IJSEAT International Journal of Science Engineering and Advance Technology

Scalable Coding of PRNG Encrypted Images

Mr. Y. Sridhar, Mr. Bighneswar Panda
M.Tech Scholar, Assistant Professor
ECE Dept., GITAS, Piridi, Bobbli, Vijayanagaram

Abstract — This paper proposes a unique scheme of scalable coding for PRNG encrypted images. In the encryption stage, the original pixel values are masked by a modulo-256 addition with pseudorandom numbers that are resulting from a secret key. After decomposing the encrypted data into a down sampled sub image and some data sets with a multiple-resolution construction, an encoder quantizes the sub image and the Hadamard coefficients of each data set to condense the data quantity. Then, the data of quantized sub image and coefficients are observed as a set of bit streams. At the receiver side, while a sub image is decrypted to provide the uneven information of the original content, the quantized coefficients can be used to reconstruct the detailed content with an iteratively updating procedure. Because of the hierarchical coding mechanism, the principal original content with advanced resolution can be reconstructed when more bit streams are received.

Index Terms—Hadamard transform, image compression, image encryption, scalable coding.

I. INTRODUCTION

In recent years, encrypted signal processing has attracted considerable research interests [1]. The discrete Fourier transform and adaptive filtering can be implemented in the encrypted domain based on the homomorphic properties of a cryptosystem [2], [3], and a composite signal representation method can be used to reduce the size of encrypted data and computation complexity [4]. In joint encryption and data hiding, a part of significant data of a plain signal is encrypted for content protection, and the remaining data are used to carry the additional message for copyright protection [5], [6]. With some buyer–seller protocols [7], [8], the fingerprint data are embedded into an encrypted version of digital multimedia to ensure that the seller cannot know the buyer's watermarked version while the buyer cannot obtain the original product.

A number of works on compressing encrypted images have been also presented. When a sender encrypts an original image for privacy protection, a channel provider without the knowledge of a cryptographic key and original content may tend to reduce the data amount due to the limited channel resource. In [9], the compression of encrypted data is investigated with the theory of source coding with side information at the decoder, and it is pointed out that the performance of compressing encrypted data may be as good as that of compressing non encrypted data in theory. Two practical approaches are also given in [9]. In the first one, the original binary image is encrypted by adding a pseudo-random string, and the encrypted data are compressed by finding the syndromes of low-density parity-check (LDPC) channel code. In the second one, the original Gaussian sequence is encrypted by adding an independent identically distributed Gaussian sequence, and the encrypted data are quantized and compressed as the syndromes of trellis code. While Schonberg *et al.* [10] study the compression of encrypted data for memory less and hidden Markov sources using LDPC codes, Lazzeretti and Barni [11] present several lossless compression methods for encrypted gray and color images by employing LDPC codes into various bit planes. In [12], the encrypted image is decomposed in a progressive manner, and the data in most significant planes are com-pressed using rate-compatible punctured turbo codes. Based on local statistics of a low-resolution version of the image, the original content can be perfectly reconstructed. By extending the statistical models to video, some algorithms for compressing encrypted video are presented in [13]. In most of aforementioned schemes, the syndrome of channel code is exploited to generate the compressed data in a lossless manner.

Furthermore, several methods for lossy compressing encrypted images have been developed. In [14], a compressive sensing mechanism is introduced to achieve the lossy compression of encrypted images, and a basis pursuit algorithm is used to enable joint decompression and decryption. In [15], the original gray image is encrypted by pixel permutation; then, the encrypted data are compressed by discarding the excessively rough and fine information of coefficients generated from orthogonal transform. When having the compressed data and the permutation way, a receiver can reconstruct the principal content of the original image by retrieving the values of coefficients. However, the rate–distortion performance in [14] is low, and there is a leakage of statistical information in [15] since only the pixel positions are shuffled and the pixel values are not masked in the encryption phase.

This paper proposes a novel scheme of scalable coding for encrypted gray images. Although there have been a lot of works on scalable coding of unencrypted images/videos [16], [17], the scalable coding of encrypted data has not been reported. In the encryption phase of the proposed scheme, the pixel values are completely concealed so that an attacker cannot obtain any statistical information of an original image. Then, the encrypted data are decomposed into several parts, and each part is compressed as a bit stream. At the receiver side with the cryptographic key, the principal content with higher resolution can be reconstructed when more bit streams are received.

