Wuhan University
Journal of Natural Sciences

2013, Vol.18 No.2, 126-132

Article ID 1007-1202(2013)02-0126-07
DOI 10.1007/s11859-013-0904-1

Reversible Watermarking in JPEG Images Based on Modified RZL Codes and Histogram Shift

□ CHEN Biao[1], ZHANG Weiming[1,2], YU Nenghai[1†]

1. Department of Electronic Engineering and Information Science, University of Science and Technology of China, Hefei 230027, Anhui, China;

2. Department of Information Research, Zhengzhou Information Science and Technology Institute, Zhengzhou 450002, Henan, China

Abstract: Reversible watermarking has extensive applications in fields such as medical data management and forensic enforcement. In this article, we propose a modified reverse zero-run length (RZL) coding method for reversible watermarking, which introduces fewer modifications to cover than previous ones under the same embedding rate. By combining the coding method with histogram shift and quantization table modification strategy, we propose a novel reversible data hiding algorithm for JPEG images. Compared with previous art, when embedding the same payload, our method has an improvement of at least 2 dB in stego image's quality, which proves the effectiveness and advantage of our algorithm.

Key words: reversible watermarking; JPEG images; image authentication; payload; PSNR

CLC number: TP 309.7

Received date: 2012-04-05

Foundation item: Supported by the National Natural Science Foundation of China (61170234 and 60803155), the National Science and Technology Major Project of China (2010ZX03004-003), and the High-Tech Research and Development Program of China (863 program) (2009AA012201).

Biography: CHEN Biao, male, Master candidate, research direction: image and video processing, information hiding. E-mail: chb893@mail.ustc.edu.cn

† To whom correspondence should be addressed. E-mail: ynh@ustc.edu.cn

0 Introduction

As a technique that embeds secret information into a cover media, digital watermarking has been used in applications such as media notation, copyright protection, and integrity authentication[1]. For most watermarking methods, the embedding process will introduce permanent distortion to the cover and the original cover can never be restored from the watermarked media. However, in some applications, such as medical imagery, military imagery, and law enforcement, no degradation of the original cover is allowed. In these situations, we need exactly to restore the original cover media after secret message is extracted. The technique for this requirement is called reversible data hiding (RDH) or reversible watermarking.

The main application of reversible watermarking is image integrity protection. In a classic framework, the hash of an image's important region is reversibly embedded into unimportant region. In fact, reversible authentication watermarks have been incorporated as an integrity protection mechanism into the ISSE military guard[2]. Integrity protection watermarks can also be embedded inside the imaging hardware of cameras for forensic personnel[2]. Since firstly introduced in a Kodak patent[3], many RDH algorithms have been proposed for grayscale images. These algorithms roughly fall into three categories. The first algorithms follow the idea of compression-embedding framework, which was first introduced by Fridrich *et al*[4]: in these algorithms, a two-value feature is calculated for each group of pixels and the features form a binary sequence. The sequence is compressible and the message can be embedded in the extra

space left by compression. An extension of this work was presented in Celik *et al*'s work[5]. The second category is based on difference expansion[6], in which the differences of pixel are expanded, for example, multiplied by 2; thus, the least significant bits of the differences can be used for embedding message. These algorithms can usually achieve a considerable high embedding capacity[7,8]. The last and most promising RDH algorithms are based on histogram modification[9]. The histogram of one special feature for natural image is quite uneven, which implies that the histogram can be compressed for embedding data. For instance, some space can be saved for watermarks by shifting the bins of histogram. These algorithms following the idea of histogram shift can often achieve a very high PSNR with an acceptable embedding payload[10,11].

