Multimedia Tools and Applications (2019) 78:27569–27590
https://doi.org/10.1007/s11042-019-07852-2

A pixel permutation based image encryption technique using chaotic map

Shamama Anwar[1] · **Solleti Meghana**[1]

Received: 30 October 2018 / Revised: 28 April 2019 / Accepted: 31 May 2019 /
Published online: 24 June 2019

Abstract
In the last decade, with a rapid increase in multimedia productions, image encryption has become a significant part of information security. The inherent image features make image encryption different from text encryption and also makes it difficult to encrypt images using the traditional encryption techniques. This paper presents an overview of the different encryption algorithms in details, analyzing its effect in the field of image cryptography. The paper further discusses the role of chaos theory in the field of image encryption and makes use of the chaos theory to propose an image encryption technique that is based on pixel permutation. Existing image encryption techniques produce encrypted images that resemble a texture or noise like image increasing the chance of attack as it becomes perceivable as an encrypted image. The proposed image encryption technique produces an encrypted image and masquerades it with any general image, hence eluding the attacker. The proposed algorithm is also compared with the Arnold's cat map algorithm visually as well as statistically using the evaluation metrics Structural Similarity Index (SSIM), Correlation coefficient (CC) and Number of changing Pixel Rate (NPCR).

Keywords Image encryption · Arnold's cat map · SSIM · Correlation coefficient · NPCR

1 Introduction

With the digitization drive attaining momentum in the current world scenario, security has become an utmost concern. Organizations and individuals are inclined to share paramount information online for ease, convenience, and speed. Safeguarding these information or data is a concern that has been an extensive research area since the inception of internet in the

✉ Shamama Anwar
shamama@bitmesra.ac.in

[1] Department of Computer Science and Engineering, Birla Institute of Technology, Mesra, Ranchi, 835215 India

1990s. The information that is shared should be in such a form that even if an unauthorized person gains access to it, it remains unintelligible or obscure to him. A long-used custom to share texts in secret has been to encrypt it. This is referred to as cryptography which is derived from the Greek word meaning "secret writing"[37]. Cryptography consists of two phases: Encryption and Decryption. Encryption refers to the process of converting plain, ordinary text into unintelligible text and Decryption is extracting the plain text back from the unintelligible format. The purpose of encryption is to prevent the text from being read from unauthorized person. Encryption does not itself prevent interference but prevents readability of the content to a would-be interceptor. In an encryption scheme, the intended information or message, referred to as plaintext, is encrypted using an encryption algorithm generating ciphertext that can only be read if decrypted using a key. The key is a secret which is ideally known to the sender and receiver of the message. It is usually a short string of characters or numbers. A lot of algorithms has been proposed for the encryption and decryption method. The most prominent ones are the Data Encryption Standard (DES), Triple Data Encryption Standard (TDES), Simplified Data Encryption Standard (SDES), Advanced Encryption Standard (AES), Modified Version of Advanced Encryption Standard (MAES), Rivest - Shamir - Adleman (RSA), Blowfish, etc.

Predominantly encryption conveys the concept of encoding and decoding text messages or documents. Besides the limitless number of documents being shared, images are also a prime bearer of crucial information. Hence, there is a requirement to encrypt images as well. Encrypting images require the pixels of the images to be manipulated so that they do not represent the original image any more. After the receiver receives the image, the image goes through the decryption phase to regain the original image back. Encrypting images insures that even if an interceptor gets access to the image, during transmission, it is unintelligible to him. Another practical utilization of encrypting images is safeguarding the biometric data. With the increase of biometric identification involving finger print and retina scans, there is a requirement to share and store these data securely. Encrypting them insures better security mechanism as even if a malicious access is made it is unintelligible to the intruder unless he can decrypt it. Further, in the medical field to ensure privacy of a patient, his medical reports and scans can be saved in the encrypted format [26]. Similarly, it can also be used for securing multimedia applications[7].

The main contribution of the paper is an image encryption technique that produces an encrypted image and masquerades it with any general image (the key image), hence eluding the attacker. General existing image encryption techniques produce encrypted image that resembles a noisy or pattern like image. Any intruder that gets access to these images perceives that these images are encrypted and hence store vital information. On the other hand, the images generated by the proposed algorithm are visually pleasing as they can be masqueraded into any image and hence the intruder is unable to perceive that these images are encrypted.

The remainder of the paper is organized as: Section 2 contains an overview of the existing encryption technique. Section 3 extends the encryption algorithm for image encryption and includes the proposed image encryption technique. Section 4 discusses the parameters for evaluating the strength of the encryption algorithm and finally section 5 shows the implementation results of the proposed technique and Arnold's cat map technique for image encryption. It also includes the statistical analysis of the both the techniques based on the performance parameters.

Table 1 Permuted choice

14	17	11	24	1	5	3	28
15	6	21	10	23	19	12	4
26	8	16	7	27	20	13	2
41	52	31	37	47	55	30	40
51	45	33	48	44	49	39	56
34	53	46	42	50	36	29	32

2 Background

One of the earliest algorithm introduced for encryption was the Data Encryption Standard commonly known as the DES algorithm. DES is a block cipher that uses a shared secret key for encryption and decryption. It is an iterative cipher mechanism that has 16 rounds of steps. It has a 56-bit key K out of which 48 bits are used for encryption and other 8 bits are parity bits. The bit position 8, 16, 24, 32, 40, 48, 56 and 64 are the parity bits and do not form a part of the key (Table 1). The 48-bit key is divided into two halves C and D. For each round i, $(1 \leq i \leq 16)$ of the DES algorithm, the key K_i is generated as in Algorithm 1. The key is known to the sender and receiver using which the sender encrypts the text and transmits it to the receiver.

Images are an array of pixel arrangements. The DES algorithm receive as input 64 pixel values from an 8×8 sliding window. For encryption, the input pixels are shuffled based on Table 2. The input is then divided into two halves L and R of 32 bits each. A feistel function *F(R,K)* is used for encryption. The feistel function first expands R into 48 bits using an expansion D box as in Table 4 (Algorithm 3) and then uses S boxes (*Substitution boxes*) to do the real mixing [9]. A total of 8 S - boxes are used. The 48 bit input into the S box is divided into 8 blocks of 6 bits each. Each S - box takes a 6-bit input $(a_1, a_2, a_3, a_4, a_5, a_6)$ and produces a 4-bit output (b_1, b_2, b_3, b_4). Each S box comprises of 4 rows and 16 columns, and every entry in it is a 4-bit number (Table 5). The first and last bit of the input (a_1, a_6) combine to form the row number and the other bits (a_2, a_3, a_4, a_5) represent the column number of the S box. The number corresponding to the S box entry is chosen as the 4 bit output (b_1, b_2, b_3, b_4) [16, 37]. The algorithm for the feistel function is represented in Algorithm 3 and the DES algorithm in Algorithm 2 (Table 3).

