A Simple, Secure, and Time-Efficient Bit-Plane Operated Bit-Level Image Encryption Scheme Using 1-D Chaotic Maps

K. Abhimanyu Kumar Patro and Bibhudendra Acharya

1 Introduction

Due to the advancements in Internet technology, large numbers of images are transmitted over the internet. As a result, privacy and security of images have become a major issue in the present scenario. Encryption is one of the well-known techniques to secure images against various attacks. Though various traditional encryption techniques [1, 2] are present, they are not much suitable to encrypt images because of some image in-built characteristics such as high-correlation of adjacent pixels, bulky data, high redundancy, etc [3–6].

Chaos-based image encryption provides a better security to the images. Chaotic maps have certain good characteristics such as sensitivity of their parameters, non-periodicity, ergodicity, nonlinearity [7, 8]. Matthews [9] first proposed chaos-based encryption technique, since then, many encryption techniques are used chaotic maps in image encryption. Chaotic maps are classified into high-dimensional and one-dimensional (1-D) [10]. High-dimensional chaotic maps have the disadvantage of complex structure and therefore requires high amount of hardware resources for implementation. Complexity can be reduced using 1-D chaotic maps which are simple in structure with low hardware resource requirement and high efficiency [11]. Hence, the proposed algorithm uses 1-D chaotic maps to perform image encryption. A number of researches have been done using 1-D chaotic maps. Multiple 1-D chaotic mapbased image encryption is proposed by Ahmed et al. in [12]. A variable-size hash function is developed using multiple chaotic maps by Musheer Ahmad et al. in [13].

K. A. K. Patro (✉) · B. Acharya
Department of Electronics and Telecommunication Engineering, National Institute of Technology Raipur, Raipur 492010, Chhattisgarh, India
e-mail: abhimanyu.patro@gmail.com

B. Acharya
e-mail: bacharya.etc@nitrr.ac.in

J. Chattopadhyay et al. (eds.), *Innovations in Soft Computing and Information Technology*, https://doi.org/10.1007/978-981-13-3185-5_23

This paper contributes the followings.

1. To reduce the total simulation time and also to make the algorithm simple, two stages of bit-plane diffusion operation and bit-level column–row shuffling operation are presented.
2. The Beta map along with the Piece-wise Linear Chaotic Map (PWLCM) system is used in this algorithm to obtain large key space and high key sensitivity.
3. Hash-based key operations are performed to prevent the algorithm against chosen-plaintext attack (CPA) and known-plaintext attack (KPA).

The remaining sections are as follows. Section 2 describes the 1-D chaotic maps. Section 3 presents the proposed key generation and encryption algorithm. The security analyses and the computer simulations are presented in Sect. 4. At last, the conclusion is reached in Sect. 5.

2 One-Dimensional Chaotic Maps

2.1 *Beta Map*

The Beta function [14] is defined as,

$$\mathrm{Beta}(x;\alpha,\beta,x_1,x_2)=\begin{cases}\left(\frac{x-x_1}{x_0-x_1}\right)^{\alpha}\left(\frac{x_2-x}{x_2-x_0}\right)^{\beta} & \text{if } x\in\,]x1,x2[\\ 0 & \text{else}\end{cases} \tag{1}$$

where α,β,x_1 and $x_2\in R, x_1<x_2$ and x_0,

$$x_0=\frac{(\alpha x_2+\beta x_1)}{(\alpha+\beta)} \tag{2}$$

By using the Beta function, the Beta map is defined as,

$$x_{n+1}=k\times\mathrm{Beta}(x_n;x_1,x_2,\alpha,\beta) \tag{3}$$

where

$$\alpha=p_1+q_1\times r \tag{4}$$

and

$$\beta=p_2+q_2\times r \tag{5}$$

where k is the control parameter, r is the bifurcation parameter, and (p_1,q_1,p_2,q_2) are the constants.

2.2 PWLCM System

Recently, in image encryptions, PWLCM systems are widely used because of its good characteristics such as simpler representation, efficient implementation, excellent ergodicity, and good dynamical behavior [15, 16].

The PWLCM system is defined as,

$$x_{n+1} = \begin{cases} \frac{(x_n)}{(\mu)} & \text{if } 0 \le x_n < \mu \\ \frac{(x_n-\mu)}{(0.5-\mu)} & \text{if } \mu \le x_n < 0.5 \\ (1 - x_n) & \text{if } 0.5 \le x_n < 1 \end{cases} \tag{6}$$

where μ lies in the range (0, 0.5).

3 Proposed Methodology

3.1 Key Generation Operation

The steps for key generation are

Step 1: Input an original grayscale image, *I*.
Step 2: Generate 32-decimal values by converting 256-bit hash values of original image. The generated hash values are represented as,

$$\text{hash} = h1, h2, h3, \ldots, h31, h32 \tag{7}$$

Step 3: Using the hash values, the secret keys are generated as,

$$\begin{cases} \text{mue} = \text{mue1} + (\text{mod}(\text{sum}(h1 : h2), 256)/2^9) \times 0.1 \\ x1(1) = x(1) + (\text{mod}(\text{sum}(h3 : h5), 256)/2^9) \times 0.1 \end{cases} \tag{8}$$

where $(\text{mue1}, \text{mue})$ are the original and generated system parameters, $(x(1), x1(1))$ are the original and generated initial values of PWLCM system only.

$$\begin{cases} xxx1(1) = xxx(1) + (\text{mod}(\text{sum}(h6 : h8), 256)/2^9) \times 0.1 \\ k = kk + (\text{mod}(\text{sum}(h9 : h11), 256)/2^9) \times 0.1 \\ x1 = xx1 + (\text{mod}(\text{sum}(h12 : h14), 256)/2^9) \times 0.1 \\ x2 = xx2 + (\text{mod}(\text{sum}(h15 : h17), 256)/2^9) \times 0.1 \\ p1 = pp1 + (\text{mod}(\text{sum}(h18 : h20), 256)/2^9) \times 0.1 \\ p2 = pp2 + (\text{mod}(\text{sum}(h21 : h23), 256)/2^9) \times 0.1 \\ q1 = qq1 + (\text{mod}(\text{sum}(h24 : h26), 256)/2^9) \times 0.1 \\ q2 = qq2 + (\text{mod}(\text{sum}(h27 : h29), 256)/2^9) \times 0.1 \\ r = rr + (\text{mod}(\text{sum}(h30 : h32), 256)/2^9) \times 0.1 \end{cases} \tag{9}$$

where $(xxx(1), xxx1(1))$ are the original and generated initial values of Beta map only. $(kk, xx1, xx2, pp1, pp2, qq1, qq2, rr)$ and $(k, x1, x2, p1, p2, q1, q2, r)$ are the original and generated system parameters of Beta map only.

