JurnalMantik
Volume 3 Number 4, February 2020, pp. 634-641 E-ISSN 2685-4236
https://iocscience.org/ejournal/index.php/mantik/index

Comparative Analysis Of Chaotic Cat MAP And Fibonacci Image Scrambling Methods To Secure Digital Image

Hendra[1],Johanus Kurniawan[2], Insidini Fawwaz[3]

Teknik Informatika,
Universitas Prima Indonesia, Jl. Sekip Sei Kambing Medan 20111, Indonesia

E-mail: hendraahaihendra@gmail.com,johanusliang@gmail.com

ARTICLE INFO

Article history:
Received: 25/01/2020
Revised: 07/ 02/2020
Accepted: 18/02/2020

Keywords:
Fibonacci transformation,
chaotic cat map method,
image scrambling,
digital image security

ABSTRACT

In the process of communication through the internet network, security is an important issue that needs serious attention. This is because the process of sending and communicating data through the internet has the possibility to be intercepted by other parties. This also applies to image data. Therefore, data to be sent via the internet must be secured first. However, the application of several cryptographic methods and several methods of randomization of images requires a relatively long time. To solve the problems encountered, the Image Scrambling Generalized Fibonacci And Chaotic Cat Map algorithm can be applied. The work process of the algorithm will start from the process of selecting the input image and charging the key value that will be used. After that, the process continues with the process of randomizing the image, so that it will produce a randomized image. The resulting image can be reconstructed again by applying an anti-scrambling algorithm. This process requires the same key to be used at the scrambling stage. The resulting application can randomize the original image by filling in the randomization key value. The resulting image can be reconstructed again using the same key. In addition, the application will also produce detailed reports of calculations performed during the scrambling and anti-scrambling processes for each of these methods.

1. Introduction

Along with the rapid development of digital technology, image transmission faces security problems, where the image may be accessible to other parties. The security of image data is very important in many areas, such as privacy and copyright protection, communication security and military applications. Image scrambling is a good tool for making scrambled images visually unidentifiable and difficult to decrypt by unauthorized users. The main purpose of image randomization techniques is to transform an original image into a randomized image to increase the strength to resist statistical attacks, ie attacks that guess the original image based on the shape seen in the scrambled image.

Some methods that can be used to randomize digital images are the Fibonacci method and the Chaotic Cat Map method. The Chaotic Cat Map method was first introduced by a Russian mathematician named Vladimir I. Arnold, in 1960 who demonstrated the algorithm using cat images. This Chaotic Cat Map method uses the idea of modular arithmetic. To apply the cat map method, several encryption parameters are required. In the results of the Arnold's paint map, each pixel point from S is transformed into another pixel point. Meanwhile, Fibonacci numbers are rows of numbers starting at 0 and 1 then followed by numbers obtained by adding the two consecutive numbers before, while the rows of Fibonacci numbers are 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597, 2584, 4181, 6765, 10946, and so on. The concept of Fibonacci numbers can be applied to randomize the pixel position of an image.

To find out the performance of the two methods of image randomization in securing digital images, it is necessary to conduct research on these two methods by taking a thesis entitled "Comparative

Accredited "Rank 4"(Sinta 4), DIKTI, No. 36/E/KPT/2019, December 13th 2019.

JurnalMantik
Volume 3 Number 4, February 2020, pp. 634-641 E-ISSN 2685-4236
https://iocscience.org/ejournal/index.php/mantik/index

Analysis of the Chaotic Cat Map Method and Fibonacci Image Scrambling for Security of Digital Images".

2. Method

2.1. Image Scrambling

Digital image scrambling is often used as an initial or final step in hiding information on an image. The main purpose of digital image scrambling is to transform an original image into an insignificant image (pixel image in a randomized position) to increase the strength to resist invalid attacks so as to increase security.

Fridrich suggested that a chaotic-based image encryption scheme must consist of two processes, namely chaotic confusion and pixel diffusion. The first process will mutate the pixels of an input image with 2D chaotic mapping, while the second process replaces the pixels (gray level) sequentially. This architecture is the basis of a number of chaos-based image ciphers introduced.