Table 2 Initial permutation

58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

Table 3 Final permutation

40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

Decryption uses a similar concept, but the keys are used in reverse order. The key K_{16} is used in the first round and so on. The DES algorithm exhibits a strong avalanche effect i.e. a minimal modification in the original image will radically change the cipher image but DES fails in front of linear crypt-analysis, because during its design this attack was not invented. The DES algorithm is not very secure due to its compact key size. Moreover, now in the era of parallel computing, deciphering DES has become unchallenging with the help of brute force attack which was infeasible at the time of its conception. Due to these restraints, various extensions to the traditional DES algorithm has been presented.

Simplified-Data Encryption Standard (S-DES) is a reduced adaptation of the Data Encryption Standard (DES) algorithm. The Simplified DES algorithm is similar to the structure of the DES but uses more simpler specifications. It is also a block cipher which works on blocks of 8 bits and uses a 10-bit initial key which goes through a series of permutation and shifting to generate keys for the different rounds. The algorithm for key generation is given in Algorithm 4. An initial permutation is performed followed by a Mangler function which takes 8 bits as input. The rightmost 4 bits of the input are left unchanged and the leftmost 4 bits are mangled. The switch function then follows, the Mangler function works again using the second key and ends with the reverse permutation to get the encrypted image [18, 33]. There are 2 *S boxes* (Table 6) having four rows and four columns and each entry is a 2-bit value. In Algorithm 6, the first and last bit of Y_1 serve as the row number and the other two bits as the column number. The corresponding entry in the *S box* is the output depicted by (q_0, q_1) for S_1 and (q_2, q_3) for S_2 respectively.

Table 4 Expansion permutation

32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

Algorithm 1 Key generation for DES algorithm.

Input: 64 bit initially chosen key K
Output: Keys K_i for each round i ($1 \leq i \leq 16$) of the DES Algorithm
Initialize: $K_i = K$ where i = 1
for i =1 to 16 **do**
  Select 48 bits from K_i (leaving bit 8, 16, 24, 32, 40, 48, 56, 64 for parity)
  Divide K_i into two C_i and D_i where
  C_i= bits at position (57,49,41,33,25,17,9,1,58,50,42,34,26,18,10,2,59,51,43,35,27,19, 11,3,60,52,44,36)
  D_i= bits at position (63,55,47,39,31,23,15,7,62,54,46,38,30,22,14,6,61,53,45,37,29, 21,13,5,28,20,12,4)
  if i= 2 or 9 or 16 **then**
    Left shift C_i and D_i by 1 bit
  else
    Left shift C_i and D_i by 2 bits
  end if
  The final key for the next round is constructed using the PC table (Table 1)
end for

Algorithm 2 DES algorithm.

Input: Input image I, key K for each round constructed using Algorithm 1
Output: The encrypted Image E
for i =1 to 16 **do**
  Shuffle the pixels in I using the permutation in Table 2 to obtain I'
  Split I' into two halves L_i and R_i
  $R_i = L_i \oplus F(R_i,K_i)$ and $L_i = R_i$ where F is computed from Algorithm 3
end for
swap(L, R)
Reshuffle the bits using Final Permutation in Table 3 to obtain E

Algorithm 3 Algorithm for fiestal function of DES.

Input: 48 bit key K and 32 bit R
Output: 32-bit output Y
Expand R to 48 bits using Table 4
$X = R \oplus K$
Split 48 bits of X into 8 blocks (X_1,X_2,X_3, ..., X_8) of 6 bits each
for i =1 to 8 **do**
  $Y_i = X_i \rightarrow S_i$ where S represents the S box (Table 5)
end for
$Y = \bigcup Y_i$ where i=1 to 8

Table 5 S boxes

14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13
(a) S1															
15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9
(b) S2															
10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12
(c) S3															
7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14
(d) S4															
2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3
(e) S5															
12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13
(f) S6															
4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
13	0	11	7	4	9	1	10	4	3	5	12	2	15	8	6
1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12
(g) S7															
13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11
(h) S8															

Algorithm 4 Key Generation algorithm for S-DES Algorithm

Input: 10 bit initially chosen key K (k_0,k_1,k_2,k_3,k_4,k_5,k_6,k_7,k_8,k_9)
Output: K_1, K_2
Apply permutation (2,4,1,6,3,9,0,8,7,5) to K get the key as
K' (k_2,k_4,k_1,k_6,k_3,k_9,k_0,k_8,k_7,k_5)
Split K' into two K_L' = (k_2,k_4,k_1,k_6,k_3) and K_R' = (k_9,k_0,k_8,k_7,k_5)
Perform left shift by 1 bit on K_L' and K_R' to obtain
K_L'' = (k_4,k_1,k_6,k_3,k_2) and K_R'' = (k_0,k_8,k_7,k_5,k_9)
Merge K_L'' and K_R'' to obtain K'' = (k_4,k_1,k_6,k_3,k_2,k_0,k_8,k_7,k_5,k_9)
Apply permutation (5,2,6,3,7,4,9,8) to K'' to obtain a 8 a bit key
K_1= (k_0,k_6,k_8,k_3,k_7,k_2,k_9,k_5)
Split K'' into two K_L'' = (k_4,k_1,k_6,k_3,k_2) and K_R''= (k_0,k_8,k_7,k_5,k_9)
Perform left shift by 2 bits on K_L'' and K_R'' to obtain
K_L'''= (k_6,k_3,k_2,k_4,k_1) and K_R''' = (k_7,k_5,k_9,k_0,k_8)
Merge K_L''' and K_R''' to obtain K'''= (k_6,k_3,k_2,k_4,k_1,k_7,k_5,k_9,k_0,k_8)
Apply permutation (5,2,6,3,7,4,9,8) to K''' to obtain a 8 bit key
K_2= (k_7,k_2,k_5,k_4,k_9,k_1,k_8,k_0)

Algorithm 5 S-DES algorithm.

Input: 8-bit input image I, key K_1,K_2 for each round constructed using Algorithm 4
Output: The encrypted image E
Shuffle the bits in I using the permutation (3,0,2,4,6,1,7,5) to obtain
I'=(I_3,I_0,I_2,I_4,I_6,I_1,I_7,I_5)
X_1=M(I',K_1) where M( ) is the Mangler function given in Algorithm 6
Switch the first four and last four bits of X_1 to obtain
X_1=($x_{1,0}$,$x_{1,1}$,$x_{1,2}$,$x_{1,3}$,$x_{1,4}$,$x_{1,5}$,$x_{1,6}$,$x_{1,7}$)
X_2=M(X_1,K_2)
Reshuffle the bits using Final Permutation (1,5,2,0,3,7,4,6) to obtain E

Algorithm 6 Algorithm for mangler function of S-DES.

