Self-adaptive Encryption of JPEG Color Image Based on Chaotic Random Sequence

Di Xiao†, Tao Xiang†, Yijun Xie†, and Yong Wang‡

†College of Computer Science, Chongqing University, Chongqing 400044, China
‡Key Laboratory of Electronic Commerce and Logistics of Chongqing, Chongqing University of Posts and Telecommunications, Chongqing 400065, China
Email: xiaodi_cqu@hotmail.com, txiang@cqu.edu.cn, 20075215@cqu.edu.cn, wangyong_cqupt@163.com

Abstract–By integrating JPEG compression and the idea of self-adaptive encryption, a self-adaptive encryption algorithm for JPEG color image is proposed. DC coefficients and the first sixteen AC coefficients in Zig-Zag order of JPEG compression are chosen to construct the corresponding matrixes. Based on chaotic random sequence, self-adaptive encryption scheme is utilized to encrypt the coefficient matrixes, and the signs of DC coefficients are also encrypted to avoid color information leaking. Experimental results and theoretical analyses demonstrate that the security of the proposed algorithm and its performance are both good, and the impact on the compression is negligible. Furthermore, it is compliant with JPEG file format.

1. Introduction

The traditional encryption method is not suitable for multimedia image security transmission and storage. In order to reduce the encrypted data and redundancy, image encryption of special format gradually becomes a hot research topic. In this field, the current research mainstream includes: 1) image encryption integrated with compression coding; 2) selective image encryption. JPEG is the most popular image compression standard. Based on the above ideas, there are already quite a few research results for JPEG encryption. In Ref. [1], 2-dimensional Coupled Map Lattices are utilized to generate chaotic sequence, and the sequence is then used to encrypt DC coefficients and all the signs of AC coefficients. However, Ref. [2] has already given an attack method: if all the DC coefficients are set to be 128, and all the signs of AC coefficients are set to be positive, the attacker can obtain the resultant plaintext image. Later, Ref. [3] has also pointed out its insecurity.

In this paper, by utilizing the self-adaptive encryption idea, a joint compression and encryption algorithm for color JPEG image is proposed. The algorithm has the merits of both the image encryption integrated with compression coding and the selective image encryption. The chaotic map with good property is used as the random sequence generator. The cipher image is with good visual quality and uniform color distribution. The impact on the compression is negligible. Furthermore, it is compliant with JPEG file format.

2. The proposed algorithm

In the proposed algorithm, Piecewise Linear Chaotic Map （PWLCM） will be utilized. PWLCM with parameter u is defined as:

$$x(k+1)=C[x(k);\mu]=\begin{cases}\dfrac{x(k)}{\mu} & x(k)\in[0,\mu)\\ \dfrac{x(k)-\mu}{0.5-\mu} & x(k)\in[\mu,0.5)\\ C[1-x(k);\mu] & x(k)\in[0.5,1)\end{cases} \tag{1}$$

where $x(k)\in[0,\ 1]$, $u\in(0,\ 0.5)$ are the iteration trajectory value and the current iteration parameter of PWLCM, respectively. According to the Ref. [4], $\{x(k)\}$ is ergodic and uniformly distributed in [0, 1], and the auto-correlation function of $\{x(k)\}$ is δ-like. The initial condition $x(0)\in[0,\ 1]$ and initial parameter $u_0\in(0,\ 0.5)$ of PWLCM are set as the secret key of the proposed algorithm.

For an image of $M\times N$, let $m=M/8$ and $n=N/8$, where $m\times n$ is the number of the blocks.

