Dong *et al. EURASIP Journal on Information Security* (2020) 2020:11
https://doi.org/10.1186/s13635-020-00107-w

EURASIP Journal on Information Security

RESEARCH Open Access

Reversible data hiding for binary images based on adaptive overlapping pattern

Keming Dong[1*], Hyoung Joong Kim[2], Xiaohan Yu[3] and Xiaoqing Feng[4]

Abstract

Pattern substitution (PS) method (Ho et al., Comput. Stand. Interfaces 31:787–794, 2009) is a recent reversible data hiding method for binary images. It generates one pattern pair, the patterns in which are called PM and PF, and substitutes between them to embed one bit. Two types of PS have been proposed: non-overlapping PS and overlapping PS. However, Dong et al. (ETRI J. 37:990–1000, 2015) states that the overlapping PS sometimes cannot decode correctly, which is called as miscoding problem. Moreover, to the PS-based methods (Ho et al., Comput. Stand. Interfaces 31:787–794, 2009; Dong et al., ETRI J. 37:990–1000, 2015; Dong et al., Lect. Notes Comput. Sci. 6526:3516–3522, 2011), although chosen of two patterns to form the pattern pair which relates the embedding performance much, the previous methods only use a fixed pattern pair during the whole data hiding process. In order to overcome these two shortcomings, this paper proposes an adaptive overlapping PS method. Firstly, two modes of the embedding are proposed so that the miscoding problem can be solved. Secondly, one window which contains the context of current processing pixel is generated and updated when the process proceeds to the next pixel. Thus, the optimum pair of the local context for embedding can be chosen. Thirdly, in order to get a smaller location map, a three-round embedding mechanism is also proposed. With these three improvements, our method can embed larger payloads with less distortion than the previous works.

Keywords: Reversible data hiding, Binary image, Adaptive embedding

1 Introduction

Due to the rapid development of internet and computer technique, more and more multimedia files are stored and transmitted through the internet. As a result, the copyright, authentication, and integrity protection problems for multimedia files raise much attention. Data hiding is considered to be an effective technique to solve these problems. Many researchers are attracted into this area. Depending on whether a technique is able to reconstruct the original files or not, the techniques are categorized into lossy data hiding and lossless data hiding (aka reversible data hiding), respectively. Lots of literatures on data hiding have been published for images and other media [1–10]. However, even through lots of digital binary images are increasingly common used in our life, such as digital signatures, halftone images, and scanned documents, there are not many data hiding methods for binary images. The reason is that, common data hiding techniques use the redundancy between the pixels in the image to embed data. On the other hand, pixels in the binary image can only be 0 or 1, which comparably contains few information, so it is very difficult to embed data into binary images. In consequence, it is necessary and important to research on data hiding techniques for binary images.

At the early stage, most works in the binary data hiding area focus on lossy data hiding. Tseng et al. [11] partition the host image into non-overlapping blocks, and then embed one bit of secret data by flipping one binary pixel in that block. Further, they proposed a quality control mechanism to improve the visual quality of the embedded image [12]. The number of black pixels in each block

*Correspondence: kevindong333@hotmail.com
[1]Yunnan Universities Research Center for Data Operation and Management Engineering, Information School, Yunnan University of Finance and Economics, Longquan Road, Kunming,China
Full list of author information is available at the end of the article

can be even or odd; therefore, another line of researches flip pixel(s) to control the odevity of the number of black pixels in each block to embed one bit of secret data. By considering the connectively and smoothness of the pixels in the block, Wu et al. [13, 14] define a *flippability score* to choose pixel to flip for embedding data. Another novel flippability criterion is proposed in [15] to find more suitable pixel to flip. Cheng [16] propose another different flippability criterion to achieve better visual quality. Tzeng [17] introduces an *optimal code holder* to embed binary data into every block. Aforementioned methods are the block-based methods. For this type of data hiding methods, an authentication scheme with tampering localization scheme is proposed by [18, 19]. Excluding the block-based methods, Yang [20] used a morphological transform to achieve high embedding capacity. However, all these algorithms are lossy and cannot be used to reversible data hiding. Since binary images have only one bit plane in every pixel, it makes reversible data hiding for a binary image very challenging.

The pair-wise logical computation (PWLC) mechanism is presented in [21]. This method firstly finds the edge area in the image. Then, the proposed PWLC mechanism is applied to the specific patterns "000000" and "111111" in this edge area to achieve reversible data hiding. The run-length method is proposed by Xuan et al. [22]; this method scans the binary image bit by bit. Thus, the image is represented as one dimensional bitstream. The run-length of each continuous black pixels and white pixels are named as BRL and WRL, respectively. The BRL and its adjacent WRL are considered as one pair, and the pair with the most frequent run-length is chosen to hide data. Moreover, one location map of the chosen pair is also needed to achieve reversibility. In summary, there are not so many "000000" and "111111" bit-streams in the edge area in PWLC method, and the size of the location map in run-length method is always big. Therefore, the performance of these two methods are not good.

Recently, the PS method is proposed in [23], and its performance is outstanding compared with the previous methods. Exclusive-OR operation is applied to the binary image to obtain the *difference matrix*. Four consecutive values in the difference matrix form one *pattern*. Among all patterns of exclusive "0000," the *most frequent pattern* (PM) and the *least frequent pattern* (PF) are chosen to hide the date. The encoder scans the difference matrix to find PM and PF patterns. Each PM or PF is rewritten to PM to embed the secret bit 0, and to PF to embed bit 1. This embedding mechanism is called as the *pattern-substitution* embedding, PS embedding for short. In order to achieve reversibility, the location of PF is recorded by the *location map* (LM), and LM also should be embedded into image by the PS mechanism.

There are two types of PS: non-overlapping PS and overlapping PS. The PFR method is presented in [24] to reduce the LM size for non-overlapping PS. Dong et al. [25] point out that the overlapping PS in [23] sometimes cannot decode correctly, which is called as "miscoding problem." Then, they propose an improved overlapping PS method which solves this problem.

Compared with the previous PS related methods, there are three improvements in the proposed method. First, the performance of the PS highly depends on the choice of the (PM, PF). In the previous works, the (PM, PF) should be decided before embedding, and it never changes during the entire embedding process. This paper proposes an adaptive method so that the (PM, PF) can be automatically chosen according to the context. Therefore, optimal (PM, PF) to the context can be chosen each time. Second, PFR method can significantly reduce the size of LM; however, it can only be used to non-overlapping PS. In order to apply the PFR method to the overlapping embedding, two novel overlapping PS embedding policies are also proposed in this paper. Third, the new LM sometimes is sparse, so it can be compressed well to decrease its size. In order to do this, three-round embedding is proposed. The experiments show that the adaptive embedding and the three-round embedding mechanism efficiently improve the PS method, and the comparison with the other state-of-the-art methods shows the superiority of the proposed method.

2 Related work

2.1 Pattern substitution method

The PS method is proposed by [23]. Assume the binary cover image is denoted as B and this image is called as image or image matrix throughout the rest of the paper; the PS method first transforms the image to *difference matrix* by Eq. 1. The difference matrix is denoted as D in Eq. 1.

$$D(i,j) = \begin{cases} B(i,j) & \text{if } i = 1 \text{ and } j = 1 \\ B(i,j) \oplus B(i-1,j) & \text{if } i \neq 1 \text{ and } j = 1 \\ B(i,j) \oplus B(i,j-1) & \text{otherwise} \end{cases} \quad (1)$$

where $\oplus$ represents the exclusive-OR logic operation, and i and j are the row's and column's position of the cover image, respectively. The element at the top-left corner of the difference matrix remains the same value as in the image matrix, and the rest elements with value 1 of the difference matrix indicate changes in the image matrix. In other words, if the pixel value varies from its previous pixel value in the image matrix, its difference value in the difference matrix is 1; otherwise, its difference value is 0. Assume that a binary bit 0 is used to represent white and bit 1 to represent black; a sample image and its difference matrix are shown in Fig. 1.

