Multimedia Tools and Applications (2020) 79:31239–31264
https://doi.org/10.1007/s11042-020-09548-4

A novel triple recovery information embedding approach for self-embedded digital image watermarking

Ertugrul Gul[1,2] · Serkan Ozturk[2]

Received: 13 January 2020 / Revised: 1 July 2020 / Accepted: 6 August 2020 /
Published online: 19 August 2020

Abstract
Image tamper detection and recovery has become an important issue in recent years. In order to detect and recover high tampering rated images, this paper presents a novel self-embedded fragile watermarking method with triple recovery information embedding approach. In this proposed method host image is divided into sixteen main blocks. Four partner blocks are selected from main blocks as a partner group using Look-Up table which is constructed for recovery against high tampering rated images. Each partner block is divided into 4×4 blocks. Recovery data of each partner block is generated from the mean values of the 4×4 blocks. Then, triple recovery information for each partner block is constructed by combining recovery data of the three other partner blocks. After the generation of the recovery information, each partner block is divided into 16×16 blocks. Then, these blocks are divided into four 8×8 sub-blocks. Recovery bits for each partner block are embedded into the first and second least significant bits (LSBs) of the first three sub-blocks. It means that triple copies of recovery information for each partner block are embedded into three other blocks in the image, which provides triple chance for recovery of the tampered areas. On the other hand, authentication information is generated from the pixels of the 16×16 recovery information embedded image blocks by using MD5 hash function. This authentication information is embedded into the quarter part of each 16×16 block by using LSB substitution. The performance of the proposed method has been demonstrated by applying different size of cropping attacks to the different areas of the watermarked images. Experimental results show that the proposed method satisfactory recovers up to 75% tampering rated images.

Keywords Integrity · Self-embedding · Watermarking · Triple recovery

✉ Ertugrul Gul
ertugrulgul@erciyes.edu.tr

Serkan Ozturk
serkan@erciyes.edu.tr

Extended author information available on the last page of the article

1 Introduction

In recent years, due to advances in internet technology and powerful image editing tools, protection of image integrity has become an important challenge in many application areas where manipulating content may lead to inadmissible consequences [5]. In order to authenticate the image content, digital signature based cryptographic methods have been developed [39, 51]. In signature-based methods, authentication is performed by comparing the attached signature information with the re-generated signature information. However, this type of methods cannot localize the tampered areas of the images [31]. Therefore, in order to overcome this challenge, researchers have proposed fragile watermarking methods [7, 9, 12, 38, 57] for image integrity protection and tamper detection. In fragile watermarking methods, a watermark is embedded into the image to detect all kind of manipulations and to localize the tampered areas.

The fragile watermarking methods only focus on detection of the tampered areas of the altered images [46]. However, detection of manipulated areas is not enough for many applications. Therefore, many researchers have proposed self-embedding watermarking methods [5, 11, 16, 31, 37, 47, 48] which recover manipulating areas after tamper localization. In self-embedding fragile watermarking methods, authentication and recovery information are generated from the image as watermarks and these watermarks are embedded into the image itself. Authentication information is used for tampering detection, while recovery information is used for reconstruction of the tampered areas.

Most of the self-embedded watermarking methods have maximum 50% tolerable tampering rate. Therefore, the recovery capability of such methods is insufficient against large sized attacks. In this paper, in order to recover high tampering rated images, we propose a novel self-embedding watermarking method, in which each block in the image contains recovery information of three other blocks. In this proposed method host image is first divided into sixteen non-overlapping main blocks. Then, four partner blocks are selected from main blocks using Look-Up table which is constructed for recovery against 75% tampered images. The partner blocks are then divided into 4×4 sub-blocks. The mean values of these sub-blocks are combined as recovery data for each partner block. Then, for each partner block, triple recovery information is constructed by combining recovery data of the three other partner blocks. After the construction of the recovery information, each partner block is divided into 16×16 blocks. Then, these blocks are divided into four sub-blocks. Finally, recovery bits for each partner block are permuted with a key and then embedded into the first and second least significant bits (LSBs) of the first three sub-blocks. Then, in order to generate authentication information, the first and second LSBs of the pixels are set equal to zero in the fourth sub-blocks. The hash values of the four sub-blocks are generated with a secret key and a block number by using MD5 hash function. Finally, for each 16×16 block, 128-bit authentication information is embedded into the first and second LSBs of fourth sub-block using LSB modification.

The major contributions of this work are as follows: 1) we introduce a novel triple recovery information embedding approach for self-embedded watermarking, in which triple copies of recovery information for each block are embedded into the three other blocks in the image; 2) we successfully recover 75% tampering rated images; 3) we improve our earlier tamper detection method [12] by reducing the block sizes to increase the precision of the tamper

detection; 4) we apply six types cropping attacks size of 75%, 50%, and 25% to the watermarked images, and we successfully detect and recover the tampered areas. To the best of our knowledge, this is the first triple recovery information embedding approach for self-embedded watermarking.

The rest of this paper is organized as follows. Section 2 presents the related works. The proposed method is described in Section 3. Section 4 shows the experimental results. Section 5 concludes the paper.

2 Related work

Digital watermarking methods are mainly used in healthcare [25], remote sensing [22] and industrial [18, 24] application areas for the purpose of ownership identification [30], copyright protection [49], fingerprinting [32, 33, 36], monitoring [10, 19, 20] and forgery detection [12]. With respect to the watermark embedding domain digital watermarking methods like steganography [26, 27, 29, 35] can be generally classified as spatial domain methods and transform domain methods such as discrete Fourier transform [34], discrete cosine transform [1, 3, 23], discrete wavelet transform [2, 17, 21, 41, 42]. Digital watermarking methods can be also classified namely robust, semi-fragile and fragile according to application areas [1, 12]. Fragile watermarking methods have been proposed by many researchers for tampering detection. Chen and Wang [7] proposed a fuzzy c-means clustering-based fragile watermarking scheme for tamper proofing and image authentication. In this method, fuzzy c-means clustering technique was used to build a relationship between image blocks. Then, the authentication bits were produced by combining a secret-key-generated random sequence with a membership matrix. An adaptive image authentication method which detects manipulations for image vector quantization was presented by Chuang and Hu [9]. Authentication information was produced by using the pseudo random sequence in this method. The authentication information size and the number of indices were adaptively determined. He et al. [14] proposed a wavelet based fragile watermarking method for tamper localization, in which watermark was generated by using discrete wavelet transform and embedded into the LSBs of the host image. Lu et al. [40] presented LSB modification based fragile watermarking method, where watermark was generated with a random binary image by using exclusive OR operations and random permutations. Peng et al. [43] proposed reversible fragile watermarking method using two host images, in which a secret data was embedded into one host image while a distortion data was embedded into the other image. Gull et al. [13] presented a fragile watermarking method for tampering localization of medical images. In this method, tamper detection information was embedded into lower half block of the image and tamper localization information was embedded into the upper half block of the image.

Self-embedding fragile watermarking methods have also been proposed by many researchers for tampering detection and recovery. Lee and Lin [31] proposed dual watermarking method for tamper detection and recovery. In this method, two copies of recovery information of each block were embedded into other blocks in the image. Therefore, this method provided second chance for recovery in situation one copy was destroyed. Cao et al. [6] presented hierarchical recovery based self-embedding watermarking method. Binary information extracted from most significant bit (MSB) layers were scrambled and interleaved with extension

ratios. Then, the interleaved information and authentication bits were embedded into the LSB layers of the image blocks in this method. Flexible restoration quality based self-embedding watermarking method was proposed by Zang et al. [59], where recovery information was generated from five MSBs of host image and authentication information was derived from recovery information and MSBs. Sing and Sing [47] presented discrete cosine transformation based self-embedded fragile watermarking method. Two authentication bits and ten recovery bits generated from MSBs of 2×2 blocks were embedded into three LSBs of the image. Authentication bits were embedded in blocks itself while recovery bits were embedded in the corresponding mapped block in this method. Multilevel authentication based fragile watermarking method was proposed by Singh and Agarwal [48], in which image recovery domain was chosen dynamically. In this method, spatial domain was selected for smooth image blocks and frequency domain was selected for rough image blocks. Ansari et al. [4] proposed singular value decomposition based fragile watermarking method for tamper detection. In this method, singular values of 4×4 blocks and mean values of 2×2 blocks were used for tamper localization and image recovery. Qin et al. [45] presented reference data interleaving and adaptive embedding mode selection based self-embedding watermarking method, in which recovery bits were derived from the interleaved MSB bits of the host image and then were combined with authentication bits for LSB embedding. Bravo-Solorio et al. [5] presented an iterative restoration mechanism based fragile watermarking method, in which two independent recovery bits sequences were generated from five MSBs of the image. In this method, reference sequences and some authentication bits were embedded into the 3 LSBs of the host image. Qin et al. [46] proposed an overlapping embedding strategy based fragile watermarking method, where recovery bits generated from average value of each overlapping block were dispersedly embedded into one or two LSB layers of the host image according to diagonal and horizontal-vertical mode. Halftone based tampering detection and content recovery method was presented by Yang et al. [56], in which watermark including authentication bits and recovery bits was only embedded one LSB layer of the host image. Huang et al. [16] proposed a self-embedding fragile watermarking method based on adaptive payload. In this method, image blocks were classified into region of interest (ROI) and background by using graph-based visual saliency model. ROI blocks were operated with superior priority for extracting backup information and allocating payload. Kim et al. [28] proposed an absolute moment block truncation based self-embedding watermarking method, in which compressed version of image was embedded as a watermark. In this method, two LSBs of the host image pixels were used for watermark embedding.

Fragile watermarking methods can only detect the tampered regions of the images. However, in most of the applications, detecting the tampered regions cannot satisfy the requirements. On the other hand, most of the self-embedded watermarking methods are insufficient against large sized attacks because of having low tolerable tampering rate. These methods generally can recover at most 50% tampered images. In this work, in order to overcome these deficiencies, we propose a high tolerable tampering rated self-embedding watermarking method. In our proposed method, each block in the image contains recovery information of three other blocks. It means that the triple copies of recovery information for each block are embedded into the three other blocks in the image. Therefore, the proposed method can successfully recover 75% tampered images. In addition, in order to increase the precision of the tamper detection, we have improved our earlier tamper detection method [12] by reducing the block sizes.

3 Proposed method

The proposed self-embedding watermarking method is designed for efficient image tamper detection and recovery. Only two LSBs of the pixels are used for efficient watermark embedding without reducing the visual quality of the image in the proposed method. Also, in order to use entire 128-bit hash value and triple recovery information copies of the blocks, 16×16 block size is selected for watermark embedding. The reliability of our method is improved by using the whole hash value. Therefore, recovery of the attacked areas is performed in selected 16×16 size block for more reliable tamper detection. The proposed method includes three main processes: (1) recovery bits embedding process, which embeds triple recovery information extracted from three partner blocks; (2) authentication bits embedding process, which is improved version of our previous work [12]; (3) tamper detection and recovery process, which can be achieved by extracting authentication and recovery bits from first and second LSBs of the image blocks. The list and descriptions of symbols used for the proposed method is presented in Table 1.

