Reversible Data Hiding in JPEG Images Using Ordered Embedding

Zhenxing Qian[1], Shu Dai[1] and Boyang Chen[2]
[1] School of Communication and Information Engineering, Shanghai University
Shanghai, 200444, China
[e-mail: zxqian@shu.edu.cn]
[2] National Satellite Meteorological Center
Beijing, 100081, China
[e-mail: chenby@cma.gov.cn]
*Corresponding author: Boyang. Chen

Received September 17, 2016; revised November 28, 2016; accepted December 17, 2016; published February 28, 2017

Abstract

This paper proposes a novel method of reversible data hiding in JPEG images. After analyzing the JPEG features, we provide a new algorithm of selecting appropriate blocks and coefficients to carry secret messages. Instead of embedding data into the histogram of all coefficients, we propose a strategy of ordered embedding to hide data by histogram shifts in several rounds. On the recipient end, secret messages can be exactly extracted, and the original JPEG image can be losslessly recovered. Experimental results show that high embedding rate can be achieved with limited distortions. Most importantly, the marked JPEG file generated by the proposed method requires less storage size than state-of-the-art works.

Keywords: information hiding, reversible data hiding, histogram shift

This work was supported by Natural Science Foundation of China (Grant 61572308, Grant U1536108)

1. Introduction

Reversible data hiding (RDH) is an important branch of information hiding technology [1]. In RDH, a sender can imperceptibly hide secret information into a multimedia, and a receiver can not only extract the embedded data exactly but also recover the original multimedia without any error [2]. This technique is mainly used in the applications where no distortion is allowed on the multimedia, such as law forensics, military communication, medical imagery and fingerprint imagery [3-5]. Digital images are the popular type of multimedia. Many RDH algorithms have been proposed for uncompressed images [6-9]. However, these works cannot be applied to JPEG images, the most widely used format of digital images. Since there are few redundancies in JPEG images, RDH in JPEG always result in more distortions and file-size increments. Although JPEG encoding itself is lossy, users always hope not to introduce further degradation to a JPEG image while uploading. That is why lossless recovery is required. RDH in JPEG images in many applications. For example, in a cloud storage system, the images are stored by JPEG format. On the one hand, the user's information, labels, and timestamps can be embedded into the image for a convenient management. On the other hand, the user can extract all embedded information exactly and recover the original images without any loss.

In recent years, RDH in JPEG has attracted much attention and many methods have been proposed [10-18]. Generally, there are four categories of this technique. The first one is the compression based methods. In [10], variable length integer (VLI) codes are compressed losslessly to generate extra space for data embedding. However, the achievable embedding capacity is small. The second one is based on Huffman table modification. In [11], Qian et al. proposes a method of embedding data into JPEG bitstream by modifying Huffman code mapping. Both [10] and [11] can preserve the JPEG file size. Nevertheless, the embedding rate is limited. The third one is based on quantization table modification. In [12], some values within the original quantization table are divided by integers, and the corresponding quantized coefficients are multiplied by the same integers, the procedure of which is used to embed information. Although high visual quality and large embedding capacity can be achieved, the increment of file size is quite large. The fourth category is the histogram shift (HS) based method, which provides much more embedding rates than previous works. These algorithms are constructed by modifying DCT coefficients. A block of quantized DCT coefficients contains one DC coefficient and 63 AC coefficients. The DC coefficient is always kept unchanged, since modifying this coefficient would introduce considerable distortions for JPEG image. Most algorithms embed data into JPEG images by modifying AC coefficients [13-18]. We focus on the fourth type of RDH for JPEG because it has better embedding efficiency than the others.

In [13], Xuan et al. propose an RDH method for JPEG images by generating new histogram pairs to hide messages. This method was improved by Sakai et al., using the variance of DC coefficients among adjacent blocks to embed data into flat areas of an image [14]. Good performance can be achieved especially for the images with many smooth contents. In [15], zero coefficients are used to carry the secret message, while the other non-zero coefficients are shifted by one. This way, the original coefficients can be losslessly recovered on the recipient end. Nikolaidis proposes a different RDH for JPEG images [16], in which non-zero coefficients of the original image are not modified during data hiding. Secret bits are embedded by modifying the coefficients with zero values following the last non-zero coefficient in each block. This approach has good embedding rate. However, modifying these coefficients would result in a large increment of JPEG file size. Another method proposed in [17] uses all coefficients whose absolute values are L. These coefficients are either changed to L+1 or –L–1 to embed “1”, or remains unchanged to embed “0”. The value L can be adaptively chosen according to the message payload. More recently, Huang et al. propose to embed data into the coefficients with the magnitude “1”s, using a block selection strategy to choose some useful coefficients [18]. Compared with [17], quality of the marked image is better and the storage size is smaller. Besides, some other works on RDH in JPEG have been proposed to embed data into the encrypted domain [4].

In this paper, we propose an improved RDH method for JPEG images. We use the quantized AC coefficients with values “1” or “–1” to embed secret message, while the other non-zero coefficients are processed by shifting the histogram. During the embedding, we propose a new embedding strategy to select pairs with smaller zero run-lengths from the blocks with fewer pairs to accommodate the secret bits. Compared with state-of-the-art work in [18], embedding capacity and image quality are kept unchanged, while the file size increment of the marked image is reduced. The rest of the paper is organized as follows. We review the related works in Section II, and describe the proposed method in Section III. In Section IV, experimental results of the new method are provided. Section V concludes the paper.

2. Related Work

Since the proposed method focuses on RDH in JPEG images, we first provide an overview of JPEG compression in this section. To achieve a fair comparison, we also introduce a state-of-the-art RDH method for JPEG proposed in [18].

