Lossless data hiding in JPEG images with segment coding

Mingming Zhang
Quan Zhou
Yanlang Hu

Mingming Zhang, Quan Zhou, Yanlang Hu, “Lossless data hiding in JPEG images with segment coding,” J. Electron. Imaging 28(5), 053015 (2019), doi: 10.1117/1.JEI.28.5.053015.
Lossless data hiding in JPEG images with segment coding

Mingming Zhang,* Quan Zhou, and Yanlang Hu
Xi’an Institute of Space Radio Technology, National Key Laboratory of Science and Technology on Space Microwave, Xi’an, China

Abstract. This study examines the lossless compression of an original JPEG bitstream such that spare space is reserved to embed secret data in a file header. After an original image is transformed using discrete cosine transform (DCT), the nonzero DCT coefficients at high frequencies are relatively sparse and their absolute values are small. To improve encoding efficiency, we propose a segment encoding algorithm that defines a high frequency in DCT blocks as the termination point and recodes coefficients separately. The coefficients above this point are encoded in the proposed Rxy code, and the coefficients below it are encoded in JPEG code. Then, to further preserve the embedding capacity’s (EC) robustness to different quality factors, we cluster blocks into large $32 \times 32$ blocks and record special block indexes that are not suitable for Rxy encoding in appended data. Hence, space is reserved for inserting secret data and appended data after the comment marker in a file header. A receiver can obtain the termination point through data analysis, extract additional data, and recover the original JPEG image in a lossless manner. Encoding redundancy is substantially reduced such that a large EC can be achieved with no file expansion. Moreover, only coefficients at high frequencies are recoded; hence, concealment is computationally simple. Experimental results demonstrate that the proposed method has a larger EC than state-of-the-art methods.© 2019 SPIE and IS&T [DOI: 10.1117/1.JEI.28.5.053015]

Keywords: JPEG bitstream; compression domain; lossless data hiding; segment compression; encoding efficiency.

Paper 190370 received May 11, 2019; accepted for publication Sep. 17, 2019; published online Oct. 9, 2019.

1 Introduction
Data hiding is a technique in which secret data are embedded in cover media. This technique can be used in numerous applications. However, it has a drawback in that noticeable distortion is caused in cover media by irreversible operations. Reversible data hiding (RDH)1 can recover host images exactly after secret data have been extracted. The technique is widely employed in the fields of law enforcement, medical systems, and military imagery. RDH for images can be divided into four categories based on the domain in which it is employed,2 namely, the spatial domain,3–7 transform domain,8–16 compression domain,13–16 and encryption domain.17,18 There are three RDH techniques in the spatial domain, specifically, lossless compression,3 difference expansion,4 and histogram shifting.5 The transform domain includes difference expansion,8 prediction-error expansion,10 and integer-to-integer transforms.11,12 The compression domain consists of lossless compression,13 difference expansion,14,15 and prediction-error expansion.16 The encryption domain consists of difference expansion17 and prediction-error expansion.18 The relationship can also be shown in Table 1.

Good performance is achieved with respect to embedding capacity (EC) and visual quality using these methods. These techniques are mostly applied to uncompressed images; however, the majority of images are compressed in Joint Photographic Experts Group (JPEG) format. JPEG compression is the most popular lossy image compression technique, in which most redundant data are discarded. The resulting file size is reduced substantially, and computational complexity is acceptable.19,20 RDH approaches for JPEG images are divided into four types, specifically, quantization of discrete cosine transform (DCT) coefficients,21–26 modification of a quantization table,27 modification of the Huffman table,28–32 and concealment in an encrypted JPEG bitstream.33,34 However, most of them cause distortion to host images and cannot recover data in a lossless manner. Lossless data hiding (LDH) can recover host images in a lossless manner after secret data are extracted. However, even though certain methods can restore host images in a lossless manner, file expansion might occur and the increase in file size is almost always larger than the size of the secret data.

To achieve LDH for JPEG images without increasing file size, Mobasseri et al.28 proposed an algorithm to flip the appended bits of variable-length codes (VLCs) and employed a VLC mapping technique to hide data. Qian and Zhang30 proposed a mapping algorithm to relate used VLCs to unused VLCs of the same size to preserve file size. Ma et al.35 improved the algorithm of Qian and Zhang30 to calculate the occurrences in used VLCs and map them according to frequencies, thereby further improving EC. Qiu et al.32 constructed an optimal mapping between used and unused VLCs in each category to achieve a high embedding rate. Ma et al.35 proposed a scheme that used the framework of reserving room before encryption (RRBE). To extend the RRBE framework in JPEG images, Chang et al.36 proposed a scheme in which a JPEG bitstream is compressed in a lossless manner before encryption. In these methods, no file expansion occurs, but hiding capacity is still quite small. We propose an LDH approach for JPEG bitstreams. By analyzing the relationship between
Table 1: Relation between domain and techniques.

| Domain          | Techniques                                               |
|-----------------|---------------------------------------------------------|
| Spatial domain  | Lossless compression, difference expansion, and histogram shifting |
| Transform domain| Difference expansion, prediction-error expansion, and integer-to-integer transforms |
| Compression domain | Lossless compression, difference expansion, and prediction-error expansion |
| Encryption domain | Difference expansion and prediction-error expansion |

frequencies and coefficients, we encode the coefficients at high frequencies in a proposed Rxy code and the coefficients at low frequencies in JPEG code. Then, similar to Zhang et al.’s method, secret data and appended data are inserted as the comment part in a file header. The proposed method does not increase file size and EC improves significantly.

The remainder of this paper is organized as follows: First, we review the previous LDH methods without file expansion in Sec. 2. Then, the principle of segment encoding is introduced in Sec. 3. In addition, we present the proposed data hiding method. Section 4 presents and discusses the results of experiments and compares the impact of different parameters on performance. Finally, we state our conclusions in Sec. 5.

2 Previous Methods

There are two classical LDH methods in JPEG images with no file expansion: (1) VLC mapping to represent secret data and (2) concealment in an encrypted JPEG bitstream.

2.1 VLC Mapping

Mobasseri et al. found that there are 162 VLCs, but only a fraction of them are actually used, resulting in a small reserved mapping space. In their method, a VLC mapping relationship is constructed, and VLC bits are flipped to represent secret data. People who are unfamiliar with the JPEG coding principle cannot recognize any visual difference between marked images and host images using the error concealment technique. Even though distortion is minimized, a slight bitwise difference still exists. This implies that this method is not actually LDH. In Qian and Zhang’s method, 162 VLCs are classified into 16 categories according to binary length \( \{C_1, C_2, \ldots, C_{16}\} \). In category \( C_i \) of \( i \) bits, there are \( p_i \) used VLCs and \( q_i \) unused VLCs. This category is denoted as \( C_j = \{VLC_{i,1}, VLC_{i,2}, \ldots, VLC_{i,q_i}; VLC_{i,1}^{(n)}, VLC_{i,2}^{(n)}, \ldots, VLC_{i,q_i}^{(n)}\} \), where \( u \) denotes used VLCs and \( n \) denotes unused VLCs. Several unused VLCs are mapped to one used VLC to represent secret data, and receivers can extract secret data without extra information. In extreme cases, modified frequencies might exceed 64, which is out of range and will lead to decoding errors. Hu et al. found that the occurrences of used VLCs differ, and if more unused VLCs are mapped to the most frequently used VLC, more secret bits can be allocated in replacement. Hu et al.’s method sorts used VLCs and maps them such that more frequently used VLCs represent more bits while less frequently used VLCs represent fewer bits. The optimization can further increase EC and potential space is well utilized. However, the decoding failure problem cannot be resolved.