3.1 Recovery bits embedding process

The block diagram of the recovery bits embedding process is shown in Fig. 1. Also, the pseudocode form of the recovery bits embedding process is presented in Algorithm 1. In this process, the original image, I^o, is first divided into sixteen non-overlapping main blocks. Then, partner blocks are selected from main blocks using Look-Up table. In the proposed method, Look-Up table is constructed for recovery against 75% tampering rated attacks. In Look-Up table constructing process, host image is first divided into the four main blocks. Then, these main blocks are subdivided into four equal-sized blocks. For each equal-sized block, three partner blocks are selected from other three main blocks. In order to recover 75% tampered images, the optimum Look-Up table has been determined as shown in Fig. 1. Partner blocks groups are formed as A, B, C, and D. Each partner block group includes four partner blocks selected from four different main blocks. It can be seen from the figure that the partner blocks group A is consisted of four partner blocks: A1 from first main block, A2 from the second main block, A3 from the third main block and A4 from the fourth main block. After the partner blocks selection process, the partner blocks are divided into 4×4 non-overlapping blocks. The mean values of 4×4 blocks are calculated and converted to binary data. The binary data of the sub-blocks are combined as recovery data for each partner block. Then, triple recovery information is constructed by combining recovery data of the three other partner blocks for each partner block. This recovery information is permuted with a key. After the generation of the recovery information, each partner block is divided into 16×16 non-overlapping blocks. Then, these 16×16 blocks are divided into non-overlapping four sub-blocks. Finally, recovery bits produced separately for each partner block are embedded into the first and second LSBs of the first three sub-blocks. In the following, detailed description of proposed recovery bits embedding process is demonstrated:

1. Divide the original host image, I^o, into sixteen non-overlapping main blocks size of $S \times T$, $N_{(i)}$(i = 1, 2, …, 16).
2. Select partner blocks group, $P_{(g)}$(g = A, B, C, D), from main blocks using Look-up table which is shown in Fig. 1.
3. Divide each partner blocks, $P_{(g,\,i)}$ (i = 1, 2, 3, 4), into 4×4 blocks,

Table 1 List and descriptions of symbols

Symbols	Descriptions
I^o	Original host image
I^{rw}	Recovery bits embedded image
I^w	Watermarked image
I^a	Attacked image
I^{td}	Tamper detected image
I^r	Recovered image
$M \times N$	The size of the original image
$S \times T$	The size of the main blocks
$N_{(i)}$	The i-th main blocks of the original image ($i = 1, 2, \ldots, 16$)
$P_{(g)}$	Partner block group ($g = A, B, C, D$)
$P_{(g,i)}$	The i-th partner block ($i = 1, 2, 3, 4$)
$P^{k}_{(g,i)}$	The k-th 4×4 block of the partner block ($k = 1, 2, \ldots, S \times T/4 \times 4$)
$M^{k}_{(g,i)}$	Average value of the k-th 4×4 block ($k = 1, 2, \ldots, S \times T/4 \times 4$)
$M_{(g,i)}$	Combined average value of i-th partner block ($i = 1, 2, 3, 4$)
$R_{(g,i)}$	Recovery bits to be embedded into i-th partner block ($i = 1, 2, 3, 4$)
$PR_{(g,i)}$	Permuted recovery bits produced for i-th partner block ($i = 1, 2, 3, 4$)
$P^{l}_{(g,j)}$	The l-th 16×16 block of the partner block ($l = 1, 2, \ldots, S \times T/16 \times 16$)
$P^{l,m}_{(g,j)}$	The m-th 8×8 block of the 16×16 block ($m = 1, 2, 3, 4$)
$IR^{l,m}_{(g,j)}$	The m-th recovery bits embedded 8×8 block ($m = 1, 2, 3$)
A_i	The i-th 16×16 block of the recovery bits embedded image ($i = 1, 2, \ldots, S \times T/16 \times 16$)
$A_{(i,j)}$	The j-th 8×8 sub-block of the 16×16 the block ($j = 1, 2, 3, 4$)
$A'_{(i,j)}$	Fourth sub-block which first and second LSBs are equal zero
H_i	The i-th hash value of the 16×16 blocks ($i = 1, 2, \ldots, S \times T/16 \times 16$)
I^{w}_{i}	The i-th hash value embedded 16×16 blocks ($i = 1, 2, \ldots, S \times T/16 \times 16$)
$P^{t}_{(g,i)}$	The t-th 16×16 block of the partner block ($t = 1, 2, \ldots, S \times T/16 \times 16$)
$P^{t}_{(g,i)}m$	The m-th 8×8 sub-block of 16×16 the block ($m = 1, 2, 3, 4$)
EA	Extracted authentication bits from the fourth sub-block
$P^{k'}_{(g,i)}4$	Fourth sub-block which first and second LSBs are equal zero
C	Re-generated hash value of the 16×16 block
P_{nt}	Non-tampered partner block
P^{t}_{nt}	The t-th 16×16 block of the non-tampered partner block ($t = 1, 2, \ldots, S \times T/16 \times 16$)
$P^{t}_{nt}m$	The m-th 8×8 sub-block of the 16×16 block ($m = 1, 2, 3, 4$)
PR^t	Extracted recovery bits of the t-th 16×16 block ($t = 1, 2, \ldots, S \times T/16 \times 16$)
PR^n	Combined recovery bits of the partner block
R^n	Permutated recovery bits of the partner block
R^n_k	Divided recovery bits for k-th 16×16 block ($k = 1, 2, \ldots, S \times T/16 \times 16$)
$R^n_{(k,l)}$	Divided recovery bits for l-th 4×4 block ($l = 1, 2, \ldots, 16$)
$RB^n_{(k,l)}$	Constructed l-th 4×4 recovery block ($l = 1, 2, \ldots, 16$)
RB^n_k	Constructed k-th 16×16 recovery block ($k = 1, 2, \ldots, S \times T/16 \times 16$)

$P^{k}_{(g,i)}$ $(\mathrm{k} = 1, 2, \ldots, \mathrm{S} \times \mathrm{T}/4 \times 4)$.

4. Calculate the average value, $M^k_{(g,i)}$, of the 4×4 blocks and convert to binary using Eq. 1.

$$M^k_{(g,i)} = \mathrm{Decimaltobinary}\left(\mathrm{Mean}\left(P^k_{(g,i)}\right)\right) \tag{1}$$

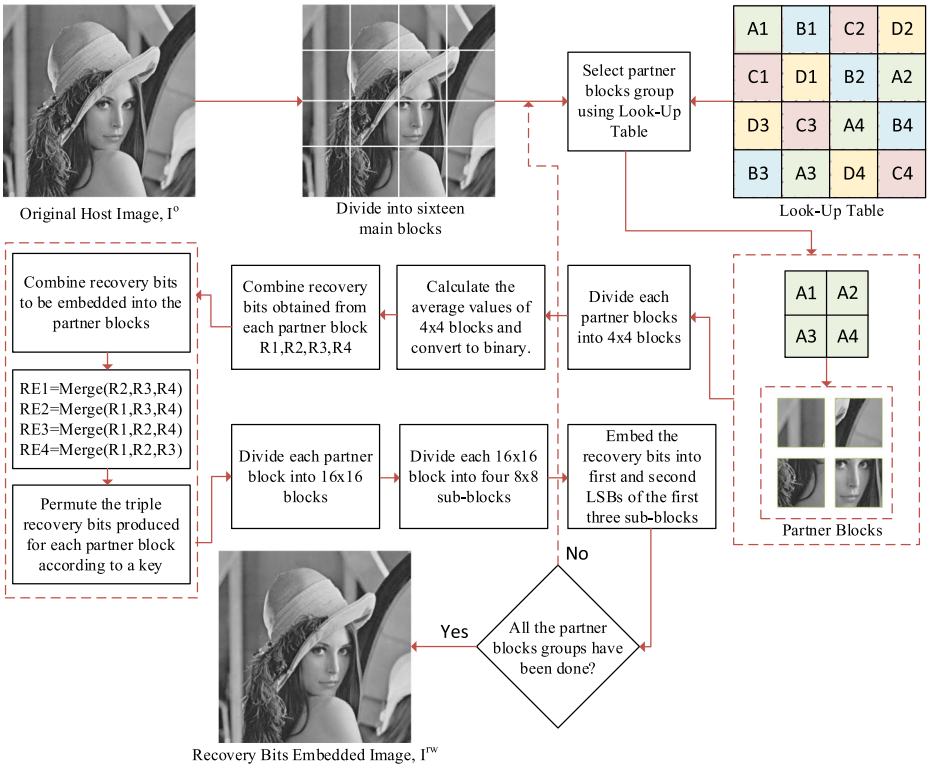

Fig. 1 Block diagram of the proposed recovery bits embedding process

5. Combine recovery bits obtained from each partner blocks using Eq. 2.

$$M_{(g,i)} = \text{Combine}\left(M^{k}_{(g,i)}\right) \tag{2}$$

6. Combine recovery bits to be embedded into each partner block using Eq. 3–6.

$$R_{(g,1)} = \text{Combine}\left(M_{(g,2)}, M_{(g,3)}, M_{(g,4)}\right) \tag{3}$$

$$R_{(g,2)} = \text{Combine}\left(M_{(g,1)}, M_{(g,3)}, M_{(g,4)}\right) \tag{4}$$

$$R_{(g,3)} = \text{Combine}\left(M_{(g,1)}, M_{(g,2)}, M_{(g,4)}\right) \tag{5}$$

$$R_{(g,4)} = \text{Combine}\left(M_{(g,1)}, M_{(g,2)}, M_{(g,3)}\right) \tag{6}$$

7. Permute the recovery bits produced for each partner block according to key using Eq. 7.

$$PR_{(g,i)} = \text{Permute}\left(R_{(g,i)}, \text{key}\right) \tag{7}$$

8. Divide each partner block, $P_{(g,i)}$, into 16×16 blocks, $P^{l}_{(g,i)} (l = 1, 2 \ldots, \text{S} \times \text{T}/16 \times 16)$.

9. Divide each 16×16 block, $P^{l}_{(g,i)}$, into four 8×8 sub-blocks, $P^{l,m}_{(g,i)}$ $(\mathrm{m} = 1, 2, 3, 4)$.
10. Embed the recovery bits into first and second LSBs of the first three sub-blocks using Eq. 8.

$$IR^{l,m}_{(g,i)} = P^{l,m}_{(g,i)} \oplus PR_{(g,i)} \tag{8}$$

11. Repeat steps 2–10 for all partner groups.
12. Obtain recovery bits embedded image, I^{rw}.

In the proposed recovery bits embedding process, as demonstrated above, generation of the recovery information is constructed between steps 3–7. Triple recovery information to be embedded into each partner block is constructed in step 6. Embedding of the recovery bits is performed between steps 8–10.

