

## Reversible Data Hiding in Encrypted Images Based on Adaptive Prediction and Labeling

Jiaohua Qin<sup>1,\*</sup>, Zhibin He<sup>1</sup>, Xuyu Xiang<sup>1</sup> and Neal N. Xiong<sup>2</sup>

<sup>1</sup>College of Computer Science and Information Technology, Central South University of Forestry & Technology, Changsha, 410004, China

<sup>2</sup>Department of Mathematics and Computer Science, Northeastern State University, Tahlequah, 74464, OK, USA

\*Corresponding Author: Jiaohua Qin. Email: qinjiaohua@163.com

Received: 24 March 2022; Accepted: 07 May 2022

**Abstract:** Recently, reversible data hiding in encrypted images (RDHEI) based on pixel prediction has been a hot topic. However, existing schemes still employ a pixel predictor that ignores pixel changes in the diagonal direction during prediction, and the pixel labeling scheme is inflexible. To solve these problems, this paper proposes reversible data hiding in encrypted images based on adaptive prediction and labeling. First, we design an adaptive gradient prediction (AGP), which uses eight adjacent pixels and combines four scanning methods (i.e., horizontal, vertical, diagonal, and diagonal) for prediction. AGP can adaptively adjust the weight of the linear prediction model according to the weight of the edge attribute of the pixel, which improves the prediction ability of the predictor for complex images. At the same time, we adopt an adaptive huffman coding labeling scheme, which can adaptively generate huffman codes for labeling according to different images, effectively improving the scheme's embedding performance on the dataset. The experimental results show that the algorithm has a higher embedding rate. The embedding rate on the test image Jetplane is 4.2102 bpp, and the average embedding rate on the image dataset Bossbase is 3.8625 bpp.

**Keywords:** Reversible data hiding; adaptive gradient prediction; huffman coding; embedding capacity

### 1 Introduction

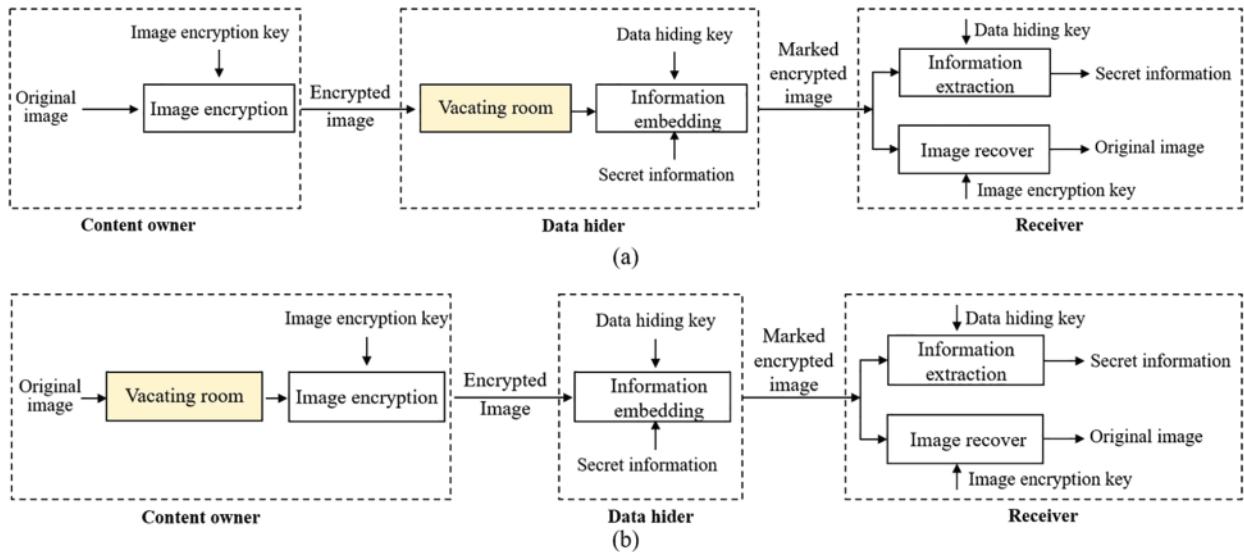
With the continuous development of digital media technology [1], the security problem of information transmission is also becoming more and more serious. Information hiding plays a vital role in information security as an essential information protection technology. Information hiding refers to the stealthy embedding of information into a carrier without arousing suspicions [2], such as steganography [3,4] and digital watermarking [5,6]. However, these methods will make certain changes to the carrier, resulting in the receiver being unable to recover the original carrier after extracting the secret information, which is terrible in the military and medical fields because the carrier's original information is also terrible. Reversible data hiding (RDH) can effectively solve this dilemma



This work is licensed under a Creative Commons Attribution 4.0 International License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

and therefore has become a popular research direction, which can extract secret information accurately and recover the original image without image quality degradation. Early proposed RDH techniques are mainly implemented in the plaintext domain and can be classified into three categories depending on the embedding ways: difference expansion [7,8], histogram shifting [9,10], and lossless compression [11,12]. However, with the increasingly severe privacy leakage problem, coverless steganography [13], information hiding technology in the ciphertext domain, and multimodality [14] have attracted more and more attention.

Therefore, based on RDH, researchers have proposed reversible data hiding in encrypted images (RDHEI), which additionally allows the data hider to hide additional information in the ciphertext. As shown in Fig. 1, the existing RDHEI technologies are mainly based on RDH combined with two mechanisms: vacating room after encryption (VRAE) [15–18] and reserving room before encryption (RRBE) [19–25]. VRAE can hide data by the redundant space after image encryption, for example, Zhang et al. [16] proposed an RDHEI scheme based on image block embedding in 2011, embedding data by modifying a part of encrypted data and supporting the receiver to extract data and recover the image in a separable manner, but the image block size limits its capacity; Zhang et al. [17] reserved more space for auxiliary information and secret data by compressing the least significant bit (LSB) of encrypted pixels; Hong et al. [18] proposed a new smoothness evaluation function to evaluate the fluctuation of pixels, which effectively improves the low embedding rate in the case of small image blocks. As seen, VRAE cannot achieve good embedding performance as the encryption operation inevitably destroys the spatial correlation of the original image. On the contrary, RRBE makes full use of the redundant information of image pixels to reserve space before image encryption and then transfers it to the encrypted image with higher embedding capacity. Obviously, it makes more sense for us to adopt RRBE as the research subject.



**Figure 1:** Two frameworks of RDHEI algorithm: (a) VRAE (b) RRBE

Yi et al. [22] proposed an RDHEI scheme based on parametric binary tree labeling (PBTL), which proposed a PBTL pixel labeling scheme to ensure that the scheme can extract data and restore images reversibly. Wu et al. [23] proposed an RDHEI scheme based on improved parametric binary tree labeling (IPBTL), which improves upon scheme [22]. The IPBTL classifies and labels pixels according

to the prediction error of the image, leaving space for data embedding and improving the embedding rate of the scheme. However, its predictor and coding efficiency still affect the embedding performance. Yin et al. [24] designed an RDHEI method based on multiple most significant bit prediction and huffman coding (MMPHC), which compared the most significant bit (MSB) of the original pixel with the predicted pixel, used the bits of the same sequence as a label, and marked the pixel with a pre-defined huffman code. However, experiments show that predefined huffman encodings do not perform markup well. To take advantage of the correlation of adjacent pixels, Yin et al. [25] proposed an RDHEI scheme based on pixel prediction and bit-plane compression (PPBC) by utilizing the correlation of neighboring pixels, which reserved more space for data embedding due to its better compression effect. Despite the exemplary achievements of all the above-mentioned methods, they share a common drawback: the used classifier median edge detector (MED) has a small receptive field, its prediction form is too homogeneous to predict complex images. Besides, the existing codes still have many shortcomings as an essential aspect. Most of the methods use the correlation between pixels to generate the embedding space and achieve the reversible embedding and extraction of secret information by establishing the mapping relationship between the prediction error of the pixel and the marking code. While this coding approach has two drawbacks: 1. The sequence of labeling depends on the set predefined parameters, and different parameter settings may yield different embedding rates, which leads to instability of embedding rates. 2. It uses a uniform sequence for labeling different types of images, so it has some limitations, e.g., it does not work well on large image datasets.

