International Journal of Security and Its Applications
Vol. 9, No. 6 (2015), pp. 279-298
http://dx.doi.org/10.14257/ijsia.2015.9.6.27

An Adaptive Multi-Layer Block Data-Hiding Algorithm that uses Edge Areas of Gray-Scale Images

Tuan Duc Nguyen[1], Somjit Arch-int[2*] and Ngamnij Arch-int[3]

[1]Student, Dept. of Computer Science, Faculty of Science, Khon Kaen University, Khon Kaen 40000, Thailand
[2,3]Associate Professor, Dept. of Computer Science, Faculty of Science, Khon Kaen University, Khon Kaen 40000, Thailand
[1]nguyenductuan1982@gmail.com, [2]somjit@kku.ac.th, [3]ngamnij@kku.ac.th

Abstract

Embedding data into smooth regions introduces stego-images with poor security and visual quality. Edge adaptive steganography, in which the flat regions are not employed to carry a message at low embedding rates, was proposed. However, for the high embedding rates, smooth regions are contaminated to hide a secret message. In this paper, we present an adaptive multi-layer block data-hiding (MBDH) algorithm, in which the embedding regions are adaptively selected according to the number of the secret message bits and the texture characteristic of a cover-image. Via employing the MBDH algorithm, more secret message bits are embedded into the sharp regions. Therefore, the smooth regions are not used, even at high embedding rates. Furthermore, most of edge adaptive steganography algorithms have a limited capacity when the smooth regions are not employed in data hiding. The proposed scheme solves this issue when it can embed more secret bits into the selected regions while the perceptual quality of stego-images is still maintained. The experimental results were evaluated on 10,000 natural gray-scale images. The visual attack, targeted steganalysis, and universal steganalysis are employed to examine the performance of the proposed scheme. The results show that the new scheme significantly overcomes the previous edge-based approaches and least significant bit (LSB) based methods in term of security and visual quality.

Keywords: *Secure, Multi-layer block data-hiding, Edge region, Gray-scale image, Steganography*

1. Introduction

For the past few years, the amount of sensitive information (*e.g.*, business transaction information) transferred via the Internet has increased significantly over the recent decades due to the fast growth of network techniques. Valuable data can suffer from security breaches that occurs when there is no protection applied to transmitted data on the Internet. We can employ cryptography techniques to encrypt valuable data before sending them to the Internet. Because modern information-hiding techniques are used, double layer security is obtained to protect valuable data against opponents.

In general, information-hiding techniques are categorised into two classes: invisible watermarking [1–3] and steganography [4–7]. Invisible watermarking is employed for copyright protection, traitor tracing, and digital authentication. Steganography, which is used for secure communication, pursues an increased embedidng capacity while sacrifices the high visual quality of stego-images. Therefore, un-detectability and embedding capacity are two major requirements that should be considered when designing new steganography algorithms, especially for image steganography. If the number of secret message bits embedded is increased, much more distortion is introduced in the stego-image. The distortion thus allows the existence of hidden message is discovered by a universal steganalyzer with high precision. In many real applications, the most fundamental criteria of image steganography is un-detectability, which means that the stego-images should have a high perceptual quality and statistically similarity to the cover-images while maintaining as high an embedding rate as possible.

Basically, the steganographic algorithms are divided into two categories: the spatial and transform domains. Least Significant Bit (LSB)-based steganography is one of the most popular image steganography algorithms in the spatial domain, in which the LSBs of pixels in cover image are replaced by secret bits. To protect the secret bits against recovery attack, the pixels, which are used in data hiding, are identified by pseudorandom number generator (PRNG). Nevertheless, some structural asymmetries have been introduced, and thus an existence of unseen messages is figured out easily, even at a low payload, when several proposed steganalytic algorithms, such as the Chi-square attack [8], regular/singular (RS) groups [9], and sample pair analysis [10] are employed.

In general, LSB matching (LSBM) is a slight enhanced version of LSB replacement algorithm. If a message bit is not equal to LSB of the considered cover pixel, then a value of this pixel is either added or subtracted randomly by one. Statistically, the probability of increasing or decreasing each altered pixel value is the same, and thus, the obviously asymmetric degradation caused by LSB modification can be easily prevented [11]. Hence, the typical steganalysis methods are inefficient for determining an existence of unseen message hidden by LSBM method.

Today, some effective steganalysis algorithms [12–13] have been proposed to analyse the LSBM method. Thus, Mielikainen proposed LSB matching revisited (LSBMR) in [11], in which a pair of pixels is utilised as a data hiding unit. One message bit is embedded into the LSB of the first pixel, and the relationship of the two pixel values carries another bit of a pair of message bits. By this way, the alternation rate of the pixels can be decreased in the case of maximum payload employed. This means that there is less distortion introduced in stego images at the same embedding rate in comparison with conventional LSB and LSBM algorithm. Hence, the LSBMR method obtains a higher security against statistical attack methods. Unfortunately, this approach modifies the smooth regions of cover-image to hide a secret message even at a low payloads. This issue increases the possibility of the secret message to be detected by visual attack techniques.

According to the aforementioned limitation of previous approaches, the pixel-value differencing (PVD)-based schemes, another type of edge-adaptive scheme were proposed [14–19]. According to these previous methods, the number of secret bits, which can be embedded, is estimated by the dissimilarity in value between the pixel and its neighbors. A larger difference value indicates that more secret bits can be hidden in considered embedding unit. PVD-based approaches futher reduce perceptible embedding noises in comparison with the typical LSB-based approaches when they have the same embedding rate. In PVD-based approaches, however, the pixels are selected to embed secret bits according to the raster scanning order. Therefore, it becomes possible to detect the pixels that carry the secret bits for detectors. Moreover, the smooth regions are still contaminated to embed secret bits; as a result, the human eye can more recognize easily distortion in the stego-image.

In [11], Lou *et al.* introduced an edge adaptive steganographic employs LSBMR (EA-LSBMR), which can adaptively choose image regions according to the length of the given message and the different value between two successive pixels of embedding units. At low embedding rate, EA-LSBMR employs textured regions to hide the secret bits. However, many smooth regions are altered at a high embedding rate while our human vision system (HVS) is sensitive to small alternations performed in these flat regions.

In [17], Sabeti *et al*. presented another edge adaptive steganographic method that based on LSB-Matching (LSBM). This method (named as CBL) employs a complexity measurement based on a local neighborhood to select the pixels are used to hide secret message bits. Nevertheless, the disadvantage in this approach is that the flat regions are contaminated to embed the secret message bits at high payloads still exists in this method.

In addition, the available capacity of the edge based image steganography methods is limited because the smooth regions in the image are avoided to be used in data hiding. In some cases, the number of available edge pixels is not enough to carry the given secret message, all pixels in the image will be used (by presented method in [11]). In this case, the security that introduced by embedding the secret bits into the edge pixels cannot be obtained.

To meliorate these disadvantages, we propose an adaptive multi-layer block data-hiding (MBDH) method that covert the secret bits in the edge pixels to increase the security performance and the perceptual quality of hidden data. The MBDH aims to solve the aforementioned weakness of previous approaches by addressing the following issues:

- For a low embedding rate, we hide the message bits in the high texture-characteristic regions. At higher embedding rate, to maintain the smooth regions, more bit-planes of pixels in selected regions are used to carry the message bits. Hence, a high perceptual quality of stego-image is obtained. Moreover, the stego-image, in which smooth regions are not altered, is secure against visual attack (such as an LSB enhancement attack).
- Based on our experiment, the maximum bit-planes (layers), which can be used to embed the secret bits, is five while the perceptual quality of the stego-image is guaranteed. As a result, the payload capacity is higher than that of previous edge adaptive-based approaches. Therefore, the proposed scheme can be applied to any image in the image database that used in the experiments.
- To guarantee the privacy of hidden data, the secret data are encrypted by XOR'ing with a random *keystream*. This *keystream*, which is simple generated by Matlab function ***rand***.
- To protect the secret data against recovery attacks, before embedding, in each divided block, two rows, which are identified by data key from the generated key stream, are exchanged. As a result, the hidden message cannot be extracted without key stream that used in embedding process.