Algorithm 1: Recovery Bits Embedding Process

Input: Original host image, $(\boldsymbol{I^o})$.
Output: Recovery bits embedded image, $(\boldsymbol{I^{rw}})$.
Step 1: $\boldsymbol{N_{(i)} \leftarrow Divide(I^o, S \times T)}$ // divide the host image into sixteen non-overlapping main blocks.
Step 2: $\boldsymbol{P_{(g)} \leftarrow SelectPartnerBlocksGroup\left(N_{(i)}, LookUp\ table\right), (g = A, B, C, D)}$ // select the partner blocks group from main blocks using Look-up table.
2.1. $\boldsymbol{P^k_{(g,i)} \leftarrow Divide\left(P_{(g,i)}, 4x4\right)(k = 1, 2, \ldots, S \times T/4 \times 4)}$ // divide the partner blocks into 4×4 blocks.
2.1.1. $\boldsymbol{M^k_{(g,i)} \leftarrow Decimaltobinary(Mean(P^k_{(g,i)}))}$ // calculate and convert the average values of the 4×4 blocks.
2.1.2. $\boldsymbol{M_{(g,i)} \leftarrow Combine(M^k_{(g,i)})}$ // combine recovery bits obtained from the 4×4 blocks.
2.2. Repeat step 2.1 for all partner blocks in partner blocks group.
2.3. $\boldsymbol{R_{(g,1)} \leftarrow Combine\left(M_{(g,2)}, M_{(g,3)}, M_{(g,4)}\right)}$, $\boldsymbol{R_{(g,2)} \leftarrow Combine\left(M_{(g,1)}, M_{(g,3)}, M_{(g,4)}\right)}$, $\boldsymbol{R_{(g,3)} \leftarrow Combine\left(M_{(g,1)}, M_{(g,2)}, M_{(g,4)}\right)}$, $\boldsymbol{R_{(g,4)} \leftarrow Combine\left(M_{(g,1)}, M_{(g,2)}, M_{(g,3)}\right)}$ // combine recovery bits to be embedded into the partner blocks.
2.4. $\boldsymbol{PR_{(g,i)} \leftarrow Permute\left(R_{(g,i)}, key\right)}$ // permute the triple recovery bits produced for each partner block with a secret key.
2.5. $\boldsymbol{P^l_{(g,i)} \leftarrow Divide\left(P_{(g,i)}, 16 \times 16\right)(l = 1, 2, \ldots, S \times T/16 \times 16)}$ // divide the partner blocks into $16x16$ blocks.
2.5.1. $\boldsymbol{P^{l,m}_{(g,i)} \leftarrow Divide\left(P^l_{(g,i)}, 8 \times 8\right)(m = 1, 2, 3, 4)}$ // divide the 16×16 blocks into four 8×8 sub-blocks.
2.5.2. $\boldsymbol{IR^{k,l}_{(i,j)} \leftarrow P^{k,l}_{(i,j)}\mathrm{m} \oplus PR_{(i,j)}m}$ // embed the recovery bits into first and second LSBs of the first three sub-blocks.
2.6. Repeat step 2.5 for all partner blocks.
Step 3: Repeat step 2 for all partner blocks groups.
Step 4: Obtain recovery bits embedded image, $(\boldsymbol{I^{rw}})$.

3.2 Authentication bits embedding process

The block diagram of the authentication bits embedding process is shown in Fig. 2. Also, the pseudocode form of the authentication bits embedding process is presented in Algorithm 2. Proposed authentication bits generation and embedding method is an improved version of our previous image integrity method [12]. The block size used in the earlier study has been

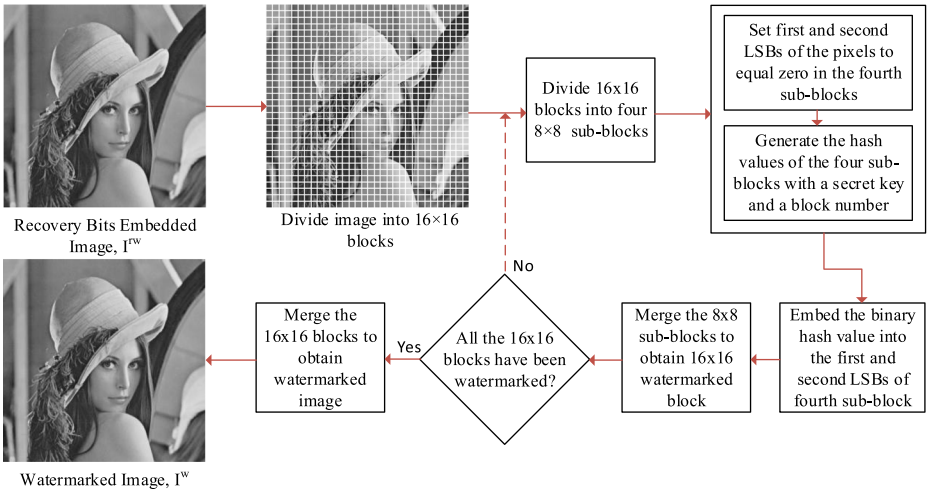

Fig. 2 Block diagram of the proposed authentication bits embedding process

reduced to 16 × 16 for increasing the precision of the tamper detection. In addition, instead of SHA-256, MD5 hash function which is appropriate to the proposed method has been used. In authentication bits generating process, recovery bits embedded image is first divided into 16 × 16 blocks. Then, each 16 × 16 blocks are divided into four 8 × 8 non-overlapping sub-blocks. The first and second LSBs of the pixels are set equal to zero in the fourth sub-blocks. The hash values of the four sub-blocks are generated with a secret key and a block number. In authentication bits embedding process, binary hash values are embedded into the first and second LSBs of fourth sub-block using LSB modification. The detailed description of proposed authentication bits embedding process is demonstrated as follows:

1. Divide the recovery bits embedded image, I^{rw}, into non-overlapping blocks size of 16 × 16, $A_i (i = 1, 2, \ldots, M \times N/16 \times 16)$.
2. Divide each 16 × 16 blocks, A_i, into 8 × 8 four sub-blocks, $A_{(i,j)}$ (j = 1, 2, 3, 4).
3. Set first and second LSBs of the pixels to equal zero in the fourth sub-blocks using Eq. 9.

$$A'_{(i,4)} = Setzero\left(LSB\left(A_{(i,4)}\right)\right) \tag{9}$$

4. Produce 128-bit hash value of the four sub-blocks with a secret key and a block number using Eq. 10.

$$H_i = MD5\left(A_{(i,1)}, A_{(i,2)}, A_{(i,3)}, A'_{(i,4)}, key, block\ number\right) \tag{10}$$

5. Embed the binary hash value into the first and second LSBs of fourth sub-block with LSB modification using Eq. 11.

$$I_i^w = A_{(i,4)} \oplus H_i \tag{11}$$

6. Repeat 2–5 steps for all 16 × 16 blocks.
7. Obtain watermarked image, I^w.

In the demonstrated proposed authentication bits embedding process, in step 3, first and second LSBs of the fourth sub-blocks are reserved for embedding of authentication bits. Therefore, first and second LSBs of the pixels in the fourth sub-blocks are set to equal zero. Authentication bits for each 16×16 block are produced in step 4. Block number and key are used for detection of the complex attacks such as copy-move. In step 5, authentication bits are embedded with LSB modification.

Algorithm 2: Authentication Bits Embedding Process

Input: Recovery bits embedded image ($\boldsymbol{I^{rw}}$).
Output: Watermarked image ($\boldsymbol{I^w}$).
Step 1: $\boldsymbol{A_{(i)} \leftarrow Divide(I^{rw}, 16 \times 16)}$ // divide the recovery bits embedded image into 16×16 non-overlapping blocks.
Step 2: $\boldsymbol{A_{(i,j)} \leftarrow Divide(A_{(i)}, 8 \times 8)(j = 1, 2, 3, 4)}$ // divide the 16×16 block into four 8×8 non-overlapping sub-blocks.
2.1. $\boldsymbol{A'_{(i,4)} \leftarrow Setzero(LSB(A_{(i,4)}))}$ // set first and second LSBs of the pixels to equal zero in fourth sub-blocks.
2.2. $\boldsymbol{H_i \leftarrow MD5(A_{(i,1)}, A_{(i,2)}, A_{(i,3)}, A'_{(i,4)}, key, blocknumber)}$ // calculate the hash value of the $16x16$ block with a secret key and a block number.
2.3. $\boldsymbol{I_i^w \leftarrow A_{(i,4)} \oplus H_i}$ // embed the hash value into the first and second LSBs of fourth sub-block.
Step 2: Repeat step 2 for all 16×16 blocks.
Step 4: Obtain watermarked image, ($\boldsymbol{I^w}$).

3.3 Tamper detection and recovery process

The block diagram of the tamper detection and recovery process is shown in Fig. 3. Also, the pseudocode form of the tamper detection and recovery process is presented in Algorithm 3. In this process, attacked Image, I^a,is first divided into sixteen non-overlapping main blocks. Partner blocks are selected from main blocks using Look-Up table. Then, partner blocks are divided into non-overlapping 16×16 blocks. These 16×16 blocks are further divided into four sub-blocks. Authentication bits are extracted from fourth sub-block and then are compared with regenerated hash values of the four sub-blocks. According to the comparison result, 16×16 blocks are classified as tampered or non-tampered block. After tamper detection process, a non-tampered partner block is selected in order to recover tampered block. Recovery bits are extracted from selected non-tampered partner block and then are permuted with the key. Then, recovery blocks for tampered blocks are generated using the permuted recovery bits. Finally, recovery blocks are replaced with the corresponding blocks classified as tampered. In the following, detailed description of proposed tamper detection and recovery process is demonstrated:

1. Divide the attacked image, I^a, into sixteen non-overlapping main blocks size of $S \times T$, B_i(i = 1, 2, …, 16).
2. Select partner blocks group, P_g(g = 1, 2, 3, 4), using Look-up table which is shown in Fig. 3.

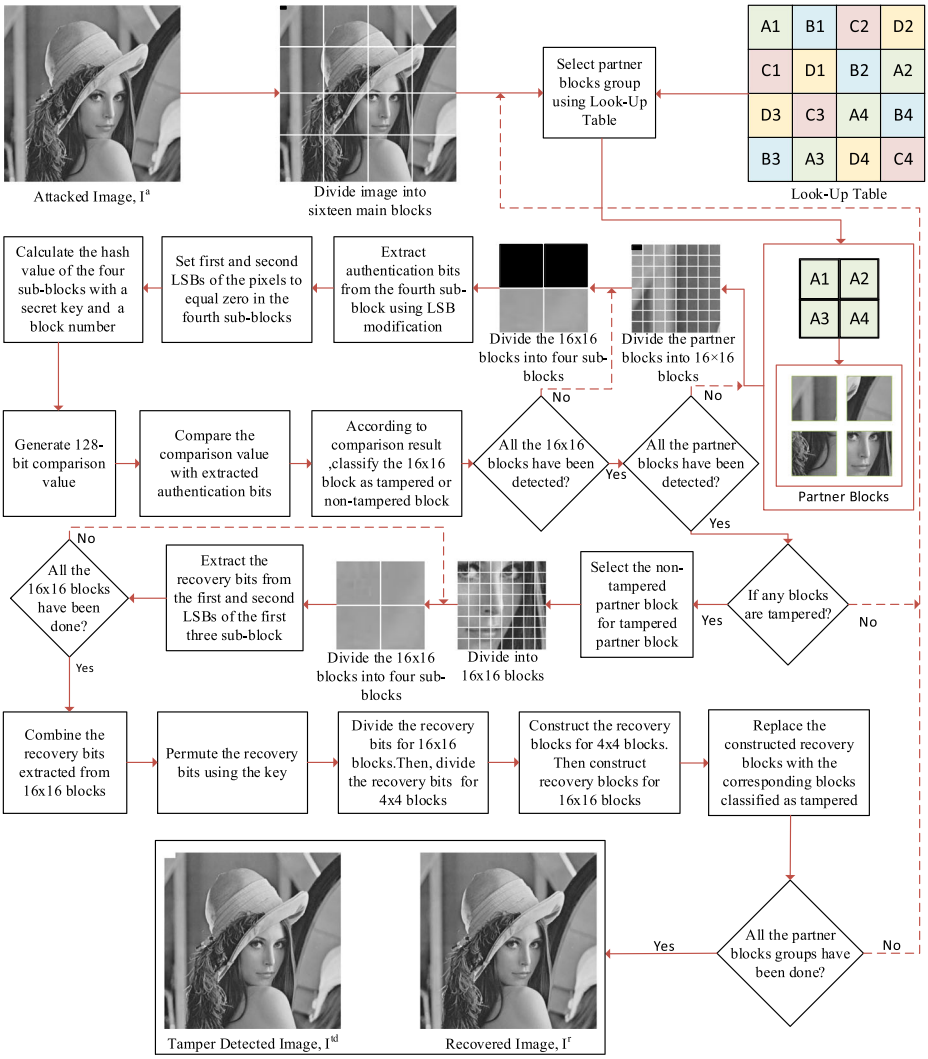

Fig. 3 Block diagram of the proposed tamper detection and recovery process

3. Divide each partner blocks, $P_{(g,i)}$ (i = 1, 2, 3, 4), into 16 × 16 blocks, $P^{t}_{(g,i)}$ $(t = 1, 2, \ldots, S \times T/16 \times 16)$.
4. Divide the 16 × 16 blocks into four 8 × 8 sub-blocks, $P^{t}_{(g,i)}\, m$ $(m = 1, 2, 3, 4)$.
5. Extract authentication bits from the fourth sub-block with LSB modification using Eq. 12.

