Information hiding based on search-order coding for VQ indices

Chin-Chen Chang a, Guei-Mei Chen a, Min-Hui Lin b,*

a Department of Computer Science and Information Engineering, National Chung Cheng University, Chiayi 621, Taiwan, ROC
b Department of Computer Science and Information Management, Providence University, Taichung 433, Taiwan, ROC

Received 19 June 2003; received in revised form 1 March 2004
Available online 18 May 2004

Abstract

Compared with various information-hiding techniques on the spatial domain proposed previously, VQ-based information hiding technique has not been paid much attention. As a result, we present a steganographic scheme based on the search-order coding (SOC) compression method of vector quantization (VQ) indices in this paper. The major goal is to embed secret data into the compression codes of the host image such that the interceptors will not notice the existence of secret data. In the proposed scheme, the embedding process induces no extra coding distortion and adjusts the bit rate according to the size of secret data. According to the experimental results, it is confirmed that the proposed scheme yields a good and acceptable compression ratio of the image. In addition, the receiver can efficiently receive both the compressed image and the embedded data almost at the same time.

© 2004 Elsevier B.V. All rights reserved.

Keywords: Information hiding; Steganography; VQ; Search-order coding compression

1. Introduction

Nowadays, information hiding and cryptography have become two significant topics of computer science due to the increasing popularity of the Internet and the essential need of data security. In cryptographic systems, messages are protected by encryption and decryption techniques such as data encryption standard (DES) (Davis, 1978; NBA FIPS PUB 46-1, 1988) or RSA (Rivest et al., 1978). The ciphertext, the encrypted message, is sent by the sender over the insecure channel—the Internet for example. Upon receiving the ciphertext, the receiver decrypts the ciphertext by using the corresponding decryption key. The main disadvantage of the cryptographic systems is that the ciphertext usually seems meaningless. Because of the disadvantage, the interceptors may notice the transmission of the secret data. As a result, the information hiding technique is proposed to overcome this weakness.
The information hiding technique is embedding data into various media, which are called host signals, such as image, audio and video. What is more, the embedded data should be invisible or inaudible to the human observer. The information hiding techniques are mainly classified into two categories, steganography and copyright marking (i.e. watermarking) (Bauer, 1996; Petitcolas et al., 1999), as shown in Fig. 1. The latter (Barni et al., 2001; Cox et al., 1997) is used to embed the copyright information, a signature or a trademark, into the host signal for ensuring the ownership of the stego-signal. Therefore, the watermark must be kept concealed in a host signal and must be able to be retrieved even if the stego-signal suffers from manipulations such as cropping, resampling, lossy compression, etc. On the other hand, watermarking composes of imperceptible watermarking and visible watermarking. Note that the size of the watermark is usually very small.

The goal of steganography (Chang et al., 2003; Johnson and Jajodia, 1998; Shaulding et al., 2002) is quite different from watermarking. Steganography is employed to embed secret data, text or image—for example, into the host signal, and the interceptors will not notice the existence of the embedded data. Instead of resisting the manipulations threatening watermarking, steganography makes the embedded data unnoticeable. In this paper, a novel steganographic method, based on the SOC compression method of vector quantization (VQ) indices (Gary and Gersho, 1992; Nassarabadi and King, 1988), is proposed without inducing extra image coding distortion. A still image, also called the host image, is taken to be the carrier of the secret data.

Previously, many works have been proposed to hide information based on VQ (Chung et al., 2001; Jo and Kim, 2002). Nevertheless most of them degrade the image quality after embedding secret data into a host image. As to our proposed scheme, the secret data are embedded into the compression codes of the host image by employing the search-order coding algorithm (Hsieh and Tsai, 1996). Our proposed scheme induces no extra distortion after the secret data are embedded. And the receiver can efficiently receive both the compressed image and the embedded data almost at the same time. Moreover, the proposed scheme can provide higher image quality of the retrieve host image even if the secret data have been embedded.

