Image Encryption Using Chaotic 3-D Arnold's Cat Map and Logistic Map

Farhan Musanna, Asha Rani and Sanjeev Kumar

Abstract Image encryption is different from that of traditional texts or binary data because of some inherent properties of images such as large data capacity, i.e., enormous size and high redundancy (statistical and psycho-visual), making them difficult to handle by traditional methods. In recent years, chaos theory has been explored to find efficient ways to develop secure image cryptosystems. Due to the desirable properties of mixing and sensitivity to initial conditions and parameters (butterfly effect), chaotic systems have found great deal in the domain of image encryption. In this paper, the 2-D chaotic cat map is generalized to its 3-D map counterpart for designing a real-time secure and reliable symmetric encryption design. This new scheme deploys the 3-D Arnold's cat map to shuffle the positions of image pixels and uses the other chaotic logistic map to perplex the relationship between the plain-image and the cipher-image, thereby significantly enhancing the robustness to differential and statistical attacks. Experimental tests are carried out with comprehensive analysis, demonstrating the high security of the scheme.

Keywords 3-D Arnold's cat map · Logistic map · Chaos · Chaotic cryptography

1 Introduction

In recent years, in the world of digital image processing and encryption, the transmission of images over an unprotected channel has been a major source of concern. The images being transferred may be of varying importance ranging from a blueprint of a

F. Musanna (✉) · S. Kumar
Department of Mathematics, IIT Roorkee, Roorkee, India
e-mail: farhanmusanna@gmail.com

S. Kumar
e-mail: malikfma@iitr.ernet.in

A. Rani
Department of Computer Science and Engineering, Indian Institute of Technology Roorkee, Roorkee 247667, India
e-mail: ashar.mcs@iitr.ac.in

B. B. Chaudhuri et al. (eds.), *Proceedings of 2nd International Conference on Computer Vision & Image Processing*, Advances in Intelligent Systems and Computing 704, https://doi.org/10.1007/978-981-10-7898-9_30

city, digital signatures, satellite images, etc. Therefore, there is a need of encryption of the image to prevent eavesdropping and tampering in order to protect the integrity and information of the images. Various traditional encryption algorithms such as Advanced Encryption Standard (AES), Data Encryption Standard (DES), Rivest, Shamir and Adleman (RSA), International Data Encryption Algorithm (IDEA) have been incorporated for encryption of text data. But due to the high redundancy and bulk capacity of images, these algorithms have not proven to be suited for encryption of images. A new encryption approach termed as **Chaotic encryption** has found its way into the thick of things and has become a new cryptographic paradigm. The desirable properties for a good encryption algorithm are sensitivity to keys, confusion, and diffusion. All these properties found in chaotic maps which possess ergodic properties are susceptible to initial conditions and achieve this by iterating them. The main difference between these two techniques is that encryption operations are operated upon finite sets, whereas chaos is defined on real numbers.

The approach used in this paper is to deploy a chaotic map for permutation of the pixels and use Chen's diffusion approach [2] with the help of 1-D logistic map [10], finally making the image completely unrecognizable. The 2-D chaotic Arnold Map has desirable properties of confusion, but we have gone a step forward and used its 3-D counterpart that mathematically has greater mixing properties due to higher dimensionality and larger Lyapunov exponent. The diffusion approach uses the logistic map for generation of chaotic points. Both the processes work in tandem in the sense that the output from the permutation phase is passed on to the substitution phase every time the iteration increases, thus adding to the novelty of the algorithm and thereby strengthening the algorithm. The encryption scheme proposed by [13] also uses 3-D chaotic maps but the implementation is different from our scheme. They have used Arnold 3-D map on the blocks of pixels and used the matrix formed by operating the map for the purpose of diffusion process along with the original image. We have applied the 3-D map directly on the image and obtained a scrambled image. Then the diffusion step is applied with the help of logistic map on the scrambled image only.