The experimental results from evaluations on 10,000 natural images using universal steganalyser and specific visual attack illustrate the performance of the proposed algorithm.

The remainder of this paper is organised as follows. The background knowledge and related work are presented in Section 2. The proposed scheme is introduced in Section 3. In Section 4, an analysis of the experimental results is reported, and the conclusions are given in Section 5.

2. Background and Related Work

2.1. Pixel-value Differencing (PVD)-based Scheme

In [15], pixel-value differencing (PVD) approach was presented to hide secret data to gray-scale image. This method can hide more secret bits into cover image while still maintains a high perceptual quality of stego-image in comparison with the classical LSB method. It employs a pair of two consecutive non overlapping pixels of cover image as an embedding block and a number of secret bits can be hidden to each embedding unit is estimated by a difference between two pixels in the pair. The larger difference between two pixels, the more secret bits that can be embedded into an embedding unit. It is because the human eye is less sensitive to a large change in textured regions in an image. Meanwhile, the difference between two pixels in edge areas is larger than that of smooth regions.

In general, the embedding process is started from the left-to-right according to the zig-zag order of an image. Denote $d_{i,j}$ is an absolute of different value between two pixels P_i and P_j with gray values g_i and g_j, respectively, where j is the index of the first pixel in the considered pair. Then a pixel-value difference is identified as follow

$$d_j = |g_i - g_j|$$

The different value is in the range r_i with lower bound l_i and width w_i. The new value of two pixels in a pair are set such that the new difference is given by:

$$d'_j = \begin{cases} l_i + b_k & for\ d \geq 0 \\ -(l_i + b_k) & otherwise \end{cases}$$

where b_k is the corresponding decimal value of secret message bits to be hidden to the considered block and d'_j is the new pixels difference. Thus, the message bits are embedded into pixels in the pair by

$$(g'_j, g'_{j+1}) = \begin{cases} (g_j - \lceil m_j \rceil, g'_{j+1} + \lfloor m_j \rfloor) & if\ d_j\ is\ odd \\ (g_j - \lfloor m_j \rfloor, g'_{j+1} + \lceil m_j \rceil) & if\ d_j\ is\ even \end{cases}$$

where $m_j = (d'_j - d_j)/2$. If the new value of the pixels are outside of the range [0, 255], the considered pair will not be used to carry message bits. Therefore, the available embedding capacity is reduced.

In addition, if an image contains several smooth regions, then avoid embedding data into these regions lead to the further reduction of embedding capacity.

2.2. Analysis of EA-LSBMR

In this section, a brief overview of EA-LSBMR is provided. This scheme first computes some parameters used to select embedding regions, and then calculates the available embedding capacity of those selected regions. If the selected areas are not large enough for carrying the given message, then the parameters modification is performed, otherwise the data embedding is executed on the selected regions. The data embedding process of this scheme is organized in four steps as follow:

- **Step 1**. The cover image is divided into non overlapping arrays of pixels with the same size, rotating each block by a degree picked randomly from the set of $\{0, 90, 180, 270\}$, as identified by a secret key generated randomly. The obtained image is then rearranged as a row vector through raster scanning. Every two consecutive pixels from the row vector are regarded as an embedding unit.
- **Step 2**. The message's length and the difference between the pixels in the pairs are considered to select embedding regions.
- **Step 3**. LSBMR is employed to embed secret data in the embedding process. The order of all embedding units (from the step 2) is identified by PRNG.
- **Step 4**. The resulting row vector is transformed into blocks that are then rotated by a random degree (that opposite with a random degree in step 1) before re-construct the stego-image.

The important contribution of EA-LSBMR is reflected in selecting embedding regions adaptively by considering the texture characteristics of the image and the length of secret message. In other words, for lower embedding rates, only higher texture characteristic areas are utilised. While the other, smoother regions are retained. At the higher embedding rates, more edge regions are adaptively selected to carry the data by adjusting only a few parameters [11]. Unfortunately, the parameter alternation process can be repeated many times until the number of selected regions is enough for embedding the given message. This process can degrade the performance of EA-LSBMR.

Based on our experiments using an embedding rate in the range of [70%– 100%], the smoother regions are contaminated inevitably to embed the secret message in the EA-LSBMR scheme. Hence, the possibility of detection of visual attack methods (such as an LSB enhancement attack) is high for stegos that are introduced by EA-LSBMR.

Furthermore, in the EA-LSBMR scheme, the LSBMR algorithm is used as data hiding algorithm to embed secret bits into image regions. This means that the EA-LSBMR cannot overcome the capacity limitation of LSBMR. Note that if the value of the threshold $T = 0$, EA-LSBMR works as the conventional LSBMR, which means that in this case, this approach cannot achieve the high security introduced by hidding secret message bits to edge pixels.

3. Proposed Scheme

In this section, we propose the scheme that employs the textured regions in an image to hide secret bits, to increase the security performance against some existing steganalysis techniques.

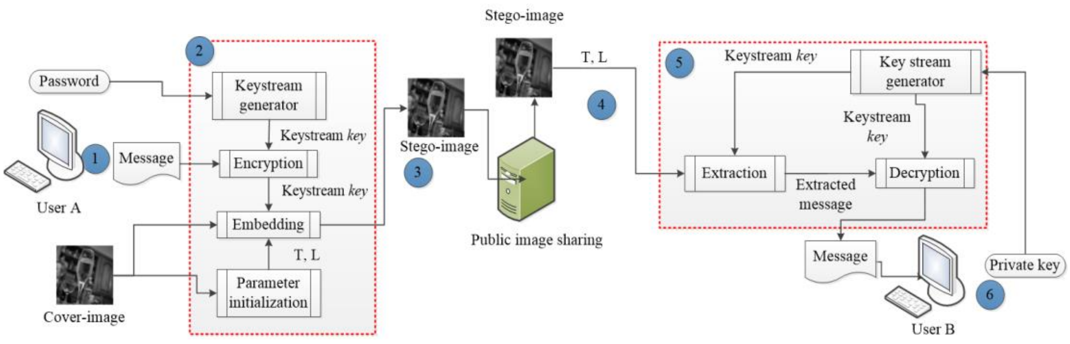


Figure 1. Conceptual Framework of the Proposed Scheme

As shown in Figure 1, the valuable data (such as business transaction information) is encrypted first. Then, this encrypted data is embedded into the digital images. A receiver then downloads the stegos from public image sharing to extract the hidden message.

The flow diagram of the proposed scheme is exhibited in Figure 2, in which the MBDH is employed to embed a secret message. In the data embedding process (Figure 2(a)), the scheme initialises some parameters, which are used for the data segmentation and region selection.

In data extraction (Figure 2(b)), the scheme first extracts the parameters, which are embedded into an unaltered region of a stego-image, then employs keystream *key* to recover the hidden message. Finally, the extracted message is decrypted by applying the bitwise XOR operator between the bits of this message and the keystream *key*.

The data embedding and data extraction algorithms are explained in details in sections 3.1 and section 3.2, respectively.