To explore more space, Qiu et al. constructed an alternative LDH embedding method using code mapping and reordering. Similar to Hu et al.’s method, they found that if an unused VLC is arbitrarily mapped to a specific used VLC, it will produce different combinations of unused VLC mapping. The EC, \( EC_1 \), in category \( C_i \) becomes equal to \( EC_{1,1} + EC_{1,2} \), where \( EC_{1,1} \) is sorted VLC mapping capacity and \( EC_{1,2} \) is unused VLC combined capacity. As a supplement, Qiu et al. also used a VLC reordering algorithm proposed by Wu and Deng, which modified the orders of run/size in \( C_i \) and constructed a new Huffman tree. For example, 125 VLCs are included in category \( C_{16} \) such that EC is \( \log_2125 \), i.e., 695 bits, which is obtained by rearranging all VLCs in category \( C_{16} \). In an alternative embedding method, the maximum EC, \( EC_{1} \), in category \( C_i \) is equal to the maximum of \( \{EC_{1,1}, EC_{1,2}\} \), where \( EC_{1,1} \) and \( EC_{1,2} \) are the EC in the improved mapping and reordering, respectively. Hence, the EC of the entire JPEG file is \( EC = \sum_{i=1}^{16} \log_2C_i \). Qiu et al. achieved the highest EC in VLC mapping. However, the latent decoding risk could not be prevented and the capacity was not extremely large.

2.2 Concealment in Encrypted JPEG Bitstream

Ma et al. first proposed the RRBE framework, in which a host image is first compressed such that spare space is reserved before encryption, and then, secret data are embedded in the reserved space using the original RDH algorithm. A receiver can extract secret data and recover the host image in a lossless manner such that no errors occur. However, this approach is not suitable for JPEG images because they are already compressed.

Chang et al. constructed a separable RDH scheme for an encrypted JPEG bitstream based on the structure developed by Ma et al. A JPEG file is first decompressed into quantized DCT blocks and divided into individual bitstreams corresponding to the blocks. Then, the blocks are shuffled using a pseudorandom permutation, and nonzero coefficients are converted into binary digits. Additionally, in the two least important bit planes, there is bias in the 0-1 distribution such that sparse space can be recompressed. The bitstream corresponding to the two planes is extracted and recompressed in a lossless manner using Chang et al.’s method to reserve the spare space. Secret data are inserted at the end of the recompressed bitstream, and a synthetic bitstream is encrypted and filled in the two least important planes. There is no file expansion, and the redundancy in low bit planes is fully exploited. The quantization table is also encrypted so that a receiver can display marked images, which are slightly deviated with respect to the original JPEG images, using a decryption key. The receiver can extract secret data by utilizing a data embedding key and then decompress the JPEG bitstream to restore the images that are lossless with respect to the original JPEG images. However, encryption modifies the distribution of original data, and only the data in low bit planes are compressed such that theoretical EC cannot be achieved.
3 Proposed Method

3.1 Segment Encoding

After the quantized DCT transform is performed, data are encoded in the form of run, size, and value (RSV) and concentrated at low frequencies, whereas the nonzero coefficients at high frequencies are sparse. Run refers to the number of zeros before a nonzero coefficient, size refers to the binary size of the nonzero coefficient, and value refers to its value.

In the JPEG encoding criterion, the zero interval and nonzero coefficients are jointly encoded in JPEG code. For example, in DCT coefficients \(0, 0, 3, 5, 0, 0, -2, 0, 0, 1\), the following four clusters are formed: \([0, 0, 3], [5], [0, 0, -2], \) and \([0, 0, 1]\). In each block, the coefficients are scanned in the “Zigzag” mode, which is shown in Fig. 1. The lowest frequency is for the direct current component, and it is denoted as the first frequency. The other frequencies are for alternating current components, and the highest frequency is denoted as the 64th frequency.

In the distribution of coefficients at high frequencies, there are frequently a few zeros before a small nonzero coefficient. If the absolute value of the nonzero coefficient is smaller than a threshold, no extra bits are required to specially encode the size of the nonzero coefficient. In this case, encoding length decreases and encoding efficiency is improved. At high frequencies, nonzero coefficients are so small that a termination point can be obtained, above which the absolute values of the coefficients are smaller than a threshold. Nonzero coefficients are encoded using the proposed Rxy code for frequencies above the termination point, and they are encoded using JPEG code for frequencies below this point.

There are 162 VLCs for encoding the DCT coefficients, but only a few VLCs are used, particularly at high frequencies. In VLC mapping methods, \(^{30-32}\) used VLCs are replaced with unused VLCs of the same binary size to represent secret bits. Additionally, the encoded bitstream of coefficients at high frequencies occupies considerable space in an entire compressed file. However, the number of nonzero coefficients is quite small, and most values are in \([-1, 1]\) such that more bits are required to encode them in JPEG code. Based on the assumption that coefficients at high frequencies are small and sparsely distributed with a large part of the bitstream, we only build a small encoding dictionary of Rxy code specific to the coefficients at high frequencies such that the zero interval and nonzero coefficients in \([-1, 1]\) can be jointly encoded, similar to JPEG code, but the size of the encoded bits is significantly reduced.

![Fig. 1 DCT coefficient arrangement in “Zigzag” mode.](Image)

The Rxy encoding scheme is proposed for nonzero coefficients, which are either “-1” or “1.” The encoding framework is shown in Fig. 2, where run indicates the length of the zero interval and requires three bits, and \(x\) refers to a nonzero coefficient.

In contrast to the form of RSV, there is no size in the proposed encoding form because a nonzero coefficient is in \([-1, 1]\) and size is always 1; hence, one bit can encode the nonzero value. The encoding of run is shown in Table 2.

The zero interval is encoded in the proposed bitstream, and “000” refers to the end of the encoding in a block.

In addition, \(x\) refers to a nonzero coefficient with a value in \([-1, 1]\). It is encoded using one bit; this mapping is shown in Table 3.

Joint encoding length is decreased by reducing the information entropy of the nonzero coefficients that are only in \([-1, 1]\). When the number of zeros, \(M\), is greater than six, the run encoding scheme in Rxy code is similar to the run encoding scheme in JPEG code, in that parameter \(m\) refers to quotient \(M\) divided by six, and parameter \(n\) is \(M\) modulo six.

\[
m = \lfloor M/6 \rfloor, \quad n = \text{mod}(M, 6), \quad M = m \times 6 + n.
\]

The joint encoding is shown in Table 4 for \(M\) in \([0, 1, 2, 3, 4, 5, 6]\).

When \(M\) is larger than 5, \([M/6]\) groups of “111” are added before the result.

We compare Rxy code and JPEG code in Table 5.

