Multi-Dimensional Histogram Modification Framework for Reversible Data Hiding in JPEG Images

Kaiyue Hou, Cheng Zeng, Xiaoyi Zhou, *Member, IEEE*, Dahao Fu, Xin Liao, *Member, IEEE*

***Abstract*—Reversible data hiding in JPEG images is critical for secure multimedia applications, while, existing histogram modification schemes are confined to fixed 1D or 2D dimensions. These methods fail to adapt to the characteristics of cover images and the demands of embedded data, resulting in suboptimal trade-offs among embedding capacity, image quality, and file size. To address these issues, this study proposes a Multi-Dimensional Histogram Modification Framework that advances histograms from "fixed dimension" to "variable dimension" and extends them to high-dimensional structures. Our key contributions are as follows: first, we propose an adaptive dimension selection strategy that dynamically determines the optimal histogram dimension by evaluating the capacity and distortion for each cover image and embedding task; second, we develop a high-dimensional histogram construction method that combines AC coefficients with absolute values of 1 and 2 at the same frequency to enhance the utilization of coefficient correlation; third, we design an optimized mapping algorithm that searches for the optimal mapping matrix to minimize both distortion and file expansion. Experimental results show that the proposed method outperforms existing 1D and 2D histogram modification schemes. On average, it increases the embedding capacity by 5000 bits, improves the image quality by 0.2 dB, and reduces the file size by 4000 bits.**

***Index Terms*—Reversible data hiding, JPEG image, multi-dimensional histogram.**

I. Introduction

REVERSIBLE data hiding(RDH) has garnered considerable interest due to its potential in enhancing the security of multimedia data [1]–[3], promoting the transmission of confidential information [4]–[6] and managing multimedia data [7]–[9]. The key feature of RDH lies in its ability to embed specific data into the cover images and restore both data and images without any loss, making it particularly important in military, medical, judicial, and other applications.

The modification of the histogram in RDH for uncompressed images has received significant attention and is widely studied [10], [11]. They involve statistical analysis of the cover image to create histograms, followed by modifications to these histograms. For instance, a two-dimensional(2D) histogram is formed by Ou *et al*. using pairwise prediction errors [12]. Zhang *et al* [13] first introduced the concept of adaptivity to histogram modification by employing adaptive two-dimensional prediction-error histograms that derive optimal modification mappings tailored to the local statistical characteristics of each image. To extend Ou's algorithm to color images, three-dimensional(3D) histograms are proposed, using double deep Q-network [14] and adaptive mapping selection [15] to construct 3D mappings. However, these methods are not suitable for JPEG-compressed images, even though JPEG remains one of the most ubiquitously adopted image formats in real-world applications, as their redundancy is limited, resulting in lower embedding capacity, and even slight modifications can cause significant image distortion and file expansion. Therefore, histogram modification schemes for compressed images have been proposed.

JPEG RDH can be classified into three types based on the location of modification. One type is quantization table modification [16], where the coefficients are modified by reducing the quantization step size in order to embed data. The second one is Huffman table modification [17]–[23], which achieves data embedding by using multiple codes to represent the same Run/Size. The last is the discrete cosine transform(DCT) coefficients modification [24]–[38]. However, in recent years, scholars have shifted their attention from compressing DCT coefficients to modifying the histograms formed by DCT coefficients, and there are one-dimensional [24]–[33] and the 2D [34]–[38] histograms for data embedding in JPEG RDH.

In the construction of one-dimensional histograms for reversible data hiding, several advancements have been proposed to enhance embedding performance. Coefficient selection strategies [25]–[29] prioritize low-frequency DCT coefficients with high embedding efficiency, thereby reducing visual distortion. To mitigate embedding induced degradation, negative influence models [30] estimate the potential impact of modifying each coefficient on visual quality and file size. Multi-objective optimization models [31] jointly consider embedding capacity, distortion, and bitrate, enabling flexible trade-offs for quality-critical applications. Multi-distortion metrics [32] extend traditional quality indicators by incorporating file expansion and perceptual quality, offering better control in the JPEG domain. In addition, multiple histogram strategies [33]create several

Kaiyue Hou, Cheng Zeng, and Dahao Fu are with the School of Cyberspace Security, Hainan University, Hainan, Haikou 570228, China (e-mail: 1213670222@qq.com, 23210839000021@hainanu.edu.cn, fudahao2021@163.com).

Xiaoyi Zhou is with the School of Cyberspace Security, Hainan University, Hainan, Haikou 570228, China (e-mail: xy.zhou@hainanu.edu.cn).(Corresponding author)

Xin Liao is with the College of Computer Science and Electronic Engineering, Hunan University, Changsha 410082, China, and also with the Shenzhen Key Laboratory of Media Security, Shenzhen University, Shenzhen 518060, China (e-mail: xinliao@hnu.edu.cn).(Corresponding author)

This work was supported by the National Natural Science Foundation of China under Grants 62362025.

sub-histograms across different regions or frequency bands to increase embedding capacity.

For two-dimensional histograms, various approaches aim to improve mapping efficiency. A representative method is the seven-type 2D mapping [36], which classifies coefficient pairs into predefined types and designs specific mappings to improve histogram concentration. In contrast, adaptive 2D mapping selection methods [37], [38] evaluate multiple candidate mappings based on capacity and distortion metrics and select the optimal one dynamically. These strategies demonstrate robustness across image types and quality factors.

The deficiencies of the above histogram modification-based JPEG RDH can be summarized as follows: (1) The preset histogram dimension fails to account for the influence of cover images and embedded data. Differences in coefficient distributions among cover images shape distinct high-dimensional histograms, affecting embedding capacity. Embedded data, meanwhile, directly alters coefficient shifts, leading to variations in file size and distortion. (2) The 3D histograms in uncompressed images excel, but current JPEG image methods are confined to 1D or 2D histograms, neglecting high-dimensional correlations. Higher dimensions complexify mappings, offering potential for optimal mapping construction.

Therefore, this paper presents a multi-dimensional histogram modification framework. Initially, a dimension selection strategy is proposed to dynamically identify the optimal-dimension histogram that balances both payload capacity and minimal distortion. This approach considers the unique attributes of both the cover image and embedding data, ensuring an optimal trade-off between capacity and distortion. Subsequently, the framework integrates sophisticated 2D and 3D histogram modification algorithms that harness the inherent correlations within DCT coefficients. Ultimately, a high-dimensional mapping strategy and optimization algorithm are introduced to explore the optimal mapping, offering greater capacity and reduced distortion. Compared to the existing framework, the proposed one achieves superior performance in embedding capacity, image quality, and file size. Specifically, in embedding capacity, the proposed framework has an average increase of 5000 bits; in image quality, it has an average improvement of 0.2 dB; in file size, it reduces the file size by an average of 4000 bits.

- To the best of our knowledge, we first propose to obtain a variable histogram through the dimension selection strategy in JPEG reversible data hiding. By comparing the capacity and distortion of the histogram at different dimensions and adaptively selects the optimal one based on the cover image and embedded data, it increases the embedding capacity, reduces embedding distortion and prevents file size from increasing.
- This paper is proposed to form a high-dimensional histogram by combining AC with absolute values of 1 and 2 at the same frequency. It maximizes the utilization of correlations between coefficients and significantly enhances the concentration of the histogram, and leading to improved information hiding performance.
- The high-dimensional mapping strategy proposed in this paper uses an optimization algorithm to search for the optimal one in a set of matrices for constructing mappings. This method considers the effect of changes in ACs on distortion and improves the efficiency of the modification process.

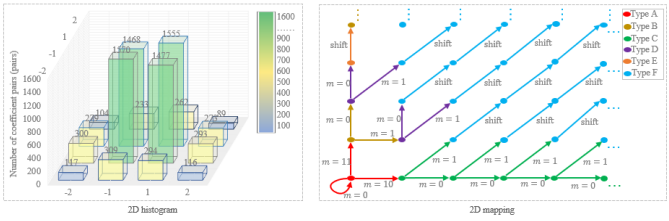


Fig. 1. 2D histogram and 2D mapping proposed by Chen *et al*. in [36]

The rest of this paper is organized as follows. Section II provides a review of traditional histogram modification algorithms. Section III further discusses the design of MHMF, aiming to address issues such as reducing file size and image distortion while increasing embedding capacity. In Section IV, a comprehensive evaluation and discussion is conducted on the performance of the proposed framework to verify the superiority and effectiveness in processing high-dimensional data. Finally, Section V summarizes the main points and analyzes its imperfection, as well as future research directions.

II. Related Work

Chen *et al*. [36] have made substantial advancements in enhancing the performance of 2D histogram modification, which will be detailed in Section II-A. Although the current application of 3D histogram modification is primarily limited to uncompressed images, an algorithm proposed by Chang *et al*. [15] will be introduced as an example in Section II-B. In Section II-C, a summary of traditional histogram modification algorithms will be provided.

A. Chen's Scheme

Firstly, in order to form an ordered sequence of non-zero ACs, the blocks and frequencies are sorted by the number of zero ACs and embedding efficiency, respectively. Secondly, to form a 2D histogram, the sequence is combined into pairs in order, including Type A = ($|a| = 1, |b| = 1$), Type B = ($|a| = 1, |b| = 2$ or $|a| = 1, |b| >= 5$), Type C = ($|a| >= 2, |b| = 1$), Type D = ($|a| = 2, |b| = 2$ or $|a| = 1, |b| = 3$), Type E = ($|a| = 1, |b| = 4$) and Type F (the remaining coefficient pairs), where ($|a|, |b|$) refers to the absolute value of pair (AC_i^{j1}, AC_i^{j2}), AC_i^{j1} refers to the AC of the ith ($1 \leq i \leq 63$) frequency and $j1$th ($1 \leq j1 \leq N,\ j1 \leq j2$) block, N refers to the number of blocks. Thirdly, to embed data, the resulting 2D histogram is mapped according to a 2D mapping in Fig. 1. Finally, to recover a marked image, the 2D histogram is transformed back into the sequence by the inverse process of the first step. Experiments are conducted on JPEG images under QF=60(Quality Factor) following this method. The number of pairs is counted, and a 2D histogram is formed as shown in Fig. 1.

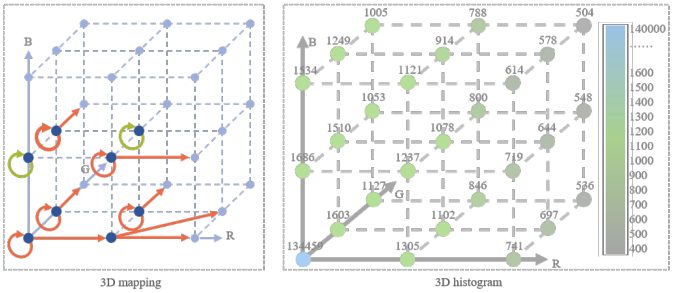


Fig. 2. 3D histogram and 3D mapping proposed by Chang *et al*. in [15]

B. Chang' s Scheme

At first, all original pixels are predicted, and prediction errors $e_{x,y} = f_{x,y} - \hat{f}_{x,y}$ are calculated to form an ordered sequence, where $x \in \{R, G, B\}$ refers to the index of the channel, $y\,(1 \le y \le k)$ is the position of the pixel, k is the number of pixels in each channel, $f_{x,y}$ and $\hat{f}_{x,y}$ are the original pixel and predicted pixel at the xth channel and yth position, respectively. Next, the sequence is combined into triples in order to form a 3D histogram. Then, the resulting 3D histogram is mapped according to a 3D mapping to embed data. It shows the 3D mapping constructed by the adaptive mapping selection when the payload is 50000 bits in Fig. 2. Lastly, the 3D histogram is transformed back into the sequence using the inverse process of step one to recover a marked image. This method is tested on the JPEG images. A tally is taken of the number of triples falling within the range of absolute values of 1 and 2, and based on that, a 3D histogram is formed, as illustrated in Fig. 2.

C. Traditional Histogram Modification framework

In Fig. 3, a traditional histogram modification framework typically comprises four essential steps: namely pretreatment, histogram formation, histogram mapping and recovery pretreatment. Drawing inspiration from this framework, many researchers have proposed excellent algorithms that significantly improve efficiency of embedding. Traditional frameworks cannot handle the correlation existing in JPEG images in higher dimensional spaces, nor can they adaptively select the optimal dimensional histogram. Thus, an improved framework is proposed. Compared with traditional frameworks, the proposed method improves histogram formation by splitting it into two steps, as shown in Fig. 3.

III. Proposed Variable High-Dimensional Histogram Modification Framework

The proposed MHMF is illustrated in Fig. 4, which, compared with the traditional histogram modification framework, MHMF mainly addresses two key issues. The first is how to upgrade a fixed histogram into a variable histogram, which will be discussed in detail in Section III-B. The second is how to construct the optimal 3D mapping, which will be discussed in Section III-C. The process of embedding, extraction and recover will be disscussed in Section III-D.

A. Overview of the Proposed Framework

In the pretreatment stage, a cover JPEG image is partially decompressed to obtain quantized DCT coefficients. Subsequently, blocks and frequencies are ordered to produce sorted DCT coefficients. From these ordered DCT coefficients, embedding coefficients, boundary coefficients, and invariant coefficients are classified. In the adaptive selection of histogram dimensions, additional data, embedding coefficients, and boundary coefficients are used to estimate the embedding capacity and embedding distortion. The optimal dimension is selected based on the criteria of maximizing capacity and minimizing distortion. During the formation of the histogram with the optimal dimension, a histogram of the optimal dimension and a histogram of boundary coefficients are constructed according to the selected optimal dimension. In the histogram mapping process, the additional data, the mapping of the optimal dimension, the histogram of the optimal dimension, and the histogram of boundary coefficients are modified to create a marked histogram of the optimal dimension and a marked histogram of boundary coefficients.

B. Adaptive Selection of Histogram Dimensions

In this section, the proposed high-dimensional modification algorithms and the adaptive dimension selection strategy are introduced.