To solve the above problem, we design an adaptive gradient predictor (AGP) and use an adaptive huffman coding labeling method to propose a new RDHEI algorithm. The proposed adaptive gradient predictor uses eight adjacent pixels that combine four scanning methods (i.e., horizontal, vertical, diagonal, and anti-diagonal) in prediction, whose weights can be adaptively adjusted to construct a local linear prediction model based on the edge properties of the neighboring pixels. The adaptive huffman coding tagging algorithm statistically analyzes the prediction errors of different images and then adaptively generates the corresponding coding rules to achieve better results on multiple datasets.

The main contributions of this paper are as follows:

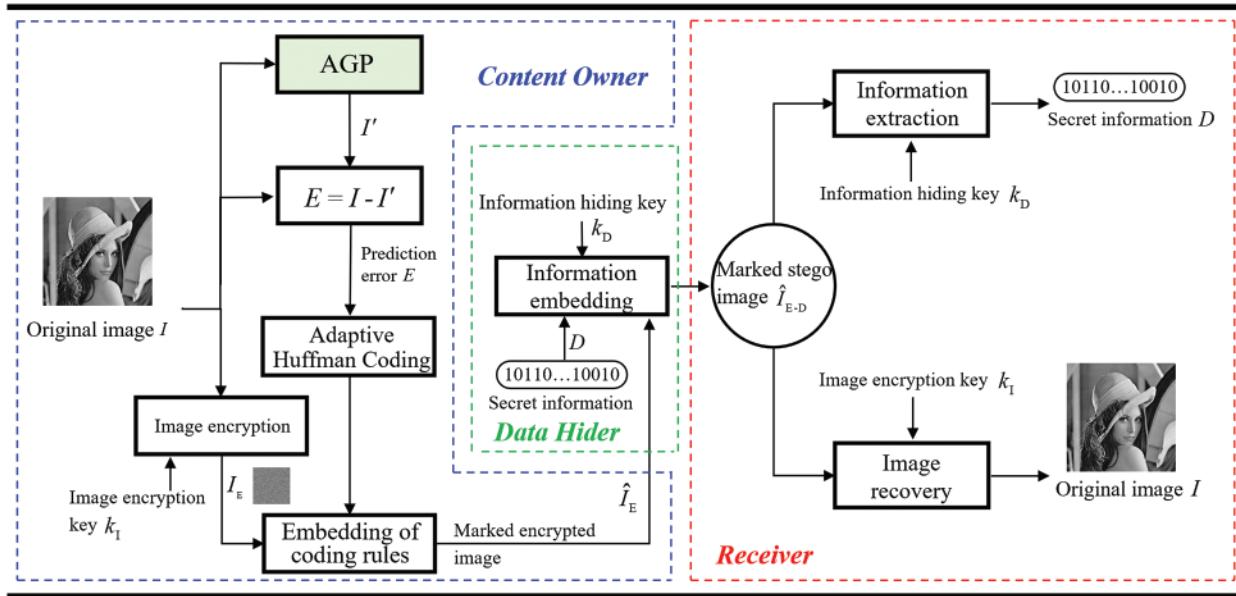
1. We propose a new and well-performing adaptive gradient prediction model, which can assign different weights to the surrounding pixels according to the neighborhood complexity of the target pixels in the prediction process, and consider the impact of pixel changes in different directions on the prediction, which better characterizes the local features of the target pixels.
2. We further combine the proposed AGP with the adaptive huffman coding labeling method to statistically analyze the prediction errors of different images and adaptively generate different huffman codes for labeling.
3. Experimental results show that our scheme achieves higher embedding rates, especially on large datasets, e.g., the average embedding rate of the BOSSbase dataset reaches 3.8625 bpp, which exceeds other state-of-the-art algorithms.

The rest of this paper are organized as follows. Section 2 presents the proposed scheme proposed, Section 3 presents the experimental evaluations, and Section 4 concludes the article.

## 2 Proposed Scheme

This section illustrates reversible data hiding in encrypted images based on adaptive gradient prediction and code labeling method. The algorithm framework is shown in Fig. 2, which can be divided into three parts. In the first part, the content owner first predicts and encrypts the original

image using AGP and image encryption keys. Then, the huffman coding is adaptively generated according to the prediction error, and the coding rules are embedded into the encrypted image. In the second part, after obtaining the marked encrypted image, the data hider first extracts the encoding rules to determine the position of the reserved room. Then the encrypted information is embedded in the reserved room to form a marked stego image. In the third part, the receiver can perform three operations. The original secret information can be extracted when the information hiding key is obtained. The original image can be recovered when the image encryption key is obtained. The original image and the secret information can be obtained simultaneously when both keys are received.



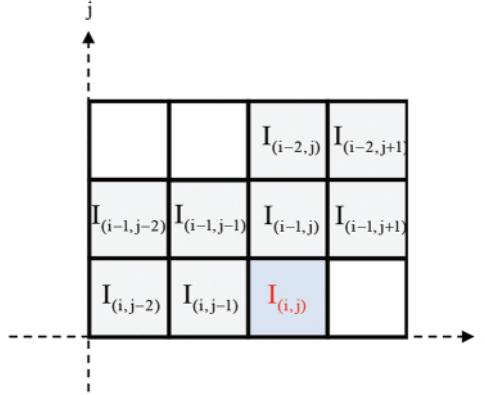
**Figure 2:** The framework of the proposed scheme

## 2.1 Image Preprocessing

Before embedding the information, the content owner preprocesses the original image to reserve room for the embedded information. The preprocessing process of our scheme mainly includes two parts. First, the prediction error is calculated according to the AGP, and then the prediction error is statistically analyzed, and the huffman code is adaptively generated.

### 2.1.1 Adaptive Gradient Prediction

The proposed adaptive gradient predictor AGP is shown in Fig. 3, which can meet the multi-directional requirements of natural images for texture attributes by perceiving pixel changes from four directions: horizontal, vertical, diagonal, and anti-angle around the target pixel. Its weight distribution of different gradient pixels is adjusted according to the position of the target pixel. The specific prediction steps can be divided into two stages. The gradient strength of the predicted pixel is calculated first. Then, according to the gradient environment where the target pixels are located, the weights are adaptively adjusted to construct a local linear prediction model for prediction. The pixels in the first row and the first column remain unchanged as reference pixels during the image prediction process.



