International Journal of Applied Information Systems (IJAIS) – ISSN : 2249-0868
Foundation of Computer Science FCS, New York, USA
Volume 10 – No.1, November 2015 – www.ijais.org

Implementation of Gray Image Encryption using Multi-Level of Permutation and Substitution

Dina Riadh Alshibani
Assistant Lecturer
Computer Science Department
University of Al Mustansiriyah / Iraq - Baghdad

Rasha Shaker Ibrahim
Assistant Chief Programmer
Computer Science Department
University of Al Mustansiriyah / Iraq - Baghdad

ABSTRACT
In this paper a new Gray image encryption system is presented. It is based on permutation and substitution of image pixels using secret keys in time domain. The system is with multilevel to increase the security and to present an encrypted image with low pixel correlation, high entropy and uniform distributed histogram. The Tinkerbell map, Zaslavsky Map and Arnold Transform are employed in keys generation to be used in the permutation and substitution process. Test results are done with definite investigation to show that the proposed image encryption is very secure because of its vast key space and robust permutation-diffusion mechanism.

General Terms
Security, Image Processing.

Keywords
Choatic maps, Arnold Transform, Tinkerbell map, Zaslavsky Map, Permutation, Subsitusion.

1. INTRODUCTION
In the course of recent decades, chaos has been discovered to be exceptionally helpful and to have extraordinary potential in diverse fields of physics, mathematics, engineering, biology, chemistry, and financial matters. Exactly toward the begin of a decade ago, the persistent and additionally discrete chaotic dynamical systems have been utilized for the advancement of cryptosystems.

The reason of applying chaos theory in cryptography lies in its intrinsic features. These properties of chaos includes: affectability to modest changes in introductory conditions, random-like manners, ergodicity, unstable periodic orbits, desired diffusion and confusion properties, and one-way property. Due to these properties, chaotic systems have become a very good candidate for use in the field of cryptography [1, 2, 3].

2. CHOATIC MAP
Brief descriptions for three types of the chaotic maps (Tinkerbell map, Zaslavsky Map and Arnold Transform) that are used in this paper are introduced in the following section.

2.1 Tinkerbell Map
The Tinkerbell map is a 2-D discrete-time dynamical system, and is defined as:

$$X_{n+1} = X_n^2 - Y_n^2 + aX_n + bY_n \quad \text{.... 1}$$

$$Y_{n+1} = 2 * X_n * Y_n + cX_n + dY_n \quad \text{.... 2}$$

Where X_n , Y_n are current chaotic values and X_{n+1} , Y_{n+1}, are next chaotic values and a, b, c, d are control parameters. Details of Tinkerbell map can be found in [4]. The key set for Tinkerbell map is $\{X_0, Y_0, a, b, c, d\}$. Commonly used initial values and parameters are a= 0.9, b= -0.6013, c= 2.0, d=0.50.

2.2 Zaslavsky Map
Zaslavsky map is a discrete-time dynamical system, and is defined as:

$$x_{n+1} = \mathrm{mod}(x_n + v(1 + \mu y_n) + \varepsilon\, v \cos(2\pi x_n), 1) \quad \text{....3}$$

$$y_n = e^{-\tau}(y_n) + \varepsilon \cos(2\pi x_n) \quad \text{....4}$$

And $\mu = \frac{1-e^{-\tau}}{\tau}$

Where v, τ, ε are control parameters and e is exponentiation. The key set for Zaslavsky map is

$\{x_0, y_0, v, \tau, \varepsilon\}$. Commonly used values for the parameters are $v = 12.6695$, $\varepsilon = 9.1$, $\tau = 3.0$. Details of Zaslavsky map can be found in [5].

2.3 Arnold Transform
Arnold transform is chosen as pre-treatment method for open image as it is simple and periodic. Therefore, the scrambling transform gives a secondary security for the digital products. In addition, after scrambling transform, the spatial relationships of the pixels of an image has been destroyed, which makes it evenly distributed in all the space, so the robustness of the algorithm was improved in this way. The discrete Arnold transformation is defined as follows:

$$\begin{bmatrix} Xn+1 \\ Yn+1 \end{bmatrix} = \begin{bmatrix} 1 & a \\ b & a+b \end{bmatrix} \begin{bmatrix} Xn \\ Yn \end{bmatrix} = A \begin{bmatrix} Xn \\ Yn \end{bmatrix} \bmod N \quad \text{....5}$$

Where, $(x, y) = \{0, 1, \ldots . N\}$ are pixel coordinates from original image. (x_{n+1}, y_{n+1}) are corresponding results after Arnold Transform, a and b positive integers, det(A) =1 [6].

