Adaptive data hiding scheme based on magic matrix of flexible dimension

Hua Wu[1], Ji-Hwei Horng[2], and Chin-Chen Chang[3,*]
[1] School of Information and Communication Engineering, Beijing Information Science & Technology University, Beijing 100101, China
[e-mail: sunshinesmilewhh@126.com]
[2]Department of Electronic Engineering, National Quemoy University, Kinmen 89250, Taiwan
[e-mail: horng@email.nqu.edu.tw]
[3]Department of Information Engineering and Computer Science, Feng Chia University, Taichung 40724, Taiwan
[e-mail: alan3c@gmail.com]
[*]Corresponding Author : Chin-Chen Chang

Received April 25, 2021; revised July 14, 2021; accepted August 29, 2021; published September 30, 2021

Abstract

Magic matrix-based data hiding schemes are applied to transmit secret information through open communication channels safely. With the development of various magic matrices, some higher dimensional magic matrices are proposed for improving the security level. However, with the limitation of computing resource and the requirement of real time processing, these higher dimensional magic matrix-based methods are not advantageous. Hence, a kind of data hiding scheme based on a single or a group of multi-dimensional flexible magic matrices is proposed in this paper, whose magic matrix can be expanded to higher dimensional ones with less computing resource. Furthermore, an adaptive mechanism is proposed to reduce the embedding distortion. Adapting to the secret data, the magic matrix with least distortion is chosen to embed the data and a marker bit is exploited to record the choice. Experimental results confirm that the proposed scheme hides data with high security and a better visual quality.

Keywords: Data hiding, embedding efficiency, flexible dimension, magic matrix

1. Introduction

With the rapid development of the Internet, the data hiding is referred to as a usual approach of embedding confidential or sensitive information into images for covert communications. It is classified into two major types: reversible data hiding and irreversible data hiding. The difference between them is that the reversible hiding scheme can reconstruct the original cover image after secret data is extracted. Many successful reversible data hiding schemes have been proposed [1-7], including the difference expansion [3,4], the prediction error expansion (PEE) [5,6], the histogram shifting [8,9], the neural network [10-12] and so on. For instance, the prediction error expansion is first proposed by Thodi et al. [13], which embeds the secret information by expanding the prediction errors obtained from the difference of original and predicted value of the target pixel. To further minimize the distortion, Ou et al. [14] proposed a method of pairwise PEE for data hiding, which simultaneously uses a pair of prediction errors obtained from consecutive pixels and modifies the corresponding pixel values using 2D histogram modification strategy. Li et al. [15] introduced a multiple histograms modification into data hiding, which is a more general framework and includes the conventional one as the special case. In addition, they verified that multiple histograms modification (MHM) embeds data with higher fidelity, and overall embedding on multiple prediction error histograms (PEHs) could be flexible by determining the target bins adaptively. However, only one pair of bins in the PEH is allowed to be expanded for data embedding according to the MHM. For this reason, an improved data hiding scheme based on MHM with high capacity is proposed, which utilized multiple pairs of bins for expansion instead of one pair in each PEH [16].

The researchers on irreversible data hiding also achieves remarkable results [17-25], including the least significant bit (LSB) substitution [17,19], the exploiting modification direction (EMD) [20,21], and the magic matrix based (MMB) schemes [23,24], and so on. In LSB substitution method, LSBs of a cover pixel are replaced by secret binary bits of the same length. However, this method is vulnerable to steganalysis. In recent decades, many data hiding methods based on different kinds of magic matrices have been proposed [26-29]. Chang et al. [23] introduced Sudoku table to construct a 256×256 Sudoku matrix for guiding the embedding process. This scheme provides a higher security than LSB substitution, because a great number of possible Sudoku solutions can prevent from malicious attacks. Chen et al. [26] proposed a multi-layer mini-Sudoku based data hiding scheme to achieve the better visual quality for a given payload by the optimal number of layers of magic matrices. In 2014, Chang et al. [24] proposed a data hide method of Turtle Shell-based (TSB) scheme with a 256×256 turtle shell matrix that is composed of a lot of adjacent turtle shells. According to this method, a secret 8-ary digit can be embedded into each pixel pair of the original image. Liu et al. [27] developed a new turtle shell matrix-based data hiding scheme to improve the embedding capacity. Xie et al. [28] proposed an extended method of two-layer turtle shell matrix-based data hiding with an extra attribute represented by a 4-ary digit, which is assigned to each element of the turtle shell matrix by exploiting symmetric properties of the matrix. In 2020, a data hiding method based on a 3D magic cube [30] was proposed, which can hide more secret data and be more unpredictable than the conventional versions [31-33]. Generally, it is more difficult to predict the embedded secret data for hiding methods with a magic matrix of high dimensions, since a high dimensional magic matrix is more complex than a low dimensional one. However, the usual magic matrix-based data hiding methods construct the whole reference matrix sized 256 in each dimension before embedding and extracting information,

which occupies too much memory space. In some applications, such as an app for portable devices, there is not enough memory for building the whole matrix. For example, a 16G memory processing device does not have enough memory to construct 4-dimensional matrix sized 256×256×256×256. Hence, a flexible design of the magic matrix is required, which should take both computational complexity and the adaptive mechanism into consideration.

In this research, we try to devise a magic matrix with flexible dimensions so that the security level under steganalysis attack can be raised. Considering the real time applications, a data hiding scheme based on a magic matrix of reduced scales is proposed to improve the computational efficiency and save the memory space. Besides, an adaptive strategy is exploited to minimize the distortion of data embedding. Adapting to the secret data in hand, the magic matrix with a minimum embedding distortion is always chosen between two predefined selections. Thus, the visual quality of stego images can be effectively enhanced.

The rest of the paper is organized as follows. Section 2 discusses the related work. The proposed scheme is introduced in Section 3. Section 4 presents the experimental results and their analysis. Finally, the paper is concluded in Section 5.

2. Related works

This section briefly reviews some data hiding methods based on different reference matrices, including the data hiding methods based on Sudoku [23], mini-Sudoku [31,32], and a 3D magic cube [30].

2.1 Chang et al.'s Scheme

Chang et al. [23] proposed a Sudoku-based data hiding scheme, which has three levels of elements, such as a 256×256 Sudoku matrix, 9×9 Sudoku tables and 3×3 sub-blocks. A basic 9×9 Sudoku table is constructed by nine 3×3 sub-blocks according to the following two rules: (1) each sub-block contains nine distinct digits from 0 to 8; and (2) each column or row of the table contains the same set of digits. Furthermore, a 256×256 Sudoku reference matrix is tiled by Sudoku tables for guiding the embedding of secret data. First, the binary secret information is converted to a sequence of 9-ary digits. Second, the values of selected cover pixel pair are modified to embed a secret digit through the guidance of Sudoku reference matrix. The embedding rate of this scheme is $1/2 \times \log_2 9$ bits per pixel (bpp).

