Advance Access publication on 17 April 2013
doi:10.1093/comjnl/bxt035

Adaptive ±1 Steganography in Extended Noisy Region

TAO HAN[1], WEIMING ZHANG[2,*], CHAO WANG[1], NENGHAI YU[2] AND YUEFEI ZHU[1]

[1]*Zhengzhou Information Science and Technology Institute, Zhengzhou 450002, China*
[2]*The School of Information Science and Technology, University of Science and Technology of China, Hefei 230026, China*
[*]*Corresponding author: weimingzhang@yahoo.cn*

A novel adaptive steganographic scheme for spatial image is proposed. A noisy function is used to measure texture complexity of 2 × 2 pixel blocks, which keeps monotonic increasing after ±1 modifications. Therefore, the message is embedded into the noisiest areas and the recipient can identify the embedding region. The 'double-layered embedding' is exploited to reduce the number of ±1 modifications, in which the fast matrix embedding and wet paper codes are applied to the least significant bit (LSB) plane and the second LSB plane, respectively. The experiments on resisting three steganalyzers show that the proposed method performs better than four typical steganographic schemes. Moreover, comparing with the extended highly undetectable steGO having parameter $T = 255$, the novel method achieves the competitive ability of resisting detection and faster embedding speed.

Keywords: steganography; wet paper codes; matrix embedding; ±1 embedding; double-layered embedding

Received 25 October 2012; revised 4 March 2013
Handling editor: Suchi Bhandarkar

1. INTRODUCTION

Adaptive steganographic schemes embed messages into digital media by minimizing a distortion function that is defined according to some characters of the media. For spatial images, texture complexity is proved to be the suitable distortion metric for steganography. In other words, embedding messages by only modifying noisy areas of the image can resist detection.

However, message extraction will be a problem because the noisy measurement on pixels is also changed after message embedding and thus the receiver cannot identify the embedding region. This problem can be solved by some coding methods, such as wet paper codes (WPCs) [1] and syndrome trellis codes (STCs) [2]. By the WPC, the sender can define noisy areas as dry and smooth areas as wet, and then embed messages by only modifying dry positions, and the receiver can extract messages without any knowledge on dry/wet positions, which is the most important contribution of WPCs. The WPC is designed for the model of two-level distortion (wet and dry), while the STC is a more flexible coding method that can minimize various kinds of distortion functions and the receiver does not need to get any knowledge about the distortion. In fact, the STC can also be applied to the wet paper model.

Based on the STC, one can design adaptive steganography by defining elaborate distortion functions according to some steganalytic features, such as the method proposed in [3]. However, because the feature space is incomplete, such kind of adaptive steganography can be accurately detected by extending the feature space [4] or by steganalytic methods based on other kinds of features.

Recent advancements on steganalysis show that, for spatial images, one secure steganographic manner is to embed messages into the noisiest areas by a WPC. However, the embedding process of the WPC is to solve a system of linear equations, which is a probabilistic algorithm, suffering high complexity of both computation and implementation. To avoid WPCs, Lu *et al.* [5] proposed a noisy region embedding (NRE) method, in which a monotonic increasing noisy function is defined to measure the noisy level of 2×2 image blocks, i.e. the noisy level will increase after the block is modified, and thus the receiver can extract messages from the noisiest areas of the stego images.

In NRE [5], one pixel can only carry one message bit at best, that is to say, the maximum embedding rate of NRE is 1. When the least significant bit (LSB) of the pixel does not match the message bit, the LSB is modified by adding 1 to or subtracting

1 from the gray value of the pixel, according to a special change pattern, which needs to confirm that the noisy function does not decrease after data hiding to ensure the correct extraction of the secret message. The '±1 embedding' (or LSB Matching) is the most popular embedding manner for steganography. In fact, by choosing adding/subtracting 1, every pixel can carry not just one bit but $\log_2 3$ bits of information. This merit of 'freely choosing ±1' can be fully exploited by the 'double-layered embedding' (DLE) [6, 7], which embeds messages not only into the LSB plane but also the second LSB plane. DLE embeds extra messages in the second LSB plane by WPCs without introducing new modifications, and thus can reduce the number of modifications for the given payload. However, to make the noisy function monotonic increasing, the NRE method [5] modifies the pixel in a specific direction, and thus DLE cannot be directly applied to NRE.

In fact, for texture-based adaptive ±1 steganography, one direct method is to apply WPCs in both LSB planes, which we call 'Twice-WPC scheme', subjecting to high complexity of computation and implementation.

In this paper, we improve NRE by extending the boundary of noisy region, which makes 'freely choosing ±1' feasible, and thus DLE still can be applied. In details, we use an NRE-like method in the LSB plane and WPCs in the second LSB plane. The main contributions of the proposed method are as follows.

(i) In the LSB plane, by inheriting the merit of NRE, the proposed method embeds messages in the noisy regions without using a WPC, and thus achieves faster embedding speed than the Twice-WPC scheme.
(ii) In the second LSB plane, by inheriting the merit of DLE, the proposed method embeds extra message bits without introducing new modifications, and thus achieves higher security than NRE.

In summary, the proposed method can realize a reasonable trade-off between computational complexity and security, which is valuable when designing secure and fast steganographic schemes. Extensive experiments on resisting detection show that the proposed method achieves better performance not only than the NRE [5] and the Twice-WPC scheme but also than other two adaptive steganographic schemes proposed in [3, 8].

The rest of this paper is organized as follows. In Section 2, we briefly introduce matrix embedding, WPCs and DLE. The proposed method is elaborated in Section 3. Section 4 presents some imperceptibility experiments and comparisons with the prior arts on resisting steganalysis. The paper is concluded with a discussion in Section 5.

2. SOME PRELIMINARIES

2.1. Matrix embedding

Matrix embedding [9] is the most popular method for reducing the number of modifications of steganography, which is based on linear codes. Assume that an $(n-k)\times n$ binary matrix $\mathbf{H}$ is the parity check matrix of an $[n,k]$ linear code, with which we can embed $n-k$ bits of messages $\mathbf{m}^{\mathrm{T}}=(m_1,m_2,\ldots,m_{n-k})$ into n bits of cover $\mathbf{a}^{\mathrm{T}}=(a_1,a_2,\ldots,a_n)$ by the following manner. First, calculate $\mathbf{s}=\mathbf{Ha}\in GF^{n-k}(2)$, where $\mathbf{s}$ is called the syndrome of $\mathbf{a}$. Secondly, compute the difference between $\mathbf{s}$ and $\mathbf{m}$ with exclusive-or operation, i.e. $\mathbf{Ha}\oplus\mathbf{m}$. Third, solve the system of linear equations:

$$\mathbf{Hx}=\mathbf{Ha}\oplus\mathbf{m}, \tag{1}$$

and find a solution vector $\mathbf{x}_{\min}$ with the minimum Hamming weight such that

$$\mathbf{x}_{\min}=\underset{\mathbf{x}\in GF^n(2),\ \mathbf{Hx}=\mathbf{Ha}\oplus\mathbf{m}}{\arg\min}\{w(\mathbf{x})\}. \tag{2}$$

