ITC 1/46

Journal of Information Technology and Control
Vol. 46 / No. 1 / 2017
pp. 16-36
DOI 10.5755/j01.itc.46.1.15253
© Kaunas University of Technology

A New Steganographic Algorithm Based on Multi Directional PVD and Modified LSB

Received 2016/10/16 | Accepted after revision 2016/02/17

http://dx.doi.org/10.5755/j01.itc.46.1.15253

# A New Steganographic Algorithm Based on Multi Directional PVD and Modified LSB

**Khalid A. Darabkh, Ahlam K. Al-Dhamari, and Iyad F. Jafar**

Department of Computer Engineering; The University of Jordan; Amman, 11942, Jordan; phone: +962-796969219; fax: +962-65300813; emails: k.darabkeh@ju.edu.jo, ahl_kal@daad-alumni.de, and iyad.jafar@ju.edu.jo

**Corresponding author:** k.darabkeh@ju.edu.jo

Steganographic techniques can be utilized to conceal data within digital images with small or invisible changes in the perceived appearance of the image. Generally, five main objectives are used to assess the performance of steganographic algorithms which include embedding capacity, imperceptibility, security, robustness and complexity. However, steganographic algorithms hardly take all of these factors into account. In this paper, a novel steganographic algorithm for digital images is proposed based on the pixel-value differencing (PVD) and modified least-significant bit (LSB) substitution (MDPVD-MLSB) techniques to address most of aforementioned objectives. Although there are many techniques for concealing data within pixels, the restricting factor is always the amount of bits adjusted in every pixel.  Therefore, the main contributions of this paper aim to achieve a balance between the amount of embedded data, the level of acceptable distortion, as well as providing high level of security. The performance of this algorithm has been extensively evaluated in terms of embedding capacity, peak signal-to-noise ratio (PSNR) and structural similarity index measure (SSIM). Simulation results and comparisons with six relevant algorithms are presented to demonstrate the effectiveness of this proposed algorithm.

**KEYWORDS:** Tri-way pixel-value differencing (TPVD), Octa pixel-value differencing (Octa-PVD), least-significant bit (LSB), optimal pixel adjustment process (OPAP), embedding capacity, imperceptibility.

## Introduction

Internet has become a vital part of everyone's lives, where anyone can send and receive data from any place in the world at any time. Wireless connection is getting popular since it is more convenient than wired Ethernet cables keeping in mind that wireless networks have numerous obstacles, among them low bandwidth, insecure links, and high error rate [1-9]. One of the primary issues of sending data over the Internet is security [10-11]. The ability to communicate securely is an important concern to organizations, people, and government. Therefore, it becomes very crucial to take data security into consideration as one of the most fundamental factors that requires attention during data transmission process. Generally, there are two popular approaches to secure data [12]. The first one is encryption, in which the original text is transformed into a cipher-text via certain algorithms. Thus, encryption changes the text into unreadable and incomprehensible text, which makes it suspicious enough to attract attackers' attention. Data Encryption Standard (DES), Advanced Encryption Standard (AES) and Rivest-Shamir-Adleman cryptosystem (RSA) are some of the common encryption methods [13]. The other approach is data hiding which is basically classified into two parts, namely, watermarking and steganography [14-17]. The main difference between them is that the former secures the carrier-object along with copyright information, while the later secures the embedded message into the carrier-object only [18-24]. Interestingly, there are other differences between the two methods based on the required performance objectives. For instance, watermarking is more sensitive to robustness than steganography. On the other hand, the steganography is more sensitive to capacity than watermarking. The security in watermarking lies in the difficulty of removing the watermark while in steganography, it lies in the difficulty of detecting/extracting the embedded data. Additionally, watermarking techniques are made between a sender and numerous receivers, whereas steganographic techniques are made between a sender and only one receiver [14].

This paper proposes an efficient and robust dynamic data hiding algorithm which mainly aims to improve the embedding capacity of the secret data, enhance the visual quality of stego-image, and make steganalysis a very hard task. The proposed algorithm is based on multi-directional pixel-value differencing (MDPVD) and modified least-significant bit (MLSB). However, the rest of the paper is organized as follows. In Section 2, a literature review of the most related works in the field is presented. Section 3 details the proposed algorithm. Experimental results and discussions are provided in Section 4. Section 5 concludes the work and provides future possible directions.

## Related work

Image steganographic algorithms, which have been discussed in literature, can be classified based on the embedding domain into two main classes: spatial domain and transform domain [14, 25-27]. The spatial domain algorithms are most frequently used because of their good concealment, great capability to hide information, and ease of realization [28]. In spatial domain methods, a steganographer modifies the pixel values of the host image directly [12]. LSB and PVD are the most common algorithms that essentially belong to this class [29-30]. In the transform domain algorithms, a steganographer embeds information into the coefficients of some transformed version of the host image [31]. The cover image can be converted into its transform domain by using one of the wavelet transformations such as: Discrete Cosine Transform (DCT), Discrete Wavelet Transform (DWT), and Fast Fourier Transform (FFT). Data insertion performed in the transform domain is greatly used for robust watermarking [25]. JSteg, F3, F4, F5, and Outguess methods are known steganographic algorithms that operate in the transform domain [25, 32].

The LSB substitution utilizes the least bits of a pixel value in the cover-image for embedding secret data. Many steganographic algorithms hide a large amount of secret data in the first least significant bits of the cover-image pixels. Because of the weak sensitivity of the human visual system (HVS), the existence of the hidden secret information can be imperceptible. The quality of the stego-image produced by simple LSB substitution may not be acceptable if a large amount of LSBs is used for data embedding. As an example, a stego-image can achieve a peak signal-to-noise ratio (PSNR) as low as 31.78 dB by using a simple LSB-4 replacement [29, 32-34]. The use of the optimal pixel adjustment process (OPAP) improves the perceptual quality of the stego-image when compared to using the simple LSB substitution method alone. The OPAP method was proposed in [29] and its idea has been described and utilized in many research papers [35-39].

It is worth stating that one of the most widespread approaches that become a base for a large amount of modern algorithms for secret data hiding is PVD, which can provide high embedding capacity and extremely good stego-image visual quality. For instance, the PVD method can achieve the capacity of 50,960 bytes (1.555 bpp) for Lena test image and its stego-image has a PSNR of 41.79 dB [34]. The main idea behind PVD is to use the difference between two consecutive pixels of a gray scale image to hide data. In [40], pixel-value differencing is used to distinguish between edge areas and smooth areas. Consequently, the capacity of embedded data in edge areas is higher than that of smooth areas. Wu and Tsai [40] proposed two types of quantization range table based on the range width of the power of two. The first range has the widths of {2, 2, 4, 4, 4, 8, 8, 32, 32, 64, 64} and is used to provide high imperceptibility or higher values of PSNR. The second range has the widths of {8, 8, 16, 32, 64, 128} and is used to provide high embedding capacity. It is worth mentioning that there are rare studies that aim to design new range tables. Tseng and Leng [41] proposed a PVD-based algorithm that mainly includes a new quantization range table based on the perfect square number, in order to obtain the secret data bits by using the difference value between consecutive pixels. They partition each range into two subranges for embedding variable number of secret data bits. After determining the perfect square number which belongs to the interval [1, 16], the ranges produced by this method are as follows {[0,1], [2,5], [6,11], [12,19], [20,29], [30,41], [42,55], [56,71], [72,89], [90,109], [110,131], [132,155], [156,181], [182,209], [210,239], [240,255]}.

In spite of PVD simplicity (i.e., being efficient in achieving large embedding capacity and extremely good stego-image visual quality) it has a limitation with respect to the capacity. In particular, the PVD does not make full use of edge areas. Motivated by the original PVD in [40], Chang et al [30] proposed a PVD-based scheme called tri-way pixel-value differencing (TPVD), which uses three different directional edges instead of one directional edge to eliminate the capacity limitation of the PVD method. Furthermore, the authors used the reference point selection and adaptive conditions in order to improve the quality of the stego- image. Based on TPVD, a data hiding algorithm was proposed in [42] in which three-directional PVD method for gray images is used. Unlike PVD and TPVD, the position of base pixel in this method is variable and depends on the pixel value of each group. Based on an index function, Jung and Yoo [43] proposed a high-capacity steganographic approach. In this method, the cover-image is partitioned into $B \times B$ sub-blocks and the base pixel is computed according to the index function. Rather than fixing the position of the base pixel, the position in this method depends on an index. The indexes must satisfy the condition: $0 \le x, y \le B-1$. After applying the index function to select the base pixel, the pixel values are sorted in ascending/descending order according to the results of the index function. Exploiting Modification Direction (EMD) method is devised in [44]. The importance of the EMD scheme lies in providing a good quality of stego-image with PSNR of more than 52 dB, since at most only one cover pixel in each pixel group needs to be incremented or decremented by one. Taking into account the EMD method, Shen and Huang [45] proposed a new scheme to enhance the embedding capacity and hide digits in any-ary notation adaptively by using the absolute difference value of pixel pairs.

Similar to TPVD, Thaneker and Pawar [46] proposed a new PVD scheme with eight directional edges, which is called Octa-PVD, to achieve a better capacity than that in PVD and TPVD. Unfortunately, it is noticed that even though this research used more directional edges, the image distortion increases dramatically. Therefore, the big challenge in PVD-based steganographic schemes, which have multi directional edges, is to achieve a balance between the amount of both the embedded data and the acceptable distortion, as well as provide high level of security.

In summary, an efficient and robust dynamic algorithm of data hiding is proposed in this paper. This algorithm aims to improve the embedding capacity of the secret data, enhance the visual quality of stego-image, and make steganalysis a very hard task. The proposed algorithm is based on multi-directional PVD and modified LSB. The idea of MLSB is to increase or decrease the pixel gray value after embedding, using simple LSB replacement by $2^k$ (*where k is the number of secret data bits that have to be embedded in LSBs of each pixel in a cover image*) in order to enhance the image quality [29, 42].

## The proposed algorithm

Since this study is based on MDPVD, the algorithm is divided into two separate schemes. These schemes are called (Quinary-PVD-MLSB) and (Octa-PVD-MLSB) because the cover-image is partitioned either into 2 × 3 pixel blocks, each of which has 5 pairs to embed secret data, or into 3 × 3 pixel blocks, each of which has 8 pairs to embed secret data. On the other hand, three branch conditions are proposed that permit automatic switching between MDPVD and MLSB (one of them can be selected only in the embedding and extracting procedures). In fact, these branch conditions can certainly reduce distortion caused by the offsetting of pairs process. If the selected branch condition is satisfied, then the current block can raise the distortion. Thus, this block must be embedded using MLSB to avoid any degradation in the visual quality of stego-image. The details about the proposed branch conditions are described in Section 3.1.

To protect the embedded secret data when using MLSB embedding, a secret key (*SK*) is used to insert secret data bits in different pixel indices according to the generated integer set $N_s$. $N_s$ is generated using the set-generation function $\mathrm{H_s}$ $(SK, N_{bp})$, where $N_{bp}$ is the number of block pixels, $N_s = \{N_{si} \mid i = 1, 2 \ldots N_{bp}\}$ and $SK \in [1, N_{bp}!]$ (i.e. $SK \in [1, 720]$ in Quinary-PVD-MLSB and $\mathrm{SK} \in [1,362880]$ in Octa-PVD-MLSB). In other words, based on $N_{bp}$, $\mathrm{H_s}$ set function will generate $(N_{bp}!)$ possible permutations and according to the selected *SK* value, $\mathrm{H_s}$ will generate the required $N_s$. Each element in the $N_s$ set is unique and its value falls within the range $[1, N_{bp}]$.