2.2 He et al.'s Scheme

He et al. [31] proposed a data hiding scheme based on mini-Sudoku matrix. The major difference between conventional Sudoku and the proposed one is that mini-Sudoku uses repetitions of a 4×4 Sudoku table to tile up a 256×256 matrix, with the values from 0 to 3. In this method, each cover pixel pair (p_i, p_{i+1}) embeds 4-bit secret data, which is divided into three groups. B_1 is a quaternary digit, while both B_2 and B_3 are a binary bit. By applying the cover pixel pair (p_i, p_{i+1}) as the coordinates, a 4×4 search region on the mini-Sudoku matrix is located. Within the search region, a pixel pair (p'_i, p'_{i+1}) satisfying $MSM(p'_i, p'_{i+1}) = B_1$, $p'_i = B_2$ and $p'_{i+1} = B_3$ can be found. Finally, the 4-bit secret data is embedded by modifying the cover pixel pair (p_i, p_{i+1}) into the stego pixel pair (p'_i, p'_{i+1}). The extraction stage is the inverse approach as that is used in the embedding one.

2.3 Horng et al.'s Scheme

Inspired by He et al. [31], Horng et al. [32] proposed a multi-dimensional mini-Sudoku

matrix-based data hiding scheme, which expands the magic matrix into multi-dimensions. The cubic mini-Sudoku matrix is constituted by 4×4×4-sized sub-cubes, each contains eight 2×2×2-sized basic structures with values from 0 to 7. In the embedding stage, each pixel-triplet (p_i, p_{i+1}, p_{i+2}) is used to embed 6-bit secret s_i=(d_i, d_{i+1}, d_{i+2}, d_{i+3}, d_{i+4}, d_{i+5}) in two layers. First, the outer layer points out the matched basic structure, whose label is s^M = (d_i, d_{i+1}, d_{i+2}), by the first three bits of secret with an offset matrix and several locator conditions. Then, the inner layer indicates the value in the obtained basic structure, which is equal to the later three bits of secret s^L=(d_{i+3}, d_{i+4}, d_{i+5}). Finally, the stego pixels are obtained as $(p'_i, p'_{i+1}, p'_{i+2})$. The extraction stage also is the inverse approach as that is used in the embedding one.

2.4 Lee et al.'s Scheme

Lee et al. [30] proposed a data hiding scheme based on a 3D magic cube. In this method, a 256×256×256 3D reference matrix (RM) is established before hiding secret data as shown in **Fig. 1**. A 3D magic cube sized $n_x \times n_y \times n_z$ is designed first. For instance, a 4×4×4 magic cube contains 64 distinct digits from 0 to 63. Then, a 3D RM denoted by $M(x, y, z)$, where $0 \leq (x, y, z) \leq 255$, is constructed by repeated stacking of the fundamental magic cube. According to the 3D RM, the embedding process and extracting process can be carried out. Details are summarized as follows.

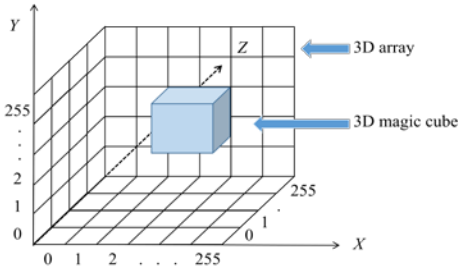


Fig. 1. 3D array and 3D magic cube

2.4.1 The data embedding process

The details of secret data embedding are as follows:

Step 1. The cover image is divided into 2×2 non-overlapping blocks. Each block has four cover pixels (p_1, p_2, p_3, p_4).

Step 2. A k_1-bit data s_1 is extracted from the binary secret stream S and embedded into the first pixel by LSB substitution. The pixel value p_1 is thus modified to q_1.

Step 3. Calculate the three absolute differences by $d_1 = |p_2 - q_1|$, $d_2 = |p_3 - q_1|$, $d_3 = |p_4 - q_1|$.

Step 4. A k_2-bit data s_2 is extracted from the remaining binary secret stream. Search the reference matrix to find the element $M(d'_1, d'_2, d'_3)$ which is closest to (d_1, d_2, d_3) and matches $M(d'_1, d'_2, d'_3) = s_2$. The embedding rules are as follows.

Case 1. If $s_2 = M(d_1, d_2, d_3)$, then the cover pixels are unchanged.

Case 2. If $s_2 \neq M(d_1, d_2, d_3)$, then find all $M(d'_{1i}, d'_{2i}, d'_{3i}) = s_2$,where $d'_{1i} \in (\lfloor d_1 \rfloor - 7 : \lfloor d_1 \rfloor + 7)$, $d'_{2i} \in (\lfloor d_2 \rfloor - 7 : \lfloor d_2 \rfloor + 7)$ and $d'_{3i} \in (\lfloor d_3 \rfloor - 7 : \lfloor d_3 \rfloor + 7)$.

(i) The target with minimum distance is obtained by $= arg \min(|d'_{1i} - d_1|^2 + |d'_{2i} - d_2|^2 + |d'_{3i} - d_3|^2)$.

(ii) The stego image pixel values are calculated as $q_2 = q_1 + d'_{1k}$, $q_3 = q_1 + d'_{2k}$ and $q_4 = q_1 + d'_{3k}$.

(iii) The cover pixel values (p_1, p_2, p_3, p_4) are modified to the stego pixel values (q_1, q_2, q_3, q_4).

The above steps are repeated until all the secret data is embedded.

2.4.2 The data extraction process

The corresponding secret data extraction process is as follows. The stego image is first divided into 2×2-sized blocks. For each stego block of four pixels (q_1, q_2, q_3, q_4), the absolute differences (d'_1, d'_2, d'_3) between q_1 and the other three pixels (q_2, q_3, q_4) are calculated. The data extraction includes two parts. The secret data s_2 is extracted by $s_2 = M(d'_1, d'_2, d'_3)$ and the secret data s_1 is extracted from LSBs of q_1. The overall secret data stream is obtained by consecutively concatenating the extracted the secret bits from each stego block.

Generally, the conventional magic matrix-based data hiding schemes exploit a reference matrix sized 256 in each dimension, which wastes too much memory space to process. It becomes very difficult to implement when the reference matrix is extended to high dimensions.

3. Proposed scheme

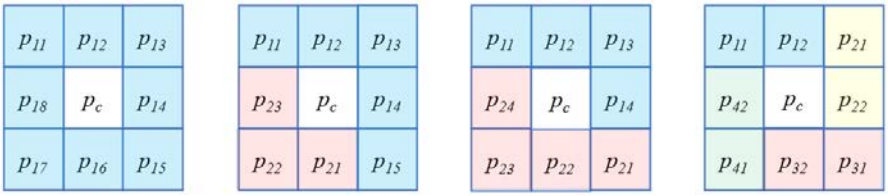
This section introduces the proposed multi-dimensional flexible magic matrix-based data hiding scheme, which extends the usual 2D and 3D magic matrix to multi-dimensional space, but with a much smaller size. For this reason, the proposed matrix can be easily switched between different dimensions during the data hiding phase, which also improves security level of the proposed scheme.

