JCSE Journal of Computing Science and Engineering

Regular Paper
Journal of Computing Science and Engineering,
Vol. 8, No. 4, December 2014, pp. 187-198

A New Approach for Image Encryption Based on Cyclic Rotations and Multiple Blockwise Diffusions Using Pomeau-Manneville and Sin Maps

Gururaj Hanchinamani* **and Linganagouda Kulakarni**
Department of Computer Science and Engineering, BVB College of Engineering & Technology, Hubli, Karnataka, India
gs_hanchinamani@bvb.edu, linganagouda@yahoo.co.uk

Abstract
In this paper an efficient image encryption scheme based on cyclic rotations and multiple blockwise diffusions with two chaotic maps is proposed. A Sin map is used to generate round keys for the encryption/decryption process. A Pomeau-Manneville map is used to generate chaotic values for permutation, pixel value rotation and diffusion operations. The encryption scheme is composed of three stages: permutation, pixel value rotation and diffusion. The permutation stage performs four operations on the image: row shuffling, column shuffling, cyclic rotation of all the rows and cyclic rotation of all the columns. This stage reduces the correlation significantly among neighboring pixels. The second stage performs circular rotation of pixel values twice by scanning the image horizontally and vertically. The amount of rotation is based on $M \times N$ chaotic values. The last stage performs the diffusion four times by scanning the image in four different ways: block of 8×8 pixels, block of 16×16 pixels, principal diagonally, and secondary diagonally. Each of the above four diffusions performs the diffusion in two directions (forwards and backwards) with two previously diffused pixels and two chaotic values. This stage makes the scheme resistant to differential attacks. The security and performance of the proposed method is analyzed systematically by using the key space, entropy, statistical, differential and performance analysis. The experimental results confirm that the proposed method is computationally efficient with high security.

Category: Ubiquitous computing

Keywords: Image encryption/decryption; Chaotic theory; Cryptanalytic attacks; Pomeau-Manneville map; Sin map

I. INTRODUCTION

Protecting the digital images from unauthorized viewing covers a wide range of applications. With advancements in internet and communication technologies, the utilization of multimedia information has become more prevalent. Hence a great deal of concerns has been raised over the security of multimedia information transmitted or stored over open networks. It is vital to ensure the security of multimedia information. Encryption is one way to protect the information, which transforms the information in a way that makes it unreadable to anybody except those with meticulous knowledge, which usually takes the form of a key. A wide range of traditional cryptosystems, such as AES, DES, IDEA, and GOST, have been proposed in literature. However, they are computationally exhaustive and are not suitable for image encryption [1-10]. Moreover, the inherent features of images such as strong correlations among adjacent pixels, high redundancy and bulk volume of data prevent the usage of

Open Access http://dx.doi.org/10.5626/JCSE.2014.8.4.187 http://jcse.kiise.org

This is an Open Access article distributed under the terms of the Creative Commons Attribution Non-Commercial License (http://creativecommons.org/licenses/by-nc/3.0/) which permits unrestricted non-commercial use, distribution, and reproduction in any medium, provided the original work is properly cited.

Received 4 August 2014; **Accepted** 5 November 2014
*Corresponding Author

Copyright © 2014. The Korean Institute of Information Scientists and Engineers pISSN: 1976-4677 eISSN: 2093-8020

traditional text schemes for images.

In recent years, chaotic based encryption schemes have received increasing interest from cryptographers. The chaotic systems have many important properties, such as ergodicity, no periodicity, pseudorandom properties and sensitive dependence on initial conditions and system parameters [1-5, 11-13]. These properties meet cryptographic requirements, such as sensitivity to keys, diffusion and mixing. Hence chaotic based cryptosystems are expected to provide an easy and fast way for developing efficient cryptosystems.

In the past decade, several chaotic based image encryption schemes have been proposed; however, some of them hinder system performance, and suffer against bruteforce, statistical, entropy and differential attacks [5]. This paper proposes an efficient image encryption scheme based on cyclic rotations and multiple blockwise diffusions with two chaotic maps. The proposed method is resistant to statistical, bruteforce, entropy and differential attacks and has high computational speed.

The rest of the paper is organized as follows. In Section II, a literature survey is presented. Sin and Pomeau-Manneville maps are discussed in Section III. In Section IV, the proposed encryption scheme is discussed in detail. Experimental results and security analysis are presented in Section V to show the effectiveness and validity of the algorithm. The last section concludes the paper.

II. LITERATURE SURVEY

The typical architecture of chaotic based image cryptosystems includes iteration of two phases: permutation and diffusion. The permutation stage is employed to decorrelate the adjacent pixels. The diffusion stage is used to ensure plain image sensitivity. However, numerous rounds of permutation and diffusion or iterations should be taken, which makes the overall encryption speed slow [1-5, 14-18].

A short summary of recently proposed chaotic based encryption schemes is given hereafter. In [1, 4, 8, 17], the key-space is increased by using multiple chaotic maps to resist bruteforce attacks. The authors of [2] proposed an image encryption scheme with a generalized Arnold map and total circular function. In [3], the avalanche effect is introduced with a pseudo-Hadamard transform. In [5], an image encryption scheme based on cyclic elliptic curves and chaotic systems is proposed. The scheme encrypts a 256-bit plain image to a 256-bit cipher image within 32-bit registers. Huang et al. [6] proposed an image encryption scheme with pixel shuffling and gray level encryption by a single chaotic system. The authors of [7] proposed an encryption framework of combinational domain encryption that encrypts significant data in the spatial domain and insignificant data in the wavelet domain to reduce the computational time. In [9], a key stream is generated by using nonlinear Chebyshev function. The authors of [11] proposed an image encryption scheme using a large pseudorandom permutation, which is combinatorially generated from small permutation matrices based on chaotic maps. In [12], a symmetric image encryption scheme is proposed using circle maps. The authors of [13] introduced a hierarchy of 2D piecewise nonlinear chaotic maps with an invariant measure. Hu and Han [19] proposed a pixel-based scrambling scheme to protect medical images. In [20], a scrambling scheme is proposed, which can implement position encryption and gray value encryption simultaneously. The authors of [15] proposed an encryption scheme based on piecewise nonlinear chaotic maps. In [16], a common framework of guidelines for image cryptosystems is provided, and addresses three issues: implementation, key management, and security analysis. In [18], an image encryption scheme with an external 80-bit secret key and two chaotic logistic maps is proposed.

However, most of the encryption schemes in the above literature have their own strengths and constraints more or less in terms of security level and computational speed. Some of the cryptosystems have been cryptanalyzed [10, 14]. This paper proposes a new chaotic image encryption scheme based on cyclic rotations and multiple blockwise diffusions with two chaotic maps. The proposed scheme is resistant to various cryptanalytic attacks such as bruteforce attacks, statistical attacks, entropy based attacks and differential attacks. The proposed approach achieves the required level of security with only two rounds of encryption, hence computationally efficient.

