The Journal of Systems and Software 86 (2013) 2166–2173

Contents lists available at SciVerse ScienceDirect

The Journal of Systems and Software

journal homepage: www.elsevier.com/locate/jss

An improved VLC-based lossless data hiding scheme for JPEG images

Yongjian Hu[a], Kan Wang[a], Zhe-Ming Lu[b,*]

[a] School of Electronic and Information Engineering, South China University of Technology, Wushan Road 381, Tianhe District, Guangzhou 510641, PR China
[b] School of Aeronautics and Astronautics, Zhejiang University, Hangzhou 310027, PR China

ARTICLE INFO

Article history:
Received 30 October 2012
Received in revised form 15 February 2013
Accepted 27 March 2013
Available online 6 April 2013

Keywords:
Information hiding
Lossless data hiding
JPEG bitstream

ABSTRACT

In this paper, a lossless data hiding scheme which directly embeds data into the bitstream of JPEG images is presented. For real cases, the JPEG code space is partly occupied, not all variable length codes (VLC) in the Huffman table are used during the JPEG image compression process. Thus, these unused VLCs can be made used of and data hiding can be performed by mapping one or more unused VLCs to one used VLC. Through analyzing the statistics of both used and unused VLCs, the proposed scheme can take full advantage of the unused VLCs by mapping Huffman codes according to a specific mapping strategy and reach higher capacity. The output stego image can keep exactly the same content as the original one and preserve the same file size, and if the file size is allowed to be enlarged, then our scheme can achieve a significant improvement of embedding capacity.

1. Introduction

With the rapid development of computer technology and the widely usage of the Internet, it becomes more and more convenient for people to access and exchange multimedia data. However, since anyone could access the data transmitted over the Internet and further tamper them, it makes the received data unreliable and thus arouses the public concern about the information security. Along with the economic globalization, secure data transmission involves not only politics and military, but also business, finance and individual privacy. Therefore, how to deliver the data safely becomes an important issue. Among various techniques, data hiding may be one of the most effective ways to solve the security problem. Data hiding technique embeds the secret data imperceptibly into the cover media by slightly modifying some of the cover elements, and thus the transmission of stego media (i.e., the media embedded with secret data) in public channels would not make irrelevant persons notice the existence of secret data. Moreover, data hiding technique can be applied to other aspects such as embedding patient's information into medical images and embedding geographic information into remote sensing satellite images, etc.

Data hiding techniques for images can be performed in various domains such as the spatial domain, transform domains and compressed domains. Most existing multimedia data are stored in compressed formats, thus the data hiding techniques for compressed images become much more practical. This paper focuses on the JPEG image format which is a widely used compressed format. For JPEG images, as early as in Upham (1997) developed a famous JPEG hiding tool Jpeg-Jsteg. Later, Westfeld (2001) developed the F5 algorithm to implement the matrix encoding in order to improve the embedding efficiency. Chang et al. (2002) modified the quantization table and hid the secret data in the cover image with its mid-frequency of the quantized DCT coefficients. Iwata et al. (2004) modified boundaries between zero and non-zero quantized DCT coefficients in each block to hide data. Tseng and Chang (2004) employed a capacity table to estimate the capacity of each DCT component and adopted the adaptive LSB substitution technique to embed data. Almohammad et al. (2009) extended Chang et al.'s method (2002) by using an optimized 16×16 quantization table and improved the quality and capacity of stego images. However, the above methods are traditional data hiding techniques which cause irreversible distortion to the cover media. Therefore, lossless data hiding has recently become a hot research topic which aims at embedding secret data into the cover media in the manner that both secret data and original cover media can be extracted and recovered without distortion. In several special fields such as medical, military and legal fields, lossless data hiding techniques are preferred compared with traditional data hiding techniques. For JPEG images, some lossless data hiding algorithms have been proposed. Fridrich et al. (2001) first presented two invertible watermarking methods for authenticating JPEG images. One losslessly compressed the LSB plane of some selected JPEG mode coefficients which made space for reversible data embedding. The other

* Corresponding author at: Room 105, Teaching Building 5, Yuquan Campus, Zhejiang University, Hangzhou 310027, PR China. Tel.: +86 571 87953349; fax: +86 571 87953349.
E-mail addresses: zheminglu@zju.edu.cn, zhemingl@yahoo.com (Z.-M. Lu).

0164-1212/$ – see front matter © 2013 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.jss.2013.03.102

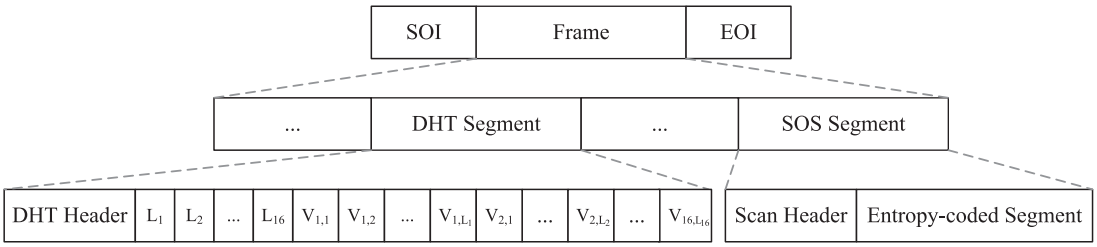

Fig. 1. JPEG bitstream structure.

modified the quantization matrix to enable lossless embedding of one bit per DCT coefficient. Chang et al. (2007) proposed a lossless and reversible steganography scheme for hiding secret data in each block of quantized DCT coefficients in JPEG images. They adopted two successive zero coefficients of the medium-frequency components in each block to hide the secret data. Xuan et al. (2007) proposed a lossless data hiding technique based on histogram pairs. It shifted the quantized DCT coefficient histogram with an optimum threshold and region and achieved good performance.

Most existing lossless data hiding techniques, including those mentioned above, increase the file size after embedding the secret data, which negates the advantages of the lossless data hiding scheme. Lossless data hiding with file size preservation is now a new important research subarea. Fridrich et al. (2004) introduced a lossless watermarking technique that preserves the file size for the first time. In their work, the watermark is embedded into the variable length integer (VLI) codes of AC coefficients encoded by run length encoding (RLE). Mobasseri et al. (2010) proposed an algorithm to embed data directly into the bitstream of JPEG image by flipping one bit of the VLC or the appended bits and applying an error concealment technique to minimize the image quality loss. Inspired by Mobasseri et al.'s method, Qian and Zhang (2012) presented a method to embed the secret data into the JPEG bitstream by Huffman code mapping, guaranteeing no quality distortion meanwhile more embedding capacity. Their method maps several unused codes to one used codes instead of flipping bits and concealing errors in the used codes.