3.1 Flexible magic matrix and magic unit

Before presenting the data hiding scheme, the proposed flexible magic matrix is introduced first. Our flexible magic matrix is a reduced version of reference matrix, which is constituted by basic magic units.

3.1.1 Dimensions of magic unit

In our data hiding scheme, the cover image is also divided into non-overlapping blocks. In each block, the center cover pixel is defined as the basic pixel p_c and the other pixels around it are defined as surrounding pixels $(p_1, p_2, \dots, p_N)$. The surrounding pixels are divided into n groups denoted by $(p_{i1}, p_{i2}, \dots, p_{ij}, .., p_{ik_i})$, where the i-th group consists of k_i pixels and $\sum_{i=1}^{n} k_i = N$. For example, in a 3×3 block, the basic pixel p_c is located at center of the block, which is surrounded by eight pixels $(p_1, p_2, p_3, p_4, p_5, p_6, p_7, p_8)$. Then, the surrounding pixels can be grouped in different ways, such as two equally sized groups, four equally sized groups, or combination of unequally sized groups, as shown in **Fig. 2**. According to the number of pixels in a group, a magic matrix with the same number of dimensions is applied as its reference matrix. The proposed flexible magic matrix is a hyper-cubic matrix of reduced size, which is twofold stacked by fundamental units called magic units. For example, to embed six bits of data in a group of three pixels, the flexible magic matrix is a cubic matrix sized 8×8×8 constituted by eight magic units sized 4×4×4.



(a) one group (b) two unequal groups (c) two equal groups (d) four equal groups

Fig. 2. One basic pixel and its eight neighboring pixels.

3.1.2 Scales of magic unit

As mentioned in the previous subsection, the magic matrix is a twofold stacked matrix of magic units. The scales of a fundamental magic unit is determined by the number of pixels in the group and the designated payload for each pixel. Assuming a particular group consists of k pixels and the payload for the j-th pixel is s_j, its corresponding magic unit is a k-dimensional hyper cubic matrix. The width $r_j, j = 1,2, \dots, k$, in the j-th dimension is determined by

$$r_j = 2^{s_j}. \tag{1}$$

In a magic unit, each element is assigned with a distinct value. The magic unit contains a set of distinct integer values from 0 to V_{max}, which are assigned randomly. V_{max} is the largest value in this unit calculated by the following equation.

$$V_{max} = 2^{\sum s_j} \text{-}1, \tag{2}$$

If the payload for all pixels in the group is the same, then $s_1 = s_2 = \cdots = s_k = s_{eq}$ and the magic unit is a hypercube with width $r_j = 2^{s_{eq}}$ in each direction of axis. V_{max} is given by

$$V_{max} = 2^{k \times s_{eq}} - 1. \tag{3}$$

For example, in **Fig. 2(d)**, suppose a group of two pixels is designated to embed 2-bit secret data for each pixel, its corresponding flexible magic matrix is sized 8×8 and stacked with four magic units, as shown in **Fig. 3**. The largest element value 15 is calculated by Eq. (3).

3	10	6	2	12	10	6	2	12
2	4	11	15	9	4	11	15	9
1	3	5	0	7	3	5	0	7
0	1	14	8	13	1	14	8	13
-1	10	6	2	12	10	6	2	12
-2	4	11	15	9	4	11	15	9
-3	3	5	0	7	3	5	0	7
-4	1	14	8	13	1	14	8	13
	-4	-3	-2	-1	0	1	2	3

Magic unit

Magic matrix

Fig. 3. Example of a 2D magic matrix with its magic units.

3.2 Data embedding phase

In this subsection, the embedding procedure of the proposed scheme is introduced. Then, details of the flexible magic matrix-guided embedding steps are presented.

3.2.1 Data embedding procedures

The data embedding procedures of the proposed scheme is given as follows.

Step 1. The given cover image I of bit-depth 8 is divided into non-overlapping blocks sized $h \times w$.

Step 2. For each block, the center pixel is labeled as p_c and the surrounding pixels are grouped according to the predefined hiding strategy.

Step 3. s_c-bit secret segment is extracted from the binary secret stream S. Apply LSB substitution and optimal pixel adjustment process (OPAP) to embed the retrieved s_c-bit secret segment into the center pixel and to get the resulting stego pixel q_c.

Step 4. The neighboring pixels are embedded group-wise. The i-th group of the surrounding pixels $(p_{i1}, p_{i2}, \dots, p_{ik})$ is used for hiding secret data s_i by referring to a k dimensional flexible magic matrix.

Step 5. Repeat Steps 2-4 until the entire secret stream is embedded, then the stego image I' is obtained.

3.2.2 Details of the pixel group embedding

The details of Step 4 in the data embedding procedures are presented as follows.

Step 1. A group of neighboring pixels $(p_1, p_2, \dots, p_k)$ are exploited to embed secret data and their corresponding payloads are denoted by $s_j, j = 1,2, \dots, k$. The width r_j in the j-th dimension of the k-dimensional magic unit is determined by s_j and given by Eq. (1). The width of flexible magic matrix in the j-th dimension is twice as r_j and its index interval is $[-r_j, r_j - 1]$.

Step 2. The differences d_j between neighboring pixel $p_j, j = 1,2, \dots, k$, and stego center pixel q_c are calculated by $d_j = p_j - q_c, j = 1,2, \dots, k$.

Step 3. Normalize the differences into coordinates $\tilde{d}_j, j = 1,2, \dots, k$, of the flexible magic matrix M_k by the following rules, where m_j is an integer that makes Eq. (5) satisfied.

$$\tilde{d}_j = d_j - m_j \times r_j, \tag{4}$$

$$-\frac{r_j}{2} \le \tilde{d}_j \le \frac{r_j}{2} - 1. \tag{5}$$

Step 4. $\sum s_j$ bits of secret data S_i is extracted from the secret stream S and converted to a value V in decimal format. Find the unique element $M_k(d'_1, d'_2, \dots, d'_k)$ that satisfies Eqs. (6) and (7).

$$M_k(d'_1, d'_2, \dots, d'_k) = V, \tag{6}$$

$$\tilde{d}_j - \frac{r_j}{2} \le d'_j \le \tilde{d}_j + \frac{r_j}{2} - 1. \tag{7}$$

Then, calculate the stego pixel values by

$$q_j = q_c + d'_j + m_j \times r_j. \tag{8}$$

Step 5. In the cases of overflow/underflow. The stego pixel value is added/subtracted by r_{ij} to fit the proper dynamic range of [0,255].

Step 6. Finally, the group of cover pixel values $(p_1, p_2, \dots, p_k)$ are modified to the stego-pixel values $(q_1, q_2, \dots, q_k)$.

3.3 Data extraction phase

The basic execution framework of data extraction procedures is similar to that of embedding

procedures. First, the stego-image is divided into non-overlapping blocks with the same size as embedding. For each image block, the center pixel q_c is retrieved, and its surrounding pixels are grouped in the same way as embedding. Then, the secret data is extracted group by group. Finally, the secret data embedded in the center pixel is extracted. The output is a concatenation of the extracted secret segments. Data extraction procedures of an image block is summarized as follows.