In Table 5, code length refers to run/size length in binary, codeword refers to run/size code, sum length refers to joint

| Run | Number of zeros |
|-----|-----------------|
| 001 | 0               |
| 010 | 1               |
| 011 | 2               |
| 100 | 3               |
| 101 | 4               |
| 110 | 5               |
| 111 | 6               |

| \(x\) | Value |
|-------|-------|
| 0     | -1    |
| 1     | 1     |

![Fig. 2 Rxy encoding scheme.](Image)
code size, and rate refers to the ratio of nonzero coefficients to run/size in binary size. Suppose that code length is $m_1$ bits, sum length is $m$ bits, and the binary size of nonzero coefficient code $m_2$ is $m - m_1$ bits. Then, rate is calculated as follows:

$$\text{Rate} = \frac{m_2}{m_1}, \quad m_2 = m - m_1. \quad (2)$$

When a nonzero coefficient is in $\{-1, 1\}$, binary size is one bit and rate is $1/m_1$. Compared with the standard JPEG Huffman table, the rate of the proposed scheme in Rxy code is larger than that of the standard scheme in JPEG code, except for run/size 0/0. Run/size 0/0 implies that no zero intervals exist between small nonzero coefficients, such as the sequence $-1, 1, -1$. A larger rate indicates a higher proportion of nonzero coefficients. The proposed scheme is also advantageous with respect to code length; the code length for Rxy coding is almost always smaller than that of the standard scheme, except for run/size 0/1, which confirms that the proposed scheme is more suitable when coefficients are sparsely distributed. When a nonzero coefficient is in $\{-1, 1\}$, a higher number of zeros leads to shorter code length using Rxy code compared to the code length in JPEG code. Furthermore, “000,” which refers to the end of the encoding, is shorter than “1010,” which is used in the standard JPEG scheme.

JPEG code is more suitable in a dense distribution in which absolute values are large at low frequencies. In contrast, at high frequencies, the encoding efficiency in JPEG

### Table 4 Joint encoding.

| $M$ | Run | Rxy |
|-----|-----|-----|
| 0   | 001 | 001x |
| 1   | 010 | 010x |
| 2   | 011 | 011x |
| 3   | 100 | 100x |
| 4   | 101 | 101x |
| 5   | 110 | 110x |
| 6   | 111 | 111001x |

### Table 5 Rxy code comparison.

| Run/size | Code length | Code word | Sum length | Rate |
|----------|-------------|-----------|------------|------|
|          | Rxy JPEG    | Rxy JPEG  | Rxy JPEG   | Rxy JPEG |
| 0/0 (EOB)| 3 4 000 1010 | 3 4 0 0    |            |
| 0/x      | 3 2 001 00   | 4 3 0.33 0.5 |          |
| 1/x      | 3 4 010 1100 | 4 5 0.33 0.25 |        |
| 2/x      | 3 5 011 11100 | 4 6 0.33 0.2 |      |
| 3/x      | 3 6 100 111010 | 4 7 0.33 0.17 |     |
| 4/x      | 3 6 101 111011 | 4 7 0.33 0.17 |     |
| 5/x      | 3 7 110 111010 | 4 8 0.33 0.14 |     |
| 6/x      | 6 7 111001 111011 | 7 8 0.17 0.14 |     |
| 7/x      | 6 8 111010 1111010 | 7 9 0.17 0.13 |     |
| 8/x      | 6 9 111011 11111000 | 7 10 0.17 0.11 |     |
| 9/x      | 6 9 111100 11111001 | 7 10 0.17 0.11 |     |
| 10/x     | 6 9 111101 11111010 | 7 10 0.17 0.11 |     |
| 11/x     | 6 10 111110 111111001 | 7 11 0.17 0.1 |     |
| 12/x     | 9 10 11111001 111111010 | 10 11 0.11 0.1 |     |
| 13/x     | 9 11 11111010 1111111000 | 10 12 0.11 0.09 |     |
| 14/x     | 9 16 11111011 1111111111011 | 10 17 0.11 0.06 |     |
| 15/x     | 9 16 11111100 111111111110101 | 10 17 0.11 0.06 |     |

Note: EOB, end of block.
code is not high because absolute values are small and their distribution is sparse, implying that a large number of bits are required. When nonzero binary size is given, only the number of zeros must be encoded. As long as this number is nonzero, code length is considerably smaller in the proposed scheme. After segment coding, spare space is reserved for embedding secret data in a file header. In a receiver, if the termination point is obtained, each 8 × 8 block can be separated and secret data can be extracted. As the coefficients at both the whole low and high frequencies are not modified, JPEG images can be displayed without distortion.

Joint encoding length is decreased by reducing the information entropy of the nonzero coefficients, which are only in \{-1, 1\} other than in \{-255, -254, ..., 254, 255\}.

For example, coefficients 0, 0, 0, 1, 0, 0, 0, -1, 0, 0, 0, 1 at high frequencies can be divided into four small categories, which are \{0, 0, 0, 1\}, \{0, 0, 0, 0\}, \{-1\}, and \{0, 0, 1\}. The encoding comparison is shown in Table 6.

Run/size joint code refers to the bitstream of run/size, the length is different because Run/Size is jointly encoded in JPEG code and Rxy code, respectively, and mostly, the encoded length in Rxy code is smaller than that in JPEG code except run/size 0/1. Value code refers to the bitstream of value, because size is always 1, one bit can encode value so that value code is the same both in Rxy and JPEG code. Then, joint code is the synthesis of run/size joint code and value code, and it shows that the size of joint code in Rxy code is mostly smaller than that in JPEG code.

For coefficients 0, 0, 0, 1, run is 3, size is 1, and value is 1. In JPEG code, run/size is jointly encoded as “111010” and value is encoded as “1.” In Rxy code, run/size is jointly encoded as “100” and value is encoded as “1.” For coefficients 0, 0, 0, 0, -1, run is 4, size is 1, and value is -1. In JPEG code, run/size is jointly encoded as “111011” and value is encoded as “0.” In Rxy code, run/size is jointly encoded as “101” and value is encoded as “0.” For coefficient -1, run is 0, size is 1, and value is -1. In JPEG code, run/size is jointly encoded as “000” and value is encoded as “0.” In Rxy code, run/size is jointly encoded as “011” and value is encoded as “1.” For coefficients 0, 0, 1, run is 2, size is 1, and value is 1. In JPEG code, run/size is jointly encoded as “11100” and value is encoded as “1.” In Rxy code, run/size is jointly encoded as “011” and value is encoded as “1.” The bitstream of these coefficients is “11101011101100 00111001” with 23 bits in JPEG code and “1001101000 100111” with 16 bits in Rxy code. Seven bits are reserved compared to JPEG code.

In the encoding procedure at high frequencies, the zero interval is mostly less than 16 such that 16 Rxy codes can be used. However, in JPEG code, at least 50 VLCs are listed considering the cases that run is more than 10 and value is larger than 1. These cases occur rarely, and even if there is such a case, we can mark the block in which the value is larger than 1 at high frequencies individually. The encoding efficiency in JPEG code is reduced at the cost of considering all cases in which run varies in [0, 16] and value varies in [1, 255]. In contrast, only considering the cases in which run varies in [0, 16] and value is 1, our Rxy code is highly efficient at high frequencies.

### 3.2 Additional Data Embedding

We decode a bitstream into quantized DCT blocks, cluster neighboring 8 × 8 blocks in the spatial domain into a 32 × 32 block, and search for the highest frequency at which the absolute value is larger than one in each 8 × 8 block. After calculating all 16 frequencies in all sixteen 8 × 8 blocks, we select the largest value among them as the termination point in the large 32 × 32 block.

