Image Compression and Encryption Using DCT and Gaussian Map

IOP Conf. Series: Materials Science and Engineering 462 (2019) 012035 doi:10.1088/1757-899X/462/1/012035

Image Compression and Encryption Using DCT and Gaussian Map

W M Rahmawati[1] and F Liantoni[1]

[1] Department of informatics Institut Teknologi Adhi Tama Surabaya

wenymistarika@gmail.com

Abstract. The use of the image in daily life keeps increasing as information technology is constantly developing. Consequently, the demand for a form of security in sending images through the internet is unavoidable. The encryption algorithm is one of the solutions. Encryption will make any image become only readable by those authorized to access. A chaos-based encryption algorithm is the most commonly used image encryption method because of its sensitivity to the parameter value. In addition to security, image transmission speed is also on demand. Images with bigger size than texts require longer transmission time. For this issue, image compression is the answer. There are many kinds of algorithm that can be used for image size compression. One of which is DCT. DCT algorithm is known to transform spatial domain to frequency. Image encryption and compression are two different processes. Nevertheless, this study combined both processes into one scheme so that they become inseparable. This study has two findings. First, in the area of compression, the scheme produced cipher image with considerable compression ratio. Second, in the area of encryption, the produced cipher image's histogram became completely different from the original histogram, indicating that the encryption algorithm used was immune to histogram analysis.

1. Introduction

Nowadays, most data transmission is done online or via the internet. The internet users begin to look for a way to improve the security and the speed of online data transmission. Many studies on data compression have been conducted [1-5]. All because smaller data size means less bandwidth needed, and the data can be transferred in a little amount of time. Many researchers also developed encryption techniques to ensure that data transferred online will not be accessible if the data ever fell to the wrong hands.

One type of data used by internet users is an image. The image has been widely used in many fields, such as medicine, transportation, military, etc. Moreover, some of those fields require high security when it comes to data transfer because most of their data are confidential. However, security alone is not enough. Data transfer is preferred to be quick as well.

Image data have different characteristics from text data. The image usually has a much bigger data size and high level of redundancy compared to text. Therefore, image compression is performed to reduce image data size, so that image's transfer takes less time. There are many algorithms available for image compression.

Image compression is classified into lossless compression and lossy compression. In lossless compression, the decrypted image has precisely the same components as the original image (before compression). That means there is no information lost from the original image. The examples of lossless compression algorithms are Huffman, LZW, etc. On the other hand, lossy compression produced a decrypted image that is different from the original image. There is information being discarded during the process[6]. However, the data being lost is not much and not recognized by naked eyes. Some examples of lossy compression algorithms are FFT, DCT, & DWT. DCT as a lossy compression algorithm, has the property of strong energy compaction. DCT can transform an image from spatial domain to frequency domain.

Image encryption is performed to ensure that unauthorized individuals can not access the image. An encryption algorithm is good if the key plays a significant role in the encryption and decryption process. This means that if a little mistake occurs when inputting the key, the produced decrypted image will be completely different from the original image. Many algorithms can be used for encryption. Some of those are DES, AES, SHA, chaos-based algorithm. In the field of image encryption, most researchers use a chaos-based method. A chaos-based method is considered as the most suitable method for image encryption because of the fact that image characteristics are different from text characteristics[7], [8]. Gaussian map as one of the chaos-based method types with a property of generating random numbers well. When it is combined with substitution and permutation, the result is an encryption method more resistant to hacking attempts[9].

Compression and encryption are two algorithms meant for different goals. Some research proposed some methods to combine compression and encryption so that both processes cannot be separable[10-13]. The proposed method is considered effective and saving more time. When the wrong hands obtain data, decompression cannot be done without decryption, and decryption cannot be done without decompression as well.

This research proposed a method of combining compression and encryption of image data. DCT algorithm was used for compression because of its strength and its convenience to compress data. A Gaussian map was used for chaos-based encryption because of its high sensitivity to an input parameter to produce random number. The combination of these methods will produce strong scheme of compression and encryption.

2. Literature Review

2.1 Discrete Cosine Transform (DCT)

Discrete Cosine Transform is one of lossy compression algorithm used to transform an image from spatial domain to frequency domain. DCT is mostly used in transforming image and video domain. DCT makes image or video into low and high-frequency form. With the limitations of humans' sight that is difficult to distinguish high frequency, the results of DCT having high frequencies can be suppressed. To suppress DCT results, a quantization process is required after the DCT process [14]. Quantisation is performed on the chrominance and luminance matrix.