II. PROPOSED SCALABLE CODING SCHEME

a. Image Encryption

The original image is in an uncompressed format and that the pixel values are within [0, 255], and denote the numbers of rows and columns as N_1 and N_2 and the pixel number as ($N=N_1$ X N_2). Therefore, the bit amount of the original image is 8N. The content owner generates a pseudorandom bit sequence with a length of 8N. Here, we assume the content owner and the decoder has the same pseudorandom number generator (PRNG) and a shared secret key used as the seed of the PRNG. Then, the content owner divides the pseudorandom bit sequence into N pieces, each of which containing 8 bits, and converts each piece as an integer number within [0, 255]. An encrypted image is produced by a one-by-one addition modulo 256 as follows:

$$g^{(0)}(i,j)=\mathrm{mod}[p(i,j)+e(i,j),256], \qquad 1\le i\le N_1, 1\le j\le N_2$$

Where $\mathrm{p}(\mathrm{i},\mathrm{j})$ represents the gray values of pixels at positions (i,j), $\mathrm{e}(\mathrm{i},\mathrm{j})$ represents the pseudorandom numbers within [0, 255] generated by the PRNG, and $\mathrm{g}^{(0)}(\mathrm{i},\mathrm{j})$ represents the encrypted pixel values. Clearly, the encrypted pixel values $\mathrm{g}^{(0)}(\mathrm{i},\mathrm{j})$ are pseudorandom numbers since $e(i,j)$ values are pseudorandom numbers. It is well known that there is no probability polynomial time (PPT) algorithm to distinguish a pseudorandom number sequence and a random number sequence until now. Therefore, any PPT adversary cannot distinguish an encrypted pixel sequence and a random number sequence. That is to say, the image encryption algorithm that we have proposed is semantically secure against any PPT adversary.

b. Encrypted Image Encoding

Although an encoder does not know the secret key and the original content, he can still compress the encrypted data as a set of bit streams. The detailed encoding procedure is as follows. First, the encoder decomposes the encrypted image into a series of sub images and data sets with a multiple-resolution construction. The sub image at the $(\mathrm{t}+1)\mathrm{th}$ level $\mathrm{G}^{(\mathrm{t}+1)}$ is generated by down sampling the sub image at the $\mathrm{t}\,\mathrm{th}$ level as follows:

$$g^{(t+1)}(i,j)=g^{(t)}(2i,2j), \qquad t=0,1,\cdots;T-1$$

Where $\mathrm{G}^{(0)}$ just the encrypted image and T is is the number of decomposition levels. In addition, the encrypted pixels that belongs to $\mathrm{G}^{(\mathrm{t}+1)}$ but do not belong to form data set $\mathrm{Q}^{(\mathrm{t}+1)}$ as follows:

$$\mathrm{Q}^{(\mathrm{t}+1)}=\left\{\mathrm{g}^{(\mathrm{t})}(\mathrm{i},\mathrm{j})\middle|\mathrm{mod}(\mathrm{i},2)=1\,\mathrm{or}\,\mathrm{mod}(\mathrm{j},2)=1\right\}, \qquad \mathrm{t}=0,1\cdots;\mathrm{T}-1.$$

That means each $\mathrm{G}^{(\mathrm{t})}$ is decomposed into $\mathrm{G}^{(\mathrm{t}+1)}$ and $\mathrm{Q}^{(\mathrm{t}+1)}$, and the data amount of $\mathrm{Q}^{(\mathrm{t}+1)}$ is three times of that of $\mathrm{G}^{(\mathrm{t}+1)}$. After the multiple-level decomposition, the encrypted image is reorganized as $\mathrm{G}^{(\mathrm{T})},\mathrm{Q}^{(\mathrm{T})},\mathrm{Q}^{(\mathrm{T}-1)},\cdots\mathrm{and}\,\mathrm{Q}^{(\mathrm{t})}$