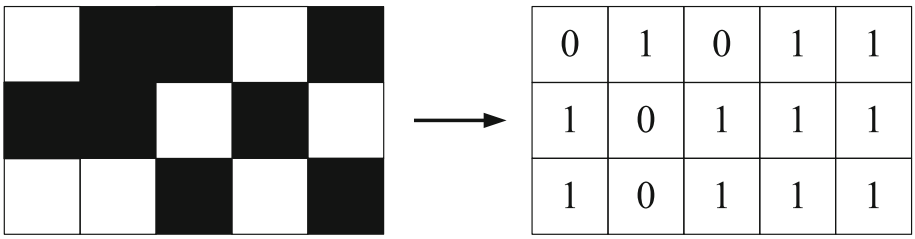

Fig. 1 A binary image and its difference matrix

In the difference matrix domain, four continuous bits form one *pattern*. Therefore, there exist sixteen patterns, from D0000 to D1111. These patterns are denoted as P0 to P15 or P0000 to P1111.

The embedding procedure is as follows. First, the PS method transforms an image to the difference matrix domain by Eq 1, and then two specific patterns are chosen to form a pattern pair before embedding. The rule to choose the pattern pair will be explained later. Then, the encoder scans the different matrix. When the encoder meets the pattern from the pattern pair, it embeds one bit of data using pattern-substitution mechanism. After all secret bits are embedded, the embedded image is generated from the modified difference matrix by the inverse Eq. 1.

Now, the problem transfers to how to choose the pattern pair. Assuming one pattern Px is given, then its suitable pattern Py for substitution should meet two conditions below.

Condition 1: When one substitution of Py for Px in the difference matrix occurs, only one pixel flips its value in the image matrix.

Condition 1 ensures that when pattern substitution occurs in the difference domain, it causes minimum distortion in the image domain, that is to say, only one bit is flipped. As shown in Fig. 2, there is a bit "0" in the image and four bits 0001 is followed. The four bits is denoted as B0001. With Eq. 1, its relevant pattern is P0001. In this manner, if P0001 in the difference matrix is substituted for P0010 by data hiding, B0001 in the image will be changed to B0011.We can see only one bit is different between B0001 and B0011. It is easy to verify that the condition is also satisfied when the previous bit is 1. Therefore, the pair of (P0001, P0010) satisfies Condition 1.

Condition 2: All sixteen patterns are divided into two groups. The pattern with even number of 1 belongs to the even group, and the pattern with odd number of 1 belongs to the odd group. Py should be picked from the same group of Px.

One example is used bellow to explain this condition. According to the definition, pattern P0011 and P0010 are from the different groups, that is, P0011 belongs to the even group and P0010 belongs to the odd group. In this example, we will substitute P0011 for P0010. Consider a binary bit stream B0010000 from the image with a previous bit 0. Since it has two times of flipping, its difference stream is P0011000 by the difference operation. If its first pattern, that is, P0011, is replaced with the pattern P0010, the changed difference stream will be P0010000. With this changed difference stream and its previous bit 0 from the image, the embedded image can be produced as B0011111 by the inverse Eq. 1. Consequently, between embedded stream B0011111 and its original stream B0010000, the rest bit string following the first four bits is inverse from each other. Therefore, if we flip these two patterns to embed data, it causes quite serious distortion. However, if two patterns are from the same group, this kind of distortion does not happen.

The condition 1 ensures that the substitution in the difference domain only makes one pixel flipped inside the pattern area in the image, and condition 2 ensures the pixels outside the pattern area in the image remains the same when the substitution occurs.

The P0 represents the area with four white pixels or black pixels in the image. In this type of region, even a single pixel flipping can cause severe distortion of human vision. Thus, the P0 is not used in the PS method. According to two conditions above, the substitutable patterns for any pattern can be deduced (see Table 1).

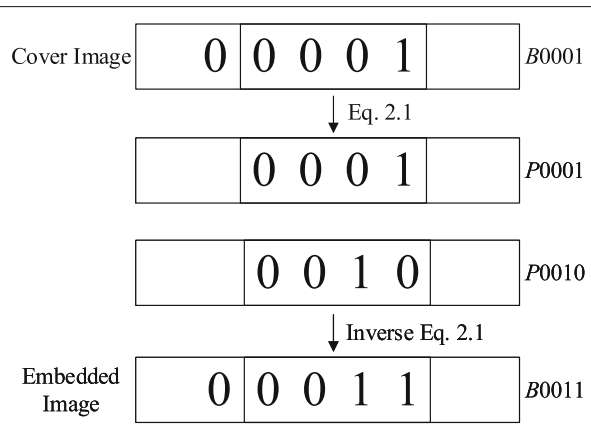

Fig. 2 An example of P_{0001} and P_{0010} which satisfies Condition 1

Table 1 Pattern and its substitutable patterns

Difference value pattern	Substitutable patterns
P_{0001}	$P_{0010}, P_{0111}, P_{1101}$
P_{0010}	$P_{0001}, P_{0100}, P_{1110}$
P_{0011}	$P_{0000}, P_{0101}, P_{1111}$
P_{0100}	$P_{0010}, P_{0111}, P_{1000}$
P_{0101}	$P_{0011}, P_{0110}, P_{1001}$
P_{0110}	$P_{0000}, P_{0101}, P_{1010}$
P_{0111}	$P_{0001}, P_{0100}, P_{1011}$
P_{1000}	$P_{0100}, P_{1011}, P_{1110}$
P_{1001}	$P_{1000}, P_{1010}, P_{1111}$
P_{1010}	$P_{0110}, P_{1001}, P_{1100}$
P_{1011}	$P_{1000}, P_{0111}, P_{1101}$
P_{1100}	$P_{0000}, P_{1010}, P_{1111}$
P_{1101}	$P_{0001}, P_{1011}, P_{1110}$
P_{1110}	$P_{0010}, P_{1000}, P_{1101}$
P_{1111}	$P_{0011}, P_{1001}, P_{1100}$

The number of all the fifteen patterns are counted by raster scanning the difference matrix, from left to right and top to bottom. The most frequent pattern is labeled as PM, and the least probable pattern among its three candidates (refer to Table 1) is named as PF. These two patterns are chosen to embed data.

The pattern-substitution embedding mechanism is simple. The encoder scans the *difference matrix*. When the pattern is PM or PF, the encoder embeds the secret bit 0 by rewriting to PM, and embeds the bit 1 by rewriting to PF.

In order to recover the cover image in the decoding process, the position of PF should be recorded in a *location map*, denoted by LM, and it has to be embedded after all secret data is embedded. The PM and PF values have to be recorded as side information and sent to the decoder before decoding through a secure channel. The decoder scans the difference matrix in the same order as the encoder. When the current pattern is PM, the decoder extracts the bit 0, and the decoder extracts 1 for PF. The extracted data includes two part: the secret data and LM. With favor of the LM, the encoder can recover the cover image. Figure 3 presents a block diagram representing the embedding and extracting processes. A simple example on the PS method is presented in Fig. 4, in which the secret data, denoted as SD, 0110 is embedded. According to the PM and PF sequence shown in the image, the LM is 00000100. Assuming after lossless compression, the compressed LM denoted as LMC is 0010, then the total data to embed is 01100010 which is SD appended with LMC. Then, we flip PM and PF patterns to embed data. After all data is embedded, the inverse Eq. 1 is used to obtain the embedded image.