III. CHAOTIC MAPS

Chaotic maps are characterized by sensitive dependence on initial conditions, similarity to random behavior, no periodicity and ergodicity. The possibility for self-synchronization of chaotic oscillations has initiated an avalanche of works on applications of chaos in cryptography. The proposed image encryption scheme uses two chaotic maps: the Sin map and Pomeau-Manneville map, which are discussed hereafter.

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

$$Z_{i+1} = r \times \sin(\pi \times Z_i) \tag{1}$$

where Z_i is the current chaotic value, Z_{i+1} is the next chaotic value, and r is the control parameter. The key set for the Sin map is $\{Z_0, r\}$. In the proposed scheme the Sin map is used to generate round keys for the Pomeau-Manneville map.

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

$$X_{i+1} = a \times X_i^2 + X_i + \varepsilon \tag{2}$$

The 2D Pomeau-Manneville map used in the proposed scheme is defined as

$$X_{i+1} = a_1 \times X_i^2 + X_i + Y_i + \varepsilon_1 \tag{3}$$

$$Y_{i+1} = a_2 \times Y_i^2 + Y_i + X_i + \varepsilon_2 \tag{4}$$

where X_i, Y_i are current chaotic values, X_{i+1}, Y_{i+1} are the next chaotic values, a_1, a_2, ε_1, ε_2 are control parameters. The key set for the Pomeau-Manneville map is $\{X_0, Y_0, a_1, a_2, \varepsilon_1, \varepsilon_2\}$. In the proposed scheme, the Pomeau-Manneville chaotic values are used during permutation, pixel value rotation and diffusion stages of the encrypt and decrypt schemes.

The propositions of chaotic maps [3, 20] are defined in Eqs. (5)–(7). The chaotic output sequence of Sin and Pomeau-Manneville maps is assessed by computing mean and self-correlations according to the propositions (5)–(7). It can be observed that the average values of the chaotic sequence are close to 0.5 and that 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 given by*

$$x_{mean} = \lim_{N\to\infty} \frac{1}{N} \sum_{k=0}^{N-1} x_k = 0.5 \tag{5}$$

PROPOSITION 2. *Self-correlation of a chaotic sequence is computed as*

$$S1(\beta) = \lim_{N\to\infty} \frac{1}{N} \sum_{k=0}^{N-1} (x_k - x_{mean})(x_{k+\beta} - x_{mean}) = 0 \tag{6}$$

PROPOSITION 3. *The self-correlation function between two chaotic sequences is calculated as*

$$S2(\beta) = \lim_{N\to\infty} \frac{1}{N} \sum_{k=0}^{N-1} (x_k - x_{mean})(y_{k+\beta} - y_{mean}) = 0 \tag{7}$$

IV. PROPOSED ENCRYPTION SCHEME

The proposed encryption scheme is composed of three stages: permutation, pixel value circular rotations, and diffusion. Also a round key generation function is used.

A. Round Key Generation and Scheming

The key set of the Pomeau-Manneville map is $\{X_0, Y_0, a_1, a_2, \varepsilon_1, \varepsilon_2\}$. In the proposed scheme, the X_0, Y_0 parameters of the Pomeau-Manneville map are generated by using a Sin map and the a_1, a_2, ε_1, ε_2 parameters are kept the same for all the rounds.

The round keys of the Pomeau-Manneville map can be generated as

$$Round\ 1: Keys = \{X_0 = Z_0,\ Y_0 = Z_1,\ a_1,\ a_2,\ \varepsilon_1,\ \varepsilon_2\}$$

$$Round\ 2: Keys = \{X_0 = Z_2,\ Y_0 = Z_3,\ a_1,\ a_2,\ \varepsilon_1,\ \varepsilon_2\}$$

$$Round\ 3: Keys = \{X_0 = Z_4,\ Y_0 = Z_5,\ a_1,\ a_2,\ \varepsilon_1,\ \varepsilon_2\}$$

and so on, where $Z_0, Z_1, Z_2, Z_3, \ldots$ are the chaotic values of the Sin map.

The key set of the Sin map is $\{Z_0, r\}$, which is used to generate the $\{X_0, Y_0\}$ values of the Pomeau-Manneville map at each round. Hence the key set used in the proposed encryption/decryption scheme is

$$Key\ set = \{Z_0, r, a_1, a_2, \varepsilon_1, \varepsilon_2\} \tag{8}$$

where $\{Z_0, r\}$ are parameters of the Sin map, and $\{a_1, a_2, \varepsilon_1, \varepsilon_2\}$ are parameters of the Pomeau-Manneville map.

The Sin map is used only to generate $\{X_0, Y_0\}$ values, and $\{a_1, a_2, \varepsilon_1, \varepsilon_2\}$ are kept constant to increase the key space.

B. Permutation

The decorrelation of adjacent pixels in an image can be achieved by employing the permutation function. Let I be a gray original image of size $M \times N$, which is a digital matrix with M rows and N columns, and whose gray values lie in the range from 0 to 255. In the process of permutation, initially $M + N$ Pomeau-Manneville chaotic values $\{(X_1, \ldots, X_M), (Y_1, \ldots, Y_N)\}$ are generated by using Eqs. (3) and (4), after doing iterations on the chaos maps. Let $SM = \{X_1, \ldots, X_M\}$ and $SN = \{Y_1, \ldots, Y_N\}$. Then SM and SN are sorted, and the positions of the sorted chaotic values in the original chaotic sequence are found and are stored in SM' and SN', respectively. The permutation stage is composed of the following four steps.

Step 1. Shuffle the row position of all values from first column to last column according to $SM'_1, \ldots, SM'_M$.

Step 2. Shuffle the column position of all values from first row to last row according to $SN'_1, \ldots, SN'_N$.

Step 3. Cyclically rotate the pixel positions row-wise from first row to last row, where the amount of rotation is based on $SM'_1, \ldots, SM'_M$. The first row is rotated by SM'_1, the second row is rotated by SM'_2, and so on.

Step 4. Cyclically rotate the pixel positions column-wise from first column to last column, where the amount of rotation is based on $SN'_1, \ldots, SN'_N$.

This stage shuffles all pixels and decorrelates the adjacent pixels.

C. Pixel Value Circular Rotations

This stage consists of two steps.

Step 1. Scan the image horizontally (left to right and then top to bottom) then apply left Circular rotation on a pixel by pixel basis.