Although Qian and Zhang (2012) have made use of the little redundancy existing in the JPEG bitstream, there is still potential free space which can be explored to hide data. In this paper, we improve their scheme and propose a lossless data hiding scheme in the JPEG bitstream that improves the hiding capacity. By analyzing the statistics of both used and unused VLCs, a specific mapping strategy is produced and the unused VLCs can be taken full advantage of. Experimental results show that the proposed scheme can reach a higher capacity. Embedding data into the JPEG bitstream, which is one of the open compression standards, has two limitations of weak security and fragility (Mobasseri et al., 2010). The JPEG bitstream must be viewable by normal viewers, and it is also available for the third party. Attackers may re-encode the JPEG image to erase the embedded data or replace it using the open algorithm. The proposed algorithm could be applied to some special fields such as embedding patient's information into medical images and embedding geographic information into remote sensing satellite images. Thus, secret information authentication is vital and the proposed scheme guarantees the integrity of the secret information. The rest of this paper is organized as follows. Section 2 briefly reviews the JPEG bitstream structure and Qian and Zhang's scheme. Section 3 describes the proposed data hiding scheme in detail. Experimental results are presented and explained in Section 4, followed by the conclusion in Section 5.

2. Related work

In this section, a brief introduction to the JPEG bitstream structure is given and then followed by a brief review of Qian and Zhang's scheme.

2.1. JPEG bitstream structure

An input gray-scale image is compressed into the JPEG format by the JPEG encoder and saved in the form of bitstream. Fig. 1 shows the structure of the JPEG bitstream. It starts with an SOI (Start of Image) marker and ends with an EOI (End of Image) marker. There is one frame containing several segments between the two markers. Among these segments, the DHT (Define Huffman Table) and SOS (Start of Scan) segments are associated with the proposed scheme as shown in Fig. 1. The data in the DHT segment are used to generate the Huffman table. For example, the AC coefficients are coded in a specific RLE format as intermediate symbols (*VLC*, *VLI*). Each *VLC* is defined as (*run*, *size*) and encoded by a Huffman code. Here, *run* denotes the zero run length, and each non-zero AC coefficient can be obtained by its corresponding *size* and *VLI* values. L_i in the DHT segment stands for the number of VLCs whose code length is i, and $V_{i,j}$ is the (*run*, *size*) value of the jth code whose code length is i. According to a specific rule, the Huffman table can be established itself with these data above. The SOS segment has a scan header and an entropy-coded segment. The image content is saved in the entropy-coded segment, and particularly the AC coefficients are represented in the form of several (*VLC*, *VLI*)s. More details can be seen in the JPEG guidelines (Int. Telecommunication Union, 1992).

2.2. Qian and Zhang's scheme

Qian and Zhang found that, in Mobasseri et al.'s method (2010), flipping one or more bits of VLC in a JPEG image to hide secret bits usually would caused the collision in the decoding process and thus error concealment steps are required to minimize the decrease of output JPEG image quality. Therefore, they mapped VLCs by direct VLC replacement in the JPEG bitstream and the image quality can be kept the same as the original (Qian and Zhang, 2012). Their scheme can be described as follows:

Step 1: Parse the JPEG bitstream and read all the VLCs to get the used and unused ones.
Step 2: Establish mapping relationships based on the number of the used VLCs and the number of the unused VLCs.
Step 3: Modify the $V_{i,j}$ value in the file header according to the mapping relationships.
Step 4: Replace the VLCs in the bitstream to embed the secret data.

All the VLCs can be classified into 16 categories $\{C_1, C_2, \ldots, C_{16}\}$ and Category C_i has L_i codes of length i (i=1, 2, . . ., 16). Thus, the used and unused VLCs in each category C_i can be expressed as follows:

$$C_i = \left\{ \mathrm{VLC}_{i,1}^{(\mathrm{u})}, \ldots, \mathrm{VLC}_{i,p_i}^{(\mathrm{u})}; \mathrm{VLC}_{i,1}^{(\mathrm{n})}, \ldots, \mathrm{VLC}_{i,q_i}^{(\mathrm{n})} \right\} \quad (1)$$

where $\mathrm{VLC}_{i,1}^{(\mathrm{u})}, \ldots, \mathrm{VLC}_{i,p_i}^{(\mathrm{u})}$ are p_i used codes, $\mathrm{VLC}_{i,1}^{(\mathrm{n})}, \ldots, \mathrm{VLC}_{i,q_i}^{(\mathrm{n})}$ q_i unused codes, and $p_i + q_i = L_i$. Here we briefly introduce the method they adopted to establish the mapping relationships. If $p_i \geq q_i > 0$, VLCs in each category are mapped by one-to-one manner:

$$M_i = \left\{ \{\mathrm{VLC}_{i,1}^{(\mathrm{u})} \leftrightarrow \mathrm{VLC}_{i,1}^{(\mathrm{n})}\}, \ldots, \{\mathrm{VLC}_{i,q_i}^{(\mathrm{u})} \leftrightarrow \mathrm{VLC}_{i,q_i}^{(n)}\} \right\} \tag{2}$$

where "↔" stands for the mapping relationship. If $0 < p_i < q_i$, VLCs are mapped by one-to-many manner for each category:

$$M_i = \left\{ \left\{ \mathrm{VLC}_{i,1}^{(\mathrm{u})} \leftrightarrow \{\mathrm{VLC}_{i,1}^{(\mathrm{n})}, \ldots, \mathrm{VLC}_{i,k_i}^{(\mathrm{n})}\} \right\}, \ldots, \left\{ \mathrm{VLC}_{i,p_i}^{(\mathrm{u})} \leftrightarrow \{\mathrm{VLC}_{i,(p_i-1)\times k_i+1}^{(\mathrm{n})}, \ldots, \mathrm{VLC}_{p_i \times k_i}^{(\mathrm{n})}\} \right\} \right\} \tag{3}$$

where $k_i = 2^{\lfloor \log_2(q_i/p_i+1) \rfloor} - 1$, and $\lfloor x \rfloor$ stands for the floor function. For Eq. (2), each code in the mapping relationship presents one secret bit "0" or "1", and for Eq. (3) each code can present $\lfloor \log_2(q_i/p_i + 1) \rfloor$ secret bits.