$$EA = Extract\left(LSB\left(P^{t}_{(g,i)}\,4\right)\right) \tag{12}$$

6. Set first and second LSBs of pixels to equal zero in the fourth sub-blocks using Eq. 13.

$$P^{t'}{}_{(g,i)}\,4 = Setzero\left(LSB\left(P^{t}{}_{(g,i)}\,4\right)\right) \tag{13}$$

7. Produce the hash values of the four sub-blocks with a secret key and a block number using Eq. 14.

$$C = MD5\left(P^{t}{}_{(g,i)}\,1, P^{t}{}_{(g,i)}\,2, P^{t}{}_{(g,i)}\,3, P^{t'}{}_{(g,i)}\,4, key, block\ number\right) \tag{14}$$

8. Compare the re-generated hash value, C, with extracted authentication bits, EA.
9. Classify the 16×16 block as tampered or non-tampered block with respect to comparison result.
10. Repeat 4–9 steps for all 16×16 blocks.
11. Repeat 3–10 steps for all partner blocks.
12. Return to step 2 for select new partner blocks group if any block is not classified as a tampered.
13. Select the non-tampered partner block, P_{nt}, for tampered partner block, P_t.
14. Divide selected non-tampered partner block into 16×16 blocks, P^{t}_{nt} $(t = 1, 2, \ldots, S \times T/16 \times 16)$.
15. Divide the 16×16 blocks into four 8×8 sub-blocks, $P_{nt}{}^{t}m$ $(m = 1, 2, 3, 4)$.
16. Extract the recovery bits from the first and second LSBs of the first three sub-block using Eq. 15.

$$PR^{t} = Extract\left(LSB\left(P^{t}_{nt}1, P^{t}{}_{nt}\,2, P^{t}{}_{nt}\,3\right)\right) \tag{15}$$

17. Repeat 15–16 steps for all 16×16 blocks.
18. Combinate the recovery bits, PR^n, extracted from 16×16 blocks.
19. Permute the recovery bits with a key using Eq. 16.

$$R^{n} = \text{Permute}(PR^{n}, \text{key}) \tag{16}$$

20. Divide the extracted recovery bits, R^n, for 16×16 blocks, R^{n}_{k} $(k = 1, 2\ldots, S \times T/16 \times 16)$.
21. Divide the recovery bits, R^{n}_{k}, for 4×4 blocks, $R^{n}_{(k,l)}$ $(l = 1, 2\ldots, 16)$.
22. Construct the recovery blocks for 4×4 blocks using Eq. 17.

$$RB^{n}_{(k,l)} = Resize\left(R^{n}_{(k,l)}\right) \tag{17}$$

23. Construct the recovery blocks for 16×16 blocks using Eq. 18.

$$RB^{n}_{k} = \text{Construct}\left(RB^{n}_{(k,l)}\right) \tag{18}$$

24. Replace the constructed recovery blocks size of 16×16 with the corresponding blocks classified as tampered.

25. Repeat 2–24 steps for all partner blocks groups.
26. Obtain tamper detected image, I^{td}, and recovered image, I^{r}.

In the proposed tamper detection and recovery process, as demonstrated above, detection of the tampered 16 × 16 blocks is performed between steps 3–10. Recovery bits of the attacked areas are generated between steps 14–23. Recovery blocks for attacked 16 × 16 blocks are constructed in step 23. In step 24, constructed recovery blocks are replaced with the corresponding blocks classified as tampered.

Algorithm 3: Tamper Detection and Recovery Process

Input: Attacked image ($\boldsymbol{I^a}$).
Output: Tamper detected image ($\boldsymbol{I^{td}}$), recovered image ($\boldsymbol{I^r}$).
Step 1: $\boldsymbol{B_{(i)} \leftarrow Divide(I^a, S \times T)}$ // divide the attacked image into sixteen non-overlapping main blocks.
Step 2: $\boldsymbol{P_{(g)} \leftarrow \mathbf{SelectPartnerBlocksGroup}\left(B_{(i)}, LookUp\ table\right), (g = A, B, C, D)}$ // Select the partner blocks group using Look-up table.
2.1. $\boldsymbol{P^t_{(g,i)} \leftarrow Divide\left(P_{(g,i)}, 16x16\right)(k = 1, 2, \ldots, S \times T/16 \times 16)}$ // divide the partner blocks into 16 × 16 blocks
2.1.1. $\boldsymbol{P^t_{(g,i)}m \leftarrow Divide\left(P^t_{(g,i)}, 8x8\right)(m = 1, 2, 3, 4)}$ // divide 16 × 16 block, into four 8 × 8 sub-blocks.
2.1.1.1. $\boldsymbol{EA \leftarrow Extract\left(LSB\left(P^t_{(g,i)}4\right)\right)}$ // extract the authentication bits from fourth sub-block.
2.1.1.2. $\boldsymbol{P^{t\prime}_{(g,i)}4 \leftarrow Setzero\left(LSB\left(P^t_{(g,i)}4\right)\right)}$ // set first and second LSBs of the pixels to equal zero in fourth sub-blocks.
2.1.1.3. $\boldsymbol{C \leftarrow MD5\ (P^t_{(g,i)}1, P^t_{(g,i)}2, P^t_{(g,i)}3, P^{t\prime}_{(g,i)}4, key, block\ number)}$ // produce the hash value of the 16 × 16 block with a secret key and a block number.
2.1.1.4. Compare the re-generated hash value with extracted authentication bits.
2.1.1.5. Classify the 16 × 16 block as tampered or non-tampered.
2.1.2. Repeat step 2.1.1 for all 16 × 16 blocks
2.2. Repeat step 2.1 for all partner blocks in partner blocks group.
2.3. Return step 2 to select new partner blocks group if any block is not classified as a tampered.
2.4. $\boldsymbol{P_{nt} \leftarrow \mathbf{SelectNonTamperedPartnerBlock}(P_t)}$ // select the non-tampered partner block for tampered partner block.
2.5. $\boldsymbol{P^t_{nt} \leftarrow Divide(P_{nt}, 16 \times 16)(t = 1, 2, \ldots, S \times T/16 \times 16)}$ // divide selected non-tampered partner block into 16 × 16 blocks.
2.5.1. $\boldsymbol{P^t_{nt}m \leftarrow Divide(P^t_{nt}, 16 \times 16)\ (m = 1, 2, 3, 4)}$ // divide the 16 × 16 blocks into four sub-blocks.
2.5.2. $\boldsymbol{PR^t \leftarrow Extract(LSB(P^t_n1, P^t_n2, P^t_n3))}$ // extract the recovery bits from first and second LSBs of the first three sub-blocks.
2.6. Repeat step 2.5 for all 16 × 16 blocks.
2.7. $\boldsymbol{PR^n \leftarrow \mathbf{Combine}(PR^t)}$ // combine recovery bits extracted from 16 × 16 blocks.
2.8. $\boldsymbol{R^n \leftarrow \mathbf{Permute}(PR^n, \mathbf{key})}$ // Permute the recovery bits with a secret key.
2.9. $\boldsymbol{R^n_k \leftarrow Divide(R^n_k, 4 \times 4)\ (k = 1, 2, \ldots, S \times T/16 \times 16)}$ // divide the recovery bits for $16x16$ blocks.
2.9.1. $\boldsymbol{R^n_{(k,l)} \leftarrow Divide(R^n, 16 \times 16)\ (l = 1, 2, \ldots,\ 16)}$ // divide the recovery bits for $4x4$ blocks.
2.9.2. $\boldsymbol{RB^n_{(k,l)} \leftarrow Resize\left(R^n_{(k,l)}\right)}$ // construct the recovery blocks for 4 × 4.
2.10. Repeat step 2.9 for all 16 × 16 blocks.
2.11. $\boldsymbol{RB^n_k \leftarrow \mathbf{Construct}\ (RB^n_{(k,l)})}$ construct the recovery blocks for 16 × 16.
2.12. Replace the constructed 16 × 16 recovery blocks with the corresponding tampered blocks
Step 3: Repeat step 2 for all partner blocks groups.
Step 4: Obtain tamper detected image, ($\boldsymbol{I^{td}}$), recovered image, ($\boldsymbol{I^r}$).

4 Experimental results

In this section, 512 × 512 Lena, Sailboat, Lake, Airplane, Baboon and Goodhill images shown in Fig. 4 are used as host images to evaluate the performance of the proposed method. In the proposed method, only two LSBs of the pixels are used for watermark embedding. Therefore, each 512 × 512 host image is watermarked with 393 216-bit recovery watermark to obtain recovery bits embedded image. Then, each recovery bits embedded image is watermarked with 131 072-bit authentication watermark to obtain watermarked image. The watermarked Lena, Sailboat, Lake, Airplane, Baboon and Goodhill images are illustrated in Fig. 5. It is clear from the figure that the visual differences between the host and the watermarked images cannot be distinguished by human eyes.

In order to determine the quality of the watermarked and the recovered images, Peak signal-to-noise ratio (PSNR) and Structural similarity index (SSIM) metrics have been used. PSNR values between the host and the measured images are calculated using the Eqs. 19 and 20 [15].

$$PSNR = 10 \times \log \frac{255^2}{MSE} \tag{19}$$

$$MSE = \frac{1}{M \times N} \sum_{i=1}^{M} \sum_{j=1}^{N} \left(I^h_{(i,j)} - I^m_{(i,j)} \right)^2 \tag{20}$$

where, M and N are the height and width of the images, $I^h_{(i,j)}$ and $I^m_{(i,j)}$ denote the pixel values at position (i, j) of the host and measured images [15].