Step 2. Scan the image vertically (top to bottom and then left to right) then apply left Circular rotation on a pixel by pixel basis.

In each of these two steps, the amount of rotation of the pixel values is based on $M \times N$ chaotic values, and is calculated with following steps.

Step 1. Generate $M \times N$ chaotic values using Eqs. (3) and (4).

Step 2. Transform the real chaotic sequence to an integer form with the following transform:

$$T_i' = (T_i * 10^8) mod\ m \quad (9)$$

where T_i is real chaotic value, T_i' is the transformed integer value and m is 256 for the gray level image.

Step 3. Apply mod 8, and replace 0 by a number between 1 and 7. This step is necessary because the rotation of a pixel value by 0 does not change the value.

The above steps generate $M \times N$ chaotic values in the range of 1 to 7, and are used as the rotation amount for the $M \times N$ pixels of the image.

D. Diffusion

The diffusion stage is employed to ensure the plain image sensitivity, i.e., a 1-bit change of the plain image should produce a radical change in the encrypted image using the same key. The proposed scheme applies the diffusion process four times by scanning the image in four different ways as given below and shown in Fig. 1.

Step 1. Scan the image in blocks of 8×8 pixels, and then apply forward and backward diffusions.

Step 2. Scan the image in blocks of 16×16 pixels, and then apply forward and backward diffusions.

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

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

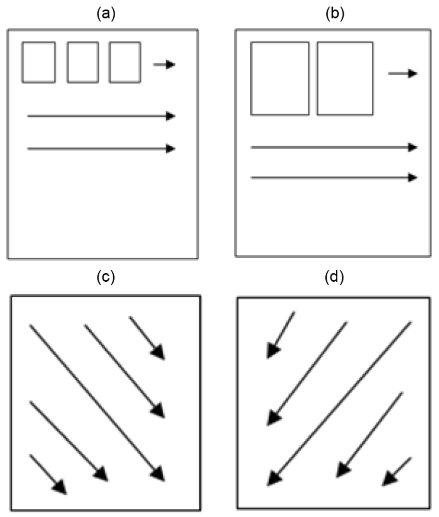

Fig. 1. scanning directions (a) block of 8×8 pixels, (b) block of 16×16 pixels, (c) principal diagonal, and (d) secondary diagonal.

The forward and backward diffusions in each of the above four steps are performed as discussed below.

The 2D image is transformed to a 1D array $P_{1\times MN}$ by scanning the image in the directions as shown in Steps 1–4. The diffusion process is performed in two directions (forwards and backwards) with two chaotic values of the Pomeau-Manneville map $\{X, Y\}$ and two previously diffused pixels. As the calculated encrypted pixel values depend on previously encrypted pixels and chaotic sequences, the algorithm shows resistance against differential attacks.

Initially $M \times N$ chaotic values $\{(X_1, ..., X_{M\times N}), (Y_1, ..., Y_{M\times N})\}$ are generated by using Eqs. (3) and (4) after doing iterations in chaos maps. Then the real chaotic sequences are transformed to integer form by using Eq. (9).

The forward diffusion is performed by using the following equation:

$$E_i = (((((P_i + E_{i-2}) mod\ 256) + E_{i-1}) mod\ 256 \oplus X_i) + Y_i)\ mod\ 256,\ i = 1, 2, ..., MN \quad (10)$$

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

The backward diffusion is performed with the following equation to make the influence of every pixel equal:

$$F_i = (((((E_i + F_{i+2}) mod\ 256) + F_{i+1}) mod\ 256 \oplus X_i) + Y_i)\ mod\ 256,\ i = MN, ..., 1 \quad (11)$$

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

E. Decryption

Decryption involves the inverse steps of the encryption process to reconstruct the original image from the encrypted image. It is a simple reverse process of the proposed encryption scheme. Initially the diffusion process is carried out in a reverse sequence: secondary diagonally, principal diagonally, block of 16×16 pixels, and block of 8×8 pixels. Then circular pixel rotation is applied in the inverse rotation direction. Lastly the permutation is performed in reverse sequence: cyclic rotation column-wise, cyclic rotation row-wise, column shuffling, and row shuffling.

F. Encryption Algorithm

The encryption algorithm is composed of a mainline routine for round key generation and an encrypt function.

1) Mainline Routine for Round Key Generation

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

Step 2. Initialize the Sin map $\{Z_0, r\}$ and generate chaotic values using Eq. (1). The number of chaotic values to be generated depends on the number of rounds. Two rounds of the encryption function give sufficient security.

Step 3. Initialize the Pomeau-Manneville map $\{X_0, Y_0, a_1, a_2, \varepsilon_1, \varepsilon_2\}$. The $\{X_0, Y_0\}$ parameters are initialized with chaotic values of the Sin map. The other four parameters $\{a_1, a_2, \varepsilon_1, \varepsilon_2\}$ are initialized with the same values in all rounds.

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

Step 5. Initialize the Pomeau-Manneville map.

Step 6. Invoke the Encrypt function for the modified image.

2) Encrypt Function

The encrypt function is composed of nineteen steps.

Step 1. Get the image.

Step 2. Generate M chaotic values of X_i sequence (X_1, ..., X_M) and N chaotic values of Y_i sequence (Y_1, ..., Y_N) using Eqs. (3) and (4).

Step 3. Copy X_i chaotic values to SM and Y_i chaotic values to SN.

Step 4. Sort SM and SN, find the position of sorted chaotic values in the original chaotic sequence and store in SM' and SN', respectively.

Step 5. Shuffle the row position of all values from first column to last column according to SM'_1, ..., SM'_M.

Step 6. Shuffle the column position of all values from first row to last row according to SN'_1, ..., SN'_N.

Step 7. Cyclically rotate the pixel positions row-wise from first row to last row, where the amount of rotation is based on SM'_1, ..., SM'_M. The first row is rotated by SM'_1, the second row is rotated by SM'_2 and so on.

Step 8. Cyclically rotate the pixel positions column-wise from first column to last column, where the amount of rotation is based on SN'_1, ..., SN'_N.

Step 9. Generate $M \times N$ chaotic values using Eqs. (3) and (4) for pixel value circular rotations.

Step 10. Transform the real chaotic sequence to an integer form with the transform given in Eq. (9).

Step 11. For the above integer chaotic sequence, apply mod 8, and replace 0 by a number between 1 and 7, and use these $M \times N$ values as rotation amounts for $M \times N$ pixel values.

Step 12. Scan the image horizontally (left to right and then top to bottom), then apply left circular rotation on a pixel by pixel basis.

Step 13. Scan the image vertically (top to bottom and then left to right), then apply left circular rotation on a pixel by pixel basis.

