Image hiding by base-oriented algorithm

Ching-Yu Yang
National Penghu University
Department of Computer Science and Information Engineering
Penghu, Taiwan, 880
E-mail: chingyu@npu.edu.tw

Ja-Chen Lin
National Chiao Tung University
Department of Computer and Information Science
Hsinchu, Taiwan, 300

Abstract. We propose a method called the base-oriented algorithm for image hiding. The method classifies each block of the host image H according to the base value (BV) that represents the block's variation. The classified blocks then either uses BV embedding or module substitution to hide data. The hiding capability of the proposed method is high, whereas the visual performance is also good. The lossless extraction of the confidential image from the mixed image is simple and uses no look-up table. The method can be used as an economic tool to hide and store confidential data in images. © *2006 Society of Photo-Optical Instrumentation Engineers.* [DOI: 10.1117/1.2395925]

Subject terms: classified blocks; base value embedding; module substitution.

Paper 050432RR received Jun. 15, 2005; revised manuscript received Apr. 20, 2006; accepted for publication May 2, 2006; published online Nov. 22, 2006.

1 Introduction

Information hiding is a technology used to hide data such as a confidential image, ownership protection, and fingerprints in a variety of forms of media such as image, video, and audio.[1,2] A number of techniques have been presented.[3–19] In the paper, we focus on hiding data in images; more specifically, we will try to embed a large amount of information (for example, a confidential image) into a so-called host image H, which is an ordinary image among many other ordinary images, to produce a mixed image H^*. The goal is to achieve lossless and high-capacity hiding of the information, while requiring that the distortion to the host image H should be very small to reduce people's attention. Its application is a space-saving tool for the hiding and storage of confidential data.

Chen et al.[12] proposed a method to hide images based on vector quantization. The method is an elegant method for image hiding. However, the method requires a look-up table to extract the hidden confidential image from the mixed image, and the extraction is lossy. Hu and Lin[13] presented another kind of image hiding scheme. Since the confidential images were compressed by vector quantization before the embedding process, the extracted image is also not error free. Wang et al.[14] developed a technique to hide confidential data by using least significant bit (LSB) substitution and a genetic algorithm. Compared with the simple LSB approach, their method improved the perceptual quality of the mixed images. But the computational complexity is high. Chang et al.[15] suggested two graceful hiding schemes. The visual performance and security generated by both schemes are very good. However, the hiding capacity is not very high. In summary, we propose here a simple method for image hiding. The recovery of the hidden information is lossless, the hiding capability is high, and the visual quality of the mixed image is also good.

The rest of the paper is organized as follows. The base value (BV) embedding scheme is introduced in Sec. 2.1. The module substitution technique, which handles the blocks rejected by the BV embedding scheme, is specified in Sec. 2.2. Section 2.3 gives the algorithm, combining the preceding two sections. Some properties of embedding distortion are discussed in Sec. 2.4. Section 2.5 presets the extraction of the hidden data. Security enhancement is discussed in Sec. 2.6 and experimental results are given in Sec. 3. Several reported data-hiding techniques are compared with our method in Sec. 4. Section 5 gives a brief conclusion.

2 Proposed Method

The proposed hiding method is called base-oriented because it classifies each block (of the host image H) according to its BV, which represents the block's variation. If the block's BV is in a predefined interval $(0.5\tau, \tau)$, then the method tries to embed confidential data into the block by using the BV. If the BV of the block is out of the predefined interval, however, then the method embeds confidential data into the block by using module function.

2.1 *BV Embedding Technique*

Assume a confidential image is the data to be embedded. Treat the confidential image as a (very long but finite) binary sequence E. The sequence is decomposed into many (much shorter) bit streams of nonfixed lengths later for hiding purposes. Let H be a host image of size $M \times N$, and let H^* be the mixed image of size $M \times N$ obtained after hiding data in H. Partition H into nonoverlapping blocks of size $n \times n$. Each time we pick up the next not-yet-processed block of H, we try to hide some data in it. The BV embedding technique consists of two possibilities: (1) without BV adjustment (for the blocks whose natural base values are already acceptable) and (2) with the BV adjustment to create an acceptable artificial base values.

2.1.1 *Part I: without BV adjustment*

Let $H_k = \{q_{kj}\}_{j=0}^{(n\times n)-1}$ be the k'th $n \times n$ block of H, and let q_{kj} the j'th pixel value in block H_k. Define the "minimum" pixel value m and BV b of the block by

$$m = m_k = \min_{0 \le j \le n \times n-1} q_{kj}, \tag{1}$$

and

$$b = b_k = \max_{0 \le j \le n \times n-1} q_{kj} - \min_{0 \le j \le n \times n-1} q_{kj} + 1. \tag{2}$$

If the value b_k does satisfy $0.5\tau < b_k < \tau$, where τ is a predefined number, then the BV embedding scheme is applied; otherwise, the block will be processed by the module substitution method introduced in Sec. 2.2.

The main idea of the BV embedding scheme is the embedding replacement of the value-reduced block $\{q_{kj} - m\}_{j=0}^{(n\times n)-1}$ of H_k by a bit stream taken from data. More specifically, let the binary bit stream to be embedded into H_k be $B = \{\beta_i\}_{i=0}^{L-1} = \Sigma_{i=0}^{L-1} \beta_i \cdot 2^i$. (The value of L is evaluated using Property 1 stated later.) Transform B into a base-b_k number $B = \Sigma_{j=0}^{n^2-1} r'_j b_k^j = (r'_{n^2-1}, \ldots, r'_1, r'_0)_{b_k}$. The resulting k'th block $H_k^* = \{r_{kj}\}_{j=0}^{n^2-1}$ of the mixed image H^* is obtained by adding m to each coefficient $\{r'_j\}_{j=0}^{n^2-1}$ of B, respectively; i.e., $\{r_{kj}\}_{j=0}^{n^2-1} = \{r'_j + m\}_{j=0}^{n^2-1}$ are the pixel value in H_k^*.

