VOL. 18, NO. 8, APRIL 2023 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

MULTIPLE ROUNDS USING MIXED CHAOTIC KEYS METHOD FOR SECURE MESSAGE CRYPTOGRAPHY

Nasser Abdellatif[1], Adnan Manasreh[1], Mohammad S. Khrisat[2], Hatim Ghazi Zaini[3] and Ziad A. Alqadi[2]
[1]Department of Electrical Engineering, Applied Science Private University, Amman, Jordan
[2]Department of Computer Engineering, Faculty of Engineering Technology, AL-Balqa Applied University, Amman, Jordan
[3]Computer and Information Technology College, Taif University, Taif, Kingdom of Saudi Arabia
E-Mail: nasser_abdellatif@asu.edu.jo

ABSTRACT

Protecting secret messages from being hacked is a vital problem. This research paper will present a simple and flexible data cryptography method. The method can encrypt-decrypt messages of any length. Also, it can be easily used to encrypt-decrypt gray and color images. The proposed method will use a private key, which contains the needed information to generate chaotic keys; these keys will be used to obtain indices keys, which will be used as a lockup table in the encryption and decryption phases. The generated key will be very sensitive to the P.K. contents. The P.K. used in the proposed method will have a complex structure, providing a huge key space that can resist attacks. The proposed method will be implemented using various messages, the obtained results will be analyzed, and several test analysis methods will be used to prove the achievements of the proposed method. The quality analysis will use the calculated values of MSE, PSNR, CC, and NSCR to prove the quality of the proposed method in both the encryption and decryption phases. A sensitivity analysis will be performed to show how the encryption and decryption phase is sensitive to the selected P.K.; any changes in the P.K. during the decryption phase will be considered a hacking attempt by producing a damaged decrypted message. A speed analysis will be provided to prove that the proposed method is efficient, and the obtained results will be compared with other methods' results to show the speedup provided by the proposed method.

Keywords: cryptography, P.K., CTK, CLK, I.K., CTMM, CLMM, MSE, PSNR, CC, NSCR.

Manuscript Received 22 January 2023; Revised 25 May 2023; Published 15 June 2023

ABBREVIATIONS

The following abbreviations will be used in this research paper:

PK: private key
IK: indices key
CTK: chaotic tent key
CLK: chaotic logistic key
CTMM: chaotic tent map model
CLMM: chaotic logistic map model
MSE: mean square error
PSNR: peak signal-to-noise ratio
CC: correlation coefficient
NSCR: number of samples change ratio
ET: encryption time
DT: decryption time
TP: throughput

INTRODUCTION

Encryption is a measure of cyber security that protects personal data through the use of unique codes that obfuscate data and make it impossible for hackers to read. Encryption ensures that an organization's private data is secure, even if attackers manage to bypass the firewall [25-30].

Companies increasingly collect a lot of private user data to prevent this data from getting into unauthorized outside hands. Therefore, companies need to ensure that they encrypt all the data in their possession.

The data encryption process is straightforward. An encryption key is used with a specific encryption algorithm to translate plain text data into unreadable data, and the data after the encryption process is called ciphertext.

Encrypted data can only be decrypted using the corresponding encryption key, so hackers will not be able to read the data even when they bypass system security measures.

Data encryption is an important and vital task because of the following reasons:

- The data being transferred is at risk

Hackers can easily attack communication channels and capture the data transmitted over the network. Data encryption ensures that confidential data cannot be stolen during its transmission through data mining techniques such as brute-force attacks.

- Security threats are evolving

Attackers are constantly learning new ways to bypass the most complex cyber security defenses, so companies need to ensure that attackers will not be able to read their data should their firewalls be compromised. Data encryption makes it impossible for attackers to access sensitive data.

- Unauthorized applications may expose sensitive data

Installing and using some unauthorized online resources and applications may cause intrusions and security breaches. When this happens, your sensitive data may be exposed to malicious agents online. Data encryption prevents this from happening by making sure

that your private data is encrypted and hard to read by unauthorized users.

- Hacking is a business

Competition and business rivalry mean companies are always looking to access their competitors' data to know their next business steps. This has led to the growth of the hacking business, whereby tech-savvy individuals hack companies' online resources for a fee. Data encryption will ensure that no one who is not authorized to access your data can read, understand or use your confidential data, even if that person manages to bypass your security systems.

The process of message cryptography can be summarized in two main steps: encryption and decryption.

First: The text we want to encrypt is taken, be it a text message or an email, and complex encryption algorithms are applied to that text and converted into an unreadable format. The text after conversion is called "ciphertext." This helps protect the confidentiality of digital data, whether stored on Computer systems or transmitted over a network such as the Internet (see Figure-1).

Second: When the encrypted text reaches the intended recipient, the information is translated back to its original form; that is, this text is returned to what it was before encryption algorithms were applied to it, and this is called decryption. Finally, for the recipient to unlock the message, both the sender and receiver must use a "secret" encryption key, a set of algorithms that defend and decrypt the data into a readable format [31-39].

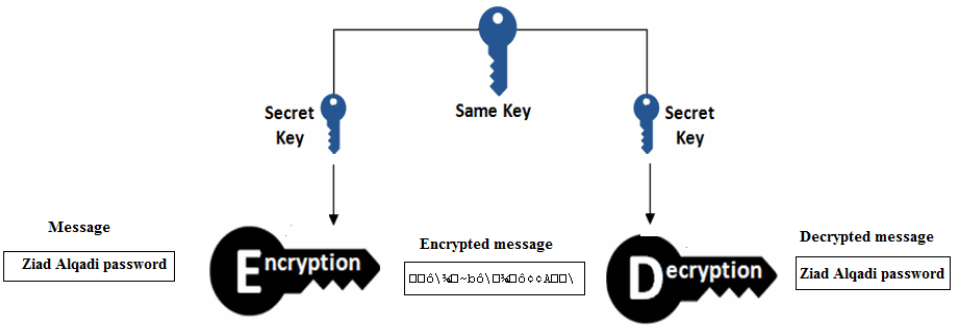

Figure-1. Message cryptography process.

The aim of this research paper is to introduce a method of message cryptography that must provide the following:

- Provide a low message quality in the encryption phase by destroying the message; here, the method must maximize MSE, NSCR and minimize CC and PSNR.
- Provide a high message quality in the decryption phase by producing a message identical to the original message; here, the method must maximize PSNR and CC and minimize MSE and NSCR.
- Increase the speed of message cryptography; here, the method must reduce the ET/DT and thus increase the throughput of the process of message cryptography.
- Increase the level of security by using a complicated PK, which must increase the key space to resist any attacks.
- It must be simple and easy to implement; changing the PK or/and the message must not require changing the algorithm.

