# An improved module-based substitution method for image hiding

Shang-Kuan Chen[1], Ja-Chen Lin[2] and Hung-Lin Fu[3]

[1]Department of Computer Science and Information Engineering, Yuanpei University, Hsinchu, Taiwan, R. O. C.
[2] Department of Computer Science, National Chiao Tung University, Hsinchu, Taiwan, R. O. C.
[3] Department of Applied Mathematics, National Chiao Tung University, Hsinchu, Taiwan, R. O. C.

***Abstract*-The paper proposes an improved module-based substitution method for image hiding. The method has the following characteristics: (1) The extracted data are lossless. (2) Each $k$-bit datum is hidden in a pixel of the host image, and the gray-value distortion is not larger than $2^{k-1}$ for most of the pixels. (3) If it is compared with reported lossless Least-Significant-Bits (LSB) or module-based methods, the proposed method is often with better stego-image quality, unless the hidden data are with very strong randomness. (4) The time complexity is also competitive.**

*Keywords:* Image hiding, LSB substitution, module-based substitution.

## I. INTRODUCTION

Digital data are widely used in current life. Since important digital data are transmitted via internet, they should be protected from illegal peeking or damaging. Hiding data in ordinary images, see [1-7] for references, is a way to accomplish this goal. In general, a stego-image is generated by hiding the important data into a host image. The data can be transmitted much safely by using the stego-image than by transmitting the data themselves directly. Data hiding schemes can be summarized into two types. In the first type, the data extracted from the generated stego-image are with some distortion. In the second type, the extracted data are lossless (without any distortion). The extracted data were distorted in [1-2], but these two methods are with extremely-high hiding capacity, for they could hide data of very large size. For example, Chung et al. [2] hid the important image in a host image of the same size. Wang and Tsai [1] even hid a larger important image in a smaller host image. In both methods, the quality of the extracted image is acceptable, although not lossless. The LSB (Least-Significant Bit) approach is a lossless approach. The simplest one is the so-called simple LSB substitution method which directly replaces the least-significant bit planes of the host image with hidden data. Wang et al. proposed an LSB-based image hiding method in [3], in which an optimal re-naming problem is defined, and then the problem's theoretical optimal solution is approximated by using a genetic algorithm in the searching. The hidden data's size can be as large as one half of the host image's size. Chang et al. [4] gracefully developed a dynamic programming strategy to replace Wang et al.'s [3] genetic algorithm to get a faster algorithm to approximate the aforementioned theoretical solution. In [5], Thien and Lin proposed another digit-by-digit lossless hiding method based on modulus function. Their stego-images' quality (PSNR values) outperformed that of Ref. [3-4], without damaging the hiding capacity or lossless recovery of the secret data. The method is also faster.

In this paper, an improved module-based substitution method for hiding data is proposed. The idea is to modify Thien and Lin's [5] module-based lossless hiding method by adding a counter that counts the heavy repetition of data pattern, should this occur.

The remainder of this paper is organized as follows. Section 2 is an introduction of module-based technique for data hiding. The encoding and decoding phases of the proposed method are illustrated in Sections 3 and 4, respectively. Experimental results are shown in Section 5. Finally, the concluding remarks are in Section 6.

## II. A SHORT REVIEW OF MODULE-BASED HIDING TECHNIQUE

The method of Thien and Lin [5] is reviewed here. Let the host image have $n$ pixels. Partition the secret data into a sequence of non-overlapping segments, and each segment $m_i$ is in the range $\{0, 1, 2, \ldots, q\text{-}1\}$ where $q$ is a given positive integer called the base for module function. We explain how a segment $m_i$ can be hidden in the $i$th pixel (whose original gray value is $p_i$) of the host image. In the simple LSB method, the $i^{th}$ pixel value of the generated stego-image is $p_i' = p_i - (p_i \bmod q) + m_i$, true for each $i$. In [5], $p_i'$ is adjusted further to a new number $p_i''$ that is closer to $p_i$, and yet still congruent to $p_i'$ on the module base $q$, i.e. $m_i = p_i'' \bmod q = p_i' \bmod q$. Therefore, the decoding (to get $m_i$ from $p_i''$) is simple and fast, and the impact to the host image is smaller.