In the extraction process, the embedded image produce the difference matrix by Eq. 1. According to the PM and PF sequence in the difference matrix, the SD and LMC is extracted. The LMC is decompressed to get LM. With the LM, we can recover the embedded difference matrix. At last, the recovered matrix produce the original image by inverse Eq. 1.

There are two kinds of the PS method. One is the non-overlapping PS, and the other one is the overlapping PS. The non-overlapping PS scans the difference matrix one pattern by one pattern so that the pattern is not overlapped with the next pattern. The overlapping PS scans the difference matrix one bit by one bit so that two neighboring patterns have three common bits. Because the pattern in the overlapping PS intersects its neighboring pattern, decoding will fail. In order to solve this problem, Ho et al. [23] proposed two rules for the overlapping PS.

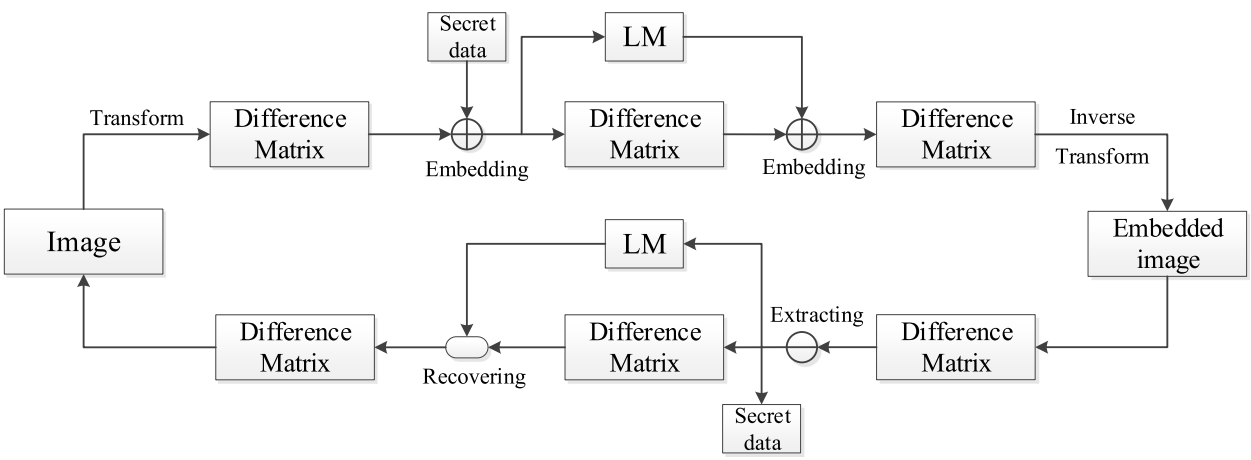

Fig. 3 Flowchart of the PS method

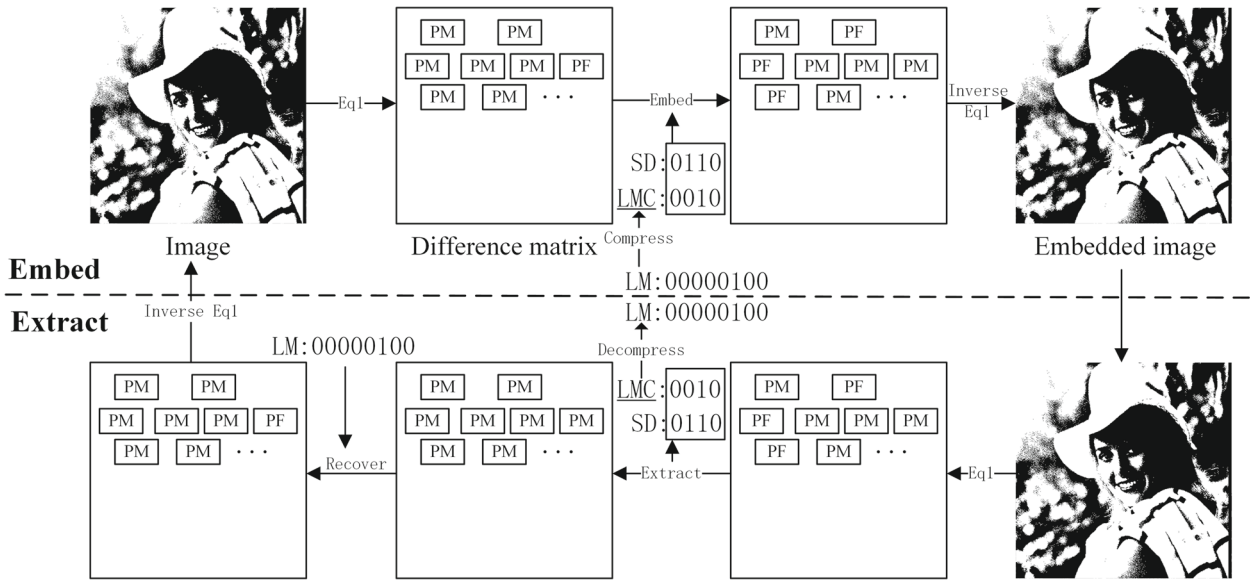

Fig. 4 An example of PS embedding

Rule 1: When the PM and PF overlap, take PF as the hiding pattern.

Rule 2: The 11111 is an unacceptable hiding place.

Dong et al. [25] points out that even these two rules cannot solve the miscoding problem which means the decoder fails to extract the secret data and cannot recover the original image. Three examples of miscoding under these two rules are shown in their paper. Furthermore, they propose one improved overlapping PS method which solves this problem.

2.1.1 PFR method

In the PS method, the location map always occupies much data hiding space. The PFR method [24] is proposed as an improved non-overlapping PS method which uses a new type of location map with much smaller size than [23].

Before embedding, the PFR method firstly chooses the pair of (PM, PF) as the PS method does, and then it finds another infrequent pattern which is labeled as PFR. The PFR is used to replace PF in the data hiding.

The PS method uses the LM to code the PM and PF sequence in the cover image; however, the PFR method firstly flips all the PF in the cover image to PFR, and then the LM is used to mark whether the PFR is original PFR or flipped from the PF. After flipping, if there is no PF in the image, then the encoder only uses the PM to embed: if the hidden bit is 0, the PM keeps the same; if it is 1, the PM is changed to PF.

The decoder extracts bit 0 when it meets PM, and extracts bit 1 when it meets PF. As all the PF and PM in the embedded image are changed from PM, all PF and PM patterns are recovered to PM. After all data is extracted, it includes two parts: secret data and LM. Note that some PFRs in the embedded image are flipped from PF by encoder. Therefore, with the favor of LM, we can recover original PF from PFR.

One example of PFR method is shown in Fig. 5. The PFR method uses LM to code the occurrence sequence of PF and PFR. As both the PF and PFR are less frequent patterns, the new LM is small.

The rule to select the PFR is as follows:

Rule: All patterns is divided to even group and odd group. The PFR should be from the same group of PF, and among the patterns in this group, the least frequent pattern exclude PF is chosen as the PFR pattern.

3 Proposed method

Even through the PFR method proposed in [24] can significantly reduce the size of the LM, it can only work for the non-overlapping PS and cannot apply to the overlapping PS in [23]. In this section, we propose one method which achieves not only overlapping PFR embedding but also adaptive embedding.

3.1 Adaptive selection of (PM, PF, PFR)

All the previous researches on the PS [23–25] use only one pair of (PM, PF) throughout the data hiding process, and this pair should be decided before data hiding. The proposed method can automatically choose (PM, PF, PFR) according to the context, while it is processing each pattern.