RELATED WORK

Many methods for message cryptography were introduced; some of these methods were based on the standards DES, 3DES, AES, BF, RC2, and RSA [1]; these methods shared some features such as fixed data block size, fixed length PK, and fixed number of rounds, many research investigates these methods. In [14], the authors provided a comparison between the performances provided by these methods; the results of the comparisons are shown in Table-1.

Table-1. Comparisons between standard methods of message cryptography.

Ref. [12]	DES	3DES	AES	BF
Throughput (K bytes per second)	7.8008	2.6006	5.1953	9.9287

In [15], the authors introduced an enhanced methods base on the standard method; the throughput was enhanced as shown in Table-2:

Table-2. Enhanced standard methods of message cryptography.

The Total Average Time and Throughput	The Algorithms AES (256-bit)	DES (56-bit)	3DES (168-bit)	RC2 (128-bit)	RSA (2048-bit)
Total Average Time	152431	166800	178494	182875	259978
Throughput(MByte/sec)	8.630882	7.887374	7.370634	7.194062	5.060482

Many other methods and techniques have been introduced for dealing with data of large sizes, such as digital images and digital letters. Some methods were based on chaos theory and used various chaotic map models to generate the key required to decrypt the cipher.

In [2], the authors made a performance comparison between the chaotic and non-chaotic methods of data encryption, the chaotic methods of speech coding improved the performance of the cipher, and it was found that the average throughput of the non-chaotic, chaotic, and excessive random methods was 170.3, 141.2 and 636.3 kbps, respectively, which is better than the standard methods.

Other researchers introduced some techniques to improve the data encoding speed by increasing the TP of the data encryption process; in [8], the researchers introduced a method, the throughput was improved to 169.1 kbps, while in [9], the introduced method boosted the throughput to 710 kilobytes per second.

Faster data encryption methods were introduced later; these methods were used to reduce ET-DT and increase data encoding speed. In [3], the researchers introduced a robust and fast image encoding scheme based on mixing technology. These methods are designed to encode digital images and can be adapted to encode digital speech files. In [4], researchers introduce a chaotic system based on cosine transform for image coding. In contrast, in [5], researchers present a new image coding algorithm based on a polynomial combination of chaotic maps and dynamic function generation. In [6], researchers introduced a multi-image coding algorithm based on DNA coding and chaotic order, while in [7], researchers produced multi-image ciphers with bit-level decomposition and chaotic maps; these methods provided good quality at different speeds, as shown in Table-3.

Table-3. Performance comparisons of method mentioned in [3.7].

Method	Throughput(K bytes per second)
In [3]	888.8867
In [4]	638.4082
In [5]	911.0352
In [6]	360.4102
In [7]	384.9609

Image coding requires additional processing due to its size and a large number of blocks; this will negatively affect the performance of the methods, so using speech files in the previous methods will drop the TP for encryption.

A recently invented method for encrypting color image files has shown that using chaotic maps to create speech coding algorithms leads to high levels of security. In [8], the researchers proposed an encryption method using confusion and diffusion based on a chaotic multi-coil system. In [9], the authors evaluated Lorenz and Rosslea's chaotic speech signal coding system. In [10], the researchers presented a more valuable paper, where the Bernoulli chaotic map to create the encryption algorithm. In [11], the researchers presented another method similar to the random audio coding algorithm. Important research in [12] has shown that using a single chaotic map is not always a guarantee of the highest levels of cryptographic security. The paper gives an overview of how cryptographic algorithms can be compromised using the Arnold Kat, Becker or 2D Logistic Chaos Map. This is one of the reasons why we use a combination of two messy maps to expand the key space (all possible values for secret keys) for additional cryptographic security. In [13], the author introduced a new audio coding algorithm with permutation substitution geometry, various analyses were carried out, including velocity analysis, and Table-4 shows what was obtained in the results of this research:

Table 4 : Results provided in [13]

Speech file	File size (K byte)	Encryption time (second)	Throughput(K bytes per second)
1	41.1	0.130	316.1538
2	98.6	0.245	402.4490
3	138	0.333	414.4144
4	277	0.694	399.1354
5	544	1.324	410.8761
6	1080	2.688	401.7857
7	2330	5.767	404.0229
Average		1.5973	405.4472

Given the previous experience in this field of data encryption, we have built a new algorithm for digital color image encryption - decryption using a complex PK; this key will have a complex structure to increase the security level and maintain the optimum encryption speed and less than the speed provided by the newly used methods of image cryptography.

Proposed Method PK

The PK used in the proposed method is a complicated key with a complex structure. The structure of the PK is shown in Figure-2. The PK has 10 components, each with a double data type. Furthermore, each component value is to be represented by 64 bits. This will provide the key with a huge key space that can resist any kind of hacking attack.

The PK contains information needed to run one CTMM and four CLMMs; these chaotic models based on the selected chaotic parameters are needed to generate 5 chaotic keys, one CTK, and 4 CLKs. The four CLKs are to be combined in one key. The length of each generated chaotic key must equal 256 to cover the ASCII characters range (0 to 255). The generated two chaotic keys are converted to indices keys, which will be used as lockup tables in the encryption and decryption phases. The encryption/decryption will be performed in two rounds. Each round will use one IK, the number of rounds may be expanded, and here we have to expand the structure of the PK by adding the necessary information to generate the required keys.

PK									
CT parameters		CL parameters							
		CLMM1		CLMM2		CLMM3		CLMM4	
u	x	r	x	r	x	r	x	r	x
Example									
1.99	0.7	3.99	0.25	3.78	0.124	3.91	0.0345	3.85	0.095

Figure-2. PK structure.

It is very easy to generate the required keys. For example, the following code can be easily used to generate the required keys based on the information included in the PK:

```
u=1.799;x(1)=0.098;%PK contents
r1=3.9; x1=0.725;
r2=3.8; x2=0.125;
r3=3.85; x3=0.065;
r4=3.79; x4=0.097;
for i=1:255              %CTK generation
    if x(i) < 0.5 %and condition
       x(i+1)=u*x(i); %is it right
    else
       if x(i) >= 0.5 %and this condition
       x(i+1)=u*(1-x(i)); %is it right
       end
    end
end
[unused,CTK]=sort(x);  %Converting to IK
CTK=CTK-1;
for i=1:64                % CLK generation
          x1=r1*x1*(1-x1);
          x2=r2*x2*(1-x2);
          x3=r3*x3*(1-x3);
          x4=r4*x4*(1-x4);
    key1(i)=x1;
    key2(i)=x2;
    key3(i)=x3;
    key4(i)=x4;
end
key=[key1 key2 key3 key4]; %Combining in one CLK
[notused,CLK] = sort(key);  %Converting to IK
CLK=CLK-1;
```

Figure-3 shows an example of the generated key (key length=16), while Figure-4 shows the contents of these keys:

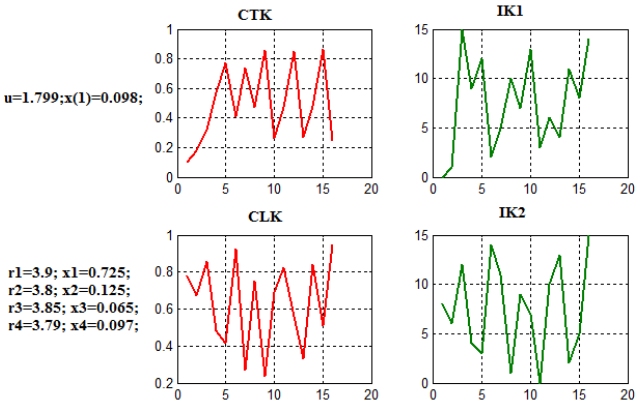

Figure-3. Plots of the generated keys (example).