For the sub image $\mathrm{G}^{(\mathrm{T})}$, the encoder quantizes each value using a step Δ as follows:

$$b(i,j)=\left\lfloor \frac{g^{(T)}(i,j)}{\Delta} \right\rfloor$$

Where the operator $\lfloor \bullet \rfloor$ takes an integer toward minus infinity and

$$\Delta = 256 / M$$

Here, M is an integer shared by the encoder and the decoder, and its value will be discussed later. Clearly $0 \le b(i,j) \le M-1$

Then, the data of b (i,j) are converted into a bit stream, which is denoted as BG. The bit amount of BG is

$$N_{BG} = \frac{N}{4^T}.\log_2 M.$$

For each data set $Q^{(t)}(t=1,2,\cdots T)$ the encoder permutes and divides encrypted pixels in it into $K^{(t)}$ groups, each of which containing $L^{(t)}$ pixels $\left(K^{(t)} x\, L^{(t)} = 3N/4^t\right)$. In this way, the $L^{(t)}$ pixels in the same group scatter in the entire image. The permutation way is shared by the encoder and the decoder, and the values of $L^{(t)}$ will be discussed later. Denote the encrypted pixels of the Kth group as $q_k^{(t)}(1), q_k^{(t)}(2), \cdots, q_k^{(t)}\left(L^{(t)}\right)\left(1 \le k \le K(t)\right)$, and perform the Hadamard transform in each group as follows:

$$\begin{bmatrix} C_k^{(t)}(1) \\ C_k^{(t)}(2) \\ \vdots \\ C_k^{(t)}\left(L^{(t)}\right) \end{bmatrix} = H \bullet \begin{bmatrix} q_k^{(t)}(1) \\ q_k^{(t)}(2) \\ \vdots \\ q_k^{(t)}\left(L^{(t)}\right) \end{bmatrix}$$

Where **H** is a $L^{(t)}$ x $L^{(t)}$ Hadamard matrix made up of +1 or -1. That implies the matrix **H** meets

H'*H = H*H' = $L^{(t)}$ * I

Where $H^{(t\ )}$ is a transpose of H,I is an $L^{(t)}$ x $L^{(t)}$ identity matrix, and $L^{(t)}$ must be a multiple of 4. For each coefficient $C_k^{(t)}(l)$, calculate

$$C_k^{(t)}(l) = \left\lfloor \frac{\operatorname{mod}\left[C_k^{(t)}(l), 256\right]}{256 / M^{(t)}} \right\rfloor \qquad 1 \le k \le K^{(t)}, 1 \le k = l \le L^{(t)}$$

Where

$$M^{(t)} = \operatorname{round}\left(M / \sqrt{L^{(t)}}\right)$$

and round (.) finds the nearest integer. The remainder of $C_k^{(t)}(l)$ modulo 256 is quantized as integer $C_k^{(t)}(l)$, $L^{(t)}$, and the quantization steps are approximately proportional to square roots of $L^{(t)}$. Then, $C_k^{(t)}(l)$ at different levels are converted into bit streams, which are denoted as $BS^{(t)}$. Since $0 \le C_k^{(t)}(l) \le M^{(t)} - 1$

and the number of $C_k^{(t)}(l)$ at the th level is $3N/4^t$ the bit amount of $BS^{(t)}$ is

$$N^{(t)} = \frac{3 \bullet N \bullet \log_2 M^{(t)}}{4^t}, \qquad t = 1,2,\cdots,T$$