The encoder scans the difference matrix one pixel by one pixel from top to bottom and left to right. For each

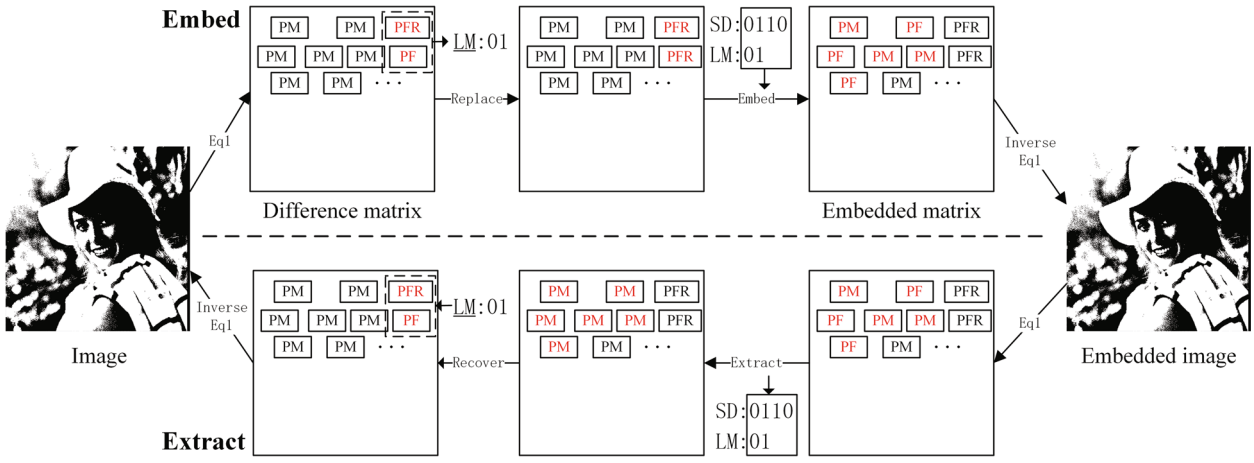

Fig. 5 An example of PFR embedding

pixel in the scanning list, it generates one window which is formed with the following "L" number of pixels. This window is called *context window*, and L is the length of the window. Then, the numbers of all fifteen patterns inside this window are counted. Note that the position of one pattern is indicated/represented by its head pixel, so the statement that one pattern is inside the windows means its head pixel is inside the window. For instance, in Fig. 6, even though the third and fourth pixels of the current pattern are out of the window, as the first pixel of this pattern is inside the window, we say the current pattern is inside the window. Depending on the count numbers of the patterns in the window, the pair of (PM, PF) is selected as the PS method [23], and the PFR is selected as the PFR method [24]. One illustration about the window is shown in Fig. 7. The gray pixels represent the pixels which have been processed. The red pixel is the current processing pixel. In this example, the size of the host image is 8 by 8, and the size of context window "L" is set with 10. After the current pixel is processed, the encoder moves to the next pixel. The window also slides one pixel, and the number of all patterns inside the window is also updated. Then, the triple of (PM, PF, PFR) for the next pattern can be chosen according to the changed context.

In this way, while the encoder is embedding to the current pattern, the optimum (PM, PF, PFR) of current context is selected and used. Note that the decoding process goes the inverse order as the encoding does, and each pattern has to be recovered just after the secret data is extracted. As a result, when the decoder processes to the same pattern as Fig. 6, the context window will also be same as it was in the encoding process. Therefore, if we apply the same choosing mechanism from [23, 24] to the window, the same triplet of (PM, PF, PFR) as the encoding can be obtained in the decoding phase. Thus, under the proposed selection mechanism, even the triplet of (PM, PF, PFR) changes throughout the embedding phase, it does not need to be recorded into the side information, and still can be gotten by the decoder.

3.2 Two embedding modes

In this subsection, we propose two different embedding modes. One is called as "embedding excluding LM" and the other one is called as "embedding including LM." The first embedding mode cannot embed the LM into the image, and the second mode can embed both the secret data and the LM into the image.

We firstly introduce the one which is called "embedding excluding LM." In this mode, the encoder uses the same scanning order as the previous subsection. If the current pattern in the scanning list is PM, then one bit can be embedded. That is to say, if the secret bit is 1, the PM is modified to PF, and if the secret bit is 0, PM does not change. If the current pattern is PF, the PF is modified to the PFR, and a mark bit 1 is appended to the LM. If the current pattern is PFR, a mark bit 0 is appended. If the current pattern is in other cases, then it cannot embed any data, and the encoder skips this pattern. This embedding process is applied on each pattern in the scanning list. Finally, the encoder produces the embedded image and the LM, and the LM has to be sent to the decoder to recover the image. The decoder processes in the inverse order, that is, the decoder extracts the bits from the last embedded pixel to the first. After the current pattern is processed, the decoder recovers this pattern with the help of the LM. Then, the decoder goes to the previous pixel, until it finishes processing the first pixel. As this type of embedding cannot embed the LM to the image, it cannot be an individual data hiding method.

In the scale of reversible data hiding for gray image, Tian proposed his famous difference expansion method in [1]. This method embeds one bit of secret data into two