Now, if the base value of the block is unchanged, i.e., if

$$c_k = \max_{0 \le j \le n \times n-1} r_{kj} - \min_{0 \le j \le n \times n-1} r_{kj} + 1. \tag{3}$$

of H_k^* still has value b_k, then we say b_k is "acceptable" because the decoder can extract later the hidden bit stream easily. This is so because the decoder must evaluate only Eq. (3) to obtain the base value c_k, which equals b_k, then treat the n^2 digit number

$$\{r_{ki} - \min_{0 \le j \le n \times n-1} r_{kj}\}_{i=0}^{n \times n-1} \tag{4}$$

as a base-b_k number, and convert this base-b_k number to the expected base-2 number B.

If b_k is not acceptable, i.e., if $c_k \neq b_k$, then we cannot recover B because we cannot recover the value b_k required to convert the base-b_k number to the equivalent. In this case, we restart the procedure handling H_k by using some other possible base values b'_k introduced in Sec. 2.1.2.

When $c_k = b_k$, the number of bits (in the binary data E) can be embedded in H_k is evaluated using the following property.

Property 1. If the $n \times n$ block H_k is an acceptable block with acceptable base value b_k, then the number of bits hidden in the block is $L = \lfloor n^2 \log_2 b_k \rfloor$.

Proof. The largest n^2-digit number in the base-b_k number system is $(b_k - 1, b_k - 1, \ldots, b_k - 1)_{b_k}$, whose decimal value is $\Sigma_{j=0}^{n^2-1} (b_k - 1) b_k^j = (b_k - 1) \; b_k^{n^2} - 1 / b_k - 1 = (b_k^{n^2} - 1)$. Similarly, the largest L-bit binary number is $(1, 1, \ldots, 1)_2$, whose decimal value is $2^L - 1$. So that $2^L - 1$ can be stored as a base-b_k number using n digits, we require $2^L - 1 \le b_k^{n^2} - 1$, i.e., $2^L \le b_k^{n^2}$. In other words, $L \le n^2 \log_2 b_k$. Now, the maximal integer L satisfying $L \le n^2 \log_2 b_k$ is $L = \lfloor n^2 \log_2 b_k \rfloor$.

Property 1 implies that the length of bits to be embedded into block H_k is determined by the base value b_k (if $c_k = b_k$). Thus, blocks at a distinct area of H might embed a different length of bits, because the BVs often vary. In the following we illustrate our idea using an example.

100	101	103
100	101	103
100	101	103

(a)

1	0	0
0	0	2
0	0	3

(b)

101	100	100
100	100	102
100	100	103

(c)

Fig. 1 Example of embedding a bit stream into a block, say, the k'th block, according to the base value of the block. This example needs no BV adjustment because (a) and (c) have identical base value, namely 4. (a) Block in the host image H; (b) the coefficients $\{r'_j\}_{j=0}^8$ of $B = \Sigma_{j=0}^8 r'_j b^j$, which was obtained from the embedding bits, and b is the BV of the block in (a), i.e., $b = 103 - 100 + 1 = 4$; and (c) the resulting mixed block $\{r_{kj}\}_{j=0}^8$ in C, where $r_{kj} = r'_j + m = r'_j + 100$.

Example 1 (see Fig. 1). Let Fig. 1(a) be a 3×3 block in the host image H. The base value of this block is $b = 103 - 100 + 1 = 4$. The length of bits can be embedded in the block is therefore $L = \lfloor n^2 \log_2 b \rfloor = \lfloor 9 \log_2 4 \rfloor = 18$. Also, assume that the security bit stream $B = \{\beta_i\}_{i=0}^{L-1} = (0100000000\ 10000011)_2$ is the 18-bit stream to be embedded in Fig. 1(a). If we treat B as a number [whose decimal value is $B = \Sigma_{i=0}^{L-1} \beta_i \cdot 2^i = 2^{16} + 2^7 + 2^1 + 2^0 = (65{,}647)_{10}$], then we can transform B into its base-4 equivalent, namely, $B = (65{,}647)_{10} = \Sigma_{j=0}^8 r'_j b^j = 1 \times 4^8 + 2 \times 4^3 + 3 \times 4^0 = (100002003)_4$. The resulting H_k^* block [Fig. 1(c)] is obtained by adding the minimum value $m = 100$ of Fig. 1(a) to each coefficients $\{r'_j\}_{j=0}^8$ of B [see Fig. 1(b)], respectively. Note that the base value of the mixed block in Fig. 1(c) is still 4. Hence, the binary embedded data B can be extracted successfully from this block. The mean square error (MSE) for the two blocks between Figs. 1(c) and 1(a) is $\text{MSE}_p = (1/9)\Sigma_{j=0}^8 (r_{kj} - q_{kj})^2 = 1/9(1 + 1 + 9 + 0 + 1 + 1 + 0 + 1 + 0) = 14/9$.

As remarked earlier, we predetermine a control parameter value τ, and require that $0.5\tau < b_k < \tau$. When the data are to be hidden in host blocks, any host block whose base value b_k satisfying $b_k \ge \tau$ or $b_k \le 0.5\tau$ will be encoded by the module substitution method introduced later in Sec. 2.2, rather than by the preceding BV embedding scheme using the BV of the block. Note that, although τ can be chosen as any positive integer greater than 1, we suggest that the readers use a τ in $4 \le \tau \le 12$. If $\tau < 4$, then almost none of the blocks can have an acceptable BV (even after the BV adjustment introduced in the next section). On the other hand, if $\tau > 12$, then the distortion to the pixels might be too large.