VQ is an effective technique for compressing images. In a VQ-based system, an image is usually divided into many blocks of the same size, and then each block is independently encoded by an

![Fig. 1. A classification of information hiding techniques based on Bauer (1996) and Petitcolas et al. (1999).](image-url)
index of the closest codeword in the codebook. Hence, an index table is generated. That is, the index table consists of the VQ indices for all blocks of the encoded image. Image compression is achieved by transmitting these VQ indices instead of the image. Because the adjacent blocks of an image are always very similar, Hsieh and Tsai (1996) proposed an algorithm, called search-order coding (SOC), to encode the VQ indices with fewer bits. In our scheme, the secret data are embedded into the SOC compression codes, and the bit rate is adjusted according to the size of secret data. In addition, for enhancing the security, the position of the index table, for hiding each bit of the secret data, is determined by using a pseudo random number generator (Hwang et al., 1999), and the secret data are encrypted by the traditional cryptography system, DES or RSA, in advance.

The rest of this paper is organized as follows. In Section 2, the search-order coding algorithm is briefly reviewed. The proposed information hiding scheme is presented in Section 3. In Section 4, some experimental results are given, followed by the conclusions in Section 5.

2. Review of search-order coding algorithm (Hsieh and Tsai, 1996)

In 1996, Hsieh and Tsai proposed the SOC algorithm to further increase the compression rate of the VQ indices of an image. In a traditional VQ-based system, an image is first divided into same-sized blocks. Each block, regarded as a multi-dimensional vector, is encoded as the index of the closest codeword in the codebook. Hence an index table is generated. Image compression is achieved by transmitting these indices instead of pixels in the spatial domain. Here, a codebook consists of multi-dimensional vectors, which are called codewords and are representative vectors selected from some standard images. In the decoder, the codebook based look-up procedure will be performed to recover the image according to the received indices.

The SOC algorithm encodes the traditional VQ indices with fewer bits by utilizing the high correlation feature of the adjacent indices, where high correlation denotes that there may exist many blocks with the same indices in the neighborhood.

First of all, the SOC algorithm encodes each index in the index table one by one and with the raster scan order, which is from left to right and top to bottom. With deep insight into the SOC algorithm, the main idea is (1) to find the same index around the current processed index in a predefined search path and (2) to encode the current processed index with the search-order code instead of its original index value. All of the indices appearing in the predefined search path are called search points (SP), and non-search points are those indices that appear after the current processed index in the raster scan order. Search-order codes are generated according to the comparison order of the SPs and the current processed index. If none of the SPs matches with the current processed index, the original index value is preserved. Note that an indicator, of which the length can be 1-bit, must be added in front of the result compression codes of each index in order to differentiate search-order codes from the original index value.

Let us give an example as shown in Fig. 2. The current processed index, the index of the point (3, 3) equals to 81. And a predefined search path is shown with arrows. It is obvious that the SPs in the predefined search path are sequentially
compared to the point (3, 3) and are denoted with the comparison order. For example, if a 2-bits search-order code is used here, the point (3, 2) is denoted as “00” since it is the first SP in the predefined search path of the current processed index (3, 3). Similarly, the point (2, 2) is denoted as “01” because it is the second SP in the predefined path of the current processed index (3, 3). It is noted that the exclusion of repetition SP can increase the possibility for finding a matched index. If the original index value of a SP is equal to the previous one in the predefined search path, the SP is just excluded, such as the points (2, 3), (2, 4) and (2, 1). As a result, if the length of search-order codes is 2-bit, there are four SPs are used for comparison without excluding the repetition points; whereas, more than four SPs when the repetition points are excluded. The length of the search-order codes, n-bit, must be less than that of the original index value so that compression can be achieved.

As shown in Fig. 2, the point (3, 3) is encoded with the search-order code “11” since the index value of the point (1, 1) is equal to that of itself. As for the compression codes, a 1-bit indicator will be added in front of the search-order codes “11”. On the other hand, if the original index value of the point (1, 1) is not equal to that of the current processed index, it denotes that no matched SP of the current processed point (3, 3) is found. Then the original index value of the point (3, 3) is preserved. That is, it is encoded with its original index value 81 (= 01010001) instead of the search-order code “11”.

First, we list the notations used in the SOC algorithms in Section 2.1. Then, the SOC encoding algorithm is presented in Section 2.2.

