A Secret Image Sharing Scheme Based on Vector Quantization Mechanism

Chin-Chen Chang [1,2], Chi-Shiang Chan [1], and Yi-Hsuan Fan [1]

[1] Department of Computer Science and Information Engineering, National Chung Cheng University, Chiayi, Taiwan 621
E-mail: {ccc, cch, fyh93}@cs.ccu.edu.tw
[2] Department of Information Engineering and Computer Science, Feng Chia University, Taichung, Taiwan 40724
E-mail: ccc@cs.ccu.edu.tw

Abstract. In this paper, we proposed an image sharing method adopting (*t*, *n*) threshold scheme. The secret image goes through Vector Quantization (VQ) scheme and permutes the locations of indices to obtain permuted indexed secret image. We select *t* indices to form a (*t*-1)-degree polynomial by taking *t* indices as its coefficients. Then, the values of blocks' indices in cover image are fed into the (*t*-1)-degree polynomial to get the output values. After embedding the output values back to the pixels in block, we can obtain meaningful shadow images. Any *t* out of *n* shadow images can reconstruct the permuted indexed secret image. By inversing permutation and decoding VQ indices, we can finally reconstruct the secret image. Experimental results show that our method can get high image quality for the resultant shadow images and reconstruct secret images as well.

Keywords: Image sharing, (*t*, *n*) threshold scheme, vector quantization.

1 Introduction

As the rapid growth of communication, texts and images are often digitized in order to be processed, stored, and transmitted on the wide Internet easily. For these exposed data, security is the most important concern anywise. To protect a secret file, the simplest way is to encrypt it with a secret key. However, there are two drawbacks for such a system. If illegal people can obtain the secret key to decrypt the secret file, the file might be likely stolen. Besides, the secret file cannot be revealed completely if the secret key is lost or destroyed accidentally. To solve the safekeeping and distribution problems, it is necessary to design a secret key management to protect the secret key on multi-party secure protocols [4, 5, 6, 8]. Secret sharing method is one formal strategy of the possible solutions.

The first secret sharing scheme called (*t*, *n*) threshold scheme was proposed by Shamir [11] and Blakley [1] independently. It can distribute and share secret information among *n* participants in such a way and no one can obtain any useful information from the shared data. Referring to security, any *t* qualified participants of

n can reconstruct the secret completely, but any *t*-1 have no information about the secret absolutely. The concept of secret sharing has been used widely in many commercial or military applications, such as launching a missile, opening a bank vault or a safety deposit box, etc. In addition, there are lots of researches and studies that have already been proposed, which focus on improving the security of the system [2, 3, 12].

Furthermore, secret sharing schemes have been applied to secret images sharing. The secret data are by far secret images in the application. However, directly using the (*t*, *n*) threshold scheme requires large memory space. To improve this drawback, Thien and Lin proposed a secret image sharing method derived from the original (*t*, *n*) threshold scheme in [9]. Thien and Lin's method generates a (*t*-1)-degree polynomial for each *t* gray value and set the *t* gray values as the coefficients. Here, we must label a fixed number for each shadow and take this number as a parameter input to feed into the (*t*-1)-degree polynomial to get result values. Consisting of those result values, then we can recover the shadow image.

Nevertheless, in Thien and Lin's method [13], we can see nothing else but some random gray level pixels. Moreover, each shadow image must be labeled a fixed number. Once the labeled number is lost unfortunately, this shadow image will be invalid. Another drawback of Thien and Lin's method is that any gray pixels, whose values are larger than 250 in a secret image, must be truncated to 250.

To improve those drawbacks, we shall propose a novel secret image sharing method in this paper. In our method, we do not need to label each shadow image. Instead, the parameter of each shadow image is set according to the VQ-index of each block. Furthermore, we embed sharing data into cover images so that the shadow images can be meaningful images to achieve the secret date transmission, rather than random-dot images.

The rest of this paper is organized as follows. To begin with, we shall review the related works in Section 2. Then, we will continue to present our method in Section 3. In Section 4, we shall offer our experimental results to demonstrate the effectiveness of our new method. Finally, the conclusions will be given in Section 5.

