http://dx.doi.org/10.5392/IJoC.2015.11.4.056

An Efficient Image Encryption Scheme Based on Quintuple Encryption Using Gumowski-Mira and Tent Maps

Gururaj Hanchinamani, Linganagouda Kulkarni
BVB College of Engineering & Technology, Hubli-580031, India

ABSTRACT

This paper proposes an efficient image encryption scheme based on quintuple encryption using two chaotic maps. The encryption process is realized with quintuple encryption by calling the encrypt(E) and decrypt(D) functions five times with five different keys in the form EDEEE. The decryption process is accomplished in the reverse direction by invoking the encrypt and decrypt functions in the form DDDED. The keys for the quintuple encryption/decryption processes are generated by using a Tent map. The chaotic values for the encrypt/decrypt operations are generated by using a Gumowski-Mira map. The encrypt function E is composed of three stages: permutation, pixel value rotation and diffusion. The permutation stage scrambles all the rows and columns to chaotically generated positions. This stage reduces the correlation radically among the neighboring pixels. The pixel value rotation stage circularly rotates all the pixels either left or right, and the amount of rotation is based on chaotic values. The last stage performs the diffusion four times by scanning the image in four different directions: Horizontally, Vertically, Principal diagonally and Secondary diagonally. Each of the four diffusion steps performs the diffusion in two directions (forward and backward) with two previously diffused pixels and two chaotic values. This stage ensures the resistance against the differential attacks. The security and performance of the proposed method is investigated thoroughly by using key space, statistical, differential, entropy and performance analysis. The experimental results confirm that the proposed scheme is computationally fast with security intact.

Key words*: Quintuple Encryption, Gumowski-Mira map, Tent Map, Statistical Attacks, Differential Attacks.*

1. INTRODUCTION

Ensuring security to the multimedia information is a crucial issue. Encryption is one of the ways to protect the information, which disguises the information to hide its substance, so that only authorized persons with legitimate key can use the information. A wide variety of conventional encryption schemes such as DES, BLOWFISH, AES, CAST etc. are offered in the literature. However, they are computationally exhaustive and are not appropriate for image encryption [1]-[8]. In addition, the intrinsic features of images such as the strong correlation among neighboring pixels, high redundancy and bulk amount of data prevent the usage of the traditional text schemes for images [1]-[5].

Chaotic map based cryptosystems typically possess high speed with low cost, which makes them better contender than the traditional cryptosystems for multimedia encryption [1]-[5]. Moreover, the chaotic properties such as: sensitive dependence on initial conditions, ergodicity and random-like behaviors [1]-[10], are extremely useful in cryptographic permutation and diffusion processes. Hence, chaotic map based cryptosystems are receiving growing research interest from cryptographers.

Recently, quite a lot of chaotic based image cryptosystems have been proposed; nevertheless, the majority of them have their own strengths and weaknesses in terms of security and performance [2]. This paper proposes an efficient image encryption scheme based on quintuple encryption with two chaotic maps. The proposed method is defiant to brute force, statistical, differential and entropy based attacks and is computationally fast.

The novelty of this paper is listed hereafter.

i) The quintuple encryption is explored for images with five different keys.
ii) The Gumowski-Mira map is explored for image encryption.
iii) The encryption operations are carried with new methods.
iv) The proposed image encryption/decryption scheme is computationally fast.

The remaining of the paper is structured as follows. In section 2, the literature survey is presented. The 1D Tent map and 2D Gumowski-Mira map are discussed in section 3. In section 4, the proposed encryption scheme is described in detail. Experimental results and security analysis are presented in section 5 to show the effectiveness and validity of the proposed algorithm. Final section concludes the paper.

* *Corresponding author, Email: gs_hanchinamani@bvb.edu*
Manuscript received Feb. 09, 2015; revised Nov. 10, 2015; accepted Nov. 18, 2015

2. LITERATURE SURVEY

Chaotic based image cryptosystems usually consists of iteration of two processes: permutation and diffusion [1]-[5]. The permutation stage is employed to attain the decorrelation of the adjacent pixels and the diffusion stage is employed to spread the effect of individual pixels across the entire image [1]-[5]. However, as many rounds of the permutation and diffusion or iterations should be taken, the overall encryption speed is slow. A concise overview of the recently proposed chaotic based encryption schemes is given hereafter.

In [1], an image encryption scheme based on a generalized Arnold cat map is presented. The typical permutation diffusion architecture is employed. Two generalized Arnold cat maps are utilized to generate the chaotic sequences. The scheme has three parts: circular permutation, positive diffusion and opposite diffusion. The authors of [2], proposed a digital image encryption scheme, by employing multiple 1D chaotic map and the cryptographic primitive operations. The chaotic maps used are: Logistic map, Cubic map, Sin map and Tent map. The algorithm consists of three components: Key expansion, encryption and decryption. The scheme encrypts 256-bit of the plain image to 256-bit of the cipher image within eight 32-bit registers. The encryption scheme consists of eight w-bit registers $R_i(i = 1,2, \dots,8)$, which contain the initial input plain image as well as the output cipher image at the end of the encryption process. The encryption is performed by using a *xor* and pixel value circular rotations. The authors of [3] employed multiple chaotic maps to increase the key space to resist the brute force attacks. In [4], an encryption scheme based on coupling of chaotic maps and a *xor* operator is presented. The method disturbs the inner binary structure of the image and progressively brings the randomness characteristics. The encryption process is straightly realized on the bits of the plain image. The image is shuffled by using a chaotic function based on linear congruence. Nidhi Taneja [5], proposed a structure of combinational domain encryption that encrypts important data in spatial domain and the irrelevant data in wavelet domain. The proposed technique considers the varying data significance and support dissimilar security level to regions of different significance. The scheme makes use of Prewitt edge detector, discrete wavelet transform, Arnold cat map and Logistic map.

In [6], an image encryption scheme based on an intertwining chaotic map is proposed to increase the key space and the security. The substitution process uses six arbitrarily chosen odd integers to permute and then *xor*ed with the first chaotic key to shuffle and change the image pixel values. The diffusion stage alters the pixel values successively with various operations which include non linear diffusion using the first chaotic value, subdiagonal diffusion of neighboring pixels and *xor*ing with the third chaotic value. In [7], an image encryption scheme by using a chaotic Chebyshev generator is proposed. The scheme employs multiple permutations and a diffusion step. The permutation is based on row and column scrambling. The diffusion is based on addition and *xor* operations. In [8], an image encryption algorithm based on self adaptive wave transmission is proposed. The self adaptive encryption is implemented by using one half of the image to encrypt the other half of the image reciprocally. In [9], a symmetric image encryption scheme based on a 3D chaotic cat map is presented. The scheme uses a 3D cat map to shuffle the positions of the image pixels and uses another chaotic map to confuse the connection between the encrypted image and the plain image. In [10], a stream encryption scheme with avalanche effect is presented. A pseudo random number generator is used to create a *d-bit* segment binary key stream. In [11], a symmetric image encryption scheme using a circle map is proposed. The scheme uses an *xor* and image rotation operations to confuse the pixel value and to shuffle the pixel position. In [12], a hierarchy of *2D* piecewise nonlinear chaotic maps with an invariant measure is introduced. In [13], chaos based image encryption is merged with pixel bit. The scheme uses single chaotic system, which is applied directly to the position scrambling encryption; moreover, it also performs the gray encryption at the same time. In [14], a common framework of guidelines for image cryptosystems is provided, and it addresses three issues: implementation, key management and security analysis. The authors of [15], proposed an image cipher based on substitution diffusion architecture by using Standard and Logistic chaotic maps. In [16], an image encryption scheme based on two Logistic maps and an external 80-bit key is proposed. The initial conditions for the Logistic maps are derived by using external keys. The encryption is performed using eight different operations.

Though, there exist several image encryption schemes in the literature, each of them have their own strengths and limitations more or less in terms of security level and computational performance. Majority of the schemes produce satisfactory security results but lack in the computational speed. This paper proposes a novel chaotic image encryption scheme based on quintuple encryption with two chaotic maps. The proposed method is computationally fast over other schemes in the literature and also provides satisfactory security level.