JPEG is the most popular image format currently, so RDH algorithms designed for JPEG images seem more useful and necessary. Some recent applications of reversible authentication watermarks for JPEG images can be found in Zhang *et al*[12]. However, there are not that many RDH algorithms for JPEG images. Fridrich and Goljan[2] first made an attempt. They selected some quantized DCT coefficients equal to 0 and 1, the 0's and 1's then form a compressible binary sequence, and the spare space after lossless compression can be used to carry message bits. Chang *et al* [13] have introduced a RDH algorithm that searches special pattern that has two distinct states in each block. Message can be embedded via representing the two states with 0 and 1, respectively. Li *et al* [14] and Xuan *et al* [15] adopted the idea of histogram shift. Li *et al*'s method[14] combines simple histogram shift with quantization table modification and has shown a satisfactory performance. Xuan *et al*'s method[15] took several factors into consideration and tried to find a best PSNR for any given payload.

Reversible watermarking appreciates large payload and low distortion between cover and stego image. There's a coding method named reverse zerorun length (RZL)[16] that can help this. However, RZL method is a binary coding method for all-zero cover. We can modify it for our application.

In this article, we propose a reversible watermarking algorithm by combining a modified RZL coding method, histogram shift, and quantization table modification. The rest of this article is as follows. In Section 1, we present our modified RZL method for ternary cover and make some theoretical analysis. In Section 2, the entire process of RDH algorithm for JPEG image is presented. The experimental results are demonstrated in Section 3, and the article is concluded with a discussion in Section 4.

1 Modified RZL Coding

1.1 RZL Coding

Li *et al*[14] proposed a simple but efficient reversible data hiding method for JPEG images, in which the quantized DCT coefficients at some positions of an 8×8 block are used to embed data. First, the histogram of the selected quantized DCT coefficients is shifted in following manner. All bins with entry larger than 0 are shifted rightward, whereas all bins with entry smaller than 0 are shifted leftward, and then the message is embedded into the zero coefficients. The empty locations where entry is equal to 1 or -1 are used to embed message. If the message bit is 0, then the coefficient holds 0; if message bit is 1, then the coefficient is changed to be 1 or -1 randomly. The recipient can extract the message by decoding 0 as bit "0" and ±1 as bits "1". The coefficients can be reconstructed by modifying ±1 to 0 and reversely shifting all non-zero coefficients.

In the above method, an all-zero cover is constructed after shifting non-zero coefficients. On this cover, every element, zero, can be modified toward two directions, 1 or -1. To reduce the embedding distortion, the key problem is to reduce the number of modified zeros for a message with given length. One coding method for this purpose has been proposed by Wong *et al*[16], which is called RZL coding method.

RZL is designed for an all-zero cover with only one modification direction, i.e., 0 can only be changed to 1. In RZL coding, the binary message sequence is first divided into disjoint segments of k bits. To embed one segment, covert the corresponding vector of k bits to a positive integer, denoted by d, and then skip d zeros in the cover, and flip the (d+1)th 0. To extract the message, the recipient only needs to convert the distance between 1's in the stego to binary vectors of k bits.

We use embedding rate versus modification rate to evaluate such kind of coding methods. The embedding rate α is defined as the average number of message bits embedded into per cover element, for a binary cover sequence, $\alpha \leqslant 1$; the modification rate c is defined as the probability of one cover element being modified. If we assume the message is encrypted sequence that is random, then it is easy to calculate and confirm that RZL method has embedding rate $\alpha = k/(2^{k-1}+0.5)$ and modification rate $c = 1/(2^{k-1}+0.5)$ [16].

We compare RZL with the ordinary representation

scheme (ORS; which means to directly replace the cover element by 0 or 1 according to message bit). RZL achieves lower modification rate than ORS for most range of embedding rate but performs poorly for larger embedding rate ($\alpha > 0.7$). On the contrary, although RZL can be directly applied to Li *et al*'s scheme[14], the embedding potential cannot be exploited sufficiently, because there are two modification directions, +1 and -1, in Li *et al*'s scheme.