3. THE PROPOSED SYSTEM
The Proposed Image Encryption Algorithm is for the most part comprises of two stages. So as to dispose of the relationship of image pixels in closest region the pixel positions is blended this present the first stage. The second stage will be performed by change of pixel dark qualities. Fig. 1 demonstrates the general structure of the proposed image encryption and unscrambling calculation. The base line of the proposed image encryption system can be summarized as follows:

1. Read a gray scale image (img) of size N*N.
2. Mixing stage (first level):
   - Generation of Transformation key1 and key2.

- Transform the image columns with the Transformation key1.
- Transform the image rows with the Transformation key2.
- Apply Arnold Transform.
- Bit pixel permutation.

3. Confusion stage (second level):
   - Generation of mask key.
   - Application of XOR operation between mask key and image pixels resulted from first level.

The general structure of the proposed image decryption design can be summarized as follows:

1. Read the encrypted gray scale image (Eimg) of size N*N.
2. Inverse Confusion stage:
   - Generation of mask key.
   - Application of XOR operation between mask key and image pixels.
3. Inverse Mixing stage:
   - Inverse bit pixel permutation
   - Apply inverse Arnold Transform.
   - Generation of Transformation key1 and Transformation key2.
   - Transform the rows of the transformed image with the Transformation key2.
   - Transform the columns of the transformed image with the Transformation key1.

3.1 Mixing Stage

The motivation behind shuffling stage is to diminish the high connection between neighboring pixels in the plain image. Let I be a gray original image of size N*N, it is a matrix containing N rows and N columns, and the gray values ranges from 0 to 255. In the process of permutation the first step in this work is to create the necessary keys for scrambling the image data.

Firstly N/2 chaotic values $\{(xtin_1, xtin_2, \dots, xtin_{N/2})$, $(ytin_1, ytin_2, \dots, ytin_{N/2})$, $(xzas_1, xzas_2, \dots, xzas_{N/2})$ $yzas_1, yzas_2, \dots, yzas_{N/2}$ $)\}$ are initiated by using Eq. (1, 2, 3, 4), then chaotic values will be cross-coupled to form one matrix M of size N+N as shown in the Fig. 2. Then the First N of matrix M will be transcribed in Per_C and the second N in Per_R. Per_C And Per_R are arranged, and the positions of sorted chaotic values in the original chaotic sequence are found and stored in $Per_C'$and Per_R' .The next step is to scramble column position of all values from first row to last row according to Per_C' . Similarly scramble row position of all values from first column to last column according to Per_R' . This will form the first and second steps of the mixing stage.

The third step of mixing stage is applying Arnold Transform to the transformed image. The steps of applying Arnold Transform can be summarized as follows:

Step 1: Partition the transformed image $I_{Tr}(r, c)$ into 8×8 size blocks,$blk_1, blk_2, \dots, blk_n$.

Step 2: Apply Arnold Transform within block B_i to rearrange the pixels of the block, where$i = 1, 2, 3, \dots, n$.

Step 3: Apply Arnold Transform within whole image $I1_{Tr}(r, c)$ to rearrange the pixels .After repeating this step for n times shuffled image S(x, y) is finally produced.

The fourth step of the mixing stage is the bit pixel permutation. The image can be seen as an array of pixels, each with eight bits for 256 gray levels. In the bit permutation technique the bits in each pixel taken from the image are permuted. The entire array of these permuted pixels forms the encrypted image. Suppose the current pixel value is 65 convert it to its binary presentation then rotate only one bit to the left. After bit pixel permutation the pixel value will be 192. Bit pixel permutation used in this paper is shown in Fig. 3. This stage shuffles all pixels and de-correlates the neighboring pixels