### 2.1. The notations

The notations used in the SOC algorithms are listed as follows:

- **m**: the number of codewords in the codebook;
- **j**: block size, that is, j-dimensional vector;
- \((b_1, b_2, \ldots, b_j)\): the pixels of a block in the input image;
- \((v_{i1}, v_{i2}, \ldots, v_{im})\): the vectors of a codeword in the codebook (where \(i = 1, 2, \ldots, m\));
- **n**: the number of bits assigned to the SOC codes.

### 2.2. The SOC encoding algorithm

The details of the SOC encoding algorithm are shown as follows:

**Input**: Each block of an image in the raster scan order.

**Output**: The compression codes of the input image.

**Step 1**: Input a block, \((b_1, b_2, \ldots, b_j)\), of the image. Generate its corresponding VQ index from the codebook according to the closest codeword principle:

\[
\min \left\{ \sqrt{(b_1 - v_{i1})^2 + (b_2 - v_{i2})^2 + \cdots + (b_j - v_{im})^2} \right\},
\]

where \(i = 1, 2, \ldots, m\).

**Step 2**: Try to find a SP (search point) with the same VQ index in the predefined search path of the index table until the current processed block cannot be encoded with any of the SOC codes, \((0)_2 \sim (2^n - 1)_2\), or a matched one is found. Note that the repeated SP, whose index value equals to the previous one in the predefined search path, should be excluded as mentioned above.

**Step 3**: If a matched SP is found, the current processed block will be encoded with a 1-bit indicator followed by the corresponding SOC code; whereas, followed by its original index value. Go to Step 1 for next block.

### 3. The proposed scheme

In the SOC coding method, all of the indices in the index table are exactly encoded with either search-order codes (SOC) or original index values (OIV). In other words, the final compression codes consist of the SOC and OIV compression codes. The receiver can distinguish them according to the indicator as described in Section 2. Based on this characteristic, secret data can be embedded into the compression codes without inducing additional coding distortion in our proposed scheme. That is, the receiver determines that each bit of secret data...
is “0” or “1” according to whether the received compression code is SOC or OIV. For example, if the receiver receives a SOC compression code, the bit of the secret data is “0”. On the contrary, the bit of the secret data is “1” when an OIV compression code is received. According to the above description, it is obvious that each index in the index table can hide one bit of the secret data. For example, if the block size is $4\times 4$ pixels, a $512\times 512$ image will first be encoded to be a $128\times 128$ index table. So the size of secret data can be $128\times 128$ bits at most.

In the hiding process, there are four categories taken into consideration. Assume that the SOC compression code represents that the hidden secret bit is “0”, and the OIV compression code represents that the hidden secret bit is “1”. For the first category, aiming at the current index, if it is originally encoded as OIV and the secret hidden bit is “1”, there is nothing needing to be changed for its compression code. For the second category, if the current index is originally encoded as SOC and the secret hidden bit is “1”, it is necessary to preserve the OIV compression code instead of the SOC one. As far as the second category is concerned, the compression rate will decrease. For the third category, there is nothing that needs to be changed if the current index is originally encoded as SOC and the hidden secret bit is “0”. Finally, for the fourth category, if it is originally encoded as OIV and the secret data is “0”, a translation technique of translating OIV into SOC is performed. The translation technique is described as follows.

In the original SOC algorithm, the number of bits, $n$, which is assigned to search-order codes, is predefined. It means that the search-order codes are in binary representation from 0 to $(2^n - 1)$. Here, the binary representation of $(2^n - 1)$ followed by the original index value is used to represent the translation of OIV into SOC. For example, if $n$ is equal to 2, the binary representation of $(2^n - 1)$ will be equal to “11”. When there is the need to translate OIV into SOC, the translated compression code will be “11” followed by the original index value of the current index. Note that the indicator is still in front of them. According to the indicator and search-order codes “11”, the receiver can retrieve the hidden secret bit “0”. Moreover, the original index value can be recovered if necessary. After this translation, the compression rate is also decreased. Additionally, note that the search-order codes of the un-translated SOC will be in the binary representation from 0 to $(2^n - 2)$ since the binary representation of $(2^n - 1)$ is used as a flag which indicates the existence of the translation of OIV to SOC.