3.2 Encryption Operation

Figure 1 shows the block diagram of the encryption operation of the proposed cryptosystem. The encryption operation is as follows:

Step 1: Input an $M \times N$ sized grayscale image, I.
Step 2: Generate bit-planes of the image, I. The generated 8 bit-planes are,

$$IBBP1, IBBP2, IBBP3, IBBP4, IBBP5, IBBP6, IBBP7, IBBP8$$

Step 3: Generate secret keys $(\text{mue}, x1(1))$ of PWLCM system as Step 3 of Sect. 3.1.
Step 4: Iterate the PWLCM system of Eq. (6) for $200 + (M \times N \times 8)$ times. To avoid transit effect, remove 200 iterations from the first position. Therefore, the generated PWLCM sequence is,

$$x1 = (x1(1), x1(2), \ldots, x1(M \times N \times 8)) \tag{10}$$

Step 5: Using the sequence $x1$ of PWLCM system, generate a key image of size $M \times N$. The key image generation process is as follows:

```
for $i = 1: (M \times N \times 8)$
    if $x1(i) < 0.5$
        $x1(i) = 0;$
    else
        $x1(i) = 1;$
    end
end
$K = reshape(x1, [(M \times N), 8]);$
$C = bi2de(K);$
$D = reshape(C, [M, N]);$                                   (11)
```

where D is the generated key image. Here, the threshold value is taken as 0.5.
Step 6: Generate bit-planes of the key image, D. The generated 8 bit-planes are, $KBBP1, KBBP2, KBBP3, KBBP4, KBBP5, KBBP6, KBBP7, KBBP8$
Step 7: Perform Stage-1 of bit-plane diffusion operation between bit-planes of the original image and the key image. Let the diffused bit-plane outputs are, $IKBP1, IKBP2, IKBP3, IKBP4, IKBP5, IKBP6, IKBP7, IKBP8$

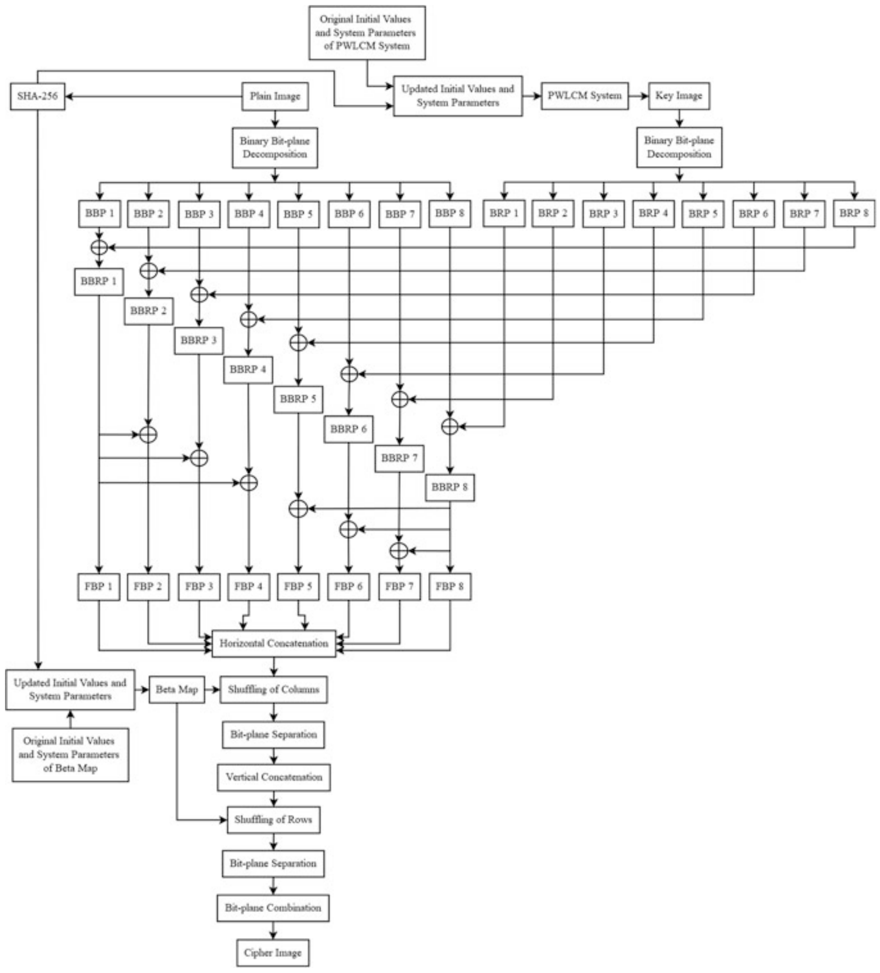

Fig. 1 Encryption system block diagram

The diffusion operation is as follows:

$$\begin{cases} IKBP1 = IBBP1 \oplus KBBP8 \\ IKBP2 = IBBP2 \oplus KBBP7 \\ IKBP3 = IBBP3 \oplus KBBP6 \\ IKBP4 = IBBP4 \oplus KBBP5 \\ IKBP5 = IBBP5 \oplus KBBP4 \\ IKBP6 = IBBP6 \oplus KBBP3 \\ IKBP7 = IBBP7 \oplus KBBP2 \\ IKBP8 = IBBP8 \oplus KBBP1 \end{cases} \tag{12}$$