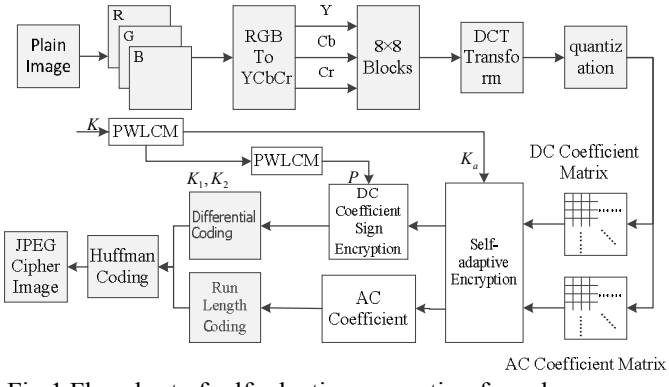


Fig 1 Flowchart of self-adaptive encryption for color JPEG

As shown in Fig. 1, the detailed algorithm is as follows:

(1) Utilize the same method in Ref. [5] to extract all the 2nd bits of $\{x_r\}$, $r=1,\ 2,\ \cdots,\ N$ and form a binary sequence, which is composed of independent and identically distributed (i.i.d.) binary random variables.

Denote the a floating point number x as

$$x = 0.b_1(x)b_2(x)\cdots b_i(x)\cdots, \quad x \in [0,1], b_i(x) \in \{0,1\} \quad (2)$$

The i^{th}-bit $b_i(x)$ can be expressed as

$$b_i(x) = \sum_{q=1}^{2^i-1} (-1)^{q-1} \Theta_{(q/2^i)}(x) \quad (3)$$

where $\Theta_t(x)$ is a threshold function which is defined as

$$\Theta_t(x) = \begin{cases} 0 & x < t \\ & . \\ 1 & x \geq t \end{cases} \quad (4)$$

Set $i = 2$, a binary sequence $B_2^r = \{b_2(x_r)\}_{r=1}^N$ (where r is the length of the sequence and x_r is the r^{th} floating point value) can be obtained.

Extract the first 8 bits as K_a, the 9th-72th bits as K_1, and the 73th-136th bits as K_2. K_1 and K_2 are further used as the initial value and parameter of PWLCM to generate a chaotic sequence P with the length of $m \times n$.

(2) Convert plaintext image to YCbCr color space, and perform the DCT and quantization process. There are two quantization tables, the Luminance Quantization Table (LQT) is for Luminance component , and the Chrominance Quantization Table (CQT) is for Chrominance component. In the process of compression, the DC coefficients and AC coefficients can be adaptively encrypted.

(3) For each block, DC coefficients and the first sixteen AC coefficients in Zig-Zag order of JPEG compression are chosen to construct the DC coefficient matrix M_{DC} sized $m \times n$ and AC coefficient matrix M_{AC} .sized $m \times n \times 16$, respectively.

(4) According to the keystream K_a, M_{DC} and M_{AC} can be adaptively encrypted. If the key bit is "0", based on the sorting result of the upper half of M_{DC}, the lower half of M_{DC} can be shuffled firstly; then the upper half can be shuffled based on the sorting result of the lower half. Similarly, if the key bit is "1", the left half of M_{AC} can be shuffled based on the sorting result of the lower half M_{AC}; then the right half of M_{AC} can be shuffled based on the sorting result of the left half of M_{AC}.

(5) K_1 and K_2 are further used as the initial value and parameter of PWLCM to generate a chaotic sequence P with the length of $m \times n$. The sequence P is used to encrypt the sign of DC coefficients as follows:

$$M_{DC}[i] = \begin{cases} M_{DC}[i] & round(P[i]) = 0 \\ -M_{DC}[i] & round(P[i]) = 1 \end{cases} \quad (5)$$

(6) After the above encryption process, the remaining task is to complete the JPEG coding process.

3. Experimental results

We use USC-SIPI as the test image database, and implement our algorithm in Matlab. Quality factor Q=85 to get a tradeoff between compression performance and the quality of images.

3.1 Encryption results

To demonstrate the encryption result, the color image Baboon is taken as the plain image as shown in Fig. 2(a), and its encrypted image using our proposed algorithm is given in Fig. 2(b). It is easy to see that the encrypted image is noise-like and it doesn't have any intelligible information about its plain image in visual. The similar results are obtained for other test images in USC-SIPI database.