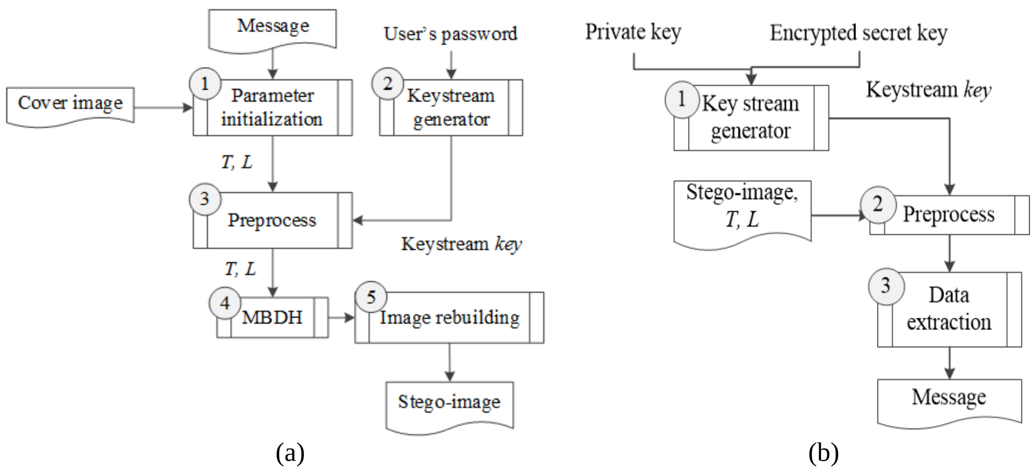


Figure 2. Proposed Approach (a) Data Hiding (b) Data Extraction

3.1. Data Hiding

- **Step 1.** A cover-image I and a length of message M are employed in the parameter initialization. The cover image is segmented into non overlapping blocks of $m \times n$ pixels (B). According to our extensive experiments, the optimized size of a block is 5 x 3 pixels to guarantee the balance between the visual quality of stego-images and security of unseen message. Based on the principle, the MBDH can employ more than one bit-plane of the pixels in the block to hide the secret message. Let denote L be the number of the bit-planes of the pixels that can be used in data hiding, T is the threshold to select the blocks of pixels. The values of two parameters T, L are determined according to the illustration in Figure 3.

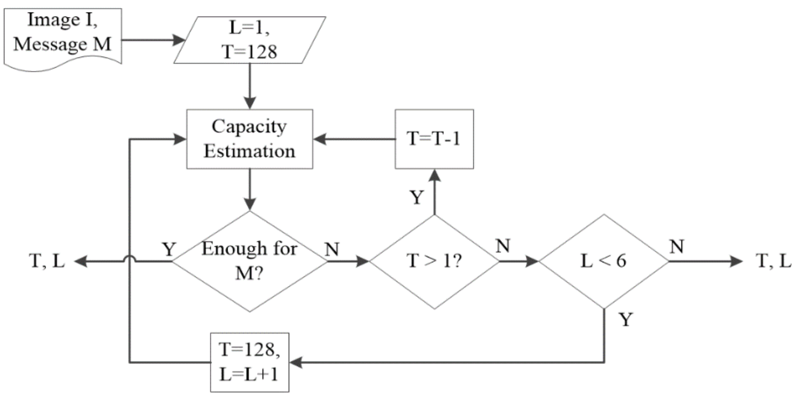


Figure 3. Parameter Initialization

The minimum value of parameter T is one, which means that the smoothest regions in the image will not be employed to hide the secret message. The number of layers (bit-planes), which can be used to carry the given secret message bits, is five. The reason is that five used bit-planes are enough to achieve the embedding rate of 100% (1 bit per pixel) for all images in the used image database.

- **Step 2.** The key stream *key* is generated by using the initial data created by a given password from the user. This key stream generator, in which a Gray-code is employed, is presented in detail in Section 3.4. The generated key stream is not truly random as the data that come from natural sources (such as radio noise, atmospheric noise, etc.). However, the non-repetitiveness of the keystream *key* is enough to guarantee the security of encrypted data.

- **Step 3.** The random row swapping is applied to each block of pixels *B*. The positions of rows, which are swapped, are determined by the keystream *key*. This process leads to the improvement of the security against recovery attacks. An attacker cannot extract the hidden message precisely without the key stream *key*.
- **Step 4.** An encrypted message is embedded into the selected regions of the stego-image by the proposed algorithm, MBDH.
- **Step 5**. In this phase, the rows of the resulting blocks (with secret bits embedded) are re-arranged as the original order in cover block and this order is identified by the keystream *key*.

3.2. Data Extraction

The data extraction constitutes of the following three steps.

- **Step 1**. With a provided private-key, the *keystream* key is generated. Then, the threshold T and the number of layers L, which are extracted from the stego-image, are used to extract the hidden data.
- **Step 2**. The stego-image is split into blocks of $m \times n$ pixels, and the rows in the blocks are then swapped. The positions of rows in the blocks, which will be swapped, are identified by the *keystream* key.
- **Step 3**. The data extraction travels through the stego-image and examine a block, which contains the secret bits, whose maximum absolute difference (*MAD*) is greater than or equal to the threshold T until the last block in the stego-image is reached. The nibbles of data are extracted from the selected blocks based on the number of layers used to hide the secret message in the data-hiding process.

The metric *MAD* that is used to select a region to embed secret bits is described in detail in Section 3.3.

3.3. Region Selection

To select the texture regions in the cover-image to embed secret bits, an absolute difference in value of adjacent pixels in the block is calculated. Horizontal and vertical absolute differences of adjacent pixels are computed as follows:

$$d_{i,j}^{h} = \left|p_{i,j} - p_{i,j-1}\right| \quad (1)$$

$$d_{i,j}^{v} = \left|p_{i,j} - p_{i-1,j}\right| \quad (2)$$

where $p_{i,\,j}$ represents the value of the pixel in the ith row and jth column of the block. The *MAD* of the xth block in the image is determined as follows:

$$MAD_x = \max_{1 \le i \le m, 1 \le j \le n} \left\{d_{i,j}^{h}, d_{i,j}^{v}\right\} \quad (3)$$

A region, which has a *MAD* greater than a threshold *T*, is selected to embed the secret bit. Because the edge regions are characterised by a high *MAD*.

3.4. Keystream Generation using Gray-code

To increase a security of hidden data against being extracted by an opponent (stegalyzer), in this Section a significantly simple *keystream* generator using Gray-code is designed.

The Gray code is a binary numerical system, in which two consecutive values differ in only one bit. This code was originally invented to avert undesired transient states or outputs from electro-mechanical switches. Nowadays, it is employed in error correction in digital communications and digital-to-analog converters [20].

Three following steps conduct the key stream generator.

- **Step 1**. User's password is employed to produce a *seed* for Matlab's function ***rand***. The *seed* is then encrypted by a public-key cipher RSA [21] and only receiver who owns the private key can decrypt the encrypted seed.
- **Step 2**. A pseudo-random data is generated by ***rand*** in such that the length is enough to encrypt the given message.
- **Step 3**. Each data byte of the generated keystream is converted to Gray-code to increase the randomness of the obtained keystream.

The *seed* in its encrypted form is transferred to the authorized receiver to re-generated the keystream which is used in data hiding and message encryption.

3.5. Multi-layer Block Data-Hiding Algorithm

For each block of $m \times n$ pixels that is selected by comparing the *MAD* of the block with threshold *T*, an array of nibbles of data (of size *L*, which is the number of layers of pixels that can be used to embed secret data) is embedded by binary block data-hiding algorithm. The L^{th} layer of pixels in the block is employed to embed secret bits first, and the lower layers $(L-1, L-2, \ldots, 0)$ are then used to avoid a conflict caused by an adaptive adjustment process.

In the embedding process, there are some embedded blocks in which the *MAD* is smaller than the threshold *T*; therefore, the variable *bEmb* is used to control the message segmentation process. If the value of *bEmb* is **true**, then the nibble is filled with the new data from the message stream; otherwise, the data from backup the nibble are used. An adjustment phase is performed to guarantee the extraction process can correctly locate the embedding blocks. In this process, the blocks, which has *MAD* is smaller than threshold *T* after embedding, are restored the same as in the cover image before altering it to guarantee that the *MAD* of this block is smaller than *T*. This alternation is applied in such a way that the degradation to the block is minimized.

Algorithm Multi-layer block data-hiding

```
Input: Gray-scale image C, secret message M, two estimated parameters L and T
Output: stego-image S
1:  bEmb ← true;
2:  while (curX < imWidth && curY < imHeight)
3:    BC ← getBlockOfPixels(C, m, n);
4:    MAD ← estimateMAD(BC);
5:    if (MAD ≥ T) then
6:      if bEmb then
7:        nibArr ← getNibbleArray(M, L); i ← 0;
8:        backupNibArr ← nibArr;
9:      else
10:       nibArr ← backupNibArr; i ← 0;
11:     end if
12:     Layer ← L;
13:     while (Layer ≥ 0)
14:       nib ← nibArr(i);
15:       BC ← binaryBlockDataHidingAlgorithm(BC, nib, Layer);
16:       Layer ← Layer - 1; i ← i + 1;
17:     end while
18:   end if
```

```
19: $bEmb \leftarrow$ estimateMAD($BC$) $\geq T$ ? true: false;
20: saveStegoArrayToImage($BC$, $S$);
21: **if** end of message **then** break;
22: **end while**
```

The binary block data-hiding algorithm (which is described in section 3.5.1) is used to hide a nibble of secret data to a block of cover bits (obtained from LSBs of pixels) at the specific layer.