The term irregularity or chaos was first used in the world of mathematics in 1975 by Tien Yien Li and James Yorke in a paper entitled "Peirod Three Implies Chaos" (Period Three which Indicates Irregularity). This term is now used to describe a trait of mathematical mapping and certain physical phenomena that at first glance look like something random or irregular, but actually have an element of order as its basis, for example (generating random numbers, shuffling cards, heartbeat disorders, flapping aircraft wings, red dot changes on Planet Jupiter and Planet Pluto's orbital deviations). (Howard Anton and Chris Rorres, 2005)

2.2. Chaotic Cat Map Methode

To explain the Arnold's map of paint, some ideas about modular arithmetic must be understood. If x is a real number, then the notation x mod 1 represents a unique number in the interval [0, 1) where the difference from x is an integer. As an example:

$$2{,}3 \bmod 1 = 0{,}3 \quad 0{,}9 \bmod 1 = 0{,}9 \quad -3{,}7 \bmod 1 = 0{,}3 \quad 2{,}0 \bmod 1 = 0$$

It appears that if x is a nonnegative number, then x mod 1 is only a fraction of x. If (x, y) is a sequence of real numbers, then the notation (x, y) mod 1 represents (x mod 1, y mod 1). As an example,

$$(2{,}3, -7{,}9) \bmod 1 = (0{,}3, 0{,}1)$$

Observe that for each real number x, the point x mod 1 is located in the unit interval [0, 1) and for each successive pair (x, y), the point (x, y) mod 1 is located in a unit square:

$$S = \{(x, y) \mid 0 \le x < 1, 0 \le y < 1\}$$

Also, note that the upper and right border of the square is not included in S. (Anton & Rorres, 2005)

Chaotic paint mapping is a discrete chaotic model introduced by Arnold and Avez. Imagery can be mutated and mapping can be defined as follows.

$$\begin{pmatrix} x_{n+1} \\ y_{n+1} \end{pmatrix} = A \begin{pmatrix} x_n \\ y_n \end{pmatrix} (\mathrm{mod}\, 1), \quad A = \begin{pmatrix} 1 & 1 \\ 1 & 2 \end{pmatrix} \qquad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.1)$$

To apply encryption on the paint map, several encryption parameters are required. Encryption parameters can be obtained by changing the elements of the matrix A. Then the paint map can be developed into N x N, and can be formulated as follows:

$$\begin{pmatrix} x_{n+1} \\ y_{n+1} \end{pmatrix} = A_d \begin{pmatrix} x_n \\ y_n \end{pmatrix} (\mathrm{mod}\, N)$$

$$A_d = \begin{pmatrix} 1 & a \\ b & ab+1 \end{pmatrix}, \quad a, b \in N \qquad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.2)$$

Theorem:

The mapping defined by equation (2.2) is a chaotic mapping and a and b must be integers smaller than N. As shown in Figure 2.4, each pixel in S can be marked by a unique pair of coordinates in the form (m / p, n / p), where p is the pixel size of the image, which identifies the lower left corner, where m and n are integers in the range 0, 1, 2, ..., p - 1. These points are referred to as pixel points, because each of these points identifies a pixel. Figure 1 shows an example of the process of reading the pixel value of an input image.

Accredited "Rank 4"(Sinta 4), DIKTI, No. 36/E/KPT/2019, December 13th 2019.

JurnalMantik
Volume 3 Number 4, February 2020, pp. 634-641 E-ISSN 2685-4236
https://iocscience.org/ejournal/index.php/mantik/index