3.2 Substitution Stage

As a start, generate $(\mathrm{N} \times \mathrm{N} \times 8)/2$ chaotic values of xtin, ytin, xzas, yzas the generated chaotic series which are a factual valued. These factual valued cross coupled into two matrix $mask\_key1$ and $mask\_key2$ of size $(\mathrm{N} \times \mathrm{N} \times 8)$ for each as shown in the Figure 2. $mask\_key1$ And $mask\_key2$ can be transformed into binary series as following:

$$\mathrm{ser1}_{i,1} = \mathrm{Round}\left(\mathrm{Abs}\left(mask\_key1_{i,1} \times 10^{14}\right)\right) \bmod 2 \dots .6$$

$$\mathrm{ser2}_{i,1} = \mathrm{Round}\left(\mathrm{Abs}\left(mask\_key2_{i,1} \times 10^{14}\right)\right) \bmod 2..7$$

Each 8 consecutive bit of $\mathrm{ser1}_{i,1}$ and $\mathrm{ser2}_{i,1}$will be converted into integer value after xored with the following 8 consecutive bit in $\mathrm{ser1}_{i,1}$and $\mathrm{ser2}_{i,1}$. Reshape integer sequence stored in array$\mathrm{ser1}_{i,1}$ and $\mathrm{ser2}_{i,1}$into a two dimensional matrix Mask1 (N*N) and Mask2(N*N). Finally, Mask1 and Mask2 are used to form the Mask in which each pixel of Shuffled image [i,j] is xored with corresponding value of Mask [i,j].
Masking operation is performed by following formula:

$$\mathrm{enc}_{\mathrm{img(i,j)}} = \left(\left(\mathrm{shu}_{\mathrm{img(i,j)}} \otimes \mathrm{Mask(i,j)}\right) \otimes \mathrm{shu}_{\mathrm{img(i-1,j)}}\right)..9$$

The decryption process similar to the encryption one, described above, but in the reverse order.

The following matlab function is used to illustrate how to mix the two masks (Mask1 and Mask2) into one mask (Mask).

```
function [Mask]=Masking_key(Mask1,Mask2)
 x=1; y=2;
for i=1:r-2
  if mod(i,2)~=0
  Mask(x,:)= bitxor(Mask1(i,:),Mask2(i+1,:));
    x=x+2;
   else
     Mask(y,:)=bitxor(Mask1(i,:),Mask2(i+1,:));
     y=y+2;
  end
end
```

