Image Encryption Based on New One-Dimensional Chaotic Map

N.F.Elabady[#1] , H.M.Abdalkader[*2] , M. I. Moussa[#3] ,S. F. Sabbeh[#4]

[#]*Computer Science Department, Faculty of Computer and Informatics, Benha University, Benha, Egypt*
[1]noga23_121986@yahoo.com
[3]mahmoud.mossa@fci.bu.edu.eg

[#]*Information System Department, Faculty of Computers and Information, Benha University, Benha, Egypt*
[4]sahar.fawzy@fci.bu.edu.eg

[*]*Information System Department, Faculty of Computers and Information, Menofia University, Shebien, Egypt*
[2]hatem6803@yahoo.com

***Abstract*— This paper proposes a new image encryption technique based on a new chaotic system consists of joining two chaotic maps: the logistic chaotic map and the cubic chaotic map. This chaotic system is used to encrypt the *R, G, B* components of a colored image at the same time and the three components affect each other. So the correlations between *R, G,* and *B* components were reduced and the security of algorithm was increased. Simulation results show that the algorithm satisfy the required performance tests such as high level security and large key space which larger than the key space of related work.**

***Keywords*—one dimensions logistic map, cubic map, new chaotic map and image encryption.**

I. INTRODUCTION

With the fast development of computer and internet technologies, a lot of information is transmitted over the internet as images. So the security of image has become more and more important. Encryption is one of the ways to ensure security. The aim of image encryption is to convert the original image to another image that is hard to understand. However, it differs from text encryption due to some intrinsic features of images which include bulk data capacities, high redundancy, strong correlations among pixels, etc. These features make conventional cipher systems such as DES, AES and RSA unsuitable for practical image encryption [1]. The chaos theory has been used in cryptography due to its intrinsic features. These properties of chaos includes: sensitivity to initial condition and control parameters, random like behavior and mixing property etc. The chaotic encryption technique was developed in [2]. Since that, a number of researches on image encryption were based on chaotic systems. Because color image consist of three component *R, G, B.* Most of recent papers encrypted the components R, *G, B* of color images with the same method which leads to neglect the correlations between *R, G,* and *B* components and are more vulnerable to be attacked.

The main objective of the paper is to overcome the correlation between R, G and B, new chaotic map which combine the logistic map and cubic map are used as chaotic system. Chaotic system is used to encrypt *R, G,* and *B* components of color image at the same time and made the three components affect each other. New equation is used to scramble the blocks of image, shuffle the position in rows and columns and finally used to change the values of pixels to make the system more complicated and more security.

The rest of paper is organized as follows: section II covers the related work. Section III describes the proposed system. Experimental results are shown on section IV. Finally section V presents the conclusion.

II. RELATED WORKS

The chaos based image encryption consists of two stages [3]. The confusion stage is the pixel permutation where the position of pixels scrambled without changing the values of pixels. The second stage is the diffusion which aims to changing the value of each pixel in the whole image. In [4], a chaotic shuffling algorithm based on sort transformation was proposed which obtained the address codes of images transposition by sorting transformation of chaotic sequence. In [5], it was proposed that an image encryption scheme based on combining two chaotic systems Lorenz and Rosslere to overcome the drawbacks of small key space, weak security and complexity. In [6, 7], the algorithms substituted the gray values of pixels to encrypt plain image without shuffling position, which could be easily attacked by correlation of pixels. In [8], it was proposed that an image encryption scheme based on two chaotic maps, logistic map to change the values of pixels and Arnold cat map to rearranges the position of pixels. In [9], a novel image encryption method based on skew tent chaotic map and permutation–diffusion architecture is proposed. In this method, the P-box is chosen as the same size of plain-image, which shuffles the positions of pixels totally. Because the color images provide more information than grey-level images, they have attracted more and more attentions. In color images, each pixel's value consists of *R, G,* and *B* color components; each color component directly determines the

intensity of red, green or blue color. There are more algorithms to encrypt color image but they used the same method to encrypt its *R, G, B* components. But these methods neglect the correlations between *R, G,* and *B* components and are more vulnerable to be attacked [10]. To overcome this problem, [11] proposed a novel color image encryption algorithm based on chaos. They used chaotic system for the encryption *R, G, B* components of color image at the same time and made the three components affect each other .But they used one dimensional chaotic map as a chaotic system.

