Sharing secret images by using base-transform and small-size host images

Chih-Ching Thien,, Wen-Pinn Fang, and Ja-Chen Lin,

Department of Computer Science, National Chiao Tung University, Hsinchu 300, Taiwan.

Summary
A method is proposed to share and hide a given secret image S. The image is transformed to digits which are then shared to create n shadow images. To avoid attackers' attention, each shadow image is hidden further in an ordinary-looking host image to form a stego image. Any r of the n hidden shadows can be used to recover S, while less than r hidden shadows cannot. Each natural-looking stego image (with the shadow image hidden inside) is usually 1/r times smaller than that of the secret image, and hence avoids the waste of storage space or transmission time. The method surpasses our previous work (Pattern Recog. 37 (2004) 1377-1385) because the images' quality is superior. Also, we have more freedom here about the choice of the number (n) of shadow images being created. Therefore, the new method is more suitable (than the old one) for a news photographer working in an enemy area and wants to transmit photos.

Key words:
base-transform, sharing, hiding, stego images.

Introduction

Blakley[13] and Shamir[14] first independently proposed the concept of secret sharing which is designed for the protection of the key. In general, the secret sharing techniques split secret data D into n shadows in such the way that:
Any r ($r \le n$) of n shadows can be used to reconstruct D.
Any r-1 or less shadows leave D completely undetermined (i.e., all its possible values are equally like).
(Usually, because of the thresholding property of secret sharing, it is also called the (r, n) threshold scheme.). To protect a key (D), the n shadows could be held by different people. Therefore, if an opponent wants to steal the key, he/she must collect at least r shadows from many people to reveal the key. However, if without secret sharing, the opponent can get the key more easily by stealing from the person who holds the key.
From the viewpoint of safeguarding the key from being destroyed, the secret sharing method still works well. Because any r shadows are sufficient to reveal the key, some (not more than n-r) damaged shadows will not spoil the reveal of the key. If without secret sharing, once the key is just one copy and unfortunately been damaged, it is never revealed; even if the key has many copies, the risk of being grabbed will also arise.
Shamir's (r, n) threshold scheme [14] is a popular method in applications. They set the secret key as a parameter combined with r-1 random numbers to form a polynomial. By evaluating the polynomial, n shadows can be generated with n different index. The method is usually implemented in the finite field. A prime number p can form a finite field in which the key and shadows are located. Therefore, the generated shadows can be restricted within a prime number that is useful for real applications. There are other useful properties of the (r, n) threshold scheme [14]. (1) The size of each shadow does not exceed the size of the original data. (2) When r is kept fixed, shadows can be dynamically added or deleted without affecting the other shadows.
Thien and Lin [1] proposed an (r, n) threshold scheme ($r \le n$) that shared a secret image among n participants, and any r participants could cooperate to reconstruct the secret image, while r-1 or fewer participants could get nothing. In Ref. [1], each participant held his own shadow image, which contained partial information of the secret image, and the size of each shadow image was 1/r of that of the secret image. Notably, the shadow images looked like random noise rather than ordinary images. Therefore, before transmitting these shadow images via public channels (for example, internet or mobile phone), some data hiding methods were utilized in Ref. [1] to hide the shadow images in some ordinary-looking host images so as to avoid attackers' attention. However, in the hiding process, the size of each host image was quite often 2 or 4 times bigger than the size of the data to be hidden inside [2-8]. (As for the hiding methods in Ref. [9-10], although the sizes of their host images were identical to that of the hidden image, these two hiding methods could not be utilized to hide shadow images because their hiding systems are lossy and hence could not extract the shadow images exactly, which in turn caused the recovered secret image to become a meaningless noisy image.) As a result, although hiding the shadow images in the host images could avoid attackers' attention, the hiding also paid the price of size expansion (2 or 4 times expansion). To solve the size expansion problem, Wu et al. [11] proposed a method for sharing and hiding secret images without size expansions. However, the method in Ref. [11] was a little more complicated, and the PSNRs of the generated stego images were only about 34 dB. The goal of the current

