

Received July 23, 2018, accepted September 17, 2018, date of publication October 15, 2018, date of current version March 18, 2019.

Digital Object Identifier 10.1109/ACCESS.2018.2875923

A Reversible Watermark With a New Overflow Solution

BIN FENG¹, BOTAO YU², YILIN BEI¹, AND XINQIANG DUAN¹

¹School of Information Science and Technology, Taishan University, Taian 271000, China

²School of Software Technology, Dalian University of Technology, Dalian 116620, China

Corresponding author: Yilin Bei (beiyilinok@163.com)

This work was supported by the National Science Foundation of China under Grant 61401060 and Grant 61771090.

ABSTRACT Overflow problem is essential to reversible watermarking algorithms, as it would lead to great distortion or irreversibility if not properly handled. In general, the existing algorithms handle overflow problem with methods that take a lot changes to pixels, which consequently brings a possibly poor quality or a high complexity. In order to solve the overflow problem specifically with a higher quality and a lower complexity, and improve the security of watermark, we put forward a new reversible watermarking scheme. We take the strategy of twice embedding—first, the proposed scheme uses wavelet histogram shifting to embed the watermark information; second, a proposed low-distortion overflow processing algorithm (LDOPA) is implemented to process the overflow problem occurred during the first step, which iteratively scans the pixels, modifies the overflow pixels one by one, and embeds the modification record with its authentication information on the image. In addition, we apply logistic mapping, torus mapping, and CRC to improve the security of the watermark, allowing users to extract watermark only when they hold the correct password. The experimental results prove that the scheme realizes the complete reversibility of watermark and carrier image. Especially, the proposed LDOPA can be easily combined with the existing watermarking methods to further improve their performance.

INDEX TERMS Reversible watermark, overflow handling, logistic mapping, torus mapping.

I. INTRODUCTION

Digital watermarking [1] is an important tool of copyright protection and content authentication for multimedia resources. Reversible watermarking, due to its feature of lossless reversibility, has become an important research direction of the field [2]. It can not only hide watermark information in carrier image, but also restore watermark information and carrier image completely through the inverse process. At present, the representative reversible watermarking algorithms are mainly based on difference expansion and histogram shifting.

Difference-expansion-based algorithm was firstly proposed by Tian [3]. They designed a reversible integer transform, expanding the difference of adjacent pixels and replacing the primitive values with new values, in which way the watermark was embedded. It works well, and has brought a profound impact to the development of reversible watermarking. Based on the work of Tian, Alattar proposed a generalized difference expansion algorithm [4], which increased the watermarking capacity; Thodi *et al.* embedded watermark into the prediction errors of each pixel [5], which improved the payload while keeping the distortion low; etc.

Histogram-shifting-based algorithm was first put forward by Ni *et al.* [6]. The principle is to modify the pixels of specific values, and embed watermark with these values, which intuitively equal to shifting the histogram of pixel values. The algorithm is simple in calculation, and has a high quality of watermark invisibility, but the embedding capacity relies too much on the carrier images. Subsequently, Xuan *et al.* [7] and Wu [8] presented algorithms of wavelet histogram shifting. Instead of modifying the pixel values in spatial domain, these methods embedded the watermark by modifying wavelet coefficients in integer wavelet transform domain, which improved the quality and capacity of the watermark. A number of improved innovations have also been proposed later.

For reversible watermarking, overflow is always a problem that we must consider and solve. The so-called overflow refers to that the pixel values exceed the boundary of the valid range. That is, for a grayscale image, a pixel overflows when its value is less than 0 or greater than 255. If not handled, the overflow would cause great damages to the image, and the loss of information leads to the failure of

reversible recovery of the carrier image and the watermark. At present, the main methods to handle the overflow include Module-C [9], [10], location mapping [3], [11] and histogram compression [12], [13]. The Module-C method adjusts the pixels with a reversible modulus operation, but values of adjusted pixels change greatly, which causes noises and decreases the quality of the image. Location mapping method works by skipping embedding into the pixels where overflow would happen, and the skipped pixels should be recorded. However, it makes the embedding capacity rely too much on the characteristics of the carrier image, and the efficiency is low. Histogram compression method makes the pixel values near the boundary closer to the middle of the valid range, and records the adjustment information [14]. Yet, it is hard to determine the adjustment threshold before embedding the watermark, and the operation is complicated.

In this paper, we do research on the reversible watermarking algorithms based on image. Aiming at solving the above problems, we propose a new reversible image watermarking scheme with a new overflow solution. It is a two-stage embedding process. Firstly, we embed watermark information into carrier image with wavelet histogram shifting method. Secondly, we adjust the overflowed pixels in the carrier image and embed the adjustment records with the proposed low distortion overflow processing algorithm (LDOPA). The contributions of this paper include: firstly, the LDOPA represents a new idea of solving the overflow problem for reversible watermarking, with a higher quality and a lower complexity; secondly, with the application of Logistic mapping, MD5 hash function, Torus mapping and Cyclic Redundancy Check (CRC) [15], the proposed scheme also enhances the security of the watermark [16].

The constitution of the paper is as follows. In the next part, we will introduce some related techniques adopted in our watermarking scheme. In the third part, we will describe the new overflow solution – LDOPA in detail. Then, we will introduce the entire process of our watermarking scheme. At last, extensive experiments are conducted to assess the performance of the LDOPA and the watermarking scheme.

II. RELATED TECHNIQUES

A. LOGISTIC MAPPING

Chaotic system is a nonlinear system with complex pseudo-random. After being processed by a chaotic system, a numerical sequence will become chaotic and seemingly random, while it is actually regular [17]. Logistic mapping is a common instance of chaotic systems [18]. Its widely-used model is

$$x_{n+1} = \mu x_n(1 - x_n), \quad (1)$$

where x_n is a variable, $x_n \in (0, 1)$, and $n \in N$. The letter μ represents the indicator of the system state. If $3.569945 \leq \mu \leq 4$, the system is in a chaotic state [19]. Since Logistic mapping is strongly sensitive to the initial value x_0 and the state indicator μ , and the correlation between adjacent numbers is weak [20], it can effectively prevent exhaustive key

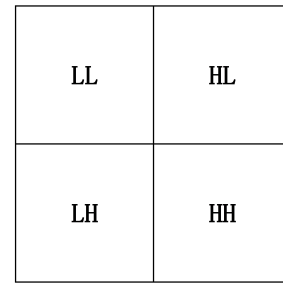


FIGURE 1. Subbands of wavelet transform.

attack and statistical attack. Therefore, we adopt it for image encryption. Set the value of μ and choose a secret initial value x_0 , and through the Logistic mapping, the encrypted image will be highly scrambled.

B. TORUS AUTOMORPHISM MAPPING

Torus automorphism mapping is a simple but useful method, with which we can get the mapping sequence $A \rightarrow B \rightarrow C \rightarrow \dots \rightarrow A$, that is, position A is mapped to position B , position B is mapped to position C , and so on, and the last position is mapped to position A . The one-dimensional Torus mapping is formulated as

$$X' = kX \pmod{M + 1}, X \in [1, M] \text{ and } \in Z, \quad (2)$$

where k is a prime number, $k \in [0, M - 1]$. M is the amount of all the positions. X and X' are the original position and its mapped position respectively.