Generally, the proposed algorithm has three main phases: the partition phase, the embedding phase, and the extracting phase, which will be described in details in Sections 3.2, 3.3 and 3.4, respectively.

### Branch conditions to reduce distortion

Although both Quinary-PVD and Octa-PVD can hide large amount of secret data if they are utilized without using MLSB, embedding such large amount of data can clearly cause a very poor visual quality due to the offsetting of pairs process. Therefore, there is a need to determine some branch conditions that allow automatic switching between Quinary-PVD and MLSB or between Octa-PVD and MLSB. Three branch conditions are designed and only one of them can be used at a time according to the requirements of the practical applications such as the embedding capacity and the visual quality of the stego-image. These conditions are called $BC_1$, $BC_2$ and $BC_3$. They are as the following:

1 At least one of the difference values is equal or greater than 8 (i.e.$\exists\ D \geq 8$).

2 At least one of the difference values is equal or greater than 16 (i.e. $\exists D \geq 16$).

3 At least one of the difference values is equal or greater than 32 (i.e. $\exists D \geq 32$).

Where $\mathrm{D} = \{\mathrm{d_i} \mid \mathrm{i} = 0, \ldots, \mathrm{N_{bp}} - 2\}$ . If the selected branch condition from the above three conditions is satisfied, the current block results in higher distortion if MDPVD is used. Therefore, MLSB is used to individually embed each pixel in the block.

### Partition phase

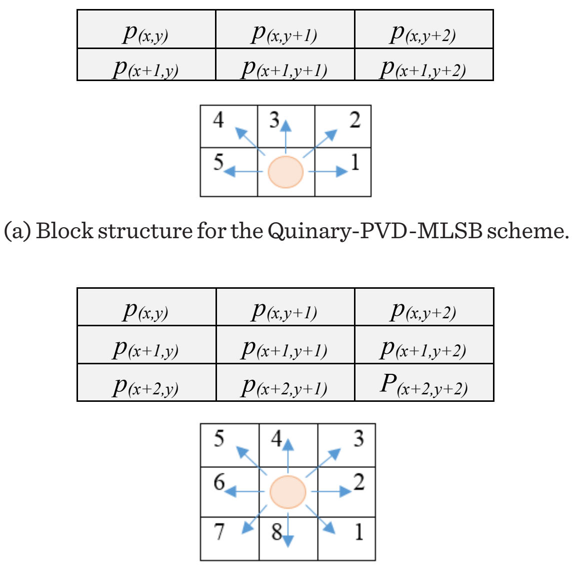
In the proposed algorithm, the cover-image (*CI*) is partitioned into 2×3 or 3×3 non-overlapping pixel blocks $B_i$, each of which has six or nine pixels in raster scan order or zig-zag scan order such that:

$$CI = \left\{ B_i \middle| i = 1, 2, \ldots, \frac{M \times N}{6} \right\} \text{ Or } CI = \left\{ B_i \middle| i = 1, 2, \ldots, \frac{M \times N}{9} \right\}. \quad (1)$$

Fig. 1 (a, b) shows the block structures for the two schemes of the proposed algorithm.

**Figure 1**

(a, b). Block structures in MDPVD-MLSB algorithm



(a) Block structure for the Quinary-PVD-MLSB scheme.

(b) Block structure for the Octa-PVD-MLSB scheme.

### Embedding phase

The embedding phase of our algorithm when the Octa-PVD-MLSB scheme is considered will be explained. As shown in Fig. 1. (b), each 3×3 block includes nine pixels $\{p_{(x,\ y)}, p_{(x,\ y+1)}, p_{(x,\ y+2)}, p_{(x+1,\ y)}, p_{(x+1,\ y+1)}, p_{(x+1,\ y+2)}, p_{(x+2,y)}, p_{(x+2,\ y+1)}, p_{(x+2,\ y+2)}\}$ where $x$ and $y$ are the pixel location in the cover image. Let $p_{(x+1,y+1)}$ be the beginning point, then eight pixel pairs can be formed by pairing the beginning point $p_{(x+1,y+1)}$ with each of its eight neighbors in the block. These eight pixel pairs are denoted by $\{P_0, P_1, P_2, P_3, P_4, P_5, P_6, P_7\}$. When using Octa-PVD to embed secret data bits, each pair has a new difference value $d_i$ $(0 \le i \le 7)$ and modified pixel pair $P_i'$ $(0 \le i \le 7)$. Consequently, each pair has new pixel values which are not the same as their original ones. That is, there are eight different values for the beginning point $p_{(x+1,y+1)}$. However, after finishing the Octa-PVD embedding process, only one value for the beginning point can be used. Thus, one of the eight pairs is selected as the optimal reference point to offset the other seven pixel values. Therefore, an approach to select the optimal reference point in the proposed algorithm is introduced. In addition, the shifting function is added to resolve the falling-off-boundary problem resulting from Octa-PVD embedding process. The details about both the optimal reference point selection approach and shifting function are left to be described later in subsections 3.3.1 and 3.3.2, respectively.

The embedding phase for the proposed algorithm uses the same equations that are employed in [29, 40, 46] with some modifications to suit the proposed algorithm. These modifications include the directions of pixel pairs. From the implementation of Octa-PVD, it is found that the best directions are those shown in Fig. 1 (b) in order to avoid excessive degradation in the stego-image. The details for embedding the secret data bits dynamically in each block $B_i$ that contains nine pixels from a cover image, are described in the following steps:

**Input:** A $W \times H$ grayscale cover-image *CI*, secret data *S*, range table *R* and secret key *SK*.

**Output:** A $W \times H$ stego-image *SI*.

**Step 1.** The secret data *S* is converted to form binary bitstream $S'$.

**Step 2.** As explained beforehand, $H_s$ $(SK, N_{bp})$ function is used to generate integer set $N_s$. According to the pixel indices in $N_s$, the secret data bits are embedded in block pixels. For instance, if it is assumed that $N_{bp}$ = 9 pixels, then all possible permutations for $N_s$ will be as shown in Table 1. Likewise, if it is assumed that *SK* = 40317, then $H_s$ (40317, 9) function will generate the following$N_s = [7,4,8,6,5,1,9,3,2]$. Therefore, the MLSB embedding procedure begins with the pixel that has the index of 7 and ends with the pixel that has the index of 2. The index of pixels is assumed as shown in Fig. 2.

**Table 1**
All possible permutations for $N_s$

| SK | Permutations |
|---|---|
| 1 | {1, 2, 3, 4, 5, 6, 7, 8, 9} |
| 2 | {2, 1, 3, 4, 5, 6, 7, 8, 9} |
| .<br>.<br>. | .<br>.<br>. |
| 40317 | {7, 4, 8, 6, 5, 1, 9, 3, 2} |
| .<br>.<br>. | .<br>.<br>. |
| 362880 | { 9, 8, 7, 6, 5, 4, 3, 2, 1} |

**Figure 2**
The index of pixels in block Bi of Octa-PVD-MLSB scheme

| | | |
|---|---|---|
| 1 | 2 | 3 |
| 4 | 5 | 6 |
| 7 | 8 | 9 |

**Step 3.** Select one of the branch conditions $BC_i$ $(1 \le i \le 3)$, which we described previously in section 3.1.

**Step 4.** Calculate the difference values for the eight pixel pairs $\{P_0, P_1, P_2, P_3, P_4, P_5, P_6, P_7\}$ where:

$$\begin{aligned}
P_0 &= (p_{(x+1,\ y+1)}, p_{(x+2,\ y+2)}),\\
P_1 &= (p_{(x+1,\ y+1)}, p_{(x+1,\ y+2)}),\\
P_2 &= (p_{(x+1,\ y+1)}, p_{(x,\ y+2)}),\\
P_3 &= (p_{(x+1,\ y+1)}, p_{(x,\ y+1)}),\\
P_4 &= (p_{(x+1,\ y+1)}, p_{(x,\ y)}),\\
P_5 &= (p_{(x+1,\ y+1)}, p_{(x+1,\ y)}),\\
P_6 &= (p_{(x+1,\ y+1)}, p_{(x+2,\ y)}),\\
P_7 &= (p_{(x+1,\ y+1)}, p_{(x+2,\ y+1)}).
\end{aligned} \tag{2}$$

Let the difference value be $d_i$ , where $0 \le i \le 7$ , then the difference values for all pixel pairs in the block $B_i$ can be calculated as follows:

$$\begin{aligned} d_0 &= p_{(x+2,\, y+2)} - p_{(x+1,\, y+1)}, \\ d_1 &= p_{(x+1,\, y+2)} - p_{(x+1,\, y+1)}, \\ d_2 &= p_{(x,\, y+2)} - p_{(x+1,\, y+1)}, \\ d_3 &= p_{(x,\, y+1)} - p_{(x+1,\, y+1)}, \\ d_4 &= p_{(x,\, y)} - p_{(x+1,\, y+1)}, \\ d_5 &= p_{(x+1,\, y)} - p_{(x+1,\, y+1)}, \\ d_6 &= p_{(x+2,\, y)} - p_{(x+1,\, y+1)}, \\ d_7 &= p_{(x+2,\, y+1)} - p_{(x+1,\, y+1)}. \end{aligned} \tag{3}$$

**Step 5.** The range table that is shown in Fig. 3 is used. It consists of six contiguous sub-ranges $R_j$ where ($j$ =1…6). In other words, $R = \{R_j = [l_i, u_i]\} = \{[0, 7], [8, 15], [16, 31], [32, 63], [64, 127], [128, 255]\}$. Thus, for a given difference value $|d_i|$, locate the suitable sub-range $R_j$ in the designed range table which has values from 0 to 255, where $l_i$ and $u_i$ are the lower and upper bounds, respectively.

**Step 6.** If the selected branch condition is satisfied, apply Steps from 7 to 10 to embed secret data using MLSB. Otherwise, go to Step 11 to embed secret data bits using MDPVD.

**Figure 3**

Range table proposed in [40]

| | |
|---|---|
| $R_1$ = [0, 7] | $t_1$ = 3 |
| $R_2$ = [8, 15] | $t_2$ = 3 |
| $R_3$ = [16, 31] | $t_3$ = 4 |
| $R_4$ = [32, 63] | $t_4$ = 5 |
| $R_5$ = [64, 127] | $t_5$ = 6 |
| $R_6$ = [128, 255] | $t_6$ = 7 |

**Step 7.** For each pixel $p_i$ in the block and using the indices of pixels in $N_s$, replace the k-rightmost LSBs of the pixel $p_i$ by the k-leftmost bits of the bitstream $S'$ ($k$ ′ {3, 4}) to obtain stego pixel $p_i'$. In addition, transform these $k$ bits-LSBs from $p_i$ and $k$ bits from the binary bitstream $S'$ into their decimal values (call them $LSB_i$ and $s_i$, respectively).

**Step 8.** Calculate the difference value $a_i$ between the values of $LSB_i$ and $s_i$ as follows:

$$a_i = LSB_i - s_i. \tag{4}$$

**Step 9.** Use the OPAP and modify the value of $p_i'$ with $p_i''$

$$p_i'' = \begin{cases} p_i' + 2^k, \text{ if } a_i > 2^{k-1} & \text{and } p_i' + 2^k \in [0,255] \\ p_i' - 2^k, \text{ if } a_i < -2^{k-1} & \text{and } p_i' - 2^k \in [0,255] \\ p_i', & \text{otherwise} \end{cases}. \tag{5}$$

**Step 10.** Move to the next block and return back to Step 4.

**Step 11.** For embedding secret data bits using Octa-PVD, use the obtained sub-ranges $R_j$ from Step 4 to calculate the number of embedding secret bits ($t_i$) for each pixel pair in the current block $B_i$. The number of embedding secret data bits can be calculated using the following equation:

$$t_i = \log_2(u_i - l_i + 1). \tag{6}$$

**Step 12.** For each pixel pair, read $t_i$ bits from the binary bitstream $S'$ and convert them into its decimal value $b_i$.