Before performing compression, an image will be divided into 8x8 blocks. Therefore, DCT for 2 dimensions will be used with the formula that can be seen in equation (1).

$$F_k(u,v)=\frac{c(u)c(v)}{4}\sum_{i=0}^{7}\sum_{j=0}^{7} f_k(i,j)\cos\left(\frac{(2i+1)u\pi}{16}\right)\cos\left(\frac{(2j+1)v\pi}{16}\right) \quad \text{with} \quad c(e)=\begin{cases}\frac{1}{\sqrt{2}} & e=0\\ 1 & e\neq 0\end{cases} \tag{1}$$

$F_k(u,v)$ is the result of DCT in the (u,v) position, while $f_k(i,j)$ is the initial image pixel value in the (i,j) position before being transformed to DCT.

To restore the compression result using DCT, decompression is done by employing Inverse Discrete Cosine Transform function using equation (2).

$$f_k(i,j)=\frac{1}{4}\sum_{u=0}^{7}\sum_{v=0}^{7} c(u)c(v)F_k(u,v)\cos\left(\frac{(2i+1)u\pi}{16}\right)\cos\left(\frac{(2j+1)v\pi}{16}\right) \text{ with } c(e)=\begin{cases}\frac{1}{\sqrt{2}} & e=0\\ 1 & e\neq 0\end{cases} \tag{2}$$

$f_k(i,j)$ is the IDCT result in the (i,j) position, while the DCT result in the $F_k(u,v)$ position is the result value of DCT in the (u,v) position.

2.2 Huffman algorithm

David A. Huffman firstly developed the Huffman Algorithm in 1952. The Huffman Algorithm is one of lossless compression because there is no lost information from its decompression results. This algorithm is widely used in compression since it is easy and straightforward by utilising the frequency of data occurrence. Frequently appear data symbols will be replaced by the shorter bits compared to appear data symbols rarely. The coding method in the Huffman algorithm is stored in a dictionary in the form of a tree called a Huffman tree. To decode the data resulted from the Huffman algorithm, the same Huffman tree for data compression is needed. That Huffman tree will give the meaning of the original symbol from the code used to decode.

2.3 Chaotic Gaussian Map

A chaotic map is an algorithm that is widely used in the encryption because it is difficult to guess despite its simplicity. A chaotic map will generate pseudorandom numbers which will be used for the encryption process. Pseudorandom results made by chaotic maps depend on the parameters input[9]. There are several chaotic maps can be used including Tent map, Circle map, Gaussian map, etc. To generate pseudorandom from Gaussian map, equation (3) can be used.

$$x_{n+1} = \exp(-\alpha x_n^2) + \beta \tag{3}$$

Where α and β is the input parameter that will significantly affect the Gaussian map results. A Gaussian map will generate some random values and form a sequence. This sequence will be used to randomise other sequences with the same length.

2.4 SHA

SHA is an encryption algorithm created by NIST. SHA has several versions, namely SHA-1, SHA-256, SHA-384, and SHA-512. The output of the SHA algorithm has the same length even though the input length is different. The maximum input and output length generated depends on the used SHA version. In SHA-1, with an input, less than 2^{64} bits long will generate a fixed period of 160 bits[15]. SHA Algorithm is widely used because it has high sensitivity. A little difference in the input will generate a completely different ciphertext result although with the same length. SHA algorithm is a reversible encryption algorithm so that there is no possibility to return the input value generated by SHA. Because of these properties, this algorithm is widely used as a part of a very secure encryption algorithm.

3. The Method Used

In the method section, it was divided into two parts, namely the compression and encryption section as well as the decompression and decryption section. In the first section, the scheme of the process to carry out a combination of compression and encryption will be explained. The input of this algorithm was an uncompressed grayscale image. The steps to do a full image compression and encryption combination are explained as follows:

Step 1: the input image was divided into 8x8 blocks. Each block through the compression process with DCT using equation (1). DCT results were then quantised. The quantisation was carried out by dividing the DCT results with chrominance and luminance matrix. The quantisation results in the form of blocks were made sequences by reading it in an indirect way so that each block generated a sequence with 64 lengths. The first three values of sequence in each block were combined to form sequence A. While the rest was combined to form sequence B.

Step 2: Sequence A was combined with all of the used keys. In this algorithm, the key consisting of $\alpha, \beta, \alpha', \beta', a, b.$. Sequence A that has been combined with the keys became input from SHA-1. In addition, sequence A was also carried out by diffusion process to increase the sensitivity. To do the diffusion process in sequence A, equation (4) was used. While sequence B was encoded using the Huffman algorithm.

$$v_1 = cyc\left[\left(t_1 + v_{i-1} + t_{i+1}\right) \bmod \left(2^{LD} - 1\right), LSB_y\left(v_{i-1} + t_{i+1}\right)\right] \tag{4}$$

Where,

$$y = \begin{cases} 1, & LD \le 3 \\ 3, & 3 < LD \le 7 \\ 7, & 7 < LD \end{cases}$$

Where t_1 is DCT value, v_1 is the result of diffusion, $cyc[x, y]$ turned the sequence x to the right as many as y the bit. $LSB_y(s)$ is the least y value significant bit off s, while LD is a representation of the number of DCT bit.