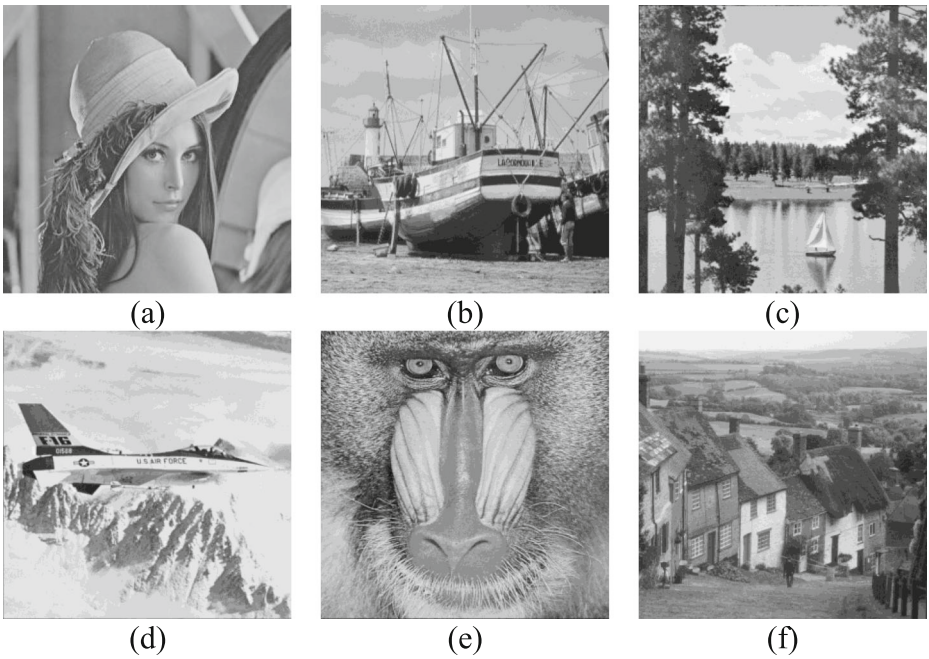

Fig. 4 Host images: (**a**) Lena, (**b**) Sailboat, (**c**) Lake, (**d**) Airplane, (**e**) Baboon, (**f**) Goodhill

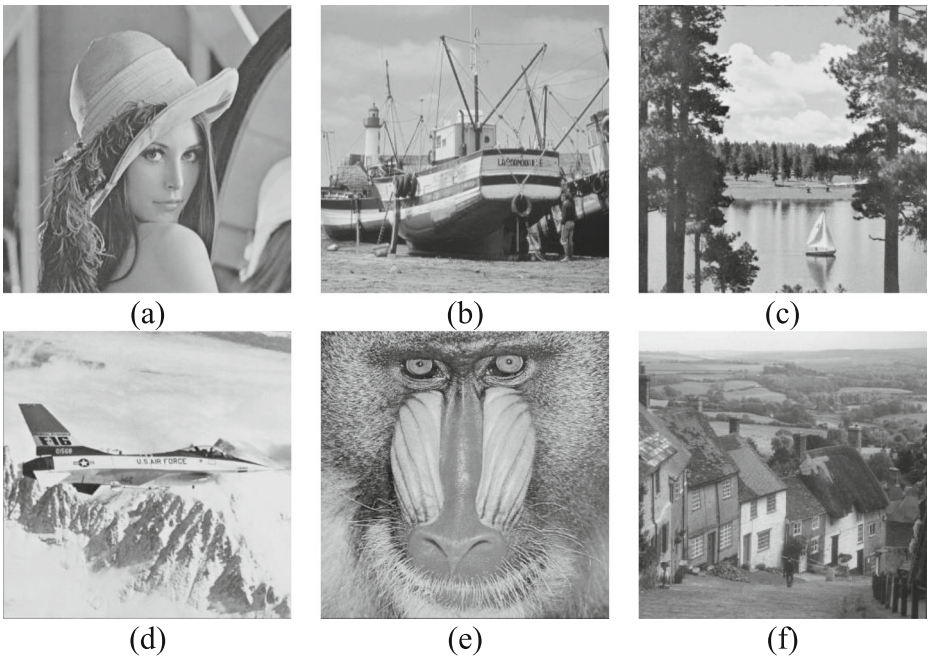

Fig. 5 Watermarked images: (**a**) Lena, (**b**) Sailboat, (**c**) Lake, (**d**) Airplane, (**e**) Baboon

SSIM values between the host and the measured images are calculated using the Eq. 21 [53].

$$SSIM = \left(\frac{\left(2\mu_x\mu_y + C_1\right)\left(2\sigma_{xy} + C_2\right)}{\left(\mu_x^2 + \mu_y^2 + C_1\right)\left(\sigma_x^2 + \sigma_y^2 + C_2\right)} \right) \tag{21}$$

where, μ_x, μ_y are the local means of the host and measured image, σ_{xy} is the co-variation of the host and measured images, and σ_x, σ_y are the standard deviations of the host and measured image. C_1 and C_2 are constants [53].

The PSNR and SSIM results of the watermarked images are shown in Table 2. It can be seen from the table that the PSNR and SSIM values of the watermarked images are higher than 44.127 *dB* and 0.9785, respectively. The results demonstrate that the visual quality of the watermarked images has been satisfactorily protected.

The performance of the proposed method against 75% tampering rated attacks has been evaluated using six types of cropping attacks. Left (LZ), bottom (BZ), right (RZ), upper (UZ), center (CZ) and outer (OZ) zone cropping attacks have been applied to Lena image, as shown in Fig. 6. Tamper detected and recovered images for corresponding attacked images are also shown in Fig. 6. It is clear from the figure that all six types of 75% cropping attacks have been successfully detected and all six types of attacked zones

Table 2 The PSNR and SSIM results of watermarked images

Image	Lena	Sailboat	Lake	Airplane	Baboon	Goodhill
PSNR (dB)	44.1416	44.1270	44.1448	44.1422	44.1514	44.1538
SSIM	0.9785	0.9816	0.9858	0.9788	0.9938	0.9866

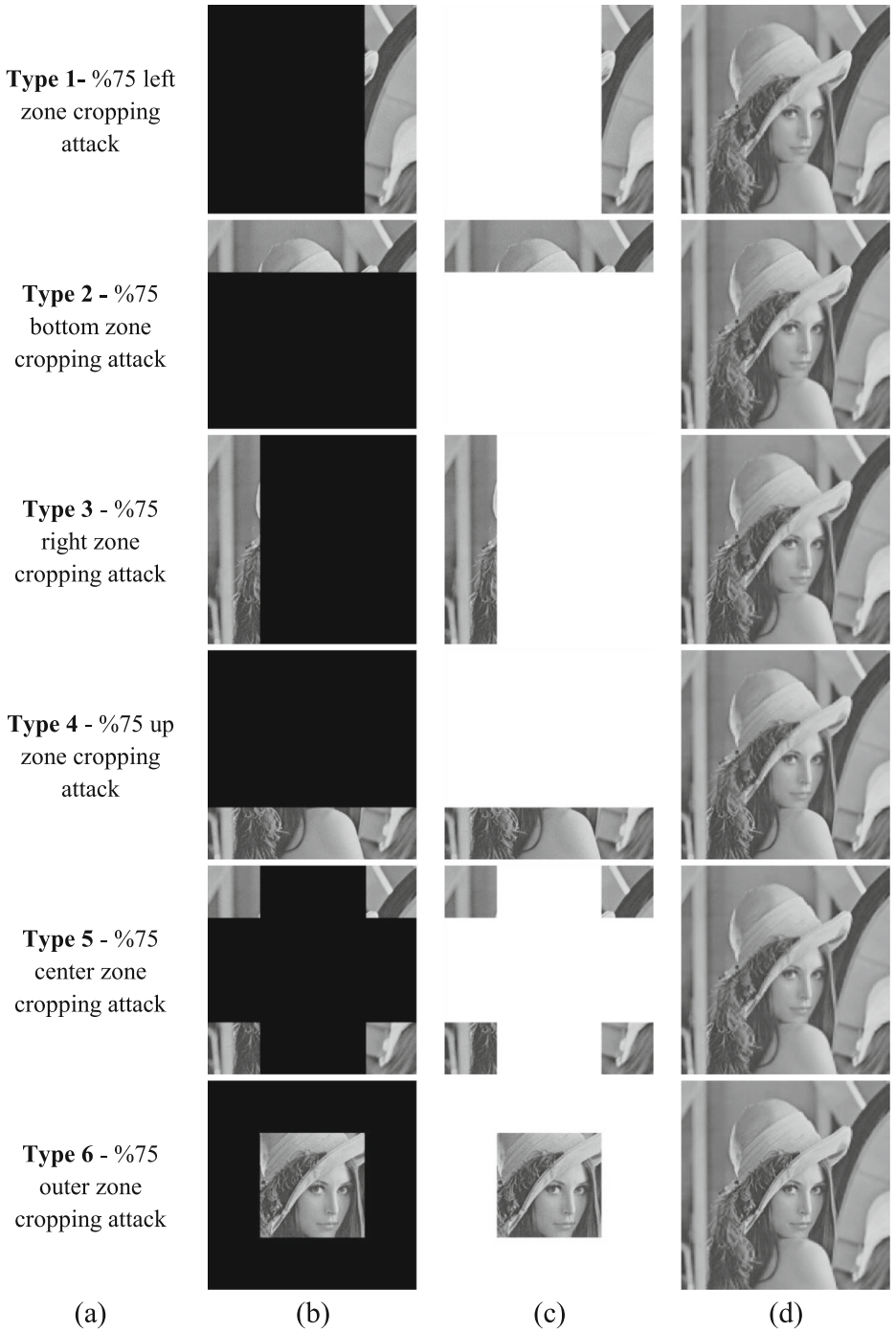

Fig. 6 75% cropping attacks and results: (**a**) attack types and descriptions, (**b**) attacked images, (**c**) tamper detected images, (**d**) recovered images

have been recovered using proposed method. Also, in order to evaluate the performance of the proposed method on different images, 75% cropping attacks have been performed to Sailboat, Lake, Airplane, Baboon and Goodhill images. The PSNR and SSIM results of the recovered images are shown in Table 3. It can be seen from the table that the PSNR results of Lena, Sailboat, Airplane and Goodhill images are higher than 25.1 *dB*. Also, the PSNR results of Lake images are higher than 24.89 *dB*. However, the PSNR results of the Baboon images are between 21.23 and 23.21 *dB*. The best PSNR results of the recovered Lena, Sailboat, Lake, Airplane, Baboon and Goodhill images against %75 cropping attack are 30.4995, 27.1618, 25.4999, 26.9780, 23.2143 and 28.2961, respectively. The highest PSNR and SSIM results of the recovered Lena, Sailboat and Airplane images are obtained against OZ cropping attack. Furthermore, the best PSNR result of the recovered Lake images is obtained against LZ cropping attack while the best SSIM result is obtained against RZ cropping attack. On the other hand, the best PSNR result of the recovered Goodhill images is obtained against RZ cropping attack while the highest SSIM result is obtained OZ cropping attack. Also, the best PSNR and SSIM results of the recovered Baboon images are obtained against BZ cropping attack. The highest PSNR and SSIM results of all recovered images are obtained from Lena images as 30.4995 *dB* and 0.8538, respectively.

