IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 16 April 2023, accepted 4 May 2023, date of publication 16 May 2023, date of current version 24 May 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3276723

RESEARCH ARTICLE

An Efficient and Secure Method of Plaintext-Based Image Encryption Using Fibonacci and Tribonacci Transformations

CHINMAY MAITI[1], BIBHAS CHANDRA DHARA[2], SAIYED UMER[3], AND VIJAYAN ASARI[4]

[1]Department of Computer Science and Engineering, College of Engineering and Management, Kolaghat 721171, India
[2]Department of Information Technology, Jadavpur University, Kolkata 700106, India
[3]Department of Computer Science and Engineering, Aliah University, Kolkata 700156, India
[4]Department of Electrical and Computer Engineering, University of Dayton, Dayton, OH 45469, USA

Corresponding authors: Bibhas Chandra Dhara (bcdhara@gmail.com) and Saiyed Umer (saiyedumer@gmail.com)

The work of Vijayan Asari was supported by the Department of Electrical and Computer Engineering, University of Dayton, Dayton, OH, USA.

ABSTRACT In this article, we have proposed an efficient and secure method of image encryption. This image encryption method is new, where the plain image is confused using Fibonacci Transformation, and Tribonacci Transformation modifies the pixel values. The Fibonacci numbers and Tribonacci numbers of the above transformations are determined using the hash value of the plain image. To our knowledge, this is the first time a Tribonacci Transformation has been used in image encryption. The performance of the present method is evaluated using some standard test images and some special images. The present method has high key space and is sensitive to the plain image. The proposed method is also robust against attacks like brute force attacks, statistical attacks, differential attacks, noise attacks, cropping attacks, and known/chosen-plaintext attacks. The performance of the proposed method is equally efficient as the state-of-the-art methods, which establishes the applicability of the present method.

INDEX TERMS Image encryption, Fibonacci transformation, Tribonacci transformation, SHA256, non-chaotic encryption.

I. INTRODUCTION

Due to the fast growth of Internet technology, popularity of the digital devices, and style of multimedia information exchanges through the Internet, increasing the information confidentiality requirement. Since the image is an essential communication medium, image security is a significant concern. In the literature, different techniques like secret image sharing [1], [2], [3], image steganography [4], [5], and image encryption [6], [7] are widely used to provide security in image communication. An image contains bulk data, and data is highly correlated and redundant. So, classical data encryption techniques like data encryption standard (DES), triple DES (3DES), and advanced encryption standard (AES) [8], [9] are not suitable for image encryption because of their slow computation [10]. A particular class of encryption methods, called image encryption, is designed to protect images.

The associate editor coordinating the review of this manuscript and approving it for publication was Jun Wang.

The image encryption algorithm transforms an image into a cipher image using an encryption key. The cipher image looks like a noisy image, so nothing cannot be predicted about the original image from the cipher image. We can achieve this by changing the position of the pixels and by modifying the intensity value of the pixels. The pixel scrambling method reduces the correlation among the adjacent pixels, resulting in a noisy image. The process of pixel scrambling is known as the confusion phase of image encryption. Since the confusion phase generates a noisy image without changing the pixel intensity, the histogram remains the same as the original image. The diffusion phase is another step of the image encryption method in which the pixels' intensity value is modified, resulting in a uniform histogram, and the system becomes robust against different attacks.

There are numerous uses of image encryption in the corporate world. In the medical industry [11], [12], health records are generated and circulated online. This record includes patient information, medical history, symptoms, and many

more. Due to the confidentiality of these health records, it is a significant task to protect them from unauthorized use. In the military field, images like maps, the positions of buildings, and enemies are critical. Images are essential in small target detection, tracking, and missile guidance. Therefore, illegal access to those images may endanger national security. In the media industry, the news is broadcast 24 × 7, and privacy of multimedia (image, audio, or video) content is essential. Image encryption is vital in protecting information from unwanted access [13]. In the cloud a third party stores client data in the cloud. Cloud has provided feasible storage for images. The privacy protection of the images in cloud applications is also essential [14], [15], [16].

In this article, we have proposed a new method for image encryption method. In this work, pixels are scrambled using Fibonacci Transform (FT), and this transformation can be achieved by matrix multiplication of two matrices of size 2×2 and 2×1. In the diffusion phase, the scrambled image is XOR-ed with a random image (generated using the key) to ensure the usability of the proposed method for any grayscale image. The significant contribution of this work is the use of the Tribonacci Transform (TT) to modify the pixels' intensity value. In this pixel modification process, a 3×3 Tribonacci matrix is multiplied with a 3×1 vector, defined by three consecutive pixels. Bitwise XOR, (circular) bit-shift, and substitution (S-Box) are commonly used in the diffusion process. To our knowledge, this is the first time a Tribonacci Transformation has been used in the diffusion process and in image encryption, which is our fundamental contribution. In this work, the 'key' for the encryption is computed from the plain image itself. For this purpose, the hash value of the image is computed. Here, SHA256 is used to compute the hash value, which returns a hash value of 256 bits, sufficiently large to protect against brute-force attacks. One of the important features of our method is that the proposed method has enough strength to handle any grayscale images (including pure black/white images) with equal efficiency. The proposed technique is simple and easy to implement. Experimental result shows that the proposed method is fast enough in terms of execution time. Analysis of the results supports that the present work is also robust against different attacks. The significant contributions of this article are highlighted below:

1) A FT is used to scramble the pixels.
2) A TT is applied to change the intensity value of the pixels.
3) The proposed method possesses high key space, and the present method depends on the plain image.
4) This method can efficiently encrypt any grayscale image (including pure black/white images).
5) The proposed method is a faster one.
6) The method is robust against brute force attacks, statistical attacks, differential attacks, known/chosen-plaintext attacks, etc.

The rest of this article is organized as follows: The current state of the literature is discussed in Section II. The background mathematics of the Fibonacci Transform and Tribonacci Transform are studied in Section III. The proposed image encryption method is presented in Section IV. In Section V, the performance of the proposed method is reported. The security analysis and the comparative study with state-of-the-art methods are given in Section VI. Finally, this article is concluded in Section VII.

II. LITERATURE SURVEY

At an earlier stage, the image encryption methods were developed on compressed data [17], [18], [19]. Among different classes of image encryption, chaotic-based techniques are quite popular due to their sensitivity to the initial parameters, pseudo-random behavior, and unpredictable motion trajectories. In the chaotic methods, a pseudo-random sequence is computed, and certain permutations are defined using this sequence. Moreover, these permutations are used to scramble the pixels, permute the bit planes, or define the substitution matrices. One intriguing aspect of chaotic methods is that a slight change in the initial value of the parameter(s) generates quite different sequences. This chaotic behavior is matched with an encryption feature. Earlier researchers have worked using classical chaotic maps like Baker's map [20], Logistic map [21], Tent map [22], delayed coupled map [23], [24], 2D logistic-Sine-coupling map [25], 1D chaotic map [26], Lorenz chaotic system [27], etc. Recently, people have been designing different hybrid chaotic image encryption methods, like the hybrid chaotic map with an optimized substitution box [28]. In [29], a hybrid chaotic method using a 2D modified Henon map with a Sine map is proposed. Combining the Sine map, Logistic map, and Tent map, a new hybrid method is proposed in [30]. A hybrid method [31] using 1D and 2D chaotic maps to achieve image encryption is proposed, where a 2D sin-cosine cross-chaotic map is used in the confusion phase, and a 1D Logistic-Tent map is used to diffuse the image. Recently, high-dimensional chaotic (hyper-chaotic) systems have gained popularity [32], [33] because these systems increase the key space and also become robust against attacks. In [34], a 5D hyper-chaotic system is proposed to encrypt color images in which the plain image is decomposed into sub-bands using a complex wavelet transform. Then sub-bands are diffused using secret keys obtained from a 5D chaotic map. A novel adjustable visual encryption scheme using a 6D hyperchaotic system is proposed in [35]. A new image cryptosystem using a hyperchaotic system and a Fibonacci Q-matrix is proposed in [36].

