Journal of Imaging Science and Technology® 55(5): 050502-1–050502-15, 2011.

Lossless Data Hiding for JPEG Pictures

Yih-Chuan Lin and Tzung-Shian Li
Department of Computer Science and Information Engineering, National Formosa University, Yunlin, Taiwan
E-mail: lyc@nfu.edu.tw

Abstract. *In this article, the authors propose a novel interpolation-based reversible data hiding algorithm for use with images that are encoded in the Joint Photographic Expert Group standard format. This algorithm is able to exactly recover the original coded quality of the image, once the hidden message has been extracted. The proposed algorithm embeds message data into discrete cosine transform blocks by calculating the difference values between the nonzero quantized AC coefficients and their interpolated counterparts and then shifting the histogram of those difference values. Simple linear interpolation is employed to exploit the correlation between the AC coefficients that are colocated across neighboring blocks for the purpose of creating hiding capacity for the insertion of data bits into the AC coefficients. The embedding of data is performed via the shifting of the histogram of interpolation differences. Our experimental results demonstrate that the proposed method not only provides an increased embedding capacity but also maintains a high image quality. Moreover, as an additional processing delay is not introduced and only simple arithmetic computations are involved, the proposed algorithm can operate in real time. © 2011 Society for Imaging Science and Technology.*
[DOI: xxxxxx]

INTRODUCTION

Due to rapid advances in imaging technology and multimedia communications, digital images, such as pictures, are now widespread in our societies and daily lives. At the current time, individuals with modern handheld devices or camcorders can capture pictures and exchange them conveniently using wireless communications networks or the public internet. In addition to the traditional data and voice information, pictorial information (for example, image and video information) has played and will continue to play an important role as communication media that are used by individuals using the public communication infrastructure.[1] With the progress in multimedia system capabilities, a wide range of convenient utilities, such as image processing software tools and database management systems, have been developed to provide support for such functions as picture composition, picture manipulation, picture storage, and database management. Through the use of these powerful tools, users of multimedia communications systems can manipulate and disseminate pictures easily and efficiently. However, these easy-to-use tools may also facilitate undesirable actions in the digital internet environment, such as illegal picture copy, illegal picture distribution, picture forgery, and fraud. Image security issues, such as copyright protection, authentication, covert communication, and access control, have become a notable problem for modern multimedia communication systems.[2]

Data encryption and data hiding are two methods that may be used to secure multimedia communications. In this study, we concentrate on methods that use data hiding approaches. Various data hiding technologies to prevent illegal exploitation of digital images have been reported.[3–6] Data hiding for images refers to the process of embedding application-specific secret messages into a host image in such a way that the requirements for securing the host image or the message hidden in the image are enforced. The level of protection that can be offered by a data hiding scheme depends on the messages that are embedded and which embedding strategies are employed for hiding the messages. For example, to provide proof of image ownership, a watermark is embedded within those components of an image that can withstand a substantial attempt to illegally modify the image data. In the case where it is desired to verify the integrity of the image content, a concise message related to the image content can be embedded into those components of the image that are most sensitive to changes to the image. In addition to applications that provide traditional multimedia security, various data hiding schemes have also been developed for other multimedia applications, e.g., monitoring video broadcast quality,[7] determining the image display quality,[8] and concealing patient records associated with medical images.[9] With traditional data hiding algorithms, an image always experiences an irreversible reduction in quality once a hidden message has been embedded. For some applications that require sensitive imaging systems, such as medical applications, fine artwork applications, and military applications, the ability to restore the original quality of an image after extracting the embedded messages is one of the most demanding requirements. For example, if a satellite image is modified only slightly, the battlefield commander may erroneously judge the situation according to the altered images. Data hiding satisfying the requirement of reversibility is referred to as reversible, or lossless, data hiding.[10]

The reversibility feature of lossless data hiding is appealing for applications that require the availability of an original image or would benefit from the availability of a higher

Received Sep. 17, 2010; accepted for publication Jun. 6, 2011; published online Nov. 21, 2011
1062-3701/2011/55(11)/050502/15/$20.00.

image quality after the embedded message is removed. In addition to use for sensitive imaging applications, lossless data hiding can also be used to associate two related data sets by uniting them into a single file or data packet in a way that does not require modification of the file format. The original data set can then be exactly recovered after the associated data set has been extracted. Examples of this type of application include image caption annotation, multilanguage subtitle embedding, and network traffic monitoring. Most lossless image data hiding schemes operate on the raw data format of images without use of compression. This method of operation constrains the scope of applications that may incorporate lossless data hiding, as most digital images captured with widely available digital cameras and scanners are stored using a standard compression format, e.g., the JPEG standard.[11] In daily life, JPEG pictures play an important role in visual communications between individuals. Thus, this article describes the design of a lossless data hiding scheme for applications that require messages to be hidden within digital images that have been compressed using the JPEG format. The length of additional message that can be hidden in an image is called the hiding capacity of the image. Without targeting a particular data hiding application, the proposed hiding scheme aims to create a larger lossless hiding capacity for JPEG pictures.

In general, image compression methods exploit statistical redundancy and/or human visual redundancy to accomplish a reduction in the amount of data that is used to represent the content of an image. Most of the currently used data hiding schemes exploit a redundancy between neighboring pixels to increase the capacity for hiding a message within the pixels of an image. As digital images in a compressed format usually have less redundancy available for data hiding than uncompressed images, embedding messages into compressed images is challenging. It is necessary to seek other types of redundancy that exist within the compressed images to enable an increase in the length of the hidden message. With the JPEG standard, the input image is sectioned into an array of nonoverlapping 8×8 blocks of image pixels, and the majority of the correlation or redundancy in each block is exploited to make a compact binary code word via a discrete cosine transform (DCT) quantization, run-length encoding, and a lossless coder. With the goal of embedding more data into JPEG-encoded pictures, this article describes a method to exploit the correlation between DCT coefficients across neighboring blocks and design an interpolation-based lossless data hiding scheme that is based on the interpolation difference values. Shifting the histogram of interpolation difference values performs the process of data embedding. The advantage of the proposed algorithm is that a picture with a hidden message remains compliant to the JPEG standard specification, i.e., a standard JPEG decoder is able to decode the content of the picture. The file format does not need to be revised to hide the other extra message. A revised JPEG decoder that is equipped with the appropriate lossless data extraction module can extract the hidden message and restore the original coded content of the picture without degradation in image quality. Furthermore, the proposed embedding process can be easily incorporated into the JPEG encoder and performs in real time without introducing an additional processing delay, making it suitable for implementation in JPEG hardware devices.

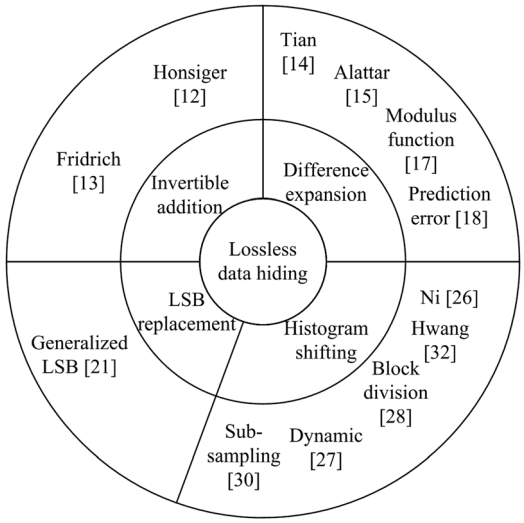

Figure 1. Classification of lossless data-hiding schemes.

BACKGROUND

In this section, background information is presented on lossless data hiding methods, and a concise explanation is given of the JPEG coding and decoding processes. First, a brief review of lossless data hiding schemes is presented, followed by a description of a baseline JPEG compression model. Then, we elaborate on our reasons for applying an interpolation-based strategy to lossless data hiding methods for JPEG-encoded pictures.

Lossless Data Hiding

Lossless data hiding schemes feature the ability to exactly recover the host image after the hidden message has been removed from the image. Due to the significant application in multimedia communications, a variety of lossless data hiding algorithms have been developed in recent years. In this subsection, we review and explain the basic strategies used by various types of lossless data hiding methods. Figure 1 shows a classification of the lossless hiding schemes according to the basic methods on which the lossless embedding strategies are derived. Selected references for each category are cited in this figure.

In the following paragraphs, the basic methods for the classification of lossless data hiding schemes are briefly explained.