1.1 3-D Cat Map

Arnold's cat map is a chaotic map from the torus into itself, named after Vladimir Arnold, who demonstrated its effects in the 1960s using an image of a cat, hence the name [1, 2, 4].

$$\Gamma \begin{bmatrix} i_{n+1} \\ j_{n+1} \end{bmatrix} = \left(\begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} i_n \\ j_n \end{bmatrix} \right) \mod 1 \tag{1}$$

The determinant of the matrix is equal to 1 that renders it area preserving. The eigenvalues λ_1 and λ_2 of the matrix in Eq. 1 are the Lyapunov characteristic exponents of the map.

$$\lambda_1 = \frac{3+\sqrt{5}}{2}, \qquad \lambda_2 = \frac{3-\sqrt{5}}{2} \tag{2}$$

Since the larger eigenvalue is greater than 1 in magnitude, the map possesses good mixing properties. The map is extended to the 3-D version [5–7] by first performing the 2-D operation in each of the three planes and then combining all the results as given below.

Keeping i_n untouched and applying 2−D map on the $j-k$ plane.

$$\begin{bmatrix} i_{n+1} \\ j_{n+1} \\ k_{n+1} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & a_i \\ 0 & b_i & a_i b_i + 1 \end{bmatrix} \begin{bmatrix} i_n \\ j_n \\ k_n \end{bmatrix} \tag{3}$$

Keeping j_n untouched and applying 2−D map on the $i-k$ plane.

$$\begin{bmatrix} i_{n+1} \\ j_{n+1} \\ k_{n+1} \end{bmatrix} = \begin{bmatrix} 1 & 0 & a_j \\ 0 & 1 & 0 \\ b_j & 0 & a_j b_j + 1 \end{bmatrix} \begin{bmatrix} i_n \\ j_n \\ k_n \end{bmatrix} \tag{4}$$

Keeping k_n untouched and applying 2−D map on the $i-j$ plane.

$$\begin{bmatrix} i_{n+1} \\ j_{n+1} \\ k_{n+1} \end{bmatrix} = \begin{bmatrix} 1 & a_k & 0 \\ b_k & a_k b_k + 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} i_n \\ j_n \\ k_n \end{bmatrix} \tag{5}$$

Combination of these three maps gives a 3-D cat map. The three maps are combined by multiplying the matrices in Eqs. 3, 4, 5.

$$\begin{bmatrix} i_{n+1} \\ j_{n+1} \\ k_{n+1} \end{bmatrix} = \left(A \begin{bmatrix} i_n \\ j_n \\ k_n \end{bmatrix} \right) \mod 1 \tag{6}$$

$$\begin{bmatrix} i_{n+1} \\ j_{n+1} \\ k_{n+1} \end{bmatrix} = \begin{bmatrix} 1 + a_i a_k b_j & a_k & a_j + a_i a_k + a_i a_j a_k b_j \\ b_k + a_i b_j + a_i a_k b_j b_k & a_k b_k + 1 & a_j a_k + a_i a_j a_k b_j b_k + a_i a_k b_k + a_i a_j b_j + a_i \\ a_i b_i b_k & b_i & a_i a_j b_i b_j + a_i b_i + a_j b_j + 1 \end{bmatrix} \begin{bmatrix} i_n \\ j_n \\ k_n \end{bmatrix} \tag{7}$$

1.2 Logistic Map

The logistic map is a polynomial mapping of degree 2, which is an equivalently recurrence relation. This is a popular example of explaining the complex, chaotic behavior that can arise from very simple nonlinear dynamical equations. The credit of popularization of this map goes to the biologist Robert May [8]. The mathematical definition of logistic map is given below:

$$x_{n+1} = \lambda x_n \left(1 - x_n\right) \tag{8}$$

Various chaotic encryption algorithms based on the chaotic properties of this map have been designed [11]. It has been found that for $\lambda = 4$, the map is chaotic [3, 10], and hence, this chaotic property is utilized in generating chaotic points in the diffusion phase. For initial seed value within (0, 1), the iterated values stay in (0, 1).

2 Encryption Scheme

The proposed encryption scheme is completed in three phases—permutation phase, key generation phase, and diffusion phase. A detailed description of each of the three phases is given in the following subsections.