3. CHAOTIC MAPS

Chaotic maps are nonlinear maps that exhibit chaotic behavior and possess the following properties: non-periodicity, non-convergent, sensitive to initial conditions and parameters, and topologically mixing [1]-[5]. The proposed image encryption scheme uses two chaotic maps: 1D Tent map and 2D Gumowski-Mira map and are discussed hereafter.

The 1D Tent map is a discrete-time dynamical system, and is defined as,

$$Z_{i+1} = \begin{cases} \mu Z_i & if \;\; Z_i < 0.5 \\ \mu(1 - Z_i) & if \;\; 0.5 \leq Z_i \end{cases} \qquad (1)$$

where Z_i is the current chaotic value, Z_{i+1} is the next chaotic value and μ is a control parameter. The key set of the Tent map is $\{Z_0, \mu\}$. In the proposed scheme the Tent map is employed to determine the initial keys for the Gumowski-Mira map.

The 2D Gumowski-Mira map is a discrete-time dynamical system, and is defined as,

$$X_{i+1} = Y_i + a\left(1 - b\, Y_i^2\right) Y_i\, f(X_i) \qquad (2)$$

$$Y_{i+1} = -X_i + f(X_{i+1}) \qquad (3)$$
$$f(X) = \mu X + \frac{2(1-\mu)X^2}{1+X^2}$$

where X_i, Y_i are the current chaotic values, X_{i+1} , Y_{i+1} are the next chaotic values, a, b are the control parameters and μ is a constant. The key set of the Gumoswki-Mira map is $\{X_0, Y_0, a, b\}$. In the proposed scheme, the Gumowski-Mira chaotic values are used during the permutation, pixel value rotation and diffusion stages of the encrypt and decrypt functions.

The propositions of the chaotic maps [13] are given in Eq.(4-6). The chaotic outputs of the above two maps are analyzed by computing the mean and self-correlations according the propositions given in Eq.(4-6). It is observed that the mean values are close to 0.5 and the self correlations within the sequence and across the two sequences are very close to 0.

Proposition 1. The mean value of the chaotic sequence is determined as,

$$x_{mean} = \lim_{N\to\infty} \frac{1}{N} \sum_{k=0}^{N-1} x_k = 0.5 \qquad (4)$$

Proposition 2. Self-correlation of a chaotic sequence is calculated as,

$$S1(\beta) = \lim_{N\to\infty} \frac{1}{N} \sum_{k=0}^{N-1} (x_k - x_{mean})(x_{k+\beta} - x_{mean}) = 0 \quad (5)$$

Proposition 3. Correlation function between two chaotic sequences is given as,

$$S2(\beta) = \lim_{N\to\infty} \frac{1}{N} \sum_{k=0}^{N-1} (x_k - x_{mean})(y_k - y_{mean}) = 0 \qquad (6)$$

4. PROPOSED ENCRYPTION SCHEME

The proposed scheme is based on quintuple encryption. The reason being; it is stronger than the triple and quadruple encryptions against the man-in-the-middle attack [17]. The proposed quintuple encryption operates on a plain image and then on a modified plain image five times with five different keys. The sender first encrypts with the first key, then decrypt with the second key, then encrypt with the third key, then encrypt with the fourth key and finally encrypt with the fifth key according to Eq.(7). This is called as encrypt-decrypt-encrypt-encrypt-encrypt ($EDEEE$) mode. The receiver decrypts with the fifth key, then decrypts with the fourth key, then decrypts with third key, then encrypts with the second key and finally decrypts with the first key according to Eq.(8).

$$C = E_{K_5}\left(E_{K_4}\left(E_{K_3}\left(D_{K_2}\left(E_{K_1}(P)\right)\right)\right)\right) \qquad (7)$$

$$P = D_{K_1}\left(E_{K_2}\left(D_{K_3}\left(D_{K_4}\left(D_{K_5}(C)\right)\right)\right)\right) \qquad (8)$$

where P is a plain image, C is a cipher image, E is an encrypt function, D is a decrypt function and K_1, K_2, K_3, K_4, K_5 are the keys at the respective invocations.

The proposed encryption scheme consists of three functionalities: i) Key generation and scheming ii) Encrypt function iii) Decrypt function, and these are discussed hereafter.

4.1 Key Generation and Scheming

Let K_1, K_2, K_3, K_4, K_5 are the keys used at the respective call to the encrypt/decrypt functions. And these keys are used as initial values to generate the Gumowski-Mira chaotic sequence. The keys can be described as,

$$\begin{aligned} K_1 &= \{ X_1 , Y_1 , a_1, b_1 \} \\ K_2 &= \{ X_2 , Y_2 , a_2, b_2 \} \\ K_3 &= \{ X_3 , Y_3 , a_3, b_3 \} \\ K_4 &= \{ X_4 , Y_4 , a_4, b_4 \} \\ K_5 &= \{ X_5 , Y_5 , a_5, b_5 \} \end{aligned} \qquad (9)$$

where X_i, Y_i are the initial values and a_i, b_i are the parameters of the Gumowski-Mira map. So, the key comprises of totally 20 values $\{ X_i, Y_i , a_i, b_i , i = 1 \; to \; 5\}$. It is difficult to remember the above key set. Hence the following key scheming approach is used to limit the key length. Generate eight chaotic values (Z_1 to Z_8) using a Tent map according to Eq.(1), then these chaotic values are assigned to $\{X_2 , Y_2 , X_3 , Y_3 , X_4, Y_4, X_5, Y_5\}$. The a and b parameters are kept the same for all the five invocations of the encrypt/decrypt functions. And X_1, Y_1 are taken as parameters to increase the key space.

So the modified keys can be described as,

$$\begin{aligned} K_1 &= \{X_1 = \quad , Y_1 = \quad , a, b\} \\ K_2 &= \{X_2 = Z_1, Y_2 = Z_2 , a, b\} \\ K_3 &= \{X_3 = Z_3, Y_3 = Z_4 , a, b\} \\ K_4 &= \{X_4 = Z_5, Y_4 = Z_6 , a, b\} \\ K_5 &= \{X_5 = Z_7, Y_5 = Z_8 , a, b\} \end{aligned} \qquad (10)$$

So the key set of the quintuple encryption/decryption process is,

$$key = \{Z_0, \mu, X_1, Y_1, a, b\} \qquad (11)$$

where $\{ Z_0 , \mu\}$ are the parameters of the Tent map, and $\{X_1, Y_1, a, b\}$ are the parameters of the Gumowski-Mira map.

4.2 Encrypt Function

The encrypt function is composed of three stages, i.e. permutation, pixel value rotation and diffusion and are discussed below.

4.2.1 Permutation

The permutation is employed to decorrelate the neighboring pixels. Let I be a gray original plain image of size $M \times N$, it is a digital matrix containing M rows and N columns,

and the gray values ranges from 0 to 255. In the process of permutation, initially $M+N$ Gumowski-Mira chaotic values $\{(X_1,\dots,X_M),(Y_1,\dots,Y_N)\}$ are generated by using Eq.(2,3), after doing iterations in chaos maps. Let $PM=\{X_1,\dots,X_M\}$ and $PN=\{Y_1,\dots,Y_N\}$. Then PM and PN are sorted, and the positions of the sorted chaotic values in the original chaotic sequence are found and are stored in PM' and PN'. The next step is to shuffle the row position of all the values from the first column to the last column according to $PM'_1,\dots,PM'_M$. Likewise shuffle the column position of all the values from the first row to the last row according to $PN'_1,\dots,PN'_N$. This stage shuffles all the pixels and decorrelates the neighboring pixels.

4.2.2 Pixel Value Rotation

Circular rotation is applied on pixel by pixel basis by scanning a block of eight pixels at a time and continued for the whole image. The amount of rotation is based on chaotic values and is computed as given below.

