of sub-vector occupies $(t+1)/(t+2)$ of $\mathbf{Z}$. For each sub-vector with length $(t+1)$, only one bit-alteration in the host data is made; therefore

$$\overline{R} = \frac{1}{t+2}. \tag{4}$$

B. Case of $2^t < k < 2^{t+1}$

If k is not an integer power of 2, i.e., $2^t < k < 2^{t+1}$, a binary generating matrix $\mathbf{G}$ can also be constructed as follows.

1) Its size is $(t+1)\times k$.
2) All elements in the first row are "1."
3) All $g(t+1,j)$ are "0" where $1\le j\le 2^t$, and all $g(t+1,j)$ are "1" where $2^t+1\le j\le k$.
4) The columns in $\mathbf{G}$ are mutually different.

For instance

$$\mathbf{G} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{5}$$

Similarly, y_s and z_s can be computed from (2) and (3). Segment the vector $\mathbf{Z}$ into sub-vectors in the following way.

1) Scan the vector $\mathbf{Z}$ from the beginning to the end.
2) If the encountered bit is "0," designate this "0" as a sub-vector, and denote this type of sub-vector as SV_0.
3) If the encountered bit is "1" and the vector containing this "1" and the following t bits is identical to a column in $\mathbf{G}$, designate the $(t+1)$ bits as a sub-vector, and denote this type of sub-vector as SV_1.
4) If the encountered bit is "1" and the vector containing this "1" and the following t bits is not the same as any column in $\mathbf{G}$, designate this "1" and the following $(t-1)$ bits as a sub-vector, and denote this type of sub-vector as SV_2. In this case, the next sub-vector must start with a "1."

Thus, a sub-vector contains only an element "0" or is identical to a column of $\mathbf{G}$ or the first t bits in a column of $\mathbf{G}$. For an SV_0 sub-vector, no bit-alteration in host data is needed. For an SV_1 or SV_2 sub-vector, on the other hand, a flip of host bit $b_{s,j}$ is made to embed the secret data.

For example, consider 60 host bits available for carrying secret message [01010, 01011, 11110, 10010, 10111, 11111, 10010, 00111, 10001, 10100, 11100, 01000] and 12 secret bits to be embedded [000 001 111 100]. Because the transmission rate $k=60/12=5$, construct a generating matrix $\mathbf{G}$ as in (5).

According to (2), the $\mathbf{Y}$ vector is [001 010 110 111]. Comparing $\mathbf{Y}$ and the secret data, the vector $\mathbf{Z} = [001\,011\,001\,011]^{\mathrm{T}}$ is obtained. Append a "0" to the end of $\mathbf{Z}$, and segment it into five sub-vectors [0], [0], $[101]^{\mathrm{T}}$, $[1001]^{\mathrm{T}}$, and $[110]^{\mathrm{T}}$. Here, appending a "1" is also allowable. Using the rule of modification as described above, the values of $b_{3,2}$, $b_{6,5}$, and $b_{10,3}$ should be flipped. So, the stego-data will be [01010, 01011, 10110, 10010, 10111, 11110, 10010, 00111, 10001, 10000, 11100, 01 000]. In other words, 12 secret bits are embedded with alterations of only three bits.

Now we calculate the expectation of the alteration rate. 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 $\{0, 0, \ldots, 0, \mathrm{SV}_2, \mathrm{SV}_2, \ldots, \mathrm{SV}_2, \mathrm{SV}_1\}$. Denote the numbers of consecutive 0s and SV_2 sub-vectors as p and q $(p, q = 0, 1, 2, \ldots)$, respectively. In the above example, the pattern of the first four sub-vectors is $\{0, 0, \mathrm{SV}_2, \mathrm{SV}_1\}$ ($p = 2$, $q = 1$). Denoting

$$\eta = \frac{k}{2^{t+1}} \tag{6}$$

the probability of a sub-vector sequence with p "0"s, q SV_2 sub-vectors, and an SV_1 is

$$P(p,q) = \left(\frac{1}{2}\right)^{p+1} (1-\eta)^q \eta. \tag{7}$$

For the sub-vector sequence, a total of $(q+1)$ bit-alterations are required for embedding $(p + q \cdot t + t + 1)$ secret bits. Thus

$$\overline{R} = \frac{\sum_{p=0}^{\infty}\sum_{q=0}^{\infty}\left[\left(\frac{1}{2}\right)^{p+1}(1-\eta)^q\eta\cdot(q+1)\right]}{\sum_{p=0}^{\infty}\sum_{q=0}^{\infty}\left[\left(\frac{1}{2}\right)^{p+1}(1-\eta)^q\eta\cdot(p+q\cdot t+t+1)\right]} = \frac{1}{2\eta+t}. \tag{8}$$

C. Performance Comparison

Comparison of performance has been made between the running coding technique and the embedding method described in [5], and the results are shown in Fig. 1. The abscissa represents the transmission rate, and the ordinate is the expectation of the alteration rate. At any given transmission rate, the running coding has a smaller alteration rate, therefore inducing less distortion, showing advantages of the proposed technique over the previous technique.

