An information hiding scheme with minimal image distortion

Ching-Chiuwan Lin *

Department of Information Management, Overseas Chinese University, Taichung 40721, Taiwan

ARTICLE INFO

Article history:
Received 19 September 2009
Received in revised form 6 February 2011
Accepted 7 February 2011
Available online 21 February 2011

Keywords:
Information hiding
Steganography
Secret communication

ABSTRACT

This paper proposes an embedding scheme which can embed a message into an image and obtain minimal image distortion for applications which need a high-visual-quality stego-image. The message to be embedded is divided into sub-messages each of which is embedded into a pixel vector with three pixels. A sub-message is extracted from a stego-pixel vector by calculating the differences between pixels. The embedding capacity of an image using the proposed scheme can be more than one bit per pixel and the modification of a pixel is not more than one. Since the modification of pixels is minimal, applications using the proposed scheme can obtain a stego-image with higher visual quality than existing studies.

© 2011 Elsevier B.V. All rights reserved.

1. Introduction

Information hiding is a technique which imperceptibly embeds important data into media such as images, voices, or documents [1,2]. Since images are popular media on the Internet, they are good candidates of cover media for secret communication. When hiding data into an image, the values of pixels in the image are modified according to the data to be embedded. To enable embedded information in an image to be imperceptible, the image should be distorted as slightly as possible. Usually, the more we embed data into an image, the more the image is distorted. Employing a good embedding method, which can embed a large amount of data and result in low image distortion, is an important issue when data would be embedded in an image.

Many studies have been proposed for embedding secret data in spatial- or transformed-domain images [3–5]. Studies for spatial-domain images are more popular than those for transformed-domain images. In general, a spatial-domain image can provide more embedding capacities and higher visual quality than a transformed-domain image. The simplest way of embedding a message into a spatial-domain image is least-significant-bit (LSB) substitution which replaces the LSBs of a pixel in the image with the message bits to be embedded. For example, if message bits (00)2, (10)2, and (11)2 would be embedded into pixels with pixel values equal to (24)16, (25)16, and (28)16, respectively, the pixel values will be changed to (24)16, (26)16, and (28)16, respectively. Note that the first pixel value is unchanged, whereas the value of the last pixel is increased by 3. If the above message bits are embedded into the most significant bits (MSBs), the pixel values will be changed to (24)16, (A5)16, and (E8)16, respectively. Note that the value of the last pixel is increased by 192 and the modification will be visually perceptible, since the intensity of the pixel is significantly changed. To decrease the modification of pixel value, the encoder usually embeds a message in LSBs instead of MSBs.

To make a stego-image more similar to its cover image such that the embedded message is less noticeable by hackers, recently, a number of studies [6–11] have been proposed. Thien and Lin [10] embedded a message with n kinds of symbols into an image by adjusting cover pixel value x to stego-pixel value x′ such that x′ mod n = i and |x′ − x| is minimized, where i is a symbol in the message to be embedded. In Wang’s method [11], a secret image is converted into a binary bit-string and embedded into a cover image. A threshold T and two modulus values u and l are selected to determine the embedding capacity of a cover pixel with pixel value x. Specifically, if x > T, then embed a sub-message m with |log2u| bits into the pixel by adjusting x to x′ such that x′ mod u = val(m), where val(m) denotes the decimal value of the sub-message m. Otherwise, embed a sub-message m′ with |log2l| bits into the pixel by adjusting x to x′ such that x′ mod l = val(m′). In his method, a pixel value with high intensity embeds a longer sub-message than a pixel value with low one, i.e. u > l. When adjusting a pixel value, an adaptable process is applied to decrease the modification of pixel values.

Based on the sensitivity of human vision in smooth area, Wu and Tsai proposed a steganographic method [12] in which a sub-message with k bits is embedded into a difference between two consecutive pixels in an image, where k is determined by the difference value. The larger the difference value is, the longer the sub-message is embedded. Later, Wu and Tsai’s method was improved by Wu et al. [13] and Chang et al. [14]. Recently, Wang et al. [15] was inspired by
Wu and Tsai’s method and proposed an optimal approach to adjusting pixel values when embedding a sub-message. Also inspired by Wu and Tsai’s method, Yang et al. [16] introduced an adaptive LSB substitution method which replaces more LSBs in edge areas with secret data to be embedded.