Step 1. Retrieve the center pixel value q_c.

Step 2. For the i-th group of pixels $(q_1, q_2, \dots, q_k)$, calculate $(\hat{d}_1, \hat{d}_2, \dots, \hat{d}_k)$ by $\hat{d}_j = q_j - q_c$, j=1,2,…,k.

Step 3. Find the normalized coordinates $(\tilde{d}_1, \tilde{d}_2, \dots, \tilde{d}_k)$ by Eqs. (4) and (5).

Step 4. Extract secret data by $V = M_k(\tilde{d}_1, \tilde{d}_2, \dots, \tilde{d}_k)$ and convert V into binary digits S_i.

Step 5. Repeat Step 2-4 to extract secret data from all groups of pixels.

Step 6. Extract secret digits S_c from LSBs of q_c.

Step 7. Concatenate $S_c \,\|\, S_1 \,\|\, S_2, \dots, \|\, S_n$.

3.4 An example of multi-dimensional flexible magic matrix-based data hiding

In this subsection, an example is provided to demonstrate the embedding and extracting procedures. In addition, a special case of overflow is also demonstrated. In **Fig. 4**, an example image block sized 3×3 is given. p_c is the center pixel and its surrounding pixels $(p_1, p_2, \dots, p_8)$ are divided into four groups distinguished by different colors. The binary secret steam is given by $S = (10010001010)_2$.

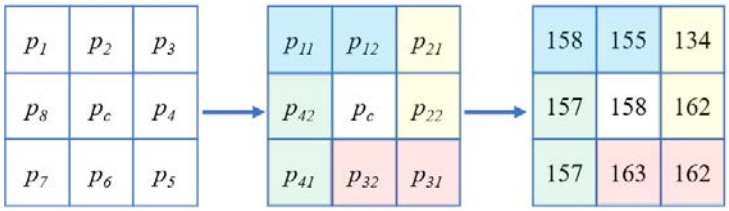


Fig. 4. A block including four groups of neighboring pixels.

(1) 3-bit secret segment $S_c = (100)_2$ is extracted from the head of the binary secret stream S. The value of basic pixel p_c is converted into 8-bit binary sequence as $158 = (10011110)_2$, whose 3 LSBs are replaced by $(100)_2$ to get $q_c = (10011100)_2 = 156$ as shown in **Fig. 5**.

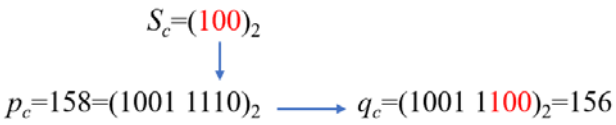


Fig. 5. Illustration of the center pixel embedding.

(2) The first group with two pixels (p_{11}, p_{12}) is used to embed the following 4 bits secret data $S_1 = (1000)_2 = 8$. First, the differences (d_{11}, d_{12}) are calculated between pixels (p_{11}, p_{12}) and q_c by $d_{11} = p_{11} - q_c = 158 - 156 = 2$ and $d_{12} = p_{12} - q_c = 155 - 156 = -1$. The coordinates $(2, -1)$ is normalized to $(-2, -1)$ by $(2 - 1 \times 4, -1 - 0 \times 4)$. Since $k = 2$, a 2D flexible magic matrix is applied as the reference matrix for embedding. Search M_2 within the index range of $(-2 - 2{:}-2 + 1, -1 - 2{:}-1 + 1)$ to find the unique solution $M_2(-2,0) = S_1 = 8$. Finally, the stego pixels (q_{11}, q_{12}) are calculated through $q_{11} = q_c +$

$d'_{11} + m_{11} \times r_{11} = 156 - 2 + 1 \times 4 = 158$ and $q_{12} = q_c + d'_{12} + m_{12} \times r_{12} = 156 + 0 + 0 \times 4 = 156$.

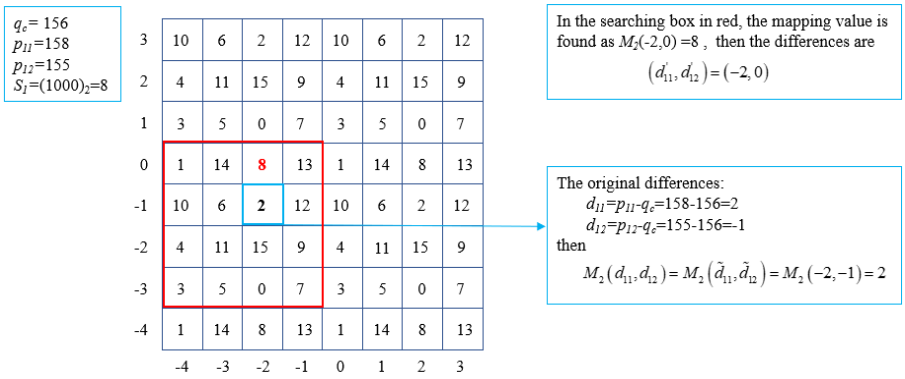


Fig. 6. Embedding process of the first pixel group.

(3) The second group with two pixels (p_{21}, p_{22}) are used for embedding the following secret data $S_2 = (1010)_2 = 10$. Similarly, the differences (d_{21}, d_{22}) are calculated between (p_{21}, p_{22}) and q_c by $d_{21} = p_{21} - q_c = 134 - 156 = -22$ and $d_{22} = p_{22} - q_c = 162 - 156 = 6$. Then, the coordinates are normalized by $(\tilde{d}_{21}, \tilde{d}_{22}) = (-22 - (-5) \times 4, 6 - 2 \times 4) = (-2, -2)$. To embed the secret value 10, the search range corresponding to the current reference location is enclosed by red square as shown in **Fig. 7**. The unique matched element is $M_2(-4, -1) = 10$. As a result, the stego pixel values are calculated by $q_{21} = 156 - 4 + (-5) \times 4 = 132$ and $q_{22} = 156 - 1 + 2 \times 4 = 163$.

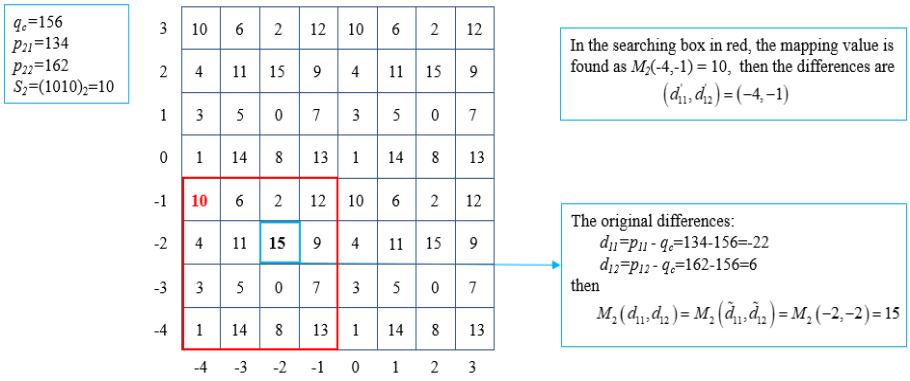


Fig. 7. Embedding process of the second pixel group.