CTK	IK1	CLK	IK2
0.0980	0	0.7776	8.0000
0.1763	1.0000	0.6745	6.0000
0.3172	15.0000	0.8562	12.0000
0.5706	9.0000	0.4802	4.0000
0.7725	12.0000	0.4156	3.0000
0.4092	2.0000	0.9229	14.0000
0.7362	5.0000	0.2702	11.0000
0.4745	10.0000	0.7494	1.0000
0.8537	7.0000	0.2340	9.0000
0.2632	13.0000	0.6901	7.0000
0.4735	3.0000	0.8234	0
0.8518	6.0000	0.5598	10.0000
0.2666	4.0000	0.3320	13.0000
0.4797	11.0000	0.8405	2.0000
0.8630	8.0000	0.5081	5.0000
0.2465	14.0000	0.9473	15.0000

Figure-4. Keys contents.

The generated keys are very sensitive to any minor changes in the PK components values, and any change will lead to a change in the keys, as shown in figures 5 and 6 (Using updated PK):

PK:

u=1.999; x (1) =0.278;
r1=3.99; x1=0.325;
r2=3.6; x2=0.025;
r3=3.85; x3=0.165;
r4=3.79; x4=0.25;

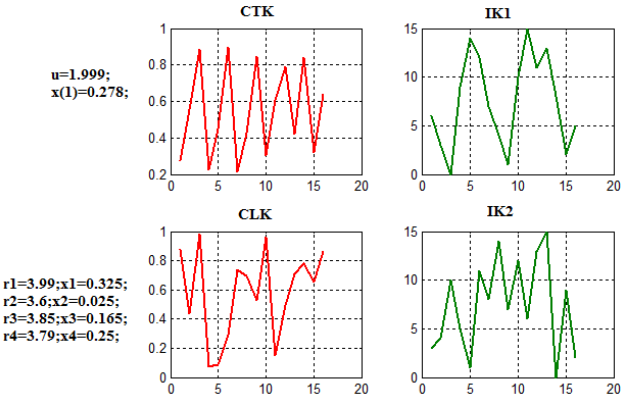

Figure-5. Keys plots after changing PK.

CTK	IK1	CLK	IK2
0.2780	6.0000	0.8753	3.0000
0.5557	3.0000	0.4355	4.0000
0.8881	0	0.9809	10.0000
0.2237	9.0000	0.0748	5.0000
0.4471	14.0000	0.0878	1.0000
0.8938	12.0000	0.2882	11.0000
0.2124	7.0000	0.7385	8.0000
0.4245	4.0000	0.6953	14.0000
0.8486	1.0000	0.5304	7.0000
0.3026	10.0000	0.9589	12.0000
0.6049	15.0000	0.1516	6.0000
0.7897	11.0000	0.4952	13.0000
0.4204	13.0000	0.7106	15.0000
0.8403	8.0000	0.7794	0
0.3192	2.0000	0.6517	9.0000
0.6381	5.0000	0.8603	2.0000

Figure-6. Keys values after changing PK.

The chaotic keys will be fixed and remain without changes when fixing the chaotic parameters; they differ from random keys, which require saving because they will change from time (run) to time, as shown in Figure-7.

First run		Second run	
0.8556	0.2904	0.8556	0.0336
0.4819	0.6171	0.4819	0.0688
0.9737	0.2653	0.9737	0.3196
0.0998	0.8244	0.0998	0.5309
0.3503	0.9827	0.3503	0.6544
0.8876	0.7302	0.8876	0.4076
0.3892	0.3439	0.3892	0.8200
0.9271	0.5841	0.9271	0.7184
0.2636	0.1078	0.2636	0.9686
0.7570	0.9063	0.7570	0.5313
0.7174	0.8797	0.7174	0.3251
0.7906	0.8178	0.7906	0.1056
0.6456	0.2607	0.6456	0.6110
0.8924	0.5944	0.8924	0.7788
0.3746	0.0225	0.3746	0.4235
0.9137	0.4253	0.9137	0.0908
0.3075	0.3127	0.3075	0.2665
0.8305	0.1615	0.8305	0.1537
0.5489	0.1788	0.5489	0.2810
0.9657	0.4229	0.9657	0.4401
0.1293	0.0942	0.1293	0.5271
0.4391	0.5985	0.4391	0.4574
0.9605	0.4709	0.9605	0.8754
0.1479	0.6959	0.1479	0.5181
0.4915	0.6999	0.4915	0.9436
0.9747	0.6385	0.9747	0.6377
CLMM key	Random key	CLMM key	Random key

Figure-7. Chaotic keys vs. random keys.

PK contains chaotic parameters required to run one CTMM and four CLMMs; each generated key is very sensitive to any changes in the associated chaotic parameter. Changing one or two keys will change the results of the IKs, and thus will change the results of the encryption/decryption phases, so the same PK must be used in both the encryption and decryption phases. Figures 8 and 9 show how CLK is sensitive to any changes in the chaotic parameters:

CLMM parameters	Generated keys	
r1=3.9;x1=0.325;	4 21 24 9 17 5 15 7 22 2 25 19 13 11 10 12 18 1 6 14 16 8 23 20 3 26	Key 1
r2=3.8;x2=0.125;	11 26 14 3 1 24 9 12 22 7 20 5 18 16 15 17 4 19 6 21 8 23 2 13 25 10	Key 2
r3=3.85;x3=0.065;	19 25 22 6 14 1 9 20 26 17 23 4 12 7 15 2 10 11 3 16 8 13 5 21 24 18	Key 3
r4=3.79;x4=0.097;	5 21 16 8 24 1 14 3 26 19 6 22 12 17 10 9 11 18 25 2 13 23 7 15 20 4	Key 4

Figure-8. Various parameters values generate various CLK.