Step 8: Perform Stage-2 of bit-plane diffusion operation between the bit-plane outputs of **Step 7**. Let the diffused bit-plane outputs are denoted as:

$$FDBP1, FDBP2, FDBP3, FDBP4, FDBP5, FDBP6, FDBP7, FDBP8$$

The diffusion operation is as follows:

$$\begin{cases} FDBP1 = IKBP1 \\ FDBP2 = IKBP1 \oplus IKBP2 \\ FDBP3 = IKBP1 \oplus IKBP3 \\ FDBP4 = IKBP1 \oplus IKBP4 \\ FDBP5 = IKBP5 \oplus IKBP8 \\ FDBP6 = IKBP6 \oplus IKBP8 \\ FDBP7 = IKBP7 \oplus IKBP8 \\ FDBP8 = IKBP8 \end{cases} \tag{13}$$

Step 9: Generate the secret keys $(xxx1(1), k, x1, x2, p1, p2, q1, q2, r)$ of Beta map as **Step 3** of Sect. 3.1.
Step 10: Iterate the Beta map of Eq. 1–5 for $200 + ((N \times 8) + (M \times 8))$ times. To avoid transit effect, remove 200 iterations from the first position. Therefore, the generated Beta map sequence is,

$$xxx1 = \begin{pmatrix} xxx1(1), xxx1(2), xxx1(3), \ldots, \\ xxx1((N \times 8) + (M \times 8)) \end{pmatrix} \tag{14}$$

Step 11: Separate the Beta map sequence into two parts which are represented as:

$$\begin{cases} xxx11 = \begin{pmatrix} xxx1(1), xxx1(2), xxx1(3), \ldots, \\ xxx1(N \times 8) \end{pmatrix} \\ xxx12 = \begin{pmatrix} xxx1((N \times 8) + 1), xxx1((N \times 8) + 2), \\ xxx1((N \times 8) + 3), \ldots, \\ xxx1((N \times 8) + (M \times 8)) \end{pmatrix} \end{cases} \tag{15}$$

Step 12: Sort the Beta map sequences as,

$$\begin{cases} [xxxC\text{sort}, xxxC\text{index}] = \text{sort}(xxx11) \\ [xxxR\text{sort}, xxxR\text{index}] = \text{sort}(xxx12) \end{cases} \tag{16}$$

where $xxxC\text{sort}$ and $xxxR\text{sort}$ are the sort sequences of $xxx11$ and $xxx12$, respectively. $xxxC\text{index}$ and $xxxR\text{index}$ are the index values of $xxxC\text{sort}$ and $xxxR\text{sort}$, respectively.
Step 13: Horizontal concatenate the bit-plane outputs of **Step 8**. Perform column shuffling operation by using the index value, $xxxC\text{index}$.
Step 14: Separate the column shuffled bit-plane outputs in **Step 13**. Vertical concatenate the bit-planes to perform row-shuffling operation by using the index value, $xxxR\text{index}$.

Step 15: Separate the bit-planes in **Step 14** and then combine all the bit-planes to generate an encrypted image.

The decryption operation is performed in the reverse way of performing encryption operation.

4 Security Analyses and Computer Simulations

Simulation results are shown in Fig. 2. In this algorithm, the keys are taken as $x(1) = 0.2, \mathrm{mue1} = 0.3, xxx(1) = -0.2, kk = 0.9, xx1 = -0.7, xx2 = 1, pp1 = 8, pp2 = 3, qq1 = 1, qq2 = -1, rr = -0.2$. In Fig. 2, we can see that the proposed

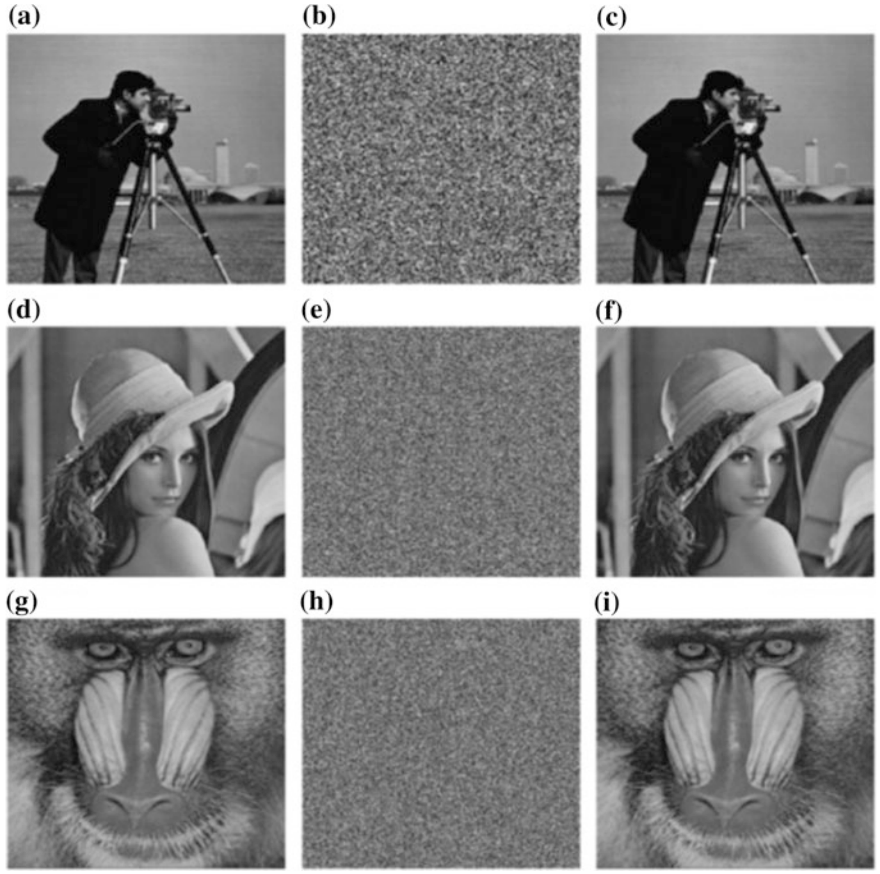

Fig. 2 Computer simulation outputs: original images of **a** "Cameraman (256×256)," **d** "Lena (512×512)," **g** "Baboon (512×512)"; encrypted images of **b** "Cameraman (256×256)," **e** "Lena (512×512)," **h** "Baboon (512×512)"; decrypted images of **c** "Cameraman (256×256)," **f** "Lena (512×512)," **i** "Baboon (512×512)"