Now, we demonstrate the secret data extraction procedures for the stego image block produced in the above embedding example.

Step 1. S_c is extracted from the 3 LSBs of q_c to get $(100)_2$.

Step 2. For the first group, the differences $(\hat{d}_{11}, \hat{d}_{12})$ between two neighboring pixels (q_{11}, q_{12}) and q_c are calculated by $\hat{d}_{11} = q_{11} - q_c = 158 - 156 = 2$ and $\hat{d}_{12} = q_{12} - q_c = 156 - 156 = 0$. Normalized $(\hat{d}_{11}, \hat{d}_{12})$ into $(\tilde{d}_{11}, \tilde{d}_{12})$ by $(\tilde{d}_{11}, \tilde{d}_{12}) = (2 - 1 \times 4, 0 - 0 \times 4) = (-2,0)$ Then, according to the 2D flexible magic matrix $M_2(-2,0) = 8$. Hence, the value 8 is the secret data, which is converted to 4-bit binary segment $S_1 = (1000)_2$.

Step 3. The final secret segment can be extracted from pixel group (q_{21}, q_{22}) to get $S_2 = (1010)_2$.

Step 4. TheS_c, S_1 and S_2 are concatenated into $S = (10010001010)_2$.

(4) Sometimes, direct application of the proposed embedding scheme may result in overflow (greater than 255) or underflow (less than 0) of pixel values. Now, we demonstrate how Step 5 of the pixel group embedding can solve this problem. Suppose $q_c = 254$, $(p_{11}, p_{12}) = (253,255)$, and the secret bits to be embedded $S_1 = (1001)_2 = 9$. Thus, $(d_{11}, d_{12}) = (\tilde{d}_{11}, \tilde{d}_{12}) = (-1,1)$ and $(m_{11}, m_{12}) = (0,0)$. Referring to **Fig. 8**, the unique solution within the search range is $M_2(-1,2) = 9$. Then, (q_{11}, q_{12}) are calculated by $(q_{11}, q_{12}) = (254 + (-1), 254 + 2) = (253,256)$. The second pixel value is overflowed. According to Step 5 of pixel group embedding, it is subtracted by r_{ij} to get $q_{12} = 256 - 4 = 252$. In the extraction phase, $(q_{11}, q_{12}) = (253,252)$ is applied to calculate $(\tilde{d}_{11}, \tilde{d}_{12}) = (-1, -2)$ and the secret value $M_2(-1, -2) = 9$. Due to repetition of magic units, adding or subtracting r_{ij} from the coordinates of magic matrix does not affect the extracted value.

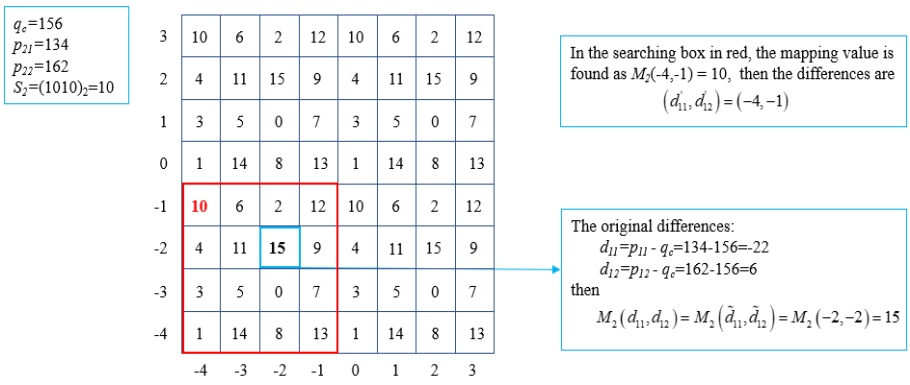


Fig. 8. The example to the q_{ij} needs fine-turning.

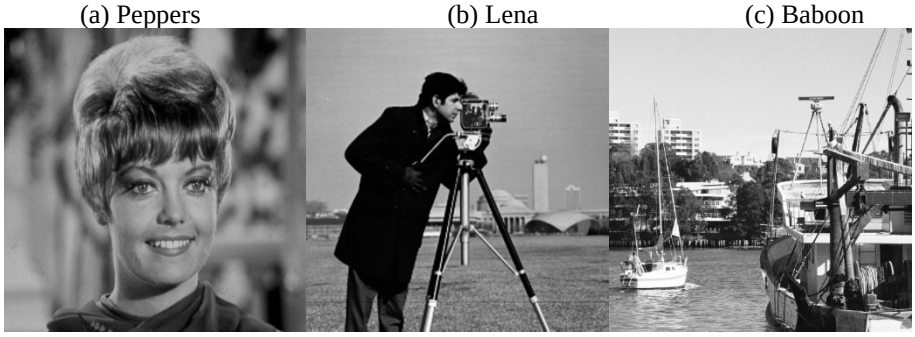
3.5 Adaptive mechanism

We design an additional adaptive mechanism to improve the visual quality of stego images. The idea is to embed secret data in a cover image with two magic matrices of different dimensions. For each image block and its corresponding secret data to be embedded, the distortions on pixel values for the two magic matrices are compared and the better solution is applied. To indicate the actual applied version of magic matrix, a certain bit in the center pixel is exploited as the marker bit to record the choice. This adaptive mechanism can effectively enhance the visual quality of stego images.

4. Experimental results and security analysis

In this section, several experiments are executed to demonstrate the applicability and evaluate the performance of the proposed scheme. Six standard gray scale images, including Peppers, Lena, Baboon, Zelda, Cameramen, and Harbour, as shown in **Fig. 9** are applied. The peak signal-to-noise ratio (PSNR) and embedding capacity (EC) are used as metrics to evaluate the performance.

(a) Peppers (b) Lena (c) Baboon



(e) Zelda (f) Cameraman (g) Harbour

Fig. 9. Six standard gray level test images.

PSNR represents the distortion between the original cover image and the stego image and defined by

$$PSNR = 10\log_{10}\left(\frac{255^2}{MSE}\right)(dB), \tag{9}$$

where *MSE* is the mean square error between the cover image and its stego image, which can be calculated by

$$MSE = \frac{1}{H\times W}\sum_{i=1}^{H}\sum_{j=1}^{W}\left(p_{ij}, q_{ij}\right)^2, \tag{10}$$

where p_{ij} and q_{ij} are the pixel values of the cover image and the stego image, respectively; H and W are the height and the width of the cover image, respectively.

The embedding capacity (EC) is measured in bits per pixel and defined by

$$EC = \frac{|L|}{H\times W}(bpp), \tag{11}$$

where $|L|$ is the length of the secret data under full embedding and the *bpp* is the abbreviation of the bits per pixel.