Step 1 Generate eight chaotic values by using a Gumowski-Mira map.
Suppose the chaotic values are,
0.6, 0.7, 0.2, 0.3, 0.8, 0.4, 0.1, 0.5

Step 2 Sort the eight chaotic values.
Sorted values: 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8

Step 3 Find the positions of the sorted chaotic values in the original sequence, and these eight values are used as the amount of rotation for a block of eight pixels.
Positions: 7, 3, 4, 6, 8, 1, 2, 5

Step 4 Replace 8 by any of the randomly chosen number in the range 1 to 7, since rotation by 8 leads to the same value.
Positions: 7, 3, 4, 6, 5, 1, 2, 5

Step 5 Scan a block of eight pixels (P_1 to P_8), and perform the left or right circular rotations as follows,

$$\begin{aligned} P_1 &= P_1 \ll 7 \\ P_2 &= P_2 \gg 3 \\ P_3 &= P_3 \ll 4 \\ P_4 &= P_4 \gg 6 \\ P_5 &= P_5 \ll 5 \\ P_6 &= P_6 \gg 1 \\ P_7 &= P_7 \ll 2 \\ P_8 &= P_8 \gg 5 \end{aligned} \qquad (12)$$

where << is circular left rotation, >> is circular right rotation, P_1 to P_8 are the eight consecutive pixels of a block scanned, and 1 to 7 are the amount of rotations.

4.2.3 Diffusion

The diffusion process is used to ensure the plain image sensitivity i.e., a 1-bit change in any one pixel of the plain image should spread out to almost all pixels in the image. Moreover, to increase the resistance against the differential attack, the proposed scheme applies the diffusion four times by scanning the image in four different directions as given below and shown in Fig.1.

Step 1 Scan the image left to right and top to bottom, and then apply forward and backward diffusions.

Step 2 Scan the image top to bottom and left to right, and then apply forward and backward diffusions

Step 3 Scan the image in principal diagonal direction and then apply forward and backward diffusions.

Step 4 Scan the image in secondary diagonal direction and then apply forward and backward diffusions.

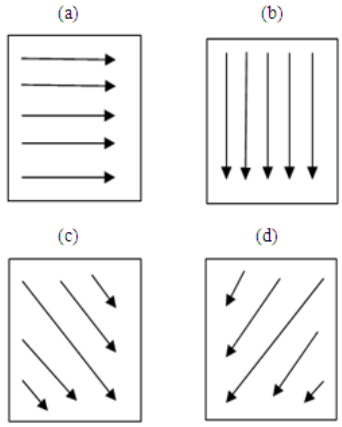

Fig. 1. Scanning directions (a) horizontal (b) vertical (c) principal diagonal (d) secondary diagonal

In each of the above four steps, the forward and backward diffusions are carried as discussed below.

The 2D image is transformed to 1D array $P_{1\times MN}$ by scanning the image in the directions as shown in Step 1-4. The diffusion process is performed in two directions (forward and backward) with two previously diffused pixels and two chaotic values of the Gumowski-Mira map $\{X,Y\}$. The computed encrypted pixel values depend on the previously encrypted pixels and chaotic sequences; hence the algorithm shows more resistance against the differential attacks.

Initially $M\times N$ chaotic values $\{(X_1,\dots,X_{M\times N}),(Y_1,\dots,Y_{M\times N})\}$ are generated by using Eq.(2,3) after doing iterations in chaos maps. The real chaotic sequences are transformed to the integer form by using Eq.(13).

$$Z'_i = (Z_i * 10^8)\ mod\ m \qquad (13)$$

where Z_i is the real chaotic value, Z'_i is the transformed integer value and m is 256 for the gray scale images.

The forward diffusion is performed by using the following equation,

$$E_i = \big((((P_i + E_{i-2})mod\ 256) + E_{i-1})mod\ 256 \oplus X_i\big) + Y_i)mod\ 256, i = 1,2,\dots,MN \qquad (14)$$

where $+$ indicates the modulo addition, $\oplus$ is the bitwise XOR, E_i is the current pixel, E_{i-1} and E_{i-2} are the previously encrypted pixels, P_i is the permuted pixels, X_i and Y_i are the 2D Gumowski-Mira chaotic values. E_{-1} and E_0 can be considered as constants.

The backward diffusion is performed by using the following equation to make the influence of every pixel equal.

$$F_i = \left((((E_i + F_{i+2}) \bmod 256) + F_{i+1}) \bmod 256 \oplus X_i\right) + Y_i) \bmod 256, i = MN, \dots, 1 \quad (15)$$

where F_i is the current pixel, F_{i+1} and F_{i+2} are the previously encrypted pixels, E_i is the forward diffused image pixels, X_i and Y_i are the 2D Gumowski-Mira chaotic values and E_{MN+1} and E_{MN+2} can be considered as constants. Finally, the encrypted image is obtained after the four diffusion steps.

4.3 Decrypt Function

Decryption involves the reconstruction of the gray levels of the original image from the encrypted image. It is an inverse process of the proposed encrypts function. Initially the diffusion process is carried out in the sequence: secondary diagonally, principal diagonally, vertically, and horizontally. Then circular rotation is applied in inverse rotate direction. Lastly the column scrambling is performed followed by the row scrambling.

4.4 Encryption Algorithm

The encryption algorithm uses following three functions: Quintuple encryption process, Encrypt function and Decrypt function.

4.4.1 Quintuple Encryption Process

Step 1 Read the original image and store the pixel values in the matrix $I_{M\times N}$.

Step 2 Initialize the Tent map and generate eight chaotic values (Z_1 to Z_8) by using Eq.(1).

Step 3 Initialize five Keys K_1, K_2, K_3, K_4, K_5 by using Eq.(10).

Step 4 Invoke the Encrypt function with K_1 for original image.

Step 5 Invoke the Decrypt function with K_2 for modified image.

Step 6 Invoke the Encrypt function with K_3 for modified image

Step 7 Invoke the Encrypt function with K_4 for modified image.

Step 8 Invoke the Encrypt function with K_5 for modified image.

4.4.2 Encrypt Function

The encrypt function is composed of following steps.

Step 1 Get the image.

Step 2 Generate M chaotic values of the X_i sequence $(X_1, \dots, X_M)$, and N chaotic values of the Y_i sequence $(Y_1, \dots, Y_N)$ by using Eq.(2,3).

Step 3 Copy the X_i chaotic values to PM and Y_i chaotic values to PN.

Step 4 Sort PM and PN, find the position of the sorted chaotic values in the original chaotic sequence and store in PM' and PN'.

Step 5 Scramble all the rows by using PM'.

Step 6 Scramble all the columns by using PN'.

Step 7 Generate eight chaotic values by using Eq.(2,3) for the circular rotations.

Step 8 Sort the chaotic values and find their position in the original chaotic sequence, and replace 8 by a randomly chosen number in the range 1 to 7.

Step 9 Scan a block of eight pixels at a time from the image and apply left or right circular rotations as given in Eq.(12), Repeat this for the entire image.

Step 10 Generate $M \times N$ chaotic values $\{(X_1, \dots, X_{M\times N}), (Y_1, \dots, Y_{M\times N})\}$ by using Eq.(2,3) for the diffusion process.

Step 11 Transform the real chaotic sequence to integer sequence by using Eq.(13).

Step 12 Scan the image in horizontal direction, and then apply forward and backward diffusions by using Eq.(14,15).

Step 13 Scan the image in vertical direction, and then apply forward and backward diffusions by using Eq.(14,15).

Step 14 Scan the image in principal diagonal direction, and then apply forward and backward diffusions by using Eq.(14,15).

Step 15 Scan the image in secondary diagonal direction, and then apply forward and backward diffusions by using Eq.(14,15).

4.4.3 Decrypt Function

The decrypt function is composed of following steps.

Step 1 Get the image.

Step 2 Generate $M \times N$ chaotic values $\{(X_1, \dots, X_{M\times N}), (Y_1, \dots, Y_{M\times N})\}$ by using Eq.(2,3) for the reverse diffusions.