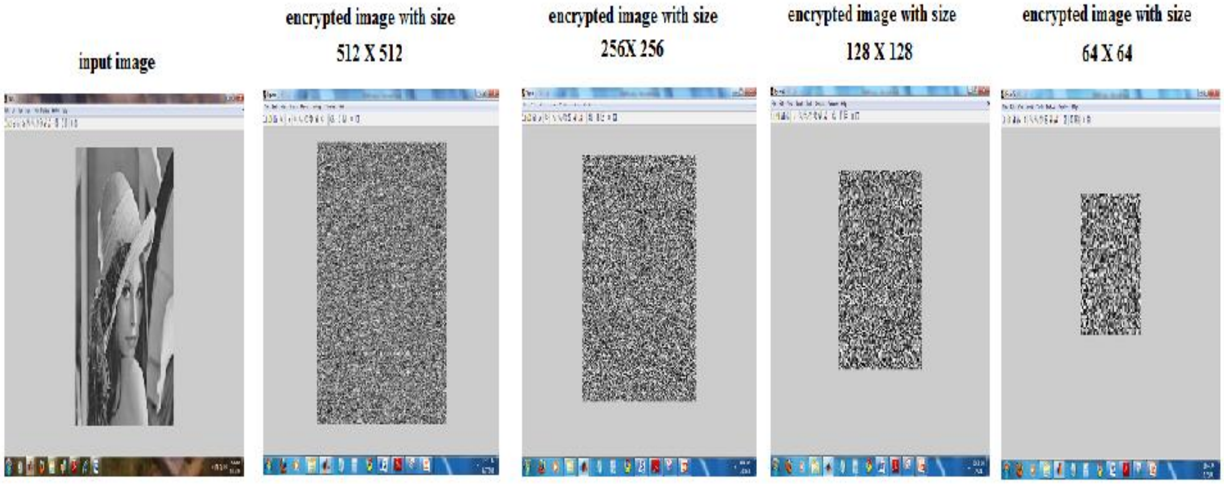

input image
encrypted image with size 512 X 512
encrypted image with size 256X 256
encrypted image with size 128 X 128
encrypted image with size 64 X 64

Fig. 1 : original image Lena and its encrypted versions

The encoder transmits the bit streams with an order of $\{\mathrm{BG}, \mathrm{BS}^{(T)}, \mathrm{BS}^{(T-1)}, \cdots, \mathrm{BS}^{(1)}\}$. . If the channel bandwidth is limited, the latter bit streams may be abandoned. A higher resolution image can be reconstructed when more bit streams are obtained at the receiver side. Here, the total compression ratio R_C, which is a ratio between the amount of the encoded data and the encrypted image data, is

$$R_C = \frac{N_{BG}}{8N} + \frac{1}{8N}\sum_{t=1}^{T} N^{(t)} = \frac{\log_2 M}{8 \bullet 4^t} + \frac{3}{8} \bullet \sum_{t=1}^{T} \frac{\log_2 M^{(t)}}{4^t}$$

c. Image reconstruction

With the bit streams and the secret key, a receiver can reconstruct the principal content of the original image, and the resolution of the reconstructed image is dependent on the number of received bit streams. While BG provides the rough information of the original content, $BS^{(t)}$ can be used to reconstruct the detailed content with an iteratively updating procedure. The image reconstruction procedure is as follows.

When having the bit stream BG, the decoder may obtain the values of $b(i,j)$ and decrypts them as a subimage, i.e.,

$$p^{(T)}(i,j) = mod\,[b(i,j).\Delta - e(2^T.i.2^T.j).256] + \frac{\Delta}{2},$$

$$1 \le i \le \frac{N_1}{2^T}, \quad 1 \le j \le \frac{N_2}{2^T}$$

Where $e(2^T.i.2^T.j)$ are derived from the secret key.

If the bit streams $BS^{(t)}$ $(\tau \le t \le T)$ are also received, an image with a size of $N_1/2^{(\tau-1)} \times N_2/2^{(\tau-1)}$ will be reconstructed. First, upsample the subimage $p^{(T)}(i,j)$ by factor $2^{(T-\tau+1)}$ to yield an $N_1/2^{(\tau-1)} \times N_2/2^{(\tau-1)}$ image as follows:

$$r\left(2^{(T-\tau+1)}.i.2^{(T-\tau+1)}.j\right) = p^{(T)}(i,j),$$

$$1 \le i \le \frac{N_1}{2^T}, \quad 1 \le j \le \frac{N_2}{2^T}$$

and estimate the values of other pixels according to the pixel values using a bilinear interpolation method.