For example, assume there are four VLCs (VLC_m, VLC_n, VLC_k and VLC_l) and they are listed in the order of what they appear in the DHT segment. If the three unused VLCs (VLC_n, VLC_k and VLC_l) are mapped to the same used VLC_m, then the mapping set is $\{\mathrm{VLC}_m \leftrightarrow \{\mathrm{VLC}_n, \mathrm{VLC}_k, \mathrm{VLC}_l\}\}$. To map codes, the (*run*, *size*) values of VLC_n, VLC_k and VLC_l are all modified to that of VLC_m, then the four VLCs present secret bits "00", "01", "10" and "11" respectively. When scanning the entropy-coded segment in the data hiding phase, if VLC_m is met and the secret bits to be embedded are "01", then VLC_m is replaced with VLC_n.

Qian and Zhang successfully hide data into the JPEG bitstream by VLC mapping and replacement. After data are embedded, both the image quality and the file size of the stego image are kept the same as the original JPEG image. However, the statistical results of the used and unused codes are not made full use of. Thus, the code mapping relationships could be able to be better explored to further increase the capacity.

3. Proposed algorithm

The proposed scheme explores the potential free space by analyzing the statistics of the used and unused VLCs. The code mapping relationships are well designed and the unused VLCs can be properly utilized. The optimization of mapping relationships can further increase the capacity. The detailed optimization algorithm, embedding and extracting procedures are presented in the following subsections respectively.

3.1. Optimization algorithm to establish mapping relationships

Take AC coefficients for instance, there are 162 predefined VLCs in total. However, not all of the codes are used in a normal JPEG image. For this reason, the unused VLCs can be mapped to the used VLCs to present secret bits. Mapping codes must observe certain rules or the standard JPEG image viewers used by normal users will be unable to decode and display the stego image properly. As mentioned in Section 2.1, VLC corresponds to the (*run*, *size*) value which means one VLC indicates the current zero run length and the size of appending VLI. The arbitrary mapping VLC may lead to the misunderstanding of zero run length or the incorrect reading of appending VLI. To preserve the file size, we adopt the following code mapping rules: (1) the length of every code in the mapping set $\{\mathrm{VLC}_m, \mathrm{VLC}_n, \ldots, \mathrm{VLC}_l\}$ should be the same and (2) all codes should correspond to the same (*run*, *size*) value. Thus, the mapping relationships can be established inside each category C_i.

The above rules are the same as those used in Qian and Zhang's scheme. Now we turn to explain our optimization scheme for establishing optimal mapping relationships. We notice that the statistics of both used and unused VLCs can help design the mapping relationships to increase the capacity. Thus, the JPEG bitstream is parsed and the number of occurrences of every VLC is recorded first. A zero number of occurrences means that the corresponding VLC is unused. The VLCs in each category C_i are then sorted in the descending order according to their numbers of occurrences, which can be presented in the following form:

$$\mathrm{C}_i = \left\{ \mathrm{VLC}_{i,1}^{(\mathrm{u})'}, \ldots, \mathrm{VLC}_{i,p_i}^{(\mathrm{u})'}; \mathrm{VLC}_{i,1}^{(\mathrm{n})}, \ldots, \mathrm{VLC}_{i,q_i}^{(\mathrm{n})} \right\} \tag{4}$$

where $\mathrm{VLC}_{i,1}^{(\mathrm{u})'}, \ldots, \mathrm{VLC}_{i,p_i}^{(\mathrm{u})'}$ are p_i sorted used codes, $\mathrm{VLC}_{i,1}^{(\mathrm{n})}, \ldots, \mathrm{VLC}_{i,q_i}^{(\mathrm{n})}$ are q_i unused codes, and $p_i + q_i = L_i$.

If we go along with Qian and Zhang's way of thinking, we can develop a mapping method which takes the above statistics into consideration and call it *mapping method 1*. For the case $0 < q_i \leq p_i$, the VLCs in each category are mapped by one-to-one manner similar to Eq. (2) as follows:

$$M_i = \left\{ \{\mathrm{VLC}_{i,1}^{(\mathrm{u})'} \leftrightarrow \mathrm{VLC}_{i,1}^{(\mathrm{n})}\}, \ldots, \{\mathrm{VLC}_{i,q_i}^{(\mathrm{u})'} \leftrightarrow \mathrm{VLC}_{i,q_i}^{(\mathrm{n})}\} \right\} \tag{5}$$

In this case, not all used VLCs could be utilized for the reason $p_i \geq q_i$, thus the sorting keeps the q_i frequently appearing used VLCs and abandons the $(p_i - q_i)$ relatively seldom-appearing used VLCs. It avoids easily abandoning the last $(p_i - q_i)$ used VLCs in the original order which may contain the frequently appearing ones, and therefore helps to increase the capacity.

For the case $0 < p_i < q_i$, VLCs are mapped by the combination of one-to-many manner and one-to-one manner. One-to-k manner, for example, can present $\log_2(k+1)$ secret bits, where k satisfies the restrictive condition that $\log_2(k+1)$ is a positive integer. In other words, one-to-$(2^j - 1)$ manner can present j secret bits. For AC coefficients, there are 162 VLCs in total, and the largest category is C_{16} which has 125 VLCs, i.e., $L_{16} = 125$. The maximum j that satisfies $2^j - 1 \leq 125$ is 6, thus j could be 1, 2, 3, 4, 5 and 6, which means the parameter of one-to-k manner could be 1, 3, 7, 15, 31 and 63, respectively. In each category C_i, assuming the number of one-to-$(2^j - 1)$ manner mapping sets is $m_{i,j}$ $(1 \leq j \leq 6)$. To efficiently utilize the unused VLCs in this case, the selection of $m_{i,j}$ should satisfy the following condition:

$$\begin{aligned} \max \quad & Z = \sum_{j=1}^{6} j \cdot m_{i,j} \\ \text{s.t.} \quad & \begin{cases} \sum_{j=1}^{6} m_{i,j} \leq p_i \\ \sum_{j=1}^{6} (2^j - 1) \cdot m_{i,j} \leq q_i \\ m_{i,j} \geq 0, j = 1, 2, \ldots, 6 \\ m_{i,j} \quad \text{integer}, j = 1, 2, \ldots, 6 \end{cases} \end{aligned} \tag{6}$$