2.1 Permutation Phase

Permutation is performed on the image considered for encryption in the following steps:

Step1: Consider a grayscale image of size $M \times N$. Total number of pixels in the image is MN. Convert the image into 3-D lattice of pixels with each side R pixels wide such that

$$R^3 = MN \tag{9}$$

If in case the number of pixels in the MN is not a perfect cube, try to do zero padding to the image to make it into a perfect cube. For example, the image used in this scheme is of size 512×512. So to convert it into a 3-D cube, the dimensions should be $64 \times 64 \times 64$.

Step2: After extraction of all R slices from the image, pile them up to form a cube and operate the 3-D cat map described above on the cube. This is the first part of the initial rounds of iteration. The 6 parameters of the 3-D map also serve as the key for encryption which will be evaluated in the key generation mechanism.

Note: In the 3-D cat map, the variables i_{n+1}, j_{n+1}, k_{n+1} will be the changed coordinates of the pixels after applying the transformation on the initial coordinates i_n, j_n, k_n. So all the 6 parameters of the transformation should be integer, because at the end what we will get is a changed coordinate which has to be an integer.

Step3: After performing the permutation above, convert the cube back to a 2-D array, i.e., a matrix.
Now collect the pixel values of the permuted matrix into 1-D array and prepare for the substitution phase.

2.2 Key Generation Mechanism

The key consists of the initial input of 16 characters by the user which is converted into a 128-bit-long key. The steps for generating the key are as follows:

Step1: Take the first eight bits of 128 bit key and XOR it with the last 8 bit of the key. Then take the 9th to 16th bit of the key and XOR it with 113th to 120th bit. Repeat the process till all the bits are exhausted. The computation of first two values for the key is demonstrated below:

$$key_1 = v(1:8) \oplus v(121-128)$$

$$key_2 = v(9:16) \oplus v(113-120)$$

$$key_3 = v(17:24) \oplus v(105:112)$$

$$key_4 = v(25:32) \oplus v(97:104)$$

$$key_5 = v(33:40) \oplus v(89:96)$$

$$key_6 = v(41:48) \oplus v(81:88)$$

$$key_7 = v(49:56) \oplus v(73:80)$$

$$key_8 = v(57:64) \oplus v(65:72)$$

where $v(i:j)$ means the starting point and the ending point chosen from the 128 bit string entered initially, and $\oplus$ is the bitxor operations between the two operands.

Step2: After getting all the 8 keys from the above procedure, assign the first 6 keys to the parameters of the 3-D map. Setting $a_i = key_1, a_j = key_2, a_k = key_3, b_i = key_4,$ $b_j = key_5, b_k = key_6$.

2.3 Diffusion

Step1: The diffusion phase starts with the choice of the initial seed to the logistic map. To make the initial seed dependent on the original image and the current round of iteration, use the seed as:

$$seed = c_1 = \frac{(\sum_{i,j} M_{i,j}) + itr}{M \times N \times 256} \tag{10}$$

where $M_{i,j}$ is the pixel value at location (i, j), M and N are the dimensions of the image, and itr is the current iteration number. The experiments have been done on square images of size 512, i.e., $M = N = 512$, and 256 is the maximum possible graylevels in the image.

Step2: The next step is to provide the seed to the logistic map to generate MN chaotic points, and these chaotic points are stored in the vector c. All the iterated values must fall in the interval (0.2, 0.8), and the value 0 or 1 must not be attained anywhere in the middle as it would imply further iterated values to be 0. To ensure this, after each iterated value is obtained, it is properly digitized by multiplying it with 10^{14} and scaled by taking it modulo 255.

$$d_i = \lfloor c_i * 10^{14} \rfloor mod\ 255, \qquad i = 2 \ldots MN \tag{11}$$

$\lfloor . \rfloor$ denotes the greatest integer function or the floor function.

$$w_1 = key_1 + key_2 + key_3 + key_4 + key_5 + key_6 + key_7 + key_8 \tag{12}$$

Step3: The governing encryption equation is as follows [2]:

$$w_i = di_i \oplus \{(v_i + di_i) \quad \text{mod } 255)\} \oplus w_{i-1} \tag{13}$$

where w_i = current diffused value, di_i = current digitized value, v_i = current pixel value that has been collected in the $1 - D$ array, and w_{i-1} = previously diffused value.