(1) Invertible addition method:[12,13] For 8-bit images, the invertible addition method uses a modulo-256 addition operator to add the hash value of the original image to each pixel in the original image for the purpose of image authentication. Using modulo-256 addition, the sum of each pixel value and hash value remains an 8-bit integer; underflow and overflow do not occur. Since the modulo-256 addition operator

performs an invertible transform on any pair of 8-bit integers, the original image can be exactly recovered by subtracting the hash value from the marked image. Honsinger et al.[12] proposed the use of modulo-256 addition for lossless image authentication verification. Due to the possibility that pixels with values close to 255 or zero may flip when the message is added, the marked image may contain a distracting artifact termed salt-and-pepper noise. Fridrich et al.[13] proposed a modified form of modulo-256 addition with shorter cycles to minimize the visual artifact from the flipped pixels in the marked image. Invertible addition is a simple and effective approach to guarantee the exact recovery of the original image, but the message that is to be embedded in the marked image must be made available prior to the recovery process using the modulo-256 addition operator.

(2) Difference expansion method:[14–20] This method finds the average of and difference between the values of two neighboring pixels in an image and expands the difference by a factor of two to create a one-bit storage space in which a message bit may be inserted [i.e., the position of the least significant bit (LSB) in the expanded difference value]. The pair of values, consisting of the modified difference value and the average value, is transformed into a pixel pair such that the average value of the pixel pair is not altered. This type of difference expansion approach was proposed by Tian[14] for the purpose of lossless data embedding. Alattar[15] have extended Tian's idea in their formulation of a more general reversible transform involving multiple pixels, rather than pairs of pixels, for increasing the embedding rate. The main disadvantage of the difference expansion methods is that, to guarantee the exact recovery of the original image, a large binary location map is required to indicate where the message is embedded in the image. Thus, data compression techniques and many variants of the difference expansion technique[16–20] have been developed to mitigate this problem. If the pixel values in the image vary smoothly, the difference expansion method can provide good performance in terms of the trade-off between hiding capacity and distortion. Smaller difference values can be significantly expanded to provide more space for embedding message bits with a relatively lower level of visual distortion. With images containing higher levels of contrast, the difference expansion method becomes less feasible, as the trade-off between hiding capacity and distortion is less favorable.

(3) Generalized-LSB (Refs. 21 and 22): This type of method originates with the well known least significant bit replacement technique. In traditional LSB replacement methods, the lowest LSB bit plane is extracted and losslessly compressed to provide sufficient space for insertion of a message. Embedding is carried out by placing the payload, which is formed from the concatenation of the compressed bit stream and the bit stream of the message to be embedded, at the lowest bit plane of the original image. If additional hiding capacity is required, two or more LSB bit planes may be used for compression and overwriting. To perform extraction and recovery, the bits scanned from the selected LSB bit planes are used to form the embedded payload, which is then used to extract the hidden message and restore the original data bits from the overwritten bit planes. Celik et al. generalized the idea of LSB replacement for high-performance data hiding in terms of a capacity-distortion trade-off.[21] The generalized LSB (G-LSB) algorithm applies a prediction-based conditional entropy coder to the lower LSB bit planes to improve the data embedding capacity. Therefore, the performance of G-LSB algorithms in terms of generating additional hiding capacity depends on the amount of correlation available for data hiding in the image LSB bit planes and the complexity of the entropy coders.[23–25]

(4) Histogram-shifting method:[26–33] In this method, a histogram of the host image is used to guide the embedding and extraction processes. The algorithm first finds the pixel value with the highest occurrence (termed the maximum point) and then it finds a pixel value without occurrence (termed the zero point) in the image. The algorithm then shifts all pixel values that lie between the maximum and minimum points by subtracting or adding one, so that they are all closer to the minimum point. This action creates one space next to the maximum point of the histogram, in which the hidden message may be placed. Message bits are distributed so that they are embedded at all of the maximum points in the image. If the message bit is a zero, the pixel that corresponds to the maximum point is not modified; otherwise, the pixel value is increased or decreased by one and becomes the value associated with a newly created bin in the shifted histogram. Given knowledge of the maximum and zero points, the extraction of a hidden message can be accomplished by using the same technique as was used with the embedding process, and in this way the original image is recovered by shifting the histogram back to its original shape. Since Ni proposed this algorithm,[26] a variety of variants and improvements have been developed, particularly as this method is quite simple and does not require complicated computations and compression techniques.[27–33] The marked image has lower degradation in quality after the message has been hidden. Therefore, in this article, we adopt histogram shifting as the embedding process for the DCT coefficients in the JPEG pictures. Before describing the proposed data hiding scheme, we first review the encoding and decoding processes specified in the JEPG standard.

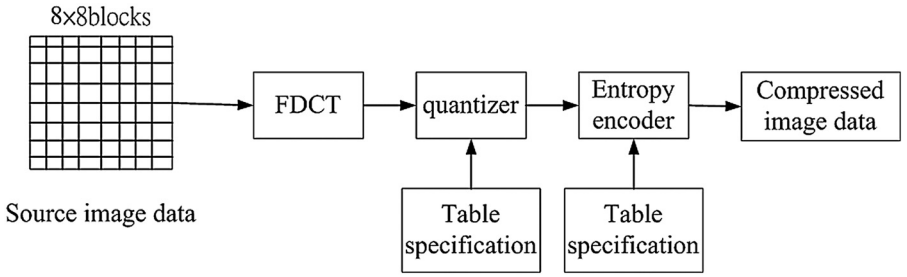

Figure 2. JPEG baseline encoding process.

$$F(u,v)=\frac{1}{4}C(u)C(v)\left[\sum_{x=0}^{7}\sum_{y=0}^{7}f(x,y)\times\cos\frac{(2x+1)u\pi}{16}\cos\frac{(2y+1)v\pi}{16}\right],\quad 0\le u,v\le 7$$
$$\text{where } C(i)=\begin{cases}\frac{1}{\sqrt{2}} & \text{if } i=0\\ 1 & \text{otherwise}\end{cases} \tag{1}$$

JPEG Standard Encoder and Decoder

The joint ISO/CCITT committee established the first international compression standard for continuous-tone still images. This standard, known as JPEG, is applicable to both grayscale and color images. Recently, the JPEG standard has become a widely used file format for encoding images. Figure 2 shows the encoding process of the JPEG baseline mode.

In the encoding procedure, the input image is divided into an array of 8×8 blocks, and the brightness of each pixel in a block is then shifted from the unsigned integer range $[0, 2^{P}\text{-}1]$ to the signed integer range $[-2^{P-1}, 2^{P-1}\text{-}1]$ where P is the number of bits representing each pixel in the image. For example, for 8-bit grayscale images, 128 is subtracted from each of the pixels in each of the 8×8 blocks to transform the grayscale range, i.e., [0, 255], to a signed integer range, i.e., [−128, 127]. Then, each shifted 8×8 block is transformed by the forward discrete cosine transform (FDCT) from a spatial domain representation, $f(x, y)$, to a frequency domain representation, $F(u, v)$. Equation (1) shows the formula that is specified in the standard to perform the 8×8 FDCT, where (u, v) is a coordinate in the frequency domain and (x, y) is a coordinate in the spatial domain.

The coefficient $F(0, 0)$, having a zero frequency value in both dimensions u and v, is termed the DC coefficient, while the remaining 63 coefficients are termed AC coefficients. Because the DC coefficient represents the average intensity of the block, the JPEG standard specifies different coding strategies for the DC and AC coefficients. Prior to entropy coding, coefficients within the transformed block are uniformly quantized according to the quantization steps that are specified in the quantization table. Quantization is accomplished by dividing each DCT coefficient by its corresponding quantizer step and then rounding the result. In the entropy encoding step, the DC coefficient and the AC coefficients are compressed using different methods. As shown in Figure 3, a differential pulse coded modulation scheme is employed to compress the DC coefficients. As the adjacent DC coefficients have a strong correlation, the difference value between them is usually small, and may be zero, a fact that leads to reduced entropy of the symbols needed for Huffman coding. Figure 4 demonstrates the prediction between DC coefficients across horizontally neighboring blocks.

Fig. 3 also shows the encoding flow for the AC coefficients. All of the quantized AC coefficients are ordered into a 1-D sequence in accordance with the zig-zag order shown in Figure 5. The DC coefficient indicated in gray in the figure is excluded from the zig-zag AC sequence. In most cases, this AC coefficient sequence contains runs of zeros interspersed between several nonzero coefficients. To capture this particular type of data distribution, run-length encoding (RLE), as shown in Fig. 3, is employed to generate suitable symbols in the form of (runs, nonzero coefficient) that are suitable for AC Huffman coding. Finally, the JPEG bit stream is constructed from the DC- and AC-coded bit streams.