1) High-dimensional modification algorithms: Firstly, among the AC coefficients obtained through pretreatment, the coefficients with values {1,2} are classified as embedding coefficients and the coefficients with values $S \in \{3, 4, ..., 1023\}$ are classified as boundary coefficients. This classification strictly keeps zero coefficients unchanged, which directly protects the sparsity of quantized coefficients. Then, to ensure that ACs with higher embedding efficiency are prioritized for embedding data, 63 frequencies are sorted according to their embedding efficiency. Generally, the embedding efficiency is defined as the amount of data that can be embedded per unit of distortion. The embedding efficiency is defined as Eq.(1).

$$E_i^{\gamma} = \frac{C_i^{\gamma}}{D_i^{\gamma}} = \begin{cases} \dfrac{\mathrm{Z}(i)\left[N_i(S)+{p_1}_i^{\gamma} N_i(1)+{p_2}_i^{\gamma} N_i(2)\right]}{\sum_{m=1}^{4}\left[N_i^2(\text{Pair } m)\log_2 \sum_{j=1}^{9} h_{mj}^{2D}\right]}, & \text{if } \gamma = 2 \\ \dfrac{\mathrm{Z}(i)\left[N_i(S)+{p_1}_i^{\gamma} N_i(1)+{p_2}_i^{\gamma} N_i(2)\right]}{\sum_{n=1}^{8}\left[N_i^3(\text{Triple } n)\log_2 \sum_{j=1}^{27} h_{nj}^{3D}\right]}, & \text{if } \gamma = 3, \end{cases} \tag{1}$$

where $\gamma(\gamma \in \{2, 3\})$ refers to the dimension of histogram, for example, $\gamma = 3$ represents 3D histogram. $i(1 \le i \le 63)$ refers to the index of the frequency, for example, $i = 2$ represents $2th$ frequency. E_i^{γ} refers to the embedding efficiency, D_i^{γ} and C_i^{γ} refer to the distortion and capacity, $N_i(S)$ refers to the number of boundary coefficients, $N_i(1)$ and $N_i(2)$ refers to the number of ACs with absolute value 1 and 2, ${p_1}_i^{\gamma}$ refers to the probability that ACs with absolute value 1 are changed to 2, ${p_2}_i^{\gamma}$ refers to the probability that ACs with absolute value 2 are changed to 3. $\mathrm{Z}(i)$ is a function used to incorporate some characteristics of the image itself into the evaluation of

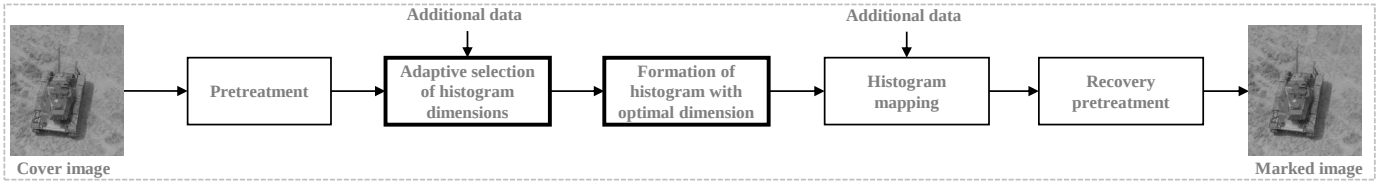


Fig. 3. The traditional histogram modification and the improved framework.

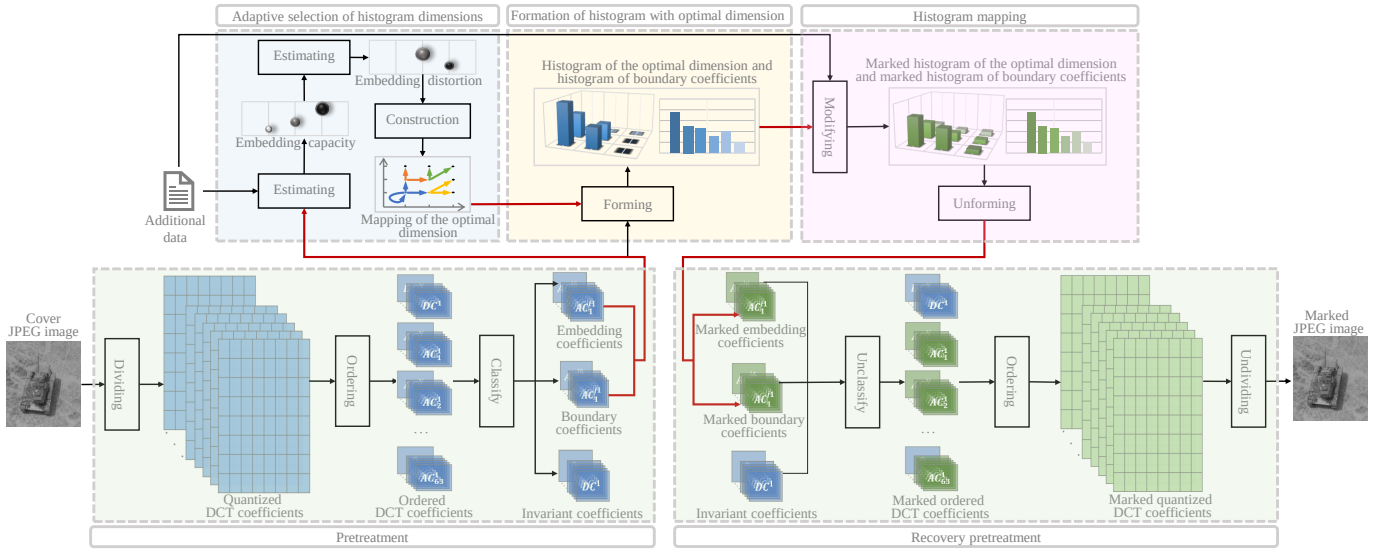


Fig. 4. The proposed framework of histogram modification with variable dimensionality.

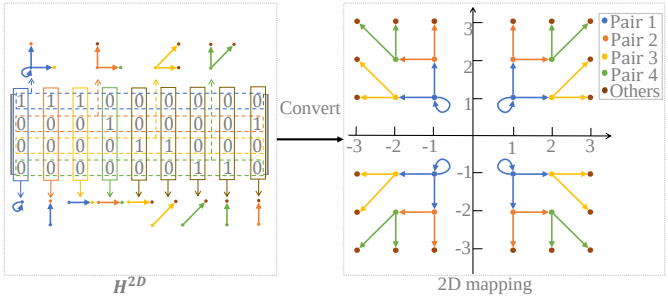


Fig. 5. The process of converting the 2D mapping matrix $\boldsymbol{H^{2D}}$ into the 2D mapping.

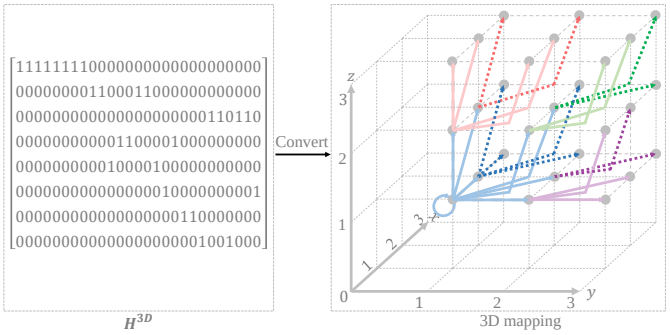


Fig. 6. The process of converting the 3D mapping matrix $\boldsymbol{H^{3D}}$ into the 3D mapping.

embedding efficiency and is defined as Eq.(2) and Eq.(3).

$$\mathrm{Z}(i) = \frac{1}{lw}\sum_{x=0}^{7}\sum_{y=0}^{7}\left[\frac{1}{4}Q_{k,g}G_{xy}(k,g)\right]^2 \tag{2}$$

$$G_{xy}(k,g) = \mathrm{c}(k)\mathrm{c}(g)\cos\frac{(2x+1)k\pi}{16}\cos\frac{(2y+1)g\pi}{16}, \tag{3}$$

where l and w refer to the length and width of an image, k and g refer to the row and column of the i th frequency in zigzag order within an 8×8 block, $Q_{k,g}$ refers to quantization step size at the corresponding position, $G_{xy}(k,g)$ is a function used to calculate DCT coefficients , $\mathrm{c}(x)$ is a coefficient function and is 1 in all case except $\mathrm{c}(0) = \sqrt{2}$. $N_i^2(\text{Pair } m)$ refers to the number of Pair m, Pair m refers to the embedding pair $(a,b) \in \{\pm1,\pm2\}^2$ with index $m(m \in \{1,2,3,4\})$, h_{mj}^{2D} refers to the (m,j) element of matrix $\boldsymbol{H^{2D}} \in \{0,1\}^{4\times9}$. The expression for $\boldsymbol{H^{2D}}$ is as Eq.(4)

$$\boldsymbol{H^{2D}} = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \end{bmatrix} \tag{4}$$

where $\boldsymbol{H^{2D}}$ represents a 2D mapping matrix that contains four embedding pairs (Pair 1 = (±1,±1), Pair 2 = (±1,±2), Pair 3 = (±2,±1) and Pair 4 = (±2,±2)) and five boundary pairs (including Pair 5 = (±3,±1), Pair 6 = (±3,±2), Pair 7 = (±3,±3), Pair 8 = (±2,±3) and Pair 9 = (±1,±3)). The row and column vectors of $\boldsymbol{H^{2D}}$ refer to the mapping relationships

and directions. For example, the first row refers to Pair 1 can map to Pair 1, Pair 2 or Pair 3, and the first column refers to Pair 1 maps to Pair 1. The process of constructing the 2D mapping from $\boldsymbol{H^{2D}}$ is illustrated in Fig. 5. N_i^3(Triple n) refers to the number of Triple n, Triple n refers to the embedding triple $(a,b,c) \in \{\pm1,\pm2\}^3$ with index n, h_{nj}^{3D} refers to the (n,j) element of matrix $\boldsymbol{H^{3D}} \in \{0,1\}^{8\times27}$, H^{3D} represents a 2D mapping matrix that contains eight embedding triples and nineteen boundary triples. The process of constructing a 3D mapping from $\boldsymbol{H^{3D}}$ is illustrated in Fig. 6. After frequency sorting, for each embedding pair $u=(a,b)$ and embedding triple $t=(a,b,c)$, the 2D and 3D histograms are defined as Eq.(5) and Eq.(6).

$$H^2(u) = \#\left\{1 \le i \le 63, 1 \le x,y \le N : AC_i^x = a, AC_i^y = b\right\}, \quad (5)$$

$$H^3(t) = \#\left\{1 \le i \le 63, 1 \le x,y,z \le N : AC_i^x = a, AC_i^y = b, AC_i^z = c\right\}, \quad (6)$$

where N refers to the number of blocks, x, y, z are the index of blocks, AC_i^x refers to the embedding coefficient of the ith frequency and xth block. Experiments were performed on all images in the database BOSSBase, and the number of pairs and triples was counted in different JPEG quality factors, as shown in Fig. 7. It can be seen that:

- Compared with existing methods, the proposed one yields a sharper histogram with more peak coefficient pairs which means a greater embedding capacity as more such pairs offer more space for data embedding, allowing more data bits to be inserted into the image.
- The peak non-zero coefficient pairs in the 2D histogram generated by our method are pair 1, followed by pair 2 and pair 3, and the least is pair 4.
- When the quality factor decreases from 90 to 60, the number of coefficient pairs(Pair 1 to Pair 4) generally decreases, which aligns with the theory that a higher compression ratio leads to lower image redundancy.

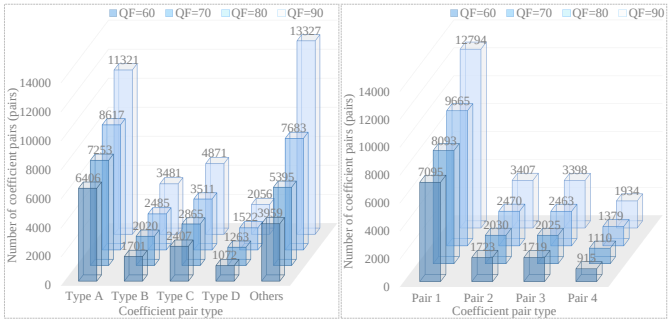
After histograms formation, the formed histograms will be mapped. There are four different transfer modes for the coefficient pairs, as shown in Fig. 5, which blue represents mode 1, orange represents mode 2, yellow represents mode 3, and green represents mode 4. According to He et al. [39], when a non-zero quantized AC coefficient is increased or decreased by one, the resulting image distortion is equal to the square of the quantization step. Thus, the embedding efficiency of the four transfer modes can be calculated as Eq.(7)–Eq.(10).

$$E_1 = \frac{\frac{1}{4}\times2+\frac{1}{4}\times2+\frac{1}{2}\times1}{\frac{1}{4}\times Q_1^2+\frac{1}{4}\times Q_2^2} = \frac{6}{Q_1^2+Q_2^2} \quad (7)$$

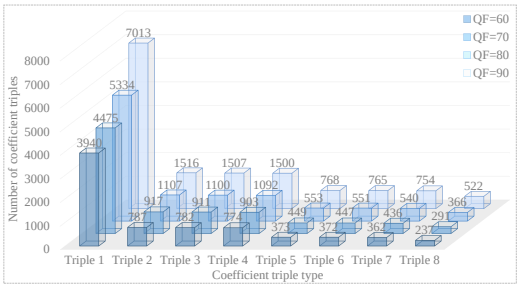
$$E_2 = \frac{\frac{1}{2}\times1+\frac{1}{2}\times1}{\frac{1}{2}\times(Q_1^2+Q_2^2)} = \frac{2}{Q_1^2+Q_2^2} \quad (8)$$

$$E_3 = \frac{\frac{1}{2}\times1+\frac{1}{2}\times1}{\frac{1}{2}\times Q_2^2+\frac{1}{2}\times(Q_1^2+Q_2^2)} = \frac{2}{Q_1^2+2\times Q_2^2} \quad (9)$$

$$E_4 = \frac{\frac{1}{2}\times1+\frac{1}{2}\times1}{\frac{1}{2}\times Q_1^2+\frac{1}{2}\times(Q_1^2+Q_2^2)} = \frac{2}{2\times Q_1^2+Q_2^2}, \quad (10)$$



(a)



(b)

Fig. 7. Distribution of pairs and triples for 10000 images from BOSSBase. (a)refers to pairs produced by the existing method(on the left) and the proposed method(on the right). (b)refers to triples produced by the proposed method.

where E_i(i=1,2,3,4) denotes the embedding efficiency of a coefficient pair using the transfer mode i, Q_1 and Q_2 denote the quantization steps corresponding to the frequency positions of the two coefficients of the pair(AC_1,AC_2), respectively. According to the Eq.(7)–Eq.(10) , the embedding efficiency of the transfer mode 1 is the highest, followed by the transfer mode 2, and the embedding efficiency of the transfer modes 3 and 4 is the lowest. Considering that the number of Pair 1(±1,±1) is the largest, the transfer mode 1 with the highest embedding efficiency is assigned to these coefficient pairs. Since the number of Pair 2(±1,±2) is usually greater than that of Pair 3(±2,±1), it is reasonable to assign pair 2 to mode 2, and assign pair 3 to mode 3. The fewest Pair 4(±2,±2) are assigned to mode 4. To provide a detailed explanation of the mapping process, for any embedding pair $u=(a,b) \in \{\pm1,\pm2\}^2$, mapping rules are as follows.

