Proceedings of the 5th WSEAS International Conference on Signal Processing, Istanbul, Turkey, May 27-29, 2006 (pp167-172)

Progressive transmission of vector-quantized images with security and fault-tolerance

Shang-Kuan Chen and Ja-Chen Lin
Department of Computer Science
National Chiao Tung University
Hsinchu, 300
Taiwan, R. O. C.

Abstract: - The proposed method is an $((r_1, \ldots, r_k), n)$-threshold sharing system: n shares are created by the sharing system to replace the index file of a given quantized image. Security is achieved through the fact that interceptors cannot view the image if they intercept less than r_1 shares. Fault-tolerance is also obtained because the VQ version of the image can be fully recovered using any r_k shares. Each share is $k / (r_1 + r_2 + \ldots + r_k)$ times smaller than the index file. The VQ image can be recovered in a progressive manner, as long as the number of received shares reaches some preset threshold values.

Key-words: vector quantization, image sharing, progressive reconstruction.

1 Introduction

Thien and Lin [1] proposed a method to share a secret image based on Shamir's [2] polynomial threshold scheme. In [1], the input image is divided into J non-overlapping sectors, and each sector has r pixels. (Therefore, the input image has $J \times r$ pixels.) For each sector j, the r coefficients $a_0, a_1, \ldots, a_{r-1}$ of the corresponding polynomial

$$q_j(x) = a_0 + a_1 \times x + \ldots + a_{r-1} \times x^{r-1} (\mathrm{mod}\, p) \text{ ------(1)}$$

are assigned as the gray values of the r pixels in the sector j. For each natural number w, people can create a share S_w, which is a finite sequence of values defined as

$$S_w = \left\{ q_j(w) \mid j = 1, 2, \ldots, J \right\} \text{.----------(2)}$$

Obviously, the size of each share is only $1/r$ of that of the original secret image, for each share only has J elements. Assume that a dealer arbitrarily picks n natural number $\{w_1, w_2, \ldots, w_n\}$ and generates n corresponding shares. According to [1-2], the secret image can be error-freely recovered using any r shares (out of the n generated shares). The method in [1] is thus "fault-tolerant" (because up to n-r shares can be missed), and is applied to other works [3-5].

Vector quantization is one of the well-known methods for image compression. When an image is quantized by a VQ scheme, the image is divided into non-overlapping blocks, and the size of each block is identical to that of each codeword of a given codebook. Then, each block is mapped to its nearest codeword in the codebook. Finally, all the mapped codeword indices are recorded in a file called the index file. Assuming that a public codebook is used; then, the index file is the only thing left (and the only thing needed) to store or unveil the image. The index file therefore plays a critical role in VQ. If the file gets lost, then the image is also gone forever. (Similarly, if the index file is stolen, then the image might also be peeped eventually.) In the reported VQ works [6-8], it is seldom discussed how to avoid the index file from getting lost. Therefore, we propose in this paper a fault-tolerant method which replaces the index file by n smaller shares. These n shares can be stored or transmitted using n distinct channels, so they can be transmitted independently. The interception of up to $r_1 - 1$ channels (r_1 is a given threshold value) will not let the thief has a chance to peep the image. On the other hand, as long as the number of shares that survive in an attack reaches another given threshold value r_k, the VQ-image can be recovered by our system exactly identical to the one recovered by traditional VQ.

The progressive image transmission is a technique to progressively approximate an image on the receiving site of the transmission. In most of previous researches [9-11], if the most significant transmitting part is lost, the recovered image is seriously damaged. Therefore, designing equally-significant and fault-tolerant parts for progressive image transmission is another kind of approach worthy a try.

When some communication channels are faster than the others, it is also quite natural to expect that the VQ-image can be recovered in a progressive manner (as long as the numbers of received shares reach some of the preset threshold values $\{r_1, r_2, \ldots, r_k\}$, where $r_1 \le r_2 \le \ldots \le r_k$). This feature is for the remote users to grasp a vague image quickly, and then obtain later finer and finer view of that image. Therefore, besides the fault-tolerance and security concern, for the vector-quantized images, the proposed method also needs to provide some vague versions of the images quickly (before the expected VQ-images are recovered completely), if some of the required r_k shares arrive much later than the other shares do.