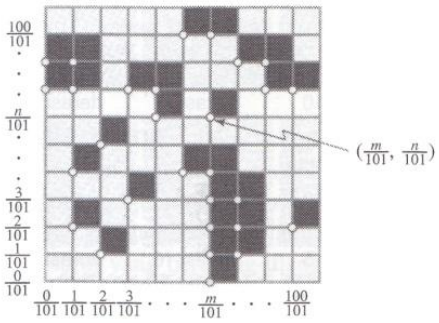

Fig 1. Illustration of input image
Source: Howard Anton dan Chris Rorres, 2005

In the results of the Arnold's paint map, each pixel point from S is transformed into another pixel point. To see how this can occur, the pixel point images (m / p, n / p) produced by are expressed in matrix form by:

$$\Gamma = \left(\begin{bmatrix} \frac{m}{p} \\ \frac{n}{p} \end{bmatrix}\right) = \begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} \frac{m}{p} \\ \frac{n}{p} \end{bmatrix} \bmod 1 = \begin{bmatrix} \frac{m+n}{p} \\ \frac{m+2n}{p} \end{bmatrix} \bmod 1$$

The sequential pair ((m + n) / p, (m + 2n) / p) mod 1 is in the form (m '/ p, n' / p), where m 'and n' lie in the range 0, 1, 2 , ..., p - 1. Specifically, m 'and n' are the remainder of the division when m + n and m + 2n are divided by p. Consequently, each point on S in the form (m / p, n / p) will be mapped to other points in the same form.

Example:

For example p = 76, the above equation becomes:

$$\Gamma\left(\begin{bmatrix} \frac{m}{76} \\ \frac{n}{76} \end{bmatrix}\right) = \begin{bmatrix} \frac{m+n}{76} \\ \frac{m+2n}{76} \end{bmatrix} \bmod 1$$

In this case, iteration at point ($^{27}/_{76}$, $^{58}/_{76}$) sequentially are:

$$\underset{0}{\begin{bmatrix} \frac{27}{76} \\ \frac{58}{76} \end{bmatrix}} \rightarrow \underset{1}{\begin{bmatrix} \frac{9}{76} \\ \frac{67}{76} \end{bmatrix}} \rightarrow \underset{2}{\begin{bmatrix} \frac{0}{76} \\ \frac{67}{76} \end{bmatrix}} \rightarrow \underset{3}{\begin{bmatrix} \frac{67}{76} \\ \frac{58}{76} \end{bmatrix}} \rightarrow \underset{4}{\begin{bmatrix} \frac{49}{76} \\ \frac{31}{76} \end{bmatrix}} \rightarrow \underset{5}{\begin{bmatrix} \frac{4}{76} \\ \frac{35}{76} \end{bmatrix}} \rightarrow \underset{6}{\begin{bmatrix} \frac{39}{76} \\ \frac{74}{76} \end{bmatrix}} \rightarrow \underset{7}{\begin{bmatrix} \frac{37}{76} \\ \frac{35}{76} \end{bmatrix}} \rightarrow \underset{8}{\begin{bmatrix} \frac{72}{76} \\ \frac{31}{76} \end{bmatrix}}$$

Because the Arnold's cat map transforms every pixel point on S to other pixel points on S and because there are only p2 different pixel points on S, the result is that any pixel point must return to its original position after most p2 iterations on the Arnold's cat map.

2.3. Metode *Fibonacci Image Scrambling*

Fibonacci is one of the most famous numerical systems now. In the Fibonacci sequence, each number is the sum of the two previous numbers, starting with zero and one. The higher the value in the row, two Fibonacci numbers that are close together in a row if divided by each other it will have an average ratio of 1: 1.618 or 0.618: 1. (Canaan, et al, 2011)

Numbers 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233 and so on are referred to as Fibonacci numbers. Suppose G1 = a, G2 = b and Gn = Gn-1 + Gn-2, n 3, where a and b are non-negative integer numbers. The {Gn} sequence is called the generalized Fibonacci sequence (GFS).

a) When a = 1, b = 1, *generalized Fibonacci sequence* $\{G_n\}$ called as Fibonacci sequence $\{F_n\}$.
b) When a = 1, b = 3, then *generalized Fibonacci sequence* $\{G_n\}$ called as Lucas sequence $\{L_n\}$.

Suppose there is a series of different consecutive integers {0, 1, 2, ..., B - 1}. For a distinguished generalized Fibonacci sequence (DGFS) $\{G_n\}$, sequence from *integer* $\{S_k\}$,

$$S_k = (kG_n + r) \bmod G_{n+1}, r \in 0, 1, 2, \ldots, B - 1$$

Accredited "Rank 4"(Sinta 4), DIKTI, No. 36/E/KPT/2019, December 13th 2019.