This paper proposes a new embedding scheme which can embed a message into an image and obtain minimal image distortion for applications which need a high-visual-quality stego-image. The message to be embedded is divided into sub-messages each of which is embedded into a pixel vector with three pixels. The sum of weighted differences between pixels in a vector is divided by an integer and, then, the remainder is used to be an identifier for embedding a sub-message. To make a pixel value less modified, a refining process is applied. A sub-message is extracted from a stego-pixel vector by calculating the remainder of the sum of weighted differences divided by the integer. The embedding capacity of an image using the proposed scheme can be more than one bit per pixel and the modification of a pixel is not more than one. Therefore, a stego-image with higher visual quality can be expected, when the proposed scheme is applied.

The rest of this paper is organized as follows. Related works are briefly reviewed in Section 2. Section 3 introduces the proposed scheme, including the embedding and extraction processes. To show the performance of the proposed scheme, Section 4 presents the experimental results and compares them with existing approaches. The contributions of the proposed scheme are discussed in Section 5, followed by conclusions in Section 6.

2. Related works

Mielikainen [17] proposed a new LSB matching method which uses a pair of pixels as an embedding unit. Let \((x, y)\) be a pair of cover pixels and \((b_1, b_2)\) be a pair of message bits to be embedded. In his method, a pixel embeds a bit of message and the stego-pixels \((x', y')\) for \((x, y)\) are determined by

\[
(x', y') = \begin{cases} 
(x, y) & \text{if } b_1 = \text{LSB}(x) \text{ and } b_2 = \text{LSB}(l(x/2l + y)), \\
(x, y+1) & \text{if } b_1 = \text{LSB}(x) \text{ and } b_2 \neq \text{LSB}(l(x/2l + y)), \\
(x-1, y) & \text{if } b_1 \neq \text{LSB}(x) \text{ and } b_2 = \text{LSB}(l(x-1/2l + y)), \\
(x+1, y) & \text{if } b_1 \neq \text{LSB}(x) \text{ and } b_2 \neq \text{LSB}(l(x+1/2l + y)).
\end{cases}
\]

(1)

where \(\text{LSB}(x)\) denotes the least significant bit of \(x\). In simple LSB substitution, if \(\text{LSB}(x) \neq b_1\) and \(\text{LSB}(y) \neq b_2\), both \(x\) and \(y\) will be modified such that \(\text{LSB}(x') = b_1\) and \(\text{LSB}(y') = b_2\). However, in Eq. (1), at most one pixel will be increased or decreased by one, even if \(\text{LSB}(x) \neq b_1\) and \(\text{LSB}(y) \neq b_2\). In the extraction process, \(b_1\) is extracted from the LSB of \(x\) and \(b_2\) is not simply extracted from the LSB of \(y\) but determined by \(\text{LSB}(l(x'/2l + y'))\). For example, if \((x, y) = (25, 26)\) and \((b_1, b_2) = (0, 1)\), then \((x, y)\) will become \((26, 26)\) after \(b_1\) and \(b_2\) are embedded. The decoder extracts \(b_1 = \text{LSB}(x') = 0\) and \(b_2 = \text{LSB}(l(x'/2l + y')) = 1\). For any \((x, y)\) and \((b_1, b_2)\), the probability of \(b_1 = \text{LSB}(x)\) is 0.5. On the basis of \(b_1 = \text{LSB}(x)\), the probability of \(b_2 = \text{LSB}(l(x/2l + y)) = 0.5\). In other words, the probability of \(b_1 = \text{LSB}(x)\) and \(b_2 = \text{LSB}(l(x/2l + y)) \geq 0.5\). Similarly, the remaining three conditions have the same probability of occurrence, i.e., 0.25. Thus, in Eq. (1), the probability of modifying a pixel in Mielikainen’s method is equal to \((0.25/2 + 0.25/2 + 0.25/2) = 0.375\). If applying simple LSB substitution to embed a message, the probability of modifying a pixel which embeds a bit of message is equal to 0.5. This implies that a stego-image using Mielikainen’s method will be more similar to its cover image than that using simple LSB substitution.

In 2004, Chan and Cheng [18] proposed an optimal pixel adjustment process (OPAP) to improve the visual quality of a stego-image in which secret data are embedded by applying simple LSB substitution. If a pixel \(x\) would embed \(k\) bits of message \(b\) in its LSBs, in the worst case, the pixel value may be increased or decreased by \(2^k - 1\). The OPAP embeds a message into a pixel by simple LSB substitution and adjusts stego-pixel \(x'\) to

\[
x' = \begin{cases} 
x' + 2^k & \text{if } x' - x > 2^{k-1}, \\
x' - 2^k & \text{if } x' - x < 2^{k-1}, \\
x' & \text{otherwise.}
\end{cases}
\]

For example, if \(x = (28)_{16} = (00101000)_{2}\), \(k = 3\) and \(b = (111)_{2}\), then the stego-pixel will become \(x' = (2F)_{16} = (00101111)_{2}\), if \(b\) is embedded into \(x\) by simple LSB substitution. In Chan and Cheng’s study, \(x'\) will be adjusted to \(x'' = (27)_{16} = (00100111)_{2}\) to decrease the modification of the pixel caused by embedding the message.