On the other hand, six types of 50% and six types of 25% cropping attacks have also been applied to the watermarked images. Upper, Left, vertical center (VCZ), horizontal center (HCZ), left-right (LRZ), upper-left and down-right (ULDRZ) zone 50% cropping attacks have been applied to Lena image, as shown in Fig. 7. Also, center, upper-left (ULZ), center-left (CLZ), upper-center (UCZ), different (DZ1 and DZ2) zone 25% cropping attacks have been performed to Lena image, as shown in Fig. 8. Tamper detected and recovered images of 50% and 25% attacked Lena images are shown in Figs. 7 and 8, respectively. It clear from the figures that all applied cropping attacks have been successfully detected and recovered. Further, six types of 25% and 50% tampering rated attacks have been applied to Sailboat, Lake, Airplane, Baboon and Goodhill images. The PSNR and SSIM results for 50% and 25% tampering rated cropping attacks are given in Tables 4 and 5, respectively. As can be seen from the Table 4 that the highest PSNR and SSIM results of the recovered images for 50% tampering rated attacks

Table 3 The PSNR and SSIM results for 75%tampering rated attacks

Image	Performance of recovered image	Tamper Type					
		LZ	BZ	RZ	UZ	CZ	OZ
Lena	PSNR	28.8530	28.6864	29.1784	29.6740	29.0286	30.4995
	SSIM	0.8158	0.8183	0.8289	0.8356	0.8246	0.8538
Sailboat	PSNR	26.3517	25.3123	26.0742	26.0763	25.7100	27.1618
	SSIM	0.7804	0.7410	0.7743	0.7923	0.7671	0.7951
Lake	PSNR	25.4999	25.4118	25.3491	24.8967	25.3814	24.9500
	SSIM	0.7625	0.7500	0.7638	0.7437	0.7603	0.7388
Airplane	PSNR	25.5423	25.1031	26.6659	26.1345	25.6047	26.9780
	SSIM	0.8083	0.8035	0.8252	0.8283	0.8123	0.8344
Baboon	PSNR	22.2015	23.2143	22.5466	21.9019	22.5520	21.2351
	SSIM	0.5997	0.6105	0.5973	0.5759	0.5976	0.5397
Goodhill	PSNR	27.3533	26.9948	28.2961	27.7906	27.3983	28.1936
	SSIM	0.7267	0.7099	0.7489	0.7387	0.7270	0.7496

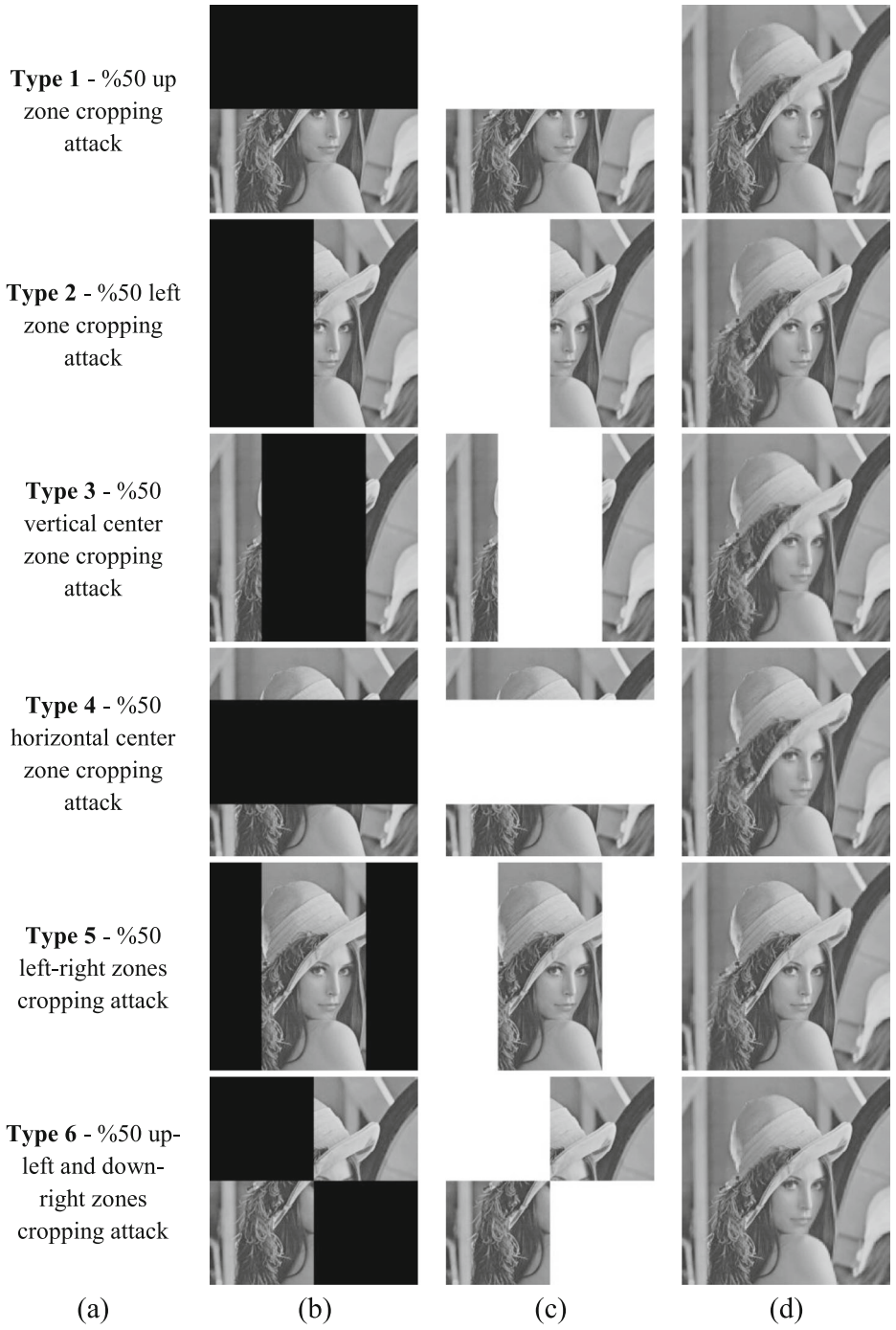

Fig. 7 50% cropping attacks and results: (**a**) attack types and descriptions, (**b**) attacked images, (**c**) tamper detected images, (**d**) recovered images

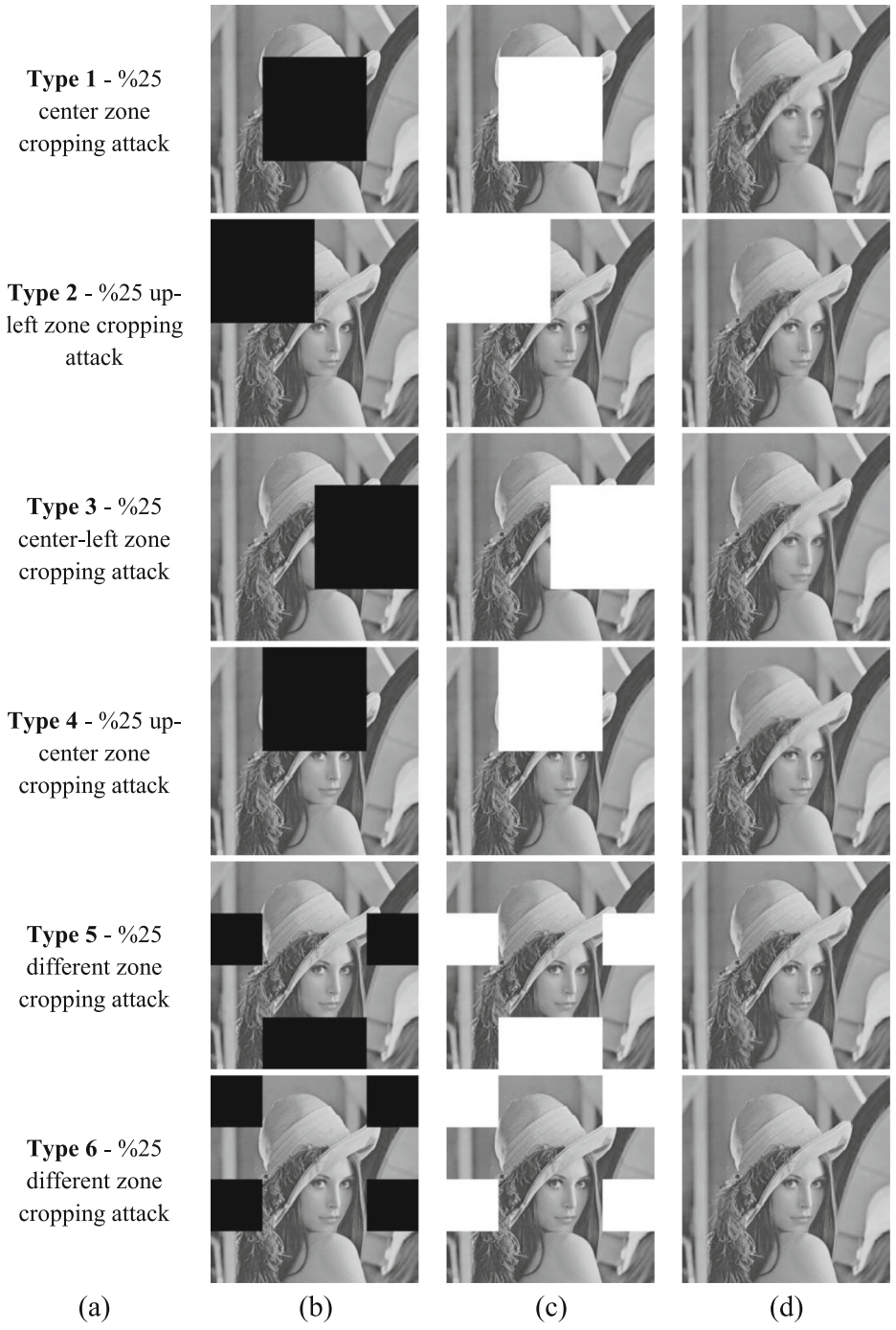

Fig. 8 50% cropping attacks and results: (**a**) attack types and descriptions, (**b**) attacked images, (**c**) tamper detected images, (**d**) recovered images

Table 4 The PSNR and SSIM results for 50%tampering rated attacks

Image	Performance of recovered image	Tamper Type					
		UZ	LZ	VCZ	HCZ	LRZ	ULDRZ
Lena	PSNR	32.4310	30.3825	30.0741	30.4398	32.6410	32.8133
	SSIM	0.8982	0.8680	0.8689	0.8780	0.9072	0.8998
Sailboat	PSNR	29.4615	28.4079	27.8483	26.4466	28.2294	28.5393
	SSIM	0.8866	0.8551	0.8494	0.8281	0.8562	0.8649
Lake	PSNR	26.1865	26.7501	27.7333	26.8541	26.2600	26.7494
	SSIM	0.8391	0.8300	0.8602	0.8275	0.8061	0.8319
Airplane	PSNR	29.0383	27.3131	27.6834	26.4164	28.1349	27.6512
	SSIM	0.9045	0.8759	0.8711	0.8693	0.8914	0.8761
Baboon	PSNR	22.8856	23.6516	24.7794	25.2804	23.0691	23.6847
	SSIM	0.6996	0.7221	0.7555	0.7449	0.6877	0.7179
Goodhill	PSNR	30.9535	28.7486	29.6079	28.4300	29.4659	29.7784
	SSIM	0.8406	0.8097	0.8268	0.7998	0.8222	0.8276

are obtained from Lena image as 32.8133 *dB* and 0.9072. The PSNR results of Lena images are between 30.07 and 32.81 *dB* for 50% tampering rated attacks. Also, it can be seen from the Table 4 that the lowest PSNR and SSIM results of the recovered images for 50% tampering rated attacks are obtained from Baboon image as 22.8856 *dB* and 0.6877, respectively. The PSNR results of recovered Baboon images against 50% tampering rated attacks are between 22.8856 and 25.2804 *dB*. The highest PSNR and SSIM results of the recovered Sailboat, Airplane and Goodhill images against 50% tampering rate are obtained against UZ cropping attack. The best PSNR result of the recovered Lake images is obtained against ULDRZ cropping attack while the best PSNR result of the recovered Lake images is obtained against VCZ cropping attack. Moreover, the best PSNR result of the recovered Baboon images is obtained against BZ cropping attack. For 25% tampering rated attacks, as shown from Table 5, the best PSNR and SSIM results of all images are obtained from Lena image as 36.2713 *dB* and 0.9565, respectively. The PSNR results of Lena images are between 31.96 and 36.27 *dB* while the SSIM results are between 0.9223 and 0.9565. The worst PSNR result of all images is

