Image Watermarking for Tamper Correction Based on Hamming Error Correcting Code

Pinapati Lakshmana Rao
Department of CSE, Koneru Lakshmaiah Education Foundation, Vaddeswaram 522302, Andhra Pradesh, India
plakshmanarao@kluniversity.in

Reshma Sonar
Department of Computer Engineering and Technology, Dr. Vishwanath Karad MIT World Peace University, Pune, India
sonarreshma@gmail.com

Nanditha Boddu
Vidya Jyothi Institute of Technology, Aziz Nagar Gate, C.B. Post, Hyderabad 500075, Telangana, India
nanditha.boddu@gmail.com

Anita Pradhan
Department of CSE, Koneru Lakshmaiah Education Foundation, Vaddeswaram 522302, Andhra Pradesh, India
anita.pradhan15@gmail.com

Ranjan K. Senapati
Department of ECE, VNR Vignana Jyothi Institute of Engineering & Technology, Bachupally, Nizampet (S.O), Hyderabad 500090, Telangana, India
ranjan_ks@vnrvjiet.in

Gandharba Swain
Department of CSE, Koneru Lakshmaiah Education Foundation, Vaddeswaram 522302, Andhra Pradesh, India
gswain1234@gmail.com (corresponding author)

Received: 21 April 2025 | Revised: 16 May 2025 | Accepted: 1 June 2025

 DOI: https://doi.org/10.48084/etasr.11632

ABSTRACT

Watermarking techniques are designed to embed a watermark in a manner that enables precise detection of tampering at the receiver side. This article presents an image watermarking and tamper detection technique based on a modified Hamming code. The watermarking process is applied to each individual pixel of the gray image, with each pixel is represented using 8 bits. The 4 higher plane bits of a pixel are designated as data bits, and based on these, 3 redundant bits are computed. These 3 redundant bits are then modified using a Logistic Map (LM). Furthermore, based on the 4 data bits and 3 modified redundant bits, a parity bit is computed. The 3 modified redundant bits along with the parity bit form the Watermark Bits (WBs), which are embedded into the Least Significant Bit (LSB) positions of the pixel. This process results in a watermarked pixel. In the event that a single bit is tampered with, it can be easily identified and corrected at the receiver according to the detection and correction logic. The Hiding Capacity (HC) is 4 bits per pixel, and the distortion measure is 31 dB. Although the similarity index is reduced the accuracy of tamper detection is significantly improved.

Keywords-tamper correction; image integrity check; watermarking; modified Hamming code; logistic map

I. INTRODUCTION

High-quality photo editing applications are widely available online, making it easy to tamper with a Watermarked Image (WI) [1]. Consequently, research on tamper correction emerged. Numerous recent approaches are currently available for this purpose [2, 3], with different kinds of watermarking schemes. According to human perception, the classification of watermarking schemes is typically divided into two categories: visible, and invisible. In addition, the invisible watermarking schemes can be categorized into two distinct types: fragile and robust. In the context of the domain, two categories of watermarking schemes are identified: frequency and spatial. With regard to file types, watermarking techniques can be categorized into four distinct types: image, video, text, and audio. In the spatial domain, block-wise watermarking is a prevalent practice. For enhanced security, Chaotic Maps (CM), including Logistic Map (LM), are often employed.

Authors in [3] proposed a block-wise embedding scheme utilizing 4-pixel blocks. Two Recovery Bits (RBs) and two Watermark Bits (WBs) are computed from the Most Significant Bits (MSBs) of pixels. The WBs and RBs are concealed in Least Significant Bits (LSBs). The presence of tampered blocks is detected at the receiver. Authors in [4] employed a comparable watermarking technique, whereby WBs and RBs are derived from n MSBs and stored in LSBs. Here, the value n changes from block to block to achieve security. Authors in [5] described a 4×4 block technique. In this context, the block average value is computed subsequent to the conversion of the two least significant bits (LSBs) to zero. The WBs are calculated from this average and are embedded in the lower part of the block. The block average is embedded in the upper part. The tamper detection functionality operates effectively. Authors in [6] attempted to enhance security by employing a specific hash function. Authors in [7] divided the cover medium into 16 blocks, wherein a set of four blocks are partners. A partner block is further divided into 4×4 pixels and these partners blocks are considered to compute WBs and RBs.

The aforementioned schemes are non-overlapped and block-based. An overlapped 9-pixel block-based watermarking technique was proposed by authors in [8], where the RBs and WBs are developed considering six MSBs of pixels. Additionally, WBs are embedded within the middle pixel LSB locations and RBs are embedded in other pixels. This approach rectifies the tampered blocks correctly. Authors in [9] scrambled the image by Arnold Map (AM), derived the WBs from MSBs, and concealed them in LSBs. Here, the tampered locations could not be detected. Authors in [10] computed the WBs from higher planes and hid them in lower planes. Authors in [11] calculated the WBs from five higher bits and used LM, shift, and mod operators to hide them in pixels. Here, tampered pixels are accurately identified. Authors in [12] used XOR and LM operations to embed WBs and identify the tampered places. This method demonstrated resilience in several attacks.

Authors in [13] also scrambled the image using CM, using 2×2 blocks. They calculated certain information bits from MSBs and stored them in LSBs. In this paradigm, the length of the information varies from block to block. The use of CM resulted in enhanced security. Authors in [14] employed 2×2 blocks. In this study, 12 WBs were derived using CM and consolidated to four bits. The WBs along with the RBs were buried in LSBs. In this approach, the tampered blocks were identified correctly.