The remaining of the paper is organized as follows. Section 2 states the encoding method. Section 3 states the decoding method. Section 4 shows the experimental results. Finally, Section 5 presents the conclusions.

2. Encoding

Firstly, a sorted codebook is generated via sorting a given public codebook so that the average gray-values of the codewords are in a non-decreasing order (for each codeword, take an average among all dimensions of the codeword). The reason that the codebook must be sorted will be explained later at the end of Section 3. Right now let us still focus on how to use this sorted codebook to quantize an image. If we have been given the index file created earlier for the given image using the "non-sorted" codebook, then, in order to obtain the new index file corresponding to the sorted codebook, we simply modify the content of that given index file. This is just an easy job of switching names, because we only have to apply the 1-to-1 mapping (which maps the codewords of the non-sorted codebook to that of the sorted codebook) to the indices of the old index file. On the other hand, if no index file is given, then, in order to get the index file, we directly use the sorted codebook and apply vector quantization to original image. In both cases, an index file is generated, and it is corresponding to the sorted codebook.

Now, let the index file be bit-transformed further by a bit-plane scanning method introduced below in Section 2.1. After that, the bit-transformed indices are shared to generate the n shares, as explained in Section 2.2.

2.1 Bit-transforming of the indices.

Let the k given threshold values be $r_1 \le r_2 \le \ldots \le r_k$, as stated earlier. The r_1 is the minimal threshold value to reveal the image in a vague version, and r_k is the threshold value to reveal the image identical to the one recovered by traditional VQ. Then, we perform the bit-transform of the index file, as described below. Firstly, the index file is divided into non-overlapping sectors so that each sector has RSUM ($= r_1 + r_2 + \ldots + r_k$) indices. The indices of each sector are then processed by a bit-plane scanning. For example, assume that RSUM=5, the five indices of a sector are (738, 770, 751, 721, 643), and each index has t bits (t = 10 if there are 1024 codewords; t = 9 if there are 512 codewords; and so on). An example of the bit-transforming using t = 10 is shown in Fig. 1, and the $\frac{5 \times t}{t} = 5$ bit-transformed values (still t bits per value) of the sector are (1000, 758, 642, 132, 935), according to the appearance order. Of course, the first several transformed values of the sector consist of the significant bits of the original indices. For instance, 1000 and 758 together record the four ($[2 \times t/5] = [2 \times 10/5]=4$) most significant bits (MSB) of the original indices in the above example.

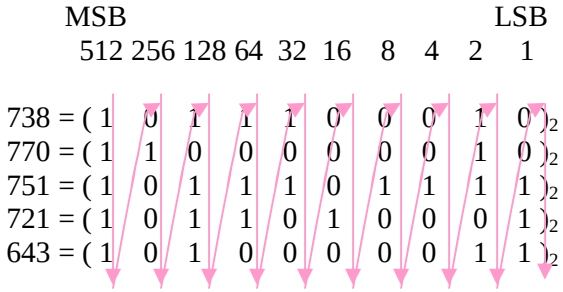

Figure 1. An example showing the bit-transforming.

2.2. Sharing the bit-transformed indices.

As stated above in Section 2.1, every $r_1 + \ldots + r_k$ indices are grouped together to form a sector, and the $r_1 + \ldots + r_k$ indices inside each sector are bit-transformed to another $r_1 + \ldots + r_k$ values. Now, these values need to be shared. In the sharing process, the $r_1 + \ldots + r_k$ values inside each sector are divided further into k sub-sectors (SS) so that each SSi has r_i values. Then, at each SS_i, the r_i values are shared using a polynomial $P_i(x)$ of degree $r_i - 1$, and its r_i coefficients are in fact the r_i values being shared. (Therefore, each sector is equipped with k polynomials (P_1(x), P_2(x), …, $P_k(x)$) of its own: polynomial P_1 is to share the first r_1 transformed values; polynomial P_2 is to share the next r_2 transformed values; and so on.) Finally, each Share j

is just a record of the values ($P_1(j)$, $P_2(j)$, …, $P_k(j)$) recorded in a sector-by-sector manner.