This is an integer linear programming (ILP) problem. After Eq. (6) is solved, all code mapping relationships can be expressed as:

$$\begin{aligned} M_i = & \left\{ \{\mathrm{VLC}_{i,1}^{(\mathrm{u})'} \leftrightarrow \{\mathrm{VLC}_{i,1}^{(\mathrm{n})}, \ldots, \mathrm{VLC}_{i,63}^{(\mathrm{n})}\}\}, \ldots, \left\{ \mathrm{VLC}_{i,m_{i,6}}^{(\mathrm{u})'} \leftrightarrow \left\{ \mathrm{VLC}_{i,63\cdot(m_{i,6}-1)+1}^{(\mathrm{n})}, \ldots, \mathrm{VLC}_{i,63\cdot m_{i,6}}^{(\mathrm{n})} \right\} \right\}, \\ & \left\{ \mathrm{VLC}_{i,m_{i,6}+1}^{(\mathrm{u})'} \leftrightarrow \{\mathrm{VLC}_{i,63\cdot m_{i,6}+1}^{(\mathrm{n})}, \ldots, \mathrm{VLC}_{i,63\cdot m_{i,6}+31}^{(\mathrm{n})}\} \right\}, \ldots, \left\{ \mathrm{VLC}_{i,\sum_{j=1}^{6} m_{i,j}}^{(\mathrm{u})'} \leftrightarrow \mathrm{VLC}_{i,\sum_{j=1}^{6} [(2^j-1)\cdot m_{i,j}]}^{(\mathrm{n})} \right\} \end{aligned} \tag{7}$$

That's to say, the first $m_{i,6}$ sorted used VLCs are mapped by the one-to-63 manner, the next $m_{i,5}$ sorted used VLCs are mapped by the one-to-31 manner, ..., and the last $m_{i,1}$ sorted used VLCs are mapped by the one-to-one manner.

The above mapping method only makes the unused VLCs be able to be mapped to as many used VLCs as possible. In fact, there is still certain potential space to further increase the capacity. In real cases, some used VLCs may appear with a very high frequency. Thus, it is even worth abandoning parts of the used VLCs and mapping more unused VLCs to the frequently used VLCs, which can help to carry more secret bits. The following method which considers the above situations is called *mapping method 2*. For the case $p_i > 0$ and $q_i > 0$, VLCs are mapped by the combination of one-to-many manner and one-to-one manner. Thus, in *mapping method 2*, we adopt the same code mapping relationships as given in Eq. (7), however the selection of $m_{i,j}$ should satisfy the following condition:

$$\begin{aligned}\max Z = {} & 6\cdot\sum_{v=1}^{m_{i,6}}\mathrm{count}(i,v) + 5\cdot\sum_{v=m_{i,6}+1}^{m_{i,6}+m_{i,5}}\mathrm{count}(i,v) + 4\cdot\sum_{v=m_{i,6}+m_{i,5}+1}^{m_{i,6}+m_{i,5}+m_{i,4}}\mathrm{count}(i,v) + 3\cdot\sum_{v=m_{i,6}+m_{i,5}+m_{i,4}+1}^{m_{i,6}+m_{i,5}+m_{i,4}+m_{i,3}}\mathrm{count}(i,v)\\ & +2\cdot\sum_{v=m_{i,6}+m_{i,5}+m_{i,4}+m_{i,3}+1}^{m_{i,6}+m_{i,5}+m_{i,4}+m_{i,3}+m_{i,2}}\mathrm{count}(i,v) + \sum_{v=m_{i,6}+m_{i,5}+m_{i,4}+m_{i,3}+m_{i,2}+1}^{m_{i,6}+m_{i,5}+m_{i,4}+m_{i,3}+m_{i,2}+m_{i,1}}\mathrm{count}(i,v)\end{aligned}$$

$$\text{s.t.}\begin{cases}\sum_{j=1}^{6} m_{i,j} \le p_i\\ \sum_{j=1}^{6}(2^j-1)\cdot m_{i,j} \le q_i\\ m_{i,j} \ge 0, j = 1,2,\ldots,6\\ m_{i,j} \text{ are integers}, \quad j = 1,2,\ldots,6\end{cases} \tag{8}$$

where the function $\mathrm{count}(i,v)$ presents the number of times that the vth sorted used code in category C_i appears in the entropy-coded segment. After Eq. (8) is solved, by substituting them into Eq. (7), we can get the optimal code mapping relationships.

3.2. Extension to the application case where file size increase is acceptable

From Section 3.1, we establish the mapping relationships inside each category C_i in order to preserve the original file size. In fact, we can extend the above scheme to the situations where slight file size increase is allowed. In this case, the mapping relationships are not required to be established inside each category. They can be established among all categories. The file size increase is induced by mapping the short used VLCs to long unused VLCs, and a threshold TH could be set to constrain the selection of used VLCs. The search of used VLCs can be performed among a limited number of categories, i.e., those categories C_i whose index i is not smaller than TH, so that certain shortest VLCs would not be chosen. However, the selection of unused VLCs is not restrained. Suppose the number of selected used VLCs is p and the number of unused ones is q. For the case $0 < p \le q$, VLCs can be mapped by the combination of one-to-many manner and one-to-one manner. However, for the case $p > q > 0$, even all unused codes are mapped by one-to-one manner, no more than q used codes will be utilized. Thus, p' used codes and q unused codes will be chosen. The definition of p' is as follows:

$$p' = \begin{cases} p, & 0 < p \le q\\ q, & p > q > 0\end{cases} \tag{9}$$

The first p' most frequently appearing used codes are selected and sorted. Likewise, the q unused codes are sorted in the ascending order of their lengths for the purpose that the shorter unused codes could be mapped to the more frequently appearing used codes:

$$C = \{\mathrm{VLC}_1^{(\mathrm{u})'}, \ldots, \mathrm{VLC}_{p'}^{(\mathrm{u})'}; \mathrm{VLC}_1^{(\mathrm{n})'}, \ldots, \mathrm{VLC}_q^{(\mathrm{n})'}\} \tag{10}$$