**Figure 3:** Predictive template used by AGP

Before calculating the local complexity, the gradient strength of the local pixel is estimated by Eq. (1).

$$\begin{aligned} G_h &= \sum_{t=0}^1 |I_{(i-1,j+1-t)} - I_{(i-1,j-t)}| + \sum_{t=0}^1 |I_{(i-t,j-1)} - I_{(i-t,j-2)}| \\ G_v &= \sum_{t=0}^1 |I_{(i-2,j+1-t)} - I_{(i-1,j+1-t)}| + \sum_{t=0}^1 |I_{(i-1,j-1-t)} - I_{(i,j-1-t)}| \\ G_d &= |I_{(i-1,j-2)} - I_{(i,j-1)}| + |I_{(i-2,j)} - I_{(i-1,j+1)}| \\ G_a &= |I_{(i-1,j-1)} - I_{(i,j-2)}| + |I_{(i-2,j+1)} - I_{(i-1,j)}| \end{aligned} \quad (1)$$

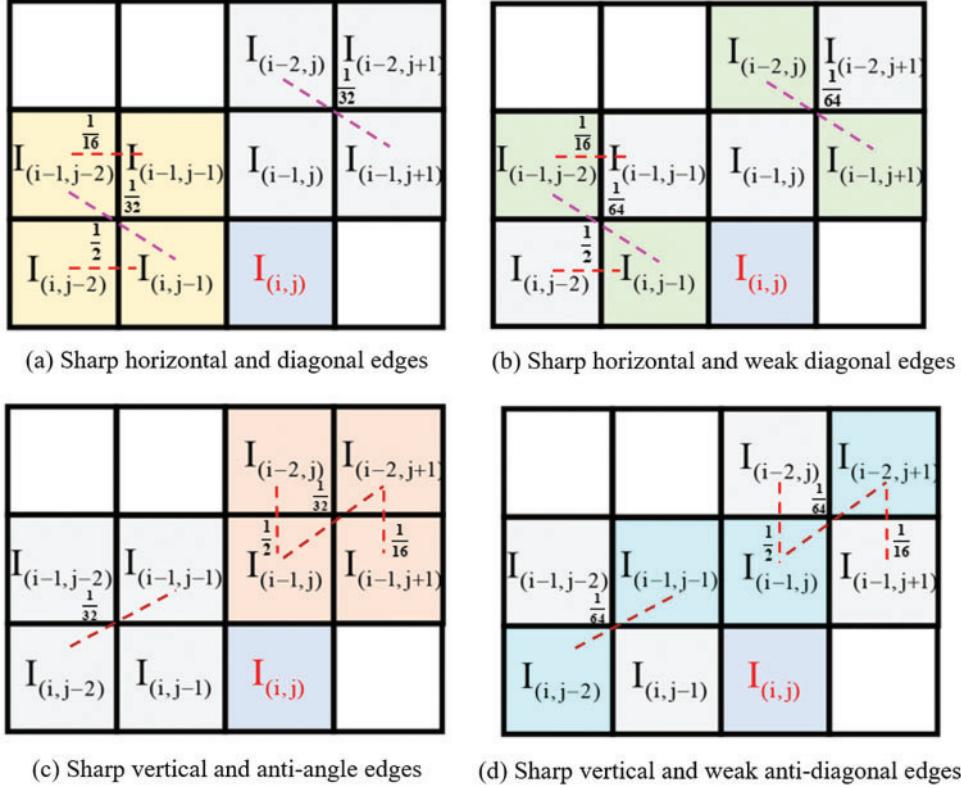
where  $G_h$ ,  $G_v$ ,  $G_d$ , and  $G_a$  represent the gradient strengths of the predicted pixels from the horizontal, vertical, diagonal, and anti-diagonal directions, respectively.

The predicted value of the pixel in our scheme is obtained according to the weighting of the domain pixels. According to the texture properties of the pixels, the prediction cases are divided into sharp edges and non-sharp edges. Sharp edges indicate significant changes in surrounding pixels, and non-sharp edges indicate regular or minor changes in surrounding pixels. When the pixel is at a sharp edge, the weight distribution of the scanning mode is shown in Fig. 4.

As shown in Fig. 4, to adapt the scheme to images with complex textures, this scheme adjusts the scanning mode's weight according to the neighborhood complexity of pixels. When the target pixel is on a sharp horizontal edge, its predicted value is mainly affected by the pixels in the horizontal direction, so the weight of the horizontal scanning method is larger. The weight of the diagonal direction is also divided into strong edges and weak edges, which are determined according to the surrounding gradient strength. When the pixel is on a sharp edge, the predicted value is obtained by

$$\begin{aligned} I'_{(i,j)} &= \lfloor \gamma_t + \omega \Delta_t \rfloor, t = \begin{cases} 0 & G_v - G_h > T_1 \\ 1 & G_h - G_v \geq T_1 \end{cases}, \omega = \begin{cases} 1 & |G_d - G_a| > T_2 \\ \frac{1}{2} & |G_d - G_a| \leq T_2 \end{cases} \\ \Delta_0 &= \frac{1}{32} (2(I_{(i-1,j-1)} - I_{(i-1,j-2)}) + (I_{(i-1,j-2)} - I_{(i,j-1)}) + (I_{(i-1,j+1)} - I_{(i-2,j)})) \\ \Delta_1 &= \frac{1}{32} (2(I_{(i-1,j+1)} - I_{(i-2,j+1)}) + (I_{(i-2,j+1)} - I_{(i-1,j)}) + (I_{(i-1,j-1)} - I_{(i,j-2)})) \\ \gamma_0 &= I_{(i,j-1)} + \frac{I_{(i,j-1)} - I_{(i,j-2)}}{2}, \gamma_1 = I_{(i-1,j)} + \frac{I_{(i-1,j)} - I_{(i-2,j)}}{2} \end{aligned} \quad (2)$$

where  $\gamma_0, \gamma_1$  represent the dominant values when predicting pixels at sharp horizontal and vertical edge locations.  $\Delta_0, \Delta_1$  represent the influence of pixels in the diagonal and anti-diagonal directions on the predicted value during prediction, respectively.  $t$  is used to judge that the predicted value is mainly affected by the pixels in the vertical or horizontal direction.  $\omega$  represents the weight of the scanning mode of different gradient strengths.



**Figure 4:** Weight assignment of scan mode for sharp edges

When the pixels are on sharp horizontal and diagonal edges, the vertical and diagonal pixels vary greatly. The predicted pixels are mainly affected by the horizontal and nearest diagonal pixels. When the pixels are on sharp vertical and diagonal edges, the horizontal and diagonal pixels vary greatly. The predicted pixels are mainly affected by the vertical and nearest diagonal pixels. When the pixels are on a non-sharp edge since the pixels from all directions do not change much, a prediction operator  $\theta$  is calculated by Eq. (3). Then the prediction value is fine-tuned according to the prediction operator and the pixel's position.

Non-sharp edges are divided into ordinary edges and weak edges according to the gradient strength around the pixel. When the pixels are on a non-sharp edge, the weight distribution of the scanning mode is shown in Fig. 5.

where the red and blue lines in Fig. 5 represent the pixels participating in the prediction when the pixels are on non-sharp horizontal and vertical edges, respectively. Pixel predictions on non-sharp

edges are calculated by Eq. (4).