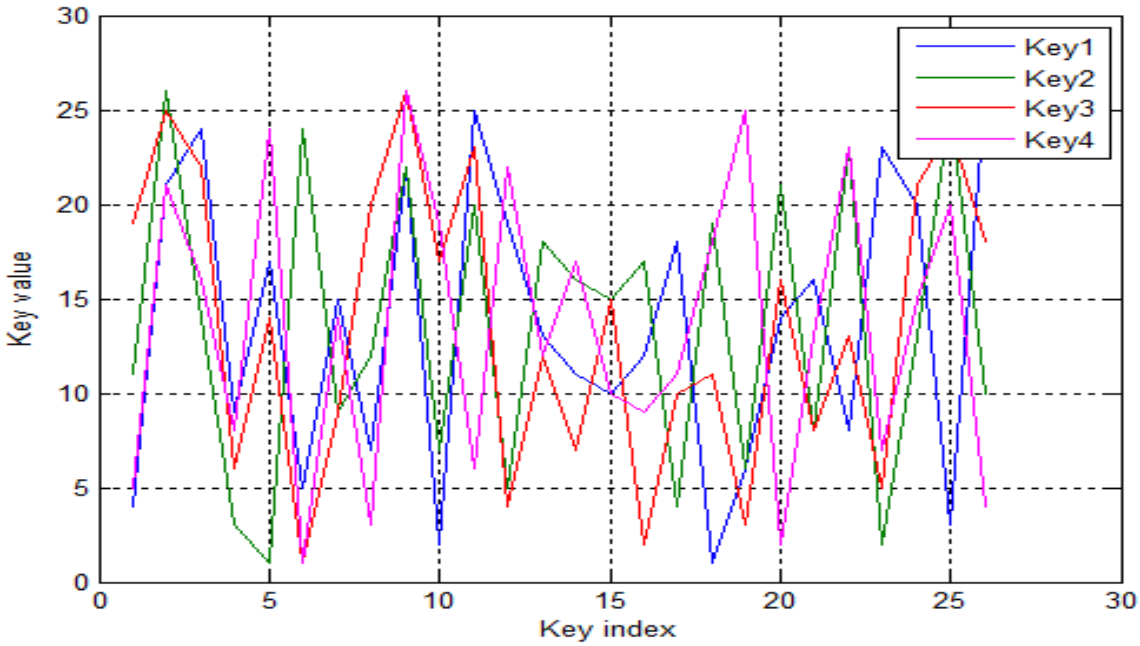

Figure-9. Various CLKs plots.

The Proposed Method

The proposed method uses the data in the PK to generate the required IKs. First, the chaotic tent parameters are used to generate a CTK key with 256 elements. Next, this key is to be converted to an indices key using the sort function; this key will be used in the first round of encryption as a lockup table to apply encryption.

The rest components of the PK are chaotic logistic parameters, and they are used to generate 4 CLKs; each CLK will have a length of 16, the four obtained CLKs will be combined to get on CLK, and this key then will be converted to IK to get the second key which will be used in the second round of encryption.

The generated IK will be used as a look-up table. In the encryption phase, the value from the image will be replaced by the index of the IK, where the value is stored to get the encrypted value.

In the decryption phase, the value from the encrypted message will search to find in what index the value is stored, and then the encrypted value will be replaced by the index.

Table-5 illustrates a worked example of how to apply encryption decryption using generated IKs:

Table-5. Worked example.

CK									
0.0973	**0.3503**	**0.9081**	**0.3330**	**0.8862**	**0.4024**	**0.9595**	**0.1552**	**0.5231**	**0.9954**
IK									
Index									
0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
IK contents									
0	**7**	**3**	**1**	**5**	**8**	**4**	**2**	**6**	**9**
Message									
4		3		3		2		7	
Encrypted: replace with index content									
5		1		1		3		2	
Decrypted: find the index content									
4		3		3		2		7	

The encryption phase, as shown in Figure-10, can be implanted by applying the following algorithm:

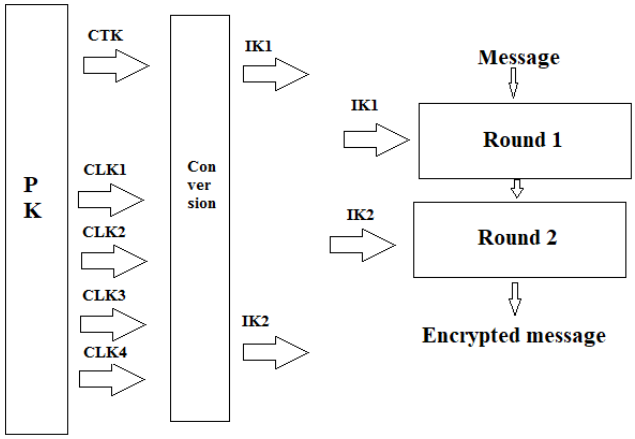

Figure-10. Encryption phase diagram.

Inputs
Message to be encrypted, PK

Output
Encrypted message

Process
1. Get the message.
2. Get the message length
3. Convert the message to decimal
4. Get the PK
5. Run CTMM to get CTK
6. Convert CTK to IK1
7. Run the CLMMs to generate 4 CLKs
8. Combine the CLKs into one CLK
9. Convert CLK to IK2
10. Apply round 1 using IK1
11. For each character value in the message do
12. Replace the character value by its index in the associated IK1
13. Apply round 2 using IK2
14. For each character value in the message encrypted in round 1 do
15. Replace the character value by its index in the associated IK1
16. Convert the encrypted message back to characters

The decryption phase can be implemented. Similarly, Figure-11 shows the decryption phase diagram, and this phase can be implemented by applying the following algorithm:

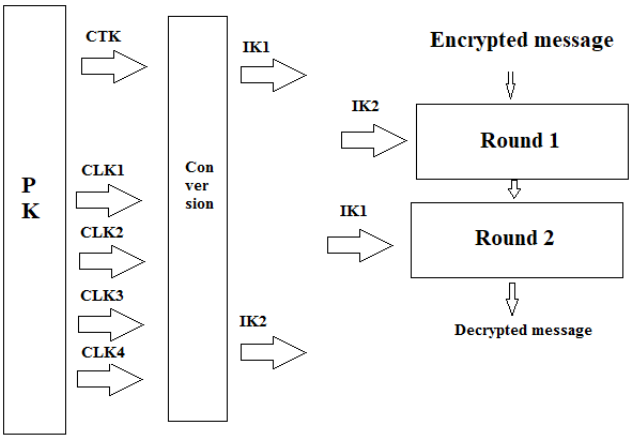

Figure-11. Decryption phase diagram.

Inputs
Encrypted message, PK

Output
Decrypted message

Process
1. Get the message.
2. Get the message length
3. Convert the message to decimal
4. Get the PK
5. Run CTMM to get CTK
6. Convert CTK to IK1
7. Run the CLMMs to generate 4 CLKs
8. Combine the CLKs into one CLK
9. Convert CLK to IK2
10. Apply round 1 using IK2
11. For each character value in the message do
12. Replace the character value by its index in the associated IK2
13. Apply round 2 using IK1

14. For each character value in the message decrypted in round 1 do

15. Replace the character value by its index in the associated IK1

16. Convert the decrypted message back to characters

Implementation and Results Analysis

The proposed method was implemented using various sizes of messages, and the obtained results were analyzed; several analysis types were performed to prove the enhancements provided by the proposed method. Below we will explain each of the analysis methods.