## III. THE ENCODING PHASE

The encoding phase of the proposed method is formed of two stages. As described in Subsec. A, Stage 1 hides each $k$-bit datum in the host image. Then, as described in Subsec. B, Stage 2 provides the pixel adjustment.

### A. *The hiding process*

As usual, let $m_i$ denotes the $i$th $k$-bits-segment of the datum, and each $m_i$ is to be hidden in a pixel of the host image. The stego-image is generated in a pixel-by-pixel manner. Each time when we want to hide a not-yet-hidden $k$-bits segment $m_i$, we also take next several segments simultaneously if their values are all identical to $m_i$; i.e., if $r$ contiguous data segments $m_i$, $m_{i+1}$, … , and $m_{i+r-1}$ are all with the same value $z$ for some positive integer $r$. (Originally, these $r$ segments are to supposed to be hidden in $r$ pixels $p_i$ , $p_{i+1}$,... , and $p_{i+r-1}$ of the host image, respectively). Now, if $r$ is 1, 2, or 3, then for each $l$ = $i$, $i+1$, …, $i+r-1$ , we still let $p_l^{'}$ equal $p_l - p_l \bmod (2^k+1) + m_l$. (Notably, as shown right above, the module base $q$ in our system is $q = 2^k + 1$ rather than $2^k$; the reason is explained below.) In a numerical system with base $q = 2^k + 1$, the available digits are {0, 1, 2, …, $2^k$}. Because each $k$-bit binary datum can be converted to a base-$q$ digit ranges from 0 to $2^k - 1$, the only unused digit is $2^k$. This special digit $2^k$ (the so-called flag-digit) is especially reserved to indicate the special case when the repetition counter value is over 3, i.e. $r > 3$.

Should this special heavy-repetition case occur, besides using a flag digit (value $2^k$) for identifying the case, the repeated value $z$ and the repetition counter value $r$ ($r > 3$) also need be recorded. We therefore need a three-digit vector [$2^k$, $z$, $r$—4.]. The third component is $r$—4, rather than $r$, because $r$ is at least 4 in this heavy-repetition case, and we can record $r$—4 rather than $r$ in order to save coding-length. Notably, in our base $q = 2^k + 1$ system, the available digits are {0, 1, 2,…, $2^k$}; so the digit ($r$—4) is forced to be in the range $0 \le (r-4) \le 2^k$. As a result, the actual value of the repetition counter value $r$ must be in the range $4 \le r \le 2^k + 4$. If we have an extremely-heavy-repetition case in which $r > 2^k + 4$, then we have to cut the repetition interval into several intervals of smaller repetition length. The detail is omitted here. So, assume that $4 \le r \le 2^k + 4$, then the stego pixels $p_i^{'}$ , $p_{i+1}^{'}$ , and $p_{i+2}^{'}$ are $p_i - p_i \bmod (2^k+1) + 2^k$ , $p_{i+1} - p_{i+1} \bmod (2^k+1) + z$ , and $p_{i+2} - p_{i+2} \bmod (2^k+1) + (r-4)$, respectively. (The remaining pixels $p_{i+3}^{'}$ , ... , *and* $p_{i+r-1}^{'}$ equal to $p_{i+3}$ , ... , and $p_{i+r-1}$ , respectively.) Notably, in this operation, the maximal ratio of the number of unchanged pixels to total number of pixels is $(2^k+1)/(2^k+4)$. Of course, with slight modification, these unchanged pixels can also be used to record some other data. This slightly-modified version can either increase *PSNR* or increase the amount-of-hidden-data (the so-called hiding-capacity). The detail is omitted to save paper length.