Hamming proposed an $(m + r, r)$ error detection and correction code, where m is the number of data bits and r is the number of redundant bits [15]. The values of m and r must satisfy the condition $2^r \geq (m + r + 1)$. The (7, 4) Hamming code consists of three redundant bits (P_1, P_2 P_3), and four data bits (d_1, d_2, d_3, d_4), as illustrated in Figure 1. The redundant bits are calculated from the data bits using (1), (2), and (3), where the symbol $\oplus$ denotes the exclusive OR (XOR) operation.

$$P_1 = d_1 \oplus d_2 \oplus d_4 \quad (1)$$

$$P_2 = d_1 \oplus d_3 \oplus d_4 \quad (2)$$

$$P_3 = d_2 \oplus d_3 \oplus d_4 \quad (3)$$

The Hamming code identifies tampered bits by assuming the association of redundant bits with data bits to attain even parity. Furthermore, if parity detected as odd at the receiver, then tampering is detected.

7	6	5	4	3	2	1
d_4	d_3	d_2	P_3	d_1	P_2	P_1

Fig. 1. The (7, 4) Hamming code bit positioning.

Equations (1), (2), and (3) are based on the reasoning explained in Table I, which provides a self-explanatory overview. Suppose the receiver receives the seven bits of information. To correct an error, the receiver computes r_1, r_2, and r_3, using (4), (5), and (6), respectively.

$$r_1 = \mathrm{XOR}(P_1, d_1, d_2, d_4) \quad (4)$$

$$r_2 = \mathrm{XOR}(P_2, d_1, d_3, d_4) \quad (5)$$

$$r_3 = \mathrm{XOR}(P_3, d_2, d_3, d_4) \quad (6)$$

TABLE I. REASONING FOR THE CALCULATIONS OF P_1, P_2, AND P_3

Position	3rd bit	2nd bit	1st bit	
$1 = 001_2$	0	0	1	P_1
$2 = 010_2$	0	1	0	P_2
$3 = 011_2$	0	1	1	d_1
$4 = 100_2$	1	0	0	P_3
$5 = 101_2$	1	0	1	d_2
$6 = 110_2$	1	1	0	d_3
$7 = 111_2$	1	1	1	d_4
	P_3 = parity bit for positions (5,6,7), i.e. (d_2, d_3, d_4)	P_2= parity bit for positions (3,6,7), i.e. (d_1, d_3, d_4)	P_1= parity bit for positions (3,5,7), i.e. (d_1, d_2, d_4)	

The error correction process at the receiver is performed using the procedure illustrated in Figure 2. This procedure is further detailed in Table II, where the first column represents the values of three redundant bits, the second column indicates

the error position, and the third column describes the correction.

If $(r_3r_2r_1 = 000)$, then there is no error.
Elseif $(r_3r_2r_1 \neq 001)$, then there is error at P_1.
So, P_1 can be inverted.
Elseif $(r_3r_2r_1 \neq 010)$, then there is error at P_2.
So, P_2 can be inverted.
Elseif $(r_3r_2r_1 \neq 011)$, then there is error at d_1.
So, d_1 can be inverted.
Elseif $(r_3r_2r_1 \neq 100)$, then there is error at P_3.
So, P_3 can be inverted.
Elseif $(r_3r_2r_1 \neq 101)$, then there is error at d_2.
So, d_2 can be inverted.
Elseif $(r_3r_2r_1 \neq 110)$, then there is error at d_3.
So, d_3 can be inverted.
Elseif $(r_3r_2r_1 \neq 111)$, then there is error at d_4.
So, d_4 can be inverted.
Else, end

Fig. 2. Error correction using the (7,4) Hamming code

TABLE II. ERROR CORRECTION USING THE (7,4) HAMMING CODE

$r_3r_2r_1$	Tamper detection	Tamper correction
001	Error at P_1	P_1 can be inverted
010	Error at P_2	P_2 can be inverted
011	Error at d_1	d_1 can be inverted
100	Error at P_3	P_3 can be inverted
101	Error at d_2	d_2 can be inverted
110	Error at d_3	d_3 can be inverted
111	Error at d_4	d_4 can be inverted

Authors in [16], proposed a technique in which the WBs are computed from four MSBs by Hamming code and masked in LSBs. This scheme also employs LM, thereby ensuring protection against potential attacks. It is also capable of detecting tampered locations with a precision of 37 dB Peak Signal-to-Noise Ratio (PSNR). Authors in [17] obtained the WBs by applying LM and Hamming code weight. WBs are concealed by adding ±0 or ±1 to the pixel value. In this technique, tamper pixels are identified successfully. In [18], authors used the Hamming code and LM with a block size of 2 pixels. Three WBs are calculated from a block using LM and Hamming code and concealed using differencing approach. Authors in [19] also applied LM with Hamming code. A generator matrix computed three WBs from four MSBs and the WBs are concealed within the LSBs. This method demonstrates that LM enhances security.

Authors in [20] devised an image tamper correction technique that utilizes the Merkle root and pixel difference, addressing the Fall Off Boundary Problem (FOBP) appropriately. This technique works on four blocks. From the four pixels, the four MSBs are termed as quotients and the four LSBs are termed as remainders. The four quotients are put on the leaf nodes of the Merkle tree and the WBs are collected from the Merkle tree root. The WBs are hidden in the remainders using a differencing approach. The FOBP avoidance logic is applied while hiding the WBs. This technique yielded 42.02 dB PSNR and 0.9765 Structural Similarity Index (SSIM), with accurate tamper correction. Authors in [21] proposed an image tamper identification and correction procedure based on the (12, 8) modified Hamming code. At a given time, it performs over three pixels. There are 24 bits from these three pixels, which are divided into two functional units with equal length. Out of the 12 bits in the unit, 8 are data and 4 are redundant bits. The WBs are derived from the data bits using the modified Hamming code and LM. The WBs are stored over redundant bits The experimental results are satisfactory, with a PSNR of 39.21 dB, a SSIM of 0.9844, and an accuracy (ACC) of 0.9999, performing at par with existing techniques. Authors in [22] proposed an image tamper identification and correction procedure using the (7,4) Hamming code over 8-pixel blocks. Here, a 2-bit error or a 1-bit error can be detected if it occurs in only one of the eight pixels. The PSNR and SSIM values are not impressive.

