Available online at www.sciencedirect.com
SCIENCE DIRECT®
ELSEVIER Applied Mathematics and Computation 164 (2005) 99–116
APPLIED MATHEMATICS AND COMPUTATION
www.elsevier.com/locate/amc

Steganography of capacity required using modulo operator for embedding secret image

Shiuh-Jeng Wang

Department of Information Management, Central Police University, Taoyuan 333, Taiwan

Abstract

In this paper, a novel mechanism using modulus operations to incorporate secret data (with image form) into a host-image is proposed. The used modulus is a threshold value that indicates how many bits of the secret image are being incorporated into the host-image. In our scheme, there are two kinds of secret images processed. The capacities of these two types are half the size and a quarter the size of the chosen host-image, respectively. As can be seen in the result of our experiment, not only can the secret images be totally embedded, but the extra information, such as the auxiliary table, assisting the secret extraction is released as well. Even then, there is still space remaining after the hiding procedure is completed. The result is a higher capacity exploration for embedding the secret into a host-image. Upon inspection of the quality of the stego-image using the measurement of PSNR it is noted that not only does the image display make the grade of the human vision system, i.e., it is imperceptible by human eyes as far as the embedded secret is concerned, but also the valued results of PSNR remain sufficiently stable for a variety of host-image selections. Therefore, our proposal scheme is notably superior to those in the previous literature [IEE Proc.—Vision Image Signal Process. 147(1) (2000) 29]. The advantages of this proposed scheme are the higher capacity exploration on secret embedding, and better stability on stego-image displays.

E-mail address: sjwang@sun4.cpu.edu.tw

0096-3003/$ - see front matter
doi:10.1016/j.amc.2004.04.059

Keywords: Modulus; Information hiding; Embedding procedure; Secret

1. Introduction

In recent years, another branch of research, in addition to the cipher-based cryptosystem has been exploring security issues. This new branch is called information hiding. The main difference between the two branches is that the secret is encrypted as cipher text of a random-number form in the cipher-based cryptosystem, whereas with the information hiding system, the secret is embedded into the meaningful medium, such as image, or video display so that the embedded secret is kept imperceptible to human vision when looking at the medium that embeds the secret. That is to say, in information hiding, not only is the secret kept inside the medium, but also the display fidelity of the medium that embeds the secret is almost the same as the original one. Therefore, several researchers [1,4,6,8–12,14–24] have paid much attention to information hiding. In principle, the explorations of information hiding are divided into two categories: watermarking and steganography issues. The former study is aimed at the protection of authorized image data and the latter one is focused on the data embedding procedure of how to develop the capacity for an embedded secret. More specifically, information hiding is a means of securely hiding representative digitized information (secret) into a cover-medium such that the activated medium is still meaningful as a visual view, where the activated medium is usually called the stego-medium. Without loss of generality, a mature information hiding exploration is comprised of some key requirements: transparency, universality, robustness, capacity and security [1]. For transparency, it is required that the two images of ''with embedding secret'' and ''without embedding secret'' are indistinguishable to human sight. The two compared images are regarded as the same without causing any suspicion. Besides, it should conform to the purpose of ''seeing is believing'' surveyed in [10]. The universality is used to authenticate the ownership for a receiving image. It is required that the content is universal, recognizable and unambiguous for the intended people, when the secret is extracted from the stego-medium. As to the feature of robustness, it is required that the stego-medium is supposed to be damage-resistant. In particular, it must be able to resist against those frequent image processes, such as image filtering, image compression. When it comes to capacity, it is required to keep the secret as much as can be embedded in a given medium. In contrast to robustness, in general, a medium holding a higher capacity is vulnerable to damage by frequent image processing. The tradeoff for gaining the benefit of strong robustness and high capacity always relies on the physical demand for

hiding the secret. And last the security requirement. For security reasons, it is required to correctly conjecture the pixels/regions into which the secret is embedded at a very low possibility. In addition, the secret must be unable to be arbitrarily removed or altered by anyone but the owner. In general implementation, the security is fulfilled by either the encryption manipulation using private-key/public-key system or through pseudo-random generation using a seed key. In either case, no matter which scenario is applied, the used keys are kept secret in the information hiding system, only known to the legal inspectors.

Over the past years, many literatures discussed the technique of information hiding. Up until now, there are two techniques developed in information hiding, spatial-domain manner [1,18,20] and frequency-domain manner [4,6,9,12,14,19]. Owing to the fact that the media considered in these literatures are image illustrations, we therefore will include images in our discussions. The so-called spatial-domain refers to the fact that the secret is mixed into the distributed pixels (regions) directly. While in the frequency-domain, it is necessary to transform the host-image first using a frequency-oriented mechanism, such as a DCT-based, wavelet-based, etc., after which the secret is then combined with the relative coefficients in the frequency-form image. Let us take another look at the spatial-domain manner. Generally speaking, it is simpler to achieve the goal of information hiding in the course of secret embedding. The least significant bit (LSB for short) secret embedding or LSB-like embedding is the most commonly used method in the spatial-domain approach. Surveying the past literatures it was found that the method proposed in [23] is not a totally genuine embedding procedure. The reason for this is that the auxiliary table generated in their scheme is needed in the course of the secret extraction phase. In other words, the secret revelation is conducted with the cooperation of the stego-image and the auxiliary table. This causes the problem that the created auxiliary table occupies extra space. It does not conceal the table so that such method becomes a partial embedding procedure, not a totally genuine embedding procedure. Even though the auxiliary table is embedded into the host-image, it will degrade the quality of the stego-image. Accordingly, we proposed a modulo-oriented mechanism to realize a genuine data embedding procedure. We were able to embed/extract a secret without the auxiliary table. In addition, we retained the same high quality after embedding the secret into the host-image. That is to say, a genuine secret embedding procedure is proposed in our scheme.