Another class, based on DNA computing, of image encryption methods is now getting popular. In recent years, many DNA sequence-based encryption methods have been designed. In [37], a DNA sequence-based image encryption method is proposed. In this method, the image is converted into a DNA matrix, and then a 2D Logistic map is used to define circular row and column permutations. Finally,

a row-by-row diffusion technique is adopted. The initial parameters are calculated from the SHA256 of the plain image. A 2D Henon-Sine map and DNA coding-based image encryption method are proposed in [38], where S-box is first employed to synthesize cryptographic effects of the complicated DNA encryption operations, and security evaluation is conducted using 2D Henon-Sine map and DNA coding. In [39], the research proposes a novel hybrid method combining the power of DNA and the randomness of a Binary Search Tree (BST), which creates a more accurate encryption method. In [40], a two-phase secure image encryption method is proposed where the concepts of DNA and RNA are used. In this method, the initial cipher image is created by applying DNA rules, the DNA XOR operator, and the chaotic function. Then, the codon's truth table for RNA and secret key are exploited to obtain the final encrypted image. A new Chaotic Evolutionary Biomolecules Model (CEBM) based on the concepts of biological molecules (DNA and RNA) is presented in [41] for image encryption. In this work, the Chaotic Rate operator is used for permutation, and DNA XOR and RNA codon complement operators are used for diffusion. Some other recent DNA-based encryption methods are noticed in [42], [43], [44], [45], [46], and [47].

Nowadays, medical images are also vital data that are communicated through the internet. So, the security of medical images like MRI, ultrasound sonography, X-ray, etc., is also essential because these images contain private information. Some recent proposals on medical image encryption are given in [11], [48], [49], [50], [51], and [52]. Chaos-based image encryption methods are popular in the research community. At the same time, researchers also proposed non-chaotic methods to scramble the images (i.e., to define the permutation). Non-chaotic techniques such as the Arnold Transform [53], [54], [55], Fibonacci Transform [56], [57], [58], Gray code [59], Rubik's cube principle [60], cyclic group [7], prime factorization [61], binary tree traversal [62], and iterative numerical methods for root finding [6], [63], [64] are used to define the permutation. The SCAN-based permutation is another group of techniques used in image encryption. Different encryption techniques based on different scan patterns are studied in [65] and [66]. A sine curve-based pattern is utilized in [67] to encrypt an image. In [68], a circular scan pattern is employed to define the permutation. A double spiral scan-based method is proposed in [69]. Recently, a novel zig-zag scan-based feedback convolution algorithm has been designed for image encryption [70].

III. BACKGROUND KNOWLEDGE

Here, we study the mathematics of the Fibonacci Transform and Tribonacci Transform in the following two subsections.

A. FIBONACCI NUMBERS

In mathematics, the sequence

$$\{1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, \cdots\}$$

TABLE 1. Fibonacci and Tribonacci numbers.

k	$\cdots$	-5	-4	-3	-2	-1	0	1	2	3	4	5	$\cdots$
f_k	$\cdots$	5	-3	2	-1	1	0	1	1	2	3	5	$\cdots$
t_k	$\cdots$	-3	2	0	-1	1	0	0	1	1	2	4	$\cdots$

is known as the Fibonacci sequence [71], [72]. The above sequence can recursively be defined in Eq. (1).

$$f_k = \begin{cases} 1, & \text{if } k = 1 \\ 1, & \text{if } k = 2 \\ f_{k-1} + f_{k-2}, & \text{otherwise} \end{cases} \tag{1}$$

The above relation is extended for any

$$k \in \{\cdots, -3, -2, -1, 0, 1, 2, 3, \cdots\},$$

the numbers are given in Table 1. The Fibonacci sequence is powerful and possesses some amazing properties. Researchers have used elementary matrix operations, determinants, and their properties to generate class identities for generalized Fibonacci numbers [71]. In this work, we study three properties of generalized Fibonacci sequences such as Cassini's identity, Catalan's identity, and d,Ocagnes's identity, which are given below.

$$\text{Cassini's identity: } f_{q+1}f_{q-1} - f_q^2 = (-1)^q \tag{2}$$

$$\text{d'Ocagne's identity: } f_{p+1}f_q - f_p f_{q+1} = (-1)^p f_{q-p} \tag{3}$$

$$\text{Catalan's identity: } f_q^2 - f_{q+p}f_{q-p} = (-1)^{q-p} f_p^2 \tag{4}$$

The above identities can be expressed as the determinants as given below.

$$\text{Cassini's identity: } T_{Cas} = \begin{vmatrix} f_{q+1} & f_q \\ f_q & f_{q-1} \end{vmatrix} \tag{5}$$

$$\text{d'Ocagne's identity: } T_{dOc} = \begin{vmatrix} f_{p+1} & f_{q+1} \\ f_p & f_q \end{vmatrix} \tag{6}$$

$$\text{Catalan's identity: } T_{Cat} = \begin{vmatrix} f_q & f_{q+p} \\ f_{q-p} & f_q \end{vmatrix} \tag{7}$$

The corresponding matrix of any of the above identities can be used to transform data in cryptography under modulo n if and only if the $gcd(\det, n) \equiv 1 \bmod n$, where 'det' is the determinant of the matrix. From the definition of the Fibonacci series, we note that $f_1 = 1$ and $f_2 = 1$. So, for any n, we may note the following about the transformation matrices:

1) $T_{Cas} = \pm 1$, for any value of q.
2) $T_{dOc} = \pm 1$, when $q - p = 1$ or 2.
3) $T_{Cat} = \pm 1$, if $p \in \{1, 2\}$.

From the above definitions (Eq. (2)-(4)), it is obvious that Cassini's identity is a special case of the other two identities as given below.

$$\text{d'Ocagne's identity} \rightarrow \text{Cassini's identity, when } p + 1 = q$$
$$\text{Catalan's identity} \rightarrow \text{Cassini's identity, when } p = 1$$

B. TRIBONACCI NUMBERS

Tribonacci numbers [73] are a generalization of Fibonacci numbers, denoted as t_k, for $k = 0, 1, 2, 3, \ldots$, and defined by the recurrence relation as given below in Eq. (8).

$$t_{k+1} = t_k + t_{k-1} + t_{k-2}, \quad \text{where } t_0 = t_1 = 0,\ t_2 = 1 \tag{8}$$

The Tribonacci negative numbers t_{-k}, for $k = 1, 2, 3, \ldots$, satisfies the recurrence relation given in Eq (9).

$$t_{-k} = \begin{vmatrix} t_{k+1} & t_{k+2} \\ t_k & t_{k+1} \end{vmatrix} \tag{9}$$

Eq (8) and (9) provide the Tribonacci numbers t_k, for $k = 0, \pm1, \pm2, \pm3, \ldots$ shown in Table 1.

In [73], a Tribonacci coding technique is proposed, and this method is based on the Tribonacci numbers. For this purpose, a matrix $H_{3\times3}$ is introduced. The matrix H is defined as

$$H = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} = \begin{pmatrix} t_3 & t_2 + t_1 & t_2 \\ t_2 & t_1 + t_0 & t_1 \\ t_1 & t_0 + t_{-1} & t_0 \end{pmatrix} \tag{10}$$

where $det(H)$ =1 and the inverse of H is given as (11), shown at the bottom of the next page.

The above article has also provided two important methods to compute H^k ($k \in \mathbb{Z}$). The positive power of H (i.e., $H^k : k \in \mathbb{N}$) is computed as

$$H^k = \begin{pmatrix} t_{k+2} & t_{k+1} + t_k & t_{k+1} \\ t_{k+1} & t_k + t_{k-1} & t_k \\ t_k & t_{k-1} + t_{k-2} & t_{k-1} \end{pmatrix} \tag{12}$$

The negative power of H (i.e., $H^{-k} : k \in \mathbb{N}$) is computed as (13), shown at the bottom of the next page.

The Eqs. (12) and (13) can be easily established by using mathematical induction. From the above definitions of H^p, $p \in \mathbb{Z}$, the following properties can be easily proved.

P1: $H^p = H^{p-1} + H^{p-2} + H^{p-3}$
P2: $H^p H^q = H^q H^p = H^{p+q}$ $(p, q = \pm0, \pm1, \pm2, \cdots)$
P3: $det\ H^p = 1$

So, from the above discussion, we may note that a Tribonacci matrix H^i for any i can be used to transform data into another domain, and the original data can be retrieved since H^i is invertible. Therefore, H^i can be used in the encryption method.

IV. PROPOSED ENCRYPTION METHOD

The proposed encryption method consists of three phases:

1) Key generation phase, in this phase, the key for the encryption is generated, and this is derived from the plain image.
2) The confusion phase scrambles the pixel positions using the Fibonacci Transformation.
3) The diffusion phase is applied to modify the intensity values of the pixels, where the Tribonacci Transformation is used.