Traditionally, three significant quality metrics have been established for image steganography techniques. Those metrics are Hiding Capacity (HC), PSNR, and security. As the Stego-Image (SI) is transmitted towards the receiver, the hidden classified data may get tampered by many factors. It has been demonstrated that traditional steganography techniques are ineffective in addressing this issue. To address this limitation, authors in [23, 24] used Hamming code-based error correction on hidden data embedded within an image. While traditional data hiding techniques concentrate on HC, PSNR, and security, these techniques also talked about correcting the errors after the classified data are extracted at the receiver.

A critical application of image watermarking is medical image security. A medical image may contain patient information, and while transmitted through the internet, an intruder may attempt to locate and modify the patient data. In addition, while the image is in transit, the intruder may try to retrieve the hidden confidential information. There is a plethora of literature on this subject. Authors in [25] used the watermarking technique to protect patient records. This technique utilized the Region of Choice (ROC) and the Region of Non-Choice (RONC) methodologies. Authors in [26] proposed a technique for hiding patient information in digital images using an enhanced 28-neighbour average interpolation approach, wherein they can detect and localize the tampered regions. Authors in [27] identified the border and non-border regions of the medical image as a method for concealing patient data and ensuring their security. Authors in [28] also suggested a watermarking technique based on ROC and RONC. They used a histogram expansion approach and achieved 100% reversibility and a fair PSNR value. Authors in [29] employed watermarking as a means of authentication.

II. RESEARCH MOTIVATION AND CONTRIBUTION

This research introduces a novel approach to image tamper detection and correction with the following key features:

- A modified Hamming code is introduced for error recognition and correction in each pixel of the image.

- The main contribution is the relocation of redundant bits towards the LSB side.
- The proposed technique is named Image Tamper Correction using Modified Hamming Code (ITCMHC).

III. THE PROPOSED IMAGE TAMPER CORRECTION USING MODIFIED HAMMING CODE SCHEME

A. *Logistic Map*

The LM produces a list of elements in a disorderly manner. The model accepts two seed values, α_0 and β, constrained such that $0 \leq \alpha_0 \leq 1$ and $0 < \beta \leq 4$. Knowing α_0 and β, we can compute $\alpha_1, \alpha_2, \alpha_3, \ldots, \alpha_n$ using (7).

$$\alpha_i = \beta \times \alpha_{i-1} \times (1 - \alpha_{i-1}) \quad (7)$$

In this watermarking technique, we have chosen $\alpha_0 = 0.0396$ and $\beta = 3.58$. When the value of β is greater than 3.57 but less than or equal to 4, the sequence of values will be significantly disordered. First, we compute α_i value, and then multiply it with 255 to generate another sequence X_i. Each value in sequence X_i shall be rounded-up to obtain the sequence Y_i. Again, we shall apply mod 8 on Y_i to generate Z_i. This procedure is described in (8). We will get the Z_i value such that $Z_i \leq 7$. The Z_i value shall be represented in 3 binary bits, and can be denoted as S_i.

$$X_i = \alpha_i \times 255, \; Y_i = \text{Round}(X_i),$$
$$Z_i = Y_i \bmod 8 \quad (8)$$

B. *Watermark Hiding Algorithm*

The watermark embedding process is performed at the pixel level using a modified (7,4) Hamming code integrated with a LM for added security. The step-by-step procedure is as follows:

- Step 1: The eight bits of a pixel are designated as in Figure 3. The first 4 MSBs are designated as db_4, db_3, db_2, and db_1, and the 4 LSBs are designated as rb_3, rb_2, rb_1, and rb_0. The MSBs are considered as data bits.

db_4	db_3	db_2	db_1	rb_3	rb_2	rb_1	rb_0

Fig. 3. The modified (7, 4) Hamming code.

- Step 2: The redundant bits rb'_3, rb'_2, and rb'_1 are computed using (9).

$$rb'_3 = db_2 \oplus db_3 \oplus db_4, \; rb'_2 = db_1 \oplus db_3 \oplus db_4,$$
$$rb'_1 = db_1 \oplus db_2 \oplus db_4 \quad (9)$$

- Step 3: Obtain the next value $S_i = b_3\, b_2\, b_1$ from the LM and compute $rb''_3\, rb''_2\, rb''_1$ using (10).

$$rb''_3\, rb''_2\, rb''_1 = rb'_3\, rb'_2\, rb'_1 \oplus b_3\, b_2\, b_1 \quad (10)$$

- Step 4: Compute rb''_0 using (11).

$$rb''_0 = db_4 \oplus db_3 \oplus db_2 \oplus db_1 \oplus rb''_3 \oplus rb''_2 \oplus rb''_1 \quad (11)$$

- Step 5: Replace rb_3 by rb''_3, rb_2 by rb''_2, rb_1 by rb''_1, and rb_0 by rb''_0. The watermarked pixel is as shown in Figure 4.

db_4	db_3	db_2	db_1	rb''_3	rb''_2	rb''_1	rb''_0

Fig. 4. Watermarked pixel structure after embedding.

Furthermore, Figure 5 illustrates the watermark embedding procedure.