Manuscript received June 5, 2006.
Manuscript revised June 25, 2006.

paper is to design a simpler approach that outperforms Ref. [11] in images' quality.
First, the original secret image (raw data) will be compressed using some lossy or lossless compression methods (such as JPEG or wavelet techniques). The compressed file is then transformed into a sequence of digits taken from a prime-number-based system. After that, the digits are then shared using a sharing procedure adopted from the one used in Ref. [11]. Finally, each generated shadow is embedded in an ordinary image by another base-transform operation and a simple hiding procedure proposed in Ref. [8].
In the remaining portion of this paper, the proposed approach is described in Sec. 2. Sec. 3 shows the experimental results. The conclusions and discussion for application are stated in Sec. 4.

2. The proposed method

As shown in Fig. 1, the proposed method could be divided into four parts: (a) Compression; (b) Base-transform; (c) Sharing; and (d) Hiding. We introduce the compression and base-transform in Subsection 2.1, and then discuss the sharing and hiding in Subsections 2.2 and 2.3, respectively. Notably, to recover the secret image is just to do the inverse operations of the above procedures.

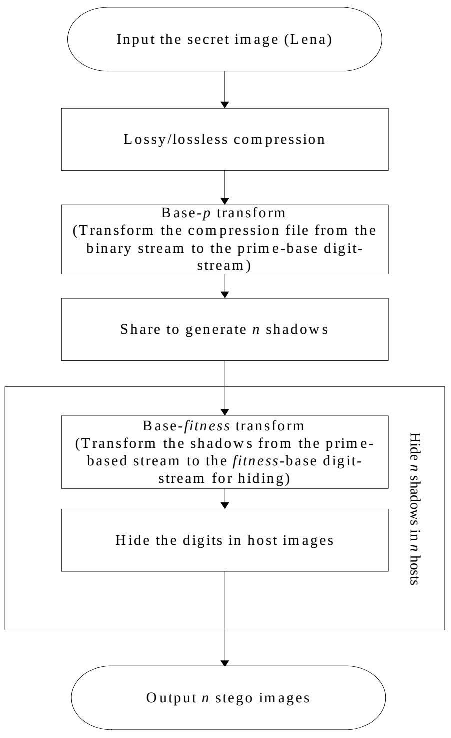

Input the secret image (Lena)
Lossy/lossless compression
Base-*p* transform
(Transform the compression file from the binary stream to the prime-base digit-stream)
Share to generate *n* shadows
Base-*fitness* transform
(Transform the shadows from the prime-based stream to the *fitness*-base digit-stream for hiding)
Hide the digits in host images
Hide *n* shadows in *n* hosts
Output *n* stego images

Fig. 1. Flowchart of the method.

2.1.2. Base-p transform

After the compression procedure, the compressed image file could be viewed as a binary stream, for example, 0011100100101…0010101, regardless of what compression type (lossless/lossy) or format (JPEG/wavelet/gif/…) was used. Now, transform the binary stream into a digit stream of which the numeric base is a specified prime number p. The reason we require that the base p is a prime number is due to the fact that the output of the transformation will be used later as the input of the sharing procedure which requires the digits being prime-number based. The transformation could be easily done using a look-up table similar to Table 1. Without the loss of generality, assume that the base is 7 (i.e. each transformed digits will fall within the range 0~6). Now, read in 2 bits from the binary stream. If the 2-bit binary code could be mapped to a digit according to the look-up table (in fact, only the 2-bit binary code 11 can be mapped to a digit [the digit 6] according to the look-up Table 1(a)), then a new digit is obtained, and we thus continue to read in the next not-yet-processed 2 bits and try to create next digit. Otherwise, read in the next not-yet-processed bit to form a 3-bit binary code, and then find the corresponding digit in the look-up table. (For example, if the 2-bit binary code is "10" and the next bit is "0", then the 3-bit binary code is formed as 100 which could be mapped to the digit "4" according to the look-up Table 1(a).) The base-p transform procedure is given below.