2.1 Overview of JPEG compression

Generally, the baseline JPEG includes three stages: block division, forward discrete cosine transform (FDCT), coefficient quantization, and entropy encoding. A diagram of the procedure is depicted in **Fig. 1**.

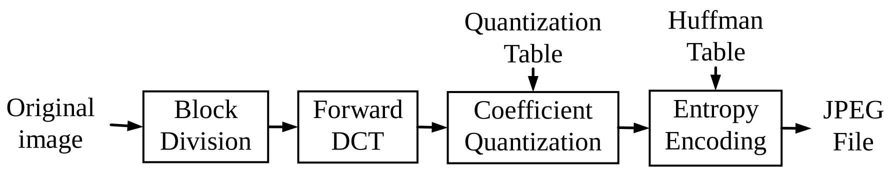


Fig. 1. Procedure of JPEG compression

For the ease of discussion, we mainly introduce the procedure of compressing a grayscale image. In the coder, an image is first divided into non-overlapped blocks, each of which containing 8×8 pixels. Each block is then transformed to coefficients by FDCT calculation. With a pre-defined quantization table, coefficients in each block are quantized and rounded to integers, resulting in many zeros in each block. After zigzag scanning the coefficients in each block, the quantized DC coefficients are encoded by differential pulse code modulation (DPCM), and the quantized AC coefficients are encoded by run length coding (RLC).

A special operation is used during RLC. The zigzag-scanned sequence containing AC coefficients in each block always contains several non-zero values and many zero values. Each sequence is represented by a number of (R, V) pairs, where R stands for *zero-run-length*, and V

the non-zero value following these zero coefficients. Each pair is named ZRV, *i.e.*, zero-run-value. For example, if a zigzag-scanned sequence containing AC coefficients in a block is {0, 2, 1,0, 0, 0, 3, 0, 1, 0, ..., 0}, ZRV pairs are represented by {(1, 2), (0, 1), (3, 3), (1, 1)}. Each ZRV pair is further converted into an intermediate symbol [(*R*, *C*) / *V*], where *C* ($1 \leq C \leq 10$) represents the category that V belongs to, i.e., the number of bits to represent *V*. After that, (*R*, *C*) are encoded with Huffman coding, and *V* encoded with VLI codes. Categories of different *V*'s and VLI codes are showed in **Table 1**. Suggested Huffman codes are listed in Table K.5 of JPEG Standard [19]. For example, for a ZRV pair (0, 5) whose intermediate symbol should be [(0, 3)/ 5], the encoded bits are therefore "100,101", where "3" is the category, "100" the Huffman code of (0, 3) according to Table K.5 in [19], and "101" the encoded bits of "5" according to **Table 1**.

Table 1. Categories of different V's and corresponding VLI codes

V	*C*	**VLI Codes**
– 1, 1	1	0,1
–3, –2, 2, 3	2	00,01,10,11
–7, …, –4, 4, …, 7	3	000, … ,111
–15, …, –8, 8, …, 15	4	0000, … ,1111
–31, …, –16, 16, …, 31	5	00000, … ,11111
–63, …, –32, 32, …, 63	6	000000, … ,111111
–127, …, –64, 64, …,127	7	0000000, … ,1111111
–255, …, –128, 128, …, 255	8	00000000, … ,11111111
–511, …, –256, 256, …, 511	9	000000000, … ,111111111
–1023, …, –512, 512, …, 1023	10	0000000000, … ,1111111111

2.2 Huang et al.'s Method

In 2015, a new RDH method for JPEG images is proposed in [18], which has better embedding capability than previous methods. Sketch of this method is shown in **Fig. 2**, including the stages of entropy decoding, block selection, HS embedding, and entropy encoding. Key idea of this method is to use a selection strategy based HS embedding. Instead of using all non-zero AC coefficients, only a part of coefficient blocks are chosen to carry secret messages. With an evaluation algorithm, blocks with smoother contents are chosen in advance. The selection is required to meet the condition that the number of AC coefficients valued "1" and "–1" in these blocks is larger than the amount of message bits.

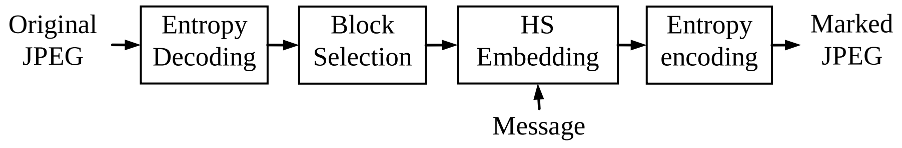


Fig. 2. Embedding process proposed by Huang's method

During data embedding, only the AC coefficients of "1" and "–1" in the selected blocks are modified to accommodate secret bits. If a bit "1" is to be embedded, these coefficients are modified to "2" or "–2", respectively. Otherwise, the coefficients are kept unchanged.

Meanwhile, the other AC coefficients with magnitudes greater than 1 are shifted by one towards the direction of their signs.

High embedding capacity can be achieved using the histogram shift based embedding. Since the way of hiding data into the selected blocks with more zero coefficients efficiently eliminates some invalid shifts, quality of the marked image is better than that in the previous works. One limitation is that the size of the JPEG file increases to a certain extent after data hiding. It would be better to reduce this artifact while preserving good embedding rate and image quality. To this end, we propose a new RDH approach for JPEG images based on zero-run-length selection.

3. Proposed Method

There are four stages in the proposed method, as shown in **Fig. 3**, including partial decoding, carrier selection, ordered embedding and partial encoding. After applying partial entropy decoding, the original JPEG file can be decoded into a series of ZRV pairs. To select proper carriers, we first choose the blocks with fewer ZRV pairs according to the message payload. Within the selected blocks, a number of ZRV pairs with smaller zero-run-lengths are chosen in advance, and an ordered embedding based on histogram shift is performed orderly on these pairs. Finally, all ZRV pairs are encoded to generate a marked JPEG file.