In our watermarking scheme, we use Torus mapping to determine the block where information will be embedded each time. For example, we divide the carrier image into $a \times b$, altogether M blocks, and number them from left to right, from top to bottom, so the order numbers will be 1 to M . With the one-dimensional Torus mapping, we can get the order number of the block where to embed information for the i -th time as

$$P(i) = ik \pmod{M + 1}, i \in [1, M] \text{ and } \in Z, \quad (3)$$

where $P(i)$ is the order number of the block. By Torus mapping, we can scramble the order of blocks, which will improve the invisibility of watermark, and avoid the problem that a serial of watermark information cannot be extracted if the image is damaged in a big scale [21].

C. INTEGER WAVELET TRANSFORM (IWT)

Wavelet transform [22] supports good time-frequency localization and multi-resolution analysis, and is wide-applied in image processing. Through the wavelet transform, image will be decomposed, as shown in Fig. 1, into four subbands: LL, HL, LH and HH. The LL subband contains most of the energy, while the HL and the LH subbands contain details of the image.

However, the transform domain coefficients of general wavelet transform are floating numbers, which will cause the loss of accuracy, thus the original image will fail to be reconstructed. Therefore, we use integer wavelet transform

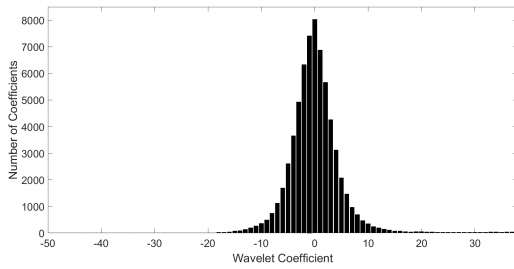


FIGURE 2. A wavelet histogram of the HH subband of an image.

(IWT, for the method please refer to [7]). It does not rely on the Fourier transform. Instead, it is based on lifting, and goes through three steps – splitting, predicting and updating, to decompose the original signal into high-frequency and low-frequency signals. Since the transform executes in integer domains, the IWT is fully reversible. In addition, it has the advantages of easy expansion, low complexity and fast computation.

D. WAVELET HISTOGRAM SHIFTING

Different from the general histogram shifting that modifies the pixel values, wavelet histogram shifting modifies the wavelet coefficients to embed the watermark information. We now give an example to introduce its principle described in [7].

For example, we implement the IWT over an image, do statistics of the wavelet coefficients in HH subband, and generate the wavelet histogram as Fig. 2. The X-axis represents the value of the wavelet coefficients, while the Y-axis represents the amount of each coefficient value in the HH subband.

When embedding, we should first choose a zero-point Z , from which we start to modify the histogram. Since wavelet coefficients in the middle-and-high-frequency subbands (HL, LH, HH) are in concordance with Laplacian-like distribution, and the mean value is near zero [23], we choose the wavelet coefficient 0 as the first zero-point, so $Z = 0$. Then, we increase the coefficients larger than Z by 1, which is equal to shifting the part of the histogram with values larger than Z towards right by 1. Thus, there is no longer any coefficient with value $Z + 1$ existing in the image, and at $Z + 1$ in the histogram forms a blank space, as shown in Fig. 3. Then, we scan all the coefficients, and for every coefficient with value Z , if the to-be-embedded bit is “1”, increase it by 1; if not, no change is made.

When extracting, for every coefficient with value $Z + 1$, bit “1” is extracted and the coefficient should be recovered to Z ; for every coefficient with value Z , bit “0” is extracted. After all the coefficients with value $Z + 1$ or Z have been processed, coefficients larger than $Z + 1$ should be subtracted by 1 to come back to the original values.

Due to the Laplacian-like distribution of the coefficients, we can embed data in both sides of the histogram alternatively. We will introduce the detailed method that adopted in our scheme later.

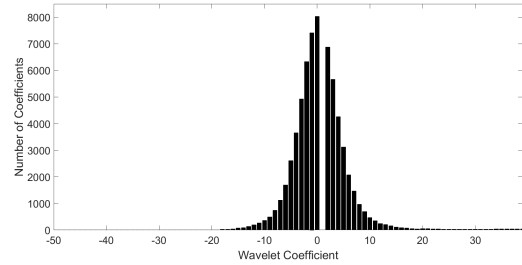


FIGURE 3. The wavelet histogram after the first shift from Fig. 2.

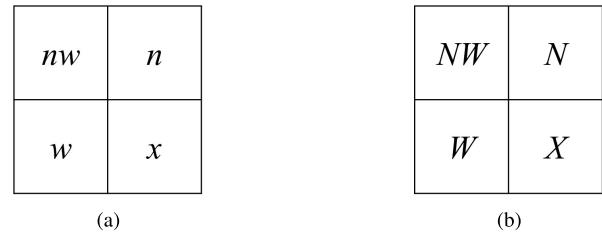


FIGURE 4. Pixels before and after the transform of the low distortion reversible watermarking algorithm. (a) Before. (b) After.

E. LOW DISTORTION REVERSIBLE WATERMARKING ALGORITHM

We use a low distortion reversible watermarking algorithm [24] to embed the overflow-related information. This algorithm can embed 1 bit into each 2×2 block through a reversible transform. Fig. 4 shows the pixels in a 2×2 block before and after the transform.

Denote b ($b = 0$ or 1) as the bit to be embedded. Then we get the predicted value of x , $\hat{x} = n + w - nw$. The error of prediction is $p = x - \hat{x}$. Calculate $p_b = p + b$. Then, the pixels after the transform will be

$$\begin{aligned}
 X &= x + \left\lfloor \frac{p_b}{4} \right\rfloor, \\
 N &= n - \left\lfloor \frac{p_b + 3}{4} \right\rfloor, \\
 W &= w - \left\lfloor \frac{p_b + 1}{4} \right\rfloor, \\
 NW &= nw + \left\lfloor \frac{p_b + 2}{4} \right\rfloor.
 \end{aligned} \tag{4}$$

This transform is reversible as the following equations show:

$$\begin{aligned}
 \hat{X} &= N + W - NW \\
 &= \hat{x} - \left\lfloor \frac{p_b + 3}{4} \right\rfloor - \left\lfloor \frac{p_b + 1}{4} \right\rfloor - \left\lfloor \frac{p_b + 2}{4} \right\rfloor, \\
 P &= X - \hat{X} \\
 &= x - \hat{x} + \left\lfloor \frac{p_b + 3}{4} \right\rfloor + \left\lfloor \frac{p_b + 1}{4} \right\rfloor + \left\lfloor \frac{p_b + 2}{4} \right\rfloor + \left\lfloor \frac{p_b}{4} \right\rfloor \\
 &= p + (p + b) \\
 &= 2p + b.
 \end{aligned} \tag{5}$$

The recovered bit is

$$b = (X - \hat{X}) - 2 \times \left\lfloor \frac{X - \hat{X}}{2} \right\rfloor, \quad (6)$$

then

$$\begin{aligned} p &= \frac{X - \hat{X} - b}{2}, \\ p_b &= p + b. \end{aligned} \quad (7)$$

And the recovered original pixels are

$$\begin{aligned} x &= X - \left\lfloor \frac{p_b}{4} \right\rfloor, \\ n &= N + \left\lfloor \frac{p_b + 3}{4} \right\rfloor, \\ w &= W + \left\lfloor \frac{p_b + 1}{4} \right\rfloor, \\ nw &= NW - \left\lfloor \frac{p_b + 2}{4} \right\rfloor. \end{aligned} \quad (8)$$

F. CYCLIC REDUNDANCY CHECK (CRC)

CRC [25] is a commonly-used validation method in communication, and the length of its information field and authentication field can be arbitrarily selected.