algorithm encrypts and decrypts images properly using the right keys. This shows that the proposed method is suitable to encrypt and decrypt grayscale images.

The security analyses are as follows.

4.1 Key Space Analysis

Key space is the number of different keys which are used in an encryption–decryption operation. The key space should be larger than 2^{128} to protect brute-force attack [17]. The secret keys used in this scheme are,

a. system parameters and initial values of chaotic maps
b. hash value of 256-bits.

To define the keys of chaotic maps, a precision value of 10^{-15} is used in this algorithm and for hash value, a key space of 2^{128} is taken in this algorithm.

So the key space in total is,

$\left(10^{15} \times 10^{15}\right) \times \left(10^{15} \times 10^{15} \times 10^{15} \times 10^{15} \times 10^{15} \times 10^{15} \times 10^{15} \times 10^{15} \times 10^{15}\right) \times 2^{128} = 10^{165} \times 2^{128} = 1.0853 \times 2^{676}$ which is larger than 2^{128} [17].

This proves that the proposed scheme strongly protects brute-force attack.

4.2 Statistical Attack Analysis

4.2.1 Histogram Analysis

Figure 3 shows the histogram results of the proposed method. In the results, we see the uniformity of grayscale values in the histograms of encrypted images and also see the differences in the histograms of encrypted and original images. In Fig. 3, we also found the similarity of histograms of decrypted and original images. This confirms the zero losses of information in the decrypted images. These in turn proves the strong resistivity of statistical attack in the encryption scheme.

The histogram uniformity can be evaluated quantitatively by calculating the variance of encrypted images. The lower is the variance, higher the uniformity of grayscale values in histogram images [18]. The variance results are shown in Table 1. In Table 1, we can see the lesser value of variance of encrypted images using the proposed method as compared to the encryption methods in [19, 20]. This proves that a greater uniformity occurs in the histograms of encrypted images using the proposed scheme.

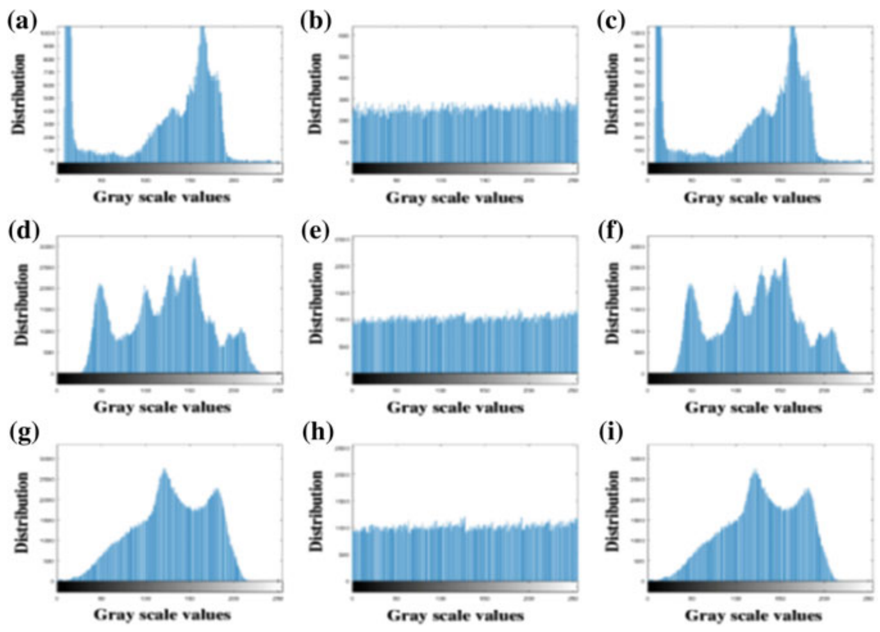

Fig. 3 Histogram outputs: original images of **a** "Cameraman," **d** "Lena," **g** "Baboon"; encrypted images of **b** "Cameraman," **e** "Lena," **h** "Baboon"; decrypted images of **c** "Cameraman," **f** "Lena," **i** "Baboon"

Table 1 Results of variance

Algorithms	Images	Original images	Encrypted images
Ours	Cameraman.tiff	110,970	334.5391
	Lena.tiff	633,400	3480.7
	Baboon.tiff	749,430	3857.3
Ref. [19]	Lena	–	5554.8293
Ref. [20]	Lena	–	5335.8309

4.2.2 Correlation Analysis

Table 2 shows the correlation results of the proposed scheme. In this result, we can see that, in all the three directions, the original images have correlation value close to 1 and the encrypted images have correlation value close to 0. The correlation plot of "Lena" image is shown in Fig. 4. These plots clearly show the high adjacent pixel correlation of original images and weak adjacent pixel correlation of encrypted images. This proves that the proposed scheme strongly resists statistical attack.