Quality Analysis

Here in this analysis, we will use the statistical parameters MSE, PSNR, CC, and NSCR, these parameters can be measured between two messages, and the values of these parameters will be used as indicators to verify the quality. The quality of the message in the encryption phase must be low, and the message must be destroyed, while in the decryption phase, the message must be recovered, and the quality must be high. Table 6 shows the requirement of good quality provided by a good method of message cryptography:

Table-6. Quality requirements.

Parameter	Between source and encrypted images	Between source and decrypted images
MSE	Very high	0
PSNR	Very low	Infinite
CCr	Very low	1
CCg	Very low	1
CCb	Very low	1
NSCR	Closed to 100%	0

MSE and PSNR can measure the quality of two messages, the high value of MSE and low value of PSNR points to low quality, while low MSE and high PSNR point to high quality. A good method of message cryptography must provide a low quality in the encryption phase and a high quality in the decryption phase.

MSE and PSNR can be calculated using equations 1 and 2:

$$PSNR = 10 \log_{10} \frac{MAX^2}{MSE} \text{dB}, \quad (1)$$

$$MSE = \frac{1}{N} \sum_{i=1}^{N} (x_i - y_i)^2, \quad (2)$$

Where: MAX is the maximum possible value of message stream (In our case, the maximum value is 255), N is the total number of samples, xi and yi are the corresponding sample values of the plain and encrypted files.

- Measuring the correlation coefficient between two messages expresses the dependency between their corresponding ASCII values. This is another statistical evaluation for testing the quality of encryption algorithms. Calculating the correlation coefficient determines the level of correlation between two messages, and the correlation coefficient is always in the range [-1, 1]. Therefore, values between |1–0.7| are considered a strong correlation (samples from the source files are similar to samples from the encrypted file), the correlation between |0.7-0.3| is considered a medium correlation and values between |0.3-0| is considered as weak correlation.

The correlation coefficient can be calculated using equation 3:

N is the total number of ASCII values, xi and yi are the sample values of the plain and encrypted messages, x^- and y^- are the mean values of ASCII values, and cov (x, y) is the covariance between both messages.

$$\mathbf{CC}_{xy} = \frac{cov(x,y)}{\sqrt{D(x)}\sqrt{D(y)}}, \quad (3)$$

where

$$D(x) = \frac{1}{N} \sum_{i=1}^{N} (x_i - \bar{x})^2,$$

$$D(y) = \frac{1}{N} \sum_{i=1}^{N} (y_i - \bar{y})^2,$$

$$cov(x,y) = \sum_{i=1}^{N} (x_i - \bar{x})(y_i - \bar{y}),$$

The number of sample change rates (NSCR) is a robustness test for establishing the quality of encryption algorithms. The purpose of the test is to compare the corresponding sample values of the original and encrypted messages and to show the percent difference. NSCR can be calculated using equation 4:

$$NSCR = \frac{\sum_{i=1}^{N} D_i}{N} \times 100\%, \quad (4)$$

where

$$D_i = \begin{cases} 1, x_i \neq y_i \\ 0, Otherwise \end{cases}$$

The selected messages were processed using the proposed method, MSE and PSNR were calculated, and table7 shows the obtained results.

Table-7. MSE and PSNR results.

Message	Length (byte)	Between source and encrypted messages		Between source and decrypted messages	
		MSE	PSNR	MSE	PSNR
1	10	12636	15.3357	0	Infinite
2	100	11069	17.5487	0	Infinite
3	200	12038	16.8668	0	Infinite
4	500	11559	17.2730	0	Infinite
5	750	12245	16.6963	0	Infinite
6	1000	12186	16.7451	0	Infinite
7	2000	11198	17.5901	0	Infinite
8	4000	11133	17.6482	0	Infinite
9	8000	11539	17.2904	0	Infinite
10	16000	11586	17.2497	0	Infinite
11	32000	11595	17.2418	0	Infinite
12	64000	11649	17.1956	0	Infinite

The CCs and NSCRs were also calculated, and Table-8 shows the obtained results:

Table-8. CC and NSCR results.

Message	Length (byte)	Between source and encrypted messages		Between source and decrypted messages	
		CC	NSCR %	CC	NSCR %
1	10	-0.1579	100	1	0
2	100	-0.0768	100	1	0
3	200	-0.1007	100	1	0
4	500	-0.0409	100	1	0
5	750	-0.0788	100	1	0
6	1000	-0.0944	100	1	0
7	2000	-0.0278	100	1	0
8	4000	-0.0428	100	1	0
9	8000	-0.0740	100	1	0
10	16000	-0.0585	100	1	0
11	32000	-0.0649	100	1	0
12	64000	-0.0685	100	1	0

From Tables 7 and 8 and refereeing to the requirements shown in Table-6, the obtained results prove that the proposed method satisfies the requirement of good message cryptography.

Sensitivity Test Analysis

The generated chaotic keys used in message cryptography are very sensitive to the chaotic parameters included in the PK. Any changes in these parameters will change the generated CTK and CLKs; thus, IK1 and IK2 will change, and thus the process of cryptography will be affected. Figures (12 and 13) show the generated keys using different contents of the PK. Here we have to notice that changing one key will affect encryption and decryption results.

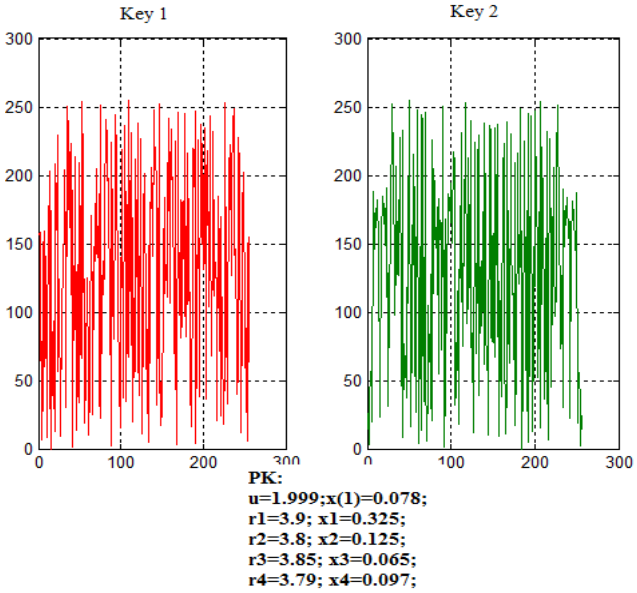

Figure-12. Generated keys using PK.