The following example illustrates the process of CRC. Suppose that the information field is a seven-bit binary string “1100111.” Select a fixed generator polynomial, such as $G(x) = x^3 + x^2 + 1$, which can be converted to binary digital “1101.” First, move information fields left by the supreme power of $G(x)$ (which is 3 here). This means to add three 0s at the end of the information field and it becomes “1100111000.” Then, according to the binary division, divide the information field by “1101”, and finally get the three-bit remainder “111”, which is exactly the CRC check code of the authentication field.

When verifying, connect the information field with the authentication field as “1100111111”, and use the binary division to divide it by “1101.” If the remainder is 0, the information is correct. Otherwise, the information field or authentication field has been tampered.

III. LOW DISTORTION OVERFLOW PROCESSING ALGORITHM (LDOPA)

In order to handle the problem that overflow probably appears after embedding the watermark information, we design a low distortion overflow processing algorithm (LDOPA). It scans the image to find out the overflowed pixels and process them one by one, and generate the modification record. Then, it encodes the records and embed them into the carrier image.

A. OVERFLOW PROCESSING PROCESS

Suppose the carrier image is an $m \times n$ grayscale image, where $m, n \leq 512$, and there exists some overflowed pixels in the image after embedding watermark. Fig. 5 shows the process of handling overflow.

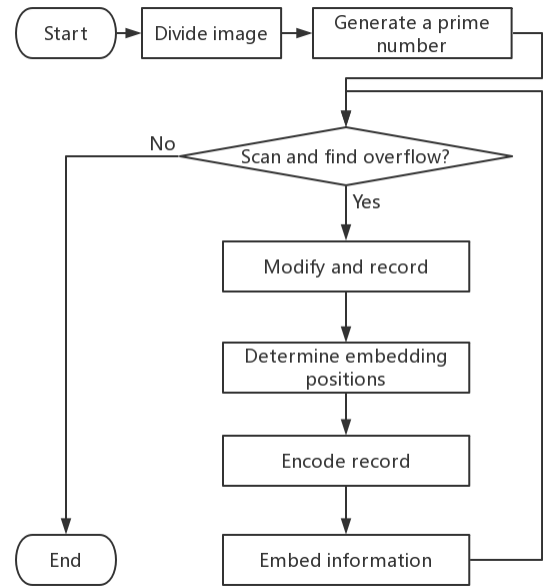


FIGURE 5. The flowchart of overflow processing process of LDOPA.

Step 1 (Divide Image): Divide the carrier image into 8×8 blocks (unit block), and number them 1 to M from left to right, from top to bottom. M is the amount of all unit blocks.

Step 2 (Generate a Prime Number): Choose a secret prime number k , where $k < M$.

Step 3 (Modify and Record): Scan every pixel of the image. Once an overflowed pixel is found, stop scanning immediately. Modify its value into a valid one, and generate a modification record. If no overflow found, jump to **Step 8**.

Suppose the pixel at row x column y , denoted as $P_t(x, y)$, is the overflowed pixel. The modified pixel $P'_t(x, y)$ will be

$$P'_t(x, y) = \begin{cases} 255, & \text{if } P_t(x, y) > 255 \\ 0, & \text{if } P_t(x, y) < 0 \end{cases} \quad (9)$$

and the modifier m is

$$m = \begin{cases} P_t(x, y) - 255, & \text{if } P_t(x, y) > 255 \\ P_t(x, y) - 0, & \text{if } P_t(x, y) < 0 \end{cases} \quad (10)$$

and the modification record $R = (x, y, m)$.

Step 4 (Determine Embedding Positions): Use the one-dimensional Torus mapping as (3), along with the prime number k to determine two unit blocks for embedding the modification record. For the j -th record, the order numbers of the two unit blocks are $p_1 = (2j - 1)k \bmod M + 1$ and $p_2 = 2jk \bmod M + 1$. We denote the blocks as $S(p_1)$ and $S(p_2)$.

Step 5 (Encode Record): Calculate $x' = x - 1, y' = y - 1$. Since $1 \leq x \leq m \leq 512$ and $1 \leq y \leq n \leq 512$, we have $x', y' \in [0, 511]$. Therefore, the location information x' and y' can be stored with 9 bits. Transform x', y' into 9-bit binary numbers, and transform $|m|$ into 7-bit binary number (fill the high empty bit(s) with 0). Suppose that x_i, y_i and m_i represent the i -th bit of the binary numbers of x', y' and $|m|$ respectively.

cx_1	cx_2	cx_3	s
x_1	x_2	x_3	m_1
x_4	x_5	x_6	m_2
x_7	x_8	x_9	m_3

(a)

cy_1	cy_2	cy_3	m_4
y_1	y_2	y_3	m_5
y_4	y_5	y_6	m_6
y_7	y_8	y_9	m_7

(b)

FIGURE 6. Auxiliary information matrix for a overflow modification record. (a) The first matrix. (b) The second matrix.

The sign bit that represents the sign of the modifier m is

$$sn = \begin{cases} 1, & \text{if } m < 0, \\ 0, & \text{otherwise.} \end{cases} \quad (11)$$

Then, joint $1, s, m_1$ to m_3, x_1 to x_9 as a long binary string, and calculate its CRC check code cx_1 to cx_3 with the generator polynomial $G(x) = x^3 + x^2 + 1$. In the same way, joint $1, m_4$ to m_7, y_1 to y_9 and calculate the check code cy_1 to cy_3 with $G(x)$. The reason why we introduce CRC is that, the check code can help to distinguish whether information is correct in the recovery process, and if not, we can make some efforts to decrease the impact of wrong records. Besides, that we add “1” before other bits to form the long binary strings is to handle the situation where the string starts with “0” or all the bits are “0.”

We define the combination of modification record information and the authentication information (the CRC check code) as the auxiliary information, and the two 4×4 auxiliary information matrixes made up by the bits we get above are shown in Fig. 6.

Step 6 (Embed Information): Use the low distortion watermarking algorithm we introduced before to embed the two matrixes in Fig. 6 into unit blocks $S(p_1)$ and $S(p_2)$. Since each 2×2 block supports 1 bit information, we can embed all the 16 bits of a matrix into an 8×8 unit block.

Step 7 Cycle the Process): Cycle **Step 3** to **Step 6** circularly, which is to iterate the process of scanning overflowed pixel, modifying it and embedding the record, until there is no overflowed pixel any longer.