The rest of our paper is organized as follows: a brief review for the past literature is given in Section 2. In Section 3, we depict our scheme using modulo operator for embedding secret image. Next, the empirical experiments and performance analyses are presented in Section 4, and we conclude our scheme in Section 5.

2. Brief review of Wu et al.

In the past, Wu and Tsai [23] proposed an image hiding scheme in which the gray values of consecutive image pixels are analyzed and the secret is then embedded against the human visual system. Owing to the fact that the intensities of neighboring pixels in an image are usually similar, a difference image is therefore generated by differing the neighboring pixels in an image. With two processed images, one is the secret image, the other one is the host-image, the histograms of the two difference images are quite similar in shape. Wu et al. embedded the secret image into the host-image by replacing the gray value between the secret difference image and the host difference image. Meanwhile, range tables are required to record the location of the host difference image when embedding the secret difference image. Afterwards, an inter-medium image is formed together with the range tables. Finally, the inter-medium image is then transformed into the stego-image by the inverse differencing manner. The range table is named as the leading information in Wu et al.'s scheme, which must be used in the extraction process to get the embedded secret image. The leading information is put on an available place in the network, or hidden into the host-image directly for later use in the extraction process. When considering the process of leading information, this is clearly a space-consuming image hiding process in that we need to create extra space to store it. If the leading information is also embedded into the host-image, the image quality of the stego-image will become degraded after the process. Even the PSNR measurement is less than 30 dB in their scheme after embedding the leading information, as for example, the case of embedding the secret image, 'Mandrill'.

The remarked advantage in [23] is the concept of difference image. Accordingly, the highest embedded capacity is able to promote half a host-image. But, the drawback in this scheme is that the leading information (range table), which is the index table indicating the pixel location associated with the embedding secret image, is an additional requirement. Even the space occupied by the leading information is much greater than the embedded secret image space. Besides, the system requires extra effort to encrypt the leading information for security considerations. The leading information could be embedded into the stego-image, but then the quality of the stego-image will become even worse than before. Therefore, the leading information becomes a significant burden in their scheme.

In this paper, we therefore proposed a scheme that remedies such deficiencies in [23]. We also performed the relevant experiments in which two kinds of secret images sized one half and one quartersize of the host-image are processed. When observing the result of our experiment, not only can we totally embed all the secret images but the extra information assisting the image extraction is released as well. Furthermore, the quality of the stego-image

measured by PSNR is acceptable for human vision and stable under diverse host-image processes.

3. Our scheme using modulo operator for embedding secret image

Owing to the technique of the data embedding procedure explored in our scheme, it is subject to the spatial-domain manner. Therefore we need to deliberately observe the special characteristics among the pixel distributions in the host-image (which means cover-image in Section 2) and the secret image, so as to imperceptibly embed the secret image into the host-image. As a matter of fact, the pixel of gray-value in the secret image is able to be treated as a bit-sting of 8-bit. By concatenating all pixels of bit-string form in the secret image, it will turn into a long 0/1 pattern bit-string representation instead. For a bit pattern with fixed-length l, the corresponding decimal integer is supposed to fall into the range of $[0, 2^l - 1]$. Clearly, an integer in the range of $[0, 2^l - 1]$ is a remainder integer on modulo 2^l. In this paper, we postulate that an integer of bit-length l, which is the part of the secret image of the bit-string is able to be generated by the pixels in the host-image, where the specified integer of bit-length l is the remainder under modulo 2^l. Collecting all the generated remainder integers, the secret image is then totally revealed. Inspired by the merits as mentioned above, we explore the modulus relationship between the chosen pixel in the host-image and the secret image of bit-string form in our scheme. Without loss of generality, there are three phases for carrying out the secret hiding procedure. The first one is the pre-processing phase of bit-string transformation in which the security consideration is imposed. The second one is the secret embedding phase in which the numeric relationship is established between the randomly chosen pixel in the host-image and the secret image. The third one is the bit-string extraction and secret image revelation phase in which each pixel of the secret image is recovered through the modulus operation. The three phases are detailed as follows:

Phase I (Pre-processing phase): [Bit-string transformation]

Input: The secret image S.
Output: The secret bit-string B_s.
Step 1: Order all pixels in S in a top-to-down and left-to-right way.
Step 2: Represent each pixel in an 8-bit length, and concatenate these bits to be a long bit-string.
Step 3: Encrypt the bit-string generated in Step 2 using the DES-like cryptosystem.
Step 4: Output the encrypted bit-string B_s.

Phase II: [Secret embedding]

Input: The host-image C, bit-string B_s, seed key SK and two modulus numbers m_u and m_l.

Output: A stego-image SI.

Step 1: Randomly choose a pixel $P_c(i)$ in C by using the pseudo-random-number generator with SK, where $P_c(i)$ denotes the intensity of the ith pixel with the linear order of top-to-down and left-to-right in C.