**Step 13.** Calculate eight new difference values $d_i'$ $(i = 0, \ldots, 7)$ to replace the eight original difference values $d_i$ $(i = 0, \ldots, 7)$

$$d_i' = \begin{cases} l_i + b_i, & \text{if } d_i \ge 0 \\ -(l_i + b_i), & \text{otherwise} \end{cases}. \tag{7}$$

**Step 14.** Compute $z_i$ as follows:

$$z_i = (d_i' - d_i)/2, \quad (i = 0, \ldots, 7). \tag{8}$$

**Step 15.** Calculate new pixel pair values ($p_i'$, $p_{i+1}'$) using

$$(p_i', p_{i+1}') = (p_i - \lceil z_i \rceil, p_{i+1} + \lfloor z_i \rfloor). \tag{9}$$

**Step 16.** Use the optimal reference point selection approach (ORPSA) that will be explained later to select the optimal reference point from the resulting eight pixel pairs and then use it to modify the other seven pixel pairs.

**Step 17.** If there is any falling-off-boundary cases happened, then use the shifting function and return back to Step 6 to re-embed this block using MLSB.

**Step 18.** Move to the next block and go to Step 4 until all secret data bits are embedded in the cover-image.

Notice that by using the shifting function, all blocks in the cover-image will be exploited to hide secret data bits, and thus giving higher embedding capacity. It is worth stating that the complexity time produced by our method is about O ($n^2$) which is less than or equal to 59s. The embedding process is illustrated in Fig. 4.

### Optimal Reference Point Selection Approach (ORPSA)

To achieve a minimum mean square error (MSE) and thus minimize stego-image distortion, the optimal reference point must be carefully obtained to adjust the other remaining gray values of the pixel pairs in each block. The authors of the TPVD algorithm [30] proposed optimal selection rules for obtaining the reference point according to $m_i$ values where $m_i = d_i - d_i'$. However, their rules for selecting the optimal reference point depend only on three values of '$m_i$' since there are three directional pixel pairs. Unfortunately, these rules are not applicable in this algorithm since there exist five or eight directional pixel pairs. Therefore, in this algorithm, a new function is proposed for selecting the optimal reference point for each block, depending on the calculations of the PSNR. Consequently, the optimal reference point can be obtained dynamically to ensure that the best selection is acquired. The overall computational complexities, induced by this method and TPVD method, are approximately the same.

### Shifting function

One of the deficiencies of PVD method is the falling-off-boundary problem. Therefore, a need to use a shifting function arises to modify one pixel value in the block $B_i$ to re-embed data in this block using MLSB method. Using shifting function in the proposed algorithm will somehow increase the embedding capacity. Moreover, this modification for one pixel value rarely affects the overall visual quality of the stego-image significantly. The following steps describe how the shifting function works considering the following variables:

- $Mp$ is the maximum value in block $B_i$.
- $Mip$ is the middle pixel in block $B_i$.
- $BC_i$ is the selected branch condition.

**Step 1.** Find the maximum pixel ($Mp$) among pixels in the block $B_i$ excluding the middle pixel $Mip$.

**Step 2.** Let $Omp = Mp$.

**Step 3.** Increase $Mp$ by 1 (i.e. $Mp = Mp + 1$). Then, compute $dif = Mp - Mip$.

**Step 4.** If only one difference value satisfies the selected $BC_i$ (i.e., if the selected branch condition is $BC_1$, that means the condition will be "if the $dif$ is equal or greater than 8"), then repeat Step 3 until this "if condition" is satisfied.

**Step 5.** If the $Mp$ is not in the range of 0-255 (i.e. $Mp$ <0 or 255<$Mp$), then make $Mp = Omp$. Otherwise, go to Step 8.

**Step 6.** Increment $Mp$ by 1 (i.e. $Mp = Mp - 1$). Then, compute $dif = Mp - Mip$.

**Step 7.** If $dif$ does not satisfy the selected $BC_i$, then repeat Step 6 until the "if condition" is satisfied.

**Step 8.** Modify the original value of the max pixel in the block $B_i$ with $Mp$.

### Embedding phase example

Assuming the bitstream of secret data is given as described in Fig. 5(a) and the sample sub-block for nine neighboring pixels is given as found in Fig. 5(b) where the sub-block has the gray-values of (192, 190, 192, 192, 193, 189, 192, 189, and 187). Consequently, the following are the outcomes (in sequence):

- The difference values are generated as shown in Fig. 5(c).
- Assuming branch condition 1 is used, all the difference values, in this sub-block, are less than 8. Thus, this block will be embedded using Octa-PVD where the optimal range for all pairs in this block is $R_1$ = [0, 7] as shown in Fig. 5(d). Consequently, the number of secret data bits can be obtained as follows: $t_i = \log_2 |7 - 0 + 1| = 3\,bits$, which will be embedded in each pixel pair.

**Figure 4**

The flowchart of the embedding procedure in MDPVD-MLSB algorithm

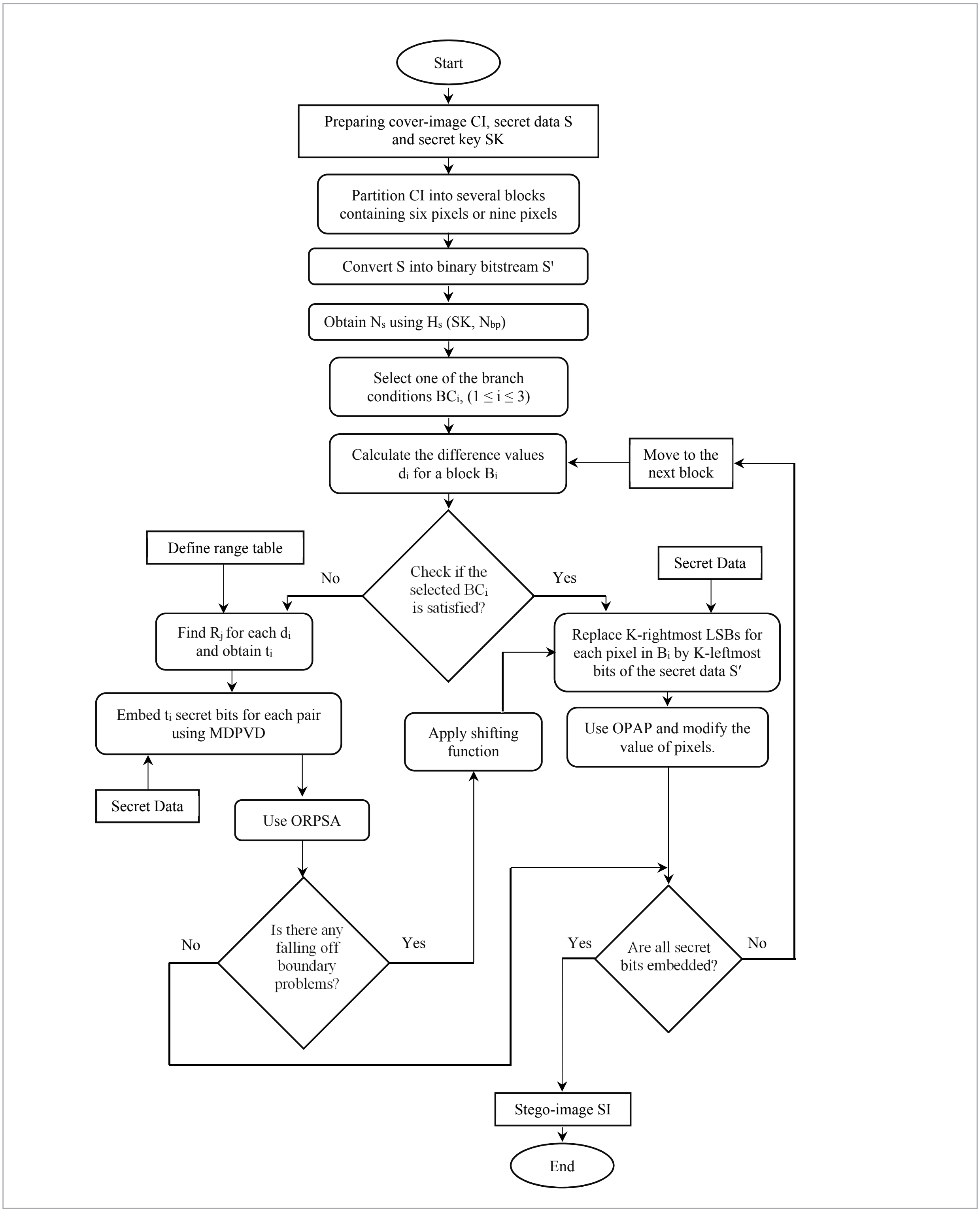
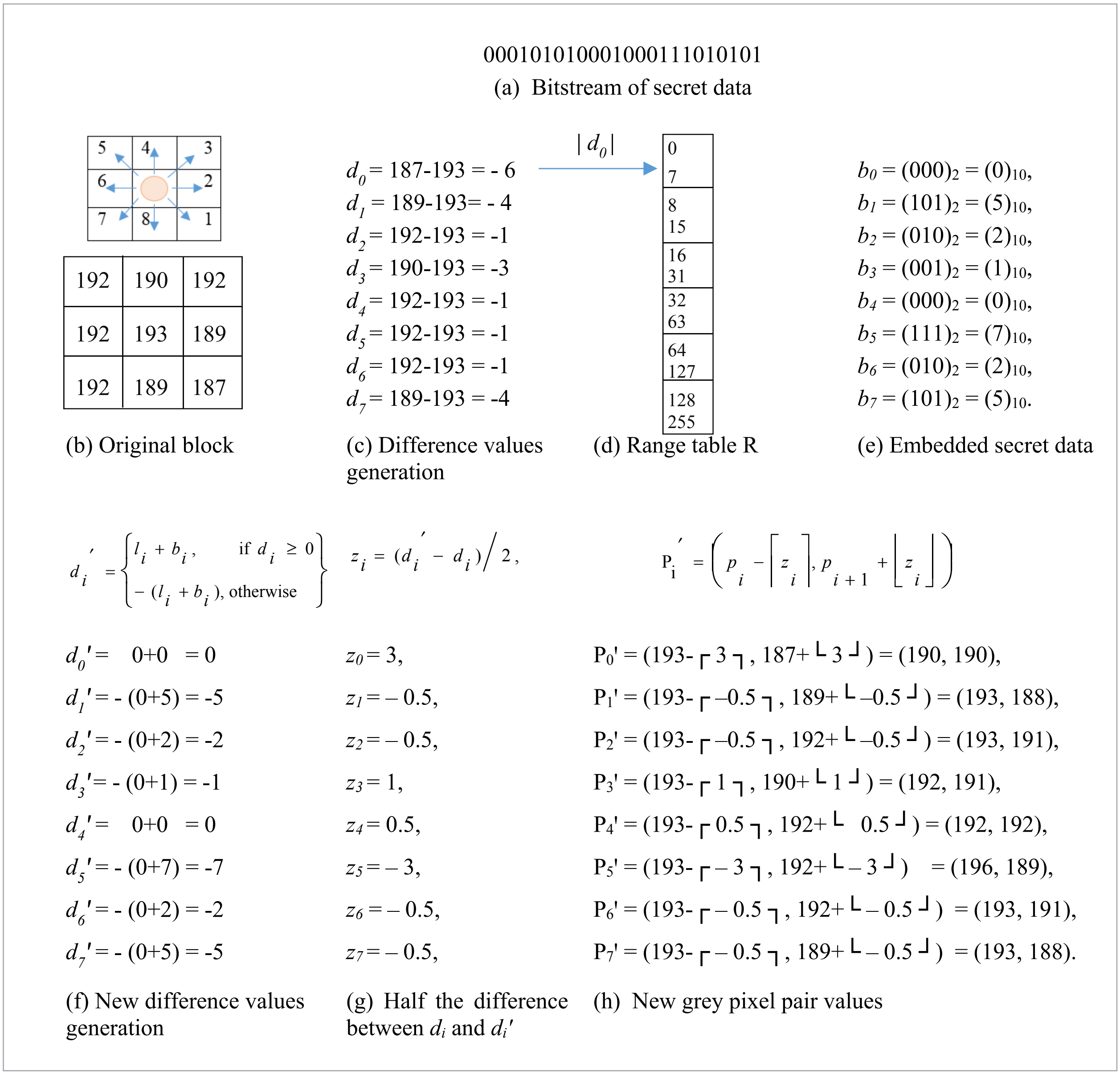