- when $|a|=1$ and $|b|=1$, then

$$u' = \begin{cases} (a,b), & \text{if } m=0 \\ (a,b+\psi(b)), & \text{if } m=10 \\ (a+\psi(a),b), & \text{if } m=11, \end{cases} \quad (11)$$

- when $|a|=1$ and $|b|=2$, then

$$u' = \begin{cases} (a,b+\psi(b)), & \text{if } m=0 \\ (a+\psi(a),b), & \text{if } m=1, \end{cases} \quad (12)$$

- when $|a|=2$ and $|b|=1$ or $|a|=2$ and $|b|=2$, then

$$u' = \begin{cases} (a+\psi(a),b+\psi(b)), & \text{if } m=0 \\ (a+\psi(a),b), & \text{if } m=1, \end{cases} \quad (13)$$

where u' refers to mapped embedding pair, m refers to embedding bits, function $\psi(x)(x \neq 0)$ is defined by the following formula.

$$\psi(x) = \begin{cases} -1, & \text{if } x < 0 \\ 1, & \text{if } x > 0 \end{cases} \tag{14}$$

For any embedding triple $t = (a, b, c) \in \{\pm 1, \pm 2\}^3$, mapping rules are as follows

- when $|a| = 1$ and $|b| = 1$ and $|c| == 1$, then

$$t' = \begin{cases} (a, b, c), \text{ if } m = 000 \\ (a, b, c + \psi(c)), & \text{if } m = 001 \\ (a + \psi(a), b, c), & \text{if } m = 010 \\ (a, b + \psi(b), c), & \text{if } m = 100 \\ (a, b + \psi(b), c + \psi(c)), & \text{if } m = 110 \\ (a + \psi(a), b + \psi(b), c), & \text{if } m = 011 \\ (a + \psi(a), b, c + \psi(c)), & \text{if } m = 101 \\ (a + \psi(a), b + \psi(b), c + \psi(c)), & \text{if } m = 111, \end{cases} \tag{15}$$

- when $|a| = 1$ and $|b| = 1$ and $|c| = 2$, then

$$t' = \begin{cases} (a, b, c + \psi(c)), & \text{if } m = 00 \\ (a, b + \psi(b), c + \psi(c)), & \text{if } m = 01 \\ (a + \psi(a), b, c + \psi(c)), & \text{if } m = 10 \\ (a + \psi(a), b + \psi(b), c + \psi(c)), & \text{if } m = 11, \end{cases} \tag{16}$$

- when $|a| = 2$ and $|b| = 1$ and $|c| = 1$, then

$$t' = \begin{cases} (a + \psi(a), b, c), & \text{if } m = 00 \\ (a + \psi(a), b + \psi(b), c), & \text{if } m = 01 \\ (a + \psi(a), b, c + \psi(c)), & \text{if } m = 10 \\ (a + \psi(a), b + \psi(b), c + \psi(c)), & \text{if } m = 11, \end{cases} \tag{17}$$

- when $|a| = 1$ and $|b| = 2$ and $|c| = 1$, then

$$t' = \begin{cases} (a, b + \psi(b), c + \psi(c)), & \text{if } m = 11 \\ (a, b + \psi(b), c), & \text{if } m = 0 \\ (a + \psi(a), b + \psi(b), c), & \text{if } m = 10, \end{cases} \tag{18}$$

- when $|a| = 1$ and $|b| = 2$ and $|c| = 2$, then

$$t' = \begin{cases} (a, b + \psi(b), c + \psi(c)), & \text{if } m = 1 \\ (a + \psi(a), b + \psi(b), c + \psi(c)), & \text{if } m = 0, \end{cases} \tag{19}$$

- when $|a| = 2$ and $|b| = 2$ and $|c| = 1$, then

$$t' = \begin{cases} (a, b + \psi(b), c + \psi(c)), & \text{if } m = 1 \\ (a + \psi(a), b + \psi(b), c), & \text{if } m = 0, \end{cases} \tag{20}$$

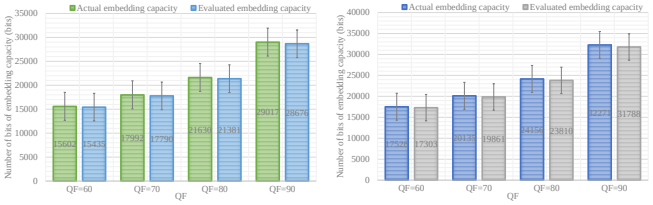
- when $|a| = 2$ and $|b| = 1$ and $|c| = 2$, then

$$t' = \begin{cases} (a + \psi(a), b, c + \psi(c)), & \text{if } m = 1 \\ (a + \psi(a), b + \psi(b), c + \psi(c)), & \text{if } m = 0, \end{cases} \tag{21}$$

- when $|a| = 2$ and $|b| = 2$ and $|c| = 2$, then

$$t' = \begin{cases} (a + \psi(a), b + \psi(b), c), & \text{if } m = 1 \\ (a + \psi(a), b + \psi(b), c + \psi(c)), & \text{if } m = 0, \end{cases} \tag{22}$$

where t' refers to mapped embedding triple.



(a) (b)

Fig. 8. Average embedding capacity of 10000 images from BOSSBase. (a) refers to actual and evaluated embedding capacity in 2D histograms. (b) refers to actual and evaluated embedding capacity in 3D histograms.

2) Adaptive dimension selection strategy: Prior to introducing the strategy, it is necessary to define 1D histograms for the two types of coefficients involved in data embedding, namely embedding coefficients and boundary coefficients, as they directly or indirectly participate in the embedding process. For any coefficient $e \in \{1, 2\}$, the embedding histogram is defined as Eq.(23).

$$H_e^1(e) = \#\left\{1 \leq i \leq 63N : \left|AC^i\right| = e\right\}, \tag{23}$$

where AC^i refers to the ith AC. For any coefficient $e \in \{3, 4, \ldots, 1023\}$, the boundary histogram is defined by the following Eq.(24).

$$H_b^1(e) = \#\left\{1 \leq i \leq 63N : \left|AC^i\right| = e\right\} \tag{24}$$

The proposed adaptive dimension selection strategy comprises three steps, including capacity estimation, distortion estimation, and construction. Firstly, the capacity of the 2D and 3D histograms will be evaluated to determine whether they are sufficient to accommodate the given payload P. Combining with the mapping rules, the maximum capacity can be defined as Eq.(25) and Eq.(26).

$$C_{max}^2 = \log_2 3N^2(\text{Pair } 1) + \log_2 \sum_{m=2}^{4} \left[N^2(\text{Pair } m)\right] \tag{25}$$

$$C_{max}^3 = \log_2 8N^3(\text{Triple } 1) + \log_2 4 \sum_{n=2}^{3} \left[N^3(\text{Triple } n)\right] + \log_2 3N^3(\text{Triple } 4) + \log_2 2 \sum_{n=5}^{8} \left[N^3(\text{Triple } n)\right], \tag{26}$$

where C_{max}^2 and C_{max}^3 refer to the maximum capacity of the 2D and 3D histograms, $N^2(\text{Pair } m)$ and $N^3(\text{Triple } n)$ refer to the number of Pair $m(m \in \{1, 2, 3, 4\})$ and Triple $n(n \in \{1, 2, ..., 8\})$. $\log_2 q$ refers to the number of bits that can be represented by q embedding directions, for example, both Pair 1 and Triple 4 have 3 embedding directions, and thus can represent $\log_2 3$ bits. $N^2(\text{Pair } m)$ and $N^3(\text{Triple } n)$ can be obtained either by directly counting the number of Pair m and Triple n in the cover image, or by estimation. For any Pair $m = (a, b) \in \{\pm 1, \pm 2\}^2$, $N^2(\text{Pair } m)$ can be estimated as Eq.(27).

$$N^2(\text{Pair } m) = \begin{cases} \left\lfloor \frac{K}{2} \right\rfloor \frac{\varphi^1\left(H_e^1(a)\right)}{\varphi^1(K)}, & \text{if } a = b \\ \left\lfloor \frac{K}{2} \right\rfloor \frac{H_e^1(a) H_e^1(b)}{\varphi^1(K)}, & \text{if } a \neq b, \end{cases} \tag{27}$$

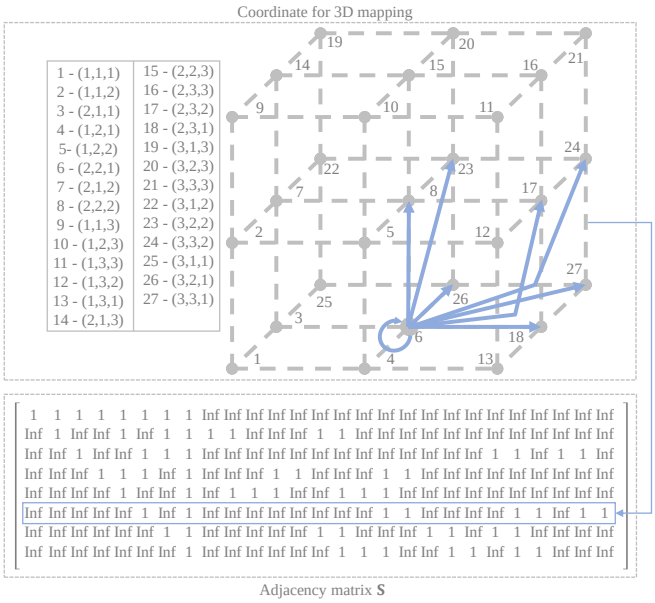


Fig. 9. Taking all mapping directions of Triple 6 as an example, the process of mapping directions represented by adjacency matrix $\boldsymbol{S}$ is shown.

where $\lfloor \cdot \rfloor$ refers to the floor function, $K = H_e^1(1) + H_e^1(2)$, $\varphi^1(x) = x(x-1)$. For any Triple $n = (a, b, c) \in \{\pm 1, \pm 2\}^3$, $N^3(\text{Triple } n)$ can be estimated as Eq.(28).

$$N^3(\text{Triple } n) = \begin{cases} \lfloor \frac{K}{3} \rfloor \frac{\varphi^2\left(H_e^1(a)\right)}{\varphi^2(K)}, & \text{if } c_1 \\ \lfloor \frac{K}{3} \rfloor \frac{\varphi^1\left(H_e^1(1)\right) H_e^1(2)}{\varphi^2(K)}, & \text{if } c_2 \\ \lfloor \frac{K}{3} \rfloor \frac{\varphi^1\left(H_e^1(2)\right) H_e^1(1)}{\varphi^2(K)}, & \text{if } c_3, \end{cases} \tag{28}$$

where $\varphi^2(x) = \varphi^1(x)(x-2)$, c_1 refers to $a = b = c$, c_2 refers to $a = b = 1 \neq c$ or $a = c = 1 \neq b$ or $c = b = 1 \neq a$, c_3 refers to $a = b = 2 \neq c$ or $a = c = 2 \neq b$ or $c = b = 2 \neq a$. To compare the capacity obtained by estimating and by directly counting, experiments were carried out on all images in the database BOSSBase using both methods. From the experimental results, shown in Fig. 8, it can be observed that the capacity through the proposed estimated method can replace the capacity obtained through statistics, within an acceptable error range. Subsequently, we need to evaluate whether the embedding capacities of the two dimensions derived from Eq.(27) and Eq.(28) can meet the payload P. The dimension that meets the payload requirement will be selected as the optimal embedding dimension. If both dimensions satisfy this constraint, we will calculate their histogram distortions and choose the dimension with the lowest distortion as the most suitable one. Specifically, according to the embedding efficiency order and coefficient mapping rules obtained in Section III, data will be embedded into the top-R frequencies, where R represents the minimum number of frequencies required to fully embed the data. After the data embedding process is completed, the distortion will be calculated. Based on the bits to be embedded, the distortion

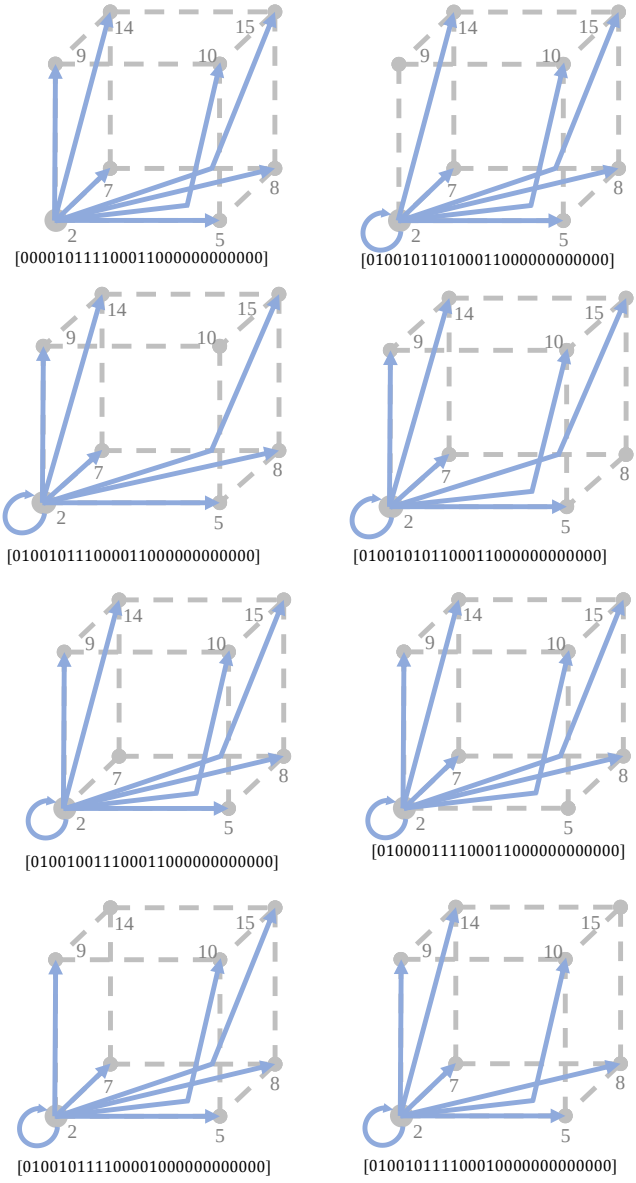


Fig. 10. Taking Triple 2 as an example, the process of obtaining M_{c2} from $W_{c2}^T X_{c2} = 7$ is shown, where M_{c2} , W_{c2} and X_{c2} represent the 2nd row of matrices $\boldsymbol{M}$, $\boldsymbol{W}$ and $\boldsymbol{X}$, respectively.

can be defined as shown in Eq.(29).

$$D_e = \sum_{i=1}^{N} \frac{1}{64} \sum_{x=0}^{7} \sum_{y=0}^{7} \Delta f^2(x, y), \tag{29}$$

where D_e refers to evaluated distortion, N refers to the number of blocks, x and y are the index of pixels, $\Delta f(x, y) = f(x, y) - f^*(x, y)$ refers to the difference between the original pixel $f(x, y)$ and the modified pixel $f^*(x, y)$. Finally, the corresponding optimal mapping can be constructed based on the optimal dimension.