Step 2: Set the threshold value T and the two modulus values m_u,m_l. Then compute the residue RES and the possible embedded capacity EC as the following procedure:

IF $P_c(i) > T$

$$\text{Compute } EC = \lfloor \log_2 m_u \rfloor, \tag{1}$$

$$\text{Compute } RES = P_c(i) \bmod m_u. \tag{2}$$

ELSE

$$\text{Compute } EC = \lfloor \log_2 m_l \rfloor, \tag{3}$$

$$\text{Compute } RES = P_c(i) \bmod m_l. \tag{4}$$

Step 3: Compute a difference value D such that

$$D = |RES - DEC|, \tag{5}$$

where DEC is the decimal value of EC bit-length fetched from B_s.

Step 4: Embed DEC into the pixel $P_c(i)$ by performing the following process (here, we pre-define $P_s(i)$ as the intensity of the ith pixel after embedding DEC).

- **Case I:** $P_c(i) < T$:

1. **IF** $P_c(i) < \frac{m_l}{2}$

$$\text{Assign } P_s(i) = 0 + DEC. \tag{6}$$

2. **IF** $\frac{m_l}{2} \leqslant P_c(i) < T - \frac{m_l}{2}$

IF $D > \frac{m_l}{2}$

$$\text{Compute the adaptable value } AV = m_l - D. \tag{7}$$

IF $RES > DEC$

$$\text{Compute } P_s(i) = P_c(i) + AV. \tag{8}$$

ELSE

$$\text{Compute } P_s(i) = P_c(i) - AV. \tag{9}$$

IF $D \leqslant \frac{m_l}{2}$

Compute $AV = D$. (10)

IF $RES > DEC$

Compute $P_s(i) = P_c(i) - AV$.

ELSE

Compute $P_s(i) = P_c(i) + AV$.

3. **IF** $\left(T - \frac{m_l}{2}\right) \leqslant P_c(i) < T$

Compute $P_s(i) = P_c(i) - RES + DEC$. (11)

- **Case II:** $P_c(i) \geqslant T$:

1. **IF** $P_c(i) > \left(255 - \frac{m_u}{2} + 1\right)$

Compute $P_s(i) = (255 - m_u + 1) + DEC$. (12)

2. **IF** $\left(T + \frac{m_u}{2}\right) < P_c(i) \leqslant \left(255 - \frac{m_u}{2} + 1\right)$

IF $D > \frac{m_u}{2}$

Compute $AV = m_u - D$. (13)

IF $RES > DEC$

Compute $P_s(i) = P_c(i) + AV$. (14)

ELSE

Compute $P_s(i) = P_c(i) - AV$.

IF $D \leqslant \frac{m_u}{2}$

Compute $AV = D$.

IF $RES > DEC$

Compute $P_s(i) = P_c(i) - AV$.

ELSE

Compute $P_s(i) = P_c(i) + AV$.

3. **IF** $T \leqslant P_c(i) \leqslant \left(T + \frac{m_u}{2}\right)$

Compute $P_s(i) = P_c(i) - RES + DEC$.

Step 5: Output the stego-image SI containing the $P_s(i)s$ of embedded secret B_s.

There numerous image experiments have been performed by the procedure of our Phase II to illustrate the feasibility and efficiency of our mechanism. We shall explain the results in later Section 4. Next we give the last phase, Phase III, extracting the secret image from the stego-image.

Phase III: [Bit-string extraction and secret revelation]

Input: The stego-image SI, the seed key SK, the threshold value T the two parameter m_u and m_l and the decryption key of DES-like crypto-system.

Output: The extracted bit-string of B'_s and original secret image S'.

Step 1: Find the secret embedding pixel $P_s(i)$ in SI by using the pseudo-random-number generator of seed SK.

Step 2: Compute RES and EC according to the following two cases.

- **Case I:** $P_s(i)<T$:

  Compute $RES = P_s(i) \bmod m_l$. (15)

  Compute $EC = \lfloor \log_2 m_l \rfloor$.

- **Case II:** $P_s(i) \geqslant T$:

  Compute $RES = P_s(i) \bmod m_u$. (16)

  Compute $EC = \lfloor \log_2 m_u \rfloor$.

Step 3: Translate the RES into the bit representations with EC bit-length.

Step 4: Repeat Steps 1–3 until all the embedded bits of B'_s are recovered. Then decrypt B'_s to the original bit-string under the DES-like crypto-system with the given decryption key.

Step 5: Separate the long bit-string decoded in Step 4 with a block of 8-bit. Then reconstruct the original secret image S' of gray-level still image.

4. Empirical experiments and discussions

4.1. Empirical experiments

In our scheme, the host-image used in our scheme is a pixel of gray-level with 256 intensity distribution, and of a size containing 512×512 pixels, as

shown in Fig. 1. With regard to the embedded secret images, which are also of gray-level images with a pixel of 256 intensity distributions. Here, there are two kinds of embedded secret images considered in our experiment, of size 256×512 and 256×256, respectively, as shown in Figs. 2 and 3. As observed from our experiments, the remarkable stability of our proposed scheme is shown when compared with the previous literature [23]. The most important aspect of our scheme is that the auxiliary table is no longer needed, but that the high capacity remained. In the following paragraphs, the practical parameters used in our experiments are depicted to illustrate the proposed secret hiding procedures.

