Hindawi
Security and Communication Networks
Volume 2022, Article ID 8757594, 12 pages
https://doi.org/10.1155/2022/8757594

Research Article

Reversible Data Hiding in Encrypted Images Based on Adaptive Gradient Prediction

Jiaohua Qin, Zhibin He, Xuyu Xiang, and Yun Tan

School of Computer and Information Engineering, Central South University of Forestry and Technology, Changsha 410004, China

Correspondence should be addressed to Jiaohua Qin; qinjiaohua@163.com

Received 11 January 2022; Revised 24 February 2022; Accepted 2 May 2022; Published 24 May 2022

Academic Editor: Andrea Michienzi

The existing reversible data hiding (RDH) technology solves the problem that the ciphertext data and carrier are easily damaged in the traditional scheme, which has become a research hotspot in information hiding. However, because of the small coverage area of the predictor and poor prediction ability of the existing reversible data hiding in encrypted images (RDH-EI) technology based on pixel predictor, the embedding performance of the algorithm is limited. Therefore, this paper proposes an adaptive gradient prediction (AGP) scheme. The AGP employs a comprehensive and efficient local complexity measurement strategy to make predictions based on pixel changes around the predicted pixel, including horizontal, vertical, and diagonal directions. The experimental results show that the AGP-RDHEI scheme has apparent advantages in embedding rate.

1. Introduction

The development of cloud computing technology has promoted the continuous sharing of massive media data [1], providing users with convenient and low-cost computing and storage services. However, in recent years, personal privacy leaks have occurred many times, resulting in the leakage of a large amount of personal information and resources, and have become the focus of continuous public attention [2]. Therefore, technologies such as digital watermarking and steganography [3, 4] have been proposed successively. Steganography focuses on the imperceptibility of hidden data and embedded data, but the carrier will be tampered. Digital watermarking technology possesses strong robustness, which ignores the integrity of the carrier. However, it is necessary to ensure the carrier's integrity while extracting data losslessly for some specific fields.

Reversible data hiding, different from the above two algorithms, can accurately restore the cover image after extracting the embedded data, and is widely used in military and medical image data hiding [5]. Existing RDH methods can be roughly classified into three parts: lossless compression-based RDH [6, 7], histogram translation-based RDH [8, 9], and differential expansion-based RDH [10, 11]. However, with the continuous improvement of users' requirements for data security, the traditional reversible data hiding technology has been unable to meet the needs of practical applications. Therefore, the researchers put forward to combine the encryption algorithm with the reversible data hiding algorithm, further ensuring the data security during the transmission process, and realizing the reversible data hiding of encrypted images.

The existing RDH-EI technologies are mainly divided into two parts. One is the vacating room after encryption (VRAE) [12, 13], which uses an encryption algorithm to encrypt the original image and then frees up room for secret data embedding. The other is reserving room before encryption (RRBE) [14, 15], which utilizes the spatial correlation of the original image to reserve room, then encrypts the image, and performs secret data embedding. However, traditional RDH-EI needs to synchronize the image restoration and data extraction process at the receiver [15], which requires the recipient to establish a high degree of trust with the content owner. To ensure that the receiver can manipulate the secret data without revealing the content, Zhang [16] first proposed separable RDH-EI, which can perform different operations by sending different keys, so that the receiver's operations are subject to the type of key obtained.

To meet different security requirements, many improved RDH-EI schemes have been proposed [13, 17–19]. Yi and Zhou [13] designed a RDH algorithm based on parametric binary tree labeling (PBTL), making full use of the spatial redundancy of the image, and solved the problem of low embedding rate (ER). Wu et al. [17] proposed an improved parametric binary tree labeling (IPBTL) RDH-EI method and achieved a higher ER than [13], but has low utilization of spatial redundancy. Meanwhile, Yin et al. proposed an RDH encryption image algorithm based on pixel prediction and multi-MSB plane rearrangement [18], which solved the problem of large space for auxiliary information through arithmetic coding and data compression, and introduced the medium edge detector (MED) that improves ER. Subsequently, Yang et al. [19] introduced block scrambling in the encryption algorithm and adaptive coding in the embedding process to improve the compression rate and embedding ability. However, both methods use MED. But MED makes predictions based on three pixels around the predicted pixel: the top pixel, the left pixel, and the top-left pixel. Since the number of pixels involved in the prediction is small, the coverage of the predictor is small, it cannot perceive images with complex textures well, resulting in low prediction accuracy. Consequently, Wu et al. [20] proposed the gradient-adjusted prediction (GAP), which only considers the horizontal and vertical gradients of the pixels in the neighborhood, and the prediction accuracy is still poor for complex images.

To fully consider the characteristics of pixel changes in different images, this paper proposes a pixel prediction scheme based on adaptive gradient based on [17, 20, 21]. When measuring complexity, we add the change in pixels in the diagonal direction to the calculation. At the same time, we combine the AGP with the PBTL scheme to free up more space in the image for secret information embedding. Compared with [17], this method takes the edge information of each image into consideration, achieving more accurate pixel prediction and higher embedding capacity.

The rest of the paper is structured as follows. Section 2 reviews some related research. Section 3 describes the RDH-EI method based on AGP. Section 4 presents the experimental results and analysis. Finally, Section 5 summarizes the paper and proposes future work.

2. Related Work

2.1. Gradient-Adjusted Prediction (GAP). The GAP, proposed by Wu and Memon [20], mainly utilizes the gradient change between adjacent pixels to estimate pixel values. Specifically, it makes predictions through the seven neighborhoods around the current pixel of the cover image, as shown in Figure 1.

2.2. Parameter Binary Tree Labeling (PBTL). The parametric binary tree labeling scheme mainly employs the binary sequence of 1 to 7 bits on the parametric binary tree to label the pixels of different categories. The construction rule of a 7-level binary tree is left 0 right 1. Giving parameters α and β,

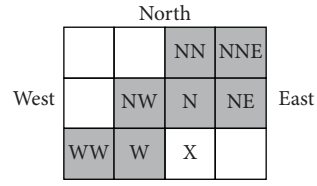


Figure 1: Predictive template used by GAP.

pixels can be divided into embeddable pixel class G_1 and non-embeddable pixel class G_2, where $1 \le \alpha, \beta \le 7$. G_1 consists of embeddable pixels P_e and G_2 includes non-embeddable pixels P_n. The overall algorithm steps can be described as follows:

Step 1: build a 7-layer full-parameter binary tree

Step 2: after determining the parameters α and β, the label sequence of the pixels can be determined according to the parameters and the binary tree.

Step 3: for G_2, all P_n are labeled by β bits of "0, . . ., 0," which is the first node at the β_{th} layer of the binary tree.

Step 4: for G_1, all pixels can be divided into n_α different sub-categories due to the different prediction errors of the pixels. The calculation of n_α is as follows:

$$n_\alpha = \begin{cases} 2^\alpha - 1, & \text{if } \alpha \le \beta, \\ \left(2^\beta - 1\right) * 2^{\alpha - \beta}, & \text{otherwise.} \end{cases} \quad (1)$$

Step 5: after determining the value of n_α, use n_α different nodes from the α_{th} layer of the binary tree to label pixels in G_1 from right to left.