Notably, after bit-transforming and sharing, the index file is replaced by n shares in our scheme, and these n shares can be transmitted using n distinct channels. Intercepting up to $r_1 - 1$ channels by the hackers will not provide them sufficient information to reveal the index file which is necessary to restore the image. This is a property related to the safety of our scheme. The reason for obtaining safety is quite obvious, as explained below. $P_1(x)$ only has r_1 coefficients; however, in order to recover back its r_1 coefficients, r_1 shares will be needed in order to provide the data $\{P_1(j)\}$ at r_1 distinct values of j. Similarly, $P_2(x)$ needs r_2 shares to recover its coefficients, and so on. Since $r_1 \le r_2 \le \ldots \le r_k$, if we get less than r_1 shares, none of the coefficients of $P_1(x)$ can be recovered, and this is true for each of the k polynomials.

Besides the secure property just mentioned, also note that, the disconnection of up to $n - r_k$ channels will not influence the recovery of the vector-quantized image. This is due to the fact that each of the k polynomials has no more than r_k coefficients; therefore, if we get r_k shares (recall that each share j keeps a record of $\{P_1(j), P_2(j), \ldots, P_k(j)\}$), then we get enough information to solve for the r_i coefficients of each polynomial $P_i(x)$.

3. Decoding

The sorted codebook discussed in the encoding phase is used again here in the decoding phase. The three cases in the decoding phase of the index file are described as below:

Case 1*:* If less than r_1 shares are collected, then the r_1 coefficients of $P_1(x)$ can not be recovered, neither can the index file.

Case 2*:* If less than r_k (but not less than r_1) shares are collected, then at least the r_1 coefficients of $P_1(x)$ can be recovered, thus bit-transformed data can be partially recovered. Then, by reversing the bit-plane scanning, the index file can also be partially recovered.

Case 3*:* If at least r_k shares are collected, then the coefficients of $P_1(x)$, $P_2(x)$, …, $P_k(x)$ are all recovered, thus the bit-transformed data can be completely recovered. Then, by reversing the bit-plane scanning, the index file can also be completely recovered. The image recovered in this case will be identical to the one recovered by ordinary vector quantization algorithm.

In the overall image decoding phase, firstly, the public codebook is sorted. When the part of the index file of VQ is decoded, the image is then decoded via the sorted codebook and with vague quality. However, when the complete index file of VQ is decoded, the image is then decoded via the sorted codebook and the same quality as ordinary VQ.

Below we explain why the codebook should be sorted. When the number of received shares is less than r_k (but still not less than r_1), in order that "partial information" of the index file can still recover a vague quality image, the estimation of the indices is required. Of course, the image distortion due to wrong estimation certainly exists. Therefore, to reduce serious distortion (Without codebook sorting, the recovered image looks like a noisy image [see Fig. 9 and Fig. 10] that people cannot have idea about what the original image looks like) the public codebook is sorted in advance so that the neighboring codewords will have similar average gray-values. By doing so, we can reduce serious distortion introduced by using the neighboring codewords, when indices are estimated. To be able to estimate the indices from their partial data (so that images can be recovered progressively), the bit-plane scanning method is also used earlier in the encoding phase to generate the bit-transformed data. By doing so, the first several elements created in each sector and stored in the transformed data, can be used to recover the most significant bits of the original indices.