The decoding process is the inverse process of the encoding process. As shown in Figure 6, the compressed image data is first passed through the Huffman decoder to restore the DC coefficients and AC coefficients and then restored to an 8×8 block in the frequency domain through the use of inverse zig-zag scanning. The decoded 8×8 block is then normalized by the dequantization step, which multiplies the decoded 8×8 block by the quantization table. Finally, the JPEG standard utilizes the inverse DCT (IDCT)

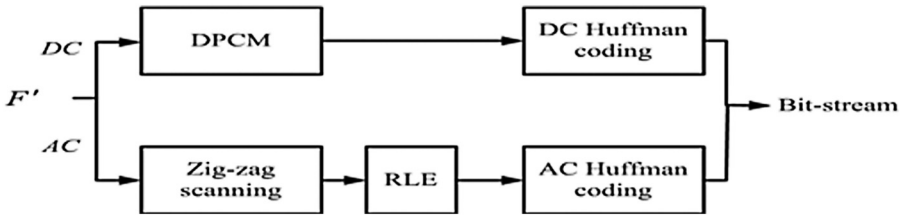

Figure 3. Flowchart of entropy coding.

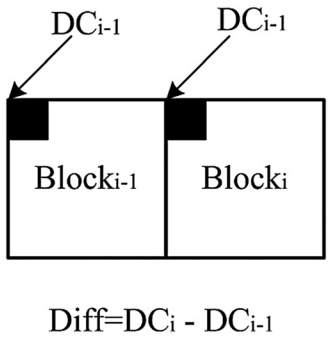

Figure 4. Preparation of DC coefficients for entropy coding.

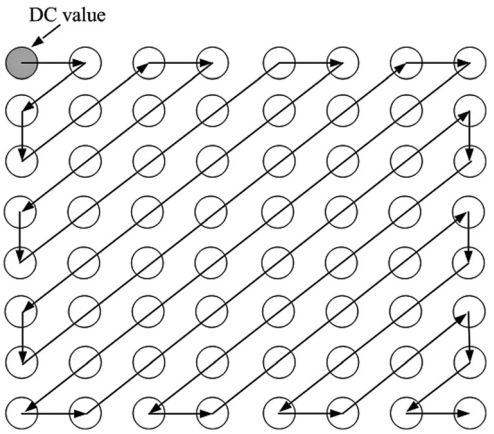

Figure 5. One-dimensional zig-zag scanning order.

to transform the block into the spatial domain $f(x, y)$. The equation for the IDCT is given by

$$f(x,y) = \frac{1}{4}\left[\sum_{u=0}^{7}\sum_{v=0}^{7} C(u)C(v)F(u,v)\cos\frac{(2x+1)u\pi}{16}\cos\frac{(2y+1)v\pi}{16}\right], 0 \le x, y \le 7$$
$$\text{where } C(i) = \begin{cases} \frac{1}{\sqrt{2}} & if\, i = 0 \\ 1 & otherwise \end{cases} . \quad (2)$$

Interpolation-Based Lossless Data Hiding

In Ref. 34, Lin et al. proposed an interpolation-based lossless data hiding method to improve the performance of a histogram-shifting hiding scheme in terms of marked image quality and hiding capacity. The proposed method functions in the spatial domain and embeds an additional message through a controlled shift of the histogram of interpolation difference values. The interpolation differences are computed between spatially subsampled pixel intensities and their interpolated counterparts. As compared to other methods, this method better utilizes the correlation between nearby pixels in an image (via simple interpolation techniques) to increase the hiding capacity for data hiding without creating significant distortion. This interpolation technique is used in this study to exploit the correlation between DCT coefficients across neighboring transformed blocks.

We now discuss how lossless data hiding methods are incorporated into JPEG-encoded pictures. As stated previously, an interpolation-based data hiding method can exploit the correlation that exists between neighboring image pixels to increase the amount of message that may be embedded within a host image. In this article, the concept of exploiting the correlation of DCT coefficients across neighboring DCT blocks is also employed for reversible hiding of data in JPEG-encoded pictures. Due to the irreversibility of the coefficient quantization process (refer to Fig. 2), the content of the host data that are used to hide the message information depends on the zig-zag scanned quantized sequence of DCT AC coefficients. Hiding data in the quantized coefficients prevents destruction of the hidden data by the quantization process. Given the continuity of pixel intensity variations in most natural images and the compactness effect of the quantized coefficients, a significant correlation between or redundancy in the quantized DCT coefficients between neighboring blocks would be expected in the lossy encoding of JPEG-coded images. For example, Figure 7 shows a set of the

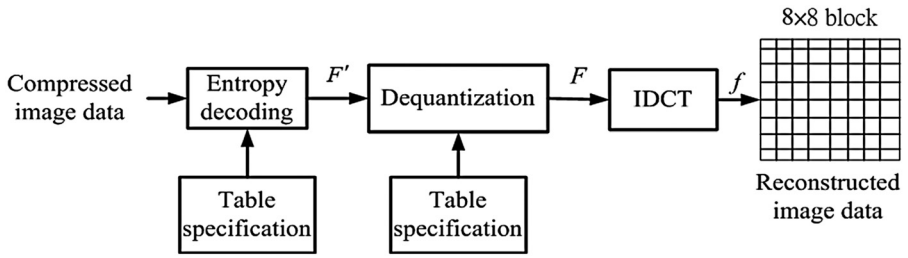

Figure 6. JPEG baseline decoding process.

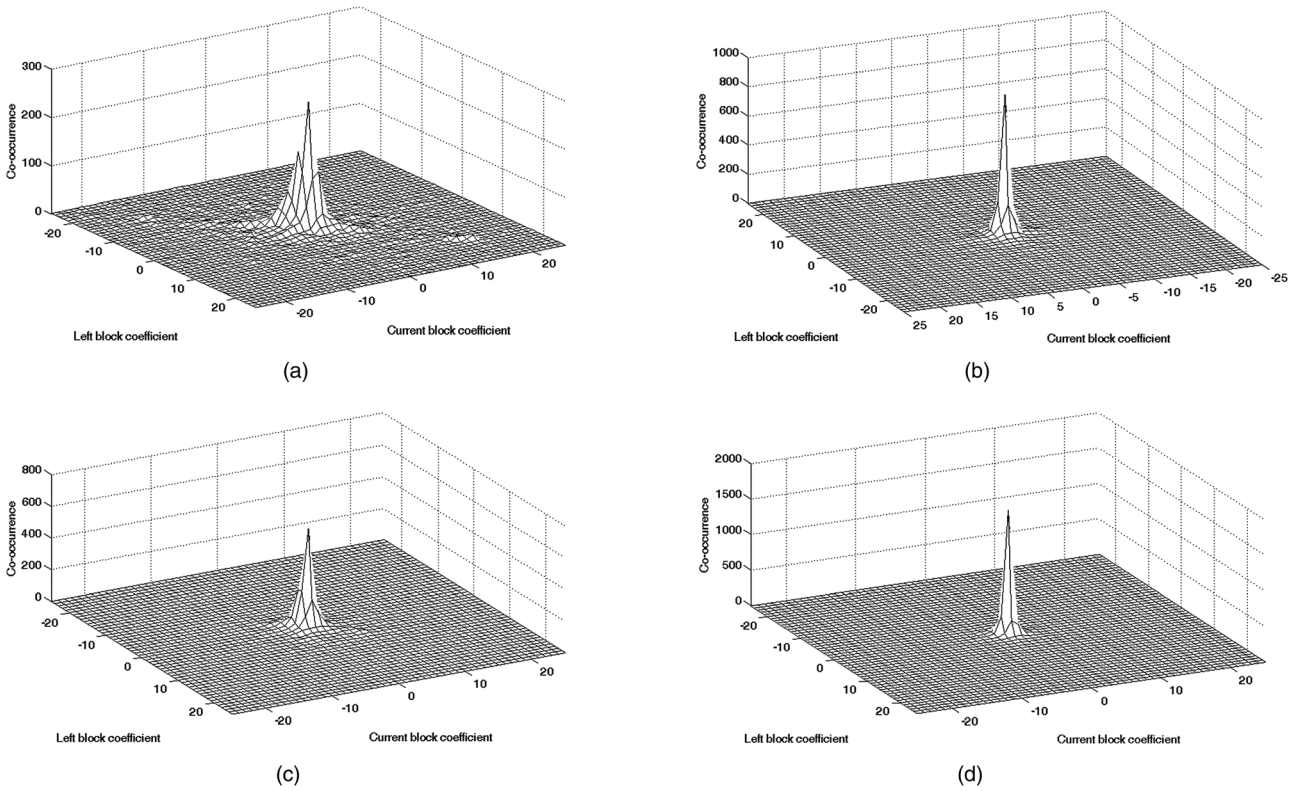

Figure 7. Examples of co-occurrence matrices between neighboring co-located AC coefficients.

