International Journal of Security and Its Applications
Vol.8, No.3 (2014), pp. 209-220
http://dx.doi.org/10.14257/ijsia.2014.8.3.22

Image Scrambling Algorithm based on Grouping Calculation

Lianyuan Jiang[1,2], Haohao Yuan[1], Jianbing Jiang[1], Yalan Zhang[1] and Jian He[1]

[1]*School of Computer, Guangxi University of Science and Technology, Liuzhou 545006, Guangxi, China*
[2]*College of Automation Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, Jiangsu, China*
yuanhao_1027@163.com

Abstract

With the rapid development of the Internet and the broadband, more and more people transit images on the Internet. Given the information security, some important images must be encrypted when transmitted. Image scrambling serves as one of the powerful encrypting tools. This paper proposes an image scrambling algorithm based on grouping calculation. This algorithm divides every byte of the stored image information into three groups, and swaps the position of each group so that the pixels' positions are scrambled. And then empower the three groups with corresponding values and do linear calculation for new byte values so that the pixels' color values are scrambled. The empirical results show that this algorithm has a satisfactory scrambling result.

Keywords: *image scrambling; grouping; weight value; byte value*

1. Introduction

Most information people received in daily life comes from sight, namely, image information. It indicates our trust on what we see. The development of information technology enables the digital media to be transmitted and spread through the Internet quickly and conveniently but also incurs many risks [1]. As the information is easy to be accessed, copied and spread illegally, it needs protections. There are mainly two techniques to protect the multi-media information: one is encryption, the other is to embed watermark. Image scrambling is in the former category. It can also be applied to pretreatment of information hiding and digital watermark [2-4].

Image scrambling is defined as scrambling the image, usually the position or the gray correlation so that people or computer systems cannot perceive the true meaning of the original image [5, 6]. Image scrambling algorithms have three categories: spatial domain scrambling, frequency domain scrambling and scrambling spatial domain and frequency domain simultaneously. Spatial domain image scrambling has two ways: based on the position transformation [2, 5, 7, 8] and based on the gray transformation [9-11]. Tang *et al*., [2] propose an image scrambling algorithm that needs no iterative calculation. This algorithm divides bits of each pixel into even and odd groups. Those in the even group are swapped so that the original higher bits become lower and vice versa. Lei *et al*., [8] propose an improved scrambling algorithm based on knight-tour transformation that enlarges the key database and enhances the scrambling effect. Shao *et al*., [11] construct an avalanche image scrambling transformation that scrambles the original image through inverse transformation and recovers the scrambled image through obverse transformation.

ISSN: 1738-9976 IJSIA

Currently, there is no dearth of researches on image scrambling based on position transformation. But the existing algorithms have many problems, such as safety concern in the transformation cycle, higher demand on the size of the image, dissatisfactory effect of image scrambling, instability of image scrambling, low efficiency of image scrambling, etc. And there are few researches discussing the image scrambling based on gray transformation, which generates a better result than the previous one. This paper combines two transformations and proposes an image scrambling algorithm based on grouping calculation. The algorithm takes two steps: first, divide the byte and swap the position; second, transform the byte value by linear calculation. The empirical results and the distortion analysis show that this algorithm has better result in visual effect and the degree of image scrambling.

2. Introduction on Typical Image Scrambling Algorithms

To better define the image scrambling, we take the image as the matrix, take the rows and columns as the pixels of the height and the width respectively and take the element value as the color value of the pixel. Thus, here is the definition of the image scrambling [12]:

The given image $G=[g(i, j)]_{m\times n}$, the transformation matrix $T=[t(i, j)]_{m\times n}$ is a permutation from 1 to $m\times n$, in which m stands for the number of the row, n stands for the number of the column. G and T are corresponded to each other in accordance with each row and column. Move the color value of the pixel at position $k(1\leq k<m\times n)$ to its corresponding position $k+1$, and that at the position $m\times n$ to its corresponding position 1, then we can get a new image N, which means that G turns to be N after the image scrambling T.