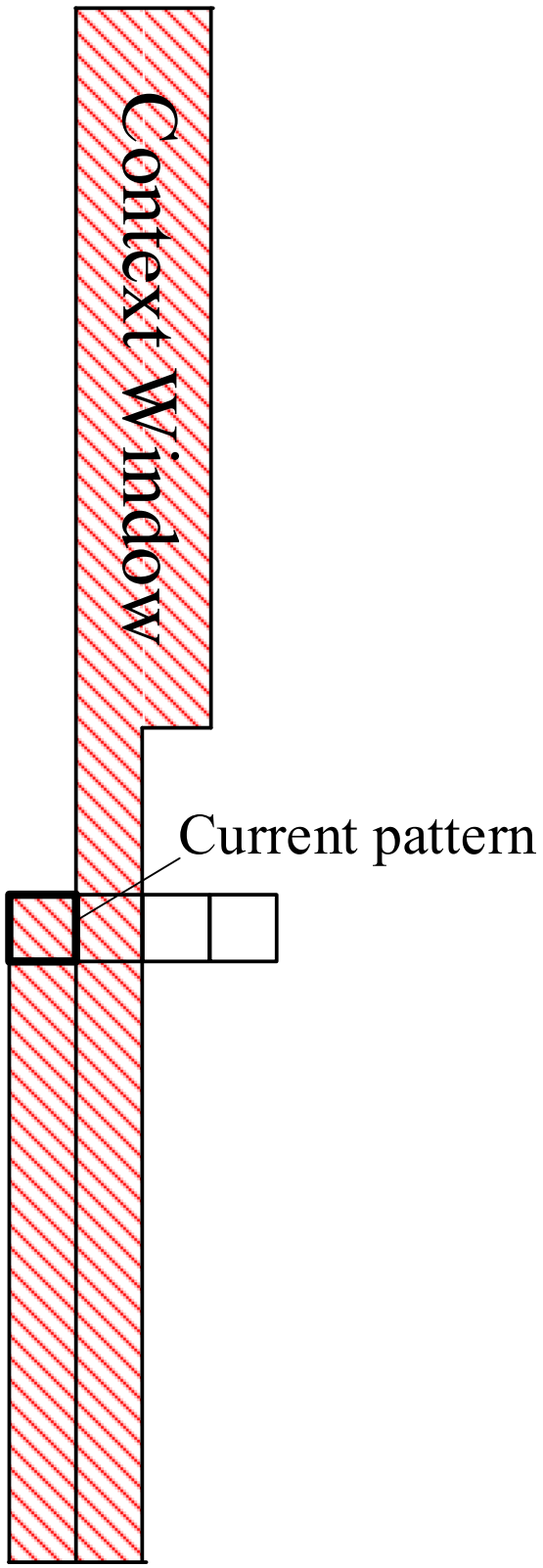

Fig. 6 An example of the current pattern

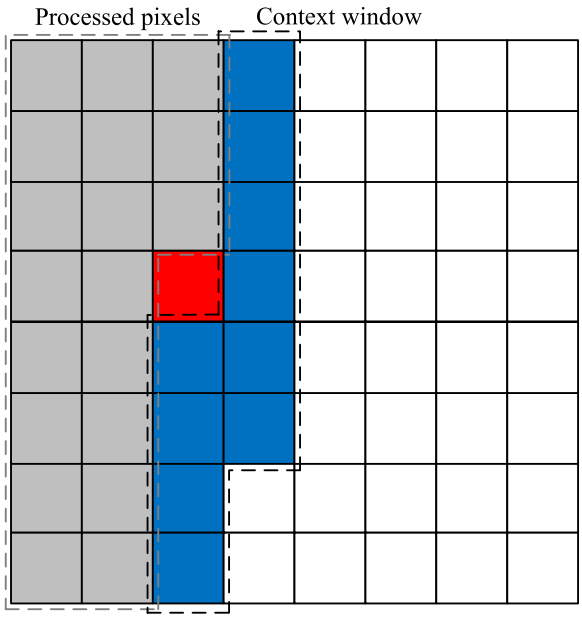

Fig. 7 An illustration of the context window

neighboring pixels, and the next bit will be embedded into the next following two pixels. As shown in Fig. 8, in Tian's method, there is no overlapping part between any adjacent data hiding units. Later, Thoid proposed another method in [2] which is called as prediction-error expansion (PEE) method. In this method, one three-neighbor context of one pixel is defined, that is, the three pixels at the left, top, and top-right of this pixel. Then, it embeds one bit of data into these four pixels. In other words, PEE uses four pixels as one embedding unit. Note that the embedding unit is overlapped with its three adjacent embedding units. In order to solve the overlapping problem, the real embedded data is mixed with the secret data and the LM bits, and the decoder uses the reverse order of the encoder. In this way, the context of any pixel in the decoding can be same as it was in the embedding process. Then, PEE can extract the data and recover the original image. Inspired by how PEE solves the overlapping embedding problem, we propose the second embedding mode.

The second embedding mode is called "embedding including LM." In this type of embedding, the bit of LM is generated and inserted to the secret data (SD) during embedding process. The bitstream which mixes the SD and the LM is denoted as LM_SD. The initial value of LM_SD is set as the SD. If the current pattern is PM, the first bit of the LM_SD is embedded, and then this bit is removed from the LM_SD. If the current pattern is PF, then PF is modified to PFR, and a LM bit 1 is appended to the front of LM_SD, which means this bit is planned to be embedded in the next embedding. If the current pattern is PFR, a LM bit 0 is appended to the front of

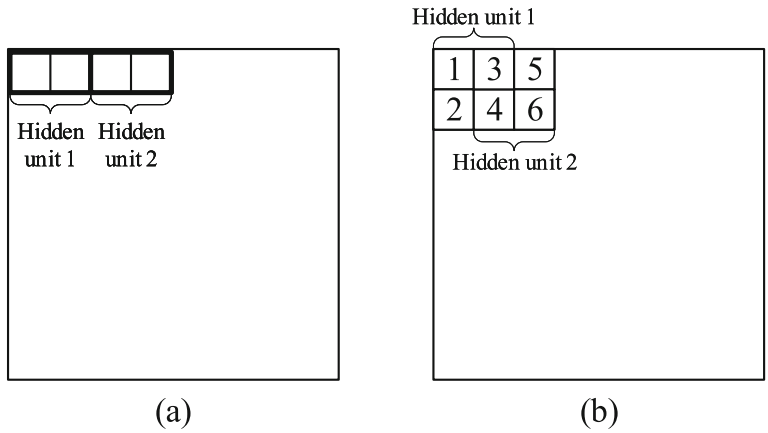

Fig. 8 a Tian's method: unit1 and unit2 have no overlapping part. **b** Thoid's method: unit1 contains pixel 1, pixel 2, pixel 3, and pixel 4. Unit2 contains pixel 3, pixel 4, pixel 5, and pixel 6. Therefore, pixel 3 and pixel 4 are overlapping pixels

the LM_SD. If it is in other cases, the encoder skips this pattern and move to the next pattern in the scanning list. The embedding process stops whenever all bits of LM_SD has been embedded, that is, the LM_SD become an empty list.

The decoder starts scanning from the last embedded pixel, and proceed in a reverse order of the encoder. The initial value of LM_SD is set as an empty list. If the current pattern is PM or PF, one bit is extracted and appended to the front of LM_SD. If the current pattern is PFR, then the head of LM_SD is a LM bit. This bit is a mark that represents the current PFR was changed from the PF or not by the encoder. Therefore, if the head is 1, the PFR is modified to PF, and then, it is removed from LM_SD. If the head is 0, it removes from LM_SD. Then, the decoder goes to the previous pixel, until it finishes processing the first embedded pixel. The final LM_SD is the secret data.

A simple example is shown in Fig. 9. This example shows that even the first three patterns, which are PM, PF, and PFR, are overlapped. This embedding mode can successfully do reversible data hiding without any miscoding problem. Note that in the third step in the extracting process, as the decoder meets PFR pattern, then it has to check the head bit of LM_SD which marks whether this PFR was from PF or not. As the head bit is 1 which