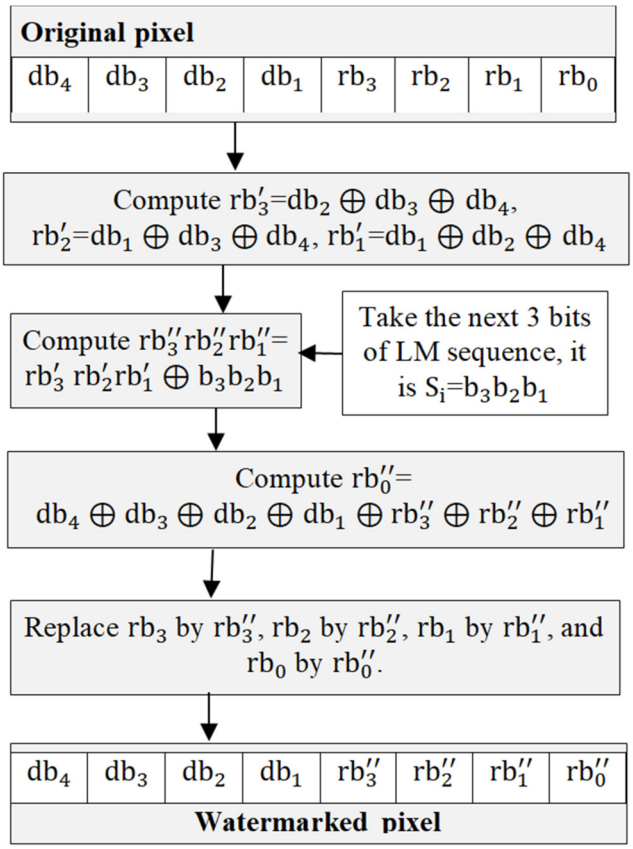

Fig. 5. Watermark hiding algorithm.

C. *Watermark Retrieval Algorithm*

The WI is raster scanned and the watermark is extracted using the below steps:

- Step 1: The eight bits of a watermarked pixel are designated as in Figure 4. Compute the parity bit (pb) using (12).

$$pb = db_4 \oplus db_3 \oplus db_2 \oplus db_1 \oplus rb''_3 \oplus rb''_2 \oplus rb''_1 \quad (12)$$

- Step 2: Obtain the next value $S_i = b_3\, b_2\, b_1$ from the LM and compute $rb'_3\, rb'_2\, rb'_1$ using (13).

$$rb'_3\, rb'_2\, rb'_1 = rb''_3\, rb''_2\, rb''_1 \oplus b_3\, b_2\, b_1 \quad (13)$$

- Step 3: Compute rb_3, rb_2, and rb_1 using (14).

$$rb_3 = rb'_3 \oplus db_2 \oplus db_3 \oplus db_4, \; rb_2 = rb'_2 \oplus db_1 \oplus db_3 \oplus db_4, \; rb_1 = rb'_1 \oplus db_1 \oplus db_2 \oplus db_4 \quad (14)$$

- Step 4: If $pb = rb''_0$ and $rb_3\, rb_2\, rb_1 = 000$, then there is no tampering, otherwise there is tampering. In the absence of tampering, the extracted WBs (EWBs) are $rb''_3\, rb''_2\, rb''_1\, rb''_0$.

- Step 5: If there has been tampering, then represent the three bits $rb_3rb_2rb_1$ to get the tampered bit location and refer to the correction procedure outlined in Figure 6. An analogous representation is shown in Table III. Note that only one-bit error detection and correction in a pixel is possible.

If $(r_3r_2r_1 = 000)$, then there is no error.
Elseif $(r_3r_2r_1 \neq 001)$, then there is error at rb_1.
So, rb_1 can be inverted.
Elseif $(r_3r_2r_1 \neq 010)$, then there is error at rb_2.
So, rb_2 can be inverted.
Elseif $(r_3r_2r_1 \neq 011)$, then there is error at db_1.
So, db_1 can be inverted.
Elseif $(r_3r_2r_1 \neq 100)$, then there is error at rb_3.
So, rb_3 can be inverted.
Elseif $(r_3r_2r_1 \neq 101)$, then there is error at db_2.
So, db_2 can be inverted.
Elseif $(r_3r_2r_1 \neq 110)$, then there is error at db_3.
So, db_3 can be inverted.
Elseif $(r_3r_2r_1 \neq 111)$, then there is error at db_4.
So, db_4 can be inverted.
Else, end

Fig. 6. Error position detection and correction using the (7,4) ITCMHC technique.

TABLE III. ERROR POSITION DETECTION AND CORRECTION USING THE (7,4) ITCMHC TECHNIQUE

$rb_3rb_2rb_1$	Tamper detection	Tamper correction
111	db_4 is erroneous	Invert db_4
110	db_3 is erroneous	Invert db_3
101	db_2 is erroneous	Invert db_2
100	rb_3 is erroneous	Invert rb_3
011	db_1 is erroneous	Invert db_1
010	rb_2 is erroneous	Invert rb_2
001	rb_1 is erroneous	Invert rb_1

Furthermore, Figure 7 illustrates the watermark extraction procedure.

IV. EXPERIMENTAL RESULTS

The proposed ITCMHC technique has been implemented in MATLAB on a computer equipped with an i5 processor and 8 GB of RAM. The USC-SIPI image database [30] was used for experimentation and the HC, PSNR, SSIM, and ACC values were computed. PSNR is computed using (15) and is expressed in dB.

$$PSNR = 10 \times \log_{10} \frac{m \times n \times 255 \times 255}{\sum_{i=1}^{m}\sum_{j=1}^{n}(P_{ij}-Q_{ij})^2} \quad (15)$$

HC refers to the total embedding capacity across all pixels in the image and the bits per pixel (bpp) represents the average HC per pixel. The SSIM metric is computed using (16) [22]. In this equation, $\overline{P}$ corresponds to the mean pixel value of the Original Image (OI), $\overline{Q}$ corresponds to the mean pixel value of the WI, σ_{pq} corresponds to the covariance between the OI and the WI, σ_p^2 corresponds to the variance of the OI, and σ_q^2 corresponds to the variance of the WI. The constants c_1 and c_2 are applied to make the terms in the denominator non-zero quantities. The value of c_1 is $(255 \times K_1)^2$and the value of c_2 is $(255 \times K_2)^2$, where $K_1 \ll 1$ and $K_2 \ll 1$. The SSIM is 1 if the WI and the OI are exactly same.

$$SSIM = \frac{(2\overline{P}\,\overline{Q}+c_1)(2\sigma_{pq}+c_2)}{(\overline{P}^2+\overline{Q}^2+c_1)(\sigma_p^2+\sigma_q^2+c_2)} \quad (16)$$

ACC is used to represent the tamper detection rate. It is calculated by dividing the sum of true negatives and true positives by the total number of possible cases.