(a) Baboon (b) Encrypted Baboon

Fig 2 Results of self-adaptive encryption for color JPEG.

3.2 Compression performance

In our proposed algorithm, the DC and AC coefficients of each JPEG block is permutated, and the signs of DC coefficients are changed; however, the values of DC and AC coefficients are intact. For this reason, the proposed algorithm has little negative impact on the entropy coding of JPEG. The file size of compressed images with/without encryption under different quality factors are plotted in Fig. 3. It can be found that the compression performance of our algorithm is very close to that of standard JPEG compression.

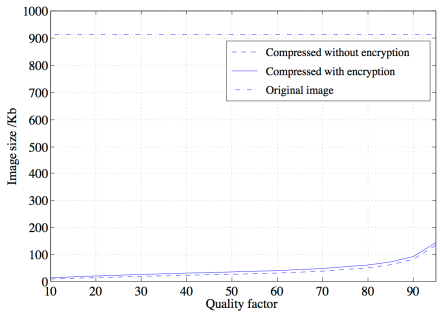


Fig 3. File size of compressed images with different quality factors

3.3 Format compliance

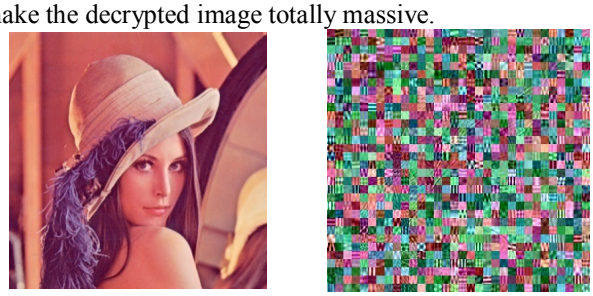
As it has been described in the procedures of our proposed algorithm, the proposed algorithm only permutes the position of DC and AC coefficients, and changes the signs of DC coefficients, it does not interface with the subsequent entropy coding in JPEG. In the reconstruction of an image, if the decoder does not have the correct key, the decoder can still get DCT values, and can therefore recover the image; except that the recovered image is unrecognizable. Only with the correct key, the decoder can recover the plain image exactly. Based on this fact, the proposed algorithm is format compliant.

4. Security analysis

In this section, we analyze the security of our proposed algorithm.

4.1 Key sensitivity

Key sensitivity is a basic requirement for a good cipher. We use chaotic map to generate the keystreams that are involved in the encryption operations, and the initial condition and parameter are used as the secret key. It is well known that chaotic maps are extremely sensitive to the change of initial condition and system parameter. As shown in Fig. 4, we take Lena as the plain image, encrypt it with a secret key, and decrypt the cipher image with a key of one-bit error. The decrypted image is shown in Fig. 4(b), and it is obvious that tiny error in the secret key can make the decrypted image totally massive.



(a) Lena (b) Decrypted Lena with 1-bit error in key

Fig 4 Key sensitivity test.

To further demonstrate the key sensitivity, 3000 secret keys which are slightly different with the correct key are randomly generated. Then 3000 decrypted images are obtained using these keys to decrypt the cipher image. The PSNR values of decrypted images are plotted in Fig. 5, and we can observe that no matter what the tiny error is, any change in the secret key will make the quality of decrypted image extremely low. The sensitivity of the key sensitivity is thus guaranteed.

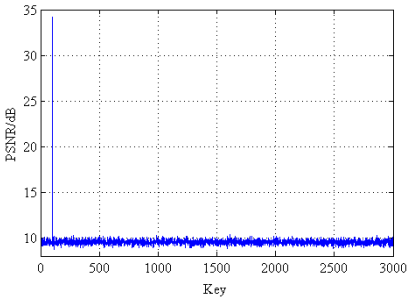


Fig 5. PSNR of decrypted images with 3000 tiny changed K

4.2 Key space

The secret key in our proposed algorithm are the initial condition $x(0) \in [0,\ 1]$ and initial parameter $u_0 \in (0,\ 0.5)$ of PWLCM. The experiments have demonstrated that even the secret key is changed 10^{-16}, the generated chaotic sequences are totally different; while the secret key is changed 10^{-17}, there is no change in the chaotic sequences. Therefore the key precision is 10^{-16}. For the initial condition $x(0) \in [0,\ 1]$ and initial parameters $u_0 \in (0,\ 0.5)$, the key space size will be $1 \times 10^{16} \times 0.5 \times 10^{16} \approx 2^{106}$.