The block diagram of the proposed method is shown in Fig. 1.

The proposed method mainly uses the Fibonacci Transformation and Tribonacci Transformation, and let us refer to the proposed method as 'FTTTIE' (Fibonacci Transformation and Tribonacci Transformation based Image Encryption). The phases of the proposed image encryption method and the summary of the proposed image encryption method are presented in the following sub-sections.

A. KEY GENERATION

In the present method, the key to encrypt the image is derived from the given plain image. The hash value of the image is computed using the SHA256 method, and the hash value of the image is *key*. The size of the key is 256 bits. The hash value is sensitive and depends on the input image. The *key* is partitioned into two halves (key_1 and key_2), each with 128 bits. To get key_i ($i = 1, 2$), we may consider $key_1 = key(1 : 128)$ and $key_2 = key(129 : 256)$ or $key_1 = key(1 : 2 : 256)$ and $key_2 = key(2 : 2 : 256)$ or some other techniques. In this work, we have considered the first one (i.e., $key_1 = key(1 : 128)$ and $key_2 = key(129 : 256)$). From these two keys, a single key is derived as $key_1 \oplus key_2$, which is considered as the confusion key (key_c) (i.e., $key_c = key_1 \oplus key_2$). The key key_2 is considered the diffusion key key_d.

In this phase, the keys key_c and key_d are used to define a key image (key_{img}) of the same size as the plain image. This key image is used in the diffusion phase to modify the pixel intensity. A sequence of Tribonacci numbers $Tri = \{t_{T+1}, \cdots, t_{T+S}\}$ and the threshold value T is computed using key_d, and t_j is the j^{th} Tribonacci number (computed using Eq. (8)) and, S is the number of pixels in the plain image. The sequence mod(Tri, 256) defines an image of the same size as the plain image. The algorithmic sketch of the key generation phase is given in **Algorithm 1: KeyGeneration()**. From the symmetries of the above steps, it may be noted that if the same key '*key*' is used then key_c, key_d, and the key image key_{img} can be obtained.

Algorithm 1 : KeyGeneration (*Img*)

Input: *Img* the plain image
Output: Keys $\{key_c, key_d\}$ and key image key_{img}

Step 1. $key_c(1 : 128) = 0$, $key_d(1 : 128) = 0$
Step 2. $[M\ N]$=Size(*Img*), $S = M \times N$
Step 3. $key(1 : 256)$= SHA256(*Img*)
Step 4. For i= 1 to 128
  a. $key_c(i) = key(i)$
  b. $key_d(i) = key(128 + i)$
Step 5. key_c=$key_c \oplus key_d$
Step 6. //Compute the key image
  a. Determine threshold T using key_d
  b. Compute the Tribonacci sequence $Tri = \{t_{T+1}, \cdots, t_{T+S}\}$
  c. $key_{img_{M\times N}} \leftarrow$ mod(Tri, 256)
Step 7. Return $\{key_c, key_d, key_{img}\}$

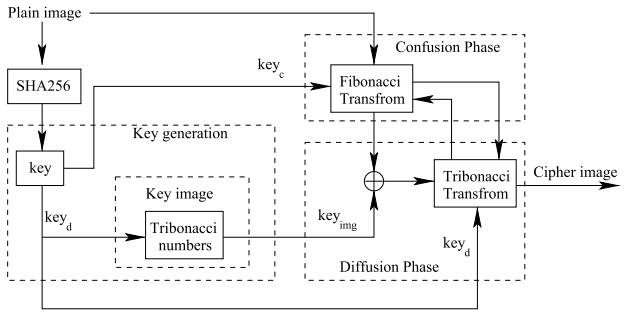

FIGURE 1. Block diagram of the proposed image encryption method.

Algorithm 2 : ConfusionPhase ($Img_{N\times N}, key_c, Img_{conf}, pr$)

Input: Plain image Img, confusion key key_c, prime pr

Output: Confused image Img_{conf}

Step 1. $q = \text{mod}\,(key_c, pr) + 3$

Step 2. For $r = 1$ to N

a. For $c = 1$ to N

i. Compute r' and c' using Eq. (14)

ii. $Img_{conf}(r', c') = Img(r, c)$

Step 3. Return Img_{conf}

B. THE CONFUSION PHASE

In the confusion phase, we applied a Fibonacci Transformation to permute the positions of the pixels. We note that any matrices corresponding to T_{Cas}, T_{dOc}, or T_{Cat} can be used as a transformation matrix to reposition the pixels. It is also worth noting that a single variable can parameterize these transformations. Again, Cassini's identity is a particular case of the other two identities. In this experiment, we employed Cassin's identity in the confusion phase. The confusion key key_c is used to determine the value q of the T_{Cas}, and then pixels are scrambled. To compute q, we consider q =mod(key_c, pr)+3 where pr is a prime number with moderate value and make it public. For an image of size $N \times N$, let (r, c) be the coordinate of a pixel, and after transformation, the new coordinate is (r', c'), then

$$\begin{pmatrix} r' \\ c' \end{pmatrix} = \begin{pmatrix} f_{q+1} & f_q \\ f_q & f_{q-1} \end{pmatrix} * \begin{pmatrix} r \\ c \end{pmatrix} \text{ mod N.} \tag{14}$$

The confusion process of the proposed method is illustrated in **Algorithm 2: ConfusionPhase()**. Concerning the same key, the same transformation can be derived; therefore, the confusion phase is invertible.

C. THE DIFFUSION PHASE

The confusion phase returns a scrambled image (Img_{conf}), where the intensity of the pixels remains the same. As a result, the intensity profile of both the plain and confused images remains unchanged. Therefore, an attacker may guess the original image from the histogram profile of the confused image. To resolve this problem, we need to modify the intensity of the pixels so that nothing can be predicted about the plain image. In this phase, two operations have been executed to achieve the goal. The operations are: i) the scrambled image is XOR-ed with the image key_{img}; ii) the pixels' values are changed using the Tribonacci Transformation.

At the first step of the diffusion process, Img_{conf} is XOR-ed with key_{img}, and this step changes the intensity of the pixels of the confused image. This XOR operation is applied at the very fast iteration, and for the remaining iterations of the diffusion phase, this XOR operation is ignored. The reason for adopting this step is discussed later. The image obtained from the previous step is transformed by applying the Tribonacci Transformation (TT), as given in Eq. (15).

$$\begin{pmatrix} p_1' \\ p_2' \\ p_3' \end{pmatrix} = H^k * \begin{pmatrix} p_1 \\ p_2 \\ p_3 \end{pmatrix} \text{ mod } 256 \tag{15}$$

where $\{p_1, p_2, p_3\}$ are pixels' intensity, H^k is the transformation matrix, and the value of k is determined from key_d. The inverse transformation (called ITT) is given in Eq. (16).

$$\begin{pmatrix} p_1 \\ p_2 \\ p_3 \end{pmatrix} = H^{-k} * \begin{pmatrix} p_1' \\ p_2' \\ p_3' \end{pmatrix} \text{ mod } 256 \tag{16}$$

$$H^{-1} = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & -1 & -1 \end{pmatrix} = \begin{pmatrix} t_0^2 - t_{-1}t_1 & t_{-1}t_2 - t_0t_1 & t_1^2 - t_0t_2 \\ t_1^2 - t_0t_2 & t_0t_3 - t_1t_2 & t_2^2 - t_1t_3 \\ t_0t_2 + t_{-1}t_2 - t_1^2 - t_0t_1 & t_1^2 + t_1t_2 - t_0t_3 - t_{-1}t_3 & t_1t_3 + t_0t_3 - t_2^2 - t_1t_2 \end{pmatrix} \tag{11}$$

$$H^{-k} = \begin{pmatrix} t_{k-1}^2 - t_{k-2}t_k & t_{k-2}t_{k+1} - t_{k-1}t_k & t_k^2 - t_{k-1}t_{k+1} \\ t_k^2 - t_{k-1}t_{k+1} & t_{k-1}t_{k+2} - t_kt_{k+1} & t_{k+1}^2 - t_kt_{k+2} \\ (t_{k-1} + t_{k-2})t_{k+1} - & t_k(t_k + t_{k+1}) - & (t_k + t_{k-1})t_{k+2} - \\ (t_k + t_{k-1})t_k & (t_{k-1} + t_{k-2})t_{k+2} & (t_{k+1} + t_k)t_{k+1} \end{pmatrix} \tag{13}$$