3. Proposed Scheme

The RDH-EI method's general framework using AGP is shown in Figure 2, which can be split into three stages. In the first stage, the content owner measures the local complexity of the predicted pixel and adaptively adjusts the prediction model to calculate the prediction error of the image. Then, the image is encrypted according to the method of stream cipher and the pixels are classified and labeled according to the PBTL scheme to obtain the labeled encrypted image in the second stage. Finally, in the third stage, the image receiver can perform data extraction or image recovery after receiving the stego image. We will introduce each component in the following subsections.

3.1. Pixel Prediction Based on AGP. Since the predicted pixels are mainly predicted according to their adjacent pixels in the process of pixel prediction, the change degree of nearby pixels and the prediction model of the predictor become the key factors affecting the prediction accuracy. However, the existing schemes ignore the pixel changes in the diagonal direction when measuring the local complexity of pixels, causing low prediction accuracy for complex images. Therefore, an adaptive gradient predictor is proposed in this section, and the complexity measurement scheme and prediction model are detailed below.

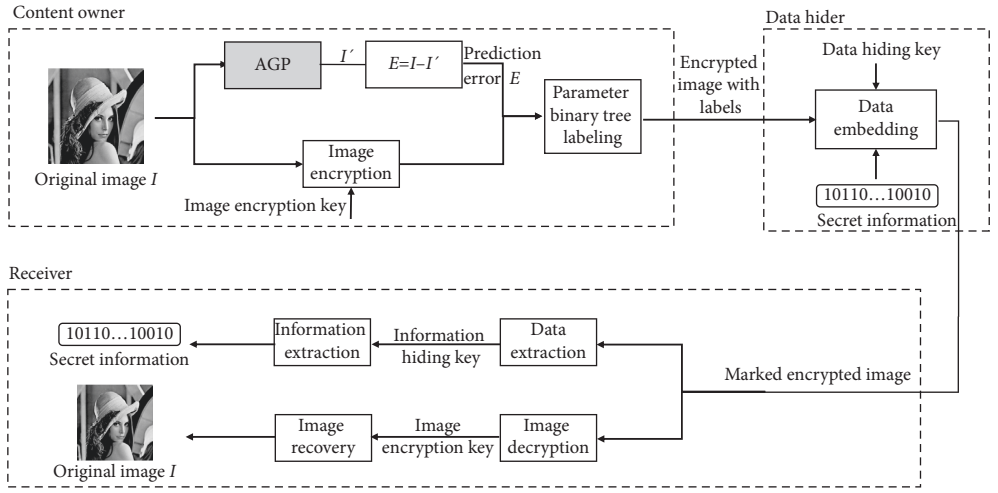


Figure 2: Framework of the proposed scheme.

3.1.1. Local Complexity Measurement. To fully estimate the nearby changes of pixels, this scheme optimizes the GAP. Specifically, we consider the pixel changes in the diagonal direction when calculating the local complexity, and introduce diagonal and anti-diagonal gradients when measuring local complexity, as shown in Figure 3.

The red and blue arrows represent the diagonal and the anti-diagonal direction. The gradient of intensity at the current pixel $I_{(i,j)}$ can be formulated as:

$$\begin{aligned} g_x &= \left|I_{(i,j-1)} - I_{(i,j-2)}\right| + \left|I_{(i-1,j)} - I_{(i-1,j-1)}\right| \\ &\quad + \left|I_{(i-1,j)} - I_{(i-1,j+1)}\right| + \left|I_{(i-1,j-1)} - I_{(i-1,j-2)}\right|, \\ g_y &= \left|I_{(i,j-1)} - I_{(i-1,j-1)}\right| + \left|I_{(i-1,j)} - I_{(i-2,j)}\right| \\ &\quad + \left|I_{(i-1,j+1)} - I_{(i-2,j+1)}\right| + \left|I_{(i,j-2)} - I_{(i-1,j-2)}\right|, \\ g'_x &= \left|I_{(i-1,j-2)} - I_{(i,j-1)}\right| + \left|I_{(i-2,j)} - I_{(i-1,j+1)}\right|, \\ g'_y &= \left|I_{(i-2,j+1)} - I_{(i-1,j)}\right| + \left|I_{(i-1,j-1)} - I_{(i,j-2)}\right|, \end{aligned} \tag{2}$$

where g_x, g_y, g'_x and g'_y represent the estimated gradient of the predicted pixel in the horizontal, vertical, diagonal, and anti-diagonal directions, respectively. The value of g_x, g_y, g'_x and g'_y are used to detect the magnitude and orientation of edges in the cover image, and make necessary adjustments in the prediction to obtain better prediction results under the local edges of the image.

The local complexity D_1, D_2 of the predicted pixel is formulated as:

$$\begin{aligned} D_1 &= g_x - g_{y'}, \\ D_2 &= g'_x - g'_y, \end{aligned} \tag{3}$$

where D_1 denotes the gradient complexity in the horizontal and vertical directions, D_2 is the gradient complexity in the diagonal and anti-diagonal directions. The value of D_1 and D_2 are used to detect the complexity around the pixel.

According to the local complexity measurement method above, we divide the situation around the predicted pixel

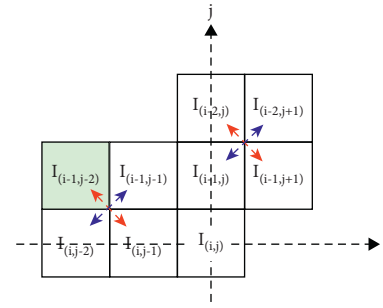


Figure 3: Diagonal gradient and anti-angle gradient.

into sharp and non-sharp edges. When the surrounding pixel value of the predicted pixel varies greatly, the value of the predicted pixel can be calculated by:

$$I'_{(i,j)} = \begin{cases} I_{(i,j-1)} + \lfloor \frac{h_1}{2} + \frac{h_2}{8} + \frac{d_1}{8} + \frac{d_2}{8}, & D_1 > T_1 \text{ and } D_2 > T_4, \\ I_{(i,j-1)} + \lfloor \frac{h_1}{2} + \frac{h_2}{8} + \frac{d_1}{64} + \frac{d_2}{64}, & D_1 > T_1 \text{ and } D_2 \le T_4, \\ I_{(i-1,j)} + \lfloor \frac{v_2}{2} + \frac{v_3}{8} + \frac{a_1}{8} + \frac{a_2}{16}, & D_1 < -T_1 \text{ and } D_2 < -T_4, \\ I_{(i-1,j)} + \lfloor \frac{v_2}{2} + \frac{v_3}{8} + \frac{d_1}{64} + \frac{d_2}{64}, & D_1 < -T_1 \text{ and } D_2 \ge -T_4. \end{cases} \tag{4}$$

Otherwise, if the position of the predicted pixel is a non-sharp edge, the predicted value is mainly affected by the prediction operator θ, which is defined as follows:

$$\begin{aligned} \theta = \lfloor & \frac{\left(I_{(i-1,j)} + I_{(i,j-1)}\right)}{2} + \frac{\left(I_{(i-1,j+1)} - I_{(i-1,j-1)}\right)}{4} \\ & + \frac{d_1}{32} + \frac{a_1}{8} + \frac{a_2}{8} + \frac{h_3}{8} + \frac{v_1}{8}. \end{aligned} \tag{5}$$

The value of θ is used to fine-tune the predicted value. The predicted value is calculated as follows:

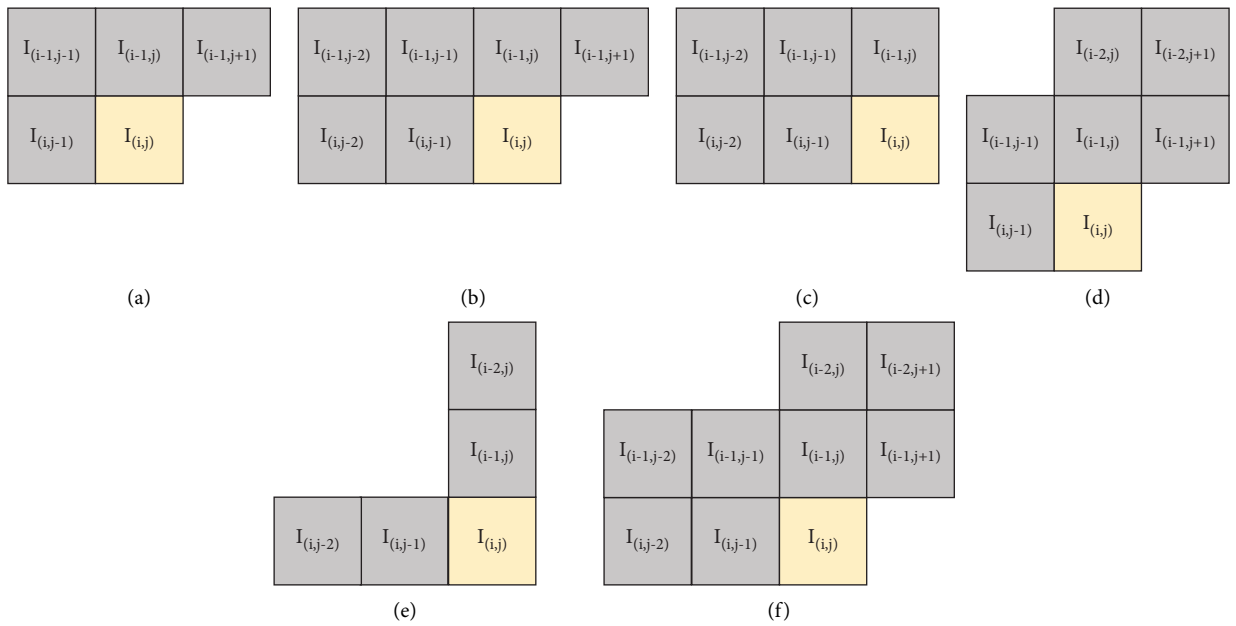


FIGURE 4: Prediction model of AGP in different locations. (a) $i = 2, j = 2$. (b) $i = 2, 2 < j <$ col. (c) $i = 2, j =$ col. (d) $2 < i <$ row, $j = 2$. (e) $i =$ row, $j =$ col. (f) Others.

$$I'_{(i,j)} = \begin{cases} \left\lfloor \dfrac{\left(\theta + I_{(i,j-1)}\right)}{2} + \dfrac{h_2}{8} + \dfrac{a_1}{32} + \dfrac{d_1}{16}, & D_1 > T_2, \\ \left\lfloor \dfrac{\left(3 * \theta + I_{(i,j-1)}\right)}{4} + \dfrac{h_2}{16} + \dfrac{a_1}{32} + \dfrac{d_1}{64}, & D_1 > T_3, \\ \left\lfloor \dfrac{\left(\theta + I_{(i-1,j)}\right)}{2} + \dfrac{h_2}{8} + \dfrac{a_2}{8} + \dfrac{d_2}{8}, & D_1 < -T_2, \\ \dfrac{\left(3 * \theta + I_{(i-1,j)}\right)}{4} + \dfrac{h_2}{8} + \dfrac{a_2}{8} + \dfrac{d_2}{8}, & D_1 < -T_3, \end{cases} \tag{6}$$

where T_1, T_2, T_3 denotes the thresholds used to determine the edge of the pixel. The value of T_1, T_2 and T_3 are used to judge horizontal and vertical edges. T_4 is adopted to judge diagonal edges. The strategy for threshold selection is described experimentally in Section 4.3.

For convenience, this part uses h_k, $v_k (k = 1, 2, 3, 4)$ to represent the horizontal and vertical gradients, and d_r, $a_r (r = 1, 2)$ to represent the diagonal and anti-diagonal gradient, which can be formulated respectively as:

$$\begin{aligned} h_1 &= I_{(i,j-1)} - I_{(i,j-2)} v_1 = I_{(i,j-1)} - I_{(i-1,j-1)}, \\ h_2 &= I_{(i-1,j)} - I_{(i-1,j-1)} v_2 = I_{(i-1,j)} - I_{(i-2,j)}, \\ h_3 &= I_{(i-1,j)} - I_{(i-1,j+1)} v_3 = I_{(i-1,j+1)} - I_{(i-2,j+1)}, \\ h_4 &= I_{(i-1,j-1)} - I_{(i-1,j-2)} v_4 = I_{(i,j-2)} - I_{(i-1,j-2)}, \\ d_1 &= I_{(i-1,j-2)} - I_{(i,j-1)} a_1 = I_{(i-2,j+1)} - I_{(i-1,j)}, \\ d_2 &= I_{(i-2,j)} - I_{(i-1,j+1)} a_2 = I_{(i-1,j-1)} - I_{(i,j-2)}. \end{aligned} \tag{7}$$

3.1.2. Adaptive Gradient Prediction Model. Most of the existing pixel prediction schemes can improve the performance of simple images, but its performance cannot be well developed for complex images with complex content and rich texture. The AGP can be shown in Figure 4. In the pixel prediction stage, the pixels in the first row and the first column of the image, as reference pixels, remain unchanged, and are used to predict the remaining pixels of the image in the image restoration stage. Since the position of the predicted pixel is constantly changing, AGP will adaptively adjust the prediction model according to the position of the pixel.

After the predicted image is calculated from the cover image by the AGP algorithm, the prediction error matrix elements are calculated by subtracting the predicted image from the cover image. The prediction error $E_{(i,j)}$ is defined as:

$$E_{(i,j)} = I_{(i,j)} - I'_{(i,j)}, \tag{8}$$

where $I_{(i,j)}$ and (i, j) is the original pixel value of the cover image and the predicted pixel value of the predicted image, respectively.

3.2. Image Encryption Based on Stream Cipher. In the stage of image encryption, we use stream cipher to encrypt the image and secret data, which can be formulated as follows:

$$I_E(i, j)_k = \left\lfloor \frac{I(i, j)}{2^{k-1}} \right\rfloor \oplus r(i, j)_k, \quad k \in \{1, 2, \ldots, 8\}, \tag{9}$$

where $\lfloor * \rfloor$ denotes round down, $I_E(i, j)_k$ denotes the result of image pixel $I(i, j)$ encryption, $\oplus$ is the exclusive OR operation, and $r(i, j)_k$ is the bitstream generated by the pseudo-random matrix $R_{m \times n}$. The content owner uses the