co-occurrence frequencies of the quantized coefficients for a block together with colocated coefficients in the left neighboring block. In this example, the frequencies were obtained by using the test image *Lena* encoded with a quality factor (QF) of 60. It is clear from Fig. 7(a) that the value of the first quantized AC coefficient in the current block's zig-zag scanned sequence has a linear correlation with that of the colocated coefficient in the block to the left. Figs. 7(b)–7(d) demonstrate that the values of the other coefficients, for example, the 4th, 5th, and 6th AC coefficients in the current block, decrease with an increase in positional index (i.e., spatial frequency) and always co-occur with their colocated coefficients near the zero value. Our development of a lossless data hiding scheme for images encoded using the JPEG standard, as presented in the Proposed Methods Section, will rely on this phenomenon.

PROPOSED METHODS

In this section, we present the method for incorporating lossless data hiding into the JPEG encoder/decoder. First, the architecture of the new JPEG coding framework that is equipped with the lossless data hiding capability is explained. Then, the data embedding algorithm and the corresponding extraction procedure that are proposed for the new framework are presented.

Incorporating the Data Hiding Process into the JPEG Standard

Figure 8 shows the proposed JPEG encoder/decoder architecture that incorporates the lossless data hiding scheme. The lossless data hiding scheme consists of two parts, a data embedding process and a data extracting process, that are located within the encoder and decoder sections of the algorithm, respectively. The JPEG encoder

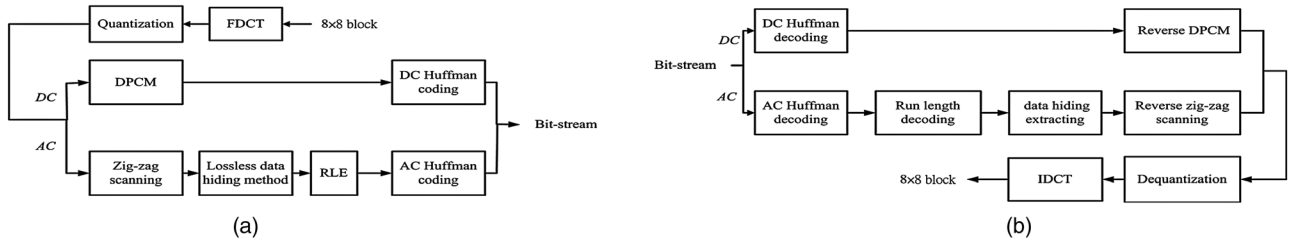

Figure 8. Lossless data-hiding scheme in the JPEG standard codec. (a) Data embedding process in the encoder; (b) Data extraction process in the decoder.

shown in Fig. 8(a) controls the data embedding process, and the decoder shown in Fig. 8(b) controls the data extracting process. To comply with the JPEG coding structure, the proposed lossless data hiding scheme operates on 8×8 blocks of pixels and hides data in the quantized DCT AC coefficients of each 8×8 block. Each block is reordered in the form of a 1-D zig-zag sequence. To avoid significant or apparent degradation of the perceptible quality in the marked JPEG pictures, the hiding process excludes the DC coefficient in each 8×8 block. Each marked sequence of DCT coefficients is then encoded into a variable-length bit stream via RLE and the JPEG Huffman encoding.

Since the output bit stream fully satisfies the JPEG standard, any picture viewer or decoder that conforms to the JPEG standard would be able to decode the marked bit stream and display the decoded host picture. To extract the data that is hidden in the JPEG picture, a decoder that implements the proposed data extraction procedure should be used to decode the marked bit stream for the purpose of extracting the message data. This process will restore the host DCT coefficients and allow display of the original encoded picture.

Embedding Algorithm

The embedding process involves a slight modification of the values of certain nonzero quantized coefficients in an 8×8 block for the purpose of hiding the bits of message data. To have a manageable degradation of the marked picture quality, a reversible method is employed in which a histogram of the interpolation differences of the AC coefficients is shifted. The proposed algorithm adopts a simple linear interpolation, such as the interpolation used in Ref. 34, to make interpolation estimates of nonzero AC coefficients. These estimates are made through the use of the colocated coefficients across neighboring coded blocks. For each nonzero quantized AC coefficient (termed a target coefficient) in the current block, the two reference coefficients colocated in those neighboring blocks that are to the left of and above the current block are used to make interpolation estimates of the target coefficient (Figure 9). Let c^l and c^u represent the left and upper colocated coefficients, respectively, of a target coefficient, c^t. The linear interpolation estimate, c^r, of c^t is computed as

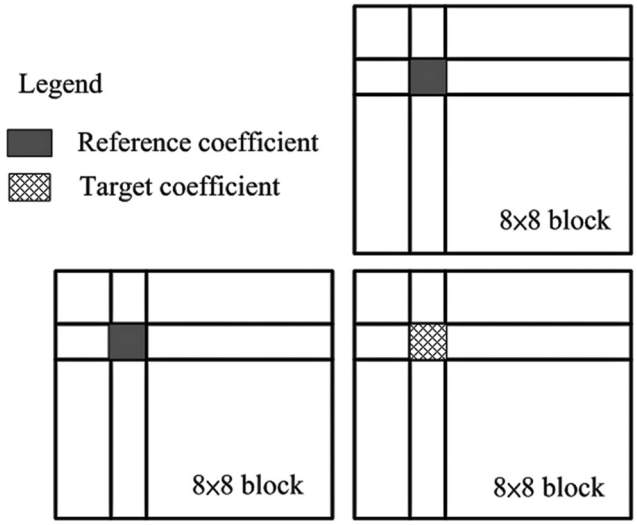

Figure 9. Co-located reference coefficients with respect to the target.

$$c^r = \frac{c^l + c^u}{2}. \tag{3}$$

Then, the interpolation error or difference is defined as

$$d = c^t - c^r. \tag{4}$$

The histogram $H(d)$ of the interpolation difference values for all of the target coefficients is shifted to embed the message data. If $H(d)$ is mostly distributed near zero, the hiding capacity and perceptible quality are maintained at an acceptable level. Figure 10 shows the flowchart of the proposed embedding algorithm. The symbols that appear in Fig. 10 are defined in Table I.

As described in Fig. 10, for each quantized AC block, the embedding algorithm first checks to see if the number of nonzero coefficients in the block is larger than 1; if this is the case, then the nonzero coefficients in the block and their colocated coefficients in the left and upper neighboring blocks are selected for calculation of the interpolation differences that are used to generate the histogram. The shifting operation that is performed on the histogram of interpolation differences to embed message bits can be performed by an equivalent modification of the values of the target coefficients. This procedure is repeated for each input block until the last block in the input picture has

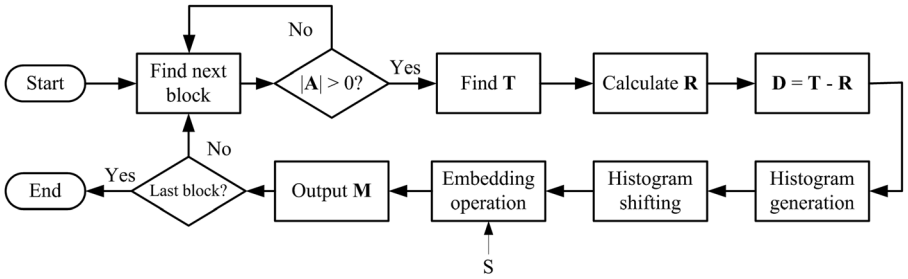

Figure 10. Flowchart of the embedding process.

Table I. Definition of symbols used in the embedding flowchart.