2 Related Works

Shamir first proposed the (*t*, *n*) threshold scheme that uses a polynomial to process the secret sharing. The final purpose is to split a secret data *S* into *n* shadows, and any *t* shadows can be used to reconstruct the secret data. If the number of shadows is less than *t*, the secret data cannot be revealed. To achieve this purpose, we first choose a prime number *p* and *t*-1 random numbers, a_1, a_2, …,a_{t-1}. Then, we build a (*t*-1)-degree polynomial with filling a_0 , a_1, …,a_{t-1} into this polynomial, where a_0 is the secret data *S*. The (*t*-1)-degree polynomial is shown below.

$$f(x) = (a_0 + a_1 x + a_2 x^2 + ... + a_{t-1} x^{t-1}) \mod p. \quad (1)$$

Once we have a (*t*-1)-degree polynomial, we can choose *n* random numbers y_0, y_1, …, y_{n-1} and calculate *n* values $d_0 = f(y_0)$, $d_1 = f(y_1)$, …,$d_{n-1} = f(y_{n-1})$. Those *n* shadows are (y_0, d_0), (y_1, d_1), …(y_{n-1}, d_{n-1}). If we get any *t* shadows, we can reconstruct

the (t-1)-degree polynomial by using Largrange's interpolation. It goes without saying that the secret data S can be obtained through the reconstructed the (t-1)-degree polynomial.

In 2002, Thien and Lin applied (t, n) threshold scheme to process the image sharing. The main idea of Thien and Lin's method is to fill t gray pixel values of the secret image into a (t-1)-degree polynomial as its coefficients. The scheme then calculates $f(1)$, $f(2)$, …, $f(n)$. The value $f(1)$ is taken as the gray pixel value of the first shadow image, and $f(2)$ is taken as the gray pixel value of the second shadow image, and likewise. Finally, we can obtain n shadow images. Having any t shadow images, we can successfully recover the original secret image. Note that the number which shadow image belongs to must be kept in mind.

One more step must been performed before applying (t, n) threshold scheme to gray level images. That is, gray pixels, whose values are larger than 250, must be adjusted to 250. The reason is described below. The range of a pixel value is from 0 to 255 and the prime number p of the (t-1)-degree polynomial must be within this range. In Thien and Lin's method, they chose 251 as the prime number p, which is the greatest prime number no larger than 255. Once we set 251 as prime number p, the reconstructed values of a_0, a_1, …, a_{t-1} are also in the range from 0 to 250. Therefore, we must truncate all pixels of values from 251 to 255 in secret images to 250.

We now describe Thien and Lin's method step by step as below.

Step 1. Adjust the gray pixels whose values are larger than 250 to 250.
Step 2. Permute the locations of the pixels in the secret image.
Step 3. Pick up t non-taken pixels in the permuted image.
Step 4. Form a (t-1)-degree polynomial by using t non-taken pixels as its coefficients.
Step 5. Calculate $f(1)$, $f(2)$, …, $f(n)$. Put the value $f(1)$ to the first shadow image, $f(2)$ to the second shadow image, and likewise.
Step 6. Mark t non-taken pixels as taken pixels. Go back to Step 3 until all pixels are marked as taken pixels.

We here give a simple example to illustrate Thien and Lin's method. The flowchart is shown as Fig. 1. Assume that the values of t and n are 2 and 3, respectively. Because the first two values of non-taken pixels are 4 and 7, the polynomial is $f(x) = 4+7x$. We label the first shadow image, the shadow image A, as number 1, label the second shadow image, the shadow image B, as number 2, and likewise. The values of $f(1)$, $f(2)$ and $f(3)$ are 11, 18 and 25, respectively, which are the first pixel values of the shadow image A, shadow image B and shadow image C, respectively. Repeating the same procedure, we can obtain three shadow images.

When reconstructing the permuted secret image, we can pick any two from these three shadow images to do our job. For example, we here have the shadow images A and C. Because the label number of these two images are 1 and 3, respectively, we can reconstruct the polynomial $f(x) = 4+7x$ by using (1, 11) and (3, 25). The first and second pixel values of the permuted secret image are thus 4 and 7, respectively.

Therefore, we can follow the same procedure to reconstruct the whole pixel values in the permuted secret image. Adopting the inverse-permutation operation, the secret image can be revealed finally.

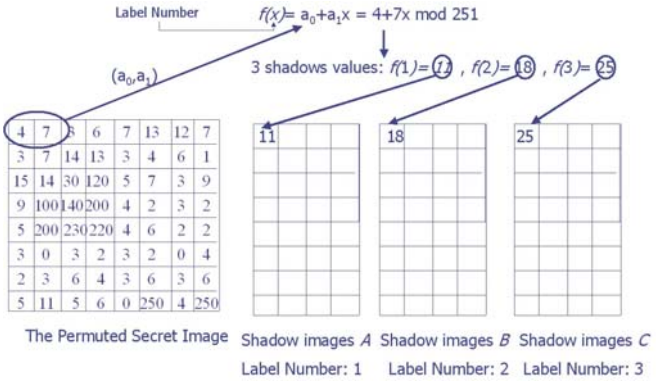

Fig. 1. An example of Thien and Lin's method

3 The Proposed Method

First of all, as we mentioned in Section 2, the results of Thien and Lin's method are *n* shadow images and those shadow images are only some random gray level pixels. This might catch the notice of grabbers. In order to enlarge the hidden quantity and make the shadow images meaningful, we encode the secret image using Vector Quantization (VQ) and embed the sharing data into cover images.