In a 512 × 512 grayscale image, there are a few special blocks in which coefficients larger than one are still at the 64th frequency. These blocks degrade the performance of the proposed method; hence, we record these special blocks and encode their indexes as 16 bits. There are 64 × 64 (i.e., 4096) blocks, and \(\log_2(64 \times 64)\) (i.e., 12) bits can be used as a special block index. Using four bits of padding “0000,” each special block index is encoded using two bytes and inserted as appended data in a JPEG header.

We assume that the number of special blocks is \(m_N\). Hence, there are \(m_N \times 2\) bytes corresponding to blocks \(m_1, m_2, ..., m_{m_N}\), and we treat the indexes as appended data. This case occurs rarely, and only a few blocks are unsuitable when the quality factor (QF) is 90.

A comment (COM) marker, 0xFFFE, is inserted in a file header, and it is followed by additional data consisting of secret and appended data. A 512 × 512 grayscale image is clustered into 256 large blocks, and we must record these 256 termination points using 256 × 6 bits as appended data. Six bits are used to represent the termination point, which is in [1, 64], and when the six bits are “111111,” they can be transformed to 63 in decimal format, which implies that the large block is invalid because the encoded size is higher than the original size. The size of the secret data is also required in the comment data. The constitution of the additional data is shown in Fig. 3.

| Categories | Run | Size | Run/size joint code | Value | Value code | Joint code |
|------------|-----|------|---------------------|-------|------------|------------|
|            |     |      | JPEG | Rxy     | JPEG | Rxy | JPEG | Rxy |
| 0, 0, 0, 1 | 3   | 1    | 111010 | 100    | 1   | 1   | 111010 | 1  |
| 0, 0, 0, 0, -1 | 4 | 1 | 111011 | 101 | -1 | 0 | 0 | 111011, 0 |
| -1         | 0   | 1    | 00    | 001   | -1 | 0   | 0   | 00, 0 |
| 0, 0, 1    | 2   | 1    | 11100 | 011   | 1   | 1   | 11100 | 1  |

Table 6 Encoding comparison of DCT coefficients.
Moreover, bytes refer to secret size, and the last byte refers to the overall additional data size. The following two bytes refer to COM marker 0xFFFE, the second two bytes only the coefficients in

In our method, we consider two as the threshold because

Based on the assumption that nonzero coefficients are small with a sparse distribution, we can also consider three as a threshold other than two. However, more bits are required to encode nonzero coefficients. Values \{-1,1\} are mostly used at high frequencies, and the number of occurrences of values \{-2,2\} is considerably less compared to \{-1,1\}. Hence, more bits are used to encode values \{-1,1\}. Even if termination points are slightly smaller, more special blocks that are not suitable for Rxy code are marked and less spare space is reserved to insert secret data. Owing to this, encoding efficiency is reduced significantly. In our method, the main idea is to take as few bits as possible to jointly encode the most used nonzero coefficients and the zero interval. When the threshold increases, more cases that occur rarely will be considered and information entropy will be reduced considerably.

3.3 Threshold Criterion

In our method, we consider two as the threshold because only the coefficients in \{-1,1\} are considered in Rxy code. Based on the assumption that nonzero coefficients are small with a sparse distribution, we can also consider three as a threshold other than two. However, more bits are required to encode nonzero coefficients. Values \{-1,1\} are mostly used at high frequencies, and the number of occurrences of values \{-2,2\} is considerably less compared to \{-1,1\}. Hence, more bits are used to encode values \{-1,1\}. Even if termination points are slightly smaller, more special blocks that are not suitable for Rxy code are marked and less spare space is reserved to insert secret data. Owing to this, encoding efficiency is reduced significantly. In our method, the main idea is to take as few bits as possible to jointly encode the most used nonzero coefficients and the zero interval. When the threshold increases, more cases that occur rarely will be considered and information entropy will be reduced considerably.

3.4 Illustration of Concealment

We consider an 8 × 8 DCT block, which is shown in Fig. 4, to illustrate concealment. The original block is shown in Fig. 4(a). As block size is small and cannot supply sufficient space to mark the secret size and termination point, we assume that a receiver has a known secret size and termination point. A large block size of 32 × 32 in our method implies that only one termination point is required other than 16 points, and the secret size is encoded with two bytes in a file header.

When the absolute value is larger than one, the highest frequency is the 16th frequency and the coefficient at this point is 0. Thus, we consider the 16th frequency as a termination point, which is shown in Fig. 4(b).

Below the termination point, coefficients 12, 17, 15, -9, 8, 10, -5, 0, 0, 3, 0, 4, 0, 0, 0, -2 are encoded in JPEG code as “1011010110100011011111111011001101100010110110000111111111011001110111101110111011000000.”

Above this point, the coefficients are recoded in Rxy code as follows:

For coefficients 0, 0, 0, 0, 1, JPEG code is “1110111,” whereas Rxy code is “1011.”

For coefficients 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, JPEG code is “1111110010,” whereas Rxy code is “1111000.”

As the end marker, JPEG code is “1010,” whereas Rxy code is “000.”

Below the termination point, coefficients are encoded in JPEG code, and this part is shown in Fig. 4(c). The most important coefficients are retained, and only small coefficients at high frequencies are removed so that basic information is retained.

Recompressed data are “1011010110100011011111111110110011101111111110110011000000.” The first part is in JPEG code, and the second part is in Rxy code, whereas original data are “101101011010001101111111111011001110111111111011001110111111111101110111101111111101010.”

Above the termination point, the original JPEG code is “1111010111110011011010,” which is 21 bits in length, whereas JPEG code is “101111101000000,” which is only 14 bits in length. Hence, this encoding reserves seven bits that can be used to embed secret data.

If we consider “1010001” as secret data, the final inserted bitstream will be “101000110101010101000111111111111111011001110111111111110011011111111111111000000.”

In a receiver, if secret size and the termination point are obtained, secret data “10100011” can be extracted in a file header. Then, the compressed data part is decoded in JPEG code from the lowest frequency, and it is decoded in Rxy code when the decoded frequency is above the 16th frequency. The DCT coefficients above the termination point are recovered in a lossless manner and can be encoded in

In Fig. 3, the first part is secret data and the second part is appended data. The total EC, \( M \), is \( 7 + m_1 + m_2 + m_3 \). Seven bytes are added as marked data, where the first two bytes refer to COM marker OxFFFE, the second two bytes refer to the overall additional data size, the following two bytes refer to secret size, and the last byte refers to the number of special blocks. Moreover, \( m_1 \) denotes secret data, \( m_2 \) denotes special 8 × 8 block indexes, and \( m_3 \) denotes 256 termination points.

![Fig. 3 Format of the additional data.](image)

![Fig. 4 8 × 8 DCT block for concealment: (a) original block, (b) termination point in frequency map, (c) retained block, and (d) recovered block.](image)
JPEG code again. The block is recovered lossless, as shown in Fig. 4(d).