Symbols	Description
A	$\mathbf{A} = (c_1, c_2, \ldots, c_{63})$. A is the zig-zag-scanned sequence of DCT AC coefficients in the current 8×8 block. If all of the coefficients in sequence **A** are zeros, the sequence is represented with one single "EOB" symbol.
T	$\mathbf{T} = (c_{i_1}, c_{i_2}, \ldots, c_{i_k})$, where $c_{i_j} \in$ A and $c_{i_j} \neq 0, \forall j \leq k$. This is the subsequence of **A**, consisting of all of the nonzero AC coefficients in **A**. Each coefficient in this sequence is treated as a target coefficient for embedding data.
R	$\mathbf{R} = (c^r_{i_1}, c^r_{i_2}, \ldots, c^r_{i_k})$. **R** contains the interpolation estimates of all of the coefficients in sequence **T**. Each estimate is obtained via (3).
D	$\mathbf{D} = (d_1, d_2, \ldots, d_k)$, where $d_j = c_{i_j} - c^r_{i_j}, 1 \leq j \leq k$. **D** is the ordered set of interpolation differences between each pair of two members, in which one member is from **R** and the other is from **T**.
S	The set of binary message data bits.
M	$\mathbf{M} = (c'_{i_1}, c'_{i_2}, \ldots, c'_{i_k})$. **M** represents the set of marked nonzero AC coefficients that are modified by the histogram-shifting operation in the embedding algorithm for the purpose of hiding message bits in the target sequence **T**.

been processed. The embedding process is not applied to those blocks located in the first row and the first column of the 8×8 block in the picture, as only one neighboring coded block is present. The following steps give a detailed explanation of the method used to embed message bits into the sequence **A.**

Step 1: Select the nonzero coefficients of **A** to form the subsequence **T** of target coefficients.

Step 2: Compute the interpolation estimates of each element in **T** to form the sequence **R** via (3).

Step 3: Compute the sequence **D** of interpolation differences between **T** and **R** via (4).

Step 4: Generate the histogram of interpolation differences, $H(d)$, from **D**.

Step 5: Shift $H(d)$ by modifying the elements c_{i_j} in **T** according to the following formula:

$$c'_{i_j} = \begin{cases} c_{i_j} + L + 1 & \text{if } d_j > L \text{ and } c_{i_j} > 0 \\ c_{i_j} - L & \text{if } d_j < -L \text{ and } c_{i_j} < 0 \\ c_{i_j} & \text{otherwise} \end{cases}, 1 \leq j \leq |\mathbf{T}|, L \geq 0,$$

where L stands for the embedding level, which is a user-specified parameter that controls the marked picture quality and the capacity created for the message.

Step 6: Embed one-bit, $s \in S$, within each element c_{i_j} in **T** if d_j meets the following condition:

$$c'_{i_j} = \begin{cases} c_{i_j} + d_j + s & -L \leq d_j \leq L \text{ and } d_j \times c_{i_j} \geq 0 \\ c_{i_j} & \text{otherwise} \end{cases}.$$

Step 7: Output the marked sequence $\mathbf{M} = (c'_{i_1}, c'_{i_2}, \ldots, c'_{i_{|\mathbf{T}|}})$.

To avoid zeroing the nonzero coefficients, which would reduce the number of nonzero coefficients after the data is embedded and thus lead to unsynchronized data extraction in the decoder side, d_j and c_{i_j} must both be either positive or negative when c_{i_j} is modified, as is done in Steps 5 and 6. The capacity provided by an 8×8 block can be measured as the number of binary bits that have been embedded in the block; capacity is then calculated as

$$\text{Capacity} = \sum_{d=-L}^{d} H(d), \text{in units of bits per block (bpb)}. \tag{5}$$

Extracting procedure

The proposed extracting procedure is simply the reverse of the embedding process. As shown in Figure 11, the extracting procedure first examines each marked sequence **M** to select nonzero coefficients c'_{i_j} to compose the subsequence $\mathbf{T}'$ of **M**. Each element of $\mathbf{T}'$ is then interpolated via (3) to generate the corresponding sequence **R** of interpolation estimates. Once both $\mathbf{T}'$ and **R** are known, the sequence of interpolation differences can be calculated as $\mathbf{D}' = \mathbf{T}' - \mathbf{R}$. Because the embedding algorithm maintains the number of nonzero coefficients, the extracting procedure is able to

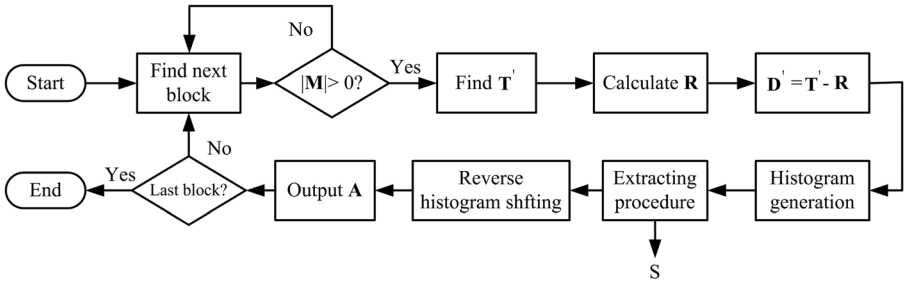

Figure 11. Flowchart of the data extracting process.

determine which of the AC coefficients are target coefficients that were modified by the embedding algorithm by determining whether or not each coefficient is nonzero. To extract the message data bits and restore **M** to the sequence given by **A**, the histogram based on **D**′ is generated and then shifted in the reverse direction by modifying each element in **T**′. As the extracting algorithm is similar to the embedding algorithm, this algorithm is not applied to those blocks that are located in the first row and the first column of 8×8 block in the picture, and the algorithm is applied to each of the remaining blocks through the last block in the picture. The following steps give a detailed explanation of the methods used for the extracting and restoring operations.

Step 1: Select the nonzero coefficients from **M** to form the sequence **T**′ of nonzero target coefficients.

Step 2: Compute the interpolation estimates of each element in **T**′ to form the sequence **R** via (3).

Step 3: Compute the sequence **D**′ of interpolation differences between **T**′ and **R** via (4).

Step 4: Generate the histogram of interpolation differences, $H(d')$, from **D**′.

Step 5: Extract the message bits, s, that may be hidden in the elements c'_{i_j} in **T**′ if c'_{i_j} satisfies the following condition:

$$s = \mathrm{mod}(c'_{i_j}, 2), \text{if } -2L \le c'_{i_j} \le 2L+1, 1 \le j \le \left|\mathbf{T}'\right|.$$

Step 6: Restore the original nonzero target coefficients, c_{i_j}, by reversely shifting $H(d')$ according to the following formula:

$$c_{i_j} = \begin{cases} c'_{i_j} - L - 1 & \text{if } d'_j > 2L+1 \text{ and } c'_{i_j} > 0 \\ c'_{i_j} + L & \text{if } d'_j < -2L \text{ and } c'_{i_j} < 0 \\ c'_{i_j} - \left\lfloor \frac{d'_j+1}{2} \right\rfloor & \text{if } -2L \le d'_j \le 2L+1 \text{ and } c'_{i_j} \times d'_j > 0 \\ c'_{i_j} & \text{otherwise} \end{cases}.$$

Step 7: Obtain the restored sequence,$\mathbf{T} = (c_{i_1}, c_{i_2}, \ldots, c_{i_{|\mathbf{T}'|}})$, and output the sequence **A** as a sequence consisting of zeros, with the nonzero coefficients of **T** replacing those sequence items of **A** with corresponding indices.

EXPERIMENTAL RESULTS

To verify the performance of the proposed lossless data hiding scheme, computer experiments were conducted to measure the hiding capacity and quality degradation for a set of test images. An image analysis application that performs face detection was utilized in the experiments to examine the impact of the proposed hiding scheme on face detection accuracy. The measure of quality that was used for our experiments is peak signal-to-noise-ratio (PSNR), defined as follows:

$$PSNR = 10 \times \log_{10}\left(\frac{65025}{E[(I - I')^2]}\right) \text{dB}, \tag{6}$$

where I and I' represent the original and marked images, respectively. $E[\cdot]$ stands for the expectation or average operator on image data sets. The quality assessment also considers the structural similarity index (SSIM) that is often used to evaluate the distortion of local patterns of pixel intensities.[35] As the proposed algorithm operates at the level of 8×8 blocks, the hiding capacities observed in the experiments were measured in terms of bits per 8×8 block, according to the formula defined in Eq. (5).