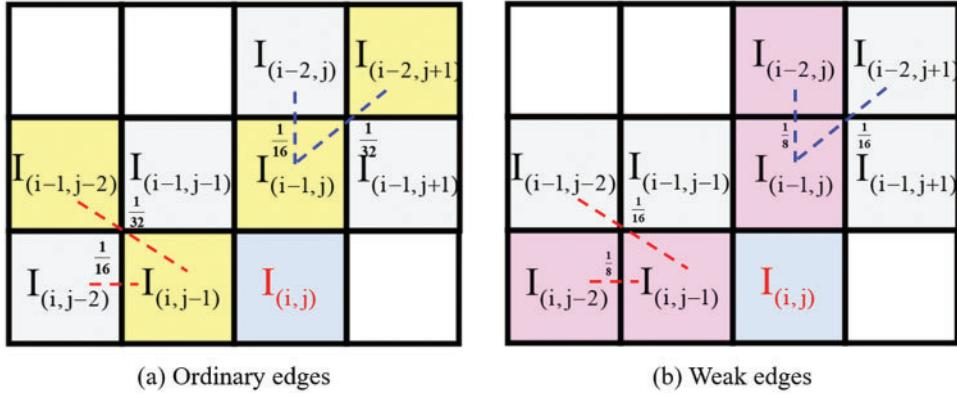
$$\theta = \left\lfloor \frac{I_{(i-1,j+1)} + 2I_{(i-1,j)} - I_{(i-2,j)}}{2} + \frac{2I_{(i,j-1)} - I_{(i,j-2)}}{8} \right\rfloor \quad (3)$$

$$I'_{(i,j)} = \begin{cases} \varepsilon t, (|G_v - G_h| > T_4) \\ \frac{\varepsilon_t + \theta}{2}, (|G_v - G_h| < T_4), \end{cases} \quad t = \begin{cases} 0 & G_v - G_h > T_3 \\ 1 & G_h - G_v > T_3 \end{cases}$$

$$\varepsilon_0 = \frac{\theta + I_{(i,j-1)}}{2} + \frac{I_{(i,j-1)} - I_{(i,j-2)}}{8} + \frac{I_{(i,j-1)} - I_{(i-1,j-2)}}{16}$$

$$\varepsilon_1 = \frac{\theta + I_{(i-1,j)}}{2} + \frac{I_{(i-1,j)} - I_{(i-2,j)}}{8} + \frac{I_{(i-1,j)} - I_{(i-2,j+1)}}{16} \quad (4)$$

where  $\varepsilon_0, \varepsilon_1$  represent the dominant values when predicting pixels at non-sharp horizontal and vertical edge locations.  $t$  is used to judge that the predicted value is mainly affected by the pixels in the vertical or horizontal direction.



**Figure 5:** Weight assignment of scan mode for ordinary edges and weak edges

The above prediction coefficients and thresholds are selected according to an exhaustive algorithm. A threshold selection experiment is set up in Section 3.1, which verifies that our scheme can achieve the best embedding rate under the selected threshold. After calculating the predicted value of the pixel, the prediction error of the pixel can be calculated by Eq. (5).

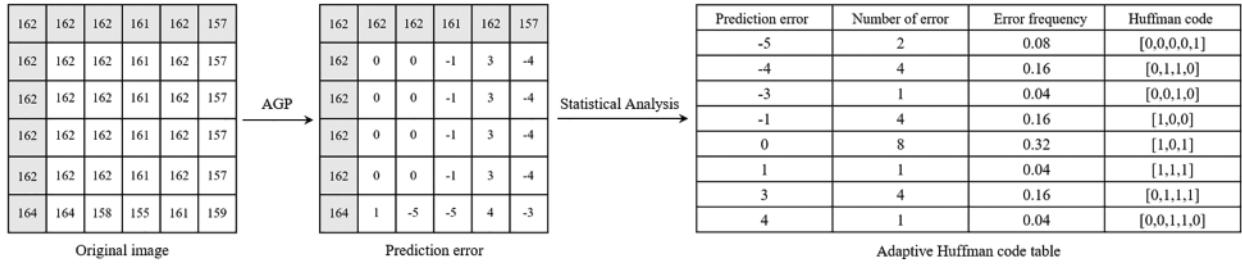
$$E_{(i,j)} = I_{(i,j)} - I'_{(i,j)} \quad (5)$$

### 2.1.2 Generation of Adaptive Huffman Codes

Our scheme adopts an adaptive huffman coding [26] for pixel marking to ensure that the scheme achieves a better coding effect. For different pictures, the scheme first performs statistical analysis on the prediction error and then adaptively generates huffman coding according to the probability of the error occurrence. The following will introduce the adaptive huffman coding marking process in detail with an example.

First, count the frequency of prediction errors. Since the length of the huffman code used for marking is no longer than 8 bits, the average frequency of each codeword is 1/256. This feature can be used as a threshold for dividing the frequency of prediction errors. Pixels with prediction error frequency within the average frequency range are classified as embeddable pixels, and those not within

the range are non-embeddable pixels. Therefore, the scheme divides the remaining pixels except for the reference pixels into two parts: embeddable pixels and non-embeddable pixels. Taking the  $6 \times 6$  block of the Lena image as an example, the adaptive huffman coding generation process shown in Fig. 6 is performed.



**Figure 6:** Adaptive huffman coding generation

The blocks selected in Fig. 6 do not contain non-embeddable pixels for the convenience of representation. The scheme uses the same code for marking for non-embeddable pixels whose error frequency is less than 1/256.

### 2.1.3 Image Encryption

After obtaining the prediction error of the original image  $I$  and huffman encoding, the original image is encrypted using a stream cipher. First, a pseudo-random matrix  $P$  of size  $m \times n$  is generated using the key. Then, convert  $I$  and  $P$  into binary form and encrypt them by bitwise XOR:

$$I_E^k(i,j) = I^k(i,j) \oplus P^k(i,j) \quad k = 1, 2, \dots, 8 \quad (6)$$

where  $1 \leq i \leq m$ ,  $1 \leq j \leq n$ .  $I_E^k(i,j)$ ,  $I^k(i,j)$ , and  $P^k(i,j)$  represent an encrypted image  $I_E$ , original image  $I$ , and the  $k^{th}$  bit of the  $(i,j)^{th}$  pixel in the error matrix  $P$ , respectively.

### 2.1.4 Labeling with Adaptive Huffman Codes

After adaptively generating the huffman code, the pixels can be labeled. All pixels are divided into reference pixels, embeddable pixels, and non-embeddable pixels. The reference pixel is not labeled, and its room is used to embed the huffman coding rule, that is, Correspondence between prediction error and huffman code. The original value of the reference pixel is stored in the reserved room as part of the auxiliary information. The bits of the non-embeddable pixels replaced by the huffman code are also stored as auxiliary information. The remaining room of the embeddable pixels is used to embed auxiliary information and secret information. Taking the  $4 \times 4$  block of the Lena image as an example, the pixel labeling process is as shown in Fig. 7.

To prevent the generated huffman code from being the same as the original bit of the labeled part of the pixel, resulting in the leakage of image information. As shown in Fig. 7c, the scheme adopts sorting in reverse order. This method ensures that the resulting marked encrypted image has a higher level of security. The part marked in red in Fig. 7d represents the labeled bit, and '-' represents the reserved room for data embedding.