Table 2 Correlation coefficient results

Image	Org. images			Enc. images		
	Diag.	Vert.	Horz.	Diag.	Vert.	Horz.
Cameraman.tiff (256 × 256)	0.9082	0.9556	0.9345	-0.0070	0.0140	0.0122
Lena.tiff (512 × 512)	0.9626	0.9853	0.9702	0.0183	0.0017	0.0200
Baboon.tiff (512 × 512)	0.7297	0.7607	0.8681	0.0022	−0.0146	−0.0093

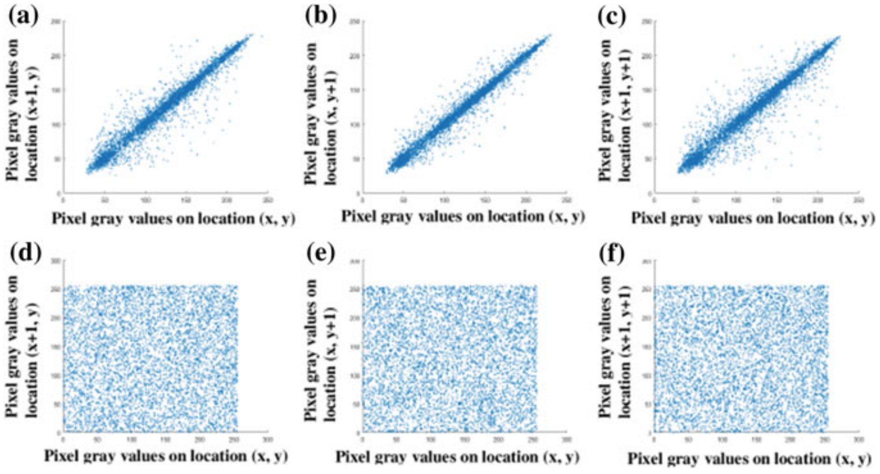

Fig. 4 Correlation outputs of original "Lena" image: **a** horz., **b** vert., and **c** diag.; correlation outputs of encrypted "Lena" image: **d** horz., **e** vert., **f** diag

4.3 Differential Attack Analysis

Two measures Unified Average Changing Intensity (UACI) and Numbers of Pixel Changing Rate (NPCR) are mostly used to analyze the differential attack in an encryption scheme. The expected value of UACI and NPCR of 256-level grayscale image are 33.4635% and 99.6094%, respectively [5]. Table 3 shows the min., max., and avg. UACI and NPCR results (100 values) of the proposed scheme. In Table 3, we can see that the average UACI and NPCR results for all the images are very close to the expected UACI and NPCR values, respectively. This proves the high resistivity of differential attack in the proposed scheme.

4.4 Information Entropy

It measures the degree of randomness of pixels in images. For an ideal encryption system, the entropy of encrypted images falls to 8. The higher is the closeness to 8,

Table 3 UACI and NPCR results

Image	UACI (%)			NPCR (%)		
	Min.	Max.	Avg.	Min.	Max.	Avg.
Cameraman.tiff (256 × 256)	33.2526	33.6811	33.4827	99.5514	99.6536	99.6060
Lena.tiff (512 × 512)	33.3694	33.6682	33.4941	99.5819	99.6414	99.6077
Baboon.tiff (512 × 512)	33.3695	33.6025	33.4727	99.5762	99.6460	99.6079

Table 4 Information entropy results

Images	Original images	Encrypted images
Cameraman.tiff	7.0097	7.9963
Lena.tiff	7.4451	7.9976
Baboon.tiff	7.3583	7.9974

the stronger is the entropy. The entropy results are shown in Table 4. The results show that the entropies of all the encrypted images are very close to 8. This confirms the high randomness of pixels in the encrypted images by using the proposed scheme.

4.5 *Key Sensitivity Analysis*

Key sensitivity measures the sensitivity of keys in decryption and encryption of images. That means a small change in the key will lead to large changes in the output. A cryptosystem should have high key sensitivity to protect brute-force attack [18, 21, 22]. The key sensitivity in both decryption and encryption process is as follows.

4.5.1 Encryption Level Key Sensitivity Analysis

In this process, the encryption operation is executed by changing a single bit in 10^{-15} position of one key and the remaining keys are unchanged and finally analyze the difference image of two encrypted images. The encryption level key sensitivity results of "Lena" image are shown in Fig. 5. The UACI and NPCR results are shown in Table 5. The difference images of Fig. 5 and the NPCR, UACI results of Table 5 shows that more than 99% pixels are changed in changing of one key. This proves the high encryption level key sensitivity of the proposed algorithm.