For example, as shown in Fig. 3, the $3\times 3$ index table is encoded with a codebook of size 256. Suppose that through the original SOC coding algorithm, where $n = 2$, the coding results are also shown in Fig. 3 before any data are hidden. And the secret data is “111110100”. In addition, the indicator of SOC is “0” and that of OIV is “1”. In the raster scan order, the corresponding positions for hiding each bit of the secret data are shown in Fig. 4. It is defined here that “0” is embedded into the SOC compression code and “1” is embedded into the OIV compression code. The embedded bit of each index in Fig. 3 will be the bit in the corresponding position in Fig. 4.

Here, $(x,y)$ denotes the position in Fig. 3 and $\ell$ denotes the corresponding bit in Fig. 4. To start the hiding procedure, first check on each pair $[(x,y), \ell]$, where $x, y \in \{1, 2, 3\}$ and $\ell \in \{0, 1\}$, in

![Fig. 3. A $3\times 3$ index table for showing the original coding results of the SOC algorithm.](image1)

![Fig. 4. The hiding position of each bit of the secret bit string “111110100” in the raster scan order.](image2)
the corresponding position, to which category it belongs. Then decide how to generate the compression codes. For example, the pair $[(1,1),1]$ belongs to the first category and its compression code is still OIV, “10010010”, because it is originally encoded as OIV and the secret data is “1”.

In addition, the pair $[(2,2),1]/C_{138}$ belongs to the second category and it is necessary to replace its compression code with the OIV compression code, “100011110”, instead of the SOC compression code, “000”. Then, the pair $[(2,3),0]/C_{138}$, which belongs to the third category is still encoded to be “010”, because it is originally encoded as SOC and the secret data is “0”. Finally, the pair $[(3,3),0]/C_{138}$, which belongs to the fourth category, is encoded with “010100000” instead of “100100000”. After receiving the above bit stream, the receiver knows the embedded secret bit is 0 since the first indicator bit, 0, means the SOC encoding. However, he also knows it is a fake because the search-order code is “11” and the original index value is attached. The notations used in our proposed scheme are listed in Section 3.1. The detail description of the proposed algorithm is given in Section 3.2.

### 3.1. Notations

The notations used in our proposed scheme are listed as follows:

- $u, v: u \times v$ means the amount of blocks in an image. Hence, the number of bits of the secret data is $u \times v$ at most;
- $c_codes$: the compression codes generated from VQ and the SOC algorithm;
- $\ell$: the bit of the secret data;
- $n$: the number of bits assigned to SOC codes.

### 3.2. The proposed information hiding algorithm

The details of the proposed algorithm are demonstrated as follows:

**Input:** (1) Each block of an image in the raster scan order and (2) the bit strings of the secret data.

**Output:** The compression codes of the input image including the secret data, where the SOC compression code means that “0” is hided, and the OIV compression code means that “1” is hided.

For $x := 1$ to $u$ do

Begin

For $y := 1$ to $v$ do

Begin

$c_codes := \text{SOC\_Algo}(2^n - 2, x, y, \text{index\_table})$

$\ell := \text{RetrieveBit}(x, y, \text{secret\_data})$

If ($c_codes \in \text{SOC}$) and ($\ell = \text{1}$) then

$c_codes := (\text{the indicator of OIV})$

$\cup (\text{the original index value})$

// change SOC into OIV

else If ($c_codes \in \text{OIV}$) and ($\ell = \text{0}$) then

$c_codes := (\text{the indicator of SOC})$

$\cup (2^n - 1)_2 \cup (\text{the original index value})$

// change OIV into SOC

else $c_codes := c_codes$

Output ($c_codes$)

End

End

End

### 4. Experimental results

As to the four categories of our proposed scheme in Section 3, a bit-based cost table for hiding data in the compression codes is illustrated in Table 1, where $n$ is defined as in Section 3 and $N_c$ denotes the size of the codebook. It denotes that there are increases of $(\log_2 N_c - n)$ bits in the compression codes of an index for using the OIV compression code instead of the SOC compression code in the second category, and $n$ bits for translating the OIV compression code to the SOC compression code in the fourth category. According to Table 1, some simulation results including the comparisons of compression rates are shown below.