Start
Preparing cover-image CI, secret data S and secret key SK
Partition CI into several blocks containing six pixels or nine pixels
Convert S into binary bitstream S'
Obtain $N_s$ using $H_s$ (SK, $N_{bp}$)
Select one of the branch conditions $BC_i$, $(1 \le i \le 3)$
Calculate the difference values $d_i$ for a block $B_i$
Move to the next block
Define range table
No
Check if the selected $BC_i$ is satisfied?
Yes
Secret Data
Find $R_j$ for each $d_i$ and obtain $t_i$
Replace K-rightmost LSBs for each pixel in $B_i$ by K-leftmost bits of the secret data S′
Embed $t_i$ secret bits for each pair using MDPVD
Apply shifting function
Use OPAP and modify the value of pixels.
Secret Data
Use ORPSA
No
Is there any falling off boundary problems?
Yes
Yes
Are all secret bits embedded?
No
Stego-image SI
End

**Figure 5 (a-h)**

An example of the embedding process in our Octa-PVD-MLSB scheme when using the branch condition 1. Since all the difference values are less than 8, the Octa-PVD is used to embed secret data bits



000101010001000111010101

(a) Bitstream of secret data

| | | |
|---|---|---|
| 192 | 190 | 192 |
| 192 | 193 | 189 |
| 192 | 189 | 187 |

(b) Original block

$d_0 = 187\text{-}193 = -6$

$d_1 = 189\text{-}193 = -4$

$d_2 = 192\text{-}193 = -1$

$d_3 = 190\text{-}193 = -3$

$d_4 = 192\text{-}193 = -1$

$d_5 = 192\text{-}193 = -1$

$d_6 = 192\text{-}193 = -1$

$d_7 = 189\text{-}193 = -4$

(c) Difference values generation

$|d_0|$

| |
|---|
| 0<br>7 |
| 8<br>15 |
| 16<br>31 |
| 32<br>63 |
| 64<br>127 |
| 128<br>255 |

(d) Range table R

$b_0 = (000)_2 = (0)_{10}$,

$b_1 = (101)_2 = (5)_{10}$,

$b_2 = (010)_2 = (2)_{10}$,

$b_3 = (001)_2 = (1)_{10}$,

$b_4 = (000)_2 = (0)_{10}$,

$b_5 = (111)_2 = (7)_{10}$,

$b_6 = (010)_2 = (2)_{10}$,

$b_7 = (101)_2 = (5)_{10}$.

(e) Embedded secret data

$$d_i' = \begin{cases} l_i + b_i, & \text{if } d_i \geq 0 \\ -(l_i + b_i), & \text{otherwise} \end{cases}$$

$$z_i = (d_i' - d_i)/2,$$

$$P_i' = \left(p_i - \lceil z_i \rceil, p_{i+1} + \lfloor z_i \rfloor\right)$$

$d_0' = \ 0+0 \ = 0$

$d_1' = -(0+5) = -5$

$d_2' = -(0+2) = -2$

$d_3' = -(0+1) = -1$

$d_4' = \ 0+0 \ = 0$

$d_5' = -(0+7) = -7$

$d_6' = -(0+2) = -2$

$d_7' = -(0+5) = -5$

(f) New difference values generation

$z_0 = 3$,

$z_1 = -0.5$,

$z_2 = -0.5$,

$z_3 = 1$,

$z_4 = 0.5$,

$z_5 = -3$,

$z_6 = -0.5$,

$z_7 = -0.5$,

(g) Half the difference between $d_i$ and $d_i'$

$P_0' = (193 - \lceil 3 \rceil, 187 + \lfloor 3 \rfloor) = (190, 190)$,

$P_1' = (193 - \lceil -0.5 \rceil, 189 + \lfloor -0.5 \rfloor) = (193, 188)$,

$P_2' = (193 - \lceil -0.5 \rceil, 192 + \lfloor -0.5 \rfloor) = (193, 191)$,

$P_3' = (193 - \lceil 1 \rceil, 190 + \lfloor 1 \rfloor) = (192, 191)$,

$P_4' = (193 - \lceil 0.5 \rceil, 192 + \lfloor 0.5 \rfloor) = (192, 192)$,

$P_5' = (193 - \lceil -3 \rceil, 192 + \lfloor -3 \rfloor) = (196, 189)$,

$P_6' = (193 - \lceil -0.5 \rceil, 192 + \lfloor -0.5 \rfloor) = (193, 191)$,

$P_7' = (193 - \lceil -0.5 \rceil, 189 + \lfloor -0.5 \rfloor) = (193, 188)$.

(h) New grey pixel pair values

- Using the bitstream of secret data bits in Fig. 5(a), the embedded bits $b_i$ for each pixel pair will be as shown in Fig. 5(e).
- The new eight difference values $d_i'$ can be generated as shown in Fig. 5(f).
- $z_i$ values are computed as shown in Fig. 5(g).
- The new pixel pairs are calculated as shown in Fig.5 (h). Therefore, the new pixel pairs will be as follows: {(190, 190), (193, 188), (193, 191), (192, 191), (192, 192), (196, 189), (193, 191), (193, 188)}.
- Finally, the optimal pixel pair can be found by using the ORPSA. This procedure is demonstrated as follows:

1 When having the pixel pairs that are shown in Fig.5 (h), which are {$P_0'$ = (190,190), $P_1'$ = (193,188), $P_2'$ = (193,191), $P_3'$ = 192,191), $P_4'$ = (192,192), $P_5'$ = (196,189), $P_6'$ = (193,191), $P_7'$ = (193,188)}, then the first pixel pair denoted by $P_0'$ is the optimal reference point. Consequently, the offsetting process for the other seven pixel pairs will be as follows: $P_1'$ = (193+190-193, 188+190-193) = (190,185), $P_2'$= (193+190-193, 191+190-193) = (190,188), $P_3'$= (192+190-192, 191+190-192) = (190,189), $P_4'$ = (192+190-192, 192+190-192) = (190,190), $P_5'$= (196+190-196, 189+190-196) = (190,183), $P_6'$ = (193+190-193, 191+190-193) = (190,188), $P_7'$ = (193+190-193, 188+190-193) = (190,185). The resulting pixel pairs are shown in the modified sub-block ‘*Msb1’* that is shown in Fig. 6 (a). After that, the PSNR of *Msb1* is found.

2 Moving to the second pixel pair ‘$P_1'$’ and assuming it is the optimal reference point, the same procedure done in Step 1 is considered to offset the other remaining pixel pairs in order to obtain ‘$Msb_2$’ and calculate its PSNR value.

3 Repeating the same work performed in Steps 1 and 2 for the rest of pixel pairs.

4 Obtaining a total of eight modified sub-blocks $(msb_1,...,msb_8)$ and eight values of the PSNR, as shown in Fig. 6 (a-h). As depicted in Fig. 6 (a-h), the optimal reference pair may be one of the following pixel pairs: {2, 3, or 7} since all of them achieve the highest PSNR (i.e., PSNR =38.5884). For simplicity, the first one is proposed to be dynamically chosen. Thus, the final embedded block is the $Msb_2$, as illustrated in Fig. 6 (b).

## Extracting phase

To extract secret data bits correctly from each block $B_i$ in a stego-image *SI*, the following steps are considered:

**Input:** A $W \times H$ stego-image *SI,* range table *R* and secret key *SK*.

**Output:** The secret data *S*.

**Step 1.** Partition *SI* into 3×3 non-overlapping sub-blocks.

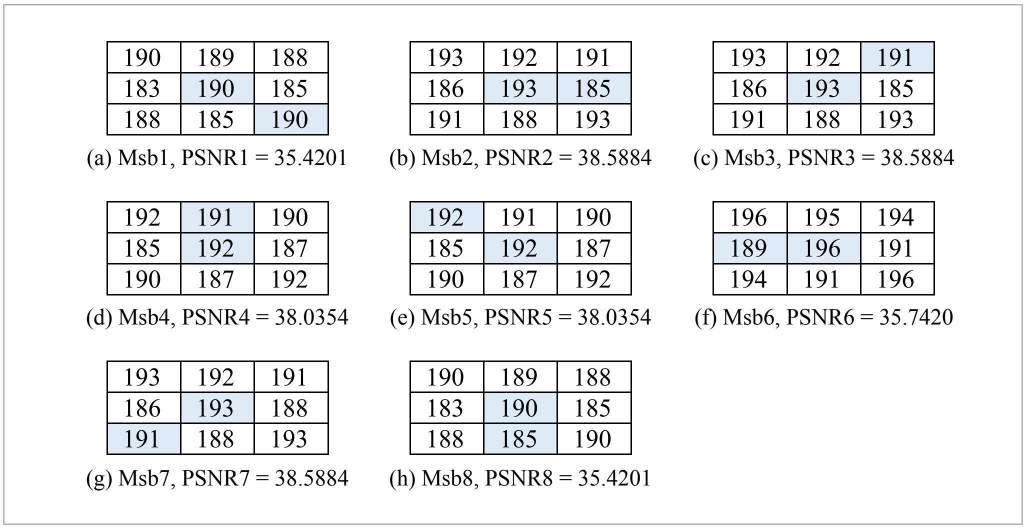
**Step 2.** As in the embedding phase, obtain $N_s$ according to $H_s$ $(SK, N_{bp})$.

**Step 3.** Choose the same branch condition that is selected by the embedding algorithm $BC_i$ $(1 \leq i \leq 3)$.

**Step 4.** Repeat Step 2 in the embedding phase to calculate the difference values $d_i'$ $(0 \leq i \leq 7)$ for eight pixel pairs.

**Step 5.** If the branch condition is satisfied, then extract this block by using MLSB. Otherwise, extract this block by using Octa-PVD and apply Steps from 8 to 10.