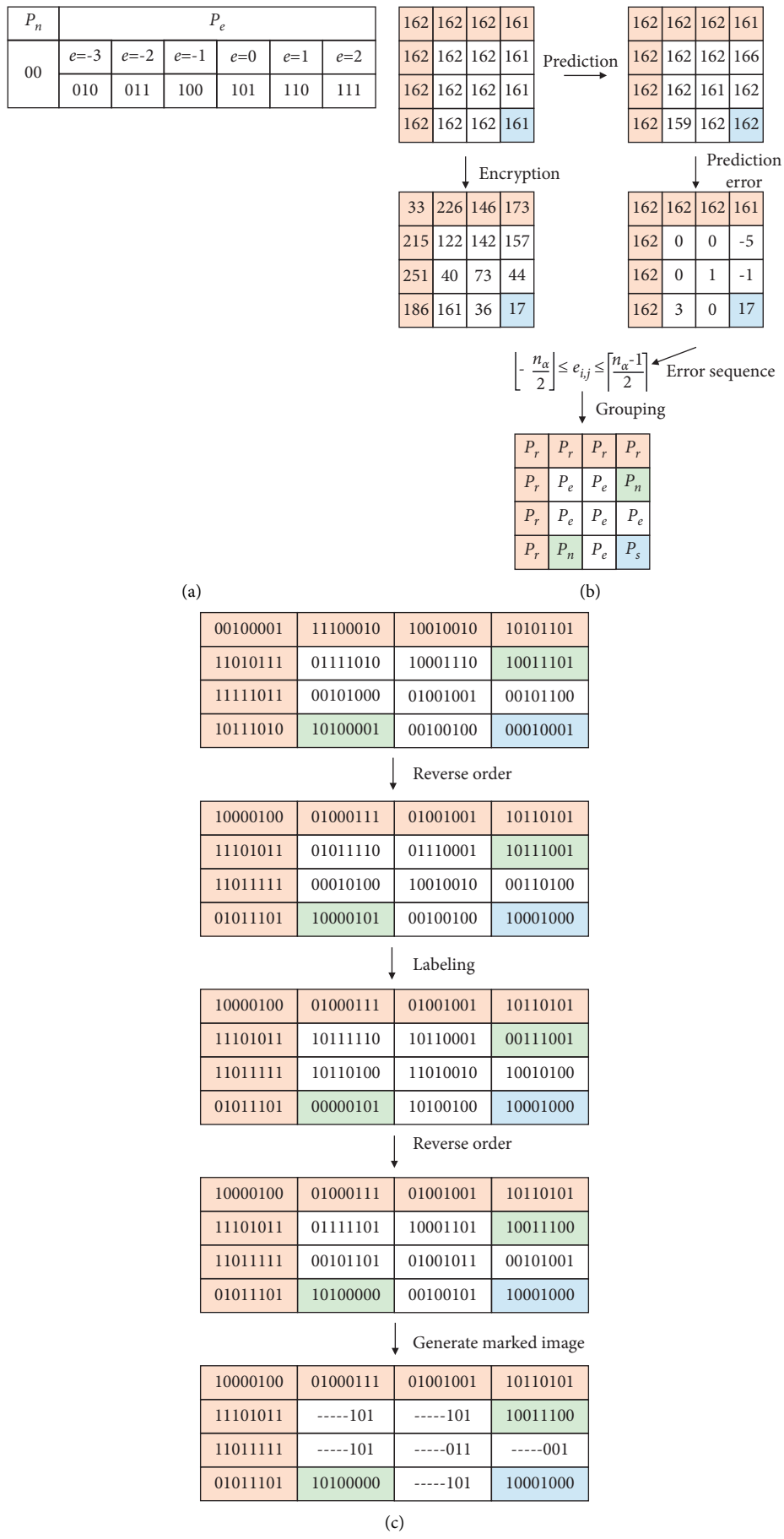
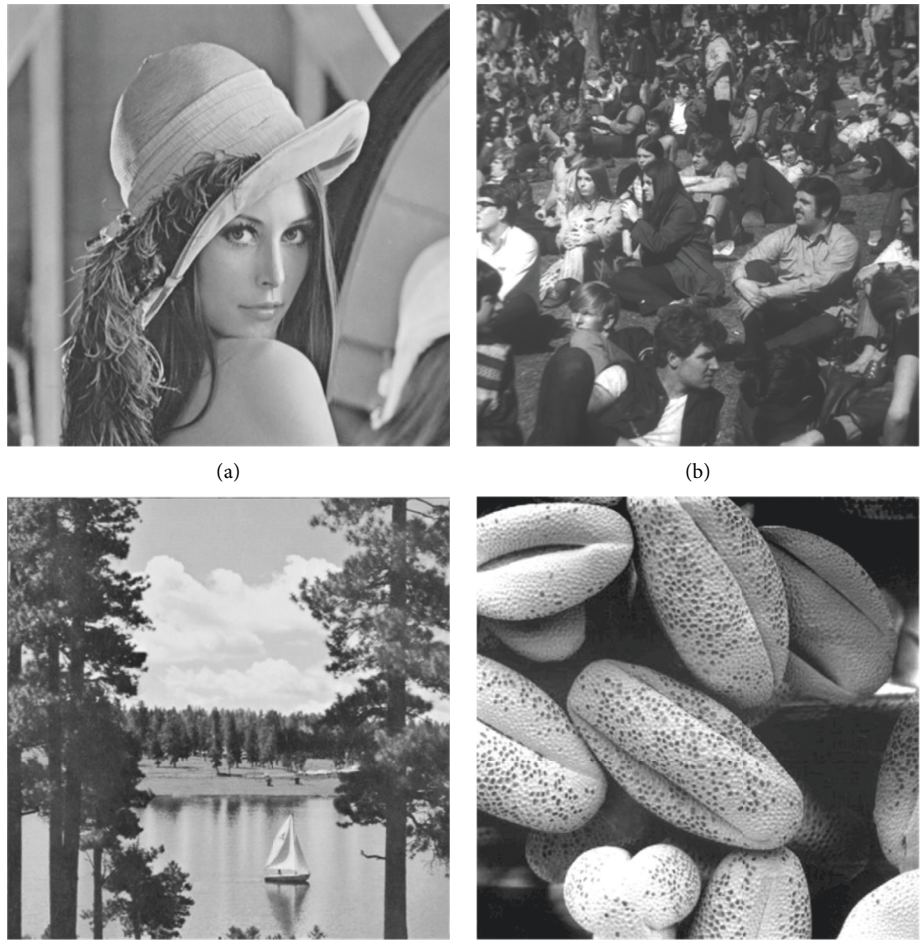


Figure 5: PBTL-based labeling process. (a) Marker bit selection. (b) Pixel grouping. (c) Pixel labeling.



(a) (b) (c) (d)

Figure 6: Test images. (a) Lena. (b) Crowd. (c) Lake. (d) Beans.

encryption key to generate a pseudo-random matrix $R_{m\times n}$, and then performs an exclusive OR operation with each bit of the image pixel to obtain the encrypted image bitstream.

3.3. PBTL-Based Pixel Grouping and Labeling. Inspired by IPBTL [17], we combine the AGP scheme and PBTL scheme in this paper. The content owner first performs pixel prediction on the original image via AGP, and then separates all pixels into four sets based on the prediction error, namely: reference pixel P_r, embeddable pixel P_e, non-embeddable pixel P_n and special pixel P_s. P_r remains unchanged during the prediction and recovery stages, which is used to predict the remaining pixels. At the same time, P_s remains unchanged during the data embedding process, which is employed to store parameters α and β. The remaining two sets are classified by equation (10).