00100001(33)	11100010(226)	10010010(146)	10101101(173)
11010111(215)	01111010(122)	10001110(142)	10011101(157)
11111011(251)	00101000(40)	01001001(73)	00101100(44)
10111010(186)	10100001(161)	00100100(36)	00010001(17)

(a) Encrypted image

00000010(2)	00000001(1)	00111110(62)	10100000(160)
11010111(215)	01011 <b>101</b> (93)	01110 <b>101</b> (117)	10111 <b>001</b> (185)
11111011(251)	00010 <b>101</b> (21)	10010 <b>101</b> (149)	00110 <b>001</b> (49)
10111010(186)	10000 <b>101</b> (133)	00100 <b>101</b> (37)	10001 <b>001</b> (137)

(b) Marked encrypted image with Huffman coding

00000010(2)	00000001(1)	00111110(62)	10100000(160)
11010111(215)	----- <b>101</b>	----- <b>101</b>	----- <b>001</b>
11111011(251)	----- <b>101</b>	----- <b>101</b>	----- <b>001</b>
10111010(186)	----- <b>101</b>	----- <b>101</b>	----- <b>001</b>

(c) Marked encrypted image

00000010(2)	00000001(1)	00111110(62)	10100000(160)
11010111(215)	----- <b>101</b>	----- <b>101</b>	----- <b>001</b>
11111011(251)	----- <b>101</b>	----- <b>101</b>	----- <b>001</b>
10111010(186)	----- <b>101</b>	----- <b>101</b>	----- <b>001</b>

(d) Marked stego image

**Figure 7:** Pixel labeling process

## 2.2 Information Embedding

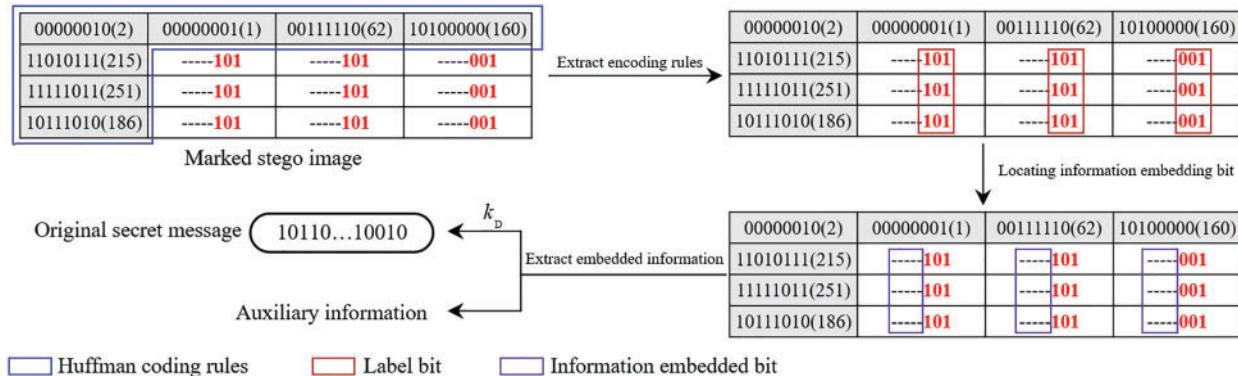
Before data hiding, the huffman coding rules are extracted from the reference pixels of  $\hat{I}_E$ , and then the position of the reserved room is determined. To ensure the security of the embedded information, the secret information needs to be encrypted by the information hiding key  $k_D$  before embedding. Finally, the information is embedded in the reserved room by bit replacement.

## 2.3 Information Extraction and Image Recovery

After receiving the marked stego image, the receiver implements different operations according to the key type obtained.

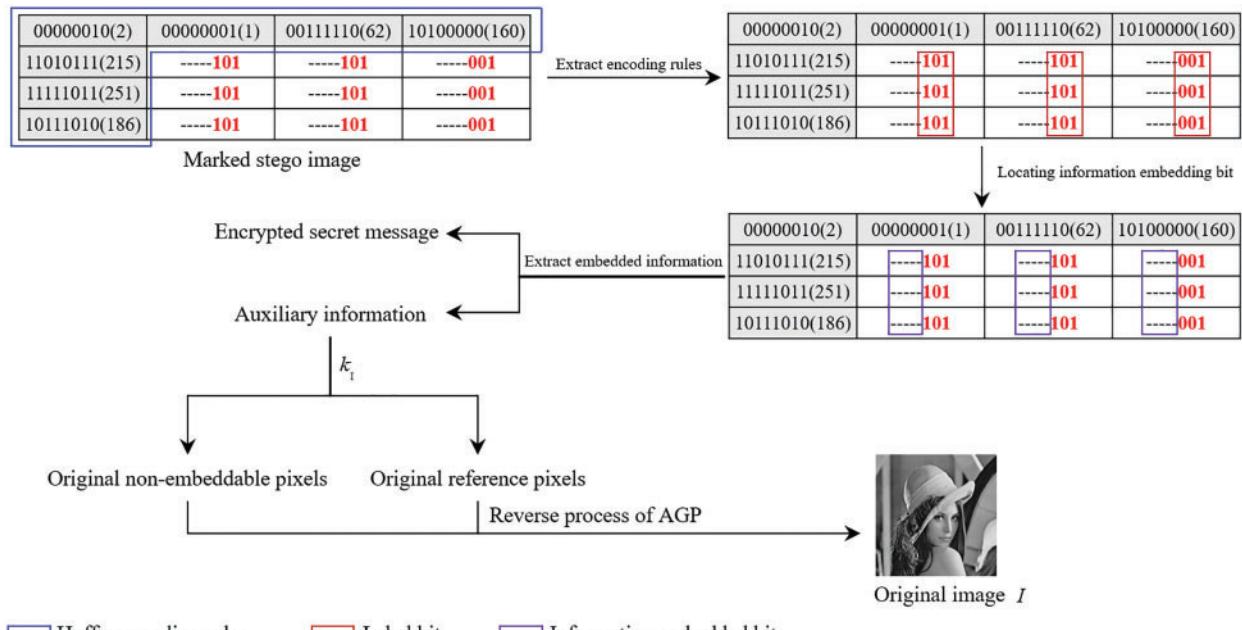
### 2.3.1 Information Extraction

The receiver who obtains the data hiding key  $k_D$  can extract the data: the receiver first extracts the coding rules from the reference pixels of  $\hat{I}_{E-D}$  and determines the position of the reserved room. Then, the embedded secret data and auxiliary information are extracted. The encrypted data can be decrypted to recover the original secret information according to the key  $k_D$ . Fig. 8 shows the process of extracting secret information.

**Figure 8:** The process of extracting secret information

### 2.3.2 Image Recovery

The receiver who obtains the image encryption key  $k_I$  can recover the original image: after extracting the auxiliary information, the encrypted reference pixels and the labeled bits of the non-embeddable pixels can be restored according to the auxiliary information. The original reference pixels and non-embeddable pixels can be recovered according to the key  $k_I$ . Then, the prediction error corresponding to the embeddable pixel can be determined according to the huffman coding rule. Finally, since the prediction process of AGP is reversible, the prediction value of the embedded pixel can be obtained from the reference pixel. The original embeddable pixels can be recovered by summing the prediction error and the predicted value. Fig. 9 shows the process of image recovery.



**Figure 9:** The process of image recovery

## 3 Experimental Results and Analysis

Experimental environment: MATLAB R2019a; Experimental setup: we test the effect of different thresholds on the performance (see in Section 3.1), the comparison and analysis with other schemes [22–25] (see in Section 3.2), the security analysis of our scheme (see in Section 3.3).

Metrics: the embedding rate (ER, expressed in bits per pixel (bpp)), the average embedding rate, Point Signal Noisy Ratio (PSNR), and Structural Similarity (SSIM).