The VQ codebook is generated by using the LBG algorithm (Linde et al., 1980) and five 512×512 standard images, Airplane, Boat, Lena, Sailboat and Toys, with 256 gray levels. There are 256 codewords in the codebook. In addition, the six host images used in our experiments are standard 256×256 images with 256 gray levels as well...
shown in Fig. 5. Here, each index table consists of $64 \times 64$ indices if the block size is of $4 \times 4$ pixels. It means that the size of secret data is $64 \times 64$ bits at most. As shown in Fig. 6, three $64 \times 64$ binary images, Barbara, Lena and Peppers, are used as the secret data. The secret data is embedded into the host images.

In our scheme, it is easy to figure out that the larger the amount of the embedded data is, the lower the compression rate of the host image is. However, the quality of the host image never gets worse. Bit rates of the host images after different sized secret binary images, “Lena” and “Peppers”, are embedded, are shown in Tables 2 and 3, respectively.

In Tables 4 and 5, the bit rates of the search-order coding (SOC) compression algorithm are compared with those of our scheme. Note that the parameter $n$, equal to 2, is defined as mentioned in Section 2. The size of the secret data in our scheme is 1024-bit. According to the analyses of Tables 1, 4 and 5 for the compression codes, it is usually not critical to use OIV to present the secret bit “0” or “1” since the bit rates are very similar.

For enhancing the security of our schemes, the secret data can be encrypted by using traditional cryptography system, such as DES (Davis, 1978; NBA FIPS PUB 46-1, 1988) or RSA (Rivest et al., 1978), before the embedding process. Furthermore, the positions for embedding secret data in the index table can be kept secret and only known by the sender and the receiver. The pseudo random number generator (Hwang et al., 1999) can be employed to achieve the goal. Under this technique, the one owning the same “seed” as that owned by the sender can obtain the same random

Table 1
The amount of increasing bits for hiding data in the compression codes

<table>
<thead>
<tr>
<th>Code category in the results of the original SOC coding method</th>
<th>Code category in the results of our information hiding code category in the results of our information hiding</th>
</tr>
</thead>
<tbody>
<tr>
<td>OIV</td>
<td>0</td>
</tr>
<tr>
<td>SOC</td>
<td>$\log_2 N_c - n$</td>
</tr>
<tr>
<td></td>
<td>$n$</td>
</tr>
<tr>
<td></td>
<td>0</td>
</tr>
</tbody>
</table>
**Fig. 6.** Secret binary images of $64 \times 64$ pixels: (a) Barbara, (b) Lena and (c) Peppers.

### Table 2
Bit rate for embedding different sized secret binary image “Lena” into six host images

<table>
<thead>
<tr>
<th>The size of secret data (bits)</th>
<th>Airplane</th>
<th>Boat</th>
<th>Girl</th>
<th>Lena</th>
<th>Peppers</th>
<th>Toys</th>
</tr>
</thead>
<tbody>
<tr>
<td>1024</td>
<td>0.3976</td>
<td>0.4162</td>
<td>0.4571</td>
<td>0.4509</td>
<td>0.4468</td>
<td>0.3733</td>
</tr>
<tr>
<td>2048</td>
<td>0.4367</td>
<td>0.4484</td>
<td>0.4821</td>
<td>0.4817</td>
<td>0.4721</td>
<td>0.4134</td>
</tr>
<tr>
<td>3072</td>
<td>0.4707</td>
<td>0.4730</td>
<td>0.5158</td>
<td>0.5144</td>
<td>0.4972</td>
<td>0.4520</td>
</tr>
<tr>
<td>4096</td>
<td>0.4962</td>
<td>0.5025</td>
<td>0.5418</td>
<td>0.5418</td>
<td>0.5196</td>
<td>0.4789</td>
</tr>
</tbody>
</table>