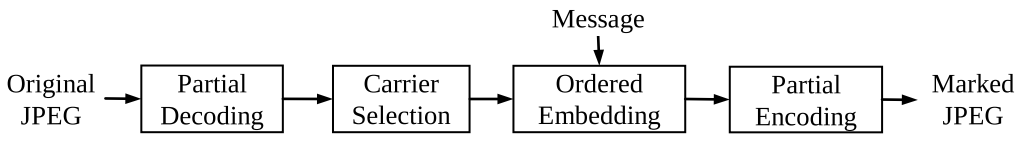


Fig. 3. Framework of the proposed method

3.1 Analyses of ZRV Pairs

To reduce the file size increment caused by data embedding in [13], the proposed method chooses proper coefficients as data carriers. When embedding data into a JPEG file using the histogram shift, all AC coefficients equaling to 0 are not modified, and none of the other coefficients is modified to 0. In other words, structure of all ZRV pairs remains unchanged after data hiding. As a result, we focus on the impacts of modifying ZRV pairs during data embedding. Let (R, V) be a ZRV pair, and (R, V') the corresponding pair carrying one secret bit. By analyzing the features of ZRV pairs in JPEG and the histogram shift based embedding, four remarks can be achieved.

Remark 1: *If the values V and V' belong to the same category, length of the encoded bits is unchanged; otherwise, the length changes.*

According to **Table 1**, if the values V and V' belong to the same category C, lengths of both Huffman code and VLI code do not change. When the value V having the magnitude 2^k-1 $(1\leq k\leq 9)$ is shifted to 2^k, length of code would be longer than the original. For example, when modifying a pair (0, 2) to (0, 3), the encoded bits change from “01/10” to “01/11”, in which the code length does not change. If we modify a pair (0, 3) to (0, 4), the codes change from “01/11” to “100/100”. The modification of non-zero AC coefficients is the main reason of JPEG file increment in HS based data embedding.

Remark 2: *The number of ZRV pairs belonging to each type in all DCT blocks of a JPEG image increases when R approaches zero.*

Statistical result corresponding to 50 images arbitrarily chosen from BOSSbase image dataset [20] is shown in **Fig. 4**. We compress these images by the toolbox of "Independent JPEG Group" (IJG) [21], using different quality factors (QF). Average amount of ZRV pairs of DCT coefficients in the compressed images are calculated. The result shows that there are more ZRV pairs when *R* gets smaller. For example, when QF=70, the average number of ZRV pairs corresponding to *R*=0 is 32882, while the number is almost zero when *R*=10. As a result, when embedding data using histogram shift based algorithm, we should use the ZRV pairs with smaller *R* to carry more bits.

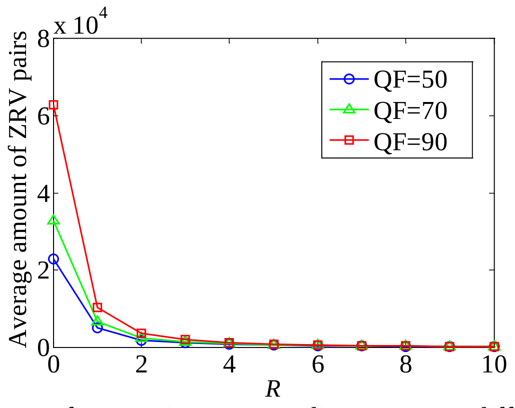


Fig. 4. Average amount of ZRV pairs corresponding to R using different quality factors

Remark 3: *Hiding data into ZRV pairs with a fixed R using histogram shift based algorithm causes the increment of JPEG file size. The increment is determined by R.*

We compress 50 arbitrarily chosen images using different quality factors, and embed a uniformly distributed binary secret message into ZRV pairs. During the embedding, (R, ±1) is used to carry secret bits. The increment rate, corresponding to different R, is referred to as the increment of file size per one bit embedded, which is defined as

$$I(R) = \frac{c_{R,1} \cdot n_{R,1} / 2 + \sum_{k \geq 2} c_{R,k} \cdot n_{R,k}}{n_{R,1}} \tag{1}$$

where $c_{R,k}$ stands for the increased code length when modifying ZRV pair (R, $\pm(2^k-1)$) to (R, $\pm 2^k$), and $n_{R,k}$ the number of (R, $\pm(2^k-1)$) pairs. For example, $c_{0,1}$ is the increased length caused by changing a ZRV pair from (0, ±1) to (0, ±2). According to Table K.5 in [19], $c_{0,1}$ is equal to 1. The parameter $n_{0,1}$ is the number of (0, ±1) pairs. During data hiding, $n_{0,1}/2$ ZRV pairs are modified when embedding "1", and the other $n_{0,1}/2$ ZRV pairs keep unchanged when embedding "0". **Table 2** shows the relationship between *R* and *I(R)* corresponding to different quality factors. The results indicate that the increment rates of JPEG file size increases when *R* is not larger than 10. Although the increment rate corresponding to 12 or 14 are smaller, there are actually few such ZRV pairs in the bitstream. This remark indicates that we should use ZRV pairs with smaller *R* to reduce the JPEG file size increment during histogram shift based embedding.