Base-p Transform Procedure

Input: a binary stream and a look-up table for base-p transform, where $p \geq 3$ is a prime number.
Output: a digit stream of base-p (i.e. each digit in the stream falls in the range of 0~p-1).
Steps:
Step 1. Calculate the values of $\lfloor \log_2 p \rfloor$ and the $\lceil \log_2 p \rceil$. (For example, if p=7, then $\lfloor \log_2 7 \rfloor = 2$ and $\lceil \log_2 7 \rceil = 3$.) Notably, in a look-up table, a binary code has either $\lfloor \log_2 p \rfloor$ bits or $\lceil \log_2 p \rceil$ bits.
Step 2. Read in the next not-yet-processed $\lfloor \log_2 p \rfloor$ bits. If the just-read-in $\lfloor \log_2 p \rfloor$ bits can be mapped to a digit according to the look-up table, then output the mapped digit and go to Step 4; otherwise, go to Step 3.
Step 3. Read in the next not-yet-processed bit i from the binary stream, and form a binary code of $\lceil \log_2 p \rceil$ bits by appending the bit i to the $\lfloor \log_2 p \rfloor$ bits that was temporarily kept in Step 2. Output the corresponding digit found in the look-up table using the just-formed binary code of $\lceil \log_2 p \rceil$ bits.

Step 4. Repeat Steps 2~3 until all bits in the binary stream are processed.

Table 1. Two example of the look-up tables used in base-transform.

Digit	Binary code
0	000
1	001
2	010
3	011
4	100
5	101
6	11

Base-7 look-up table

2.2 Sharing

We had described in Subsection 2.1 that the image compression and base-transform together convert the original secret image into a digit stream of base-p (note that p is a prime number and used as an input parameter here). In the current section, the digit stream of base-p is shared. The sharing procedure is just a slight modification of the one that we designed in Ref. [11]. For the readers' benefits, we will still list below the sharing procedure. The procedure creates n shadow images; and some days later, we may collect any r of these n shadow images to reconstruct the digit stream of base-p.

Sharing Procedure

Input: the digit stream of base-p.

Parameters setting: r and n where $r \le n \le p$.

Output: n shadow images.

Steps:

Step 1. Divide the digit stream into non-overlapping blocks, each containing r digits. Set counter j to the initial value j=0.

Step 2. j←j +1. Then read in the j-th block of the digit stream. Assume that the r digits of the block j are $(a_0, a_1, \cdots, a_{r-1})$.

Step 3. Use the polynomial

$$q_j(x) = (a_0 + a_1 x + \cdots + a_{r-1} x^{r-1}) \bmod p, \qquad (1)$$

to generate n values $q_j(1) \sim q_j(n)$. Then use these n values, respectively, as the j-th pixel value of the n shadow images 1~n.

Step 4. Repeat Steps 2~3 until all blocks are processed.

2.3 Hiding

As stated in Sec. 2.2, the digit stream of base-p is shared among n shadow images. However, each shadow image looks like random noise and it will attract the attackers' attention. Hence, the probability of incurring destruction also increases. To overcome this problem, a high capacity hiding method that we proposed earlier in Ref. [8] is adopted and used here to hide the shadow images. First, we take n ordinary-looking gray-value images (non-secret images) called host images. Second, in order to obtain better image quality of the stego images (the modified host images that contains the embedded shadow images), each shadow image (in which each pixel is a base-p digit) should be transformed further into a digit stream of base-fitness. The value of "fitness" is an integer calculated according to the sizes of the shadow and the host image. The primary goal of this transformation is to evenly hide the shadow data in each pixel of the host image. For example, assuming that the shadow is a base-7 digit stream of 5 000 digits, and the host image is of size 100×100 $(= 10\ 000)$. If the 5 000 base-7 digits are hidden in the host image in a manner of one digit per pixel, then the first 5 000 pixels will degrade quite a lot, while the other 5000 pixels have no distortion at all. In this case, the stego image will look quite abnormal. Thus, if we transform the 5 000 base-7 digits into a stream of 10 000 base-4 digits, then each pixel of the stego image will contain a base-4 digit so that the distortions can be averagely distributed among all pixels of the stego image, and hence make the stego image look more natural. In this example, the suitable base (base-4) is called base-fitness. In general, after the n shadow images are transformed into n streams of base-fitness digits, we respectively hide the n streams (of base-fitness) in n selected host images using the hiding method in Ref. [8] with the parameter m needed in Ref. [8] be set to m = fitness (because all digits in the digit stream of base-fitness are 0~ fitness-1). The hiding shadow procedure is listed below.