Finally, the stego object $\mathbf{b}$ is obtained by

$$\mathbf{b}=\mathbf{a}\oplus\mathbf{x}_{\min}. \tag{3}$$

The recipient can extract $\mathbf{m}$ from $\mathbf{b}$ by computing

$$\mathbf{Hb}=\mathbf{H}(\mathbf{a}\oplus\mathbf{x}_{\min})=\mathbf{Ha}\oplus\mathbf{Hx}_{\min}=\mathbf{Ha}\oplus\mathbf{Ha}\oplus\mathbf{m}=\mathbf{m}. \tag{4}$$

By random linear codes, the matrix embedding can greatly reduce the modification number. However, if $\mathbf{H}$ is a random matrix, it is hard to search for a solution with the minimum Hamming weight for the system of Equation (1). To solve (1) with feasible computational complexity, Fridrich *et al.* [10] proposed to use a parity check matrix in a form

$$\mathbf{H}=[\mathbf{I}_{n-k},\mathbf{D}]. \tag{5}$$

Herein, $\mathbf{I}_{n-k}$ is an $(n-k)\times(n-k)$ identity matrix, and the k columns of $\mathbf{D}$ are randomly generated. This can achieve embedding rate $(n-k)/n$ with computation load $O(n2^k)$. Therefore, to keep the complexity requirement low, the code dimension k should be small and the embedding rate must be large enough.

Recently, Wang *et al.* [11] improved the embedding speed of the method of Fridrich *et al.* [10] by appending h check columns after the random matrix $\mathbf{D}$, having embedding rate $(n-k)/(n+h)$. For instance, when appending only one check column, i.e. $h=1$, the parity check matrix has the form

$$\mathbf{H}=[\mathbf{I}_{n-k},\mathbf{D},\mathbf{1}], \tag{6}$$

where $\mathbf{1}^{\mathrm{T}}=(1,\ldots,1)$ is the all-one column. In this case, if $\mathbf{x}=(\mathbf{e},\mathbf{d},0)$ is a solution of Equation (1), the vector $\mathbf{x}'=(\bar{\mathbf{e}},\mathbf{d},1)$ is another solution of Equation (1). Herein, $\mathbf{e}$ is the first $n-k$ bits of $\mathbf{x}$, $\mathbf{d}$ is the following k bits and $\bar{\mathbf{e}}$ is obtained by flipping all bits of $\mathbf{e}$, i.e. the complement vector of $\mathbf{e}$. The Hamming weights of the pair of solutions, $\mathbf{x}$ and $\mathbf{x}'$, are equal to $w(\mathbf{e})+w(\mathbf{d})$ and $n-k-w(\mathbf{e})+w(\mathbf{d})+1$, respectively. In other words, we can try a pair of solutions at one time and only need to record the solution with minimal weight. The searching

speed can be further increased by increasing the number of check columns.

In general, we can replace the Hamming weight, w, with other measurement. Assume that we have a value ρ_i to measure the distortion caused by modifying the ith cover element, from which we can define a new weight function w^* such that

$$w^*(\mathbf{x}) = \sum_{i=1}^{n} x_i \rho_i. \tag{7}$$

Obviously, the methods in [10, 11] can also be used to search for minimal weight solutions in the metric of w^*. In the next section, we will define the distortion ρ_i according to texture complexity and apply the fast matrix embedding [11] with the weight function w^*.

2.2. Wet paper codes

WPCs are designed to embed messages in a cover with defective elements that are determined by the sender. Assume $\mathbf{a}^{\mathrm{T}} = (a_1, a_2, \ldots, a_n)$ is an n-length binary cover. First, the sender determines a selection channel with k changeable bits a_j, $j \in J \subset \{1, 2, \ldots, n\}$, $|J| = k$, which is not shared with the recipient. The bits in the selection channel are called dry bits, while the other bits a_j, $j \notin J$, are labeled as defective and called wet bits. The sender will embed a secret message vector $\mathbf{m}^{\mathrm{T}} = (m_1, m_2, \ldots, m_m)$ into $\mathbf{a}$ by only modifying some dry bits.

WPCs can be viewed as a special case of matrix embedding, in which, we want to construct a solution $\mathbf{b}$ by only modifying some dry bits of $\mathbf{a}$, satisfying

$$\mathbf{Hb} = \mathbf{m}. \tag{8}$$

Herein, $\mathbf{H}$ is an $m \times n$ binary pseudo-random matrix and can be generated from a stego key shared by the sender and the recipient. Therefore, the recipient can extract $\mathbf{m}$ from $\mathbf{b}$ by only calculating $\mathbf{Hb}$ without any knowledge about the selection channel J. To synchronize the matrix $\mathbf{H}$, the message length m and the cover length n should be communicated to the recipient.

Fridrich *et al.* [1] proposed a fast algorithm for WPCs based on LT codes, by which the message length m can approach the number of dry cover bits k when the cover length n is long enough. However, the coding method in [1] is a probabilistic algorithm, and to let m approach k, we have to suffer high computational complexity. Therefore, in practice, we propose to set $m = \alpha_w k$, $0 < \alpha_w < 1$, and call α_w the embedding rate of the WPC. In this paper, we set $\alpha_w = 0.9$ for the LT-based WPC [1]. WPCs can be used to reduce the modification number of ±1 steganography via the DLE.

2.3. Double-layered embedding

The DLE for ±1 embedding was proposed by Zhang *et al.* [6, 7], which embeds messages into the LSB layer and the second LSB layer of pixels, respectively. In fact, if the LSB of a pixel g needs to be changed, its LSB can be flipped to $g + 1$ or $g - 1$. By the choice of ±1, we can control the value of the second LSB of g, i.e. $\lfloor g/2 \rfloor \bmod 2$, and thus embed message bits in the LSB and the second LSB with one modification. For instance, assume that the gray value of pixel g is equal to 5, and we want to embed one bit $m_1 = 0$ in its LSB, which can be realized by modifying 5 to 4 or 6. The modification direction is determined by the bit m_2, which is embedded into the second LSB of g. If $m_2 = 0$, we change 5 to 4; and if $m_2 = 1$, we change 5 to 6.

In DLE, first, some bits are embedded into the LSB plane with a coding method, e.g. matrix embedding. Assume that L bits are embedded in the LSB plane with R modifications. No actual modification is done in this step, and the sender only labels the R pixels to be modified. Secondly, the sender uses WPCs to embed messages in the second LSB plane, in which the R bits of the labeled pixels are defined as dry and other bits are defined as wet. Therefore, in the second LSB plane, $\alpha_w R$ bits can be embedded with the WPC. After that, the sender can determine the modification directions of the R labeled pixels and finish the modifications by $+1$ or -1.