It helps to control the increase of file size. Note that all VLCs are taken here, i.e., there are 162 VLCs in total for AC coefficients. Hence, the maximum j that satisfies $2^j - 1 \le 162$ is 7 and we suppose the number of one-to-(2^j-1) manner mapping sets is m_j $(1 \le j \le 7)$. The rest part is similar to *mapping method 2* as depicted in Section 3.1. The selection of m_j should satisfy the following condition:

$$\begin{aligned}\max Z = {} & 7\cdot\sum_{v=1}^{m_7}\mathrm{count}(v) + 6\cdot\sum_{v=m_7+1}^{m_7+m_6}\mathrm{count}(v) + 5\cdot\sum_{v=m_7+m_6+1}^{m_7+m_6+m_5}\mathrm{count}(v) + 4\cdot\sum_{v=m_7+m_6+m_5+1}^{m_7+m_6+m_5+m_4}\mathrm{count}(v) + 3\cdot\sum_{v=m_7+m_6+m_5+m_4+1}^{m_7+m_6+m_5+m_4+m_3}\mathrm{count}(v)\\ & +2\cdot\sum_{v=m_7+m_6+m_5+m_4+m_3+1}^{m_7+m_6+m_5+m_4+m_3+m_2}\mathrm{count}(v) + \sum_{v=m_7+m_6+m_5+m_4+m_3+m_2+1}^{m_7+m_6+m_5+m_4+m_3+m_2+m_1}\mathrm{count}(v)\end{aligned}$$

$$\text{s.t.}\begin{cases}\sum_{j=1}^{7} m_j \le p'\\ \sum_{j=1}^{7}(2^j-1)\cdot m_j \le q\\ m_j \ge 0, j = 1,2,\ldots,7\\ m_j \text{ are integers}, j = 1,2,\ldots,7\end{cases} \tag{11}$$

where the function $\mathrm{count}(v)$ stands for the number of times that the vth sorted used code appears in the entropy-coded segment. After

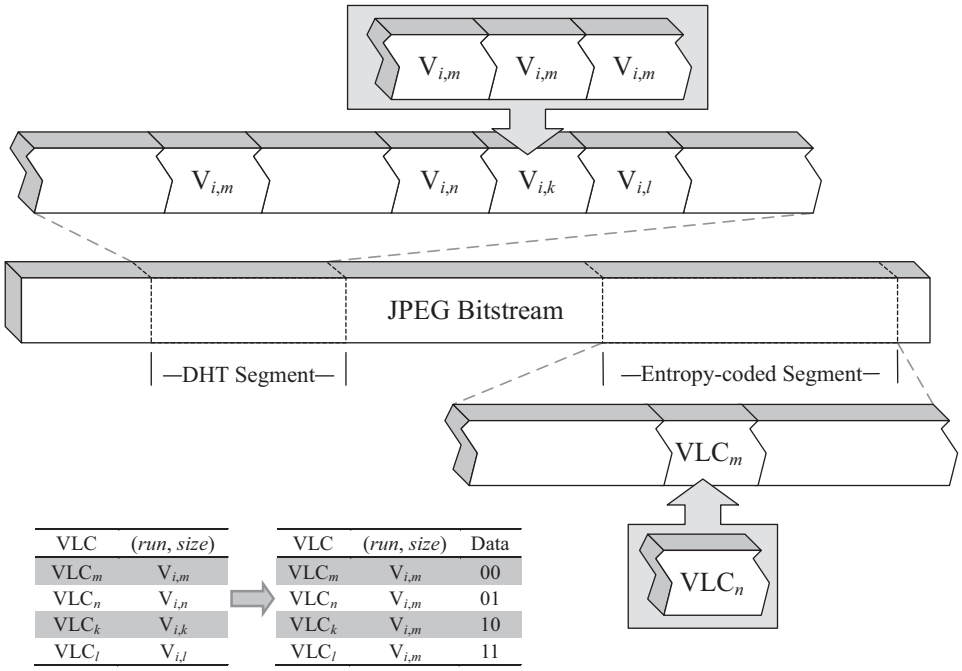

Fig. 2. The sketch map of the proposed data hiding scheme by replacing VLCs.

Eq. (11) is solved, the code mapping relationships can be expressed as:

$$M_i = \left\{ \{VLC_1^{(u)'} \leftrightarrow \{VLC_1^{(n)}, \ldots, VLC_{127}^{(n)}\}\}, \ldots, \left\{ VLC_{m_7}^{(u)'} \leftrightarrow \{VLC_{127\cdot(m_7-1)+1}^{(E)}, \ldots, VLC_{127\cdot m_7}^{(n)}\} \right\}, \right.$$
$$\left. \left\{ VLC_{m_7+1}^{(u)'} \leftrightarrow \{VLC_{127\cdot m_7+1}^{(n)}, \ldots, VLC_{127\cdot m_7+63}^{(n)}\} \right\}, \ldots, \left\{ VLC_{\sum_{j=1}^{7} m_j}^{(u)'} \leftrightarrow VLC_{\sum_{j=1}^{7}[(2^j-1)\cdot m_j]}^{(n)} \right\} \right\} \quad (12)$$

That is to say, the first m_7 sorted used VLCs are mapped by the one-to-127 manner, the next m_6 sorted used VLCs are mapped by the one-to-63 manner, ..., and the last m_1 sorted used VLCs are mapped by the one-to-one manner.

3.3. Data embedding and extracting procedures

The proposed scheme embeds data into the bitstream of JPEG images based on VLC mapping and replacing. Our scheme can be divided into two procedures: the data embedding procedure and the data extracting procedure. After collecting the VLC statistical information and establishing optimization mapping relationships, the proposed scheme embeds data into the VLCs in the entropy-coded segment. Fig. 2 shows how secret bits are embedded by replacing VLCs. The detailed data embedding steps are as follows:

Step 1: Parse the JPEG bitstream, read all the VLCs in the entropy-coded segment and get statistical results of the number of occurrences for every used VLC.
Step 2: According to the statistical results, establish the mapping relationships based on the mapping method mentioned in Sections 3.1 and 3.2.
Step 3: Encrypt the original secret data and then modify the $V_{i,j}$ (*run*, *size*) value in the DHT segment in accordance with the mapping relationships.
Step 4: Replace the VLCs in the entropy-coded segment to embed secret bits.

Because the data embedding procedure will keep the original image content without impacting the image quality, no restoring procedure is required. The data hidden in the entropy-coded segment can be extracted according to a lookup table, which is generated by scanning the Huffman table. Fig. 3 shows an example of creating a lookup table. The VLCs with the same (*run*, *size*) value are divided into groups, and the bits that each VLC could present can be calculated by counting the number of occurrences of the same (*run*, *size*) value, i.e., the binary logarithm of the number of occurrences. Four rows with gray background in Fig. 3 are the VLCs with the same (*run*, *size*) value 83. Therefore each of them can present $\log_2 4 = 2$ bits and the data are listed in the column *Value* and recorded in decimal numbers. After a stego JPEG image is received, secret data can be extracted by following steps:

Step 1: Read the DHT segment of the stego JPEG image to reconstruct the Huffman table.
Step 2: Scan the Huffman table and find the VLC mapping sets with the same (*run*, *size*) value to create a lookup table containing VLCs and corresponding secret bits.
Step 3: Parse the JPEG bitstream, read the VLCs in the entropy-coded segment to extract secret bits according to the lookup table and then decrypt them back to achieve original secret data.