In the above transformation, we need three pixels at a time. So, the pixels of the image are partitioned into groups, and each group has three pixels, except the last group may have fewer pixels. Assume we have r groups, $G_1, G_2, \cdots, G_r$ and pixels in group G_i are denoted as $\{p_{i,1}, p_{i,2}, p_{i,3}\}$ (i.e., $G_i = \{p_{i,1}, p_{i,2}, p_{i,3}\}$). There are three cases:

1) The Number of pixels in the image is $3r$,
   a) At the time of encryption, each group G_i ($1 \le i \le r$) is transformed by Eq. (15) and obtained G'_i, i.e., $G'_i = \text{TT}(G_i)$.
   b) At the time of decryption, G'_i (for $1 \le i \le r$) is transformed back into G_i with the help of Eq. (16) (i.e., $G_i = \text{ITT}(G'_i)$).
2) The Number of pixels is $3(r-1)+1$, i.e., $G_r = \{p_{r,1}\}$
   In the encryption process,
   a) First $(r-1)$ groups $\{G_1, G_2, \cdots, G_{r-1}\}$ are transformed by Eq. (15).
   b) Then, G_r is redefined as $G_r = \{p'_{r-1,2}, p'_{r-1,3}, p_{r,1}\}$ and transformed as $G'_r = \{p''_{r-1,2}, p''_{r,3}, p'_{r,1}\} = \text{TT}(G_r)$ using Eq. (15).
   c) The diffused image is defined as

$$G'_1||G'_2||\cdots||G'_{r-2}||G'_r||\{p'_{r-1,1}\}.$$

   In the decryption process,
   a) First $r-2$ groups of the diffuse image are inversely transformed as $\{G_1, G_2, \cdots, G_{r-2}\}$ (i.e., $G_i = ITT(G'_i)$, for $1 \le i \le r-2$).
   b) Then, compute $\{p'_{r-1,2}, p'_{r-1,3}, p_{r,1}\} = \text{ITT}(G'_r)$.
   c) Next, compute

$$\{p_{r-1,1}, p_{r-1,2}, p_{r-1,3}\} = \text{ITT}(p'_{r-1,1}, p'_{r-1,2}, p'_{r-1,3}\}).$$

   The previous version of the diffused image can be obtained as

$$G_1||G_2||\cdots||G_{r-2}||\{p_{r-1,1}, p_{r-1,2}, p_{r-1,3}\}||\{p_{r,1}\}$$

3) Consider the case of $3(r-1)+2$ pixels, i.e., $G_r = \{p_{r,1}, p_{r,2}\}$
   In the encryption process,
   a) Transform first $(r-1)$ groups and gives $G'_1, G'_2, \cdots, G'_{r-1}$.
   b) $p'_{r-1,3}$ of G'_{r-1} is combined with G_r and gives $G_r = \{p'_{r-1,3}, p_{r,1}, p_{r,2}\}$
   c) G_r is transformed by Eq. (15), obtained

$$Gr' = \{p''_{r-1,3}, p'_{r,1}, p'_{r,2}\}.$$

   d) The diffuse image is

$$G'_1||G'_2||\cdots||G'_{r-2}||Gr'||\{p'_{r-1,1}, p'_{r-1,2}\}.$$

   In the decryption process,
   a) First $r-2$ groups of the diffuse image are inversely transformed as $\{G_1, G_2, \cdots, G_{r-2}\}$ (i.e., $G_i = \text{ITT}(G'_i)$, for $1 \le i \le r-2$).
   b) Then, compute $\{p'_{r-1,3}, p_{r,1}, p_{r,2}\} = \text{ITT}(G'_r)$.
   c) Next, compute

$$\{p_{r-1,1}, p_{r-1,2}, p_{r-1,3}\} = \text{ITT}(p'_{r-1,1}, p'_{r-1,2}, p'_{r-1,3}\}).$$

The previous version of the diffused image can be obtained as

$$G_1||G_2||\cdots||G_{r-2}||\{p_{r-1,1}, p_{r-1,2}, p_{r-1,3}\}||\{p_{r,1}, p_{r,2}\}$$

Algorithm 3 : DiffusionPhase (Img_{conf}, key_{img}, key_d, i)

Input: Confused image Img_{conf}, key image key_{img}, diffusion key key_d, iteration number i
Output: Cipher image Img_{enc}

Step 1. If ($i == 1$) $Img_{temp} = Img_{conf} \oplus key_{img}$
  1) Else $Img_{temp} = Img_{conf}$
Step 2. k =mod(key_d, 5)+1 //Determine transformation matrix H^k
Step 3. $Img_{enc} \leftarrow \phi$; $temp \leftarrow \phi$
Step 4. Select 3 pixels $\{p_1, p_2, p_3\}$ from Img_{temp}
  a. $Img_{enc} = Img_{enc} \,||\, temp$
  b. $\{p'_1, p'_2, p'_3\} = TT(\{p_1, p_2, p_3\})$
  c. $temp = \{p'_1, p'_2, p'_3\}$
  d. If three or more pixels are available in Img_{temp} GOTO Step 4
Step 5. Is there is no pixel in Img_{temp}
  a. $Img_{enc} = Img_{enc} \,||\, temp$
  b. GOTO Step 8
Step 6. If there is one pixel, p_1, in Img_{temp}
  a. $\{p''_2, p''_3, p'_1\} = TT(\{temp(2), temp(3), p_1\})^{\xi}$
  b. $Img_{enc} = Img_{enc} \,||\, \{p''_2, p''_3, p'_1\} \,||\, \{temp(1)\}$
  c. GOTO Step 8
Step 7. If there are two pixels, $\{p_1, p_2\}$, in Img_{temp}
  a. $\{p''_3, p'_1, p'_2\} = TT(\{temp(3), p_1, p_2\})$
  b. $Img_{enc} = Img_{enc} \,||\, \{p''_3, p'_1, p'_2\} \,||\, \{temp(1), temp(2)\}$
Step 8. Return Img_{enc}

ξ '$temp(i)$' represents the i^{th} element of $temp$

In the above process, to handle the pixels, when we have either one or two pixels in the last group, we merge these pixels with some transformed pixels from the immediate previous group. This technique ensures that the size of the input and output images will remain the same. This technique is known as 'cipher stealing' in cryptography. The outline of the proposed diffusion phase is presented in **Algorithm 3: DiffusionPhase()**. Here, the two operations, namely, TT and XOR, are used, and these operations are invertible. Therefore, the diffusion process of the proposed method is also invertible.

D. IMAGE ENCRYPTION

In this section, we summarize the proposed image encryption technique. From the plain image, the hash value of the image

is computed using the SHA256 algorithm, and then keys (key_c, key_d) and key image (key_{img}) are computed. Then, the confusion and diffusion phases are executed within the loop. The algorithmic structure of the proposed image encryption method is demonstrated in **Algorithm 4: ImageEncryption()**. Since each step of the proposed image encryption method is invertible, we also have an image decryption method to reconstruct the plain image.

Algorithm 4 : ImageEncryption ($Img_{N\times N}$, Img_{enc}, itr)

Input: Plain image Img, no of iterations itr
Output: Cipher image Img_{enc}

Step 1. $[key_c, key_d, key_{img}]$ = KeyGeneration(Img)
Step 2. For i = 1 to itr
  a. Img_{conf} = ConfusionPhase(Img, key_c)
  b. Img_{enc} = DiffusionPhase(Img_{conf}, key_{img}, key_d, i)
  c. $Img = Img_{enc}$
Step 4. Return Img_{enc}

V. EXPERIMENTAL RESULT

In this section, the performance of the proposed method is presented, and the performance is compared with the state-of-the-art (SoA) methods. In this experiment, we have used ten gray-scale images as test images of size 512 × 512. Among these images, five images are standard test images commonly used by the image processing community (USC-SIPI database) [74], and the other five images are synthetic (generated by the computer program). The test images are shown in Fig. 2. Images shown in Fig. 2(a)-(e) are standard test images, (f) represents an entirely black image with an intensity value of 0 (binary representation is '00000000' for an 8-bit image), (g) is a purely white image where pixel intensity is 255 (binary representation '11111111'), (h) is a checkerboard image with 50% black pixels and 50% white pixels, (i) is a constant image with an intensity value of 170 (binary representation '10101010'), and (j) is also a constant an image whose intensity value is 85 (binary representation '01010101'). In this figure, we include the bounding box to show the presence of the 'White' image.