Hiding Shadow Procedure

Input: a shadow and a host image.

Output: a stego image.

Steps:

Step 1. (Base- fitness transform)

a. Calculate the parameter

$$fitness = \left\lceil p^{(SS/HS)} \right\rceil \qquad (2)$$

where SS and HS are, respectively, the number of pixels in the shadow image and the host image.

b. Transform the shadow image to a digit stream of base-fitness.

Step 2. Set pixel counter i to the initial value i=0.

Step 3. i←i+1. Then read in the i-th pixel value H_i of the host image.

Step 4. Read in the i-th digit g_i of the digit stream of base-fitness.

Step 5. Using Eqs. (3)-(6) in Ref. [8] to hide g_i in H_i (by replacing the y_i in Ref. [8] with H_i; and the x_i with g_i).

Step 6. Repeat Steps 3~5 until all pixels are processed.

3. Experimental Results

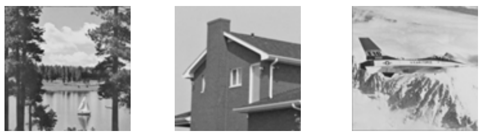
In the first experiment, we set the parameter values n=6, r=4, and the original secret image Lena (shown in Fig. 2(a)) was compressed by the lossy JPEG technique with compression ratio being 9. Six stego images (shown in Figs. 3 (a1)-(a6)) were generated, and the lossy secret image could be reconstructed by collecting any four of these six stego images. The only distortion of the secret image was due to the use of the lossy JPEG compression in Sec. 2.1.1, no further loss was caused by the operations in Sections 2.1.2, 2.2, and 2.3. Notably, the secret image was 512×512 in size, while all host images and stego images were 256×256 (and hence 1/r = 1/4 times smaller than the secret image). We had tested several values for the setting of the prime number p (see Sec. 2.1.2) by using p=11, 13, 17, and 251. Using different values of p seemed to make no noticeable differences to the final quality of the stego and recovered secret images. This is because the adjustment equation (2) automatically fine tunes the pixel distribution of hiding. However, since we require $n \leq p$ (see Sec. 2.2), if the readers wish to create more shadow images (and hence, more stego images), then a bigger value should be used for the parameter p. Also note that $n \leq 17$ was required in Ref. [11]; therefore, we have more freedom here in this paper about the choice of the number (n) of shadow images being created (as long as we use a bigger p).
The PSNRs of the six stego images were not identical, but all were around the value of 52.2 dB. When people want to retrieve the secret image, they could reconstruct it from any four of the six stego images. The reconstructed secret image was shown in Fig. 3(b), and its PSNR was 38.06 dB. Note that Fig. 3(b) was independent of which four of (a1)~(a6) in Fig. 3 were used. Any combination gave this identical 38.06 dB image.

(a)

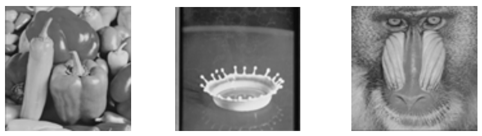
(b1) (b2) (b3)

(b4) (b5) (b6)

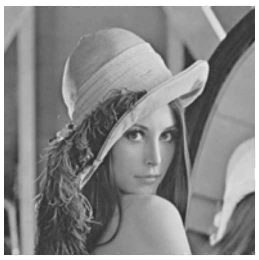
Fig. 2. The input. (a) the 512×512 secret image Lena, (b) the 256×256 host images.



(a1) (a2) (a3)



(a4) (a5) (a6)



(b)

Fig. 3. The output when Lena was first compressed using a ratio of 9. (a) the stego images (PSNRs are all about 52.20dB), and (b) the recovered secret image (whose PSNR was 38.06dB) obtained using "any" four of the six stego images.