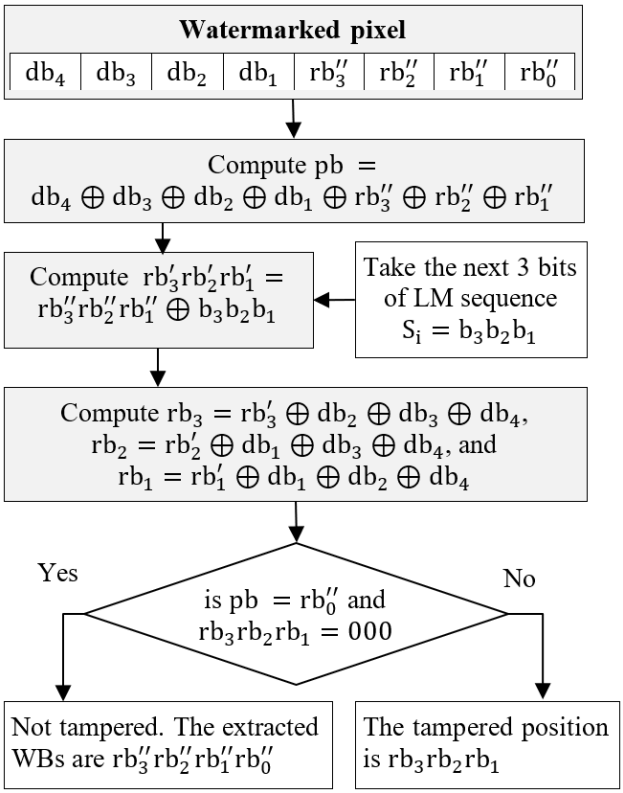

Fig. 7. Watermark retrieval algorithm.

As demonstrated in Table IV, the PSNR consistently exceeds 31 dB, which is generally accepted as adequate. The HC is 4 bpp which is relatively high. Due to this increased HC, the PSNR is reduced, highlighting the tradeoff between PSNR and HC. The SSIM is 0.8212, indicating that both WI and OI are 82% structurally similar. The value of ACC is 1.0, which indicates that all tampered pixels were detected. Table V presents the PSNR, bpp, SSIM, and ACC values as an average of eight images at tampering rates of 5%, 10%, 15%, and 20%. As illustrated in Table V, even when the tampering level exceeds 20%, the PSNR remains above 31 dB, SSIM exceeds 0.81, and ACC continues to exceed 0.99.

Table VI presents a comparison of the proposed ITCMHC scheme with those introduced in [13, 16, 18, 19]. The scheme proposed in [13] is based on CM, whereas the three remaining schemes utilize the Hamming code. Figure 8 depicts a bar diagram comparing the HC and PSNR values of the ITCMHC

scheme against these existing methods. Figure 9 illustrates a bar diagram comparing ACC and SSIM. Figure 8 reveals that the HC of the ITCMHC scheme exceeds that of the existing schemes; however, this is accompanied by a reduction in PSNR. Despite the fact that the PSNR of the ITCMHC technique is lower, it is at least 31 dB, which renders it acceptable. As illustrated in Figure 9, the SSIM values of the ITCMHC scheme are lower than those of existing schemes. However, the ACC of the ITCMHC technique is higher than that of existing schemes.

TABLE IV. PERFORMANCE RESULTS OF THE ITCMHC TECHNIQUE

Image (512×512)	PSNR (dB)	ACC	SSIM	bpp
Lena	31.77	1.0	0.7746	4
Baboon	31.85	1.0	0.9104	4
Goldhill	31.86	1.0	0.8282	4
Crowd	31.92	1.0	0.8323	4
Cameraman	31.82	1.0	0.7626	4
Pepper	31.90	1.0	0.7779	4
Barbara	31.85	1.0	0.8569	4
Boat	31.94	1.0	0.8267	4
Average	31.86	1.0	0.8212	4

TABLE V. TAMPERING RATE VS EFFICACY PARAMETERS FOR THE ITCMHC TECHNIQUE

Tampering rate (%)	PSNR (dB)	ACC	SSIM	bpp
5	31.75	0.9991	0.8197	4
10	31.65	0.9984	0.8184	4
15	31.56	0.9977	0.8175	4
20	31.47	0.9969	0.8164	4

TABLE VI. EFFICACY COMPARISON OF ITCMHC SCHEME WITH EXISTING SCHEMES

Scheme	PSNR (dB)	ACC	SSIM	bpp
ITCMHC	31.86	1.0	0.8212	4.0
[13]	36.50	0.9845	0.9928	1.66
[18]	42.09	0.9995	0.9994	1.5
[16]	37.88	0.9969	0.9844	3.0
[19]	37.94	0.9990	0.9861	3.0

Figure 10 illustrates an OI, WI, and two tampered images. In images that have undergone tampering, the pixels that have been altered are indicated by a white color. A pixel is considered tampered if its value is modified by at least one bit. The subsequent discussion in case 1 provides an explanation of watermark embedding in a pixel. Case 2 provides further elaboration on the process of watermark extraction. Case 3 provides a detailed explanation of tamper detection and correction.