3.5.1. Binary Block Data-Hiding Algorithm: In this section, the binary block data-hiding, which embeds a data byte d into a binary block of size of $\boldsymbol{m} \times \boldsymbol{n}$, is presented. The major principle of this method is that the secret bits are carried by a total S calculated by the value of the bits and their positions in the binary block ($\boldsymbol{m} \times \boldsymbol{n}$). There are 2 bits at most in the block are modified without employing an additional matrix or vector.

We denote F is a binary block size of $m \times n$, and d $(0 \leq d \leq m \times n)$ is a message value which is embedded into F. S is defined as

$$S = \sum_{i=1}^{m} \sum_{j=1}^{n} F(i,j)\big((i-1)n + j\big)(mod\ K) \tag{4}$$

where K is assigned as $K = K^* + 1$ (with $K^* = m \times n$). To hide d to the binary block F (as shown in Figure 4(a)), the important process of this algorithm is determination the positions of bits in F that can be altered in value to make the total S equal to d. In the extraction stage, the value of d is calculated via Eq. (4) from the binary block of a stego object.

Example 1. Consider the given binary block size of 3 x 5

0	1	0	0	0
0	1	1	0	1
1	0	0	0	1

(a)

0	1	0	0	0
0	1	1	1	1
1	0	0	0	1

(b)

Figure 4. Binary Block *F* (a) and *F'* (b) (after embedding)

From F (shown in Figure 4(a)), $S = 5$ (calculated using Eq. (4)) and $K = 16$. Assume that $d = 14$, flipping the bit at position (2, 4) causes the value of S is increased by an amount $((i-1)n + j) = ((2-1)5 + 4) = 9(mod\ 16)$ or $S' = S + 9(mod\ 16) = 14(mod\ 16)$.

Lemma *For all integers $d \in [0, K)$, a maximum of two bits in block F need to be altered to transform F into F'.*

$$S' = \sum_{i,j}^{m} \sum_{i,j}^{n} F'(i,j)\big((i-1)n + j\big)(mod\ K) = d(mod\ K) \tag{5}$$

Assume that Ω_1 is a collection of the position of bits that satisfy:

$$\Omega_1 = \{(i,j): F(i,j) = 1, (i-1)n + j = b\} \cup \{(i,j): F(i,j) = 0, (i-1)n + j = K - b\} \tag{6}$$

We denote Ω_2 as a collection that contains the position of a pair of bits and modification these bits in this set makes the value of S equal to d. Ω_2 is enumerated based on criteria as follows:

$$\begin{aligned}\Omega_2 = &\big\{\{(i,j),(p,q)\}: F(i,j)=0, F(p,q)=1, in+j = pn+q-b\big\}\\ &\cup \big\{\{(i,j),(p,q)\}: F(i,j)=1, F(p,q)=1, (i-1)n+j+(p-1)n+q\\ &= b(mod\ K)\big\}\\ &\cup \big\{\{(i,j),(p,q)\}: F(i,j)=0, F(p,q)\\ &= 0, (i-1)n+j+(p-1)n+q+b = 0(mod\ K)\big\}\end{aligned} \quad (7)$$

In example 1, we can enumerate

$$\Omega_1 = \{(2,2),(2,4)\}$$

and

$$\Omega_2 = \big\{\{(1,1),(2,3)\},\{(1,3),(2,5)\},\{(1,4),(3,1)\},\{(2,3),(3,5)\}\big\} \cup \big\{\{(1,3),(2,1)\},\{(1,4),(1,5)\},\{(3,2),(3,3)\}\big\}$$

Clause 1. *For any binary block F and given d, two sets Ω_1 and Ω_2 are not simultaneously empty.*

Clause 2. *In the binary block F, flipping the bit at (i, j) or the pair of bits at the set $\{(i,j),(p,q)\}$ causes $S' = d(mod\ K)$ when $(i,j) \in \Omega_1$ or $\{(i,j),(p,q)\} \in \Omega_2$.*

Mathematical proof of Lemma, Clause 1, and Clause 2 were given in details in [22].

Algorithm

From a previous discussion that explain a principle of binary block data-hiding algorithm. Assume that B (size $K^* = m \times n$) is a block of pixels of a gray-scale image and a message byte $d \in [0, K^*]$. Denote k as the kth layer (bit-plane) of a pixel and $B(i,j).k$ as the kth bit of the pixel (i,j) in the block of pixels B.

- **Step 1**. Set $F(i,j) = B(i,j).k$ with $1 \le i \le m$, $1 \le j \le n$. Then, Ω_1 and Ω_2 are determined by formulas (6) and (7).
- **Step 2.** Measure the changed values if the pixels in Ω_1 and Ω_2 are modified.
  If Ω_1 is not empty, denote
  $$C_1 = \arg\min\{LN(B(i,j),k): (i,j) \in \Omega_1\} \quad (8)$$
  and the position (i_0, j_0), which is in this set Ω_1, is selected to flip if the condition in Eq. (9) below is satisfied:
  $$(i_0 j_0): LN(B(i_0,j_0),k) = C_1 \quad (9)$$
  If Ω_2 is not empty, C_2 is identified as follow:
  $$C_2 = \arg\min\{LN(B(i,j),k) + LN(B(p,q),k): \{(i,j),(p,q)\} \in \Omega_2\} \quad (10)$$
  the pair of positions {(i_o, j_o), (p_o, q_o)} ∊ Ω_2 that satisfies:
  $$\{(i_0, j_0),(p_0,q_0)\}: LN(B(i_0,j_0),k) + LN(B(p_0,q_0),k) = C_2 \quad (11)$$
  is selected to altered. In Eq. (11), $LN(B(i_0,j_0), k)$ is the value of the pixel after the bit at the kth layer of the pixel is flipped. This value of modified pixel is then adjusted by the adaptive adjustment (which is described in detail in section 3.5.2) to reduce the degradation that caused by bit flipping.
- **Step 3**. If $C_1 \le C_2$ then the bit (i_0, j_0) is flipped, otherwise, the bits at the set {(i_o, j_o), (p_o, q_o)} are flipped.

Example 2. Suppose that there is a block of pixels B of size 5 x 3.

68	58	43
56	48	48
103	103	110
127	123	124
127	109	102

Suppose a data byte $d = 13$ and the binary block F that contains the bit obtained from the 4^{th} layer of the pixels in block B:

0	1	1
1	0	0
0	0	1
1	1	1
1	1	0

We have $S = 14$, $\Omega_1 = (5, 3)$ and
$\Omega_2 = \{\{(1,1),(1,2)\},\{(3,2),(3,3)\},\{(2,1),(5,1)\},\{(5,2),(1,3)\},\{(3,1),(3,2)\}\}$

To embed the value d into F, the bit in the collection Ω_1 is flipped, and the pixel at (5, 3) is then adjusted by an adaptive adjustment. As a result, the changed value is 2 (the new value of the pixel at (5, 3) is 104 instead of 110 after adjustment applied), and only one pixel in the block is altered to embed the data byte $d = 13$ at the 4^{th} of the pixels in the block.

3.5.2. An Adaptive Adjustment Process: Embedding the secret bits into a high LSB layer introduces more distortion to the pixels. For this reason, an adaptive adjustment process is applied to minimize the distortion.

