IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 19, 2021, accepted December 1, 2021, date of publication December 27, 2021,
date of current version January 25, 2022.

Digital Object Identifier 10.1109/ACCESS.2021.3138895

Modified Multiway Pixel-Value Differencing Methods Based on General Quantization Ranges for Image Steganography

DA-CHUN WU, ZONG-NAN SHIH, AND JHENG-HAN WU
Department of Computer and Communication Engineering, National Kaohsiung University of Science and Technology, Kaohsiung 824005, Taiwan
Corresponding author: Da-Chun Wu (dcwu@nkust.edu.tw)

ABSTRACT A modified multiway pixel-value differencing method for image steganography using general quantization ranges of pixel pairs' difference values is proposed, where the widths of the quantization ranges are not limited to be powers of two. The method can be employed to embed a secret message into a cover image with a higher embedding rate. The cover image is partitioned into non-overlapping complete or incomplete blocks using 1×3, 2×2, 2×3, or 3×3 block templates. The message bitstream, whose bit positions are randomized in advance by use of a random number generator, can be embedded in the resulting blocks. More specifically, one pixel in each block is set as a shared pixel, and the least-significant-bit substitution method with the optimal pixel adjustment process is employed to embed message bits into the shared pixel. Next, the shared pixel in the block is combined with each remaining pixel to form a pixel pair, which is then utilized to embed message bits by a new pixel-value differencing method. Also, the use of non-power-of-two range widths is accomplished by a multiple-based number conversion mechanism, and the shared pixel in each block, as well as each related pixel pair whose two grayscale values pass a falling-off-boundary check, are used to embed the digits of the resulting multiple-based number. The experimental results show that the embedding rates yielded by the proposed method are higher than those yielded by existing multiway pixel-value differencing methods. The stego-images resulting from embedding secret messages retain good image quality, and the RS steganalysis process cannot detect the presence of the embedded secret messages.

INDEX TERMS General quantization ranges, multiway pixel-value differencing, steganography, data hiding, multiple-based number conversion.

I. INTRODUCTION

Steganography [1]–[4] and *watermarking* [5], [6] techniques are important research topics in the field of multimedia security. Watermarking is the technique of embedding copyright-related or authentication information into digital media to protect the copyright or verify the authenticity of the digital media. Steganography is the technique of concealing a secret message in a cover object in an undetectable manner to hide the existence of the message. Steganography should have the three characteristics of imperceptibility, high embedding capacity, and security. In the field of steganography, the *least-significant-bit* (*LSB*) *substitution* method [7] and the *pixel-value differencing* (*PVD*) method [8] are both of the type of embedding messages in the spatial domains of images. The LSB substitution method [7] aims at embedding messages by replacing the LSBs [1] of image pixels. Chan and Cheng [9] proposed an LSB substitution method with an optimal pixel adjustment process (OPAP) to reduce pixel value changes to improve the resulting image quality. The PVD method [8] is based on the utilization of the characteristics of human vision, where a quantization range table is designed for adjusting the pixel-value differences of 1×2 blocks, with fewer bits of information embedded in the smooth blocks and more bits in the rough blocks. Wu *et al.* [10] used a *hybrid* technique of using the PVD and LSB substitution methods to increase the embedding capacity of the images. Wang *et al.* [11] used the PVD method and the modulus function to reduce the distortions of the images after embedding secret messages. Chang *et al.* [12] partitioned images into

The associate editor coordinating the review of this manuscript and approving it for publication was Muhammad Imran Tariq.

2×2 non-overlapping blocks and used a *tri-way* PVD technique to increase the embedding capacity. Shukla *et al.* [13] used an arithmetic encoding technique to increase the embedding capacity of the PVD method. Khodaei and Faez [14] and Swain [15] enlarged the 1×2 block size used in the PVD method [8] and used a *multiway* PVD technique with the LSB substitution method together with the OPAP [9] to embed messages. Specifically, the central pixel of the block is first used to embed secret messages by the LSB substitution method with the OPAP [9]. Then, the central pixel and the surrounding pixels form multiple pixel pairs, which are employed to embed secret messages by PVD using a *single-sided* pixel-value adjustment technique. In the existing PVD methods [8], [10]–[15], the quantization range tables all consist of quantization ranges with widths of *powers of two* to facilitate direct embedding of bit data. Wu [16] proposed a *general* pixel-value differencing (GPVD) method based on the PVD method [8] but with the used quantization range widths not limited to be powers of two, and adopted the mechanism of multiple-based number conversion [17]–[21] to embed secret messages into the pixel pairs in images.

In this study, a modified multiway general pixel-value differencing (MMGPVD) method for image steganography with the widths of the quantization ranges not limited to be powers of two is proposed. The method is based on the multiway pixel-value differencing methods proposed in [14], [15], which allows the use of 1×3, 2×2, 2×3, and 3×3 block templates to partition images into non-overlapping blocks. Secret messages can be embedded into each partitioned block no matter whether the block is complete or not. The proposed method yields higher embedding rates than multiway PVD methods [14], [15], and maintains good stego-image quality. A random numbering scheme is used in the method to randomize the bit positions of secret messages *before* embedding them to prevent hackers from knowing about the contents of the hidden messages. Finally, it is demonstrated that it is impossible to use the RS steganalysis method [22] to detect the presence of the secret messages embedded in the stego-images yielded by the proposed method.

The remainder of this paper is organized as follows. In Section 2, relevant techniques used in this paper are reviewed briefly. The process of the proposed MMGPVD method is described in Section 3. Experimental results and comparisons of them with those yielded by some existing methods are illustrated in Section 4. Finally, some concluding remarks and discussions are stated in Section 5.

II. RELATED TECHNIQUES

In this section, information hiding techniques related to the proposed method are reviewed, including the PVD method [8], the multiway PVD methods [14], [15], and the GPVD method [16].

A. THE PVD METHOD

The PVD data hiding method proposed by Wu and Tsai in 2003 [8] is reviewed at first here. The method begins with partitioning a given grayscale cover image into 1×2 non-overlapping blocks, each block containing a pair of neighboring pixels. Let g_1 and g_2 be the pixel values of the two pixels in a pixel pair P in a block. Denote the *pixel-value difference* of P as δ, i.e.,

$$\delta = g_2 - g_1. \tag{1}$$

By dividing the grayscale range [0, 255] into six *quantization ranges* as shown in Table 1, the number of embeddable bits in P is decided by the width of the quantization range to which the absolute value $|\delta|$ of δ belongs. Denote the number of bits embeddable in P with $|\delta|$ in the k-th quantization range $[l_k, u_k]$ as n_k where $k = 1, 2, \ldots, 6$. Then, n_k is computed as

$$n_k = \lfloor \log_2 (u_k - l_k + 1) \rfloor, \tag{2}$$

where $\lfloor \cdot \rfloor$ is the floor function. It can be figured out that the smaller the width of the quantization range is, the fewer the bits embeddable into the pixel pair P, and vice versa, according to Table 1. Furthermore, let M be a message bitstream with n_k bits, which may be converted to be a decimal value b. Then, embedding of M into P is conducted by the following two steps.

(1) Compute a new pixel-value difference δ' for P as

$$\delta' = \begin{cases} l_k + b & \text{if } \delta \geq 0; \\ -(l_k + b) & \text{if } \delta < 0. \end{cases} \tag{3}$$

(2) Adjust the original pixel values g_1 and g_2 of P to be g_1' and g_2', respectively, in the following way:

$$\begin{aligned} &(g_1', g_2') \\ &= \begin{cases} (g_1 - \left\lceil \frac{\delta' - \delta}{2} \right\rceil, g_2 + \left\lfloor \frac{\delta' - \delta}{2} \right\rfloor) \\ \qquad \text{if } \delta \bmod 2 \neq 0; \\ (g_1 - \left\lfloor \frac{\delta' - \delta}{2} \right\rfloor, g_2 + \left\lceil \frac{\delta' - \delta}{2} \right\rceil) \\ \qquad \text{if } \delta \bmod 2 = 0, \end{cases} \end{aligned} \tag{4}$$

where the operation of mod yields the remainder of integer division, and $\lceil \cdot \rceil$ and $\lfloor \cdot \rfloor$ are the ceiling and floor functions, respectively.

Because the adjusted pixel values g_1' and g_2' may be out of the pixel-value range [0, 255], creating black and white sesame seed-like noise in the resulting image, the method [8] performs a *falling-off-boundary check* on the pixel values g_1 and g_2 in P *before* embedding the data to determine whether message bits can be embedded into P without causing such noise. The checking scheme is mainly to *simulate* the embedding by using the maximum value of the quantization range corresponding to $|\delta|$ (i.e., the upper bound value u_k of the range) to replace $l_k + b$ mentioned in Eq. (3) in the following way:

$$\delta' = \begin{cases} u_k & \text{if } \delta \geq 0; \\ -u_k & \text{if } \delta < 0. \end{cases} \tag{5}$$

The new pixel values $\hat{g}_1$ and $\hat{g}_2$, after the simulated embedding process is completed, are computed according to Eqs. (4)

TABLE 1. The quantization range table for the pixel-value differencing method proposed by Wu and Tsai [8].

Index	1	2	3	4	5	6
Range	[0, 7]	[8, 15]	[16, 31]	[32, 63]	[64, 127]	[128, 255]
Width	8	8	16	32	64	128
No. of embedded bits	3	3	4	5	6	7

and (5) above as well. Subsequently, to determine if g_1 and g_2 are likely to cause pixel values to be out of bounds, the following examination is conducted:

$$fall\_off(g_1, g_2) = \begin{cases} false & \text{if } 0 \le \hat{g}_1 \le 255 \text{ and } 0 \le \hat{g}_2 \le 255; \\ true & \text{otherwise.} \end{cases} \quad (6)$$

If neither $\hat{g}_1$ nor $\hat{g}_2$ is out of bounds, the value of the function $fall\_off(g_1, g_2)$ will be *false*. In this case, b can be embedded into the block by Eqs. (3) and (4); or, in other words, any message is *embeddable* into the pixel pair P. On the contrary, if the value of $fall\_off(g_1, g_2)$ is *true*, it means that data embedded in g_1 or g_2 might exceed the boundary of 0 or 255, and no data are embedded in P in this case by the method [8].

When extracting an embedded secret message, at first the stego-image is partitioned into the same 1×2 non-overlapping blocks as those obtained in the message embedding process with each block being a pixel pair. Let g_1^* and g_2^* be the values of two pixels in a pixel pair P. The pixel-value difference of the two pixels is denoted as δ^* so that

$$\delta^* = g_2^* - g_1^*. \quad (7)$$

The absolute value of δ^* is then used to determine the corresponding quantization range R of P in Table 1, and the pixel values g_1^* and g_2^* of the two pixels in P are taken to go through the falling-off-boundary check in the same way described by Eqs. (4), (5) and (6) as done in the embedding process to determine whether g_1^* or g_2^* is likely to be out of bounds when data embedding is performed. If it is judged that no possibility of out-of-bounds will result, it means that P has previously been used to embed a secret message b^*, which can then be extracted from P to be

$$b^* = |\delta^*| - l_k \quad (8)$$

where l_k is the lower bound value of the quantization range R.