In the above process, $\alpha_w R$ extra bits are embedded into the second LSB plane without introducing new modifications. The key point of DLE is that the sender can modify the LSB of a pixel by freely choosing $+1$ or -1.

If we embed messages in the LSB plane with the maximum embedding rate 1, about half of pixels need to be changed on average. In this case, DLE can approach the embedding rate $1+0.5\alpha_w$, which is called the maximum embedding rate of DLE.

Especially, if the embedding rate of WPC $\alpha_w = 0.9$, the maximum embedding rate of the Twice-WPC scheme can approach $\alpha_{\max}^{\mathrm{TW}} = 0.9 + \frac{0.9}{2} \cdot 0.9 = 1.305$. Therefore, if a cover image consists of $M \times N$ pixels, in order to embed messages with embedding rate α, we only need to choose $\alpha MN/\alpha_{\max}^{\mathrm{TW}}$ pixels in the regions of the most complicated texture to carry the payload. According to DLE, the messages are embedded in the LSB plane and second LSB plane of these pixels separately by using the WPC.

3. PROPOSED METHOD

3.1. Motivation

Next we propose an adaptive steganographic scheme for spatial images, which is greatly inspired by NRE [5]. Lu *et al.* [5] proposed to embed messages into the LSBs of the noisiest region of images by ±1 modification. To do that, a noisy function is first defined to measure the texture complexity of 2×2 image blocks and then a noisy level is assigned to pixels of each block, according to which the pixels are rearranged from a small noisy level to large noisy level as shown in Fig. 1. To embed messages with embedding rate α, α portion of pixels with the largest noisy level are selected to carry the payload, which are identified by a noisy threshold T. In [5], the noisy function is monotonic increasing by specific $+1$ or -1 modification. Therefore, the

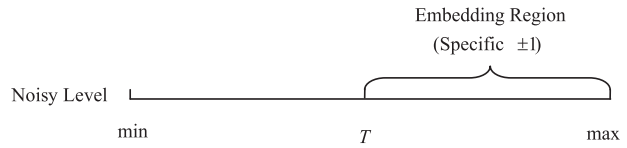

FIGURE 1. Illustration of the NRE method.

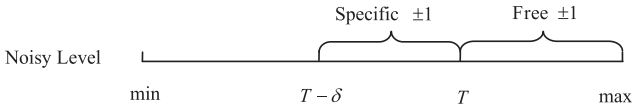

FIGURE 2. Illustration of the extended NRE method.

recipient can extract messages from the LSBs of pixels with noisy level larger than T.

NRE enables the sender to embed messages in the LSB plane of the noisiest areas, but it cannot embed extra messages in the second LSBs as DLE does, because the modification direction of ± 1 is specific to keep the noisy level from descending.

In this section, we will modify NRE to fit DLE. The main idea is shown in Fig. 2. If we embed messages in the pixels with noisy level not $< T$ by free ± 1, the noisy level of modified pixels may decrease. If the boundary is decreased to $T - \delta$, we can embed messages in the pixels with noisy level larger than $T - \delta$ with DLE. First, embed messages in the LSB plane and there are two cases when the LSB of one pixel needs to be changed.

Case 1. If the pixel's noisy level belongs to $[T - \delta, T)$, we change the pixel by specific $+1$ or -1 to keep the noisy level from further descending.

Case 2. If the noisy level of the pixel is not $< T$, we do not change the pixel but only label it.

Second, in the second LSBs of pixels with noisy level larger than or equal to $T - \delta$, we define the bits as dry for the labeled pixels and other bits as wet, and then embed messages with WPCs. After receiving the parameter $T - \delta$, the receiver can extract messages from the LSB plane and the second LSB plane respectively.

To concentrate the modifications on the noisiest region, we demond the decreasing range of noisy level, δ, to be as small as possible after freely ± 1, which is determined by the noisy function. However, the noisy function in [5] is defined by squared sum whose values change acutely, and thus we should first modify the noisy function.

3.2. Modified noisy function

We modify the noisy function in [5] by using the sum of absolute values. First, the cover image is divided into non-overlapped 2×2 blocks and four pixels in each block are ordered (e.g. in clockwise order). Secondly, a noisy level is assigned to each block. For example, on block $\mathbf{B}$, composed of four pixels a, b, c and d, the noisy level is defined by

$$f(\mathbf{B}) = \tfrac{1}{2}[|a-b| + |b-c| + |c-d| + |d-a|], \quad (9)$$

where $a, b, c, d \in \{0, 1, 2, \ldots, 255\}$. Furthermore, we assign the same distortion value for the four pixels of the block, that is,

$$\rho = \frac{1}{f(\mathbf{B}) + \mathbf{1}}. \quad (10)$$

If a cover block $\mathbf{B}_c$ is modified to $\mathbf{B}_s$ by ± 1, we conclude that the change range of noisy level is limited up to 4, i.e. $|f(\mathbf{B}_s) - f(\mathbf{B}_c)| \leq 4$.

Next, we prove that we can modify the LSBs of the four pixels a, b, c and d by specially $+1$ or -1 and keep the noisy level to be monotonic increasing, i.e. $f(\mathbf{B}_s) \geq f(\mathbf{B}_c)$.

Define that $\Delta f \hat{=} f(\mathbf{B}_s) - f(\mathbf{B}_c)$. We complete the proof through the following discussions in Case 1–4. Without loss of generosity, we assume that $255 \geq a \geq b \geq c \geq d \geq 0$ and then obtain $f(\mathbf{B}_c) = a - d$.

Case 1: If only one pixel needs to be changed, without loss of generosity, assume that it is the pixel a and the corresponding modification is denoted by Δa with $\Delta a = \pm 1$ (Δb, Δc and Δd are defined in a similar manner). Taking $\Delta a = +1$, we get $f(\mathbf{B}_s) = a - d + 1$ and $\Delta f = 1 > 0$.

Case 2: If two pixels need to be changed, there are two different cases. Case (2.1): two diagonal pixels, e.g., a and c, need to be modified; Case (2.2): two adjacent pixels, e.g. a and b, need to be modified.

In the Case (2.1), by taking $\Delta a = +1$ and $\Delta c = +1$, we can obtain that

$$f(\mathbf{B}_s) = \begin{cases} a - d + 1 & \texttt{if } b > c, \\ a - d + 2 & \texttt{if } b = c, \end{cases}$$

and thus $\Delta f > 0$ in both cases.

In the Case (2.2), by taking $\Delta a = +1$ and $\Delta b = +1$, we get $f(\mathbf{B}_s) = a - d + 1$ and $\Delta f = 1 > 0$.