Despite the embedded secret image being 256×512 or 256×256, the threshold value T picked in our algorithm is empirically set at the intensity value of 160. Consider the first case of embedded image with size 256×512. Owing to the fact that this size is one half of the host-image, the possible changed bits of a pixel in the host-image are thus at least 4 bits. Accordingly, the modulus values m_u and m_l

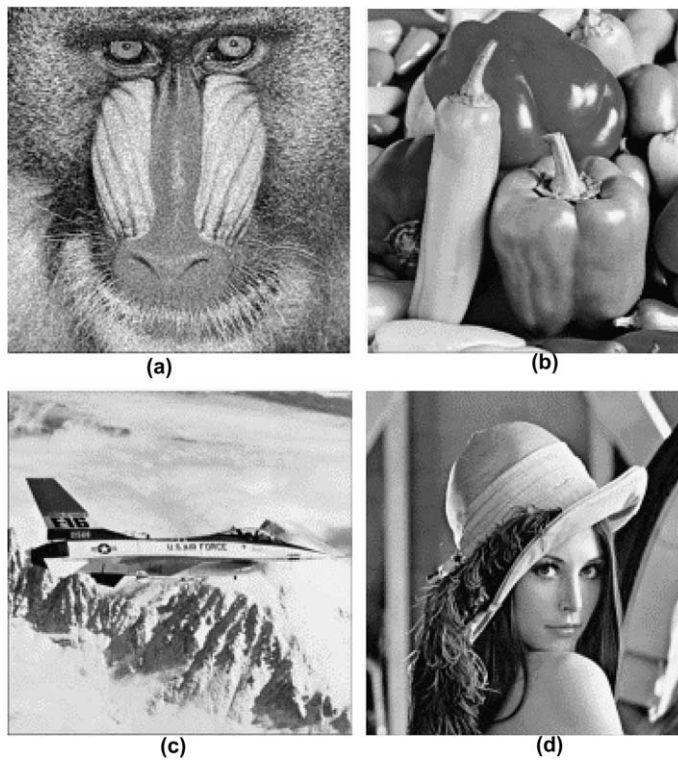

Fig. 1. Four host-images with sizes of 512×512 for experiment use: (a) mandrill, (b) peppers, (c) jet, and (d) Lena.

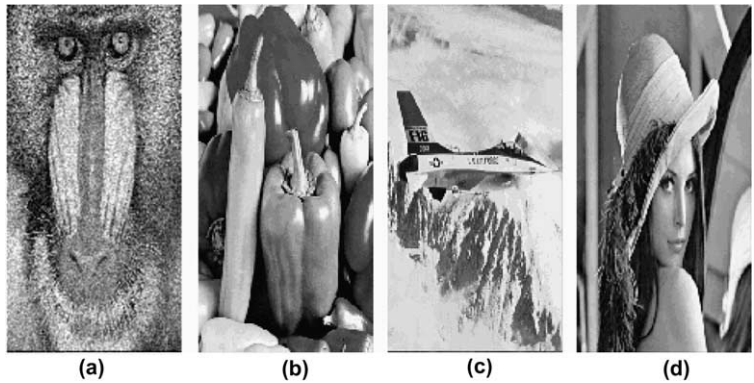

Fig. 2. Four embedded images with sizes of 256×512: (a) mandrill, (b) peppers, (c) jet, and (d) Lena.

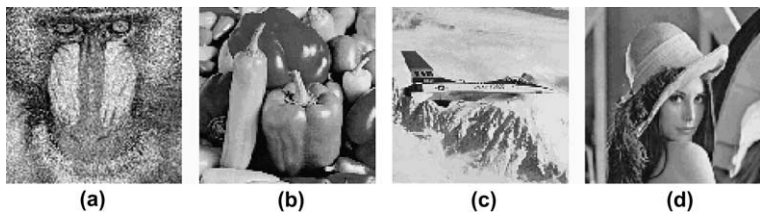

Fig. 3. Four embedded images with sizes of 256×256: (a) mandrill, (b) peppers, (c) jet, and (d) Lena.

are set at 32 and 16, respectively. In a sense, the modulus 32 is set to accommodate the embedding of a 5-bit pattern in the situation that the chosen pixel of the host-image is greater than T. On the other hand, the modulus 16 is set to accommodate the embedding of a 4-bit pattern on the condition that the pixel of the host-image is less than T. Next we turn to examine the case of embedding the image of size 256×256. Similarly, the two values $m_u = 8$ and $m_l = 4$ are set, respectively. This means that the input embedded secret is either a 3-bit or a 2-bit pattern each time in the image embedding procedure. So, it is perceivable, due to the fact that there are less bits being mixed with the chosen pixel in the host-image so that the display quality will not degrade too much, i.e. the visual effect with the smaller secret image embedding is better than that of the larger one. In Figs. 4–7, we show the relevant results of our experiment with embedding/extracting the two difference kinds of secret images of sizes 256×512 and 256×256, respectively. Furthermore, Table 1 is given to show the qualities of the experimented images in our scheme, where the posted numbers are measured by the criterion estimation of peak-signal-to-noise ratio (PSNR).