Different reconstructed image qualities (using different compression ratios in Sec. 2.1.1) of the secret image Lena were also tested. As shown in Table 2, when the secret image Lena was compressed with lossy version and the compression ratios were 3.03~12.58, then the corresponding PSNRs of the reconstructed Lena were 44.03~36.5 dB, and the PSNRs of the stego images became 41.25~53.6 dB. Notably, the tradeoff of the image quality between the stego images and the recovered image was obvious: the higher the PSNR of the recovered secret image, the lower the PSNRs of the stego images. The explanation is quite simple: the higher the quality of the recovered Lena means the smaller the compression ratio, which also means the larger size of the compressed file, and thus makes the hiding load heavier, and therefore causes more distortion to the host images. In application, people can select the proper compression ratio to meet their own needs. As a general rule, the choice with compression ratio being around 4 should be a good choice when a reader wants to have a balance between the qualities of the stego images and recovered Lena, because both are above 41 dB.

Also note that if people wish to recover the loss-free secret image Lena, then the PSNR of the (256×256)-size stego images will only be 32 dB. If people are not satisfied with the 32 dB quality of the stego images, then the host images of a larger size can be used to improve the stego images' quality (because larger host images have more pixels, and hence, yield a smaller value for the parameter "fitness" (see Eq. (2)) which in turn results in less distortion of the host images). This is described in Table 3. In that experiment, host images of different sizes were used to hide the lossless compressed secret image. (Notably, other parameters were not changed, that is, n=6, r=4, and the secret image Lena was still 512×512). As expected, the image quality of the stego image was improved as the sizes of the host images increased.

Table 2. The PSNRs of the recovered Lena and the stego images (different compression ratios for Lena were used). The four lossy versions of the secret image Lena were compressed using the JPEG technique, while the lossless one (the last column) was compressed by a lossless wavelet compression method. All stego images were 256×256.

Compression ratio for Lena	12.58	9.00	4.67	3.03	1.85
PSNR of recovered Lena	36.50 dB	38.06 dB	41.21 dB	44.03 dB	Lossless
Average PSNR of 6 stego images	53.60 dB	52.20 dB	46.28 dB	41.25 dB	32.00 dB

Table 3. The PSNRs of the stego images with different sizes of host images. The recovery of the 512×512 secret image Lena was lossless for each case listed in this table. (the compression ration is the same with Table 2)

Size of each stego image	256x256	280x280	300x300	420x420	512x512
verage PSNR of 6 tego images	32.00 dB	35.59 dB	38.39 dB	46.38 dB	49.00 dB

Finally, we listed in Table 4 the comparison of the experimental results between the current approach and Ref. [11]. With the same size of the host images, the proposed method had better PSNRs than the method in Ref. [11] on both the recovered and the stego images. In fact, even if the proposed method used the smaller 256×256 host images, the PSNRs of the recovered Lena (49.73dB) and the stego images (35.65dB) were still competitive to their opponents (49.21dB and 34.00dB, respectively) when the method in Ref. [11] used 280×280 host images.

Table 4. Comparison of the PSNRs between the proposed method and Ref. [11].

	The proposed approach			Method in Ref. [11]		
ze of the stego images	256x256		280x280	256x256		280x280
PSNR of recovered image (Lena)	8.06dB	.73 dB	.73 dB	ossless	7.9 dB	49.21 IB
verage PSNR of stego images	2.20dB	.65 dB	.09 dB	.59dB	.00 dB	34.00 IB

4. Summary and Concluding Remarks

In this paper, an approach to generate n stego images (each contains one shadow image) has been proposed so that any r of the n stego images can be used to recover the secret image with acceptable quality (or with perfect loss-free quality, if moderate size expansion is allowed), while less than r stego images cannot (because the recovery of the r coefficients $(a_0, a_1, \cdots, a_{r-1})$ in Eq. (1) requires r of the n values $q_j(1) \sim q_j(n)$). The secret image is first compressed using some lossy or lossless compression methods. Then the compressed file is transformed into digits based on a prime number p. The transformed digits are then shared and n shadow images are thus created. In order to reduce the impact to host images in which these shadow images are to be hidden, each generated shadow image is further transformed to obtain a digit stream of base-fitness according to the sizes of the generated shadow and the host image. Each digit stream of base-fitness is then embedded in a host image to get the desired stego image by a simple hiding method that we proposed in Ref. [8].