3. Proposed scheme

The proposed scheme includes an encoding process, which embeds a message into a grayscale cover image in spatial domain, and a decoding process which extracts the message from the stego-image. It is feasible to extend the proposed scheme to a true-color image in which many shades are included and the difference between one shade and the next is visually imperceptible. For example, in a digitalized true-color image which consists of Red, Green, and Blue color planes, a color plane (for example, Red) includes 256 shades and the proposed scheme can be applied to the color plane if we regard it as a grayscale image. Alternatively, if a true-color pixel is digitalized and denoted by three components: Red, Green, and Blue, each of which is an integer between 0 and 255, then a pixel in a true-color image is analogous to a vector with three pixels in a grayscale image. For example, \((24, 192, 85)\) is a digitized true-color pixel and its intensity values for Red, Green, and Blue are 24, 192, and 85, respectively. When applying the proposed scheme to embed a message into a true-color image, the pixel in the above example may be regarded as a vector with three pixels in a grayscale image. In this situation, if \((24, 192, 85)\) is adjusted to \((24, 193, 84)\) to embed a message, the stego-pixel is visually indistinguishable from its cover pixel. This implies that the way of embedding a message into a grayscale image and that of embedding a message into a true-color image are the same. The embedding and extraction processes for a grayscale image are presented in the following. However, they can also be applied to a true-color image.

3.1. Embedding a message

Let the message to be embedded be a binary bit string \(S\). Given a grayscale cover image \(C\) with \(N\) pixels, the encoding process for embedding a message into the image is listed below.

1. Convert \(S\) into a large integer \(I\) and denote \(I\) by a 9-ary notational system. Specifically, if \(S\) is a bit string with \(n\) bits denoted by \(s_{n-1}s_{n-2}...s_0\), where \(s_i \in \{0, 1\}\), then convert \(S\) into

\[
I = s_{n-1} \times 2^{n-1} + s_{n-2} \times 2^{n-2} + ... + s_1 \times 2^1 + s_0 \times 2^0
\]

\[
= k_{m-1} \times 9^{m-1} + k_{m-2} \times 9^{m-2} + ... + k_1 \times 9^1 + k_0 \times 9^0
\]

where \(0 \leq k_i \leq 8, k_0 = 0\) mod 9, \(k_1 = q_1\) mod 9, ..., \(k_{m-1} = q_{m-1}\) mod 9, \(q_0 = I\) mod 9, ..., \(q_{m-1} = I\) mod 9. In this paper, \(k_i\) is called a sub-message of \(S\).

2. Divide cover image \(C\) into disjoint vectors of three pixels \((\alpha, \beta, \gamma)\), where \(i \in \{0, 1, 2, ...\}\) is the identification of vector.

3. Calculate \(d_i = (\alpha - \beta) \mod 3, d_i = (\gamma - \alpha) \mod 3, \) and \(k = 3 \times d_1 + d_2\) for each vector \(i\), where \(d_1 \geq 0\) and \(d_2 \geq 0\). For example, if \(\alpha = 24, \beta = 17\) and \(\gamma = 31\), then \((\alpha - \beta) \mod 3 = 7 - 3 \times (17/3) = 1\) and
\( (\alpha - \gamma) \mod 3 = -7 - 3 \times 1 - 7/3 \equiv 2 \). If the sub-message to be embedded into vector \( i \) is equal to \( k' \), calculate

\[
\beta' = \begin{cases} 
\beta + 1 & \text{if } \Delta_1 = 2, \\
\beta - 1 & \text{if } \Delta_1 = -2, \\
\beta & \text{otherwise},
\end{cases}
\]

\[
\gamma' = \begin{cases} 
\gamma + 1 & \text{if } \Delta_2 = 2, \\
\gamma - 1 & \text{if } \Delta_2 = -2, \\
\gamma & \text{otherwise},
\end{cases}
\]

where \( \Delta_1 = \lfloor k/3 \rfloor - \lfloor k/3 \rfloor \) and \( \Delta_2 = (k_i \mod 3) - (k \mod 3) \).