Through the process above, we handled the overflow problem, not only the overflowed pixels originally existing in the image, but also the ones appear during the process. We also stored all the records properly, so the reversibility is ensured. In addition, the LDOPA also supports overlapping embedding, that is, even after we run out of all the unit blocks in the image, the remain records can still be embedded into the used unit blocks, overlapping the previous one. Thus, there is no limit for the amount of overflowed pixels $N_{overflow}$.

B. OVERFLOW RECOVERY PROCESS

The recovery process is to restore the pixels back to their overflowed values. The brief description is as follows:

Step 1 (Input Parameters): Require user to input the amount of overflowed pixels $N_{overflow}$ and the prime number k .

Step 2 (Divide Image): Divide the carrier image into unit blocks and number them in the same way as before.

Step 3 (Determine Positions): Use Torus mapping in the same way to get the unit blocks of a modification record. At the first time, we should determine the two blocks for the last modification record according to $N_{overflow}$.

Step 4 (Extract Information): Use the inverse transform of the low distortion watermarking algorithm to extract the auxiliary information from the two unit blocks. Decode and rebuild the modification record R . During the process, restore the original values of each unit blocks.

Step 5 (Verify and Recover): Verify the extracted information. If correct, recover the overflowed pixel, otherwise drop the record. The verification includes two checks: (1) CRC: check whether the auxiliary information can pass the CRC; (2) boundary check: check whether the recorded pixel has the value at the corresponding boundary, that is, when the sign bit sn is 1, whether the pixel value is equal to 0; when the sign bit sn is 0, whether the pixel value is equal to 255. Only when both the two checks pass can the information be correct, and then recover the overflowed value of the pixel.

Step 6 (Cycle the Process): Run **Step 3** to **Step 5** repeatedly, and recover every record from the last one to the first one, until all the $N_{overflow}$ records done.

IV. REVERSIBLE WATERMARKING SCHEME

For definiteness and without loss of generality, we use grayscale images of size 512×512 as the carrier images, and a grayscale image of different sizes as the watermark image. First, we introduce the embedding process.

A. EMBEDDING PROCESS

In the embedding process, we adopt wavelet histogram shifting to embed watermark information, and use the LDOPA to solve the overflow problem. We changed the process of wavelet histogram shifting, and all we have to do is complete embedding a bit sequence of a specific length into each unit block. Let’s first illustrate the specific process of wavelet histogram shifting for each unit block in detail.

	1	2	3	4	5	6	7	8
1	55	62	71	91	-8	7	-3	47
2	59	75	92	102	0	-6	-4	49
3	74	89	99	110	0	0	4	45
4	104	110	116	133	-6	5	1	61
5	1	4	-2	7	-1	2	-2	-1
6	1	3	-6	-1	4	7	-1	-6
7	-7	-1	-1	-2	-2	-1	1	-3
8	47	49	51	59	-3	2	0	27

FIGURE 7. Wavelet coefficients of one unit block.

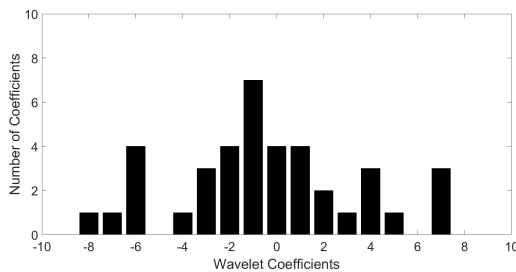


FIGURE 8. Wavelet coefficients histogram of the unit block in Fig. 7.

For instance, Fig. 7 shows the wavelet coefficients of a unit block in 512×512 “Lena” image. First of all, we define the area in subband HL, LH, and HH except the shaded parts as the embeddable area C . Denote $C_{i,j}$ as the coefficient at row i column j in the unit block, the embeddable area will be

$$C = \{c_{i,j} | i \in \{5, 6, 7\} \text{ or } j \in \{5, 6, 7\}\} \quad (12)$$

As we can see, in the area C of Fig 7, the absolute values of wavelet coefficients are small and closed to the zero value. This feature does not fit for any unit block in any image, but all the small absolute values of coefficients in any unit block assemble in the scope of area C .

We only do statistics over and modify the coefficients in the embeddable area, for doing which we can, on the one hand, avoid modifying the big coefficients, thus decrease the extent of pixel change, reduce overflow and improve the quality, on the other hand, reduce the number of modified coefficients and improve the algorithm efficiency. Fig. 8 shows the histogram of the wavelet coefficients of the embeddable area C of Fig. 7.

When embedding, first, shift the part greater than 0 towards right by 1 unit, so the histogram transforms to Fig. 9.

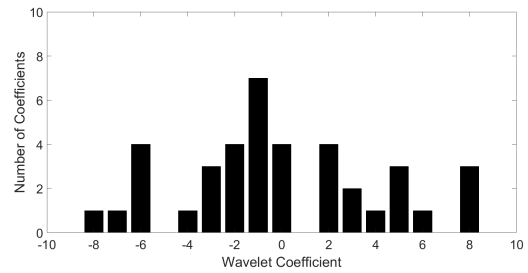


FIGURE 9. Wavelet coefficients histogram after the first move from Fig. 8.

Then, embed bits at coefficient 0 in the way we introduced before. After all the wavelet coefficients with value 0 are used, and if there is still more information to be embedded, shift the part less than 0 of the histogram towards left by 1 unit, and continue embedding at the wavelet coefficients with value -2 (the value was -1 before the second move). Repeat shifting the right part and the left part of the histogram alternatively and embed until all the bits for the unit block is embedded, and finish the embedding mission in this unit block.

Next, let’s introduce the whole process of the embedding process, as is shown in Fig. 10.

Step 1 (Divide Image): Divide the carrier image P into 8×8 blocks (unit blocks), and number them from left to right, from top to bottom. M is the total amount of all unit blocks, which is 4096 in our example.

Step 2 (Generate Parameters): Generate several secret parameters for the later steps, including the initial value of Logistic model x_0 , and two prime numbers k_1, k_2 . We use the following method to generate these parameters:

- 1) Require the user to input the password of any length s ;
- 2) Use MD5 hash function to generate the 32-bit digest of s , denoted as a , where all the letters are lowercase;
- 3) Get the ASCII codes for each letter in a , to form an ASCII code vector a' ;
- 4) Calculate $r = \prod_{i=2k+1} a'_i, k \in N \text{ and } \in [1, 15]$;
- 5) Calculate $x_0 = r \times 10^{-n}$, where n is the smallest natural number that let x_0 be in the range $[0.1, 1]$;
- 6) Calculate the product of the 4th and the 5th number in a' as m_1 . Denote the largest prime number that less than $m_1/24$ as k_1 ; calculate the product of the 12nd and the 13rd number in a' as m_2 . Denote the largest prime number that less than $m_2/13$ as k_2 .

In this way, the parameters are all related to the password s , and they will determine the encryption process or the embedding positions in the following steps.

Step 3 (Encrypt Watermark): Use Logistic mapping, as (1), to encrypt the watermark. The state indicator μ is set to 4 to make the system chaotic. The initial value x_0 is what we generate in the last step. For example, if the password s is “Dalian, Liaoning, China”, we get $x_0 = 0.308397$. And a 64×64 grayscale watermark image and its encrypted version are shown in Fig. 11. After the encryption, the content becomes not recognizable, so the effect is good.