Denote the interpolated pixel values of the *K*th group at the *t*th level as $r_k^{(t)}(1), r_k^{(t)}(2) \ldots\ldots . r_k^{(t)}(L^{(t)})$ $(1 \le k \le K^{(t)}, \tau \le t \le T)$ and their corresponding original pixel values as $p_k^{(t)}(1), p_k^{(t)}(2) \ldots\ldots p_k^{(t)}(L^{(t)})$. The errors of interpolated values are

$$\Delta p_k^{(t)}(l) = p_k^{(t)}(l) - r_k^{(t)}(l),$$

$$1 \le l \le L^{(t)}, \qquad 1 \le k \le K^{(t)}, \tau \le t \le T.$$

Define the encrypted values of $r_k^{(t)}(l)$ as

$$\hat{r}_k^{(t)}(l) = mod\left[r_k^{(t)}(l) + e_k^{(t)}(l), 256\right],$$

$$1 \le l \le L^{(t)}, \qquad 1 \le k \le K^{(t)}, \tau \le t \le T.$$

Where $e_k^{(t)}(l)$ are pseudorandom numbers derived from the secret key and corresponding to $r_k^{(t)}(l)$. Then

$$\Delta p_k^{(t)}(l) \equiv q_k^{(t)}(l) - \hat{r}_k^{(t)}(l)\ mod\ 256.$$

We also define

$$\begin{bmatrix} \Delta C_k^{(t)}(1) \\ \Delta C_k^{(t)}(2) \\ \cdot \\ \cdot \\ \cdot \\ \Delta C_k^{(t)}(L^{(t)}) \end{bmatrix} = \boldsymbol{H}\ .\begin{bmatrix} \Delta p_k^{(t)}(1) \\ \Delta p_k^{(t)}(2) \\ \cdot \\ \cdot \\ \cdot \\ \Delta p_k^{(t)}(L^{(t)}) \end{bmatrix}$$

Where **H** is a $L^{(t)} \times L^{(t)}$ Hadamard matrix made up of +1 or -1. Since only the addition and subtraction are involved in the Hadamard transform

$$\begin{bmatrix} \Delta C_k^{(t)}(1) \\ \Delta C_k^{(t)}(2) \\ \cdot \\ \cdot \\ \cdot \\ \Delta C_k^{(t)}(L^{(t)}) \end{bmatrix} \equiv \boldsymbol{H}\ .\begin{bmatrix} \Delta p_k^{(t)}(1) \\ \Delta q_k^{(t)}(2) \\ \cdot \\ \cdot \\ \cdot \\ \Delta q_k^{(t)}(L^{(t)}) \end{bmatrix} - \boldsymbol{H}\ .\begin{bmatrix} \hat{r}_k^{(t)}(1) \\ \hat{r}_k^{(t)}(2) \\ \cdot \\ \cdot \\ \cdot \\ \hat{r}_k^{(t)}(L^{(t)}) \end{bmatrix} mod\ 256$$

That means the transform of errors in the plain domain is equivalent to the transform of errors in the encrypted domain with the modular arithmetic. Denoting

$$\begin{bmatrix} \hat{C}_k^{(t)}(1) \\ \hat{C}_k^{(t)}(2) \\ \cdot \\ \cdot \\ \cdot \\ \hat{C}_k^{(t)}(L^{(t)}) \end{bmatrix} = \boldsymbol{H}\ .\begin{bmatrix} \hat{r}_k^{(t)}(1) \\ \hat{r}_k^{(t)}(2) \\ \cdot \\ \cdot \\ \cdot \\ \hat{r}_k^{(t)}(L^{(t)}) \end{bmatrix}$$

We have

$$\Delta C_k^{(t)}(l) \equiv C_k^{(t)}(l) - \hat{C}_k^{(t)}(l)\ mod\ 256$$

With the bit streams $BS^{(t)}$ $(\tau \le t \le T)$, the values of $C_k^{(t)}(l)$ can be retrived, which provide the information of $C_k^{(t)}(l)$. Therefore, the receiver may use an iterative procedure to progressively improve the quality of the reconstructed image by updating the pixel values according to $C_k^{(t)}(l)$. The detailed procedure is as follows.