4. For each vector \( i \), calculate

\[
(\alpha', \beta', \gamma') = \begin{cases} 
(\alpha + 1, \beta, \gamma) & \text{if } |\beta'| = |\beta| - 1 \text{ and } |\gamma'| = |\gamma| - 1, \\
(\alpha - 1, \beta, \gamma) & \text{if } |\beta'| = |\beta| + 1 \text{ and } |\gamma'| = |\gamma| + 1, \\
(\alpha, \beta', \gamma') & \text{otherwise},
\end{cases}
\]

and adjust cover pixels \( \alpha, \beta, \gamma \) to stego-pixels \( \hat{\alpha}, \hat{\beta}, \hat{\gamma} \), respectively, where

\[
\hat{\alpha} = \begin{cases} 
\alpha' - 3 & \text{if } \alpha' > 255, \\
\alpha' + 3 & \text{if } \alpha' < 0, \\
\alpha' & \text{otherwise},
\end{cases}
\]

\[
\hat{\beta} = \begin{cases} 
\beta' - 3 & \text{if } \beta' > 255, \\
\beta' + 3 & \text{if } \beta' < 0, \\
\beta' & \text{otherwise},
\end{cases}
\]

\[
\hat{\gamma} = \begin{cases} 
\gamma' - 3 & \text{if } \gamma' > 255, \\
\gamma' + 3 & \text{if } \gamma' < 0, \\
\gamma' & \text{otherwise}.
\end{cases}
\]

Fig. 1 shows an example illustrating the proposed scheme, where \( \alpha \) is denoted by an underlined pixel. Fig. 1(a) is a cover image with 12 grayscale pixels. Let the message to be embedded be \( S = 01010111100010 \). Before embedding the message, \( S \) is converted into \( I = 5602 = 7 \times 9^3 + 6 \times 9^2 + 1 \times 9^1 + 4 \times 9^0 \). \( k = 3 \times 6 + 2 = 20 \). The encoded message contains one bit of \( S \) without providing extra information (for example, the length of \( S \) or the number of bits of prefix 0’s) if they are converted into integers and embedded in stego-images. The encoder and decoder can make an agreement for the extra information embedded, including its format. As long as the agreement is reached, the encoder constructs the extra information and embeds it into the selected image as the way of embedding a message. Alternatively, the encoder may separately send the extra information to the decoder through a secure channel, instead of embedding it into the selected image. Since the proposed scheme focuses on minimizing distortion of a stego-image, this paper will not include details for embedding extra information into an image or separately sending it to the decoder.

Given that \( k_6 = 4 \) and \( (\alpha, \beta, \gamma) = (20, 18, 25), \) in step 3, \( d_1 = (20 - 18) \mod 3 = 2, \) \( d_2 = (20 - 25) \mod 3 = 1, \) \( k = 3 \times d_1 + d_2 = 7, \) \( \Delta_1 = k_6 \lfloor k/3 \rfloor - \lfloor k/3 \rfloor = 1, \) \( \Delta_2 = (k_i \mod 3) - (k \mod 3) = 0, \) \( \beta' = \beta - \Delta_1 = 19, \) and \( \gamma' = \gamma - \Delta_2 = 25 \) are calculated. In step 4, the stego-pixels become \( \hat{\alpha} = 20, \) \( \hat{\beta} = 19, \) and \( \hat{\gamma} = 25. \)

When embedding \( k'_1 = 1 \) into \( (\alpha', \beta, \gamma') = (23, 28, 27), \) as \( d_1 = 1, \) \( d_2 = 2, \) \( k = 3 \times d_1 + d_2 = 5, \) \( \Delta_1 = k_5 \lfloor k/3 \rfloor - \lfloor k/3 \rfloor = -1, \) and \( \Delta_2 = (k_i \mod 3) - (k \mod 3) = -1, \) we have \( \beta' = \beta - \Delta_1 = 29 \) and \( \gamma' = \gamma - \Delta_2 = 28. \) Note that, in the vector, \( \beta' = \beta + 1 \) and \( \gamma' = \gamma + 1. \) Consequently, in step 4, we have \( \hat{\alpha} = \alpha' = \alpha - 1 = 22, \) \( \hat{\beta} = \beta' = \beta = 28, \) and \( \hat{\gamma} = \gamma' = \gamma = 27. \)

Similarly, since \( k'_2 = 6, (\alpha, \beta, \gamma) = (2, 254, 1), \) \( d_1 = 0, d_2 = 1, k = 3 \times d_1 + d_2 = 1, \) \( \Delta_1 = k_6 \lfloor k/3 \rfloor - \lfloor k/3 \rfloor = 2, \) and \( \Delta_2 = (k_2 \mod 3) - (k \mod 3) = -1, \) we have \( \beta' = \beta + 1 = 255 \) and \( \gamma' = \gamma - \Delta_2 = 2. \) The stego-pixels of the last vector are \( \hat{\alpha} = 31, \hat{\beta} = 26, \) and \( \hat{\gamma} = 33. \)