III. PROPOSED SYSTEM

The proposed chaotic system consisted of the new equation (1) that was derived from logistic map and cubic map. By combining the logistic map and cubic map a new chaotic map was derived which was used as iterative equation. The new equation was written as follows:

$$x_{n+1=}\mu\, x_n(1-\ x_n)(2+\ x_n) \qquad (1)$$

When 1.41 < μ<1.59, the system comes into chaotic state and can generate a chaotic sequence in the region (0, 1].

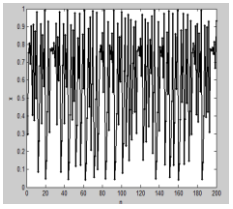
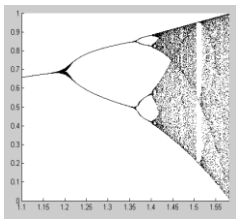
(a)  (b) 

Fig.1 (a) Plot of x component of new map
(b) Bifurcation diagram of new map

Suppose that the color plain image *P* has size *M*×*N*. Convert *P* into its three *R, G,* and *B* each of size *M* × *N* where *M* is the number of row and *N* is the number of column. The proposed algorithm was divided into four phases as follows:

A. Block shuffling

Each matrices R, G, and B of the image P was divided into # (V, U) of blocks. The position of these blocks of pixels was shuffled. This process induced the three matrices R_1, G_1 , B_1 which were combined in a new matrix p_1 = [R_1, G_1 , B_1]. Its data were converted into a one dimensional vectorV_{P1}= $\{p_0, p_1 ,\dots , p_{3*U*V-1}\}$. Then equation (1) was used to shuffle the blocks ofV_{P1}. Given the initial parameters μ, and x_0

1) Iterate equation (1) *(m+3*V*U)* times to generate a sequence of real numbers, $S_{mVU} = \{x_1, x_2, \dots, x_m, x_{m+1}, x_{m+2}, \dots, x_{m+3*V*U}\}$

2) Ignore the first *m* outcomes of S_{mVU}, to obtain the sequence of values $S_{VU} = \{ x_{m+1}, x_{m+2}, \dots, x_{m+3*V*U}\}$

3) Sort the sequence S_{VU} and get $S'_{VU} = \{ x'_{m+1}, x'_{m+2}, \dots, x'_{m+3*V*U}\}$.

4) Determine the position of the values S'_{VU} *in* $S_{VU,}$ and mark the transform map $t_{VU} = \{t_1, t_2, \dots, t_{3*V*U}\}$ where the value x'_{m+i} is the same value x_{t_i} .

5) Rearrange the one dimensional vector $V_{P1} = \{ p_0, p_1 \dots p_{3*U*V-1}\}$ according to t_{VU}.

6) Decompose the one dimensional vector V_{P1} into three $U \times V$ matrices R_2, G_2 and B_2 then reconstruct the blocks in matrices R_2, G_2 and B_2 to create R_3, G_3 and B_3 matrices.

B. Row Scrambling

In this phase the position of pixels was scrambled on rows using equation (1). Equation (1) came into chaotic state for a given initial values μ_1 , x_{00} .

1) Iterate equation (1) (*m+3*M*) that induce the sequence $S_{mM} = \{x_1, x_2, \dots, x_m, x_{m+1}, \dots, x_{m+3*M}\}$.

2) Ignore the first m outcomes, to obtain $S_M = \{x_{m+1} x_{m+1}, x_{m+2}, \dots, x_{m+3*M}\}$.

3) Sort this sequence S_M and get the sequence $S'_M = \{x'_{m+1}\ x'_{m+1}, x'_{m+2}, \dots, x'_{m+3*M}\}$.

4) Determine the position of values of S'_M in S_M and mark the transform maps $t_M = \{t_1, t_2, \dots, t_{3*M}\}$, where the value x'_{m+i} is the same value x_{t_i}.