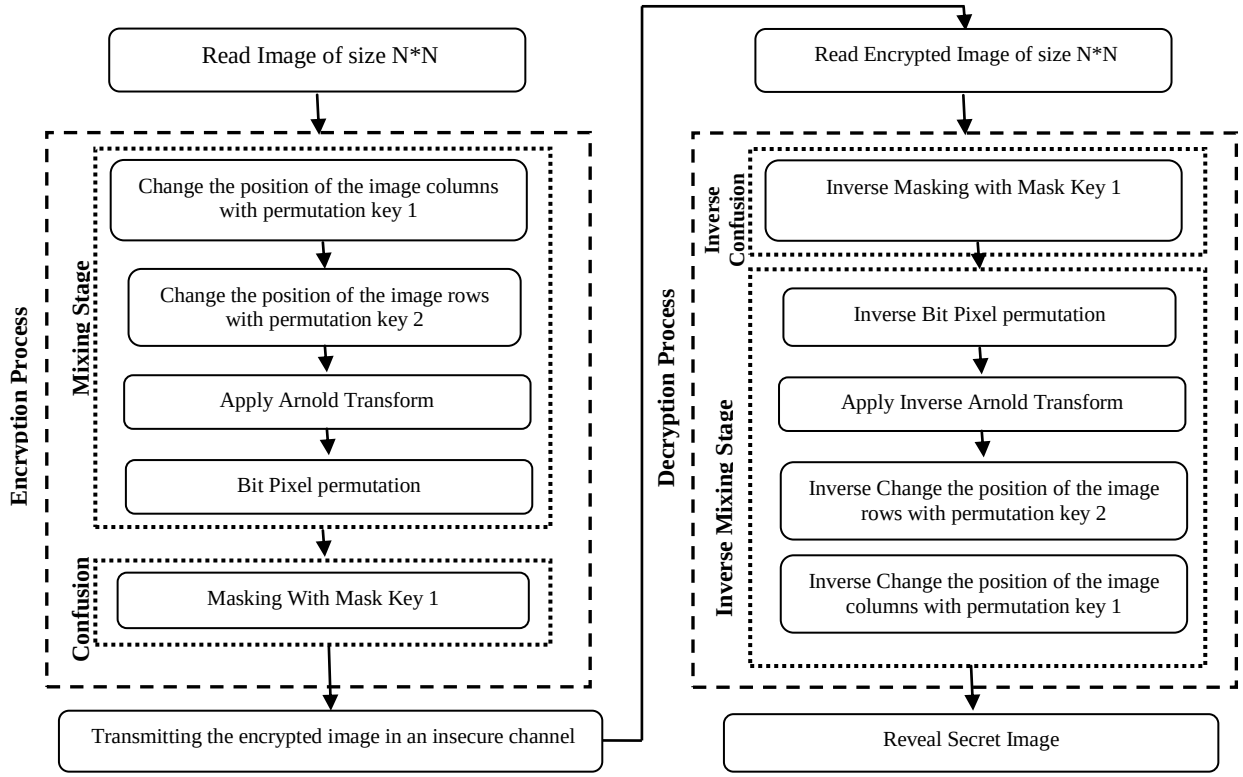


Fig 1: The General Structure of Encryption and Decryption Processes

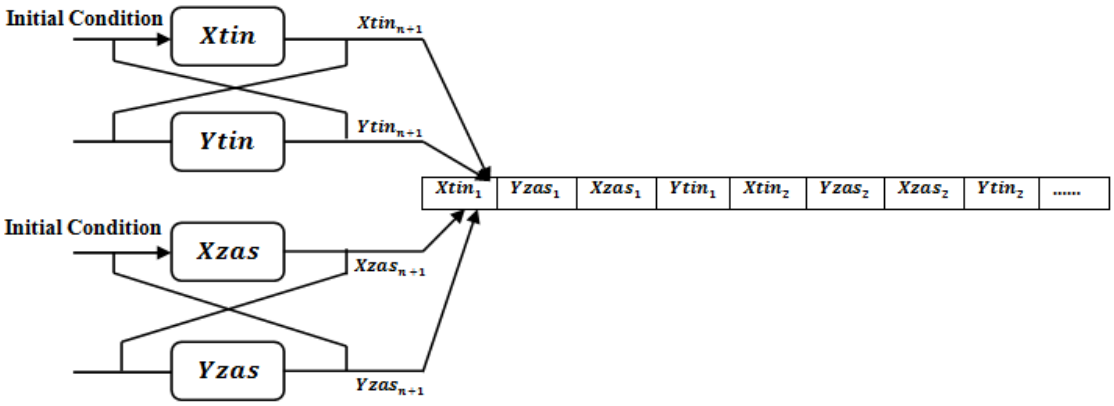


Fig 2: Mixing Keys Generation

Pixel Value = 65								
Pixel Binary presentation	0	1	0	0	0	0	0	1
Index	1	2	3	4	5	6	7	8

index	8	2	3	4	5	6	7	1
Pixel Binary presentation	1	1	0	0	0	0	0	0
Pixel Value = 192								

Fig 3: Bit-Pixel Permutation

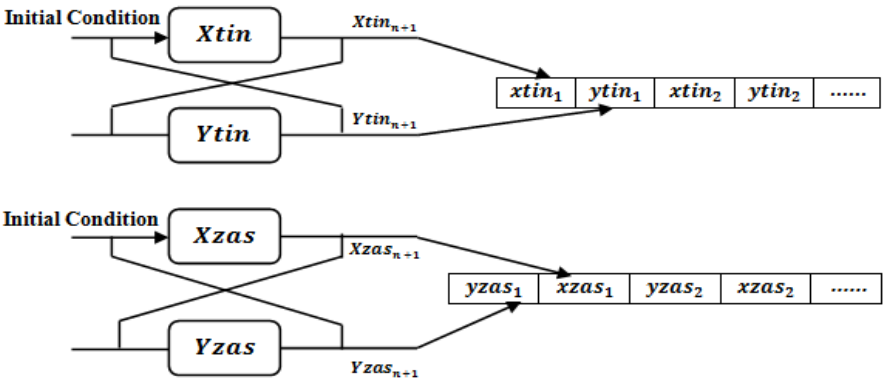


Fig 4: Substitution Keys Generation

4. SIMULATION RESULT

The following sub sections are devoted to present and discuss the results of the conducted tests to assess the performance of the proposed systems. Simulation of the proposed image encryption scheme was implemented using Matlab R2013a. six images of size 256×256 with gray-scale (0-255) were used .The outcomes of the performance of the proposed image encryption are showed as Fig. 5.