Input: 8-bit key K (k_0,k_1,k_2,k_3,k_4,k_5,k_6,k_7) and 8-bit Y (y_0,y_1,y_2,y_3,y_4,y_5,y_6,y_7)
Output: 8-bit output Z
Obtain Y_1= (y_7+k_0,y_4+k_1,y_5+k_2,y_6+k_3) and Y_2=(y_5+k_4,y_6+k_5,y_7+k_6,y_4+k_7)
(q_0,q_1)= $Y_1 \rightarrow S_1$ and (q_2,q_3) = $Y_2 \rightarrow S_2$ where S_1 and S_2 are S box (Table 6)
Apply permutation (1,3,2,0) on Q = (q_0,q_1,q_2,q_3) to obtain Q'

$$Z = \left(\sum_{i=0}^{3} (y_i + q_i'),\ y_4,\ y_5,\ y_6,\ y_7 \right)$$

Table 6 S box

(a) S1			
1	0	3	2
3	2	1	0
0	2	1	3
3	3	1	2
(b) S2			
2	3	1	1
1	2	3	1
1	1	2	3
3	1	1	2

Another variation of the DES algorithm is the Triple DES. Triple DES (3 DES) is an enhancement of DES; it has a 64-bit block size with 192 bits key size. This encryption method is same as original DES but applied 3 times to increase the encryption level and the average safe time. Initially, three different keys are used for the encryption algorithm to generate cipher image on the original image. This makes TDES a strong encryption algorithm since it is extremely hard to break 2^{168} possible combinations but is slower in comparison to the other variations of DES [5, 28].

A more advanced encryption standard is the Advanced Encryption Standard (AES). The length of both the block to be encrypted and the encryption key are not fixed. They can be independently specified as 128, 192 or 256 bits. The number of rounds, however, varies according to the key length. It can be equal to 10, 12 and 14 when the key length is 128 bits, 192 bits and 256 bits, respectively [31, 35]. Algorithm 7 represents the algorithm for standard AES. The first step is to determine the block size which is the number of column where each row in the column consists of 4 cells of 8 bytes each, hence making each row of 32 bits. For AES-128, the block size (B_s) will be equal to $(128/32) = 4$. The pixels are arranged in an array of size $B_s \times B_s$. Similarly, the key size (K_s) will also be equal to 4, hence that is also arranged in an array of size $K_s \times K_s$. Next the number of rounds for AES-128 is 10. It is chosen as $r = 6 + max(B_s, K_s)$, which is 10. After the key is expanded, each entry of the block represents the row and column number in the *S box*. The corresponding value is replaced in the image *I*. AES consists of four transformation: (i) S box substitution; (ii) Row Shifting; (iii) Column mixing, which is achieved by using modulo multiplication using Galios Field (GF) represented as: $\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix}$; and (iv) Adding the round key.

A Galios Field (GF) is useful for cryptography because its arithmetic properties allows it to be used for scrambling and descrambling of data. It can be combined with other bit shuffling techniques, resulting in strong symmetric algorithms such as the AES. A Galois field $GF(p^n)$, is a finite field with p^n members, where p is a prime and n is a positive integer. The AES encryption technique uses a $GF(2^8)$. AES has also been used with other soft computing techniques to give rise to robust encryption schemes meant for the vast multimedia data [2, 3].

Algorithm 7 AES algorithm.

Input: 128-bit input image I and key K
Output: The encrypted image E
Determine B_S (block size) and K_S (key size)
Find (number of rounds) r = 6 + max (B_S,K_S)
Key Expansion (Algorithm 8)
RK_1 = I⊕K
for i = 1 to r-1 **do**
I_1 = I → S
for j = 1 to 3 **do**
Shift Row_j by j bytes to obtain I_2
end for
I_3= I_2×GF where GF is the Galios Field
I_4 = I_3 (C_i) ⊕ RK_i (C_i) where C_i represents the column number and RK_i represents the round key generated in Algorithm 8.
end for

Blowfish is another encryption technique that uses a 64-bit symmetric block cipher and has a variable-length key from 32 to 448-bits (14 bytes). As with DES, Blowfish incorporates a 16 round Feistel network for encryption and decryption for which it requires 18 keys (16 keys for the 16 fiestal rounds and 2 for the final round). But during each round of Blowfish, the left and right 32-bits of data are modified unlike DES which only modifies the right 32-bits to become the next round's left 32- bits. Blowfish incorporates a bit wise exclusive-or operation to be performed on the left 32-bits before being modified by the *F* function or propagated to the right 32-bits for the next round. Blowfish also uses two exclusive-or operations to be performed after the 16 rounds and a swap operation [25, 34].

Algorithm 8 Key expansion for AES algorithm.

Input: Key K
Output: Round Key (RK_1, RK_2,...., RK_r)
for i = 1 to r **do**
Rotate K(C_4) by 1 bit to obtain vector V
V′ = V→S
$RK_i(C_i)$ = K(C_1)⊕V′⊕Rcon(4) where Rcon = $[2^{(x-1)}, 0, 0, 0]'$
for j = 2 to 4 **do**
RK_i (C_j) = K(C_j) ⊕ RK_i($C_{(j-1)}$)
end for
K =RK_i
Output RK_i
end for

Algorithm 9 Key generation algorithm for Blowfish.

Input: 32 bit initially chosen values (P_1,P_2,....,P_{18}), n bit C (where n is a multiple of 32)
Output: Keys K_i for each round i ($1 \leq i \leq 18$) of the Blowfish Algorithm
Divide C into m block of 32 bits each (C_1, ..., C_m) (where m = n / 32)
j = 1
for i =1 to 18 **do**
    $K_i = P_i \oplus C_j$
    if ($j \geq m$) **then**
        j = 1
    else
        j = j+1
    end if
end for

Algorithm 10 Blowfish algorithm.

Input: Input image I, key K for each round constructed using Algorithm 9
Output: The encrypted image E
Divide I in to two equal portions I_L and I_R
for i =1 to 16 **do**
    $I_L = I_L \oplus K_i$
    $I_R = F(I_L) \oplus I_R$
    swap(I_L,I_R)
end for
swap(I_L,I_R)
$I_R = I_R \oplus K_{17}$
$I_L = I_L \oplus K_{18}$
E is obtained by concatenating I_L and I_R

Algorithm 11 Fiestal function algorithm for Blowfish.