1.2 Modified RZL Coding for Binary Embedding

As mentioned before, RZL performs poorly for large embedding rate, we can use some flag bits to improve its performance. First, we consider the case of binary embedding, in which the cover element, "0", can only be changed to "1". In RZL coding, the integer $l = \sum_{i=1}^{k} a_{i,1} 2^{k-i}$ determines the number of cover elements cost by embedding the binary vector $(a_1 \cdots a_k)$. Therefore, small l implies large embedding rate. To decrease the expectation value of l, we use a flag bit f to control the most significant bits (MSBs) of several vectors, and make the majority of them to be "0". Based on this idea, the RZL coding is modified as follows.

Encoding:

ⓐ Divide the message sequence to segments of k bits, and group every m segments into one group, where m is an odd number larger than 1;

ⓑ For each group $\{(a_{1,1} \cdots a_{1,k}) \cdots (a_{m,1} \cdots a_{m,k})\}$, set $s = \sum_{i=1}^{m} a_{i,1}$; if s is larger than $m/2$, then flip the $a_{i,1}$ to be $1 - a_{i,1}$, for $i = 1, \cdots, m$, and set the flag bit $f = 1$; if s is smaller than $m/2$, then $a_{i,1}$ are not changed, and $f = 0$;

ⓒ Embed the group of bits (maybe have been changed) into cover segment by segment via RZL embedding;

ⓓ Finally, embed the flag bit f into cover using ORS.

Decoding:

ⓐ Continuously extract m segments of k bits from marked cover using RZL message extraction method[16], and set these m segments into a group $\{(a'_{1,1} \cdots a'_{1,k}) \cdots (a'_{m,1} \cdots a'_{m,k})\}$;

ⓑ Extract the corresponding flag bit f from the cover; if $f = 1$, then flip $a'_{i,1}$ to be $1 - a'_{i,1}$, for $i = 1, \cdots, m$; else, if $f = 0$, do nothing;

ⓒ Concatenate every group of $k \cdot m$ bits to form the final extracted message sequence.

Now we make a theoretical analysis about the embedding rate and modification rate of the above method. For embedding one segment of message $(a_{i,1} \cdots a_{i,k})$, if $a_{i,1} = 1$, then the expectation of occupied locations of cover is $n_1 = 2^{k-1} + 2^{k-2} + 0.5$; otherwise, if $a_{i,1} = 0$, then the expectation of occupied locations of cover is $n_0 = 2^{k-2} + 0.5$.

For $H = (a_{1,1} \cdots a_{m,1})$, the number of 1's x in H is binomial distributed: $P(x = i) = C_m^i \cdot 2^{-m}, i = 0, \cdots, m$; then, in modified RZL (MRZL) coding, if $x > m/2$, then the header bits are flipped; if $x < m/2$, then $H' = H$. Therefore, the distribution of number of 1's x' is $P(x'=i) = P(x=i) + P(x=m-i) = 2 \cdot C_m^i \cdot 2^{-m}$, $i = 0, \cdots, (m-1)/2$. If $x' = i$, then to embed $\{(a'_{1,1} \cdots a'_{1,k}), \cdots, (a'_{m,1} \cdots a'_{m,k})\}$ in RZL embedding, the expectation of occupied locations of cover is $N_i = i \cdot (0.75 \cdot 2^k + 0.5) + (m-i) \cdot (0.25 \cdot 2^k + 0.5)$. Therefore, the expectation of occupied locations of cover for MRZL coding is

$$n = \sum_{i=0}^{(m-1)/2} \{P(x' = i) \cdot \frac{1}{m} \cdot N_i\} + \frac{1}{m}.$$

It can be simplified as

$$n = \frac{1}{m} \cdot 2k - m \cdot \sum_{i=0}^{(m-1)/2} (i \cdot C_m^i) + 2^{k-2} + \frac{1}{m} + \frac{1}{2}, \quad k > 1 \tag{1}$$

For each segment, the expectation of the modification locations is

$$R = 1 + \frac{1}{2 \cdot m}, \tag{2}$$

where 1 is from RZL coding, $\frac{1}{2 \cdot m}$ is for the flag bit f.

Therefore, we obtain the embedding rate α and modification rate c as

$$\alpha = \frac{k}{n}, \quad c = \frac{R}{n}, \qquad k > 1 \tag{3}$$

In practice, we suggest setting the number of segments $m = 3$.