Finally, when embedding \( k'_2 = 7 \) into the last vector, as \( d_1 = 1, d_2 = 0, k = 3, \Delta_1 = 1, \) and \( \Delta_2 = 1, \) we have \( \beta' = \beta - 1 = 25 \) and \( \gamma' = \gamma - 1 = 32. \) Therefore, the stego-pixels of the last vector are \( \hat{\alpha} = 31, \hat{\beta} = 26, \) and \( \hat{\gamma} = 33. \)

To reduce image distortion when embedding a sub-message into a vector, a two-step refining process for stego-pixels is applied. The first refining step is injected in step 3 in the embedding process if a difference value between two pixels would be increased or decreased by two. After applying the refining process, the difference value is only decreased or increased by one. For example, if \( a \) and \( b \) are two pixel values and their difference would be increased or decreased by two, we have \( (a - b + 2) \mod 3 = (a - b - 1 + 3) \mod 3 = (a - b - 1) \mod 3 \equiv 2. \)

---

**Fig. 1.** An embedding example: (a) cover image; and (b) stego-image.

**Fig. 2.** Examples illustrating the refining process.
Fig. 3. Test images: (a)Lena; (b)Baboon; (c)F16; (d)Barbara; (e)Sailboat; (f)Pepper; (g)Boat; and (h)Goldhill.
mod 3 or \((a - b - 2) \mod 3 = (a - b + 1 - 3) \mod 3 = (a - b + 1) \mod 3\), respectively. The second refining step is injected in step 4 in the embedding process. If \(a\) and \(b\) would be decreased to \(a-1\) and \(\gamma - 1\), respectively, the refining process simply increases \(a\) to \(a+1\) and leaves \(b\) and \(\gamma\) unchanged, since \((a - (b - 1)) = ((a+1) - b)\)
and \((a - (\gamma - 1)) = ((a+1) - \gamma)\). Similarly, if they would be increased to \(b+1\) and \(\gamma + 1\), respectively, it simply decreases \(a\) to \(a-1\) and also leaves \(b\) and \(\gamma\) unchanged.

Fig. 2 is an example illustrating the contributions of the proposed scheme. Cover pixels are shown in Fig. 2(a), where \(a-b = -4\) and \(a-\gamma = -7\). In Fig. 2(b), both \(a\) and \(\gamma\) are decreased by one. In Fig. 2(c), \(a\) is increased by one, and \(b\) and \(\gamma\) are unchanged. However, the difference between \(a\) and \(b\) and that between \(a\) and \(\gamma\) in Fig. 2(b) are equal to those in Fig. 2(c), respectively. Note that two pixels are adjusted in Fig. 2(b). Nevertheless, to obtain the same differences as those in Fig. 2(c), only one pixel needs to be adjusted in Fig. 2(c). In contrast, if the absolute difference between \(a\) and \(b\) and that between \(a\) and \(\gamma\) would be increased by one as shown in Fig. 2(d), we can simply decrease \(a\) by one and leave \(b\) and \(\gamma\) unchanged as shown in Fig. 2(e). We can see that the values of \(a-b\) in Fig. 2(d) and (e) are equal and those of \(a-\gamma\) are also equal in the two figures.

Similarly, when \(\beta \leq a \leq \gamma\) (or \(\gamma \leq a \leq \beta\)) as shown in Fig. 2(f), increasing \(\beta\) and \(\gamma\) each by one, as shown in Fig. 2(g), to adjust the difference between \(a\) and \(\beta\) and that between \(a\) and \(\gamma\) is equivalent to decreasing \(a\) by one as shown in Fig. 2(h). Fig. 2(i) shows the results when \(\beta\) and \(\gamma\) in Fig. 2(f) would be decreased by one. Instead of decreasing \(\beta\) and \(\gamma\) each by one in Fig. 2(i), we can simply increase \(\alpha\) by one and leave \(\beta\) and \(\gamma\) unchanged, as shown in Fig. 2(j). The refining process is also workable if \(\gamma \leq \beta \leq a\) or \(\beta \leq \gamma \leq a\). With applying the refining process, a stego-image is more similar to its cover image than without applying it.