Step4: After all the diffused values have been obtained, convert the 1-D array into a 2-D matrix and replace the original image matrix with this diffused matrix. If more rounds of encryption are needed, just increase the iteration counter and repeat from *Step1* of the permutation round.

3 Security Analysis

A good encryption algorithm should be sufficiently robust against several kinds of statistical, cryptanalytic, and brute-force attacks. To prove the robustness of the proposed encryption algorithm against several kinds of attacks, security analysis is performed in this section. In particular, statistical analysis, differential attack analysis, sensitivity analysis with respect to the plaintext and key, etc., has been done. An encrypted image is said to be sufficiently robust to a differential attack, if a minute variation in the plain-image causes a considerable disparity in the cipher-image, i.e., even an iota of change in the original text spreads like an avalanche over the entire encrypted image. The sensitivity of the chaotic maps to initial conditions and parameters is responsible for this kind of effect. This phenomenon is termed as measure of the plaintext sensitivity [9] that can be computed using the following procedures. First of all, a plain-image is encrypted to a cipher-image C_1 and then a pixel is chosen randomly in the plain-image. The selected pixel is changed a little, say, a decrement or increment of 1 is made to its decimal value. For example, if the random pixel selected has a intensity value of say 162, then it is changed to 161. The modified image is now encrypted using the same key (used to encrypt the plain-image) to produce a new cipher-image C_2. A quantitative comparison is made between the two cipher-images using the number of pixel change rate (NCPR) [12]. A standard benchmark graylevel image 'Lena' of size 512×512 has been taken. NPCR is computed between Lena plain-image and its cipher-image using our encryption scheme and found to be more than 99%, thus proving the plaintext sensitivity of the proposed algorithm. The quantitative results with respect to different permutation and diffusion rounds are shown in Table 1, with m: permutation round and n: diffusion round.

1. **Key Sensitivity analysis**: The encryption must be highly sensitive to the variation in the encryption keys. Even a one bit difference in the key should cause large amount of diversity in the encrypted images. In our experiment on key sensitivity, key used initially was 'ASDFGHJKLZXCVBNM' and the changed key was 'ASDFGHJKLZXCVBNP'. The change in the encrypted images was above 99%, and in some cases, it was as much as 99.62%. The results of this experiment prove that the proposed encryption scheme is highly sensitive to key change.
2. **Number of pixels change rate (NPCR)**: This is a measure of plaintext sensitivity. We take the original image and form a new image from it by randomly selecting a pixel from the original image and increase/decrease its pixel value by 1. Encrypt both the images by the same key and observe the difference in the encrypted images. This test gives a percentage of the count of number of pixels that differ in value at corresponding locations in the two cipher-images. It can be formulated as given below [12]:

$$\mathbf{NPCR} = \sum_{j=1}^{N}\sum_{i=1}^{M} \frac{D(i,j)}{M \times N} \times 100$$

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

   where M and N are the width and height of the cipher-image. The numerical results are shown in Table 1.
3. **Unified average changing intensity (UACI)**: This is a measure of the average intensity of differences between the cipher-images C_1 and C_2, defined as below [12]:

Table 1 NPCR values for different rounds of encryption

(m, n)	Lian	Modified	Ours	Mao et al. [2]
(1, 1)	0.0179	68.6642	99.6136	≤50
(1, 3)	0.0252	99.4370	99.6128	≤50
(1, 4)	0.0423	99.6040	99.6128	≤50
(2, 2)	0.9903	99.6086	99.6086	≤50
(3, 3)	99.6063	64.7816	99.6082	≤50
(4, 4)	99.2676	99.6143	99.6220	≤50
(5, 5)	99.5892	99.5956	99.6056	≤50
(6, 6)	–	–	99.6281	–
(7, 7)	–	–	99.6140	–