Table 5 The PSNR and SSIM results for 25%tampering rated attacks

Image	Performance of recovered image	Tamper Type					
		CZ	ULZ	CLZ	UCZ	DZ1	DZ2
Lena	PSNR	31.9647	36.1727	34.5460	34.4603	34.1343	36.2713
	SSIM	0.9223	0.9451	0.9498	0.9377	0.9496	0.9565
Sailboat	PSNR	29.1282	33.8193	28.9799	31.7278	31.6912	31.4472
	SSIM	0.9104	0.9506	0.9121	0.9363	0.9252	0.9361
Lake	PSNR	30.6943	28.8903	29.7395	30.6020	29.2310	29.4268
	SSIM	0.9275	0.9174	0.9143	0.9465	0.9093	0.8965
Airplane	PSNR	29.0807	31.0219	30.9116	32.5729	29.9533	32.4056
	SSIM	0.9281	0.9470	0.9472	0.9550	0.9336	0.9479
Baboon	PSNR	31.3361	25.6223	28.7825	28.0954	25.6899	25.3222
	SSIM	0.9029	0.8488	0.8710	0.8812	0.8407	0.8370
Goodhill	PSNR	31.4902	33.1702	32.7948	34.3130	32.2938	32.5787
	SSIM	0.8996	0.9145	0.9114	0.9223	0.9068	0.9184

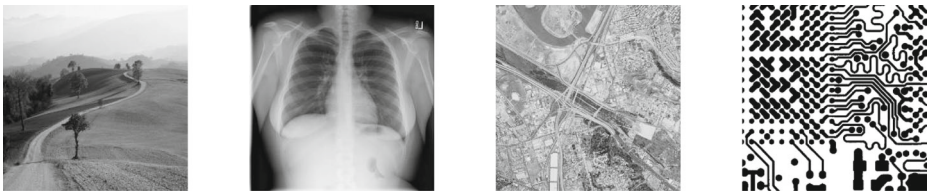

Fig. 9 Original host images: (**a**) Tree, (**b**) ChestX-ray, (**c**) San Diego, (**d**) Printed Circuit Board

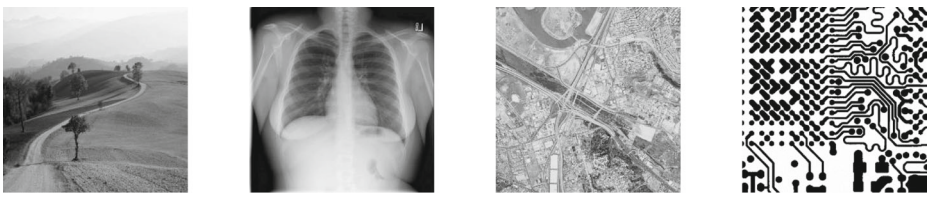

Fig. 10 Watermarked images: (**a**) Tree, (**b**) ChestX-ray, (**c**) San Diego, (**d**) Printed Circuit Board

obtained from Baboon image as 25.3222 *dB*. The PSNR results of Baboon images are between 25.3222 and 31.3361 *dB* while the SSIM results are between 0.9029 and 0.8370.

In order to demonstrate the performance of the proposed method for different application areas such as healthcare, remote sensing, and industrial; 512×512 Tree [8], ChestX-ray [52], San Diego [54] and Printed Circuit Board [50] images shown in Fig. 9 are used. The corresponding images that are watermarked by the proposed method are illustrated in Fig. 10. The PSNR results are obtained from Tree, ChestX-ray, San Diego and Printed Circuit Board images as 44.1846, 44.1846, 44.1511, 43.2620 *dB*, respectively.

Different kinds and shapes of attacks are performed to the watermarked Tree, ChestX-ray, San Diego and Printed Circuit Board images, as shown in Fig. 11. Copy-move attacks are applied to Tree and ChestX-ray watermarked images, as illustrated in Fig. 11a, b. Splicing attack is applied to San Diego image, as shown in Fig. 11c. Manipulation attack is performed to Printed Circuit Board image, as illustrated in Fig. 11d. Figures 12 and 13 shows the tamper detected and recovered images, respectively. It can be seen from the figures that detection and recovery of the tampered regions have been successfully performed.

Fig. 11 Attacked images: (**a**) Tree, (**b**) ChestX-ray, (**c**) San Diego, (**d**) Printed Circuit Board

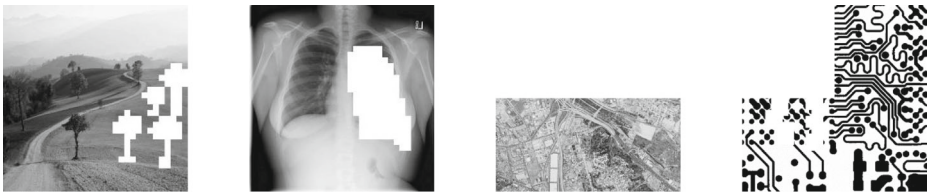

Fig. 12 Tamper detected images: (**a**) Tree, (**b**) ChestX-ray, (**c**) San Diego, d) Printed Circuit Board

Fig. 13 Recovered images: (**a**) Tree, (**b**) ChestX-ray, (**c**) San Diego, (**d**) Printed Circuit Board

In order to evaluate the time complexity of the proposed method, Lena, Sailboat, Lake, Airplane, Baboon, and Goodhill images are used. Watermark embedding time and tamper detection time against 25% VCZ cropping attack, 50% CZ cropping attack and %75 CZ cropping attack are measured. The measurements have been carried out on a computer with a CPU Intel Core i5–7500 3.40 GHz and 4 GB RAM. The recovery bits embedding, authentication bits embedding and total watermark embedding time are given in Table 6. Also, tamper detection and recovery time against 25% VCZ cropping attack, 50% CZ cropping attack and %75 CZ cropping attack are shown in Table 6. As shown from the table that, recovery bits embedding time have been measured approximately three times more than authentication bits embedding time. Also, the total watermark embedding time of all images are approximately the same. Moreover, it can be seen from the table that, as the tampering rate increases, the tamper detection and recovery time increases.

Table 6 Time complexity of the proposed method

Image	Watermark embedding time (sec)			Tamper detection and recovery time (sec)		
	Recovery bits embedding	Authentication bits embedding	Total watermark embedding	%25 CZ	%50 VCZ	%75 CZ
Lena	133.128627	46.839553	179.968180	180.296284	189.142496	198.514490
Sailboat	132.485935	46.644259	179.130194	180.821834	190.707774	199.018766
Lake	132.879485	46.652739	179,532,224	179.811014	189.824777	199.324460
Airplane	132.357713	46.389058	178,746,771	180.477713	188.998814	198.762606
Baboon	133.314231	46.128864	179,443,095	180.199268	189.820586	198.406937
Goodhill	133.624994	46.151119	179,776,113	180.330294	189.730310	198.392834

Table 7 illustrates the theoretical performance comparison of our method with other methods in terms of watermarked image PSNR value, recovered image PSNR value, and condition of recovery with a Lena image. It can be seen from the table that the PSNR value of watermarked image obtained by the proposed method is higher than Yang and Shen's [55], Zhang et al's [58], Qian et al's [44], Kim et al's [28] methods. Although the PSNR values of the watermarked image obtained by Yang et al's [56] and Qin et al's [46] methods are higher than our proposed method, PSNR values of the recovered image is lower than ours. Also, it is evident from the table that the recovered image PSNR value obtained by the proposed method is higher than Yang et al's [56], Zhang et al's [58], and Qin et al's [46] methods. The Qian et al's [44] method has the best quality of recovered image when the tampering rate is smaller than 35%. Moreover, Kim et al's [28] and Yang and Shen's [55] methods have the good quality of recovered image against %50 tampering rated attacks. On the other hand, the recovery condition of our method is %75 which is higher than the other methods [28, 44, 46, 55, 56, 58].

5 Conclusions

In this study, in order to recover up to 75% tampering rated images, a novel self-embedding watermarking method has been proposed. In this method host image is divided into sixteen main blocks. Four partner blocks are selected from main blocks using Look-Up table. Triple copies of recovery information for each block are embedded into the three other blocks by using LSB substitution. Then, by using MD5 hash function, authentication information is generated from 16×16 recovery information embedded image blocks. The generated authentication information is embedded into the quarter part of each 16×16 block using LSB modification.

In the proposed method, each block in the image contains recovery information of three other blocks, which provides triple chance for recovery of the tampered areas. With the triple recovery information generation and by scrambling the recovery bits, the proposed method is not affected by tamper coincidence problem. The achievement of the proposed method has been demonstrated by applying different size of cropping attacks to the different areas of the watermarked images. Experimental results show that the proposed method satisfactorily recovers up to 75% tampering rated images.

As a future work, generation and embedding of the recovery bits can be performed with frequency domain techniques. In addition, the proposed method can be combined with robust watermarking methods for dual watermarking.

Table 7 Theoretical performance comparisons of proposed methods and the other methods

Methods	Watermarked Image PSNR (dB)	Recovered Image PSNR (dB)	Condition of recovery
Yang and Shen [55]	40.7	32	50%
Yang et al. [56]	51.3	24.36	50%
Zhang et al. [58]	37.9	29.9	59%
Qin et al. [46]	44.27	29.41	45%
Qian et al. [44]	37.9	35	35%
Kim et al. [28]	43.7	33.6	50%
Proposed	44.14	30,49	75%

References

1. Abdulrahman AK, Ozturk S (2019) A novel hybrid DCT and DWT based robust watermarking algorithm for color images. Multimed Tools Appl 78(12):17027–17049
2. Aslantas, V, Dogan, AL and Ozturk, S (2008, June). DWT-SVD based image watermarking using particle swarm optimizer. In 2008 IEEE international conference on multimedia and expo (pp. 241-244). IEEE
3. Aslantas V, Ozer S, Ozturk S (2009) Improving the performance of DCT-based fragile watermarking using intelligent optimization algorithms. Opt Commun 282(14):2806–2817
4. Ansari IA, Pant M, Ahn CW (2016) SVD based fragile watermarking scheme for tamper localization and self-recovery. Int J Mach Learn Cybern 7(6):1225–1239
5. Bravo-Solorio S, Calderon F, Li CT, Nandi AK (2018) Fast fragile watermark embedding and iterative mechanism with high self-restoration performance. Digital signal processing 73:83–92
6. Cao F, An B, Wang J, Ye D, Wang H (2017) Hierarchical recovery for tampered images based on watermark self-embedding. Displays 46:52–60
7. Chen WC, Wang MS (2009) A fuzzy c-means clustering-based fragile watermarking scheme for image authentication. Expert Syst Appl 36(2):1300–1307
8. Christlein V, Riess C, Jordan J, Riess C, Angelopoulou E (2012) An evaluation of popular copy-move forgery detection approaches. IEEE Transactions on information forensics and security 7(6):1841–1854
9. Chuang JC, Hu YC (2011) An adaptive image authentication scheme for vector quantization compressed image. J Vis Commun Image Represent 22(5):440–449
10. Depovere, G, Kalker, T, Haitsma, J, Maes, M, De Strycker, L, Termont, P, ... and O'Reilly, G (1999, October). The VIVA project: digital watermarking for broadcast monitoring. In Proceedings 1999 International Conference on Image Processing (Cat. 99CH36348) (Vol. 2, pp. 202–205). IEEE
11. Di Martino F, Sessa S (2012) Fragile watermarking tamper detection with images compressed by fuzzy transform. Inf Sci 195:62–90
12. Gul E, Ozturk S (2019) A novel hash function based fragile watermarking method for image integrity. Multimed Tools Appl 78(13):17701–17718
13. Gull, S, Loan, NA, Parah, SA, Sheikh, JA and Bhat, GM (2018). An efficient watermarking technique for tamper detection and localization of medical images. Journal of ambient intelligence and humanized computing, 1-10
14. He, H, Zhang, J, & Tai, HM (2006, November). A wavelet-based fragile watermarking scheme for secure image authentication. In international workshop on digital watermarking (pp. 422-432). Springer, Berlin, Heidelberg
15. Hore, A and Ziou, D (2010, August). Image quality metrics: PSNR vs. SSIM. In 2010 20th international conference on pattern recognition (pp. 2366-2369). IEEE
16. Huang R, Liu H, Liao X, Sun S (2019) A divide-and-conquer fragile self-embedding watermarking with adaptive payload. Multimed Tools Appl 78(18):26701–26727
17. Jindal H, Kasana SS, Saxena S (2016) A novel image zooming technique using wavelet coefficients, In proceedings of the international conference on recent cognizance in Wireless Communication & Image Processing (pp. 1–7). Springer, New Delhi
18. Jindal H, Kasana SS, Saxena S (2018) Underwater pipelines panoramic image transmission and refinement using acoustic sensors. International journal of wavelets, multiresolution and information processing 16(03): 1850013
19. Jindal H, Saxena S, Kasana SS (2017) Sewage water quality monitoring framework using multi-parametric sensors. Wirel Pers Commun 97(1):881–913
20. Jindal H, Saxena S, Kasana SS (2018) A sustainable multi-parametric sensors network topology for river water quality monitoring. Wirel Netw 24(8):3241–3265
21. Kaur S, Jindal H (2017) Enhanced image watermarking technique using wavelets and interpolation. International journal of image, graphics and signal processing 11(7):23
22. Khosravi MR, Rostami H, Samadi S (2018) Enhancing the binary watermark-based data hiding scheme using an interpolation-based approach for optical remote sensing images. International Journal of Agricultural and Environmental Information Systems (IJAEIS) 9(2):53–71
23. Khosravi MR, Samadi S (2019) Efficient payload communications for IoT-enabled ViSAR vehicles using discrete cosine transform-based quasi-sparse bit injection. EURASIP J Wirel Commun Netw 2019(1):262
24. Khosravi MR, Samadi S (2019) Reliable data aggregation in internet of ViSAR vehicles using chained dual-phase adaptive interpolation and data embedding. IEEE Internet Things J 7(4):2603–2610
25. Khosravi MR, Yazdi M (2018) A lossless data hiding scheme for medical images using a hybrid solution based on IBRW error histogram computation and quartered interpolation with greedy weights. Neural Comput & Applic 30(7):2017–2028