JurnalMantik
Volume 3 Number 4, February 2020, pp. 634-641 E-ISSN 2685-4236
https://iocscience.org/ejournal/index.php/mantik/index

is a permutation of the original sequence {0, 1, 2, ..., B - 1}.

Suppose G_n and G_{n+1} are two different generalized Fibonacci numbers that are adjacent. The following transformation is referred to as the Generalized Fibonacci transformation.

$$S_k = (kG_n + r) \bmod G_{n+1} \quad , k = 0, 1, 2, \ldots, G_{n-1} - 1$$

r can be considered as a key to randomization transformation. For sequential applications of this transformation, different r values can be chosen.

Suppose F_n and F_{n+1} is two consecutive Fibonacci numbers, then the following transformation is called the Fibonacci transformation.

$$S_k = (kF_n + r) \bmod F_{n+1} \quad , k = 0, 1, 2, \ldots, F_{n+1} - 1$$

The Fibonacci sequence can be defined as follows:

$$F(n) = \begin{cases} 0, & \text{jika } n = 0; \\ 1, & \text{jika } n = 1; \\ F(n-1) + F(n-2) & \text{jika tidak.} \end{cases}$$

By applying the above equation, a Fibonacci sequence can be generated consisting of numbers (0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, ...). (Nidhal Khdhair El Abbadi, et al, 2016)

3. Results and Discussion

To scramble an input image, you can click on the 'Scrambling' button, so that the following Phase 1 Scrambling form will be displayed:

a. First Stage Scrambling Form:
   The first step of the image shuffle process is to select the input image, by clicking the '...' button so that the Open dialog box will appear that will be used to find and select the desired file
   Select the file you want to shuffle. For example the file '5k.jpg' is selected, then click the 'Open' button so the system will read and display the image, as shown in the following screen:

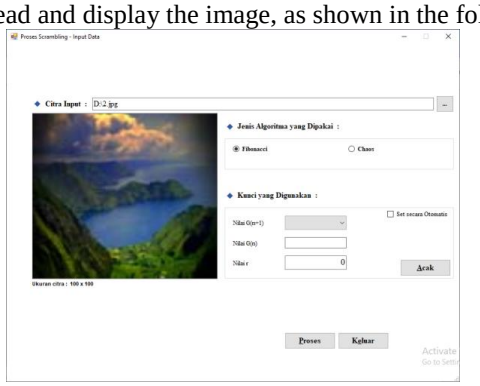

Fig 2. Display Scrambling Form After Image File Selection

After that, determine the type of algorithm to be used. If you want to use the Fibonacci sequence, then click on the 'Fibonacci' radiobutton, whereas if you want to use Chaos, then click on the 'Chaos' radiobutton. The process continues again by determining the key to be used. If the user wants the system to determine the key automatically based on the size of the input image, then click on the 'Set Automatically' checkbox so that the system will automatically determine all key input data. Likewise, the value of r to be used. If the user wants to be determined automatically, then you can click the 'Random' button so the system will take a random value as the value of r.

After all data has been entered, click the 'Process' button so the system will display the Second Stage Scrambling form if all input data is valid. If not, the system will display an error message.

In this Second Stage Scrambling form an image of the randomization results will be displayed as well as the required execution time. In addition, in this form also provided two facilities, namely:

a. The 'Generate Report' button is used to generate a detailed report calculation regarding the scrambling process for the input image that has been performed. Display report form can be seen in the following image:

Accredited "Rank 4"(Sinta 4), DIKTI, No. 36/E/KPT/2019, December 13th 2019.