C. High-Dimensional Mapping Strategy

In general, a 2D mapping in JPEG reversible data hiding consists of a maximum of four mapping directions for each

Algorithm 1 Proposed Optimization Algorithm

Input: solution set of mapping matrix M, distortion matrix $\boldsymbol{D}$, set of eligible number matrix X.

Output: Optimal mapping matrices H^{opt}.

Initializing $H^{opt} = []$;
Get $\{M_{X^1}, M_{X^2}, ..., M_{X^{size(X)}}\}$;
for $i = 1, 2, ..., size(X)$ **do**
  Find the set of mapping matrices M_{X^i} obtained from the number matrix X^i;
end for
Find mapping matrices with the smallest global distortion in each M_{X^i} and assign to M_0^D;
for $i = 1, 2, ..., size(X)$ **do**
  $temp_i = \min_{\boldsymbol{M^h} \in M_{X^i}} sum(\boldsymbol{M^h} \cdot \boldsymbol{D})$;
end for
$M_0^D = \{temp_1, temp_2, ..., temp_{size(X)}\}$;
Find mapping matrices with the largest local capacity in M_0^D and assign to M_8^C;
$M_0^C = M_0^D$;
for $i = 1, 2, ..., 8$ **do**
  $M_i^C = \max_{\boldsymbol{M^h} \in M_{i-1}^C} \sum_{j=1}^{27} \log_2 m_{ij}^h$;
end for
Find mapping matrices with the smallest local distortion in M_8^C and assign to M_8^D;
$M_0^D = M_8^C$;
for $i = 1, 2, ..., 8$ **do**
  $M_i^D = \min_{\boldsymbol{M^h} \in M_{i-1}^D} sum(M_{ci}^i \cdot D_{ci})$;
end for
Find optimal mapping matrices;
$H^{opt} = M_8^D$;
return H^{opt}.

pair, while a 3D mapping involves a greater number of mappings for each triple. It is worth noting that existing 3D mappings typically have their centers positioned at (0,0,0), whereas the center of 3D mappings to be constructed in this paper is (±1,±1,±1). Therefore, 3D mappings for JPEG RDH cannot be obtained by improving 2D mappings or by directly using existing 3D mappings. To address these issues, this paper proposes a high-dimensional mapping strategy. According to the absolute value set of embedding coefficients {1,2}, there are eight embedding triples used for data embedding, namely (1,1,1), (1,1,2), (2,1,1), (1,2,1), (1,2,2), (2,2,1), (2,1,2) and (2,2,2). However, an additional nineteen triples that cannot be used for data embedding are required, and they are called boundary triples. To facilitate further discussion, the numbering of triples is shown in Fig.9. It is evident that each embedding triple can be assigned a different number of mapping directions. For instance,in Fig.2, (0,0,0) is assigned to two mapping directions, while (0,0,1) is only assigned to one. In this paper, $\boldsymbol{X} \in \mathbb{Z}^{8\times8}$ is used to represent the number of mapping directions assigned to each embedding triple, and is called quantity matrix. In order to find all quantity matrices, this paper analyzes the maximum number of mapping directions that each embedding triple can have. Taking image quality into consideration, all embedding triples

TABLE I
AVERAGE EMBEDDING CAPACITY (BITS) OF USC-SIPI UNDER DIFFERENT QFS

Method	QF=60	QF=70	QF=80	QF=90	Avg
Huang *et al.* [24]	24073	27813	34160	46480	33131
Wedaj *et al.* [25]	24073	27813	34160	46480	33131
He *et al.* [30]	24073	27813	34160	46480	33131
Li *et al.* [34]	30013	35096	43020	61482	42402
Chen *et al.* [36]	31788	36799	44588	62805	43995
2D	32710	37990	46191	63047	44986
Proposed	**36403**	**42034**	**50952**	**71919**	**50327**

TABLE II
AVERAGE EMBEDDING CAPACITY (BITS) OF KODAK UNDER DIFFERENT QFS

Method	QF=60	QF=70	QF=80	QF=90	Avg
Huang *et al.* [24]	22741	26235	32597	43234	31202
Wedaj *et al.* [25]	22741	26235	32597	43234	31202
He *et al.* [30]	22741	26235	32597	43234	31202
Li *et al.* [34]	28185	32172	38754	51932	37761
Chen *et al.* [36]	29637	33681	40118	53414	39212
2D	30560	34912	41684	53802	40239
Proposed	**34253**	**38957**	**46446**	**61675**	**45333**

cannot map in the direction smaller than their absolute value, and each of their coordinates can only change by at most 1. Calculations reveal that each embedded triplet can have up to eight mapping directions. This paper represents all the mapping directions with a matrix $\boldsymbol{S} \in \{1, \text{Inf}\}^{8\times27}$, which is called adjacency matrix. For example, Triple $6 = (2, 2, 1)$ can map to Triple 6 referring to $s_{6,6} = 1$, while Triple 6 cannot map to Triple 1 referring to $s_{6,1} = \text{Inf}$, where $s_{i,j}$ refers to the (i, j) elements of $\boldsymbol{S}$. $\boldsymbol{S}$ and all mapping directions that can be assigned to Triple 6, are shown in Fig.9. In theory, there can be $(C_8^1 C_8^2 \dots C_8^8)^8$ quantity matrices, but only quantity matrices that satisfy constraints are useful, where C_n^m refers to the number of combinations of $m(m \le n)$ elements that can be taken from n different elements. The three constraints are defined as Eq.(30-32).

$$1 \le W_{cj}^T X_{cj} \le 8, \tag{30}$$

$$\text{sum}(\boldsymbol{X}) = 27, \text{sum}(\boldsymbol{W} \cdot \boldsymbol{X}) = 27, \tag{31}$$

$$W_{c1}^T X_{c1} \ge W_{c2}^T X_{c2} \ge \cdots \ge W_{c8}^T X_{c8}, \tag{32}$$

where W_{cj}^T refers to the transpose of the jth row of $\boldsymbol{W}$, $\boldsymbol{W} \in \{0, 1\}^{8\times8}$ refers to the identity matrix with all elements being 0 except for 1 on the diagonal, X_{cj} refers to the jth row of $\boldsymbol{X}$. $\text{sum}\left(\boldsymbol{A}^{r\times c}\right) = \sum_{i=1}^{r}\sum_{j=1}^{c} a_{i,j}$, $a_{i,j}$ refers to the (i, j) elements of matrix $\boldsymbol{A}$. $\boldsymbol{A}\cdot\boldsymbol{B}$ refers to Dot Product of matrix $\boldsymbol{A}$ and $\boldsymbol{B}$. In addition to the number of mapping directions, it is also necessary to determine which specific mapping directions are assigned. To represent this information, $\boldsymbol{M} \in \{0, 1\}^{8\times27}$ is used and is called mapping matrix, which satisfies $\text{sum}(\boldsymbol{M} \cdot \boldsymbol{S}) \ne \text{Inf}$, and Eq.(33).

$$\sum_{i=1}^{8} m_{ij} = 1, (j = 1, 2, \cdots, 27), \tag{33}$$

TABLE III
AVERAGE PSNR OF JPEG (QF=60, 70, 80 AND 90) COMPRESSED FROM FOUR FREQUENTLY USED IMAGES

Image	Method	QF=60/payload(bits)			QF=70/payload(bits)			QF=80/payload(bits)			QF=90/payload(bits)		
		13000	14000	15000	13000	14000	15000	13000	14000	15000	13000	14000	15000
Man	Huang *et al.* [24]	47.58	47.24	46.84	49.58	49.18	48.82	52.03	51.61	51.23	55.57	55.12	54.70
	Wedaj *et al.* [25]	46.74	46.05	45.19	48.43	47.70	46.85	50.72	50.11	49.48	53.49	52.98	52.46
	He *et al.* [30]	47.86	47.45	47.11	49.81	49.41	49.06	52.05	51.65	51.28	55.76	55.28	54.86
	Li *et al.* [34]	47.63	47.26	46.90	49.80	49.43	49.05	52.33	51.92	51.60	56.43	55.95	55.54
	Chen *et al.* [36]	48.78	48.33	47.97	50.70	50.28	49.93	52.97	52.52	52.16	56.72	56.23	55.85
	2D	48.84	48.45	48.09	50.79	50.37	49.98	53.03	52.60	52.22	56.54	55.99	55.72
	Proposed	**48.84**	**48.45**	**48.09**	**50.79**	**50.37**	**49.98**	**53.03**	**52.60**	**52.22**	**56.85**	**56.37**	**55.97**
Pepper	Huang *et al.* [24]	39.32	38.67	38.01	42.29	41.76	41.25	45.15	44.63	44.22	47.63	47.16	46.73
	Wedaj *et al.* [25]	39.58	39.04	38.47	42.33	41.71	41.49	45.42	44.41	44.27	48.31	48.13	48.05
	He *et al.* [30]	39.68	39.02	38.38	42.85	42.28	41.74	45.75	45.26	44.82	49.43	48.97	48.54
	Li *et al.* [34]	38.19	N/A	N/A	41.86	41.10	40.42	45.45	44.93	44.42	49.68	49.19	48.77
	Chen *et al.* [36]	39.27	38.43	N/A	42.63	41.93	41.23	46.14	45.61	45.06	50.09	49.64	49.16
	2D	39.57	38.69	37.54	42.85	42.18	41.49	46.27	45.69	45.15	49.49	48.98	48.62
	Proposed	**39.87**	**39.25**	**37.59**	**42.87**	**42.31**	**41.73**	**46.27**	**45.69**	**45.15**	**50.18**	**49.76**	**49.32**
Boat	Huang *et al.* [24]	38.29	37.78	37.23	40.48	39.97	39.48	42.80	42.33	41.90	46.01	45.55	45.16
	Wedaj *et al.* [25]	38.58	38.13	37.70	40.58	40.24	39.55	43.12	42.60	42.38	46.70	46.48	46.17
	He *et al.* [30]	38.77	38.24	37.75	41.04	40.50	40.01	43.84	43.33	42.86	47.93	47.45	47.01
	Li *et al.* [34]	38.25	37.62	37.03	40.62	40.03	39.44	43.57	43.05	42.51	48.26	47.71	47.21
	Chen *et al.* [36]	38.87	38.23	37.63	41.19	40.61	40.04	44.13	43.58	43.03	48.60	48.01	47.53
	2D	39.10	38.48	37.93	41.37	40.80	40.23	43.97	43.47	43.01	48.01	47.60	47.11
	Proposed	**39.10**	**38.48**	**37.93**	**41.37**	**40.85**	**40.29**	**44.24**	**43.74**	**43.19**	**48.67**	**48.20**	**47.67**
House	Huang *et al.* [24]	36.87	36.40	35.87	39.02	38.50	37.98	41.81	41.28	40.76	47.02	46.40	45.80
	Wedaj *et al.* [25]	37.40	36.97	36.52	39.34	38.90	38.43	41.90	41.38	40.90	45.97	45.55	45.19
	He *et al.* [30]	37.75	37.22	36.70	39.56	38.99	38.48	42.41	41.83	41.31	47.49	46.84	46.20
	Li *et al.* [34]	37.12	36.57	36.06	39.23	38.63	38.05	41.98	41.36	40.77	46.94	46.27	45.63
	Chen *et al.* [36]	37.71	37.13	36.61	39.81	39.19	38.60	42.50	41.85	41.22	47.40	46.67	46.09
	2D	37.91	37.33	36.83	40.02	39.44	38.86	42.75	42.10	41.51	47.61	46.96	46.30
	Proposed	**37.91**	**37.33**	**36.83**	**40.02**	**39.44**	**38.86**	**42.75**	**42.10**	**41.51**	**47.61**	**46.96**	**46.30**

TABLE IV
AVERAGE PSNR (DB) OF USC-SIPI UNDER DIFFERENT PAYLOADS (BITS)

QF	Method	7000	9000	11000	13000	15000
60	Huang *et al.* [24]	43.48	42.03	40.82	39.77	38.79
	Wedaj *et al.* [25]	43.18	41.74	40.79	39.77	38.83
	He *et al.* [30]	43.98	42.51	41.28	40.19	39.21
	Li *et al.* [34]	43.85	42.32	41.02	40.05	39.27
	Chen *et al.* [36]	44.57	42.99	41.67	40.52	39.84
	2D	44.70	43.16	41.84	40.79	39.88
	Proposed	**44.78**	**43.24**	**41.95**	**40.97**	**39.98**
70	Huang *et al.* [24]	46.48	44.99	43.72	42.63	41.61
	Wedaj *et al.* [25]	45.98	44.31	43.42	42.47	41.35
	He *et al.* [30]	47.02	45.47	44.19	43.06	42.02
	Li *et al.* [34]	46.67	45.12	43.75	42.48	41.24
	Chen *et al.* [36]	47.49	45.89	44.47	43.19	41.98
	2D	47.76	46.18	44.74	43.51	42.44
	Proposed	**47.84**	**46.26**	**44.89**	**43.71**	**42.64**
80	Huang *et al.* [24]	49.19	47.72	46.50	45.40	44.45
	Wedaj *et al.* [25]	48.43	46.86	46.09	45.12	44.10
	He *et al.* [30]	49.8	48.28	47.03	45.92	44.94
	Li *et al.* [34]	49.77	48.14	46.81	45.66	44.59
	Chen *et al.* [36]	50.4	48.76	47.4	46.25	45.14
	2D	50.61	49.04	47.65	46.52	45.51
	Proposed	**50.69**	**49.12**	**47.81**	**46.68**	**45.66**
90	Huang *et al.* [24]	50.92	49.45	48.24	47.22	46.31
	Wedaj *et al.* [25]	50.03	48.88	47.91	47.09	46.28
	He *et al.* [34]	51.80	50.33	49.09	48.05	47.14
	Li *et al.* [36]	52.31	50.81	49.57	48.50	47.53
	Chen *et al.* [26]	52.66	51.18	49.92	48.83	47.85
	2D	52.80	51.32	50.03	48.94	47.98
	Proposed	**52.83**	**51.35**	**50.11**	**49.03**	**48.08**