To better understand this definition, we provide a specific example as follows:

25	30	88
39	96	155
87	118	173

(a) The original image G

5	7	4
3	9	1
2	6	8

(b) The transformation matrix T

173	25	39
87	118	30

96	155	88

(c) The scrambled image *N*

Figure 1. A Particular Example of Image Scrambling

2.1. Image Scrambling Algorithm based on Position Transformation

Image scrambling algorithm based on position transformation refers to scrambling the information of the original image by swapping the position of pixels, which makes the image unable to identify. This algorithm is featured by simplicity and efficacy. Typical algorithms include Arnold transformation, Fibonacci transformation, magic square transformation, affine transformation, orthogonal Latin square transformation, Knight-tour transformation, *etc*. We will take Arnold transformation and Fibonacci transformation as examples.

(1) Arnold transformation is proposed by Arnold in his research on ergodic theory. It draws a cat-like face in the square and makes it obscure through transformation. The Arnold transformation is shown as Eq. (1):

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} (\mathrm{mod}\ \ m) \tag{1}$$

$x, y \in \{0, 1, 2, \ldots, m\text{-}1\}$ refers to the abscissa and ordinate of pixels before Arnold transformation. x', y' refer to the abscissa and ordinate of pixels after Arnold transformation. m is the order of the image matrix. Mod refers to modular arithmetic, the purpose of which is to ensure x', y' of sitting in $\{0, 1, 2, \ldots, m\text{-}1\}$.

(2) Fibonacci series is a typical recurrence relation and is widely accepted. Fibonacci series is defined as the following: make $F_0=1$, $F_1=1$, then $F_n=F_{(n-1)}+F_{(n-2)}(n\geq 2)$. We call $\{F_n\}$ the Fibonacci series. For natural number $m\geq 2$, Fibonacci series is shown as Eq. (2) [13]:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} (\mathrm{mod}\ \ m) \tag{2}$$

$x, y \in \{0, 1, 2, \ldots, m\text{-}1\}$ refers to the abscissa and ordinate of pixels before Fibonacci transformation. x', y' refer to the abscissa and ordinate of pixels after Fibonacci transformation. m is the order of the image matrix.

2.2. Image Scrambling Algorithm based on Gray Transformation

This image scrambling algorithm works by gray transformation of the pixels of the image, creating a higher safety. But the efficiency is not satisfying because of the mathematical operations and such algorithm brings some difficulty in relevant position scrambling. So there are few researchers on this category of algorithms. Typical algorithms are Gray code transformation, bit plane transformation, exclusive OR transformation, etc. We will take Gray code transformation as an example to introduce the image scrambling algorithm based on gray transformation.

For any nonnegative integer u, record its binary code as $u=(u_{p-1}u_{p-2}\ldots u_1u_0)_2$, define:

$$g_{p-1}=u_{p-1},\ g_i=u_i \oplus u_{i+1} \tag{3}$$

i=0, 1, 2, …, p-2, "⊕" is the modulo 2 addition. Then we can get an integer in the way of binary code $g(u)=(g_{p-1}g_{p-2}\ldots g_1g_0)_2$, Eq. (3) is called the Gray transformation. $g(u)$ is called the Gray code of u [14]. Gray transformation is shown as follows:

$$\begin{bmatrix} g_{p-1} \\ g_{p-2} \\ \cdots \\ g_2 \\ g_1 \\ g_0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 & 0 \\ 1 & 1 & \cdots & 0 & 0 & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & 1 & 0 & 0 \\ 0 & 0 & \cdots & 1 & 1 & 0 \\ 0 & 0 & \cdots & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} u_{p-1} \\ u_{p-2} \\ \cdots \\ u_2 \\ u_1 \\ u_0 \end{bmatrix} (\text{mod} \quad 2) \tag{4}$$

3. The Proposed Algorithm

Image scrambling algorithms based on position transformation and gray transformation are very common ways of image scrambling. This paper discusses these two methods and proposes an image scrambling algorithm based on grouping calculation. The algorithm first divides the bytes into groups and swaps their positions and then employs the linear calculation to three groups for byte value transformation.

3.1. Grouping and Rows and Columns Swapping

(1)This paper deals with bitmap images. Common bitmap images are 24-bit true color image and 8-bit gray image. Every pixel of the true color image needs 3 bytes and every pixel of the gray image needs 1 byte. To ensure the image scrambling result, this paper divides every byte into 3 groups. We will take $(10011010)_2$ as an example shown in Figure 2. Two bits "10" on the left are the first group, three bits "011" in the middle are the second group and three bits "010" on the right are the third group.