4. Experimental results

The experimental results are discussed in this section. Fig. 2 shows the original image Lena, which is vector-quantized to obtain an index file according to a public codebook with 1024 codewords, and the reconstructed VQ-image is shown in Fig. 3. So far, not a little bit of the proposed method is used. Then, when the codebook is sorted, the index file is also modified accordingly. Then, in the first experiment, $k = 2$ thresholds $\{r_1 = 2, r_k = r_2 = 3\}$ are used, and $n = 4$ shares are generated. Notably, each share is $\frac{k}{r_1 + \ldots + r_k} = \frac{k}{r_1 + r_2} = \frac{2}{5}$ times smaller than the index file in size, for each sector of $r_1 + \ldots + r_k$ transformed values only contributes k values $\{P_1(j), \ldots, P_k(j)\}$ to Share j (see Section 2.2), true for any $j = 1, 2, \ldots, n$. Fig. 4 shows the image recovered from the "partial" index file, which is generated when only 2 shares are received. Fig. 5 shows the recovered image when 3 or more shares are received. Note that

this recovered image is identical to the traditional VQ-recovered image shown in Fig. 3, and it should be of no surprise, since $r_k = r_2 = 3$ implies that the index file is fully reconstructed when 3 or more shares are received. In the second experiment, $k = 3$ thresholds $\{r_1 = 3, r_2 = 4, r_3 = 5\}$ are used, and $n = 6$ shares are generated. Each share is thus $\frac{3}{r_1 + r_2 + r_3} = \frac{3}{12} = \frac{1}{4}$ times smaller than the index file in size. Fig. 6 and Fig. 7 show the recovered images from the partial index files, which are generated when 3 and 4 shares are received, respectively. Fig. 8 shows the recovered image when 5 or more shares are received. Again, the image in Fig. 8 is identical to the ordinary VQ-recovered image shown in Fig. 3, since $r_k = r_3 = 5$ implies that the index file is fully reconstructed when 5 or more shares are received.

When the codebook is not sorted, for explaining how poor the quality of recovered images from the partial index files is, the above second experiment, $k = 3$ thresholds $\{r_1 = 3, r_2 = 4, r_3 = 5\}$, is adopted. Fig. 9 and Fig. 10 show the recovered images from the partial index files, which are generated when 3 and 4 shares are received, respectively.

Fig. 2. Original image

Fig. 3. The recovered image (PSNR = 31.60 dB) using ordinary VQ [6].

Fig. 4. The recovered image when only 2 shares are received in the $(r_1, r_2) = (2, 3)$ case. The PSNR is 20.97 dB.

Fig. 5. The recovered image (identical to the 31.60dB image in Fig. 5) when 3 or more shares are received in the $(r_1, r_2) = (2, 3)$ case.

Fig. 6. The recovered image when 3 shares are received in the $(r_1, r_2, r_3) = (3, 4, 5)$ case. The PSNR is 19.24 dB.

Fig. 7. The recovered image when 4 shares are received in the $(r_1, r_2, r_3) = (3, 4, 5)$ case. The PSNR is 22.37 dB.

Fig. 8. The recovered image (identical to the 31.60dB image in Fig. 5) when 5 or more shares are received in the (r_1, r_2, r_3) = (3, 4, 5) case.

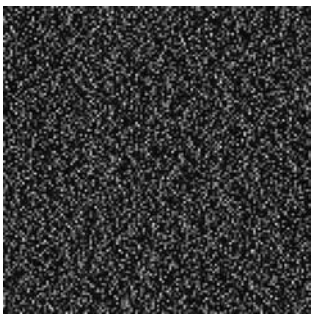

Fig. 9. The recovered image when 3 shares are received in the (r_1, r_2, r_3) = (3, 4, 5) case. The PSNR is 8.02 dB (The codebook is not sorted).

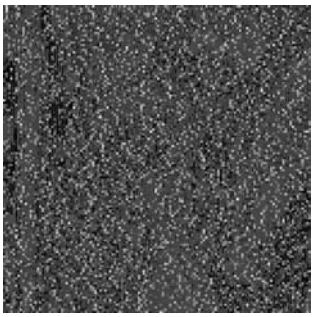

Fig. 10. The recovered image when 4 shares are received in the (r_1, r_2, r_3) = (3, 4, 5) case. The PSNR is 10.13 dB (The codebook is not sorted).