### Table 1
Comparisons of PSNR values (dB) for an embedding capacity of 1 bpp.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Lena</td>
<td>51.1302</td>
<td>52.3849</td>
<td>51.1302</td>
<td>52.6799</td>
</tr>
<tr>
<td>Baboon</td>
<td>51.1417</td>
<td>52.3942</td>
<td>51.1417</td>
<td>52.6854</td>
</tr>
<tr>
<td>F16</td>
<td>51.1458</td>
<td>52.3945</td>
<td>51.1458</td>
<td>52.6742</td>
</tr>
<tr>
<td>Barbara</td>
<td>51.1381</td>
<td>52.3847</td>
<td>51.1381</td>
<td>52.6843</td>
</tr>
<tr>
<td>Sailboat</td>
<td>51.1328</td>
<td>52.3866</td>
<td>51.1328</td>
<td>52.6816</td>
</tr>
<tr>
<td>Pepper</td>
<td>51.1312</td>
<td>52.3910</td>
<td>51.1312</td>
<td>52.6856</td>
</tr>
<tr>
<td>Boat</td>
<td>51.1297</td>
<td>52.3816</td>
<td>51.1297</td>
<td>52.6705</td>
</tr>
<tr>
<td>Goldhill</td>
<td>51.1374</td>
<td>52.3847</td>
<td>51.1374</td>
<td>52.6769</td>
</tr>
</tbody>
</table>

### Table 2
Comparisons of PSNR values (dB) for an embedding capacity of \((\log_2 9)/3 = 1.05664\) bpp.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Lena</td>
<td>50.3742</td>
<td>–</td>
<td>50.7470</td>
<td>52.4392</td>
</tr>
<tr>
<td>Baboon</td>
<td>50.4041</td>
<td>–</td>
<td>50.7580</td>
<td>52.4573</td>
</tr>
<tr>
<td>F16</td>
<td>50.4047</td>
<td>–</td>
<td>50.7583</td>
<td>52.4349</td>
</tr>
<tr>
<td>Barbara</td>
<td>50.3914</td>
<td>–</td>
<td>50.7531</td>
<td>52.4431</td>
</tr>
<tr>
<td>Sailboat</td>
<td>50.3776</td>
<td>–</td>
<td>50.7485</td>
<td>52.4616</td>
</tr>
<tr>
<td>Pepper</td>
<td>50.3890</td>
<td>–</td>
<td>50.7454</td>
<td>52.4471</td>
</tr>
<tr>
<td>Boat</td>
<td>50.3753</td>
<td>–</td>
<td>50.7426</td>
<td>52.4385</td>
</tr>
<tr>
<td>Goldhill</td>
<td>50.3849</td>
<td>–</td>
<td>50.7482</td>
<td>52.4488</td>
</tr>
</tbody>
</table>

Fig. 4. Test color-images: (a) Lena; (b) Baboon; (c) Tiffany; and (d) House.
3.2. Extracting a message

To extract the message embedded in the stego-image, the decoder divides the stego-image into vectors as the image was divided in the embedding process. Then, for each vector $i$, the decoder calculates

$$d_1 = (\hat{\alpha} - \hat{\beta}) \mod 3,$$

$$d_2 = (\hat{\alpha} - \hat{\gamma}) \mod 3,$$

and $k'_i = 3 \times d_1 + d_2$,

where $\hat{\alpha}$, $\hat{\beta}$, and $\hat{\gamma}$ are stego-pixel values for vector $i$. The integer $I$ can be calculated by

$$I = k'_{m-1} \times 9^{m-1} + k'_{m-2} \times 9^{m-2} + \ldots + k'_1 \times 9^1 + k'_0 \times 9^0.$$

Finally, the decoder converts $I$ into

$$I = s_{n-1} \times 2^{n-1} + s_{n-2} \times 2^{n-2} + \ldots + s_1 \times 2^1 + s_0 \times 2^0$$

and obtains the embedded binary message, excluding prefix 0's, $s_{n-1} s_{n-2} \ldots s_0$.

To extract the embedded message from the stego-image in Fig. 1(b), the decoder calculates $k'_0 = 3 \times ((20 - 19) \mod 3) + (20 - 25) \mod 3 = 4$, $k'_1 = 3 \times 0 + 1 = 1$, $k'_2 = 6$, and $k'_3 = 7$. The integer $I$ is obtained by calculating $I = 7 \times 9^3 + 6 \times 9^2 + 1 \times 9^1 + 4 \times 9^0 = 5602$. From the extra information which was provided by the encoder, the decoder is aware that there is one bit of prefix 0. Finally, $I$ is converted into $S = (0101011100010)_2$ and the embedded message is completely extracted.