B. THE MULTIWAY PVD METHODS

Khodaei and Faez [14] extended the block size used in the PVD method [8] from 1×2 to 1×3. Table 2 shows the quantization range table used in the method. The central pixel in each block is chosen as a *shared pixel* denoted as p_s, and the k-bit LSB substitution method [7] is applied to the pixel value g of p_s to obtain a new pixel value g'. The difference value δ between g and g' is calculated as

$$\delta = g - g',$$

and the OPAP [9] is performed on g' to obtain an adjusted value g'' of p_s as described in the following:

$$g'' = \begin{cases} g' + 2^k & \text{if } \delta > 2^{k-1} \text{ and } 0 \le g' + 2^k \le 255; \\ g' - 2^k & \text{if } \delta < -2^{k-1} \text{ and } 0 \le g' - 2^k \le 255; \\ g' & \text{otherwise.} \end{cases} \quad (9)$$

Subsequently, the shared pixel p_s is combined with its left and right pixels, respectively, to form two pixel pairs, and message bits are embedded into each pixel pair P_i by keeping the pixel value g'' of p_s unchanged while modifying the pixel value g_i of the other pixel p_i in P_i by the following steps.

(1) Compute the absolute difference value δ_i of p_s and p_i as
$$\delta_i = |g_i - g''|.$$
(2) Let the quantization range to which the value δ_i belongs be $[l_k, u_k]$ whose embedding capacity is defined to be t_k in advance as shown in the last row of Table 2.
(3) Convert the bitstream of an input message M with t_k bits into a decimal value b.
(4) Create a new difference δ_i' between the two pixels in P_i as
$$\delta_i' = l_k + b.$$
(5) Compute two candidate pixel values g_{i1}' and g_{i2}' in terms of δ_i' for p_i as follows:
$$\begin{cases} g_{i1}' = g'' - \delta_i'; \\ g_{i2}' = g'' + \delta_i'. \end{cases} \quad (10)$$
(6) Select one of g_{i1}' and g_{i2}' for use as the final pixel value g_i'' of p_i in P_i by the following way to end the message-bit embedding process:
$$g_i'' = \begin{cases} g_{i1}' & \text{if } |g_i - g_{i1}'| < |g_i - g_{i2}'| \\ & \text{and } 0 \le g_{i1}' \le 255; \\ g_{i2}' & \text{otherwise.} \end{cases} \quad (11)$$

Swain [15] modified the method proposed by Khodaei and Faez [14] by using 2×3 and 3×3 blocks, and used

TABLE 2. Quantization range table for the method proposed by Khodaei and Faez [14].

Index	1	2	3	4	5
Range	[0, 7]	[8, 15]	[16, 31]	[32, 63]	[64, 255]
Width	8	8	16	32	192
No. of embedded bits	3	3	4	5	6

the shared pixel to form five and eight pixel pairs with the surrounding pixels for the two block types, respectively, to increase the embedding capacity. A problem that the pixel values yielded by Khodaei and Faez's method [14] after message embedding may exceed the boundaries of 0 or 255 is solved by modifying Eq. (11) described previously for selecting the pixel value g_i'' for p_i in P_i to be

$$g_i'' = \begin{cases} g_{i2}' & \text{if } g_{i1}' < 0; \\ g_{i1}' & \text{if } g_{i2}' > 255; \\ g_{i1}' & \text{if } \left|g_i - g_{i1}'\right| < \left|g_i - g_{i2}'\right|, \quad g_{i1}' \geq 0, \\ & \text{and } g_{i2}' \leq 255; \\ g_{i2}' & \text{otherwise.} \end{cases} \tag{12}$$

C. THE GPVD METHOD

A PVD method based on the use of *general* quantization ranges (GPVD) was proposed by Wu [16] in 2021. Similar to the original PVD method [8], this method was also proposed to partition the cover image into 1×2 non-overlapping blocks but removed the restriction of limiting the widths of the quantization ranges to be powers of two, as shown in Table 3. Also, the secret message bits are converted into a *multiple-based number* [17]–[21] and each digit in the number is embedded into a corresponding block in the image.

The so-called multiple-based number is one with the bases of the digits being not all identical. An n-digit multiple-based number N can be represented by

$$N = d_{n-1(b_{n-1})} d_{n-2(b_{n-2})} \ldots d_{1(b_1)} d_{0(b_0)}$$

where b_i is the base of the i-th digit d_i, $b_i > 0, 0 \leq d_i \leq b_i - 1$, for $i = 0, 1, \ldots, n-1$.

1) THE MESSAGE EMBEDDING PROCESS

Specifically, in Wu [16] a given B-bit message bitstream is transformed into a decimal value m at first. Then, the pixel pairs P_i with $i = 0, 1, 2, \ldots$ of the cover image C, which *can* be utilized to embed data *without causing noise* as mentioned in Section II.A, are identified and used in order for embedding m in the message embedding process described by the following steps.

(1) Let P_i be the i-th pixel pair in the cover image with pixel values (g_{i1}, g_{i2}), which can be used to embedded data.

(2) Compute the pixel-value difference δ_i of g_{i1} and g_{i2} according to Eq. (1).

(3) Let the quantization range R corresponding to δ_i in Table 3 be $[l_k, u_k]$.

(4) Define the base value b_i of digit d_i in the multiple-based number N described above, which corresponds to the pixel pair P_i, to be the width of the quantization range R, namely, $u_k - l_k + 1$, as shown in Table 3.

(5) Define the value of digit d_i in N as

$$d_i = m \bmod b_i \tag{13}$$

where the operation of mod yields the remainder of integer division.

(6) Compute a new pixel-value difference δ_i' for pixel pair P_i in terms of the lower bound value l_k in the quantization range R and the digit value of d_i as

$$\delta_i' = \begin{cases} l_k + d_i & \text{if } \delta_i \geq 0; \\ -(l_k + d_i) & \text{if } \delta_i < 0. \end{cases} \tag{14}$$

(7) Adjust the two pixel values (g_{i1}, g_{i2}) of pixel pair P_i as follows to conduct the message embedding of the digit value d_i in P_i:

$$\begin{aligned} &\left(g_{i1}', g_{i2}'\right) \\ &= \begin{cases} \left(g_{i1} - \left\lceil \dfrac{\delta_i' - \delta_i}{2} \right\rceil, g_{i2} + \left\lfloor \dfrac{\delta_i' - \delta_i}{2} \right\rfloor\right) \\ \qquad \text{if } \delta_i \bmod 2 \neq 0; \\ \left(g_{i1} - \left\lfloor \dfrac{\delta_i' - \delta_i}{2} \right\rfloor, g_{i2} + \left\lceil \dfrac{\delta_i' - \delta_i}{2} \right\rceil\right) \\ \qquad \text{if } \delta_i \bmod 2 = 0 \end{cases} \end{aligned} \tag{15}$$

where the operation of mod yields the remainder of integer division, and $\lceil \cdot \rceil$ and $\lfloor \cdot \rfloor$ are the ceiling and floor functions, respectively.

(8) If the condition $\prod_i b_i \geq 2^B$ is met, then end the message bitstream embedding process with the resulting image C' as the output; otherwise, compute a new decimal value m as follows and continue:

$$m = m \text{ div } b_i, \tag{16}$$

where the operation of div yields the quotient of integer division.

(9) Repeat the previous steps with the new value of m to embed more digits of the multiple-based number N until done.

As an illustration example, suppose that it is desired to embed a 10-bit secret bitstream $1011001000_2 = 712_{10}$ into four pixel pairs whose pixel values are (90, 110), (73, 60), (100, 250), and (150, 153), respectively, as shown in Fig. 1. Firstly, all pixel-value differences are calculated according to Eq. (1). The resulting differences are 20, -13, 150, and 3 which correspond to respectively the quantization ranges [20, 34], [9, 19], [131, 255], and [0, 8] in Table 3. Then, each of the four pixel pairs are taken to go through the falling-off-boundary check using equations similarly to

TABLE 3. Quantization range table for the method proposed by Wu [16].

Index	1	2	3	4	5	6
Range	[0, 8]	[9, 19]	[20, 34]	[35, 60]	[61, 130]	[131, 255]
Width	9	11	15	26	70	125
No. of embedded bits	3.16	3.45	3.90	4.705	6.12	6.69

Eqs. (4) through (6). Among them, the pixel pair (100, 250) is found to be out of bounds, as shown by the following computations:

pixel value difference $\delta = 250 - 100 = 150$ by Eq. (1);
$\delta \in [l_6, u_6] = [131, 255]$ according to Table 3;
$\delta' = u_6 = 255$ ($\because \delta > 0$) by Eq. (5);
$(\delta - \delta')/2 = 52.5$,
$(100 - \lfloor 52.5 \rfloor, 250 + \lceil 52.5 \rceil) = (48, 303)$($\because \delta \bmod 2 = 0$) by Eq. (4);
fall_off$(100, 250) = true$ by Eq (6).

Therefore, no data can be embedded into this pixel pair and the pixel values are kept unchanged. As to the pixel-value differences of the other three pixel pairs, the corresponding widths of the quantization ranges are 15, 11, and 9, respectively. By using Eq. (13), the corresponding digit values in the multiple-based number can be calculated to be 7, 3, and 4, respectively, as shown in the following: 712 mod $15 = 7$ with quotient $Q_1 = 47$; 47 mod $11 = 3$ with quotient $Q_2 = 4$; 4 mod $9 = 4$ with quotient $Q_3 = 0$.

Finally, these digit values are embedded into the corresponding pixel pairs by use of Eqs. (14) and (15), resulting in the new pixel differences 27, −12, and 4, respectively, and the new pixel values (87, 114), (72, 60), and (149, 153) of the pixel pairs, as illustrated in Fig. 1.

2) THE MESSAGE EXTRACTION PROCESS

To retrieve the secret message from a stego-image S, the same group of pixel pairs which were used to embed message data *without causing noise* as mentioned in the Section II.A are used in the message extraction process. Firstly, denote the message data to be extracted as m with its initial value set to be 0. Also, define a parameter c_i and set its initial value c_0 to be 1. Then, the B-bit message bitstream M can be extracted from the pixel pairs in S in order for $i = 0, 1, \ldots$ by the following steps.