3.4. Security analysis

In the proposed scheme, the mapping relationships are established by modifying the DHT segment of the JPEG header. Because the JPEG bitstream must be viewable by common viewers, the

No.	VLC	(*run*, *size*)	Bits	Value
1	00	1	0	-
2	01	2	0	-
3	100	3	0	-
4	1010	0	0	-
…	…	…	…	…
64	1111111110011100	147	4	3
65	1111111110011101	147	4	4
66	1111111110011110	83	2	0
67	1111111110011111	84	1	0
68	1111111110100000	85	3	0
…	…	…	…	…
154	1111111111110110	83	2	1
155	1111111111110111	83	2	2
156	1111111111111000	83	2	3
157	1111111111111001	84	1	1
158	1111111111111010	99	1	1
159	1111111111111011	23	1	1
160	1111111111111100	115	1	1
161	1111111111111101	131	1	1
162	1111111111111110	67	1	1

Fig. 3. An example of creating a lookup table in the extracting phase.

mapping relationships cannot be concealed. The modified Huffman table is viewable by the third party, thus carefully observation may cause the finding of the existence of the mapping relationships. However, it could be encrypted to further improve the security of embedded information. For example, there is a mapping set {VLC$_m$, VLC$_n$, VLC$_k$, VLC$_l$} and the four VLCs are listed in the order of what they appear in the DHT segment, they present secret bits "00", "01", "10" and "11" respectively. If we use a secret key to shuffle the order of the whole predefined VLCs and generate a new order list, which leads to a different lookup table, then VLC$_m$, VLC$_n$, VLC$_k$ and VLC$_l$ may present "10", "00", "11" and "01" according to the lookup table. The shuffle is equivalent to encrypting the mapping relationships. On the receiver side, the same secret key should be used to get the shuffled order and extract the secret information.

In the data embedding phase, a secret key k_1 encrypting the secret information and another secret key k_2 encrypting the mapping relationships guarantee the security of the embedded data. In some situations, attackers may even erase the embedded data and replace them with other information. Thus, we can add the secret information authentication part to the scheme by computing the MIC (Message Integrity Code) and storing it in the user fields of JPEG header. MIC is computed by hashing the original embedded data with a secret key k_3. For example, MD2 is one of the useful hashing functions (Mobasseri et al., 2010; Qian and Zhang, 2012). The receiver extracts the embedded data and calculates the new MIC from them. By comparing the calculated MIC and the stored MIC, the receiver can authenticate the embedded data.

4. Experimental results

To evaluate the performance of the proposed scheme, ten test images obtained from the USC-SIPI image database (http://sipi.usc.edu/database) are used. These images are the same as those are used in Qian and Zhang (2012) and provided in 512×512 TIFF format which are further converted to grayscale JPEG images with different quality factors using the Netpbm image format conversion program (Netpbm, http://netpbm.sourceforge.net/). Comparisons of hiding capacity

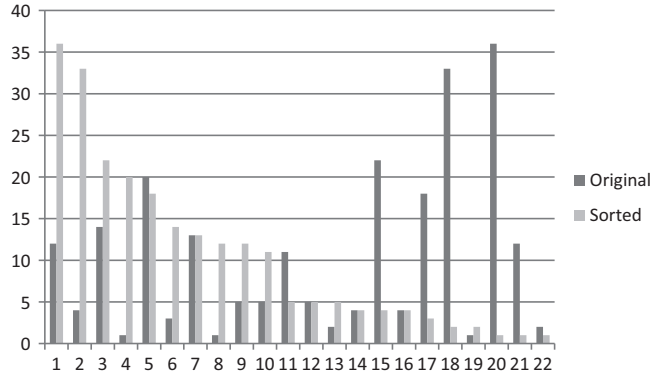

Fig. 4. Statistical results of 22 used VLCs in Category C_{16} (for the test image F16 with quality factor 70).

and embedding efficiency between Qian and Zhang's scheme and our scheme are discussed.

Experimental results tested on ten test images with different quality factors are listed in Table 1, where the results of Qian and Zhang's method (2012) are also listed. From Table 1, we can see that the capacities of both our method 1 and our method 2 are larger than Qian and Zhang's method and so is our method 2 than our method 1. To demonstrate the capacity increase of our methods compared with Qian's methods more clearly, Table 2 presents the comparison results from the perspective of average capacity across all quality factors. Our method 1 increases the embedding bits by 28–43% compared with Qian and Zhang's method, while our method 2 by 31–63%.

The capacities of our methods and Qian and Zhang's method are not fixed, and they depend on the statistical results of the used and unused VLCs. Different image contents lead to different statistical results. Table 2 reveals that the image with more similar content blocks usually has higher capacity. Take the test JPEG image F16 with quality factor 70 for example, there are 22 used VLCs and 103 unused VLCs in Category C_{16}. The occurrence distribution of the used VLCs in Category C_{16} appearing in the entropy-coded segment is given in Fig. 4. The left blue bar in each bin is in the original order (12, 4, 14, 1, 20, 3, 13, 1, 5, 5, 11, 5, 2, 4, 22, 4, 18, 33, 1, 36, 12 and 2) and the right red one (36, 33, 22, 20, 18, 14, 13, 12, 12, 11, 5, 5, 5, 4, 4, 4, 3, 2, 2, 1, 1 and 1) is in the sorted order. For Qian and Zhang's method, VLCs are mapped in the original order, while they are mapped in the sorted order for our methods. Priority in mapping by the one-to-many manner with higher capacity will be given to those with larger occurrence, which helps to increase the embedding capacity. In Qian and Zhang's scheme, every used VLC in Category C_{16} can carry $k_{16} = \lfloor \log_2(q_{16}/p_{16} + 1) \rfloor = \lfloor \log_2(103/22 + 1) \rfloor = 2$ bits, thus all VLCs in Category C_{16} in the entropy-coded segment can carry 456 bits. For our method 1, $m_{16,1}$, $m_{16,2}$, $m_{16,3}$, $m_{16,4}$, $m_{16,5}$ and $m_{16,6}$ are 0, 13, 9, 0, 0 and 0 respectively, thus all VLCs in Category C_{16} can carry 636 bits. For our method 2, $m_{16,1}$, $m_{16,2}$, $m_{16,3}$, $m_{16,4}$, $m_{16,5}$ and $m_{16,6}$ are 6, 1, 7, 3, 0 and 0, respectively, thus all VLCs in Category C_{16} can carry 699 bits. From Fig. 4 it can be seen that the values of first several red bars are much higher than the last several ones, therefore our method 2 abandons the last several ones and thus more unused VLCs can be mapped to the first several ones.