TABLE V
AVERAGE PSNR (DB) OF KODAK UNDER DIFFERENT PAYLOADS (BITS)

QF	Method	7000	9000	11000	13000	15000
60	Huang *et al.* [24]	44.46	43.01	41.8	40.75	39.77
	Wedaj *et al.* [25]	44.16	42.72	41.57	40.57	39.66
	He *et al.* [30]	45.05	43.49	42.26	41.17	40.19
	Li *et al.* [34]	44.83	43.3	42	40.94	40.05
	Chen *et al.* [36]	45.48	43.89	42.57	41.43	40.4
	2D	45.65	44.07	42.76	41.56	40.5
	Proposed	**45.65**	**44.08**	**42.84**	**41.69**	**40.66**
70	Huang *et al.* [24]	46.52	45.01	43.77	42.63	41.64
	Wedaj *et al.* [25]	46.02	44.52	43.44	42.50	41.62
	He *et al.* [30]	45.99	45.45	44.17	43.05	41.98
	Li *et al.* [34]	46.70	45.09	43.81	42.59	41.41
	Chen *et al.* [36]	47.47	45.88	44.59	43.38	42.22
	2D	47.85	46.25	44.91	43.71	42.62
	Proposed	**47.85**	**46.25**	**44.94**	**43.93**	**42.9**
80	Huang *et al.* [24]	49.20	47.72	46.52	45.41	44.46
	Wedaj *et al.* [25]	48.39	46.87	45.92	45.11	44.19
	He *et al.* [30]	49.78	48.25	47.01	45.93	44.91
	Li *et al.* [34]	49.85	48.22	46.85	45.7	44.63
	Chen *et al.* [36]	50.41	48.78	47.38	46.24	45.13
	2D	50.65	49.11	47.70	46.48	45.39
	Proposed	**50.70**	**49.13**	**47.83**	**46.73**	**45.69**
90	Huang *et al.* [24]	53.63	52.15	50.94	49.92	49.04
	Wedaj *et al.* [25]	52.63	51.44	50.51	49.71	49.04
	He *et al.* [30]	54.39	52.93	51.69	50.65	49.74
	Li *et al.* [34]	54.81	53.29	51.92	50.73	49.73
	Chen *et al.* [36]	55.16	53.66	52.27	51.06	50.04
	2D	55.36	53.85	52.51	51.29	50.21
	Proposed	**55.36**	**53.85**	**52.56**	**51.45**	**50.45**

where $m_{i,j}$ refers to the (i, j) elements of $\boldsymbol{M}$. $X_{c2} = [07000000]$ is used to explain how $\boldsymbol{M}$ can be obtained from the $\boldsymbol{X}$, as shown in Fig. 10. All quantity matrices and mapping matrices are denoted by the sets X and M, respectively. The optimization algorithm shown in **Algorithm 1** progressively approximates the mapping matrix that is optimal in terms

of both capacity and distortion. The matrix $\boldsymbol{H^{3D}}$ mentioned earlier in this paper is the optimal mapping matrix $\boldsymbol{H^{opt}}$ found by **Algorithm 1**. In **Algorithm 1**, C_i^{M} refers to the capacity of $\boldsymbol{M^i} \in M$ and is defined by Eq.(34). C_z^{T} refers to the capacity of Triple $z(1 \le z \le 8)$ in $\boldsymbol{M^i}$ and is defined by Eq.(35).

$$C_i^{\mathrm{M}} = \sum_{z=1}^{8} \log_2 \sum_{j=1}^{27} m_{zj} \tag{34}$$

$$C_z^{\mathrm{T}} = \log_2 \sum_{j=1}^{27} m_{zj}, \tag{35}$$

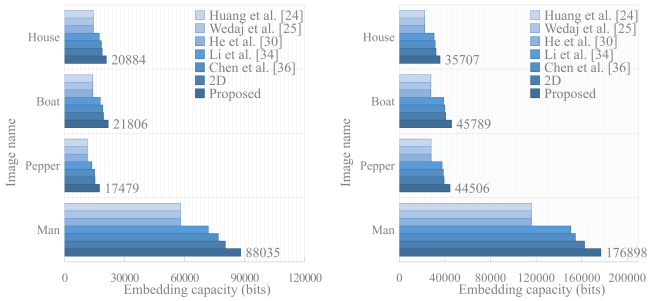
It is not difficult to observe that the mapping matrices $M_{X^k} \subseteq M$ derived from the same quantity matrix $\boldsymbol{X^k} \in X$ have the same $C_i^{\mathrm{M}}(\boldsymbol{M^i} \in M_{X^k})$ and $C_z^{\mathrm{T}}(1 \le z \le 8)$, where $\boldsymbol{X^k}$ refers to the kth quantity matrix in the set X, M_{X^k} refers to the set of all mapping matrices obtained from $\boldsymbol{X^k}$ in M. D_i^{M} refers to the distortion of $\boldsymbol{M^i}$ and is defined as

$$D_i^{\mathrm{M}} = \mathrm{sum}\left(\boldsymbol{M^i} \cdot \boldsymbol{D}\right), \tag{36}$$

where $\boldsymbol{D} \in \{0,1,2,3\}^{8\times 27}$ refers to distortion matrix and is determined by the mapping direction. As shown in Table I , it displays d_{zj} for each value of Triple j when Triple $z = (a,b,c)$ is mapped to Triple j where d_{zj} refers to the (i, j) elements of D. D_z^{T} refers to the capacity of Triple z in $\boldsymbol{M^i}$ and is defined as

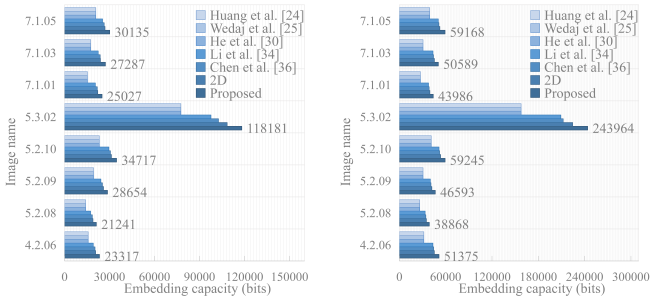
$$D_z^{\mathrm{T}} = \mathrm{sum}\left(M_{cz}^i \cdot D_{cz}\right), \tag{37}$$

where M_{cz}^i and D_{cz} refer to the zth row vector of $\boldsymbol{M^i}$ and $\boldsymbol{D}$, respectively.



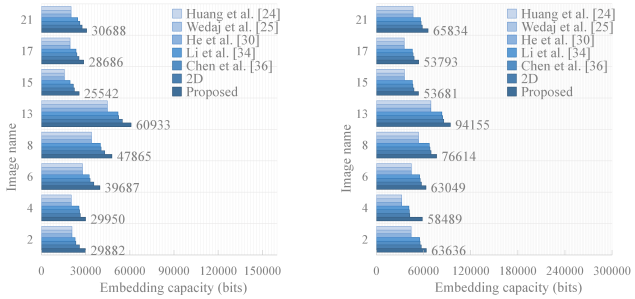
(a) QF=60 (b) QF=90

Fig. 11. Average embedding capacity of four frequently used images.



(a) QF=60 (b) QF=90

Fig. 12. Embedding capacity of eight images from dataset USC-SIPI.



(a) QF=60 (b) QF=90

Fig. 13. Embedding capacity of eight images from dataset KODAK.

D. Process of embedding, extraction and recovery

In this section, the processes of data embedding, embedded data extraction, and cover image recovery are presented. In most RDH schemes, side information is necessary to ensure their reversibility, and should be saved in marked JPEG images. The length of auxiliary information required for data extraction in our scheme is 94 bits, including length of the embedded data(18 bits), frequency selection matrix(63 bits), the number of selected blocks(12 bits) and the selected dimension(1 bit, where '0' means 2D, and '1' means 3D). This 94-bit side information will be embedded into the first 94 DC coefficients using the traditional LSB replacement method. And the original LSBs of the modified DC coefficients need to be embedded into the AC coefficients along with the secret data.

1) Data Embedding: As illustrated in Fig. 4, the data embedding workflow comprises four consecutive stages: Pretreatment, Histogram Formation, Histogram Mapping, and Recovery Pretreatment.

Step 1: Partially decode the cover JPEG image to obtain the 8×8 quantized DCT coefficient blocks.

Step 2: Append the original LSBs of the first 94 DC coefficients to the secret data.

Step 3: Sort the quantized coefficient blocks according to the number of zero AC coefficients.

Step 4: Classify embedding coefficients, boundary coefficients directly from DCT coefficients.

Step 5: Estimate the embedding efficiency of each frequency position j(j = 1,2...,63) using Eq.(1). Then, according to the estimated embedding efficiency, 63 frequency positions are sorted in descending order.

Step 6: Use additional data, embedding coefficients, and boundary coefficients to estimate the embedding capacity and embedding distortion according Eq.(27-28) and Eq.(29).

Step 7: Select the optimal dimension based on the criteria of maximizing capacity and minimizing distortion.

Step 8: Embed data into the coefficient pairs at the top-R frequency positions of Nb blocks using the proposed method of coefficient pairing and mapping presented in Section III.

Step 9: Embed side information into the first 94 DC coefficients with the LSB replacement method.

Step 10: Entropy-encode the modified DCT coefficients to obtain a marked JPEG image.

2) Data extraction and cover image recovery: Process of extraction and recovery can be considered as the inverse of the embedding process, whereby the dimension, frequencies, and blocks used during the embedding stage can be directly retrieved from the auxiliary information. As a result, there is no requirement for additional computations during the extraction and recovery process.

Step 1: Partially decode the marked JPEG image to obtain the quantized DCT coefficients.

Step 2: Extract the embedded side information including 63 bits for marking R frequency positions, the length of data bits from the LSBs of the first 94 DC coefficients.

Step 3: Sort the quantized coefficient blocks according to the number of zero AC coefficients.

Step 4: Pair the coefficients used for data embedding according to the *R* frequency positions and *Nb* blocks.

Step 5: Refer to the coefficient pair mapping shown in Fig. 5 and Fig. 6, extract the data embedded and recover the original DCT coefficients.

Step 6: Entropy-encode the recovered DCT coefficients to obtain the cover image.

It is critical to emphasize that in the embedding process, Steps 3-5 (frequency and block sorting) utilize the same sorting strategy as Steps 3-5 in the extraction process. This ensures identical ordering of blocks and frequencies during both embedding and extraction. Furthermore, Step 8 of embedding(coefficient pair mapping) and Step 5 of extraction employ the same mapping strategy(illustrated in Fig. 5 and 6), but the extraction step performs the inverse operation. Collectively, these mechanisms ensure that all modification operations within the MHMF framework-namely the coefficient pair mapping and DC LSB substitution–constitute fully reversible transformations. Moreover, the auxiliary information contains all necessary parameters required for the decoding of the marked image.

E. Complexity analysis

The main code modules of this research consist of image blocking and coefficient preprocessing, 3D histogram construction and dimension selection, high-dimensional mapping matrix generation, as well as data embedding and Huffman dynamic encoding. Each module is analyzed as follows.

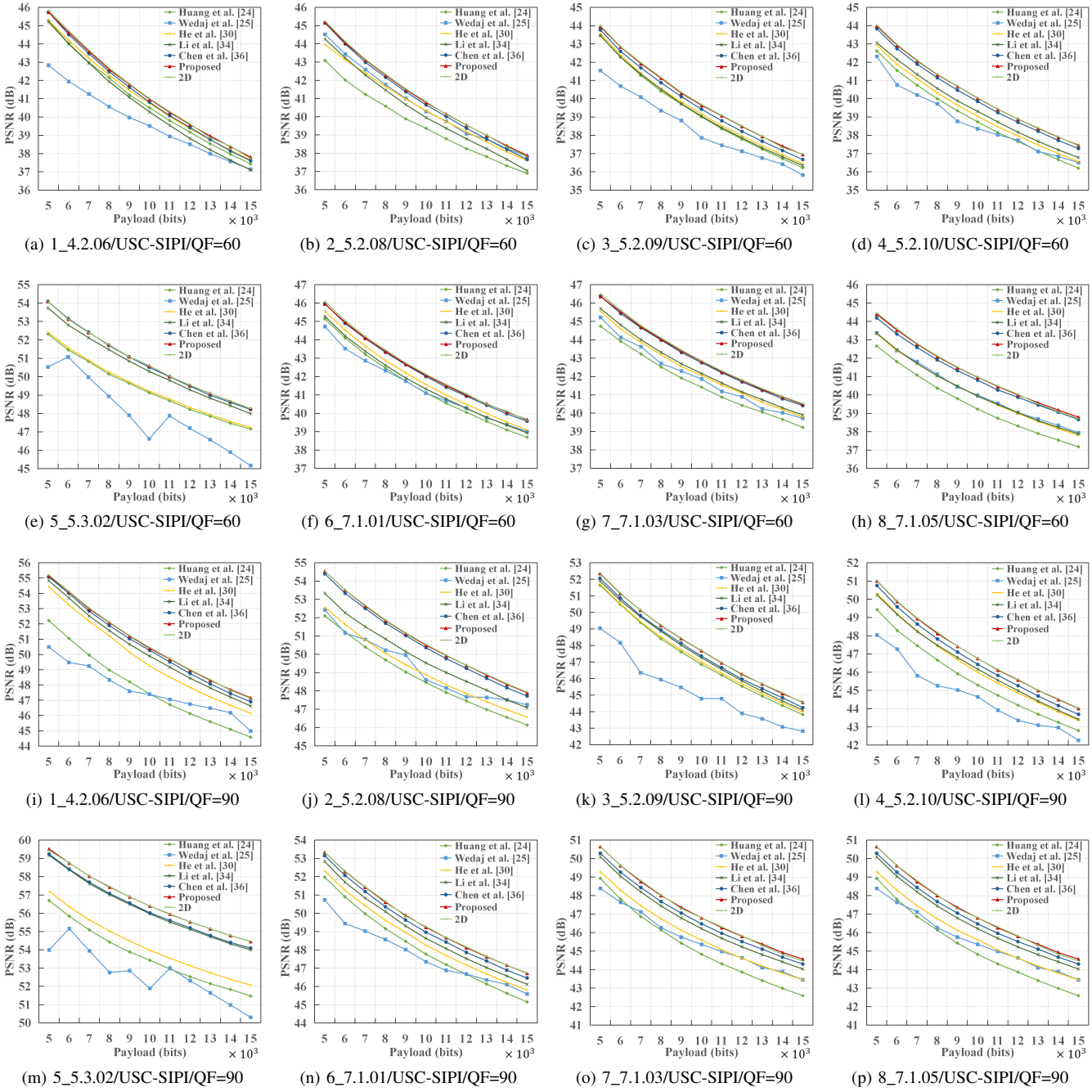
In the image blocking and coefficient preprocessing module, the algorithm first decompresses the JPEG image and extracts DCT coefficients. It converts the 8×8 DCT blocks into one-dimensional sequences through Zigzag scanning, and the time complexity of this operation is $O(B)$, where B is the number of image blocks. Subsequently, the image blocks are sorted according to the number of zero AC coefficients or embedding efficiency, and a sorting algorithm is adopted to make the time complexity of this step reach $O(B \times logB)$. Finally, the DCT coefficients are classified to distinguish embedded coefficients, boundary coefficients, and invariant coefficients. This process needs to traverse all coefficients, and the time complexity is $O(B)$. Comprehensively, the time complexity of the preprocessing stage is mainly determined by the block sorting step, and the overall performance is $O(B \times logB)$.