Second, in Thien and Lin's method, each shadow image must be labeled a fixed number and all these numbers must be kept in mind. If we forget any labeled number of a shadow image, this shadow image will become invalid. For instance, if we forget which label number the shadow image *A* belongs to in Fig. 1, that means we will lose one parameter to reconstruct the (*t*-1)-degree polynomial. To solve this problem, the parameter of each shadow image is set according to the VQ-index.

Third, Thien and Lin's method chose 251 as the prime number *p*. Those gray pixels, whose values are larger than 250 in a secret image, must be truncated to 250. In our proposed method, we chose 257 as the prime number *p*, so that we do not need to truncate any VQ-index value of a secret image. However, the value of $f(x)$ might be larger than 255, that is, 256. In this case, we can adjust x to make $f(x)$ in the range from 0 to 255. The details will be explained in the following paragraphs.

We now briefly introduce Vector Quantization (VQ) technique first. VQ technique is an image-compression technique. In this VQ mechanism, the codebook plays an important role, which consists of elements called codewords. Each codeword has its corresponding index that can be used to reach the codeword. All codewords are obtained from training several candidate blocks in the images using the LBG algorithm [10]. In the encoding phase, the original image is divided into non-overlapping blocks as codewords, each of which is of the same size. Therefore, we can find the most similar codewords for a sequence of non-overlapping blocks. By

only recording the corresponding indices for codewords, the image then could be compressed using VQ technique. In the decoding phase, because of holding a sequence of indices, we can reconstruct the image by looking up codewords in the codebook according to the indices. The flowchart of VQ technique is drawn as Fig. 2.

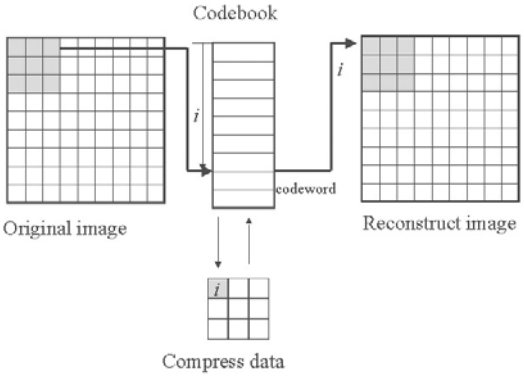

Fig. 2. The flowchart of the *VQ* method

Now, we introduce our secret image sharing scheme based on Vector Quantization (VQ). There exist two phases in our proposed method. One is sharing phase and the other is revealing phase.

Step 1. Transform a secret image into VQ indices using Vector Quantization (VQ) technique. Permute the locations of the indices of the secret image.

Step 2. Set the one least significant bit of all pixels in cover images as 0. Partition all cover images into non-overlapping blocks and use Vector Quantization (VQ) to find out indices of those non-overlapping blocks.

Step 3. Pick up t non-taken indices in the permuted indexed image and form a $(t-1)$-degree polynomial by taking t non-taken indices as its coefficients, where the value of prime number P is 257.

Step 4. Pick a block that has capacity to hide data from each cover image. Assume their indices are $I_1, I_2, \ldots, I_n$.

Step 5. Check whether there exist two picked indices whose values are the same. If there is so, randomly select one block from those two, and replace this block with another similar block in the codebook and store this similar block back to cover images. Repeat this step until all picked blocks have different indices.

Step 6. Calculate $f(I_1), f(I_2), \ldots, f(I_n)$. Check whether there exists any result valued 256. If so, replace this block with another similar one in the codebook and store this similar block back into cover images. Then, go back to Step 5.

Step 7. Embed the result value $f(I_1)$ to the one least

significant bit of all pixels in the picked block in the first shadow image, embed $f(I_2)$ in the second shadow image, and likewise.

Step 8. Mark t non-taken pixels as taken pixels. Go back to Step 3 until all pixels are marked as taken pixels.

We give a simple example with a flowchart Fig. 3 to illustrate our proposed method in the following. Assume that the values of t and n are 2 and 3, respectively.