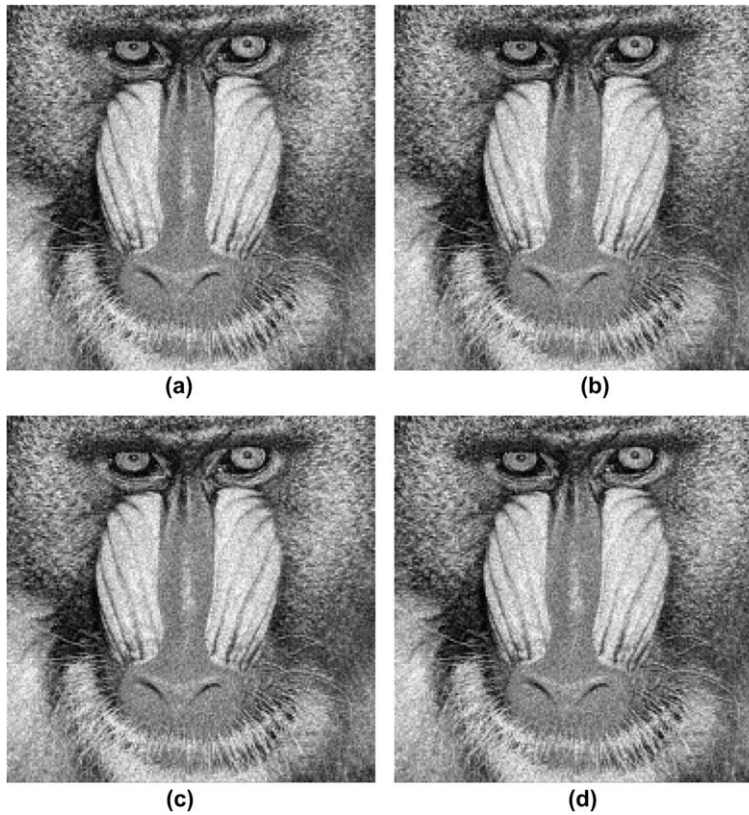

Fig. 4. Four stego-images after embedding Fig. 2(a)–(d) using the host-image of 'mandrill'.

Let us analyze Table 1 as shown in this paper. The PSNR performed for embedding the image of size 256×512 in our scheme ranged in the intervals of 30.23–31.65, which is higher than the standard measurement of 30 dB. This means that the secret which is stored behind the host-image is imperceptible to human vision. Although the PSNR measure in the compared literature [23] is almost higher than our scheme, the problem is that the auxiliary table is not included into the embedding procedure. This is contrary to a genuine secret embedding procedure. Moreover, the range of 28.2–40.53, it turns out that for measuring a variety of host-images the PSNR is unstable and is dependent on the chosen host-images. As regarding the case of embedding the secret image of size 256×256, the PSNR measurement in our scheme ranged between 42.92–44.39, which is considerably higher than 34.61–41.58 of [23] with the same chosen host-image as observed from Table 1. In both compared schemes with the smaller secret image embedded, all relevant information is embedded

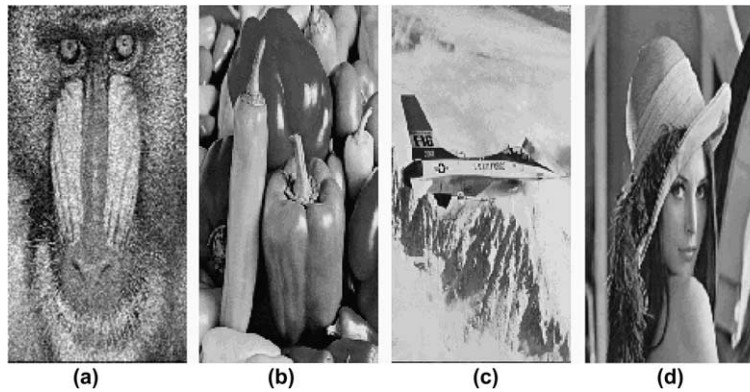

Fig. 5. The extracted images with sizes of 256×512 from Fig. 4(a)–(d): (a) mandrill, (b) peppers, (c) jet, and (d) Lena.

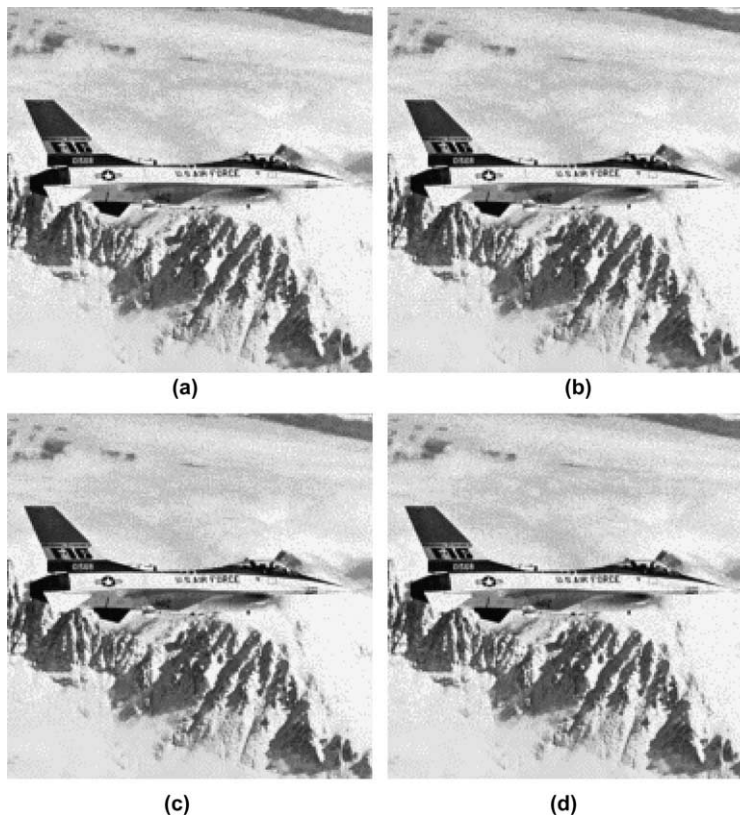

Fig. 6. Four stego-images after embedding Fig. 2(a)–(d) using the host-image of 'jet'.