### 3.1 Threshold Selection Experiment

From Section 2.1.1, we know that AGP needs to adjust the threshold according to the edge case where the pixel is located. Therefore, to verify the effectiveness of the threshold, we take the Lena image as an example and conduct the threshold selection experiments shown in Tab. 1.

It can be seen from Tab. 1 that when the predicted pixel is at a sharp edge, the embedding rate is optimal when  $T_1$  is 80 and  $T_2$  is 80. Similarly, when it is on a non-sharp edge, the scheme's performance is optimal when the threshold  $T_3$  is 8 and  $T_4$  is 32.

**Table 1:** ER under different threshold selection

$T_1$	ER	$T_2$	ER	$T_3$	ER	$T_4$	ER
70	3.4707	70	3.4717	<b>8</b>	<b>3.4718</b>	8	3.4698
<b>80</b>	<b>3.4718</b>	<b>80</b>	<b>3.4718</b>	16	3.4579	16	3.4661
90	3.4709	90	3.4718	32	3.4326	<b>32</b>	<b>3.4718</b>
100	3.4702	100	3.4718	64	3.4326	64	3.4504
110	3.4699	110	3.4718	128	3.4326	128	3.4432

### 3.2 Analysis of Embedding Performance

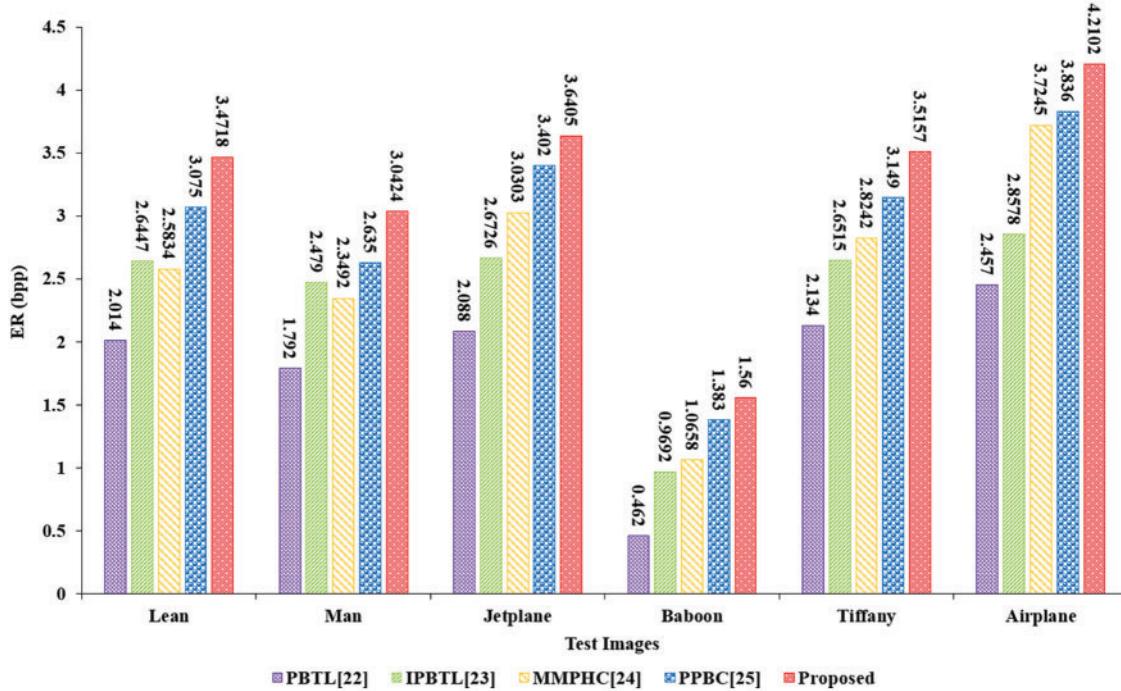
To verify the embedding performance of our scheme, we first select six standard grayscale images of Lena, Man, Jetplane, Baboon, Tiffany, and Airplane shown in Fig. 10 for embedding rate experiments. To further verify the generality of the scheme, we conduct average embedding rate experiments on UCID [27], BOSSbase [28], and BOWS-2 [29].



**Figure 10:** Six standard grayscale images

The comparison results of our scheme and different methods on six images are shown in Fig. 11. Our scheme outperforms other schemes [22–25]. Taking the Jetplane image as an example, the embedding rate of our algorithm is 1.5525, 0.9679, 0.6102, 0.2385 bpp higher than

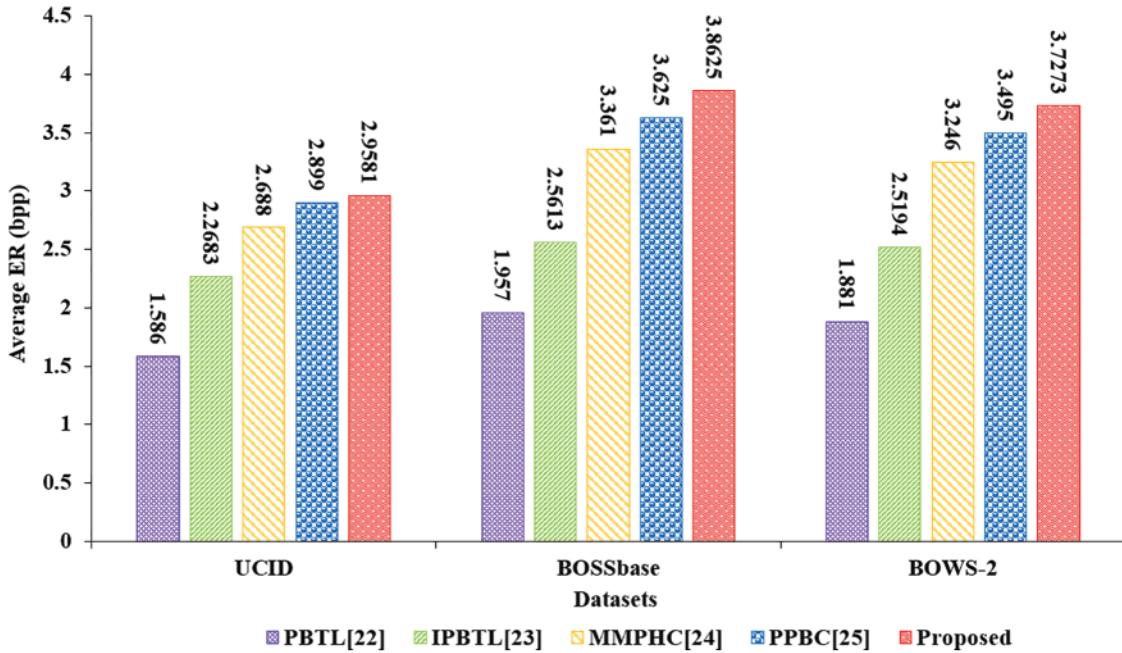
[22–25], respectively. Since Wu et al. [23] uses MED with a smaller receptive field for prediction, the perception ability of complex images is insufficient, thus affecting the overall embedding performance of the scheme. Yin et al. [24] use the predefined huffman coding, but for different images, the unified huffman coding is not the best coding rule, which leads to a decrease in the embedding rate. Our algorithm utilizes the pixel correlation of the whole image. It improves the prediction accuracy of the predictor for pixels in the texture area of the image through four scanning methods. At the same time, adaptive huffman coding is used to generate huffman codewords for different images adaptively, so the embedding rate is higher.