If $E_{(i,j)}$ satisfies the following condition, the pixel $I_{(i,j)}$ belongs to P_e. Otherwise, it belongs to P_n.

$$I_{(i,j)} \in \begin{cases} P_e, & \left\lfloor -\frac{n_\alpha}{2} \right\rfloor \le E_{(i,j)} \le \left\lceil \frac{n_\alpha - 1}{2} \right\rceil, \\ P_n, & \text{others}, \end{cases} \tag{10}$$

where $E_{(i,j)}$ represents the prediction error of $I_{(i,j)}$, $\lceil * \rceil$ and $\lfloor * \rfloor$ represent the ceil and floor operations, respectively. n_α can be calculated by equation (1). N_r, N_e, and N_n denote the number of P_r, P_e, and P_n, respectively. P_s is the pixel for storing parameters. Thus, the whole image $m \times n = N_r + N_e + N_n + 1$.

Since pixels P_r and P_s are pre-defined, they can be easily distinguished, and only pixels in P_e and P_n need to be labeled. First, two parameters, α and β are obtained. Then, we use the binary code generated by PBTL to label pixels in P_e and P_n. Precisely speaking, for each pixels in P_n, it is only necessary to replace β bits binary sequence "0, ..., 0" with the most significant bits (MSB) of P_n by bit replacement, and the remaining other $(8 - \beta)$ bits unchanged. Simultaneously, for each pixel in P_e, we classify it into n_α different subcategories according to different values of $E_{(i,j)}$, and label it by n_α different α bits binary codes. Due to the correlation between pixels, adjacent pixels may have the same prediction error which can be labeled with the same binary code. Since marking the MSB of a pixel by bit substitution may lead to the leakage of the content information of the image, this paper inverts the binary code of all pixels in the labeling process to ensure the security of the image content. Figure 5(a) shows the labeling sequence corresponding to

Table 1: Threshold selection strategy.

T_1	ER(bpp)	T_2	ER(bpp)	T_3	ER(bpp)	T_4	ER(bpp)
50	2.7504	2	2.7420	2	2.7546	50	2.7520
60	2.7527	4	2.7448	4	2.7550	60	2.7525
70	2.7537	8	2.7479	**8**	**2.7551**	70	2.7528
80	**2.7551**	16	2.7526	16	2.7530	**80**	**2.7551**
90	2.7550	**32**	**2.7551**	32	2.7461	90	2.7530
100	2.7548	64	2.7510	64	2.7461	100	2.7530
110	2.7544	128	2.7460	128	2.7461	110	2.7530

different prediction errors; Figure 5(b) represents the classification of pixels according to the prediction error, and the pixel labeling process based on PBTL can be seen in Figure 5(c).

3.4. Data Extraction and Image Recovery. After receiving the marked image, we extract the parameters α and β from P_s firstly, by which the binary tree can be restored. The positions of P_e and P_n can be determined by checking the labeled bits of the non-reference pixels. The extracted data includes two parts: secret data and auxiliary information. Auxiliary information is applied to recover the cover image, which consists of the replaced original β bits of pixel in P_n and the original 8 bits of pixel in P_s.

Subsequently, we employ the embedding data for the original data recovery. The embedding pixel P_e includes α bits labeling bits and $(8-\alpha)$ bits for embedding data. Therefore, the entire embedded data includes $(8-\alpha)*N_e$ bits, where the number of auxiliary information and secret data is $(8+\beta*N_n)$ bits and $(8-\alpha)*N_e-(8+\beta*N_n)$ bits, respectively. To ensure the security of the approach, the embedded data and cover image are encrypted through k_d and k_e, respectively. Finally, the encrypted stego image can be generated.

In order to effectively evaluate the embedding performance of our method, we adopt the embedding rate $r_{\alpha,\beta}$ as the measurement index, which is defined as:

$$r_{\alpha,\beta}=\frac{(8-\alpha)*N_e-(8+\beta*N_n)}{m*n}, \tag{11}$$

where the maximum embedding rate $r_{\max}$(bpp) can be formulated as:

$$r_{\max}=\max\left(r_{\alpha,\beta}\right)_{\alpha,\beta=1}^{7}. \tag{12}$$

To show more details of the scheme, we take the pixel labeling process when the parameter is $\alpha=3,\beta=2$ as an example and display it in Figure 5. The labeling bits selection is shown in Figure 5(a), where P_n is labeled with "00," and P_e is labeled with "010," "011," "100," "101," "110" and "111." For example, pixels with a prediction error of "−3" are labeled with the sequence of "010." The detailed process of secret data extraction can be summarized as follows:

Step 1: after receiving the marked encrypted image I_E and data hiding key k_d, the receiver firstly extracts parameters α and β from the pre-defined special pixel P_s.

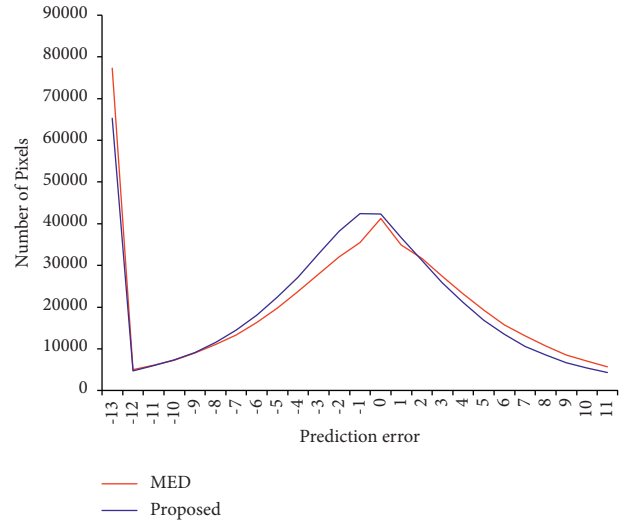


Figure 7: Prediction performance comparison.

Step 2: check α or β bits label of remaining pixels in the 8 bits binary sequence in the reverse order.

Step 3: classify them into embeddable pixel set P_e and non-embeddable pixel set P_n.

Step 4: extract $(8-\alpha)$ bits of the payload from the pixel in turn, and obtain the encrypted data.

Step 5: decrypt the original secret information D by using k_d.

After obtaining the marked image, the recipient can restore the original image only by obtaining the image encryption key k_e. First, the auxiliary information is extracted in the embedded data. Next, we recover the replaced bits in P_n and the original 8 bits binary sequence in P_s by the first part of the auxiliary information and the second part of the auxiliary information, respectively. Then, the original values of P_n and P_s are restored according to the auxiliary information. Finally, the key k_e is used to fully obtain the original image.

4. Experimental Results and Analysis

To verify the performance of the proposed method, this section designs many experiments. This section consists of five parts. The experimental environment and the dataset used are introduced in Sections 4.1 and 4.2. Threshold selection experiments and comparative experiments for

Table 2: The embedding rate of the test images when $\beta = 3, \alpha = 1, \ldots, 7$.

Image	(α, β)	(2, 3)	(3, 3)	(4, 3)	(5, 3)	(6, 3)	(7, 3)
Lena	MED	—	1.7087	2.7872	2.6770	1.9407	0.9929
	GAP	—	2.0343	2.9882	2.7369	1.9537	0.9945
	Proposed	—	**2.0664**	**3.0642**	**2.7874**	**1.9689**	**0.9957**
Crowd	MED	0.9243	2.2093	2.6540	2.5488	1.9133	0.9932
	GAP	0.9153	2.2540	2.7320	2.5863	1.9245	0.9937
	Proposed	**0.9985**	**2.3671**	**2.8291**	**2.6379**	**1.9370**	**0.9939**
Lake	MED	—	—	1.2060	2.0836	1.8390	0.9906
	GAP	—	0.1794	1.5029	2.2203	1.8547	0.9901
	Proposed	—	**0.2786**	**1.5751**	**2.2413**	**1.8665**	**0.9909**
Beans	MED	—	**0.8664**	1.7448	2.2484	1.8867	0.9912
	GAP	—	0.7109	1.6675	2.1988	1.8780	0.9909
	Proposed	—	0.7937	**1.7787**	**2.2785**	**1.8996**	**0.9927**

Table 3: The embedding rate of the test images when $\beta = 2, \alpha = 1, \ldots, 7$.