1.3 MRZL Coding for ±1 Embedding

In this section, we further modify the coding method in Section 1.2 with ±1 embedding, which means the cover element "0" can be changed to +1 or -1. In fact, a ternary coding construction hides behind the ±1 embedding. Ordinarily, we can convert the binary message sequence to a ternary sequence and then embed the ternary sequence into the cover with a ternary code. However, this method requires extra computational complexity for the conversion between binary and ternary sequences. Instead of the

common method, we propose the following two-layer embedding strategy that is much simpler and faster.

We divide the original message sequence into two parts: basic sequence and extra sequence. The basic sequence is encoded to be an intermediate sequence by using the coding method described in Section 1.2. Then, the intermediate sequence is embedded into the cover. When the embedded bit of intermediate sequence is 0, we do not change the cover; if the embedded bit is 1, we embed a message bit of the extra sequence in following manner. If the bit picked up from the extra sequence is 0, change the cover element to 1; otherwise, change cover element to -1.

In the above two-layer scheme, whenever a modification in the first layer appears, an extra bit can be embedded without introducing any more modification. Therefore, comparing the coding method with binary embedding, we can increase embedding rate while keeping the modification rate the same, and the increment of embedding rate is just equal to the modification rate. Based on Eqs. (1) and (2), we get the embedding rate and modification rate of the two layer scheme as

$$\alpha = \frac{k+R}{n}, c = \frac{R}{n} \tag{4}$$

As shown in Fig. 1, the proposed method significantly outperforms ORS and RZL. As this is in fact a ternary coding construction, the largest theoretical embedding rate increases from 1 to $\log_2 3$.

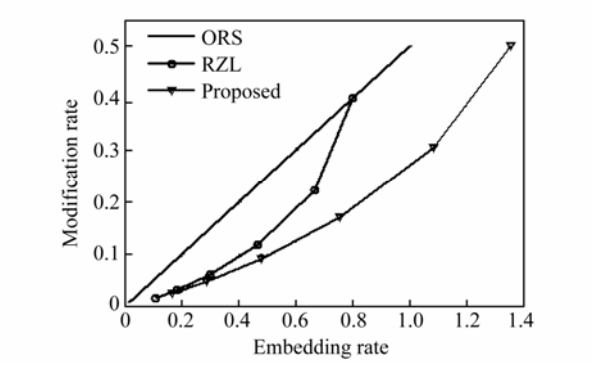

Fig. 1 Comparison of ORS, RZL, and proposed coding method

Next, we provide a simple example to describe the encoding and decoding process of proposed coding method.

Example 1

Encoding:

The cover is an all-zero sequence: $[0\,0\cdots0\cdots]$. Assume the message sequence is [1 0 0 1 1 1 0 1 1 0 ⋯], and set k=2 and m=3. First, we pick up m segments of k bits to form a group: $\{(1\,0)(0\,1)(1\,1)\}$, the MSBs of each segment are (1 0 1), which include two 1's, larger than $m/2$=1.5), so the MSBs are flipped to be (0 1 0) and set $f=1$; then, the modified segments, {(0 0) (1 1) (0 1)}, are embedded into cover by RZL coding segment by segment: (1) (0 0 0 1) (0 1), followed by the flag $f=1$; thus, the intermediate sequence is [1 0 0 0 1 0 1 1], in which totally four 1's exist. Therefore, we continue to read four extra bits from original message sequence: (0 1 1 0); if extra bit is 0, then hold the 1 unchanged; else, if extra bit is 1, the corresponding 1 is changed to −1; so, after this process, the final ternary sequence becomes [1 0 0 0 −1 0 −1 1]. The following message sequence is encoded in the same way.