When the secret image is compressed with lossy but acceptable quality (say, 38~44 dB), experimental results show that the PSNR quality of the stego images is high (say, 41~52 dB), and the visual quality is also good (see Figs. 3(a1)~(a6)). All these happen when the size of each stego image is only 1/r of that of the secret image. In other words, all these happen when there is no size extension (because the total size of the r stego images needed for the reconstruction is r$\times$1/r =100% of the secret image size). However, if lossless recovery of the secret image is desired, then size expansion might be necessary. Without size expansion, the stego images are about 32 dB. If the readers are not satisfied with the 32dB quality of the stego images, they can use larger size images for hiding. For example, as shown in Table 3, the stego images have 49 dB quality when each stego image has size identical to that of the 512×512 secret image Lena.

The experiments in Table 4 also show that the proposed method surpasses Ref. [11] in the quality of both the stego and the recovered secret image. Besides, the current approach is simpler than Ref. [11] and hence easier to implement. Also note that Ref. [11] required the number (n) of generated stego images to satisfy $n \leq 17$ (while we require $n \leq p$); therefore, we have more freedom here about the choice of the number of stego images being created (as long as we use a bigger value for the prime-number parameter p; for example, set p to 29 or 251).

It has been shown in Ref. [11] that, when the factor of the total-storage-space needed is also considered, the method in Ref. [11] outperforms almost all existing image-sharing methods in image quality (the discussion is given in Pages 1382-1383 of Ref. [11]). For example, it is quite obvious that the quality of the image LENA shown in Fig. 4(b) of Ref. [11] is better than that of the image shown in Fig. 14 of Ref. [12] although the decoding of [12] is faster. (Interested readers may also inspect the image quality of the remaining sharing methods appeared in [11-15] of the reference list of Ref. [11]. Some of these sharing methods are visual-cryptography-based, some are vector-quantization-based.) Thus, the image-quality superiority of the current method to the one in Ref. [11] means that the image quality is a major advantage of the proposed method. The proposed method still has the two good properties that Ref. [11] had: the small-size property of stego images (because each one is usually 1/r times smaller than the secret image), and the fault-tolerant ability of the reconstruction (in the sense that n-r of the n transmitted channels are allowed to be collapsed in the reconstruction of the secret image). Notably, as discussed in Ref. [11], the fault-tolerant ability enables a photographer who works in an enemy area to send a sensitive photo back to his news agency. The photographer can create n stego images of that secret photo and transmit the stego images (along with many other coy photos) through n distinct channels such as internet, e-mail, ftp, cell phone, or even public-accessible web pages where everybody posts photos. The real-time interception and decoding by the enemy is usually very difficult because they have to know which channels to intercept and they have to intercept at least r channels, and in each of these channels they have to identify which of the intercepted photos are not coy. After that, the enemy still has to extract and decode the noisy shadow images which could have been encrypted earlier by the photographer using special encryption keys. The decoding of each shadow image will not be easy for the enemy because it requires not only the value of the parameter p (see Eq. (1)) but also the encryption key specially designed for the channel transmitting that shadow image. Notably, although encryption can improve the security of sharing, encryption cannot replace the role of image sharing. Interested readers can refer to Ref. [11] for more detail about this fact.

Acknowledgments

This work is supported by National Science Council, Taiwan, R.O.C., under Grant NSC 94-2213-E-009-093.

References

[1] C.C. Thien and J.C. Lin, Secret image sharing, Computers & Graphics 26(5) (2002) 765-770.

[2] F.A.P. Petitcolas, R.J. Anderson and M.G. Kuhn, Information hiding – A survey, Proceedings of IEEE 87 (7) (1999) 1062-1078.