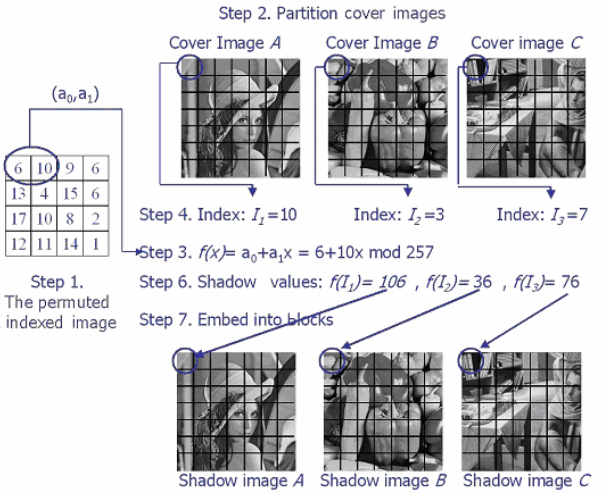

Fig. 3. The flowchart of our proposed method

The first two values of non-taken indices are 6 and 10, and the polynomial is $f(x) = 6+10x$. As for cover images, they are partitioned into 4×4 non-overlapping blocks, that is totally, there are 16 pixels in each block. As shown in Fig. 3, the index value of the first block in cover image A is $I_1 = 10$, the index value of the first block in cover image B is $I_2 = 3$, the index value of third block in cover image C is $I_3 = 7$, and the values of $f(I_1)$, $f(I_2)$ and $f(I_3)$ are 106, 36 and 76, respectively. We directly substitute one least significant bit of pixels with these result values. To be more precise, the 8-bit representation of 106 is $(01101010)_2$. And because the bit value of first bit is 0, we replace the least significant bit of first pixel in the first block of cover image A with 0. The bit value of the second bit is 1, so we replace the least significant bit of second pixel in first block of cover image A with 1, and likewise.

Note that there exist 16 pixels in each block, which means the total number of bits that can be hidden in one block is 16. However, we only embed 8 bits into a block. There still have 8 bits that can be used to hide data. Therefore, we carry out the same procedure to produce another polynomial $f(x) = 9+6x$ from another two non-taken indices. Then, we calculate the values of $f(I_1)$, $f(I_2)$ and $f(I_3)$ and they are 69, 27 and 51, respectively. The value 69 is embedded into another non-embedded 8 bits of the same block in the cover image A. After going through the same procedure, we can obtain three shadow images.

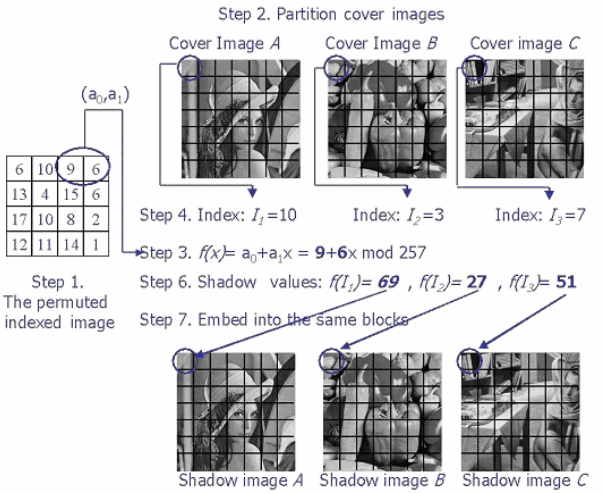

Fig. 4. The flowchart of our proposed method

In the revealing phase, if we have any two shadow images, we can reconstruct the permuted indexed secret image. Here, we pick up the shadow images *A* and *C*. We first partition these two images into 4×4 non-overlapping blocks. For the first block, we can find out its corresponding index 10 for shadow image *A* and index 7 for shadow image *C*. After that, we extract 8 bits from least significant bit of 8 pixels in first block. The extracted value from the shadow image *A* is 106 and that from the shadow image *C* is 76. Through two points (10, 106) and (7, 76), the polynomial $f(x)=6+10x$ can be reconstructed. Therefore, the first and second pixel values of the permuted indexed secret image are 6 and 10, respectively. We can use the same procedure to reconstruct the whole pixel values in the permuted indexed secret image. Performing the inverse-permutation operation, the secret indexed image can finally be revealed entirely.

4 Experimental Results

In this section, we will demonstrate our experimental results and show the image quality of our method. Our secret image F-16 is shown in Fig. 5, of size 512 × 512 pixels, and cover images are named Lena, Pepper, Barb and Baboon, the size of which is 256 × 256, as shown in Fig. 6.