**Step 6.** Extract the secret bits $s_i$ from k-rightmost LSBs of each pixel $p_i'$ in this sub-block. As in the embedding process, the secret bits are extracted according to the pixel indices in $N_s$ set.



| | | |
|---|---|---|
| 190 | 189 | 188 |
| 183 | 190 | 185 |
| 188 | 185 | 190 |

(a) Msb1, PSNR1 = 35.4201

| | | |
|---|---|---|
| 193 | 192 | 191 |
| 186 | 193 | 185 |
| 191 | 188 | 193 |

(b) Msb2, PSNR2 = 38.5884

| | | |
|---|---|---|
| 193 | 192 | 191 |
| 186 | 193 | 185 |
| 191 | 188 | 193 |

(c) Msb3, PSNR3 = 38.5884

| | | |
|---|---|---|
| 192 | 191 | 190 |
| 185 | 192 | 187 |
| 190 | 187 | 192 |

(d) Msb4, PSNR4 = 38.0354

| | | |
|---|---|---|
| 192 | 191 | 190 |
| 185 | 192 | 187 |
| 190 | 187 | 192 |

(e) Msb5, PSNR5 = 38.0354

| | | |
|---|---|---|
| 196 | 195 | 194 |
| 189 | 196 | 191 |
| 194 | 191 | 196 |

(f) Msb6, PSNR6 = 35.7420

| | | |
|---|---|---|
| 193 | 192 | 191 |
| 186 | 193 | 188 |
| 191 | 188 | 193 |

(g) Msb7, PSNR7 = 38.5884

| | | |
|---|---|---|
| 190 | 189 | 188 |
| 183 | 190 | 185 |
| 188 | 185 | 190 |

(h) Msb8, PSNR8 = 35.4201

**Figure 6 (a-h)**

Obtaining the optimal reference point for example 1

**Step 7.** Move to the next block and then return back to Step 4.

**Step 8.** Find the value of lower bound $l_i$ by obtaining the optimal range $R_{i.}$

**Step 9.** Extract the embedding secret data bits from each pixel pair in a block using:

$$s_i = | d_i' | - l_i . \tag{10}$$

Move to the next block and then go to Step 4 until the recovery process of all the secret bits is finished. Then, concatenate all the secret bits $s_i$ in the same order as in the embedding process to obtain $S'$. Finally, convert $S'$ to form the original secret data $S$.

The extraction process is illustrated in the flowchart shown in Fig. 7.

It is noteworthy to mention that all of the side information used in this algorithm such as the secret key and the branch condition are offline, except the amount of the payload which occupies only 32 bits of the embedding capacity.

## Experimental results and discussion

In this section, the performance of the proposed algorithm is evaluated, discussed and justified.

Firstly, the simulation setup and its parameters are introduced. Secondly, a detailed description of the used metrics is presented. Lastly, the simulation results and comparisons are presented.

### Simulation setup

#### Simulation parameters

The proposed algorithm has been implemented and simulated using MATLAB 8.2.0.701 (R2013b) on Windows 7 platform with an Intel Core i7-4600U CPU working at 2.1 GHz with a 4 MB cache and 4 GB RAM. The results for different grayscale images are obtained for various random secret data. The various simulation parameters are as given in Table 2.

#### Performance metrics

The performance of the proposed algorithms was evaluated in terms of imperceptibility and embedding capacity which are defined as follows:

**Table 2**

Simulation parameters

| | |
|---|---|
| Cover image pixel size (N×N) | N = 512 |
| Image type | Tiff, jpg, bmp, gif |
| Simulation tool | MATLAB 8.2.0.701 |
| Secret data | Random strings using randseq ( ) |

- **Imperceptibility:** To measure the imperceptibility or the perceptual quality of a stego-image, the PSNR is used as well as the structural similarity index measure (SSIM) metrics. PSNR is the simplest and most widely used metric [48-51]. It can be found as follows [40, 47]:

$$PSNR = 10 \times \log_{10}\left(\frac{(255)^2}{MSE}\right), \tag{11}$$

where

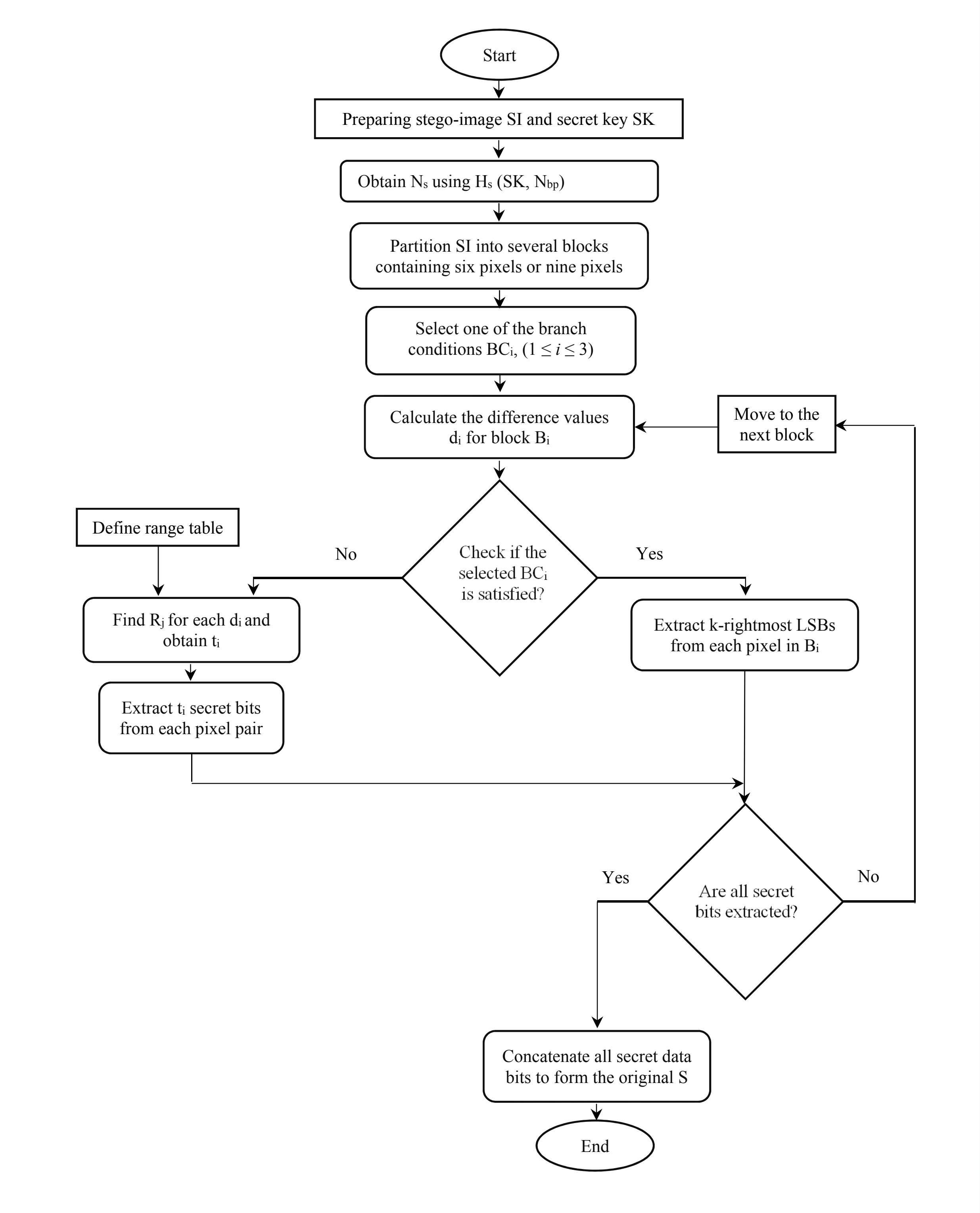
the value 255 refers to the maximum value of the pixel intensity for 8-bit grayscale images and MSE is the mean square error, which is used to compute the average square of the difference between the grey-scale cover image and its stego-image and is calculated using [52-56]:

$$MSE = \frac{1}{M \times N} \times \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} (x_{ij} - x_{ij}')^2 . \tag{12}$$

In this equation, "$M\times N$" represents the size of the cover-image and stego-image. In addition, $x_{ij}$ and $x_{ij}'$ represent the pixel values at certain index $(i, j)$ of the cover-image and stego-image, respectively. In general, to keep the grayscale cover-image and stego-image indiscernible to the human eye, a high value for the PSNR should be kept. Thus, a high PSNR means that the cover-image and stego-image are very similar to each other whereas a low PSNR means the opposite [57-58]. Besides PSNR, which is error-based metric, SSIM is very popular due to the fact that the human visual system is highly sensitive to structural information. In other words, any loss of structural data can give a good approximation for distortion of the per-

**Figure 7**

The flowchart of the extracting procedure in MDPVD-MLSB algorithm


Start
Preparing stego-image SI and secret key SK
Obtain $N_s$ using $H_s$ (SK, $N_{bp}$)
Partition SI into several blocks containing six pixels or nine pixels
Select one of the branch conditions $BC_i$, ($1 \leq i \leq 3$)
Calculate the difference values $d_i$ for block $B_i$
Move to the next block
Define range table
No
Check if the selected $BC_i$ is satisfied?
Yes
Find $R_j$ for each $d_i$ and obtain $t_i$
Extract $t_i$ secret bits from each pixel pair
Extract k-rightmost LSBs from each pixel in $B_i$
Yes
Are all secret bits extracted?
No
Concatenate all secret data bits to form the original S
End

ceived image. However, if the cover and stego images are defined as $x$ and $y$ respectively, then SSIM is computed using [59]:

$$SSIM(x,y)=\frac{(2\mu_x\mu_y+C_1)(2\sigma_{xy}+C_2)}{(\mu_x{}^2+\mu_y{}^2+C_1)(\sigma_x{}^2+\sigma_y{}^2+C_2)}, \quad (13)$$

where

$\mu_x$ and $\sigma_x{}^2$ refer to the mean and variance of $x$, respectively, $\mu_y$ and $\sigma_y{}^2$ refer to the mean and variance of $y$, respectively, $\sigma_{xy}$ denotes for the covariance of $x$ and $y$. $C_1=(k_1 L)^2$ and $C_2=(k_2 L)^2$ are two constants required to stabilize the division when the mean and variance get close to zero, where $k_1=0.01$, $k_2=0.03$, and $L=2^{Nbpp}-1$ (where $N_{bpp}$ is the number of bits per pixel), representing the maximum possible value of the image pixel. It is good to mention that the results of SSIM fall within the range of {0, 1} in which '1' indicates that the two images are completely identical, while '0' indicates that the two images are entirely different. However, for each image, there are several SSIM indices where each one is calculated within 11 × 11 local window using a certain circular-symmetric Gaussian weighting value (between 0 and 1) and the final SSIM image index is the average of these indexes.

- **Embedding Capacity (EC):** The embedding capacity is defined as the number of secret data bits that can be hidden in a cover-image. It is also known as embedding rate (ER), which represents the number of secret data bits that can be hidden per pixel, that is basically measured as the ratio of the total number of embedded secret bits to the total number of pixels in a cover-image. Actually, whenever this ratio exceeds one, it indicates that the steganographic embedding scheme has high embedding capacity [3, 19]. It can be calculated as follows:

$$EC = Total\ number\ of\ embedded\ secret\ bits. \quad (14)$$

$$ER=\frac{Total\ number\ of\ embedded\ secret\ bits}{Total\ number\ of\ \ pixels\ in\ CI}\ (bits/pixel). \quad (15)$$

$$ER=\frac{EC}{M\times N}(bpp). \quad (16)$$

## Experimental results

In our experiments, eight 8-bit 512x512 benchmark images, which are obtained from *USC-SIPI Image* database and *UWATERLOO-LINKS Image Repository* [60, 61], are used.