2.1.2 *Part II. BV adjustment*

As stated just before Property 1, if $c_k \neq b_k$, the resulting block cannot extract the hidden bit stream from itself. To overcome the $c_k \neq b_k$ problem, a BV adjustment policy is introduced here. The idea is try to dig out the possibility of using an artificial base to hide data. In other words, we try

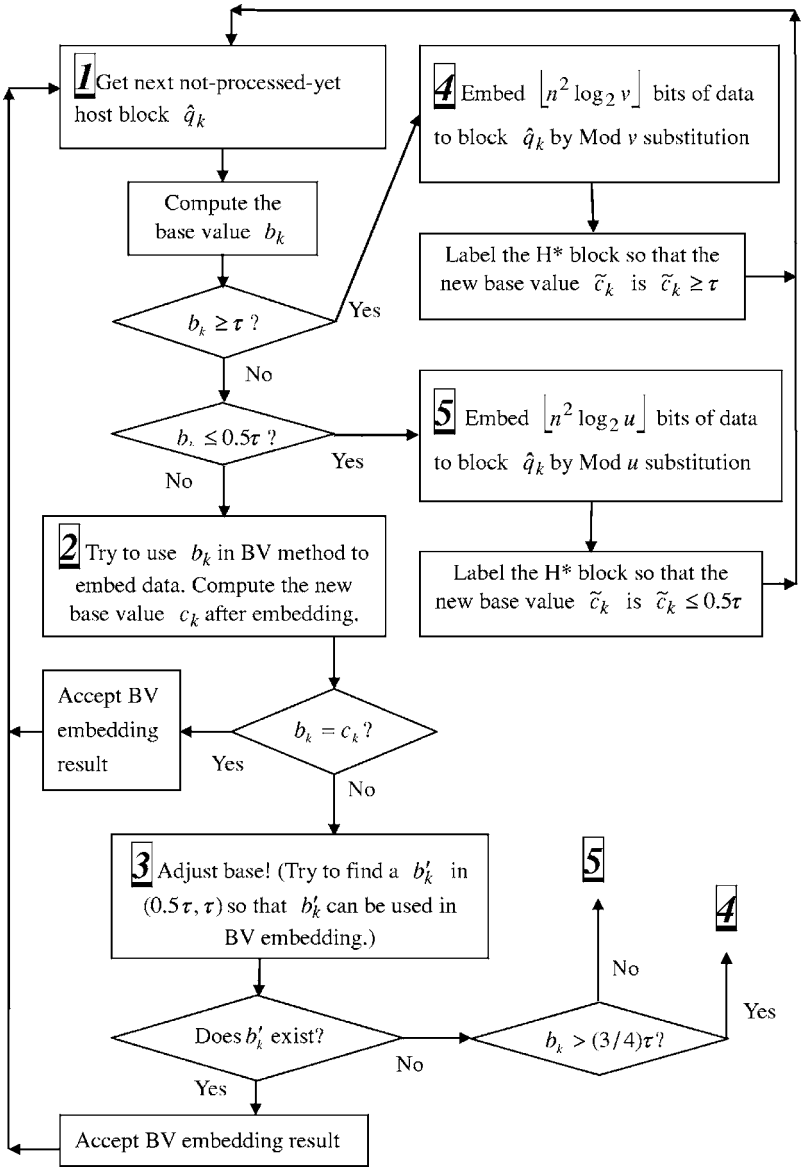


Fig. 2 Encoding scheme of the proposed method.

to determine whether there exists a new base b'_k so that $c_k = b'_k$ after we hide binary data in the block by using artificial base b'_k (rather than by the natural base b_k). The search of this b'_k is proceeded alternatively in two directions according to the order: b_k-1, b_k+1, b_k-2, b_k+2,.... (Of course, b'_k still has to be in the range $0.5\tau < b'_k < \tau$.)

Once an acceptable base value b'_k can be found, the number of embedded bits turns from original L to $L' = \lfloor n^2 \log_2 b'_k \rfloor$. Notably, if an acceptable b'_k is used to embed data, the minimal value of the host block will be identical to the minimal value of the mixed block, i.e., the minimal value is preserved in the embdding. The proof is trivial and omitted. If no value in the set $\{b_k \mp 1, b_k \mp 2, \ldots\} \cap \{0.5\tau < b'_k < \tau\}$ can be used as a b'_k to encode the bit stream of length L', then the block is forced to be encoded by the module substitution technique discussed next.

2.2 *Module Substitution Scheme*

If the BV b_k of a host block satisfies $b_k \notin (0.5\tau, \tau)$, or if the BV is nonacceptable even after BV adjustment (i.e., $c_k \neq b_k$ and $c_k \neq b'_k$), then the block will hide data bits using the module substitution scheme.

Assume that a module substitution, namely, mod u substitution, is to be applied to hide data. Take $\lfloor (n \times n) \log_2 u \rfloor$ bits from the binary data. Convert this binary number to a (base u) value of n^2 digits, and each digit is in the range $\{0, 1, 2, \ldots, u-1\}$. To hide a data digit $d \in \{0, 1, 2, \ldots, u-1\}$ in a pixel with gray value x, we first evaluate $g_0 = (x - x_{\mathrm{mod}u}) + d$. Then let $g_0^+ = g_0 + u$ and $g_0^- = g_0 - u$. Choose from $\{g_0^-, g_0, g_0^+\}$ the one whose distortion to x is the smallest, and

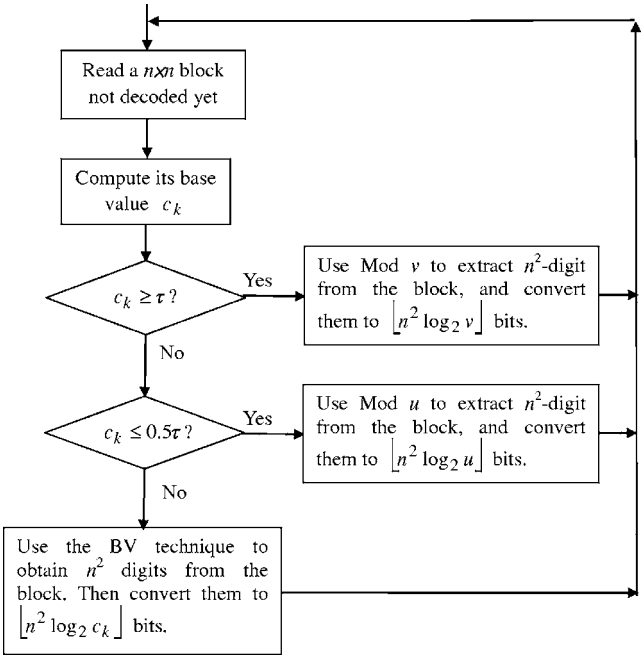


Fig. 3 Decoding scheme of the proposed method.