First	Second	Third
10	011	010

Figure 2. A Particular Example of Grouped Byte

Here we need to examine the third group of every byte for later linear calculation of byte value. If the third group of a byte is "100", then we should revise it to "101".

(2) There are many ways to swap rows and columns [15]. This paper swaps (or moves) those groups, that is, for two bytes A and B, swap the first group, or the second group, or the third group of A and B.

3.2. Calculation Method of the Byte Value

For every byte after grouping transformation, the calculation is done in the following ways: suppose the decimal numbers of the three groups of byte A are a, b, c, the byte value is:

$$g=a\times 100+b\times 10+c \tag{5}$$

For example, the binary number of byte A is expressed as $(10101001)_2$, the first group is "10", the second group is "101" and the third group is "001". The three decimal numbers are 2, 5 and 1. Then the byte value is g=2×100+5×10+1=251. Obviously, the range of the

calculated byte value is [0,377], but that of a normal byte should be [0,255]. Thus, two cases can be partitioned:

(1) If the byte value g is smaller or equals to 255, there is no need to calculate a second time. If the third group of the binary number is "100", then revise it to "101".

(2) If the byte value g is bigger than 255, then calculate a second time in the way of:

$$g=g-255 \quad (6)$$

After executing Eq. (6), the byte value sits among [1,122], which meets the requirement. The highest bit of the byte must be 0. To successfully restore the scrambled image, the third group of the binary number corresponding byte value g is forcibly revised to "100". To reduce the distortion as much as possible, we can do some changes before revising it to "100": if the value of the third group is smaller than "100", replace the highest bit of the byte by 0; otherwise replace it by 1.

3.3. Inverse Calculation of the Byte Value

For any byte A of the scrambled image, the inverse calculation of the byte value is described as follows:

(1) When the third group of A is not "100", suppose the byte value of A in the hundred's place, the ten's place and the unit's place are h, t, u, then transform them to binary numbers of two bits, three bits and three bits. Thus we get a binary number of eight bits, which is the byte value of A after inverse calculation. For example, if the byte value of A is 136 with 1 in the hundred's place, 3 in the ten's place and 6 in the unit's place, these three numbers are expressed by "01", "011", "110" respectively as binary numbers. And finally we can get $(01011110)_2$.

(2) When the third group of A is "100", the inverse calculation takes three steps to finish. First, if the highest bit of A is 0, revise the third group of A to "010"; if the highest bit of A is 1, revise the third group of A to "110". Second, add 255 to the byte value of A. Third, execute the calculation in (1) and get the byte value of A after inverse calculation.

3.4. Scrambling Algorithm

Every pixel of the true color image needs three bytes and that of the gray image needs 1 byte. For an image of any size (the true color or gray image), the proposed scrambling algorithm is described as follows:

(1) Read the original image. Group every byte that stores the image according to the method mentioned in section 3.1, and swap (or move) relevant rows and columns.

(2) Calculate the byte value of every byte using the method mentioned in section 3.2 and get the scrambled image.

3.5. Inverse Scrambling Algorithm

Inverse scrambling algorithm is similar to scrambling algorithm. This algorithm can be described as follows:

(1) Calculate the byte value of every byte in the scrambled image using the method mentioned in section 3.3.

(2) For all groups, reverse the process of swapping rows and columns in scrambling

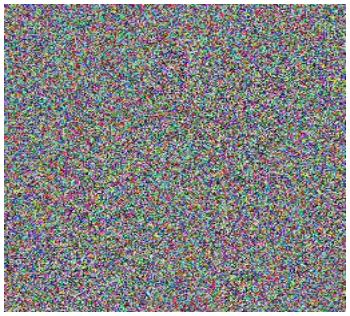
algorithm to restore the image.

4. Empirical Results

We have applied the proposed algorithm to many experiments and the empirical results show that this algorithm generates expected results. Because of the limited space, here we only take four examples including two standard test images, one photographed car image and one painted image.

Experiment 1. Figure 3(a) is a Lena true color image of 256×256 (unit: pixel, the same below). After applying it to our scrambling algorithm, we get Figure 3(b), which is unable to distinguish. The pixels in Figure 3(b) are evenly distributed and the information is destroyed. And later we restore it to Figure 3(c), which is almost the same as the original image.