### 3.5 Hiding Secret Data

Using the scheme described above, the following steps are used to hide secret data in a JPEG image by employing the proposed method:

1. Decompose the original JPEG image into its DCT coefficients and establish the corresponding relationship between frequencies and a bitstream in each \(8 \times 8\) block.
2. Cluster 16 neighboring blocks into large blocks of size \(32 \times 32\) and consider 16 corresponding bitstreams as a set.
3. Consider the highest frequency at which the absolute value is larger than one as the termination point for each \(8 \times 8\) block.
4. Use the maximum of the termination points in the 16 neighboring blocks as the termination point of the large \(32 \times 32\) block and record special block indexes.
5. In each block, encode the coefficients below the termination point using JPEG code and the coefficients above the point using the proposed Rxy code.
6. Calculate the recompressed file size and compare it with the original JPEG file size to obtain EC.
7. Insert marker 0xFFFE and then embed appended and secret data in a JPEG header.
8. Synthesize the modified header and recoded data part as a marked JPEG bitstream.

A block diagram of the embedding procedure is shown in Fig. 5.

### 3.6 Extraction of Secret Data to Restore Marked Image

The secret data are extracted in one step. We are not required to decompress the JPEG file. We need to only extract the bitstream with marker 0xFFFE in a file header, and a secret range is obtained according to the two following bytes.

The recovery of a marked image is the opposite of data embedding in that the most important procedure is the decomposition of DCT coefficients. The recovery consists of the following steps:

1. Extract appended data from the comment part in a file header. The special block indexes and termination points are listed within additional data.
2. Divide the compressed file into the header and data body using marker 0xFFFD and obtain image size from the file header.
3. Construct a mapping using the image size and calculate the new block indexes.
4. With the indexes and mapping, decompose a JPEG file into its \(8 \times 8\) DCT blocks, except for the special blocks, as recorded in the file header. Decode the coefficients below the termination point in JPEG code and the coefficients above the point in Rxy code.
5. Decode the DCT coefficients to restore the image.

The extraction and recovery procedures are shown in Fig. 6.

### 4 Results

We obtained eight grayscale images scaled to eight bits from the standard library\(^5\) for the evaluation and compressed them using \(QF = 90, 80, 70, 60, 50, 40, 30, 20,\) and 10 to create original host images. Secret data were created using a random bitstream.

One measure used in this work is EC, which denotes the size of concealed secret data. The other measure is the peak signal-to-noise ratio (PSNR), which compares retained images with the original JPEG images. In the \(8 \times 8\) blocks of retained images, the coefficients below the termination point are only encoded in JPEG code and the coefficients above the point are zero such that the bitstream of low-frequency coefficients is not modified.

---

**Fig. 5** Proposed embedding procedure.

**Fig. 6** Proposed data extraction and recovery procedure.
The platform used in the experiments was a Windows 7 64-bit system running MATLAB 2013b and equipped with a four-core E3V3 Xeon processor at 3.3 GHz.

There are $64 \times 64$ blocks in a $512 \times 512$ image, and we used two as the threshold in most experiments. The experimental results are shown in Table 7. Our approach outperforms the methods of Qian and Zhang, Hu et al., Chang et al., and Qiu et al. with respect to EC.

Table 7 Comparison of EC (bits) versus QF for the proposed and previous methods.

| Image | Method       | 10 | 20 | 30 | 40 | 50 | 60 | 70 | 80 | 90 |
|-------|--------------|----|----|----|----|----|----|----|----|----|
| Airplane | Qian and Zhang | 615 | 639 | 722 | 737 | 760 | 536 | 487 | 404 | 464 |
|        | Hu et al.     | 791 | 764 | 904 | 961 | 886 | 860 | 741 | 553 | 595 |
|        | Qiu et al.    | 1486 | 1066 | 1097 | 1215 | 1179 | 1160 | 1069 | 881 | 913 |
|        | Chang et al.  | 395 | 527 | 601 | 675 | 823 | 943 | 1003 | 1204 | 1275 |
|        | Proposed      | 1259 | 2029 | 2591 | 3043 | 3403 | 3726 | 4349 | 4995 | 9034 |
| Bridge | Qian and Zhang | 733 | 675 | 614 | 755 | 781 | 501 | 436 | 306 | 350 |
|        | Hu et al.     | 1802 | 999 | 713 | 860 | 1035 | 714 | 817 | 596 | 615 |
|        | Qiu et al.    | 2473 | 1514 | 1099 | 1193 | 1269 | 926 | 1002 | 782 | 858 |
|        | Chang et al.  | 307 | 374 | 425 | 462 | 510 | 542 | 580 | 634 | 660 |
|        | Proposed      | 3097 | 3222 | 4213 | 5006 | 5478 | 6075 | 7146 | 9041 | 11,288 |
| Baboon | Qian and Zhang | 4954 | 2263 | 1278 | 1588 | 1080 | 1005 | 1082 | 615 | 346 |
|        | Hu et al.     | 6576 | 3478 | 1758 | 1790 | 1268 | 1304 | 1353 | 757 | 681 |
|        | Qiu et al.    | 7169 | 3767 | 1970 | 2077 | 1482 | 1582 | 1634 | 1033 | 939 |
|        | Chang et al.  | 1145 | 1207 | 1213 | 1275 | 1233 | 1420 | 1499 | 1555 | 1686 |
|        | Proposed      | 2066 | 3843 | 4726 | 5465 | 6217 | 7001 | 8166 | 9858 | 10,525 |
| Boat   | Qian and Zhang | 484 | 388 | 522 | 481 | 522 | 714 | 370 | 510 | 978 |
|        | Hu et al.     | 950 | 551 | 691 | 571 | 687 | 811 | 679 | 1000 | 1929 |
|        | Qiu et al.    | 1635 | 1185 | 1242 | 1020 | 1079 | 1139 | 962 | 1249 | 2172 |
|        | Chang et al.  | 359 | 409 | 456 | 562 | 722 | 854 | 930 | 1032 | 1204 |
|        | Proposed      | 1255 | 2410 | 3041 | 3563 | 4118 | 4733 | 5534 | 6816 | 10,046 |
| Couple | Qian and Zhang | 893 | 275 | 316 | 394 | 461 | 362 | 432 | 568 | 834 |
|        | Hu et al.     | 1128 | 467 | 417 | 552 | 588 | 477 | 650 | 818 | 1428 |
|        | Qiu et al.    | 1806 | 1080 | 970 | 1046 | 1035 | 3206 | 1030 | 1101 | 1677 |
|        | Chang et al.  | 385 | 445 | 541 | 620 | 747 | 854 | 921 | 1088 | 1204 |
|        | Proposed      | 806 | 1434 | 1962 | 2375 | 2815 | 4913 | 5851 | 7491 | 11,265 |
and EC is larger. However, EC is quite small in a few categories because most VLCs are used and no extra unused VLCs exist. Qiu et al.\textsuperscript{32} constructed a method that combined advanced VLC mapping and VLC arrangement in each category. Several unused VLCs are arbitrarily mapped to one used VLC such that mapping capacity is improved further. However, there are at most 162 available VLCs, and the number of VLCs is insufficient. Chang et al.\textsuperscript{36} decompressed JPEG images to quantified DCT blocks, extracted data from the two least significant bit planes of DCT coefficients, and reshaped the data to a bitstream. Chang et al.\textsuperscript{36} recompressed a bitstream in a lossless manner using the fact that data in the two planes occur in bias. However, as file size increases, the bias in \{0,1\} decreases to 0.5 such that no more spare space can be provided.