- Case 1: Watermark embedding: Consider embedding the watermark in a pixel as shown in Figure 11. Using (9), the redundant bits are computed as: $rb'_3 = 0$, $rb'_2 = 0$, and $rb'_1 = 1$. Suppose the next value from the LM sequence is $S_i = 101$, i.e., $b_3b_2b_1 = 101$. According to (10), $rb''_3rb''_2rb''_1 = rb'_3rb'_2rb'_1 \oplus b_3b_2b_1 = 001 \oplus 101 = 100$. Next, compute rb''_0 using (11): $rb''_0 = 1 \oplus 0 \oplus 1 \oplus 1 \oplus 1 \oplus 0 \oplus 0 = 0$. After replacing rb_3 by rb''_3, rb_2 by rb''_2, rb_1 by rb''_1, and rb_0 by rb''_0, the watermarked pixel is as shown in Figure 12.

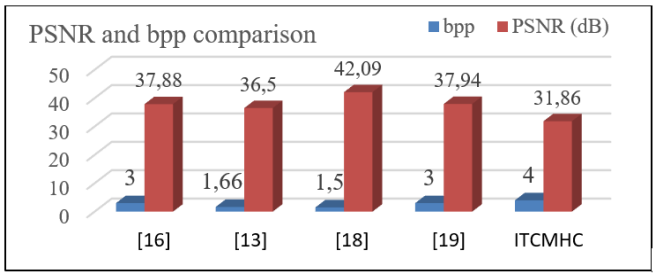

Fig. 8. Comparison of PSNR and bpp between ITCMHC and existing schemes.

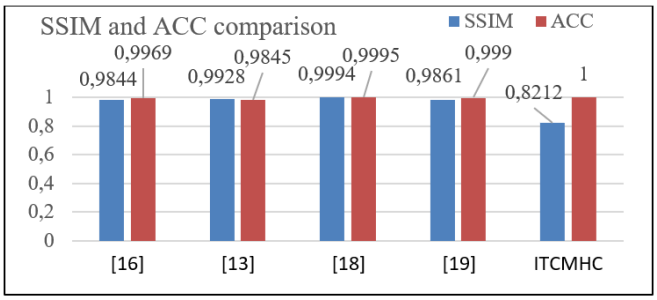

Fig. 9. Comparison of SSIM and ACC between ITCMHC and existing schemes.

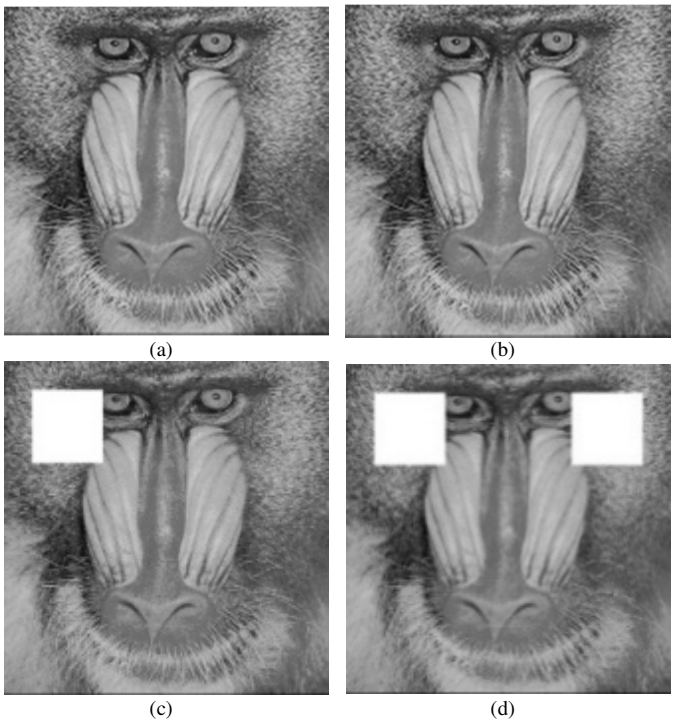

Fig. 10. Watermarking and tampering illustration: (a) Original Baboon image, (b) WI, (c) 5% tampered WI, and (d) 10% tampered WI.

db_4	db_3	db_2	db_1	rb_3	rb_2	rb_1	rb_0
1	0	1	1	1	1	0	0

Fig. 11. The 8-bit representation of a sample pixel.

db_4	db_3	db_2	db_1	rb_3''	rb_2''	rb_1''	rb_0''
1	0	1	1	1	0	0	0

Fig. 12. The resulting watermarked pixel.

- Case 2: Watermark extraction: The extraction of the watermark from the pixel shown in Figure 12 can be performed as follows: Compute pb using (12), i.e., $pb = db_4 \oplus db_3 \oplus db_2 \oplus db_1 \oplus rb_3'' \oplus rb_2'' \oplus rb_1''$. Substituting the values, $pb = 0$. Here, $pb = rb_0''$. The next three bits from the LM sequence are 101, i.e. $b_3b_2b_1 = 101$. Using (13), $rb_3'rb_2'rb_1' = 100 \oplus 101 = 001$. Hence, we get $rb_3' = 0$, $rb_2' = 0$, and $rb_1' = 1$. Using (14), we get $rb_3 = 0$, $rb_2 = 0$, and $rb_1 = 0$. Here $rb_3rb_2rb_1 = 000$, so there has been no tampering. The EWBs are $rb_3''rb_2''rb_1''rb_0''$, i.e., 1000.
- Case 3: Tamper correction: Suppose the value db_1 is tampered from 1 to 0. Then, the receiver receives the bits as in Figure 13. Compute pb using (12), $pb = 1$, and here $pb \neq rb_0''$. So, the tampered location is to be detected. The next three bits from the LM sequence are 101, i.e. $b_3b_2b_1 = 101$. Using (13), we have $rb_3'rb_2'rb_1' = 100 \oplus 101 = 001$. Hence, we get $rb_3' = 0$, $rb_2' = 0$, and $rb_1' = 1$. Using (14), we get $rb_3 = 0$, $rb_2 = 1$, and $rb_1 = 1$. Here, $rb_3rb_2rb_1 = 011$, and as per Table I, the tampered position is db_1. So, the db_1 bit can be altered. After, the corrected block is shown in Figure 14.