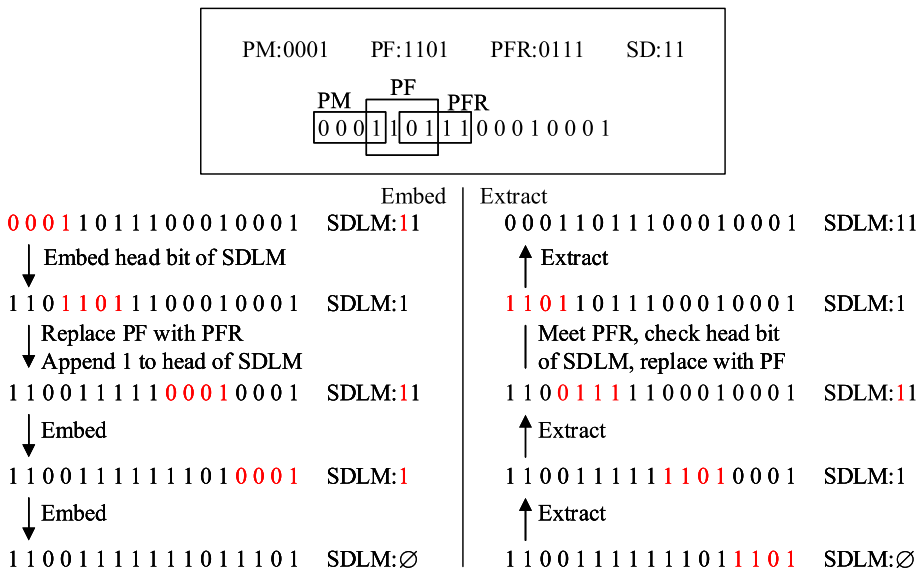

Fig. 9 An example of data hiding with "embedding including LM" mode

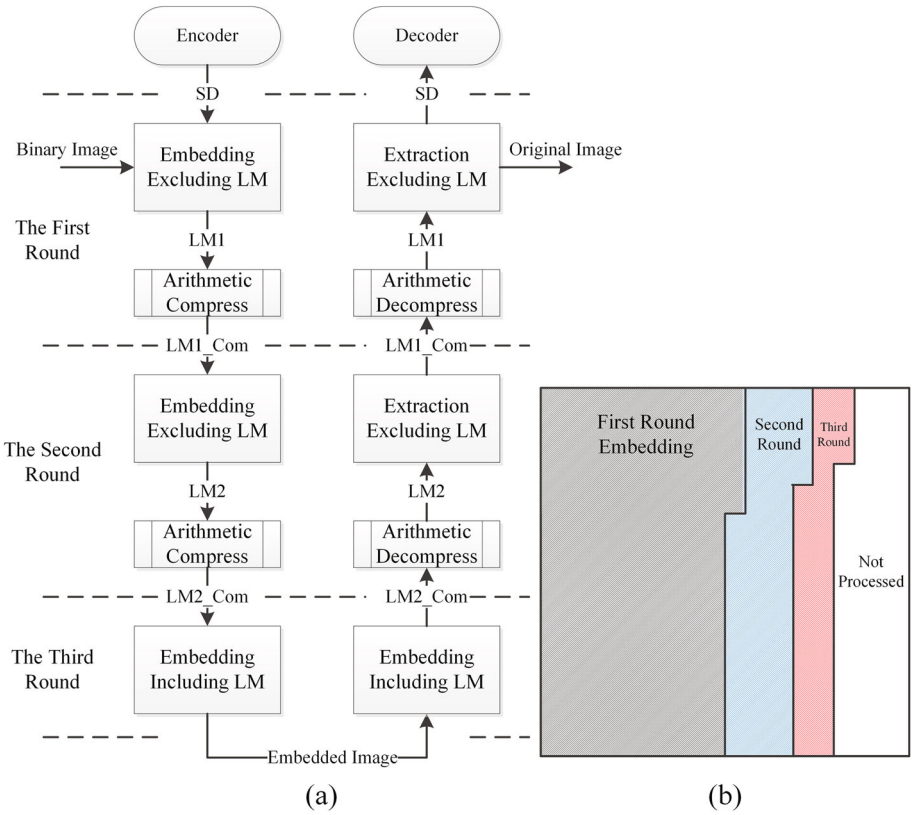

Fig. 10 The three-round embedding mechanism: **a** the flowchart; **b** the three-round embedding on the binary image

is colored with red in the figure, the PFR is replaced for PF.

3.3 Three-round embedding

The mode of "embedding including LM" can be an individual data hiding method, because it embeds both the secret data and related LM into the image. However, the other one cannot be. In the excluding mode, after all secret data is embedded, it still leaves the LM which cannot be embedded into the image. As the number of PF and PFR sometimes has great disparity, which means the LM can be easily compressed. Therefore, we design three-round embedding mechanism which uses both of two embedding modes. The first two rounds use the excluding-mode embedding, and the last round uses the including-mode embedding.

As shown in Fig. 10, in the first round, we embed the SD using the mode of "embedding excluding LM." The output of this round is the location map which is notated as LM1, and then LM1 is compressed using arithmetic

Table 2 The encoder algorithm

1:	Obtain D0 from B0 by the Eq. 1
2:	Set the value of L
3:	Embed SD to D0 by the excluding mode, and produce D1 and LM1
4:	Compress LM1 to LM1C
5:	Embed LM1C to D1 by excluding mode, get D2 and LM2
6:	Compress LM2 to LM2C
7:	Embed LM2C to D2 by the including mode, get D3
8:	Obtain B3 from D3 by the inverse of Eq. 1

Table 3 The decoder algorithm

1:	Obtain D3 from B3 by the Eq. 1
2:	Get L, EndPos3, and BegPos3 from side information
3:	Extract LM2C from D3, and restore D3 to D2 by including mode
4:	Decompress LM2C to LM2
5:	Get EndPos2 and BegPos2 from side information
6:	Extract LM1C form D2 by the excluding mode, and restore D1 under LM2
7:	Decompress LM1C to LM1
8:	Extract SD from D1 by the excluding mode, and restore D0 under LM1
9:	Obtain B0 form D0 by the inverse of Eq. 1

Table 4 Mean and standard deviation of the PSNR values with different context windows, and optimal *l*

	500	1500	2500	3500	4500
Blonde	(61.17, 1.33), 3	(52.22, 1.53), 3	(46.90, 1.03), 3	(44.38, 0.88), 8	(42.07, 0.57), 3
Couple	(64.05, 1.03), 2	(54.57, 0.44), 10	(50.11, 0.64), 8	(47.09, 0.74), 10	(44.24, 0.95), 10
Bridge	(66.86, 1.04), 7	(55.24, 0.61), 2	(49.58, 0.58), 3	(46.06, 0.61), 4	(43.47, 0.56), 2
Jet	(69.14, 0.24), 3	(57.38, 0.54), 1	(51.86, 0.45), 1	(48.50, 0.34), 1	(46.19, 0.20), 1

coding. The second-round embedding starts from the next pixel of the first-round tail pixel. In this round, the compressed LM1 (LM1C) is embedded using the excluding mode. The output of the second round is called LM2, and LM2 is also compressed using arithmetic coding. In the third round, the compressed LM2 (LM2C) is embedded using the mode of "embedding including LM." The decoding follows the inverse way to get the secret data and original image. The positions of the beginning pixel of the second round and third round, and the arithmetic coding parameters for the first and second round are considered as the side information, and it should be sent to the decoder before decoding.

Assume the host binary image, the difference matrix, the difference matrix after the first-round, second-round, and the third-round embedding, and the final watermarked image are B0, D0, D1, D2, D3, and B3, respectively, then the encoder algorithm is shown in Table 2

The decoder processes in the inverse order as the encoder does. Before decoding, the decoder receives the side information. The position of the beginning and last embedded pattern of the third-round embedding are denoted by BegPos3 and EndPos3, respectively. In the same manner, the position of the beginning and last pattern of the second round and the first round are denoted as BegPos2, EndPos2, BegPos1,and EndPos1, respectively. The algorithm for decoder is shown as Table 3.

4 Experiments

4.1 Adaptive selection of (PM, PF, PFR)

The only parameter for the proposed method is the length of the context window. To show how the adaptive embedding mechanism affects the performance, the same secret data is embedded to one image several times, and for each time, only the length of the window is set by different value. Assume the size of the cover is $n_x \times n_y$, where n_x and n_y is the number of rows and columns, respectively.

The length of the context windows is set as $[n_x \times n_y/l]$, where $[X]$ produces the nearest integer of X and l is set from 1 to 10. Therefore, ten different PSNR values with the same data embedded can be derived. The mean value and a corresponding standard deviation based on these ten PSNRs are calculated. Table 4 shows the mean value, the deviation value, and the l value with the optimum performance when the different sizes of secret data are embedded.

The decision of (PM,PF) is essential in the PS method, that is, the greater disparity in number between PM and PF is, the better the performance is. In the previous works, the (PM,PF) is decided before the embedding and never changes during the data hiding process. However, the disparity between the PM and PF varies as the data hiding process proceeds. Therefore, automatically updating the (PM,PF) pair depending on the context should be necessary.