Input: n bit input X
Output: n bit output Y
Divide X into blocks containing 8 bits each
for i = 1 to n / 8 **do**
    $y_i = x_i \rightarrow S_i$ where S represents the S box
end for
$Y = ((y_1+y_2) \bmod 2^n \oplus y_3) + y_4 \bmod 2^n$

The algorithms described above are the symmetric key algorithms that use the same key for encryption and decryption. The decryption process follows a similar approach but in the reverse sequence to get the initial image back. Another category of encryption uses

two divergent keys for encryption and decryption which are termed as Asymmetric key algorithms.

A prevalent one amid them is the RSA algorithm. It was proposed by Ron Rivest, Adi Shamir and Leonard Adleman. It requires generating two large distinct prime numbers p and q. The pair *(e,n)* forms the public key, where $n = p \times q$ and e is a co-prime to ϕ(n) where $\phi(n) = (p-1)(q-1)$ with $1 \leq e \leq \phi(n)$. Similarly, the pair *(d,n)* forms the private key where d is an integer and is calculated as the modular inverse of e with respect to $\phi(n)$ and $e \times d \ mod(n) = 1$. For encryption the image is divided into a succession of blocks. These blocks are then encrypted as $C \equiv E(I) \equiv I^e (mod\ n)$ where I is the message. For decryption the private key is used as $D(C) \equiv C^d (mod\ n)$[16, 32]. RSA is secure and reliable since it uses factorization of prime numbers but that also makes RSA slow in cases where large data needs to be encrypted.

Algorithm 12 Key generation for RSA.

Input: Two large prime numbers p and q
Output: Public and Private Key pair
Calculate RSA modulus as n = p×q
Calculate totient of RSA modulus as ϕ(n) = (p-1)(q-1)
Select e such that $1 \leq$ e $\leq \phi$(n) and gcd(e,ϕ(n))=1
Calculate d as modular inverse of e w.r.t ϕ(n) as: e×d mod (n)=1
Private key = (d,n)
Public key = (e,n)

3 Image encryption techniques

Image cryptography has become a dominant research discipline and it has broad application prospects. Innumerable image encryption algorithms have been proposed which safeguards the image from profuse attacks. Using the encryption technique, the input image is transformed into another image, so that the transformed image is difficult to apprehend by unauthorized person and hence maintain its secrecy. Image encryption is different from text encryption due to some inherent features of image (such as bulk data capacity and high correlation among pixels). The existing encryption techniques have been applied to image encryption as well [38, 47]. Based on the study and analysis of the different encryption techniques, it can be categorized into three board spectrum: Value transformation, Pixels position permutation and Chaotic systems. Value transformation technique changes or transforms the original value of the pixel to get an encrypted image. Pixel position transformation methods are based on the ideology of repositioning or shuffling the pixels in the original image to obtain a new image which later becomes the encrypted image. Chaotic map, on the other hand, produces a chaotic sequence that controls the encryption process.

3.1 Value transformation

In unprocessed images, neighboring pixels are highly correlated. Owing to this correlation any pixel value can be predicted from their neighboring pixels. Therefore, in image encryption it becomes desirable to diminish this correlation. Value transformation algorithms are

based on the technique in which the value of each pixel is changed to some other value. These algorithms involve some mathematical computation to transform the image pixels. Following the same concept, A. Nag et.al., in their work, redistributed the pixel values to diverse positions using affine transform technique followed by XOR operation with four 8 bits keys. They experimentally proved that after these transformations, the correlation between the pixels remarkably declined [24]. Other methods include using (i) AND operation followed by random permutation shuffling [1]; (ii) fractional Mellin Transformation which was proven to outperform the conventional encryption methods [49] and (iii) random phase encoding using Fourier transform followed by scrambling [22].

3.2 Pixels position permutation

Transposition means repositioning elements in the image. This repositioning of elements can be done at the bit, pixel or block level. In the bit permutation technique, the B bits in each pixel are permuted using the permutation key. Bit level encryption utilizes bit-level decomposition and stacking operations to both encrypt and decrypt an image. A perfect reconstruction is achieved by performing decryption through simple logical operations in the decomposed bit-levels without the need for any post processing operations [23, 50]. Similar use of bits transformation can been seen in [12, 21, 48]. The pixel level shuffling method for image encryption shuffles the original pixel position to obtain an encrypted image. These methods are generally used in combination of other methods like chaotic systems [11, 14, 15], compression [29], etc. In a block-based transformation the original image is divided into blocks. The blocks are first transformed followed by the actual encryption process [10, 13, 19]. The encryption process may use a secret key, and since it is a block encryption scheme the output of encrypting process may serve as the key for encrypting the next block, hence increasing the security [30].

3.3 Chaotic systems

Chaotic maps describe the behavior of certain nonlinear dynamic system that under specific conditions exhibit dynamics that are sensitive to initial conditions. The two basic properties of chaotic systems are the sensitivity to initial conditions and mixing property. Chaotic map is used to produce the chaotic sequence that controls the encryption process. The chaos streams are generated by using various chaotic maps. A vast amount of work has been done using chaotic maps. Some major contributions include [6, 27, 45, 46]. An interesting wok employs Josephus traversing and mixed chaotic map which is secure as it can resist common attacks [42].

A relatively modernist approach of image encryption is the Arnold's Cat Map algorithm. This method is based on the chaos theory [8]. Chaos theory is a field of study in mathematics, with applications in several disciplines including physics, engineering, economics, biology, and philosophy. Chaos theory studies the behavior of dynamic systems that are highly sensitive to initial conditions, an effect which is popularly referred to as the butterfly effect [39]. Minor variation in the initial conditions (such as those due to rounding errors in numerical computation) yield widely diverging outcomes for such dynamic systems, rendering long-term prediction impractical. This develops even though these systems are deterministic, that is the subsequent performance of these systems is fully decided by their initial conditions. This behavior is known as deterministic chaos, or simply chaos. This chaos theory forms the basis of the Arnold's cat map.

Arnold's cat map is a special type of chaotic map [20, 36] and it can be represented in a matrix form as follows:

$$\begin{bmatrix} x_i \\ y_i \end{bmatrix} = \begin{bmatrix} 1 & \varepsilon \\ \varphi & \varepsilon\varphi + 1 \end{bmatrix} \begin{bmatrix} x_{i-1} \\ y_{i-1} \end{bmatrix} \tag{1}$$

where x, y are the i^{th} pixel position of a $n\times$ n image, ε and φ are positive constant parameters. These parameters are chosen such that the determinant is equal to 1.

The Arnold's cat map based image encryption algorithm states that when an image is transformed the original positions of the pixels are randomized. Yet, the original image emerges again if iterated for a specified time. The number of such iterations is known as the Arnold's period. This period depends on the image size; i.e., for different sized images, Arnold's period will be different. It is calculated as: $T = 1.4938N + 40.8689$, where $2 \leq \mathrm{N} \leq 2000$ [40]. The main advantage of this encryption technique is that it can safeguard the image well without reducing the value or information of the image.