JurnalMantik
Volume 3 Number 4, February 2020, pp. 634-641 E-ISSN 2685-4236
https://iocscience.org/ejournal/index.php/mantik/index

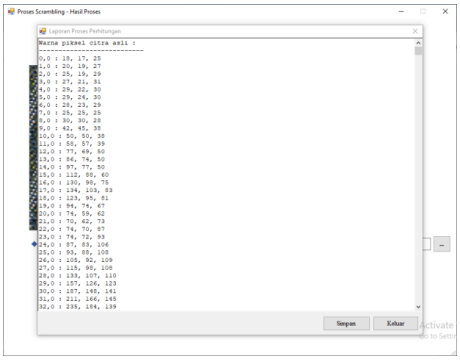

Fig 7. Display Report Form

b. Save button which functions to save the randomized image into a digital image file

After completing the image randomization process, the process can be continued by reconstructing the scrambled image. The process can be done by clicking on the 'Anti Scrambling' button found on the 'Play' form. The first display that appears is the 'Anti-Scrambling-Input Data Process' form as shown in the following image:

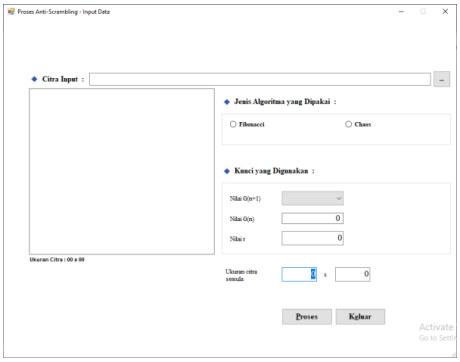

Fig 9. Form Display 'Anti-Scrambling-Input Data Process'

Fill in all the values needed to reconstruct the original image, such as the type of algorithm used and the key value used. Display system output after filling all the required data can be seen in the following figure:

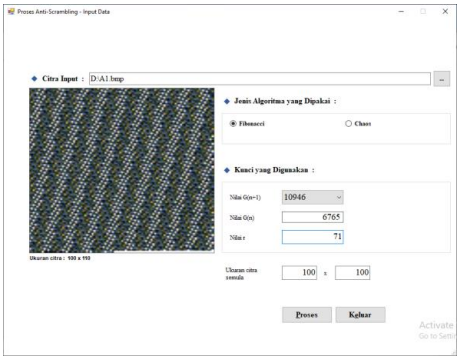

Fig 10. Form Display 'Anti-Scrambling-Input Data Process' After Selecting Image File

Click the 'Process' button to begin the anti-scrambling process. After the process is complete, the system will display the following 'Anti Scrambling Process' form:

Accredited "Rank 4"(Sinta 4), DIKTI, No. 36/E/KPT/2019, December 13th 2019.

JurnalMantik
Volume 3 Number 4, February 2020, pp. 634-641 E-ISSN 2685-4236
https://iocscience.org/ejournal/index.php/mantik/index

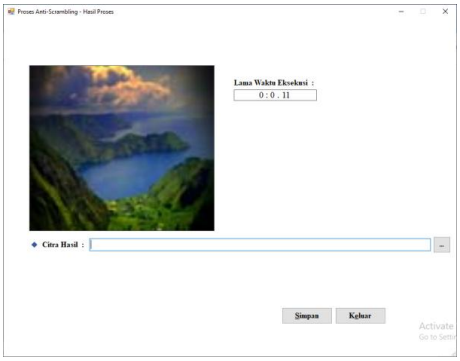

Fig 11. Form Display 'Anti Scrambling Process Results'

Meanwhile, the scrambling and unscrambling processes using the Chaos method can be detailed as follows:

Select the desired image file by clicking on the 'Open Image File' link and entering key input data. Key data can be entered manually via the keyboard or by generating it randomly by clicking the 'Random' button. Display form Scrambling Process can be seen in the following image:

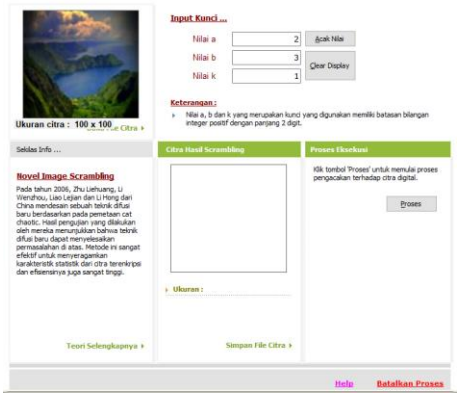

Fig 13. Display Form Scrambling Process After All Input Data Entered

Click the 'Process' button to start the scrambling process. The appearance of the Scrambling Process form will look like in the following image:

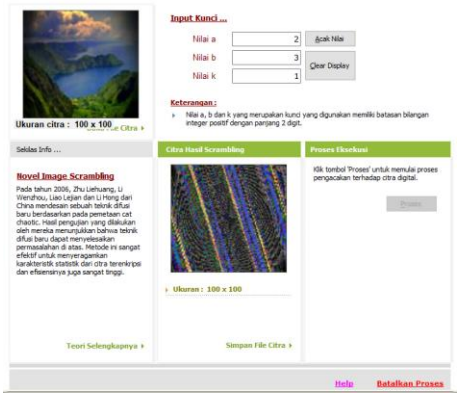

Fig 14. Display Form Scrambling Process After Generating Scrambling Process

This process can be done by clicking the 'Decryption' link or clicking the 'Un-Scrambling' link so the system will display the Dekripsi form. Select the image that will be returned to the original image and fill in the key input that corresponds to the previous scrambling process. After that, click the 'Process' button so that the Un-Scrambling Process form will look like the following image:

Accredited "Rank 4"(Sinta 4), DIKTI, No. 36/E/KPT/2019, December 13th 2019.

JurnalMantik
Volume 3 Number 4, February 2020, pp. 634-641 E-ISSN 2685-4236
https://iocscience.org/ejournal/index.php/mantik/index

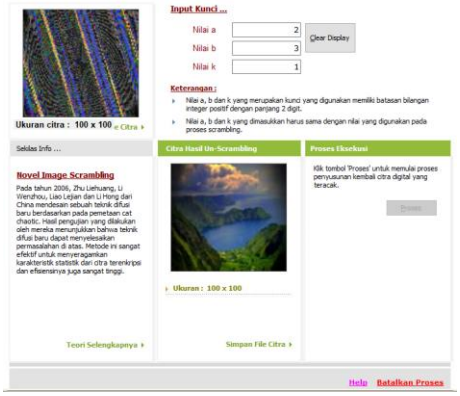

Fig 15. Display the Un-Scrambling Process Form After the Process

The Test Results conducted by the author, i.e.:

Pixel Size	Image Quality	Scrambling Time (m:s.ms)		Un-Scrambling Time (m:s.ms)	
		Fibonacci	Chaos	Fibonacci	Chaos
780 x 390	RGB	0 : 0.936	0 : 1.251	0:0.780	0:0.996
1417 x 945	RGB	0 : 2.979	0 : 3.021	0:3.338	0:4.452
1600 x 1200	RGB	0 : 6 . 146	0 : 7.001	0:5.70	0:6.25
1600 x 1198	RGB	0 : 4.555	0 : 5.21	0:0.226	0:0.331
1024 x 576	RGB	0 : 1.653	0 : 2.205	0:1.809	0:2.254
1024 x 681	RGB	0 : 1.731	0 : 2.308	0:1.934	0:2.658
1024 x 768	RGB	0 : 1.809	0 : 2.259	0:2.43	0:3.645
1600 x 1169	RGB	0 : 4.602	0 : 6.254	0:5.7	0:6.954
900 x 577	RGB	0 : 1.591	0 : 2.267	0:1.825	0:2.257
2048 x 1536	RGB	0 : 7.472	0 : 7.995	0:8.190	0:9.669

Pixel Size	Image Quality	Scrambling Time (m:s.ms)		Un-Scrambling Time (m:s.ms)	
		Fibonacci	Chaos	Fibonacci	Chaos
1920 x 1200	GrayScale	0:8.377	0:9.558	0:7.472	0:8.513
800 x 636	GrayScale	0:1.388	0:2.326	0:1.279	0:2.712
2500 x 1674	GrayScale	0:13.868	0:16.168	0:12.417	0:14.587
900 x 720	GrayScale	0:2.90	0:3.615	0:1.872	0:2.963
1024 x 768	GrayScale	0:2.199	0:3.663	0:2.12	0:3.461
1095 x 729	GrayScale	0:2.246	0:3.698	0:2.12	0:3.697
2322 x 1524	GrayScale	0:13.306	0:16.559	0:11.840	0:14.645
1024 x 664	GrayScale	0:2.121	0:3.697	0:1.887	0:2.631
800 x 450	GrayScale	0:1.279	0:2.465	0:1.201	0:2.006
900 x 675	GrayScale	0:2.12	0:3.331	0:1.840	0:2.649