**Figure 11:** ER (bpp) comparison of different methods on six test images

In Fig. 12, we compare the average embedding rate of our scheme with different methods on three datasets. The average embedding rate of our scheme is higher than [22–25]. Specifically, our scheme achieves an average embedding rate of 3.5160 bpp on three datasets. While the literatures [22–25] are 1.808, 2.4497, 3.0983 and 3.3397 bpp respectively, we have an increase of 94.5%, 43.5%, 13.5% and 5.3%.

To further verify the advantages of our scheme, we test the embedding rate (bpp), PSNR (dB), and SSIM of different schemes in three datasets, results are shown in Tab. 2. Since scheme [22] is older and scheme [23] is an improvement of the scheme [22], we only compare this scheme with schemes [23–25] in detail. Where  $\text{PSNR} = +\infty$ ,  $\text{SSIM} = 1$  means that the recovered image has the same structure as the original image, and our scheme can recover the original image losslessly. Since AGP can combine the edge features of the predicted pixels and achieve better prediction results, the worst embedding rates on the three datasets are significantly improved. The scheme has better embedding performance on complex images.



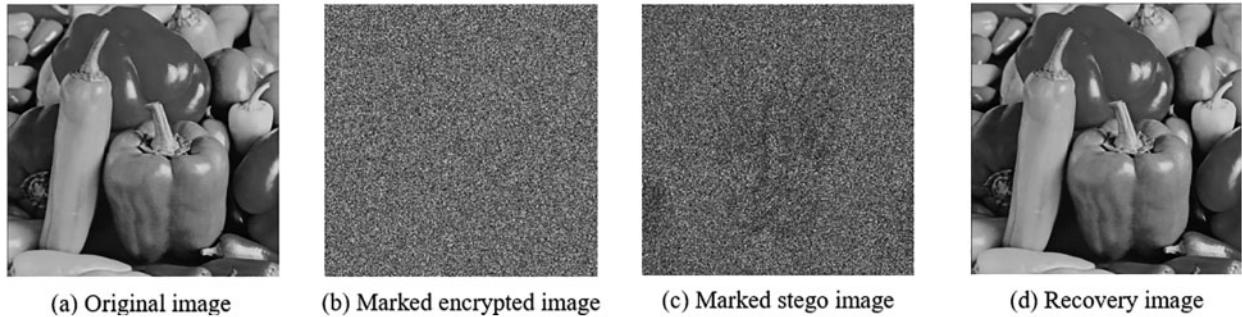
**Figure 12:** Average ER (bpp) comparison of different methods on three datasets

**Table 2:** Test results on three datasets

Datasets	Index	Best case				Worst case				Average			
		[23]	[24]	[25]	Proposed	[23]	[24]	[25]	Proposed	[23]	[24]	[25]	Proposed
UCID	ER	2.9759	5.010	5.2344	<b>6.4267</b>	0	0.397	0.2563	<b>0.8451</b>	2.2683	2.688	2.8990	<b>2.9581</b>
	PSNR	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞
	SSIM	1	1	1	1	1	1	1	1	1	1	1	1
BOSS base	ER	2.9883	5.898	7.8305	6.7850	0.0713	0.674	0.7416	<b>0.9588</b>	2.5613	3.361	3.6248	<b>3.8625</b>
	PSNR	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞
	SSIM	1	1	1	1	1	1	1	1	1	1	1	1
BOWS-2	ER	2.9883	5.622	7.0190	6.4080	0.0484	0.628	0.6239	<b>0.8350</b>	2.5194	3.246	3.4948	<b>3.7273</b>
	PSNR	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞	+∞
	SSIM	1	1	1	1	1	1	1	1	1	1	1	1

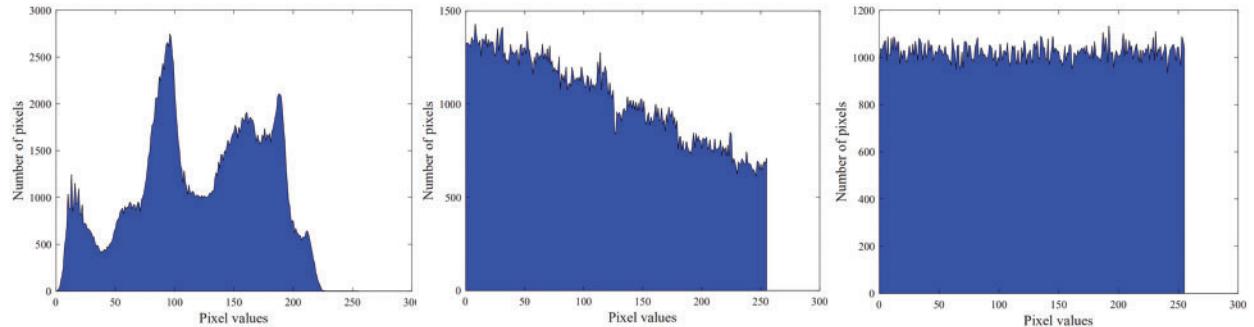
### 3.3 Safety Performance Analysis

This section first shows the stage image of the original image in the process of reversible data hiding, then compares the distribution of image histogram before and after encryption, and finally gives the results of common security indicators to illustrate the algorithm's security. Fig. 13 shows the results of Peppers after encrypting, embedding data, and recovering.



**Figure 13:** The results of peppers after encrypting, embedding data, and recovering

The histogram distribution of each stage of Peppers is shown in Fig. 14. The histogram containing the original image information is displayed in Fig. 14a. The histogram after stream cipher encryption is shown in Fig. 14b. The pixels are uniformly distributed, and the feature information of the image cannot be obtained. The histogram after embedding the data is shown in Fig. 14c. The content of the original image cannot be obtained from the encrypted image. The above experiments verify that the algorithm has good security performance and can resist standard cryptographic attack methods, such as statistical analysis attacks.



**Figure 14:** Histogram distribution of each stage of peppers

To further verify the algorithm's security, Tabs. 3 and 4 shows the PSNR and SIM of the scheme on six standard grayscale images.

**Table 3:** PSNR and SSIM of original and encrypted images

Encrypted image	Lena	Man	Jetplane	Baboon	Tiffany	Airplane
PSNR	9.2255	7.9937	8.0077	9.5108	6.8839	8.9521
SSIM	0.0341	0.0681	0.0346	0.0299	0.0389	0.0403

As shown in Tabs. 3 and 4, the PSNR of both the encrypted and marked encrypted images are very low, and their SSIM are almost 0. Therefore, the information of the original image cannot be obtained from the encrypted version image, which means that the proposed method safely protects the privacy of the original image and can be applied to RDH in the encrypted domain.

**Table 4:** PSNR and SSIM of original and marked encrypted images

Marked encrypted image	Lena	Man	Jetplane	Baboon	Tiffany	Airplane
PSNR	9.7857	8.9257	6.9484	9.3547	6.2359	8.3739
SSIM	0.0507	0.1183	-0.0090	0.0113	0.0554	0.0584

#### 4 Conclusion

In this paper, we propose an RDHEI algorithm based on adaptive gradient prediction (AGP) and code labeling. An adaptive gradient predictor is proposed to aim at the multi-directional requirement of texture attributes in natural images. AGP can adaptively adjust the weights of the four scanning methods to predict the target pixel according to the texture features around the pixel. This method combines the advantages of AGP and adaptive huffman coding and reserves more space for data embedding. Compared with existing RDHEI schemes, our scheme provides a higher embedding rate while ensuring that the receiver can accurately extract the secret information and recover the original image without image quality degradation.