Fig. 5. Stego-images of test color-images: (a) Lena; (b) Baboon; (c) Tiffany; and (d) House.

(a) (b) (c) (d)

Fig. 6. Visually comparing cover pixels with stego-pixels.

<table>
<thead>
<tr>
<th>(a)</th>
<th>(b)</th>
<th>(c)</th>
</tr>
</thead>
</table>
|\begin{tabular}{|c|c|c|}
\hline
\alpha & \beta & \gamma \\
\hline
0 & 0 & 0 \\
0 & 0 & +1 \\
0 & +2 & 0 \\
+1 & 0 & 0 \\
+1 & +1 & 0 \\
+1 & +2 & 0 \\
+2 & +1 & 0 \\
+2 & +2 & -1 \\
\hline
\end{tabular} & \begin{tabular}{|c|c|c|}
\hline
\alpha & \beta & \gamma \\
\hline
0 & 0 & 0 \\
0 & 0 & +1 \\
0 & 0 & -1 \\
0 & +1 & 0 \\
0 & +1 & +1 \\
0 & +1 & -1 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
+1 & 0 & 0 \\
\hline
\end{tabular} & \begin{tabular}{|c|c|c|}
\hline
\alpha & \beta & \gamma \\
\hline
0 & 0 & 0 \\
0 & 0 & +1 \\
0 & 0 & -1 \\
0 & 0 & +1 \\
0 & 0 & +1 \\
-1 & 0 & 0 \\
1 & 0 & 0 \\
1 & 0 & 0 \\
+1 & 0 & 0 \\
\hline
\end{tabular} |

Fig. 7. Adjustments of $\alpha$, $\beta$, and $\gamma$ for embedding a desired sub-message: (a) simple adjustment; (b) in step 3; and (c) in step 4.
4. Experimental results

To evaluate the performance of the proposed scheme, a Java program was developed on a desktop computer. Eight grayscale images, as shown in Fig. 3, were selected as test images. The dimension of each test image is 512-pixel × 512-pixel and all pixel values are between 0 and 255. Peak signal to noise ratio (PSNR) is a popular criterion for measuring the similarity between two images [19,20]. Most studies in the information hiding field use PSNR to evaluate whether a stego-image is similar to its cover image or not. In the paper, the visual quality of a stego-image with N pixels was also evaluated by PSNR defined as

\[
\text{PSNR} = 10 \times \log_{10} \left( \frac{255 \times 255}{\text{MSE}} \right),
\]

where MSE denotes mean square error calculated by

\[
\text{MSE} = \frac{1}{N} \sum_{j=1}^{N} (x_j - \hat{x}_j)^2,
\]

where \( x_j \) and \( \hat{x}_j \) are the values of pixel \( j \) in the cover image and stego-image, respectively. A randomly generated bit string was used to simulate the message to be embedded.

Table 1 shows the comparisons of PSNR values (dB) for an embedding capacity of one bit per pixel (bpp). When embedding a bit of message into a pixel, the OPAP [18] degrades to a method of simple LSB substitution. Therefore, in the table, the PSNR values of an image using simple LSB substitution and the OPAP to embed the same message are equal. Mielikainen’s method can obtain a better image quality than both simple LSB substitution and the OPAP on the basis of an embedding capacity of one bpp. In Table 1, the image quality of an image using the proposed scheme to embed a message is better than the others.

Table 2 shows the comparisons of PSNR values (dB) for an embedding capacity of \((\log_29)/3\) = 1.05664 bpp. In the table, the PSNR values for Mieliikainen’s method [17] is not available, since the maximum embedding capacity of an image using Mieliikainen’s method is not more than one bpp.

In addition to the above test images, more standard grayscale images from Ref. [21], including texture and aerial images, were also tested by applying the proposed scheme to embed messages into them. The PSNR values are almost equal to those in Tables 1 and 2 when embedding the same amount of data as those in Tables 1 and 2, respectively. This shows that the proposed scheme can be applied to various images.

To show that the proposed scheme can be applied to true-color images, randomly generated messages were embedded into the four test color-images (also from Ref. [21]) shown in Fig. 4(a)–(d). In the above color-images, a pixel is made up of three components: Red, Green, and Blue, and each component is an 8-bit unsigned integer denoting its intensity. Therefore, the payload capacity of a true-color image is up to three times of a grayscale one if their number of pixels is equal. In the experiment, a color pixel was regarded as a vector with three components. The embedding results (stego-images) of Fig. 4 are shown in Fig. 5(a)–(d), respectively. The PSNR value of Fig. 5(a) is 52.4402 dB and the image is visually indistinguishable from its cover one in Fig. 4(a). Fig. 5(b)–(d) has similar image quality to that in Fig. 5(a).