4.1 HISTOGRAM

The original image and the encrypted image are shown in figure 6 depicting the histograms. The histogram of the encrypted image is nearly uniformly distributed, which can well protect the information of the image to withstand the statistical attack.

4.2 CORRELATION ANALYSIS

The reliance of two neighboring variables at a certain bearing is measured by correlation coefficient. The more closely related these two variables are, the closer the correlation coefficient approaches 1. If they are less closely related, the value of correlation coefficient approaches 0. The correlation coefficient of adjacent pixels is calculated according to the following equations:

$$Cr = \frac{cov_{x,y}}{\sqrt{D_x}\sqrt{D_y}} \ldots.10$$

$$cov_{x,y} = \frac{1}{T}\sum_{i=1}^{T}\big(x_i - E(x)\big)\big(y_i - E(y)\big) \ldots.11$$

$$E(x) = \frac{1}{T}\sum x_i \ , D(x) = \frac{1}{T}\sum\big(x_{i-} E(x)\big)^2 \ldots.12$$

Where x and y are gray values of two adjacent pixels in an encrypted image. 5000 pairs of horizontally were randomly selected, vertically, and diagonally neighboring pixels and calculate the correlation coefficients in three directions separately [7]. The correlation coefficient of the pixel pair is calculated and the result is listed in Table 1 and the distribution is shown in Figure 5.

It is clear that neighboring pixels of the encrypted image have no relationship. Both the correlation coefficients in Table 1 and figure 5 give explanation for the claim that neighboring pixels of the plain image are uncorrelated by the proposed cryptosystem effectively.

4.3 Entropy

Average information of the image content is called entropy and should be nearly equals to 8. The entropy H(m) of a message source m can be calculated as:

$$H(m) = \sum \ P(m_i)\ \log_2 \frac{1}{p(m_i)} \ldots.13$$

Where p(mi) represents the probability of symbol mi and the entropy is expressed in bits. Details of entropy can be found in [8]. The qualities computed in Table 2 are near the perfect value.

4.4 KEY SPACE ANALYSIS:

Key space size is the total number of different keys that can be used in the encryption. For a 10^{-10} floating point precision, all keys parameters (4 introductory status, 8 control parameters to Transformation keys and 4 introductory status , 8 control parameters to mask key) can take 10^{10} possible values. Therefore, the key space comes out as $((10^{10})^{24}$, which are big as much as necessary to defend against the brute force-attack.

4.5 NPCR & UACI

The numbers of pixels change rate (NPCR) have also measured to see the influence of changing a single pixel in the original image on the encrypted image by the proposed algorithm. The NPCR measure the percentage of different pixel numbers between the two images. The NPCR is defined by the following equation:

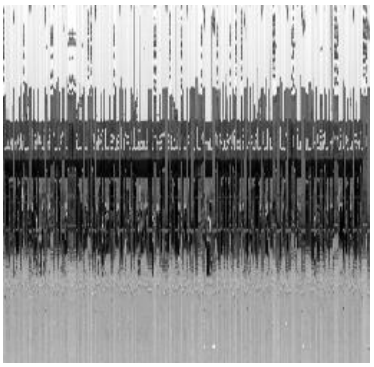
$$NPCR = \frac{\sum_{i,j} D(i,j)}{M \times N} \times 100\% \ldots.14$$

The UACI measures the normalized mean difference rate between the plain image and the encrypted one. UACI is defined by the following formula [9, 10]:

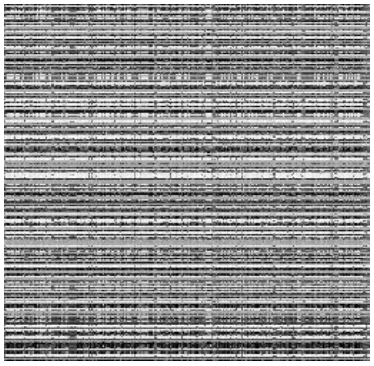
$$\text{UACI (p, s)} = \left(\frac{1}{n}\sum\frac{|p_i - S_i|}{2^{32}-1}\right) \times 100\% \ldots.15$$

Table 3, showing The NPCR and UACI values. The obtained NPCR for images encrypted by using the proposed encryption scheme found to be over 99% showing thereby that the proposed encryption algorithm is very sensitive with respect to small changes in the plaintext. NPCR and UACI prove it is possible to obtain excellent pixel alteration effects.