Table 2. Average increment rate per embedded bit

QF	*R*							
	0	2	4	6	8	10	12	14
50	1.0	2.1	2.6	3.0	3.5	4.0	3.5	0.5
70	1.1	2.2	2.6	3.0	3.5	4.0	3.5	0.5
90	1.3	2.2	2.6	3.0	3.5	4.0	3.5	0.5

***Remark 4**: Embedding data into the blocks with fewer ZRV pairs using histogram shift based algorithm can result in fewer shifts.*

Table 3 shows a group of statistical relationships between the payloads and the shifts. 50 images are arbitrarily chosen and compressed into JPEG. According to the number of ZRV pairs in each block, we evenly separate the blocks into two sets: $\mathbf{S}_1$ and $\mathbf{S}_2$. The set $\mathbf{S}_1$ contains the blocks with fewer ZRV pairs, and another the other blocks.The histogram shift based embedding is used to embed data into the ZRV pairs with a fixed R inside $\mathbf{S}_1$ and $\mathbf{S}_2$, respectively. For a 512*512 image, when R=0, 4096 8×8 blocks are sorted according to the number of (0, V) pairs. The set $\mathbf{S}_1$ contains the 2048 blocks with fewer (0, V) pairs, and $\mathbf{S}_2$ the other 2048 blocks. We calculate the average embedding payload N_p (bits) and the average number of ZRV shifts N_s. In **Table 3**, different quality factors are used to compress the original image, and R=0. Ratio N_s/N_p is smaller when embedding data into the blocks with fewer ZRV pairs. As a result, during the histogram shift based embedding, we should hide data in the blocks with less ZRV pairs to reduce the amount of shifts, meaning a smaller file size increment. Different parameter R has also been used, and the same conclusion is achieved.

Table 3. Payloads and ZRV shifts

QF	$\mathbf{S}_1$			$\mathbf{S}_2$		
	N_p	N_s	N_s/N_p	N_p	N_s	N_s/N_p
50	2210	1867	0.84	8843	9968	1.13
70	3420	3468	1.01	10734	15261	1.42
90	6497	9923	1.53	15541	30957	1.99

3.2 Carrier Selection and Ordered Embedding

According to these analyses, we choose appropriate ZRV pairs as carriers. After partially decoding the JPEG file, we calculate the number of ZRV pairs in each block. These numbers are denoted as A_i, where i=1,2,…,M, and M is the number of 8×8 blocks in the image. We also calculate the number of ZRV pairs with the values of (R, 1) or (R, −1) in each block, in which 0≤R≤15. Denote these numbers as B_i, where i=1,2,…,M.

With a threshold T (0≤T≤63), we select the blocks satisfying A_i≤T. Denote the indices of selected blocks as {s(1),s(2),⋯,s(L)}, in which L stands for the number of selected blocks and 1≤L≤M. Assuming there are P secret bits to be embedded, we use a threshold T such that

$$P \le \sum_{k=1}^{L} B_{s(k)} \tag{2}$$

The parameter T is used to choose blocks from the image. If the number of ZRV pairs in a block is smaller than T, this block will be used to embed message. Otherwise, this block will not be used. According to ***Remark 4*** and (2), we should choose T as small as possible. As a result, this parameter is identified by the embedding payload.

For all the selected blocks, we construct a map $\mathbf{W}(i, R)$ to indicate the number of ZRV pairs having the zero-run-length equaling to R in the $s(i)$-th block, where $i=1,2,\ldots,L$ and $R=0,1,\ldots,15$. For example, $\mathbf{W}(4, 2)=3$ means there are three $(2, V)$ ZRV pairs in the fourth selected block, including all possible values of V.

With the map, we use an ordered data embedding based on histogram shift. A flowchart of the embedding procedure is shown in **Fig. 5**. The embedding procedure includes several rounds. According to ***Remark 2*** to ***Remark 4***, selected blocks with fewer pairs of $R=0$ will used for embedding in advance. In the first round, we find the blocks with $\mathbf{W}(i, 0)=1$ from the selected blocks, meaning that each of these blocks contains only one $(0, V)$ ZRV pair. With the histogram shift based embedding algorithm, secret bits are hidden into these blocks. In the second round, we find the blocks with $\mathbf{W}(i, 0)=2$ from the selected blocks to carry more secret bits. After all pairs of $R=0$ in selected blocks are used in m_0 rounds, we use the pairs of $R=1$. In this way, m_a rounds of embedding are implemented until all messages are accommodated in the selected ZRV pairs, where

$$m_a = \sum_{j=0}^{15} m_j \tag{3}$$

and m_j is the maximal number of ZRV pairs with $R=j$ $(0 \le j \le 15)$ existing in any selected block.

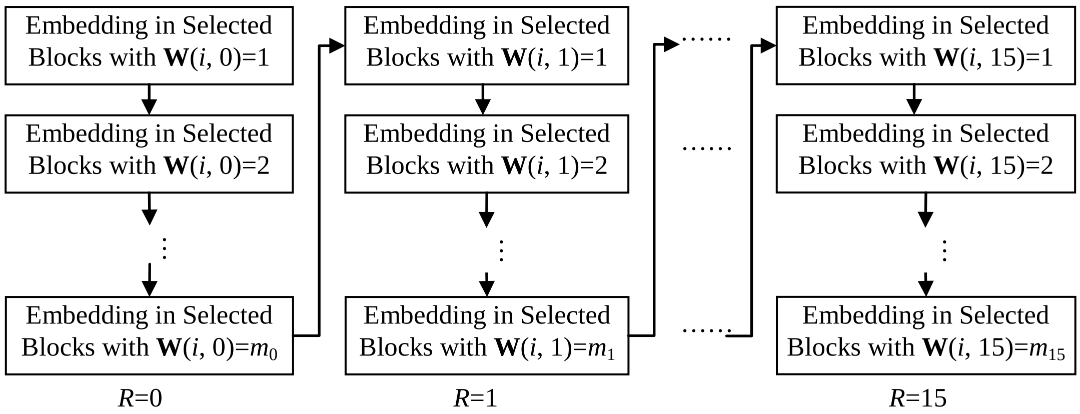

Embedding in Selected Blocks with W(i, 0)=1
Embedding in Selected Blocks with W(i, 0)=2
Embedding in Selected Blocks with W(i, 0)=m0
Embedding in Selected Blocks with W(i, 1)=1
Embedding in Selected Blocks with W(i, 1)=2
Embedding in Selected Blocks with W(i, 1)=m1
Embedding in Selected Blocks with W(i, 15)=1
Embedding in Selected Blocks with W(i, 15)=2
Embedding in Selected Blocks with W(i, 15)=m15
R=0
R=1
R=15