3.4 Proposed image encryption algorithm

Encryption schemes based only on permutation has been found to be insecure due to high information redundancy. Most prevailing image encryption algorithms fabricate the encrypted image to resemble a texture like or noise like image (as for the Arnold's cat map encryption) which is, however, an apparent observable evidence of the existence of an encrypted image and thus results in a remarkably large number of attacks. To confront this issue, this paper proposes a new image encryption concept to transform an original image into a visually meaningful encrypted one.

The proposed algorithm is based on pixel permutation using chaotic map. It uses a variation of the Arnold's cat map algorithm. For an image of size $r\times c$, each pixel is replaced by another pixel from the image. This is termed as pixel permutation. The new pixel permutation is chosen as:

$$\begin{bmatrix} x_i \\ y_i \end{bmatrix} = \begin{bmatrix} 1 & \varepsilon \\ \varphi & \varepsilon\varphi + 1 \end{bmatrix} \begin{bmatrix} x_{i-1} \\ y_{i-1} \end{bmatrix} \quad (mod\ n) \tag{2}$$

The image so obtained is then normalized. A random pixel *(a,b)* is then selected and the pixel value, K, at that location serves as the key. A new matrix P (of size $r \times c$) is then constructed and initialized by K. A bit wise XOR is then performed between the normalized image and the newly constructed matrix P to obtain the encrypted image E. This encrypted image is then masked into a general image, called the key image, to elude the intruder. In contrast to the Arnold's cat map encryption scheme, the proposed algorithm produces visually pleasing cipher images. For the decryption process, the encrypted image, the key image and the key has to be shared. The decryption process begins with using the key image and the received encrypted image to obtain an intermediary image Z. The final decryption is performed by undoing the pixel permutation by using the inverse of $\begin{vmatrix} 1 & \varepsilon \\ \varphi & \varepsilon\varphi + 1 \end{vmatrix}$.

Algorithm 13 Proposed image encryption algorithm.

Input: Image I of size $r \times c$ and the key image K_{Ref}

Output: Encrypted Image E

Intialization: Choose Period T, φ, ε such that $\begin{vmatrix} 1 & \varepsilon \\ \varphi & \varepsilon\varphi + 1 \end{vmatrix} = 1$

for k = 1 to T **do**
  for i = 1 to r **do**
    for j = 1 to c **do**
      $r' = \text{mod}((x_i + \varepsilon \times y_j), \text{rows})+1$
      $c' = \text{mod}((\varphi \times x_i + (\varepsilon\ \varphi+1) \times y_j), \text{rows})+1$
      pixel(i, j) = pixel(r', c')
    end for
  end for
end for

Image M is obtained after pixel permutation

N = M *255;

Randomly select any pixel position (x, y) as the Key K

Generate a matrix P of size r× c such that all values are intialized to K

Z = N⊕P

for i = 1 to r **do**
  for j = 1 to c **do**
    E(i,j) = K_{Ref}(i,j) - Z(i,j)
  end for
end for

Algorithm 14 Proposed image decryption algorithm.

Input: Encrypted Image E of size $r \times c$, P obtained by initializing array with key K and the key image K_{Ref}

Output: Original Image I

for i = 1 to r **do**
  for j = 1 to c **do**
    Z(i,j) = K_{Ref}(i,j) - E(i,j)
  end for
end for

N = Z⊕P

for k = 1 to T **do**
  for i = 1 to r **do**
    for j = 1 to c **do**
      $I(x_i,y_j) = \begin{vmatrix} 1 & \varepsilon \\ \varphi & \varepsilon\varphi + 1 \end{vmatrix}^{-1} .N(x_i,y_j)$
    end for
  end for
end for

The key space of the proposed algorithm is infinite as the key chosen can be any random image and has a sufficiently large number of possible variations. It is impossible for unauthorized users to decode the encrypted image by means of an exhaustive searching for the possible choices in the key space. As a result, the image is protected by a high level of security. In cryptanalysis, the chosen-plaintext attack is an attack model in which the attacker can choose a number of plaintexts and then get their corresponding cipher texts. In this manner, the attacker can choose any useful information as plaintext in order to deduce the security keys of encryption algorithms, or to reconstruct the original plaintexts from the unknown cipher texts [4]. If the image pixel values are not changed by the encryption process, the chosen-plaintext attack can break the encrypted image without knowing the encryption algorithm or its security keys. The presented image encryption algorithm changes image pixel values while changing the locations of all image pixels. This ensures that the encrypted image data is not useful in the case of a chosen-plaintext attack. As a result, the presented algorithm is able to withstand chosen-plaintext attacks.

4 Evaluating the strength of image encryption algorithm

The traditional method of image assessment have been Mean Squared Error (MSE), Signal to Noise Ratio (SNR) and Peak Signal to Noise Ratio (PSNR). These assessment methods are based on absolute error of images. Other assessment methods are based on Differential analysis. Number of changing pixel rate (NPCR) and the Unified averaged changed intensity (UACI) are the two most common evaluation indicators which have ability to test resistance against differential attack.

4.1 Mean square error

The Mean Square Error (MSE) is used as a part of digital image processing method to check for errors. Intuitively, the addition of the pixel-by-pixel squared errors between two images is the square of the distance between them. MSE is the average squared distance for the corresponding pixel in two images. MSE can be calculated using:

$$\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}\frac{[X(i,\,j)-Y(i,\,j)]^2}{m\times n} \tag{3}$$

Here, *X(i, j)* and *Y(i, j)* represent the $(i,\,j)^{th}$ pixel of the two images being compared, and m, n are the row and column of the image matrix [41]. A lower value of MSE indicates higher image quality. SNR and PSNR are both derived from MSE. The PSNR value determines the peak value of signal-to-noise ratio between the two images and is generally used as a feature measurement between the images.