Table 2 NPCR values for different rounds of encryption

(m, n)	Lian	Modified lian	Ours	Chen, Mao, Chui [2]
(1, 2)	0.0040	20.8793	33.4266	≤25.3
(1, 3)	0.0061	32.8191	33.4031	≤25.3
(1, 4)	0.0093	33.4905	33.4031	≤25.3
(2, 2)	0.2623	33.4273	33.4365	≤25.3
(3, 3)	17.6647	33.5562	33.4379	≤25.3
(4, 4)	31.7068	33.4972	33.4360	≤25.3
(5, 5)	32.7871	33.3184	34.4776	≤25.3
(6, 6)	–	–	33.3901	–
(7, 7)	–	–	33.3962	–

$$\textbf{UACI} = \sum_{j=1}^{N}\sum_{i=1}^{M} \frac{|C_1(i,j) - C_2(i,j)|}{L} \times 100$$

where L is the number of possible graylevel values. As the experiment is random in nature, i.e., the pixel chosen for modification is random, the experiment should be repeated suitably large number of times with diverse images. The ideal UACI value should be near 33 for the encryption to resist differential attacks. The numerical results for the $UACI$ values are given in Table 2 and are found to be favoring the proposed encryption scheme.

3.1 Statistical Analysis

An encryption algorithm must be robust to statistical attacks as well. The following measures are defined to evaluate the statistical robustness of the encryption algorithm.

1. **Histogram analysis**: The histogram is plotted for the encrypted image and checked for uniformity. The histogram of the encrypted image should be sufficiently uniform to resist statistical attack. A uniform histogram indicates that there is uniform probability of finding any pixel value in the image. The histograms plotted for plain-image and cipher-image are shown in Figs. 1 and 2. It is clearly visible from the figures that the histograms of cipher-images are very much uniform (Fig. 3).

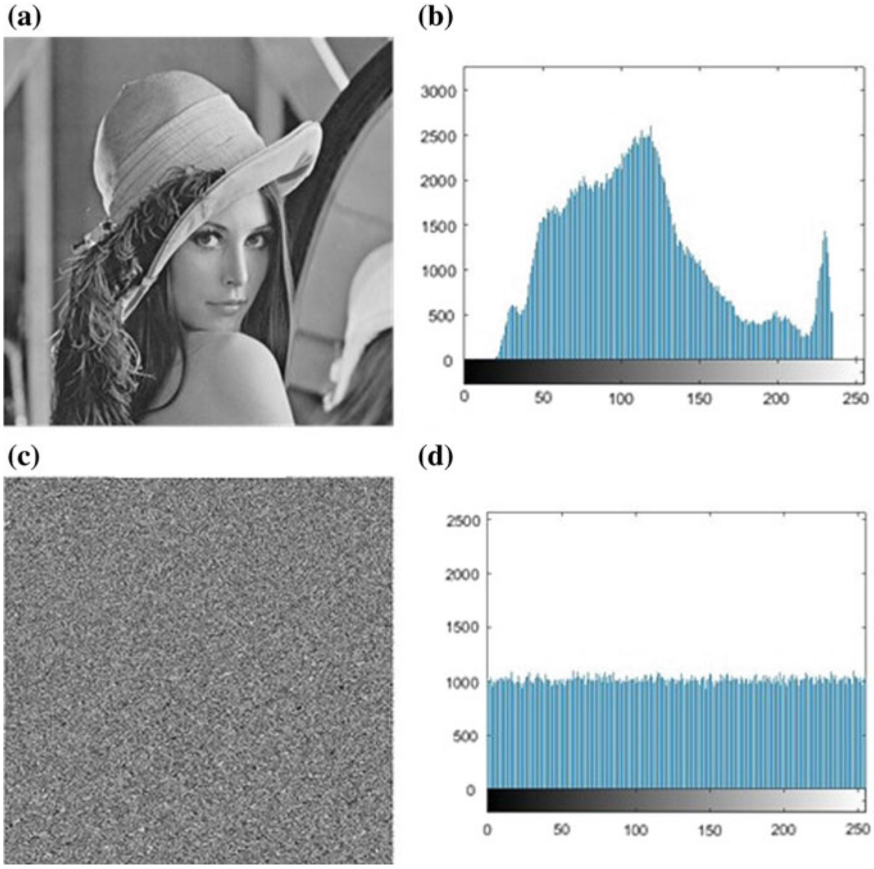

Fig. 1 Encryption results: **a** original image; **b** histogram of original image; **c** encrypted image; **d** histogram of encrypted image