5) Combine the R_3, G_3, B_3 matrices vertically and get matrix P_2 with 3*M* rows and *N* columns.

6) Rearrange the rows of P_2 according to t_M, which move the t_1 row to the first row, t_2 row to the second row, etc. until the rows have been moved. We get transformed matrix P'_2.

C. Columns scrambling

In this phase the position of pixels was scrambled on columns using equation (1). Equation (1) came into chaotic state for a given initial values μ_2 *and* y_0 .

1) Iterate equation (1) (*n+3*M*N*) to create sequence $S_{nMN} = \{y_1, y_2, \dots, y_n, y_{n+1}, y_{n+2}, \dots, y_{n+3*M*N}\}$ of real number.

2) Ignore the first *n* outcomes to obtain $S_{MN} = \{y_{n+1}, y_{n+2}, \dots, y_{n+3*M*N}\}$

3) Divide S_{MN} into *M* chaotic sequence S_N (i), where (i=1,2,…,*M).*

4) sort each S_N *(i)* and get the transform position $T_{iN}=\{p_{i1}, p_{i2}, p_{i3},..., p_{i(3N)}\}$, *(i=1,2,3,…,M).*

5) Partition the transformed matrix P'_2 into three matrices R_4, G_4 and B_4 from top to bottom; the size of each matrix *M*×*N.*

6) Combine R_4, G_4 and B_4 horizontally and get matrix P_3 with *M* rows and *3N* columns. Then rearrange the columns of each row of P_3 according toT_{iN}, that is, move the p_{i1} column of the ith row to the first column, p_{i2} row column of the ith row to the second column, etc. until all the columns of each row have been transformed and get the transformed matrixP'_3.

D. pixels diffusion

In this phase the value of pixels were changed after the pixel permuted. Given initial values μ_3 and y_{00} .

1) Use (1) to compute the chaotic sequence $Y_{MN} = \{y_1, y_2, \ldots, y_{3*M*N}\}$ and get sequences

z_{1n}= *mod(*Y_{NM}**10^14,3) ,(n = 1,2,. . .,3MN),* where z_{1n} integer range 0-2.

z_{2n}= *mod (*Y_{NM}*10^14,256), (n = 1, 2,…, 3MN),* where z_{2n} integer range 0-255.

2) Partition the transformed matrix P'_3 into three matrices R_5, G_5 and B_5 from left to right; the size of each matrix is *M* ×*N*.

3) R_5, G_5 *and* B_5 are converted into vectors V_{Ri} , V_{Gi} , V_{Bi}*(i = 1, 2,…, L)* each vector has

A length of *L=M×N.*

4) Take z_{1n}as reference,if z_{1n}*=0,* diffuse the current value of V_{Ri} , and If all the values of V_{Ri} have been changed, let z_{1n}*=1*; if z_{1n}*=1,* diffuse the current value of V_{Gi}, and if all the values of V_{Gi}, have been changed ,let z_{1n}*=2*; if z_{1n}=2, diffuse the current value of V_{Bi} , and if all the values of V_{Bi}have been changed ,let z_{1n}*=0,* until all values of the three vectors have been changed .

5) The diffusion formula is

$$C_{now}=(P_{now}+z_{2n}+C_{pre}+P_{pre}) mod\ 256,$$

where C_{now} is the current ciphered value, P_{now} is the current plain value, $C_{pre}$is the previous ciphered value and P_{pre} is the previous plain value .Set initial values C_0*=0,* P_0*=0.*

6) After the diffusion stage, we get three ciphered vectorsV'_{Ri} , V'_{Gi} , V'_{Bi}*, (i = 1, 2, …, L)* .

7) The vectors V'_{Ri} , V'_{Gi} , V'_{Bi} are converted into three matrices with size *M×N*; these matrices are the *R*, *G*, *B* components of ciphered image. So we get the ciphered color image.