Let the ordered $n$-tuples $\langle h_1, h_2, \dots, h_n \rangle$ be the hidden data, the ordered $n$-tuples $\{g_1, g_2, \dots, g_n\}$ be the pixels of an $n$-pixels image, and the ordered $n$-tuples [$d_1$, $d_2$, …, $d_n$] be the differences between the pixels of the host image and the stego-image. For example, assume that the data <4, 0, 6, 6, 6, 6, 6, 7, 2, 2>, each has 3-bits (so $k$ = 3), are to be hidden in the ten pixels {183, 187, 186, 191, 196, 196, 193, 190, 187, 186} of the host image. The generated stego-pixels will be {184, 180, 188, 195, 190, 196, 193, 196, 182, 182}. The differences between the host image and the stego-image are thus [1, 7, 2, 4, 6, 0, 0, 6, 5, 4]. Notably, both the 6th and 7th pixels of host image and stego-image are the same, because 3rd, 4th, and 5th pixels of the stego-image have already recorded the information of <6, 6, 6, 6, 6> of the hidden data, by hiding a three-tuple ($2^3$=8, $z$=6, $r$—4=1) in pixels 3, 4, and 5. Here, $2^3 = 8$ is the heavy-repetition-flag, $z = 6$ is the heavy-repeated-value, and $r$+4 = 1 + 4 = 5 is the count that $z$ is repeated.

### B. *The adjusting process*

After the proposed hiding process, the difference between the pixel value $p_i^{'}$ of the stego-image and the corresponding $p_i$ of the host image will belong to the following cases:

Case 1: ($2^{k-1} < p_i^{'} - p_i < 2^k + 1$ and $p_i^{'} > 2^k + 1$).

Case 2: ($-2^k - 1 < p_i^{'} - p_i < -2^{k-1}$ and $p_i^{'} < 255 - 2^k$).

Case 3: ($-2^{k-1} \le p_i^{'} - p_i \le 2^{k-1}$).

Case 4: ($2^{k-1} < p_i^{'} - p_i < 2^k + 1$ and $p_i^{'} \le 2^k + 1$).

Case 5: ($-2^k - 1 < p_i^{'} - p_i < -2^{k-1}$ and $p_i^{'} \ge 255 - 2^k$).

In the adjusting process, $p_i^{'}$ is adjusted to $p_i^{''}$. In Case 1, $p_i^{''} = p_i^{'} - 2^k - 1$, and in Case 2, $p_i^{''} = p_i^{'} + 2^k + 1$. However, in Case 3, 4, and 5, let $p_i^{''}$ equal to $p_i^{'}$, i.e. no adjustment. It can be shown that the $|p_i^{''} - p_i| \le 2^{k-1}$ in Cases 1-3.

**Proposition 1.** In Cases 1 and 2 of the proposed adjusting process, the difference of $p_i^{''}$ and $p_i$ is $|p_i^{''} - p_i| \le 2^{k-1}$.

Proof:

In Case 1:

$$p_i^{''} - p_i = p_i^{'} - 2^k - 1 - p_i > 2^{k-1} - 2^k - 1 = -2^{k-1} - 1$$

and

$$p_i^{''} - p_i = p_i^{'} - 2^k - 1 - p_i < 2^k + 1 - 2^k - 1 = 0,$$

this implies that $|p_i^{''} - p_i| \le 2^{k-1}$.

In Case 2:

$$p_i^{''} - p_i = p_i^{'} + 2^k + 1 - p_i < -2^{k-1} + 2^k + 1 = 2^{k-1} + 1$$

and

$$p_i^{''} - p_i = p_i^{'} + 2^k + 1 - p_i > -2^k - 1 + 2^k + 1 = 0 .$$

Thus, $|p_i^{''} - p_i| \le 2^{k-1}$.

In Case 3:

$|p_i'' - p_i| \le 2^{k-1}$ is trivial. □

In Case 4 and Case 5, because $p_i''$ is a gray-pixel value that ranges from 0 to 255, to avoid overflow or underflow, still let $p_i''$ equal to $p_i'$ rather than $p_i' - 2^k - 1$ and $p_i' + 2^k + 1$, respectively. According to our experience, Cases 4 and 5 seldom happen in reality when $k$ = 1, 2, 3, 4.