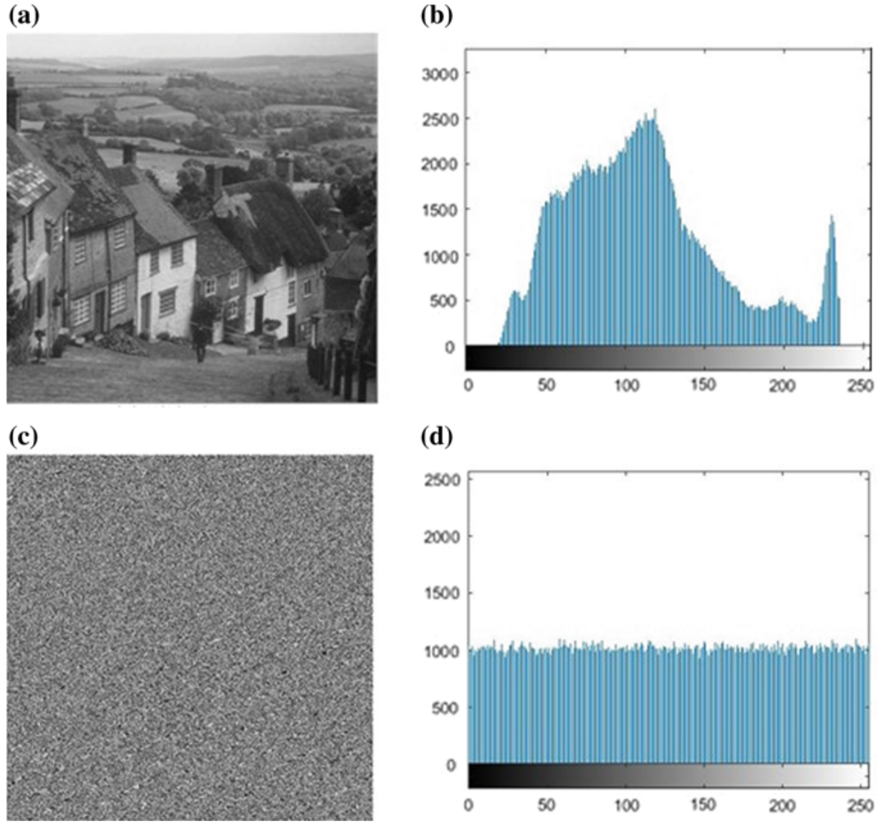

Fig. 2 Encryption results: **a** original image; **b** histogram of original image; **c** encrypted image; (d) histogram of encrypted image

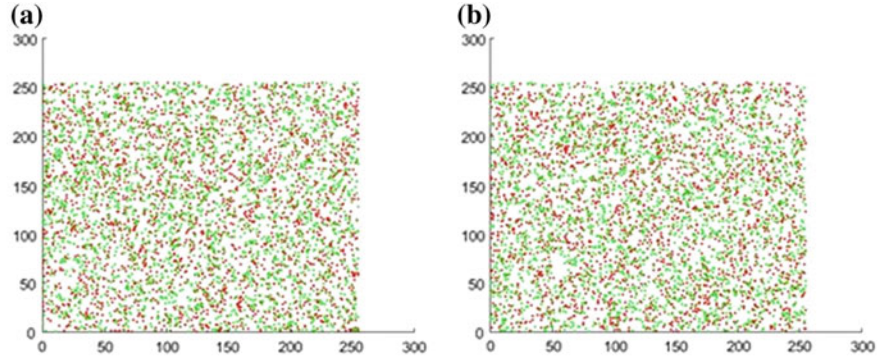

Fig. 3 Encryption results: **a** Correlation of pixels in horizontal direction; **b** Correlation of pixels in vertical direction