Fig. 5. Flowchart of ordered embedding

In the j-th round $(0 \le j \le m_a)$, assuming there are u $(0 \le u \le L)$ blocks and $R=r$, from these blocks we find all ZRV pairs whose zero-run-lengths are r. Let these pairs be (r, v_1), (r, v_2), …, (r, v_w), where w is the number of useful pairs in these blocks. If there are t pairs having values of $(r, 1)$ or $(r, -1)$, t secret bits can be embedded into these pairs by histogram shift using

$$v_i' = \begin{cases} v_i - 1 & \text{if } v_i < -1 \\ v_i - b_k & \text{if } v_i = -1 \\ v_i + b_k & \text{if } v_i = 1 \\ v_i + 1 & \text{if } v_i > 1 \end{cases} \tag{4}$$

where b_k stands for the secret bit, $0 \le k \le t$ and $0 \le i \le w$.

After all message bits are embedded during m_a rounds, all ZRV pairs are entropy encoded to generate a new JPEG file. The threshold T and the amount of secret bits are encrypted and written in the reserved segment inside the JPEG header. This way, the marked JPEG file is constructed.

3.3 Data Extraction and Image Recovery

On the recipient end, a receiver extracts the encrypted parameters from the reserved segment of JPEG header. After decryption, the parameter T and the amount of secret bits are obtained. To extract the secret message, the receiver first decodes the marked JPEG file to ZRV pairs, and calculates the number of ZRV pairs in each block. With the threshold T, L blocks whose pairs are not more than T are selected.

Using the same algorithm described in *Section B*, the receiver can reconstruct the map $\mathbf{W}(i, R)$ by analyzing all pairs in the selected blocks, where i=1,2,…,L and R=0,1,…,15. Since the data hiding procedure did not change the zero-run-length R, reconstructed map is the same as the one constructed from the original image. Then, the receiver uses m_a rounds to do the extraction. Orders of the rounds are the same as the flowchart depicted in **Fig. 5**, from $\mathbf{W}(i, 0)=1$ to $\mathbf{W}(i, 15)=m_{15}$. During the j-th round ($0 \le j \le m_a$), we find all ZRV pairs (r, v_1), (r, v_2), …, (r, v_w) whose $R=r$, and extracts t secret bits from these pairs by

$$b_k = \begin{cases} 0 & \text{if } |v_i| = 1 \\ 1 & \text{if } |v_i| = 2 \end{cases} \quad (5)$$

where $0 \le k \le t$ and $0 \le i \le w$. Meanwhile, the original ZRV pairs used in the j-th round are recovered by

$$v_i = \begin{cases} v_i'+1 & \text{if } v_i' < -2 \\ -1 & \text{if } -2 \le v_i' \le -1 \\ 1 & \text{if } 1 \le v_i' \le 2 \\ v_i'-1 & \text{if } v_i' > 2 \end{cases} \quad (6)$$

Through m_a rounds of extraction and recovery, the receiver can exactly extract the hidden message and recover the original ZRV pairs. These pairs are further entropy encoded to construct the original JPEG file.

4. Experimental Results

To evaluate the performance of the new proposed method, we use grayscale images sized 512×512 arbitrarily chosen from Bossbase [20]. These images are compressed by JPEG using different quality factors. The features of embedding capacity, image quality and file size are compared with the results of state-of-the-art work in [18].

Fig. 6 shows a group of experimental results, in which (a)-(c) are the images "Lena", "Peppers" and "Baboon" compressed by JPEG using a quality factor 70. We hide 26442 bits, 21919 bits and 42855 bits into these images using the thresholds T as 36, 36, and 42, respectively. **Fig. 6**(d)~(e) shows the marked JPEG images, PSNR of which are 39.4dB, 40.6dB, and 33.2dB, respectively, indicating that the marked JPEG images preserves good visual quality. From the marked JPEG images, secret bits are extracted without any error, and the original JPEG images are losslessly recovered.