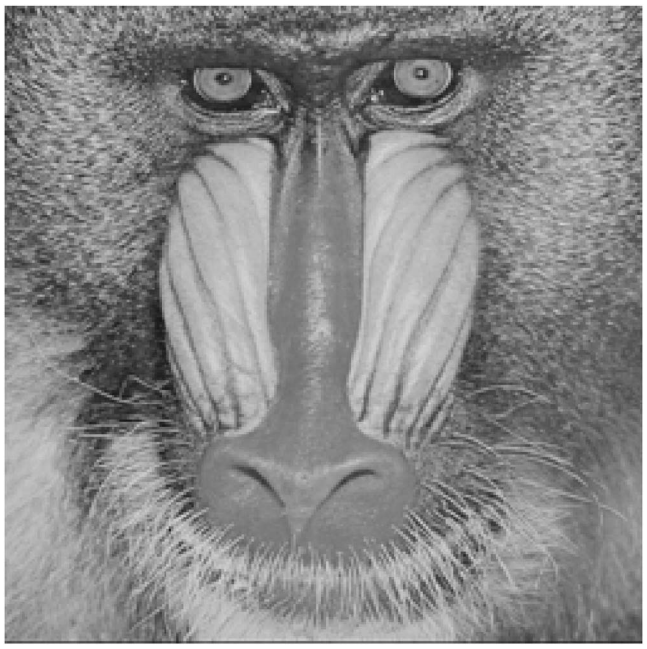


Fig. 4 Secret image "Baboon" (256×256).

call it g. Then, replace the gray value x of the host pixel by the new value g. Note that the data digit d can be extracted later easily because $d=(g)_{\mathrm{mod}u}=(g_0)_{\mathrm{mod}u}$.

After the embedding, to help the decoder identify the block type, we must label the resulting mixed block so that the new base value of the block is out of the range $(0.5\tau, \tau)$. To do labeling, we add or subtract full multiples of u to or from the gray values of certain pixels. This adding u or subtracting u action will not affect the value of the extracted data, because the decoder will take $\mathrm{gray}_{\mathrm{mod}u}$ as the extracted value for each pixel of the mixed block, and $g_{\mathrm{mod}u}=(g\pm u)_{\mathrm{mod}u}$ for any value g.

2.3 *Whole Encoding Algorithm*

Sections 2.1 and 2.2 can be combined and summarized as follows:

1. *Input*: Input is a finite binary bit sequence (representing a security image E), an $M\times N$ host image H divided into a series of nonoverlapping blocks of size $n\times n$ (say, 3×3), a control parameter τ, and two integers $\{u\leqslant v\}$ for module substitution use.
2. *Output*: Output is an $M\times N$ image H^* containing E.
3. *Steps*: Figure 2 illustrate the steps. The left (right) part of Fig. 2 corresponds to Sec. 2.1 and 2.2, respectively.

Note that when $\tau<6$, then it is often too hard to do the labeling task after mod u substitution (to make the new base value of the resulting H^* block be at most 0.5τ). Therefore, we do not have step 5 (mod u substitution) if $\tau<6$. Only BV embedding and mod v substitution are used. The BV adjustment performed in this case is with the b'_k searched in $(1,\tau)$ rather than $(0.5\tau,\tau)$. If no acceptable b'_k can be found in $(1,\tau)$, just go to step 4 to use mod v substitution.

2.4 *Properties Analysis*

Some properties of the proposed method are analyzed in this subsection.

Lemma 1. Let b_k be the natural base value of the k'th block in the host image. If b_k is acceptable (or, if b'_k is acceptable after BV adjustment), then the MSE for the $n\times n$ block between the host image H and the mixed image H^* cannot exceed $(b_k-1)^2$ [or, respectively, cannot exceed $(b'_k-1)^2$ if the BV adjustment is used]. Therefore, the MSE of the block cannot exceed $(\tau-2)^2$, since we require both $b_k<\tau$ and $b'_k<\tau$.

Proof. Without the loss of generality, we only prove the case in which the base value b_k is used in the BV embedding technique. As for the case in which the embedding is with adjusted BV b'_k, the proof is similar.

The MSE of the k'th block in host image is

$$\mathrm{MSE}_k=\frac{1}{n\times n}\sum_{j=0}^{n^2-1}(r_{kj}-q_{kj})^2, \tag{5}$$

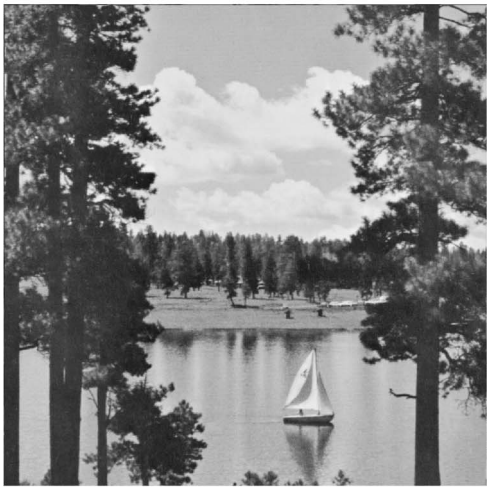
where r_{kj} and q_{kj} are the j'th pixel values of the k'th blocks of H^* and H, respectively. In addition, r_{kj} and q_{kj} can be rewritten as

$$r_{kj}=m_k+r'_{kj} \text{ and } q_{kj}=m_k+q'_{kj}, \tag{6}$$

respectively, with

(a)