These grayscale images namely Lena, Peppers, Boat, Jet, Splash, Airplane, Baboon and Tiffany are shown in Fig. 8 (a-h). Data to be embedded is randomly generated. The range table used in the experiments is shown in Fig. 3. The maximum embedding capacity in this method can be calculated as follows:

1 Find the embedding capacity for MDPVD blocks:

$$EC_1=ASDB\times Nb_1, \quad (17)$$

where

*ASDB* is the average of the secret data bits per block and $Nb_1$ is the number of MDPVD blocks (either using Quinary-PVD or Octa-PVD). Moreover, *ASDB* can be calculated as follows:

$$ASDB=\frac{\sum\left(\sum SDB/Np\right)/\left(L/2\right)}{Np}, \quad (18)$$

where

*SDB* is the sum of all secret data bits in a block, *Np* is the number of pixels for each block and *L* is the length of all pixels that is used in embedding.

2 Find the capacity for MLSB blocks as:

$$EC_2=(k\times Np)\times Nb_2, \quad (19)$$

where

*k* may be 3 bits or 4 bits and $Nb_2$ is the number of MLSB blocks.

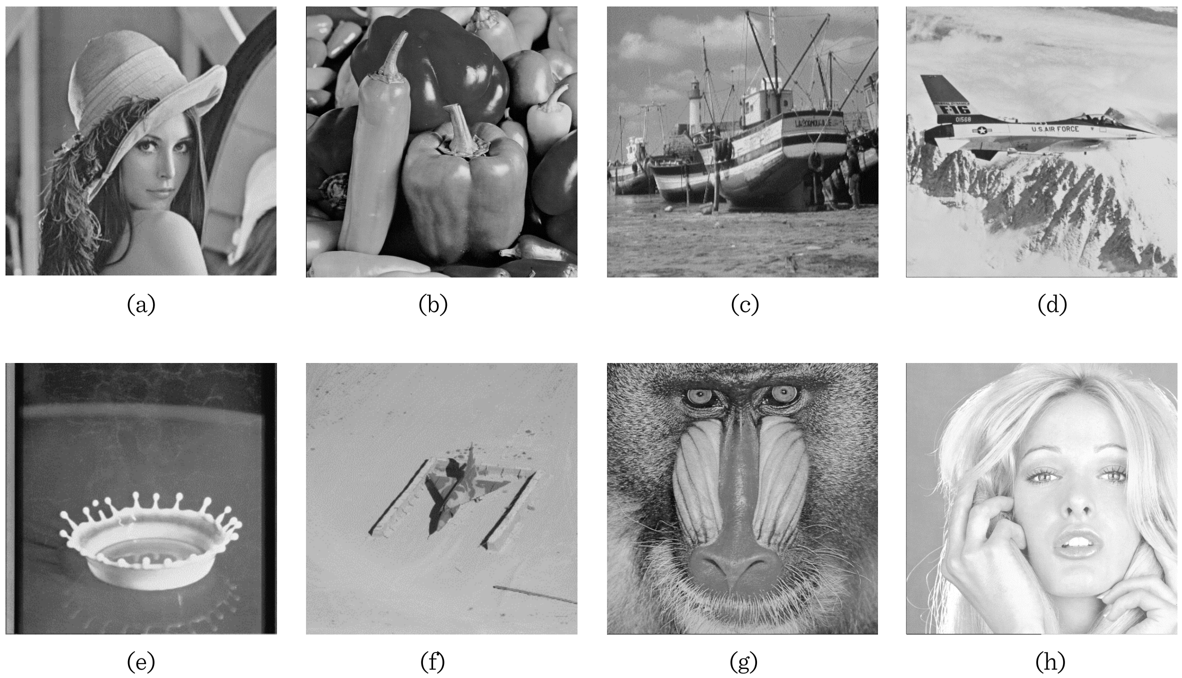
From (16) and (18), we can obtain the maximum embedding capacity (in bits) as follows:

$$EC=\lfloor EC_1+EC_2\rfloor. \quad (20)$$

Figs. 9 and 10 show the stego images produced by the Quinary-PVD-MLSB scheme for Lena and Peppers cover images when using $k$ = 3 bits, to be embedded utilizing the MLSB, and using all branch conditions introduced.

**Figure 8 (a-h)**

Test images used in the experiments (a) Lena, (b) Peppers, (c) Boat, (d) Jet, (e) Splash, (f) Airplane, (g) Baboon, (h) Tiffany

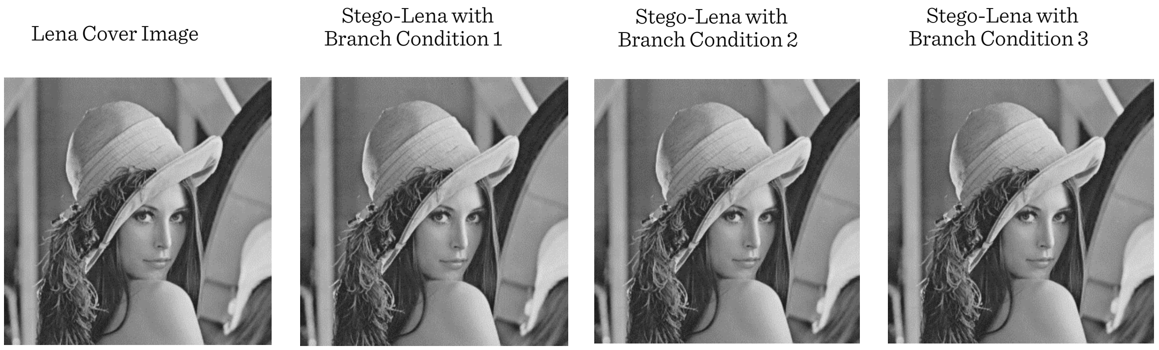


(a) (b) (c) (d)

(e) (f) (g) (h)

**Figure 9**

Results of our proposed Quinary-PVD-MLSB scheme on Stego-Lena with $k$ = 3 and all three branch conditions



| | Lena Cover Image | Stego-Lena with Branch Condition 1 | Stego-Lena with Branch Condition 2 | Stego-Lena with Branch Condition 3 |
|---|---|---|---|---|
| EC (Bits) | | 715,976 | 679,848 | 663,024 |
| SSIM | | 0.9950 | 0.9957 | 0.9955 |
| PSNR | | 40.1495 | 40.2623 | 39.7181 |

**Figure 10**

Results of our proposed Quinary-PVD-MLSB scheme on Stego-Peppers with $k$ = 3 and all three branch conditions

| Peppers Cover Image | Stego-Peppers with Branch Condition 1 | Stego-Peppers with Branch Condition 2 | Stego-Peppers with Branch Condition 3 |
|---|---|---|---|
| EC (Bits) | 727,208 | 679,976 | 783,328 |
| SSIM | 0.9946 | 0.9955 | 0.9951 |
| PSNR | 39.8849 | 39.8061 | 38.6543 |

As the figures show, there are no visual artifacts present between the cover images and their corresponding stego images. Similarly, in case of using Octa-PVD-MLSB scheme, the results are shown in Figs. 11 and 12.

When comparing the resulting stego images along with their corresponding cover images, it is obvious that all the stego images have an amazing visual quality for any human eye.

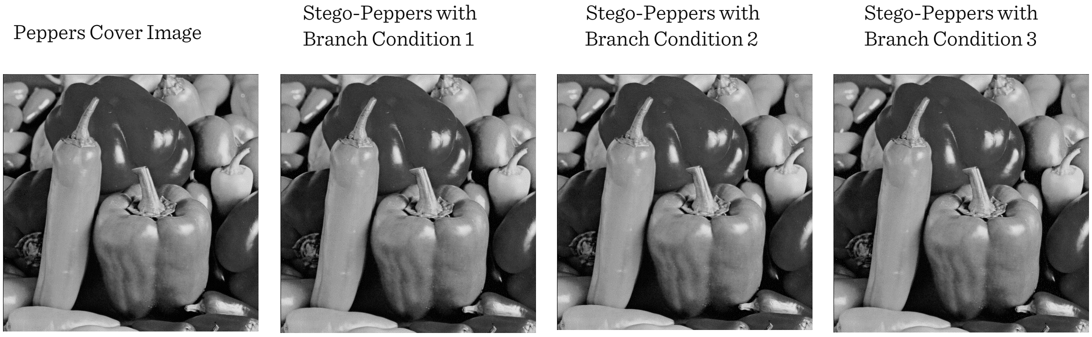
**Figure 11**

Results of our proposed Octa-PVD-MLSB scheme on Stego-Lena with $k$ = 3 and all three branch conditions

| Lena Cover Image | Stego-Lena with Branch Condition 1 | Stego-Lena with Branch Condition 2 | Stego-Lena with Branch Condition 3 |
|---|---|---|---|
| EC (Bits) | 741,464 | 715,056 | 702,304 |
| SSIM | 0.9942 | 0.9945 | 0.9943 |
| PSNR | 40.0316 | 39.9513 | 39.3894 |

**Figure 12**

Results of our proposed Octa-PVD-MLSB scheme on Stego-Peppers with $k$ = 3 and all three branch conditions


Peppers Cover Image
Stego-Peppers with Branch Condition 1
Stego-Peppers with Branch Condition 2
Stego-Peppers with Branch Condition 3

| | Stego-Peppers with Branch Condition 1 | Stego-Peppers with Branch Condition 2 | Stego-Peppers with Branch Condition 3 |
|---|---|---|---|
| EC (Bits) | 750,952 | 716,264 | 780,264 |
| SSIM | 0.9942 | 0.9945 | 0.9940 |
| PSNR | 39.8919 | 39.5898 | 38.5076 |

## Discussion

To improve the stego-image visual quality in the Octa-PVD method proposed in [46], the proposed approach is used to select the optimal reference point. In addition, the pair directions are changed by taking the directions for each pixel block as proposed in MDPV-MLSB algorithm, see Fig. 1 (b). However, Table 3 shows the results of our improved Octa-PVD and original Octa-PVD algorithms. The average improvement ratio (AIR) in regards to the PSNR is about 36% bearing in mind that the improvement ratio can rise up to 52%.

To have a comprehensive evaluation, our MDPVD-MLSB algorithm is further compared with the original PVD algorithm [40] and five different PVD-based steganographic algorithms proposed in [30,

**Table 3**

The embedding capacity and PSNR for our improved Octa-PVD method and the original Octa-PVD method proposed in [46] for eight cover images

| Cover-image | Octa-PVD [46] | | Our improved Octa-PVD | |
|---|---|---|---|---|
| | EC (Bits) | PSNR (dB) | EC (Bits) | PSNR (dB) |
| Lena | 706,872 | 26.4350 | 708,496 | 34.6617 |
| Peppers | 705,480 | 24.3073 | 709,272 | 33.1697 |
| Boat | 721,008 | 23.6600 | 721,688 | 32.5751 |
| Jet | 704,360 | 24.6409 | 713,336 | 34.1273 |
| Splash | 700,688 | 25.5480 | 701,408 | 36.0862 |
| Airplane | 698,056 | 28.9938 | 699,760 | 37.4130 |
| Baboon | 777,944 | 19.1521 | 787,664 | 29.1745 |
| Tiffany | 703,800 | 27.3457 | 705,264 | 34.0199 |
| Average | 714,776 | 25.0104 | 718,360 | 33.9034 |

42-43, 45-46]. Table 4 (a-b) shows the results of our algorithm and these six algorithms with respect to the maximum EC and PSNR. As shown in Table 4 (a), our proposed algorithm, the Octa-PVD-MLSB, provides higher values of EC than those obtained by PVD, TPVD, and Octa-PVD algorithms. The EC AIRs of our algorithm and these three methods are about 80%, 21% and 4%, respectively. Moreover, our algorithm provides higher PSNR than that achieved by TPVD and Octa-PVD. The PSNR AIRs are about 5% and 62%, respectively (keeping in mind that the improvement ratio can reach up to 15% and 108%, respectively). When comparing our algorithm with PVD in regards to PSNR, the average degradation ratio (ADR) of our algorithm is only about 4%. Table 4 (b) illustrates that our proposed algorithm outperforms those proposed in [42, 43, and 45]. Specifically, the EC AIRs are about 16%, 19% and 84%, respectively (the improvement ratio can reach up to 22%, 23% and 87%, respectively). In addition, our proposed algorithm achieves higher values of PSNR than those in methods [42] and [43]. In other words, the PSNR AIRs are about 9% and 36%, respectively (the improvement ratio can rise up to 16% and 54%, respectively).