(1) Let P_i denote the i-th pixel pair with pixel values (g_{i1}^*, g_{i2}^*) in the stego-image S, which can be used for extracting data.
(2) Compute the pixel-value difference δ_i^* of g_{i1}^* and g_{i2}^* according to Eq. (1).
(3) Let the quantization range R corresponding to δ_i^* in Table 3 be $[l_k, u_k]$.
(4) Define the base value b_i of digit d_i in the multiple-based number N described previously, which corresponds to the pixel pair P_i, as the width of the quantization range R, namely, $u_k - l_k + 1$, as shown in Table 3.
(5) Extract the value of digit d_i from P_i as
$$d_i = |\delta_i^*| - l_k. \tag{17}$$
(6) Create a new *digit place value* represented by d_i in m by modifying m in the following way:
$$m = m + d_i \times c_i. \tag{18}$$
(7) Compute c_{i+1} by multiplying c_i by the base value b_i of digit d_i corresponding to P_i, i.e., set
$$c_{i+1} = c_i \times b_i. \tag{19}$$
(8) If the condition $c_{i+1} \geq 2^B$ is met, end the message extraction process and convert m to be a bit string of length B as output; otherwise, continue the processing of the next pixel pair by repeating the previous steps.

As an illustrative example of message extraction, suppose that it is desired to extract the embedded 10-bit secret message from the four pixel pairs (87, 114), (72, 60), (100, 250), and (149, 153) mentioned in the previous example of message embedding. Firstly, the pixel-value differences of the pixel pairs are calculated to be 27, −12, 150, 4, respectively, using Eq. (1). From Table 3, these differences can be determined to belong to the quantization ranges [20, 34], [9, 19], [131, 255], and [0, 8], respectively. Then, the four pixel pairs are taken to go through the falling-off-boundary checks with pixel pair (100, 250) being checked to be out of the bounds as described above in the message embedding process, showing that no secret message was embedded into the pixel pair previously. The range widths of the pixel-value differences for the remaining three pixel pairs are 15, 11, and 9, respectively. Accordingly, Eq. (17) is used to extract respectively from these three pixel pairs the values 7, 3, and 4 of the corresponding digits in the multiple-based number. Therefore, the resulting multiple-based number is $4_9 3_{11} 7_{15}$, which is then converted into the decimal value 712_{10}, alternatively into the 10-digit bitstream 1011001000_2, and finally taken to be the extracted secret message as the output.

III. PROPOSED MMGPVD METHOD

In the proposed MMGPVD method, which is based on the GPVD technique [16], the adopted quantization range table is shown in Table 4. The method can be applied to block templates of various sizes, including 1×3, 2×2, 2×3, and 3×3, as shown in Fig. 2, where g_0 denotes the shared pixel of a given block and can be used to form multiple pixel pairs with other pixels in the block.

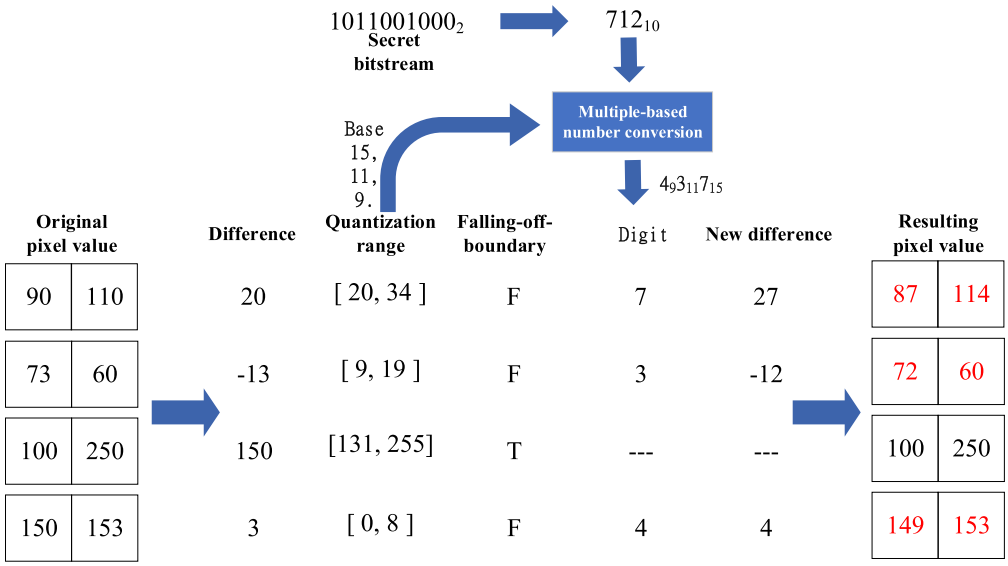

FIGURE 1. Schematic diagram of GPVD embedding.

TABLE 4. Quantization range table for the proposed MMGPVD method.

Index	1	2	3	4	5	6	7
Range	[0, 8]	[9, 19]	[20, 34]	[35, 60]	[61, 110]	[111, 173]	[174, 255]
Width	9	11	15	26	50	63	82
No. of embedded bits	3.16	3.45	3.90	4.70	5.64	5.97	6.35

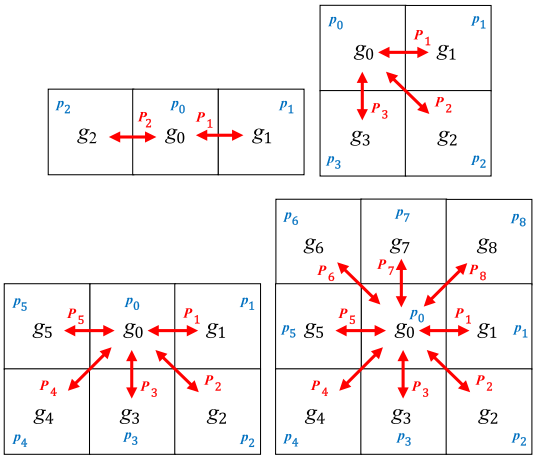

FIGURE 2. The four block templates in this paper.

The proposed data embedding method partitions a cover image sequentially into non-overlapping blocks in a raster-scan order according to the sizes of the selected block templates as shown in Fig. 2. When partitioning the image in this way, it is possible that there will leave *residual pixels* to the right or on the bottom of the cover image. In this study, the same block templates are applied to these residual pixels to form as many *incomplete blocks* as possible in order to embed as many secret message bits as possible to raise the embedding rate. Fig. 3 is a schematic diagram of the partitioning of a 3×5 image using the 2×2 block template. Fig. 3(a) shows the first *complete* block in a raster-scan order, while Fig. 3(b) shows the first *incomplete* block. A total of two complete blocks and four incomplete blocks will be formed after the image partitioning is completed.

A. THE MESSAGE EMBEDDING PROCESS

Like the original PVD method [8], the proposed MMGPVD method performs a falling-off-boundary check on each pixel pair *before* embedding the secret message to avoid the problem that the new pixel values may be out of the grayscale range of 0 to 255 after message embedding. Because the proposed method only adjusts the pixel value of a single-pixel in the pixel pair when *embedding* data, unlike the traditional PVD method [8] which adjusts the pixel values of both pixels, the proposed method also performs a falling-off-boundary check in a way that only adjusts the pixel value of a single pixel in the pixel pair.

1) FALLING-OFF-BOUNDARY CHECK

In addition, the proposed method also performs the falling-off-boundary checking on each pixel pair before *extracting*

the secret message. If the checking result reveals that message embedding may cause the pixel values of a pixel pair to be out of the range of 0 to 255, it means that the pixel pair has not been used to embed message bits, and nothing should be extracted from it. Algorithm 1 below explains how the proposed MMGPVD steganographic method performs the falling-off-boundary check. Algorithm 1 will be called both in Algorithms 2 and 3 in the following.

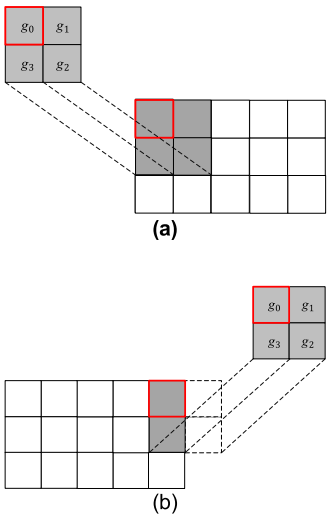

FIGURE 3. Schematic diagram of partitioning a 3 × 5 image in a raster-scan order using the 2 × 2 block template. (a) The first complete block. (b) The first incomplete block.

Algorithm 1 (Falling-Off-Boundary Checking for Identifying Usable Pixel Pairs):

Input: a pixel pair $P(g_0, g_1)$.

Output: checking result S (*true* or *false*), indicating whether the pixel values of the input pixel pair fall possibly outside the grayscale range of 0 to 255 when P is used to embed a secret message.

Steps:

Step 1. Calculate the pixel-value difference δ of the two pixels g_0 and g_1 of the pixel pair P as

$$\delta = g_1 - g_0. \tag{20}$$

Step 2. Assume the quantization range corresponding to $|\delta|$ to be the k-th quantization range $[l_k, u_k]$ of the quantization range table, Table 4, with l_k and u_k satisfying the condition $l_k \le |\delta| \le u_k$.

Step 3. Use u_k as the new pixel difference value for performing falling-off-boundary checks in the following way to determine whether the boundaries are likely to be exceeded after message embedding.

3.1 *//Simulating message embedding by adjusting g_1 while keeping g_0 unchanged*
Compute a new pixel value $\hat{g}_1$ for g_1 as

$$\hat{g}_1 = \begin{cases} g_0 + u_k & \text{if } \delta \ge 0; \\ g_0 - u_k & \text{if } \delta < 0. \end{cases} \tag{21}$$

3.2 Determine if $\hat{g}_1$ is outside the pixel value range of 0 to 255 to obtain the checking result S as

$$S = \begin{cases} false & \text{if } 0 \le \hat{g}_1 \le 255; \\ true & \text{otherwise.} \end{cases} \tag{22}$$

Step 4. Exit with S as the desired output.

2) THE MESSAGE EMBEDDING ALGORITHM

When using the proposed MMGPVD steganographic method to embed secret messages, the cover image is firstly partitioned sequentially in a raster-scan order into non-overlapping blocks using a block template. Each block so obtained may be a complete block or an incomplete one. The message bits are firstly embedded in the block's shared pixel using the t-bit LSB substitution method and the OPAP [9]. Then, message bits are embedded in the pixel pairs in the block in order according to the concept of multiple-based number proposed in the GPVD method [16]. The proposed method can use block templates of the sizes of 1×3, 2×2, 2×3, or 3×3. In addition, the bit order of the secret message is randomized before message embedding using a random number generator controlled by a secret key. The secret key can be produced by any appropriate key generator. The same secret key should be used in the embedding and extraction processes. Algorithm 2 below includes the details of the message embedding process. It is noted that in the following algorithm, an n-digit multiple-based number N described in the following is assumed to be created with its digits being embedded in the pixels of the cover image:

$$N = d_{n-1(b_{n-1})} d_{n-2(b_{n-2})} \ldots d_{1(b_1)} d_{0(b_0)}$$

where b_r is the base of the r-th digit d_r, $b_r > 0$, $0 \le d_r \le b_r - 1$, for $r = 0, 1, \ldots, n-1$.