db_4	db_3	db_2	db_1	rb_3''	rb_2''	rb_1''	rb_0''
1	0	1	0	1	0	0	0

Fig. 13. Received pixel with tampered bit db_1.

db_4	db_3	db_2	db_1	rb_3''	rb_2''	rb_1''	rb_0''
1	0	1	1	1	0	0	0

Fig. 14. Corrected pixel after recovering the tampered bit db_1.

V. CONCLUSION

This article proposes a watermarking technique called Image Tamper Correction using Modified Hamming Code (ITCMHC). The main contribution of this study is the development of a new Error Correction Code (ECC), designated as the modified Hamming code, which is implemented on seven bits of each image pixel, excluding the Least Significant Bit (LSB). Unlike the conventional Hamming code, in this novel ECC, the data bits are positioned on the extreme left, and the redundant bits are positioned on the extreme right of the bit pattern. The Hiding Capacity (HC) of this technique is 4 bits per pixel (bpp), whereas the Peak Signal-To-Noise Ratio (PSNR) consistently exceeds 31 dB. While a higher HC results in a modest decrease in PSNR, the values persist within the acceptable range of 30-40 dB. If a single bit is tampered in the Watermarked Image (WI) during transit from the sender to the receiver, it can be efficiently identified at the receiver and corrected accurately. In future work, the focus will be on the correction of multiple bit errors in a pixel.

REFERENCES

[1] N. B. A. Warif *et al.*, "Copy-move forgery detection: Survey, challenges and future directions," *Journal of Network and Computer Applications*, vol. 75, pp. 259–278, Nov. 2016, https://doi.org/10.1016/j.jnca.2016.09.008.

[2] A. R. Gottimukkala, N. Kumar, J. K. Dash, and G. Swain, "Image watermarking based on remainder value differencing and extended Hamming code," *Journal of Electronic Imaging*, vol. 33, no. 1, Nov. 2023, Art. no. 011003, https://doi.org/10.1117/1.JEI.33.1.011003.

[3] D. Singh and S. K. Singh, "Effective self-embedding watermarking scheme for image tampered detection and localization with recovery capability," *Journal of Visual Communication and Image Representation*, vol. 38, pp. 775–789, Jul. 2016, https://doi.org/10.1016/j.jvcir.2016.04.023.

[4] F. Cao, B. An, J. Wang, D. Ye, and H. Wang, "Hierarchical recovery for tampered images based on watermark self-embedding," *Displays*, vol. 46, pp. 52–60, Jan. 2017, https://doi.org/10.1016/j.displa.2017.01.001.

[5] S. Gull, N. A. Loan, S. A. Parah, J. A. Sheikh, and G. M. Bhat, "RETRACTED ARTICLE: An efficient watermarking technique for tamper detection and localization of medical images," *Journal of Ambient Intelligence and Humanized Computing*, vol. 11, no. 5, pp. 1799–1808, May 2020, https://doi.org/10.1007/s12652-018-1158-8.

[6] S. Bhalerao, I. A. Ansari, and A. Kumar, "A secure image watermarking for tamper detection and localization," *Journal of Ambient Intelligence and Humanized Computing*, vol. 12, no. 1, pp. 1057–1068, Jan. 2021, https://doi.org/10.1007/s12652-020-02135-3.

[7] E. Gul and S. Ozturk, "A novel triple recovery information embedding approach for self-embedded digital image watermarking," *Multimedia Tools and Applications*, vol. 79, no. 41, pp. 31239–31264, Nov. 2020, https://doi.org/10.1007/s11042-020-09548-4.

[8] C. Qin, P. Ji, X. Zhang, J. Dong, and J. Wang, "Fragile image watermarking with pixel-wise recovery based on overlapping embedding strategy," *Signal Processing*, vol. 138, pp. 280–293, Sep. 2017, https://doi.org/10.1016/j.sigpro.2017.03.033.

[9] S. Rawat and B. Raman, "A chaotic system based fragile watermarking scheme for image tamper detection," *AEU - International Journal of Electronics and Communications*, vol. 65, no. 10, pp. 840–847, Oct. 2011, https://doi.org/10.1016/j.aeue.2011.01.016.

[10] M. Botta, D. Cavagnino, and V. Pomponiu, "A successful attack and revision of a chaotic system based fragile watermarking scheme for image tamper detection," *AEU - International Journal of Electronics and Communications*, vol. 69, no. 1, pp. 242–245, Jan. 2015, https://doi.org/10.1016/j.aeue.2014.09.004.

[11] S. Prasad and A. K. Pal, "A Secure Fragile Watermarking Scheme for Protecting Integrity of Digital Images," *Iranian Journal of Science and Technology, Transactions of Electrical Engineering*, vol. 44, no. 2, pp. 703–727, Jun. 2020, https://doi.org/10.1007/s40998-019-00275-7.

[12] A. K. Sahu, "A logistic map based blind and fragile watermarking for tamper detection and localization in images," *Journal of Ambient Intelligence and Humanized Computing*, vol. 13, no. 8, pp. 3869–3881, Aug. 2022, https://doi.org/10.1007/s12652-021-03365-9.