**Table 4 (a-b)**

The performance efficiency of our MDPVD-MLSB method against PVD [40], TPVD [30], Octa-PVD [46], and methods in [42, 43, and 45]

| Cover image name | PVD [40] | | TPVD [30] | | Octa-PVD [46] | | Quinary-PVD-MLSB 3 LSBs, Branch condition 1 | | Octa-PVD-MLSB 3 LSBs, Branch condition 1 | |
|---|---|---|---|---|---|---|---|---|---|---|
| | EC (Bytes) | PSNR (dB) | EC (Bytes) | PSNR (dB) | EC (Bytes) | PSNR (dB) | EC (Bytes) | PSNR (dB) | EC (Bytes) | PSNR (dB) |
| *Panel a* | | | | | | | | | | |
| Lena | 50,960 | 42.12 | 75,836 | 38.84 | 88,359 | 26.44 | 89,497 | 40.15 | 92,683 | 40.03 |
| Peppers | 50,685 | 41.89 | 75,579 | 38.42 | 88,185 | 24.31 | 90,901 | 39.88 | 93,869 | 39.89 |
| Boat | 52,635 | 40.01 | 77,982 | 37.12 | 90,126 | 23.66 | 93,383 | 39.89 | 95,356 | 39.89 |
| Jet | 51,025 | 41.79 | 76,123 | 38.43 | 88,045 | 24.64 | 87,208 | 40.49 | 91,037 | 40.26 |
| Splash | 49,932 | 42.18 | 74,700 | 39.29 | 87,586 | 25.55 | 86,057 | 40.38 | 90,584 | 40.15 |
| Airplane | 49,738 | 43.48 | 74,372 | 40.38 | 87,257 | 28.99 | 84,017 | 40.74 | 88,782 | 40.46 |
| Baboon | 56,291 | 38.10 | 82,407 | 34.62 | 97,243 | 19.15 | 96,645 | 39.89 | 97,987 | 39.94 |
| Tiffany | 50,919 | 42.16 | 75,650 | 38.92 | 87,975 | 27.35 | 88,511 | 39.96 | 92,033 | 39.93 |
| Average | 51,523 | 41.47 | 76,581 | 38.25 | 89,347 | 25.01 | 89,527 | 40.17 | 92,791 | 40.07 |

| Cover image name | Method in [42] | | Method in [43] | | Method in [45] | | Quinary-PVD-MLSB 3 LSBs, Branch condition 1 | | Octa-PVD-MLSB 3 LSBs, Branch condition 1 | |
|---|---|---|---|---|---|---|---|---|---|---|
| | EC (Bytes) | PSNR (dB) | EC (Bytes) | PSNR (dB) | EC (Bytes) | PSNR (dB) | EC (Bytes) | PSNR (dB) | EC (Bytes) | PSNR (dB) |
| *Panel b* | | | | | | | | | | |
| Lena | 77,265 | 41.25 | 76,849 | 31.94 | 50,311 | 42.46 | 89,497 | 40.15 | 92,683 | 40.03 |
| Peppers | 76,938 | 37.62 | 76,424 | 30.42 | 50,136 | 42.68 | 90,901 | 39.88 | 93,869 | 39.89 |
| Boat | 80,839 | 36.33 | -- | -- | 51,097 | 41.61 | 93,383 | 39.89 | 95,356 | 39.89 |
| Airplane | 78,039 | 38.89 | 76,853 | 30.66 | -- | -- | 84,017 | 40.74 | 88,782 | 40.46 |
| Baboon | 92,382 | 34.33 | 85,777 | 25.96 | 55,434 | 38.88 | 96,645 | 39.89 | 97,987 | 39.94 |
| Average | 81,093 | 37.68 | 78,976 | 29.75 | 51,745 | 41.41 | 90,889 | 40.11 | 93,735 | 40.04 |

## Conclusions and future work

Although the TPVD and Octa PVD methods, which are based on multi-directional PVD, provide high capability of embedding data, these methods, especially Octa PVD method, may suffer from unacceptable distortion in image quality due to the offsetting process of the pixel pairs employed. Therefore, this article proposes an efficient and robust dynamic algorithm through taking the advantage of both multi-directional PVD and MLSB, namely, MDPVD-MLSB algorithm, which mainly intends to improve the capacity, maintain good visual quality, and provide high protection for embedded data. The performance of MDPVD-MLSB algorithm is comprehensively evaluated and compared with other six related algorithms where the experimental results demonstrate the strength and effectiveness of our proposed algorithm.

Many directions can be given for further enhancement to the proposed algorithm. The algorithm's framework can be extended to the RGB color images for enhancing the capability of embedding. Moreover, it can be a good addition to develop an approach that takes into account the hybrid domain. Additionally, there are future plans to develop PVD-based schemes for another media such as audios and videos.

## References

1. K. A. Darabkh, R. S. Aygün. TCP traffic control evaluation and reduction over wireless networks using parallel sequential decoding mechanism. EURASIP Journal on Wireless Communications and Networking, 2007, 1-16, Article ID 52492.
2. K. A. Darabkh. Queuing analysis and simulation of wireless access and end point systems using Fano decoding. Journal of Communications, 2010, 5(7), 551-561. https://doi.org/10.4304/jcm.5.7.551-561
3. K. A. Darabkh. Evaluation of channel adaptive access point system with Fano decoding. International Journal of Computer Mathematics, 2011, 88(5), 916-937. https://doi.org/10.1080/00207160.2010.485249
4. K. A. Darabkh. Fast and upper bounded Fano Decoding Algorithm: Queuing Analysis. Transactions on Emerging Telecommunications Technologies, 2015, DOI: 10.1002/ett.2929. https://doi.org/10.1002/ett.2929
5. K. A. Darabkh, B. N. Abu-Jaradeh, I. F. Jafar. Incorporating Automatic Repeat Request and Thresholds with Variable Complexity Decoding Algorithms over Wireless Networks: Queuing Analysis. IET Communications, 2011, 5(10), 1377-1393. https://doi.org/10.1049/iet-com.2010.0698
6. K. A. Darabkh, I. Jafar, G. Al Sukkar, G. Abandah, R. Al-Zubi. An Improved Queuing Model for Packet Retransmission Policy and Variable Latency Decoders. IET Communications, 2012, 6(18), 3315–3328. https://doi.org/10.1049/iet-com.2012.0410
7. K. A. Darabkh, I. F. Jafar, R. T. Al-Zubi, M. Hawa. An improved image least significant bit replacement method. In: Proceedings of the 37th IEEE International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), Opatija, Croatia, 2014, 1182-1186. https://doi.org/10.1109/mipro.2014.6859747
8. K. A. Darabkh, O. Alsukour. Novel protocols for improving the performance of ODMRP and EODMRP over mobile Ad hoc networks. International Journal of Distributed Sensor Networks, 2015, 2015, 1-18, Article ID 348967.
9. K. A. Darabkh, H. Ibeid, I. F. Jafar, R.T. Al-Zubi. A generic buffer occupancy expression for stop-and-wait hybrid automatic repeat request protocol over unstable channels. Telecommunication Systems, 2016, 63(2), 205–221. https://doi.org/10.1007/s11235-015-0115-5
10. I. Jafar, K. A. Darabkh, R. Saifan. SARDH: A novel sharpening-aware reversible data hiding algorithm. Journal of Visual Communication and Image Representation, 2016, 39, 239–252. https://doi.org/10.1016/j.jvcir.2016.06.002
11. I. Jafar, K. A. Darabkh, R. Al-Zubi, R. Saifan. an efficient reversible data hiding algorithm using two steganographic images. Signal Processing, 2016, 128, 98–109. https://doi.org/10.1016/j.sigpro.2016.03.023
12. M. S. Subhedar, V. H. Mankar. Current status and key issues in image steganography: A survey. Computer Science Review, 2014, 13, 95-113. https://doi.org/10.1016/j.cosrev.2014.09.001
13. W. Stallings. Cryptography and network security principles and practices. Practice Hall, 2010.
14. I. F. Jafar, K. A. Darabkh, R. T. Al-Zubi, R. Nam'neh. Efficient reversible data hiding using multiple predictors.

The Computer Journal, 2016, 59(3), 423-438. https://doi.org/10.1093/comjnl/bxv067

15. I. Jafar, S. Hiary, K. A. Darabkh. An improved reversible data hiding algorithm based on modification of prediction errors. In: Proceedings of 2014 6th International Conference on Digital Image Processing (ICDIP 2014), Athens, Greece, 2014.

16. K. A. Darabkh, I. F. Jafar, R. T. Al-Zubi, M. Hawa. A new image steganographic approach for secure communication based on LSB replacement method. Information Technology and Control, 2015, 44(3), 315–328. https://doi.org/10.5755/j01.itc.44.3.8949

17. K. A. Darabkh. Imperceptible and robust DWT-SVD-based digital audio watermarking algorithm. Journal of Software Engineering and Applications, 2014, 7, 859-871. https://doi.org/10.4236/jsea.2014.710077

18. R. Chandramouli, M. Kharrazi, N. Memon. Image steganography and steganalysis: concepts and practice. In: Proceedings of the 2004 2nd International Workshop, Book Chapter in Digital Watermarking, Series: Lecture Notes in Computer Science, Springer, Berlin-Heidelberg, Seoul, Korea, 2004, 35-49. https://doi.org/10.1007/978-3-540-24624-4_3

19. K. Rabah. Steganography-the art of hiding data. Information Technology Journal, 2004, 3(3), 245-269. https://doi.org/10.3923/itj.2004.245.269

20. N. Johnson, Z. Duric, and S. Jajodia. Information hiding: steganography and watermarking: attacks and countermeasures. Kluwer Academic Publishers. http://www.jjtc.com/Steganography/. Accessed on March 15, 2016.

21. A. Tiwari, S. Yadav, N. Mittal. A review on different image steganography techniques. International Journal of Engineering and Innovative Technology (IJEIT), 2014, 3(7), 121-124.

22. D. Yadav, M. Agrawal, A. Arora. Performance evaluation of LSB and LSD in steganography. In: Proceedings of the 2014 5th IEEE International Conference on Confluence the Next Generation Information Technology Summit (Confluence), Noida, India, 2014, 515-520. https://doi.org/10.1109/CONFLUENCE.2014.6949380

23. Y. Zhou, W. Ng. A study of influence between digital watermarking and steganography. In: Proceedings of the 2013 IEEE International Conference on Wavelet Analysis and Pattern Recognition (ICWAPR), Tianjin, China, 2013, 49-55. https://doi.org/10.1109/ICWAPR.2013.6599291

24. S. Saini, K. Brindha. Improved data embedding into images using histogram shifting. International Journal of Emerging Research in Management & Technology (IJERMT), 2014, 3(5), 83-86.