III. Discussion

The transmission rate k has been assumed to be a positive integer in the above. If it is not an integer, one can perform two running coding processes with different integer transmission rates to accomplish the embedding. For instance, if the required transmission rate is 3.4, one can carry out running coding with $k = 3$ to embed 60% of the secret data and another coding with $k = 4$ for the rest of the secret data. Thus, the total transmission rate is 3.4 and the total alteration rate $R_{\mathrm{T}} = 0.6\, R_3 + 0.4\, R_4$,

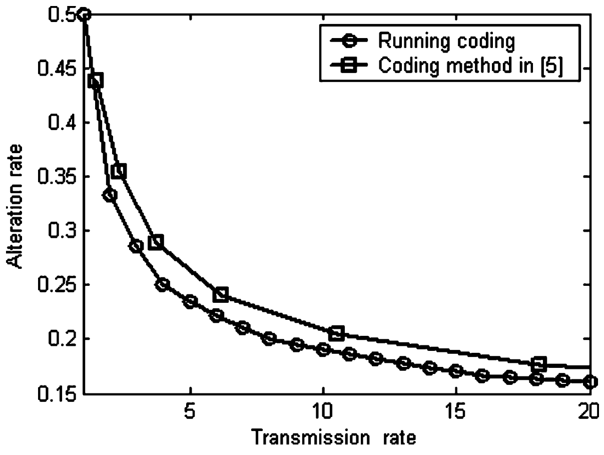

Fig. 1. Performance comparison between the proposed method and coding method in [5].

where R_3 and R_4 are the alteration rates of running coding with $k = 3$ and 4, respectively. In fact, performance of the running coding with noninteger transmission rates is given by the solid curve in Fig. 1.

In the proposed running coding method, each cover-bit-alteration is used to embed several consecutive secret bits. This implies that channel noise is fatal for secret data extraction since any cover-bit-error will cause several bit-errors in the extracted data. Therefore, applications of the running coding scheme should be limited to noise-free environments only. Hidden communication over the Internet can generally meet the requirement, unless an active warden is present.

The running coding technique effectively enhances security by reducing the amount of host data alterations and is independent of the selection of available host data. In other words, the running coding can be used in conjunction with various data embedding approaches, and both the application of running coding and the selection of sophisticated embedding approach contribute to the steganographic security. For example, when the running coding is combined with the simple LSB replacement method, a number of powerful steganalytic tools are still capable of detecting the slight abnormality caused by data hiding. If, on the other hand, an LSB matching technique that changes LSB by randomly adding 1 to or subtracting 1 from the gray value of a cover pixel, the steganalyst can no longer detect the presence of hidden message, unless the percentage of modified pixels is very high, which results in a detectable abnormality in the histogram of the cover image [8]. When the running coding method is combined with an LSB matching technique with a low bit-alteration rate, steganalysis becomes very difficult. Furthermore, in order to avoid potential risks, the data-hider can select a more secure LSB steganographic technique such as that described in [9] in combination with the running coding method.

References

[1] H. Wang and S. Wang, "Cyber warfare: Steganography vs. steganalysis," *Commun. ACM*, vol. 47, no. 10, pp. 76–82, 2004.
[2] J. Fridrich and M. Goljan, "Practical steganalysis of digital images—state of the art," in *Proc. SPIE Security Watermarking Multimedia Contents IV*, , vol. 4675, San Jose, CA, Jan. 2002, pp. 1–13.
[3] P. Sallee, "Model-based steganography," in *Lecture Notes in Computer Science*. New York: Springer-Verlag, 2004, vol. 2939, pp. 154–167.

[4] X. Zhang and S. Wang, "Vulnerability of pixel-value differencing steganography to histogram analysis and modification for enhanced security," *Pattern Recognit. Lett.*, vol. 25, pp. 331–339, 2004.

[5] 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, pp. 1227–1231, Aug. 2002.

[6] A. Westfeld, "F5—a steganographic algorithm," in *Proc. 4th Int. Workshop nformation Hiding, Lecture Notes in Computer Science*, vol. 2137, 2001, pp. 289–302.

[7] M. Dijk and F. Willems, "Embedding information in grayscale images," in *Proc. 22nd Symp. Information Theory Benelux*, Enschede, The Netherlands, 2001, pp. 147–154.

[8] A. D. Ker, "Steganalysis of LSB matching in grayscale images," *IEEE Signal Processing Lett.*, vol. 12, no. 6, pp. 441–444, Jun. 2005.

[9] X. Zhang, S. Wang, and K. Zhang, "Steganography with least histogram abnormality," in *Computer Network Security, Lecture Notes in Computer Science*. New York: Springer-Verlag, 2003, vol. 2776, pp. 395–406.