Algorithm 2 (Message Data Embedding Based on the Concept of Multiple-Based Number):

Input:

(1) a cover image C with size $W \times H$;
(2) a block template T with size $w \times h$ and pixels p_0, $p_1, \ldots, p_{(w \times h-1)}$;
(3) a secret message M with bit length ℓ;
(4) a secret key K and a random number generator G; and
(5) a parameter t of the number of bits embedded in each shared pixel of the partitioned blocks.

Output: stego-image S.

Steps:

Stage 1: Initialization.

Step 1. Use template T to partition C into n non-overlapping blocks in a raster-scan order with $n = \lceil W/w \rceil \times \lceil H/h \rceil$ with the resulting blocks either being complete or incomplete; let pixel $p_{i,j}$ be the j-th pixel in the i-th block B_i, where $j = 0, 1, \ldots, (w \times h - 1)$, and $i = 1, 2, \ldots, n$; and let $p_{i,0}$ be the selected shared pixel of B_i, and $P_{i,j}(g_{i,0}, g_{i,j})$ be the j-th pixel pair of B_i with pixel values $g_{i,0}$ and $g_{i,j}$, where $j = 1, 2, \ldots, (w \times h - 1)$, and $i = 1, 2, \ldots, n$.

Step 2. Use the secret key K and the random number generator G to randomize the bit positions in M; convert the resulting M into a big-integer m; and define a *big-integer* c with an initial value 1 for use in deciding whether message embedding into shared pixels and pixel pairs should be ended.

Stage 2: Processing of the blocks one by one.

Step 3. Set r to be -1, and perform Step 3.1 through Step 3.8 for $i = 1, 2, \ldots, n$ in order.

Stage 2.1 Processing of the shared pixel of each block.

3.1 If the shared pixel $p_{i,0}$ of the i-th block B_i is not within the scope of C, then regard B_i as *unusable* for embedding messages, and skip to Step 3.8 to process the next block; otherwise, continue.

3.2 Set $r = r+1$; assign the value 2^t as the base b_r of shared pixel $p_{i,0}$ where t is a pre-selected parameter; and calculate the value d_r of the r-th digit in the multiple-based number N as

$$d_r = m \bmod b_r \tag{23}$$

where the operation of mod yields the remainder of integer division.

3.3 Embed d_r into shared pixel $p_{i,0}$ with value $g_{i,0}$ using the t-bit LSB substitution technique, resulting in a new pixel value $g'_{i,0}$ computed as follows for use as the new pixel value of $p_{i,0}$:

$$g'_{i,0} = \left(g_{i,0} \operatorname{div} b_r\right) \times b_r + d_r \tag{24}$$

where the operation of div yields the quotient of integer division.

3.4 Adjust the pixel value $g'_{i,0}$ of $p_{i,0}$ by the OPAP method [9] according to the following formula, resulting in a new pixel value $g''_{i,0}$ of the shared pixel $p_{i,0}$:

$$g''_{i,0} = \begin{cases} g'_{i,0} - 2^t & \text{if } g'_{i,0} - g_{i,0} > 2^{t-1} \\ & \text{and } 0 \le g'_{i,0} - 2^t \le 255; \\ g'_{i,0} + 2^t & \text{if } g'_{i,0} - g_{i,0} < -2^{t-1} \\ & \text{and } 0 \le g'_{i,0} + 2^t \le 255; \\ g'_{i,0} & \text{otherwise.} \end{cases} \tag{25}$$

3.5 Compute a new value for m by the following formula:

$$m = m \operatorname{div} b_r \tag{26}$$

where the operation of div yields the quotient of integer division.

3.6 *//Test whether the message embedding task is completed.*
Set c to be $c \times b_r$; and check if $c \ge 2^\ell$; if yes, then end the entire message embedding process and skip to Step 4; otherwise, continue.

Stage 2.2 Processing of the pixel pairs of each block.

3.7 Perform Step 3.7(a) through Step 3.7(k), for $j = 1, 2, \ldots, (w \times h - 1)$ in order.

(a) If pixel $p_{i,j}$ of the j-th pixel in the i-th block B_i is not within the scope of C, then regard pixel pair $P_{i,j}$ consisting of $p_{i,0}$ and $p_{i,j}$ as *unusable* for embedding messages, and skip to Step 3.7(k) to process the next pixel pair; otherwise, continue.

(b) Perform the falling-off-boundary check on pixel pair $P_{i,j}(g''_{i,0}, g_{i,j})$ using Algorithm 1 where $g''_{i,0}$ is the *adjusted* pixel value of shared pixel $p_{i,0}$; and if the checking result is *true*, then regard the boundary as likely to be exceeded so that pixel pair $P_{i,j}$ is unusable for message embedding and skip to Step 3.7(k); otherwise, continue.

(c) Calculate the pixel-value difference $\delta_{i,j}$ of the two pixel values $g''_{i,0}$ and $g_{i,j}$ of the pixel pair $P_{i,j}$ as

$$\delta_{i,j} = g_{i,j} - g''_{i,0}. \tag{27}$$

(d) Assume that the quantization range corresponding to $|\delta_{i,j}|$ is the k-th range $[l_k, u_k]$ in the difference quantization table, with l_k and u_k satisfying the condition $l_k \le |\delta_{i,j}| \le u_k$.

(e) *//If the width of* $[l_k, u_k]$ *is* 1, *the pixel pair is unusable for embedding.*
If $u_k - l_k + 1 = 1$, then skip to Step 3.7(k); otherwise, continue.

(f) Set $r = r+1$; assign the value $u_k - l_k + 1$ as the base b_r of the pixel pair $P_{i,j}$; and calculate the value d_r of the r-th digit in the multiple-based number N as

$$d_r = m \bmod b_r \tag{28}$$

where the operation of mod yields the remainder of integer division.

(g) Compute a new pixel-value difference δ' as

$$\delta'_{i,j} = \begin{cases} l_k + d_r & \text{if } \delta_{i,j} \ge 0; \\ -(l_k + d_r) & \text{if } \delta_{i,j} < 0. \end{cases} \tag{29}$$

(h) *//Embedding the digit* d_r *by adjusting* $g_{i,j}$ *while keeping* $g''_{i,0}$ *unchanged*
Compute a new pixel value $g'_{i,j}$ for $g_{i,j}$ of pixel $p_{i,j}$ as

$$g'_{i,j} = g''_{i,0} + \delta'_{i,j}. \tag{30}$$

(i) Compute a new value for m by:

$$m = m \operatorname{div} b_r \tag{31}$$

where the operation of div yields the quotient of integer division.

(j) *//Test whether the embedding task is completed.*
Set c to be $c \times b_r$, and check if $c \ge 2^\ell$; if yes, then end the entire message embedding process and skip to Step 4; otherwise, continue.

(k) Continue.
3.8 Continue.
Stage 3 Ending.
Step 4. Exit with the resulting image as the desired stego-image S.

The block template T, bit length ℓ of the secret message, secret key K, and bit number parameter t are the required side information for use in the message embedding process of the proposed method as can be seen in Algorithm 2 above. All the information needs to be transmitted to the message extraction side for use in the message extraction process.

3) AN ILLUSTRATIVE EXAMPLE OF MESSAGE EMBEDDING

The following example illustrates the message embedding process described by Algorithm 2. Suppose that it is desired to embed an 18-bit secret message bitstream $M = 101100100000101100$ into a 2×3 cover image using the 2×2 block template shown in Fig. 2 and the quantization range table shown in Table 4. This bitstream M is equivalent to a decimal value $m = 182316$ after it is converted from binary to decimal. As shown in Fig. 4, the cover image C is partitioned into two non-overlapping blocks using the 2×2 block template.

a: PROCESSING OF THE FIRST BLOCK

Embedding of the decimal value m starts from dealing with the first block B_1. The pixel value of the shared pixel p_{10} of B_1 is $g_{10} = 96 = 01100000_2$. If the number t of bits to be embedded in p_{10} by the LSB substitution method is 4, the proposed method treats p_{10} as a pixel that provides a digit b_0 of the multiple-based number N, and sets the corresponding base to be $b_0 = 2^4 = 16$. Then, the value of m mod 16 is computed according to Eq. (23), resulting in the remainder $12 = 1100_2$. It means that the resulting least significant digit d_0 of the multiple-based number is 12_{16}. The lowest four bits of the shared pixel p_{10} are then replaced with 1100_2, giving a new pixel value $g'_{10} = 108 = 01101100_2$ according to Eq. (24) as shown by the following computation:

$$b_0 = 16; \quad d_0 = 12; \; g_{10} = 96;$$
$$g'_{10} = (g_{10} \text{ div } b_0) \times b_0 + d_0 = 96 + 12 = 108.$$

Then, the OPAP [9] is performed on the pixel value of p_{10} according to Eq. (25) as shown in the following computation:

$$g'_{10} - g_{10} = 108 - 96 = 12; \quad t = 4;$$
$$g''_{10} = g'_{10} - 2^4 = 108 - 16 = 92$$
$$(\because \; g'_{10} - g_{10} = 12 > 2^{t-1} = 8$$
$$\text{and } 0 \le g'_{10} - 2^t = 92 \le 255),$$

yielding an adjusted pixel value $92 = 01011100_2$. After processing the shared pixel p_{10}, the value of m div 16 is computed according to Eq. (26), resulting in the new value $m = 11394$.

The shared pixel p_{10} and the other pixels in the first block form three pixel pairs, denoted in order as P_{11}, P_{12}, and P_{13} with their respective pixel values being (92, 117), (92, 82), and (92, 228) where 92 is the *new* value g''_{10} of the shared pixel p_{10}. The pixel-value differences of these pixel pairs are 25, -10, and 136, respectively, which correspond to respectively the quantization ranges of [20, 34], [9, 19], and [111, 173] in Table 4. The falling-off-boundary check described by Algorithm 1 is then performed on the three pixel pairs P_{11}, P_{12}, and P_{13}. One of the new values (92, 265) of the two pixels in P_{13} resulting from applying Eq. (21) on the original values (92, 228), namely, 265 is found to be out of bounds; therefore, P_{13} is not used for message embedding. The widths of the quantization ranges corresponding to the pixel-value differences δ_{11} and δ_{12} of the remaining two pixel pairs P_{11} and P_{12} are 15 and 11, respectively, according to Table 4. The proposed method treats each of P_{11} and P_{12} as a pixel pair that provides a digit of the multiple-based number N, and sets the corresponding bases to be $b_1 = 15$ and $b_2 = 11$, respectively. Accordingly, a digit $d_1 = 9_{15}$ of N is formed from the computation of m mod 15 with 9 as the remainder. The value 9 of the digit d_1 is then embedded into pixel pair P_{11} with values (92, 117), resulting in the new pixel values $(g''_{10}, g'_{11}) = (92, 121)$, where the value $g'_{11} = 121$ comes from the following computations according to Eqs. (29) and (30):

$[l_3, u_3] = [20, 34]$; $g''_{10} = 92$; $\delta_{11} = 117 - 92 = 25$; $d_1 = 9$;
$\delta'_{11} = l_3 + d_1 = 20 + 9 = 29$ ($\because \delta_{11} = 25 > 0$) by Eq. (29);
$g'_{11} = g''_{10} + \delta'_{11} = 92 + 29 = 121$ by Eq. (30).