4. Conclusions

After completing the creation of this software, the author can draw some conclusions as follows:

a. Chaos algorithm and Fibonacci scrambling are able to restore the original image as a whole.
b. Image chaotic results and Fibonacci scrambling don't reduce the statistical characteristics of the original image, so the scrambling image doesn't contain any information about the original image.
c. Small changes in key values will cause changes in the results of chaos and Fibonacci scrambling, so both of these algorithms are resistant to key attacks.
d. The Fibonacci scrambling algorithm has a relatively faster execution time than the scrambling chaos algorithm, but the scrambling image size of the Fibonacci scrambling algorithm is greater than the chaos algorithm.
e. The chaos algorithm can produce scrambling images with the same pixel size as the original image, but the Fibonacci scrambling algorithm is not able to produce scrambling images with the exact same pixel size as the original image.

Accredited "Rank 4"(Sinta 4), DIKTI, No. 36/E/KPT/2019, December 13th 2019.

JurnalMantik
Volume 3 Number 4, February 2020, pp. 634-641 E-ISSN 2685-4236
https://iocscience.org/ejournal/index.php/mantik/index

5. Reference

[1] Anton, H. dan C. Rorres. (2005). **Aljabar Linear Elementer**, Edisi Kedelapan – Jilid 2, Versi Aplikasi, Penerbit Erlangga.
[2] Canaan, C., M. S., Garai dan M. Daya, (2011). ***All About Fibonacci: A python approach***, World Applied Programming Vol (1), No (1), ISSN 2222-2510.
[3] Chandrasekhar, A., D. Chaya Kumari, CH. Pragathi dan Ashok Kumar, (2016). ***Multiple Encryption of Independent Ciphers***, International Journal of Mathematical Archive-7(2), ISSN 2229-5046.
[4] El Abbadi, N. K., S. T. Abaas dan A. A. Alaziz, (2016). ***New Image Encryption Algorithm Based on Diffie-Hellman dan Singular Value Decomposition***, International Journal of Advanced Research in Computer and Communication Engineering, Vol.5, ISSN 2319-5940.
[5] Jiancheng Zou, Rabab K. Ward dan Dongxu Qi (2004). ***The Generalized Fibonacci Tranformations and Application to Image Scrambling***. IEEE.
[6] Liu Xiangdong, Zhang Junxing, Zhang Jinhai, He Xiqin. (2008). ***Image Scrambling Algorithm Based on Chaos Theory and Sorting Transformation***, IJCSNS International Journal of Comptuer Sicence and Network Security, Vol 8 No.1.
[7] Rinaldi Munir, 2004, **Pengantar Pengolahan Citra**, PT. Elex Media Komputindo, Jakarta.
[8] Siregar, S. D., Lestari, Ernala, I., Simarmata, D. P., Naingolan, A. S., (2019). **Pencocokan Foto Berdasarkan wajah dengan Menggunakan Metode *Kohonen***. Journal of Informatic Pelita Nusantara, Vol 4 No. 1.
[9] T. Sutoyo, Edy Mulyanto, Dr.Vincent Suhartono, Oky Dwi Nurhayati, Wijanarto, 2009, ***Teori Pengolahan Citra Digital***, Penerbit Andi Offset, Yogyakarta.
[10] Zhu Liehuang, Li Wenzhou, Liao Lejian dan Li Hong (2006). ***A Novel Image Scrambling Algorithm for Digital Watermarking Based on Chaotic Sequence***. IJCSNS, International Journal of Computer Science and Network Security, Vol 6, No. 8B.

Accredited "Rank 4"(Sinta 4), DIKTI, No. 36/E/KPT/2019, December 13th 2019.