Step 14. Generate $M \times N$ chaotic values $\{(X_1, ..., X_{M \times N}), (Y_1, ..., Y_{M \times N})\}$ using Eqs. (3) and (4) for the diffusion process.

Step 15. Transform real chaotic sequence to an integer sequence using Eq. (9).

Step 16. Scan the image in blocks of 8 × 8 pixels, and then apply forward and backward diffusions using Eqs. (10) and (11).

Step 17. Scan the image in blocks of 16 × 16 pixels, and then apply forward and backward diffusions using Eqs. (10) and (11).

Step 18. Scan the image in principal diagonal direction and then apply forward and backward diffusions using Eqs. (10) and (11).

Step 19. Scan the image in secondary diagonal direction and then apply forward and backward diffusions using Eqs. (10) and (11).

G. Decryption Algorithm

The decryption algorithm is composed of a mainline routine for round key generation and a decrypt function.

1) Mainline Routine for Round Key Generation

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

Step 2. Initialize the Sin map $\{Z_0, r\}$ and generate chaotic values using Eq. (1). The number of chaotic values to be generated depends on the number of rounds.

Step 3. Initialize the Pomeau-Manneville map $\{X_0, Y_0, a_1, a_2, \varepsilon_1, \varepsilon_2\}$. The $\{X_0, Y_0\}$ parameters are initialized with chaotic values of the Sin map in the reverse way of their invocations. The other four parameters $\{a_1, a_2, \varepsilon_1, \varepsilon_2\}$ are initialized with the same values in all rounds.

Step 4. Invoke the Decrypt function for encrypted image.

Step 5. Initialize the Pomeau-Manneville map.

Step 6. Invoke the Decrypt function for modified image.

2) Decrypt Function

The decrypt function is composed of nineteen steps.

Step 1. Get the image.

Step 2. Generate $M \times N$ chaotic values $\{(X_1, ..., X_{M \times N}), (Y_1, ..., Y_{M \times N})\}$ using Eqs. (3) and (4) for reverse diffusions.

Step 3. Transform real chaotic values to integers using Eq. (9).

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 blocks of 16 × 16 pixels, and then apply reverse forward and backward diffusions.

Step 7. Scan the image in blocks of 8 × 8 pixels, and then apply reverse forward and backward diffusions.

Step 8. Generate $M \times N$ chaotic values using Eqs. (3) and (4) for reverse pixel value circular rotations.

Step 9. Transform the real chaotic sequence to an integer form with the transform given in Eq. (9).

Step 10. For the above integer chaotic sequence, apply mod 8, and replace 0 by a number between 1 and 7, and use these $M \times N$ values as rotation amounts for $M \times N$ pixel values.

Step 11. Scan the image vertically (top to bottom and then left to right), then apply right circular rotation on a pixel by pixel basis.

Step 12. Scan the image horizontally (left to right and then top to bottom), then apply right circular rotation on a pixel by pixel basis.

Step 13. Generate M chaotic values of X_i sequence (X_1, ..., X_M) and N chaotic values of Y_i sequence (Y_1, ..., Y_N) using Eqs. (3) and (4).

Step 14. Copy X_i chaotic values to SM and Y_i chaotic values to SN.

Step 15. Sort SM and SN, find the position of the sorted chaotic values in the original chaotic sequence and store in SM' and SN', respectively.

Step 16. Cyclically rotate the pixel positions column-wise from first column to last column in reverse direction, and the amount of rotation is based on SN'_1, ..., SN'_N.

Step 17. Cyclically rotate the pixel positions row-wise from first row to last row in reverse direction, where the amount of rotation is based on SM'_1, ..., SM'_M.

Step 18. Inverse shuffle the column position of all values from first row to last row according to SN'_1, ..., SN'_N.

Step 19. Inverse shuffle the row position of all values from first column to last column according to SM'_1, ..., SM'_M.

V. EXPERIMENTS AND SECURITY ANALYSIS

The proposed work is implemented using the C programming language on a Linux platform running on a personal computer with an Intel Core i3-2120 CPU at 3.30 GHz with 2.91 GB of RAM. The initial values and system parameters of the chaotic system are randomly set according to Eq. (11) as $\{Z_0 = 0.3, r = 0.99, a_1 = 1.0, a_2 = 1.01, \varepsilon_1 = 0.20, \varepsilon_2 = 0.30\}$, where $\{Z_0, r\}$ are the parameters of the Sin map and $\{a_1, a_2, \varepsilon_1, \varepsilon_2\}$ are the parameters of the Pomeau-Manneville map. The test images are 256×256 gray scale images chosen from the *USC-SIPI* image database (http://sipi.usc.edu/database/). This section analyzes the security of the proposed scheme to show its effectiveness in resisting attacks, such as brute-force attacks, statistical attacks, entropy-based attacks, and differential attacks.

The proposed encryption scheme has been tested with a variety of images with differing content. Fig. 2 shows the visual assessment of the encrypted images and decrypted images for four different images. The first row shows the original plain images, second row shows the encrypted images and the last row shows the decrypted images. The encrypted images are totally unrecognizable, disordered, unintelligible, incomprehensible, random, and noise-like images without any leakage of the original information. The decrypted images exactly match the original plain images.

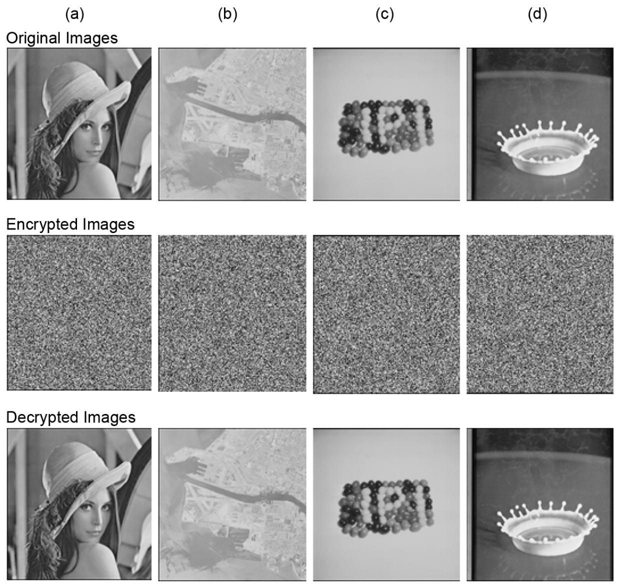

Fig. 2. Original images, encrypted images and decrypted images with proposed algorithm (a) Lena, (b) Oakland, (c) Jellybeans, and (d) Splash.