After processing the first pixel pair P_{11}, the computation of m div 15 yields the quotient 759 for use as the new value of m. With the next pixel pair P_{12} with values (92, 82), the computation of m mod 11 yields the remainder value 0. So, the resulting third digit d_2 of the multiple-based number N is $d_2 = 0_{11}$. After embedding this digit into P_{12} with original values (92, 82), the new values of P_{12} becomes $(g''_{10}, g'_{12}) = (92, 83)$ where the value 83 comes from the following computations according to Eqs. (29) and (30):

$[l_2, u_2] = [9, 19]$; $g''_{10} = 92$; $\delta_{12} = 82 - 92 = -10$; $d_2 = 0$;
$\delta'_{12} = -(l_2 + d_2) = -(9 + 0) = -9$ ($\because \delta_{12} = -10 < 0$) by Eq. (29);
$g'_{12} = g''_{10} + \delta'_{12} = 92 - 9 = 83$. by Eq. (30).

After processing the second pixel pair P_{12}, the computation of m div 11 yields the quotient 69 for use as the new value of m.

b: PROCESSING OF THE SECOND BLOCK

The pixel value of the shared pixel p_{20} of B_2 is $32 = 00100000_2$. Like the shared pixel p_{10} of B_1, p_{20} provides a digit d_3 of the multiple-based number N, with the corresponding base $b_3 = 2^4 = 16$. Then, the value of m mod 16 is computed according to Eq. (23) to yield the remainder $5 = 0101_2$, which means that digit d_3 of N is 5_{16}. The lowest four bits of p_{20} are then replaced with 0101_2 according to 4-bit LSB substitution described by Eq. (24), giving a new

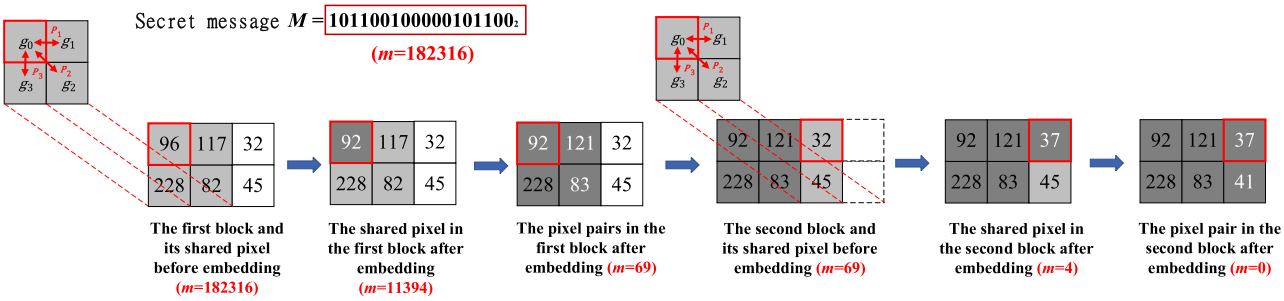

FIGURE 4. Schematic diagram of the proposed MMGPVD embedding process using the 2 × 2 block template and the quantization range table shown in Table 4.

pixel value $37 = 00100101_2$ to p_{20}. Then, the OPAP [9] is performed on the pixel value g_{20} of p_{20} according to Eq. (25), yielding an unchanged value $g'_{20} = 37$. Subsequently, the value of m div 16 is computed according to Eq. (26), resulting in the new value $m = 4$.

Furthermore, it is noted that the second block is an incomplete block, so the shared pixel p_{20} can only be used to form one pixel pair with the other pixel in B_2, denoted as P_{23} with pixel values (37, 45) where 37 is the previously-mentioned *new* value g''_{20} of p_{20}. The pixel-value difference δ_{23} of P_{23} is $45 - 37 = 8$, to which the corresponding quantization range in Table 4 is [0, 8]. The falling-off-boundary check described by Algorithm 1 is then performed on P_{23}, and neither pixel value of P_{23} is found out of bounds; therefore, P_{23} can be used for message embedding. The widths of the quantization range [0, 8] is 9, meaning that the base b_4 provided by P_{23} is 9. Also, the computation of m mod 9 yields a remainder 4; therefore, a digit $d_4 = 4_9$ of N is formed. The value 4 of the digit d_4 is then embedded into pixel pair P_{23} with values (37, 45), resulting in the new pixel values $(g''_{20}, g'_{23}) = (37, 41)$ where $g'_{23} = 41$ is computed as follows:

$$[l_1, u_1] = [0, 8]; g''_{20} = 37; \delta_{23} = 8; d_4 = 4;$$
$$\delta'_{23} = l_4 + d_4 = 0 + 4 = 4 (\because \delta_{23} = 8 > 0); \text{by Eq. (29)};$$
$$g'_{23} = g''_{20} + \delta'_{23} = 37 + 4 = 41 \text{by Eq. (30)}.$$

After processing pixel pair P_{23}, computation of div 9 on m yields 0. At this point, the product of all created bases which form the multi-based number N is

$$16 \times 15 \times 11 \times 16 \times 9 = 380160.$$

It can be verified that $380160 \geq 2^{18}$, meaning that the embedding of the 18-bit secret message is complete, and the resulting multiple-based number N is

$$N = 4_9 5_{16} 0_{11} 9_{15} 12_{16} = 182316$$

which is exactly the decimal value $m = 182316$ of the message M desired to be embedded, as expected.

B. THE MESSAGE EXTRACTION PROCESS

When using the proposed MMGPVD steganographic method to extract secret messages, the stego-image is first partitioned into non-overlapping blocks in a raster-scan order using the same block template as that specified in the side information. The image is sequentially partitioned, creating both complete blocks and incomplete ones identical to those obtained in the embedding process. The message bitstream is first extracted for the shared pixel in each block using the multi-bit LSB substitution method [7]; then, the multiple pixel pairs in the block are processed according to the GPVD concept [16] in order. Finally, the bit positions of the extracted data are de-randomized using the same secret key and the same random number generator as used in the embedding process. Algorithm 3 below shows how to extract a secret message bitstream from a stego-image using the proposed method, where the input items (2) through (4) are the side information used in the message embedding process (Algorithm 2).

1) THE MESSAGE EXTRACTION ALGORITHM

Algorithm 3 (Message Data Extraction):

Input:

(1) a stego-image S with size $W \times H$;
(2) the block template T with size $w \times h$ and pixels p_0, $p_1, \ldots p_{(w \times h-1)}$;
(3) the secret key K, and the random number generator G;
(4) the parameter t of the number of bits embedded in each shared pixel of the partitioned blocks

Output: Secret message M with bit length ℓ.

Steps:

Stage 1: Initialization.

Step 1. Use template T to partition S into n non-overlapping blocks in a raster-scan order with $n = \lceil W/w \rceil \times \lceil H/h \rceil$ where the resulting blocks contain both complete and incomplete blocks; let pixel $p_{i,j}$ be the j-th pixel in the i-th block B_i, where $j = 0, 1, \ldots, (w \times h - 1)$, and $i = 1, 2, \ldots, n$; and let $p_{i,0}$ be the shared pixel of the i-th block, and $P_{i,j}(g_{i,0}, g_{i,j})$ be the j-th pixel pair of the i-th block with pixel values $g_{i,0}$ and $g_{i,j}$, where $j = 1, 2, \ldots, (w \times h - 1)$, and $i = 1, 2, \ldots, n$.

Step 2. Set a big-integer m to be 0 and define a *big-integer* c with an initial value 1 for use in deciding whether message extraction from the shared pixels and pixel pairs is completed.

Stage 2: Processing of the blocks one by one.

Step 3. Set r to be -1, and perform Step 3.1 through Step 3.6, for $i = 1, 2, \ldots, n$ in order.

Stage 2.1 Processing of the shared pixel of each block.

3.1 If the shared pixel $p_{i,0}$ of the i-th block B_i is not within the scope of S, then regard B_i as *unusable* for message extraction, and skip to Step 3.6; otherwise, continue.

3.2 Set $r = r+1$, extract the value d_r of the r-th digit in the multiple-based number N from the t LSBs of shared pixel $p_{i,0}$ as

$$d_r = g_{i,0} \bmod 2^t \tag{32}$$

where the operation of mod yields the remainder of integer division.

3.3 Assign the value 2^t as the base b_r of the shared pixel $p_{i,0}$, add the place value represented by d_r in the multiple-based number N to m as

$$m = m + (d_r \times c)\,. \tag{33}$$

3.4 *//Test whether the* extraction *job is completed.*
Set c to $c \times b_r$; and check if $c \geq 2^\ell$; if yes, then end the entire message extraction process and skip to Step 4; otherwise, continue.

Stage 2.2 Processing of the pixel pairs of each block.

3.5 Perform Step 3.5(a) through Step 3.5(i) for $j = 1, \ldots, (w \times h - 1)$ in order.

(a) If pixel $p_{i,j}$ is not within the scope of S, then regard the j-th pixel pair of the i-th block as unusable for secret message extraction, and skip to Step 3.5(i); otherwise, continue.

(b) Perform the falling-off-boundary check on pixel pairs $P_{i,j}(g_{i,0}, g_{i,j})$ using Algorithm 1. If the checking result is *true*, which means that the pixel pair has not used for embedding secret messages previously in the embedding process, then skip to Step 3.5(i) accordingly; otherwise, continue.

(c) Calculate the pixel-value difference $\delta_{i,j}$ of the two pixels $g_{i,0}$ and $g_{i,j}$ of pixel pair $P_{i,j}$ as

$$\delta_{i,j} = g_{i,j} - g_{i,0}. \tag{34}$$

(d) Assume that the quantization range corresponding to $|\delta_{i,j}|$ is the k-th range $[l_k, u_k]$ in the quantization range table, with l_k and u_k satisfying the condition $l_k \leq |\delta_{i,j}| \leq u_k$.

(e) *//If the width of* $[l_k, u_k]$ *is* 1, *the pixel pair cannot be used to embed secret messages, and no information can be retrieved from it.*
If $u_k - l_k + 1 = 1$, then skip to Step 3.5(i); otherwise, continue.