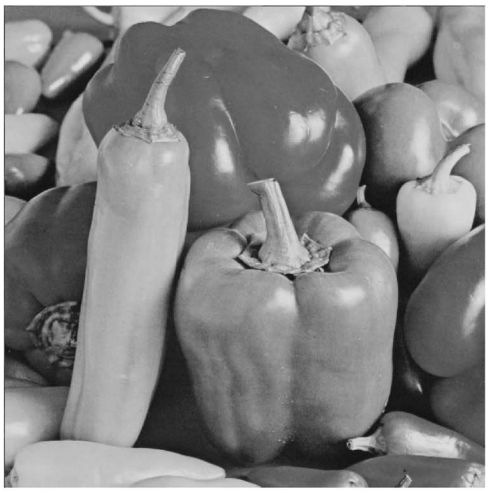
(b)



(c)

Fig. 5 Three 512×512 host images H: (a) "Lena," (b) "Peppers," and (c) "Sailboat."

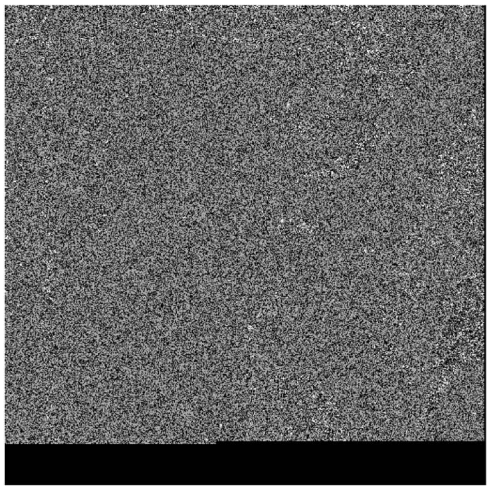
(a)



(b)

(c)

Fig. 6 Three mixed images H^* generated by hiding Fig. 4 in one of the three host images in Fig. 5, respectively.



(a)

(b)

(c)

Fig. 7 Absolute value difference images $|H\text{-}H^*|$: (a) the one whose H^* [Fig. 6(a)] is obtained using (τ=6, u=3, v=4), and the PSNR is 45.99, while the hiding rate is 0.2483; (b) the one from another experiment in which H^* is obtained using (τ=12, u=3, v=16), and the PSNR is 36.50, while the hiding rate is 0.3876; and (c) the H^* mentioned in (b).

Table 1 Experimental results when $\tau \geqq 6$. A portion of a 512×512 secret image "Baboon" is hidden in a host image 512×512 "Lena." Since τ is at least 6, both mod u and mod v substitution are used, along with the BV embedding; u=3 in this table.

v	τ	PSNR	Hiding Rate (%)
3	6	48.72	19.77
	8	45.84	21.33
	10	43.49	23.01
	12	41.96	23.96
4	6	45.99	24.83
	8	44.44	25.60
	10	42.78	26.50
	12	41.50	26.85
5	6	44.92	27.36
	8	44.76	27.73
	10	42.38	28.23
	12	41.24	28.31
6	6	43.13	31.15
	8	42.51	30.93
	10	41.58	30.90
	12	40.73	30.43
7	6	42.19	33.67
	8	41.80	33.07
	10	41.10	32.64
	12	40.42	31.87
8	6	40.89	36.20
	8	40.74	35.19
	10	40.30	34.41
	12	39.79	33.35
9	6	40.11	37.46
	8	40.11	36.28
	10	39.83	35.28
	12	39.46	34.04
12	6	37.59	42.53
	8	37.89	40.53
	10	38.00	38.78
	12	38.01	36.93

Table 2 Experimental results when τ is 4 or 5. A portion of a 512 $\times$512 secret image "Baboon" is hidden in a host image 512$\times$512 "Lena." Since $\tau<6$, there is no mod u operation in the hiding, just BV embedding and the mod v operation are used.

v	τ	PSNR	Hiding Rate (%)
3	4	49.88	19.28
	5	49.60	19.40
4	4	46.40	24.74
	5	46.33	24.72
5	4	45.17	27.47
	5	45.13	27.38
6	4	43.18	31.57
	5	43.19	31.37
7	4	42.18	34.38
	5	42.22	34.03
8	4	40.79	37.03
	5	40.86	36.70
9	4	39.95	38.40
	5	40.03	38.03
12	4	37.36	43.86
	5	37.44	43.35

Table 3 Experimental results when τ is 5. A portion of a 512$\times$512 secret image "Baboon" is hidden into a host image 512$\times$512 "Peppers" or "Sailboat." Since $\tau<6$, there is no mod u operation in the hiding; just BV embedding and the mod v operation are used.

v	τ	PSNR	Hiding Rate (%)
3	5	49.84 (Peppers)	19.32
	5	49.75 (Sailboat)	19.35
4	5	46.39 (Peppers)	24.79
	5	46.36 (Sailboat)	24.75
5	5	45.15 (Peppers)	27.52
	5	45.16 (Sailboat)	27.44
6	5	43.17 (Peppers)	31.62
	5	43.19 (Sailboat)	31.49
7	5	42.16 (Peppers)	34.36
	5	42.19 (Sailboat)	34.19
8	5	40.78 (Peppers)	37.09
	5	40.83 (Sailboat)	36.89
9	5	39.94 (Peppers)	38.46
	5	39.99 (Sailboat)	38.23
12	5	37.35 (Peppers)	43.92
	5	37.39 (Sailboat)	43.63

$$0 \leqslant r'_{kj} \leqslant b_k - 1, \quad 0 \leqslant q'_{kj} \leqslant b_k - 1, \tag{7}$$

and m_k is the minimum value of k'th block. (Since $c_k=b_k$, the minimum value of the host block is also the minimum value of the mixed block.) Substituting Eqs. (6) and (7) into Eq. (5), we obtain

$$\mathrm{MSE}_k = \frac{1}{n\times n}\sum_{j=0}^{n^2-1}(r_{kj}-q_{kj})^2 = \frac{1}{n\times n}\sum_{j=0}^{n^2-1}(r'_{kj}-q'_{kj})^2$$
$$\leqslant \frac{1}{n\times n}\sum_{j=0}^{n^2-1}(b_k-1)^2 = (b_k-1)^2.$$

Lemma 2. If a block of the host image H is encoded by the mod u substitution (or mod v substitution), then, before the labeling to mark the block type, the MSE of the $n\times n$ block (between the host block and the mixed block) cannot exceed $(0.5u)^2$ or $(0.5v)^2$, respectively.