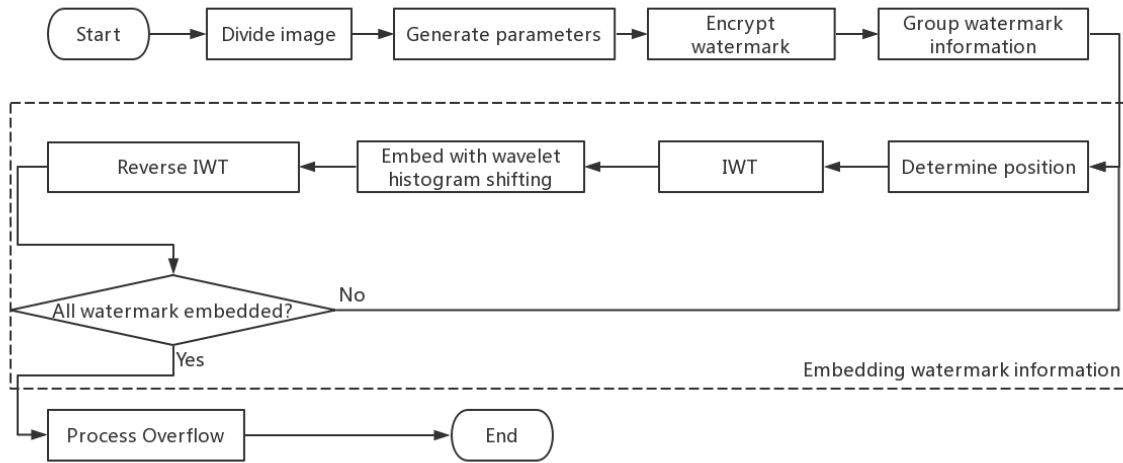


FIGURE 10. Flowchart of the embedding process of the reversible watermarking scheme.



FIGURE 11. Watermark image and the encrypted watermark image. (a) Watermark image (64 × 64). (b) Encrypted watermark image.

TABLE 1. Experiment for testing the password mechanism.

Experiment	Test of the password mechanism
Carrier image	“Lena”
Watermark image	Fig. 11(a)
Pwd_1 (right password)	“Dalian, Liaoning, China”
Pwd_2 (wrong password)	“Dalian, Liaonin, China”
Operations	1. Embed the watermark image on the carrier image, with password set as Pwd_1 . 2. Extract the watermark image from the carrier image, with password set wrongly as Pwd_2 . 3. Compare the original watermark image and the extracted watermark image.

Step 4 (Group Watermark Information): Transform the value of each pixel in the watermark image into binary number, and connect them as a binary sequence. Divide them into groups in sequence. Except the last group, every group contains t bits, and the last group contains all the remained bits.

$$t = \left\lceil \frac{n_w}{M} \right\rceil \quad (13)$$

where n_w is the amount of watermark information bit. $\lceil x \rceil$ represents rounding up to the bigger integer. Denote all the watermark information groups as I , and I_i represents the i -th group.

TABLE 2. Experiment for assessing the distortion of LDOPA.

Experiment	Test of distortion level of LDOPA
Carrier image	“Peppers” (an easily-overflowed image)
Watermark image	Fig. 11(a) of different size (payload)
Operations	1. Embed the watermark image of different size (different payload) on the carrier image. Record the amount of overflowed pixels for each payload. 2. Apply LDOPA to process the overflow problem. 3. Calculate PSNR and SSIM between image with overflowed pixels and image processed with LDOPA. 4. Plot the result.

TABLE 3. Experiment for testing the reversibility of LDOPA.

Experiment	Test of reversibility of LDOPA
Carrier image	“Lena”, “Plane”, “Peppers”, “Baboon”
Watermark image	Fig. 11(a)
Operations	1. Embed the watermark image on different carrier images. 2. Apply LDOPA to process the overflow problem. 3. Apply the inverse process of LDOPA to recover the overflowed pixels. 4. Compare the image with overflow (before processing) and the overflow-recovered image (after recovering) to check whether they are the same.

Step 5 (Embedding Watermark Information): Embed the groups I into unit blocks. For each group I_i :

- 1) Determine position: Use the Torus mapping, with the prime number k_1 , to get the order of a unit block $P(i) = k_1 i \bmod 4096 + 1$. Denote the corresponding unit block as $S(P(i))$;
- 2) IWT: Implement integer wavelet transform on $S(P(i))$;
- 3) Embed with wavelet histogram shifting: Use wavelet histogram shifting to embed the watermark information

TABLE 4. Result of processing and recovering overflowed pixels with LDOPA. The overflowed pixels are highlighted with 9×9 white blocks.

Image name	Amount of overflowed pixels	Image with overflow (before processing)	Overflow-processed image (after processing)	Overflow-recovered image (after recovering)
Lena	2			
Plane	46			
Peppers	171			
Baboon	7			

group I_i on the embeddable area C of the unit block $S(P(i))$;

- Reverse IWT: Implement the reverse IWT on unit block $S(P(i))$;

Step 6 (Process Overflow): Use the LDOPA to make all the pixel values become in the valid range, and finally generate the image with watermark information P' .

B. EXTRACTING AND RECOVERING PROCESS

The extracting and recovering process is the reverse process of embedding. Its simple steps are as follow:

Step 1 (Divide Image): Divide the image with watermark P' into unit blocks and number them in the same way as before.

Step 2 (Generate Parameters): Require user to input the password, and with the same method to generate x_0 , k_1 and k_2 .

Step 3 (Recovering Overflow): Follow the recovering process of LDOPA to recover the overflowed pixels. Use k_2 as the necessary prime number in this process.

Step 4 (Extracting Watermark): Use Torus mapping and prime number k_1 to determine the location of unit blocks one by one. Use wavelet histogram shifting to recover the

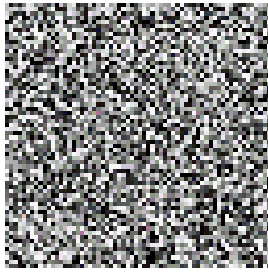


FIGURE 12. The extracted watermark image using a wrong password.

watermark information and the original coefficients where watermark is embedded.

Step 5 (Rebuild and Decrypt the Watermark): Rebuild the watermark information and recover the watermark image. Use x_0 as the initial value of Logistic mapping to decrypt the watermark, and get the extracted watermark image W' .

V. EXPERIMENTS AND ANALYSIS

A. SECURITY ANALYSIS

1) TEST OF PASSWORD MECHANISM

The proposed watermarking scheme requires the user to provide password of arbitrary length, which is used for encrypting and determining locations for embedding. When recovering, watermark image and carrier image can be completely restored only if user provide the same password. To prove this, we implement an experiment of Table 1. The extracted watermark with wrong password is shown in Fig. 12.

As we can see, even the password is slightly different from the correct one, the content of the extracted watermark is unrecognizable, which proves that the encryption effect is good, and the password mechanism works well.