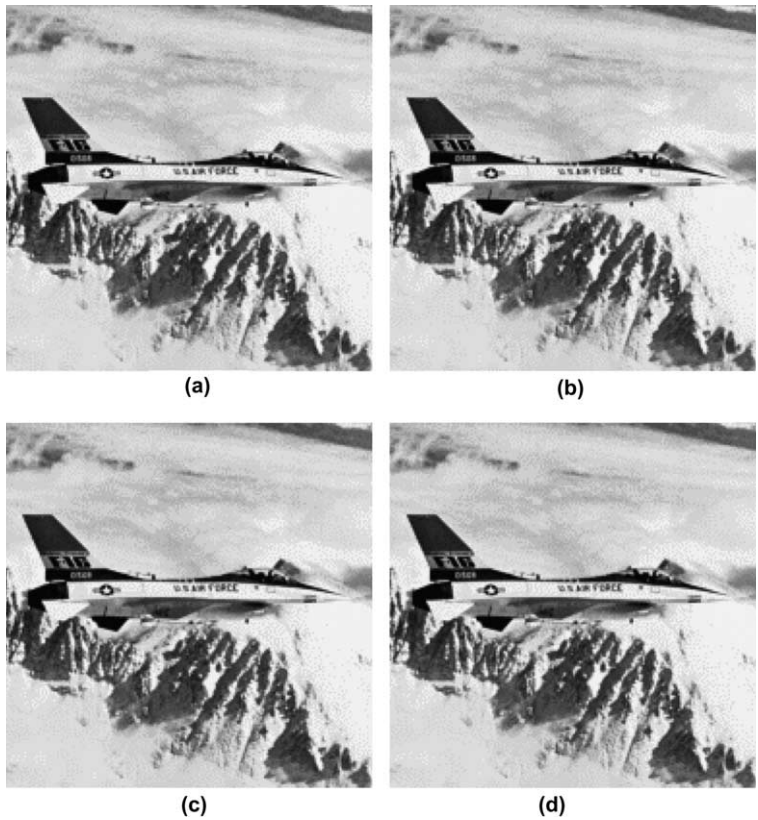

Fig. 7. Four stego-images after embedding Fig. 3(a)–(d) using the host-image of 'jet'.

into the host-image, so that they are a genuine data hiding procedure. Basically, what it boils down to in our scheme is that, no matter if the secret image is of a larger or smaller size, they are all totally embedded into the host-image without any extra information aid, such as the need of an auxiliary table. And, in addition, sufficient free space remains after the completion of the hiding procedure. It is therefore concluded that our proposed scheme is feasible, flexible (larger secret image or smaller secret image) and quality-stable for the diverse host-image choices.

4.2. Computational aspect

Actually, in our scheme, before the bit-string B_s goes into Phase II, it is completed by encryption manner in order to promote the security. Here the DES which is the popular symmetric cryptosystem, is assumed to be used in an encryption/decryption manner. Therefore, let the total length of the B_s be L.

Table 1
The PSNR comparisons between our scheme and [23] after embedding two kinds of image with size of 256×512 and 256×256

Host-image	Scheme	Secret image				
		Mandrill	Peppers	Jet	Lena	The rest capacity (in bit)
Mandrill	(a) [23] with size of 256×512	32.65	32.05	32.24	33.77	–
	(b) [23] with size of 256×512	33.87	32.26	33.00	33.97	–
	(c) [23] with size of 256×256	39.08	39.98	40.22	39.87	–
	(d) [23] with size of 256×256	40.44	41.12	40.96	41.00	–
	(e) Our scheme with size of 256×512	31.24	31.28	31.09	31.25	84019
	(f) Our scheme with size of 256×256	44.07	44.08	44.03	44.07	84019
Peppers	(a) [23] with size of 256×512	28.05	37.36	33.82	35.82	–
	(b) [23] with size of 256×512	28.45	38.59	34.68	36.93	–
	(c) [23] with size of 256×256	37.01	39.80	38.81	39.09	–
	(d) [23] with size of 256×256	36.15	41.15	39.89	40.28	–
	(e) Our scheme with size of 256×512	31.60	31.65	31.42	31.61	71871
	(f) Our scheme with size of 256×256	44.37	44.38	44.31	44.39	71871
Jet	(a) [23] with size of 256×512	28.05	38.69	35.25	36.41	–
	(b) [23] with size of 256×512	28.45	40.53	36.42	37.80	–
	(c) [23] with size of 256×256	34.61	40.05	40.19	39.76	–
	(d) [23] with size of 256×256	34.76	41.58	40.85	39.29	–
	(e) Our scheme with size of 256×512	30.23	30.27	30.24	30.29	192738
	(f) Our scheme with size of 256×256	42.94	42.98	42.92	42.93	192738
Lena	(a) [23] with size of 256×512	28.02	37.47	33.96	35.75	–
	(b) [23] with size of 256×512	28.31	38.35	34.53	36.54	–
	(c) [23] with size of 256×256	37.25	39.44	38.99	39.17	–
	(d) [23] with size of 256×256	37.03	40.31	39.77	39.94	–
	(e) Our scheme with size of 256×512	31.28	31.33	31.05	31.27	69511
	(f) Our scheme with size of 256×256	44.17	44.20	44.08	44.18	69511

(a): Not included range table 1 (auxiliary table).
(b): Not included range table 2 (auxiliary table).
(c): Included range table 1 (auxiliary table).
(d): Included range table 2 (auxiliary table).
(e,f): Our proposed scheme (without auxiliary table).