E. Image decryption

The decryption procedure was similar to that of the encryption process but in the reversed order. Given the initial parameters and initial values as same as these in encryption process and used the same methods to obtain t_{VU}, t_M, t_{iN}, Y_{MN}, z_{1n}, *and* z_{2n}. *C* is converted into its *R*, *G*, *B* components, then the components are converted into vectors V'_{Ri} , V'_{Gi} , V'_{Bi} , *(i = 1,2,…,L)* .each vector has a length of *L=M×N.* Take z_{1n} , as reference to diffuse V'_{Ri} , V'_{Gi} , V'_{Bi} reversely. The diffusion formula

$$P_{now}=(C_{now}- z_{2n}- C_{pre}- P_{pre}) mod 256$$

Where $P_0 = 0$,$C_0 = 0$. After the reverse diffusion stag, we get three vectors V_{Ri} , V_{Gi} , V_{Bi}, *(i = 1, 2,…, L).* V_{Ri} , V_{Gi} , V_{Bi} (Are converted to three matrices R_5, G_5 , B_5 with size *M×N.* R_5, G_5 , B_5 horizontally and get matrix P'_3 with *M* rows and *3N* columns. Reverse permute the columns of each row of P'_3 according to t_{iN} and get the reverse permutation matrix P_3. Partition P_3 into three matrices R_4, G_4 , B_4 from left to right; the size of each matrix is *M×N.* Combine R_4, G_4 , B_4vertically and get image $P'_2$with *3M* rows and *N* columns. Reverse permute the rows of P'_2 according to t_{VU} and get the reverse permutation matrix P_2. Partition the transformed matrix P_2 into three matrices R_3, G_3 , B_3 from top to bottom, the size of each matrix is *M×N.* Each matrix, R_3, G_3 , $B_3$was divided into *# (V, U)* of blocks. This process induced the three matrices R_2, G_2 , B_2 which were combined in a new matrix P_1*= [* R_2, G_2 , B_2*].* Its data were converted into a one dimensional vector V_{P1}. Reverse permute the blocks of V_{P1}according t_m. Reconstruct the one dimensional V_{P1} into three *M×N* matrices R_1, G_1 and B_1 then reconstruct the blocks in each matrix, R_1, G_1 and $B_1$to create *R, G,* and *B* matrices. These three matrices are the *R, G, B* components of decrypted color image, so we get the decrypted color image.

IV. EXPERIMENTAL RESULTS

A. Encrypted result

The System is developed by MATLAB. MATLAB R2012a is a version that we have chosen for implementation. This implementation is tested in Windows 7 64-bit operating system with an Intel® core™i5-2430M CPU (2.40GHz) and 4GB of RAM. For the performance evaluation of the proposed method, we used 'lena.tif' and 'mandril.tif' images of size 256x256. The initial parameters and values were (μ =1.57,x_0, =0.12345678912345, μ_1 =1.58,x_{00} =0.2345678 9123456, μ_2 =1.55, y_0= 0.56789123456789, μ_3 =1.56 y_{00} = 0.34567891234567). Fig.2 (a)-(d) shows the original color image and the components *R, G,* and *B* of the original image. Fig.3 (a)-(d) shows the encrypted image and the components *R, G,* and *B* of encrypted image.

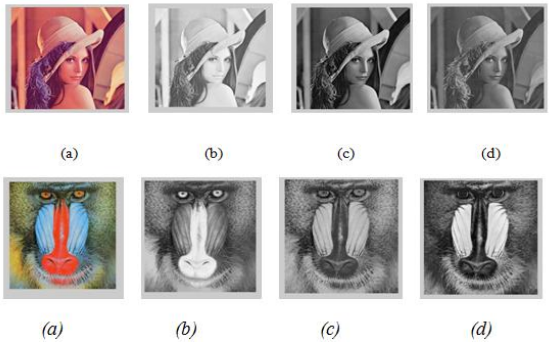
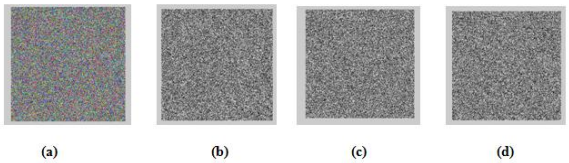
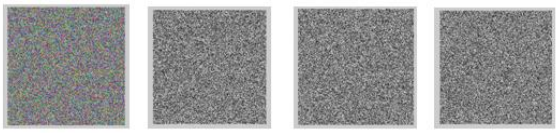


*Fig.*2 Original image *P* and its *R, G, B* components. (a) Original image *P*. (b) *R* component of original image. (c) *G* component of original image. (d) *B* component of original image.