Step 3 Transform the real chaotic values to integer form by using Eq.(13).

Step 4 Scan the image in secondary diagonal direction, and then apply reverse forward and backward diffusions.

Step 5 Scan the image in principal diagonal direction, and then apply reverse forward and backward diffusions.

Step 6 Scan the image in vertical direction, and then apply reverse forward and backward diffusions.

Step 7 Scan the image in horizontal direction, and then apply reverse forward and backward diffusions.

Step 8 Generate eight chaotic values by using Eq.(2,3) for reverse circular rotations.

Step 9 Sort the chaotic values and find their position in the original chaotic sequence and replace 8 by the same number as in the encrypt function.

Step 10 Scan a block of eight pixels at a time from the image and apply reverse left or right circular rotations with rotation amount of (8 - original rotation) as given in Eq.(12). Repeat this for the entire image.

Step 11 Generate M chaotic values of the X_i sequence $(X_1, \dots, X_M)$, and N chaotic values of the Y_i sequence $(Y_1, \dots, Y_N)$ by using Eq.(2,3).

Step 12 Copy X_i chaotic values to PM and Y_i chaotic values to PN.

Step 13 Sort PM and PN, find the position of the sorted chaotic values in the original chaotic

sequence and store in PM' and PN'.

Step 14 Reverse scramble all the columns by using PN'.

Step 15 Reverse scramble all the rows by using PM'.

4.5 Decryption Algorithm

The decryption algorithm uses the following three functions: Quintuple decryption process, Encrypt function and Decrypt function.

4.5.1 Quintuple Decryption Process

Step 1 Read the encrypted image and store the pixel values in the matrix $I_{M\times N}$.

Step 2 Initialize the Tent map and generate eight chaotic values (Z_1 to Z_8) by using Eq.(1).

Step 3 Initialize five Keys K_1, K_2, K_3, K_4, K_5 by using Eq.(10).

Step 4 Invoke the Decrypt function with K_5 for encrypted image.

Step 5 Invoke the Decrypt function with K_4 for modified image.

Step 6 Invoke the Decrypt function with K_3 for modified image.

Step 7 Invoke the Encrypt function with K_2 for modified image.

Step 8 Invoke the Decrypt function with K_1 for modified image.

4.5.2 Encrypt and Decrypt Functions

The encrypt and decrypt functions for decryption process are the same as given in the encryption process.

5. EXPERIMENTS AND SECURITY ANALYSIS

An image encryption scheme is anticipated to resist the various attacks such as the brute-force attacks, statistical attacks, differential attacks and entropy based attacks [1]-[9]. This section analyzes the properties of the proposed encryption algorithm to show its effectiveness in resisting these attacks.

5.1 Experimental Setup

The proposed algorithm is implemented by using C programming language under the Linux platform with a personal computer configured with intel (R) Core(TM) i3-2120 CPU at 3.30 GHz with 2.91 GB of RAM. According to Eq.(11) the initial parameters of the quintuple encryption process is randomly set to $\{Z_0 = 0.3,\ \mu = 3.81, X_1 = 0.3, Y_1 = 0.6, a = 0.4, b = 0.6\}$. Where $\{Z_0, \mu\}$ are the parameters of the Tent map and $\{X_1, Y_1, a, b\}$ are the parameters of the Gumowski-Mira map. The test images are chosen from the *USC-SIPI* image database (sipi.usc.edu/database/) and are gray-scale images of size 256×256.

5.2 Visual Assessment

The proposed encryption scheme has been tested with a number of test images of differing content. Fig.2 shows the visual assessment of the original, encrypted and decrypted images for four different images. The first row shows the original images, second row shows the encrypted images and the final row shows the decrypted images. The encrypted images are unintelligible, non-recognizable in appearance, random and noise-like images without any leakage of the original information. The decrypted images are exactly alike to the original images.

5.3 Key Space Analysis

Brute-force attack is an attack model, where an intruder tries to break the cryptosystem by trying each and every possible key [1]-[9]. It can be resisted by increasing the key space. The proposed encryption scheme uses two chaotic maps. The key set of the Tent map is $\{Z_0, \mu\}$ and for Gumowski-Mira map is $\{X_1, Y_1, a, b\}$. So the key set is $\{Z_0, \mu, X_1, Y_1, a, b\}$. With 64 bits for each parameter and there are six real values, so the key-length is 384 bits and the key space is 2^{384}. Hence the proposed algorithm has adequate key space and is resistant to brute-force attacks. Table 1 shows the key space of the proposed scheme and some of the other methods in the literature. It can be seen that the proposed scheme has competitive and adequate key space of 2^{384}.

Table 1. Key space of the proposed method and some of the other methods in the literature

Method	Key space
Proposed algorithm	2^{384}
Ref.[1]	Not specified
Ref.[2]	2^{349}
Ref.[3]	2^{400}
Ref.[4]	2^{128}
Ref.[5]	2^{135}
Ref.[6]	2^{216}
Ref.[7]	10^{56}
Ref.[8]	2^{128}
Ref.[9]	2^{128}

5.4 Statistical Analysis

Statistical attacks are based on the assumptions about the distribution of the pixel intensity values and the correlations of the adjacent pixels. Statistical analysis can be performed by plotting the histograms and by computing the correlations as discussed below.

5.4.1 Histogram Analysis

An image histogram plots the frequency of each gray intensity level. The histogram of an encrypted image is anticipated to be sufficiently uniform to prevent the outflow of information [1-9]. The histogram of a number of plain images are plotted and assessed. The histograms of the original and encrypted images for four images are shown in Fig.3. The histograms of the encrypted images are uniformly spread and are entirely different from that of the original image, and bear no statistical similarity with the original images. Thus, the proposed scheme is defiant to histogram based statistical attacks. The histograms are almost uniformly spread for most of the schemes in the literature.

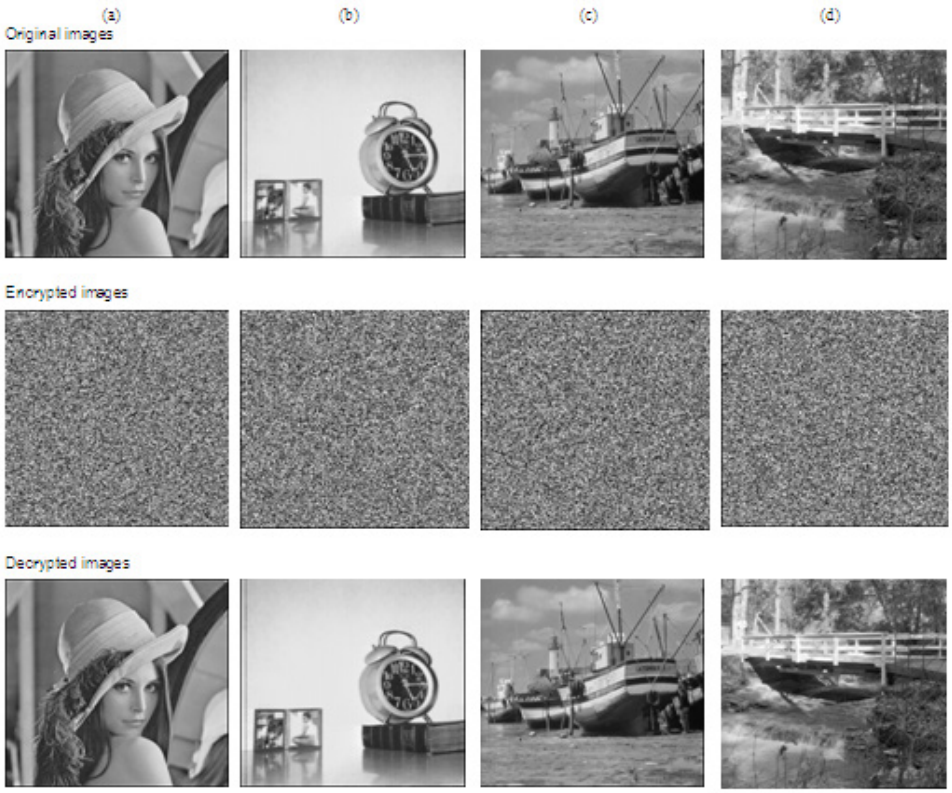

Fig. 2. Original images, encrypted images and decrypted images with the proposed algorithm (a) Lena (b) Clock (c) Boat (d) Bridge