4.2 Structural similarity index

Pixels have intense inter-dependencies substantially when they are spatially close and this reliance convey significant information concerning their structure. SSIM makes use of this dependency and quantifies visual supremacy with a similarity metric between two images as the product of three components: luminance comparison, contrast comparison and structure comparison [43].

$$SSIM=[l(x,\,y)]^{\alpha}\times[c(x,\,y)]^{\beta}\times[s(x,\,y)]^{\gamma} \tag{4}$$

Where *l(x,y)* represents the luminance comparison, *c(x,y)* represents the contrast comparison, *s(x,y)* represents the structure comparison and α, β, γ are positive exponents that adjusts the components contribution to the overall SSIM. The luminance, contrast and structure comparison parameters are calculated using mean, variance and standard deviation as follows:

$$l(x, y) = \frac{(2\mu_x\mu_y + C_1)}{(\mu_x^2 + \mu_y^2 + C_1)}, c(x, y) = \frac{(2\sigma_x\sigma_y + C_2)}{(\sigma_x^2 + \sigma_y^2 + C_2)}, s(x, y) = \frac{(\sigma_{xy} + C_3)}{(\sigma_x + \sigma_y + C_3)} \quad (5)$$

μ_x, μ_y denote the mean of image x and y and is calculated as:

$$\mu_x = \frac{1}{N}\sum_{i=1}^{N} x_i \quad and \quad \mu_y = \frac{1}{N}\sum_{i=1}^{N} y_i \quad (6)$$

σ_x, σ_y denote the standard deviation and is calculated as:

$$\sigma_x = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \mu_x)^2} \quad and \quad \sigma_y = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(y_i - \mu_y)^2} \quad (7)$$

σ_{xy} denote the cross correlation calculated as:

$$\sigma_{xy} = \frac{1}{N-1}\sum_{i=1} N(x_i - \mu_x)(y_i - \mu_y) \quad (8)$$

C_{i} (for i = 1, 2, 3) are small positive constants which combat stability issues when either $(\mu_x^2 + \mu_y^2)$ or $(\sigma_x^2 + \sigma_y^2)$ is close to zero and are calculated as:

$$C_1 = (K_1.L)^2, \quad C_2 = (K_2.L)^2, \quad C_3 = \frac{C_1}{2} \quad (9)$$

where K_1 and K_2 are small constant less than 1 and L is the dynamic range of pixel values. The paper uses the default value of K_1 and K_2 as 0.01 and 0.03, L as 255 and $\alpha = \beta = \gamma = 1$[43]. The SSIM index ranges between -1 and 1. The SSIM index for two images will be 1 if they are identical.

4.3 Correlation coefficient

The correlation coefficient is used to quantitatively evaluate the supremacy of the output images. This coefficient is widely applied in statistics to estimate how strong is the association between two variables. In general images, each pixel is distinctly correlated with its adjoining pixels. A good encryption technique should produce the cipher images with less correlation with the adjacent pixels. The correlation coefficient is calculated as [41]:

$$r_{xy} = \frac{\sigma_{xy}}{\sigma_x \times \sigma_y} \quad (10)$$

where σ_{xy} is calculated as in (8) and σ_x and σ_y is calculated as in (7). For an efficient algorithm the correaltion coefficient should be less.

4.4 Number of changing pixel rate

In image encryption, a cipher's defiance to attacks is commonly examined using the NPCR tests. The NPCR is delineated to examine the number of changing pixels in encrypted image when its difference from the original image is subtle (usually a single pixel). Although this test is concisely defined and is easy to evaluate, test scores are difficult to interpret. For

example, the upper-bound of the NPCR score is 100, and hence, the NPCR score of a secure cipher should be very close to this upper-bound [17, 44]. The results are included in Table 7.

$$NPCR = \sum_{i,j} \frac{D(i,\ j)}{T} \times 100\% \tag{11}$$

where

$$D(i,\ j) = \begin{cases} 0, & \text{if I(i, j)} = \text{E(i, j)} \\ 1, & \text{if I(i, j)} \neq \text{E(i, j)} \end{cases} \tag{12}$$

and T denotes the total number of pixels in the encrypted image. I and E represents the original image and encrypted image respectively.

5 Results and discussion

The proposed image encryption algorithm (Algorithm 13) and the Arnold's cat map based encryption technique has been implemented on a set of 10 different images. The results included in the paper are of two benchmark images;(i) the Baboon image (ii) the Lena image; and a sample image. Figure 1 represents the original images used to test the algorithm and the key image. The key image camouflages the actual image to be encrypted.

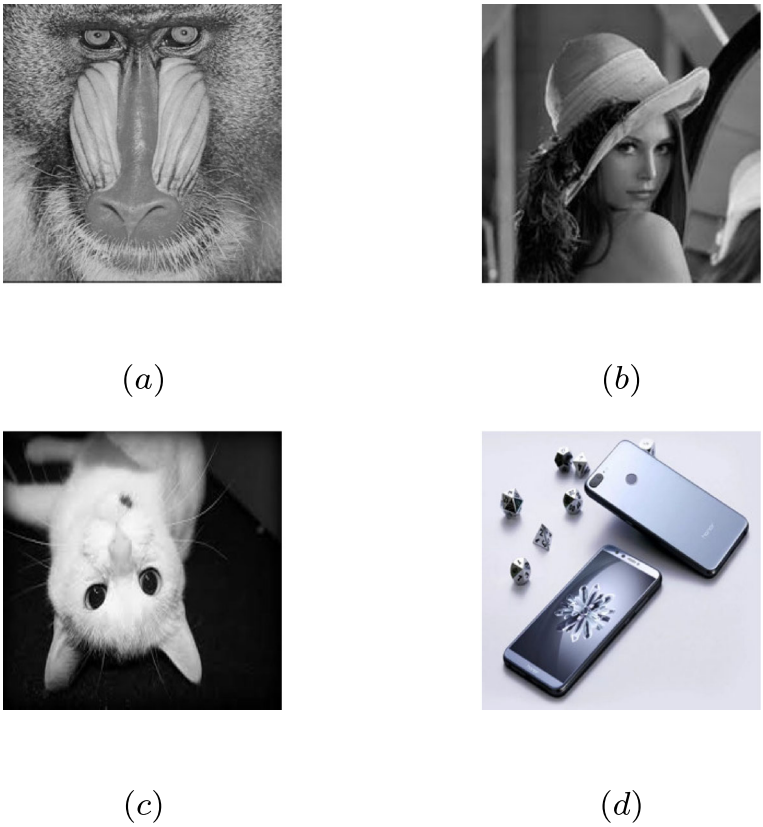

Fig. 1 Original Images: **a** Baboon; **b** Lena; **c** Sample Image; **d** Key image