(a) The original image



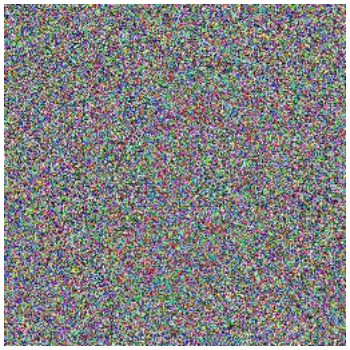
(b) The scrambled image

(c) The restored image

Figure 3. The Experiment on the Lena Image

Experiment 2. Figure 4(a) is a Baboon true color image of 256×256. After applying it to our scrambling algorithm, we get Figure 4(b), which is unable to distinguish. The pixels in Figure 4(b) are evenly distributed and the information is destroyed. And later we restore it to Figure 4(c), which is almost the same as the original image.

(a) The original image



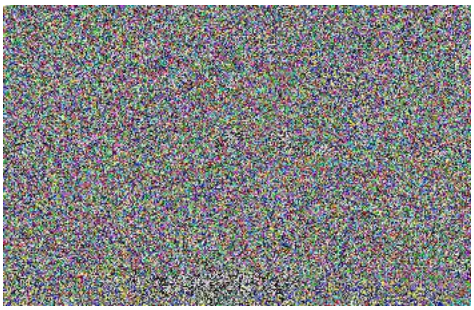
(b) The scrambled image

(c) The restored image

Figure 4. The Experiment on the Baboon Image

Experiment 3. Figure 5(a) is a car image of true color in the size of 351×225. After applying it to our scrambling algorithm, we get Figure 5(b). And later we restore it to Figure 5(c), which is almost the same as the original image. Scrambling and restoring are satisfactory.

(a) The original image



(b) The scrambled image

(c) The restored image

Figure 5. The Experiment on the Car Image

Experiment 4. Figure 6(a) is a painted image of true color in the size of 300×227. After applying it to our scrambling algorithm, we get Figure 6(b). And later we restore it to Figure 6(c), which is almost the same as the original image. Scrambling and restoring are satisfactory.

(a) The original image

(b) The scrambled image

(c) The restored image

Figure 6. The Experiment on the Painted Image

5. Discussion and Conclusion

This paper proposes an image scrambling algorithm based on grouping calculation. This algorithm works well in image scrambling. From empirical results, we have proved that distortion is hardly caught by naked eyes. Here we conclude the following distortion analysis for the restored image:

(1) In Section 3.1, the third group of the byte has been revised from "100" to "101", causing the distortion value to be 1. The distortion probability of every byte is 1/8.

(2) For the first case in Section 3.2, the third group of the byte has been revised from "100" to "101", causing the distortion value to be 1. The distortion probability of every byte is approximately 1/8. For the second case in Section 3.2, if the third group is "010" or "110", then there is no distortion; if the third group is "001", "011", "101" or "111", the distortion

value is 1; if the third group is "000" or "100", the distortion value is 2. Therefore, the average distortion value is 1 for the second case in Section 3.2.

Thus we can get that the average distortion value is approximately 0.5 for every byte applied by the proposed algorithm.

If we want to get a restored image without any distortion, we can add a new space as big as the original image for storing the distortion value of every byte. But the disadvantage is that the scrambled image is twice as big as the original image.

Acknowledgements

This work was supported by the Science Research Foundation of Guangxi University of Science and Technology (No. 1307102) and the Guangxi Natural Science Foundation (No. 2013GXNSFAA019336, No. 2013GXNSFBA019268). The authors would like to thank the anonymous referees for their valuable comments and suggestions.