4.3 Resistance against Known-plaintext attack and Chosen-plaintext attack

Known-plaintext and Chosen-plaintext attack are widely used method to attack many image encryption ciphers. The reason why they can work lies on the fact that the generation of shuffling parameters depends solely on the key and has nothing to do with the pending plain image. As long as the key keeps unchanged, the shuffling parameters generated are always the same regardless of different plain images. Consequently, the flaw may arise from the use of the same key for every plain image. However, self-adaptive encryption scheme is inherently immune to both Known-plaintext attack and Chosen-plaintext attack. Its whole encryption process establishes very complicated nonlinear connections with both the secret key and the pending plain image. In other words, the shuffling parameters rely on both of them. Since different plain images correspond to different shuffling parameters, the base of chosen-plaintext/known-plaintext attack has been destroyed.

4.4 Resistance against image processing attack

If image encryption can be regarded as a kind of image degradation to some extent, then parts of original plain image may be restored through some image processing techniques. It is a possible effective attack on image encryption algorithm. Therefore, it is necessary for us to investigate whether common image processing techniques can restore the encrypted image. Since filters are often utilized to carry out such an attack, we choose four popular filters to restore the encrypted image. The restored results are listed in Table 1, where PSNR is Peak Signal to Noise Ratio, and MSSIM is Mean Structural Similarity. As can be seen, none of them can make the encrypted image quality better. Thus, the proposed encryption algorithm can resist image processing attack.

Table 1 PSNR and MSSIM obtained by applying filters on cipher image

Filters	Mean filter	Median filter	Wiener filter	Fuzzy contrast enhancement filter
PSNR(dB)	10.818	9.907	10.975	7.408
MSSIM	0.030	0.021	0.039	0.0103

5. Conclusion

In this paper, we have proposed a format compliant self-adaptive encryption algorithm joint with JPEG coding. In the proposed algorithm, chaotic map is used to generate keystreams to shuffle the positions of DC and AC coefficients; the signs of DC coefficients are also encrypted. Theoretical analysis and experiments both demonstrate that our proposed algorithm is secure and has very little impact on the compression performance at the same time.

Acknowledgments

The work in this paper was supported by National Natural Science Foundation of China (Grant No. 61103211), the open research fund of Chongqing Key Laboratory of Emergency Communications (Grant No. CQKLEC, 20140504) and Project Nos. 106112013CDJZR180005, 106112014CDJZR185501 supported by the Fundamental Research Funds for the Central Universities.

References

[1] S. Lian, “Efficient image or video encryption based on spatiotemporal chaos system,” *Chaos Soliton Fract.* , vol.40, pp.2509–2519, 2009.

[2] C. P. Wu, C.-C. J. Kuo, “Design of integrated multimedia compression and encryption systems,” *IEEE Trans. Multimedia*, vol.7, pp. 828-839, 2005.

[3] C. H. Yuen, K. W. Wong, “Chaos-based encryption for fractal image coding,” *Chin. Phys. B.* , vol.21, pp. *010502*, 2012.

[4] A. Baranousky and D. Daems, “Design of one-dimensional chaotic maps with prescribed statistical properties, ” *Int. J. Bifur. Chaos*, vol.5 (6), pp. 1585-1598, 1995.

[5] T. Kohda, A. Tsuneda, “Statistics of chaotic binary sequences”, *IEEE Trans. Inform. Theory,* vol.43, pp. 104-112, 1997.