Decoding:

The received ternary sequence is [1 0 0 0 −1 0 −1 1]. We first wipe out their signs: [1 0 0 0 1 0 1 1]; read the sequence until three "1"s appear: [1 0 0 0 1 0 1], and decode it with RZL: (0 0) (1 1) (0 1); read the next flag bit $f=1$, which indicates that the MSBs of the decoded vector should be flipped: (1 0) (0 1) (1 1); finally, as the four non-zeros in this round are (1 −1 −1 1), then we additionally extract four bits: (0 1 1 0). Thus, the decoded binary sequence is [1 0 0 1 1 1 0 1 1 0].

2 Proposed RDH Algorithm for JPEG Image

2.1 Proposed RDH Algorithm

In this section, we improve Li *et al*'s method[14] by using the coding method proposed in Section 1.3.

In his original method, a quantization table modification technique is used to restrain degradation of the marked image.

Suppose original quantization step is q and quantized DCT coefficient is v. Then, via histogram shift, v is changed to v'=v+1, and the error of dequantized DCT coefficient is $|q \cdot v - q \cdot v'| = q$. However, if we modify q to be $p = s \cdot q$, where $0 < s < 1$ is the scaling factor. Meanwhile, set $v' = \left\lfloor \frac{q \cdot v}{p} \right\rfloor = \left\lfloor \frac{v}{s} \right\rfloor$, where $[x]$ is rounding function. Then, the error of dequantized DCT coefficient is $|q \cdot v - p \cdot v'|$. Denote $[x] = x + \theta$, $-0.5 < \theta \leqslant 0.5$, we have $|q \cdot v - p \cdot v'| = |q \cdot v - s \cdot q \cdot (\frac{v}{s} + \theta)| = |s \cdot q \cdot \theta| < q$; thus, the degradation of image quality is restrained.

By incorporating this modification technique and the coding method in Section 1.3, we proposed the following RDH algorithm for JPEG images.

Algorithm 1

Data embedding process

ⓐ Convert the original binary message sequence to ternary sequence via proposed coding method in Section 1.3.

ⓑ Select some entries in quantization table, and the corresponding quantized DCT coefficients will be used to embed data. For these entries, modify quantization step from q to $p = s \cdot q$; meanwhile, calculate A and B. Herein, A is the smallest positive integer that satisfies $A > 1$ and $[A \cdot p] = [(A+1) \cdot p] = B$.

ⓒ For each 8×8 quantized DCT coefficient block of luminance component from a JPEG image after Huffman decoding and run-level decoding, pick up coefficients that will be used to embed message.

ⓓ For each selected quantized DCT coefficient v, do the following steps:

if v=0, then embed one "bit" t from ternary sequence by setting $v' = t$;

if $0 < |v| < B$, $v' = \text{sign}(v) \cdot (|v| + 1)$;

if $|v| = B$, $v' = \text{sign}(v) \cdot (A+1)$;

if $|v| > B$, choose v' that minimizes $|p \cdot v' - q \cdot v|$.

ⓔ Recode the modified quantization table and changed quantized DCT coefficients to form a marked JPEG image via run-level coding and Huffman coding.

Data extraction process

ⓐ For each 8×8 quantized DCT coefficient block of luminance component from a JPEG stego image after Huffman decoding and run-level decoding, pick up coefficients that may carry message bits.

ⓑ For each selected quantized DCT coefficient v', do the following steps:

if $|v'| \leqslant 1$, set v=0; meanwhile, a ternary message bit t is extracted: t=v';

if $1 < |v'| \leqslant A$, then $v = \text{sign}(v') \cdot (|v'| - 1)$;

if $|v'| > A$, then choose v that minimizes $|p \cdot v' - q \cdot v|$.

ⓒ Concatenate every extracted bit to form a ternary sequence, and convert the ternary sequence to a binary sequence via inverse coding method.

Then, the original binary message sequence is extracted and the original cover is exactly restored.

2.2 Determine the Number of DCT Coefficients

When embedding data into quantized DCT coefficients, modification to low-frequency coefficients will result in serious degradation to subjective visual quality; on the contrary, modification to high-frequency coefficients will cause distinct decrease of PSNR, which represents image's objective visual quality. Thus, in our algorithm, we choose mid-frequency coefficients to embed message.