References

[1] L. Zhao, A. Adhikari, D. Xiao and K. Sakurai, "On the security analysis of an image scrambling encryption of pixel bit and its improved scheme based on self-correlation encryption", Communications in Nonlinear Science and Numerical Simulation, vol. 17, no. 8, (**2012**), pp. 3303-3327.
[2] Z. Tang, X. Lu, W. Wei and S. Wang, "Image scrambling based on bit shuffling of pixels", Journal of Optoelectronics.Laser, vol. 18, no. 12, (**2007**), pp. 1486-1488, 1495.
[3] Z. Zhong, J. Chang, M. Shan and B. Hao, "Double image encryption using double pixel scrambling and random phase encoding", Optics Communications, vol. 285, no. 5, (**2012**), pp. 584-588.
[4] S. Liu and J. T. Sheridan, "Optical encryption by combining image scrambling techniques in fractional Fourier domains", Optics Communications, vol. 287, (**2013**), pp. 73-80.
[5] L. Shao, Z. Qin, B. Liu, H. Gao and J. Qin, "2D bi-scale rectangular mapping and its application in image scrambling", Journal of Computer-Aided Design & Computer Graphics, vol. 21, no. 7, (**2009**), pp. 1025-1034.
[6] G. Ye, "Image scrambling encryption algorithm of pixel bit based on chaos map", Pattern Recognition Letters, vol. 31, no. 5, (**2010**), pp. 347-354.
[7] W. Ding, W. Yan and D. Qi, "Digital image scrambling", Progress in Natural Science, vol. 11, no. 6, (**2001**), pp. 454-460.
[8] Z. Lei, Q. Sun and X. Ning, "Image scrambling algorithms based on knight-tour transform and its applications", Journal of Chinese Computer Systems, vol. 31, no. 5, (**2010**), pp. 984-989.
[9] D. Qi, J. Zou and X. Han, "A new class of scrambling transformation and its application in the image information covering", Science in China (Series E), vol. 43, no. 3, (**2000**), pp. 304-412.
[10] L. Shao, Z. Qin, X. Heng and H. Gao, "Solution for the Inverse Problem of Matrix Transform Based Image Scrambling", Acta Electronica Sinica, vol. 36, no. 7, (**2008**), pp. 1355-1363.
[11] L. Shao, Z. Qin, X. Heng, H. Gao and X. Wang, "Avalanche image scrambling transformation based on high-dimension matrix transformation", Journal of Image and Graphics, vol. 13, no. 8, (**2008**), pp. 1429-1436.
[12] Y. Li, "Research on digital image scrambling algorithm", XiDian University, (**2011**).
[13] D. Yin and B. Li, "Using improved Fibonacci hash transform to increase the robustness of meaningful watermarking algorithm", Journal of Wuhan University of Science and Technology (Natural Science Edition), vol. 28, no. 3, (**2005**), pp. 266-269.
[14] Y. Tan and M. Ma, "Image scrambling algorithm based on bit-plane and Gray code", Computer Engineering and Applications, vol. 46, no. 16, (**2010**), pp. 174-177.
[15] L. Yang, "A row column partnership and recurrence displacement scrambling method for digital image", Journal of Jiamusi University (Natural Science Edition), vol. 27, no. 1, (**2009**), pp. 58-59,66.

Authors

Lianyuan Jiang received the M.S. degree in computer application technology from Guangxi Normal University, China in 2007. Currently, he is a lecturer at Guangxi University of Science and Technology, China. He is working toward the Ph.D. degree in the College of Automation Engineering, Nanjing University of Aeronautics and Astronautics, China. His research interests include pattern recognition and image processing. He has published more than 20 papers in international/ national journals. E-mail address: jly_jly_jly@163.com.

Haohao Yuan received the M.S. degree in precision instruments and machinery from North University of China, China in 2007. Currently, she is a lecturer at Guangxi University of Science and Technology, China. She is working toward the Ph.D. degree in the College of Information Engineering, Wuwhan University of Technology, China. Her research interests include signal processing and embedded system. She is the corresponding author. E-mail address: yuanhao_1027@163.com.

Jianbing Jiang received the M.S. degree in computer application technology from Guangxi University, China in 2008. Currently, he is a lecturer at Guangxi University of Science and Technology, China. He is working toward the Ph.D. degree in the College of Automation Engineering, Nanjing University of Aeronautics and Astronautics, China. His research interests include image processing and pattern recognition.

Yalan Zhang received the M.S. degree in electric theory and new technology from Chongqing University, China in 1999. Currently, she is an associate professor at Guangxi University of Science and Technology, China. Her research interests include image processing, pattern recognition and automatic control.

Jian He received the M.S. degree in educational technology from East China Normal University, China in 2007. Currently, he is an associate professor at Guangxi University of Science and Technology, China. He is working toward the Ph.D. degree in the College of Automation Engineering, Nanjing University of Aeronautics and Astronautics, China. His research interests include pattern recognition and image processing.