**Funding Statement:** This work was supported in part by the Natural Science Foundation of Hunan Province (No. 2020JJ4141), author X. X, <http://kjt.hunan.gov.cn/>; in part by the Postgraduate Excellent teaching team Project of Hunan Province under Grant (No. ZJWKT202204), author J. Q, <http://zfsg.gd.gov.cn/xxfb/ywsd/index.html>.

**Conflicts of Interest:** The authors declare that they have no conflicts of interest to report regarding the present study.

#### References

- [1] S. Zhao, M. Hu, Z. Cai, Z. Zhang, T. Zhou *et al.*, “Enhancing Chinese character representation with lattice-aligned attention,” *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–10, 2021, <https://doi.org/10.1109/TNNLS.2021.3114378>.
- [2] Q. Liu, X. Xiang, J. Qin, Y. Tan and Y. Luo, “Coverless steganography based on image retrieval of DenseNet features and DWT sequence mapping,” *Knowledge-Based Systems*, vol. 192, no. 1, pp. 105375–105389, 2020.
- [3] C. L. Wang, Y. L. Liu, Y. J. Tong and J. W. Wang, “GAN-GLS: Generative lyric steganography based on generative adversarial networks,” *Computers, Materials & Continua*, vol. 69, no. 1, pp. 1375–1390, 2021.
- [4] Y. J. Luo, J. H. Qin, X. Y. Xiang and Y. Tan, “Coverless image steganography based on multi-object recognition,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 7, pp. 2779–2791, 2021.
- [5] X. R. Zhang, J. Zhou, W. Sun and S. K. Jha, “A lightweight CNN based on transfer learning for COVID-19 diagnosis,” *Computers, Materials & Continua*, vol. 72, no. 1, pp. 1123–1137, 2022.
- [6] X. R. Zhang, X. Sun, W. Sun, T. Xu and P. P. Wang, “Deformation expression of soft tissue based on BP neural network,” *Intelligent Automation & Soft Computing*, vol. 32, no. 2, pp. 1041–1053, 2022.
- [7] J. Tian, “Reversible data embedding using a difference expansion,” *IEEE Transactions on Circuits & Systems for Video Technology*, vol. 13, no. 8, pp. 890–896, 2003.
- [8] Y. Ren, J. Qin, Y. Tan and N. N. Xiong, “Reversible data hiding in encrypted images based on adaptive prediction-error label map,” *Intelligent Automation & Soft Computing*, vol. 33, no. 3, pp. 1439–1453, 2022.
- [9] Z. Ni, Y. Shi, N. Ansari and W. Su, “Reversible data hiding,” *IEEE Transactions on Circuits & Systems for Video Technology*, vol. 16, no. 3, pp. 354–362, 2006.

- [10] X. Li, W. Zhang, X. Gui and B. Yang, "Efficient reversible data hiding based on multiple histograms modification," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 9, pp. 2016–2027, 2017.
- [11] J. Fridrich, M. Goljan and R. Du, "Lossless data embedding—New paradigm in digital watermarking," *Eurasip Journal on Advances in Signal Processing*, vol. 2002, no. 2, pp. 185–196, 2002.
- [12] M. Celik, G. Sharma, A. Tekalp and E. Saber, "Lossless generalized-LSB data embedding," *IEEE Transactions on Image Processing*, vol. 12, no. 2, pp. 253–266, 2005.
- [13] Y. Luo, J. Qin, X. Xiang, Y. Tan, Z. He *et al.*, "Coverless image steganography based on image segmentation," *Computers, Materials & Continua*, vol. 64, no. 2, pp. 1281–1295, 2020.
- [14] S. Zhao, M. Hu, Z. Cai and Z. F. Liu, "Dynamic modeling cross-modal interactions in two-phase prediction for entity-relation extraction," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–10, 2021, <https://doi.org/10.1109/TNNLS.2021.3104971>.
- [15] W. Puech, M. Chaumont and O. Strauss, "A reversible data hiding method for encrypted images," in *Proc. SPIE security, forensics, steganography, and watermarking of multimedia*, San Jose, CA, USA, pp. 1–9, 2008.
- [16] X. Zhang, "Reversible data hiding in encrypted image," *IEEE Signal Processing Letters*, vol. 18, no. 4, pp. 255–258, 2011.
- [17] X. Zhang, "Separable reversible data hiding in encrypted image," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 2, pp. 826–832, 2011.
- [18] W. Hong, T. Chen and H. Wu, "An improved reversible data hiding in encrypted images using side match," *IEEE Signal Processing Letters*, vol. 19, no. 4, pp. 199–202, 2012.
- [19] S. Xiang and X. Luo, "Reversible data hiding in homomorphic encrypted domain by mirroring ciphertext group," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 11, pp. 3099–3110, 2018.
- [20] P. Puteaux and W. Puech, "An efficient MSB prediction-based method for high-capacity reversible data hiding in encrypted images," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 7, pp. 1670–1681, 2018.
- [21] K. Chen and C. Chang, "High-capacity reversible data hiding in encrypted images based on extended run-length coding and block-based MSB plane rearrangement," *Journal of Visual Communication and Image Representation*, vol. 58, pp. 334–344, 2018.
- [22] S. Yi and Y. Zhou, "Separable and reversible data hiding in encrypted images using parametric binary tree labeling," *IEEE Transactions on Multimedia*, vol. 21, no. 1, pp. 51–64, 2019.
- [23] Y. Wu, Y. Xiang, Y. Guo, J. Tang and Z. Yin, "An improved reversible data hiding in encrypted images using parametric binary tree labeling," *IEEE Transactions on Multimedia*, vol. 22, no. 8, pp. 1929–1938, 2020.
- [24] Z. Yin, Y. Xiang and X. Zhang, "Reversible data hiding in encrypted images based on multi-MSB prediction and huffman coding," *IEEE Transactions on Multimedia*, vol. 22, no. 4, pp. 874–884, 2020.
- [25] Z. Yin, Y. Peng and Y. Xiang, "Reversible data hiding in encrypted images based on pixel prediction and bit-plane compression," *IEEE Transactions on Dependable and Secure Computing*, vol. 19, no. 2, pp. 992–1002, 2022.
- [26] Y. Wu, Y. Guo, J. Tang, B. Luo and Z. Yin, "Reversible data hiding in encrypted images using adaptive huffman encoding strategy," *Chinese Journal of Computers*, vol. 40, no. 4, pp. 846–858, 2021.
- [27] G. Schaefer and M. Stich, "UCID: An uncompressed color image database," in *Proc. SPIE electronic imaging, storage and retrieval methods and applications for multimedia*, San Jose, CA, USA, vol. 5307, pp. 472–480, 2003. Available: <http://vision.doc.ntu.ac.uk/>.
- [28] P. Bas, T. Filler and T. Pevny, "Break our steganographic system: The ins and outs of organizing BOSS," in *Proc. 13th Int. Conf. on Information Hiding*, Prague, CZE, pp. 59–70, 2011. Available: <http://dde.binghamton.edu/download/>.
- [29] P. Bas and T. Furon, "Image database of bows-2," Accessed: Jun. 20, 2017. [Online]. Available: <http://bows2.ec-lille.fr/>.