25. G. Sravanthi, S. Devi, S. Reddy. A spatial domain image steganography technique based on plane bit substitution method. Global Journal of Computer Science and Technology Graphics and Vision, 2012, 12(15), 1-8.

26. P. Singh, R. Chadha. A survey of digital watermarking techniques, applications and attacks. International Journal of Engineering and Innovative Technology (IJEIT), 2013, 2(9), 165-175.

27. M. Iranpour, M. Rahmati. An efficient steganographic framework based on dynamic blocking and genetic algorithm. Multimedia Tools and Applications, 2015, 74(24), 11429–11450. https://doi.org/10.1007/s11042-014-2237-2

28. N. Tiwari, M. Sandilya, M. Chawla. Spatial domain image steganography based on security and randomization. International Journal of Advanced Computer Science and Applications, 2014, 5(1), 156-159. https://doi.org/10.14569/IJACSA.2014.050121

29. C. Chan, L. Cheng. Hiding data in images by simple LSB substitution. Pattern recognition, 2004, 37(3), 469-474. https://doi.org/10.1016/j.patcog.2003.08.007

30. K. Chang, C. Chang, P. Huang, T. Tu. A novel image steganographic method using tri-way pixel-value differencing. Journal of multimedia, 2008, 3(2), 37-44. https://doi.org/10.4304/jmm.3.2.37-44

31. A. Kaur, R. Dhir, G. Sikka. A new image steganography based on first component alteration technique. International Journal of Computer Science and Information Security, 2009, 6(3), 53-56.

32. N. Provos, P. Honeyman. Hide and seek: an introduction to steganography. IEEE Security & Privacy, 2003, 1(3), 32-44. https://doi.org/10.1109/MSECP.2003.1203220

33. X. Li, B. Yang, D. Cheng, T. Zeng. A generalization of LSB matching. IEEE Signal Processing Letters, 2009, 16(2), 69-72. https://doi.org/10.1109/LSP.2008.2008947

34. N. Wu, K. Wu, C. Wang. Exploring pixel-value differencing and base decomposition for low distortion data embedding. Applied Soft Computing, 2012, 12(2), 942-960. https://doi.org/10.1016/j.asoc.2011.09.002

35. C. H. Yang, C. Y. Weng, S. J. Wang, H. M. Sun. Adaptive data hiding in edge areas of images with spatial LSB domain systems. IEEE Transactions on Information Forensics and Security, 2008, 3(3), 488-497. https://doi.org/10.1109/TIFS.2008.926097

36. M. Khodaei, K. Faez. New adaptive steganographic method using least-significant-bit substitution and pixel-value differencing. IET image processing, 2012, 6(6), 677-686. https://doi.org/10.1049/iet-ipr.2011.0059

37. W. Hong, T. S. Chen. A novel data embedding method using adaptive pixel pair matching. IEEE Transactions on Information Forensics and Security, 2012, 7(1), 176-184. https://doi.org/10.1109/TIFS.2011.2155062

38. S. Xu, S. Lai. An optimal least significant bit based image steganography algorithm. In: Proceedings of the 2014 ACM International Conference on Internet Multimedia Computing and Service, Xiamen, China, 2014. https://doi.org/10.1145/2632856.2632877

39. K. Gupta, R. Roy, S. Changder. A secure image steganography technique with moderately higher significant bit embedding. In: Proceedings of the 2014 IEEE International Conference on Computer Communication and Informatics (ICCCI), Coimbatore, India, 2014, 1-6. https://doi.org/10.1109/iccci.2014.6921726

40. D. Wu, W. Tsai. A steganographic method for images by pixel-value differencing. Pattern Recognition Letters, 2003, 24(9), 1613-1626. https://doi.org/10.1016/S0167-8655(02)00402-6

41. H. Tseng, H. Leng. A steganographic method based on pixel-value differencing and the perfect square number. Journal of Applied Mathematics, 2013, Article ID 189706. https://doi.org/10.1155/2013/189706

42. K. Jung, K. Yoo. Three-directional data hiding method for digital images. Cryptologia, 2014, 38(2), 178-191. https://doi.org/10.1080/01611194.2014.885817

43. K. Jung, K. Yoo. High-capacity index based data hiding method. Multimedia Tools and Applications, 2015, 74(6), 2179-2193. https://doi.org/10.1007/s11042-014-2081-4

44. X. Zhang, S. Wang. Efficient steganographic embedding by exploiting modification direction. IEEE Communications Letters, 2006, 10(11), 781-783. https://doi.org/10.1109/LCOMM.2006.060863

45. S. Shen, L. Huang. A data hiding scheme using pixel value differencing and improving exploiting modification directions. Computers and Security, 2014, 48, 131-141. https://doi.org/10.1016/j.cose.2014.07.008

46. S. Thanekar, S. Pawar. Octa (star) PVD: a different approach of image steganography. In: Proceedings of the 2013 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC), Enathi, India, 2013, 1-5. https://doi.org/10.1109/ICCIC.2013.6724139

47. P. Gupta, R. Roy, S. Changder. A secure image steganography technique with moderately higher significant bit embedding. In: Proceedings of the 2014 IEEE International Conference on Computer Communication and Informatics (ICCCI), Coimbatore, India, 2014, 1-6. https://doi.org/10.1109/iccci.2014.6921726

48. I. F. Jafar, R. A. AlNa'mneh, K. A. Darabkh. Efficient improvements on the BDND filtering algorithm for the removal of high-density impulse noise. IEEE Transactions on Image Processing, 2013, 22(3), 1223-1232. https://doi.org/10.1109/TIP.2012.2228496

49. I. Jafar, K. A. Darabkh, G. Al-Sukkar. A rule-based fuzzy inference system for adaptive image contrast enhancement. The Computer Journal, 2012, 55(9), 1041-1057. https://doi.org/10.1093/comjnl/bxr120

50. I. Jafar, K. A. Darabkh. Image contrast enhancement based on equalization of edge histograms. IAENG International Journal of Computer Science, 2011, 38(3), 192-204.

51. I. Jafar, K. A. Darabkh. A modified unsharp-masking technique for image contrast enhancement. In: Proceedings of the 8th IEEE/SSD'11 International Multi-Conference on Systems, Signals and Devices, Sousse, Tunisia, 2011, 1-6. https://doi.org/10.1109/SSD.2011.5767489

52. K. A. Darabkh, A. M. Awad, A. F. Khalifeh. New video discarding policies for improving UDP performance over wired/wireless networks. International Journal of Network Management, 2015, 25(3), 181-202. https://doi.org/10.1002/nem.1888

53. K. A. Darabkh, A. M. Awad, A. F. Khalifeh. Efficient PFD-based networking and buffering models for improving video quality over congested links. Wireless Personal Communications, 2014, 79(1), 293-320. https://doi.org/10.1007/s11277-014-1857-1

54. K. A. Darabkh, A. M. Awad, A. F. Khalifeh. Intelligent and selective video frames discarding policies for improving video quality over wired/wireless networks. In: Proceedings of the 2013 IEEE International Symposium on Multimedia (ISM 2013), Anaheim, California, USA, 2013, 297-300. https://doi.org/10.1109/ISM.2013.57

55. K. A. Darabkh, R. Aygun. Improving UDP performance using intermediate QoD-aware hop system for wired/wireless multimedia communication systems. International Journal of Network Management, 2011, 21(5), 432–454. https://doi.org/10.1002/nem.768

56. K. A. Darabkh, R. S. Aygun. Performance evaluation of sequential decoding system for UDP-based systems for wireless multimedia networks. In: Proceedings of 2006 International Conference on Wireless Networks (ICWN'06), Las Vegas, Nevada, 2006, 365-371.

57. A. Cheddad, J. Condell, K. Curran, P. Kevitt. Digital image steganography: survey and analysis of current methods. Signal Processing, 2010, 90(3), 727-752. https://doi.org/10.1016/j.sigpro.2009.08.010

58. K. A. Darabkh, W. Y. Albtoush, I. F. Jafar. Improved clustering algorithms for target tracking in wireless sensor

networks. The Journal of Supercomputing, 2016, DOI: 10.1007/s11227-016-1898-1. https://doi.org/10.1007/s11227-016-1898-1

59. Z. Wang, A. Bovik, H. Sheikh, P. Simoncelli. Image quality assessment: from error visibility to structural similarity. IEEE Transactions on Image Processing, 2004, 13(4), 600-612. https://doi.org/10.1109/TIP.2003.819861

60. USC-SIPI Image Database Website. http://sipi.usc.edu/database/database.php?volume=misc. Accessed on April 15, 2015.

61. UWATERLOO-LINKS Image Repository Website. http://links.uwaterloo.ca/Repository.html. Accessed on April 15, 2015.

## Summary / Santrauka

Steganographic techniques can be utilized to conceal data within digital images with small or invisible changes in the perceived appearance of the image. Generally, five main objectives are used to assess the performance of steganographic algorithms which include embedding capacity, imperceptibility, security, robustness and complexity. However, steganographic algorithms hardly take all of these factors into account. In this paper, a novel steganographic algorithm for digital images is proposed based on the pixel-value differencing (PVD) and modified least-significant bit (LSB) substitution (MDPVD-MLSB) techniques to address most of aforementioned objectives. Although there are many techniques for concealing data within pixels, the restricting factor is always the amount of bits adjusted in every pixel.  Therefore, the main contributions of this paper aim to achieve a balance between the amount of embedded data, the level of acceptable distortion, as well as providing high level of security. The performance of this algorithm has been extensively evaluated in terms of embedding capacity, peak signal-to-noise ratio (PSNR) and structural similarity index measure (SSIM). Simulation results and comparisons with six relevant algorithms are presented to demonstrate the effectiveness of this proposed algorithm.

Steganografinės technikos gali būti panaudotos paslėpti duomenims skaitmeniniuose atvaizduose, padarant mažų arba nepastebimų pokyčių atvaizdo vaizdinėje kokybėje. Įprastai steganografiniai algoritmai yra vertinami pagal penkis siekinius, kurie apima įterptinę talpą, nepastebimumą, saugumą, patikimumą ir sudėtingumą. Deja, steganografiniai algoritmai retai atsižvelgia į šiuos faktorius. Atsižvelgiant į minėtus siekinius, šis straipsnis siūlo naujovišką steganografinį algoritmą, paremtą pikselių vertės diferencijavimu (angl. Pixel-value differencing (PVD)) ir modifikuotomis mažiausiai reikšmingo bito (angl. Least significant bit (LSB)) pakeitimo (MDPVD-MLSB) technikomis. Nors yra daug technikų duomenų paslėpimui pikseliuose, ribojantysis faktorius visuomet yra kiekviename pikselyje pakeistų pikselių kiekis. Šiuo straipsniu siekiama užtikrinti balansą tarp įtvirtintųjų duomenų ir priimtino atvaizdo iškraipymo, kartu garantuojant aukštą saugumo lygį. Algoritmo efektyvumas išsamiai įvertintas atsižvelgiant į įtvirtintuosius gebėjimus, signalo maksimumo taškų santykį su triukšmu (angl. Peak Signal-to-Noise Ratio (PSNR)) ir struktūrinį panašumo indekso parametrą (angl. Structural Similarity Index Measure (SSIM)). Kad būtų parodytas siūlomo algoritmo efektyvumas, pateikiami simuliacijos rezultatai ir palyginimai su šešiais aktualiais algoritmais.