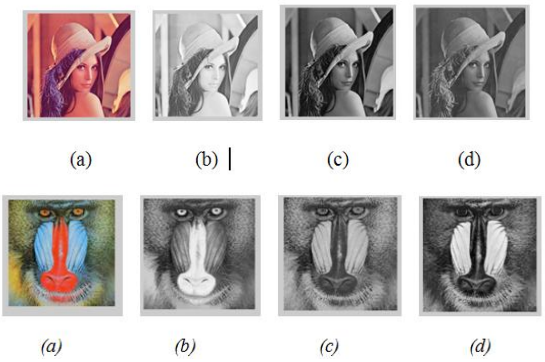
(a) (b) (c) (d)

Fig.3 Encrypted image *P* and its components *R*, *G*, *B*. (a) The original encrypted image *P* (b) the encrypted component *R*. (c) the encrypted component *G*. (d) the encrypted component *B*.

B. Security analysis

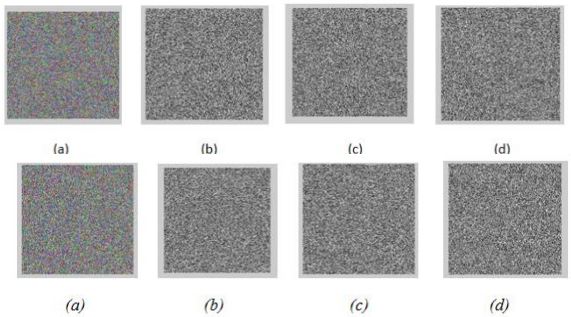
1) Key space analysis: Key space size is the total number different keys which can be used in the encryption. A good encryption algorithm should be sensitive to the secret keys. The key space should be large enough to make brute-force attacks infeasible. In this algorithm, the initial conditions and parameters μ, x_0, μ_1 , x_{00}, μ_2, y_0 μ_3, y_{00} were used as secret key. If the precision is 10^-14, the key space size can reach to 10^112, it is bigger than 2^128. So the key space was large enough to resist the brute-force attacks.

2) Key sensitivity analysis: A good encryption algorithm should be sensitive to the encryption keys in process of both encryption and decryption. When encrypt image, tiny change of keys receive two different cipher images and when decrypt image, if we use wrong key we receive different image. Fig. 4(a)-(d) show the decrypted image of Lena with the correct key λ=1.55. Fig. 5 (a)-(d) show the decrypted image of Lena with the wrong encryption key λ= 1.55000000000001, respectively. So it can be concluded that the algorithm was sensitive to the key, a small change of the key will generate a completely different decryption result and cannot get the correct plain-image.



(a) (b) (c) (d)

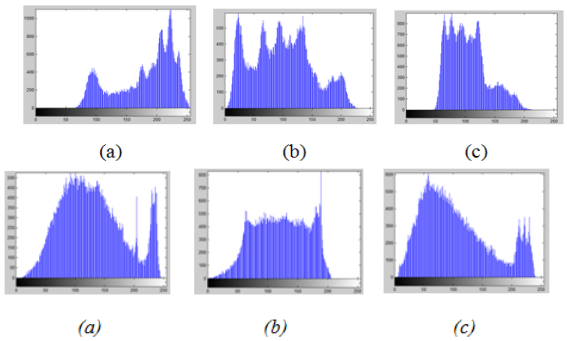
Fig. 4 Decrypted image with correct parameters and its components. (a) The decrypted image (b) the decrypted component *R*. (c) the decrypted component *G*. (d) the decrypted component *B*.



(a) (b) (c) (d)

Fig. 5 Decrypted image with incorrect parameters and its components (a) The decrypted image the decrypted component *R*. (c) the decrypted component *G*. (d) the decrypted component *B*.