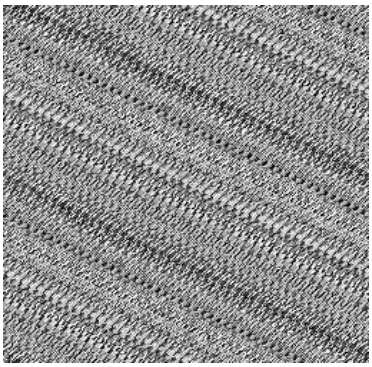
(A) Mustansiria University



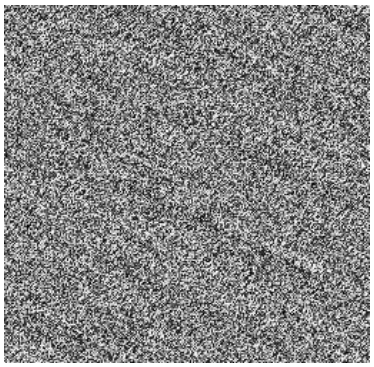
(B) Colum Permutation



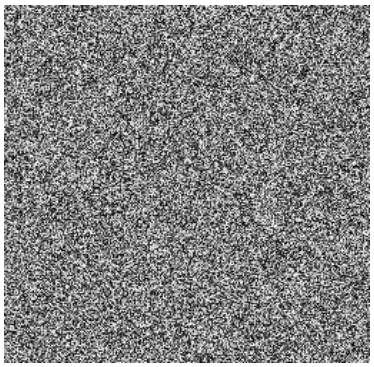
(C) Row Permutation



(D) Arnold Transform



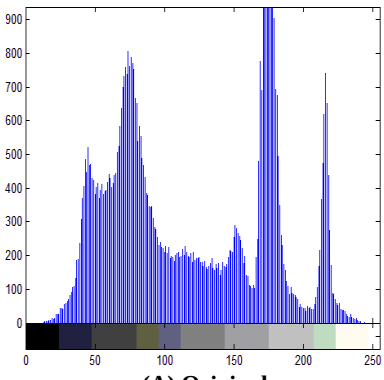
(E) Bit Pixel Permutation



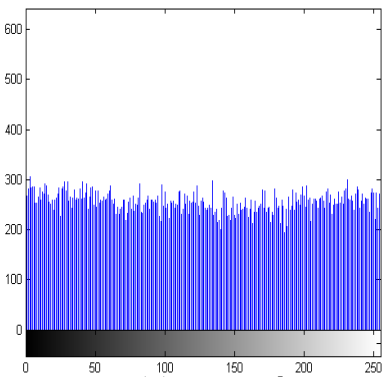
(F) Encrypted Image

(G) Decrypted Image

Fig 5: Image Encryption Algorithm Result



(A) Original



(B) Encrypted

Fig 6: Original Image Histogram and Encrypted

Table 1: The correlation of adjacent pixels

case	Original Image		
	Horizontal	Vertical	Diagonal
1.	0.9631	0.9759	0.9521
2.	0.8208	0.9144	0.8073
3.	0.9732	0.9588	0.9412
4.	0.9477	0.9296	0.8879
5.	0.9817	0.9904	0.9749
6.	0.9685	0.9680	0.9499
Case	**Encrypted Image**		
	Horizontal	**Vertical**	**Diagonal**
1.	0.0024	-0.0017	0.0033
2.	0.0036	-0.0006	-0.0019
3.	-0.0073	0.0116	-0.0040
4.	-0.00005	0.0001	-0.0035
5.	-0.0042	0.0192	-0.0058
6.	-0.0026	-0.0038	0.0043