The numbers of VLC mapping sets generated by Qian and Zhang's method and our method 2 are compared in Fig. 5. Since the number of our method 1 is similar to that of Qian and Zhang's method, we do not present it in Fig. 5. The abandon of last several sorted used VLCs and mapping more unused VLCs to first several sorted used VLCs make the number of sets by our method 2 below that of Qian and Zhang's. Fig. 6 shows the comparison results of the

Table 1
Comparison of embedding capacity (bits) between Qian and Zhang's method and our methods with different quality factors.

Image	Method	Quality factor								
		10	20	30	40	50	60	70	80	90
Baboon	Qian and Zhang's	4954	2263	1278	1588	1080	1005	1082	615	346
	Our method 1	6569	3463	1692	1746	1204	1172	1219	669	518
	Our method 2	6576	3478	1758	1790	1268	1304	1353	757	681
Boat	Qian and Zhang's	484	388	522	481	522	714	370	510	978
	Our method 1	950	551	680	548	633	782	572	725	1408
	Our method 2	950	551	691	571	687	811	679	1000	1929
Bridge	Qian and Zhang's	733	675	614	755	781	501	436	306	350
	Our method 1	1802	996	699	833	912	599	635	446	491
	Our method 2	1802	999	713	860	1035	714	817	596	615
Elaine	Qian and Zhang's	116	142	177	328	402	218	272	576	1002
	Our method 1	341	264	200	414	531	302	365	662	1328
	Our method 2	341	264	200	414	553	326	409	788	1745
F16	Qian and Zhang's	615	639	722	737	760	536	487	404	464
	Our method 1	791	729	865	881	812	763	678	507	493
	Our method 2	791	764	904	961	886	860	741	553	595
Gray21	Qian and Zhang's	201	447	694	640	767	796	913	1286	2154
	Our method 1	235	502	865	987	900	897	1463	1616	2868
	Our method 2	235	502	904	987	900	929	1510	1675	3207
Lena	Qian and Zhang's	261	210	250	291	364	188	204	249	294
	Our method 1	565	326	257	296	368	198	256	316	436
	Our method 2	565	326	259	296	369	204	286	351	576
Peppers	Qian and Zhang's	391	380	491	383	389	473	295	280	768
	Our method 1	567	513	698	487	534	616	415	280	1065
	Our method 2	567	517	717	493	557	652	455	374	1562
Splash	Qian and Zhang's	330	481	632	724	871	1112	653	714	284
	Our method 1	497	581	967	1147	1273	1334	979	953	566
	Our method 2	497	597	1064	1345	1382	1547	1180	1147	710
Tiffany	Qian and Zhang's	4404	500	345	485	467	563	724	590	256
	Our method 1	4721	1138	663	888	705	803	920	742	506
	Our method 2	4721	1153	681	907	786	875	1002	956	720

Table 2
Comparison of average capacity (bits) between Qian and Zhang's method and our methods across JPEG quality factors from 10 to 90.

Image	Qian and Zhang's method	Our method 1		Our method 2	
		Capacity	Increase (%)	Capacity	Increase (%)
Baboon	1579.00	2028.00	28.44	2107.22	33.45
Boat	552.11	761.00	37.83	874.33	58.36
Bridge	572.33	823.67	42.91	905.67	58.24
Elaine	359.22	489.67	36.31	560.00	55.89
F16	596.00	724.33	21.53	783.89	31.52
Gray21	877.56	1148.11	30.83	1205.44	37.36
Lena	256.78	335.33	30.59	359.11	39.85
Peppers	427.78	575.00	34.42	654.89	53.09
Splash	644.56	921.89	43.03	1052.11	63.23
Tiffany	926.00	1231.78	33.02	1311.22	41.06

embedding efficiency which is defined as the capacity dividing by the number of mapping sets. It can be seen that our method 2 has higher efficiency for it can achieve higher capacity with the lower number of mapping sets.

Table 3 lists the performance of the proposed method in terms of the image file size when the file size change is allowed. Experiments are performed on five test images with quality factor 70. The capacities are not fixed, either. The image content will affect the capacity and file size. From Table 3 we can see that as the threshold *TH* increases, the capacity decreases and so does the file size. For small *THs*, the capacity is very high. Nonetheless, the file size growth after hiding data is not acceptable. Large *THs*

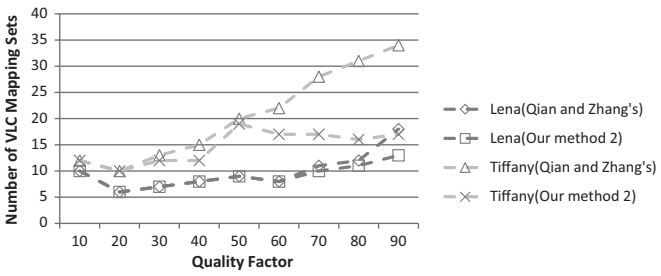

Fig. 5. Comparisons of the number of VLC mapping sets between our method 2 and Qian and Zhang's method with different quality factors.

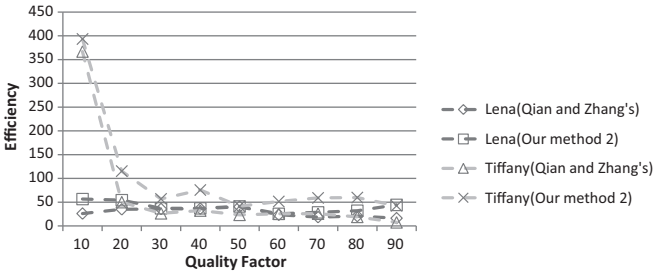

Fig. 6. Comparisons of embedding efficiency between Qian and Zhang's method and our method 2 with different quality factors when the file size change is allowed.

Table 3
Performance of our scheme in the application case where the file size increase is allowed (quality factor = 70).