Step 3: The result of SHA in step 2 was 160 bits used to disrupt the value of Gaussian map input parameter. Initially, 160-bit SHA result was divided into five equal lengths of d_0 to d_4. Furthermore, the Gaussian input parameter was changed by SHA result with equation (5) and (6).

$$\alpha' = \left(\alpha + \left(d_0 + d_1 + U\left(d_2\right)\right) / 2^{32}\right) \bmod 1 \tag{5}$$

$$\beta' = \left(\beta + \left(L\left(d_2\right) \oplus d_3 \oplus d_4\right) / 2^{32}\right) \bmod 1 \tag{6}$$

Where $U(A)$ are the upper 16 bits and $L(A)$ are the lower 16 bits.

Step 4: Doing Gaussian map iteration using α' and β' resulted from step 3 as the input parameter. The result of the Gaussian map was a random sequence used to mask the results of Huffman encoding resulted in step 1. The masking process can be done using the following equation:

$$c_i = \left(r_i + m1_{c_{i-1} + r_{i+1} \bmod L} + c_{i-1} + r_{i+1}\right) \bmod 2^{32} \tag{7}$$

Where $m1$ is the result of the Huffman algorithm and m1 is the sequence of Gaussian map results.

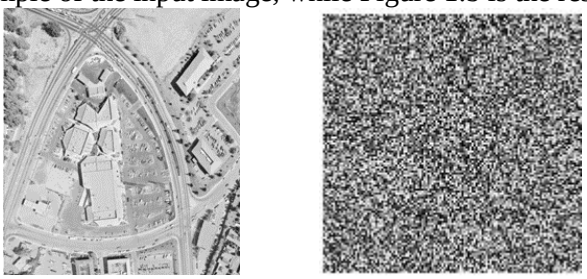
Step 5: The diffusion results of sequence A from step 2 was mutated using cat map. The mutated result encoded using the Huffman algorithm. Next, as in step 4 generated the new Gaussian value to be masked with Huffman Algorithm result using equation (7).

Step 6: Combining the masking results in step 4 and step 5 as a cipher image.

Meanwhile, to carry out decompression and description, the steps were the opposite of step 1 to 6.

4. Result and Analysis

The results of the program done were cipher images in the form of random images in which the original form cannot be recognized, while the input was an uncompressed grayscale image. As can be seen, Figure 1.a is the example of the input image, while Figure 1.b is the result of the chiper image.



(a) (b)

Figure 1. (a) Input image (b) Cipher image

The analysis section was divided into the compression process and part of encryption. In the compression section, the result of compression can be seen from the compression ratio formulated in equation (8).

$$ComprRatio = \frac{SizeofPlain\ \mathrm{Im}age}{SizeofChiper\ \mathrm{Im}age} \quad (8)$$

The higher the compression ratio, the better the compression performance. The result of compression ratio calculation can be seen in table 1. From table 1, it can be seen that the cipher image generated, on average, had four up to five times smaller than the original image.

Table 1. The result of image compression ratio

Image name	Compression ratio	Image name	Compression ratio
Aerial	3.90793083	Bridge	4.1282519
Alarm	5.31948052	Goldhill	4.5893
Baboon	4.16101587	Pepper	4.78784
Barb	4.82876510	Pirate	4.513498
Bird	5.67116649	Zelda	5.178255

While from the encryption section, it was analysed using histogram analysis test. It can be seen in Figure 2 the histogram of the original result was completely different from the histogram of the cipher image and did not have a pattern similar to the histogram of the original image. The histogram of the cipher image that did not have pattern in which the original image was unpredictable.