26. Kim C, Shin D, Leng L, Yang CN (2018) Lossless data hiding for absolute moment block truncation coding using histogram modification. J Real-Time Image Proc 14(1):101–114
27. Kim C, Shin D, Leng L, Yang CN (2018) Separable reversible data hiding in encrypted halftone image. Displays 55:71–79
28. Kim C, Shin D, Yang CN (2018) Self-embedding fragile watermarking scheme to restoration of a tampered image using AMBTC. Pers Ubiquit Comput 22(1):11–22
29. Kim C, Yang CN, Leng L (2020) High-capacity data hiding for ABTC-EQ based compressed image. Electronics 9(4):644
30. Kunhu, A, Al-Ahmad, H and Al Mansoori, S (2017, November). A reversible watermarking scheme for ownership protection and authentication of medical images. In 2017 international conference on electrical and computing technologies and applications (ICECTA) (pp. 1-4). IEEE
31. Lee TY, Lin SD (2008) Dual watermark for image tamper detection and recovery. Pattern Recogn 41(11): 3497–3506
32. Leng L, Li M, Kim C, Bi X (2017) Dual-source discrimination power analysis for multi-instance contactless palmprint recognition. Multimed Tools Appl 76(1):333–354
33. Leng, L, Zhang, J, Xu, J, Khan, MK and Alghathbar, K (2010, November). Dynamic weighted discrimination power analysis in DCT domain for face and palmprint recognition. In 2010 international conference on information and communication technology convergence (ICTC) (pp. 467-471). IEEE
34. Liao X, Li K, Yin J (2017) Separable data hiding in encrypted image based on compressive sensing and discrete fourier transform. Multimed Tools Appl 76(20):20739–20753
35. Liao X, Qin Z, Ding L (2017) Data embedding in digital images using critical functions. Signal Process Image Commun 58:146–156
36. Li C, Wang Y, Ma B, Zhang Z (2013) Multi-block dependency based fragile watermarking scheme for fingerprint images protection. Multimed Tools Appl 64(3):757–776
37. Lin PL, Hsieh CK, Huang PW (2005) A hierarchical digital watermarking method for image tamper detection and recovery. Pattern Recogn 38(12):2519–2529
38. Lin PY, Lee JS, Chang CC (2009) Dual digital watermarking for internet media based on hybrid strategies. IEEE transactions on circuits and systems for video technology 19(8):1169–1177
39. Lu CS, Liao HY (2003) Structural digital signature for image authentication: an incidental distortion resistant scheme. IEEE transactions on multimedia 5(2):161–173
40. Lu H, Shen R, Chung FL (2003) Fragile watermarking scheme for image authentication. Electron Lett 39(12):898–900
41. Mander K, Jindal H (2017) An improved image compression-decompression technique using block truncation and wavelets. International journal of image, graphics and signal processing 9(8):17
42. Mittal A, Jindal H (2017) Novelty in image reconstruction using DWT and CLAHE. International journal of image, graphics and signal processing 9(5):28
43. Peng Y, Niu X, Fu L, Yin Z (2018) Image authentication scheme based on reversible fragile watermarking with two images. Journal of information security and applications 40:236–246
44. Qian Z, Feng G, Zhang X, Wang S (2011) Image self-embedding with high-quality restoration capability. Digital Signal Processing 21(2):278–286
45. Qin C, Wang H, Zhang X, Sun X (2016) Self-embedding fragile watermarking based on reference-data interleaving and adaptive selection of embedding mode. Inf Sci 373:233–250
46. Qin C, Ji P, Zhang X, Dong J, Wang J (2017) Fragile image watermarking with pixel-wise recovery based on overlapping embedding strategy. Signal Process 138:280–293
47. Singh D, Singh SK (2016) Effective self-embedding watermarking scheme for image tampered detection and localization with recovery capability. J Vis Commun Image Represent 38:775–789
48. Singh P, Agarwal S (2016) An efficient fragile watermarking scheme with multilevel tamper detection and recovery based on dynamic domain selection. Multimed Tools Appl 75(14):8165–8194
49. Singh P, Agarwal S (2017) A self recoverable dual watermarking scheme for copyright protection and integrity verification. Multimed Tools Appl 76(5):6389–6428
50. Tang, S, He, F, Huang, X and Yang, J (2019). Online PCB defect detector on a new PCB defect dataset. arXiv preprint arXiv:1902.06197
51. Tsai P, Hu YC, Chang CC (2005) Novel image authentication scheme based on quadtree segmentation. The Imaging Science Journal 53(3):149–162
52. Wang, X, Peng, Y, Lu, L, Lu, Z, Bagheri, M and Summers, RM (2017). Chestx-ray8: hospital-scale chest x-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases. In proceedings of the IEEE conference on computer vision and pattern recognition (pp. 2097-2106)
53. Wang Z, Bovik AC, Sheikh HR, Simoncelli EP (2004) Image quality assessment: from error visibility to structural similarity. IEEE Trans Image Process 13(4):600–612

54. Weber AG (1997) The USC-SIPI image database version 5. USC-SIPI Rep 315:1–24
55. Yang CW, Shen JJ (2010) Recover the tampered image based on VQ indexing. Signal Process 90(1):331–343
56. Yang, S, Qin, C, Qian, Z and Xu, B (2014, August). Tampering detection and content recovery for digital images using halftone mechanism. In 2014 tenth international conference on intelligent information hiding and multimedia signal processing (pp. 130-133). IEEE
57. Zhang X, Wang S (2007) Statistical fragile watermarking capable of locating individual tampered pixels. IEEE Signal processing letters 14(10):727–730
58. Zhang, X, Wang, S and Feng, G (2009, August). Fragile watermarking scheme with extensive content restoration capability. In international workshop on digital watermarking (pp. 268-278). Springer, Berlin, Heidelberg
59. Zhang X, Wang S, Qian Z, Feng G (2011) Self-embedding watermark with flexible restoration quality. Multimed Tools Appl 54(2):385–395

Affiliations

Ertugrul Gul[1,2] · **Serkan Ozturk**[2]

[1] Computer Engineering Department, Nigde Omer Halisdemir University, Nigde, Turkey

[2] Computer Engineering Department, Erciyes University, Kayseri, Turkey

Terms and Conditions

Springer Nature journal content, brought to you courtesy of Springer Nature Customer Service Center GmbH ("Springer Nature").

Springer Nature supports a reasonable amount of sharing of research papers by authors, subscribers and authorised users ("Users"), for small-scale personal, non-commercial use provided that all copyright, trade and service marks and other proprietary notices are maintained. By accessing, sharing, receiving or otherwise using the Springer Nature journal content you agree to these terms of use ("Terms"). For these purposes, Springer Nature considers academic use (by researchers and students) to be non-commercial.

These Terms are supplementary and will apply in addition to any applicable website terms and conditions, a relevant site licence or a personal subscription. These Terms will prevail over any conflict or ambiguity with regards to the relevant terms, a site licence or a personal subscription (to the extent of the conflict or ambiguity only). For Creative Commons-licensed articles, the terms of the Creative Commons license used will apply.

We collect and use personal data to provide access to the Springer Nature journal content. We may also use these personal data internally within ResearchGate and Springer Nature and as agreed share it, in an anonymised way, for purposes of tracking, analysis and reporting. We will not otherwise disclose your personal data outside the ResearchGate or the Springer Nature group of companies unless we have your permission as detailed in the Privacy Policy.

While Users may use the Springer Nature journal content for small scale, personal non-commercial use, it is important to note that Users may not:

1. use such content for the purpose of providing other users with access on a regular or large scale basis or as a means to circumvent access control;
2. use such content where to do so would be considered a criminal or statutory offence in any jurisdiction, or gives rise to civil liability, or is otherwise unlawful;
3. falsely or misleadingly imply or suggest endorsement, approval , sponsorship, or association unless explicitly agreed to by Springer Nature in writing;
4. use bots or other automated methods to access the content or redirect messages
5. override any security feature or exclusionary protocol; or
6. share the content in order to create substitute for Springer Nature products or services or a systematic database of Springer Nature journal content.

In line with the restriction against commercial use, Springer Nature does not permit the creation of a product or service that creates revenue, royalties, rent or income from our content or its inclusion as part of a paid for service or for other commercial gain. Springer Nature journal content cannot be used for inter-library loans and librarians may not upload Springer Nature journal content on a large scale into their, or any other, institutional repository.

These terms of use are reviewed regularly and may be amended at any time. Springer Nature is not obligated to publish any information or content on this website and may remove it or features or functionality at our sole discretion, at any time with or without notice. Springer Nature may revoke this licence to you at any time and remove access to any copies of the Springer Nature journal content which have been saved.

To the fullest extent permitted by law, Springer Nature makes no warranties, representations or guarantees to Users, either express or implied with respect to the Springer nature journal content and all parties disclaim and waive any implied warranties or warranties imposed by law, including merchantability or fitness for any particular purpose.

Please note that these rights do not automatically extend to content, data or other material published by Springer Nature that may be licensed from third parties.

If you would like to use or distribute our Springer Nature journal content to a wider audience or on a regular basis or in any other manner not expressly permitted by these Terms, please contact Springer Nature at

onlineservice@springernature.com