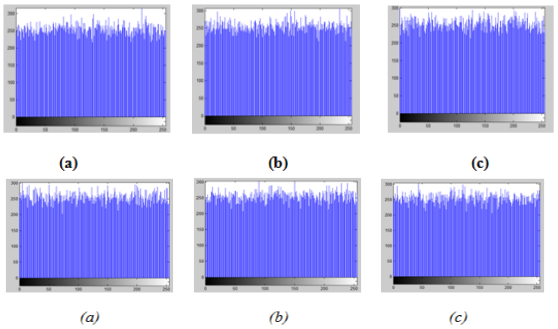
3) Histogram analysis: The image histogram illustrated how pixels in an image are distributed by graphing the number of pixels at each color intensity level. Fig. 6(a)-(c) show the histogram of plain components *R*, *G*, *B*. Fig. 7(a)-(c) show the histogram of cipher components' *R*, *G*, and *B*. The histogram of the cipher image was fairly uniform and significantly different from that of the original image.



(a) (b) (c)

(a) (b) (c)

Fig. 6 The histogram of original image's *R*, *G*, and *B* components. (a) The histogram of *R* component. (b) The histogram of *G* component. (c) The histogram of *B* component.



(a) (b) (c)

(a) (b) (c)

Fig. 7 The histogram of cipher image's R, G, and B components. (a) The histogram of cipher image's R component. (b) The histogram of cipher image's G component. (c) The histogram of cipher image's B component.

4) Correlation analysis: In order to test the correlation among pixels, we randomly select 3000 pairs of adjacent pixels in three dimension (vertical, horizontal and diagonal), and plot the corresponding distributions. Fig. 8(a)-(f) show the results of correlation property analysis. The correction coefficient was calculated for each pair using the following formulas:

$$r_{xy} = \frac{cov(x,y)}{\sqrt{D(x)}\sqrt{D(y)}} \tag{2}$$

$$cov(x,y) = \frac{1}{N}\sum_{i=1}^{N}\big(x_i - E(x)\big)\big(y_i - E(y)\big) \tag{3}$$

$$E(x) = \frac{1}{N}\sum_{i=1}^{N}(x_i) \tag{4}$$

$$D(x) = \sum_{i=1}^{N}(x_i - E(x))^2 \tag{5}$$

Where *x, y* are gray value of two adjacent pixels. Results for vertical and diagonal directions were obtained, which are shown in table 1. The correlation of two adjacent pixels in plain image closed to 1, while in ciphered image it closed to 0 which demonstrate that the image encryption could effectively resist statistical attack.

TABLE I

CORRELATIONCOEFFICIENT OF VERTICAL ADJACENT PIXELS

Image	Component	Plain image	Cipher image	
			proposed algorithm	Ref [11]
Lena	R	0.9781	-0.00092	-0.-00002
	G	0.9695	-0.0030	-0.0038
	B	0.9495	-0.0026	-0.0020
Mandril	R	0.9211	0.0014	0.0062
	G	0.8498	-0.0028	-0.0060
	B	0.9120	0.0055	0.0077

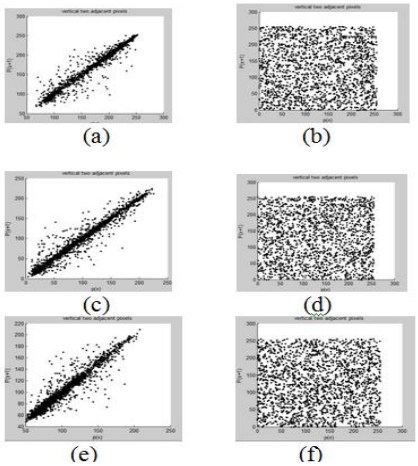


Fig. 8 The correlation test of *R, G, B* components of the original image and the decrypted respectively