The number of message bits embedded into cover image is called payload. The required payload may vary drastically in different applications. In general, we are happy to see that our algorithm works well within large payload range. This can be fulfilled by adopting different numbers of coefficients in every 8×8 block.

To test the performance of our algorithm under various payloads, we implement several experiments for a Lena JPEG image with quantization parameter Q=50. We first array the DCT coefficients in a zigzag order and then select coefficients in the following manner: the numbers of coefficients (and corresponding positions) are 1 (12th), 2 (11th and 12th), 5 (10th to 14th), 8 (10th to 17th), and 11 (10th to 20th), respectively. The performances of these experiments are depicted in Fig. 2. Then, for every given payload, we choose the highest PSNR from five candidates. Then, by connecting the best PSNR points, we can draw our ultimate performance curve, as shown in Fig. 2. Every point on this curve stands for an achievable payload-PSNR pair. By carefully choosing coding parameter k and coefficient numbers, we can achieve the best image quality for any given payload.

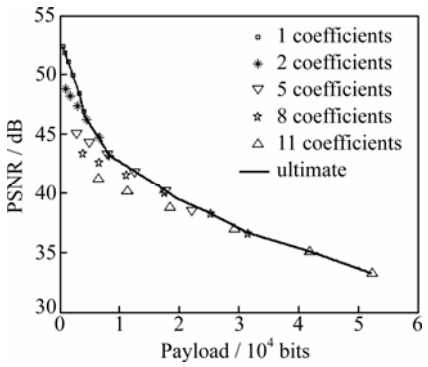

Fig. 2 Ultimate performance curve of proposed RDH algorithm

3 Experimental Results

To demonstrate the performance of proposed algorithm, we make a series of comparisons between the proposed algorithm and three previous arts in Refs. [13-15]. The test images are four standard images: Lena, Barbara, Airplane, and Baboon (Fig. 3). The raw images are in PGM format and we compress them with standard JPEG coder, during which the Q-factor is set as 50. If a reversible data embedding method can support large payload, low distortion, and low file-size increment, then we believe the method is good. Therefore, we compare PSNR and file-size increment, respectively. The comparison results are illustrated in Figs. 4 and 5.

(a) Lena(512×512) (b) Barbara(720×576) (c) Airplane(512×512) (d) Baboon(500×480)