The 3D histogram construction and dimension selection module is responsible for counting the distribution characteristics of coefficient pairs or triples and selecting the optimal embedding dimension. For the construction of the 2D histogram, the algorithm counts the coefficients of cross-block pairs with the same frequency, and the time complexity is $O(B \times Z)$, where Z represents the average number of non-zero AC coefficients per block. The construction of the 3D histogram theoretically has a time complexity of $O(B \times Z^2)$, but it can be reduced to $O(B \times Z)$ through locality optimization. The dimension selection step compares the capacity and distortion characteristics of 2D and 3D histograms, and its time complexity is a constant level of $O(1)$. Therefore, the overall time complexity of this module is $O(B \times Z)$.

The high-dimensional mapping matrix generation module operates as an offline calculation stage, dedicated to precomputing the optimal 3D mapping matrix H^{opt}. While the theoretical number of candidate matrices grows exponentially with the problem dimension, the search space is drastically reduced through adaptive selection rules, such as prioritizing coefficient pairs and triples from low-frequency DCT components which offer higher embedding efficiency. The module employs the multi-stage optimization strategy detailed in **Algorithm 1**. This strategy successively controls the search complexity by first minimizing the global distortion across all candidate matrices, then iteratively refining the solution set by maximizing local capacity and minimizing local distortion for each embedding triple. The core of the optimization involves evaluating a set of mapping matrices against a predefined distortion matrix D, with a computational complexity of $O(k \times 8 \times 27)$, where the critical parameter K represents the number of feasible solutions after applying the constraints. Extensive experimental data on datasets like BOSSBase confirm that the value of K is orders of magnitude smaller than the theoretical maximum. Consequently, the time complexity of this module is effectively constant, i.e., $O(1)$, making the precomputation feasible and efficient.

In the data embedding and Huffman dynamic encoding module, the algorithm first modifies the selected coefficients according to the mapping matrix. The time complexity of processing each selected frequency and block is $O(1)$, and the overall complexity is $O(R \times N)$. Subsequently, the Huffman table is updated dynamically, including steps such as counting R/S frequencies and re-encoding AC coefficients, and their time complexities are $O(B)$, and $O(B \times Z)$ respectively. Comprehensively, the time complexity of this module is $O(B \times Z)$). The statistics and sorting operations in the dynamic encoding process ensure the high efficiency of the algorithm.

To sum up, the total time complexity of the algorithm in this research is $O(B \times logB + B \times Z)$, where B represents the number of image blocks, and Z is the average number of non-zero AC coefficients per block. In practical application scenarios, since the Z value is small and stable, and the growth rate of the $logB$ term is relatively slow, the overall algorithm shows the characteristic of approximately linear time complexity of $O(B)$, which fully verifies the high efficiency in handling the reversible information hiding task of JPEG images.



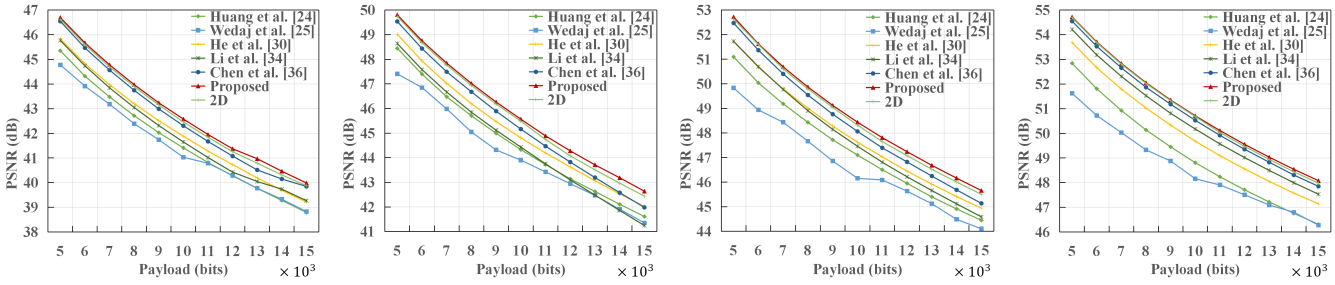
(a) 1_4.2.06/USC-SIPI/QF=60 (b) 2_5.2.08/USC-SIPI/QF=60 (c) 3_5.2.09/USC-SIPI/QF=60 (d) 4_5.2.10/USC-SIPI/QF=60

(e) 5_5.3.02/USC-SIPI/QF=60 (f) 6_7.1.01/USC-SIPI/QF=60 (g) 7_7.1.03/USC-SIPI/QF=60 (h) 8_7.1.05/USC-SIPI/QF=60

(i) 1_4.2.06/USC-SIPI/QF=90 (j) 2_5.2.08/USC-SIPI/QF=90 (k) 3_5.2.09/USC-SIPI/QF=90 (l) 4_5.2.10/USC-SIPI/QF=90

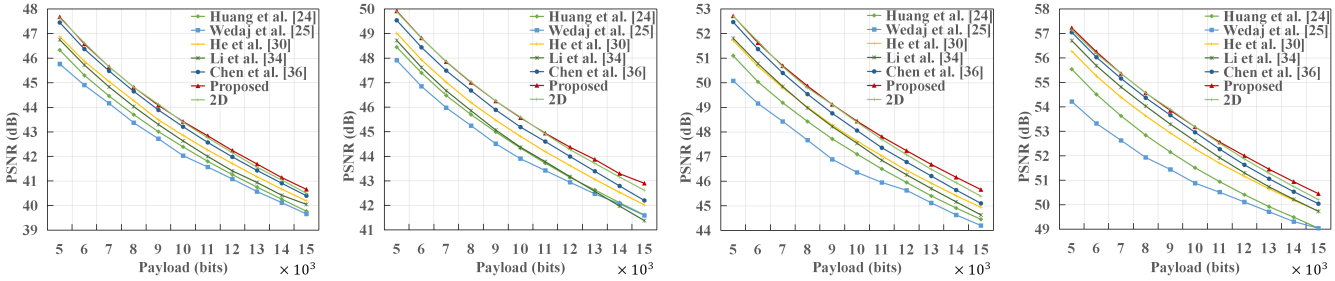
(m) 5_5.3.02/USC-SIPI/QF=90 (n) 6_7.1.01/USC-SIPI/QF=90 (o) 7_7.1.03/USC-SIPI/QF=90 (p) 8_7.1.05/USC-SIPI/QF=90

Fig. 14. PSNR of eight JPEG images from dataset USC-SIPI(QF=60 and 90).



(a) QF=60 (b) QF=70 (c) QF=80 (d) QF=90

Fig. 15. Average PSNR of dataset USC-SIPI.



(a) QF=60 (b) QF=70 (c) QF=80 (d) QF=90

Fig. 16. Average PSNR of dataset KODAK.

IV. Experimental Results

In this section, experimental setups are introduced, and performance of embedding capacity, image quality and file size is presented.

A. Experimental Setups

Four widely used test images include Man, Pepper, Boat, and House. Eight test images were randomly selected from the mainstream dataset USC-SIPI (https://sipi.usc.edu/database) and KODAK (https://gitcode.com/Universal-Tool/38ba0). Each test image is compressed into JPEG format using the default Huffman table in MATLAB. The specific QF values {60,70,80,90} are used to comprehensively cover diverse image quality and compression ratios while aligning with comparable methodologies. To evaluate the performance of proposed scheme, several representative JPEG RDH schemes are selected for comparison, including [24], [25], [30], [34], [36] and the proposed scheme using only 2D mapping. The parameter of α in paper [30] is set to 0.4 for consistency and analysis.

B. Embedding Capacity

The embedding capacity is measured by the maximum number of bits that can be embedded in the cover JPEG image.

1) Experimental Results with Four Commonly Used Images: The findings depicted in Fig. 11a-11b demonstrate the substantial increase in embedding capacity achieved by the proposed method for Four frequently utilized images. Notably, the image Man exhibits the most significant enhancement in embedding capacity.

2) Tests with the mainstream dataset: The experimental results displayed in Fig. 12a-12b and Fig. 13a-13b provide compelling evidence of the considerable improvement in embedding capacity achieved by the proposed method when applied to the USC-SIPI dataset and KODAK dataset. Specifically, the embedding capacity of image USC-SIPI-5.3.02 can be significantly improved by the proposed method.

3) Tests under different QFs: The experimental results from Table I and Table II demonstrate that the proposed method can increase the embedding capacity of images under different QFs. The average embedding capacity of the proposed method is 50327 bits in USC-SIPI dataset, which is higher than that of [36](43995 bits), [34](42402 bits), [30](33131 bits), [25](33131 bits), and [24](33131 bits). The average embedding capacity of the proposed method is 45333 bits in KODAK dataset, which is higher than that of [36](39212 bits), [34](37761 bits), [30](31202 bits), [25](31202 bits), and [24](31202 bits). Moreover, the proposed method performs better as QF increases. Therefore, the proposed method outperforms 1D and 2D histogram modification algorithms in embedding capacity.

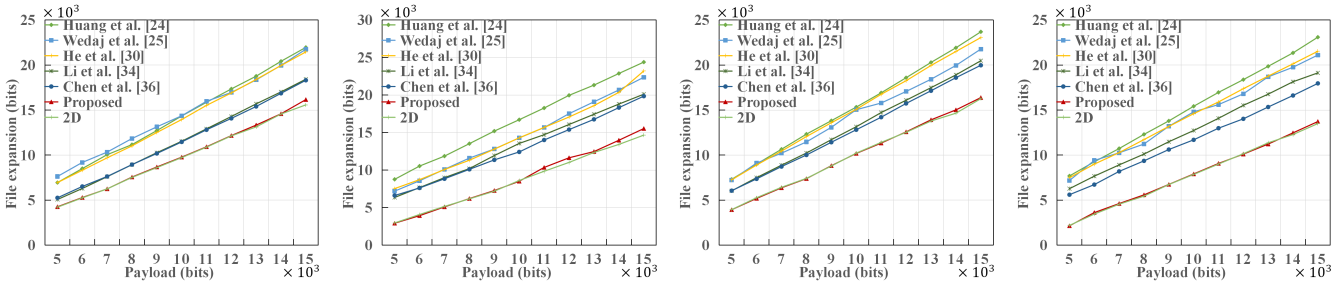
C. Image Quality

The image quality is evaluated by measuring the Peak Signal-to-Noise Ratio (PSNR) between the cover JPEG image and the marked JPEG image.

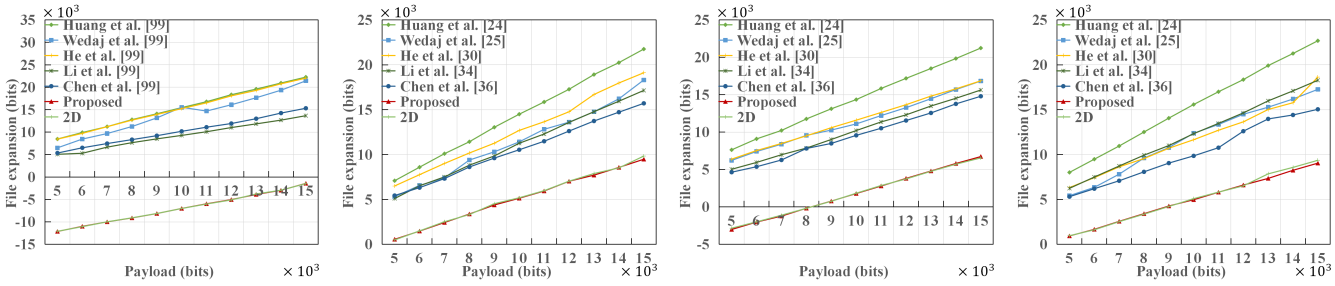
1) Tests with four frequently used images: The experimental results in Table III show that the proposed method can improve the image quality of four frequently used images. In particular, the image quality of House (QF=60/payload=13000 bits, QF=70/payload=13000, 14000 and 15000 bits) and Pepper (QF=60/payload=13000 and 14000 bits, QF=70/payload=15000 bits) can be improved by about 0.3dB.

2) Tests with the mainstream dataset: Based on the experimental results in Fig. 14 and Fig. 15, it can be observed that the proposed method markedly improves the image quality of the dataset USC-SIPI. Particularly, the image quality of image 5.2.09, 5.2.10, and 7.1.05 under QF=60 marked by the proposed method is better than that marked by other methods. Based on the experimental results in Fig. 16, the proposed method also improves the image quality of the dataset KODAK . Furthermore, with the increase of QF, the proposed method performs better. Additionally, the image quality is better than the proposed scheme using only 2D mapping for the payload of 12000, 13000, 14000 and 15000 bits.