[13] M. Nazari, A. Sharif, and M. Mollaeefar, "An improved method for digital image fragile watermarking based on chaotic maps," *Multimedia Tools and Applications*, vol. 76, no. 15, pp. 16107–16123, Aug. 2017, https://doi.org/10.1007/s11042-016-3897-x.

[14] K. Sreenivas and V. Kamakshiprasad, "Improved image tamper localisation using chaotic maps and self-recovery," *Journal of Visual Communication and Image Representation*, vol. 49, pp. 164–176, Nov. 2017, https://doi.org/10.1016/j.jvcir.2017.09.001.

[15] R. W. Hamming, "Error detecting and error correcting codes," *The Bell System Technical Journal*, vol. 29, no. 2, pp. 147–160, Apr. 1950, https://doi.org/10.1002/j.1538-7305.1950.tb00463.x.

[16] C.-C. Chang, K.-N. Chen, C.-F. Lee, and L.-J. Liu, "A secure fragile watermarking scheme based on chaos-and-hamming code," *Journal of*

Systems and Software, vol. 84, no. 9, pp. 1462–1470, Sep. 2011, https://doi.org/10.1016/j.jss.2011.02.029.

[17] S. Trivedy and A. K. Pal, "A Logistic Map-Based Fragile Watermarking Scheme of Digital Images with Tamper Detection," *Iranian Journal of Science and Technology, Transactions of Electrical Engineering*, vol. 41, no. 2, pp. 103–113, Jun. 2017, https://doi.org/10.1007/s40998-017-0021-9.

[18] S. Prasad and A. K. Pal, "A tamper detection suitable fragile watermarking scheme based on novel payload embedding strategy," *Multimedia Tools and Applications*, vol. 79, no. 3, pp. 1673–1705, Jan. 2020, https://doi.org/10.1007/s11042-019-08144-5.

[19] S. Prasad and A. K. Pal, "Hamming code and logistic-map based pixel-level active forgery detection scheme using fragile watermarking," *Multimedia Tools and Applications*, vol. 79, no. 29, pp. 20897–20928, Aug. 2020, https://doi.org/10.1007/s11042-020-08715-x.

[20] S. N. V. J. Devi Kosuru, G. Swain, N. Kumar, and A. Pradhan, "Image tamper detection and correction using Merkle tree and remainder value differencing," *Optik*, vol. 261, Jul. 2022, Art. no. 169212, https://doi.org/10.1016/j.ijleo.2022.169212.

[21] A. R. Gottimukkala, A. Pradhan, P. Patro, J. Hemalata, R. K. Senapati, and G. Swain, "Image Tamper Detection and Correction Using Modified Hamming Code with Eight Data Bits and Four Redundant Bits," *International Journal of Safety and Security Engineering*, vol. 14, no. 6, pp. 1871–1881, Dec. 2024, https://doi.org/10.18280/ijsse.140621.

[22] A. R. Gottimukkala, A. Pradhan, N. Kumar, A. K. Pradhan, R. K. Senapati, and G. Swain, "Digital Image Watermarking for Image Integrity Verification and Tamper Correction," *Contemporary Mathematics*, vol. 6, no. 2, pp. 1853–1873, Mar. 2025, https://doi.org/10.37256/cm.6220256272.

[23] D. B. Khadse and G. Swain, "Data Hiding and Integrity Verification based on Quotient Value Differencing and Merkle Tree," *Arabian Journal for Science and Engineering*, vol. 48, no. 2, pp. 1793–1805, Feb. 2023, https://doi.org/10.1007/s13369-022-06961-9.

[24] S. N. V. J. D. Kosuru, A. Pradhan, K. A. Basith, R. Sonar, and G. Swain, "Digital Image Steganography With Error Correction on Extracted Data," *IEEE Access*, vol. 11, pp. 80945–80957, 2023, https://doi.org/10.1109/ACCESS.2023.3300918.

[25] A. Singh and M. K. Dutta, "A robust zero-watermarking scheme for tele-ophthalmological applications," *Journal of King Saud University - Computer and Information Sciences*, vol. 32, no. 8, pp. 895–908, Oct. 2020, https://doi.org/10.1016/j.jksuci.2017.12.008.

[26] S. Gull, R. F. Mansour, N. O. Aljehane, and S. A. Parah, "A self-embedding technique for tamper detection and localization of medical images for smart-health," *Multimedia Tools and Applications*, vol. 80, no. 19, pp. 29939–29964, Aug. 2021, https://doi.org/10.1007/s11042-021-11170-x.

[27] P. Singh, K. J. Devi, H. K. Thakkar, M. Bilal, A. Nayyar, and D. Kwak, "Robust and Secure Medical Image Watermarking for Edge-Enabled e-Healthcare," *IEEE Access*, vol. 11, pp. 135831–135845, 2023, https://doi.org/10.1109/ACCESS.2023.3335172.

[28] S. Bhalerao, I. A. Ansari, and A. Kumar, "A Reversible Medical Image Watermarking for ROI Tamper Detection and Recovery," *Circuits, Systems, and Signal Processing*, vol. 42, no. 11, pp. 6701–6725, Nov. 2023, https://doi.org/10.1007/s00034-023-02416-0.

[29] A. Tareef, K. Al-Tarawneh, and A. Sleit, "Block-based Watermarking for Robust Authentication and Integration of GIS Data," *Engineering, Technology & Applied Science Research*, vol. 14, no. 5, pp. 16340–16345, Oct. 2024, https://doi.org/10.48084/etasr.8197.

[30] "SIPI Image Database." Signal and Image Processing Institute, University of Southern California (USC). [Online]. Available: https://sipi.usc.edu/database/database.php?volume=misc.