Fig. 3 Test images

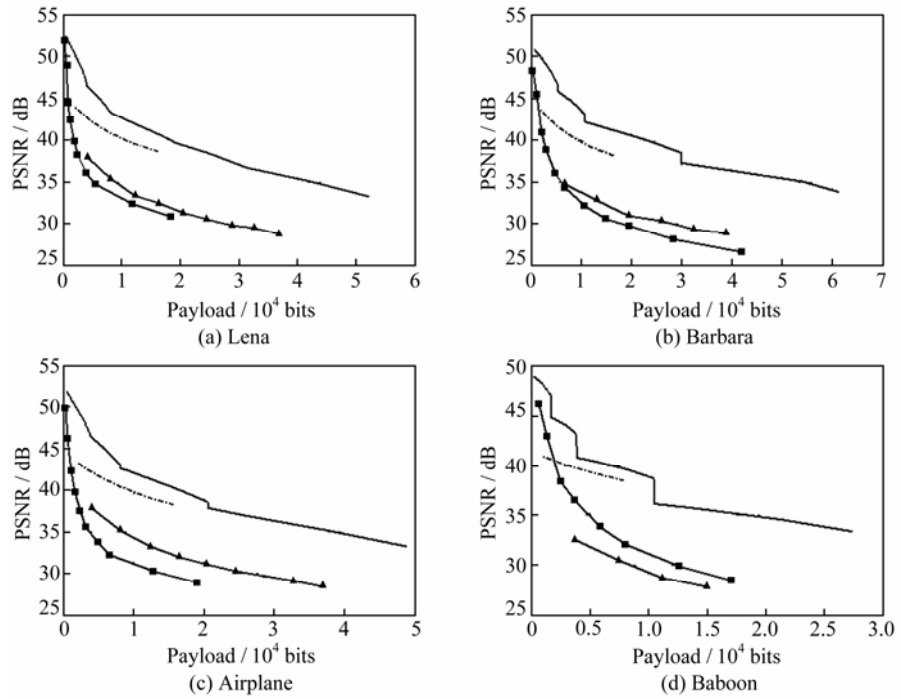

Fig. 4 PSNR comparison

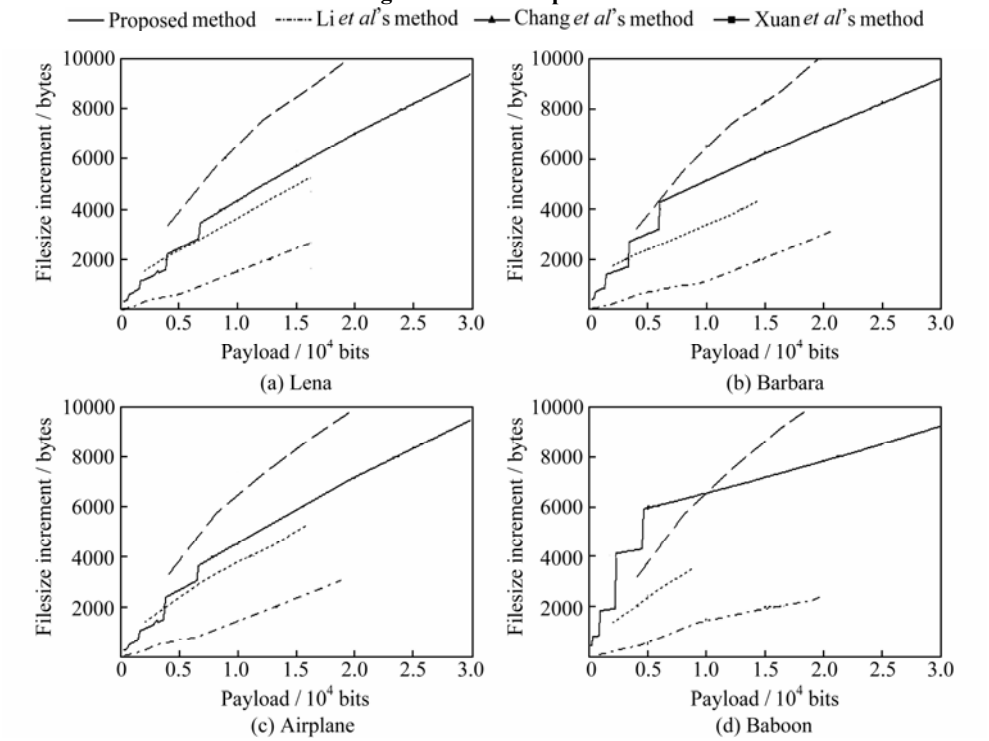

Fig. 5 File-size increment comparison

—Proposed method ····Li *et al* 's method – – Chang *et al* 's method -·-· Xuan *et al* 's method

Figure 4 shows the PSNR comparison, from which we can surely draw the conclusion that the proposed algorithm outperforms previous arts with regard to image quality. For each test image, the proposed algorithm can achieve a larger PSNR for all payload. It can provide an improvement of 2-6 dB in PSNR compared with Li *et al*'s work[14] throughout the payload axis. When compared with Chang *et al*'s work[13] and Xuan *et al*'s work [15], the improvement can be surprisingly more than 8 dB. Figure 5 shows the file-size increment comparison, which is not an advantage of our algorithm. The advantage of proposed method compared with Ref. [14] is preserving stego image's quality.

4 Conclusion

Li *et al*[14] proposed a reversible data hiding algorithm for JPEG images via histogram shift and quantization table modification, in which we found a hidden coding model of all-zero cover with two modification directions. If the model's character is sufficiently exploited, the performance of RDH algorithm can be improved considerably. Inspired by RZL coding in Ref. [16], we have designed a modified RZL coding method for the coding model behind Li *et al*'s method.