For the same aforementioned example, after the adjusting process, the pixels {184, 180, 188, 195, 190, 196, 193, 196, 182, 182} of stego-image will be adjusted to {184, 189, 188, 195, 199, 196, 193, 187, 191, 182}. After the adjustment, the total difference, 23, of [1, 2, 2, 4, 3, 0, 0, 3, 4, 4] between the host image and the adjusted stego-image is less than the total difference, 35, of [1, 7, 2, 4, 6, 0, 0, 6, 5, 4] between the host image and the unadjusted stego-image.

## IV. THE DECODING PHASE

The following reversal process is to extract the original data without any distortion. For each pixel value $p_i''$ of the stego-image, if $p_i''$ mod( $2^k + 1$) $\ne 2^k$, then the extracted datum $m_i$ is $p_i''$ mod( $2^k + 1$) . However, if $p_i''$ mod( $2^k + 1$) = $2^k$, then the data $m_i$, $m_{i+1}$, … , and $m_{i+r-1}$ will be extracted from $p_{i+1}''$ and $p_{i+2}''$ of the stego-image by the following steps:

(1). $z \leftarrow p_{i+1}''$ mod( $2^k + 1$),

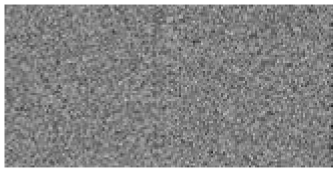
(2). $r \leftarrow 4 + ( p_{i+2}''$ mod($2^k$+1))

(3). For all $j$ from $i$ through $i + r - 1$, let $m_j \leftarrow z$.

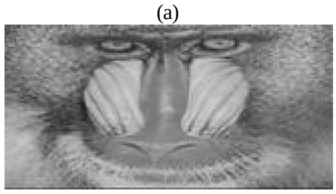
Following the aforesaid example, when extracting the first datum from the first two pixel values 184 and 189 of the stego-image {184, 189, 188, 195, 199, 196, 193, 187, 191, 182}, because 184 mod ($2^3$+1) = 4 $\ne 2^3$ and 189 mod ($2^3$+1) = 0 $\ne 2^3$, the first two datum are 4 and 0, respectively. Then, for the 3rd pixel value 188, because $188 \bmod (2^3 + 1) = 8\ (= 2^k = 2^3)$, it is a heavy-repetition-flag. As a result, the next two pixels 195 and 199 together recorded the repetition of the datum. The datum value $z$ that is repeated again and again is $z$=6=195 mod ($2^3$+1). The 5th pixel is 199; thus, $r$ - 4 is 1=199 mod ($2^3$ + 1). In other words, $z$=6 is repeated $r$=4+1=5 times. In summary, the 3rd, 4th, …, and (3+$r$-1)th=(3+5-1)th=7th data are <6, 6, 6, 6, 6>. Finally, after extracting data from the remaining pixels 8th, 9th and 10th, the original data <4, 0, 6, 6, 6, 6, 6, 7, 2, 2> are recovered without any loss.

## V. EXPERIMENTAL RESULTS

This section presents the experimental results. The simple LSB substitution method, Thien and Lin 's method [5], are both implemented to provide a comparison. Let $k$ = 4 in all experiments. Figs. 1(a)-(h) show the secret images of size 256×512 each, and Figs. 2(a)-(b) show the host images of size 512×512 each. To reduce paper length, only the stego-images generated by hiding Fig. 1(a), Fig. 1(e), and Fig. 1(h) in Fig. 2(a) are shown in Figs. 3(a)-(c), respectively.