Image	(α, β)	(2, 2)	(3, 2)	(4, 2)	(5, 2)	(6, 2)	(7, 2)
Lena	MED	0.3933	1.6609	2.6867	2.6447	1.9285	0.9919
	GAP	**0.6109**	2.0127	2.8977	2.7055	1.9440	0.9935
	Proposed	0.6022	**2.0573**	**2.9596**	**2.7551**	**1.9595**	**0.9951**
Crowd	MED	1.4856	2.3263	2.6530	2.5180	1.8966	0.9906
	GAP	1.4777	2.3745	2.7126	2.5536	1.9071	0.9912
	Proposed	**1.5516**	**2.4751**	**2.8050**	**2.6053**	**1.9228**	**0.9922**
Lake	MED	—	0.2035	1.2433	1.9997	1.8080	0.9856
	GAP	—	0.4672	1.5093	2.1449	1.8284	0.9852
	Proposed	—	**0.5587**	**1.5769**	**2.1707**	**1.8399**	**0.9866**
Beans	MED	**0.5456**	**1.1069**	1.8032	2.1669	1.8569	0.9880
	GAP	0.2821	0.9647	1.7169	2.1335	1.8461	0.9877
	Proposed	0.2842	1.0453	**1.8053**	**2.2070**	**1.8691**	**0.9906**

predictors are presented in Section 4.3. Four gray images, Lena, Crowd, Lake, and Beans shown in Figure 6 are used to display the specific performance in our experiments in Section 4.4. To reduce the effect of image randomness on the authenticity of the results, the average embedding rate on three large image datasets: UCID [22], BOSSbase [23], and BOWS-2 [24]is compared. Some evaluation indexes estimate the performance of the proposed scheme, i.e., embedding rate, peak signal-to-noise ratio (PSNR), and structural similarity index (SSIM) experiments.

4.1. Experimental Environment. This experiment uses Intel(R) Core (TM) i7-10750H CPU @ 2.60 GHz, 16.00 GB RAM, and Nvidia GeForce RTX 2060 gpu. All experiments are done under MATLAB R2019a.

4.2. Datasets. The specific description of the three datasets is as follows:

(1) UCID: The UCID dataset was created by Gerald Schaefer et al. It contains 1338 uncompressed TIFF images, involving various subjects such as indoor and outdoor natural scenes and artificial objects, all taken by a digital color camera. This dataset is mainly used to evaluate image compression and color quantization.

(2) BOSSbase: Patrick Bas and others created BOSSbase. The BOSSbase dataset consists of uncompressed images taken by seven different cameras. All images are created from full-resolution color images and are finally converted into grayscale images through operations such as resizing and cropping. BOSSbase has gone through three versions: 7518 for the 0.9 version in June 2010, 9074 for the 0.92 version, and finally 10000 for the 1.0 version in May 2011.

(3) BOWS-2: BOWS-2 was created by P. Bas et al. It contains 10000 512×512 PGM images. This dataset is taken by 7 different cameras. The creation of this dataset is mainly provided for participants who participate in the BOSS steganalysis contest.

4.3. Predictor Analysis. To verify the effectiveness of threshold selection, this section conducts performance experiments under different threshold settings. The optimal or near-optimal result of the target image is selected as the threshold, and the effectiveness of the threshold is verified on a large image dataset. For ordinary edges and weak edges, the

Table 4: The embedding rate of the test images when $\beta = 4, \alpha = 1, \ldots, 7$.

Image	(α, β)	(2, 4)	(3, 4)	(4, 4)	(5, 4)	(6, 4)	(7, 4)
Lena	MED	—	1.2997	2.7703	2.6693	1.9423	0.9933
	GAP	—	1.6660	2.9584	2.7341	1.9559	0.9950
	Proposed	—	**1.7021**	**3.0398**	**2.7869**	**1.9707**	**0.9958**
Crowd	MED	0.3629	1.8629	2.5719	2.5336	1.9146	0.9938
	GAP	0.3529	1.9132	2.6538	2.5759	1.9266	0.9942
	Proposed	**0.4454**	**2.0404**	**2.7535**	**2.6312**	**1.9391**	**0.9943**
Lake	MED	—	—	1.0276	2.0587	1.8432	0.9919
	GAP	—	—	1.3471	2.2013	1.8567	0.9914
	Proposed	—	—	**1.4239**	**2.2261**	**1.8687**	**0.9918**
Beans	MED	—	**0.3522**	1.5942	2.2255	1.8912	0.9919
	GAP	—	0.1773	1.5036	2.1805	1.8829	0.9917
	Proposed	—	0.2704	**1.6302**	**2.2675**	**1.9060**	**0.9934**

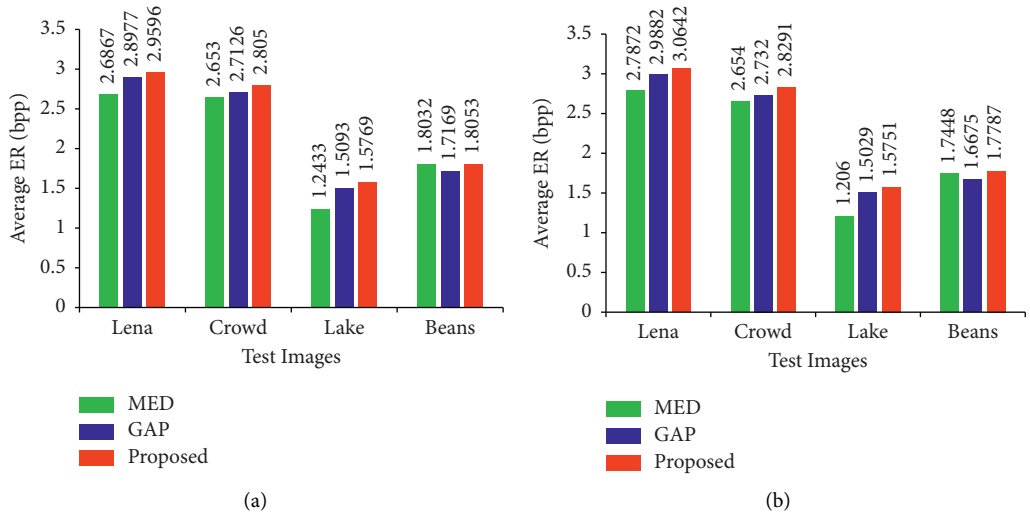


Figure 8: Comparison of the average ER (bpp) of test images between different methods. (a) $\alpha = 4$, $\beta = 2$. (b) $\alpha = 4$, $\beta = 3$.

power of 2 is taken as the threshold selection strategy. The threshold is dynamically adjusted according to the neighborhood complexity of the predicted pixel to achieve local optimization. The neighborhood gradient produces a weak influence when the predicted pixel is in the sharp edge. When it is in the ordinary edge or the soft edge, the effect produced by the neighborhood gradient increases slightly. Table 1 analyzes the embedding performance of Lena under different thresholds.