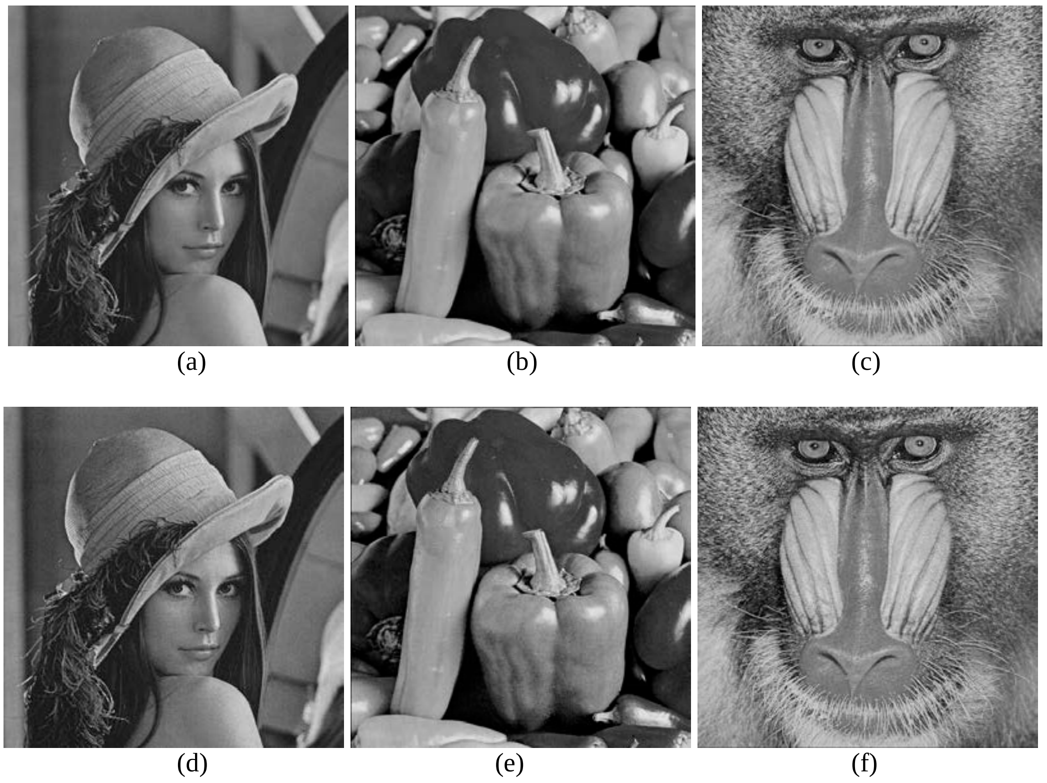


Fig. 6. A group of results, (a)~(c): the original JPEG images "Lena", "Peppers" and "Baboon"; (d)~(f): the marked JPEG images

For a fair comparison, both [18] and the present method use the coefficients with values ±1 to carry the secret message. We arbitrarily choose 50 grayscale images from Bossbase and compress these images with JPEG using different quality factors. Threshold T used in the experiments is adaptive to the embedding payload. The average maximal embedding capacity is 18999, 24015, and 36455 bits corresponding to the quality factors 50, 70 and 90, respectively. The embedding capacities are the same as those in [18], as both methods use the same ZRV pairs. Meanwhile, the average PSNR values of marked JPEG images corresponding to different payloads and quality factors are listed in **Table 4**. Results of the proposed method are also identical to [18].

Table 4. Average PSNR (dB) corresponding to different payloads using different quality factors

QF	**Methods**	**Payload** (Bits)								
		1000	5000	10000	15000	18000	20000	24000	30000	36000
50	[18]	54.4	45.8	41.5	38.7	37.6	-	-	-	-
	Proposed	54.4	45.8	41.5	38.7	37.6	-	-	-	-
70	[18]	58.2	49.4	45.0	42.0	40.4	39.4	38.1	-	-
	Proposed	58.2	49.4	45.0	42.0	40.4	39.4	38.1	-	-
90	[18]	64.5	56.2	51.8	48.7	47.2	46.3	44.5	42.2	40.6
	Proposed	64.5	56.2	51.8	48.7	47.2	46.3	44.5	42.2	40.6

In **Fig. 7**, the average increments of file size are evaluated. We compress the 50 images using different quality factors and embed different amount of secret bits. **Fig. 7**(a)~(c) show the results corresponding to quality factors of 50, 70 and 90, respectively. Generally, more payloads result in more increments of JPEG file size. Compared with [18], the proposed method has smaller increments of file size.