In the proposed method, first, we compute the hash value of the plain image using SHA256, and this hash value is considered the key of the proposed method. The size of the key is 256 bits. From this key, the keys of the confusion phase (key_c) and diffusion phase (key_d) are computed, and a key image (key_{img}) is generated. In the confusion phase, to define the Fibonacci transform, we set $pr = 5$, which is public information. In the diffusion phase, the transformation matrix is H^k where k=mod(key_d, 5)+1. The number of iterations is 3 (i.e., $itr = 3$). The output of the proposed method on the test images is shown in Fig. 3. From this result, we may note that all the output images are noisy irrespective of the input image, so the plain image cannot be predicted from the encryption image. Due to the space problem, henceforth, for illustration purposes, we have used five images, say, 'Lena', 'Cameraman', 'Black', 'Checkerboard', and 'Constant170'. The reconstructed images with the correct and incorrect decryption keys are shown in Fig. 4(a) and (b), respectively. This result shows that the encrypted images are reconstructed exactly using the correct decryption key. Nothing can be guessed about the original image from the reconstructed image when we use the wrong decryption key. The performance of the proposed method is convincing and may be applicable to different applications.

VI. SECURITY ANALYSIS

To establish the usability of the proposed method, in this section, we have analyzed the performance of the proposed method in terms of key space analysis, execution time, statistical analysis, and key sensitivity. We also test the robustness of the proposed method against different attacks like differential attacks, known/chosen plaintext attacks, noise attacks, and cropping attacks. Also, we have compared the performance with state-of-the-art (SoA) methods M1 [70], M2 [40], M3 [41], M4 [39], M5 [63], M6 [6], and M7 [75]. We have summarized the encryption process of the SoA methods and the proposed method in Table 2. This presentation will help to visualize all these methods and make them easy to understand for comparison purposes.

A. KEY SPACE ANALYSIS

As mentioned earlier, the keys for the confusion phase (key_c) and diffusion phase (key_d) are derived from the SHA value of the plain image. Since SHA256 is used to find the hash value of the plain image, the size of the key space of the proposed method is 256 bits, which is significantly high and sufficient to resist the brute-force attack.

B. COMPLEXITY AND EXECUTION TIME

From the above discussion about the proposed method, we note the following.

1) First, we compute the key, using SHA256, from the given plain image. The complexity of this computation is O($M \times N$), where $M \times N$ is the size of the image.
2) Next, we define a random image (key_{img}) of the same size as the input image. Its complexity is O($M \times N$).
3) In the confusion phase, pixels are scrambled by using Eq. (14), which is nothing but a multiplication between 2×2 and 2×1 matrices. So, for each iteration of the confusion phase, the complexity is O($M \times N$).
4) Case I. In the diffusion phase, at the very first iteration, the random image (key_{img}) is XOR-ed with the confused image, and its complexity is O($M \times N$).
5) Case II. In all iterations the pixels' values are modified using Eq. (15), a multiplication of two matrices of size 3×3 and 3×1. So, O($M \times N$) is the complexity of each iteration of the diffusion phase.

Therefore, if the confusion-diffusion phase is iterated itr times, then the complexity of the proposed image encryption

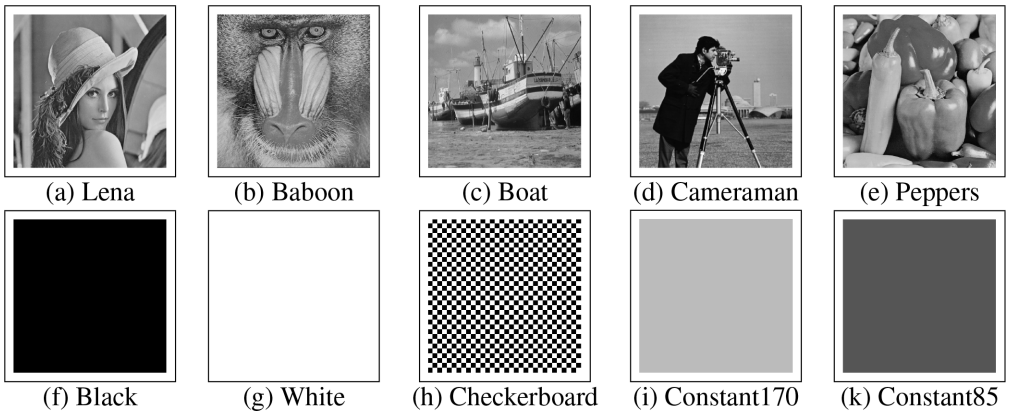

FIGURE 2. Test images used in this experiment.

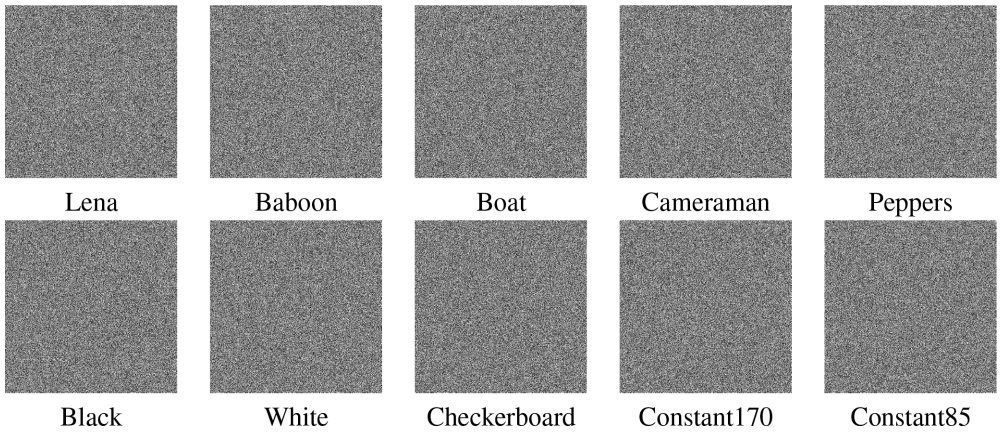

FIGURE 3. Encrypted images.

is O($M \times N$)+ O($M \times N$)+ O($M \times N$)+ $itr \times$(O($M \times N$) + O($M \times N$)). Hence, the complexity of the proposed image encryption is O(($itr+3$) $\times M \times N$). In this experiment, we set $itr = 3$ (see Section V). The actual execution time of an image encryption method is very crucial. Using the above parameters, we do the experiment in two environments:

1) Using MATLAB Version: 8.5.0.197613 (R2015a) in the platform of Windows 7 Professional Version 6.1 with the system has Intel(R) Core(TM) i5-3230M CPU @ 2.60GHz and 4GB RAM. Let us refer to the proposed method under this environment as FTTIE (Fibonacci and Tribonacci Transform-based Image Encryption).
2) Using MATLAB Version: 9.12.0.1884302 (R2022a) in the platform of Microsoft Windows 10 Pro Version 10.0 with the system has Intel(R) Core(TM) i5-6500 CPU @ 3.20GHz 3.19 GHz 4.00 GB (3.88 GB usable). The second platform is more advanced than the first. Let us refer to the proposed method under this advanced environment as $FTTIE_{adv}$.

We report the execution time of the proposed method in Table 3. Here, we run the program ten times for each test image, and we consider the average execution time. The overall average, considering all images, is 0.531 sec, and 0.288 sec, respectively. The comparison of the execution time with the SoA method is given in Table 4. We also include the system configuration of the respective methods, which helps us to compare. We note that the proposed method is faster than the SoA methods. For execution time only, we run the proposed model in two different environments. For other analyses, there is no effect of advanced architecture, i.e., we will have the same result irrespective of these two platforms. Hence, hereafter, we use the term FTTIE to refer to the proposed method.