We take our coding method as a preprocessing tool to binary message sequence. Then, by combining the tool with Li *et al*'s reversible data hiding algorithm, we propose a novel reversible data hiding algorithm for JPEG images under preprocessing-embedding framework.

To test the performance of our algorithm, we implement our algorithm with several previous arts for different test images. The experiments demonstrate that our method is satisfactory in preserving stego image's visual quality but fails to have a prevailing advantage in restricting file-size increment.

References

[1] Petitcolas F, Anderson R, Kuhn M. Information hiding: A survey [J]. *Proceedings of the IEEE*, 1999, **87**: 1062-1078.

[2] Fridrich J, Goljan M. Lossless data embedding for all image formats [C]//*Proceedings of Photonics West*, *Electronic Imaging*, 2002, San Jose: SPIE Security and Watermarking of Multimedia Contents, 2002: 572-583.

[3] Honsinger C W, Jones P W, Rabbani M, *et al*. *Lossless Recovery of an Original Image Containing Embedded Data* [P]. US Patent #6278791, 2001.

[4] Fridrich J, Goljan M, Du R. Lossless data embedding: New paradigm in digital watermarking [J]. *EURASIP Journal on Applied Signal Processing*, 2002, **2002**(2): 185-196.

[5] Celik M U, Sharma G, Tekalp A M, *et al*. Lossless generalized-LSB data embedding [J]. *IEEE Trans on Image Processing*, 2005, **14**(2): 253-266.

[6] Tian J. Reversible data embedding using a difference expansion [J]. *IEEE Trans on Circuits System and Video Technology*, 2003, **13**(8): 890-896.

[7] Thodi D M, Rodriguez J. Expansion embedding techniques for reversible watermarking [J]. *IEEE Trans on Image Processing*, 2007, **16**(3): 721-730.

[8] Hu Y, Lee H K, Li J. DE-based reversible data hiding with improved overflow location map [J]. *IEEE Trans on Circuits System and Video Technology*, 2009, **19**(2): 250-260.

[9] Ni Z, Shi Y Q, Ansari N, *et al*. Reversible data hiding [J]. *IEEE Trans on Circuits System and Video Technology*, 2006, **16**(3): 354-362.

[10] Tsai P, Hu Y C, Yeh H L. Reversible image hiding scheme using predictive coding and histogram shifting [J]. *Signal Processing*, 2009, **89**: 1129-1143.

[11] Luo L, Chen Z, Chen M, *et al*. Reversible image watermarking using interpolation technique [J]. *IEEE Trans on Information Forensics and Security*, 2010, **5**(1): 187-193.

[12] Zhang X, Wang S, Qian Z, Feng G. Reversible fragile watermarking for locating tampered blocks in JPEG images [J]. *Signal Processing*, 2010, **90**: 3026-3036.

[13] Chang C C, Lin C C, Tseng C S, *et al* . Reversible hiding in DCT-based compressed images [J]. *Information Sciences*, 2007, **177**: 2768-2786.

[14] Li Q, Wu Y, Bao F. A reversible data hiding scheme for JPEG images [C]//*Advances in Multimedia Information Processing-Pacific Rim Conference on Multimedia* (*PCM* 2010) (*LNCS* 6297), Shanghai: Springer-Verlag, 2010: 653-664.

[15] Xuan G, Shi Q, Ni Z, *et al*. Reversible data hiding for JPEG images based on histogram pairs [C]//*International Conference on Image Analysis and Recognition* (*ICIAR* 2007) (*LNCS* 4633), Montreal: Springer-Verlag, 2007: 715-727.

[16] Wong K, Tanaka K, Takagi K, *et al*. Complete video quality-preserving data hiding [J]. *IEEE Trans on Circuits System and Video Technology*, 2009, **19**(10): 1499-1512.

□