1) For each group $\left[r_k^{(t)}(1), r_k^{(t)}(2) \ldots\ldots r_k^{(t)}(L^{(t)})\right]$, calculate $\hat{r}_k^{(t)}(l)$ and $\hat{C}_k^{(t)}(l)$.
2) Calculate

$$D_k^{(t)}(l) = mod\left[c_k^{(t)}(l)\,.\,\Delta^{(t)} + \Delta^{(t)}/2 - \hat{C}_k^{(t)}(l)\,.256\right]$$

$$\widehat{D}_k^{(t)}(l) = \begin{cases} D_k(l), & if\ d_k(l) < 128 \\ D_k(l) - 256, & if\ d_k(l) \ge 128 \end{cases}$$

$\widehat{D}_k^{(t)}(l)$ are the differences between the values consistent with the corresponding $c_k^{(t)}(l)$ and $\hat{C}_k^{(t)}(l)$. Then, considering $\widehat{D}_k^{(t)}(l)$ as an estimate of $\Delta C_k^{(t)}(l)$, modify the pixel values of each group as follows:

$$\begin{bmatrix} \bar{r}_k^{(t)}(1) \\ \bar{r}_k^{(t)}(2) \\ \cdot \\ \cdot \\ \cdot \\ \bar{r}_k^{(t)}(L) \end{bmatrix} = \begin{bmatrix} r_k^{(t)}(1) \\ r_k^{(t)}(2) \\ \cdot \\ \cdot \\ \cdot \\ r_k^{(t)}(L) \end{bmatrix} + \frac{\boldsymbol{H}'}{L^{(t)}}\cdot\begin{bmatrix} \widehat{D}_k^{(t)}(1) \\ \widehat{D}_k^{(t)}(2) \\ \cdot \\ \cdot \\ \cdot \\ \widehat{D}_k^{(t)}(L^{(t)}) \end{bmatrix}$$

And enforce the modified pixel values into [0,255] as follows:

$$\bar{r}_k^{(t)}(l) = \begin{cases} 0, & if\ \bar{r}_k^{(t)}(1) < 0 \\ \bar{r}_k^{(t)}(1), & if\ 0 \le \bar{r}_k^{(t)}(1) \le 255 \\ 255, & if\ \bar{r}_k^{(t)}(1) > 255 \end{cases}$$

3) Calculate the average energy of difference due to the modification as follows:

$$D = \frac{\sum_{t=\tau}^{T} \sum_{k=1}^{K^{(t)}} \sum_{l=1}^{L^{(t)}} \left[\hat{r}_k^{(t)}(l) - r_k^{(t)}(l)\right]^2}{\sum_{t=\tau}^{T} 3N/4^t}$$

If D is not less than a given threshold of 0.10, for each pixel $\hat{r}_k^{(t)}(l)$, after putting it back to the position in the image and regarding the average value of its four neighbor pixels as its new value $r_k^{(t)}(l)$, go to step 1. Otherwise, terminate the iteration, and output the image as a final reconstructed result.

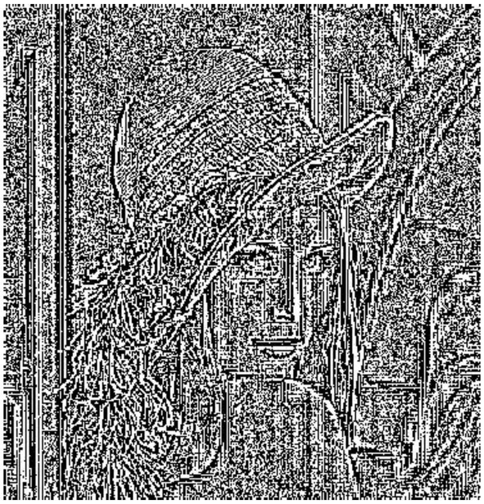


Fig. 2. Interpolated image used in both compression and decompression techniques.

III. EXPERIMENTAL RESULTS AND DISCUSSION

The test image Lena that is sized 512 X 512, 256 X 256, 128 X128 and 64X64 were used as the original images in the experiment. We let T=3 and encoded the encrypted images using M = 24, $L^{(3)}$ = 4, $L^{(2)}$ =8 and $L^{(1)}$ =24 to produce the bit streams BG, BS^3 , BS^2 , and BS^1 . In this case, the total compression ratio R_c = 0.318 . Fig. 3 gives the reconstructed Lena using {BG} , {BG $BS^{(3)}$} , {BG $BS^{(3)}$ $BS^{(2)}$} and {BG $BS^{(3)}$ $BS^{(2)}$ $BS^{(1)}$ }, respectively. Reconstructed results with higher resolution were obtained when more bit streams were used. When regarding the corresponding down sampled versions of original images as reference, the values of PSNR in reconstructed results are denoted as $PSNR_2$, $PSNR_1$. While the PSNR values of Lena are 38.4, 34, 37.1,