5) NPCR and UACI analysis: NPCR stands for the number of pixels change rate while one pixel of plain image changed. UACI stands for the average intensity of differences between the plain image and ciphered image. The NPCR and UACI measure tested the different range between two images. Calculate $\mathrm{NPCR_{R,G,B}}, \mathrm{UACI_{R,G,B}}$ using the following formulas:

$$NPCR_{R,G,B} = \frac{\sum_{ij} D_{R,G,B}(i,j)}{W \times H} \times 100\% \quad (6)$$

$$UACI_{R,G,B} = \frac{1}{W \times H}\left[\frac{\sum_{ij} |C_{R,G,B}(i,j) - C'_{R,G,B}(i,j)|}{255}\right] \times 100\% \quad (7)$$

Where W, H are the width and height of the image, $C_{R,G,B}(i,j)$ and $C'_{R,G,B}(i,j)$ are the two encrypted images before and after one pixel of the plain image is changed respectively. $D_{R,G,B}(i,j)$ is determined by the following rules: when $C_{R,G,B}(i,j) = C'_{R,G,B}(i,j)$,then $D_{R,G,B}(i,j) = 0$; otherwise it is 1. The results are shown in table 2. We can find that the $NPCR_{R,G,B}$ is over 99% and the $UACI_{R,G,B}$is over 33%; the results show that the algorithm was very sensitive to tiny changes in the plain image, even if there is only one bit difference between the two plain images, the decrypted images will be different completely. Thus, the algorithm is robust against differential attack.

TABLE II

NPCR AND UACI RESULTS

Image		Proposed algorithm			Ref [11]		
		R	G	B	R	G	B
Lena	NPCR %	99.64	99.580	99.655	99.623	99.554	99.632
	UACI %	33.383	33.508	33.568	33.416	33.423	33.471
Mandril	NPCR %	99.589	99.573	99.637	99.626	99.609	99.649
	UACI %	33.640	33.340	33.470	33.376	33.515	33.491

6) PSNR analysis: Peak Signal-to-Noise Ratio (PSNR). PSNR reflects the encryption quality. Mean square error (MSE) is the cumulative squared error between original and decrypted image. Lower value of MSE means lesser error. Formula for MSE is:

$$\mathrm{MSE_{R,G,B}} = \sum_i \sum_j \frac{\mathrm{P_{R,G,B}(i,j) - C_{R,G,B}(i,j)}}{\mathrm{MxN}} \quad (8)$$

$$\mathrm{PSNR_{R,G,B}} = 20 * \log 10\left(\frac{255}{\mathrm{sqrt(MSE_{R,G,B})}}\right) \quad (9)$$

where $\mathrm{P_{R,G,B}}$is the original image, $\mathrm{C_{R,G,B}}$is encrypted image.The lower value of PSNR represents better encryption quality. The results are showed in table 3.

TABLE III

PSNR RESULTS

Image	proposed algorithm			Ref [11]		
	R	G	B	R	G	B
Lena	7.8992	8.5765	9.6785	7.8809	8.5394	9.6342
Mandril	8.9581	9.4143	8.4156	8.9763	9.4190	8.4378

7) Randomness test: To ensure the security of a cryptosystem the cipher must have some properties such as good distribution, long period, high complexity and efficiency. In particular, the outputs of a cryptosystem must be unpredictable in the absence of knowledge of the inputs. Recently, the NIST designed a set of different statistical tests to test randomness of binary sequences produced by either hardware or software based cryptographic random or pseudorandom number generators. These tests focus on a variety of different types of non-randomness that could exist in a sequence. The mathematical description of each test can be found at [12]. So, we used the NIST test suite in order to test the randomness of the proposed algorithm. In all tests if

the computed P-value is < 0.01, then conclude that the sequence is non-random. Otherwise, conclude that the sequence is random. Table 4 depicts the NIST tests results and illustrates that the image sequences encrypted by the proposed system pass all the statistical tests with high P-values.

TABLE IV.