Case 3: If three pixels need to be changed, e.g. a, b and c, we can get $f(\mathbf{B}_s) = a - d + 1$ and $\Delta f = 1 > 0$ by taking $\Delta a = \Delta b = \Delta c = +1$.

Case 4: If all the four pixels need to be changed, we can easily obtain that $f(\mathbf{B}_s) = a - d$ and $\Delta f = 0$ by taking $\Delta a = \Delta b = \Delta c = \Delta d = +1$.

In conclusion, there always exists at least one change pattern to satisfy $\Delta f \geq 0$.

3.3. Procedure of data embedding

The following method will embed messages into the LSBs and second LSBs of pixels in the blocks with large noisy levels.

Step 1*: Calculating noisy level*. Assume that a cover image $\mathbf{I}_c$ consists of $M \times N$ pixels. For simplicity, we assume that both $M/2$ and $N/2$ are integers. Divide $\mathbf{I}_c$ into non-overlapped 2×2 blocks and compute the noisy level of each block with (9). Permute these blocks in a pseudo-random order by the stego key shared by the sender and the recipient. Scan the permuted blocks from left to right and from top to bottom to get a block sequence, define by $\mathbf{C} = (\mathbf{B}_1, \ldots, \mathbf{B}_{M \times N/4})$. Divide $\mathbf{C}$ into two disjoint

segments $\mathbf{C}_H = (\mathbf{B}_1, \ldots, \mathbf{B}_d)$ and $\mathbf{C}_P = (\mathbf{B}_{d+1}, \ldots, \mathbf{B}_{M\times N/4})$. Herein, $\mathbf{C}_H$ is used to carry the overhead that consists of some parameters needed by the recipient, and the payload (message) is embedded into $\mathbf{C}_P$.

Step 2*: Estimating the threshold of noisy level*. In $\mathbf{C}_P$, we use $G_T(\mathbf{C}_P)$ to denote the set of blocks whose noisy level is larger than or equal to T, i.e. $G_T(\mathbf{C}_P) = \{\mathbf{B}_i \mid \mathbf{B}_i \in \mathbf{C}_P \text{ and } f(\mathbf{B}_i) \geq T\}$, and $|G_T(\mathbf{C}_P)|$ denotes the number of blocks in $G_T(\mathbf{C}_P)$. Obviously, if $i \leq j$, then $G_i(\mathbf{C}_P) \supseteq G_j(\mathbf{C}_P)$. For simplicity, we also say that a pixel x belongs to $G_T(\mathbf{C}_P)$, meaning that there exists a block $\mathbf{B}_i$ such that $\mathbf{B}_i \in G_T(\mathbf{C}_P)$ and $x \in \mathbf{B}_i$.

Next, assume that the needed embedding rate is α, that is, $L = \alpha MN$ bits of messages need to be embedded. To embed the messages in the noisiest region, we should determine a proper threshold T and the block set $G_T(\mathbf{C}_P)$ to carry the messages. Since the maximum embedding rate of DLE is $1 + 0.5\alpha_w$, we should find the threshold T satisfying

$$|G_{T+1}(\mathbf{C}_P)| \leq \frac{\alpha MN}{4(1+0.5\alpha_w)} < |G_T(\mathbf{C}_P)|. \tag{11}$$

As pointed out in Section 3.1, to synchronize the embedding region and use DLE at the same time, we should loosen the threshold T by 4 levels that is equal to the fluctuating range of the noisy function f. In other words, we should embed messages in $G_{T-4}(\mathbf{C}_P)$. However, experimental results show that the set $G_{T-4}(\mathbf{C}_P)\backslash G_T(\mathbf{C}_P)$ may include too many blocks and thus modifications may be extended to some smooth areas. Therefore, we use a dynamic loosening range δ and embed messages in the blocks belonging to $G_{T-\delta}(\mathbf{C}_P)$, where δ is a non-negative integer and is usually <4. We embed messages in $G_{T-\delta}(\mathbf{C}_P)$ with DLE.

Step 3*: Embedding in the LSB plane*. In the first layer, we embed messages in the LSBs of pixels belonging to $G_{T-\delta}(\mathbf{C}_P)$ with the fast matrix embedding proposed in [11]. When using the matrix embedding, we adopt the weight function (7) with distortion ρ_i determined by Equation (10).

As shown in Fig. 3, if the LSBs need to be modified, there are two modification manners in $G_{T-\delta}(\mathbf{C}_P)$.

Case 1. For the pixels belonging to $G_{T-\delta+4}(\mathbf{C}_P)$, free ±1 is allowed, which is used to control the LSBs as well as the second LSBs of pixels. In fact, in this layer, we only label the pixels in $G_{T-\delta+4}(\mathbf{C}_P)$ that need to be modified and the modifications will be finished in the second layer.

Case 2. For the pixels in $G_{T-\delta}(\mathbf{C}_P)\backslash G_{T-\delta+4}(\mathbf{C}_P)$, the specific modification manner described in Section 3.2 is used to modify the LSBs of pixels, keeping the noisy level monotonic increasing.

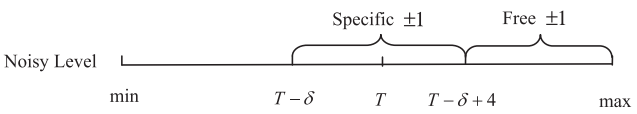

FIGURE 3. Illustration of the proposed method.

To use the fast matrix embedding [11], we set the number of random columns $k = 6$, and the number of check columns $h = 4$. Varying the code length $n_i = i + 45$ for $1 \leq i \leq 999$, we get 999 matrices with different embedding rates β_i and change rates c_i. The embedding rate is calculated by $\beta_i = (n_i - k)/(n_i + h)$, $1 \leq i \leq 999$. The empirical value of change rate c_i can be obtained by embedding random messages into random covers.

The parameters k and h are fixed and shared with the recipient. Both β_i and c_i increase with increasing n_i, and the index i can be used to identify the code. Note that $0.8 \leq \beta_i < 1$. The embedding rate 1 can be realized by LSB replacement, which is denoted by the index 1000.

By the matrix with (β_i, c_i), we can embed $\beta_i \cdot 4 \cdot |G_{T-\delta}(\mathbf{C}_P)|$ bits in the LSB plane, and there are $c_i \cdot 4 \cdot |G_{T-\delta+4}(\mathbf{C}_P)|$ pixels in $G_{T-\delta+4}(\mathbf{C}_P)$ that need to be modified on average. Note that only the second LSBs of modified pixels in $G_{T-\delta+4}(\mathbf{C}_P)$ will be defined as dry, which means that $\alpha_w \cdot c_i \cdot 4 \cdot |G_{T-\delta+4}(\mathbf{C}_P)|$ bits of messages will be embedded in the second layer on average. Therefore, to embed $L = \alpha MN$ bits of secret messages in the two layers, we should select (β_i, c_i) satisfying

$$4 \cdot (\beta_i \cdot |G_{T-\delta}(\mathbf{C}_P)| + \alpha_w \cdot c_i \cdot |G_{T-\delta+4}(\mathbf{C}_P)|) \geq \alpha MN, \tag{12}$$

where α_w is the embedding rate of WPCs and we usually set $\alpha_w = 0.9$. We will select the matrix with the smallest (β_{t_0}, c_{t_0}) satisfying (12) to embed the messages. In other words, we construct a matrix by the method of Wang *et al.* [11] by setting $n = n_{t_0}, k = 6, h = 4$, and the k random columns are generated from the stego key. Therefore, the recipient can generate the same matrix after receiving the index t_0.