4.1 Performance of four typical parameter settings

In our first experiment, we apply four models to investigate the performance of the proposed data hiding scheme. In all the four models, the center pixel is embedded with 3 bits (s_c) of secret data and the surrounding pixels are embedded with 2 bits per pixel (s_e). The different settings in block size and grouping are listed in **Table 1**. For example, Model 3 divides cover image into blocks sized 3 × 3 and the surrounding pixels are divided into four groups of two pixels. According to the number of pixels in a group and the payload per pixel, the width of the magic unit and the range of matrix element values can be determined as listed in the table. The embedding capacity (EC) for the four models are also given, which can be calculated using block size, s_c, and s_e.

Table 1. Various kinds of hiding strategy for the proposed scheme.

Model	Block size	s_c bits	Group	k	s_e bits	Width	Values	EC(*bpp*)
1	2×2	3	1	3	2	4	0-63	2.25
2	2×3	3	1	5	2	4	0-1023	2.17
3	3×3	3	2	4	2	4	0-255	2.11
4	3×3	3	4	2	2	4	0-15	2.11

To further evaluate the performance of the four typical models under different arrangement of distinct numbers in a magic unit, we apply three randomly arranged versions Secret 1, Secret 2, and Secret 3 to the experiments. The PSNR values of the resulting stego images are listed in **Table 2**, where the experimental values of state-of-the-art method proposed by Lee et al. [30] are also given. The EC of Model 1 is the same as [30] and the PSNR values of the two model are comparable. PSNR values of Model 2, 3, and 4 are higher than Model 1, however, their EC are slightly lower. There is a trade-off between PSNR value and EC. Besides, experimental data show that the arrangement of distinct numbers in a magic unit does not affect the result under random embedded data.

Table 2. Comparison of the four models with Lee et al.'s method [30]

Model	Secret	Image					
		Peppers	Lena	Baboon	Zelda	Cameraman	Harbour
Model 1	Secret1	43.30	43.23	43.28	43.27	43.26	43.26
	Secret2	43.30	43.23	43.29	43.28	43.27	43.26
	Secret3	43.31	43.23	43.29	43.29	43.26	43.27
	Average	43.30	43.23	43.29	43.28	43.26	43.26
Model 2	Secret1	44.08	44.07	44.09	44.07	44.09	44.09
	Secret2	44.09	44.06	44.09	44.07	44.09	44.10
	Secret3	44.09	44.07	44.09	44.07	44.10	44.09
	Average	44.09	44.07	44.09	44.07	44.09	44.09
Model 3	Secret1	44.75	44.70	44.73	44.71	44.69	44.72
	Secret2	44.75	44.69	44.73	44.71	44.71	44.73
	Secret3	44.75	44.70	44.73	44.71	44.71	44.72
	Average	**44.75**	**44.70**	**44.73**	**44.71**	44.70	**44.72**
Model 4	Secret1	44.75	44.69	44.73	44.70	44.70	44.72
	Secret2	44.74	44.70	44.74	44.70	44.71	44.73
	Secret3	44.75	44.69	44.73	44.71	44.72	44.72
	Average	**44.75**	44.69	**44.73**	44.70	**44.71**	**44.72**
[30]	Secret1	43.21	43.13	43.24	43.17	43.13	43.18
	Secret2	43.17	43.09	43.22	43.13	43.08	43.14
	Secret3	43.18	43.09	43.22	43.13	43.07	43.15
	Average	43.19	43.10	43.23	43.14	43.09	43.16

4.2 Three adaptive methods based on 2D&4D flexible magic matrices

In this subsection, three adaptive methods based on 2D&4D flexible magic matrices are introduced to improve security of reference matrix-based data hiding scheme. To provide more flexibility, the cover image is divided into blocks of size 3×3 in all the three adaptive methods. In addition, the payload is fixed at two bits per pixel to compare the performance of different methods. When marker bits are required, the payload of the center pixel is sacrificed to meet the requirement.

4.2.1 Adaptive method based on 2D&4D flexible magic matrices without marker bit

To interpret the design of the center pixel's data structure, the center pixel value is represented by binary format as $p_c = (b_7b_6b_5b_4b_3b_2b_1b_0)_2$. In the first method, two secret bits $(\beta_1\beta_0)_2$ are embedded to p_c by applying LSB substitution and OPAP. That is, $q_c = \mathcal{O}((b_7b_6b_5b_4b_3b_2\beta_1\beta_0)_2)$, where $\mathcal{O}(\cdot)$ denotes the OPAP operation. The third LSB b_2 is served as the natural marker bit and the embedding strategy is determined by it. More specifically, when $b_2 = 0$, the surrounding pixels of this block are partitioned into (4,4) groups to embed secret data; when $b_2 = 1$, they are partitioned into (2,2,2,2) groups. In addition, the Data 1, Data 2, and Data 3 are three different random streams. The extraction of secret data is also according to the third LSB b_2. The PSNR values of stego-images are listed in **Table 3**. Due to the random selection of hiding strategy, the security of data embedding can be enhanced.

Table 3. Adaptive method based on 2D&4D matrices without marker bit (CM1).

	Block size	s_c bits	s_e bits	Data 1	Data 2	Data 3
Peppers	3×3	2	2	46.21	46.25	46.23
Lena	3×3	2	2	46.23	46.24	46.24
Baboon	3×3	2	2	46.22	46.25	46.25
Zelda	3×3	2	2	46.23	46.26	46.23
Cameraman	3×3	2	2	46.23	46.26	46.25
Harbour	3×3	2	2	46.23	46.23	46.24

4.2.2 Adaptive method based on 2D&4D flexible magic matrices with one marker bit

In the second method, two LSBs of the center pixel is embedded with one secret bit β_0 and one marker bit α_0 as $q_c = \mathcal{O}((b_7b_6b_5b_4b_3b_2\beta_0\alpha_0)_2)$. Two versions of stego center pixel are generated, which are $q_{c0} = \mathcal{O}((b_7b_6b_5b_4b_3b_2\beta_00)_2)$ and $q_{c1} = \mathcal{O}((b_7b_6b_5b_4b_3b_2\beta_01)_2)$. Then, two partition strategies are applied to embed the secret data into this block. Based on q_{c0}, the (4,4) grouping is applied; based on q_{c1}, the (2,2,2,2) grouping is applied. After embedding of the surrounding pixels, MSE (mean-square-error) of the two stego blocks are compared. The minimum distorted block is recorded to the stego image. PSNR values of the stego-images produced by this method are listed in **Table 4**. Compared with Table 3, this adaptive method can significantly improve the visual quality of stego image.

Table 4. Adaptive method based on 2D&4D matrices with one marker bit (CM2).