As shown in Table 4, when 500 bits of secret data is embedded to couple image ten times, the mean value and the deviation of the PSNR values are 64.05 db and 1.03 db, respectively, and the l value with the best PSNR is 2. The deviation value shows that even the same secret data is embedded, if we use different length of context, the PSNR values have large difference. In addition, the optimum length of context window is also different when different secret data is embedded. This result indicates that the

Table 5 Size of location map with different size of SD embedded (in bit): 500, 2500, 4500 bits

	Blonde			Bridge			Jet			Couple		
×100	5	25	45	5	25	45	5	25	45	5	25	45
LM1	182	808	1341	57	453	1147	5	402	649	178	929	1936
LM1C	182	790	1339	56	448	1118	5	247	448	156	723	1208
LM2	40	237	332	14	78	265	0	34	30	62	378	727
LM2C	32	231	332	14	78	262	0	34	30	62	173	403

context windows affect the performance much, and this verifies the necessity of adaptive embedding.

4.2 Three-round embedding

PF is the least frequent pattern, and the location map records the PF and PFR stream in the data hiding process. The location map sometimes can be sparse and easily compressed. The proposed method uses three-round embedding. In this way, the location map generated by embedding the SD can be compressed before it is embedded to the image.

For the case in Table 5, with 4500 bits embedded to the couple image, if the three-round-embedding mechanism is not used, then the location map with size of 1936 bits has to be embedded to the image, which will occupy much embedding space. However, after the compression in the first round, the location map is compressed to 1208 bits. In this way, this mechanism can improve the performance. The compression result in the second round is not as good as the first round. This is mainly because the data embedded to the second round, LM1C, is already very short.

4.3 Comparison with the previous works

Figure 11 shows the test images and their embedded images with 1500 bits of secret data embedded. We can see the distortion caused by embedding only occurs in the edge area, and it is not so easy to notice.

Figure 12 shows the comparison with the researches [23–25]. We generate a random value for the hiding gap in [25]. The adaptive embedding makes the method always select the optimum PM, PF, and PFR of the context for each embedding. Secondly, the proposed embedding mode, embedding including LM, successfully applies PFR to the overlapping embedding, which not only solves the miscoding problem but also decreases the location map. Thirdly, if only the "embedding including LM" is used, then the original LM has to be embedded to the image. However, with the three-round embedding mechanism, the LM is compressed at first and then embedded to the image which save much space. These three improvements makes the proposed method get noticeable improvement in both distortion and capacity. The experiment result also verifies that.

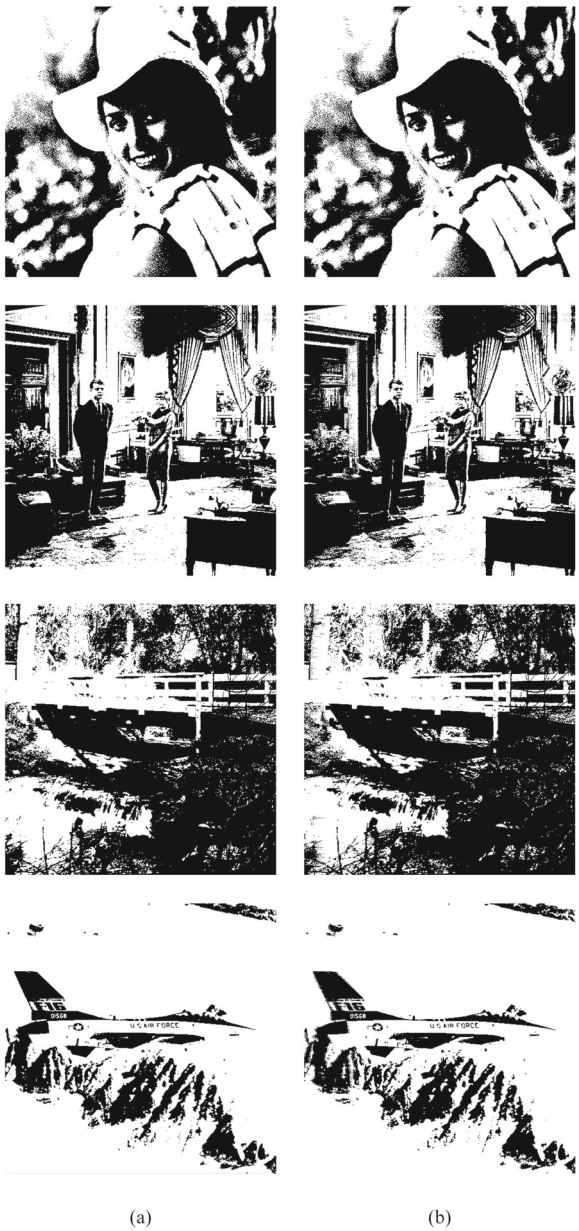

Fig. 11 a The test images. **b** Images with 1500 bits embedded

5 Conclusion and further research

This paper proposes one adaptive reversible data hiding method for binary images. The triplet of (PM; PF; PFR) can be automatically chosen according to the context so that the optimum triplet for the context can be always used. The paper also applies PFR method to the overlapping PS, so the new location map with small size is used. Three-round embedding mechanism is presented to compress the location map. The experiment verifies the adaptive embedding and three-round embedding significantly improve the performance, and the comparison with the previous work shows the superiority of our method.

Although the proposed method has achieved the adaptive selecting of PM, PF, and PFR, the length of the context window still has to be set manually. Moreover, in the original PS method, the pattern is defined to be one-dimensional with 4 members. Is it possible to use two-dimensional pattern or one-dimensional with different members? The further research will focuses on these two topics.