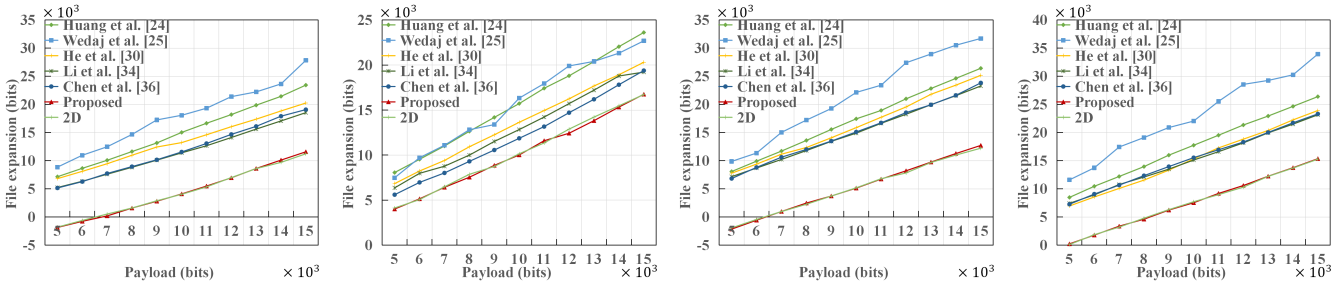
3) Tests under different QFs: As shown in Table IV and Table V, the proposed method can effectively enhance the image quality at different QF in USC-SIPI dataset and KODAK dataset, and the image quality is better in KODAK dataset. The image quality of the proposed method improves as the payload increases. When embedding a 15000 bit payload under QF=60, the PSNR of the proposed method is lower than that of [36] due to the inability to embed the entire payload in [36].



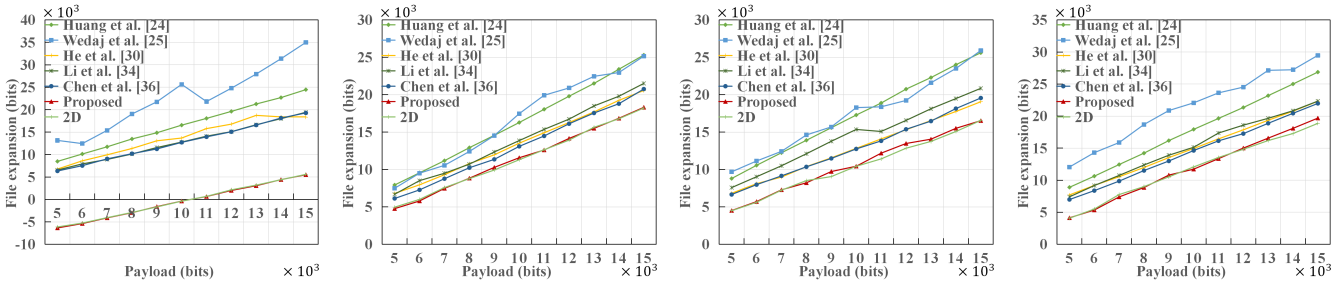
(a) 1_4.2.06/USC-SIPI/QF=60 (b) 2_5.2.08/USC-SIPI/QF=60 (c) 3_5.2.09/USC-SIPI/QF=60 (d) 4_5.2.10/USC-SIPI/QF=60



(e) 5_5.3.02/USC-SIPI/QF=60 (f) 6_7.1.01/USC-SIPI/QF=60 (g) 7_7.1.03/USC-SIPI/QF=60 (h) 8_7.1.05/USC-SIPI/QF=60

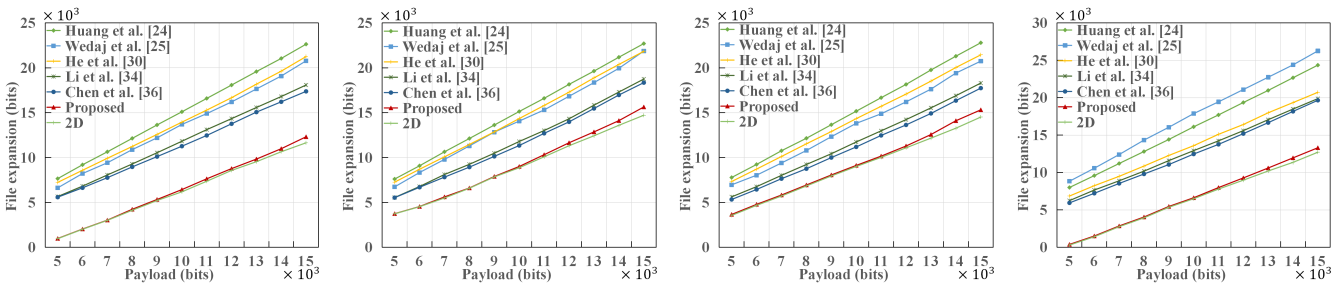


(i) 1_4.2.06/USC-SIPI/QF=90 (j) 2_5.2.08/USC-SIPI/QF=90 (k) 3_5.2.09/USC-SIPI/QF=90 (l) 4_5.2.10/USC-SIPI/QF=90



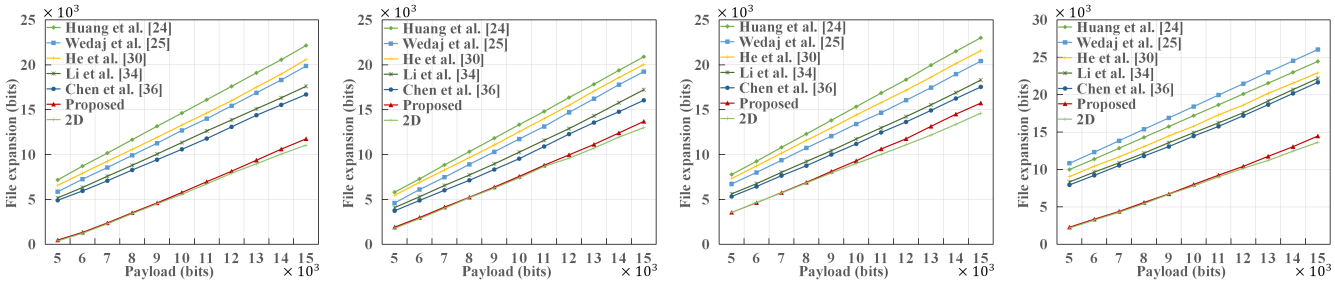
(m) 5_5.3.02/USC-SIPI/QF=90 (n) 6_7.1.01/USC-SIPI/QF=90 (o) 7_7.1.03/USC-SIPI/QF=90 (p) 8_7.1.05/USC-SIPI/QF=90

Fig. 17. file expansion of eight JPEG images from USC-SIPI (QF=60 and 90).



(a) QF=60 (b) QF=70 (c) QF=80 (d) QF=90

Fig. 18. Average file expansion of dataset USC-SIPI.



(a) QF=60 (b) QF=70 (c) QF=80 (d) QF=90

Fig. 19. Average file expansion of dataset KODAK.

TABLE VI
AVERAGE FILE EXPANSION OF JPEG BITSTREAM(QF=60,70,80 AND 90) COMPRESSED FROM FOUR FREQUENTLY USED IMAGES

Image	Method	QF=60/payload(bits)			QF=70/payload(bits)			QF=80/payload(bits)			QF=90/payload(bits)		
		13000	14000	15000	13000	14000	15000	13000	14000	15000	13000	14000	15000
average file size(bits)			115220			139150			177780			280890	
Man	Huang *et al.* [24]	19502	20926	22296	20220	21763	23104	20152	21539	23192	19232	20693	22126
	Wedaj *et al.* [25]	17734	19465	21651	17728	19601	22040	14553	16050	17679	15263	16646	18326
	He *et al.* [30]	20250	21729	23050	20637	21922	23402	20087	21673	23189	17368	18756	20269
	Li *et al.* [34]	14144	15173	16632	14783	15918	17177	14936	16035	17184	14494	15764	17035
	Chen *et al.* [36]	13672	14414	15534	14288	15271	15781	13963	14858	15804	14091	15016	16604
	2D	6508	7372	8645	8524	9341	10861	10869	11780	12543	5256	6927	8233
	Proposed	**6508**	**7372**	**8645**	**8524**	**9341**	**10861**	**10869**	**11780**	**12543**	**7132**	**8057**	**9945**
average file size(bits)			24160			29560			38500			65030	
Pepper	Huang *et al.* [24]	18702	20363	22031	18633	20121	21481	18371	19753	20989	19226	20882	22232
	Wedaj *et al.* [25]	18030	19617	21411	16275	18033	19853	15205	17270	18025	18452	19597	20128
	He *et al.* [30]	18111	19844	21301	17625	18942	20229	16480	18061	19335	15323	16456	17777
	Li *et al.* [34]	15712	N/A	N/A	14617	16080	17356	13990	15126	16401	14215	15439	16531
	Chen *et al.* [36]	15454	16860	N/A	14172	15848	17345	12955	14030	15391	13395	14673	15752
	2D	12899	14120	15327	12088	13560	14656	11176	12184	13184	4880	6832	7928
	Proposed	**13908**	**15011**	**16319**	**13589**	**14949**	**16061**	**11172**	**12149**	**13256**	**6760**	**7860**	**9736**
average file size(bits)			30330			36630			47910			75220	
Boat	Huang *et al.* [24]	19677	21020	22755	20087	21593	23192	21158	22800	24380	21681	23752	25430
	Wedaj *et al.* [25]	18488	19934	21457	19436	20460	22205	19500	21554	22547	21179	22546	23843
	He *et al.* [30]	18404	20216	21880	18690	20408	21778	19176	20684	22031	16950	18481	19899
	Li *et al.* [34]	16405	17794	19320	16547	18018	19620	16420	17848	19358	15662	17257	18763
	Chen *et al.* [36]	16020	17649	19109	16241	17817	19376	15886	17355	18746	15941	17240	19061
	2D	13088	14512	15936	13744	14896	16446	12304	14120	14920	11336	12448	14288
	Proposed	**13185**	**14859**	**17062**	**13592**	**14894**	**16510**	**14056**	**15624**	**16720**	**11736**	**13376**	**14696**
average file size(bits)			31330			36870			45820			66620	
House	Huang *et al.* [24]	20425	21838	23635	20552	22140	23931	20579	22302	23960	19772	21609	23350
	Wedaj *et al.* [25]	19884	21291	22700	20514	22083	23983	21575	23326	25100	24211	25910	27484
	He *et al.* [30]	19377	20957	22366	19453	21105	22593	19457	21022	22611	18655	20293	22248
	Li *et al.* [34]	17194	18732	20124	17410	19030	20629	17724	19463	21090	17848	19542	21334
	Chen *et al.* [36]	16678	18085	19530	17308	18907	20602	17859	19588	21424	18018	20090	20980
	2D	13358	14748	16085	14390	15876	18017	15816	17436	19004	14303	15461	16945
	Proposed	**13358**	**14748**	**16085**	**14390**	**15876**	**18017**	**15816**	**17436**	**19004**	**14303**	**15461**	**16945**

D. File Size

The file size is measured by the number of bits added to the marked JPEG bitstream compared to the cover JPEG bitstream.

1) Tests with four frequently used images: The experimental results presented in Table VI, demonstrate the effectiveness of the proposed method in reducing the file size of five commonly employed images. For Man (QF=60 and 70), the proposed method can reduce the file size by nearly 6000 bits.

2) Tests with the mainstream dataset: The experimental results from Fig. 17 and Fig. 18 demonstrate that the proposed method can suppress the file size of the dataset USC-SIPI. The experimental results from Fig. 19 demonstrate that the proposed method can also suppress the file size of the dataset KODAK. Unexpectedly, for images 5.3.02 (QF=60 and 90), 7.1.03 (QF=60), 4.2.06 (QF=90) and 5.2.09 (QF=90), the file size reduction achieved by the proposed method is so significant that it results in negative values, originating from encoding efficiency gains via mapping-induced histogram concentration. From the fundamental principles of JPEG encoding, during JPEG compression, AC coefficients are encoded using Run-Size Encoding. The file size of a JPEG follows Shannon's information theory: shorter codewords are assigned to high-frequency symbols, while longer codewords are allocated to low-frequency symbols. Specifically, taking image USC-SIPI/7.1.03/QF=60 under 5000 bits payload as an example , the AC coefficient huffman encoding lengths for the original image and the image with embedded secret information are

TABLE VII
AVERAGE FILE EXPANSION(BITS) OF JPEG UNDER DIFFERENT PAYLOADS(BITS) BITSTREAM FROM USC-SIPI

QF	Method	7000	9000	11000	13000	15000
average file size(bits)				47628		
60	Huang *et al.* [24]	10642	13635	16589	19581	22632
	Wedaj *et al.* [25]	9420	12199	14897	17642	20776
	He *et al.* [30]	9921	12579	15288	18178	21253
	Li *et al.* [34]	8061	10535	13109	15576	18084
	Chen *et al.* [36]	7758	10104	12462	15062	17375
	2D	3000	5214	7359	9538	11644
	Proposed	**3028**	**5329**	**7629**	**9829**	**12313**
average file size(bits)				57356		
70	Huang *et al.* [24]	10769	13796	16742	19745	22784
	Wedaj *et al.* [25]	9458	12493	15306	18227	21489
	He *et al.* [30]	9939	12784	15610	18488	21383
	Li *et al.* [34]	8171	10573	13139	15801	18496
	Chen *et al.* [36]	7776	10042	12452	15031	17780
	2D	5496	7842	10087	12425	14715
	Proposed	**4345**	**6513**	**8830**	**11276**	**13874**
average file size(bits)				72179		
80	Huang *et al.* [24]	10842	13814	16858	19976	23064
	Wedaj *et al.* [25]	9880	12775	15725	18998	22021
	He *et al.* [30]	10030	12812	15547	18345	21210
	Li *et al.* [34]	8337	10812	13348	15985	18741
	Chen *et al.* [36]	7795	10177	12623	15216	17919
	2D	5714	7949	10020	12166	14512
	Proposed	**5173**	**7454**	**9761**	**12139**	**14659**
average file size(bits)				113588		
90	Huang *et al.* [24]	11204	14418	17700	20977	24356
	Wedaj *et al.* [25]	12402	16050	19249	22739	26260
	He *et al.* [30]	9488	12282	15109	17986	20723
	Li *et al.* [34]	8892	11592	14235	17062	19878
	Chen *et al.* [36]	8539	11061	13761	16666	19654
	2D	2755	5352	7784	10204	12711
	Proposed	**2854**	**5461**	**7984**	**10603**	**13310**