The colors of the four pixels from left side of the top one line in Figs. 4(a) and 5(a) are shown in Fig. 6, where pixel values are neighboring their colors. To compare the similarity between stego and cover pixels, stego-pixels are arranged below their cover ones, respectively. The first stego-pixel (226, 137, 124) embeds a sub-message of 6, since \( 3 \times ((226 - 137) \mod 3) + (226 - 124) \mod 3 = 6 \). The remaining three pixels embed a sub-message of 5, 1, and 4, respectively. The figure shows that the pixel values of the first two cover pixels are equal. However, their stego-pixel values are different, since they embed different sub-messages. Obviously, these four stego-pixels are visually indistinguishable from their cover ones, respectively.

5. Discussion

The proposed scheme embeds a sub-message in a vector by adjusting the difference between \( \alpha \) and \( \beta \) and that between \( \alpha \) and \( \gamma \). If the reﬁning process is not applied, each difference may be unchanged or adjusted by one or two so that the two differences in a vector can embed any sub-message \( k_i \), where \( 0 \leq k_i \leq 8 \). A set of feasible adjustments, called simple adjustment, of pixels is listed in Fig. 7(a).

In step 3 of the embedding process, if \( \beta \) or \( \gamma \) needs to be increased by two to obtain a desired value of difference, the proposed scheme subtracts one from \( \beta \) or \( \gamma \) to obtain an equivalent desired difference value with modulo 3. Details are shown in Fig. 7(b). In step 4 of the embedding process, if \( \beta \) and \( \gamma \) would be adjusted to \( \beta + 1 \) and \( \gamma + 1 \), respectively, the proposed scheme subtracts one from \( \alpha \) to replace the adjustments of \( \beta \) and \( \gamma \). Similarly, if \( \beta \) and \( \gamma \) would be adjusted to \( \beta - 1 \) and \( \gamma - 1 \), respectively, it adds one to \( \alpha \) and leaves \( \beta \) and \( \gamma \) unchanged. Details are shown in Fig. 7(c).

From Fig. 7(c), when applying the proposed scheme to embed a randomly generated message into an image, the probability of modifying a pixel by one can be calculated by \( 10/27 = 0.3704 \). If the message to be embedded is divided into the sub-messages shown in step 1 in the embedding process, on average, each vector in a cover image can embed \( \log_29 = 3.169925 \) bits of message. If an embedding capacity of one bpp is desired, the probability of modifying a pixel by one can be calculated by \( (10/27)/(\log_29/3) = 0.350517 \), which is smaller than that in Mieliikainen’s method.

In the case of an embedding capacity of one bpp, the PSNR value of an image using the proposed scheme can be estimated by

\[
\text{PSNR} = 10 \times \log_{10} \left( \frac{255 \times 255}{0.350517} \right) = 52.6837 \text{ dB}
\]

and that using Mieliikainen’s method can be estimated by

\[
\text{PSNR} = 10 \times \log_{10} \left( \frac{255 \times 255}{0.375} \right) = 52.3905 \text{ dB}
\]

Thus the proposed scheme can obtain a stego-image with higher visual quality compared to Mieliikainen’s method.

6. Conclusions

A good embedding scheme can provide required embedding capacity for applications and obtain a stego-image with higher visual quality so that the stego-image is visually indistinguishable from its cover image. When hiding a message into an image, an important issue is that the embedded message would be imperceptible to the human eye. Otherwise, the embedded message may be disclosed by third parties. We have proposed an embedding scheme which can embed more than one bit per pixel into an image. In the proposed scheme, each pixel in the stego-image is either remained unchanged or modified by one. Besides, the number of pixels modified in a three-pixel vector is not more than two pixels. Therefore, the modification of each pixel is minimal and it may not attract attention from a malicious attacker. The experimental results have demonstrated that the performance of the proposed scheme outperform Mieliikainen’s [17] method and Chan and Cheng’s approach [18], including embedding capacity and visual quality of an image.
References


http://sipi.usc.edu/database/

Ching-Chiuau Lin received his Ph.D. degree in Information Engineering and Computer Science from Feng Chia University, Taichung, Taiwan, in 2008. He is an Associate Professor with the Department of Information Management, Overseas Chinese University. His research interests include image processing, data hiding, network computing, and software engineering. Dr. Lin was selected as an honorary member of Phi Tau Phi Scholastic Society of The Republic of China in 2008.