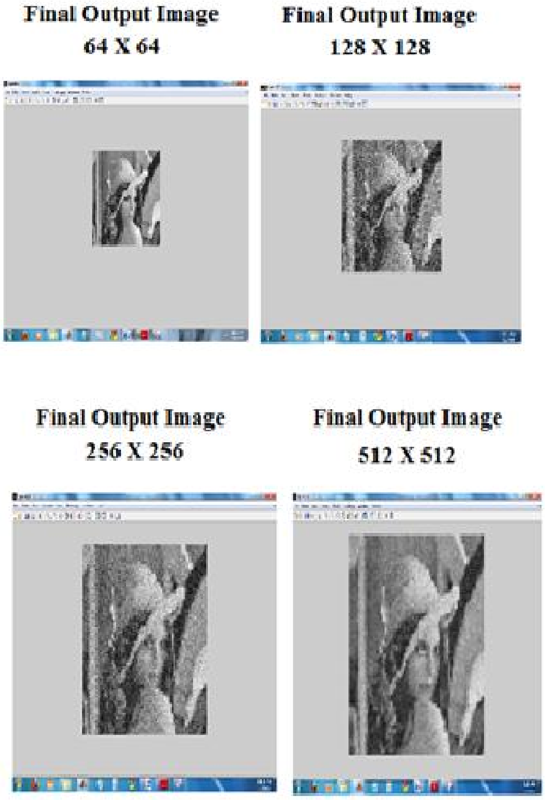
In the iterative procedure, while the decrypted pixels $p^{(T)}(i,j)$ are used to give an initial estimation of other pixels, the values of $c_k^{(t)}(l)$ in bitstreams $BS^{(t)}$ provide more detailed information to produce the final reconstructed result with satisfactory quality. In step 2, by estimating $\Delta C_k^{(t)}(l)$ according to $c_k^{(t)}(l)$, the pixel values are modified to lower the reconstruction errors. If the image is uneven and $L^{(t)}$is big, the absolute value of actual $\Delta C_k^{(t)}(l)$ may be more than 128 due to error accumulation in a group,

so that $\widehat{D}_k^{(t)}(l)$ maybe not close to $\Delta C_k^{(t)}(l)$. To avoid this case, we let $L^{(t)}$ decrease with a increasing t since the spatial correlation in a sub image with lower resolution is weaker. For instance, $L^{(1)} = 24$, $L^{(2)} = 8$, $L^{(3)} = 4$ for T=3.

M	16	18	22	24	26	30
R_C	0.235	0.275	0.287	0.318	0.323	0.356
$PSNR_2$(dB) Iteration number	34.2,32.6 16,14	34.7,32.8 4,13	36.7,33.9 9,10	37.1,33.9,**41.5**9,11	37.7,35.0 4,15	39.1,35.7 4,11
$PSNR_1$(dB) Iteration number	34.3,33.2 6,25	36.2,35.3 5,22	36.8,35.7 5,15	38.4,37.1,**18.49** 5,15	38.6,37.3 5,10	40.2,38.7 5,18

and 38.4 dB. In addition, the iterative updating procedure significantly improved the reconstruction quality. For example, while PSNR in an interpolated 512 2 512, Lena is 23.9 dB; this value in the final reconstructed image is 38.4 dB with a gain of 14.5 dB.

Table I lists the compression ratios; the PSNR in reconstructed results and the numbers of iterations with respect to different M when T=3 and encoded the encrypted images using M = 24, $L^{(3)}$ = 4, $L^{(2)}$ =8 and $L^{(1)}$ =24 and were used for image Lena.



Final Output Image 64 X 64

Final Output Image 128 X 128

Final Output Image 256 X 256

Final Output Image 512 X 512

Fig. 3. Final output images

IV. CONCLUSION

This paper has proposed a novel scheme of scalable coding for encrypted images. The original image is encrypted by a modulo-256 addition with pseudorandom numbers, and the encoded bit streams are made up of a quantized encrypted sub image and the quantized remainders of Hadamard coefficients. At the receiver side, while the sub image is decrypted to produce an approximate image, the quantized data of Hadamard coefficients can provide more detailed information for image reconstruction. Since the bit streams are generated with a multiple-resolution construction, the principal content with higher resolution can be obtained when more bit streams are received. The lossy compression and scalable coding for encrypted image with better performance deserves further investigation in the future.