If the k^{th} layer is altered to carry the secret bit, and k is equal to L (L is the number of layers that are used to embed the secret bits into block of pixels), then the adjustment algorithm in [23] is employed; otherwise, the algorithm proposed in [24] is applied. Nevertheless, the algorithm in [24] was revised due to errors in sub-case 1.4 and 2.4. In sub-case 1.4, if there is no bit "1" to the left of the k^{th} bit, then all of the bits that are "0" will be set to "1". In the sub-case 2.4, if there is no bit that is "0" on the left of the k^{th} bit, then all of the bits that are "1" will be set to "0". The revision is done by checking the existence of the bit "1" for sub-case 1.4 and the bit "0" for sub-case 2.4. Then the result of this stage is used to identify that the adjustment is performed at both sides of the k^{th} bit or only to the right of this bit.

4. Experimental Results and Analysis

In this section, obtained experimental results are presented to demonstrate the effectiveness of the proposed algorithm in comparison with previous algorithms.

The image database break our steganography system (BOSS) [25] is used to evaluate the performance of the proposed approach. This database contains 10,000 gray-scale images with a size of 512 x 512 pixels in PGM format. These images were converted to bitmap (BMP) format via command **imwrite** of Matlab.

4.1. Embedding Capacity and Image Visual Quality Discussion

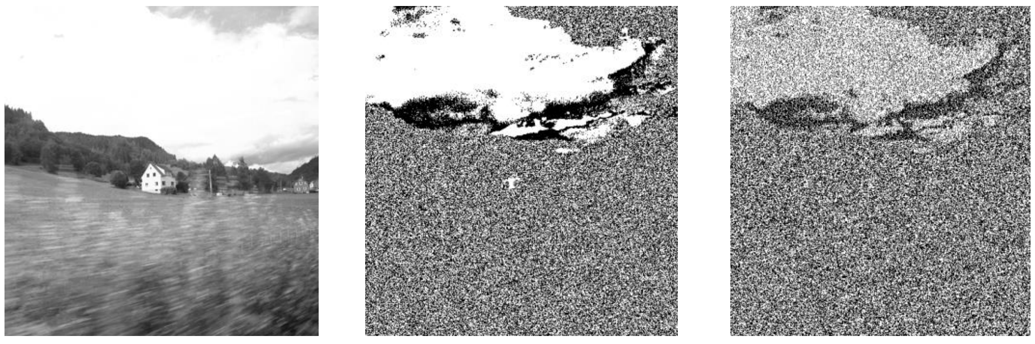
One of the important improvements of the proposed method is that it can employ more than one layer of the pixels to hide the secret message bits. This means that employing the high texture characteristic regions in data hiding to obtain the high security is an advantage of edge-based approaches.

The proposed scheme firstly chooses the texture regions for embedding according to the length of the secret message by adjusting the estimated threshold T, and the number of bit-planes (Layers-L) of pixels that can be used in the data hiding process. The block, which has a MAD value greater than or equal to the threshold T, is selected to hide the secret bits. The minimum value of the threshold T is 1, which means that the smoothest regions are never used to embed the secret bits. If there are not enough texture blocks that can be used to embed a given secret message, more secret bits are embedded into selected regions (the number of layers L is increased). An experiment shows that when the value of L is 5, the proposed algorithm can obtain an embedding capacity of more than 100% while the perceptual quality is guaranteed. This is because the size of the block is $5\,x\,3$, the block data-hiding algorithm can hide a value d ($0 \le d \le 5 \times 3$, which corresponds to 4 bits) in a binary block that is formed by the bits at one layer (bit-plane) of the pixels in the block. Thus, when the value of L is 5, there are 20 bits are embedded into a block of pixels size $5\,x\,3$, the availabe embedding rate is 1.33 bpp. Consequently, the embedding payload of our novel approach is higher than that of CBL, LSBM, LSBMR [26], and EA-LSBMR [11].

Table 1. The Percentage of the Available Images that can be used to Hide Data for Embedding Rate of [80%, 90%] for Three Steganographic Schemes

Embedding rate	**Percentage**		
	EA-LSBMR	CBL	MBDH
80	43.51%	99.41%	100%
90	12.14%	97.77%	100%

As seen in the Table 1, the proposed scheme can obtain the embedding rate of [80%–90%] for all images in the database. CBL can hide secret data to 99.41% and 99.77% of images at embedding rates of 80% and 90% respectively. Therefore, the values of some metrics (PSNR, wPSNR, and SSIM) cannot measure from the stegos that introduced by this steganographic algorithm. This is because if a cover image cannot embed the data at the high embedding rate, it then is kept as an original.



(a) (b) (c)

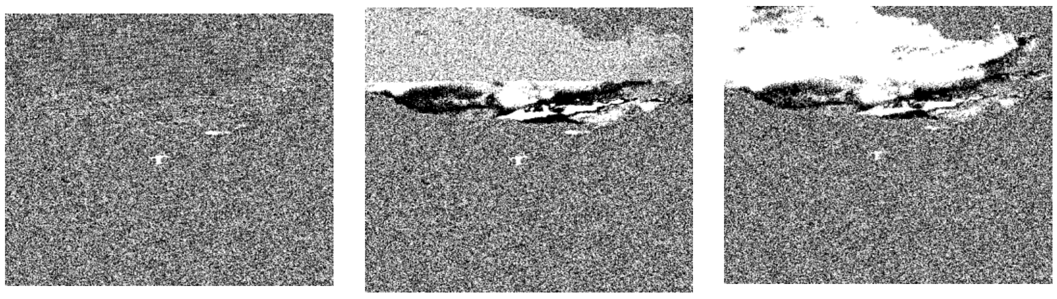


Figure 5. (a) Cover Image and its LSB (b), (c)-(f) LSBs of Stego-Images Produced by the Four Previous Steganographic Approaches and Proposed Approach at Embedding Rate of 70%; (c) LSBMR, (d) PVD, (e) Triple PVD and (f) MBDH

In contrast to CBL, EA-LSBMR employs all the pixels in the image if the texture characteristic of the considering image is not enough to hide all the secret bits to edge pixels. Nevertheless, in this case, the anti-detection capability is lower than that of the stegos that created by EA-LSBMR scheme.

Figure 5 indicates that for the embedding rate is 70%, the sky in the stego-image of MBDH (as shown in Figure 5(f)) is not modified, while this region of other stego-images (created by three previous steganographic methods: LSBMR, PVD, TPVD) is contaminated to hide the secret message. Thus, the existence of hidden data can be easily identified by visual attacks [8].

The average peak signal-to-noise-ratio (PSNR), weigh-PSNR (wPSNR is an enhanced image quality metric that is adopted in [27]) estimated from 10,000 cover images and their corresponding stego-images under various embedding rates for five steganographic methods is shown in Table 2 (the numbers in bold point out the best values in the corresponding cases). The string "N/A" marks that the CBL steganographic algorithm cannot hide secret message at embedding rate of 90% or greater than.

wPSNR metric accounts for human visual system (HVS) characteristics and enhances the PSNR metric using

$$wPSNR = 10\,log_{10}\frac{max(I)^2}{\|NVF(I'-I)\|^2}$$

where I is a cover image, and I' is a corresponding stego-image. NVF indicates the noise visibility function described in [27].

Table 2. Average PSNR, wPSNR over 10,000 stego-images for Different Steganographic Algorithms and Embedding Rates

Embedding rate	LSBMR		PVD		CBL		EA-LSBMR		MBDH	
	PSNR	wPSNR	PSNR	wPSNR	PSNR	wPSNR	PSNR	wPSNR	PSNR	wPSNR
10	62.3	62.9	57.3	62.6	59.3	66.6	61.5	66.3	**61.7**	**67.0**
20	59.2	61.2	54.1	59.5	56.3	65.3	58.3	64.8	**58.7**	**65.8**
30	57.5	61.2	52.3	57.2	54.5	64.2	56.4	63.8	54.8	**65.2**
40	56.2	61.2	51.0	56.8	53.3	63.4	54.9	62.9	53.5	**64.8**
50	55.3	60.3	49.9	54.4	52.3	63.0	53.6	62.3	52.6	**64.1**
60	54.5	60.7	49.1	54.2	51.5	62.1	52.1	61.9	51.7	**63.3**
70	53.8	60.6	48.3	53.1	50.8	61.9	50.3	61.7	**51.0**	**62.8**
80	53.2	60.1	47.7	52.7	50.3	61.7	43.8	61.4	**50.4**	**62.4**
90	52.7	60.1	47.0	52.0	N/A	N/A	32.7	61.2	**49.8**	**62.1**

For the average PSNR, the LSBMR method is superior to the other methods because the ±1 embedding scheme is employed and the value of PSNR is independent of the position of the altered pixels.