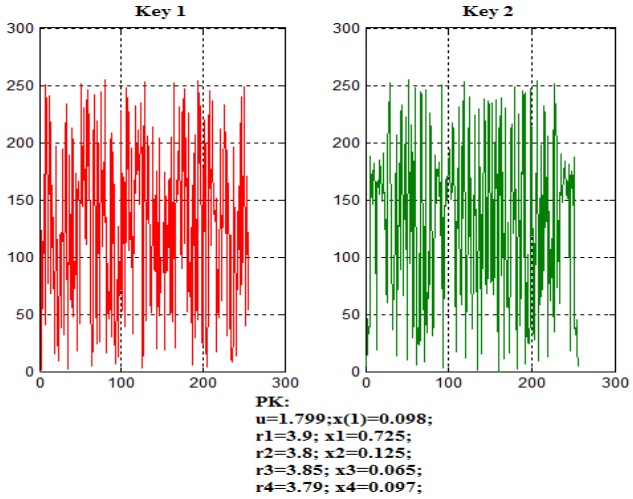

Figure-13. Changing PK changes the generated keys.

The Encryption and decryption phases must use the same PK; any changes in the PK during the decryption phase will be considered a hacking attempt by producing a damaged, corrupted decrypted message. The following example will illustrate this fact. Table-9 shows some selected various PKs. These keys were used in the encryption and decryption phases, and Table-10 shows the results of using these keys in the encryption and decryption phases using the source message 'Ziad Alqadi':

Table-9. Selected PKs (example).

PK1	PK2	PK3
u=1.999; x(1)=0.078; r1=3.9; x1=0.325;r2=3.8; x2=0.125;r3=3.85; x3=0.065; r4=3.79; x4=0.097;	u=1.899; x(1)=0.078; r1=3.88; x1=0.325;r2=3.8; x2=0.125;r3=3.85; x3=0.065; r4=3.79; x4=0.097;	u=1.999;x(1)=0.378; r1=3.72; x1=0.325;r2=3.8; x2=0.125;r3=3.85; x3=0.065; r4=3.79; x4=0.097;
PK4	PK5	PK6
u=1.799;x(1)=0.098; r1=3.9; x1=0.725;r2=3.8; x2=0.125;r3=3.85; x3=0.065; r4=3.79; x4=0.097;	u=1.999;x(1)=0.078; r1=3.9; x1=0.725;r2=3.8; x2=0.125;r3=3.85; x3=0.165; r4=3.79; x4=0.097;	u=1.999;x(1)=0.078; r1=3.9; x1=0.725;r2=3.8; x2=0.125;r3=3.85; x3=0.065;r4=3.79; x4=0.197;

Table-10. Results of using Message='Ziad Alqadi'.

Encryption key	Decryption key	Encrypted message	Decrypted message
PK1	PK1	|u□□□-:¤□□u	Ziad Alqadi
PK1	PK2	|u□□□-:¤□□u	□'□ F□~$□ '
PK1	PK3	|u□□□-:¤□□u	I@p·6□□gp·@
PK1	PK4	|u□□□-:¤□□u	~X□Z‰□□_□ZX
PK1	PK5	|u□□□-:¤□□u	û1□□7`C@□□1
PK1	PK6	|u□□□-:¤□□u	K□□□p□□□□□□

Table-10 results show that using different PKs in the decryption phase will produce a damaged message. Finally, the results of damaging the message are measured using MSE and PSNR, as shown in Table-11; here, the MSE between the source message and the decrypted one is

high (instead of being zero). At the same time, the PSNR is low (instead of being infinite).

Table-11. MSE and PSNR for the example.

Encryption key	Decryption key	Between source and encrypted		Between source and decrypted	
		MSE	PSNR	MSE	PSNR
PK1	PK1	8997.2	17.7149	0	Infinite
PK1	PK2	8997.2	17.7149	6238.1	21.2901
PK1	PK3	8997.2	17.7149	2144.4	27.4837
PK1	PK4	8997.2	17.7149	6193.5	22.9561
PK1	PK5	8997.2	17.7149	7819.5	20.8652
PK1	PK6			6881.3	17.2985

Speed Test Analysis

The selected messages were treated using the proposed method, the ET and DT were measured, and the TPs were calculated; Table-12 shows the obtained results:

Table-12. Speed results.

Message	Length (byte)	ET/DT (second)	TP(k bytes per second)
1	10	0.0010	9.7656
2	100	0.0015	65.1042
3	200	0.0019	102.7961
4	500	0.0021	232.5149
5	750	0.0022	332.9190
6	1000	0.0025	390.6250
7	2000	0.0032	610.3516
8	4000	0.0040	976.5625
9	8000	0.0045	1736.1
10	16000	0.0060	2604.2
11	32000	0.0065	4807.7
12	64000	0.0080	7812.5
13	128000	0.0100	12500
14	256000	0.0150	16667
15	512000	0.0250	20000
16	1024000	0.0420	23810
Average	128035	0.0085	5791.1

From Table-12, we can see that the proposed method is efficient by providing a high throughput with average equals to 5791.1 K bytes per second, the encryption time will slowly increase when increasing the message size, and the throughput will grow faster when increasing the message size as shown in Figure-14.

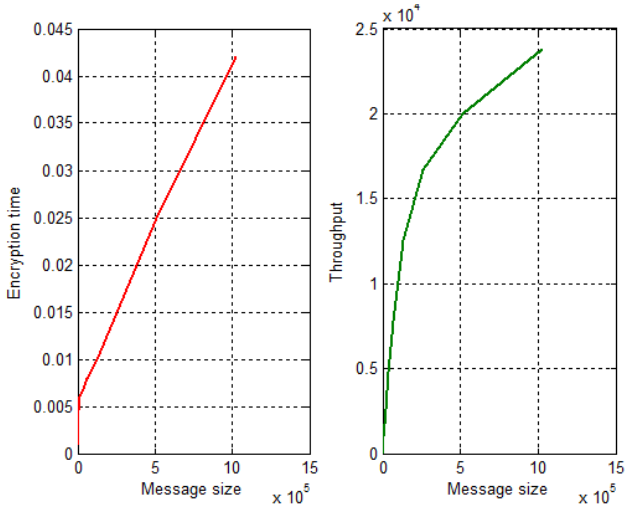

Figure-14. ET and TP VS message size.

For comparison purposes, the messages used in [14] (see Table-2) were encrypted-decrypted using the proposed method. The speed results are shown in Table-13; here, the average TP equals 23.0349 M bytes per second, which means that the proposed method enhanced the performance of the message cryptography and that the proposed method has a significant speedup compared with other existing methods.

Table-13. Results for comparisons with results in Table-2.