2) OTHER TECHNIQUES

We applied many techniques to improve the security of the scheme:

First of all, we used MD5 hash function to generate the initial value x_0 of Logistic mapping, and prime number k_1, k_2 . Due to the collision resistance of hash function, the different password, even if the difference is tiny, will result in a completely different digest, and generate completely different parameters x_0, k_1 and k_2 . Thus, the security of secret parameters generation is ensured. In addition, since MD5 supports any length of data as input and outputs a digest of fixed length, the length of the user’s password is unrestricted, which is also simple to operate.

We also used the Logistic mapping to disarrange the watermark, which makes the information contained in the watermark be disturbed. The sensitivity towards the initial value allows that slight difference of the initial value have a great impact on the result of scrambling. Even if the watermark is extracted, the attacker still cannot know the original information of the watermark.

Besides, the locations of watermark information and the auxiliary information are disarranged further with

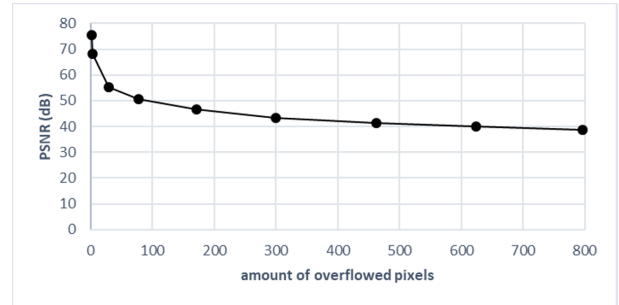


FIGURE 13. PSNR of overflow processing of "Peppers"

TABLE 5. Experiment for testing the reversibility of the watermarking scheme.

Experiment	Test of reversibility of the watermarking scheme
Carrier image	"Lena", "Plane", "Peppers", "Baboon" (512×512)
Watermark image	Fig. 11(a)
Password	"Dalian, Liaoning, China"
Operations	<ol style="list-style-type: none"> 1. Embed the watermark image on different carrier images with the proposed watermarking scheme. 2. Extract the watermark and recover the watermark image with the inverse process of the watermarking scheme. 3. Compare the original carrier image and the recovered image to check whether they are the same.

Torus mapping. Locations are determined by the prime number generated by the user password. The scattered information improved the invisibility and reduced the impact on extracting watermark if a piece of the image is attacked.

Finally, we proposed a low distortion overflow processing algorithm (LDOPA). This algorithm is a good solution to overflow problem, and the verification mechanism helps to judge the correctness of information and avoid the larger damage caused by wrong overflow records.







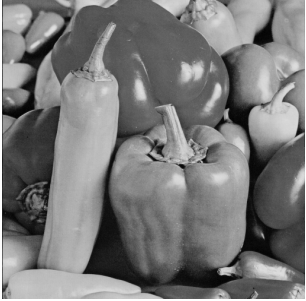
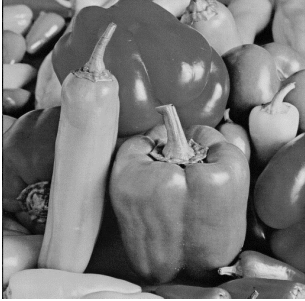

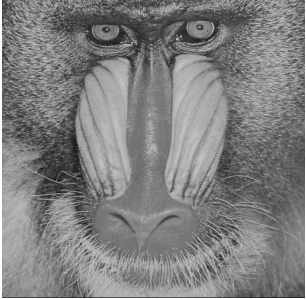
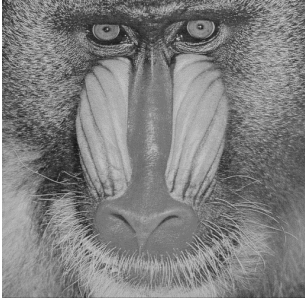
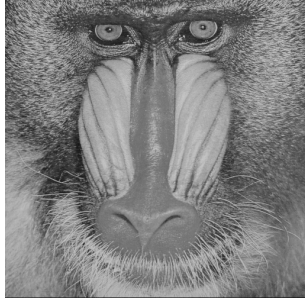
B. LDOPA PERFORMANCE ANALYSIS

1) TEST OF DISTORTION OF LDOPA

Different watermark payload embedded in different carrier images will lead to different amount of overflowed pixels. For some images, even a large payload will not cause overflow, while for others, a small payload will lead to overflowed pixels. To assess the distortion level of LDOPA, we implement an experiment of Table 2. Fig. 13 shows PSNR, and Fig. 14 shows SSIM, with respect to different amount of overflowed pixels.

From the result, we can get that, when there are only a few overflowed pixels, the PSNR is at a high level, yet it decreases with the increasement of the amount. However, even the amount increases up to 800, the PSNR remains larger than 35, which means the difference is hard to be recognized. Besides, the SSIM remains larger than 97% all the time, which also proves the low distortion of LDOPA. Considering that for

TABLE 6. Result of reversibility test on different carrier images.

Image name	Original image	Image with watermark	Recovered image
Lena			
Plane			
Peppers			
Baboon			

most carrier images, there will not be too many overflowed pixels after embedding the watermark, the effect of LDOPA is acceptable.

2) TEST OF REVERSIBILITY OF LDOPA

For testing the reversibility of LDOPA, we implement the experiment of Table 3. The result of processing and recovering overflow is shown in Table 4, where the third column shows the images with overflow after embedding watermark but before processing overflow with LDOPA, the fourth column shows the images after processing overflow with LDOPA, and the last column shows the images recovered by

the inverse process of LDOPA. In order to see the overflowed pixels clearly, we highlight each overflowed pixels with a 9×9 white block.

As we can see, after the processing of LDOPA, the image (in the fourth column) contains no overflowed pixel, either those originally existing in images or those caused by embedding the modification record during the process of LDOPA. Besides, the position and the amount of overflowed pixels in the “before processing” image (third column) and the “after processing” image (fifth column) are exactly the same, which proves that the LDOPA is completely reversible.

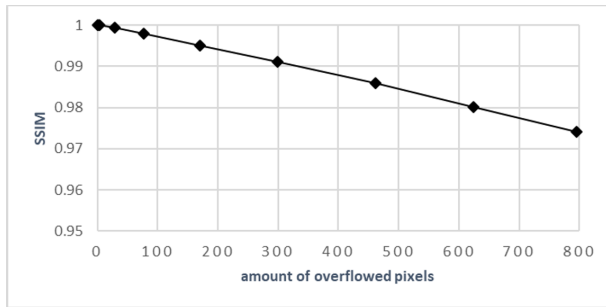


FIGURE 14. SSIM of overflow processing of “Peppers.”

TABLE 7. Experiment for assessing the quality of the watermarking scheme.

Experiment	Test of reversibility of the watermarking scheme
Carrier image	“Lena”, “Plane”, “Peppers”, “Baboon” (512×512)
Watermark image	Fig. 11(a) of different size
Operations	<ol style="list-style-type: none"> 1. Embed the watermark image of different size on different carrier images with the proposed watermarking scheme. 2. Calculate PSNR and SSIM between the original carrier image and the image with watermark embedded. 3. Plot the result

In addition, the LDOPA adopts a different embedding algorithm from the one for embedding watermark information, and overlaps the auxiliary information onto the image with watermark information, so it does not occupy the embedding watermark. Therefore, the LDOPA is proved to be a new effective way for handling overflow problem in watermarking.