Owing to the fact that the DES is operated in a 64-bit block, we therefore denote the DES block computing time by DES(64). This way the computing time for DES is measured as the form:

$$CT_L = \left\lceil \frac{L}{64} \right\rceil \mathrm{DES}(64). \tag{17}$$

Examining our embedding procedure in Phase II, we find that the secret images are two kinds of sizes, 256×512 and 256×256. The time measured is further expressed as

$$CT_{256\times 512}\left\lceil\frac{8192}{64}\right\rceil \mathrm{DES}(64) = 128\mathrm{DES}(64)$$

and

$$CT_{256\times 256}\left\lceil\frac{4096}{64}\right\rceil \mathrm{DES}(64) = 64\mathrm{DES}(64).$$

In general, the hard implementation for a cipher system is much faster than the software implementation manner. For example, with the DES used in our scheme, the rate of encrypting the block DES(64) is 1 Giga-bit per second in hardware implementation [5]. In other words, there are 16.8 million blocks per second being manipulated in the DES encryption process. As a result, around 8 and 4 μs are required, respectively, for $CT_{256\times 512}$ and $CT_{256\times 256}$ in DES hardware implementation, which is extremely fast for hardware implementation. Then we consider the software implementation to encrypt DES. According to relevant literature [17], the software implementation on an Intel 80486/33 MHz microprocessor required 2.6 million-bit per second, and 0.22 million-bit per second are required on a Motorola 68020/16 MHz microprocessor. As the evaluation in [17], if the Intel-PC system (with the slower Intel 80486/33 MHz microprocessor) is used for the test, the time required for $CT_{256\times 512}$ and $CT_{256\times 256}$ is around 3.15 and 1.58 ms, respectively. Accordingly, there will be much fast to complete the pre-processing for encrypting the secret image in now Intel-PC system. As a whole, the pre-processing with the DES encryption for the embedding secret image is time-efficient as per our analysis.

Let us move to the investigation of the computational requirement in Phase II and Phase III. Owing that our scheme is dominated by the operation of a modulo-oriented mechanism to embed the secret image and extract the original secret, the modulus operations become the main concerns in the time requirement. Consider the modulus operations in (1), (4), (15) and (16). Assuming there are r pixels in the host-image chosen to embed the secret, where these chosen pixels are assigned by using the pseudo-random-number generator manner. Consequently there will be r*MOD(8) operations required in Phase II, and Phase III, respectively, where MOD(8) denotes the modulus operation in 8-bit integer. Examine the modulus operation computation in 32-bit microprocessors. Basically, the operation of MOD(32) is able to be completed within a number of microseconds under a variety of 32-bit microprocessor architectures [7,9]. It turns out that the computation of MOD(8) is completed in a very short time. We further take a look at the number of r in the time requirement of r* MOD(8). For convenience, $r = 32\,768$ is counted if the B_s of secret image of size 256×512 is processed in a block of 4-bit, which means that modulus m_u/m_l in (1) and (3) is set as 16. In this way, the total computation time for the low-bit modulus operations is still within a number of milliseconds by using a 32-bit

based microprocessor. The efficiency of our proposed scheme is thus achieved regarding executing time.

4.3. Security aspect

In this subsection, we investigate the security issue in our proposed scheme. We examine our phases, to secure the secret embedding procedures which are imposed on DES encryption in Phase I, and the generation of pseudo-random pixels using seed SK in Phase II. First we discuss the security aspect of DES. As a matter of fact, DES has become a popular manner to encrypt the block message in now several commercial applications and encrypted products being marketed for the sake of its simple, fast and secure guarantee. Therefore, we also incorporate this popular cipher into our implementation of secret image embedding. The encryption key in DES is 56-bit length in use. That is, the possibility is 1 in 2^{56} to obtain the correct one that we use. Assume a computer which performs 1000 million instructions per second is adopted to find out the correct key in our scheme. The computational time is around $\frac{2^{56}}{60 * 60 * 24 * 365 * 10^6 * 1000}$, or more than 500-year so that the illegal attempt to reveal the secret is not feasible within an acceptable computational time. Nevertheless, there have been some attacks proposed for breaking the traditional DES cipher, such as the differential cryptanalysis attack [3] and the linear cryptanalysis attack [13]. However, these attacks are capable of being foiled by some variations of DES [2], these variants can be implemented in existing hardware so that their high speed and secure feature are both retained. Therefore, the DES-like crypto-system is still sufficiently secure to be considered in our scheme. Next, we consider the security manner of the pseudo-random-number generator used to randomly choose the embedded pixel in the host-image. In our scheme, the host-image is given as 512×512. The possible combinations for this host-image to accommodate the secret images of the sizes 512×256 and 256×256 are $P(512 \times 512,\ 512 \times 256)$ and $P(512 \times 512,\ 256 \times 256)$, respectively, where $P(n, r) = \frac{n!}{(n-r)!}$ denotes the number of r-permutations of n distinct objects. The resulting combinations are far greater than 2^{56} of the previous number in DES cryptanalysis. That is to say, the possibility to conjecture a right location distributed among the pixels of stego-image is very low, so that an illegal attempt to locate the right one is time-consuming and out of the limited computation time.