Message size	ET(second)	TP (M byte per second)
915 KB	0.0410	21.7939
5.384 MB	0.2260	23.8232
11.804 MB	0.5480	21.5400
35.350 MB	1.4150	24.9824
Average	0.5575	23.0349

Security Analysis

The PK key contains ten components, and each of these components has a double data type; the value of each component is to be represented by 64 bits, so the number

of combinations to guess this value equals 2 raised to the power of 64. So taking the ten values together, the total combinations will equal 2 raised to the power 640. This will provide a huge space that can resist any kind of attack. Also, if we take the IK with 256 values (from 0 to 255), the number of combinations to guess the IK equals the factorial of 256, which is also very huge and provides a huge key space.

Simplicity and Flexibility Analysis

The proposed method is very simple and easy to implement; changing the message or/and changing the PK does not require any changes in the method algorithm.

The proposed method is flexible; it can be used for gray and color image cryptography [17-24]. The pixel values are ASCII values and within the range of 0 to 255. Therefore, the generated IKs can also suit the gray and color images; the image to be encrypted/decrypted must be reshaped to a one-row matrix (it will like the message). This row matrix can be encrypted/decrypted using the same algorithm. After encryption/decryption, the produced image must be reshaped back for one row to a 2D matrix if the image is gray or to a 3D matrix if the image is a color, Figure-15 shows the results of encryption-decrypting a gray image, while Figure-16 shows the results of encrypting-decrypting color image [13-16].

Figure-15. Gray image encryption-decryption.

Figure-16. Color image encryption-decryption.

CONCLUSIONS

A simple and easy-to-implement method of message cryptography was presented. The method can be easily used to encrypt-decrypt any message with any length. Changing the message or/and changing the PK does not require any changes in the algorithm. Moreover, the proposed method is flexible can be used to encrypt-decrypt gray and color images without changing the algorithm. Two rounds of message cryptography were used; the number of rounds may be expanded depending on the user's wishes. The proposed method used a complicated PK that provides a huge key space that can resist attacks. The private key was used to generate chaotic keys very sensitive to the selected values of chaotic parameters; the chaotic keys were converted to indices keys, which were used as lockup tables to simplify the encryption and decryption processes.

The proposed method was implemented, several types of results analysis was performed and proved that the proposed method satisfied the quality requirements of good cryptography during the encryption and decryption phases.

The speed analysis proved that the proposed method is very efficient. The proposed method decreased ET and DT and increased the speed of message cryptography by maximizing the throughput. The proposed method has a significant speedup compared with other existing methods of message cryptography.

Appendix

For researchers who want to reproduce the outputs of the proposed method, the following Matlab codes may be useful:

```
%Preparing the message
clear all,clc,close all
in='Ziad Alqadi password';
ss =length(in);
im=uint8(in);

%First key generation
tic
N=256;
u=1.799;x(1)=0.098;
tic
for i=1:N-1
      if x(i) < 0.5 %and condition
         x(i+1)=u*x(i); %is it right
      else
         if x(i) >= 0.5 %and this condition
         x(i+1)=u*(1-x(i)); %is it right
         end
      end
end
[unused,CTK]=sort(x);
CTK=CTK-1;

%Second key generation
R1=64;
r1=3.9; x1=0.725;
r2=3.8; x2=0.125;
r3=3.85; x3=0.065;
r4=3.79; x4=0.097;
for i=1:R1
            x1=r1*x1*(1-x1);
           x2=r2*x2*(1-x2);
           x3=r3*x3*(1-x3);
           x4=r4*x4*(1-x4);
      key1(i)=x1;
      key2(i)=x2;
      key3(i)=x3;
      key4(i)=x4;

end
key=[key1 key2 key3 key4];
[notused,CLK] = sort(key);
CLK=CLK-1;

%First round encryption
numbers =double(im);
codednumbers = zeros(1,length(numbers));
for i = 1:length(numbers)
   ff=numbers(i)+1;
   codednumbers(i) = CTK(ff);

end
en1=codednumbers;

%Second round encryption
numbers =double(en1);
codednumbers = zeros(1,length(numbers));
for i = 1:length(numbers)
   ff=numbers(i)+1;
   codednumbers(i) =CLK(ff);
end
codednumbers=codednumbers;
en=codednumbers;
en=uint8(en);
ttt=toc;
```

REFERENCES

[1] Aamer Nadeem, Dr M. Younus Javed. 2005. A Performance Comparison of Data Encryption Algorithms, Conference Paper. DOI: 10.1109/ICICT.2005.1598556 · Source: IEEE Xplore.

[2] M. Bala Kumara, P. Karthikkab, N. Dhivyac, T. Gopalakrishnan. 2014. A Performance Comparison of Encryption Algorithms for Digital Images. International Journal of Engineering Research & Technology (IJERT). 3(2).

[3] Lee Mariel Heucheun Yepdia, Alain Tiedeu and Guillaume Kom. A Robust and Fast Image Encryption Scheme Based on a Mixing Technique, Security and Communication Networks, Vol. 2021 |Article ID 6615708 | https://doi.org/10.1155/2021/6615708

[4] Z. Hua, Y. Zhou and H. Huang. 2019. Cosine-transform-based chaotic system for image encryption. Information Sciences. 480: 403-419.

[5] M. Asgari-chenaghlu, M.-A. Balafar and M.-R. Feizi-Derakhshi. 2019. A novel image encryption algorithm based on polynomial combination of chaotic maps and dynamic function generation. Signal Processing. 157: 1.

[6] X. Zhang and X. Wang. 2019. Multiple-image Encryption Algorithm Based on DNA Encoding and Chaotic System, Springer, New York, NY, USA.

[7] J. S. Zhenjun and R. Sun. 2016. Multiple-image encryption with bit-plane decomposition and chaotic maps. Optics and Lasers in Engineering. 80: 1-11.

[8] Liu H., Kadir A., Li Y. 2016. Audio encryption scheme by confusion and diffusion based on multi-scroll chaotic system and one-time keys. Optik. 127, 7431-7438.

[9] Hato E., Shihab D. 2015. Lorenz and Rossler Chaotic System for Speech Signal Encryption. Int. J. Comput. Appl. 128, 09758887.

[10] Sathiyamurthi P., Ramakrishnan S. 2017. Speech encryption using chaotic shift keying for secured speech communication. EURASIP J. Audio Speech Music Process. 2017, 20.

[11] Tamimi A.A., Abdalla A.M. 2014. An Audio Shuffle-Encryption Algorithm. In Proceedings of the World Congress on Engineering and Computer Science, San Francisco, CA, USA, 22–24 October. Vol. 1.

[12] Preishuber M., Hütter T., Katzenbeisser S., Uhl A. 2018. Depreciating motivation and empirical security analysis of chaos-based image and video encryption. IEEE Trans. Inf. Forensics Secur. 13, 2137-2150.