REFERENCES

[1] Z. Erkin, A. Piva, S. Katzenbeisser, R. L. Lagendijk, J. Shokrollahi, G. Neven, and M. Barni, "Protection and retrieval of encrypted multimedia content: When cryptography meets signal processing," EURASIP J. Inf. Security, vol. 2007, pp. 1–20, Jan. 2007.

[2] T. Bianchi, A. Piva, and M. Barni, "On the implementation of the discrete Fourier transform in the encrypted domain," IEEE Trans. Inf. Forensics Security, vol. 4, no. 1, pp. 86–97, Mar. 2009.

[3] J. R. Troncoso-Pastoriza and F. Pérez-González, "Secure adaptive filtering," IEEE Trans. Inf. Forensics Security, vol. 6, no. 2, pp. 469–485, Jun. 2011.

[4] T. Bianchi, A. Piva, and M. Barni, "Composite signal representation for fast and storage-efficient processing of encrypted signals," IEEE Trans. Inf. Forensics Security, vol. 5, no. 1, pp. 180–187, Mar. 2010.

[5] S. Lian, Z. Liu, Z. Ren, and H. Wang, "Commutative encryption and watermarking in video compression," IEEE Trans. Circuits Syst. Video Technol., vol. 17, no. 6, pp. 774–778, Jun. 2007.

[6] M. Cancellaro, F. Battisti, M. Carli, G. Boato, F. G. B. Natale, and A. Neri, "A commutative digital image watermarking and encryption method in the tree structured Haar transform domain," Signal Process.Image Commun., vol. 26, no. 1, pp. 1–12, Jan. 2011.

[7] N. Memon and P. W. Wong, "A buyer-seller watermarking protocol,"IEEE Trans. Image Process., vol. 10, no. 4, pp. 643–649, Apr. 2001.

[8] M. Kuribayashi and H. Tanaka, "Fingerprinting protocol for images based on additive homomorphic property," IEEE Trans. Image Process., vol. 14, no. 12, pp. 2129–2139, Dec. 2005.

[9] M. Johnson, P. Ishwar, V. M. Prabhakaran, D. Schonberg, and K. Ramchandran, "On compressing encrypted data," IEEE Trans. Signal Process., vol. 52, no. 10, pp. 2992–3006, Oct. 2004.

[10] D. Schonberg, S. C. Draper, and K. Ramchandran, "On blind compression of encrypted correlated data approaching the source entropy rate,"

in Proc. 43rd Annu. Allerton Conf., Allerton, IL, 2005.

[11] R. Lazzeretti and M. Barni, "Lossless compression of encrypted greylevel and color images," in Proc. 16th EUSIPCO, Lausanne, Switzerland, Aug. 2008 .

[12] W. Liu, W. Zeng, L. Dong, and Q. Yao, "Efficient compression of encrypted grayscale images," IEEE Trans. Signal Process., vol. 19, no. 4,pp. 1097–1102, Apr. 2010.

[13] D. Schonberg, S. C. Draper, C. Yeo, and K. Ramchandran, "Toward compression of encrypted images and video sequences," IEEE Trans. Inf. Forensics Security, vol. 3, no. 4, pp. 749–762, Dec. 2008.

[14] A. Kumar and A. Makur, "Lossy compression of encrypted image by compressing sensing technique," in Proc. IEEE TENCON, 2009, pp. 1–6
.

[15] X. Zhang, "Lossy compression and iterative reconstruction for encrypted image," IEEE Trans. Inf. Forensics Security, vol. 6, no. 1, pp. 53–58, Mar. 2011.

[16] A. Bilgin, P. J. Sementilli, F. Sheng, and M. W. Marcellin, "Scalable image coding using reversible integerwavelet transforms," IEEE Trans.Image Process., vol. 9, no. 11, pp. 1972–1977, Nov. 2000.

[17] D. Taubman, "High performance scalable image compression with EBCOT," IEEE Trans. Image Process., vol. 9, no. 7, pp. 1158–1170, Jul. 2000.