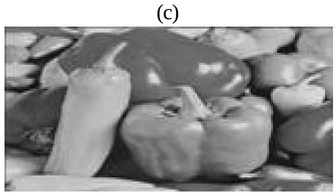


(a)

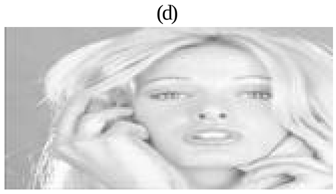


(b)

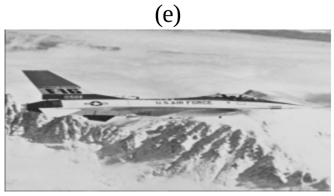
(c)



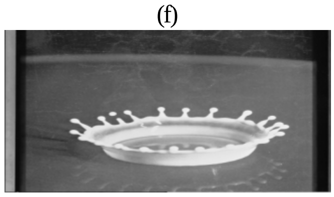
(d)



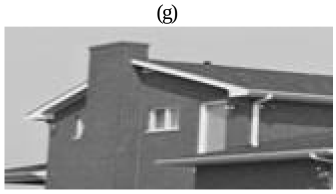
(e)



(f)



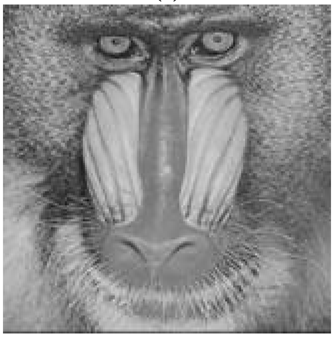
(g)



(h)

Figure 1. The secret images are of size 256×512.

(a)



(b)

Figure 2. The host images are of size 512×512.

(a)

(b)

(c)

Figure 3. (a): The stego-image of hiding Fig. 1(a) in Fig. 2(a). (b): The stego-image of hiding Fig. 1(e) in Fig. 2(a). (c): The stego-image of hiding Fig. 1(h) in Fig. 2(a).

The *PSNR* values are used for measuring the quality of stego-images. Notably, $PSNR = 10 \times \boldsymbol{log}\left(\frac{255^2}{MSE}\right)$ dB where 255 is because gray values are in 0-255. Assuming that $H$ is the host image and $S$ is the stego-image, and both are with size $m \times n$. $H_i$ and $S_i$ denote the $i^{th}$ pixel values of $H$ and $S$, respectively. Then the *MSE* is defined as $MSE = \frac{1}{m \times n}\sum_{i=1}^{m\times n}(S_i - H_i)$. The *PSNR* of the stego-images is listed in Table 1 for comparison. In Table 1, ours are usually the best. Notably, when the secret data are of very slight adjacent-repetition (for example, Fig. 1(b)) or almost no adjacent-repetition (for example, Fig. 1(a)), the quality of our stego-image is a little worse than that of Thien and Lin's method [5]. The reason is that our space-saving hiding policy by using the three-digit vector $[2^k, z, \quad r\text{-}4]$ is no longer useful (see Sec. 3.1 to understand this three-digit vector); while we waste an extra digit as the flag-digit (rather than using it as the data digit). Finally, Fig. 4(a)-(c) show the stego-images of hiding Fig. 1(f) in Fig. 2(a) using, respectively, simple LSB substitution method, Thien and Lin's method [5], and the proposed method. To clearly show these images, larger scale is used.

TABEL I

The *PSNR* of stego-images (with $k = 4$) generated using the simple LSB substitution, Thien and Lin's method, and the proposed method. The proposed method outperformed the other two methods, unless the smooth area is too small in the secret image (for example*, Baboon image, or another image formed of random noise only.).