Proof. Without the loss of generality, we prove only the mod u case. There are n^2 digits hidden in the n^2 pixels of the block, and each digit is in the range $\{0,1,2,\ldots,u-1\}$. As stated earlier, to hide a data digit $d\in\{0,1,2,\ldots,u-1\}$ in a pixel with gray value x, we first evaluate $g_0=(x-x_{\mathrm{mod}u})+d$. Then let $g_0^+=g_0+u$ and $g_0^-=g_0-u$. Then we choose from $\{g_0^-,g_0,g_0^+\}$ the one whose distortion to x is the smallest, and call it g. Then, use g to replace the gray value x. Since $(x-x_{\mathrm{mod}u})\leqslant g_0=(x-x_{\mathrm{mod}u})+d<(x-x_{\mathrm{mod}u})+u$ and $(x-x_{\mathrm{mod}u})\leqslant x<(x-x_{\mathrm{mod}u})+u$, both x and g_0 are in the interval $[x-x_{\mathrm{mod}u},x-x_{\mathrm{mod}u}+u]$. Therefore, $g_0^-=g_0-u$ is smaller than the left boundary of this interval of length u, while $g_0^+=g_0+u$ is greater than its right boundary. Now, if $x\leqslant g_0$, then we have $g_0^-=g_0-u\leqslant x\leqslant g_0$; so $\min\{|x-g_0|,|x-g_0^-|\}\leqslant 0.5u$. Similarly, if $g_0\leqslant x$, then we have $g_0\leqslant x\leqslant g_0^+=g_0+u$, so $\min\{|x-g_0|,|x-g_0^+|\}\leqslant 0.5u$. Together, we always have $|x-g|\leqslant 0.5u$. It is true for each pixel; so, MSE $\leqslant(0.5u)^2$ for the block.

In Lemma 2, after the module substitution, we have $|x-g|\leqslant 0.5u$ for each pixel (without the loss of generality, assume that mod u substitution is the one being discussed; the case for mod v can be analyzed likewise). But since we must still adjust some of the n^2 pixel values of the block so that the decoder can recognize the block type as a mod-u-type, we add or substract full multiples of u on some pixels' gray values g until the base value $(\max-\min+1)$ of the block is at most 0.5τ. This labeling work to mark the block type will affect the distortion of some (not all) pixels,

Table 4 The suggested combination of values of τ, u, and v at various hiding rates.

	Hiding Rate						
Parameters	0.15	0.20	0.25	0.31	0.35	0.40	0.45
τ	4	6	6	6	6	7	6
u	3	3	3	3	3	3	3
v	3	3	4	6	8	11	14

and $|x-(g\pm ku)| \leqslant 0.5u+ku$ if k times u is added to or subtracted from the gray value g in the labeling process. Because our τ, u, and v are all not too large [although there is no natural constraint (except that τ, u, and v should all be at least 2); we typically use $\tau=4,5,\ldots,12$; $u=2,3,4$; and $v=u,u+1,u+2,\ldots,12$]. As a result, the k is usually 0 or ± 1, and only some rare cases use $k=\pm 2$. Therefore, each block has small error; which causes good peak SNR (PSNR) value.

2.5 *Decoding Algorithm*

The decoding part is much simpler than the encoding part, and is self-explained in Fig. 3. (Assuming $\tau \geqq 6$, so the encoding is not the simplified version given at the end of Sec. 2.3.) Note that one cannot extract a correct image if she or he uses wrong values for τ, u, and v during the extraction.

2.6 *Security Enhancement*

To improve the security, instead of using a single and fixed value for the control parameter τ, a dynamic value for τ might also be employed in the proposed method. For example, a variety of τ, say, $\Phi=\{\tau_i\}_{i=1}^{|\Phi|}$ can be used to embed the data. These variant parameters $\{\tau_i\}$ can be cyclically used in a way that changes τ_i every other m blocks. Moreover, we can randomly choose a block as the leading block to start the hiding of data bits. In addition, it is not necessary to scan a $n\times n$ block according to rowwise, columnwise, or zigzag order. In fact, even the blocks' position or pixels' position of the whole image can be randomly perturbed by a random number generator using a secret key. All these approaches increase the difficulty for the adversary to grab the secret data from the mixed images.

3 Experimental Results

A 256×256 image "Baboon" (Fig. 4) is used as a test secret image. In each experiment, only one of the three 512×512 gray-scale images shown in Figs. 5(a)–5(c), namely, "Lena," "Peppers," and "Saiboat," is used as the host image H. The block size is $n\times n=3\times 3$. The generated mixed images are in Fig. 6. The control parameter τ used here was 6, while $u=3$ and $v=4$. From Fig. 6 we can see that the perceptual quality of the mixed images is good. Their PSNRs are 45.99 ("Lena"), 45.49 ("Peppers"), and 45.47 dB ("Sailboat"). The absolute value difference image $D=|H-H^*|$ is shown in Fig. 7(a). The absolute value difference at each pixel was amplified to show it clearly. The "brighter" pixel in D indicates bigger error there. The darkest pixel in D indicates no error there [for example, see the black horizontal band at the bottom of Fig. 7(a)]. It is because no more data are to be hidden there). From Fig. 7(a), we can see that, when $\tau=6$, $u=3$, and $v=4$ are used, the error is very small everywhere (this makes the PSNR become 45.99 dB); thus, there is no distinction between edge areas or smooth areas. On the other hand, when we do another experiment with larger τ and v, namely, $\tau=12$, $u=3$, and $v=16$, we yield an image "Lena*" [shown in Fig. 7(c)], whose PSNR is 36.50, and the hiding rate is 0.3876. The difference image $|H\text{-}H^*|$ in Fig. 7(b) has more distinction between uniform areas and nonuniform areas.