Fig. 5. The secret image

(a) Lena (b) Pepper (c) Barb (d) Baboon

Fig. 6. The grayscale cover images

In our first experiment, (2, 3) threshold secret sharing scheme is performed, and the secret image is shared by three shadow images. Any two shadow images can be chosen to reveal the original secret image. The experimental results are shown in Figs. 7 and 8.

(a) A secret image (b) The VQ-compressed image of (a) (c) The reconstructed secret image

Fig. 7. The experimental results for the secret image with size 512 × 512 pixels

(a) (b) (c)

Fig. 8. Three meaningful shadow images with size 256 × 256 pixels

In our second experiment, (2, 4) threshold secret sharing scheme is carried out, and the secret image is shared by four shadow images. Similarly, any two shadow images can reveal the original secret image. The experimental results are shown in Figs. 9 and 10.

(a) A secret image (b) The VQ-compressed image of (a) (c) The reconstructed secret image

Fig. 9. The experimental results for the secret image with size 512 × 512 pixels

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

Fig. 10. Four meaningful shadow images with size 256 × 256

The quality of three shadow images in (2, 3) threshold secret sharing scheme and four shadow images in (2, 4) threshold secret sharing scheme is shown in Table 2.

Table 1. The *PSNR*'s adopting (2, 3) threshold secret sharing scheme

Image Types	Images	*PSNR* (in dB)
Shadow images	Lena	48.0931
	Pepper	46.2767
	Barb	43.4503
Reconstructed image	F-16	30.5798

Table 2. The *PSNR*'s adopting (2, 4) threshold secret sharing scheme

Image Types	Images	*PSNR* (in dB)
Shadow images	Lena	48.0931
	Pepper	46.2630
	Barb	43.4483
	Baboon	39.6525
Reconstructed image	F-16	30.5798

5 Conclusions

The (*t*, *n*) threshold scheme was proposed to process secret sharing. In 2002, Thien and Lin proposed a (*t*, *n*) threshold scheme to deal with secret image sharing. Although it works, there exist some drawbacks. In this paper, we proposed an improved method to solve those drawbacks. Moreover, Vector Quantization (VQ) is employed to further increase the hiding capacity. According to our experimental results, shadow images and the secret image can still have high quality. In a word, our new scheme, as a whole, is quite a practical method to process image sharing.

References

1. Blakley, G.. R., "Safeguarding Cryptographic Keys," *Proceedings AFIPS 1979 National Computer Conference*, vol. 48, New York, USA, no. 4-7, 1979, pp. 313-317.
2. Beimel, A. and Chor, B., "Universally Ideal Secret-sharing Schemes," *IEEE Transactions on Information Theory*, vol. 40, no. 3, 1994, pp. 786-794.
3. Beimel, A. and Chor, B., "Secret Sharing with Public Reconstruction," *IEEE Transactions on Information Theory*, vol. 44, no. 5, 1998, pp.1887-1896.
4. Brickell, E. F. and Davenport, D. M., "On the Classification of Ideal Secret Sharing Schemes," *Journal of Cryptology*, vol. 4, no. 73, 1991, pp. 123-134.
5. Chan, C. C. and Chang, C. C., "A Scheme for Threshold Multi-secret Sharing," *Applied Mathematics and Computation*, vol. 166, no. 1, 2005, pp. 1-14.
6. Chang, C. C. and Chuang, J. C., "An Image Intellectual Property Protection Scheme for Gray-level Images using Visual Secret Sharing Strategy," *Pattern Recognition Letters*, vol. 23, no. 8, 2002, pp. 931-941.
7. Gray, R. M., "Vector Quantization," *IEEE ASSP Magazine*, 1984, pp. 4-29.
8. Hwang, K. F. and Chang, C. C., "Recent Development of Visual Cryptography," *Intelligent Watermarking Techniques*, World Scientific Publishing Company, chapter 16, 2004, pp. 459-480.
9. Kamin, E. D., Greene, J. W., and Hellman, M. E., "On Secret Sharing Systems," *IEEE Transaction on Information Theory*, vol. IT-29, no. 1, 1983, pp. 35-41.
10. Linde, Y., Buzo, A., and Gray, R. M., " An Algorithm for Vector Quantizer Design, " *IEEE Transactions on Communications*, vol. 28, 1980, pp. 84-95.
11. Shamir, A., "How to Share a Secret," *Communication of the ACM*, vol. 22, no. 11, 1979, pp. 612-613.
12. Stinson, D. R., "Decomposition Constructions for Secret-sharing Schemes," *IEEE Transaction on Information Theory*, vol. 40, no. 1, 1994, pp. 118-124.
13. Thien, C. C. and Lin, J. C., " Secret Image Sharing, " *Computer & Graphics*, vol. 26, 2002, pp. 765-770.