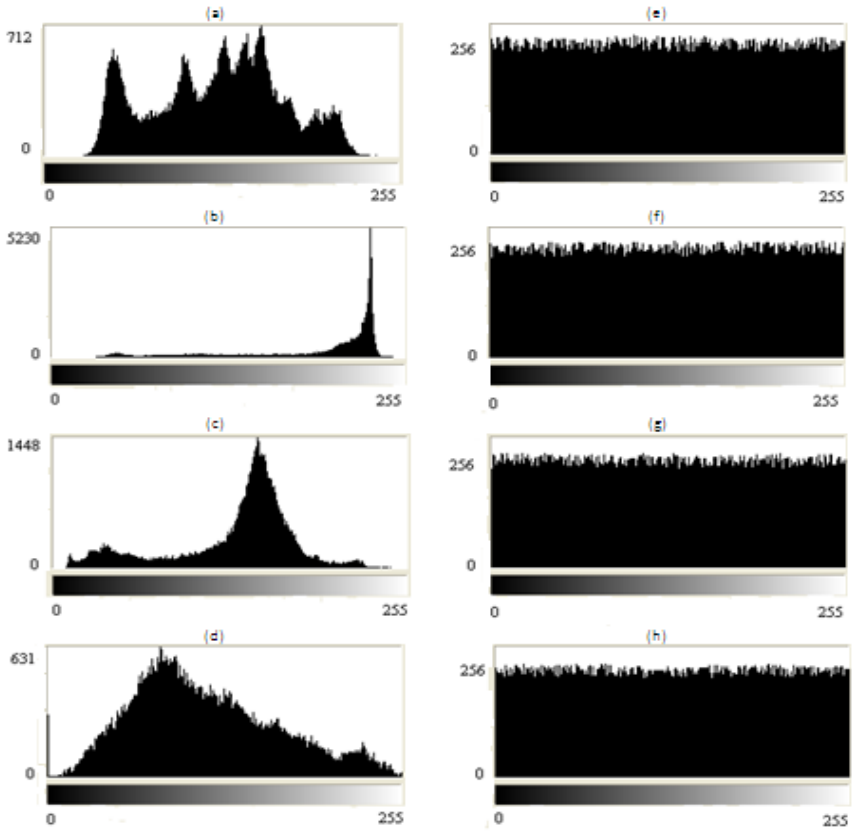

Fig. 3. Histograms of the original and encrypted images (a-d) Histograms of the original images Lena, Clock, Boat, Bridge (e-f) Histograms of the respective encrypted images

5.4.2 Correlation Analysis

Generally, for any original plain image having specific visual content, each pixel is strongly correlated with its neighboring pixels in all the three directions: horizontal, vertical and diagonal. Hence, image encryption algorithms are anticipated to generate the encrypted images with no such correlations among the neighboring pixels [1]-[9]. The correlation coefficient of the adjacent pixels is computed according to Eq.(16-19).

$$E(x) = \frac{1}{N}\sum_{i=1}^{N} x_i \quad (16)$$

$$D(x) = \frac{1}{N}\sum_{i=1}^{N} (x_i - E(x))^2 \quad (17)$$

$$cov(x,y) = \frac{1}{N}\sum_{i=1}^{N} \big(x_i - E(x)\big)\big(y_i - E(y)\big) \quad (18)$$

$$r_{xy} = \frac{cov(x,y)}{\sqrt{D(x)}\sqrt{D(y)}} \quad (19)$$

where x and y are the adjacent pixels of the original or encrypted images, $E(x)$ is the mean value, $D(x)$ is the deviation with respect to the mean, $cov(x,y)$ is the covariance between the adjacent pixels, and r_{xy} is the correlation coefficient. To assess the correlation in the original and encrypted images, 4096 pairs of adjacent pixels are arbitrarily chosen in horizontal, vertical and diagonal directions, and their correlation coefficients are calculated by using Eq.(19). Table 2 shows the computed correlation coefficients of the original and encrypted images for different images. From Table 2 it is observed that the two adjacent pixels in the original image are highly correlated to each other, whereas the correlation coefficients for the encrypted images are extremely close to zero. Thus the proposed scheme is defiant to the correlation based statistical attacks. The comparison of the correlation results with other schemes are given in Table 3. It can be seen that correlation results of the proposed scheme is extremely close to zero and comparable to the other schemes in the literature.

5.5 Sensitivity Tests

The differential attacks are based on how the differences in input can influence the resultant difference at the output. An image encryption scheme is anticipated to be sensitive to both the keys and the plain image and these are discussed below.

5.5.1 Key Sensitivity Analysis

Key sensitivity implies that the a small change in the secret key should produce entirely different encrypted image [1-5]. The key sensitivity assessment is demonstrated with the following approach.

Step 1 The original image is encrypted using a test key K_1 to produce the cipher image C_1.

Step 2 The original image is encrypted again with a very small change in the test key K_1, i.e. K_2 to produce cipher image C_2.

Step 3 The two cipher images C_1 and C_2 with slight different keys are compared pixel by pixel to observe the number of differing pixels.

Table 2. Correlation coefficients of the adjacent pixels in different directions for the original and encrypted images

Image	Correlation coefficients for encrypted images			Correlation coefficients for original images		
	Horizontal	Vertical	Diagonal	Horizontal	Vertical	Diagonal
Lena	-0.001906	0.005104	0.003114	0.968683	0.943269	0.933408
Clock	0.003761	-0.009090	-0.001476	0.973337	0.958875	0.933373
Boat	-0.009919	-0.001658	-0.000514	0.938983	0.918620	0.881951
Bridge	-0.003673	0.006024	-0.003264	0.899550	0.932778	0.868487
Tank	0.010287	-0.000518	0.001194	0.915934	0.962062	0.902080
House	-0.001778	-0.003027	0.001664	0.925962	0.902565	0.866506
Truck	-0.008543	0.009809	-0.002927	0.878351	0.939607	0.851138
Couple	-0.007798	-0.004387	0.002106	0.858304	0.918562	0.815066

Table 3. Comparison of the correlation coefficients of the proposed scheme with other methods

Method	Correlation coefficients		
	Horizontal	Vertical	Diagonal
Proposed algorithm	-0.001906	0.005104	0.003114
Ref.[1]	0.077000	-0.072360	-0.061530
Ref.[2]	0.003200	0.002424	0.001520
Ref.[3]	0.006300	0.005900	0.007300
Ref.[4]	-0.001700	0.005900	-0.001900
Ref.[5]	-0.044500	0.004100	Not specified
Ref.[6]	0.004800	0.002700	0.003600
Ref.[7]	-0.097360	-0.070680	0.048440
Ref.[8]	0.011830	0.001600	0.014800
Ref.[9]	0.012700	-0.019000	-0.001200

The key sensitivity is assessed by using $NPCR$ and $UACI$ parameters as discussed below.

$NPCR$ (Number of Pixels Change Rate) is used to compute the percentage of the total number of differing pixels in two images and is computed by using Eq.(20).

$$NPCR = \frac{\sum_{i,j} D(i,j)}{M \times N} \times 100\% \qquad (20)$$

$$D(i,j) = \begin{cases} 1, & if\ C_1(i,j) \neq C_2(i,j) \\ 0 & ,Otherwise \end{cases} \qquad (21)$$

where C_1 and C_2 are two encrypted images with slightly different keys K_1 and K_2. $C_1(i,j)$ and $C_2(i,j)$ are the pixel values of C_1 and C_2 at position (i,j). D is a bipolar array with the same size as C_1 and C_2 and its contents are either 0 or 1 based on Eq.(21).