A. Histogram Analysis

An image histogram is a plot that shows the frequency distribution of the pixel intensity values. The histograms present the statistical properties of the images [1, 3, 8]. An encrypted image is anticipated to have no statistical similarity with the original plain image. The histograms of several plain images and encrypted images are computed and analyzed and are shown in Fig. 3. From Fig. 3 it can be seen that the histograms of the encrypted images are uniformly distributed and are completely different from that of the original plain image, and bear no statistical similarity to the original images. Hence the proposed scheme is resistant to histogram based statistical attacks.

B. Key-Space Analysis

Bruteforce attack is an attack model, where an invader tries to break the cryptosystem by exhaustive search with each and every possible key [3]. It can be resisted by increasing the key-space. The key set of the proposed encryption scheme is $\{Z_0, r, a_1, a_2, \varepsilon_1, \varepsilon_2\}$, where $\{Z_0, r\}$ are the keys of the Sin map and $\{a_1, a_2, \varepsilon_1, \varepsilon_2\}$ are the keys of the Pomeau-Manneville map. With 64 bits for each parameter and there are six parameters, 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 lists the key-space size

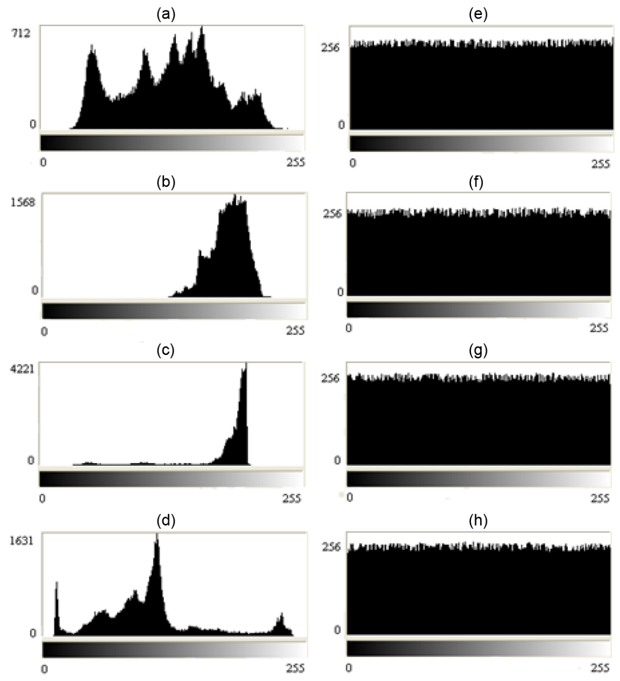

Fig. 3. Histograms of original and encrypted images. (a-d) Histograms of original images Lena, Oakland, Jellybeans, and Splash. (e-f) Histograms of respective encrypted images.

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

Encryption scheme	Proposed approach	Ye [20]	Chattopadhyay et al. [12]
Key-space size	2^{384}	2^{128}	2^{256}

of the proposed scheme compared to other approaches.

C. Information Entropy Analysis

In information theory, entropy is a measure of the uncertainty associated with a random variable [2, 3, 8]. This quantifies the expected value of the information contained in a message, and it is a measure of the amount of randomness in the information content, defined as

$$H(S) = \sum_{i=0}^{r-1} P(S_i) log_2 \frac{1}{P(S_i)} \tag{12}$$

where S_i represents the pixel intensity values, $P(S_i)$ is the probability of the symbol S_i, and r is the total number of symbols (256 for gray level image). Suppose that the gray level image has 2^8 = 256 gray levels with identical

Table 2. Entropy values for original and encrypted images for different images

Image	Entropy	
	Original image	Encrypted image
Lena	7.426985	7.997390
Oakland	6.062537	7.997442
Jellybeans	5.723857	7.997538
Splash	7.232135	7.997480
Moon surface	6.711624	7.997070
San Francisco	5.894227	7.997397
Pentagon	6.549220	7.997143
Man	7.534870	7.997478
Airplane	6.712137	7.997325
San Diego	6.361934	7.997047

probabilities $S = (S_0, S_1, S_2, ..., S_{255})$. According to Eq. (12), we obtain an entropy value $H(S) = 8$. Generally, the entropy of the original plain image is smaller than the ideal value of 8, due to high redundancy and correlations. The entropy reaches the maximum ideal value of 8 when all pixels are distributed randomly. The entropies of the original plain images and encrypted images are listed in Table 2. From the results, it is clear 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 entropy based attacks. The comparison of entropy values with other schemes is listed in Table 3.

D. Correlation Analysis

By and large, for any original plain image having specific visual content, each pixel is greatly correlated with its adjacent pixels in all three directions: horizontal, vertical, and diagonal [3]. Therefore, an encryption scheme is anticipated to generate encrypted images with no such correlations to the neighboring pixels. The correlation coefficient of neighboring pixels in an image is calculated according to Eqs. (13)–(16).

$$E(x) = \frac{1}{N}\sum_{i=1}^{N} x_i \tag{13}$$

$$D(x) = \frac{1}{N}\sum_{i=1}^{N} (x_i - E(x))^2 \tag{14}$$

Table 3. Comparison of entropy values of the proposed scheme with other methods for Lena image

Method	Proposed approach	RC5	RC6	Patidar et al. [17]	Pareek et al. [18]
Entropy values	7.997390	7.9812	7.9829	7.9923	7.9884

Table 4. Correlation coefficients of adjacent pixels in different directions for original and encrypted images

Image	Correlation coefficients for encrypted images			Correlation coefficients for original images		
	Horizontal	Vertical	Diagonal	Horizontal	Vertical	Diagonal
Lena	0.006620	-0.001182	-0.001825	0.968683	0.943269	0.933408
Oakland	-0.004383	-0.002966	0.006973	0.853729	0.885210	0.799235
Jellybeans	-0.008395	0.007179	0.004751	0.980616	0.978190	0.963775
Splash	0.001427	-0.006721	-0.001450	0.979588	0.970683	0.957963
Moon surface	0.004252	-0.002863	0.002611	0.932532	0.899993	0.870292
San Francisco	0.004762	-0.005266	0.000895	0.792038	0.800018	0.760571
Pentagon	-0.011602	0.001178	-0.003034	0.803426	0.794238	0.718547
Man	-0.008528	-0.013527	-0.001581	0.950883	0.937704	0.912943
Airplane	-0.004987	-0.003271	0.000374	0.926508	0.932170	0.876223
San Diego	-0.000259	0.004623	0.003210	0.813044	0.816869	0.738995

Table 5. Comparison of correlation coefficients of the proposed scheme with other methods for Lena image