	Block size	s_c bits	s_e bits	Data 1	Data 2	Data 3
Peppers	3×3	2	2	47.11	47.13	47.11
Lena	3×3	2	2	47.11	47.12	47.10
Baboon	3×3	2	2	47.13	47.12	47.12
Zelda	3×3	2	2	47.13	47.12	47.11
Cameraman	3×3	2	2	47.10	47.11	47.10
Harbour	3×3	2	2	47.11	47.11	47.12

4.2.3 Adaptive method based on 2D&4D flexible magic matrices with two marker bits

In third adaptive method, the center pixel is embedded with two marker bits $(\alpha_1\alpha_0)_2$ as $q_c = \mathcal{O}((b_7b_6b_5b_4b_3b_2\alpha_1\alpha_0)_2)$. This time, four versions of center pixel are generated. The four

values of marker bits $(00)_2$, $(01)_2$, $(10)_2$, $(11)_2$ are applied to indicate the groupings (4,4), (2,2,2,2), (4,2,2), and (2,4,2), respectively. Again, the MSE of different groupings are compared to determine the optimal solution and the corresponding version of stego block is recorded to the output stego image. PSNR values of this method are listed in **Table 5**. As shown in the table, the image quality can be further improved. However, fewer secret data is embedded since the embedding room of the center pixel is sacrificed to record the marker bits.

Table 5. Adaptive method based on 2D&4D matrices with two marker bits (CM3).

	Block size	s_c **bits**	s_e **bits**	**Data 1**	**Data 2**	**Data 3**
Peppers	3×3	2	2	48.10	48.10	48.10
Lena	3×3	2	2	48.10	48.08	48.10
Baboon	3×3	2	2	48.10	48.10	48.09
Zelda	3×3	2	2	48.10	48.09	48.11
Cameraman	3×3	2	2	48.11	48.09	48.08
Harbor	3×3	2	2	48.10	48.09	48.10

4.3 Comparison with general reference matrix-based data hiding schemes

To compare the proposed models with the related methods based on different kinds of magic matrices, **Table 6** lists the average PSNR values of six images, EC, and matrix-dimensions of these methods. The proposed scheme is based on 2D&4D flexible magic matrices, which is more security than the static matrix-based methods [29-33]. As shown in the table, the proposed adaptive mechanism can significantly improve the PSNR value with a slight sacrifice of embedding capacity.

Table 6. Comparison of proposed method with existing methods.

	Average RSNR	**EC**	**Dimension**
[30]	43.17	2.25	3
Proposed Model1	**43.27**	2.25	3
[29] (*3D-64*)	**47.10**	2	3
[31]	46.36	2	2
[32]	46.37	2	2
[33]	41.64	2	2
Proposed *CM1*	46.25	2	2&4
Proposed *CM2*	**47.13**	1.89	2&4
Proposed *CM3*	**48.10**	1.78	2&4

5. Conclusions

To improve the security and efficiency of the magic matrix-based data hiding scheme, a magic matrix with a flexible dimension and a restricted size is proposed. An efficient data hiding scheme is devised to embed secret data through the guidance of flexible magic matrix. Within the restricted search range, a unique target position can be found to embed secret data. This can significantly improve the embedding efficiency.

In addition, an adaptive mechanism is proposed to embed secret data with an optimal strategy according to the actual combination of secret data and image block. Experimental results confirm that the proposed scheme can effectively enhance the visual quality of stego images. Besides, the proposed adaptive data hiding scheme based on flexible magic matrix outperforms reference matrix-based state-of-the-art methods.

References

[1] Q. Li, X. Wang, X. Wang, et al., "An Encrypted Coverless Information Hiding Method Based on Generative Models," *Information Sciences*, vol. 553, no. 3, pp. 19-30, 2020. Article (CrossRef Link)

[2] O. S. Faragallah, M. A. Elaskily, A. F. Alenezi, et al., "Quadruple Histogram Shifting-based Reversible Information Hiding Approach for Digital Images," *Multimed Tools Appl*, vol. 80, pp. 26297–26317, 2021. Article (CrossRef Link)

[3] Y. Ke, M. Q. Zhang, J. Liu, et al., "Fully Homomorphic Encryption Encapsulated Difference Expansion for Reversible Data Hiding in Encrypted Domain," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 30, no. 8, pp. 2353-2365, 2020. Article (CrossRef Link)

[4] S. Gujjunoori, M. Oruganti, "Difference Expansion Based Reversible Data Embedding and Edge Detection," *Multimed Tools Appl*, vol. 78, pp. 25889–25917, 2019. Article (CrossRef Link)

[5] A. Roy, R. S. Chakraborty, "Toward Optimal Prediction Error Expansion-Based Reversible Image Watermarking," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 30, no. 8, pp. 2377-2390, 2020. Article (CrossRef Link)

[6] I. Caciula, H. G. Coanda, D. Coltuc, "Multiple Moduli Prediction Error Expansion Reversible Data Hiding," *Signal Processing: Image Communication*, vol. 71, pp. 120-127, 2019. Article (CrossRef Link)

[7] P. V. Sabeen Govind, B. M. Varghese, M.V. Judy, "A high imperceptible data hiding technique using quorum function," *Multimed Tools Appl*, vol. 80, pp. 20527–20545, 2021. Article (CrossRef Link)

[8] Y. Jia, Z.Yin, X. Zhang, et al., "Reversible data hiding based on reducing invalid shifting of pixels in histogram shifting," *Signal Processing*, vol. 163, pp. 238-246, 2019. Article (CrossRef Link)

[9] F. Peng, Y. Zhao, X. Zhang, et al., "Reversible Data Hiding Based on RSBEMD Coding and Adaptive Multi-segment Left and Right Histogram Shifting," *Signal Processing: Image Communication*, vol. 81, pp. 115715, 2020. Article (CrossRef Link)

[10] C. C. Chang, "Adversarial learning for invertible steganography," *IEEE Access*, vol. 8, pp. 198425-198435, 2020. Article (CrossRef Link)

[11] C. C. Chang, "Cryptospace invertible steganography with conditional generative adversarial networks," *Security and Communication Network*s, vol. 2021, 2021. Article (CrossRef Link)

[12] L. Li, W. Zhang, C. Qin, et al., "Adversarial batch image steganography against CNN-based pooled steganalysis," *Signal Processing*, vol. 181, pp. 107920, 2021. Article (CrossRef Link)

[13] D. M. Thodi, J. J. Rodriguez, "Expansion Embedding Techniques for Reversible Watermarking," *IEEE Trans. Image Process*, vol. 16, no. 3, pp. 721–730, 2007. Article (CrossRef Link)

[14] B. Ou, X. Li, Y. Zhao, et al., "Pairwise Prediction-Error Expansion for Efficient Reversible Data Hiding," *IEEE Trans. Image Process*, vol. 22, no. 12, pp. 5010–5021, 2013. Article (CrossRef Link)

[15] X. Li, W. Zhang, X. Gui, et al., "Efficient Reversible Data Hiding Based on Multiple Histograms Modification," *IEEE Trans. Inf. Forensics Security*, vol. 10, no. 9, pp. 2016–2027, 2015. Article (CrossRef Link)