$UACI$ (Unified Average Changing Intensity) is used to compute the percentage of the average changing intensity difference between two encrypted images and is computed as,

$$UACI = \frac{1}{M \times N}\left[\sum_{ij} \frac{C_1(i,j) - C_2(i,j)}{255}\right] \times 100\% \qquad (22)$$

The key sensitivity is analyzed by testing one parameter at a time with a very minute change in the key. The proposed scheme has six parameters $\{Z_0, \mu, X_1, Y_1, a, b\}$. Table 4 lists the $NPCR$ and $UACI$ values for six different parameters. From Table 4 it is observed that the $NPCR$ and $UACI$ values are extremely close to their ideal values of 99.6% and 33.4%. Thus, the proposed scheme has high key sensitivity. The key sensitivity results in the literature are satisfactory and are demonstrated in the visual form hence are not shown here.

Table 4. Key sensitivity results for the Clock image with different parameters of the chaotic map

Parameter changed	$NPCR$ (%)	$UACI$ (%)
Z_0	99.610901	33.584370
μ	99.624634	33.570103
X_1	99.604797	33.454929
Y_1	99.623108	33.543621
a	99.586487	33.406181
b	99.612427	33.421890

Furthermore, the key sensitivity is also tested visually with the following scheme. The original key is altered with a little change and different keys are generated. The keys can be described as,

original key, $\boldsymbol{Key_1}$= (0.3, 3.81, 0.3, 0.6, 0.4, 0.6)
and the slightly modified keys,
$\boldsymbol{Key_2}$= (**0.30000000001**, 3.81, 0.3, 0.6, 0.4, 0.6),
$\boldsymbol{Key_3}$= (0.3, **3.81000000001**, 0.3, 0.6, 0.4, 0.6),
$\boldsymbol{Key_4}$= (0.3, 3.81, **0.30000000001**, 0.6, 0.4, 0.6),
$\boldsymbol{Key_5}$= (0.3, 3.81, 0.3, **0.60000000001**, 0.4, 0.6),
$\boldsymbol{Key_6}$= (0.3, 3.81, 0.3, 0.6, **0.40000000001**, 0.6),
$\boldsymbol{Key_7}$= (0.3, 3.81, 0.3, 0.6, 0.4, **0.6000000001**).

The encrypted images with the correct key $\{C_1\}$ and the slightly altered keys $\{C_2, C_3, C_4, C_5, C_6, C_7\}$ are given in Fig.4b-h. Although, all look alike, they are totally dissimilar from each other. This can be confirmed by finding the difference image between C_1 and other encrypted images $\{C_2, C_3, C_4, C_5, C_6, C_7\}$. Fig.4i-n shows the difference images $\{C_1 - C_2, C_1 - C_3, C_1 - C_4, C_1 - C_5, C_1 - C_6, C_1 - C_7\}$. From the difference images, it is observed that the majority of the pixels in the difference image are nonzero.

Moreover, the key sensitivity test is also demonstrated for the decryption process. The decryption is performed with the correct key and with slightly altered keys. The Fig.5a shows the decrypted image with the correct key $\boldsymbol{Key_1}$ and Fig.5b-g are the decrypted images with slightly altered keys $\boldsymbol{Key_2}, \boldsymbol{Key_3}, \boldsymbol{Key_4}, \boldsymbol{Key_5}, \boldsymbol{Key_6}, \boldsymbol{Key_7}$. From Fig.5 it is observed that the correct decryption cannot be achieved if there is a minute change in the key.

5.5.2 Plain Image Sensitivity Analysis

Plain image sensitivity means, a 1-bit change in the original plain image should cause a significant change in the encrypted image [1]-[9]. The plain image sensitivity test is demonstrated with the following scheme.

Step 1 Encrypt the original plain image to produce a cipher image C_1.

Step 2 Change one bit of the original plain image at any random location, and encrypt again to produce a cipher image C_2.

Step 3 The two cipher images C_1 and C_2 are compared pixel by pixel to examine the number of differing pixels.

The $NPCR$ and $UACI$ parameters are used to test the plain image sensitivity as given in Eq.(20-22). The $NPCR$ and $UACI$ values are computed for different randomly chosen locations by changing one bit at a time, and it is observed that these values are close to their ideal values of 99.6% and 33.4% irrespective of the pixel position selected. Table 5 lists the $NPCR$ and $UACI$ values for a randomly chosen position (60, 95) and the pixel intensity value is 156. The $NPCR$ and $UACI$ are calculated by changing one bit at a time from bit 0 to bit 7, the average $NPCR$ and $UACI$ values are found to be 99.608738% and 33.465564%. Hence the proposed scheme is extremely sensitive to the 1-bit alterations in the original plain image and is resistant to the differential attacks. Table 6 lists the plain image sensitivity results of the proposed scheme and the other schemes in the literature. The plain image sensitivity results of the proposed scheme are comparable to other schemes and are close the ideal values.

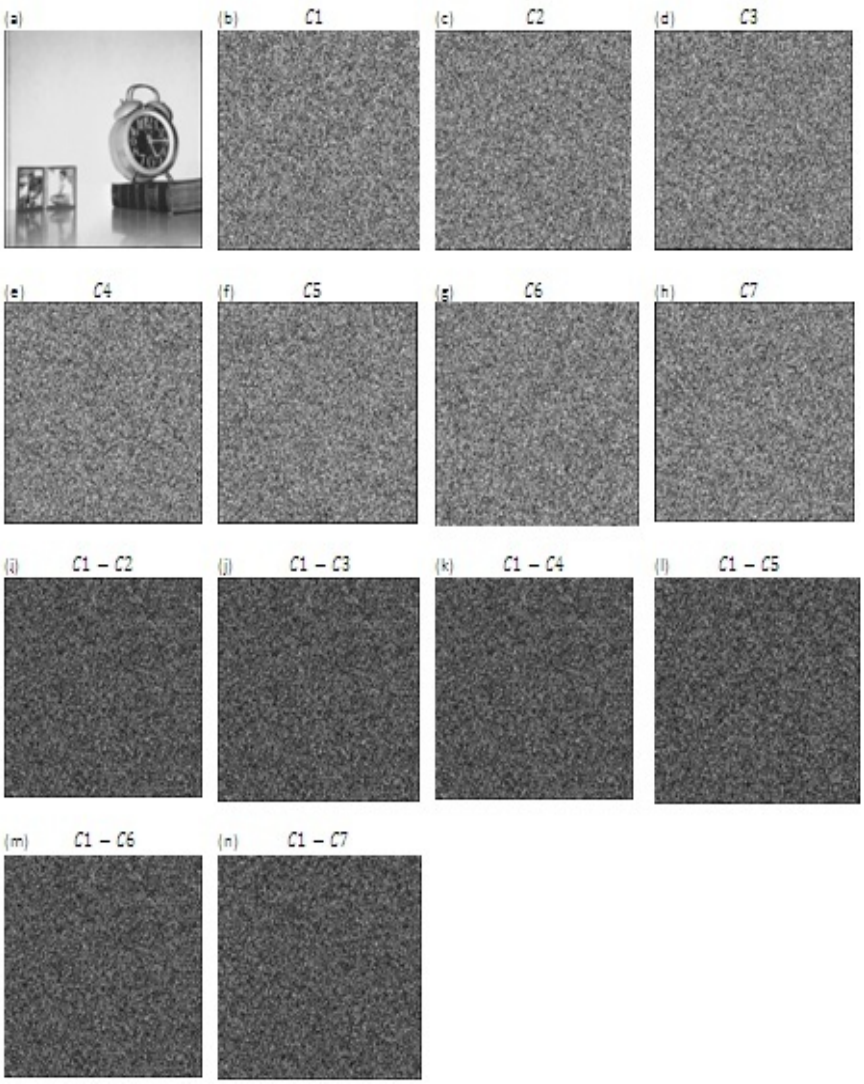

Fig. 4. Key sensitivity analysis for the encryption process for clock image (a) original image (b) encrypted image with correct key Key_1 (c-h) encrypted images with slightly different keys (i-n) difference image between C_1 and other incorrect encrypted images