[13] Krasimir Kordov. 2019. A Novel Audio Encryption Algorithm with Permutation-Substitution Architecture Electronics. 8(5): 530; https://doi.org/10.3390/electronics805053

[14] Priya Dhawan. 2002. Performance Comparison: Security Design Choices. Microsoft Developer Network October 2002. http://msdn2.microsoft.com/en-us/library/ms978415.aspx

[15] Israa Hashim Latif. 2020. Time evaluation of different cryptography algorithms using labview, IOP Conf. Series: Materials Science and Engineering 745, 012039, doi:10.1088/1757-899X/745/1/012039.

[16] Dr. Amjad Hindi, Dr. Majed Omar Dwairi, Prof. Ziad Alqadi. 2020. Analysis of Procedures used to build an Optimal Fingerprint Recognition System. International Journal of Computer Science and Mobile Computing. 9(2): 21-37.

[17] M. Abu-Faraj, A. Al-Hyari, K. Aldebei, B. Al-Ahmad and Z. Alqadi. 2022. Rotation Left Digits to Enhance the Security Level of Message Blocks Cryptography. IEEE Access, 10: 69388-69397, doi:10.1109/ACCESS.2022.3187317.

[18] M. Abu-Faraj, A. Al-Hyari and Z. Alqadi. 2022. Experimental Analysis of Methods Used to Solve Linear Regression Models. CMC-Computers, Materials & Continua, 72(3): 5699-5712, doi:10.32604/cmc.2022.027364. (Web of Science Indexed, Scopus Indexed).

[19] M. Abu-Faraj, A. Al-Hyari and Z. Alqadi. 2022. Complex Matrix Private Key to Enhance the Security Level of Image Cryptography. Symmetry, 14(4): 664-678, doi:10.3390/sym0664. (Web of Science Indexed, Scopus Indexed)

[20] M. Abu-Faraj, K. Aldebei, and Z. Alqadi. 2022. Simple, Efficient, Highly Secure, and Multiple Purposed Method on Data Cryptography. Traitement du Signal, 39(1): 173-178, doi:10.18280/ts.390117. (Web of Science Indexed, Scopus Indexed)

[21] M. Abu-Faraj and Z. Alqadi. 2021. Rounds Reduction and Blocks Controlling to Enhance the Performance of Standard Method of Data Cryptography. International Journal of Computer Science and Network Security (IJCSNS), 21(12): 648-656, doi: 10.22937/IJCSNS.2021.21.12.89. (Web of Science Indexed)

[22] M. Abu-Faraj and Z. Alqadi. 2021. Improving the Efficiency and Scalability of Standard Methods for Data Cryptography. International Journal of Computer Science and Network Security (IJCSNS), 21(12): 451-458, doi:10.22937/IJCSNS.2021.21.12.61. (Web of Science Indexed)

[23] M. Abu-Faraj and Z. Alqadi. 2021. Using Highly Secure Data Encryption Method for Text File Cryptography. International Journal of Computer Science and Network Security (IJCSNS), 21(12): 53-60, doi:10.22937/IJCSNS.2021.21.12.8. (Web of Science Indexed)

[24] M. Abu-Faraj and M. Zubi. 2020. Analysis and Implementation of Kidney Stones Detection byApplying Segmentation Techniques on Computerized Tomography Scans. Italian Journal of Pure and Applied Mathematics. (43): 590-602, (Scopus Indexed)

[25] Naseem Asad, Ismail Shayeb, Qazem Jaber, Belal Ayyoub, Ziad Alqadi, Ahmad Sharadqh. 2019. Creating a Stable and Fixed Features Array for Digital Color Image. IJCSMC. 8(8): 50-62.

[26] Majed O. Al-Dwairi, Amjad Y. Hendi, Mohamed S. Soliman, Ziad A.A. Alqadi. 2018. A new method for voice signal features creation. International Journal of Electrical and Computer Engineering (IJECE), 9(5): 4092-4098.

[27] Akram A. Moustafa and Ziad A. Alqadi. 2009. A Practical Approach of Selecting the Edge Detector Parameters to Achieve a Good Edge Map of the Gray Image. Journal of Computer Science. 5(5): 355-362.

[28] ZA Alqadi, MusbahAqel, Ibrahiem MM El Emary. 2008. Performance analysis and evaluation of parallel matrix multiplication algorithms. World Applied Sciences Journal. 5(2): 211-214.

[29] Ayman Al-Rawashdeh, Ziad Al-Qadi. 2018. Using wave equation to extract digital signal features. Engineering, Technology & Applied Science Research. 8(4): 1356-1359.

[30] ZiadAlqadi, Bilal Zahran, Qazem Jaber, Belal Ayyoub, Jamil Al-Azzeh. 2019. Enhancing the Capacity of LSB Method by Introducing LSB2Z Method. International Journal of Computer Science and Mobile Computing. 8(3): 76-90.

[31] Ziad A. Alqadi, Majed O. Al-Dwairi, Amjad A. Abu Jazar and Rushdi Abu Zneit. 2009. Optimized True-RGB color Image Processing. World Applied Sciences Journal. 8(10): 1175-1182, ISSN 1818-4952.

[32] Waheeb, A. and Ziad AlQadi. 2009. Gray image reconstruction. Eur. J. Sci. Res. 27: 167-173.

[33] A. A. Moustafa, Z. A. Alqadi. 2009. Color Image Reconstruction Using a New R'G'I Model. Journal of Computer Science. 5(4): 250-254.

[34] K Matrouk, A Al-Hasanat, H Alasha'ary, Z. Al-Qadi Al-Shalabi. 2014. Speech fingerprint to identify isolated word person. World Applied Sciences Journal. 31(10): 1767-1771.

[35] J. Al-Azzeh, B. Zahran, Z. Alqadi, B. Ayyoub, M. Abu-Zaher. 2018. A Novel zero-error method to create a secret tag for an image. Journal of Theoretical and Applied Information Technology. 96(13): 4081-4091.

[36] Prof. Ziad A.A. Alqadi, Prof. Mohammed K. Abu Zalata, Ghazi M. Qaryouti. 2016. Comparative Analysis of Color Image Steganography. JCSMC. 5(11): 37-43.

[37] M. Jose. 2014. Hiding Image in Image Using LSB Insertion Method with Improved Security and Quality. International Journal of Science and Research. 3(9): 2281-2284.

[38] M. Juneja, P. S. Sandhu. 2013. An improved LSB based Steganography with enhanced Security and Embedding/Extraction. 3rd International Conference on Intelligent Computational Systems, Hong Kong China, January 26-27.

[39] H. Alasha'ary, K. Matrouk, A. Al-Hasanat, Z. Aalqadi, H. Al-Shalabi. 2013. Improving Matrix Multiplication Using Parallel Computing. International Journal on Information Technology (I.RE.I.T.). 1(6), ISSN 2281-2911.