Experiment for Hiding Capacities and Marked Qualities

The proposed lossless data hiding scheme was used on the set of four natural grayscale images shown in Figure 12 to determine the performance in terms of hiding capacities and marked image qualities. Each test image was first compressed using the standard JPEG encoder, and then the data embedding operation was performed using the proposed lossless data hiding scheme. In these experiments, the quality factor QF and the embedding level L were varied to determine the variation in performance of the proposed scheme. The messages that were embedded within the JPEG-encoded images were composed of pseudorandomly generated bit streams. Table II shows the results of the experiments. By examining each row in Table II, it is clear that the hiding capacities decrease gradually as a function of the quality factor. Choosing a larger QF results in better retention of quality in the coded image. For the example of hiding a message in the *Lena* image with $L = 0$, the hiding capacity declines to 0.39 bits-per-block (bpb) when the QF is decreased from 70 to 30. This result occurs because the larger quantization step zeroes most of the AC coefficients

Figure 12. Test images. (a) Lena; (b) Baboon; (c) F16; (d) Goldhill.

Table II. Hiding capacities (in bpb) and coding bit rates (in bpp).

		QF = 70		QF = 60		QF = 50		QF = 40		QF = 30	
Image	*L*	Capacity	Bitrate	Capacity	Bitrate	Capacity	Bitrate	Capacity	Bitrate	Capacity	Bitrate
Lena	0	0.69	0.96	0.58	0.80	0.52	0.70	0.42	0.61	0.39	0.55
	1	2.83	1.03	2.38	0.86	2.12	0.75	1.80	0.65	1.64	0.59
	2	4.30	1.07	3.61	0.88	3.21	0.77	2.72	0.67	2.48	0.61
Baboon	0	1.59	2.24	1.40	1.90	1.31	1.66	1.18	1.47	1.11	1.34
	1	6.24	2.39	5.58	2.03	5.13	1.80	4.67	1.60	4.40	1.45
	2	9.93	2.37	8.91	2.02	8.20	1.78	7.49	1.59	7.05	1.45
F16	0	0.68	1.06	0.58	0.89	0.54	0.79	0.46	0.70	0.42	0.64
	1	2.81	1.13	2.44	0.95	2.23	0.84	1.95	0.74	1.82	0.68
	2	4.38	1.17	3.81	0.98	3.49	0.87	3.04	0.77	2.82	0.70
Goldhill	0	1.05	1.26	0.94	1.06	0.87	0.93	0.74	0.81	0.70	0.73
	1	4.00	1.36	3.60	1.14	3.32	1.00	2.94	0.87	2.74	0.79
	2	6.30	1.37	5.63	1.17	5.17	1.03	4.52	0.90	4.17	0.81

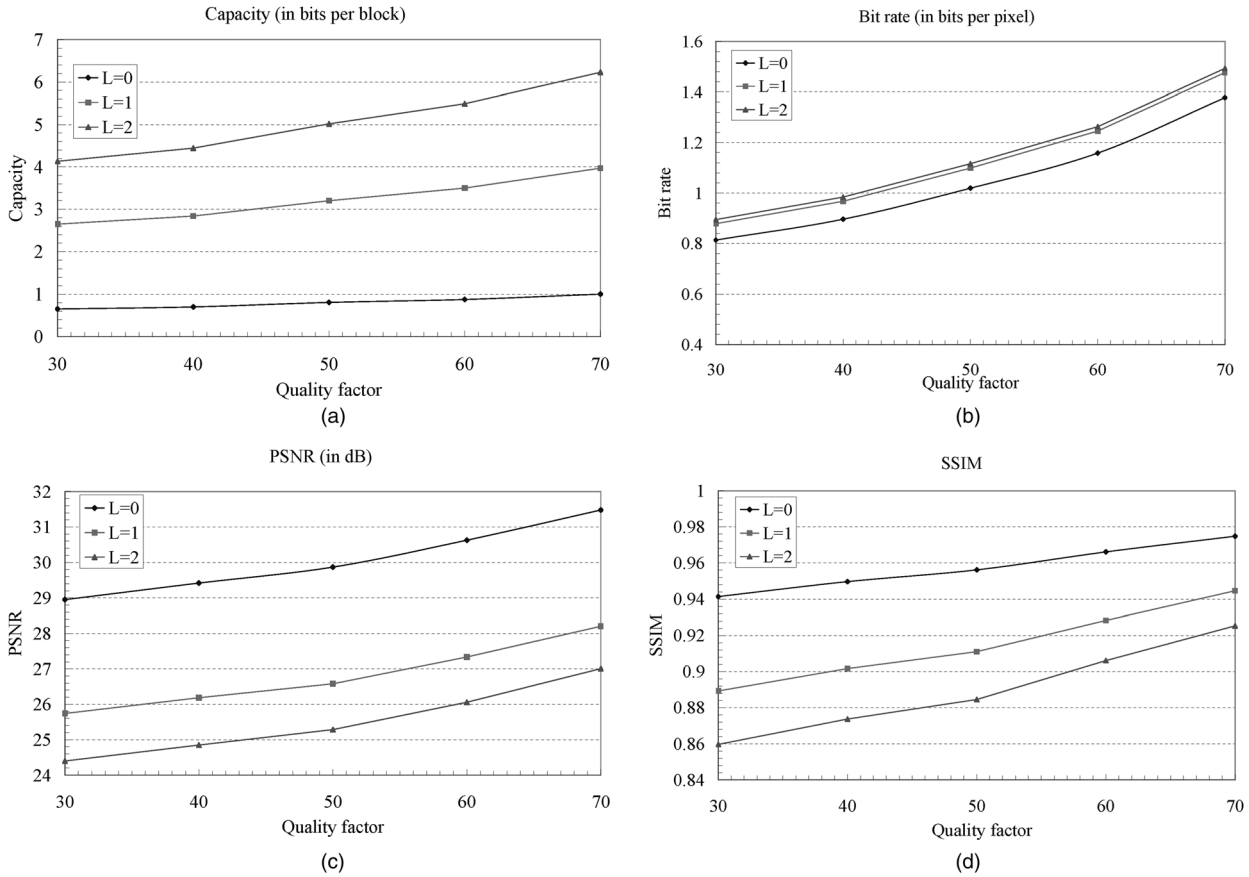

Figure 13. Average performance measurements for the test images. (a) Hiding capacities; (b) JPEG coded bit rates; (c) PSNRs of decoded images; (d) SSIM indices of decoded images.

in a block, and thus only a few of the nonzero quantization coefficients are available for the embedding process. As an illustration of the dependence on image complexity, the hiding capacities shown in Table II for the *Baboon* and *Goldhill* images are much larger than those of other images that are smoother. In particular, the *Goldhill* image has an almost 80% greater hiding capacity than the *Lena* image, when the QF is decreased from 70 to 30 and data is embedded with $L = 0$. In a more extreme case, the *Baboon* image has a greater than 180% increase in hiding capacity under the same conditions, as compared with the *Lena* image. If a higher capacity is preferred, the parameter L may be adjusted to provide the desired capacity. In Table II, it can be seen that all of the test images exhibit significant increases in capacity when L is set to 1 or 2. Referring to (5), when L is increased, there are additional candidate coefficients in each block that may be used for hiding data. However, increases in capacity may introduce corresponding increases in coded bit rate. For example, by examining the "QF = 70" column of Table II, it is clear that the bit rate of the *Lena* image increases from 0.96 to 1.07 bits per pixel (bpp), while its capacity increases from 0.69 to 4.3 bpb when L is increased to 2. This increase of bit rate, which leads to a larger file size, is acceptable, as the proposed method increases the magnitudes of each of the nonzero coefficients when embedding data, resulting in a somewhat longer AC Huffman code. Figures 13(a) and 13(b) summarize the average capacities and bit rates for the four test images.

In these experiments, we also decoded the JPEG-coded images (each of which had been embedded with a hidden message) using a standard JPEG decoder. We then observed the degradation in the quality of the decoded image that occurred for various coding and embedding parameter settings. Table III lists the outcome in terms of the PSNR and the SSIM index. Embedding data in the coded image is likely to cause degradation of the decoded quality of the marked images, and this degradation becomes more significant as L is increased. As can be seen in Table II, the degradation is at most 5 dB of the PSNR for all test images when the value of L is increased to 2. This degradation in quality was found to have no significant dependence on the variation of the quality factor. The SSIM indices listed Table III reveal that the structural distortion of the marked image content is limited, while the embedding distortion in the PSNR is significant. In other words, the local pattern of pixel intensities in the marked image is on average similar to that in the original image, even though the objective distortion is noticeable. This result implies that the marked images are useful for most applications that perform image analysis or recognition of structural content within images. Figs. 13(c) and 13(d) show the average PSNRs and SSIM indices for the four test images.