In the proposed method, if the *MAD*s of the embedded blocks are smaller than the threshold *T* after embedding, then these blocks are adjusted in such a way that the distortion is minimized, and their *MAD* must be smaller than *T* to achieve the correct data-extraction. Thus, the average PSNR of stego-images introduced by our proposed approach is not much better than that of EA-LSBMR at the low embedding rates (from 10% to 60%). With the higher embedding rates (from 70% to 90%), the average PSNR estimated from cover-images and their corresponding stego-images produced by our proposed method is higher than that of EA-LSBMR. Because the EA-LSBMR employs more pixels for data hiding at the high embedding rates.

For average wPSNR, our proposed method overcomes the previous algorithms in term of texture characteristics at different embedding rates. This achievement is obtained due to the use of pixels at sharper edge regions to carry a secret message even at a high embedding rate. As reported by the definition of NVF in [27], the weighting for changes in the higher texture characteristic regions is smaller than that in the flat regions. Consequently, the wPSNR value of stegos introduced by MBDH should become higher than those of the stegos that produced by a random embedding scheme (LSB-based approaches) and EA-LSBMR.

To further evaluate the perceptual quality of stego-images produced by MBDH, a structural similarity index (SSIM) [28], which is a method for measuring the similarity between the cover and stego images, is performed. The SSIM indexes are measured from the stegos of the proposed scheme, and four previous approaches. In Table 3, the average SSIM values measured from 10,000 cover images and their corresponding stego-images were listed under different embedding rates.

Table 3. Average SSIM Values over 10,000 stego-images with Different Steganography and Embedding Rates

Embedding rate	SSIM				
	LSBMR	PVD	CBL	EA-LSBMR	MBDH
10%	0.9997	0.9992	0.9998	0.9998	0.9999
20%	0.9993	0.9982	0.9992	0.9994	0.9996
30%	0.9989	0.9973	0.9985	0.9989	0.9990
40%	0.9986	0.9964	0.9975	0.9979	0.9984
50%	0.9983	0.9956	0.9967	0.9967	0.9977
60%	0.9980	0.9948	0.9953	0.9927	0.9969
70%	0.9976	0.9940	0.9943	0.9818	0.9960
80%	0.9973	09934	0.9929	0.9119	0.9949
90%	0.9970	0.9927	N/A	0.7695	0.9937

It is observed from Table 3, that the average SSIM values for the proposed algorithm are higher than that of three previous steganographic methods (PVD, CBL, and EA-LSBMR) at different embedding rates. As it can be seen, the SSIM of LSBMR algorithm is higher than that of the proposed scheme at embedding rate of [40%–90%] due to its high embedding efficiency (number of secret message bits embedded per embedding change).

4.2. Visual Attack

The LSB enhancement attack method is performed by a StegSecret tool [29] to measure the security of our proposed approach against visual attack. From Figure 6 it can be seen that the LSB bit-plane (results from LSB enhancement attack) of the cover images and the corresponding stegos that introduced by our proposed method, with an embedding rate of 90%, are very similar. While the smooth regions of the stegos that introduced by EA-LSBMR are contaminated to carry secret bits because a higher embedding rate is used. Therefore, the existence of hidden information in the stegos of EA-LSBMR is detected with a high precision by the LSB enhancement attack. This problem also exists in LSBM, LSBMR, and PVD that employ a PRNG to spread the message over the stegos; the smooth regions would be certainly disturbed and thus become more random [30].

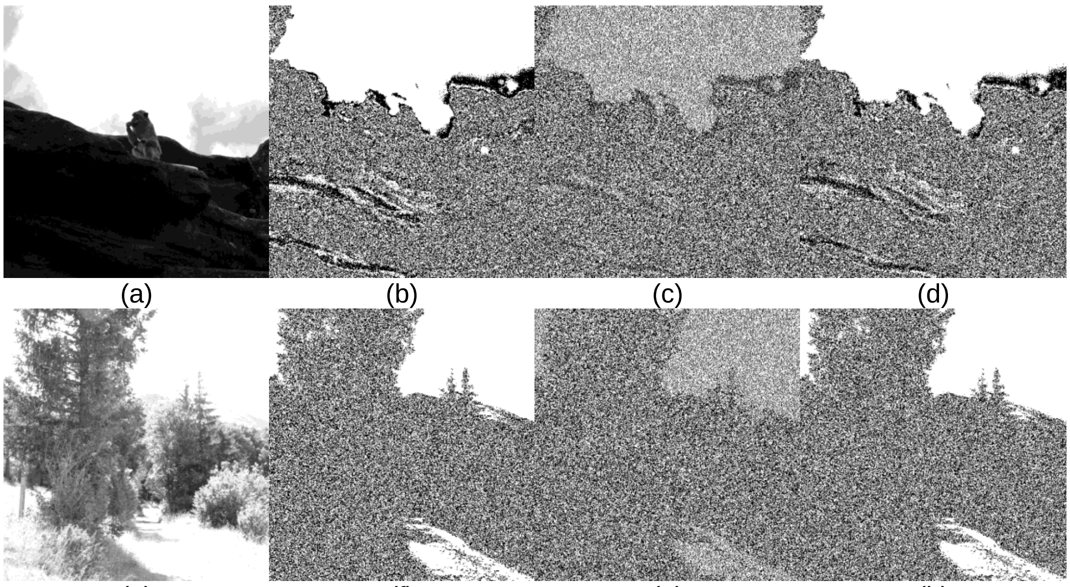


Figure 6. The Cover Images (a, e) and their LSB Planes (b, f). (c, g) LSB of stegos of EA-LSBMR and (d, h) LSB of stegos of the Proposed Method with the Same Embedding Rate of 90%

It is evident that the stego-images, which are created by the proposed approach, is secure against LSB enhancement attack when compared with other corresponding previous approaches.

4.3. Steganalysis using Amplitude of Local Extrema (ALE)

According to the principle of CBL and EA-LSBMR, these two schemes are edge adaptive approach based on LSBM. Thus, the LSB matching steganalysis is employed to examine the security of our proposed method and these two previous approaches (CBL and EA-LSBMR).

In this experiment, the sums of absolute differences between each local extremum and its neighbors in the histogram are calculated. Assume that D_c and D_s are the sum estimated from cover and stego images, respectively. It is claimed that $D_c > D_s$ for any stego-image created by LSB matching steganography [31]. Detection accuracy (P_{detect}) is computed using below equation.

$$P_{detect} = 1 - P_{error}$$

and

$$P_{error} = \frac{1}{2} \times P_{FP} + \frac{1}{2} \times P_{FN}$$

where P_{FP} , P_{FN} are the probabilities of false positive and false negative, respectively. A value of $P_{detect} = 0.5$ indicates that the classification is as good as random guessing and $P_{detect} = 1.0$ indicates a classification with 100% accuracy.

Figure 7 illustrates the security of the proposed scheme and two previous edge adaptive steganography against steganalysis based on ALE. At low embedding rates of [10%–20%], the number of used bit-planes of pixels in the block in hiding process are small. Therefore, the change caused by data embedding is same as the effects that introduced by LSB matching steganography on the histogram. Hence, the P_{detect} of the proposed scheme is higher than that of CBL but is still lower than that of EA-LSBMR.