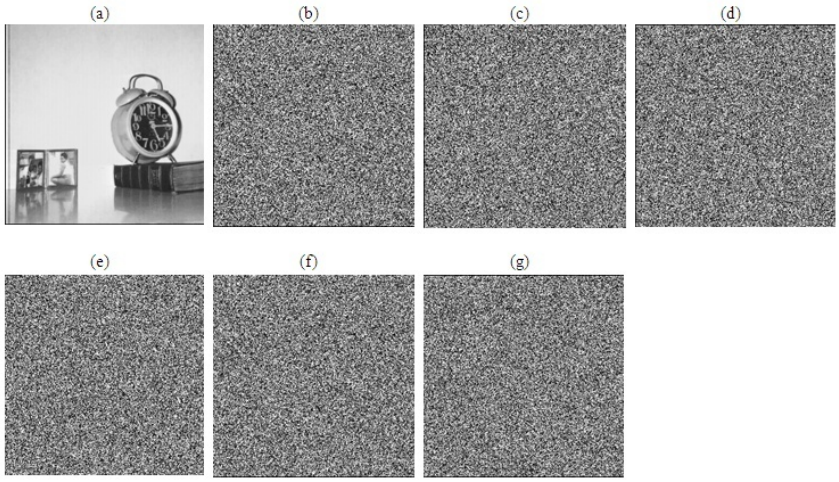

Fig. 5. Key sensitivity analysis for decryption process for Clock image (a) decryption with correct key (b-g) decryption with slightly changed keys

Table 5. Plain image sensitivity test

Bit Position	Changed Pixel values	$NPCR$ (%)	$UACI$ (%)
0	157	99.620056	33.491440
1	158	99.588013	33.344589
2	152	99.604797	33.369267
3	148	99.636841	33.545628
4	140	99.624634	33.386986
5	188	99.589702	33.467438
6	220	99.618530	33.552040
7	28	99.587332	33.567127
Average values		99.608738	33.465564

Table 6. Comparison of the plain image sensitivity results with other schemes in the literature

Method	Plain image sensitivity	
	NPCR %	*UACI %*
Proposed algorithm	99.608738	33.465564
Ref.[1]	99.576000	33.441000
Ref.[2]	99.610000	33.450000
Ref.[3]	99.609200	33.489100
Ref.[4]	99.610000	33.450000
Ref.[5]	60.00 – 94.00	15.00 – 38.00
Ref.[6]	99.562100	33.432400
Ref.[7]	99.233000	33.479000
Ref.[8]	50.200000	25.200000
Ref.[9]	99.650000	33.480000

5.6 Information Entropy Analysis

Information entropy is a measure of the amount of randomness in information content, and is defined as,

$$H(K) = \sum_{i=0}^{r-1} P(K_i)\, log_2 \frac{1}{P(K_i)} \qquad (23)$$

where K_i represents the pixel values, $P(K_i)$ is the probability of the symbol K_i and r is the number of symbols and is 256 for gray level image. Suppose the gray level image has 2^8 gray values with equal probabilities, $K = (K_0, K_1, K_2, \dots, K_{255})$, according to Eq.(23), we obtain its entropy value $H(K) = 8$. The entropy reaches the maximum ideal value of 8 when all the pixel values are randomly spread [1]-[5]. Table 7 shows the entropy values for the original plain images and the encrypted images. From the entropy results it is observed that the entropies of the encrypted images are extremely close to the ideal value of 8. The information outflow in the proposed encryption scheme is insignificant and is secure against the entropy based attacks. The comparison of the entropy outcome with the other schemes is listed in Table 8. The entropy outcome of the proposed scheme is close to the ideal value of 8 and is comparable to other schemes in the literature.

Table 7. Entropy values for the original and encrypted images for different images

Image	Entropy	
	original image	encrypted image
Lena	7.426985	7.997352
Clock	6.707120	7.997435
Boat	7.161232	7.997118
Bridge	7.669804	7.997503
Tank	6.920753	7.997265
House	7.212776	7.997278
Truck	7.057452	7.997301
Couple	7.145428	7.997145

Table 8. Comparison of the entropy values of the proposed scheme with other methods

Method	Entropy
Proposed algorithm	7.997352
Ref.[2]	7.996991
Ref.[3]	7.993000
Ref.[6]	7.994000
Ref.[1,4,5,7,8,9]	Not specified

5.7 Other Tests

5.7.1 Peak Signal to Noise Ratio (PSNR) Analysis

By considering the original plain image as a signal and the encrypted image as a noise [5], the objective assessment of the encryption scheme can performed by computing the $PSNR$. The $PSNR$ is computed as follows,

$$PSNR = 20 \times log_{10}\left(\frac{255}{\sqrt{MSE}}\right) dB \qquad (24)$$

where MSE is the mean square error and is determined according to Eq.(25).

$$MSE = \frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}(\,|I(i,j) - I'(i,j)|\,)^2 \qquad (25)$$

where $I(i,j)$ is the pixel value of the original plain image and $I'(i,j)$ is the pixel value of the encrypted image at position (i,j). The $PSNR$ values are computed for the different test images and are listed in Table 9. From Table 9 it is observed that the $PSNR$ values are lower, which indicates the complexity in getting the original plain image from the encrypted image for invaders. The comparison of the $PSNR$ outcome with the other schemes is listed in Table 10. The entropy outcome of the proposed scheme is lower and is comparable to other schemes in the literature.

Table 9. $PSNR$ values for different test images

Image	$PSNR$ (dB)
Lena	9.252720
Clock	7.273054
Boat	9.373762
Bridge	8.850304
Tank	9.875631
House	8.662225
Truck	9.665390
Couple	9.686495

Table 10. Comparison of the $PSNR$ values of the proposed scheme with other methods

Method	$PSNR$(dB)
Proposed algorithm	7.273054 – 9.686495
Ref.[5]	8.2462 – 9.7322
Ref.[1,2,3,4,6,7,8,9]	Not specified

5.7.2 Gray Value Degree (GVD) Analysis

The gray difference of a pixel with its four neighborhood pixels can be computed as [13],

$$G = \frac{\sum [I(m,n) - I(m',n')]^2}{4},$$

$$here\ (m',n') = \begin{cases} (m-1,n) \\ (m+1,n) \\ (m,n-1) \\ (m,n+1) \end{cases} \qquad (26)$$

where $I(m,n)$ represents the pixel value at location (m,n), and $I(m',n')$ denotes the pixel values of the four neighborhood pixels. The average neighborhood gray difference for the entire image can be calculated by Eq.(27).

$$W(\,G(m,n)\,) = \frac{\sum_{m=2}^{M-1}\sum_{n=2}^{N-1} G(m,n)}{(M-2)\times(N-2)} \qquad (27)$$

where M and N are the height and width of the image. By using Eq.(26,27) the gray value degree is computed as,

$$GVD = \frac{W'\big(G(m,n)\big) - W(G(m,n))}{W'\big(G(m,n)\big) + W(G(m,n))} \qquad (28)$$

where W' and W denote the average neighborhood gray difference of the original image and the encrypted image. The GVD value ranges between -1 to +1. Table 11 lists the computed gray value degree values for the different images by the proposed scheme. From Table 11 it is observed that the gray value degrees calculated by the proposed scheme are very close to the ideal value of +1. Table 12 lists the comparison of the GVD with the other approaches in the literature. It can be seen that the GVD value is close to the ideal value of 1 and is moderately better than the other schemes in the literature.

Table 11. Gray value degree values for different test images.