It can be seen from Table 1 that the predicted pixel on sharp edges has the best case when T_1 is 80. Similarly, the predicted pixel on normal edges has the best case when T_2 is 32. When the predicted pixel is on a weak edge, the threshold T_3 is 8. For diagonal gradient edges, when the threshold T_4 is 80, the solution's performance is optimal.

To verify the predictive performance of the scheme, three gray images of Lena, Man, and Baboon were selected to conduct prediction error experiments. As shown in Figure 7,

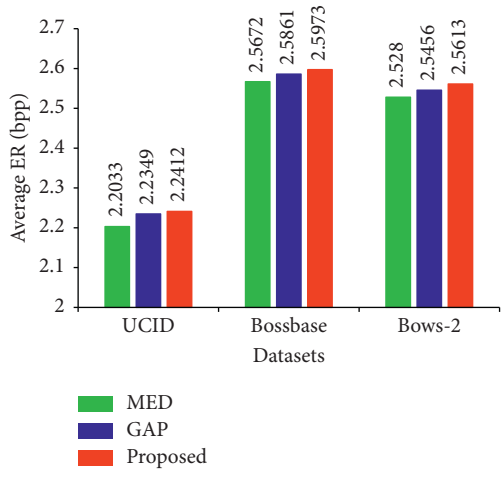


Figure 9: Comparison of the average ER (bpp) of datasets between different methods.

Table 5: Comparison of embedding performance on three datasets.

Datasets	(α, β)	Methods	Best case	Average
UCID	$\alpha = 5, \beta = 2$	MED	2.9759	**2.2683**
		GAP	2.9794	2.2412
		Proposed	**2.9860**	2.2473
	$\alpha = 4, \beta = 2$	MED	3.9349	2.2551
		GAP	3.9364	2.2604
		Proposed	**3.9367**	**2.2621**
	$\alpha = 5, \beta = 3$	MED	2.9793	2.2033
		GAP	2.9834	2.2349
		Proposed	**2.9883**	**2.2412**
BOSSbase	$\alpha = 5, \beta = 2$	MED	2.9882	2.5613
		GAP	2.9882	2.5756
		Proposed	2.9882	**2.5869**
	$\alpha = 4, \beta = 2$	MED	3.9843	2.9332
		GAP	3.9843	2.9218
		Proposed	3.9843	**2.9424**
	$\alpha = 5, \beta = 3$	MED	2.9882	2.5672
		GAP	2.9882	2.5861
		Proposed	2.9882	**2.5973**
BOWS-2	$\alpha = 5, \beta = 2$	MED	2.9882	2.5194
		GAP	2.9882	2.5320
		Proposed	2.9882	**2.5475**
	$\alpha = 4, \beta = 2$	MED	3.9838	2.8311
		GAP	3.9838	2.8144
		Proposed	**3.9841**	**2.8415**
	$\alpha = 5, \beta = 3$	MED	2.9882	2.5280
		GAP	2.9882	2.5456
		Proposed	2.9882	**2.5613**

the x-axis is the prediction error, and the y-axis is the number of pixels under that error.

It can be seen that in the prediction error interval [−12, 2], the number of pixels of this scheme is significantly higher than that of MED, which means that there are more embedded pixels. In contrast, this scheme has significantly fewer non-embeddable pixels than MED. Therefore, this scheme has better prediction performance.

4.4. Comparison and Analysis of Embedding Capacity. To verify the performance of the AGP-RDHEI scheme. This section compares the embedding rate of this scheme with the MED method [17] and the GAP method [20]. As shown Table 2 in Tables 2–4, four test images were selected for ER experiments. The parameter settings of experiments are $\beta = 2, 3, 4$ and $\alpha = 2, 3, \ldots, 7$.

It can be seen that when α is set to a small value, the encrypted image with labels cannot embed secret data. The "/" in Tables 2–4 indicates that the auxiliary information is larger than the reserved room, so the secret data cannot be embedded. Tables 2–4 shows that the maximum embedding rate can be achieved by adjusting the parameter settings. Furthermore, this scheme has better results for images with complex textures. The maximum ER on the four images can reach 3.0642bpp, 2.8291bpp, 1.5751bpp and 1.7787bpp, respectively.

To more intuitively reflect the performance advantages of this solution, Figure 8 visualizes the embedding rate experiment with the parameter set to $\alpha = 4, \beta = 2, 3$.

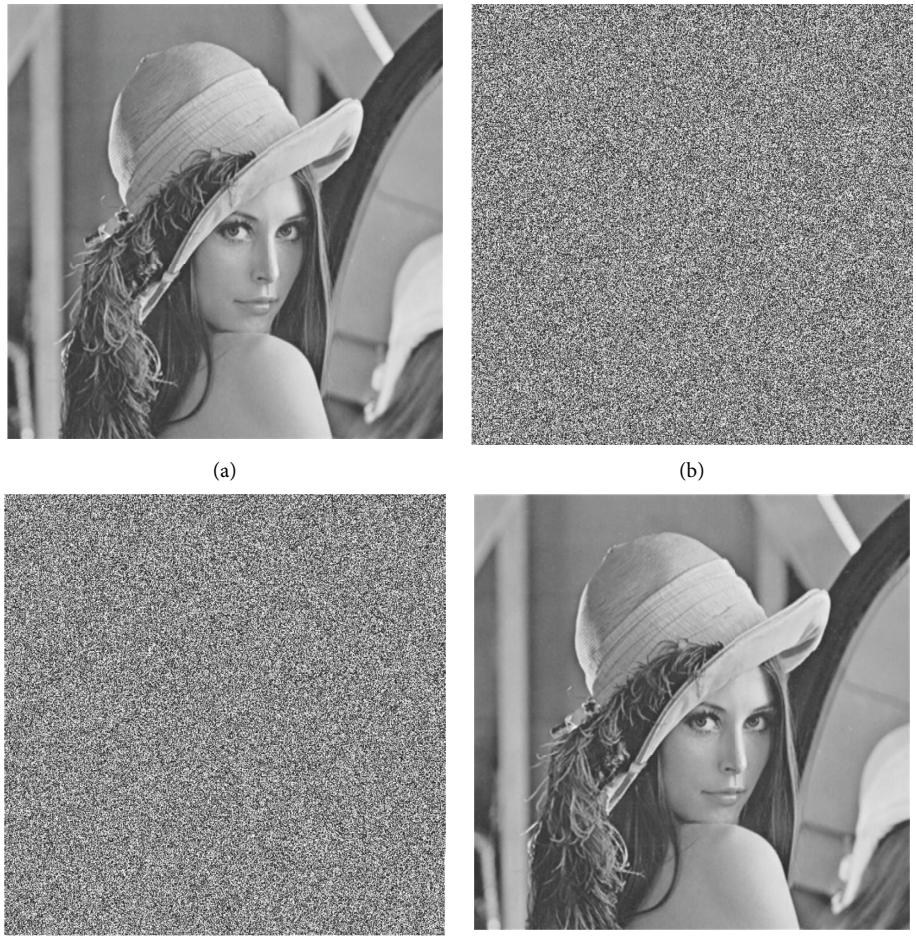
Figure 9 shows the comparative experiments of different schemes on three datasets with the parameter set to $\alpha = 5, \beta = 3$ to reduce the randomness of the selected test images.