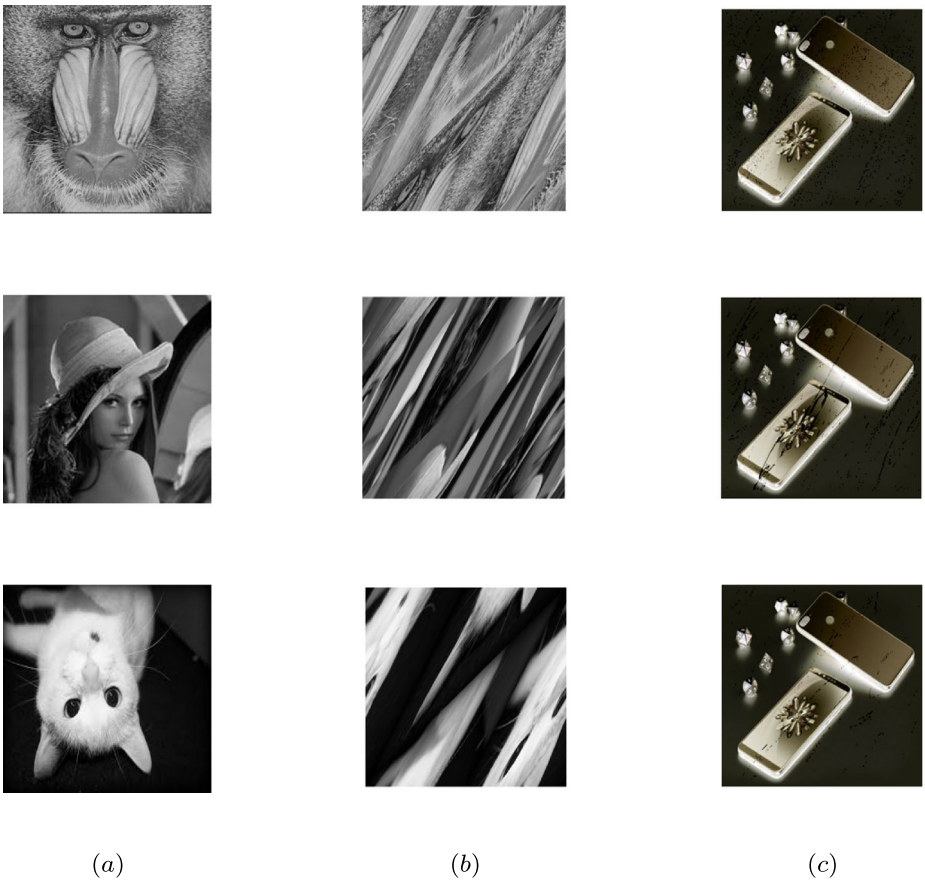

Fig. 2 Encrypted Images for Baboon, Lena and a Sample Image; **a** Original Image; **b** Encrypted Image using Arnold's catmap; **c** Encrypted Image using the Proposed Algorithm

The results of applying both the encryption techniques are represented in Fig. 2. Figure 2a represents the original image to be encrypted (the Baboon image, Lena image and a sample image), Fig 2b are the results of the Arnold's catmap encryption technique. It is evident from visual inspection that the encrypted images do not convey any meaning and are hence prone to attacks as the intruder can make a guess that it is an encrypted image and try to intrude it. Figure 2c represents the result of the proposed algorithm where the encryption is camouflaged with the key image. These images are visually pleasing and gives no clue that the image is actually an encrypted image.

Beside the fact that the encrypted image are visually pleasing, a comparative analysis of both the algorithms have been done in terms of the performance parameters.

Table 7 shows the result of the evaluation parameters of the Proposed algorithm and the Arnold's Cat Map Algorithm (ACM). The SSIM value of the original and decrypted images are compared and it can be observed from the table that the proposed algorithm performs slightly better than the ACM method as it shows a greater SSIM value for the decrypted image. Similarly, the proposed algorithm has a lesser CC value that can be deduced form the table. The NPCR value calculated for the both the algorithms were similar at around 98 - 99

Table 7 Evaluation parameters

Images	ACM			Proposed		
	SSIM	CC	NPCR	SSIM	CC	NPCR
Lena	0.613	0.9667	98.3	0.613	0.8454	98.9
Baboon	0.1379	0.643	97.8	0.1505	0.4852	98
Sample	0.2792	0.9936	98.8	0.7752	0.8762	99.3

%. Visually the output of the proposed algorithm is more pleasing as it does not resemble a texture or noise like pattern and hence is difficult for the intruder to detect it as an encrypted image.

6 Conclusion

Encryption is a vital part in security due to increased usage of multimedia and web services for sharing data. Image encryption has equal importance and is still a major research area with new encryption techniques being developed. The paper proposed an encryption technique based on pixel permutation using chaotic maps. The algorithm is simulated on a set of 10 different images, 3 of which are included in the paper. Apart from being visually misleading as a cipher image, the algorithm also marginally performs better than the Arnold's cat map algorithm. The same can be concreted from the evaluation of the performance parameters. The encryption technique can be widely used to share vital information in form of images. Future work may include a detailed crypt analysis of the proposed algorithm and its complexity. Further, the possibility of combining other approaches to this technique can lead to more robust encryption algorithms.

References

1. Al-Khassaweneh M, Tawalbeh S (2013) A value transformation and random permutation-based coloured image encryption technique. Int J Inf Comput Secur 5(4):290–300
2. Aljawarneh S, Yassein MB (2017) A resource-efficient encryption algorithm for multimedia big data. Multimed Tools Appl 76(21):22703–24
3. Aljawarneh S, Yassein MB (2018) A multithreaded programming approach for multimedia big data: encryption system. Multimed Tools Appl 1:1–20
4. Badve O, Gupta BB, Gupta S (2016) Reviewing the security features in contemporary security policies and models for multiple platforms. In: Handbook of Research on Modern Cryptographic Solutions for Computer and Cyber Security. IGI Global, pp 479–504
5. Bhanot R, Hans R (2015) A review and comparative analysis of various encryption algorithms. International Journal of Security and Its Applications 9(4):289–306
6. Chen G, Mao Y, Chui CK (2004) Symmetric image encryption scheme based on 3D chaotic cat maps. Chaos. Solitons & Fractals. 21(3):749–61
7. Dang PP, Chau PM (2000) Image encryption for secure internet multimedia applications. IEEE Trans Consum Electron 46(3):395–403
8. Davies B (2018) Exploring chaos: theory and experiment. CRC Press, Boca Raton
9. Dey S, Ghosh R (2018) A review of cryptographic properties of S-boxes with generation and analysis of crypto secure S-boxes. PeerJ Preprints
10. Fouda JA, Effa JY, Sabat SL, Ali M (2014) A fast chaotic block cipher for image encryption. Commun Nonlinear Sci Numer Simul 19(3):578–88