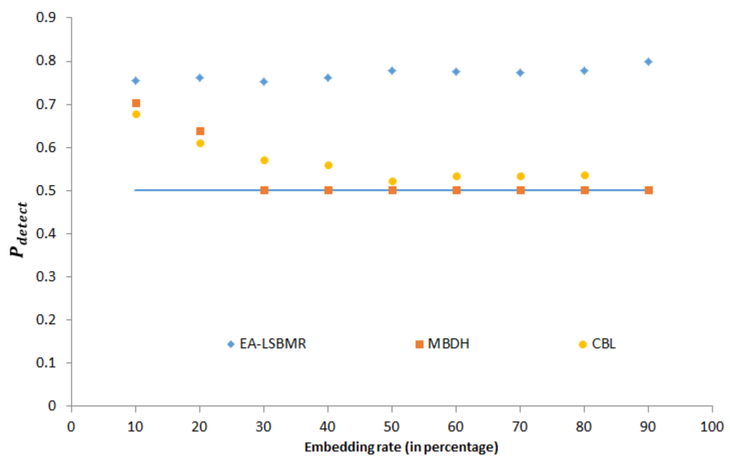


Figure 7. Security of the Proposed Scheme Compared with CBL and EA-LSBMR for Different Embedding Rates when Attacked by ALE

For high embedding rates in the range of [30%–90%], the proposed scheme hides more secret message bits to an edge region. This leads to the increasing of the change in modified pixels and therefore the degradation of data hiding is not same as the effect that causes by LSB matching methods.

To further evaluate the security of the proposed scheme against the steganalyzer using ALE, for every cover-images and stego-images (at 50% of embedding rate), the ALE features are obtained and the cover-features database and stego-features database then randomly split into a training set (80% of database size) and a test set (the remaining 20% of the database). These two divided feature sets are then used in the classification by the LibSVM toolbox [32]. The hyper-parameters (C, γ) are optimized employing five-fold cross-validation to achieve a higher accuracy for the classification. After the optimized parameter pair (C, γ) is selected, it will be used to get the training model by employing the *svmtrain* function of LibSVM toolbox. At last, the obtained training model is used to predict the testing data.

Table 4. Detection Accuracy Rates of the Proposed Scheme Compared with CBL and EA-LSBMR

Embedding rate	Accuracy		
	CBL	EA-LSBMR	MBDH
50%	60.25%	61.0%	57.0%

As seen in the Table 4, the detection accuracy rate of the steganalysis using ALE on MBDH is lower than that of CBL and EA-LSBMR. This means that the proposed algorithm is more secure than two previous algorithms against the steganalysis using ALE feature set.

4.4. Security Analysis under Universal Steganalyzers

4.4.1. Ensemble Classifier: The ensemble classifier introduced in [33] was designed to keep low complexity and overall simplicity. Ensemble classifiers were implemented that was composed of random forests, and the authors argued that this method is ideally suited for steganalysis. Ensemble classifiers scale much more favourably with respect to the number of training examples and the feature dimensionality, and they have performances comparable to those of the much more complex support vector machines (SVMs). Out-of-bag (OOB), an unbiased estimate of a real testing error, is obtained from ensemble classifier. The high value of OOB indicates the high security against this attack.

Table 5. The Average OOB Errors of our Proposed Method (MBDH) and the Previous Algorithms

	OOB					
Embedding Rate	**PVD**	**LSBMR**	**CBL**	**EA-LSBMR**	**MBDH**	**Improvement**
10%	0.1634	0.1844	0.2321	0.3660	0.4587	24.3 %
20%	0.0906	0.1302	0.0923	0.3388	0.4252	25.0 %
30%	0.0776	0.1104	0.0281	0.2804	0.3346	19.3 %
40%	0.0652	0.0910	0.0121	0.2164	0.2468	14.0 %
50%	0.0501	0.0796	0.0043	0.1304	0.1746	32.7 %
60%	0.0416	0.0562	0.0040	0.0718	0.1227	70.9 %
70%	0.0188	0.0326	0.0027	0.0358	0.0771	115.4 %
80%	0.0166	0.0180	0.0023	0.0198	0.0413	108.6 %
90%	0.0106	0.0160	N/A	0.0126	0.0277	119.8 %

The feature extractor SPAM [34] is employed to extract features from cover images and corresponding stegos, and SPAM has 686 features to use in ensemble classifier.

Table 5 shows the average OOB errors (obtained error from classification) for our proposed method (MBDH) and for the previous approaches (PVD, LSBMR, EA-LSBMR, and CBL) at difference embedding rates. Percentage of improvement illustrates the improved security of MBDH in comparison to EA-LSBMR. It is clear that the average OOB errors of our proposed method are higher than those of all of the previous algorithms.

By using the low embedding rates in the range of [10%–20%], and employing the binary block data-hiding approach, the OOB errors of our proposed method are significantly higher than that of EA-LSBMR. For the embedding rate of 30% to 40%, the parameter initialisation process sets the value of parameter L and threshold T as same as these values of lower embedding rate (10%-20%) while the number of secret bits is increased. As a result, although the OOB errors of our proposed method are relatively higher than those of EA-LSBMR, the MBDH cannot obtain a higher security as same as at other embedding rates.

4.4.2. Quantitative Steganalysis using Rich Models: In [35], Kodovsky *et al.*, presented a regression framework for steganalysis of digital images that uses the recently proposed rich models – high-dimensional statistical image descriptors. According to illustrated experimental results, the proposed scheme outperforms previous quantitative steganalysis (both structural and feature-based) under various steganographic schemes.

In this section, we attack the proposed scheme and two previous steganographic algorithms (EA-LSBMR and CBL) using the 686-dimensional SPAM. The image database BOSS is randomly split into training and testing sets. The stego-images were generated with payloads uniformly distributed in the range of [0%, 80%] numbers of pixels of cover-image are used to carry secret message bits.

The mean square errors (MSEs) measured from the testing set illustrate the performance of the proposed scheme and two previous steganography algorithms (CBL and EA-LSBMR).

Table 6. Security of the Proposed Scheme and other Algorithms (CBL and EA-LSBMR) against Quantitative Steganalysis

MSE			
EA-LSBMR	**CBL**	**MBDH**	**Improvement**
$0.95 \bullet 10^{-3}$	$1.08 \bullet 10^{-3}$	$1.42 \bullet 10^{-3}$	31.5%

Table 6 shows the final MSE of testing of three steganography algorithms (EA-LSBMR, CBL and MBDH). The last column shows the improvement over the CBL method. From Table 6, it is evident that the anti-detection capability of the proposed scheme is higher than that of two algorithms (CBL and EA-LSBMR) for quantitative steganalysis.

5. Conclusion

In this paper, an adaptive multi-layer block data-hiding approach that is based on edge detection is proposed. By ultilization the block data-hiding algorithm, the secret bits are embedded with at most two positions changed in a block of pixels at a specified layer (bit-plane). As a result, the perceptual quality of the stegos is guaranteed. Furthermore, only sharper regions in the image are contaminated to hide the secret message even at the high embedding rates. Thus, the stegos that created by proposed algorithm is secure against visual attack. An experiment that used 10,000 natural images and performed different types of steganalysis showed that both the perceptual image quality and the security of stego-images, which are introduced by our proposed scheme, are significantly enhanced in comparison with edge-adaptive-based and LSB-based approaches.

Moreover, the proposed scheme provides a higher embedding capacity in comparison with the existing edge-based approaches. By employing more bit-planes (layers) of pixels in the block, the proposed scheme can hide the given message at embedding rate of 100% for all images in the used image database.

Our future work will be directed toward improving the proposed algorithm to further reduce the distortion to cover-image and apply it to another digital format mediums, such as video format.

Acknowledgement

We would like to express the deepest sincere gratitude to Khon Kaen University, Faculty of Science, Department of Computer Science for their support and assistance.

References

[1] Dubey, N.K., Kumar, S., An effective approach of distortion-resistant video watermarking for piracy deterrence. International Journal of Security and its Applications, vol. 9, issue. 1, (**2015**), p. 283–294.

[2] Cho, D.-J., Watermarking scheme of MPEG-4 LASeR object for mobile device. International Journal of Security and its Applications, vol. 9, issue. 1, (**2015**), p. 305–312.

[3] Zhao, H., Fang, Y., On the security and robustness with fingerprint watermarking signal via compressed sensing. International Journal of Security and its Applications, vol. 9, issue. 1, (**2015**), p. 221–236.