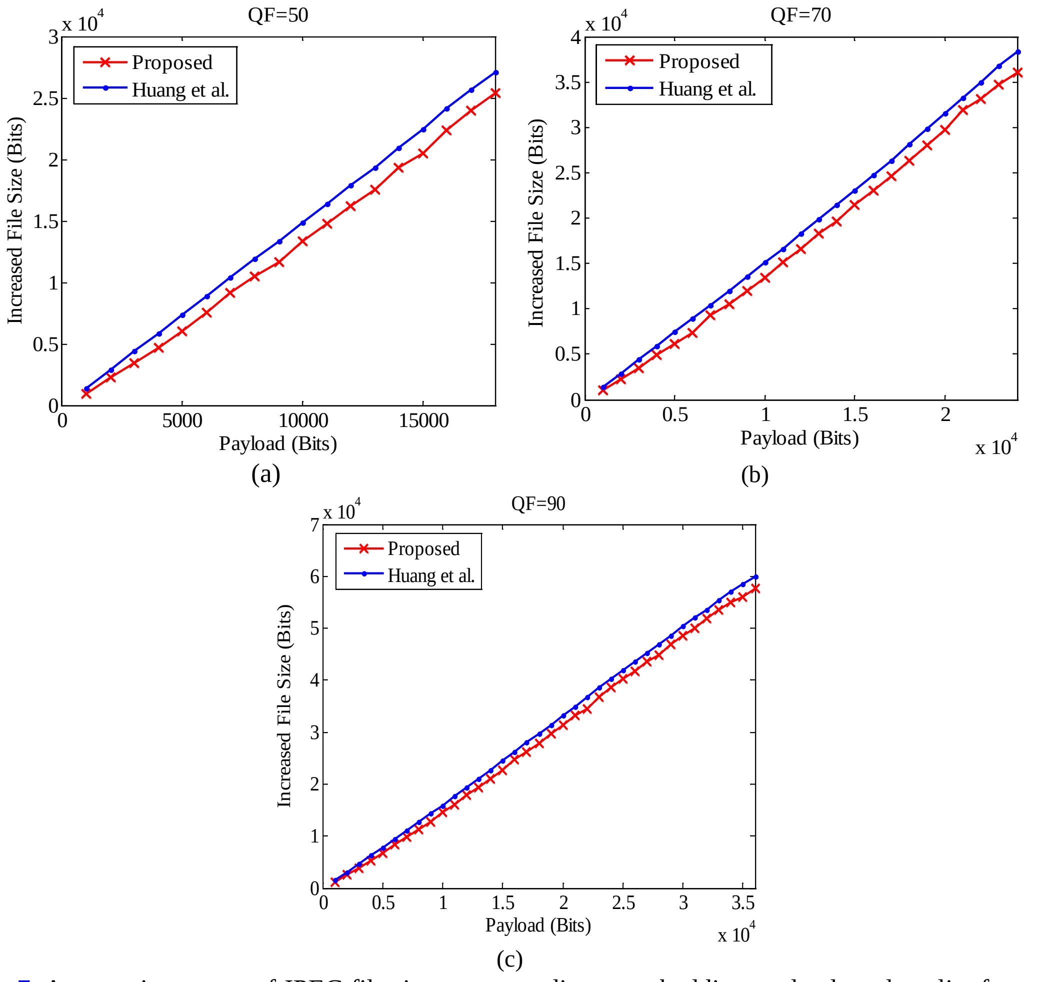


Fig. 7. Average increment of JPEG file size corresponding to embedding payloads and quality factors

Fig. 8 shows the average increments of file size corresponding to the quality of marked JPEG images. When the quality of marked JPEG image is fixed, the proposed method has better capability of preserving JPEG file size. This achievement in the proposed method is resulted by the ordered embedding. Instead of embedding bits in one histogram, we embed secret bits in many rounds, *i.e.*, shift coefficients in many histograms. This procedure successfully avoids many unnecessary shifts. As a result, file size increment is reduced.

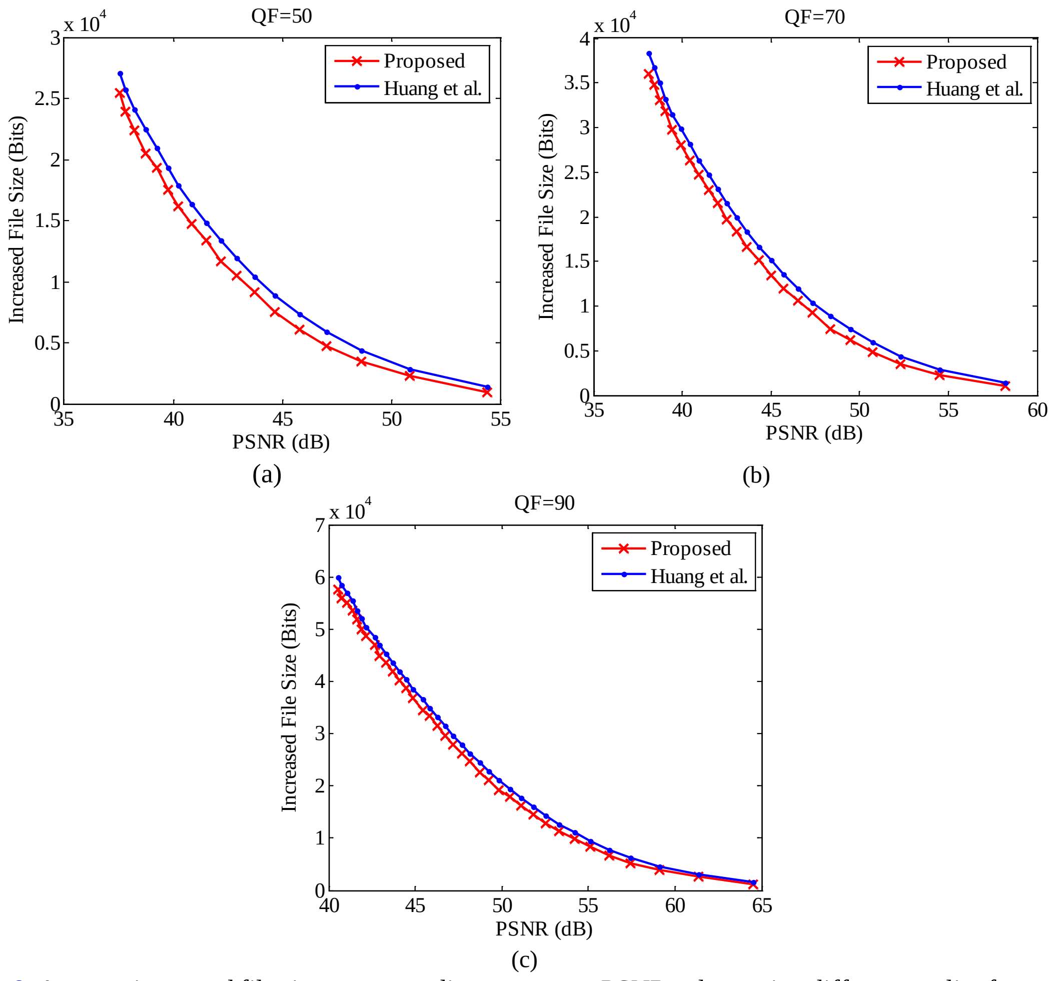


Fig. 8. Average increased file size corresponding to average PSNR values using different quality factors

5. Conclusion

Generally, reversible data hiding in JPEG files by modifying image content always result in the artifact of file size increment. This paper provides an ordered embedding based method to reduce this artifact. We analyze the features of JPEG embedding and achieve a number of remarks. According to these remarks, a number of blocks are chosen to accommodate the secret message, and the data embedding procedure is implemented in several rounds using histogram shift based algorithm. With the proposed method, large embedding capacity can be achieved and the marked image preserves good quality. Benefit from the ordered embedding, the proposed method efficiently reduces the increment of JPEG file size.

References

[1] Y. Qiu, Z. Qian, and L. Yu, “Adaptive reversible data hiding by extending the generalized integer transformation,” *IEEE Signal Processing Letters*, vol. 23, no.1, pp. 81-85, 2016. Article (CrossRef Link).