Image	Original file size (KB)	Evaluation criteria	*TH*						
			4	5	6	7	8	9	10
Baboon	61.0	Capacity (bit)	119,513	56,672	31,421	21,176	12,486	8436	3493
		File size (KB)	113.0	84.6	73.0	68.6	65.1	63.6	61.9
Boat	36.6	Capacity (bit)	73,174	33,552	18,891	12,223	7223	5347	2925
		File size (KB)	69.1	50.8	44.1	41.3	39.0	38.2	37.2
F16	30.0	Capacity (bit)	64,676	27,829	13,987	8252	3969	2520	1470
		File size (KB)	58.6	41.6	35.3	33.0	31.3	30.7	30.3
Lena	28.6	Capacity (bit)	65,325	29,198	15,976	8879	4874	3010	1557
		File size (KB)	56.7	40.4	34.6	31.8	30.1	29.5	28.9
Peppers	29.5	Capacity (bit)	65,759	29,287	15,642	8630	4586	3002	1466
		File size (KB)	58.7	41.5	35.5	32.8	31.1	30.5	29.9

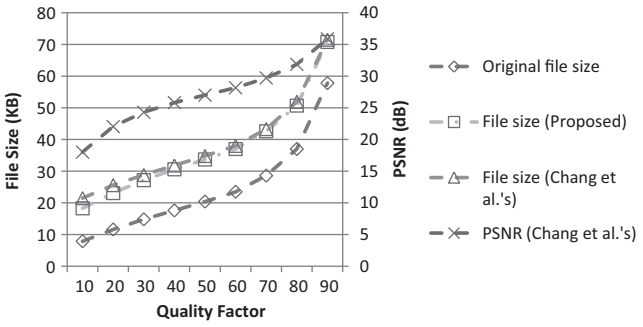

Fig. 7. Comparisons of image file size and PSNR between the proposed method and Chang et al.'s method.

result in a satisfactory file size growth at the expense of lowing capacity. A proper threshold balances the capacity and the file size. Threshold could be selected according to the data to be hidden. Suppose we need to hide 32,768 bits into the JPEG image Lena, i.e., the embedding rate is 0.125 bpp, *TH* is recommended to be set to 4.

Fig. 7 compares the proposed method with Chang et al.'s method (2007) in terms of the image file size and PSNR when the file size change is allowed. Experiments are performed on the test image Lena with quality factors ranging from 10 to 90. Fig. 7 shows the results when hiding 32,768 bits. For each case, the file size of the proposed method is similar to that of Chang et al.'s and a little closer to the original file size. Note that the PSNR of Chang et al.'s changes from 18 dB to 36 dB, while the proposed method can keep the original image's content and quality. Therefore, the proposed method can provide high capacity, acceptable file size growth and image quality preservation.

5. Conclusions

In this paper, a lossless data hiding scheme with file size preservation is proposed. Through analyzing the code space and the statistics of both used and unused VLCs, we find there is still potential free space in the JPEG bitstream that can be explored to hide data. The unused VLCs are made the best of and mapped to the used VLCs in specific mapping manners. The proposed scheme embeds data into the VLC codes and what mapped VLCs present are not changed, therefore the image content after data hiding is exactly the same as the original one. Experimental results demonstrate that the proposed scheme can reach better performance no matter whether the file size preservation is required or not.

References

Almohammad, A., Ghinea, G., Hierons, R.M., 2009. JPEG steganography: a performance evaluation of quantization tables. In: Proc. 2009 International Conference on Advanced Information Networking and Applications, Bradford, United Kingdom, pp. 471–478.

Chang, C.C., Chen, T.S., Chung, L.Z., 2002. A steganographic method based upon JPEG and quantization table modification. Information Sciences 141 (1–2), 123–138.

Chang, C.C., Lin, C.C., Tseng, C.S., Tai, W.L., 2007. Reversible hiding in DCT-based compressed images. Information Sciences 177 (13), 2768–2786.

Fridrich, J., Goljan, M., Du, R., 2001. Invertible authentication watermark for JPEG images. In: Proc. IEEE International Conference on Information Technology: Coding and Computing, Las Vegas, Nevada, USA, pp. 223–227.

Fridrich, J., Goljan, M., Chen, Q., Pathak, V., 2004. Lossless data embedding with file size preservation. In: Proc. EI SPIE Security and Watermarking of Multimedia Contents VI, vol. 5306, San Jose, California, pp. 354–365.

Int. Telecommunication Union CCITT Recommendation T.81, Infor-mation Technology-Digital Compression and Coding of Continuous-tone Still Images-Requirements and Guidelines 1992.

Iwata, M., Miyake, K., Shiozaki, A., 2004. Digital steganography utilizing features of JPEG images. IEICE Transactions on Fundamentals E87-A (4), 929–936.

Mobasseri, B.G., Berger, R.J., Marcinak, M.P., NaikRaikar, Y.J., 2010. Data embedding in JPEG bitstream by code mapping. IEEE Transactions on Image Processing 19 (4), 958–966.

Qian, Z., Zhang, X., 2012. Lossless data hiding in JPEG bitstream. The Journal of Systems and Software 85 (2), 309–313.

Tseng, H.W., Chang, C.C., 2004. High capacity data hiding in JPEG-compressed images. Informatica 15 (1), 127–142.

D. Upham, 1997. JPEG-JSteg, http://www.funet.fi/pub/crypt/steganography/jpeg-jsteg-v4.diff.gz

Westfeld, A., 2001. F5-a steganographic algorithm: high capacity despite better steganalysis. In: Proc. 4th International Workshop on Information Hiding, Pittsburgh, PA, USA, pp. 289–302.

Xuan, G.R., Shi, Y.Q., Ni, Z.C., et al., 2007. Reversible data hiding for JPEG images based on histogram pairs. In: Proc. International Conference on Image Analysis and Recognition, Montreal, Canada, pp. 715–727.

Yong-Jian Hu received the B.S. degree in Electrical Engineering from Hubei University of Automotive Technology, Shiyan, P. R. China, in 1984, and the M.S. degrees in Information and Control Engineering from Xi'an Jiaotong University, Xi'an, P. R. China, in 1990, and the Ph. D. degree in Electronic and Information Engineering at South China University of Technology, Guangzhou, P. R. China, in 2002. He is currently working as an full professor in South China University of Technology, Guangzhou, P. R. China. His research interests include multimedia single analysis and processing, information hiding and astronautics signal processing, etc.

Kan Wang received the B.S. degree in Electronic and Information Engineering from Guangdong University of Technology, Guangzhou, P. R. China in 2011 and is currently working toward the master degree in Electronic and Information Engineering at South China University of Technology, Guangzhou, P. R. China. His research interests include information hiding and image processing.

Zhe-Ming Lu received the B.S. and M.S. degrees in Electrical Engineering from Harbin Institute of Technology (HIT), Harbin, P. R. China, in 1995 and 1997 respectively, and the Ph. D. degree in instrument science and technology from HIT, Harbin, P. R. China, in 2001. He is currently working as an full professor in Zhejiang University, Hangzhou, China. His research interests include multimedia single analysis and processing, information hiding and astronautics signal processing, etc.