C. REVERSIBILITY ANALYSIS

To demonstrate the proposed watermarking scheme is reversible, we implement the experiment of Table 5. The result is shown in Table 6, where the second column shows the original carrier images, the third column shows the images with watermark embedded using the proposed entire watermarking scheme, and the fourth column shows the recovered carrier images with the inverse process of the scheme.

As the result tells, the recovered images are always exactly the same as the original ones, and the extracted watermark images are also completely the same as the original one. Thus, the reversibility of the scheme has been proved.

D. QUALITY ANALYSIS

We implement the experiment of Table 7 to assess the quality of the watermarking scheme. Fig. 15 shows PSNR, Fig. 16 shows SSIM, with respect to the different payload.

As we can see, when the payload is low, the PSNR is at a high level, and it decreases with the increasing payload. For most images, the PSNR remains over 30 dB, and the

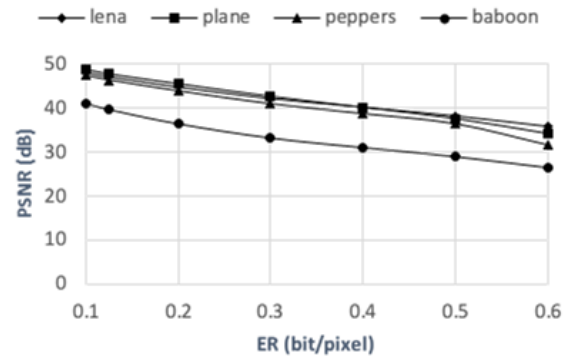


FIGURE 15. PSNR under different watermark payload.

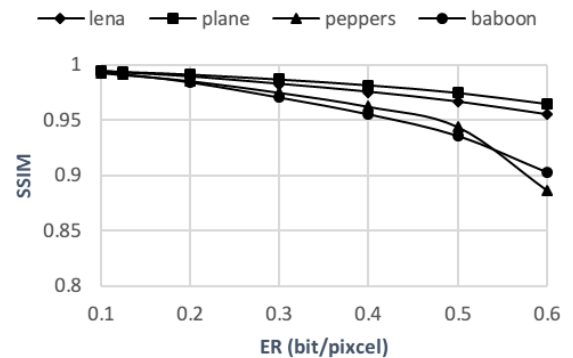


FIGURE 16. SSIM under different watermark payload.

TABLE 8. Experiment for testing the robustness: modification and cutting.

Experiment	Test of robustness: modification and cutting
Carrier image	“Lena” (512×512)
Watermark image	Fig. 11(a)
Operations	<ol style="list-style-type: none"> 1. Embed the watermark image on the carrier images with the proposed watermarking scheme. 2. Implement some modifications or cutting over the image with watermark embedded. 3. Extract the watermark and compare it with the original one.

SSIM remains over 90% when ER is up to 0.6, which means it is hard for human to recognize the existence of the watermark. Therefore, there is still room for increasing the quality further, yet in most circumstances, the quality of the watermarking scheme is acceptable.

E. ROBUSTNESS ANALYSIS

1) TEST OF MODIFICATION AND CUTTING

To test the robustness of the scheme, firstly, we implement the experiment of Table 8. The modifications and cutting, as well as their corresponding extracted watermark image, are displayed in Table 9.

From the result, we can get that, even we modify a lot, or remove a large part of the image, the extracting process

TABLE 9. Result of robustness test, adding some modification or cut over the image and extract the watermark.



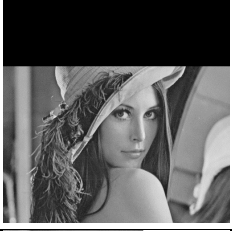
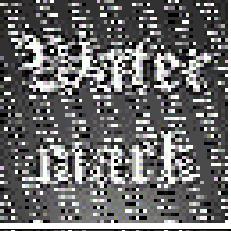

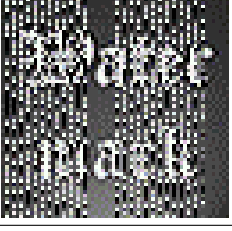
Attack	Modified image	Extracted watermark
Modify		
Cut 1		
Cut 2		

TABLE 10. Experiment for testing the robustness: noise.

Experiment	Test of robustness: noise
Carrier image	“Lena” (512×512)
Watermark image	Fig. 11(a)
Operations	<ol style="list-style-type: none"> 1. Embed the watermark image on the carrier images with the proposed watermarking scheme. 2. Add some salt & pepper noises, with different variance, to the image with watermark embedded. 3. Extract the watermark and compare it with the original one.






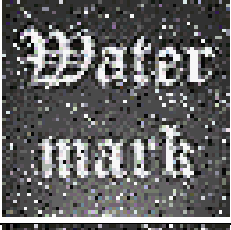


can still work, and the recovered watermark remains recognizable. That is because we applied Logistic mapping and Torus mapping in our scheme. They effectively distributes the watermark information in the image, so the loss of a large part of the image will not hinder the most and other information to be restored.

2) TEST OF NOISE

Next, we implement an experiment of Table 10 to value the robustness against noise. The result is shown in Table 11, which shows the extracted watermark images extracted from images with salt & peppers noises of different variance added.

As we can see, even we add the noise with a large variance up to 0.02, the watermark can still be recognized. Therefore, the scheme is capable to resist some attacks. The robustness of our watermarking scheme is satisfying.

TABLE 11. Result of robustness test, adding salt & peppers noises on the image and extract the watermark.

Variance	Modified image	Extracted watermark
0.001		
0.005		
0.01		
0.02		

VI. CONCLUSION

We proposed a new reversible watermarking scheme with a new overflow solution. The main idea is to embed twice – First, embed watermark information with wavelet histogram shifting. Then, embed overflow modification records to solve the overflow problem with a low distortion reversible watermarking algorithm. The biggest contribution of this paper is that we designed a new low distortion overflow processing algorithm (LDOPA). It has a low distortion, with SSIM always greater than 97% even if the amount of overflowed pixels is up to 800. Meanwhile, it is proved that LDOPA is completely reversible, therefore it can be used in reversible watermarking techniques. Different from general overflow solutions that operates before embedding or during embedding, the new solution is implemented independently after embedding watermark. So, LDOPA can be used for any reversible watermarking algorithm theoretically. In addition, we embed the watermark information evenly into the blocks of the carrier image, and no threshold should be determined ahead of time, so the operation is simple, and the application of encryption and scrambling methods ensures the security of the watermark.

The whole process of reversible watermarking scheme ensured the complete reversibility. Carrier image and watermark image can be recovered lossless if the image is not damaged. Thus, the new scheme can be applied in the fields where high integrity of multimedia data is required, such as medical images, military image, judicial authentication, etc.