In this work, we first select coefficients in \(f^{-1}g^1\) with 1-bit encoding, which are mostly at high frequencies so that no degradation of the original image occurs. A zero interval frequently exists between these nonzero values in \(f^{-1}g^1\), and then, extra bits are required to encode the zero interval. We recode these zero coefficients, thereby substantially reducing redundancy. In the proposed method, the correlation among values \{\(-1,1\)\} are fully utilized and EC is clearly larger than state-of-the-art methods. When QF is low, the quantization step is large. In this case, even though nonzero coefficients are small with numerous zero interval and the termination point is quite low, more seriously, the number of nonzero coefficients in \(f^{-1}g^1\) above the termination point is low. Hence, EC is insufficient when QF is small. Absolute values increase with increasing QF. Even though the termination point is higher, the number of nonzero coefficients in \{\(-1,1\)\} above the termination point increases, and EC increases substantially. When QF increases further, there is still general improvement in EC. However, compressed file size increases considerably. Additionally, in certain extremely large blocks, termination points are high and recompressed file size might be larger than the original size. Thus, these large blocks become invalid.

| Image   | Method                              | 10  | 20  | 30  | 40  | 50  | 60  | 70  | 80  | 90  |
|---------|-------------------------------------|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| Clown   | Qian and Zhang\textsuperscript{30}  | 965 | 388 | 467 | 526 | 617 | 660 | 559 | 634 | 594 |
|         | Hu et al.\textsuperscript{31}       | 1180| 584 | 705 | 863 | 865 | 884 | 907 | 938 | 1083|
|         | Qiu et al.\textsuperscript{32}      | 1858| 1155| 1108| 1250| 1262| 3559| 1261| 1238| 1376|
|         | Chang et al.\textsuperscript{36}    | 398 | 342 | 452 | 479 | 528 | 596 | 645 | 778 | 915 |
|         | Proposed                            | 1395| 2549| 3306| 3932| 4453| 5174| 5974| 7547| 10,101|
| Peppers | Qian and Zhang\textsuperscript{30}  | 391 | 380 | 491 | 383 | 389 | 473 | 295 | 280 | 764 |
|         | Hu et al.\textsuperscript{31}       | 567 | 517 | 717 | 493 | 557 | 652 | 455 | 374 | 1562|
|         | Qiu et al.\textsuperscript{32}      | 1263| 989 | 1078| 923 | 928 | 964 | 774 | 695 | 1775|
|         | Chang et al.\textsuperscript{36}    | 245 | 291 | 302 | 385 | 450 | 568 | 820 | 960 | 1085|
|         | Proposed                            | 1158| 1958| 2558| 3019| 3500| 3955| 4640| 6036| 10,155|
| Hill    | Qian and Zhang\textsuperscript{30}  | 916 | 366 | 443 | 538 | 630 | 670 | 581 | 743 | 830 |
|         | Hu et al.\textsuperscript{31}       | 1105| 557 | 686 | 902 | 926 | 925 | 948 | 1084| 1473|
|         | Qiu et al.\textsuperscript{32}      | 1785| 1135| 1084| 1248| 1254| 1254| 1268| 1339| 1742|
|         | Chang et al.\textsuperscript{36}    | 441 | 501 | 561 | 594 | 674 | 785 | 841 | 968 | 1041|
|         | Proposed                            | 983 | 1996| 2665| 3316| 3928| 4499| 5172| 6188| 8526|
| Average | Qian and Zhang\textsuperscript{30}  | 1244| 672 | 607 | 675 | 655 | 615 | 530 | 510 | 645 |
|         | Hu et al.\textsuperscript{31}       | 1762| 990 | 824 | 874 | 852 | 828 | 819 | 765 | 1171|
|         | Qiu et al.\textsuperscript{32}      | 2434| 1486| 1206| 1247| 1186| 1724| 1125| 1040| 1432|
|         | Chang et al.\textsuperscript{36}    | 459 | 512 | 569 | 632 | 711 | 820 | 905 | 1027| 1134|
|         | Proposed                            | 1502| 2430| 3133| 3715| 4239| 5010| 5854| 7247| 10,118|
We can also analyze the influence of termination points on EC. We calculate the valid termination points as the average for the blocks that can be recompressed. The average points for eight test images are shown in Table 8. When QF is low, termination points are also low, and original information is concentrated at low frequencies. There are only a few nonzero coefficients in \( f_{-1} \) at high frequencies, and EC is relatively small. Even though termination points become higher as QF increases, more details are retained so that more nonzero coefficients occur at high frequencies and the interval between them is smaller. The number of nonzero coefficients increases and the number of coefficients in \( f_{-1} \) above the termination point increases to a considerably larger extent. Hence, EC increases significantly. Even though there are a few special blocks, where the nonzero values at the 64th frequency are larger than one, we can record these block indexes in a file header as appended data. There are only four blocks that are not suitable for our method in JPEG image Baboon at QF 90, whereas EC in JPEG image Baboon is larger than 10,000 bits. At the same QF, the complexity of images influences termination points. This implies that the nonzero coefficients at high frequencies indicate complexity; that is, the number of small nonzero coefficients increases at high frequencies. This can provide more spare embedding space at the minor cost of increase in termination points. The EC of Baboon is larger than that of airplane at the same QF because the termination points of Baboon are higher on average and more coefficients in \( f_{-1} \) at high frequencies are retained after quantization.

In a receiver, host JPEG coefficients are recovered after the extraction of secret data. It is important to decompress a bitstream into a block using a termination point so that the bitstream below the point is decoded using JPEG code and the bitstream above the point is decoded using Rxy code. The recovered images are lossless with respect to the host JPEG images. The retained images are only constituted with low-frequency coefficients using JPEG code such that the corresponding original bitstream is not modified. Moreover, retained images can be displayed normally because the coefficients above the termination point are zero and the corresponding bitstream is removed. The PSNRs of retained images with respect to host JPEG images are shown in Table 9. The retained and original host JPEG images at QF 50 are also shown in Fig. 7. The figure shows that even

| QF  | 10  | 20  | 30  | 40  | 50  | 60  | 70  | 80  | 90  |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| Airplane | 6.8 | 10.2 | 12.0 | 13.1 | 14.1 | 15.2 | 17.6 | 19.7 | 35.5 |
| Baboon   | 8.2 | 15.0 | 18.5 | 21.3 | 24.3 | 27.3 | 31.9 | 38.5 | 52.3 |
| Boat     | 5.9 | 10.0 | 12.1 | 14.0 | 16.2 | 18.5 | 21.6 | 26.6 | 39.4 |
| Bridge   | 7.1 | 12.6 | 16.5 | 19.6 | 21.5 | 23.8 | 27.9 | 35.3 | 46.8 |
| Couple   | 6.1 | 9.9  | 13.1 | 15.3 | 17.4 | 19.2 | 22.9 | 29.3 | 44.9 |
| Hill     | 4.6 | 8.3  | 10.9 | 13.4 | 15.9 | 18.1 | 20.4 | 24.2 | 33.3 |
| Clown    | 6.1 | 10.6 | 13.2 | 15.5 | 17.5 | 20.2 | 23.3 | 29.6 | 40.7 |
| Peppers  | 5.3 | 8.0  | 10.4 | 12.0 | 13.7 | 15.5 | 18.1 | 23.6 | 44.0 |