The experiments using different values of τ are shown in Tables 1–3. In Table 1, $\tau \geqq 6$. Thus, in addition to mod v substitution and the BV embedding technique, we also use mod u substitution, which is for the blocks with extremely small variation ($b_k \leqslant 0.5\tau$). The u used in Table 1 is 3. As for Tables 2 and 3, $\tau < 6$, so there is no mod u substitution (see the final paragraph of Sec. 2.3). From these three tables, we can say that, roughly speaking, smaller values of τ are better. In addition, when the value of τ is fixed, we can modify the value of v to change the hiding rate.

Here we discuss the roles of u and v. Since mod u substitution is originally used to handle extremely smooth blocks (their base values are at most 0.5τ), the value of u should be very small. Therefore, we set $u=2$, 3, 4. On the other hand, mod v substitution is primarily used to process high frequency area (the blocks whose base values are at least τ); as a result, $v \geqq u$. Moreover, since mod v is for the high-frequency area, and we know that high-frequency area can hide more data since human vision is less sensible to the change in it, we can use a large value of v if higher hiding rate is needed. The better combinations of values for τ, u, and v at various hiding rates are tabulated in Table 4. The value of u is 3, the value of τ locates around 6, whereas the value of v increases when higher hiding rate is expected. Here, the hiding rate is a percentage defined as the ratio between the number of bits being hidden and $8\times 512\times 512$ (the number of bits of the host image).

4 Comparison

First, a graceful technique called MBNS (multiple-base notational system) presented by Zhang and Wang[16] is compared with our method. MBNS dynamically chooses the base of each pixel, and we use dynamic bases for blocks. In Table 5, the average PSNR of the two methods is compared. We can see that our method has better PSNRs and higher hiding capacity than the MBNS technique has. How-

Table 5 The average PSNR comparison between the MBNS method[16] and the proposed method.

	Hiding Rate			
Method	0.20	0.25	0.30	0.35
MBNS (Ref. 16)	43.5	41.2	39.9	38.1
Proposed method	48.6	45.3	43.7	41.3

ever, the MBNS method yields better Q-index,[20] namely, 0.999. Our Q-indices are slightly less than theirs by a scale of 0.027.

Several reported hiding techniques[17,18] use an adaptive LSB substitution technique, and our method is also adaptive to local performance. Our PSNR and theirs are compared in Table 6 for the image "Lena." Obviously, our method has the largest PSNR value, and the improvements are about 4 dB (Table 6 shows only the image "Lena" because it was the image used by all researchers in these reports.)

We can also compare our method with other types of image hiding methods. Both the vector quantization (VQ)-based[12] and VQ with LSB substitution[13] techniques have good performance in theft-proof extraction and hiding capability; but both techniques must establish a look-up table (LUT) during the embedding process, and must preprocess both the security and the host images. Their extracted image was lossy. Ours is lossless and uses no LUT. However, their host image can be smaller than the secret image due to lossy compression; their hiding rate is thus better than ours. As for the LSB substitution method with a genetic algorithm,[14] its PSNR is a little worse than ours, and it requires the use of a more complicated genetic algorithm. The pixel difference[15] scheme has very good performance in theft proofing, and it uses no LUT. However, its hiding capacity is not as high as ours. For example, when the mixed image "Lena*" has a PSNR around 40 dB, its hiding rate is around 0.27, but ours can be 0.36.

In summary, our method uses no LUT, no preprocessing, and no extraction error and the hiding capacity is high when similar PSNR values of the mixed image H^* are required among different lossless methods.

Robust hiding methods[5,7,10,19] have the advantages of being robust against some (but not all) image processing operations. Our fragile method has the advantage of space saving, as explained in the following. Assume that some private data (for example, the description about students' weaknesses) are to be hidden in a teacher's PC for his own use. If the teacher uses the elegant method in Ref. 19, which is a robust one, a high PSNR (44.4 dB) can be achieved when the hiding rate is $R=0.0078=0.78\%$. However, we have 45.3 dB when $R=0.25=25\%$. In general, if a robust method uses, say, 10% as the hiding rate, to achieve a high PSNR of the mixed images, then it uses 10 host images to hide a secret. Then, if it uses a lossless compression to further compress the mixed images with a compression rate of 40%, the final storage space is $10\times 40\%=400\%$ times larger than the secret image. Obviously, the method costs more storage space than ours. [When the hiding rate is 0.25, our final storage space is only $(1/0.25)\times 40\%=160\%$ times larger than the secret image.] Finally, if the hiding rate of a robust method is very low so that it certainly needs a compression as postprocessing, then we may even skip this postprocessing compression and still defeat the robust method in storage space. In this case, the data extraction of the robust method contains two parts—decompression and inverse-hiding—but we require only inverse-hiding. From the preceding analysis, we suggest the use of robust methods if the secret is both private and important (both the unauthorized disclosure and data disappearance cause disasters). However, if the data are just private, such as a sequence of criticisms against some people (the unauthorized disclosure will cause an unpleasant result, but destruction of the data is not a big loss), then our method is still worthy. This is because our method saves storage space in PC, and the PC owner certainly has the power of choosing not to use image processing (except lossless compression) on certain directories.

5 Summary

We presented a simple method to hide data in images. The method is base-oriented, because it classifies each block of the host image H according to the block's BV, that represents the block's variation. If the BV of the block is in a predefined interval $(0.5\tau, \tau)$, then the method tries to embed confidential data into the block by using the BV. If the BV of the block is out of the predefined interval, however, then the method embeds confidential data into the block by using module function. Preprocessing the host image before embedding the information is not required. Good hiding capacity is obtained while maintaining a good visual performance. At the receiver end, the decoder uses neither the

Table 6 PSNR generated by the proposed method and other adaptive LSB techniques on the image "Lena" at various hiding rates.