(f) Add 1 to r, and extract the value d_r of the r-th digit in the multiple-based number N, which corresponds to the pixel pair $P_{i,j}(g_{i,0}, g_{i,j})$, as

$$d_r = |\delta_{i,j}| - l_k. \tag{35}$$

(g) Add the place value represented by d_r in the multiple-based number N to m as

$$m = m + (d_r \times c)\,. \tag{36}$$

(h) *//Testing* whether *the extraction task is completed.*
Set c to be $c \times b_r$, and check if $c \geq 2^\ell$; if yes, then end the entire message extraction process and skip to Step 4; otherwise, continue.

(i) Continue.

3.6 Continue.

Stage 3 Ending.

Step 4. Convert m into an ℓ-bit bitstream, and de-randomize the bit positions using secret key K and random number generator G. Exit with the resulting bitstream as the desired secret message M.

2) AN ILLUSTRATIVE EXAMPLE OF MESSAGE EXTRACTION

The following is an example to illustrate the message extraction process of the proposed MMGPVD method. As depicted by Fig. 5, it is assumed that a secret message bitstream M of 18 bits is to be extracted from the resulting stego-image of the previous example. Above all, the 2×3 stego-image is partitioned into two non-overlapping blocks B_1 and B_2 using the same 2×2 block template as that used in the message embedding process.

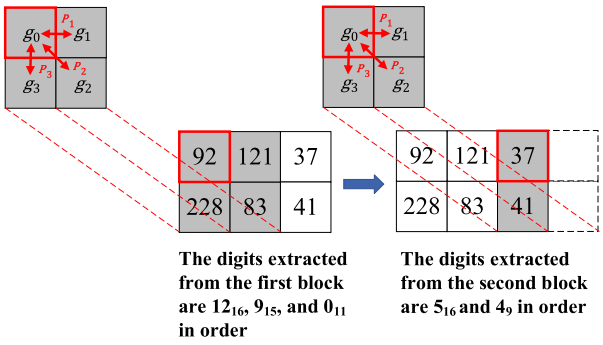

FIGURE 5. Schematic diagram of the proposed MMGPVD extraction using the 2 × 2 block template and the quantization range table in Table 4.

a: PROCESSING OF THE FIRST BLOCK

Extraction of the decimal value m of the stego-message M starts from dealing with the first block B_1. The value of m is set to be 0 initially. The pixel value of the shared pixel p_{10} of B_1 is $92 = 01011100_2$. Presumably, the number t of bits to be extracted from p_{10} by the LSB substitution method is known to be 4. Also, p_{10} is treated as a pixel that provides a digit d_0 of the multiple-based number N, and the corresponding base is taken to be $b_0 = 2^4 = 16$. Therefore, according to Eq. (32), the value $92 \bmod 2^4 = 12$ is computed, which means that the extracted least significant digit d_0 of the multiple-based number is $d_0 = 12_{16}$. After processing the shared pixel p_{10}, the place value of d_0 of the multiple-based number N is added to m according to Eq. (33), resulting in the new value $m = 12$.

The shared pixel p_{10} and the other pixels in the first block form three pixel pairs, denoted in order as P_{11}, P_{12}, and P_{13} with their respective pixel values being (92, 121), (92, 83), and (92, 228). The pixel-value differences of these pixel pairs are $\delta_{11} = 29$, $\delta_{12} = -9$, and $\delta_{13} = 136$, respectively, which correspond to respectively the quantization ranges of $[l_3, u_3] = [20, 34]$, $[l_2, u_2] = [9, 19]$, and $[l_6, u_6] = [111, 173]$ in Table 4. The falling-off-boundary check described by Algorithm 1 is then performed on the three pixel pairs P_{11}, P_{12}, and P_{13}. One of the values (92, 265) of the two pixels in P_{13} resulting from applying Eq. (21) on the original values (92, 228), namely, 265 is found to be out of bounds; therefore, P_{13} is decided to be *unused previously* in the message embedding process, and no information should be extracted from it. The quantization ranges corresponding to the values of the pixel-value differences δ_{11} and δ_{12} of the remaining two pixel pairs P_{11} and P_{12} are $[l_3, u_3] = [20, 34]$, $[l_2, u_2] = [9, 19]$, and their widths are 15 and 11, respectively. Each of P_{11} and P_{12} is a pixel pair into which a digit of the multiple-based number N was embedded in the message process, and the corresponding bases of the embedded digits d_1 and d_2 are taken to $b_1 = 15$ and $b_2 = 11$, respectively. Accordingly, d_1 and d_2 of N are computed by use of Eq. (35), resulting in the values:

$$d_1 = |\delta_{11}| - l_3 = |29| - 20 = 9;$$
$$d_2 = |\delta_{12}| - l_2 = |-9| - 9 = 0.$$

After processing pixel pairs P_{11} and P_{12}, the place value of each of d_1 and d_2 in the multiple-based number N is added to m according to Eq. (36), resulting in a new value of m, namely, $m = 0_{11}9_{15}12_{16} = 156$, completing the message extraction from the first block.

b: PROCESSING OF THE SECOND BLOCK

The pixel value of the shared pixel p_{20} of B_2 is $37 = 00100101_2$. Like the shared pixel p_{10} of B_1, p_{20} provides a digit d_3 of the multiple-based number N, with the base $b_3 = 2^4 = 16$. The lowest four bits 0101_2 of p_{20} are then extracted according to 4-bit LSB substitution described by Eq. (32). It means that the extracted least significant digit d_3 of the multiple-based number is $0101_2 = 5_{16}$. After processing the shared pixel p_{20}, the place value of d_3 of the multiple-based number N is added to m according to Eq. (33), resulting in a new value of m, namely, $m = 5_{16}0_{11}9_{15}12_{16} = 13356$.

Furthermore, it is noted that the second block is an incomplete block, so the shared pixel p_{20} can only be used to form one pixel pair with the other pixel in B_2, denoted as P_{23} with pixel values (37, 41). The pixel-value difference δ_{23} of P_{23} is $41 - 37 = 4$, to which the corresponding quantization range in Table 4 is $[l_1, u_1] = [0, 8]$. The falling-off-boundary check described by Algorithm 1 is then performed on P_{23}, and neither of the two pixel values of P_{23} is found out of bounds, indicating that P_{23} has previously been used for message embedding. The width of the quantization range $[l_1, u_1] = [0, 8]$ is 9, meaning that the digit d_4 provided by P_{23} has a base be $b_4 = 9$. Accordingly, d_4 may be computed according to Eq. (35), resulting in the value

$$d_4 = |\delta_{23}| - l_1 = |4| - 0 = 4.$$

In addition, the place value of d_4 of the multiple-based number N can now be added to m according to Eq. (36), resulting in the new value $m = 4_9 5_{16} 0_{11} 9_{15} 12_{16} = 182316$, completing the message extraction of the second block B_2.

Now, the product of all the bases of the extracted digits in the multi-based number N can be computed as follows:

$$16 \times 15 \times 11 \times 16 \times 9 = 380160.$$

It can be verified that $380160 \geq 2^{18}$, meaning that the extraction of the 18-bit secret message M is complete, and the resulting multiple-based number N is

$$N = 4_9 5_{16} 0_{11} 9_{15} 12_{16} = 182316.$$

Finally, the decimal value 182316 is converted to be an 18-bit bitstream 101100100000101100, which is the extracted secret message M, as expected.

IV. EXPERIMENTAL RESULTS AND COMPARISION WITH OTHER METHODS

In this section, some experimental results yielded by the proposed method are described, followed by a comparison of the results with those yielded by four other related methods.

A. EXPERIMENTAL RESULTS

In the experiments conducted in this study, six cover images named Baboon, Jet, Peppers, Boat, Tree, and House were used, as shown in Fig. 6, all of which are 512×512 grayscale images. Experiments were proceeded by using the computer language C# with the *BigInteger class* for big-integer processing. As shown in Table 5, the experiment began with partitioning the 512×512 images by 1×3, 2×2, 2×3, and 3×3 block templates, and then compared the results in the aspects of the numbers of complete and incomplete blocks, the number of shared pixels, and the number of pixel pairs obtained. Some incomplete blocks were formed when using the 1×3, 2×3, and 3×3 block templates. The use of the 1×3 block template yields the largest number of shared pixels, with the rest being 2×2, 2×3, and 3×3 block templates in order. The use of the 3×3 block template yields the largest number of pixel pairs, with the rest being 2×3, 2×2, and 1×3 block templates in order.

In Table 6, the embedding rate and the resulting image quality of the stego-images for each image of Fig. 6 after message embedding using the proposed method are shown. The embedding rate is defined as the number of embedded bits per pixel in the image. The value of the peak signal-to-noise ratio (PSNR) is used to assess the quality of each resulting image. The higher the PSNR value of a resulting image, the lower the difference of the image from the original one and the less likely the human eye will detect the embedded information.

The quantization range table used in the experiments is presented in Table 4. The 4-bit LSB substitution method [7] with

the OPAP [9] was employed. The embedded secret messages are random bitstreams. In Table 6, it can be observed that except for the Baboon image, the highest embedding rate was obtained by use of the 1×3 block template, followed by the uses of the block templates of sizes 2×2, 2×3, and finally 3×3 in order. The reason for this can be seen from Table 5. The number of blocks obtained by using the 1×3 block template is the largest, with the shared pixel in *each* block being usable for embedding data in the four LSBs; therefore, the amount of message embedding is also larger.

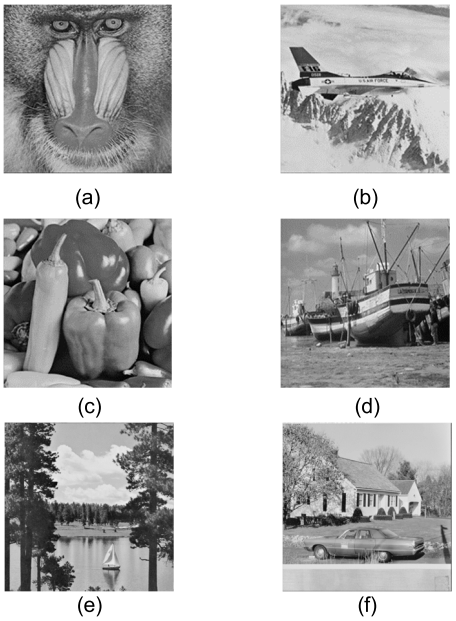

FIGURE 6. Experimental images. (a) Baboon image; (b) Jet image; (c) Peppers image; (d) Boat image; (e) Tree image; (f) House image.

TABLE 5. Comparisons of partitioning 512 × 512 images using various block templates of different sizes.