| QF  | 10  | 20  | 30  | 40  | 50  | 60  | 70  | 80  | 90  |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| Airplane | 32.29 | 34.91 | 36.51 | 38.29 | 39.08 | 39.41 | 40.76 | 41.48 | 37.97 |
| Baboon   | 26.84 | 28.97 | 29.09 | 29.11 | 29.61 | 30.36 | 31.68 | 33.77 | 31.48 |
| Boat     | 31.10 | 33.64 | 34.72 | 35.49 | 36.16 | 37.38 | 38.74 | 39.99 | 41.41 |
| Bridge   | 28.77 | 30.95 | 31.88 | 32.35 | 31.89 | 31.83 | 32.55 | 34.29 | 39.23 |
| Couple   | 17.51 | 34.06 | 33.99 | 37.12 | 35.03 | 38.62 | 32.65 | 37.72 | 64.63 |
| Hill     | 31.99 | 33.97 | 35.30 | 36.38 | 37.26 | 37.84 | 37.94 | 38.81 | 40.67 |
| Clown    | 30.92 | 33.04 | 33.95 | 34.22 | 34.48 | 35.25 | 35.86 | 37.60 | 41.01 |
| Peppers  | 33.21 | 36.10 | 37.04 | 39.12 | 37.89 | 40.86 | 35.09 | 37.98 | 59.32 |
if the coefficients at high frequencies are removed, the basic information is still preserved so that it is difficult to recognize the distortion of host JPEG images. When QF decreases to 10, the PSNRs of retained images are mostly approximately 30 dB. Less information is preserved when the termination point decreases because of the decline in host JPEG image quality with higher quantization and more zero coefficients at high frequencies.

We perform supplemental analysis using three as the threshold. Hence, the coefficients in \(-2, -1, 1, 2\) are used for encoding. In the Rxy code for this case, the three bits of run are not modified. The difference is that the nonzero coefficients with values in \(-2, -1, 1, 2\) are encoded using two bits, as shown in Table 10. We search for the highest termination point in each block, just as in the basic proposed method, and recode coefficients using the redefined Rxy code to leave spare space. The average EC for both modes is compared in Fig. 8. In the one-bit mode, EC is improved such that even though nonzero coefficients are limited to \([-1, 1]\), they are encoded with a shorter length when the

### Table 10 Nonzero coefficient mapping for two bits.

| XY | Value |
|----|-------|
| 00 | \(-2\) |
| 01 | \(-1\) |
| 10 | 1     |
| 11 | 2     |
interval is longer than zero. However, in the two-bit mode, when nonzero coefficients are in \{-2, -1, 1, 2\}, encoded length is expanded for intervals of 0 and 1. Hence, we use two as the threshold. This causes termination points to be higher but improves encoding efficiency.

We also compared our results on the standard Kodak dataset. There were 24 gray images with a size of 3072 × 2048, which were compressed with QF 90, 80, 70, 60, 50, 40, 30, 20, and 10 as host JPEG images. There were 3072/8 × 2048/8 (i.e., 98,304) blocks, and \log_2(98304) (i.e., 17) bits could be used as a special block index. With seven bits padding “0000000,” three bytes were used to represent a special block index. The performance is shown in Fig. 9. The average EC at each QF exhibits considerable improvement compared with previous works in general. In addition, when nonzero is in \{-1, 1\}, the image size of the standard Kodak dataset is 24 times that of the USC-SIPI dataset, and the average EC is more than 10 times that of the USC-SIPI dataset. This shows that image size and resolution strongly affect EC. When the resolution is higher, the correlation between adjacent pixels is stronger, such that most nonzero DCT coefficients are concentrated at low frequencies after JPEG compression. No sufficient spare space is reserved in a block. However, with blocks number increasing greatly, average EC in the Kodak dataset still has a great improvement. The concealment in the JPEG image Mountain with QF 50 is shown in Fig. 10. Figure 10(a) is the retained image, and Fig. 10(b) is the host image. The EC in the concealment is 78881 bits, and the average termination point is 12.88. Surprisingly, the PSNR is 40.41 dB, and the distortion cannot be recognized by human eyes. The concealment in the JPEG image Wall with QF 50 is shown in Fig. 11. Figure 11(a) is the retained image, and Fig. 11(b) is the host image. The EC in the concealment is 55526 bits, the average termination point is 9.49, and also surprisingly, the PSNR is 39.13 dB, and it also confirms our assumption.

5 Conclusion

We proposed an LDH scheme for JPEG bitstreams, in which the original bitstreams are compressed in a lossless manner to achieve efficient balancing of encoding DCT coefficients at high and low frequencies. JPEG bitstreams are fragile to
attacks, and even a bit error can lead to decoding failure. The LDH in our method ensures that marked images can be restored in a lossless manner such that no original information is missing in concealment. The JPEG Huffman table prioritizes the preservation of low frequencies. Hence, we also propose a segment coding scheme in which the coefficients below a termination point are encoded in JPEG code and the coefficients above the point are encoded in Rxy code. The Rxy encoding scheme retains nonzero coefficients that are only in \{-1, 1\}, and it is only necessary to encode the number of zeros such that the encoding length is decreased. Compressed data are further divided into sets corresponding to large blocks in sizes of 32 × 32, which are composed of 16 blocks to obtain the optimal termination point. Thus, spare space is reserved, appended data and secret data are inserted in a JPEG header following COM marker 0xFFFE. Secret data can be extracted in a blind manner from a file header, and retained images, which consist of the coefficients below the termination points and are only encoded in JPEG code, can be displayed at high visual quality. Furthermore, marked images are restored in a lossless manner with low-frequency coefficients using JPEG code and high-frequency coefficients using Rxy code. The comparison of experimental results shows that the proposed concealment is lossless and EC is improved without file expansion. EC is sufficient for a JPEG image such that if there is side information belonging to host JPEG images, particularly target attributes and diagnostic reports, it can be inserted in the JPEG bitstream to be conveyed together. This technique can be widely applied in the fields of law enforcement, medical systems, and military imagery. However, when an image is compressed with QF in \{10, 20\}, image quality is just acceptable for human eyes and the number of nonzero coefficients in \{-1, 1\} is small, such that EC improves negligibly or even decreases. In future work, we will formulate another Rxy code that is suitable for JPEG images with a low QF to improve EC further.

Acknowledgments
This work was supported by the Natural Science Foundation of China (Grant No. 61372175) and the National Key Laboratory Foundation of China (Grant Nos. 20188SFNKLSTM-13, HTKJ2019KL504006, and HTKJ 2019KL504007).

Fig. 11 Concealment in JPEG Image Wall from the Kodak dataset.40 (a) retained image and (b) host JPEG image.