C. RANDOMNESS ANALYSIS

It is known that there are strong correlations among the adjacent pixels of the original image. Due to these correlations, the value of the current pixel can be predicted from the neighboring pixels in the case of plain images. In image encryption, one of the primary goals is that nothing can be guessed about the plain image from the encrypted image. The goal can be fulfilled if the encrypted image is random and noisy. The proposed method returns noisy-like images,

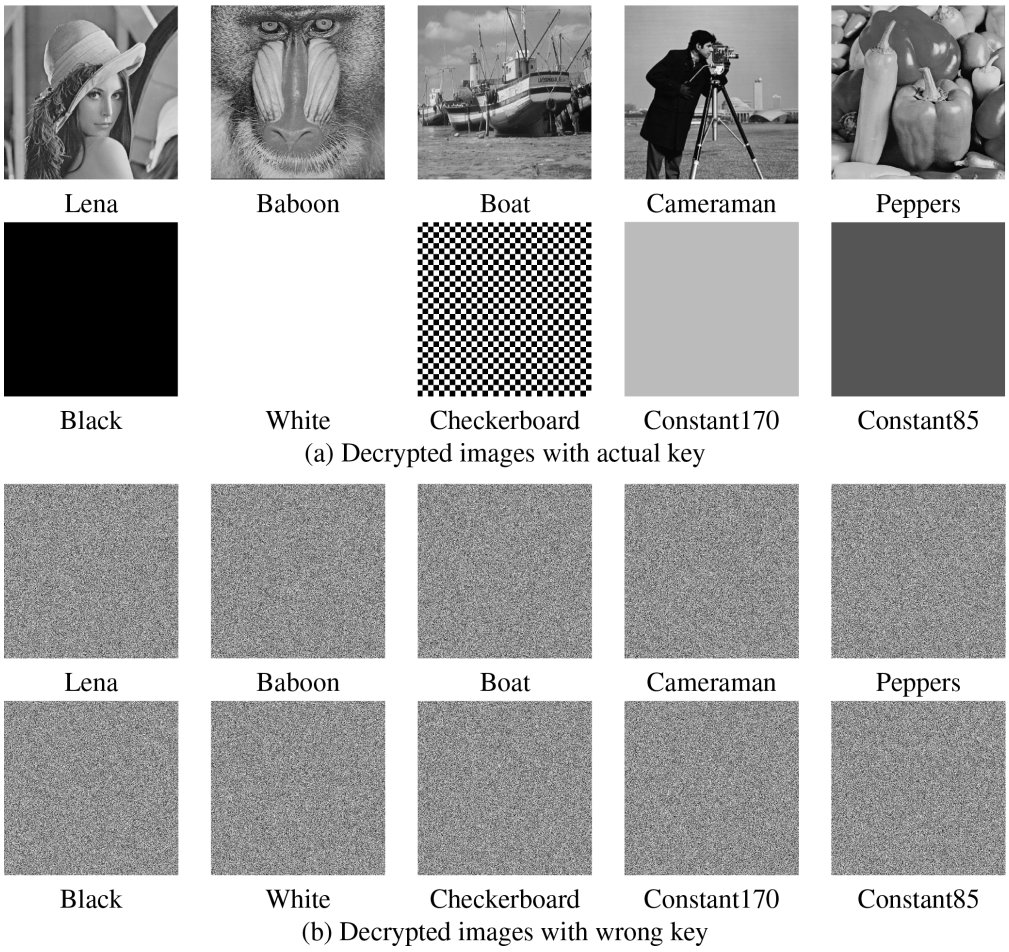

FIGURE 4. Decrypted images: (a) with the actual decryption key, (b) with the wrong decryption key.

as shown in Fig. 3. The point that we need to measure is the randomness of the encrypted image. Here, to study the randomness, we use different analyses like i) histogram analysis, ii) entropy analysis, iii) χ^2 analysis, and iv) correlation analysis.

1) HISTOGRAM ANALYSIS

A histogram attack is a well-known tool for an attacker. Here, an attacker accesses the cipher image and then tries to predict the plain image. The histogram of an image shows the intensity profile of an image. The attacker may be guessed the plain image from known histograms with varying small amounts of the computed histogram. The variation of an almost uniform histogram is negligible; thus, it is impossible to find any clue about the frequency distribution of the image. Therefore, nothing can be guessed about the plain image. Hence, the uniform distribution of the histogram of a cipher image is an essential feature of any image encryption method. Here, the histogram of some of the test images is given in Fig. 5.(a). From these histograms, we note that i) they have specific peaks and ii) they do not cover the entire domain. These characteristics of the histogram of the plain images are due to the spatial correlations among the adjacent pixels. We can say that an image is random if the intensity of a pixel can be any value of the domain, i.e., $P(intensity = i) = P(intensity = j)$, for all $i, j \in \{0, 1, \cdots, 255\}$ (for 8-bit grayscale image). All the intensity values are equally probable (i.e., the frequency of each intensity is almost the same). This phenomenon is reflected in Fig. 5.(b). The figure shows that the histograms are uniformly distributed and cover the entire intensity domain. Since our proposed method returns a nosy image and a uniform histogram, we may assume that encrypted images are random.

2) ENTROPY ANALYSIS

The entropy of a source measures the information contained in the source. A source has more information (less predictable or highly random) if the entropy of the source is large. The entropy of a source is measured by Eq. (17), where $p(s_i)$ is the probability of the i^{th} symbol, s_i, of the source S.

$$H(S) = -\sum_{i=0}^{N-1} p(s_i) log_2\, p(s_i) \tag{17}$$

TABLE 2. Studied SoA methods along with the proposed method.

Schemes	Process of encryption	Techniques introduced/employed
M1 [70]	1) Permutation is generated using a zig-zag scan-based pattern 2) The initial seed of the chaotic map is generated from the plain image 3) Diffusion is done using XOR operation	zig-zag scan based chaotic feedback convolution model
M2 [40]	1) An intermediate cipher image is created using DNA module 2) RNA module is used to produce the final cipher image	DNA rules, DNA-XOR operator, and chaotic map
M3 [41]	1) Hash value of the plain image is used for the initial value of the chaos and to permute the pixels of the image 2) DNA XOR and RNA codons complement operators are used in diffusion.	A new Chaotic Evolutionary Biomolecules Model, DNA, and RNA computing
M4 [39]	1) The plain image is converted to the DNA image using binary coding rules 2) A BST is created from the DNA image using Logistic map 3) DNA-BST is superimposed on DNA Image using XOR operation to get encrypted image	Binary search tree, DNA coding, Logistic map
M5 [63]	1) Given a polynomial, the value of the polynomial is computed at some selected points where the points are selected by interval bisection method 2) This sequence is used to define i) circular shift in confusion phase and ii) substitution matrix and mask image same size as the given image of the diffusion phase.	Interval bisection method, Circular shift, Substitution and XOR method
M6 [6]	1) The permutation is defined using points selected by Regula-Falsi method 2) image encryption is achieved by pixel value substitution and iterative addition with the cyclic shift.	Regula-Falsi method, Substitution, Iterative addition, Circular shift
M7 [75]	1) A unified key is used to select the key pixels 2) DNA encryption is performed on the key pixels in their original locations 3) Other pixels of the image are encrypted by using the key stream generated by the hyperchaotic Lorenz system to perform operations such as scrambling, DNA encoding, cyclic shift, etc.	Key pixels, hyperchaotic Lorenz system Scrambling, DNA encoding, Cyclic shift
Prop.	1) The hash value of image is used as key 2) The pixels are permuted using Fibonacci Transformation 3) Scrambled image is XOR-ed with a key image generated using the key 4) The pixel values are changed using Tribonacci Transformation	Key image, Fibonacci Transformation, and Tribonacci Transformation

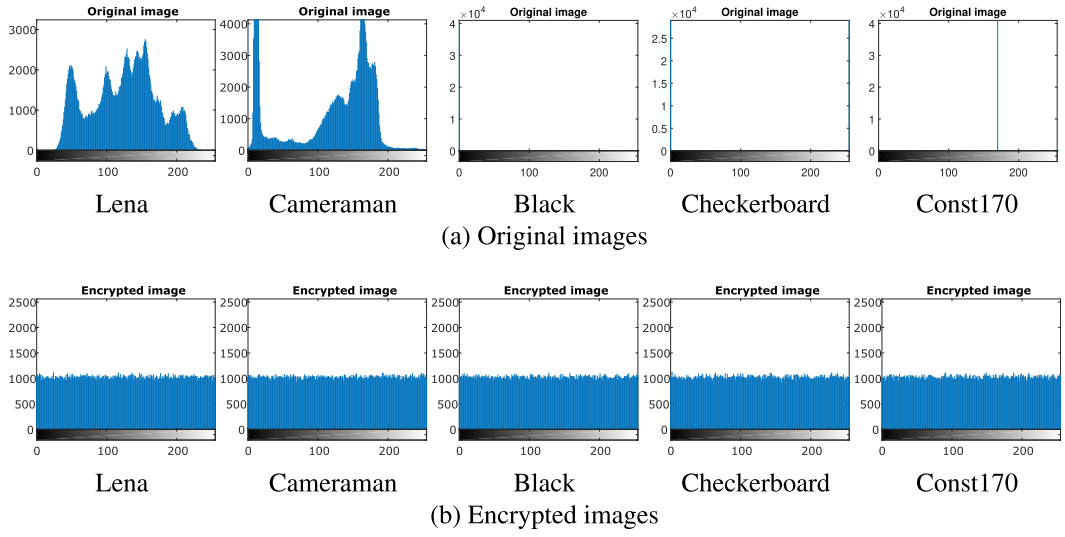

FIGURE 5. Histogram of the images: (a) Original images, (b) Encrypted images.