Table III. Decoded qualities of marked images in terms of PSNR and SSIM index.

		QF = 70		QF = 60		QF = 50		QF = 40		QF = 30	
Image	*L*	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
Lena	0	34.4	0.975	33.5	0.967	32.8	0.958	32.4	0.953	31.9	0.946
	1	31.1	0.947	30.3	0.934	29.5	0.919	29.2	0.914	28.6	0.904
	2	29.6	0.926	28.7	0.910	27.9	0.892	27.5	0.886	27.0	0.874
Baboon	0	25.7	0.976	24.7	0.966	24.0	0.955	23.5	0.946	23.2	0.937
	1	22.4	0.943	21.4	0.923	20.7	0.902	20.3	0.887	20.0	0.871
	2	21.7	0.928	20.8	0.905	20.2	0.882	19.8	0.865	19.4	0.848
F16	0	33.8	0.977	33.0	0.970	32.1	0.961	21.5	0.957	30.9	0.950
	1	30.4	0.950	29.5	0.938	28.6	0.924	28.0	0.919	27.5	0.910
	2	28.7	0.930	27.7	0.916	26.9	0.899	26.3	0.896	25.7	0.885
Goldhill	0	32.1	0.972	31.3	0.962	30.6	0.950	30.3	0.942	29.8	0.932
	1	28.9	0.938	28.2	0.919	27.5	0.899	27.2	0.887	26.8	0.872
	2	28.0	0.918	27.0	0.894	26.2	0.865	25.8	0.848	25.4	0.832

Table IV. Decoded image PSNR and SSIM after removal of the message from the marked images.

		QF = 70		QF = 60		QF = 50		QF = 40		QF = 30	
Image	*L*	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
Lena	0,1,2	37.3	0.989	36.5	0.985	35.8	0.981	35.3	0.977	34.8	0.973
Baboon	0,1,2	29.8	0.990	28.4	0.986	27.5	0.982	26.8	0.977	26.2	0.972
F16	0,1,2	37.1	0.990	36.1	0.986	35.4	0.982	34.7	0.978	34.1	0.974
Goldhill	0,1,2	35.2	0.989	34.2	0.984	33.6	0.980	33.0	0.975	32.6	0.970

To test the extraction and restoration processes, a nonstandard JPEG decoder was implemented by modifying the standard JPEG decoder to incorporate the proposed data extraction and recovery procedure. This nonstandard JPEG decoder was used with all of the marked JPEG images to extract the hidden messages and to restore each of them to a standard decoded JPEG format. Table IV lists the results, in terms of PSNR and SSIM, for the images with various embedded message sizes. The associated parameter settings are also shown. These results verify that each of the decoded images is identical to the original JPEG-decoded version of the image after removal of the hidden message.

Influence on the Accuracy of Object Recognition

Analyzing an image to detect the position and size of human faces that are rendered in the image is an interesting and currently popular topic in the field of automatic object recognition. Representing a facial image using the JPEG format is common practice for many face detection applications. Therefore, in this section of our experiments, we investigate the influence of the proposed lossless data hiding scheme on the accuracy of face recognition. These experiments involve the detection of faces that are present within the marked JPEG images that have been embedded with additional messages. We have implemented a face detection system that is based on the library of detection functions in Ref. 36. Faces that are detected by this system are indicated by squares that bound each area in the image containing a face. The detected square is represented as a triplet of integers, (x, y, z), where the integers x and y stand for the center point position of the square (in pixels) in the image coordinate system, and the integer z is the width or length of the square (in pixels). Figure 14 illustrates one example of face detection; here, the face is indicated by the white bounding square with $(x, y, z) = (112, 130, 66)$.

The face detection system was used with a set of thirty images containing faces that were chosen from the Yale Face Database B.[37] This database includes the faces of ten unique individuals who were photographed in various poses; these images were used to test for the presence of detection distortion that may be caused by the data hiding operation. In this experiment, each face image is first encoded in a JPEG format with QF = 60, and then a random bit stream is hidden within the image using the

Figure 14. Illustration of face detection.

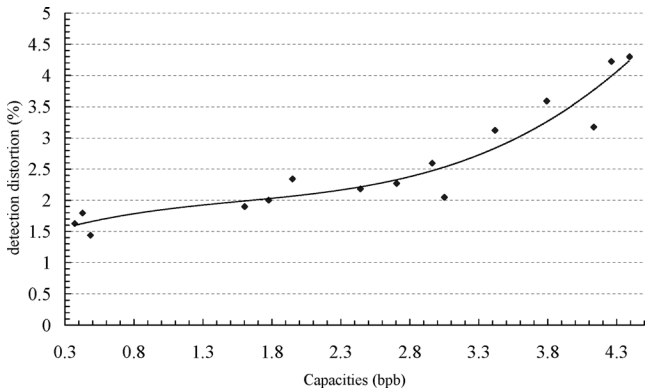

Figure 15. Influence of capacity on the detection accuracy.

Figure 16. Samples of the results of the face detection process for JPEG images without (upper) and with (bottom) the embedding of hidden data. (a) capacity=0.28 bpb; d_3= 1.13%; (b) capacity=3.50 bpb; d_4= 2.89%; (c) capacity=3.96 bpb; d_{29}= 7.74%.

proposed data scheme. By varying the embedding level L and the position of the starting coefficient that is hidden in each block, bit streams of various sizes can be hidden within the JPEG images. For the ith face image, F_i, in the image set, let F_i^J and F_i^M stand for the JPEG-coded and the marked JPEG versions of the images, respectively. Then, the distortion between the faces that was detected within F_i^J and F_i^M is defined as:

$$d_i = \frac{100}{3 \times z_i^J}\left(\left|x_i^J - x_i^M\right| + \left|y_i^J - y_i^M\right| + \left|z_i^J - z_i^M\right|\right)\%, \quad (7)$$

where (x_i^J, y_i^J, z_i^J) and (x_i^M, y_i^M, z_i^M) denote the detected areas of the squares that bound the faces from the JPEG image and the marked JPEG image, respectively. Distortion in the face detection process is primarily measured as the average differences between the positions and the widths of the two detected squares and is expressed as a percentage of z_i^J. To improve reliability, all of the distortion values obtained for the thirty images were averaged. To examine how the detection accuracy is affected by the data embedding process, we adjusted the appropriate parameter setting in the data hiding scheme to vary the hiding capacity and then observed the resulting change in detection distortion. Figure 15 shows the results of this experiment.

The results shown in Fig. 15 demonstrate that the distortion in detection accuracy increases with hiding capacity. A rapid increase in distortion is observed when the hiding capacity is increased to greater than 3 bpb, and the rate

continues to increase at higher capacities. Figure 16 shows three pairs of images that illustrate the differences observed between face detection performance, both with and without the use of the data embedding process. The images shown in the upper row of Fig. 16 are the JPEG images, and the images in the bottom row are the corresponding marked JPEG images. The facial detection results are indicated by white squares in the images. It is clear that those images that had been embedded with a significant amount of data exhibit a perceptible distortion in the size and position of the detected faces. For example, the two images shown in Figs. 16(b) and 16(c) exhibit significant distortion, while the image shown in Fig. 16(a), which contains a smaller amount of hidden data, exhibits no noticeable distortion in detection accuracy. Although there is a trade-off between the detection accuracy and the hiding capacity, the proposed lossless data hiding scheme is capable of offering sufficient hiding capacity for most multimedia applications without a significant sacrifice in the accuracy of analyzing the visual content in the marked media.

CONCLUSIONS

This article addresses a method for hiding additional message data into images that have been encoded using the JPEG format. The proposed method adopts an interpolation technique to exploit the correlation between quantized DCT coefficients across neighboring 8×8 blocks within the image and mitigates the degradation of the JPEG coding performance that may occur when the hiding capacity is large. Although the marked images may experience some degradation of the objective quality, the original quality of the host JPEG image may be exactly recovered once the message hidden in the image has been extracted. Our results show that the proposed lossless data hiding scheme not only guarantees the exact recovery of the host JPEG image but also maintains an acceptable quality level in the marked images. In summary, the proposed lossless data hiding scheme that is described in this article possesses the following advantageous properties:

The marked images are fully compatible with the standard JPEG format, i.e., they are decodable by the standard JPEG decoder. Applications that act on JPEG images will function properly with the marked images.