Method	Direction		
	Horizontal	Vertical	Diagonal
Plain-image	0.968683	0.943269	0.933408
Proposed scheme	0.006620	-0.001182	-0.001825
Francois et al. [8]	0.0089	-0.0215	-0.0074
AES	-0.0160	0.8018	-0.0140
Chen's	0.0442	0.9728	0.0469
Arnold's	0.0787	-0.0793	-0.0633

$$cov(x, y) = \frac{1}{N}\sum_{i=1}^{N}(x_i - E(x))(y_i - E(y)) \quad (15)$$

$$r_{xy} = \frac{cov(x, y)}{\sqrt{D(x)}\sqrt{D(y)}} \quad (16)$$

where x and y are adjacent pixels of the original or encrypted images, $E(x)$ is the mean value, $D(x)$ is the deviation with regard to mean, $cov(x, y)$ is the covariance among adjacent pixels, and r_{xy} is the correlation coefficient. To assess the correlation within the original and encrypted images, pairs of 4096 adjacent pixels are randomly selected in horizontal, vertical and diagonal directions, and their correlation coefficients are calculated using Eq. (16). The correlation coefficients of the original plain images and encrypted images for various images are listed in Table 4. From the correlation results it is observed that the adjacent pixels in the original image are extremely correlated to each other, but that the correlation coefficients for encrypted images are very close to zero. Thus the proposed scheme is resistant to correlation based statistical attacks. The comparison of correlation results with other approaches is given in Table 5.

Table 6. Gray value degree (GVD) values for different test images

Image	GVD value
Lena	0.962221
Oakland	0.986025
Jellybeans	0.989301
Splash	0.978644
Moon surface	0.977346
San Francisco	0.982308
Pentagon	0.959951
Man	0.937024
Airplane	0.947789
San Diego	0.957429

E. Gray Value Degree (GVD) Analysis

The gray difference of a pixel with its four neighborhood pixels is calculated as in [3, 20]:

$$G = \frac{\Sigma[I(m, n) - I(m', n')]^2}{4},$$
$$here(m', n') = \begin{cases} (m-1, n) \\ (m+1, n) \\ (m, n-1) \\ (m, n+1) \end{cases} \quad (17)$$

where $I(m, n)$ indicate the pixel intensity value at location (m, n), and $I(m', n')$ is the pixel intensity values of the four neighboring pixels. The average neighborhood gray difference for the whole image can be calculated by Eq. (18):

$$W(G(m, n)) = \frac{\sum_{m=2}^{M-1}\sum_{n=2}^{N-1}G(m, n)}{(M-2)\times(N-2)} \quad (18)$$

Table 7. The gray value degree (GVD) values of the proposed scheme and other approaches

Image	GVD value		
	Proposed approach	Arnold's	Ye [20]
Lena	0.962221	0.89	0.954

where M and N are the numbers of rows and columns of the image, respectively. By using Eqs. (17) and (18), the gray value degree is computed as

$$GVD = \frac{W'(G(m,n)) - W(G(m,n))}{W'(G(m,n)) + W(G(m,n))} \tag{19}$$

where W' and W indicate the average neighborhood gray difference of the original plain image and encrypted image. The GVD value ranges between -1 and +1. The gray value degree values for different images computed by the proposed scheme are listed in Table 6. From Table 6 it can be observed that the gray value degrees are close to the ideal value of +1. Table 7 shows the evaluation of GVD compared to other methods.

F. Peak Signal-to-Noise Ratio (PSNR) Analysis

By considering the original plain image as a signal and the encrypted image as a noise [3, 7], an objective assessment of the encryption scheme can be performed by computing the PSNR. The PSNR is defined as

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

where MSE indicates the mean square error and is calculated according to Eq. (21).

$$MSE = \frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}(|I(i,j) - I'(i,j)|)^2 \tag{21}$$

where $I(i,j)$ indicates 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 for different test images are computed and listed in Table 8. From Table 8 it can be observed that the PSNR values are smaller, which indicates the increased complexity in getting the original plain image from the encrypted image for opponents.

G. Key Sensitivity Analysis

Key sensitivity means that a one-bit change of the key should produce a radical change in the encrypted image using the same plain image [1-10]. Key sensitivity analysis is conducted with the following scheme.

Step 1. The original plain image is encrypted by using a test key K_1 to create cipher image C_1.

Step 2. The original plain image is encrypted another time with a minuscule change in the test key K_1, i.e., K_2 to create cipher image C_2.

Table 8. The PSNR values for different test images

Image	PSNR (dB)
Lena	9.255873
Oakland	8.859804
Jellybeans	8.451614
Splash	8.792241
Moon surface	10.186010
San Francisco	10.150878
Pentagon	10.247757
Man	8.083686
Airplane	8.043295
San Diego	9.315116

PSNR: peak signal-to-noise ratio.

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

The number of pixels change rate (NPCR) and unified average changing intensity (UACI) parameters are used to evaluate the key sensitivity and are discussed after this.

NPCR is the percentage of the number of differing pixels between two images and is defined as

$$NPCR = \frac{\sum_{i,j} D(i,j)}{M \times N} \times 100\% \tag{22}$$

$$D(i,j) = \begin{cases} 1, \ if\ C_1(i,j) \neq C_2(i,j) \\ 0 \qquad , Otherwise \end{cases} \tag{23}$$

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 of the same size as C_1 and C_2 and its contents are either 0 or 1 based on Eq. (23).

UACI is the percentage of the average changing intensity difference between two images and is defined as

$$UACI = \frac{1}{M \times N}\left[\sum_{ij} \frac{C_1(i,j) - C_2(i,j)}{255}\right] \times 100\% \tag{24}$$

The key sensitivity is assessed by testing one parameter at a time with a slightly change in the keys. The proposed scheme has six parameters $\{Z_0, r, a_1, a_2, \varepsilon_1, \varepsilon_2\}$. Table 9 shows the NPCR and UACI values for six different parameters. From Table 9 it can be observed that the NPCR and UACI values are extremely close to their ideal values of 99.6% and 33.4%, respectively. Thus the proposed scheme has great key sensitivity.

Furthermore, the key sensitivity can also be demonstrated visually with the following approach. The original

Table 9. Key sensitivity results for Splash image with different parameters of the chaotic map

Parameter changed	NPCR (%)	UACI (%)
Z_0	99.591064	33.408295
r	99.603271	33.395805
a_1	99.610901	33.406994
a_2	99.594961	33.392109
ε_1	99.612427	33.344021
ε_2	99.595642	33.402004

NPCR: number of pixels change rate, UACI: unified average changing intensity.