[16] B. Ou, Y. Zhao, "High Capacity Reversible Data Hiding Based on Multiple Histograms Modification," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 8, pp. 2329-2342, 2020. Article (CrossRef Link)

[17] D. Shehzad, T. Dag, "LSB Image Steganography Based on Blocks Matrix Determinant Method," *KSII Transactions on Internet and Information Systems*, vol. 13, no. 7, pp. 3778-3793, 2019. Article (CrossRef Link)

[18] R. Shreelekshmi, M. Wilscy, C. E. Veni Madhavan, "Undetectable least significant bit replacement steganography," *Multimed Tools Appl*, vol. 78, pp. 10565-10582, 2019. Article (CrossRef Link)

[19] C. K. Chan, L. M. Cheng, "Hiding Data in Images by Simple LSB Substitution," *Pattern Recognition*, vol. 37, no. 3, pp. 469-474, 2004. Article (CrossRef Link)

[20] Y. Liu, C. Yang, Q. Sun, "Enhance Embedding Capacity of Generalized Exploiting Modification Directions in Data Hiding," *IEEE Access*, vol. 6, pp. 5374-5378, 2017. Article (CrossRef Link)
[21] C. Kim, "Data Hiding by an Improved Exploiting Modification Direction," *Multimed Tools Appl*, vol. 69, pp. 569–584, 2014. Article (CrossRef Link)
[22] X. Li, Y. Luo, W. Bian, "Retracing extended sudoku matrix for high-capacity image steganography," *Multimed Tools Appl*, vol. 80, pp. 18627–18651, 2021. Article (CrossRef Link)
[23] C. C. Chang, Y. C. Chou, T. D. Kieu, "An information hiding scheme using Sudoku," in *Proc. of 3rd Int. Conf. Innov. Comput. Inf. Control (ICICIC)*, Dalian, China, pp. 17–22, Jun. 2008. Article (CrossRef Link)
[24] C. C. Chang, Y. Liu, T. S. Nguyen, "A novel turtle shell based scheme for data hiding," in *Proc. of 10th Int. Conf. Intell. Inf. Hiding Multimedia Signal Process. (IIHMSP)*, Kitakyushu, Japan, pp. 89–93, Aug. 2014. Article (CrossRef Link)
[25] H. Liu, P. Su, M. Hsu, "An Improved Steganography Method Based on Least-Significant-Bit Substitution and Pixel-Value Differencing," *KSII Transactions on Internet and Information Systems*, vol. 14, no. 11, pp. 4537-4556, 2020. Article (CrossRef Link)
[26] W. Chen, C. C. Chang, S. W. Weng, et al., "Multi-layer mini-Sudoku based high-capacity data hiding method," *IEEE Access*, vol. 8, pp. 69256-69267, 2020. Article (CrossRef Link)
[27] Y. Liu, C. C. Chang, T. S. Nguyen, "High capacity turtle shell-based data hiding," *IET Image Process*, vol. 10, no. 2, pp. 130–137, 2016. Article (CrossRef Link)
[28] X. Z. Xie, C. C. Lin, C. C. Chang, "Data hiding based on a two-layer turtle shell matrix," *Symmetry*, vol. 10, no. 2, pp. 47, 2018. Article (CrossRef Link)
[29] J. Lin, J. H. Horng, Y. J. Liu, et al., "An Anisotropic Reference Matrix for Image Steganography," *Journal of Visual Communication and Image Representation*, Vol. 74, pp. 102969, 2021. Article (CrossRef Link)
[30] C. F. Lee, J. J. Shen, S. Agrawal, et al., "Data Hiding Method Based on 3D Magic Cube," *IEEE Access*, Vol. 8, pp. 39445-39453, 2020. Article (CrossRef Link)
[31] M. He, Y. Liu, C. C. Chang, et al., "A mini-Sudoku matrix-based data embedding scheme with high payload," *IEEE Access*, vol. 7, pp. 141414-141425, 2019. Article (CrossRef Link)
[32] J. H. Horng, S. Xu, C. C. Chang, et al., "An Efficient Data-Hiding Scheme Based on Multidimensional Mini-SuDoKu," *Sensors*, vol. 20, no. 9, pp. 2739, 2020. Article (CrossRef Link)
[33] X. Z. Xie, Y. Liu, C. C. Chang, "Extended Squared Magic Matrix for Embedding Secret Information with Large Payload," *Multimed Tools Appl*, vol. 78, pp. 19045-19059, 2019. Article (CrossRef Link)

Hua Wu received a Ph.D. degree from University of Chinese Academy of Sciences in 2016. Currently, she is a lecturer in Beijing Information Science & Technology University. Her research interests include image processing, pattern recognition, information security, and artificial intelligence.

Ji-Hwei Horng received a B.S. degree from the Department of Electronic Engineering, Tamkang University, Taipei, Taiwan in 1990 and M.S. and Ph.D. degrees from the Department of Electrical Engineering, National Taiwan University, Taipei, Taiwan in 1992 and 1996, respectively. He was a professor and Chairman of the Department of Electronic Engineering from 2006 to 2009 and the Dean of the College of Science and Engineering from 2011 to 2014 at National Quemoy University (NQU), Kinmen, Taiwan. Currently, he is the Vice President of Academic Affairs in NQU. His research interests include image processing, pattern recognition, information security, and artificial intelligence.

Chin-Chen Chang received a B.S. degree in Applied Mathematics, an M.S. degree in Computer and Decision Sciences from National Tsing Hua University, and a Ph.D. degree in Computer Engineering from National Chiao Tung University. He was with the National Chung Cheng University from 1989 to 2005. He is currently the Chair Professor of the Department of Information Engineering and Computer Science, Feng Chia University. Prior to joining Feng Chia University, he was an Associate Professor with Chiao Tung University, a Professor with National Chung Hsing University, and a Chair Professor at National Chung Cheng University. He has also been a Visiting Researcher at Tokyo University, Japan, and a Visiting Scientist at Kyoto University, Japan. During his service in Chung Cheng, he served as the Chairman of the Institute of Computer Science and Information Engineering, Dean of College of Engineering, Provost, and then the Acting President of Chung Cheng University and the Director of Advisory Office in Ministry of Education, Taiwan. He has won many research awards and has honorary positions in prestigious organizations both nationally and internationally. He is currently a Fellow of IEEE, U.K. Since his early years of career development, he has consecutively won awards, including the Outstanding Talent in Information Sciences of the R.O.C., the Acer Dragon Award of the Ten Most Outstanding Talents, the Outstanding Scholar Award of the R.O.C., the Outstanding Engineering Professor Award of the R.O.C., the Distinguished Research Awards of National Science Council of the R.O.C., the Top Fifteen Scholars in Systems and Software Engineering of the Journal of Systems and Software, and so on. On numerous occasions, he has been invited to serve as a Visiting Professor, Chair Professor, Honorary Professor, Honorary Director, Honorary Chairman, Distinguished Alumnus, Distinguished Researcher, and Research Fellow by various universities and research institutes. His current research interests include database design, computer cryptography, image compression, and data structures.