After the embedding in the LSB plane, about $c_i \cdot 4 \cdot |G_{T-\delta+4}(\mathbf{C}_P)|$ pixels are labeled. Denote the length of messages embedded in the LSB plane by L_1, and there are $L_2 = L - L_1$ bits left for the second layer.

Step 4*: Embedding in the second LSB plane*. In the second layer, we embed messages into the second LSBs of pixels belonging to $G_{T-\delta}(\mathbf{C}_P)$ with WPCs. We define the second LSBs of the labeled pixels as dry and the second LSBs of other pixels in $G_{T-\delta}(\mathbf{C}_P)$ as wet, and call the second LSB plane of $G_{T-\delta}(\mathbf{C}_P)$ as a wet paper cover whose length is equal to $4 \cdot |G_{T-\delta}(\mathbf{C}_P)|$. To reduce the computational complexity, we can divide the cover and the remaining messages into several segments and embed messages in segment by segment. Assume that the number of segments is s, and then the message length m_i and the cover length c_i for the ith segment are defined as

$$m_i = \begin{cases} \lceil L_2/s \rceil, & 1 \leq i \leq s-1, \\ L_2 - (s-1)\lceil L_2/s \rceil, & i = s, \end{cases} \tag{13}$$

$$c_i = \begin{cases} \lfloor 4|G_{T-\delta}(\mathbf{C}_P)|/s \rfloor, & 1 \leq i \leq s-1, \\ 4|G_{T-\delta}(\mathbf{C}_P)| - (s-1)\lfloor 4|G_{T-\delta}(\mathbf{C}_P)|/s \rfloor, & i = s. \end{cases} \tag{14}$$

For each segment, generate a random matrix and embed messages with the WPCs [1]. After the wet paper coding, we determine whether the second LSBs of the labeled pixels need to be modified or not, according to which we choose $+1$ or -1 to modify these pixels.

Step 5: Communicating parameters. For the convenience of extraction, some parameters need to be communicated to the recipient, which can be embedded into $\mathbf{C}_H = (\mathbf{B}_1, \ldots, \mathbf{B}_d)$, i.e. the first d blocks of the permuted images. The parameters needed by the recipient include the total length L of the message, the length L_1 of the messages embedded in the first layer, the threshold $T - \delta$ of noisy level, the index t_0 for the fast matrix embedding and the number s of segments for WPCs. We embed these parameters in the LSBs of pixels in $\mathbf{C}_H$ randomly by ± 1. In practice, 40 bits are enough for carrying the coded parameters and thus we set the number of blocks $d = 10$ for the overhead.

Step 6: Generating the stego image. After embedding the messages and parameters, we get the modified block sequence, defined by $\mathbf{C}' = (\mathbf{B}'_1, \ldots, \mathbf{B}'_{M\times N/4})$. Permute the order of the blocks back with the stego key and arrange these blocks into an $M \times N$ image, denoted by $\mathbf{I}_s$. Therefore, $\mathbf{I}_s$ is just the stego image that will be sent to the recipient.

3.4. Procedure of data extraction

Step 1: Calculating noisy level and extracting parameters. After receiving the stego image $\mathbf{I}_s$, the recipient first divides it into non-overlapped 2×2 blocks and computes the noisy level of each block with (9), and then permutes these blocks in a pseudo-random order by the stego key. Secondly, scan the permuted blocks from left to right and from top to bottom to get the block sequence $\mathbf{C}' = (\mathbf{B}'_1, \ldots, \mathbf{B}'_{M\times N/4})$ and divide $\mathbf{C}'$ into two disjoint segments $\mathbf{C}'_H = (\mathbf{B}'_1, \ldots, \mathbf{B}'_d)$ and $\mathbf{C}'_P = (\mathbf{B}'_{d+1}, \ldots, \mathbf{B}'_{M\times N/4})$, where $d = 10$. From $\mathbf{C}'_H$, extract the parameters, including the message length L, the length L_1 of the messages embedded in the first layer, the threshold $T - \delta$ of noisy level, the index t_0 of the fast matrix embedding and the number s of segments for WPCs.

Step 2: Extracting messages from the LSB plane. From $\mathbf{C}'_P$, extract the blocks with noisy level larger than or equal to $T - \delta$, and denote the set of such blocks by $G_{T-\delta}(\mathbf{C}'_P)$. If the index $t_0 < 1000$, generate a matrix $\mathbf{H}$ by the method of Wang *et al.* [11] by setting $n = t_0 + 45$, $k = 6$, $h = 4$, and the k random columns are generated from the stego key. With $\mathbf{H}$, extract messages from the LSBs of pixels in $G_{T-\delta}(\mathbf{C}'_P)$. If $t_0 = 1000$, extract the LSBs of the pixels in $G_{T-\delta}(\mathbf{C}'_P)$ directly.

Step 3: Extracting messages from the second LSB plane. The length of messages embedded in the second layer can be calculated by $L_2 = L - L_1$. Note that $|G_{T-\delta}(\mathbf{C}'_P)| = |G_{T-\delta}(\mathbf{C}_P)|$, and thus, with the parameter s, the recipient can calculate the message length and the cover length of each segment according to (13) and (14). Furthermore, the recipient can generate the random matrix for each segment by the stego key, and extract the message from each segment.

4. EXPERIMENTAL RESULTS

4.1. Imperceptibility experiment

The 1000 images randomly chosen from BossBase1.01 image database [12], which contains 10 000 gray-scale images of fixed size 512×512, are used as a group of cover images in this subsection. When processing the proposed method on each cover image, the secret data are randomly generated by using different seeds.

4.1.1. *Visual imperceptibility*

In the paper, the visual quality of a stego image with $M \times N$ pixels was also evaluated by the PSNR defined as

$$\mathrm{PSNR} = 10 \times \log_{10}\left(\frac{255 \times 255}{\mathrm{MSE}}\right), \tag{15}$$

where MSE denotes the mean square error calculated by

$$\mathrm{MSE} = \frac{\sum_{i=1}^{M}\sum_{j=1}^{N}(x(i,j) - y(i,j))^2}{M \times N}, \tag{16}$$

where $x(i, j)$ and $y(i, j)$ are the values of pixel (i, j) in the cover image and stego image, respectively.