11. Fu C, Chen JJ, Zou H, Meng WH, Zhan YF, Yu YW (2012) A chaos-based digital image encryption scheme with an improved diffusion strategy. Opt Express 20(3):2363–78
12. Fu C, Lin BB, Miao YS, Liu X, Chen JJ (2011) A novel chaos-based bit-level permutation scheme for digital image encryption. Opt Commun 284(23):5415–23
13. Guan ZH, Huang F, Guan W (2005) Chaos-based image encryption algorithm. Phys Lett A 346(1-3):153–7
14. Huang CK, Liao CW, Hsu SL, Jeng YC (2013) Implementation of gray image encryption with pixel shuffling and gray-level encryption by single chaotic system. Telecommun Syst 52(2):563–71
15. Huang CK, Nien HH (2009) Multi chaotic systems based pixel shuffle for image encryption. Opt Commun 282(11):2123–7
16. Kahate A (2013) Cryptography and network security. Tata McGraw-Hill Education, New York, pp 94–107
17. Khan MF, Ahmed A, Saleem K (2019) A novel cryptographic substitution box design using gaussian distribution. IEEE Access 7:15999–6007
18. Konikoff J, Toplosky S (2010) Analysis of simplified DES algorithms. Cryptologia 34(3):211–24
19. Krikor L, Baba S, Arif T, Shaaban Z (2009) Image encryption using DCT and stream cipher. Eur J Sci Res 32(1):47–57
20. Li C, Lin D, Feng B, Lu J, Hao F (2018) Cryptanalysis of a chaotic image encryption algorithm based on information entropy. IEEE Access 6:75834–75842
21. Liu H, Wang X (2011) Color image encryption using spatial bit-level permutation and high-dimension chaotic system. Opt Commun 284(16-17):3895–903
22. Liu Z, Li S, Liu W, Wang Y, Liu S (2013) Image encryption algorithm by using fractional Fourier transform and pixel scrambling operation based on double random phase encoding. Opt Lasers Eng 51(1):8–14
23. Lukac R, Plataniotis KN (2005) Bit-level based secret sharing for image encryption. Pattern Recogn 38(5):767–72
24. Nag A, Singh JP, Khan S, Biswas S, Sarkar D, Sarkar PP (2011) Image encryption using affine transform and XOR operation. In: 2011 international conference on signal processing, communication, computing and networking technologies (ICSCCN). IEEE, pp 309–312
25. Nie T, Zhang T (2009) A study of DES and Blowfish encryption algorithm. In: Tencon 2009-2009 IEEE region 10 conference. IEEE, pp 1–4
26. Norcen R, Podesser M, Pommer A, Schmidt HP, Uhl A (2003) Confidential storage and transmission of medical image data. Comput Biol Med 33(3):277–92
27. Pareek NK, Patidar V, Sud KK (2006) Image encryption using chaotic logistic map. Image Vis Comput 24(9):926–34
28. Patil P, Narayankar P, Narayan DG, Meena SM (2016) Comprehensive evaluation of cryptographic algorithms: DES, 3DES, AES, RSA and Blowfish. Procedia Computer Science. 78:617–24
29. Qian Z, Zhou H, Zhang X, Zhang W (2016) Separable reversible data hiding in encrypted JPEG bitstreams. IEEE Trans Dependable Secure Comput 15(6):1055–1067
30. Reyad O, Mofaddel MA, Abd-Elhafiez WM, Fathy M (2017) A novel image encryption scheme based on different block sizes for grayscale and color images. In: 2017 12th international conference on computer engineering and systems (ICCES). IEEE, pp 455–461
31. Rijmen V, Daemen J (2001) Advanced encryption standard. Proceedings of federal information processing standards publications. National Institute of Standards and Technology, pp 19–22
32. Rivest RL, Shamir A, Adleman L (1978) A method for obtaining digital signatures and public-key cryptosystems. Commun ACM 21(2):120–6
33. Schaefer EF (1996) A simplified data encryption standard algorithm. Cryptologia 20(1):77–84
34. Schneier B (1993) Description of a new variable-length key, 64-bit block cipher (Blowfish). InInternational Workshop on Fast Software Encryption. Springer, Berlin, pp 191–204
35. Selent D (2010) Advanced encryption standard. Rivier Academic Journal 6(2):1–4
36. Singh P, Yadav AK, Singh K (2017) Phase image encryption in the fractional Hartley domain using Arnold transform and singular value decomposition. Opt Lasers Eng 91:187–95
37. Smid ME, Branstad DK (1988) Data encryption standard: past and future. Proc IEEE 76(5):550–9
38. Tang L (1997) Methods for encrypting and decrypting MPEG video data efficiently. In: Proceedings of the fourth ACM international conference on Multimedia. ACM, pp 219–229
39. Vaidyanathan S, Rajagopal K (2017) LabVIEW implementation of chaotic masking with adaptively synchronised forced Van der Pol oscillators and its application in real-time image encryption. Int J Simul Process Model 12(2):165–78
40. Wahballa O, Wahaballa A, Li F, Idris II, Xu C (2017) Medical image encryption scheme based on arnold transformation and ID-AK protocol. IJ Network Security 19(5):776–84

41. Wang J, Zheng N, Chen B, Principe JC (2017) Associations among image assessments as cost functions in linear decomposition: MSE, SSIM, and Correlation Coefficient. arXiv:1708.01541
42. Wang X, Zhu X, Zhang Y (2018) An image encryption algorithm based on Josephus traversing and mixed chaotic map. IEEE Access 6:23733–46
43. Wang Z, Bovik AC, Sheikh HR, Simoncelli EP (2004) Image quality assessment: from error visibility to structural similarity. IEEE Trans Image Process 13(4):600–12
44. Wu Y, Noonan JP, Agaian S (2011) UACI Randomness tests for image encryption NPCR Cyber journals: multidisciplinary journals in science and technology. Journal of Selected Areas in Telecommunications (JSAT) 1(2):31–8
45. Xie EY, Li C, Yu S, Lu J (2017) On the cryptanalysis of Fridrich's chaotic image encryption scheme. Signal Process 132:150–154
46. Ye G, Wong KW (2012) An efficient chaotic image encryption algorithm based on a generalized Arnold map. Nonlinear Dyn 69(4):2079–87
47. Yun-Peng Z, Wei L, Shui-Ping C, Zheng-Jun Z, Xuan N, Wei-Di D (2009) Digital image encryption algorithm based on chaos and improved DES. In: SMC 2009 IEEE international conference on systems, man and cybernetics, 2009. IEEE, pp 474–479
48. Zhang G, Liu Q (2011) A novel image encryption method based on total shuffling scheme. Opt Commun 284(12):2775–80
49. Zhou N, Wang Y, Gong L (2011) Novel optical image encryption scheme based on fractional Mellin transform. Opt Commun 284(13):3234–42
50. Zhu ZL, Zhang W, Wong KW, Yu H (2011) A chaos-based symmetric image encryption scheme using a bit-level permutation. Inform Sci 181(6):1171–86

Shamama Anwar received her M. Tech degree in Computer Science from Birla Institute of Technology, Mesra, India and Ph.D. from NIFFT, India in 2017. She is currently employed at Birla Institute of Technology, Mesra, India as an Assistant Professor in the department of Computer Science and Engineering. Her research interest includes Artificial Intelligence, Soft computing, Image processing and Social Intelligence.

Solleti Meghana is an engineer graduate from the Department of Computer Science and Engineering at Birla Institute of Technology, Mesra, India. She is currently employed as a Research and Innovation Engineer at Adrosonic IT Consultancy services, India.