REFERENCES

- [1] R. G. Van Schyndel, A. Z. Tirkel, and C. F. Osborne, "A digital watermark," in *Proc. 1st IEEE Int. Conf. Image Process.*, vol. 2, Nov. 1994, pp. 86–90.
- [2] N. Thilagavathi, D. Saravanan, S. Kumarakrishnan, S. Punniakodi, J. Amudhavel, and U. Prabu, "A survey of reversible watermarking techniques, application and attacks," in *Proc. Int. Conf. Adv. Res. Comput. Sci. Eng. Technol. (ICARCSET)*, 2015, Art. no. 37.
- [3] J. Tian, "Reversible data embedding using a difference expansion," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 8, pp. 890–896, Aug. 2003.
- [4] A. M. Alattar, "Reversible watermark using the difference expansion of a generalized integer transform," *IEEE Trans. Image Process.*, vol. 13, no. 8, pp. 1147–1156, Aug. 2004.
- [5] D. M. Thodi and J. J. Rodríguez, "Reversible watermarking by prediction-error expansion," in *Proc. 6th IEEE Southwest Symp. Image Anal. Interpretation*, Mar. 2004, pp. 21–25.
- [6] Z. Ni, Y.-Q. Shi, N. Ansari, and W. Su, "Reversible data hiding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 16, no. 3, pp. 354–362, Mar. 2006.
- [7] G. Xuan et al., "Lossless data hiding using histogram shifting method based on integer wavelets," in *Proc. Int. Workshop Digit. Watermarking*. Berlin, Germany: Springer, 2006, pp. 323–332.
- [8] X. Wu, "Reversible semi-fragile watermarking based on histogram shifting of integer wavelet coefficients," in *Proc. Inaugural IEEE-IES Digit. EcoSyst. Technol. Conf.*, Feb. 2007, pp. 501–505.
- [9] C.-C. Lin, W.-L. Tai, and C.-C. Chang, "Multilevel reversible data hiding based on histogram modification of difference images," *Pattern Recognit.*, vol. 41, no. 12, pp. 3582–3591, 2008.
- [10] K.-S. Kim, M.-J. Lee, H.-Y. Lee, and H.-K. Lee, "Reversible data hiding exploiting spatial correlation between sub-sampled images," *Pattern Recognit.*, vol. 42, no. 11, pp. 3083–3096, Nov. 2009.
- [11] D. M. Thodi and J. J. Rodríguez, "Expansion embedding techniques for reversible watermarking," *IEEE Trans. Image Process.*, vol. 16, no. 3, pp. 721–730, Mar. 2007.
- [12] C.-F. Lee, H.-L. Chen, and H.-K. Tso, "Embedding capacity raising in reversible data hiding based on prediction of difference expansion," *J. Syst. Softw.*, vol. 83, no. 10, pp. 1864–1872, 2010.
- [13] H.-W. Tseng and C.-P. Hsieh, "Prediction-based reversible data hiding," *Inf. Sci.*, vol. 179, no. 14, pp. 2460–2469, 2009.
- [14] X. Deng, Z. Chen, H. Liu, and P. Jian, "Study on preventing overflow and underflow of reversible digital watermarking for medical images," *Comput. Eng. Appl.*, vol. 49, no. 23, pp. 162–165, 2013.
- [15] W. Zhao, Q. Wu, A. Reinthal, and N. Zhang, "Design, implementation, and field testing of a privacy-aware compliance tracking system for bedside care in nursing homes," *Appl. Syst. Innov.*, vol. 1, no. 1, p. 3, 2017.
- [16] W. Zhao, M. A. Reinthal, D. D. Espy, and X. Luo, "Rule-based human motion tracking for rehabilitation exercises: Realtime assessment, feedback, and guidance," *IEEE Access*, vol. 5, pp. 21382–21394, 2017.
- [17] M. Li and X.-F. Liao, "A new method for digital watermarking of wavelet transform combined chaos," *Comput. Sci.*, vol. 34, no. 8, pp. 245–247, 2007.
- [18] W. Tang and X. Chen, "Chaos theory and research on its applications," *Autom. Electr. Power Syst.*, vol. 24, no. 7, pp. 67–70, 2000.
- [19] L. Chen, H. Liu, and X. Deng, "A digital image reversible watermarking algorithm based on integer wavelet transform," *Comput. Appl. Softw.*, vol. 33, no. 4, pp. 286–291, 2016.
- [20] K. Li et al., "Application of logistic mapping in digital image encryption algorithm," *Inf. Commun.*, vol. 2017, pp. 139–140, Jan. 2017.
- [21] W. Zhao et al., "A human-centered activity tracking system: Toward a healthier workplace," *IEEE Trans. Human-Mach. Syst.*, vol. 47, no. 3, pp. 343–355, Jun. 2017.
- [22] M. Antonini, M. Barlaud, P. Mathieu, and I. Daubechies, "Image coding using wavelet transform," *IEEE Trans. Image Process.*, vol. 1, no. 2, pp. 205–220, Apr. 1992.
- [23] H. Yingqing, J. Xiangqi, and Z. Zhiqian, "Distribution of wavelet coefficients and the effect to the image quality," *Infr. Laser Eng.*, vol. 29, no. 3, pp. 15–18, 2000.
- [24] Y. Wang, "Research on robust and reversible watermarking algorithm," M.S. thesis, Dept. Electron. Eng., Jinan Univ. Guangzhou, Guangdong, China, 2016.
- [25] W. W. Peterson and D. T. Brown, "Cyclic codes for error detection," *Proc. IRE*, vol. 49, no. 1, pp. 228–235, Jan. 1961.



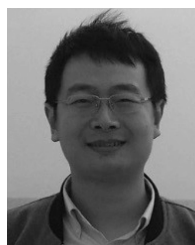
BIN FENG received the B.S. degree in computer science and technology from Liaocheng University, Liaocheng, Shandong, China, in 2002, and the M.S. degree in software engineering and the Ph.D. degree in computer application technology from the Dalian University of Technology, Dalian, China, in 2006 and 2015, respectively.

He was an Assistant with Taishan University from 2002 to 2004, where he has been a Professor with Taishan University 2018. He was a Full Engineer of computer science with the Dalian University of Technology. His research interests include data hiding, image processing, and network and information security.



BOTAO YU was born in Changsha, Hunan, China, in 1997. He is currently pursuing the bachelor's degree with the School of Software Technology, Dalian University of Technology, Dalian, Liaoning, China.

From 2016 to 2018, he was a Research Assistant with the Feng's Academic Team. He has been researching on digital watermarking techniques, especially on reversible watermarking algorithms. His research interests include data hiding, big data, data mining, and machine learning.



YILIN BEI received the B.S. degree in computer science and technology from Shandong Normal University, Shandong, China, in 2003, and the M.S. degree in computer software and theory from Liaoning Normal University, Dalian, China, in 2008.

He is currently an Associate Professor with Taishan University. His research interests include digital watermarking, virtual reality, and information security.



XINQIANG DUAN received the B.S. degree in computer science and technology from Liaocheng University, Liaocheng, Shandong, China, in 2002, and the M.S. degree in computer science and engineering from Jiangsu University, Zhenjiang, Jiangsu, China, in 2009.

He is currently a Full Teacher of information science and technology with Taishan University. His research interests include data hiding, big data, and information security.

• • •