NIST RANDOMNESS RESULT

Image	Test Name	P-value
Lena	Approximate Entropy	0.655694
	Frequency	0.788766
	Block Frequency	0.756104
	Cumulative sums	0.701203
	FFT	0.186835
	Linear complexity	0.908002
	Run	0.937761
	Longest Run	0.267841
	Overlapping template	0.769208
	Non overlapping template	Success
	Random Excursion	Success
	Random Excursion variant	Success
	Rank	0.159511
	Serial P-vaue1	0.653835
	Serial P-value2	0.478615
	Universal	0.423873
Image	Test Name	P-value
Mandril	Approximate Entropy	0.916894
	Frequency	0.272568
	Block Frequency	0.212805
	Cumulative sums	0.491608
	FFT	0.494345
	Linear complexity	0.782149
	Run	0.598042
	Longest Run	0.302303
	Overlapping template	0.643756
	Non overlapping template	Success
	Random Excursion	Success
	Random Excursion variant	Success
	Rank	0.943630
	Serial P-vaue1	0.599654
	Serial P-value2	0.451544
	Universal	0.709256

V. CONCLUSSION

An image encryption based on new one-dimensional chaotic map is proposed in this paper. We use new one-dimensional chaotic map to encrypt R, G, and B components of color image at the same time and make these three components affect each other. New equation is used to scramble the blocks of image, shuffle the position in rows and columns and finally used to change the values of pixels to make the system more complicated and more security. The System is developed by MATLAB R2012a. This implementation was tested in Windows 7 64-bit operating system with an Intel® core™i5-2430M CPU (2.40GHz) and 4GB of RAM. Simulation results showed that The correlation Coefficient between two adjacent pixels in encrypted image is far less than that of plain image, it indicates that this image encryption have high performance of resisting statistical attack. The key space is about 10^112, it is bigger than 2^128.which is large enough to resist brute-force attack. It is high sensitive to security key so that encrypted image couldn't be decrypted correctly even only one digit of the security key is changed.

REFERENCES

[1] S. Li, G. Chen, A. Cheung, B. Bhargava, K.-T. Lo, On the Design of Perceptual MPEG video Encryption Algorithms, CoRR abs/cs/0501014, 2005.Encryption Algorithms, CoRR abs/cs/0501014, 2005.

[2] C. Fu, Z. Zhang and Y. Cao, "An improved image encryption algorithm based on chaotic maps," Third In-ternational Conference on Natural Computation, Vol. 3, Washington, 2007, pp. 24-27.

[3] C. Fu, Z. Zhang and Y. Cao, "An improved image encryption algorithm based on chaotic maps," Third In-ternational Conference on Natural Computation, Vol. 3, Washington, 2007, pp. 24-27.

[4] LIU Xiangdong, Zhang Junxing, Zhang Jinhai, He Xiqin,"Image scrambling algorithm based on chaos theory and Sorting transformation" ,IJCSNS International Journal of computer Science and Network Security, VOL.8 No.1, January 2008.

[5] Qais H. Alsafasfeh, Aouda A. Arfoa," Image encryption based on the general approach for multiple Chaotic systems", Journal of Signal and Information Processing, 2011, 2, 238-244.

[6] H. Zhang, X.F. Wang, Z.H. Li, and D.H. Liu, "A fast Image encryption algorithm based on chaos system and Henon Map", Chinese Journal of Computer Research and Development,2005,pp.2137-2142.

[7] X.J. Li, J.H. Peng, and N. Xu, et al. "Image encryption algorithm based on 2D Hyper chaotic sequences", Journal of Image and Graphics, 2003, pp. 1172-1177.

[8] 1Sudhir Keshari, 2Dr. S. G. Modani," Image encryption algorithm based on chaotic map Lattice and Arnold cat map for secure transmission", IJCST Vol. 2, Issue 1, March 2011.

[9] Guoji Zhang a, Qing Liu b,," A novel image encryption method based on total shuffling scheme", Optics Communications 284 (2011) 2775–2780.[10] M. Sahar, M.E. Amir, "Colour image encryption based on coupled nonlinear chaotic map", Chaos, Solitons & Fractals 42 (3) (2009) 1745–1754.

[10] Xingyuan Wangn, LinTeng,XueQin," A novel colour image encryption algorithm based on chaos", Signal Processing 92 (2012) 1101–1108.

[11] Rukhin, A. et al. (2010b). A Statistical Test Suite for Random and Pseudorandom Number Generators forCryptographic Applications. NIST Special Publication 800-22, Revision 1a.