[3] D.C. Wu and W.H. Tsai, Data hiding in images via multiple-based number conversion and lossy compression, IEEE Transaction on Consumer Electronics, 44(4) (1998) 1406-1412.

[4] C.K. Chan and L.M. Cheng, Hiding data in images by simple LSB substitution, Pattern Recognition 37(3) (2004) 469-474.

[5] M. Ramkumar and A.N. Akansu, Capaciry Estimates for Data Hiding in Compressed Images, IEEE TRANSACTIONS ON IMAGE PROCESSING, 10(8) (2001) 1252-1263.

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

[7] D.C. Wu and W.H. Tsai, Spatial-domain image hiding using image differencing, Vision, Image and Signal Processing, IEE Proceedings 147(1) (2000), 29-37.

[8] C.C. Thien and J.C. Lin, A simple and high-hiding capacity method for hiding digit-by digit data in images based on modulus function, Pattern Recognition 36(12) (2003) 2875-2881.

[9] Y.C. Hu, Grey-level image hiding scheme based on vector quantization, Electronics Letters 39(2) (2003), 202-203.

[10] K.L. Chung, C.H. Shen and L.C. Chang, A novel SVD- and VQ-based image hiding scheme, Pattern Recognition Letters 22(9) (2001), 1051-1058.

[11] Y.S. Wu, C.C. Thien, and J.C. Lin, Sharing and hiding secret images with size constraint, Pattern Recognition 37(7) (2004) 1377-1385.

[12] C.C. Lin and W.H. Tsai, Visual cryptography for gray-level images by dithering techniques, Pattern Recognition Letters 24 (2003) 349-358.

[13] G.R. Blakley, “Safeguarding cryptographic keys,” Proceedings AFIPS 1979 National Computer Conference, New York, vol. 48, pp. 313-317, 1979.

[14] A. Shamir, “How to share a secret,” Communication of the ACM, vol. 22, no. 11, pp. 612-613, 1979.

[15] J. Benaloh and J. Leichter, "Generalized Secret Sharing and Monotone Functions," Advances in Cryptology - CRYPTO '88, Volume: 403, Springer-Verlag, 1989, pp. 27-35.

[16] C.S. Tsai and C.C. Chang, "A Generalized Secret Image Sharing and Recovery Scheme," Advanced in Multimedia

Information Processing, Lecture Notes in Computer Science, Volume: 2195, 2001, pp. 963-968.
[17] C.S. Tsai, C.C. Chang, and T.S. Chen, "Sharing Multiple Secrets in Digital Images," Journal of Systems and Software, Volume: 64, Issue: 2, November 15, 2002, pp. 163-170.
[18] Jen-Bang Feng, Hsien-Chu Wu, Chwei-Shyong Tsai, and Yen-Ping Chu, "A New Multi-Secret Images Sharing Scheme Using Largrange's Interpolation," Journal of Systems and Software (JSS), Vol. 76, No.3, pp. 327-339, June 2005.

Chih-Ching Thien was born in Taiwan, Republic of China, in 1975. He received his Ph.D degree in computer and information science from National Chiao Tung University in 2003. His recent research interests include data hiding, pattern recognition, and secret image sharing. He is a member of the Phi-Tau-Phi Scholastic Honor society.

Wen-Pinn Fang received his BS degree in mechanical engineering in 1994 from National Sun-Yet-Sen University and his MS degree in mechanical engineering in 1998 from National Chiao Tung University, where he is currently a PhD candidate in the Computer and Information Science Department. His recent research interests include pattern recognition and image processing..

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 from Purdue University, U.S.A., in 1988. In 1981 to 1982 he was an instructor with the National Chiao Tung University and from 1984 to 1988 he was a graduate instructor with Purdue University. In August 1988 he joined the Department of Computer and Information Science at National Chiao Tung University, where he is currently a professor. His recent research interests include pattern recognition and image processing. Dr. Lin is a member of the Phi- Tau-Phi Scholastic Honor Society.