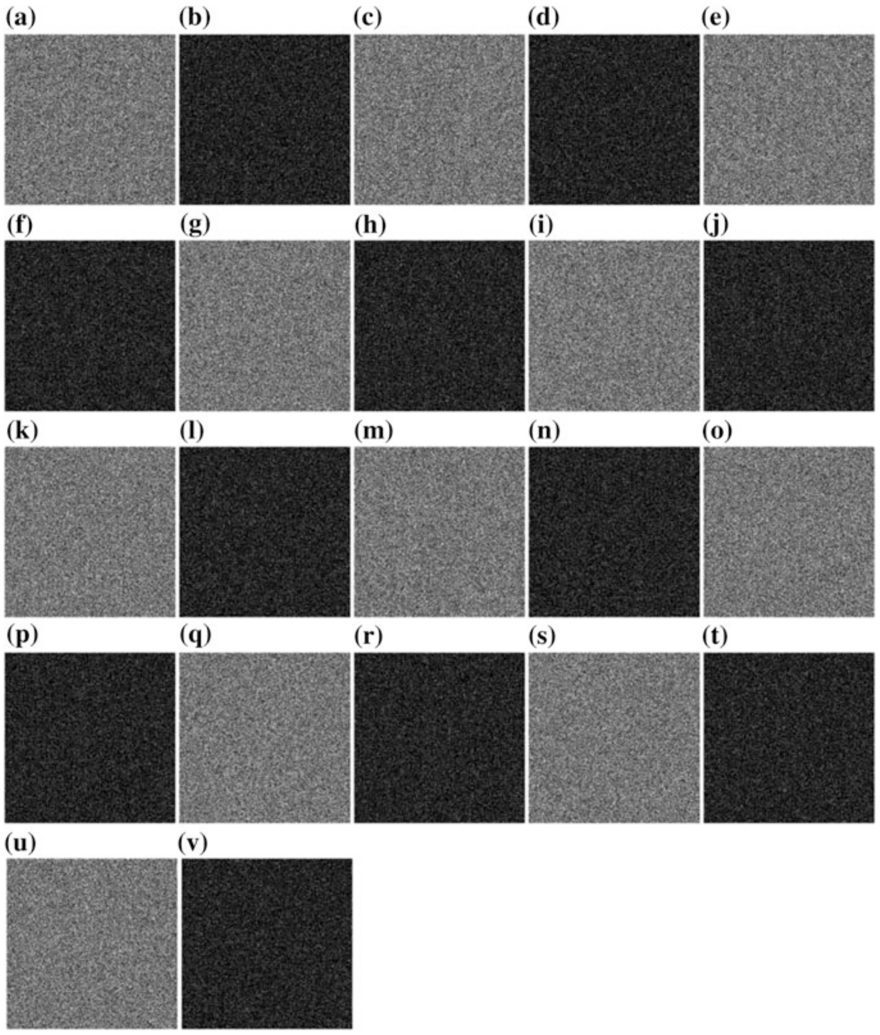

Fig. 5 Encryption level key sensitivity results of “Lena” image: encrypted images when changing of key from **a** mue1 to mue1 $+ 10^{-15}$, **c** $x(1)$ to $x(1) + 10^{-15}$, **e** $xxx(1)$ to $xxx(1) + 10^{-15}$, **g** kk to $kk + 10^{-15}$, **i** $xx1$ to $xx1 + 10^{-15}$, **k** $xx2$ to $xx2 + 10^{-15}$, **m** $pp1$ to $pp1 + 10^{-15}$, **o** $pp2$ to $pp2 + 10^{-15}$, **q** $qq1$ to $qq1 + 10^{-15}$, **s** $qq2$ to $qq2 + 10^{-15}$, **u** rr to $rr + 10^{-15}$; Difference of **b** Figs. 2e and 5a, **d** Figs. 2e and 5c, **f** Figs. 2e and 5e, **h** Figs. 2e and 5g, **j** Figs. 2e and 5i, **l** Figs. 2e and 5k, **n** Figs. 2e and 5m, **p** Figs. 2e and 5o, **r** Figs. 2e and 5q, **t** Figs. 2e and 5s, **v** Figs. 2e and 5u

Table 5 UACI and NPCR results of “Lena” image

Encryption keys	$\text{mue1} + 10^{-15}$	$x(1) + 10^{-15}$	$xxx(1) + 10^{-15}$	$kk + 10^{-15}$	$xx1 + 10^{-15}$	$xx2 + 10^{-15}$
NPCR (%)	99.4892	99.4579	99.6166	99.6235	99.5949	99.5892
UACI (%)	32.7018	32.6263	33.4090	33.4851	33.4971	33.4666
Encryption keys	$pp1 + 10^{-15}$	$pp2 + 10^{-15}$	$qq1 + 10^{-15}$	$qq2 + 10^{-15}$	$rr + 10^{-15}$	
NPCR (%)	99.6090	99.6235	99.6006	99.6220	99.6151	
UACI (%)	33.4944	33.5019	33.5604	33.5239	33.5390	

4.5.2 Decryption Level Key Sensitivity Analysis

In this process, the decryption operation is executed by changing a single bit in 10^{-15} position of one decryption key and the remaining decryption keys are unchanged and finally analyze the key changed decrypted images. The key sensitivity results of “Lena” image are shown in Fig. 6. The NPCR and UACI results are shown in Table 6. Figure 5 and Table 6 results show that more than 99% pixels are changed in changing of one decryption key. This shows the high decryption level key sensitivity of the algorithm.

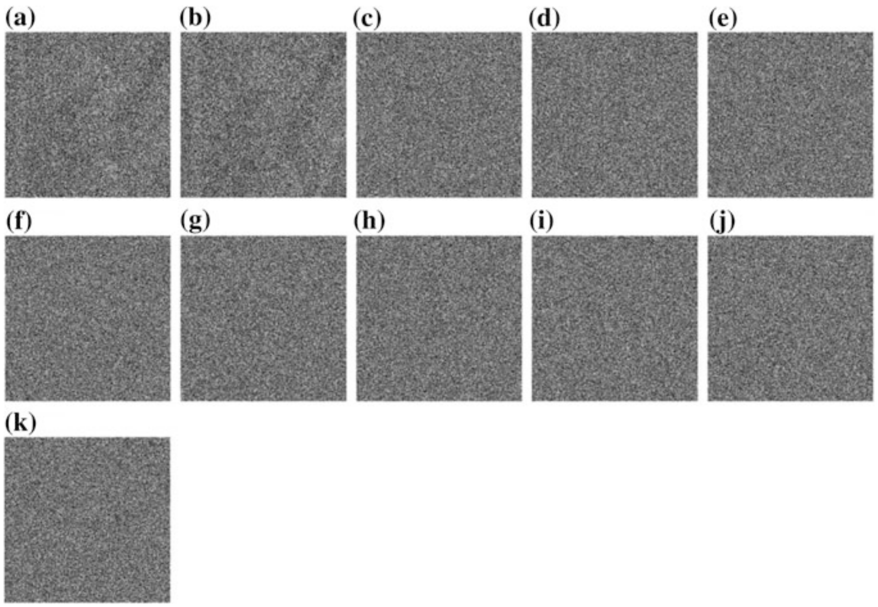

Fig. 6 Decryption level key sensitivity results of “Lena” image: Decrypted images when changing of key from **a** mue1 to $\text{mue1} + 10^{-15}$, **b** $x(1)$ to $x(1) + 10^{-15}$, **c** $xxx(1)$ to $xxx(1) + 10^{-15}$, **d** kk to $kk + 10^{-15}$, **e** $xx1$ to $xx1 + 10^{-15}$, **f** $xx2$ to $xx2 + 10^{-15}$, **g** $pp1$ to $pp1 + 10^{-15}$, **h** $pp2$ to $pp2 + 10^{-15}$, **i** $qq1$ to $qq1 + 10^{-15}$, **j** $qq2$ to $qq2 + 10^{-15}$, **k** rr to $rr + 10^{-15}$