Block template	No. of pixel pairs per block	No. of complete blocks	No. of incomplete blocks	No. of shared pixels	No. of pixel pairs
1×3	2	87,040	512	87552	174,592
2×2	3	65,536	0	65536	196,608
2×3	5	43,520	256	43776	218,368
3×3	8	28,900	341	29241	232,903

In addition, the image Baboon includes rougher texture, and the pixel-value differences of the adjacent pixels in the image are larger. Therefore, by the proposed method more message bits can be embedded in Baboon than in the other images with smoother texture. Also, the embedding amount using this image is the highest by the use of the 2×2 block template and the lowest by the use of the 1×3 block template, unlike the other images. In Table 6, the PSNR values of all the stego-images are above 30 dB, which means that the changes to the images are not easily detectable by the human eye, demonstrating the imperceptibility of the hidden data embedded by the proposed method.

Figs. 7 and 8 show some resulting stego-images after using the proposed method with 1×3, 2×2, 2×3, and 3×3 block templates for embedding secret messages. Fig. 7 shows the stego-images of Fig. 6(a), and Fig. 8 shows the stego-images of Fig. 6(c). As can be seen from the images in the two figures, the differences between the stego-images and the original ones are invisible to the human eye. The resulting image quality remains good after embedding the secret messages.

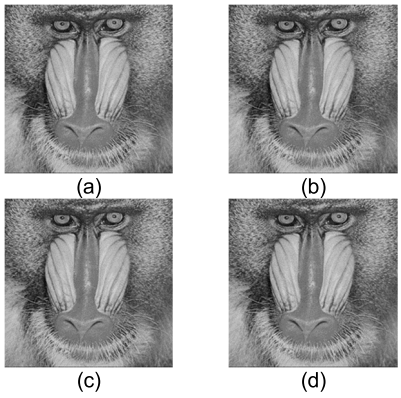

FIGURE 7. The stego-images after embedding using various block templates for Fig. 6(a). (a) using the 1 × 3 block template; (b) using the 2 × 2 block template; (c) using the 2 × 3 block template; (d) using the 3 × 3 block template.

Also, the RS steganalysis method [22] was used to check whether the hidden secret messages in the stego-images can be detected. In the diagrams of the results yielded by this method [22], there are four curves of R_M, R_{-M}, S_M, and S_{-M}, respectively. If the RS steganalysis method detects the presence of hidden information in the stego-image, the R_M and R_{-M} curves will be clearly separated, as will the S_M and S_{-M} curves. If the RS steganalysis method does not detect the hidden information, the two curves R_M and R_{-M} will be very close to each other, as will the two curves S_M and S_{-M}. Figs. 9 and 10 show some resulting diagrams of the RS steganalysis for the stego-images. Specifically, Fig. 9 shows the results for stego-images in Fig. 7, and Fig. 10 shows the results for stego-images in Fig. 8. It can be observed in all these results that R_M and R_{-M} almost overlap, and S_M and S_{-M} also appear to be so. It means that the RS steganalysis does not detect the hidden secret messages in the stego-images yielded by the proposed MMGPVD method using the four different sizes of block templates. These observations prove that the proposed method is secure from the viewpoint of RS steganalysis [22].

B. COMPARISONS WITH OTHER METHODS ABOUT EMBEDDING RATES

In Table 7, a comparison of the embedding rates and the PSNR values yielded by the proposed method with those yielded by the methods proposed by Wu and Tsai [8], Wu [16], Khodaei and Faez [14], and Swain [15] are shown. The embedding rates of the proposed method using 1×3

TABLE 6. Comparison of the embedding rates and PSNR values of the proposed method using various block templates.

512×512 Tested Image	1×3 block		2×2 block		2×3 block		3×3 block	
	Embedding rate (bits / pixel)	PSNR (dB)	Embedding rate (bits / pixel)	PSNR (dB)	Embedding rate (bits / pixel)	PSNR (dB)	Embedding rate (bits / pixel)	PSNR (dB)
Baboon	1.857	32.87	1.876	31.45	1.861	31.24	1.865	30.81
Jet	1.766	34.79	1.744	34.43	1.720	34.23	1.701	34.26
Peppers	1.766	35.09	1.741	34.83	1.711	34.90	1.691	34.90
Boat	1.793	34.24	1.768	34.22	1.744	34.08	1.728	34.03
Tree	1.795	34.39	1.780	33.86	1.757	33.71	1.743	33.59
House	1.781	34.62	1.770	34.05	1.743	34.05	1.729	33.97

TABLE 7. The embedding rates (bits per pixel) and the PSNR values (dB) of the proposed MMGPVD method and the methods proposed by Wu and Tsai [8], Wu [16], Khodaei and Faez [14], and Swain [15].

Tested Image 512 × 512 (pixels)	Wu and Tsai [8] (PVD)		Wu [16] (GPVD)		Khodaei and Faez [14] (PVD + 4-LSB with OPAP)		Swain [15] (PVD + 4-LSB with OPAP)		Proposed MMGPVD method (GPVD + 4-LSB with OPAP)							
	1 × 2 Block		1 × 2 Block		1 × 3 Block		3 × 3 Block		1 × 3 Block		2 × 2 Block		2 × 3 Block		3 × 3 Block	
	Embedding rate	PSNR	Embedding rate	PSNR	Embedding rate	PSNR	Embedding rate	PSNR	Embedding rate	PSNR	Embedding rate	PSNR	Embedding rate	PSNR	Embedding rate	PSNR
Baboon	1.743	37.02	1.782	36.95	3.660	31.65	3.661	28.20	3.715	32.87	3.753	31.45	3.722	31.24	3.730	30.81
Jet	1.562	40.50	1.640	39.88	3.410	34.45	3.229	32.19	3.533	34.79	3.487	34.43	3.439	34.23	3.402	34.26
Peppers	1.547	41.56	1.634	41.08	3.404	32.96	3.191	31.10	3.532	35.09	3.482	34.83	3.422	34.90	3.382	34.90
Boat	1.600	39.54	1.679	39.18	3.479	32.00	3.285	30.81	3.586	34.24	3.535	34.22	3.487	34.08	3.455	34.03
Tree	1.609	39.76	1.681	39.64	3.486	32.69	3.329	30.68	3.591	34.39	3.561	33.86	3.514	33.71	3.487	33.59
House	1.598	39.84	1.664	39.66	3.465	32.77	3.311	31.26	3.563	34.62	3.540	34.05	3.486	34.05	3.459	33.97

TABLE 8. The embedding rates (bits per pixel) and the PSNR values (dB) of the proposed MMGPVD method and the methods proposed by Wu and Tsai [8], Wu [16], Khodaei and Faez [14], and Swain [15] using the quantization rage table shown in Table 4.

Tested Image 512 × 512 (pixels)	Wu and Tsai [8] (PVD)		Wu [16] (GPVD)		Khodaei and Faez [14] (PVD + 4-LSB with OPAP)		Swain [15] (PVD + 4-LSB with OPAP)		Proposed MMGPVD (GPVD + 4-LSB with OPAP)							
	1 × 2 Block		1 × 2 Block		1 × 3 Block		3 × 3 Block		1 × 3 Block		2 × 2 Block		2 × 3 Block		3 × 3 Block	
	Embedding rate	PSNR	Embedding rate	PSNR	Embedding rate	PSNR	Embedding rate	PSNR	Embedding rate	PSNR	Embedding rate	PSNR	Embedding rate	PSNR	Embedding rate	PSNR
Baboon	1.567	39.98	1.777	38.27	3.413	34.22	3.293	31.01	3.715	32.87	3.753	31.45	3.722	31.24	3.730	30.81
Jet	1.520	42.14	1.639	40.41	3.346	35.70	3.133	34.14	3.533	34.79	3.487	34.43	3.439	34.23	3.402	34.26
Peppers	1.510	42.38	1.636	41.24	3.338	34.95	3.116	33.16	3.532	35.09	3.482	34.83	3.422	34.90	3.382	34.90
Boat	1.528	41.41	1.679	39.93	3.361	34.50	3.139	32.92	3.586	34.24	3.535	34.22	3.487	34.08	3.455	34.03
Tree	1.525	41.73	1.681	40.24	3.356	35.14	3.156	32.70	3.591	34.39	3.561	33.86	3.514	33.71	3.487	33.59
House	1.525	41.49	1.667	40.14	3.356	34.96	3.144	33.23	3.563	34.62	3.540	34.05	3.486	34.05	3.459	33.97

block template and the 3×3 block template are higher respectively than those yielded by the methods of [14] and [15] which use block templates of the same sizes. There are two possible reasons for this comparison result. The first reason is that the pixel-value differences of adjacent pixels in a general image are primarily distributed in quantization ranges near the value zero. The quantization range table used in the proposed method is designed to have a larger range width in the two ranges near zero, when compared with the quantization range table used in the methods proposed by

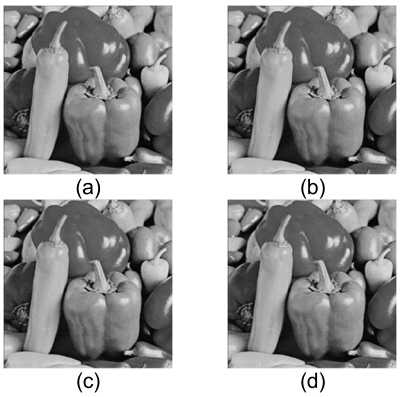

FIGURE 8. The stego-images after embedding using various block templates for Fig. 6(c). (a) using the 1 × 3 block template; (b) using the 2 × 2 block template; (c) using the 2 × 3 block template; (d) using the 3 × 3 block template.

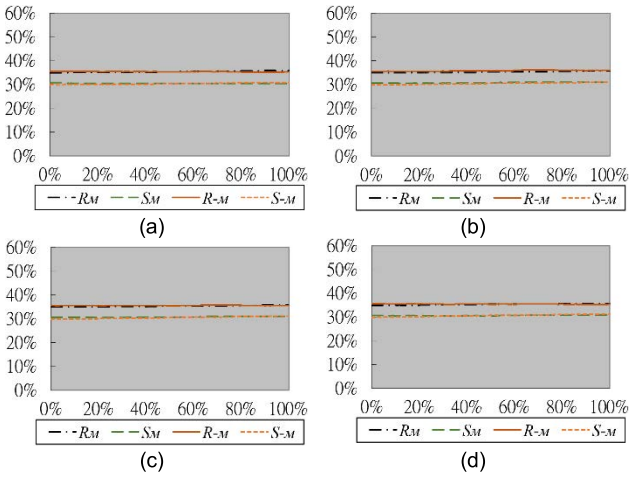

FIGURE 9. Results of RS steganalysis for each of the stego-images in Fig. 7. (a) using the 1 × 3 block template; (b) using the 2 × 2 block template; (c) using the 2 × 3 block template; (d) using the 3 × 3 block template.