A variety of options are available to modify hiding capacity according to application requirements, for example, the adjustment of the embedding level and/or the selection of the starting coefficient position, which determines the location where the message is embedded.

Real time embedding of data may be realized without involved complex computations and a large buffering resource. The proposed method is highly suitable for incorporation into hardware devices that operate on images.

In future work, it would be worthwhile to further explore the characteristics of the DCT coefficients within JPEG images, with the goal of providing a higher capacity and/or developing more advanced image data hiding schemes.

ACKNOWLEDGMENTS

This research is financially supported in part by the National Science Council, Taiwan, under the contract NSC-100-2221-E-150-071. Additionally, the authors would like to express thanks to the anonymous reviewers for their constructive suggestions to improve the presentation of this article.

REFERENCES

[1] K. R. Rao, Z. S. Bojkovic, and D. A. Milovanovic, *Multimedia Communication Systems: Techniques, Standards, and Networks* (Prentice-Hall, Englewood Cliffs, New Jersey, 2002), p. 5.
[2] L. M. Mayron, "Secure multimedia communications", IEEE Secur. Privacy **8**, 76–79 (2010).
[3] C. I. Podilchuk and E. J. Delp, "Digital watermarking: Algorithms and applications", IEEE Signal Process. Mag. **18**, 33–46 (2001).
[4] X. Zhang and S. Wang, "Efficient steganographic embedding by exploiting modification direction", IEEE Commun. Lett. **10**, 781–783 (2006).
[5] C. F. Lee, Y. R. Wang, and C. C. Chang, "A steganographic method with high embedding capacity by improving exploiting modification direction", *Proc. Intelligent Information Hiding and Multimedia Signal Processing*, (IEEE, Piscataway, NJ, 2007), pp. 497–500.
[6] W. C. Kuo, L. C. Wu, C. N. Shyi, and S. H. Kuo, "A data hiding scheme with high embedding capacity based on general improving exploiting modification direction method", *Proc. Hybrid Intelligent Systems* (IEEE, Piscataway, NJ, 2009), pp. 69–72.
[7] C. C. Wang, and Y. C. Lin, "An automatic system for monitoring the visual quality and authenticity of satellite video streams using a fragile watermarking approach," Digit. Signal Process. **20**, 780–792 (2010).
[8] A. Phadikar and S. P. Maity, "Quality of access control of compressed color images using data hiding," Int. J. Electron. Commun. **64**, 833–843 (2010).
[9] S. Miaou, C. Hsu, Y. Tsai, and H. Char, "A secure data hiding technique with heterogeneous data-combining capability for electronic patient records", *Proc. IEEE 22nd Annual EMBS International Conference* (IEEE, Piscataway, NJ, 2000), pp. 280-283.
[10] Y. Q. Shi, Z. Ni, D. Zou, C. Liang, and G. Xuan, "Lossless data hiding: fundamentals, algorithms and applications", *Proc. IEEE International Symposium Circuits and Systems* (IEEE, Piscataway, NJ, 2004), pp. 33–36.
[11] G. K. Wallace, "The JPEG still picture compression standard", Commun. ACM **34**, 30–44 (1991).
[12] C. W. Honsinger, P. Jones, M. Rabbani, and J. C. Stoffel, "Lossless recovery of an original image containing embedded data", U.S. Patent 6,278, 791 (2001).
[13] J. Fridrich, M. Goljan, and R. Du, "Invertible authentication", *Proc. Security Watermarking Multimedia Contents* (SPIE, Bellingham, WA, 2001), pp. 197–208.
[14] J. Tian, "Reversible data embedding using a difference expansion", IEEE Trans. Circuits Syst. Video Technol. **13**, 890–896 (2003).
[15] A. M. Alattar, "Reversible watermarking using the difference expansion of a generalized integer transform", IEEE Trans. Image Process. **13**, 1147–1156 (2004).
[16] D. Coltuc and J. M. Chassery, "High capacity reversible watermarking", *Proc. IEEE International Conference on Image Processing* (IEEE, Piscataway, NJ, 2006), pp. 2565–2568.
[17] D. Coltuc and J. M. Chassery, "Very fast watermarking by reversible contrast mapping", IEEE Signal Process. Lett. **14**, 255–258 (2006).
[18] D. M. Thodi and J. J. Rodriguez, "Expansion embedding techniques for reversible watermarking", IEEE Trans. Image Process. **16**, 721–730, (2007).
[19] G. Z. Chuan, W. J. Lin, and N. Hong, "A low complexity reversible data hiding method based on modulus function", *Proc. IEEE International Conference Signal Processing* (IEEE, Piscataway, NJ, 2008), pp. 2221–2224.
[20] C. C. Lin, S. P. Yang, and N. L. Hsueh, "Lossless data hiding based on difference expansion without a location map", *Proc. IEEE Congress on Image and Signal Processing* (IEEE, Piscataway, NJ, 2008), pp. 8–12.
[21] M. U. Celik, G. Sharma, A. M. Tekalp, and E. Saber, "Lossless generalized-LSB data embedding", IEEE Trans. Image Process. **14**, 253–266 (2005).
[22] S. Ohyama, M. Niimi, K. Yamawaki, and H. Noda, "Lossless data hiding using bit-depth embedding for JPEG2000 compressed bit-stream", *Proc. International Conference on Intelligent Information Hiding and Multimedia Signal Processing* (IEEE, Piscataway, NJ, 2008), pp. 151–154.

[23] E. Bodden, M. Clasen, and J. Kneis, "Arithmetic coding revealed", Sable Technical Report 2007-5, Sable Research Group, School of Computer Science (McGill University, Montréal, Canada, 2007).
[24] J. Jiang, "Novel design of arithmetic coding for data compression", IEEE Proc. Comput. Digit. Tech. **142**, 419–424 (1995).
[25] A. Said, "Introduction to arithmetic coding—theory and practice", in *HPL-2004-76, Imaging Systems Laboratory* (HP Laboratories, Palo Alto, CA, 2004).
[26] Z. Ni, Y. Q. Shi, N. Ansari, and W. Su, "Reversible data hiding", IEEE Trans. Circuits Syst. Video Technol. **16**, 354–362 (2006).
[27] K. L. Chung, Y. H. Huang, W. N. Yang, and Y. C. Hsu, "Capacity maximization for reversible data hiding based on dynamic programming approach", Appl. Math. Comput. **208**, 284–292 (2009).
[28] W. C. Kuo, D. J. Jiang, and Y. C. Huang, "A reversible data hiding based on block division", *Proc. IEEE Congress on Image and Signal Processing* (IEEE, Piscataway, NJ, 2008), pp. 365–369.
[29] K. S. Kim, M. J. Lee, and Y. H. Suh, "Histogram-based reversible data hiding technique using subsampling", *Proc. 10th ACM Workshop on Multimedia and Security* (ACM, New York, NY, 2008), pp. 69–74.
[30] K. S. Kim, M. J. Lee, and Y. H. Suh, "Reversible data exploiting spatial correlation between sub-sampled image", Pattern Recognit. **42**, 3083–3096 (2009).
[31] Z. Ni, Y. Q. Shi, N. Ansari, and W. Su, "Robust lossless image data hiding designed for semi-fragile image authentication", IEEE Trans. Circuits Syst. Video Technol. **18**, 497–509 (2008).
[32] J. H. Hwang, J. W. Kim, and J. U. Choi, "A reversible watermarking based on histogram shifting", *Lecture Notes in Computer Science* (Springer), Vol. 4283, pp. 248–351 (2006).
[33] Y. C. Lin and T. S. Li, "Reversible image data hiding using quad-tree segmentation and histogram shifting", J. Multimedia (in press).
[34] Y. C. Lin, T. S. Li, Y. T. Chang, C. C. Wang, and W. T. Chen, "A subsampling and interpolation technique for reversible histogram shift data hiding", *Lecture Notes in Computer Science* (Springer), Vol. 6134, pp. 384–393 (2010).
[35] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity", IEEE Trans. Image Process. **13**, 600–612 (2004).
[36] W. Kienzle, G. Bakir, M. Franz, and B. Scholkopf, "Face detection-efficient and rank deficient", Proc. Adv. Neural Inf. Process. Syst. **17**, 673–680 (2005).
[37] A. S. Georghiades, P. N. Belhumeur, and D. J. Kriegman, "From few to many: Illumination cone models for face recognition under variable lighting and pose", IEEE Trans. Pattern Anal. Mach. Intell. **23**, 643–660 (2001).