When all symbols of S are equally likely, the source's entropy is maximum. So, when a source is genuinely random, its entropy is maximized, which is a necessary condition (i.e., maximum entropy does not indicate that the source is random). In the case of 8-bit grayscale images, the maximum entropy is eight, and the computed entropy of the plain and encrypted images is reported in Table 5. The entropy values of the encrypted images are almost 8, and the noisy structure of the cipher images guarantees that the cipher images are random. Table 6 reports the comparative performance of the proposed method with the SoA methods. From this table, we observe that the performance of the proposed method is comparable with the SoA methods.

TABLE 3. Average execution time in sec.

Image	Execution time	
	FTTIE	$FTTIE_{adv}$
Lena	0.536126	0.284235
Baboon	0.568398	0.281999
Boat	0.538379	0.300635
Cameraman	0.517202	0.284769
Peppers	0.548757	0.308050
Black	0.504455	0.269685
White	0.511780	0.286664
Checkerboard	0.519854	0.286427
Constant170	0.540580	0.290878
Constant85	0.527916	0.290954
Average	0.531345	0.288429

TABLE 4. Comparison of execution time (in seconds).

Method (image size)	System specification	Execution Time
M1 [70] (256×256)	Intel Pentium N3540, 2.16 GHz CPU, 8 GB RAM, Windows 10, Matlab R2018a	10.440
M2 [40] (512×512)	Intel Core i7, 2.3 GHz CPU, 8 GB RAM, Windows 8, Python 3	1.494
M4 [39] (512×512)	Intel Core i7, 2.3 GHz CPU, 8 GB RAM, Windows 8, MATLAB 2014	3.009
M6 [6] (512×512)	Intel core i5, 2.3 GHz CPU, MATLAB R2018b	0.620
M7 [75] (512×512)	Intel Core i7-8700, 3.20 GHz CPU 8 GB RAM, Windows 10, Matlab R2020b	0.798
FTTIE (512×512)	Intel Core i5-3230M CPU, 2.60GHz, 4 GB RAM, Windows 7, MATLAB R2015a	0.531
$FTTIE_{adv}$ (512×512)	Intel Core i5-6500 CPU, 3.20GHz 4 GB RAM, Windows 10, MATLAB R2022a	0.288

TABLE 5. Entropy of the original and encrypted images.

Image	Original	Encrypted
Lena	7.445061	7.999287
Baboon	7.358337	7.999289
Boat	7.191370	7.999302
Cameraman	7.047955	7.999367
Peppers	7.593654	7.999270
Black	-0.000000	7.999353
White	-0.000000	7.999410
Checkerboard	1.000000	7.999244
Constant170	-0.000000	7.999199
Constant85	-0.000000	7.999191

From above, we note that the performance of the proposed method in term of the entropy analysis is acceptable. The global (image) entropy is significantly high; it does not imply that it is also high in every region of the cipher image. It may happen that for some blocks, entropy is low. To test the robustness of the proposed method, here, we also compute the entropy at the block level. For this purpose, (k, T_B)-local Shannon entropy is computed as

$$H_{(k,T_B)} = \frac{1}{k}\sum_{i=1}^{k} H(B_i) \tag{18}$$

where k is the number of blocks and B_i's (for $1 \leq i \leq k$) are randomly selected non-overlapping blocks with T_B pixels. The (30, 1936)-local Shannon entropy lies in the range [7.901901305, 7.903037329] [76]. The performance of the proposed method in (30, 1936) setup is given in Table 7. From this table, we observe that the proposed method passed the test for all the images. Therefore, the cipher images are highly random.

3) CORRELATION ANALYSIS

The correlation coefficient (ρ_{xy}), between two random variables X and Y, determines the level of accuracy of a predicted value x' of $x \in X$ using $y \in Y$. The correlation coefficient (ρ_{xy}) between the random variables X and Y is computed as given in Eq. (19).

$$\rho_{xy} = \frac{cov(X,Y)}{\sqrt{D(X)}\sqrt{D(Y)}}$$

where

$$cov(X,Y) = \frac{1}{N}\sum_{i=1}^{N}(x_i - E(X))(y_i - E(Y))$$

$$E(X) = \frac{1}{N}\sum_{i=1}^{N} x_i, \quad E(Y) = \frac{1}{N}\sum_{i=1}^{N} y_i$$

$$D(X) = \frac{1}{N}\sum_{i=1}^{N}(x_i - E(X))^2, \; D(Y) = \frac{1}{N}\sum_{i=1}^{N}(y_i - E(X))^2 \tag{19}$$

In the case of an original image, the level of accuracy is very high when a pixel is predicted from neighboring pixels (known as spatial correlation). Hence, in cryptographic attacks, the spatial correlation of the pixels is exploited. One crucial objective of image encryption is to reduce the spatial correlation of the encrypted image. We consider three adjacency relations: horizontal, vertical, and diagonal. As a result, a collection of adjacent pixels is considered in this analysis phase. In this implementation, we consider 5000 pairs of adjacent pixels where X denotes the intensity of the first pixels of the selected pairs, and the intensity of the second pixels is denoted by Y. The correlation of the adjacent pixels of the plain images and the cipher images is given in Table 8. The table shows that the correlation in plain images is very high, a prominent characteristic of an original image. For the encrypted images, the correlation is close to zero; therefore, an attacker cannot obtain any information about the cipher image from some known pixels. So, nothing can be guessed about the plain image, an essential feature of an image encryption method. Hence, the performance of the proposed method is quite good and acceptable. To establish the efficacy of the proposed method, we compared the correlation coefficients of the test images given by the proposed method with the SoA methods. The comparative result is shown in Table 9. If we compare the performance image-wise, then we have mixed observations. For example, the performance of the FTTIE method for Baboon is better for horizontal correlation.

TABLE 6. The performance of the proposed method and SoA methods in terms of entropy.

Method→ Image ↓	M1 [70]	M2 [40]	M3 [41]	M4 [39]	M5 [63]	M6 [6]	M7 [75]	FTTIE
Lena	7.9994	7.9994	7.9995	7.9992	7.9993	7.9993	7.9994	7.9993
Baboon	7.9991	7.9994	7.9994	7.9990	7.9993	7.9993	7.9992	7.9993
Boat	7.9990	7.9992	7.9994	7.9991				7.9993
Cameraman	7.9992	7.9993	7.9994	7.9991	7.9993	7.9993		7.9994
Peppers	7.9993	7.9993	7.9994	7.9983	7.9994	7.9993	7.9993	7.9993
Black					7.9994	7.9993		7.9994
Checkerboard					7.9993	7.9993		7.9992

TABLE 7. (30, 1936)-local Shannon entropy on the cipher images.

Image	local Shannon entropy	Remark
Lena	7.902567	Pass
Baboon	7.902946	Pass
Boat	7.903012	Pass
Cameraman	7.902496	Pass
Peppers	7.902021	Pass
Black	7.902365	Pass
White	7.901960	Pass
Checkerboard	7.902347	Pass
Constant170	7.903025	Pass
Constant85	7.901948	Pass

TABLE 8. Correlation coefficients of the images.