Figure 4 shows the PSNRs of the 1000 stego images when the embedding rate of the proposed method is 1. From the picture, it can be seen that the PSNR of each image is larger than 51 dB, and so we can conclude that the human visual characteristics are preserved perfectly, meeting the requirement of visual imperceptibility.

4.1.2. *Statistical imperceptibility*

We use the relative entropy to measure the difference between distributions of the cover and stego images. The relative entropy,

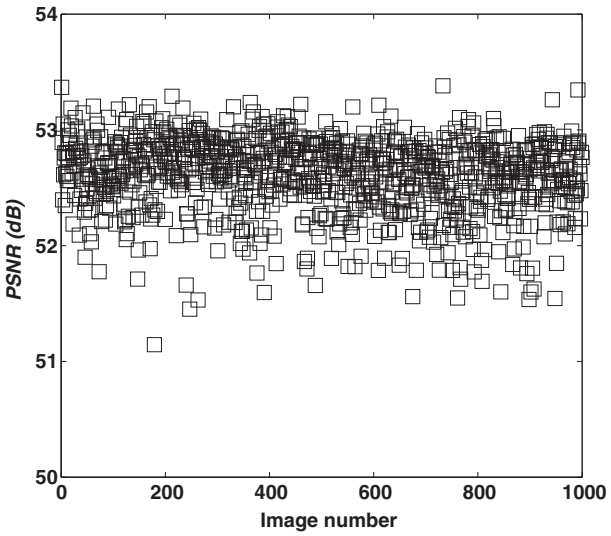

FIGURE 4. PSNRs of the 1000 test images.

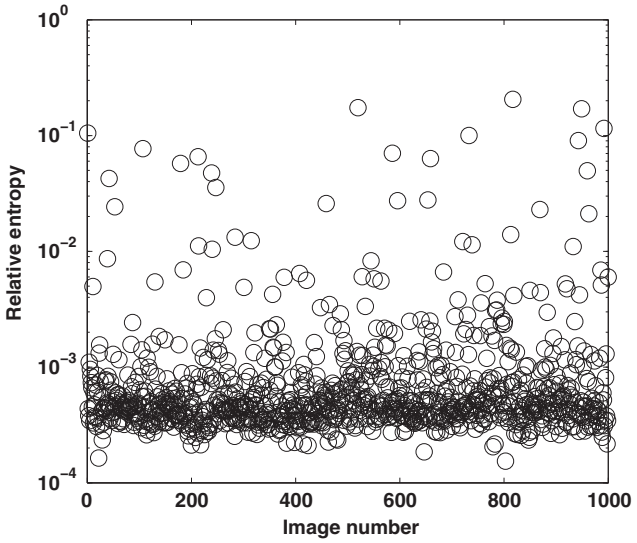

FIGURE 5. Relative entropy of the 1000 test images.

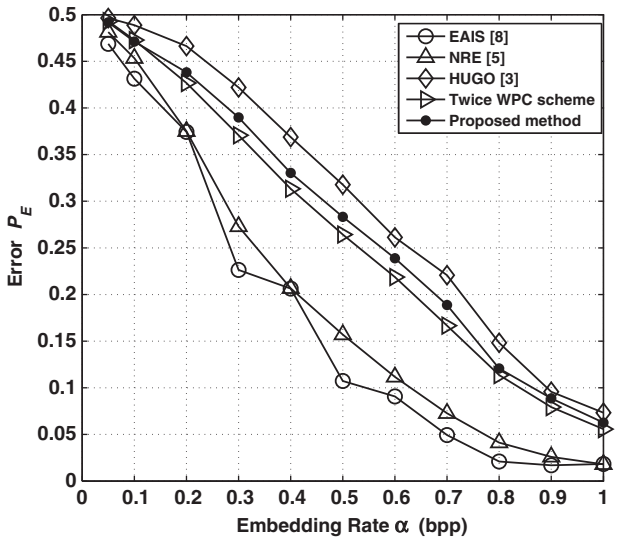

FIGURE 6. The comparisons of five embedding methods resisting the SPAM steganalyzer [13].

also known as Kullback–Leibler divergence, is a measurement of the difference between two probability distributions. The relative entropy between two probability distributions P and Q is defined as

$$D_{\mathrm{KL}}(P \parallel Q) = \sum_i P(i) \log \frac{P(i)}{Q(i)}. \tag{17}$$

We see that $D_{\mathrm{KL}}(P \parallel Q)$ is non-negative and it is equal to 0 if and only if the distributions P and Q are equal. We calculate the relative frequency of the gray value i ($0 \leq i \leq 255$), denoted by $P(i)$ and $Q(i)$ for the cover image and stego image, respectively, and then compute the relative entropy with Equation (19).

Figure 5 shows the relative entropy when the proposed method was processed on the 1000 test images at the embedding rate 1. It can be seen that the relative entropy is very close to zero, which means that the changes of the cover images' histograms, caused by the embedding process, are very small.

4.2. Steganalysis experiment

In this subsection, we compared the proposed method with three adaptive ±1 steganographic algorithms: highly undetectable steGO (HUGO) [3], NRE [5] and EAIS [8]. As mentioned in Section 1, one natural method for steganography is to apply WPCs in both LSB planes, called the Twice-WPC scheme, which is also used for comparison. First, two efficient steganalyzers against ±1 steganography, the second-order SPAM [13] (dim 686) and RHF [14] (dim 70), were used to detect these five steganographic algorithms. The 10 000 grayscale images of fixed size 512 × 512 in the BossBase image database [12] were used for experiments. Among these images, we used 50% of them for training and the remaining 50% for test. Both steganalyzers used the Ensemble Classifier [15] as the classifier. Eleven embedding rates $\alpha = 0.05, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9$ and 1.0 are considered.

We use the minimum average classification error P_E, such that

$$P_E = \min_{P_{\mathrm{FA}}}[(P_{\mathrm{FA}} + P_{\mathrm{MD}}(P_{\mathrm{FA}}))/2], \tag{18}$$

to measure the undetectability of the steganographic schemes, where P_{FA} is the false-alarm probability and P_{MD} is the missed-detection probability. Larger values of P_E correspond to higher undetectability, i.e. stronger security.

As shown in Fig. 6, for the SPAM steganalyzer, the proposed method outperforms NRE, EAIS and the Twice-WPC scheme. HUGO resists the SPAM steganalyzer better than all the other four schemes. As shown in Fig. 7, for the RHF steganalyzer, the proposed method outperforms NRE and EAIS, and has comparable performance with the Twice-WPC scheme. And the proposed method outperforms HUGO when the embedding rate $\alpha \geq 0.5$.