Table 3 Entropy values of two different images

Rounds	Lena image	Scenery image
(1, 1)	7.9992	7.9993
(2, 2)	7.9992	7.9993
(3, 3)	7.9993	7.9994
(4, 4)	7.9992	7.9992
(5, 5)	7.9994	7.9994
(6, 6)	7.9993	7.9992
(7, 7)	7.9993	7.9992
(8, 8)	7.9992	7.9992

2. **Entropy**: The entropy of a system as defined by 'Shannon' gives a measure of uncertainty about its genuine structure. Let t be the message source, then the entropy $H(t)$ is defined as below:

$$H(t) = -\sum_{i=1}^{S} p(t_i) \times \log_2 t_i$$

where S is the total number of symbols, which in our case is 256, $p(t_i)$ stands for the probability of occasions of symbol t_i, and the $\log_2$ is used so that the entropy is measured in bits. An entropy more closer to the figure $\log_2 L$ bits implies more uniform histogram that in turn ensures the security of encrypted information. A quantitative evaluation is given in the Table 3. From the table, it is clear that the entropy values obtained are very close to $\log_2 256 = 8$, which again indicates a uniform histogram and hence encrypted information is secure enough.

3. **Correlation of adjacent pixels**: The correlation between the pixels of an encrypted image should be close to zero. To compute the correlation between two adjacent pixels, select P number of pairs of horizontally/vertically adjacent pixels randomly from the image. Compute the correlation coefficient ρ for each pair using the following equations:

$$\rho = \frac{cov(u, v)}{\sigma_u \cdot \sigma_v}$$

where u and v are the values of the adjacent pixels.

$$cov(u, v) = \mathbf{E}\{(u - \mathbf{E}(u))(v - \mathbf{E}(v))\}.$$

$$\mathbf{E}(u) = \frac{\sum_{i=1}^{P} u_i}{P}.$$

Table 4 Horizontal correlation between the pixels in the encrypted image

Rounds	Lena image	Scenery image
(1, 1)	−0.2439	−0.0237
(2, 2)	−0.1357	−0.0079
(3, 3)	−0.0796	−0.0120
(4, 4)	−0.0716	−0.0094
(5, 5)	0.0480	0.0269
(6, 6)	0.0459	−0.0434
(7, 7)	−0.0299	−0.0344
(8, 8)	−0.0230	−0.0278
(20, 20)	−0.0060	−0.0058

Table 5 Vertical correlation between the pixels in the encrypted image

Rounds	Lena image	Scenery image	Wong [6] for Lena image
(1, 1)	0.0108	0.0450	−0.06538
(2, 2)	0.0211	−0.0139	−0.06538
(3, 3)	−0.0439	0.0214	−0.06538
(4, 4)	−0.0115	−0.0163	−0.06538
(5, 5)	−0.0129	0.0248	−0.06538
(6, 6)	0.0205	0.0478	−0.06538

$$\sigma_u^2 = \frac{\sum_{i=1}^{P}(u_i - \mathbf{E}(u))}{P}$$

The results are given in Tables 4 and 5 for $P = 5000$. Correlation between the horizontal adjacent pixels and correlation between the vertical adjacent pixels have been plotted to visualize the outcome as shown in Fig. 3.

4. **Parameter sensitivity**: The parameters in this algorithm are the entries of the Arnold 3-D map and the seed value to the logistic map. These values in turn depend on:
   (a) The user entered key :If the key entered is altered by even a single bit, then the key scheduling algorithm gives an entirely different digest of the input that is further processed by the maps.
   (b) The size of the image : The size of the image plays an important role since for our encryption, the basic need is that the size of the image should be a perfect cube in order to apply the 3-D map. Secondly, the sum of all the pixels is one more key aspect in our encryption algorithm which in turn depends on size.
   (c) To check the parameter sensitivity, the UACI and the NPCR tests have already been performed, but we try to give a correlation test analysis as

Table 6 Correlation of original cipher with incorrect cipher image

Rounds	Correlation of C_1 and C_2
(1, 1)	−0.0050
(2, 2)	−0.0025
(3, 3)	0.0029
(4, 4)	0.0021

well. We encrypt the same image by two different sets of keys and find the correlation between the two images. The results in Table 6 show that the correlation between the two cipher images C_1 and C_2 is close to zero, where C_1 is the original cipher and C_2 is the changed cipher, thereby strengthening our claims of parameter sensitivity.