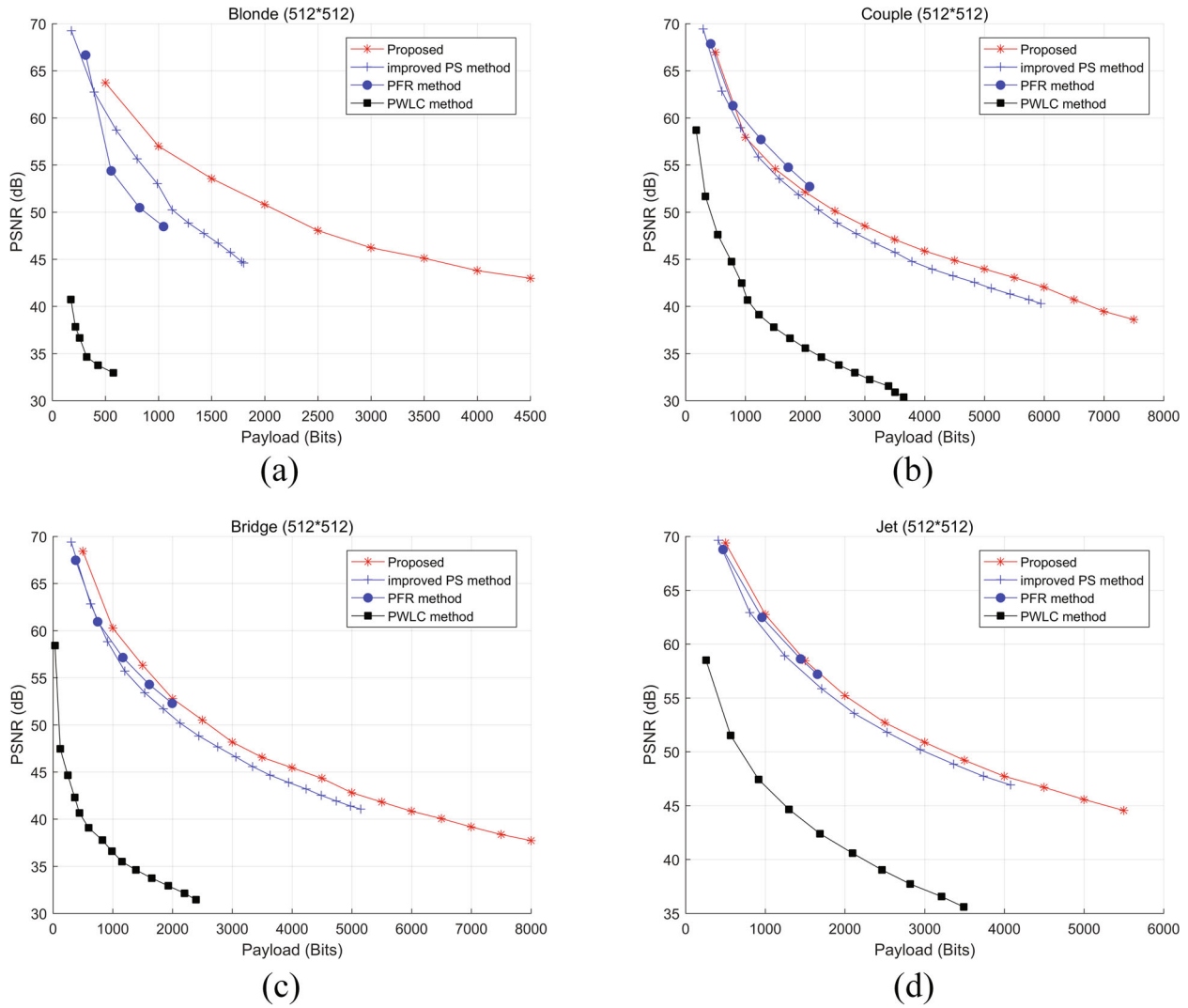

Fig. 12 Comparisons of PSNR and capacity (in bits)

Author's contributions

KD and HJK conceived and designed the study. KD, XY, XF, and HJK performed the experiments. KD wrote the paper. KD, XY, XF, and HJK reviewed and edited the manuscript. All authors read and approved the manuscript.

Funding

This work was supported in part by the National Natural Science Foundation of China (No. 61701441).

Availability of data and materials

The data used to support the findings of this study are available from the first author upon request.

Competing interests

The authors declare that they have no competing interests.

Author details

[1]Yunnan Universities Research Center for Data Operation and Management Engineering, Information School, Yunnan University of Finance and Economics, Longquan Road, Kunming,China. [2]Graduate School of Information Security, Korea University, Anam dong, 136-713 Seoul, South Korea. [3]School of Information and Electronic Engineering, Zhejiang Gongshang Univeristy, Xuezheng Street, 24105 Hangzhou, China. [4]Department of Information School, Zhejiang University of Finance and Economics, Xueyuan Street, 24105 Hangzhou, China.

Received: 9 January 2020 Accepted: 22 April 2020
Published online: 01 June 2020

References

1. J. Tian, Reversible data embedding using a difference expansion. IEEE Trans. Circuits Syst. Video Technol. **13**, 890–896 (2003)
2. D. M. Thodi, J. J. Rodriguez, Expansion embedding techniques for reversible watermarking. IEEE Trans. Image Process. **16**, 721–730 (2007)
3. V. Sachnev, H. Kim, J. Nam, S. Suesh, Y. Q. Shi, Reversible watermarking algorithm using sorting and prediction. IEEE Trans. Circuits Syst. Video Tech. **19**, 989–999 (2009)
4. L. Li, H. Xu, C. C. Chang, Y. Ma, A novel image watermarking in redistributed invariant wavelet domain. J. Syst. Softw. **84**, 923–929 (2011)
5. B. Ou, X. Li, Y. Zhao, R. Ni, Y. Q. Shi, Pairwise prediction-error expansion for efficient reversible data hiding. IEEE Trans. Image Process. **22**, 5010–5021 (2013)

6. C. Qin, C. Chang, Y. Chiu, A novel joint data-hiding and compression scheme based on SMVQ and image inpainting. IEEE Trans. Image Process. **23**, 969–978 (2014)
7. X. Li, W. Zhang, X. Gui, B. Yang, Efficient reversible data hiding based on multiple histograms modification. IEEE Trans. Inf. Forensics Secur. **10**, 2016–2027 (2015)
8. B. Ou, X. Li, J. Wang, F. Peng, High-fidelity reversible data hiding based on geodesic path and pairwise prediction-error expansion. Neurocomputing. **226**, 23–34 (2017)
9. F. Huang, J. Huang, Y. Shi, New framework for reversible data hiding in encrypted domain. IEEE Tran. Inf. Forensics Secur. **11**(12), 2777–2789 (2016)
10. J. Qin, F. Huang, Reversible data hiding based on multiple two-dimensional histograms modification. IEEE Sig. Process. Lett. **26**(6), 843–847 (2019)
11. Y. C. Tseng, H. K. Pan, A secure data hiding scheme for two-color images. IEEE Symp. Comput. Commun. **51**, 750–755 (2000)
12. Y. C. Tseng, H. K. Pan, Data hiding in 2-color images. IEEE Trans. Comput. **51**, 873–878 (2002)
13. M. Wu, E. Tang, B. Liu, in *2000 IEEE International Conference on Multimedia and Expo. ICME2000. Proceedings. Latest Advances in the Fast Changing World of Multimedia (Cat. No.00TH8532)*. Data hiding in digital binary image (IEEE, 2000). https://doi.org/10.1109/icme.2000.869623
14. M. Wu, B. Liu, Data hiding in binary images for authentication and annotation. IEEE Trans. Multimedia. **6**, 528–538 (2004)
15. H. Yang, A. C. Kot, Pattern-based data hiding for binary image authentication by connectivity-preserving. IEEE Trans. Multimedia. **9**, 475–486 (2007)
16. J. Cheng, A. C. kot, Objective distortion measure for binary text image based on edge line segment similarity. IEEE Trans. Image Process. **16**, 1691–1695 (2007)
17. C. H. Tzeng, W. H. Tsai, A new approach to authentication of binary images for multimedia communication with distortion reduction and security enhancement. IEEE Trans. Commun. Lett. **7**, 443–445 (2003)
18. H. Yang, A. C. Kot, Binary image authentication with tampering localization by embedding cryptographic signature and block identifier. IEEE Trans. Sig. Process. Lett. **13**, 741–744 (2006)
19. Y. Lee, H. Kim, Y. Park, A new data hiding scheme for binary image authentication with small image distortion. Inf. Sci. **179**, 3866–3884 (2009)
20. H. Yang, A. C. Kot, S. Rahardja, Orthogonal data embedding for binary images in morphological transform domain: a high-capacity approach. IEEE Trans. Multimedia. **10**, 339–351 (2008)
21. H. F. Chiang, K.-C. Fan, C.-L. Tsai, C. D. Chung, Reversible data hiding and lossless reconstruction of binary images using pair-wise logical computation mechanism. Pattern Recog. **11**, 1993–2006 (2005)
22. G. Xuan, Y. Q. Shi, P. Chai, X. Tong, J. Teng, J. Li, in *2008 19th International Conference on Pattern Recognition*. Reversible binary image data hiding by runlength histogram modification (IEEE, 2008). https://doi.org/10.1109/icpr.2008.4761613
23. Y. A. Ho, Y. K. Chan, H. C. Wu, Y. P. Chu, High-capacity reversible data hiding in binary images using pattern substitution. Comput. Stand. Interfaces. **31**, 787–794 (2009)
24. K. Dong, H. Kim, An efficient pattern substitution watermarking method for binary images. Lect. Notes Comput. Sci. **6526**, 3516–3522 (2011)
25. K. Dong, H. Kim, Y. Choi, S. Joo, B. Chung, Reversible binary image watermarking method using overlapping pattern substitution. ETRI J. **37**, 990–1000 (2015)

Publisher's Note