HUGO was designed to preserve a model for the cover source, and the model is represented by a feature vector computed from co-occurrence matrices whose dimension is limited by a threshold T for the differences of neighboring pixel pairs. The default setting for T was $T = 90$. As pointed out in [4], there is a weakness of HUGO caused by the abrupt end of the model. In fact, by taking the difference histogram around the value of 90, one can obtained a detectable artifact. Therefore, we used the combination of the histogram features around 90 (denoted by $\mathbf{h}^{\mathbf{x}}$, dim 4) and the SQUARE features (dim 338) adopted in [4] to detect HUGO. And the Ensemble Classifier [15] was also used for the classifier in this steganalyzer. As

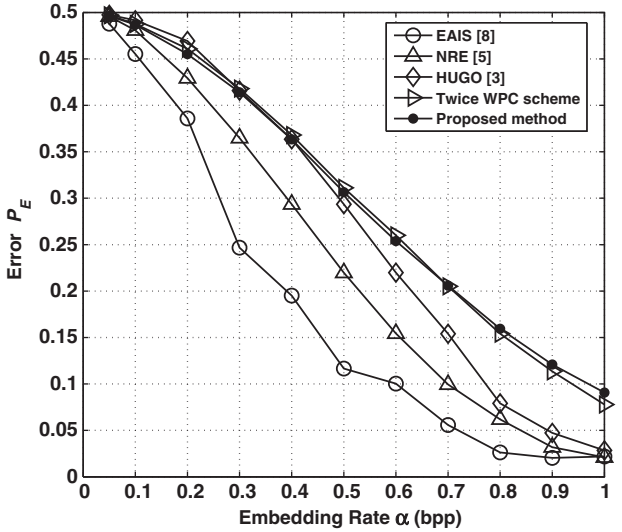

FIGURE 7. The comparisons of five embedding methods resisting the RHF steganalyzer [14].

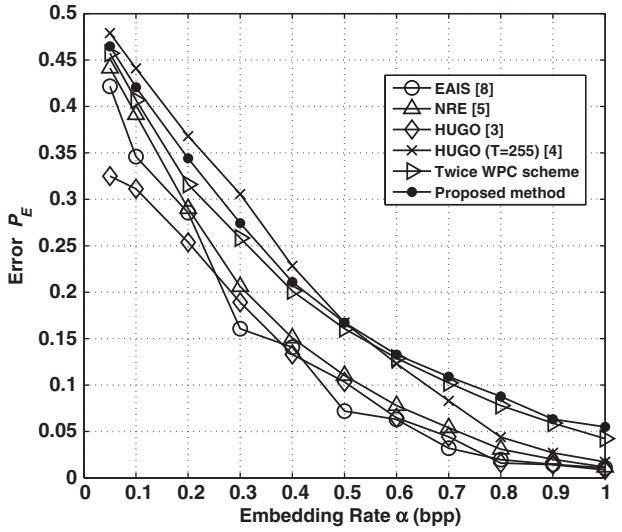

FIGURE 8. The comparisons of the previous five embedding methods and highly undetectable steGO ($T = 255$) [4] resisting the steganalyzer with the combination of the histogram feature (dim 4) and the SQUARE feature (dim 338) used in [4].

shown in Fig. 8, the minimum average classification error P_E for HUGO is greatly reduced. As mentioned in [4], this flaw of HUGO can be prevented by adjusting the parameter as $T = 255$. For the detection of 'SQUARE+$\mathbf{h}^{\mathbf{x}}$' features (dim 342), the proposed method performs better than NRE, EAIS, the Twice-WPC scheme and HUGO. In addition, when $\alpha \geq 0.5$, the proposed method outperforms the extended HUGO ($T = 255$).

One steganographic algorithm will be insecure as long as there exists one effective steganalyzer for it, and so we use the measure $Global\_P_E$, to evaluate the ability of a steganographic method resisting a series of steganalyzers.

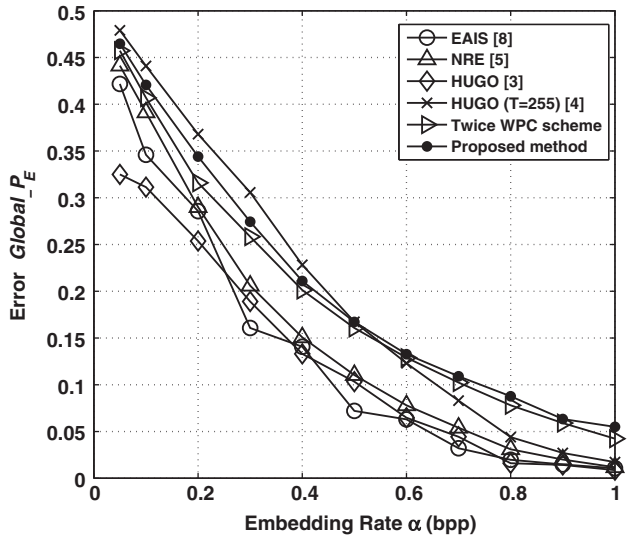

FIGURE 9. The $Global\_P_E$ of six embedding methods resisting the steganalyzers SPAM [13], RHF [14] and 'SQUARE + $\mathbf{h}^{\mathbf{x}}$' features [4].

$$Global\_P_E = \min_{i \in \mathcal{S}}(P_{Ei}), \tag{19}$$

where $\mathcal{S}$ is the index set of steganalyzers and P_{Ei} is the P_E for resisting the ith steganalyzer. Herein, $\mathcal{S} = \{1, 2, 3\}$ and the steganalyzers include SPAM, RHF and 'SQUARE+$\mathbf{h}^{\mathbf{x}}$' features. Figure 9 depicts the $Global\_P_E$ of six steganographic algorithms for resisting three steganalyzers. The proposed method outperforms NRE, EAIS, the Twice-WPC scheme and HUGO. Comparing with the extended HUGO ($T = 255$), the proposed method performs better when $\alpha \geq 0.5$.

4.3. Embedding speed experiment

The embedding speed is also an important criterion in some practical situation, such as the steganographic system based on real-time communications [16]. In the present paper, one purpose of defining monotonic increasing noisy function is to avoid using a WPC and reduce the computational complexity. Therefore, we also compare the embedding speed of the proposed scheme with the Twice-WPC scheme, HUGO [3] and the extended HUGO ($T = 255$) [4]. We embedded messages into the 512×512 grayscale image, Lena, with five embedding rates ($\alpha = 0.2, 0.4, 0.6, 0.8$ and 1.0) by the four methods, respectively. The tests were performed on Intel Core i5 750 CPU running at 2.67 GHz with 3 GB RAM, and the proposed scheme and Twice-WPC scheme were implemented in C++ program and compiled under Microsoft Visual C++ 6.0. The C++ source codes of HUGO and the extended HUGO ($T = 255$) were downloaded from the BOSS Web Page [17]. Although the coding method STC in HUGO can be fast implemented, the distortion function defined in HUGO ($T = 90$) and HUGO

TABLE 1. Embedding time (in seconds) of HUGO [3], HUGO ($T = 255$) [4], the Twice-WPC scheme and the proposed method with the 512 × 512 grayscale image Lena as the cover.