Table 6 UACI and NPCR results of "Lena" image

Encryption keys	$\text{mue1} + 10^{-15}$	$x(1) + 10^{-15}$	$xxx(1) + 10^{-15}$	$kk + 10^{-15}$	$xx1 + 10^{-15}$	$xx2 + 10^{-15}$
NPCR (%)	99.0734	99.1058	99.5991	99.6090	99.5789	99.6246
Encryption keys	$pp1 + 10^{-15}$	$pp2 + 10^{-15}$	$qq1 + 10^{-15}$	$qq2 + 10^{-15}$	$rr + 10^{-15}$	
NPCR (%)	99.6128	99.5998	99.5995	99.5998	99.6246	

4.6 Plaintext Sensitivity Analysis

Plaintext sensitivity measures the sensitivity of pixel values in original images. This is done first by randomly changing one-pixel value and constant other pixel values in original image. Then the proposed encryption algorithm is applied to get the difference of two encrypted images (one-pixel value changed encrypted image and original encrypted image). Figure 7 shows the results of plaintext sensitivity at positions (1, 1), (64, 64), (128, 128), and (256, 256). Table 7 shows the corresponding UACI and NPCR results when the pixel values are changed at random positions. The images of Fig. 7b, d, f, h and the UACI, NPCR results of Table 7 shows that more than 99% pixels are changed in encrypted image while changing of one pixel in original image. This proves the high sensitivity of the plaintext of the proposed scheme.

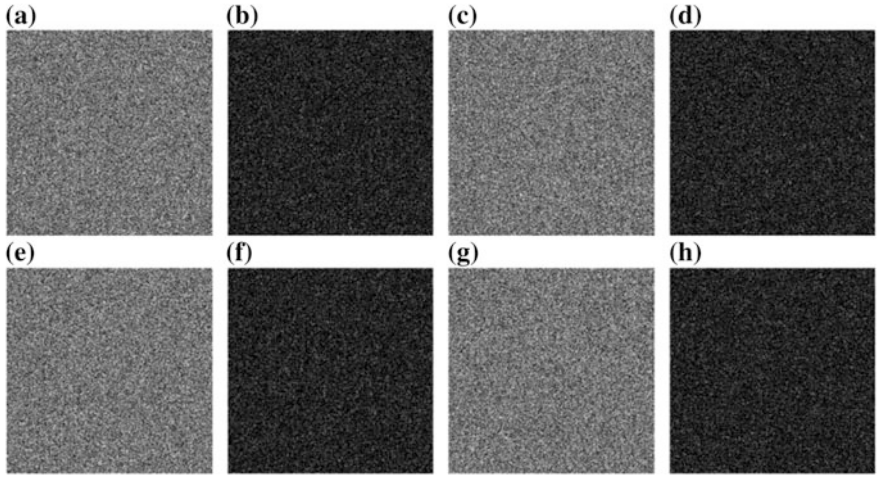

Fig. 7 Plaintext sensitivity results of "Lena" image: encrypted images when pixels changed at **a** (1, 1), **c** (64, 64), **e** (128, 128), **g** (256, 256) positions; Difference of **b** Figs. 2e and 7a, **d** Figs. 2e and 7c, **f** Figs. 2e and 7e, **h** Figs. 2e and 7g

Table 7 UACI and NPCR results of “Lena” image

Pixel positions	(1, 1)	(64, 64)	(128, 128)	(256, 256)
NPCR (%)	99.5926	99.6067	99.5995	99.6132
UACI (%)	33.4845	33.5027	33.4715	33.4923

4.7 Chosen-Plaintext Attack (CPA) and Known-Plaintext Attack (KPA) Analysis

The CPA and KPA are easily attacked by the attacker when the algorithm is not much depended on the original image. The proposed scheme highly depends on the original image in both diffusion and permutation operations. This shows the high resistivity of the proposed algorithm against CPA and KPA.

4.8 Comparison Analysis

The comparative results are shown in Table 8. The comparison is done on “Lena” image. By observing Table 8 comparison results, we found the followings:

- Higher key space of the proposed scheme as compared to the schemes in [19, 20].
- The correlation values of all the schemes are close to 0.
- Better UACI and NPCR results of the proposed scheme as compared to the schemes in [19, 20].
- The entropy values of all the schemes are close to 8.

Table 8 Comparison of “Lena (512 × 512)” image

Algorithms	Key space	Correlation coefficients			NPCR	UACI	Entropy
		Horizontal	Vertical	Diagonal			
Ours (1 round)	1.0853×2^{676}	0.0200	0.0017	0.0183	99.6077	33.3694	7.9976
Ref. [19] (1 round)	$10^{42} \approx 1.4349 \times 2^{139}$	0.0020	−0.0009	0.0016	0.4222	0.1365	7.9993
Ref. [20] (1 round)	More than $10^{120} \approx 2^{400}$	0.0013	−0.0002	−0.0003	99.5838	17.0035	–

4.9 Encryption Time

The proposed algorithm is simulated using MATLAB R2012a in a system having configuration 3.40 GHz processor with 4 GB RAM. The total simulation time required to simulate 256 × 256 and 512 × 512 sized images are 0.148 s and 0.733 s, respectively. Due to the requirement of less simulation time, the proposed algorithm is suitable to use in real-time applications.