Image	GVD value
Lena	0.962112
Clock	0.957340
Boat	0.945906
Bridge	0.948403
Tank	0.972316
House	0.939243
Truck	0.962644
Couple	0.949856

Table 12. The GVD values of the proposed scheme and other approaches

Method	GVD
Proposed algorithm	0.962112
Ref.[13]	0.954000
Arnold's	0.890000
Ref.[1]-[9]	Not specified

5.8 Computational Speed

The running time of an encryption scheme can be determined by several factors such as the programming language, operating system, system configuration etc. Table 13 shows the computational time requirement for the proposed scheme and the other schemes in the literature. The proposed scheme requires only 0.000018 seconds, and is fast compared to [3], [6] with similar computer speed. The proposed scheme is also faster compared to [2], [4] with similar programming language. The comparison of the time complexities is listed in Table 14. It can be seen that the time complexity of the proposed scheme is lower and is fast compared to other schemes.

Table 13. Comparison of the computational time with other schemes in the literature

Method	Computational time in seconds	Programming language	Operating system	System
Proposed algorithm	0.000018	C	Linux	Intel core i3 at 3.30 GHz
Ref.[1]	0.150	Matlab	Windows XP	Intel core 2 at 2.00 GHz
Ref.[2]	0.093	Visual C++	Windows XP	Intel dual core at 2.7 GHz
Ref.[3]	0.00297	Matlab	Windows Vista	Pentium Core 2 Duo at 3 GHz
Ref.[4]	0.90	C	Linux	Pentium M at 1.7GHz
Ref.[5]	0.2	Matlab	Windows	Pentium P4 dual core at 2.16 GHz
Ref.[6]	0.8645	Matlab	Windows	Pentium Core 2 Duo at 3 GHz
Ref.[7]	0.547	Matlab	Windows XP	Intel core 2 at 2.00 GHz
Ref.[8]	0.312	Matlab	Windows	Celeron processor at 1.8 Ghz
Ref.[9]	0.4	Matlab	Windows	Pentium IV at 1 GHz

Table 14. Comparison of the time complexity with other schemes in the literature

Method	Time complexity
Proposed algorithm	$O(1) + O(M \log M + N \log N) + O(MN) + O(MN)$
Ref.[1]	$O(MN) + O(MN) + O(MN)$
Ref.[2]	$O(M) + O(MN) + O(MN) + O(MN)$
Ref.[3]	$O(MN) + O(MN) + O(MN) + O(MN)$
Ref.[4]	$O(MN) + O(MN) + O(MN) + O(MN)$
Ref.[5]	$O(MN) + O(MN) + O(MN) + O(MN)$
Ref.[6]	$O(MN) + O(MN) + O(MN) + O(MN) + O(MN)$
Ref.[7]	$O(MN) + O(MN) + O(MN)$
Ref.[8]	$O(MN) + O(MN)/4) + O(MN) + O(MN) + O((MN)/2) + O((MN)/2)$
Ref.[9]	$O(MN) + O(MN) + O(MN) + O(MN) + O(MN)$

6. CONCLUSIONS

In this paper, a new approach for image encryption based on Quintuple encryption using a Gumowski-Mira and a Tent chaotic map has been proposed. The proposed approach has key space of 2^{384}, which is large enough to prevent the brute-force attacks. Moreover, the correlations for encrypted images are close to zero and the histogram is almost uniformly spread, hence the scheme is defiant to the statistical based attacks. Furthermore, the $NPCR$ and $UACI$ values are extremely close to the ideal values of 99.6% and 33.4% for both the key sensitivity and plain image sensitivity assessments respectively and hence the propose scheme is resistant to the differential attacks. The average entropy achieved is 7.997299, which is close to the ideal value of 8, and hence the information outflow is insignificant. The $PSNR$ in the proposed scheme is lower than other schemes and the GVD is near to 1. The encryption speed attained is 0.000018 seconds for 256×256 image. Accordingly, the image encryption scheme proposed in this paper will render excellent image encryption effects and high speed. The proposed scheme can also be applied for color images.

REFERENCES

[1] Guodong Ye and Kwok Wo Wong, "An efficient chaotic image encryption algorithm based on a generalized Arnold map," Nonlinear Dynamics, vol. 69, no. 4, 2012, pp. 2079-2087.

[2] Ahmed A. Abd El-Latif, Li Li, Tiejun Zhang, Ning Wang, Xianhua Song, and Xiamu Niu, "Digital image encryption scheme based on multiple chaotic systems," Sensing and Imaging, vol. 13, no. 2, 2012, pp. 67-88.

[3] Shatheesh Sam, P. Devaraj, and R. S. Bhuvaneswaran, "A novel image cipher based on mixed transformed logistic maps," Multimedia Tools and Applications, vol. 56, no. 2, 2012, pp. 315-330.

[4] M. Francois, T. Grosges, D. Barchiesi, and R. Erra, "A new image encryption scheme based on a chaotic function," Signal Processing: Image Communications, vol. 27, no. 3, 2012, pp. 249-259.

[5] Nidhi Taneja, Balasubramanian Raman, Indra Gupta, "Combinational domain encryption for still visual data," Multimedia Tools and Applications, vol. 159, no. 3, 2012, pp. 775-793.

[6] Shantheesh Sam, P. Devaraj, and R. S. Bhuvaneswaran, "An intertwining chaotic maps based image encryption scheme," Nonlinear Dynamics, vol. 69, no. 4, 2012, pp. 1995-2007.

[7] Xiaoling Huang, "Image encryption algorithm using chaotic Chebyshev generator," Nonlinear Dynamics, vol. 67, no. 4, 2012, pp. 2411-2417.
[8] Xiaofeng Liao, Shiyue Lai, and Qing Zhou, "A novel image encryption algorithm based on self-adaptive wave transmission," Signal processing, vol. 90, no. 9, 2010, pp. 2714-2722.
[9] Guanrong Chen, Yaobin Mao, and Charles K. Chui' "A symmetric image encryption scheme based on 3D chaotic cat maps," Chaos solitons and fractals, vol. 21 no. 3, 2004, pp. 749-761.
[10] Lequan Min and Guanrong Chen, "A novel stream encryption scheme with avalanche effect," The European Physical Journal B, vol. 86, 2013, p. 459.
[11] D. Chattopadhyay, M. K. Mandal, and D. Nandi, "Symmetric key chaotic image encryption using circle map," Indian Journal of Science and Technology, vol. 4, 2010, pp. 593-599.
[12] A Akhshani, S Behnia, A Akhavan, H Abu Hassan, and Z Hassan, "A novel scheme for image encryption based on 2D piecewise chaotic maps," Optics Communications, vol. 283, no. 17, 2010, pp. 3259-3266.
[13] Guodong Ye, "Image scrambling encryption algorithm of pixel bit based on chaos map," Pattern Recognition Letters, vol. 31, no. 5, 2010, pp. 347-354.
[14] Gonzalo Alvarez and Shujun Li, "Some basic cryptographic requirements for chaos-based cryptosystems," International Journal of Bifurcation and Chaos, vol. 16, no. 8, 2006, pp. 2129-2151.
[15] V. Patidar, N. K. Pareek, and K. K. Sud, "A new substitution diffusion based image cipher using chaotic standard and Logistic maps," Communications in Nonlinear Science and Numerical Simulations, vol. 14, no. 7, 2009, pp. 3056-3075.
[16] N. K. Pareek, V. Patidar, and K. K. Sud, "Image encryption using chaotic Logistic map," Image Vision and Computing, vol. 24, no. 9, 2006, pp. 926-934.
[17] Bruce schneier, "Applied cryptography," John Wiley & sons, 2001.

Gururaj Hanchinamani
He received ME degree in computer science and engineering from Walchand college of engineering Sangli, India. He has submitted his PhD thesis at Visvesvaraiah technological university Belgaum. His research interests are information security and computer architectures. He is currently working as associate professor at computer science department, BVB college of engineering and technology, Hubli, India.

Linganagouda Kulkarni
He received PhD degree in pattern recognition from Mysore university, India. His research interests are image processing, computer networks and information security. He is currently working as professor at computer science department, BVB college of engineering and technology, Hubli, India.