Embedding rates	0.2	0.4	0.6	0.8	1.0
HUGO [3]	5.26	5.47	5.90	6.25	6.80
HUGO ($T = 255$) [4]	5.88	6.27	6.56	7.08	7.66
Twice-WPC scheme	3.57	4.73	7.45	12.52	24.88
Proposed method	1.23	1.47	1.83	2.06	2.26

HUGO, highly undetectable steGO.

($T = 255$) need to compute more than 10^7 and 10^8 features, respectively. That is why HUGO has to be subject to higher computational complexity. As shown in Table 1, the proposed method is about 3.93 times faster than the extended HUGO ($T = 255$) and 6.24 times faster than the Twice-WPC scheme on average. In fact, the proposed method uses a WPC only once and the WPC is limited in a shorter cover, i.e. the noisiest region. That is why the proposed method can greatly increase the embedding speed.

5. CONCLUSIONS

In this paper, we present an adaptive steganographic algorithm for spatial images, which embeds messages by only modifying the noisy region of the image and efficiently exploits the merit of '±1 steganography' by DLE. The experimental results on resisting three steganalyzers show that the proposed method outperforms four typical adaptive steganographic schemes, including HUGO [3], NRE [5], EAIS [8] and the Twice-WPC scheme, and has competitive ability with the extended HUGO ($T = 255$). On the other hand, the novel method inherits the merit of NRE [5], avoiding using WPCs in the LSB plane, and thus greatly increases the embedding speed when comparing with the Twice-WPC scheme and the extended HUGO ($T = 255$).

One potential flaw of the proposed method is that the attacker may improve the detecting accuracy by extracting features only from noisy areas of the image. However, the attacker cannot determine the embedding region without the noisy threshold $T - \delta$. On the other hand, it is hard to extract effective features for detecting changes in noisy areas. If such kinds of methods are possible, they can also be used to detect other adaptive steganographic schemes, such as those in [3, 5, 8], because all these adaptive schemes for spatial images tend to avoid modifications in smooth areas. Therefore, how to improve detection on adaptive steganography by only extracting features from noisy regions is an interesting and important problem for steganalysis.

FUNDING

This work was supported in part by the Natural Science Foundation of China (61170234, 60803155); by the Strategic and Piloted Project of CAS (XDA06030601); and by the Municipal Science and Technology Innovation Team Project of Zhengzhou (10CXTD150).

REFERENCES

[1] Fridrich, J., Goljan, M. and Soukal, D. (2005) Efficient Wet Paper Codes. *Proc. 7th Int. Workshop on Information Hiding*, Barcelona, Spain, June 6–8, Lecture Notes in Computer Science 3727, pp. 204–218. Springer, Berlin.

[2] Filler, T., Judas, J. and Fridrich, J. (2010) Minimizing Embedding Impact in Steganography using Trellis-Coded Quantization. *Proc. SPIE, Electronic Imaging, Media Forensics and Security XII*, San Jose, CA, January 17–21, pp. 0501–0514. SPIE Digital Library, Bellingham, WA.

[3] Pevný, T., Filler, T. and Bas, P. (2010) Using High-Dimensional Image Models to Perform Highly Undetectable Steganography. *Proc. 12th Int. Workshop on Information Hiding*, Calgary, Alberta, Canada, June 28–30, Lecture Notes in Computer Science 6387, pp. 161–177. Springer, Berlin.

[4] Kodovský, J., Fridrich, J. and Holub, V. (2011) On Dangers of Overtraining Steganography to an Incomplete Cover Model. *Proceedings of ACM Multimedia and Security Workshop*, Niagara Falls, New York, USA, September 29–30, pp. 69–76. ACM, New York, NY.

[5] Lu, Y., Li, X. and Yang, B. (2009) A Secure Steganography: Noisy Region Embedding. *Proc. 5th Int. Conf. on Intelligent Information Hiding and Multimedia Signal Processing*, Kyoto, Japan, September 12–14, pp. 1046–1051. IEEE Computer Society, Los Alamitos, California.

[6] Zhang, X., Zhang, W. and Wang, S. (2007) Efficient double-layered steganographic embedding. *Electron. Lett.*, **43**, 482–483.

[7] Zhang, W., Zhang, X. and Wang, S. (2007) A double layered "plus-minus one" data embedding scheme. *IEEE Signal Process. Lett.*, **14**, 848–851.

[8] Luo, W., Huang, F. and Huang, J. (2010) Edge adaptive image steganography based on LSB matching revisited. *IEEE Trans. Inf. Forensics Sec.*, **5**, 201–214.

[9] Crandall, R. Some notes on steganography. http://www.dia.unisa.it/ads/corso-security/www/CORSO-0203/steganografia/LINKS%20LOCALI/matrix-encoding.pdf (22 October 2012).

[10] Fridrich, J. and Soukal, D. (2006) Matrix embedding for large payloads. *IEEE Trans. Inf. Forensics Sec.*, **1**, 390–394.

[11] Wang, C., Zhang, W., Liu, J. and Yu, N. (2012) Fast matrix embedding by matrix extending. *IEEE Trans. Inf. Forensics Sec.*, **7**, 346–350.

[12] Bas, P., Filler, T. and Pevný, T. (2011) Breaking Our Steganographic System: The Ins and Outs of Organizing Boss. *Proc. 13th Int. Workshop on Information Hiding*, Prague, Czech Republic, May 18–20, Lecture Notes in Computer Science 6958, pp. 59–70. Springer, Berlin.

[13] Pevný, T., Bas, P. and Fridrich, J. (2010) Steganalysis by subtractive pixel adjacency matrix. *IEEE Trans. Inf. Forensics Sec.*, **5**, 215–224.

[14] Cai, K., Li, X., Zeng, T., Yang, B. and Lu, X. (2010) Reliable Histogram Features for Detecting LSB Matching. *Proc. IEEE 17th Int. Conf. on Image Processing*, Hong Kong, September

26–29, pp. 1761–1764. IEEE Signal Processing Society, Piscataway, NJ.

[15] Kodovský, J., Fridrich, J. and Holub, V. (2012) Ensemble classifiers for steganalysis of digital media. *IEEE Trans. Inf. Forensics Sec.*, **7**, 432–444.

[16] Shirali-Shahreza, M.H. and Shirali-Shahreza, S. (2010) Real-time and MPEG-1 layer III compression resistant steganography in speech. *IET Inf. Sec.*, **4**, 1–7.

[17] Filler, T., Pevný, T. and Bas, P. BOSS Web Page. http://exile.felk.cvut.cz/boss/BOSSFinal/ (22 October 2012).