	Hiding Rate				
Methods	0.20	0.25	0.31	0.37	0.44
Liu et al.[17]	45.12	—	39.26	—	32.66
Wu et al.[18]	—	38.80	—	36.16	—
Proposed method	48.72	45.99	43.13	40.11	36.54

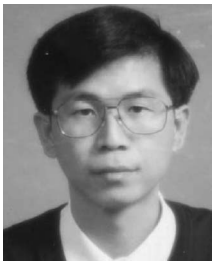
original host image nor a large LUT, such as a codebook, to extract the hidden information from the mixed image. The extraction process is simple and lossless. In our experiments, we used a Pentium IV PC with 128 MB of memory, and the computation times of the encoding and decoding parts to hide data in a 512×512 image took only about 0.30 and 0.20 s, respectively. With good mixed image quality, and fast and lossless recovery, the proposed method can be used as a space-saving and fast tool to hide private data.

Acknowledgments

The authors would like to thank the reviewers and the editor for suggestions that greatly improved the paper.

References

1. S. Katzenbeisser and F. A. P. Petitcolas, *Information Hiding Techniques for Steganography and Digital Watermarking*, Artech House, Norwood, MA (2000).
2. M. Wu and B. Liu, *Multimedia Data Hiding*, Springer-Verlag, New York (2003).
3. I. J. Cox, J. Kilian, F. T. Leighton, and T. Shamoon, "Secure spread spectrum watermarking for multimedia," *IEEE Trans. Image Process.* **6**(12), 1673–1686 (1997).
4. I. J. Cox and J. P. Linnartz, "Some general methods for tampering with watermarks," *IEEE J. Sel. Areas Commun.* **16**(4), 587–593 (1998).
5. C. T. Hsu and J. L. Wu, "Hidden digital watermarks in images," *IEEE Trans. Image Process.* **8**(1), 58–68 (1999).
6. I. J. Cox, M. Miller, and J. A. Bloom, *Digital Watermarking*, Morgan Kaufmann, San Francisco (2002).
7. D. A. Winne, H. D. Knowles, and C. N. Canagarajah, "Digital watermarking in wavelet domain with predistortion for authenticity verification and localization," *Proc. SPIE* **4675**, 349–356 (2002).
8. C. T. Li and F. M. Yang, "One-dimensional neighbourhood forming strategy for fragile watermarking," *J. Electron. Imaging* **12**(2), 284–291 (2003).
9. C. T. Li, "Digital fragile watermarking scheme for authentication of JPEG images," *IEE Proc. Vision Image Signal Process.* **151**(6), 460–466 (2004).
10. Y. Wang and A. Pearmain, "Blind image data hiding based on self reference," *Pattern Recogn. Lett.* **25**, 1681–1689 (2004).
11. W. Benders, D. Gruhl, N. Morimoto, and A. Lu, "Techniques for data hiding," *IBM Syst. J.* **35**(3–4), 313–336 (1996).
12. T. S. Chen, C. C. Chang, and M. S. Hwang, "A virtual image cryptosystem based on vector quantization," *IEEE Trans. Image Process.* **7**(10), 1485–1488 (1998).
13. Y. C. Hu and M. H. Lin, "Secure image hiding scheme based upon vector quantization," *Int. J. Pattern Recognit. Artif. Intell.* **18**(6), 1111–1130 (2004).
14. R. Z. Wang, C. F. Lin, and J. C. Lin, "Image hiding by LSB substitution and genetic algorithm," *Pattern Recogn.* **34**(3), 671–683 (2001).
15. C. C. Chang, J. C. Chuang, and Y. P. Lai, "Hiding data in multitone images for data communications," *IEE Proc. Vision Image Signal Process.* **151**(2), 137–145 (2004).
16. X. Zhang and S. Wang, "Steganography using multiple-base notational system and human vision sensitivity," *IEEE Signal Process. Lett.* **12**(1), 67–70 (2005).
17. S. H. Liu, T. H. Chen, H. X. Yao, and W. Gao, "A variable depth LSB data hiding technique in images," in *Proc. 3rd Int. Conf. Machine Learning and Cybernetics*, August 2004, Shanghai, pp. 26–29 (2004).
18. H. C. Wu, N. I. Wu, C. S. Tsai, and M. S. Hwang, "Image steganographic scheme based on pixel-value differencing and LSB replacement methods," *IEE Proc. Vision Image Signal Process.* **152**(5), 611–615 (2005).
19. L. H. Chen and J. J. Lin, "Mean quantization based image watermarking," *Image Vis. Comput.* **21**, 717–727 (2003).
20. Z. Wang and A. C. Bovik, "A universal image quality index," *IEEE Signal Process. Lett.* **9**(3), 81–84 (2002).



Ching-Yu Yang received his BS degree in electronic engineering in 1983 from National Taiwan Institute of Technology, his MS degree in electrical engineering in 1990 from National Cheng Kung University, Taiwan, and his PhD degree in computer and information science in 1999 from National Chiao Tung University. From 1999 to 2005, he was a senior engineer with Chunghwa Telecommunications Co. Ltd., Taiwan. In February 2005 he joined the Department of Computer Science and Information Engineering at National Penghu University, where he is currently an associate professor. His recent research interests include image compression and information hiding.

Ja-Chen Lin received his BS degree in computer science in 1977 and his MS degree in applied mathematics in 1979, both from National Chiao Tung University, Taiwan, and his PhD degree in mathematics in 1988 from Purdue University. In 1981 and 1982, he was an instructor with the National Chiao Tung University. From 1984 to 1988, he was a graduate instructor at Purdue University. In August 1988 he joined the Department of Computer and Information Science National Chiao Tung University, where he is currently a professor. His recent research interests include pattern recognition and image processing. He is a member of the Phi-Tau-Phi Scholastic Honor Society.