TABLE IX
AVERAGE FILE EXPANSION(BITS) OF JPEG UNDER DIFFERENT PAYLOADS(BITS) BITSTREAM FROM KODAK

QF	Method	7000	9000	11000	13000	15000
average file size(bits)				45349		
60	Huang *et al.* [24]	10163	13155	16109	19102	22152
	Wedaj *et al.* [25]	8554	11263	13992	16877	19860
	He *et al.* [30]	9258	11916	14625	17515	20590
	Li *et al.* [34]	7581	10055	12629	15097	17605
	Chen *et al.* [36]	7078	9425	11783	14382	16695
	2D	2282	4510	6724	8964	11052
	Proposed	**2382**	**4610**	**6974**	**9364**	**11752**
average file size(bits)				54894		
70	Huang *et al.* [24]	8840	11823	14800	17834	20899
	Wedaj *et al.* [25]	7465	10305	13127	16221	19236
	He *et al.* [30]	8278	11059	14004	17077	20039
	Li *et al.* [34]	6555	8960	11585	14311	17223
	Chen *et al.* [36]	6020	8332	10898	13566	16047
	2D	4015	6267	8676	10712	12992
	Proposed	**4140**	**6376**	**8806**	**11126**	**13692**
average file size(bits)				68583		
80	Huang *et al.* [24]	10803	13813	16855	19964	23013
	Wedaj *et al.* [25]	9365	12050	14660	17450	20410
	He *et al.* [30]	10155	12962	15796	18624	21565
	Li *et al.* [34]	8016	10423	12977	15536	18300
	Chen *et al.* [36]	7644	9997	12465	14917	17536
	2D	5714	7949	10020	12166	14612
	Proposed	**5758**	**8120**	**10620**	**13142**	**15734**
average file size(bits)				102045		
90	Huang *et al.* [24]	12850	15750	18650	21550	24450
	Wedaj *et al.* [25]	13850	16900	19950	23000	26050
	He *et al.* [30]	11688	14483	17309	20187	22923
	Li *et al.* [34]	10893	13593	16235	19162	22178
	Chen *et al.* [36]	10539	13062	15761	18666	21655
	2D	4278	6674	9023	11212	13628
	Proposed	**4394**	**6734**	**9238**	**11762**	**14478**

TABLE VIII
THE AC COEFFICIENT RUN-SIZE ENCODING LENGTHS(BIT), FREQUENCY AND THE FILE EXPANSION(BIT) FOR THE ORIGINAL AND EMBEDDED IMAGE OF 7.1.03/USC-SIPI/QF=60

Run-Size	OrigLen	EmbLen	OrigFreq	EmbFreq	Expansion
(0,1)	3	3	15551	14621	-2790
(0,2)	4	4	8002	8787	3140
(0,3)	6	7	2751	2895	3759
(0,4)	8	11	612	613	1847
(0,5)	10	14	103	103	412
(1,1)	5	4	5365	4962	-6977
(1,2)	7	7	1538	1898	2520
(1,3)	10	10	256	298	420
(2,1)	6	5	2459	2200	-3754
(2,2)	10	8	418	663	1124
(2,3)	13	12	38	52	130
(3,1)	7	6	1403	1281	-2135
(3,2)	11	9	129	241	750
(4,1)	7	6	912	861	-1218
(4,2)	12	11	58	105	459
(5,1)	8	7	661	638	-822
(5,2)	13	12	15	37	249
(6,2)	14	13	8	19	135
(7,2)	14	13	3	13	127
(8,1)	10	9	157	156	-166

shown in Table VIII. Calculations reveal that the file size of the Embedded Image (7.1.03/QF=60) is smaller than that of the Original Image (7.1.03/USC-SIPI/QF=60).

3) Tests under different QFs: From the experimental results in Table VII and Table IX, it can be seen that the proposed method significantly suppresses the file size at different QFs. Specifically, as QF increases, the proposed method demonstrates even greater suppression of file size for marked images. Furthermore, for payloads of 12000, 13000, 14000, and 15000 bits, the file size expansion is slightly larger than that of the proposed scheme which uses only 2D mapping.

V. CONCLUSION

To address issues such as increasing embedding capacity while suppressing file size and image distortion, this study proposes MHMF which employs a variable-dimensional histogram and high-dimensional histogram enhancement. Compared with traditional histogram modification frameworks, MHMF improves traditional fixed dimensional histograms by introducing a dynamic method. It adaptively determines the dimension of the histogram based on the characteristics of the cover image and embedding data. In addition, MHMF extends the histogram from 1D and 2D to higher dimension. It focuses on forming more centralized high-dimensional histograms leveraging the inherent correlations between coefficients. The experimental results demonstrate that compared with 1D and 2D JPEG RDH schemes, the proposed scheme has advantages in embedding capacity, image quality, and file size.

Despite its significant improvements, MHMF still faces limitations and opportunities for further development. Its current focus on JPEG images restricts its application scope, and

future research could expand its use to other image formats and multimedia types.

REFERENCES

[1] G. Hou, B. Ou, M. Long, and F. Peng, "Separable reversible data hiding for encrypted 3d mesh models based on octree subdivision and multi-msb prediction," *IEEE Transactions on Multimedia*, vol. 26, pp. 2395–2407, 2024.

[2] K. Gao, J.-H. Horng, and C.-C. Chang, "Reversible data hiding for encrypted 3d mesh models with secret sharing over galois field," *IEEE Transactions on Multimedia*, vol. 26, pp. 5499–5510, 2024.

[3] Y.-Y. Tsai, "Separable reversible data hiding for encrypted three-dimensional models based on spatial subdivision and space encoding," *IEEE Transactions on Multimedia*, vol. 23, pp. 2286–2296, 2021.

[4] D. Fu, X. Zhou, L. Xu, K. Hou, and X. Chen, "Robust reversible watermarking by fractional order zernike moments and pseudo-zernike moments," *IEEE Transactions on Circuits and Systems for Video Technology*, 2023.

[5] X. Liao, Y. Yu, B. Li, Z. Li, and Z. Qin, "A new payload partition strategy in color image steganography," *IEEE transactions on circuits and systems for video technology*, vol. 30, no. 3, pp. 685–696, 2019.

[6] Y. Zhuang, S. Liu, C. Ding, and X. Zhou, "Reversible watermarking based on extreme prediction using modified differential evolution," *Applied Intelligence*, vol. 52, no. 12, pp. 14 406–14 425, 2022.

[7] Z. Qian, H. Xu, X. Luo, and X. Zhang, "New framework of reversible data hiding in encrypted jpeg bitstreams," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 2, pp. 351–362, 2018.

[8] S. Sheidani, A. Mahmoudi-Aznaveh, and Z. Eslami, "Cpa-secure privacy-preserving reversible data hiding for jpeg images," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 3647–3661, 2021.

[9] Z. Hua, Z. Wang, Y. Zheng, Y. Chen, and Y. Li, "Enabling large-capacity reversible data hiding over encrypted jpeg bitstreams," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 33, no. 3, pp. 1003–1018, 2022.

[10] Y. Wu, R. Hu, and S. Xiang, "Pvo-based reversible data hiding using global sorting and fixed 2d mapping modification," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 34, no. 1, pp. 618–631, 2024.

[11] D. Coltuc and H. G. Coanda, "Reversible contrast enhancement by histogram specification and very low distortion data hiding," *IEEE Transactions on Information Forensics and Security*, vol. 19, pp. 529–539, 2024.

[12] B. Ou, X. Li, Y. Zhao, R. Ni, and Y.-Q. Shi, "Pairwise prediction-error expansion for efficient reversible data hiding," *IEEE Transactions on image processing*, vol. 22, no. 12, pp. 5010–5021, 2013.

[13] C. Zhang and B. Ou, "Reversible data hiding based on multiple adaptive two-dimensional prediction-error histograms modification," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 7, pp. 4174–4187, 2022.

[14] J. Chang, G. Zhu, H. Zhang, Y. Zhou, X. Luo, and L. Wu, "Reversible data hiding for color images based on adaptive 3d prediction-error expansion and double deep q-network," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 8, pp. 5055–5067, 2022.

[15] Q. Chang, X. Li, and Y. Zhao, "Reversible data hiding for color images based on adaptive three-dimensional histogram modification," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 9, pp. 5725–5735, 2022.

[16] J. Fridrich, M. Goljan, and R. Du, "Lossless data embedding for all image formats," *Security and watermarking of multimedia contents IV*, vol. 4675, pp. 572–583, 2002.

[17] B. G. Mobasseri, R. J. Berger, M. P. Marcinak, and Y. J. NaikRaikar, "Data embedding in jpeg bitstream by code mapping," *IEEE Transactions on Image Processing*, vol. 19, no. 4, pp. 958–966, 2010.

[18] Z. Qian and X. Zhang, "Lossless data hiding in jpeg bitstream," *Journal of Systems and Software*, vol. 85, no. 2, pp. 309–313, 2012.

[19] Y. Hu, K. Wang, and Z.-M. Lu, "An improved vlc-based lossless data hiding scheme for jpeg images," *Journal of Systems and Software*, vol. 86, no. 8, pp. 2166–2173, 2013.

[20] Y. Qiu, H. He, Z. Qian, S. Li, and X. Zhang, "Lossless data hiding in jpeg bitstream using alternative embedding," *Journal of Visual Communication and Image Representation*, vol. 52, pp. 86–91, 2018.

[21] C. Zhang, B. Ou, and D. Tang, "An improved vlc mapping method with parameter optimization for reversible data hiding in jpeg bitstream," *Multimedia Tools and Applications*, vol. 79, no. 27-28, pp. 19 045–19 062, 2020.

[22] Y. Qiu, Z. Qian, H. He, H. Tian, and X. Zhang, "Optimized lossless data hiding in jpeg bitstream and relay transfer-based extension," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 4, pp. 1380–1394, 2021.

[23] Y. Du and Z. Yin, "New framework for code-mapping-based reversible data hiding in jpeg images," *Information Sciences*, vol. 609, pp. 319–338, 2022.

[24] F. Huang, X. Qu, H. J. Kim, and J. Huang, "Reversible data hiding in jpeg images," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 9, pp. 1610–1621, 2015.

[25] F. T. Wedaj, S. Kim, H. J. Kim, and F. Huang, "Improved reversible data hiding in jpeg images based on new coefficient selection strategy," *EURASIP Journal on Image and Video Processing*, vol. 2017, pp. 1–11, 2017.

[26] D. Hou, H. Wang, W. Zhang, and N. Yu, "Reversible data hiding in jpeg image based on dct frequency and block selection," *Signal Processing*, vol. 148, pp. 41–47, 2018.

[27] J. He, X. Pan, H.-t. Wu, and S. Tang, "Improved block ordering and frequency selection for reversible data hiding in jpeg images," *Signal Processing*, vol. 175, p. 107647, 2020.

[28] F. Li, Z. Qi, X. Zhang, and C. Qin, "Jpeg reversible data hiding using dynamic distortion optimizing with frequency priority reassignment," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 12, pp. 8849–8863, 2022.

[29] X. Yang, T. Wu, and F. Huang, "Reversible data hiding in jpeg images based on coefficient-first selection," *Signal Processing*, vol. 200, p. 108639, 2022.

[30] J. He, J. Chen, and S. Tang, "Reversible data hiding in jpeg images based on negative influence models," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 2121–2133, 2019.

[31] Z. Yin, Y. Ji, and B. Luo, "Reversible data hiding in jpeg images with multi-objective optimization," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 8, pp. 2343–2352, 2020.

[32] K. Chen, H. Zhou, D. Hou, W. Zhang, and N. Yu, "Reversible data hiding in jpeg images under multi-distortion metric," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 10, pp. 3942–3953, 2020.

[33] M. Xiao, X. Li, B. Ma, X. Zhang, and Y. Zhao, "Efficient reversible data hiding for jpeg images with multiple histograms modification," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 7, pp. 2535–2546, 2020.

[34] N. Li and F. Huang, "Reversible data hiding for jpeg images based on pairwise nonzero ac coefficient expansion," *Signal Processing*, vol. 171, p. 107476, 2020.

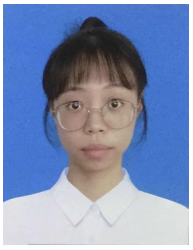
[35] M. Xiao, X. Li, and Y. Zhao, "Reversible data hiding for jpeg images based on multiple two-dimensional histograms," *IEEE Signal Processing Letters*, vol. 28, pp. 1620–1624, 2021.

[36] Y. Chen, J. He, and Y. Xian, "Reversible data hiding for jpeg images based on improved mapping and frequency ordering," *Signal Processing*, vol. 198, p. 108604, 2022.

[37] S. Weng, Y. Zhou, T. Zhang, M. Xiao, and Y. Zhao, "General framework to reversible data hiding for jpeg images with multiple two-dimensional histograms," *IEEE Transactions on Multimedia*, 2022.

[38] X. Zhou, K. Hou, Y. Zhuang, Z. Yin, and W. Han, "General pairwise modification framework for reversible data hiding in jpeg images," *IEEE Transactions on Circuits and Systems for Video Technology*, 2023.

[39] J. He, J. Chen, and S. Tang, "Reversible data hiding in jpeg images based on negative influence models," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 2121–2133, 2020.



Kaiyue Hou received the master's degree from the School of Cybersace Security Academy, Hainan University, in 2024. Her current research interest includes reversible data hiding.
Affiliations: The School of Cyberspace Security, Hainan University, Hainan, Haikou 570228, China
Contact details: 1213670222@qq.com

Cheng Zeng received the bachelor's degree from the School of Cyberspace Security Academy, Hainan University, in 2022, where he is currently pursuing the master's degree. His current research interest includes reversible data hiding.
Affiliations: The School of Cyberspace Security, Hainan University, Hainan, Haikou 570228, China
Contact details: 23210839000021@hainanu.edu.cn

Xiaoyi Zhou serves as Professor and Doctoral Supervisor at Hainan University, recognized as a Hainan Provincial Outstanding Talent and Associate Dean of the School of Cyberspace Security. Her primary research focuses on multimedia security and AI security based on cryptography and information hiding. Her contributions have been honored with the Third Prize of Hainan Provincial Science and Technology Progress Award and the First Prize of Hainan Provincial Teaching Achievement Award.
Affiliations: The School of Cyberspace Security, Hainan University, Hainan, Haikou 570228, China
Contact details: xy.zhou@hainanu.edu.cn

Dahao Fu received the master's degree from the School of Cyberspace Security Academy, Hainan University, in 2024. His current research interest includes reversible data hiding.
Affiliations: The School of Cyberspace Security, Hainan University, Hainan, Haikou 570228, China
Contact details: fudahao2021@163.com

Xin liao is a Professor and Doctoral Supervisor at Hunan University, serving as Associate Dean of the School of Cyberspace Security. His research focuses on information content security and artificial intelligence (AI) security. He holds key leadership roles as Deputy Director of the Hunan Provincial Key Laboratory and Director of the Hunan Provincial Cyberspace Security Outstanding Innovative Talent Development Base. Recognized as a globally leading scholar, he has been listed in the Global Top 2% Scientists Ranking.
Affiliations: the College of Computer Science and Electronic Engineering, Hunan University, Changsha 410082, China, and also with the Shenzhen Key Laboratory of Media Security, Shenzhen University, Shenzhen 518060,China
Contact details: xinliao@hnu.edu.cn