### Table 3
Bit rate for embedding different sized secret binary image “Peppers” into six host images

<table>
<thead>
<tr>
<th>The size of secret data (bits)</th>
<th>Airplane</th>
<th>Boat</th>
<th>Girl</th>
<th>Lena</th>
<th>Peppers</th>
<th>Toys</th>
</tr>
</thead>
<tbody>
<tr>
<td>1024</td>
<td>0.3935</td>
<td>0.4209</td>
<td>0.4571</td>
<td>0.4510</td>
<td>0.4351</td>
<td>0.3729</td>
</tr>
<tr>
<td>2048</td>
<td>0.4297</td>
<td>0.4539</td>
<td>0.4806</td>
<td>0.4788</td>
<td>0.4602</td>
<td>0.4180</td>
</tr>
<tr>
<td>3072</td>
<td>0.4487</td>
<td>0.4774</td>
<td>0.5015</td>
<td>0.4943</td>
<td>0.4897</td>
<td>0.4384</td>
</tr>
<tr>
<td>4096</td>
<td>0.4961</td>
<td>0.5124</td>
<td>0.5459</td>
<td>0.5433</td>
<td>0.5374</td>
<td>0.4790</td>
</tr>
</tbody>
</table>

### Table 4
Bit rates of the SOC scheme and our information hiding method with secret binary image “Barbara” of 1024 bits

<table>
<thead>
<tr>
<th>Methods</th>
<th>Images</th>
<th>Airplane</th>
<th>Boat</th>
<th>Girl</th>
<th>Lena</th>
<th>Peppers</th>
<th>Toys</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOC (Hsieh and Tsai, 1996)</td>
<td></td>
<td>0.3602</td>
<td>0.3814</td>
<td>0.4319</td>
<td>0.4206</td>
<td>0.4168</td>
<td>0.3316</td>
</tr>
<tr>
<td>OIV represents to hide “1”</td>
<td></td>
<td>0.3980</td>
<td>0.4227</td>
<td>0.4558</td>
<td>0.4525</td>
<td>0.4471</td>
<td>0.3749</td>
</tr>
<tr>
<td>OIV represents to hide “0”</td>
<td></td>
<td>0.3983</td>
<td>0.4197</td>
<td>0.4561</td>
<td>0.4514</td>
<td>0.4419</td>
<td>0.3740</td>
</tr>
</tbody>
</table>

### Table 5
Bit rates of the SOC scheme and our information hiding method with secret binary image “Lena” of 1024 bits

<table>
<thead>
<tr>
<th>Methods</th>
<th>Images</th>
<th>Airplane</th>
<th>Boat</th>
<th>Girl</th>
<th>Lena</th>
<th>Peppers</th>
<th>Toys</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOC (Hsieh and Tsai, 1996)</td>
<td></td>
<td>0.3602</td>
<td>0.3814</td>
<td>0.4319</td>
<td>0.4206</td>
<td>0.4168</td>
<td>0.3316</td>
</tr>
<tr>
<td>OIV represents to hide “1”</td>
<td></td>
<td>0.3976</td>
<td>0.4162</td>
<td>0.4571</td>
<td>0.4509</td>
<td>0.4468</td>
<td>0.3733</td>
</tr>
<tr>
<td>OIV represents to hide “0”</td>
<td></td>
<td>0.3987</td>
<td>0.4262</td>
<td>0.4548</td>
<td>0.4531</td>
<td>0.4422</td>
<td>0.3756</td>
</tr>
</tbody>
</table>
positions. Even though the interceptors are aware of the existence of the secret data, they still cannot get the secret data.

5. Conclusions

In this paper, a novel steganographic scheme is proposed to embed the secret data into multimedia such as digital images. The purpose of steganographic techniques is different from traditional cryptography and watermarking, where the former encrypts messages into the meaningless data and the latter is used to protect the copyright. Steganography covers the secret data with the host image as camouflage and is regarded as an extension of traditional cryptography. As far as we know, our proposed steganographic scheme is the first scheme that embeds the secret data into the compression codes of the VQ indices directly. Moreover, embedding the secret data will not incur any distortion, and the receiver can efficiently receive both the compressed image and the embedded data almost at the same time.

References