5 Conclusion

This paper proposes a bit-level image encryption scheme using 1-D chaotic maps. In this proposed method, two-stages of bit-plane diffusion operations using PWLCM system and one-stage of bit-level column–row shuffling operation using Beta map are performed. This scheme yields good encryption outputs, produces higher key space, less simulation time to execute the algorithm, and resists all the widely known security attacks. In this algorithm, simple bit-plane diffusion operations and bit-level column–row shuffling operations are used, and hence, the proposed method provides simplicity in the algorithm. The comparison results show the better security of the proposed scheme as compared to other reported schemes. All these features show that the proposed algorithm is time-efficient, secure, and simple to encrypt grayscale images.

References

1. Coppersmith D (1994) The Data Encryption Standard (DES) and its strengths against attacks, IBM J Res Dev 38:243–250.
2. Pub NF. 197 (2001) Advanced encryption standard (AES). Federal Information Processing Standards Publication. 197, 441-0311.
3. Gao H, Zhang Y, Liang S, Li D (2006) A new chaotic algorithm for image encryption, Chaos Solitons Fractals 29:393–399.
4. Samhita P, Prasad P, Patro KAK, Acharya B (2016) A Secure Chaos-based Image Encryption and Decryption Using Crossover and Mutation Operator, Int J Cont T Appl 9:17–28.
5. Gupta A, Thawait R, Patro KAK, Acharya B (2016) A Novel Image Encryption Based on Bit-Shuffled Improved Tent Map, Int J Cont T Appl 9:1–16.
6. Shadangi V, Choudhary SK, Patro KAK, Acharya B (2017) Novel Arnold Scrambling Based CBC-AES Image Encryption, Int J Cont T Appl 10:93–105.
7. Guesmi R, Amine M, Farah B et al. (2016) Hash key-based image encryption using crossover operator and chaos, Multimed Tools Appl 75:4753–4769.
8. Guesmi R, Farah MAB, Kachouri A, Samet M (2016) A novel chaos-based image encryption using DNA sequence operation and Secure Hash Algorithm SHA-2, Nonlinear Dyn 83:1123–1136.
9. Matthews R (2010) On the derivation of a chaotic encryption algorithm, Cryptologia XIII 37–41.
10. Liu W, Sun K, Zhu C (2016) A fast image encryption algorithm based on chaotic map, Opt Lasers Eng 84:26–36.

11. Wang X, Wang S, Zhang Y, Guo K (2017) A novel image encryption algorithm based on chaotic shuffling method, Inf Secur J A Glob Perspect 0:1–10.
12. Abd El-Latif AA, Li L, Zhang T, Wang N, Song X, Niu X (2012) Digital image encryption scheme based on multiple chaotic systems, Sens Imaging 13:67–88.
13. Ahmad M, Khurana S, Singh S, AlSharari HD (2017) A Simple Secure Hash Function Scheme Using Multiple Chaotic Maps, 3D Res 8.
14. Zahmoul R, Ejbali R, Zaied M (2017) Image encryption based on new Beta chaotic maps, Optics and Lasers in Engineering 96:39–49.
15. Wang XY, Yang L (2012) Design of pseudo-random bit generator based on chaotic maps, Int J Mod Phy B 26:1250208.
16. Li S, Chen G, Mou X (2005) On the dynamical degradation of digital piecewise linear chaotic maps, Int J Bifurcation Chaos 15:3119–3151.
17. Kulsoom A, Xiao D, Abbas SA (2016) An efficient and noise resistive selective image encryption scheme for gray images based on chaotic maps and DNA complementary rules, Multimed Tools Appl 75:1–23.
18. Chai X, Chen Y, Broyde L (2017) A novel chaos-based image encryption algorithm using DNA sequence operations, Opt Laser Engg 88:197–213.
19. Zhu ZL, Zhang W, Wong KW, Yu H (2011) A chaos-based symmetric image encryption scheme using a bit-level permutation, Information Sciences 181:1171–1186.
20. Zhang YQ, Wang XY (2014) A symmetric image encryption algorithm based on mixed linear–nonlinear coupled map lattice, Information Sciences 273:329–351.
21. Patro KAK, Acharya B (2018) Secure multi–level permutation operation based multiple colour image encryption, J Inf Secur Appl 40:111–133.
22. Patro KAK, Banerjee A, Acharya B (2017) A Simple, Secure and Time Efficient Multi-way Rotational Permutation and Diffusion Based Image Encryption by Using Multiple 1-D Chaotic Maps, In International Conference on Next Generation Computing Technologies, Springer, Singapore, 396–418.

K. Abhimanyu Kumar Patro received the M.Tech. degree in Electronics and Telecommunication Engineering from the National Institute of Science and Technology, Berhampur, India.

He is currently pursuing Ph.D. in the Department of Electronics and Telecommunication Engineering, National Institute of Technology, Raipur, India. His research interests include Cryptography and Network Security, Digital Signal Processing, Digital Image Processing. He has more than 15 research publications in National/International Journals and conferences. He also served as a reviewer of several journals, including IEEE, Journal of Electronic Imaging.

Mr. Patro is a member of Cryptology Research Society of India, International Association of Engineers, Universal Association of Computer and Electronics Engineers, International Association of Computer Science and Information Technology, and Internet Society.

Bibhudendra Acharya received Post Graduation and Ph.D. degree in Electronics and Communication Engineering from National Institute of Technology, Rourkela, India.

He is currently Head and Assistant Professor in the Electronics and Telecommunication Engineering Department, National Institute of Technology, Raipur, India. His research interests include Cryptography and Network Security, Microcontroller and Embedded system, Signal Processing, Mobile Communication. He has more than 60 research publications in National/ International Journals and conferences. He also served as a reviewer of several journals, including IEEE.

Dr. Acharya is a member of Computer Society of India, The Institution of Electronics and Telecommunication Engineers (India), Institute of Engineers (India), Indian Society for Technical Education (India), Cryptology Research Society of India, International Association of Engineers, Universal Association of Computer and Electronics Engineers, International Association of Computer Science and Information Technology, and The Indian Science Congress Association.