[2] Y. Hu, B. Jeon, "Reversible visible watermarking and lossless recovery of original images," *IEEE Trans. Circuits and Systems for Video Technology*, vol.16, no.11, pp. 1423-1429, 2006. Article (CrossRef Link).

[3] G. Coatrieux, Pan W, N. C. Boulahia, F. Cuppens, and C. Roux, "Reversible watermarking based on invariant image classification and dynamic histogram shifting," *IEEE Trans. Information Forensics and Security*, vol.8, no.1, pp.111-120, 2013. Article (CrossRef Link).

[4] Z. Qian, X. Zhang, and S. Wang, "Reversible data hiding in encrypted JPEG bitstream. IEEE Trans," *Multimedia*, vol.16, no.5, pp.1486-1491, 2014. Article (CrossRef Link).

[5] H. Wu, J. L. Dugelay, and Y. Shi, "Reversible image data hiding with contrast enhancement," *IEEE Signal Processing Letters*, vol. 22, no.1, pp. 81-85, 2015. Article (CrossRef Link).

[6] W. Zhang, X. Hu, X. Li, and N. Yu, "Recursive histogram modification: establishing equivalency between reversible data hiding and lossless data compression," *IEEE Trans. Image Processing*, vol 22, no.7, pp. 2775-2785, 2013. Article (CrossRef Link).

[7] W. Tai, C. M. Yeh, and C. C. Chang, "Reversible data hiding based on histogram modification of pixel differences," *IEEE Trans. Circuits and Systems for Video Technology*, vol.19, no.6, pp. 906-910, 2009. Article (CrossRef Link).

[8] C. Qin, C. C. Chang, Y. Huang, and L. Liao, "An inpainting-assisted reversible steganographic scheme using a histogram shifting mechanism," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 23, no. 7, pp. 1109-1118, 2013. Article (CrossRef Link).

[9] X. Li, W. Zhang, X. Gui, and B. Yang, "A novel reversible data hiding scheme based on two-dimensional difference-histogram modification," *IEEE Trans. Information Forensics and Security*, vol.8, no.7, pp. 1091-1100, 2013. Article (CrossRef Link).

[10] J. Fridrich, M. Goljan, Q. Chen, and V. Pathak, "Lossless data embedding with file size preservation," in *Proc. of the SPIE, Int. Society for Optics and Photonics, Electronic Imaging*, San Jose, America, Jun 22, pp. 354-365, 2004. Article (CrossRef Link).

[11] Z. Qian, X. Zhang, "Lossless data hiding in JPEG bitstream," *Journal of Systems and Software*, vol. 85, no.2, pp. 309–313, 2012. Article (CrossRef Link).

[12] K. Wang, Z. -M. Lu, and Y. -J. Hu, "A high capacity lossless data hiding scheme for JPEG images," *Journal of Systems and Software*, vol. 86, no.7, pp. 1965–1975, 2013. Article (CrossRef Link).

[13] G. Xuan, Y. Q. Shi, Z. Ni, P. Chai, X. Cui, and X. Tong, "Reversible data hiding for JPEG images based on histogram pairs," in *Proc. of Int. Conf. on Image Analysis and Recognition*, Montreal, Canada, Aug. 22–24, pp. 715–727, 2007. Article (CrossRef Link).

[14] H. Sakai, M. Kuribayashi, and M. Morii, "Adaptive reversible data hiding for JPEG images," in *Proc. of Int. Symp. on Inf. Theory and its Applications*, Auckland, New Zealand, Dec. 7-10, pp. 1–6, 2008. Article (CrossRef Link).

[15] Q. Li, Y. Wu, and F. Bao, "A reversible data hiding scheme for JPEG images," in *Proc. of 11th Pa cific Rim Conf. on Advances in multimedia information processing*, Berlin, Germany, Sep. 21-24, pp. 653–664, 2010. Article (CrossRef Link).

[16] A. Nikolaidis, "Reversible data hiding in JPEG images utilizing zero quantized coefficients," *IET Image Processing*, vol.9, no.7, pp. 560-568, 2015. Article (CrossRef Link).

[17] A. Nikolaidis, "Low overhead reversible data hiding for color JPEG image," *Multimedia Tools and Applications*, vol.75, no.4, pp.1869-1881, 2016. Article (CrossRef Link).

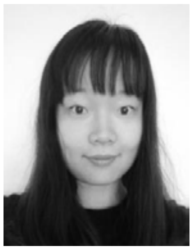
[18] F. Huang, X. Qu, H. J. Kim, and J. Huang, "Reversible Data Hiding in JPEG Images, *IEEE Trans*," *Circuits and Systems for Video Technology*, 2015. Article (CrossRef Link).

[19] Int. Telecommunication Union, CCITT Recommendation T.81, Information Technology-Digital Compression and Coding of Continuous-tone Still Images-Requirements and Guidelines 1992.

[20] Bossbase Database, [Online], Available: http://dde.binghamton.edu/download/, 2013.

[21] Independent JPEG Group, [Online], http://www.ijg.org/, 2016

Zhenxing Qian received both the B.S. and the Ph.D. degrees from University of Science and Technology of China (USTC), in 2003 and 2007, respectively. He is currently a professor with the School of Communication and Information Engineering, Shanghai University, China. His research interests include data hiding and multimedia security.



Shu Dai received the B.S. degree from the School of Communication and Information Engineering, Shanghai University, China, in 2014. She is currently pursuing the MS degree in Shanghai University, Shanghai, China. Her research interests include data hiding and image processing.

Boyang Chen received the B.S. degree from University of Science and Technology of China (USTC) in 2003 and the Ph.D. degree from Chinese Academy of Science in 2008. He is currently a faculty of National Satellite Meteorological Center, Beijing, China. His research interests include image processing and image assessment