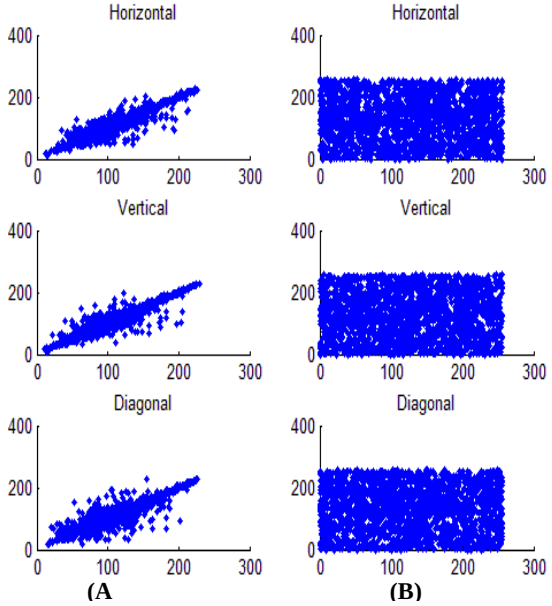


Fig 7: Correlation of two adjacent pixels: (A) for plain image, and (B) for its encrypted image

5. CONCLUSIONS

In this paper, a new algorithm of encryption and decryption of images is proposed. The encryption algorithm use two concepts, i.e., confusion and diffusion which is also called permutation and substitution among the pixels of the gray scale image. To perform the confusion in the plain-image"s pixels, a blend of 2D chaotic map is used which include Tinkerbell map, Zaslavsky map and Arnold Transform. While only Tinkerbell map and Zaslavsky map are used to performing the diffusion.

Experimental tests are carried out with detailed numerical analysis, including key space analysis, statistical attack analysis and differential attack analysis. All the experimental results demonstrate that the proposed encryption algorithm is secure thanks to its bulky key space; it's vastly sensitivity to the encryption keys. For future work 3D chaotic maps can replace the 2D chaotic maps used in the paper.

Table 2: Entropy value

case	Original Image	Encrypted Image
1.	7.4197	7.9932
2.	7.6863	7.9930
3.	7.5544	7.9942
4.	7.1333	7.9576
5.	6.2507	7.9108
6.	7.6134	7.9955

Table 3: NPCR and UACI values

case	NPCR	UACI
1.	0.99559	33.53
2.	0.99577	33.525
3.	0.9958	33.48
4.	0.9949	33.67
5.	0.99003	33.05
6.	0.9962	33.48

6. REFERENCES

[1] Akhavan A., Samsudin A. , Akhshani A., " A novel parallel hash function based on 3D chaoticmap ", EURASIP Journal on Advances in Signal Processing 2013.

[2] Gopalakrishnan T., Ramakrishnan S., Balakumar M., " Image Encryption using Chaos and Parity based Pixel Modification in Permutation", International Journal of Computer Applications® (IJCA) , 2014.

[3] Akhshani A. , Behnia S., A. Akhavan, Hassan H. Abu, Hassan Z. , "A novel scheme for image encryption based on 2D piecewise chaotic maps", Optics Communications 283 (2010) 3259–3266.

[4] Goldsztejn A., "Tinkerbell is chaotic", SIAM Journal applied dynamical system, 2011, vol. 10, No. 4, pp. 1480-1501.

[5] Stoyanov B., Kordov K., "Novel Zaslavsky Map Based Pseudorandom Bit Generation Scheme ", Applied Mathematical Sciences, Vol. 8, 2014, no. 178, 8883 - 8887.

[6] Tian-gong P., Li Da-yong, " A Novel Image Encryption Using Arnold Cat", International Journal of Security and Its Applications, Vol.7, No.5 (2013), pp.377-386.

[7] Tiegang G., Zengqiang C., "A new image encryption algorithm based on hyper-chaos", Elsevier B.V. All rights reserved 2007.

[8] Hanchinamani G., Kulakarni L., " A Novel Approach for Image Encryption based on Parametric Mixing Chaotic System ", International Journal of Computer Applications (0975 – 8887) ,Volume 96– No. 11, June 2014

[9] Huang C.K., Liao C.W., Hsu S.L., Jeng ·Y.C., "Implementation of gray image encryption with pixel shuffling and gray-level encryption by single chaotic system ", © Springer Science Business Media, LLC 2011.

[10] Chen G., Mao Y., & Chui, C. K., "A symmetric image encryption scheme based on 3D chaotic cat maps.Chaos", Solitons and Fractals, 21, 749–761, 2004.