4 Conclusion

As a concluding note, we would like to summarize the proposed algorithm's pros and some cons that are seen from the numerical results. Firstly, the use of 3-D map makes it more topologically mixing and thereby producing more confusion in lesser number of rounds. The NPCR results show that the sensitivity to plaintext is around 100%, i.e., if there is even the slightest change in the plaintext, the final encrypted text will be approximately 100% changed. The UACI results show that there is $>33.3\%$ change in the final encrypted image if the original image is changed. The encryption approach shows sensitivity to the key also. Even one bit change in the encryption key can produce a difference over 99% in the cipher-image. The entropy value is close to 8, which shows that in the final encrypted image, each pixel has maximum amount of information content in it, i.e., $\log_2 L$ bits. The correlation between the pixels horizontally initially is not close to 0 but as the number of rounds increases, the correlation gets reduced subsequently and tends toward 0, thereby showing improvement of the scheme. Hence, looking over all results, we have got some concrete and substantial proofs to think that our encryption approach can withstand the major cryptographic attacks and can serve as a secure image encryption approach.

References

1. Vladimir I. Arnold and A. Avez. Ergodic problems in classical mechanics. *ZAMM-Journal of Applied Mathematics and Mechanics/Zeitschrift fr Angewandte Mathematik und Mechanik*, 506(7–9):506, 1970.
2. G. Chen, Y. Mao, and C.K Chui. A symmetric image encryption scheme based on 3d chaotic cat maps. *Chaos, Solitons and Fractals*, 21:749, 2004.
3. Wikipedia contributors. Logistic map. *Wikipedia, The Free Encyclopedia*, (16 February 2017).
4. J. Fridrich. Symmetric ciphers based on two-dimensional chaotic maps. *Int. J. Bifurcat Chaos*, 8:1259–1284, 1998.

5. L. Kocarev, Z. Galias, and S Lian. *Intelligent Computing Based on Chaos*, volume 184. Springer, 2008.
6. H.S. Kwok and K.S. Tang Wallace. A fast image encryption system based on chaotic maps with finite precision representation. *Chaos, Solitons and Fractals*, 32:1518–1529, 2007.
7. Y. Mao, G. Chen, and S. Lian. A novel fast image encryption scheme based on 3d chaotic baker maps. *Int. J. Bifurcat. Chaos*, 14(10):3613– 3624, 2004.
8. R. May. Simple mathematical models with very complicated dynamics. *Nature*, 261(5560):459–467, 1976.
9. A. Mohamed. Efficient modified rc5 based on chaos adapted to image encryption. *Journal of Electronic imaging*, 19(1):013012, March 2010.
10. W. Morris, S. Hirsch, R. Smale, and L. Devaney. *Differential Equations, Dynamical Systems, And An Introduction To Chaos*. Elsevier Academic Press, 2004.
11. N.K Pareek, V. Patidar, and K.K Sud. Image encryption using chaotic logistic map. *Image and Vision Computing*, 24:926–934, 2006.
12. Y. Wu, P. Noonan Joseph, and A. Sos. Npcr and uaci randomness tests for image encryption. *Multidisciplinary Journals in Science and Technology, Journal of Selected Areas in Telecommunications (JSAT)*, April Edition, 2011.
13. Zhou Zhe, Yang Haibing, Zhu Yu, Pan Wenjie, and Zhang Yunpeng. A block en-cryption scheme baes on 3D chaotic arnold maps. In 2009 International conference on Intellegent Interaction and Affective Computing, pages 15–20, Wuhan, TBD, China, 2009. IEEE.