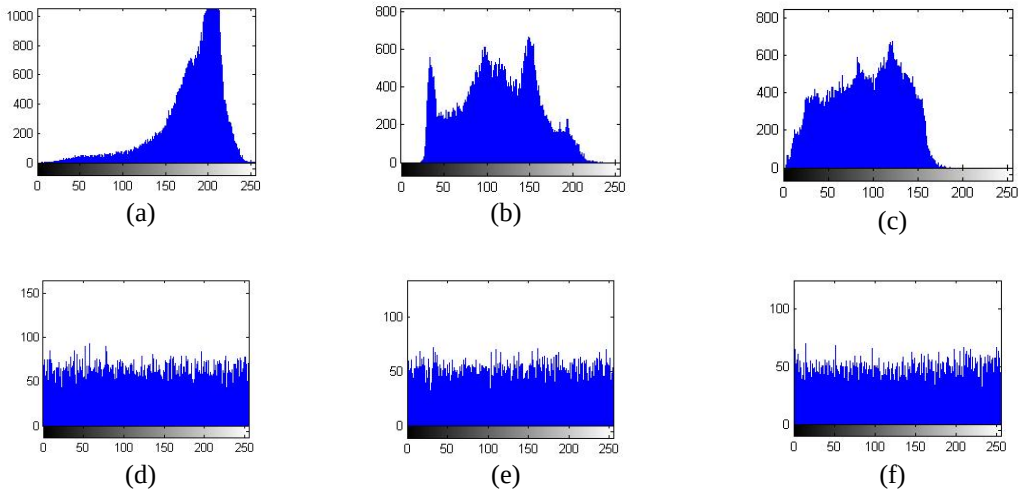


Figure 2 (a) histogram of original Aerial image (b) histogram of original Barb image (c) histogram of original Zelda image (d) histogram of cipher Aerial image (e) histogram of cipher Barb image (f) histogram of cipher Zelda image

5. Conclusion

From the conducted research on image compression and encryption put in on the scheme, the compression carried out using DCT resulted in excellent compression ratio that was 4 to 5 times smaller than the original size. While the chaos-based encryption, Gaussian map was carried out resulted in cipher image that withstood the histogram analysis attack since it resulted in cipher image in which the histogram had a completely different pattern with the original image histogram.

References

[1] A. Chu, "LZAC lossless data compression," in *Proceedings DCC 2002. Data Compression Conference*, 2002, p. 449.
[2] F. Rashid, A. Miri, and I. Woungang, "Secure image deduplication through image compression," *J. Inf. Secur. Appl.*, vol. 27–28, pp. 54–64, Apr. 2016.
[3] H. Wu, X. Sun, J. Yang, W. Zeng, and F. Wu, "Lossless Compression of JPEG Coded Photo Collections," *IEEE Trans. Image Process.*, vol. 25, no. 6, pp. 2684–2696, Jun. 2016.
[4] M. Iwahashi, H. Kobayashi, and H. Kiya, "Lossy compression of sparse histogram image," in *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2012, pp. 1361–1364.
[5] S. K. Roy, S. Kumar, B. Chanda, B. B. Chaudhuri, and S. Banerjee, "Fractal image compression using upper bound on scaling parameter," *Chaos, Solitons & Fractals*, vol. 106, pp. 16–22, Jan. 2018.
[6] I. M. Pu, *Fundamental Data Compression*, 1st ed. Oxford: Butterworth-Heinemann, Elsevier, 2006.
[7] E. Yavuz, R. Yazıcı, M. C. Kasapbaşı, and E. Yamaç, "A chaos-based image encryption algorithm with simple logical functions," *Comput. Electr. Eng.*, vol. 54, pp. 471–483, Aug. 2016.
[8] Y. Zhou, L. Bao, and C. L. P. Chen, "A new 1D chaotic system for image encryption," *Signal Processing*, vol. 97, pp. 172–182, Apr. 2014.
[9] A. Sahay and C. Pradhan, "Gauss iterated map based RGB image encryption approach," in *2017 International Conference on Communication and Signal Processing (ICCSP)*, 2017, pp. 0015–0018.
[10] C.-H. Yuen and K.-W. Wong, "A chaos-based joint image compression and encryption scheme using DCT and SHA-1," *Appl. Soft Comput.*, vol. 11, no. 8, pp. 5092–5098, Dec. 2011.
[11] Y. Zhang, B. Xu, and N. Zhou, "A novel image compression–encryption hybrid algorithm based on the analysis of sparse representation," *Opt. Commun.*, vol. 392, pp. 223–233, Jun. 2017.
[12] M. Brindha and N. Ammasai Gounden, "A chaos-based image encryption and lossless compression algorithm using hash table and Chinese Remainder Theorem," *Appl. Soft Comput.*, vol. 40, pp. 379–390, 2016.
[13] H. Zhu, C. Zhao, and X. Zhang, "A novel image encryption–compression scheme using hyper-chaos and Chinese remainder theorem," *Signal Process. Image Commun.*, vol. 28, no. 6, pp. 670–680, Jul. 2013.
[14] D. Salomon and G. Motta, *Handbook of Data Compression*, Fifth. New York: Springer, 2010.
[15] W. Stallings, *Cryptography and network security : principles and practice*, Fifth. Prentice Hall, 2011.

Acknowledgement

The authors would like to thank Institut Teknologi Adhi Tama that has given the facility to conduct this research. The authors would also like to thank the Directorate General of Higher Education that has provided a chance and financial support so that this research can be well-conducted.