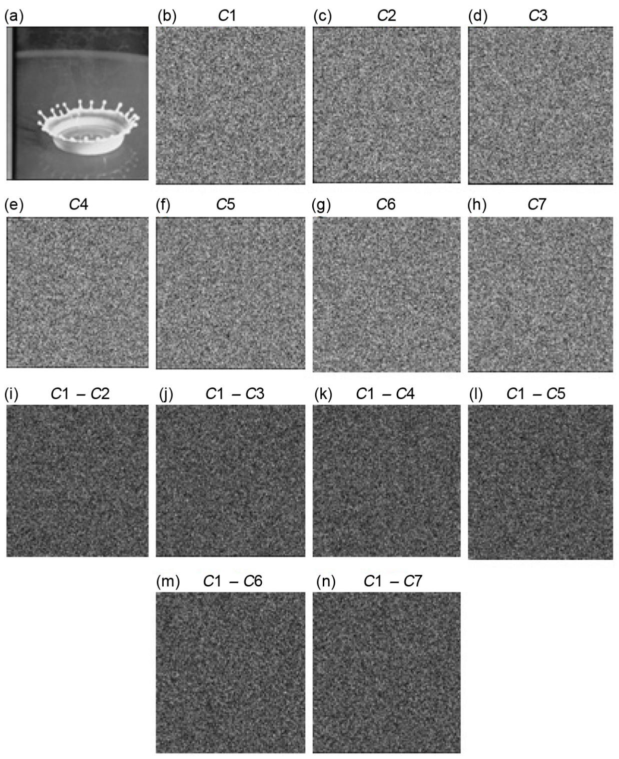

Fig. 4. Key sensitivity analysis for encryption process for Splash image. (a) Original image. (b) Encrypted image with correct key Key_1. (c–h) Encrypted images with slightly different keys. (i–n) Difference image between C_1 and other incorrect encrypted images.

key is altered with a slightly change and different keys are generated. The keys can be summarized as,

Original key $\boldsymbol{Key_1}$ = (0.3, 0.99, 1.0, 1.01, 0.2, 0.3), and the slightly altered keys,

$\boldsymbol{Key_2}$ = (**0.30000000001**, 0.99, 1.0, 1.01, 0.2, 0.3),
$\boldsymbol{Key_3}$ = (0.3, **0.99000000001**, 1.0, 1.01, 0.2, 0.3),
$\boldsymbol{Key_4}$ = (0.3, 0.99, **1.0000000001**, 1.01, 0.2, 0.3),
$\boldsymbol{Key_5}$ = (0.3, 0.99, 1.0, **1.01000000001**, 0.2, 0.3),
$\boldsymbol{Key_6}$ = (0.3, 0.99, 1.0, 1.01, **0.20000000001**, 0.3),
$\boldsymbol{Key_7}$ = (0.3, 0.99, 1.0, 1.01, 0.2, **0.30000000001**).

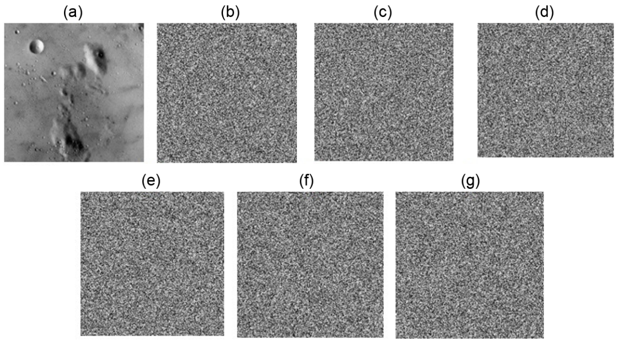

Fig. 5. Key sensitivity analysis for decryption process for Moon surface image. (a) Decryption with correct key. (b–g) Decryption with slightly changed keys.

The encrypted images with the correct key $\{C_1\}$ and the bit altered keys $\{C_2, C_3, C_4, C_5, C_6, C_7\}$ are given in Fig. 4(b)–(h). Even though all look related, they are wholly dissimilar from each other. This can be confirmed by finding the difference image between C_1 and the other encrypted images $\{C_2, C_3, C_4, C_5, C_6, C_7\}$. Fig. 4(i)–(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 can be observed that: a large amount of the pixels are nonzero, thus the difference is adequate.

Furthermore, the key sensitivity test is also assessed using the decryption process. Decryption is performed with the right key and slightly different keys. Fig. 5(a) shows the decrypted image with the correct key $\boldsymbol{Key_1}$ and Fig. 5(b)–(g) are the decrypted images with the slightly altered keys $\boldsymbol{Key_1}$, $\boldsymbol{Key_2}$, $\boldsymbol{Key_3}$, $\boldsymbol{Key_4}$, $\boldsymbol{Key_6}$, $\boldsymbol{Key_7}$. The decrypted images with the slightly altered keys are non-recognizable, thus the correct decryption cannot be achieved if there is a small alteration in the key.

H. Plain-Image Sensitivity Analysis

Plain-image sensitivity means that a 1-bit change of the plain image should produce a radical change in the encrypted image using the same key [1-10]. The plain-image sensitivity assessment is performed with following steps.

Step 1. Encrypt the original plain-image to create cipher image C_1.

Step 2. Change 1-bit of original plain-image at any random position, and encrypt another time to create cipher image C_2.

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

The plain image sensitivity is assessed by using NPCR and UACI parameters as specified in Eqs. (22)–(24). The NPCR and UACI values are calculated for different arbitrarily chosen locations by changing one bit at a time. It is observed that they are close to their ideal values of 99.6% and 33.4%, respectively, irrespective of the pixel position chosen. Table 10 shows the NPCR and UACI values for a

Table 10. Plain-image sensitivity test for Jellybeans image

Bit position	Changed pixel values	NPCR (%)	UACI (%)
0	196	99.629211	33.477940
1	199	99.662781	33.445370
2	193	99.617004	33.376114
3	205	99.594116	33.432789
4	213	99.574280	33.438553
5	229	99.594116	33.354668
6	133	99.632263	33.523315
7	69	99.591064	33.488377
Average values		99.611854	33.442140

NPCR: number of pixels change rate, UACI: unified average changing intensity.

Table 11. Execution time comparison for 256×256 image

Method	Encryption time (sec)
Ye [20]	0.150
Gao and Chen [21]	0.633
Ye [22]	>10
Huang [9]	0.547
Proposed scheme	0.000006

randomly chosen location (50, 75) and with a pixel intensity value of 197. The NPCR and UACI are computed by changing one bit at a time from the lower bit (bit 0) to upper bit (bit 7), where the average NPCR and UACI values are found to be 99.611854% and 33.442140%. Thus, the proposed approach has higher sensitivity to plain-images and is resistant to differential attacks.