[14] and [15], so that the embedding rates yielded by the proposed method are higher. The second reason is that the embedding is performed regardless of whether the block is complete or incomplete in the proposed method; however, the methods proposed by [14] and [15] seem to embed messages only in partitioned complete blocks. Therefore, the embedding rates yielded by the proposed method are higher. The proposed method also has higher PSNR values when using the 1 × 3 and 3 × 3 block templates than those respective PSNR values resulting from the methods proposed by [14] and [15]. This is probably because the proposed method performs the falling-off-boundary checks on the pixel pairs before message embedding, and only embeds messages into pixel pairs that do not exceed the boundaries. In contrast, the method proposed by [14] and [15] embeds messages in all pixel pairs, causing the gray-value differences between the selected pixel values by Eqs. (11) and (12) and the original pixel values too large and affecting the overall resulting image quality.

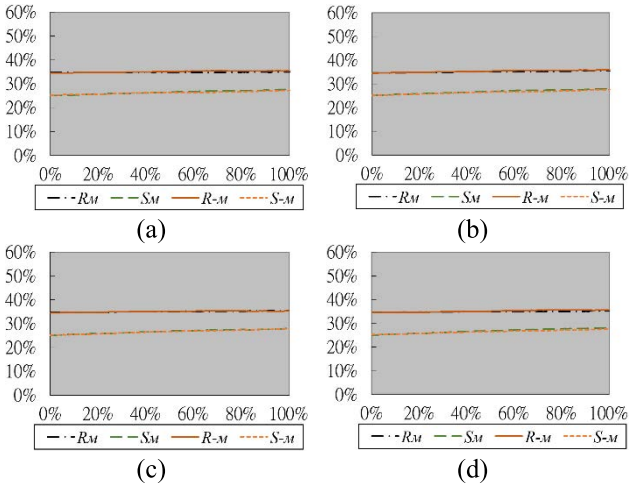

FIGURE 10. Results of RS steganalysis for each of the stego-images in Fig. 8. (a) using the 1 × 3 block template; (b) using the 2 × 2 block template; (c) using the 2 × 3 block template; (d) using the 3 × 3 block template.

Table 8 shows another comparison of the embedding rates and the PSNR values yielded by the embedding processes according to the quantization rage table shown in Table 4 using five methods, including the one proposed in this paper and four ones proposed respectively by Wu and Tsai [8], Wu [16], Khodaei and Faez [14], and Swain [15]. Note that the widths of the quantization ranges used by the proposed method and those proposed by Wu [16] are *not* limited to powers of two while the three methods of [8], [14], [15] on the contrary use quantization range tables with power-of-two ranges *originally*. Therefore, as a unification of the used range widths, the experiment conducted for this comparison has used Table 4 as the quantization range table for all the three methods of [8], [14], [15] where the range widths used in Table 4 are not limited to be powers of two. As shown in Table 8, the embedding rates of the method proposed in this paper are higher than those yielded by the methods proposed in [8], [14]–[16]. When comparing the results coming from using block templates of the same size, the proposed method has 5% to 8% higher embedding rates than the method proposed by [14] and 8% to 13% higher embedding rates than the method proposed by [15]. For example, for the 1 × 3 block template, the proposed method yields the embedding rate of 3.715 with the image of Baboon as the input and the method by Khodaei and Faez [14] yields the embedding rate of 3.413, leading to a promotion of 8.8% created by the proposed method as shown in the following:

$$(3.715 - 3.413)/3.413 = 0.088.$$

In addition, the PSNR values of the stego-images yielded by the proposed method are all above 30.8dB, indicating that the difference of the stego-images from the input images cannot be easily perceived by the human eye.

V. CONCLUSION AND DISCUSSIONS

A modified multiway general pixel-value differencing (MMGPVD) method has been proposed in this study,

which improves the MPVD methods of Khodaei and Faez [14], and Swain [15]. The main improvements include: (1) the block templates include more versions, namely, 1×3, 2×2, 2×3, and 3×3, which are more diverse and can be employed to raise possibly message embedding rates; (2) the partitioned image blocks can be used to embed message bits regardless of whether they are complete or incomplete blocks, allowing more embedding space; (3) a falling-off-boundary check is performed on the pixel pairs before using them to embed secret data, in order to prevent the grayscale values of the pixel pairs from getting out of bounds after message embedding, so that the resulting stego-image qualities are better than those of the results yielded by the methods of [14] and [15]; (4) the widths of the adopted quantization ranges are not limited to be powers of two, which are more flexible than those used by the methods of [14] and [15] which utilize quantization ranges with power-of-two widths. In addition, the proposed method uses a secret key and a random number generator to randomize the bit positions of the secret message bitstream to enhance the security of the hidden message. The experimental results show that the proposed MMGPVD method yields better stego-images than the existing multiway pixel-value differencing methods in the aspects of embedding rate and image quality.

REFERENCES

[1] I. J. Cox, M. L. Miller, J. A. Bloom, J. Fridrich, and T. Kalker, *Digital Watermarking and Steganography*, 2nd ed. Burlington, MA, USA: Morgan kaufmann, 2007.

[2] D. L. Page, *History and the Homeric Iliad*, Berkeley, CA, USA: Univ. California Press, 1976.

[3] C.-C. Chang, "Adversarial learning for invertible steganography," *IEEE Access*, vol. 8, pp. 198425–198435, 2020.

[4] J. Qin, Y. Luo, X. Xiang, Y. Tan, and H. Huang, "Coverless image steganography: A survey," *IEEE Access*, vol. 7, pp. 171372–171394, 2019.

[5] N. Agarwal, A. K. Singh, and P. K. Singh, "Survey of robust and imperceptible watermarking," *Multimedia Tools Appl.*, vol. 78, no. 7, pp. 8603–8633, 2019.

[6] M. Begum and M. S. Uddin, "Digital image watermarking techniques: A review," *Information*, vol. 11, no. 2, p. 110, Feb. 2020.

[7] R.-Z. Wang, C.-F. Lin, and J.-C. Lin, "Image hiding by optimal LSB substitution and genetic algorithm," *Pattern Recognit.*, vol. 34, no. 3, pp. 671–683, 2001.

[8] D.-C. Wu and W.-H. Tsai, "A steganographic method for images by pixel-value differencing," *Pattern Recognit. Lett.*, vol. 24, pp. 1613–1626, Jun. 2003.

[9] C.-K. Chan and L. M. Cheng, "Hiding data in images by simple LSB substitution," *Pattern Recognit.*, vol. 37, no. 3, pp. 469–474, Mar. 2004.

[10] H.-C. Wu, N.-I. Wu, C.-S. Tsai, and M.-S. Hwang, "Image steganographic scheme based on pixel-value differencing and LSB replacement methods," *IEE Proc.-Vis., Image Signal Process.*, vol. 152, no. 5, pp. 611–615, Oct. 2005.

[11] C.-M. Wang, N.-I. Wu, C.-S. Tsai, and M.-S. Hwang, "A high quality steganographic method with pixel-value differencing and modulus function," *J. Syst. Softw.*, vol. 81, no. 1, pp. 150–158, Jan. 2008.

[12] K.-C. Chang, C.-P. Chang, P. S. Huang, and T.-M. Tu, "A novel image steganographic method using tri-way pixel-value differencing," *J. Multimedia*, vol. 3, no. 2, pp. 1–9, Jun. 2008.

[13] A. K. Shukla, A. Singh, B. Singh, and A. Kumar, "A secure and high-capacity data-hiding method using compression, encryption and optimized pixel value differencing," *IEEE Access*, vol. 6, pp. 51130–51139, 2018.

[14] M. Khodaei and K. Faez, "New adaptive steganographic method using least-significant-bit substitution and pixel-value differencing," *IET Image Process.*, vol. 6, no. 6, pp. 677–686, Aug. 2012.

[15] G. Swain, "High capacity image steganography using modified LSB substitution and PVD against pixel difference histogram analysis," *Secur. Commun. Netw.*, vol. 2018, pp. 1–14, Sep. 2018.

[16] D. C. Wu, "A pixel-value differencing technique for images steganography using general quantization ranges," in *Proc. Conf. Inf. Technol. Appl. Outlying Islands (ITOAI)*, Kinmen, Taiwan: ROC, 2021, pp. 1138–1142.

[17] D.-C. Wu and W.-H. Tsai, "Data hiding in images via multiple-based number conversion and lossy compression," *IEEE Trans. Consum. Electron.*, vol. 44, no. 4, pp. 1406–1412, Nov. 1998.

[18] D.-C. Wu and W.-H. Tsai, "Embedding of any type of data in images based on a human visual model and multiple-based number conversion," *Pattern Recognit. Lett.*, vol. 20, no. 14, pp. 1511–1517, Dec. 1999.

[19] S. Geetha, V. Kabilan, S. P. Chockalingam, and N. Kamaraj, "Varying radix numeral system based adaptive image steganography," *Inf. Process. Lett.*, vol. 111, no. 16, pp. 792–797, Aug. 2011.

[20] M. Tang, W. Song, X. Chen, and J. Hu, "An image information hiding using adaptation and radix," *Optik*, vol. 126, no. 23, pp. 4136–4141, 2015.

[21] V. Dimitrov, G. Jullien, and R. Muscedere, *Multiple-Base Number System: Theory and Applications*. Boca Raton, FL, USA: CRC Press, 2017.

[22] J. Fridrich, M. Goljan, and R. Du, "Reliable detection of LSB steganography in color and grayscale images," in *Proc. Workshop Multimedia Secur.*, Ottawa, ON, Canada: New Challenges, 2001, pp. 27–30.

DA-CHUN WU received the B.S. degree in computer science and the M.S. degree in information engineering from Tamkang University, Taipei, Taiwan, in 1983 and 1985, respectively, and the Ph.D. degree in computer and information science from the National Chiao Tung University, Hsinchu, Taiwan, in 1999.

He joined the Faculty of the Department of Information Management, Ming Chuan University, Taipei, in 1987. From 2002 to 2018, he was with the National Kaohsiung First University of Science and Technology, Kaohsiung, Taiwan. From 2010 to 2014, he was the Director of the Library and Information Center, National Kaohsiung First University of Science and Technology, and from 2015 to 2018, he was the Head of the Department of Computer and Communication Engineering. He is currently a Professor with the Department of Computer and Communication Engineering, National Kaohsiung University of Science and Technology, Kaohsiung. His research interests include multimedia security, image processing, and machine learning.

ZONG-NAN SHIH received the B.S. degree from the Department of Communication Engineering, Oriental Institute of Technology, Banqiao, Taiwan, in 2019, and the M.S. degree from the Department of Computer and Communication Engineering, National Kaohsiung University of Science and Technology, Kaohsiung, Taiwan, in 2021. His current research interests include information hiding and mobile applications.

JHENG-HAN WU received the B.S. degree from the Department of Computer Science and Information Engineering, National Taitung University, Taitung, Taiwan, in 2016, and the M.S. degree from the Department of Computer and Communication Engineering, National Kaohsiung University of Science and Technology, Kaohsiung, Taiwan, in 2020. His current research interests include information hiding and system analysis.