5. Conclusions

In this paper, we have proposed a method that generates *n* small shares from the index file of VQ, and then recovers the vector-quantized image (VQ-image) in a progressive and fault-tolerant manner. The method is achieved by using the "sorted" codebook, and the sharing of bit-transformed indices.

The method can be treated as a post-processing tool for any VQ algorithm. With this post-processing, there are some extra properties added to the VQ algorithm being used: 1). The proposed method reduces the chance that the VQ-image is unveiled when the index file is intercepted, for the *n* shares can be stored or transmitted in *n* distinct channels, and the thief has to intercept at least r_1 channels before he can see a vague quality image. 2). It also increases the survival rate of the VQ-image after a sequence of attacks from the attackers, for up to $n - r_k$ shares can be lost (in fact, even if $n - r_1$ shares are lost, we can still get a vague image, for example Fig. 6, of the VQ-image). 3). When the *n* shares are received from *n* distinct channels, some shares might arrive earlier than the others, so the progressive reconstruction is useful. 4). Each share is equally important, so it is unnecessary to worry about which share is transmitted first in the transmission period, or worry about which share is lost in the recovery period. 5). Since the index file is already much smaller than the image itself ($\frac{\log_2 m}{8 * z}$ times smaller, when the codebook size is *m* and each codeword has *z* gray-pixels), the fact that each share is smaller than the index file means that the storage space of each share in any of the *n* distinct channels (one share per channel) is very compact. 6). If an image is quite confidential, then we may just give up the progressive property, and use only one threshold value by setting $k = 1$ and $r_1 = r_k = r$. This ensures that the VQ-image (Fig. 3) is either fully recovered (when at least *r* shares are received) or completely invisible (when less than *r* shares are received).

References:

[1] C. C. Thien and J. C. Lin, Secret image sharing, *Computers & Graphics*, Vol. 26, No. 5, 2002, pp. 765-770.

[2] A. Shamir, How to share a secret, *Communication of the ACM*, Vol. 22, No. 11, 1979, pp. 612-613.

[3] J. B. Feng, H. C. Wu, C. S. Tsai, Y. P. Chu, A new multi-secret images sharing scheme using Largrange's interpolation, *The Journal of Systems & Software*, Vol. 76, No. 3, 2005, pp. 327-339.

[4] Y. S. Wu, C. C. Thien, J. C. Lin, Sharing and hiding secret images with size constraint, *Pattern Recognition*, Vol. 37, No. 7, 2004, pp. 1377-1385.

[5] C. C. Thien, J. C. Lin, An image-sharing method with user-friendly shadow images, *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 13, No. 12, 2003, pp. 1161 – 1169.

[6] Y. Linde, A. Buzo, and R. M. Gray. An algorithm for vector quantizer design, *IEEE Transactions on Communications*, Vol. 28, 1980, pp. 84-95.

[7] N. G. Omer and C. Hatice, Segmentation based coding of human face images for retrieval, *Signal Processing*, Vol. 84, 2004, pp. 1041-1047.

[8] A.M. Eftekhari-Moghadam, J. Shanbehzadeh, F. Mahmoudi, H. Soltanian-Zadeh, Image retrieval based on index compressed vector quantization, *Pattern Recognition*, Vol. 36, 2003, pp. 2635-2647.

[9] G. Pavlidis, A. Tsompanopoulos, N. Papamarkos, C. Chamzas, A multi-segment image coding and transmission scheme, *Signal Processing*, Vol. 85, No. 9, 2005, pp.1827-1844.

[10] H. F. Franky and D. Zhang, A new progressive colour image transmission scheme for the World Wide Web, *Computer Networks and ISDN Systems*, Vol. 30, No. 20-21, 1998, pp. 2059-2064.

[11] Y. C. Hu and J. H. Jiang, Low-complexity progressive image transmission scheme based on quadtree segmentation, *Real-Time Imaging*, Vol. 11, No. 1, 2005, pp. 59-70.