I. Computational Speed Analysis

The time complexity of the proposed scheme is O($M \times N$), where M and N are the height and width of the image, respectively. Only two rounds of the encryption process provide adequate security. The time required to encrypt 256 × 256 gray scale image is 0.000006 second and for decryption it is the same. Hence the proposed approach can offer a fast and competent way for image encryption. The comparison of the encryption time with other schemes is listed in Table 11.

VI. CONCLUSIONS

In this paper, an efficient image encryption scheme based on cyclic rotations and multiple blockwise diffusions with two chaotic maps was proposed. It was implemented using the C programming language on a Linux platform and the speed attained was 0.000006 seconds for a 256 × 256 images, hence it is computationally efficient. The proposed scheme has a key space of 2^{384}, which is adequate to avoid bruteforce attacks. The average entropy achieved is 7.997331, which is close to the ideal value of 8, and hence the information outflow is insignificant. The NPCR and UACI values are close to their ideal values of 99.6% and 33.4%, respectively, for both key sensitivity and plain image sensitivity, hence the method is secure against differential attacks. The correlations are close to zero and the histogram is almost uniformly spread, thus statistical attacks are resisted. The GVD is close to 1 and the PSNR is smaller. The results shown in Section V are the obtained results after two rounds of encryption. Hence, the proposed scheme offers high security and high speed. The proposed scheme is also applicable to larger sized images.

REFERENCES

1. A. A. A. El-Latif, L. Li, T. Zhang, N. Wang, X. Song, and Niu, X, “Digital image encryption scheme based on multiple chaotic systems,” *Sensing and Imaging: An International Journal*, vol. 13, no. 2, pp. 67-88, 2012.
2. G. Ye and K. W. Wong, “An efficient chaotic image encryption algorithm based on a generalized Arnold map,” *Nonlinear Dynamics*, vol. 69, no. 4, pp. 2079-2087, 2012.
3. G. Hanchinamani and L. Kulakarni, “Image encryption based on 2-D Zaslavskii chaotic map and pseudo hadmard transform,” *International Journal of Hybrid Information Technology*, vol. 7, no. 4, pp. 185-200, 2014.
4. I. S. Sam, P. Devaraj, and R. S. Bhuvaneswaran, “An intertwining chaotic maps based image encryption scheme,” *Nonlinear Dynamics*, vol. 69, no. 4, pp. 1995-2007, 2012.
5. A. A. A. El-Latif, L. Li, and X. Niu, “A new image encryption scheme based on cyclic elliptic curve and chaotic system,” *Multimedia Tools and Applications*, vol. 70, no. 3, pp. 1559-1584, 2012.
6. C. K. Huang, C. W. Liao, S. L. Hsu, and Y. C. Jeng, “Implementation of gray image encryption with pixel shuffling and gray-level encryption by single chaotic system,” *Telecommunication Systems*, vol. 52, no. 2, pp. 563-571, 2013.
7. N. Taneja, B. Raman, and I. Gupta, “Combinational domain encryption for still visual data,” *Multimedia Tools and Applications*, vol. 59, no. 3, pp. 775-793, 2012.
8. M. François, T. Grosges, D. Barchiesi, and R. Erra, “A new image encryption scheme based on a chaotic function,” *Signal Processing: Image Communication*, vol. 27, no. 3, pp. 249-259, 2012.
9. X. Huang, “Image encryption algorithm using chaotic Chebyshev generator,” *Nonlinear Dynamics*, vol. 67, no. 4, pp. 2411-2417, 2012.
10. L. Zhao, A. Adhikari, D. Xiao, and K. Sakurai, “On the security analysis of an image scrambling encryption of pixel bit and its improved scheme based on self-correlation encryption,” *Communications in Nonlinear Science and*

Numerical Simulation, vol. 17, no. 8, pp. 3303-3327, 2012.
11. J. W. Yoon and H. Kim, "An image encryption scheme with a pseudorandom permutation based on chaotic maps," *Communications in Nonlinear Science and Numerical Simulation*, vol. 15, no. 12, pp. 3998-4006, 2010.
12. D. Chattopadhyay, M. K. Mandal, and D. Nandi, "Symmetric key chaotic image encryption using circle map," *Indian Journal of Science and Technology*, vol. 4, no. 5, pp. 593-599, 2011.
13. A. Akhshani, S. Behnia, A. Akhavan, H. A. Hassan, and Z. Hassan, "A novel scheme for image encryption based on 2D piecewise chaotic maps," *Optics Communications*, vol. 283, no. 17, pp. 3259-3266, 2010.
14. C. Cokal and E. Solak, "Cryptanalysis of a chaos-based image encryption algorithm," *Physics Letters A*, vol. 373, no. 15, pp. 1357-1360, 2009.
15. S. Behnia, A. Akhshani, S. Ahadpour, H. Mahmodi, and A. Akhavan, "A fast chaotic encryption scheme based on piecewise nonlinear chaotic maps," *Physics Letters A*, vol. 366, no. 4, pp. 391-396, 2007.
16. G. Alvarez and S. Li, "Some basic cryptographic requirements for chaos-based cryptosystems," *International Journal of Bifurcation and Chaos*, vol. 16, no. 8, pp. 2129-2151, 2006.
17. 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 Simulation*, vol. 14, no. 7, pp. 3056-3075, 2009.
18. N. K. Pareek, V. Patidar, and K. K. Sud, "Image encryption using chaotic logistic map," *Image and Vision Computing*, vol. 24, no. 9, pp. 926-934, 2006.
19. J. Hu and F. Han, "A pixel-based scrambling scheme for digital medical images protection," *Journal of Network and Computer Applications*, vol. 32, no. 4, pp. 788-794, 2009.
20. G. Ye, "Image scrambling encryption algorithm of pixel bit based on chaos map," *Pattern Recognition Letters*, vol. 31, no. 5, pp. 347-354, 2010.
21. T. Gao and Z. Chen, "A new image encryption algorithm based on hyper-chaos," *Physics Letters A*, vol. 372, no. 4, pp. 394-400, 2008.
22. R. Ye, "A novel chaos-based image encryption scheme with an efficient permutation-diffusion mechanism," *Optics Communications*, vol. 284, no. 22, pp. 5290-5298, 2011.

Gururaj Hanchinamani

He received an M.E. degree in computer science and engineering from Walchand college of engineering Sangli, India. He is pursuing his Ph.D. degree at Visvesvaraiah Technological University Belgaum, India. His research interests are information security and computer architectures. He is currently working as associate professor at the computer science department, BVB college of engineering and technology, Hubli, Karnataka, India.

Linganagouda Kulakarni

He received his Ph.D. degree in pattern recognition from Mysore university, India. His research interests are image processing, computer networks and information security. He is currently working as a professor in the computer science department, BVB college of engineering and technology, Hubli, Karnataka, India.