5. Conclusions

In this paper, a novel and efficient mechanism using a modulus operation to incorporate the secret data (with image form) into a host-image was presented. In our scheme, there are two kinds of capacity sizes with secret embedded

images. The sizes are one half and one quarter the size of the host-image, respectively. The two kinds of secrets are capable of being totally embedded into a variety of host-images in our scheme so that only the stego-image is generated, i.e., we do not need to generate the auxiliary table used in the secret extraction phase. As such a genuine secret embedding procedure is achieved in our scheme when compared to previous literature. The main advantage in our scheme is that the auxiliary table required in [23] is totally released. Furthermore, the image qualities, after the secrets are embedded, remain quite high under the criteria of PSNR measurements. In addition, the evaluated PSNRs are quite stable for accommodating the diverse host-image adoptions, as observed from our experiments. Also, in our scheme the computational complexities are simply efficient in that the DES-like hardware/software implementation is fast, and the low-bit modulus operations are applied in the secret embedding. At the same time, the security analysis shows that there is a low possibility for breaking down the secret for anyone wanting to attempt to reveal the secret on purpose. Therefore, in this study, a truly secret scheme of hiding a secret was achieved. It also satisfied the requirements of information hiding in transparency, universality, capacity and security, except for robustness. That is because we focused on the higher capacity in secret embedding.

Acknowledgements

I am indebted to K.S. Yang for our discussion on the results of experiment in the course of manuscript preparation. This work was supported in part by the National Science Council in R.O.C. under Grant No. NSC93-2213-E-015-001.

References

[1] W. Bender, D. Gruhl, N. Morimoto, Techniques for data hiding, IBM Systems Journal 35 (3–4) (1996) 313–336.
[2] E. Biham, A. Biryukov, How to strengthen DES using existing hardware, in: Advances in Cryptology: Asiacrypt'94 Proceedings, Springer-Verlag, 1995.
[3] E. Biham, A. Shamir, Differential cryptanalysis of DES-like cryptosystems, Journal of Cryptology 4 (1) (1991) 3–72.
[4] I.J. Cox, J. Kilian, T. Leighton, T. Shamoon, Secure spread spectrum watermarking for multimedia, NECI Technical Report 95-10, NEC Research Institute, Princeton, NJ, 1995.
[5] H. Eberle, A high-speed DES implementation for network application, in: Advance in Cryptology: CRYPTO'92, Springer-Verlag, New York, 1992, pp. 527–545.
[6] J. Fridrich, Robust bit extraction from images, in: 1999 IEEE International Conference on Multimedia Computing and Systems (CMCS 99), Florence, Italy, vol. 2, 1999, pp. 536–540.
[7] A. Gupta, H.M.D. Toong, An architectural comparison of 32-bit microprocessors, IEEE Micro 3 (1) (1983) 9–22.

[8] A. Herrigel, J.J.K.Ó Ruanaidh, H. Petersen, S. Pereira, T. Pun, Secure copyright protection techniques for digital images, in: D. Aucsmith (Ed.), Information Hiding, Lecture Notes in Computer Science, vol. 1525, 1998, pp. 169–190.
[9] C.T. Hsu, J.L. Wu, Hidden digital watermarks in images, IEEE Transactions on Image Processing 8 (1) (1999) 58–68.
[10] N.F. Johnson, S. Jajodia, Steganography: seeing the unseen, IEEE Computer (February) (1998) 26–34.
[11] C. Kotropoulos, I. Pitas, Adaptive LMS L-filters for noise suppression in images, IEEE Transactions on Image Processing 5 (12) (1996) 1596–1609.
[12] C.S. Lu, S.K. Huang, C.J. Sze, H.Y.M. Liao, Cocktail watermarking for digital image protection, IEEE Transactions on Multimedia 2 (4) (2000).
[13] M. Matsui, Linear cryptanalysis method for DES cipher, in: Advances in Cryptology: Eurocrypt'93 Proceedings, Springer-Verlag, 1994, pp. 386–397.
[14] N. Nikolaidis, I. Pitas, Copyright protection of images using robust digital signatures, in: IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP-96), vol. 4, May 1996, pp. 2168–2171.
[15] F.A.P. Petitcolas, R.J. Anderson, M.G. Kuhn, Information hiding—a survey, Proceedings of the IEEE 87 (7) (1999) 1062–1078.
[16] B. Pfitzmann, Information hiding terminology, in: Proceedings of the First Workshop on Information Hiding, Lecture Notes in Computer Science, Springer-Verlag, Berlin, 1996, pp. 347–350.
[17] B. Schneier, Applied cryptography, second ed., John Wiley, New York, 1996.
[18] R.G. van Schyndel, A.Z. Tirkel, C.F. Osborne, A digital watermark, in: Proceeding of IEEE International Conference of Image Processing, Austin, TX, vol. 2, November 1994, pp. 86–90.
[19] M.D. Swanson, B. Zhu, A.H. Tewfik, Robust data hiding for images, in: Proceedings of IEEE Digital Signal Processing Workshop, Loen, Norway, September 1996, pp. 37–40.
[20] G. Voyatzis, I. Pitas, Digital image watermarking using mixing systemsComputer & Graphics, vol. 22(3), Elsevier, 1998, pp. 405–416.
[21] S.J. Wang, The high capacity in embedding logo implementing the intellectual property using tree-oriented coding mechanism, Bulletin of the College of Engineering, National Taiwan University, Taiwan, no. 90, February 2004, pp. 119–126.
[22] S.J. Wang, C.K. Lu, A scheme of non-sensible document in transit with secret hiding, Journal of Information Management 9 (2) (2003) 169–182.
[23] D.C. Wu, W.H. Tsai, Image hiding in spatial domain using an image differencing approach, IEE Proceedings—Vision, Image and Signal Processing 147 (1) (2000) 29–37.
[24] W.H. Yeh, J.J. Hwang, Hiding digital information using a novel system scheme, Computers & Security 20 (6) (2001) 533–538.