[4] Yang, W.-C., Chen, L.-H., Lee, C.-H., A difference expansion based reversible data hiding algorithm using edge-oriented prediction. International Journal of Security and its Applications, vol. 8, issue. 6, (**2014**), p. 173–184.
[5] Wang, B., Qian, H., Sun, X., Shen, J., Xie, X., A secure data transmission scheme based on information hiding in wireless sensor networks. International Journal of Security and its Applications, vol. 9, issue. 1, (**2015**), p. 125–138.
[6] Kasana, G., Singh, K., Bhatia, S.S., Steganography technique for JPEG2000 compressed images using histogram in wavelet domain. International Journal of Security and its Applications, vol. 8, issue. 6, (**2014**), p. 211–224.
[7] Swain, G., Steganography in digital images using maximum difference of neighboring pixel values. International Journal of Security and its Applications, vol. 7, issue. 6, (**2013**), p. 285–294.
[8] Westfeld, A., Pfitzmann, A., Attacks on Steganographic Systems. Proceeding of Information Hiding (Editor: A. Pfitzmann). Berlin, Heidelberg: Springer Berlin Heidelberg, (**2000**), p. 61–76.
[9] Fridrich, J., Goljan, M., Rui Du, Detecting LSB steganography in color, and gray-scale images. IEEE Multimedia, vol. 8, issue. 4, (**2001**), p. 22–28.
[10] Dumitrescu, S., Xiaolin Wu, Zhe Wang, Detection of LSB steganography via sample pair analysis. IEEE Transactions on Signal Processing, vol. 51, issue. 7, (**2003**), p. 1995–2007.
[11] Weiqi Luo, Fangjun Huang, Jiwu Huang, Edge Adaptive Image Steganography Based on LSB Matching Revisited. IEEE Transactions on Information Forensics and Security, vol. 5, issue. 2, (**2010**), p. 201–214.
[12] Harmsen, J.J., Pearlman, W.A., Steganalysis of additive-noise modelable information hiding. Proceeding of SPIE 5020. Santa Clara, CA, (**2003**), p. 131–142.
[13] Andrew D. Ker, Steganalysis of LSB matching in grayscale images. IEEE Signal Processing Letters, (**2005**), p. 441–444.
[14] Yang, C.-H., Weng, C.-Y., Wang, S.-J., Sun, H.-M., Adaptive Data Hiding in Edge Areas of Images With Spatial LSB Domain Systems. IEEE Transactions on Information Forensics and Security, vol. 3, issue. 3, (**2008**), p. 488–497.
[15] Wu, D.-C., Tsai, W.-H., A steganographic method for images by pixel-value differencing. Pattern Recognition Letters, **24** (9-10), (**2003**), p. 1613–1626.
[16] Zhang, X., Wang, S., Vulnerability of pixel-value differencing steganography to histogram analysis and modification for enhanced security. Pattern Recognition Letters, vol. 25, issue. 3, (**2004**), p. 331–339.
[17] Sabeti, V., Samavi, S., Shirani, S., An adaptive LSB matching steganography based on octonary complexity measure. Multimedia Tools and Applications, vol. 64, issue. 3, (**2013**), p. 777–793.
[18] Lou, D.-C., Wu, N.-I., Wang, C.-M., Lin, Z.-H., Tsai, C.-S., A novel adaptive steganography based on local complexity and human vision sensitivity. Journal of Systems and Software, vol. 83, issue. 7, (**2010**), p. 1236–1248.
[19] Wu, H.-C., Lee, C.-C., Tsai, C.-S., Chu, Y.-P., Chen, H.-R., A high capacity reversible data hiding scheme with edge prediction and difference expansion. Journal of Systems and Software, vol. 82, issue. 12, (**2009**), p. 1966–1973.
[20] Jorge Jasso, Gray Code - algorithm in Matlab bin/dec conversions. (**2009**).
[21] Rivest, R.L., Shamir, A., Adleman, L., A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM, vol. 21, issue. 2, (**1978**), p. 120–126.
[22] Nguyen, T.D., Arch-int, S., Arch-int, N., A novel secure block data-hiding algorithm using cellular automata to enhance the performance of JPEG steganography. Multimedia Tools and Applications, (**2014**).
[23] Cvejic, N., Seppänen, T., Increasing Robustness of LSB Audio Steganography by Reduced Distortion LSB Coding. JUCS - Journal of Universal Computer Science, (1), (**2005**).
[24] Samir Kumar Bandyopadhyay, Biswajita Datta, Higher lsb layer based audio steganography technique. International Journal on Electronics & Communication Technology, vol. 2, issue. 4, (**2011**).
[25] Tomáš Pevný, Tomáš Filler, Patrick Bas, Break Our Steganography System. Available: http://boss.gipsa-lab.grenoble-inp.fr/Warming/Materials/BOSSRank-covers.tar.bz2, (**2013**).
[26] Mielikainen, J., LSB matching revisited. IEEE Signal Processing Letters, vol. 13, issue. 5, (**2006**), p. 285–287.
[27] Pereira, S., Voloshynovskiy, S., Madueno, M., Marchand-Maillet, S., Pun, T., Second Generation Benchmarking and Application Oriented Evaluation. Proceedings of the 4th International Workshop on Information Hiding. London, UK, UK: Springer-Verlag, (**2001**), p. 340–353.
[28] Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P., Image Quality Assessment: From Error Measurement to Structural Similarity. IEEE TRANS. IMAGE PROCESSING, **13**, (**2004**), p. 600–612.

[29] Alfonso Muñoz, StegSecret. A simple steganalysis tool. Available: http://stegsecret.sourceforge.net/, (**2007**).
[30] Voloshynovskiy, S., Herrigel, A., Baumgaertner, N., Pun, T., A Stochastic Approach to Content Adaptive Digital Image Watermarking. Proceeding of Information Hiding (Editor: A. Pfitzmann). Berlin, Heidelberg: Springer Berlin Heidelberg, (**2000**), p. 211–236.
[31] Zhang, J., Cox, I.J., Doërr, G., Steganalysis for LSB Matching in Images with High-frequency Noise. Proceeding of IEEE 9th Workshop on Multimedia Signal 950 Processing (MMSP). (**2007**), p. 385–388.
[32] Chih-Chung Chang, Chih-Jen Lin, LIBSVM—A Library for Support Vector Machines.
[33] Kodovsky, J., Fridrich, J., Holub, V., Ensemble Classifiers for Steganalysis of Digital Media. IEEE Transactions on Information Forensics and Security, vol. 7, issue. 2, (**2012**), p. 432–444.
[34] Pevny, T., Bas, P., Fridrich, J., Steganalysis by Subtractive Pixel Adjacency Matrix. IEEE Transactions on Information Forensics and Security, vol. 5, issue. 2, (**2010**), p. 215–224.
[35] Kodovský, J., Fridrich, J., Quantitative steganalysis using rich models. (Editors: A. M. Alattar et al.), Proceeding of SPIE 8665, Burlingame, California, USA, (**2013**), p. 86650O.

Authors

Tuan Duc Nguyen, received the M. S degree from Le Qui Don Technical University, Vietnam, in 2008. He is currently a Ph.D student in Department of Computer Science, Faculty of Science, Khon Kaen University, Thailand. His research interests are cryptography, steganography and steganalysis. Contact him at nguyenductuan1982@gmail.com.

Somjit Arch-int, received the PhD degree in computer science from the Asian Institute of Technology (AIT) in 2002. He is currently an associate professor in the Department of Computer Science, Khon Kaen University, Thailand. His previous experiences include the development of several industry systems and consulting activities. His research interests are working on traceability system in a supply chain network, information integration, data mining, semantic web and security in the relevance problem domains. He is a member of IEEE Computer Society.

Ngamnij Arch-int, received the PhD degree in computer science from Chulalongkorn University, Thailand in 2003. She is currently an associate professor in the Department of Computer Science at Khon Kaen University, Thailand. Her research interests include the semantic web, web services, semantic web services, and heterogeneous information integration. Contact her at ngamnij@kku.ac.th