The abscissa is the name of the image library, and the ordinate is the average embedding rate in Figure 9. It can be seen that the algorithm in this paper also has promising results on large image datasets. Its average embedding rate is better than literature [20] and significantly better [17].

To further verify the performance of the scheme. As shown in Table 5, we added two sets of experiments with different parameter settings. It can be seen that the results of this scheme under different parameter settings have advantages. It can achieve 2.9424bpp on BOSSbase and 2.8415bpp on BOWS-2.

4.5. Security Analysis. Figure 10 shows the simulation result of Lena, and the sub-figure are the process images of the scheme at different stages. In addition, since the original image and data are encrypted, which can effectively prevent the leakage of image content and data during transmission. It can be seen that it is not easy to detect the content of the original image, e.g., Figures 10(b) and 10(c).

To further verify the security of the scheme, PSNR and SSIM experiments had also been conducted. The smaller the value of SSIM, the greater the distortion of the image. As shown in Table 6 and Tables 6–8, the SSIM value of the encrypted image is almost zero, which effectively protects the security of the image.



(a) (b) (c) (d)

Figure 10: Simulation results of the solution on the Lena image. (a) Original image. (b) Encrypted image. (c) Secret image. (d) Recovered image.

Table 6: PSNR and SSIM of the original images and tagged encrypted images when $\alpha = 5, \beta = 2$.

Encrypted image with label	Lena	Crowd	Lake	Beans
PSNR	9.2231	8.2648	8.2204	7.6936
SSIM	0.0342	0.0318	0.0288	0.0235

Table 7: PSNR and SSIM of the original images and encrypted images when $\alpha = 5, \beta = 2$.

Encrypted image	Lena	Crowd	Lake	Beans
PSNR	9.2255	8.2341	8.2198	7.6256
SSIM	0.0341	0.0301	0.0291	0.0219

Table 8: PSNR and SSIM of the original images and the marked encrypted images when $\alpha = 5, \beta = 2$.

Marked encrypted image	Lena	Crowd	Lake	Beans
PSNR	9.2295	8.2721	8.2327	7.7018
SSIM	0.0365	0.0327	0.0281	0.0254

5. Conclusion

In this paper, we propose a new RDH-EI algorithm based on adaptive gradient prediction, which can be widely applied to various existing schemes. AGP can adaptively adjust the prediction model with the position of the predicted pixel, which improves the predictor's ability to perceive local pixel changes. In addition, the diagonal gradient is introduced when measuring local complexity, effectively enhancing the sensitivity of the predictor to pixel changes. Moreover, we combine the AGP with the PBTL scheme, which reserves more room for embedding data to achieve a high embedding capacity. In the future, we can try to optimize the predictor and labeling scheme to further improve the embedding performance of the scheme.

Data Availability

The datasets UCID, BOSSbase, and BOWS-2 used in this paper can be available from [22–24]. All the experiment results, codes, and data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the Soft Science Research Project of Guangdong Digital Government Reform and Construction Expert Committee (No. ZJWKT202204), the National Natural Science Foundation of China under Grant (62002392), and the Natural Science Foundation of Hunan Province under Grant (2020JJ4140 and 2020JJ4141).

References

[1] T. Zheng, Y. Luo, T. Zhou, and Z. Cai, "Towards differential access control and privacy-preserving for secure media data sharing in the cloud," *Journal of Computer Security*, vol. 113, 2022.

[2] X. Zhang, "Reversible data hiding with optimal value transfer," *IEEE Transactions on Multimedia*, vol. 15, no. 2, pp. 316–325, 2013.

[3] C. Wang, Y. Liu, Y. Tong, and J. Wang, "GAN-GLS: generative lyric steganography based on generative adversarial Networks," *Computers, Materials & Continua*, vol. 69, no. 1, pp. 1375–1390, 2021.

[4] Q. Liu, X. Xiang, J. Qin, Y. Tan, and Q. Zhang, "A robust coverless steganography scheme using camouflage image," *IEEE Transactions on Circuits and Systems for Video Technology*, 2022.

[5] Y. Luo, J. Qin, X. 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.

[6] J. Fridrich, M. Goljan, and R. Du, "Lossless data embedding: new paradigm in digital watermarking," *EURASIP Journal on Applied Signal Processing*, vol. 2002, no. 1, pp. 185–196, 2002.

[7] M. U. Celik, G. Sharma, A. M. Tekalp, and E. Saber, "Lossless generalized-LSB data embedding," *IEEE Transactions on Image Processing*, vol. 14, no. 2, pp. 253–266, 2005.

[8] L. Lixin Luo, Z. Zhenyong Chen, M. Ming Chen, X. Zhang Xiong, and Z. Xiong, "Reversible image watermarking using interpolation technique," *IEEE Transactions on Information Forensics and Security*, vol. 5, no. 1, pp. 187–193, 2010.

[9] X. Xiaolong Li, W. Weiming Zhang, X. Bin Yang, and B. Yang, "A novel reversible data hiding scheme based on two-dimensional difference-histogram modification," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 7, pp. 1091–1100, 2013.

[10] W. Puech, M. Chaumont, and O. Strauss, "A reversible data hiding method for encrypted images," Security, Forensics, Steganography, and Watermarking of Multimedia Contents X," *Proceedings of SPIE*, vol. 6819, 2008.

[11] X. Zhang, "Reversible data hiding in encrypted image," *IEEE Signal Processing Letters*, vol. 18, no. 4, pp. 255–258, 2011.

[12] J. Zhou, W. Sun, L. Dong, X. Liu, O. C. Au, and Y. Y. Tang, "Secure reversible image data hiding over encrypted domain via key modulation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 3, pp. 441–452, 2016.

[13] 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.

[14] 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.

[15] S. Yi and Y. Zhou, "Binary-block embedding for reversible data hiding in encrypted images," *Signal Processing*, vol. 133, pp. 40–51, 2017.

[16] X. Zhang, "Separable reversible data hiding in encrypted image," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 2, pp. 826–832, 2012.

[17] 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.

[18] Z. Yin, X. She, J. Tang, and B. Luo, "Reversible data hiding in encrypted images based on pixel prediction and multi-MSB planes rearrangement," *Signal Processing*, vol. 187, no. 2, Article ID 108146, 2021.

[19] Y. Yang, H. He, and F. Chen, "Reversible data hiding of image encryption based on prediction error adaptive coding," *Journal of Computer Research and Development*, vol. 58, no. 6, pp. 1340–1350, 2021.

[20] X. Wu and N. Memon, "Context-based, adaptive, lossless image coding," *IEEE Transactions on Communications*, vol. 45, no. 4, pp. 437–444, 1997.

[21] R. Uyyala and R. Pal, "Reversible data hiding using improved gradient based prediction and adaptive histogram bin shifting," in *Proceedings of the 2020 7th International Conference on Signal Processing and Integrated Networks (SPIN)*, pp. 720–726, Noida, India, February 2020.

[22] G. Schaefer and M. Stich, "Ucid: An uncompressed color image database," vol. 5307, pp. 472–480, 2003, https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.101.8709&rep=rep1&type=pdf.

[23] P. Bas, T. Filler, and T. Pevný, "Break our steganographic system: the ins and outs of organizing BOSS," in *Proceedings of the International Workshop on Information Hiding*, pp. 59–70, Prague, Czech, October 2011.

[24] P. Bas and T. Furon, "Image database of bows-2," *Access*, vol. 20, 2017.