References
1. A. Khan et al., “A recent survey of reversible watermarking techniques,” Inf. Sci. 279, 251–272 (2014).
2. M. Chandan and M. Pandey, “A review on current methods and application of digital image steganography,” Int. J. Multidiscip. Approach Stud. 2(2), 163–170 (2015).
3. W. L. Tai, C. M. Yeh, and C. C. Chang, “Reversible data hiding based on histogram modification of pixel differences,” IEEE Trans. Circuits Syst. Video Technol. 19(6), 906–910 (2009).
4. Y. Hu, H. K. Lee, and J. Li, “DE-based reversible data hiding with improved overflow location map,” IEEE Trans. Circuits Syst. Video Technol. 19(2), 250–260 (2009).
5. C. C. Lin, W. L. Tai, and C. C. Chang, “Multilevel reversible data hiding based on histogram modification of difference images,” Pattern Recognit. 41(12), 3582–3591 (2008).
6. S. Lee, C. D. Yoo, and T. Kalker, “Reversible image watermarking based on integer-to-integer wavelet transform,” IEEE Trans. Inf. Forensics Secur. 2(3), 321–330 (2007).
7. F. Y. Shih, Digital Watermarking and Steganography: Fundamentals and Techniques, 2nd ed., pp. 324–327, CRC Press, Boca Raton (2017).
8. C. Chang, Y. Hsieh, and C. Lin, “Lossless data embedding with high embedding capacity based on declustering for VQ-compressed codes,” IEEE Trans. Inf. Forensics Secur. 2(3), 341–349 (2007).
9. W. L. Tai and C. C. Chang, “Data hiding based on VQ compressed images using hamming codes and declustering,” Int. J. Innovative Comput. Inf. Control 5(7), 576–584 (2009).
10. P. Y. Lin and C. S. Chan, “Invertible secret image sharing with steganography,” Pattern Recognit. Lett. 31(13), 1887–1893 (2010).
11. K. Wang, Z. M. Lu, and Y. J. Hu, “A high capacity lossless data hiding scheme for JPEG images,” J. Syst. Software 86(1), 1965–1975 (2013).
12. R. Bhardwaj, “Enhanced encrypted reversible data hiding algorithm with minimum distortion through homomorphic encryption,” J. Electron. Imaging 27(2), 023017 (2018).
13. S. Yi, Y. Zhou, and Z. Hua, “Reversible data hiding in encrypted images using adaptive block-level prediction-error expansion,” Signal Process. Image Commun. 64, 78–88 (2018).
14. J. Fridrich, M. Goljan, and R. Du, “Lossless data embedding for all image formats,” Proc. SPIE 4675, 572–583 (2002).
15. J. Tian, “Reversible data embedding using a difference expansion,” IEEE Trans. Circuits Syst. Video Technol. 13(8), 890–896 (2003).
16. D. M. Thodi and J. J. Rodriguez, “Expansion embedding techniques for reversible watermarking,” IEEE Trans. Image Process. 16(3), 721–730 (2007).
17. Z. Ni et al., “Reversible data hiding,” IEEE Trans. Circuits Syst. Video Technol. 16(3), 354–362 (2006).
18. C. Y. Lin, “A reversible data transform algorithm using integer transform for privacy-preserving data mining,” J. Syst. Software 117, 104–112 (2016).
19. International Telecommunication Union, “Information technology-digital compression and coding of continuous-tone still images-requirements and guidelines,” CCITT Recommendation T.81 (1992).
20. A. M. Raud et al., “JPEG image compression using discrete cosine transform—a survey,” Int. J. Comput. Sci. Eng. Surv. 5(2), 39–47 (2014).
21. G. Xuan et al., “Reversible data hiding for JPEG images based on histogram pairs,” Lect. Notes Comput. Sci. 4633, 713–727 (2007).
22. H. Sakai, M. Kuribayashi, and M. Morii, “Adaptive reversible data hiding for JPEG images,” in Int. Symp. Inf. Theory and Its Appl., IEEE, Auckland, New Zealand, pp. 1–6 (2008).
A. Nikolaidis, “Reversible data hiding in JPEG images utilising zero quantised coefficients,” *IET Image Process.*, 9(7), 560–568 (2015).

F. Huang et al., “Reversible data hiding in JPEG images,” *IEEE Trans. Circuits Syst. Video Technol.*, 26(9), 1610–1621 (2016).

Y. Liu and C. C. Chang, “Reversible data hiding for JPEG images employing all quantized non-zero AC coefficients,” *Displays*, 51, 51–56 (2018).

D. Hou et al., “Reversible data hiding in JPEG image based on DCT frequency and block selection,” *Signal Process.*, 148, 41–47 (2018).

C. C. Chang et al., “Reversible hiding in DCT-based compressed images,” *Inf. Sci.*, 177(13), 2768–2786 (2007).

B. G. Mobasseri et al., “Data embedding in JPEG bitstreams by code mapping,” *IEEE Trans. Image Process.*, 19(4), 958–966 (2010).

X. Zhang et al., “Reversible fragile watermarking for locating tampered blocks in JPEG images,” *Signal Process.*, 90(12), 3026–3036 (2010).

Z. Qian and X. Zhang, “Lossless data hiding in JPEG bitstreams,” *J. Syst. Software*, 85(2), 309–313 (2012).

Y. Hu, K. Wang, and Z. M. Lu, “An improved VLC-based lossless data hiding scheme for JPEG images,” *J. Syst. Software*, 86(8), 2166–2173 (2013).

Y. Qiu et al., “Lossless data hiding in JPEG bitstreams using alternative embedding,” *J. Visual Commun. Image Represent.*, 52, 86–91 (2018).

Z. Qian, X. Zhang, and S. Wang, “Reversible data hiding in encrypted JPEG bitstreams,” *IEEE Trans. Multimedia*, 16(5), 1486–1491 (2014).

Z. Qian et al., “Separable reversible data hiding in encrypted JPEG bitstreams,” *IEEE Trans. Dependable Secure Comput.*, 15(6), 1055–1067 (2018).

K. Ma et al., “Reversible data hiding in encrypted images by reserving room before encryption,” *IEEE Trans. Inf. Forensics Secur.*, 8(3), 553–562 (2013).

J. C. Chang, Y. Z. Lu, and H. L. Wu, “A separable reversible data hiding scheme for encrypted JPEG bitstreams,” *Signal Process.*, 133, 135–143 (2017).

S. R. Zhang, Q. D. Li, and Q. Zhou, “A JPEG image information hiding method based on datastreams,” CN. Patent No. 107071455A (2017).

Y. Wu and R. H. Deng, “Zero-error watermarking on jpeg images by shuffling Huffman tree nodes,” in *Visual Commun. and Image Process. (VCIP)*, IEEE, Tainan, Taiwan, pp. 1–4 (2011).

R. Franzen, “The standard Kodak dataset,” http://r0k.us/graphics/kodak/ (accessed 15 November 1999).

A. G. Weber, “The usc-sipi image database,” http://sipi.usc.edu/database/ (5 August 2009).

Mingming Zhang received his BS degree in simulation engineering from National University of Defense Technology, Changsha, China, in 2011. He received his MS degree in communication engineering from China Academy of Space Technology, Xi’an, China, in 2015. He is currently working toward his PhD in aircraft designing at China Academy of Space Technology, Xi’an, China. His research interests include data hiding and image compression.

Quan Zhou received his BS, MS, and PhD degrees in communication engineering from Xidian University, Xi’an, China, in 1986, 1989, and 1992, respectively. He is currently working as a full professor in the National Key Laboratory of Science and Technology on Space Microwave at China Academy of Space Technology, Xi’an, China. His research interests include digital image processing, data hiding, and image compression.

Yanlang Hu received his BS degree in electronic and information engineering from Hohai University, Nanjing, China, in 2005. He received his MS degree in communication engineering from China Academy of Space Technology, Xi’an, China, in 2008. He is currently working as a senior engineer in National Key Laboratory of Science and Technology on Space Microwave at China Academy of Space Technology, Xi’an, China. His research interests include computer vision and data transmission.