| Secret images (256×512) | Host image = Lena (512×512) | | |
|---|---|---|---|
| | The simple LSB substitution method | Thien and Lin's method | The proposed method |
| House | 32.69 | 34.78 | 36.37 |
| Milk | 32.26 | 34.86 | 35.76 |
| Jet | 31.95 | 34.76 | 35.73 |
| Tiffany | 31.32 | 34.80 | 35.49 |
| pepper | 32.44 | 34.79 | 35.46 |
| Baboon* | 32.65 | 34.78 | 34.73 |
| Random-noise* | 31.82 | 34.81 | 34.33 |

| Secret images (256×512) | Host image = Baboon (512×512) | | |
|---|---|---|---|
| | The simple LSB substitution method | Thien and Lin's method | The proposed method |
| House | 32.70 | 34.79 | 36.37 |
| Milk | 32.26 | 34.80 | 35.80 |
| Jet | 32.04 | 34.82 | 35.72 |
| Tiffany | 31.40 | 34.82 | 35.52 |
| pepper | 32.46 | 34.80 | 35.44 |
| Lena | 32.58 | 34.82 | 35.36 |
| Random-noise* | 31.83 | 34.80 | 34.34 |

(a)

(b)

(c)

Figure 4. The stego-images of hiding Fig. 1(f) in Fig. 2(a) using (a) simple LSB substitution method, (b) Thien and Lin's method, and (c) the proposed method. (larger scale used)

## VI. CONCLUDING REMARKS

Traditional $k$-bits LSB-substitution methods can be viewed as methods that use $2^k$ as the base for module function. The proposed method is with module-base ($2^k + 1$), in which the $2^k$ digits {0, 1, 2, …, $2^k$-1} are used as ordinary numbers, but the special digit $2^k$ is a flag for identifying the heavy repetition case. The quality of the stego-images generated by the proposed method are often better than that generated by Thien and Lin's method [5], which in turn are better than that generated by [3-4]. On the other hand, when the data is with very slight repetition, Thien and Lin's method [5] outperformed ours. (In Table 1, Ref. [5] lead in *PSNR* by a 0.48dB difference when the secret data is formed of noise only; and a 0.05db difference when the data is the image Baboon.) In short, the proposed method benefits from hiding heavy-repetition data, especially a smooth image.

Notably, in decoding, the proposed method has the same very-low computation complexity (*O*(image size)) that the reported LSB substitution methods [3-4] or module-based method [5] have. In encoding, its computation complexity is the same as [5], and hence, much faster than the methods [3-4] which searched for best renaming using genetic algorism or dynamic programming.

In summary, without time overhead, we have proposed a lossless hiding method that has smaller impact to host images.

## ACKNOWLEDGMENT

This work was supported by the National Science Council, Republic of China, under grant NSC96-2218-E-264-002. The authors also thank to the valuable comments from reviewers.

## REFERENCES

[1] R. Z. Wang, Y. D. Tsai, "An image-hiding method with high hiding capacity based on best-block matching and k-means clustering," *Pattern Recognition*, vol. 40, 2007, pp. 398-409.

[2] K. L. Chung, C. H. Shen, L. C. Chang, "A novel SVD- and VQ-based image hiding scheme," *Pattern Recognition Lett.*., vol. 22, 2001, pp. 1051-1058.

[3] R. Z. Wang, C. F. Lin, J. C. Lin, "Image hiding by optimal LSB substitution and genetic algorithm," *Pattern Recognition*, vol. 34, 2001, pp. 671-683.

[4] C. C. Chang, J. Y. Hsiao, C. S. Chen, "Finding optimal Least-Significant-Bit substitution in image hiding by dynamic programming strategy," *Pattern Recognition*, vol. 36, 2003, pp.1583-1595.

[5] C. C. Thien, J. C. Lin, "A simple and high-hiding capacity method for hiding digit-by-digit data in images based on modulus function," *Pattern Recognition*, vol. 36, 2003, pp. 2875-2881.

[6] C. C. Chang, C. Y. Lin, Y. Z. Wang, "New image steganographic methods using run-length approach," *Information Sciences*, vol. 176, 2006, pp. 3393-3408.

[7] C. C. Chang, W. L. Tai, C. C. Lin, "A reversible data hiding scheme based on side match vector quantization," *IEEE Transactions on circuits and systems for video technology*, vol. 16, no. 10, 2006, pp. 1301-1308.