Image		Horizontal	Vertical	Diagonal
Lena	Org img:	0.969557	0.985108	0.961273
	Enc img:	-0.014825	-0.000664	0.007183
Baboon	Org img:	0.869555	0.759337	0.731440
	Enc img:	0.000300	-0.003300	0.002763
Boat	Org img:	0.931975	0.970481	0.918183
	Enc img:	-0.007857	0.005314	-0.003888
Cameraman	Org img:	0.982116	0.989307	0.971991
	Enc img:	-0.003485	-0.012058	-0.002873
Peppers	Org img:	0.975656	0.980522	0.959344
	Enc img:	-0.015383	-0.009301	0.004001
Black	Org img:	1.000000	1.000000	1.000000
	Enc img:	-0.007286	-0.002917	-0.000073
White	Org img:	1.000000	1.000000	1.000000
	Enc img:	-0.015854	0.009324	-0.005013
Checkerboard	Org img:	0.880613	0.875203	0.776411
	Enc img:	-0.004700	0.014261	0.011992
Constant170	Org img:	1.000000	1.000000	1.000000
	Enc img:	-0.001335	-0.024532	-0.019457
Constant85	Org img:	1.000000	1.000000	1.000000
	Enc img:	-0.020115	-0.004376	-0.003340

Again, for vertical correlation, the performance of the FTTIE is superior to M2, M4, and M7 but inferior to M1, M5, and M6. At the same time, the FTTIE method is superior to M1, M2, and M7 concerning the diagonal correlation, whereas FTTIE is inferior to M4, M5, and M6. A similar type of mix observations is there for different images when we compare the performance of the FTTIE method with SoA methods. So, the performance of the proposed method is similar to that of the SoA methods.

Also, to visualize the relationships among the adjacency pixels of the images, we consider the selected pairs of pixels and plot the intensity value as the scatter diagram, which is a joint distribution. Here, horizontal, vertical, and diagonal neighbors are taken. The scatter diagrams are shown in Fig. 6. This figure has three parts (a) - (c). In each part, the first row shows the scatter diagram of the original images, and the second row represents the scatter diagram of the encrypted images. In the case of the original images, (i) For 'Lena' and 'Cameraman', the points of the scatter diagrams are distributed along the line $y = x$, and this shows that there is a high correlation among the adjacent pixels of these images; (ii) The 'Black' image has only one dot at $(0, 0)$ location, and this implies that pixels are 100% correlated; (iii) All the scatter diagrams of the 'Checkerboard' image has four dots at the locations $\{(0, 0), (0, 255), (255, 0), (255, 255)\}$ and due to the off-diagonal points $(0, 255)$ and $(255, 0)$ the correlation among the pixels is comparatively less; (iv) The scatter diagrams of the 'Constant170' has only one point at location (170, 170), so pixels are 100% correlated. We also have similar observations for the other original images. The result of Table 8 supports our above observations. For the cipher images, we note that the scatter diagrams are spared over the entire domain [0-255, 0-255], which implies that the correlation among the adjacent pixels of the cipher images is almost zero (this is also supported by Table 8). So, the performance of the proposed method is pretty good.

4) χ^2-TEST

The χ^2-test calculates the difference between the observed and expected values of statistical samples. The χ^2-test can be used as another important measure to analyze the uniformity of the cipher image's histogram (i.e., randomness) [75]. Here, we expect that the histogram is uniform and the expected frequency of any gary value $e \in [0, 255]$ is $f_e = \frac{M \times N}{256}$, for an 8-bit grayscale image of size $M \times N$. The χ^2 value of a cipher image can be defined by Eq (20):

$$\chi^2 = \sum_{i=0}^{255} \frac{(f_i - f_e)^2}{f_e} \tag{20}$$

where f_i is the frequency of the gray value i in the cipher image. When the χ^2 value is high, the distribution is less analogous to the uniform distribution. If two distributions are identical, the χ^2 value is 0, indicating that the theoretical value is consistent. When the significant level is 0.05, the ideal value is $\chi^2_{255,0.5} = 293.2478$, where the degree of freedom is 255 for an 8-bit image [75]. The χ^2 value of the cipher images is given in Table 10. Table 10 supports our conclusions: i) original images do not have a uniform

TABLE 9. The performance of the proposed method and SoA methods in terms of correlation coefficients.

Method→ Image ↓	Direction	M1 [70]	M2 [40]	M4 [39]	M5 [63]	M6 [6]	M7 [75]	FTTIE
Lena	Horizontal	–0.002062	0.0054	0.0007	0.002097	0.001853	-0.0026	-0.014825
	Vertical	0.003685	0.0192	0.0017	0.002767	0.001984	-0.0033	-0.000664
	Diagonal	0.000249	0.0055	0.0008	-0.002901	0.000743	0.0004	0.007183
Baboon	Horizontal	–0.005890	0.0059	0.0009	-0.002659	0.000433	0.0019	0.000300
	Vertical	–0.001884	0.0047	0.0026	-0.001401	0.001475	-0.0041	-0.003300
	Diagonal	–0.009670	0.0058	0.0006	-0.002419	0.000648	-0.0099	0.002763
Boat	Horizontal	–0.002261	0.0052	0.0007				-0.007857
	Vertical	–0.000162	0.0007	0.0031				0.005314
	Diagonal	0.000207	0.0151	0.0007				-0.003888
Cameraman	Horizontal	–0.000280	0.0068	0.0035	0.004615			-0.003485
	Vertical	–0.000445	0.0023	0.0041	-0.000353			-0.012058
	Diagonal	0.000259	0.0092	0.0014	0.003272			-0.002873
Peppers	Horizontal	–0.000434	0.0149	0.0029	-0.001076	0.001180	-0.0063	-0.015383
	Vertical	–0.000446	0.0077	0.0059	0.001394	0.000721	-0.0006	-0.009301
	Diagonal	0.000195	0.0002	0.0018	-0.002952	0.001784	-0.0046	0.004001
Black	Horizontal				0.001646			-0.007286
	Vertical				-0.001508			-0.002917
	Diagonal				0.005440			-0.000073
Checkerboard	Horizontal				-0.002520			-0.004700
	Vertical				-0.001171			0.014261
	Diagonal				0.004867			0.011992

TABLE 10. χ^2 value of the original and cipher images.

Image	Original	Cipher
Lena	158349.355469	259.451172
Baboon	187356.572266	259.167969
Boat	383969.687500	253.525391
Cameraman	418530.146484	229.529297
Peppers	120165.796875	265.492188
Black	66846720.000000	234.914063
White	66846720.000000	214.857422
Checkerboard	33292288.000000	275.041016
Constant170	66846720.000000	290.718750
Constant85	66846720.000000	289.310547

distribution (see Fig. 5(a)), and the χ^2 values are very high; ii) for cipher images, the value is less than 293.2478, indicating an almost uniform histogram (see Fig. 5(b)). Therefore, the proposed image encryption method can successfully counter statistical attacks.

5) PIXEL DISPARITY ANALYSIS

A straightforward approach to obtaining the relationship between the plain and cipher images is to find the pixel-wise difference between images. Commonly used parameters are mean square error (MSE), peak signal-to-noise ratio (PSNR), and mean absolute error (MAE), which are defined in Eq. 21).

$$MSE = \frac{1}{M \times N} \sum_{i=1}^{M} \sum_{j=1}^{N} (P(i,j) - C(i,j))^2$$

$$PSNR = 10 \times log_{10} \frac{255^2}{MSE} \; dB$$

TABLE 11. Pixel disparity between the plain image and cipher image.

Image	MAE	MSE	PSNR
Lena	73.017334	7775.909210	9.223292
Baboon	70.936752	7236.867638	9.535297
Boat	72.429760	7628.511555	9.306406
Cameraman	79.425556	9396.519512	8.401133
Peppers	75.414028	8396.533710	8.889803
Black	127.426994	21708.028267	4.764600
White	127.302147	21667.287865	4.772758
Checkerboard	127.459908	21698.784279	4.766450
Constant170	70.995754	7256.970516	9.523250
Constant85	71.022579	7265.353962	9.518236

$$MAE = \frac{1}{M \times N} \sum_{i=1}^{M} \sum_{j=1}^{N} |P(i,j) - C(i,j)| \qquad (21)$$

where P and Q are the corresponding plain and cipher images, respectively, MSE and PSNR have an inverse relation, i.e., when two images are similar, MSE will be less, and in that case, the PSNR value will be high. In the case of image encryption, we expect that MSE will be as large as possible (i.e., PSNR is less). On the other hand, MSE and MAE have similar behavior, i.e., when MSE is large, then MAE also has a high value and vice versa. Table 11 reflects our observations, and we note that the PSNR value is very low (or MSE/MAE is high). So, predicting the plain image from the cipher image is impossible.

D. KEY SENSITIVITY ANALYSIS OF ENCRYPTION PROCESS

Key sensitivity analysis means we must test whether the proposed method is key sensitive. An image encryption method is said to be key sensitive if the method produces two different cipher images from the same plain image when two different

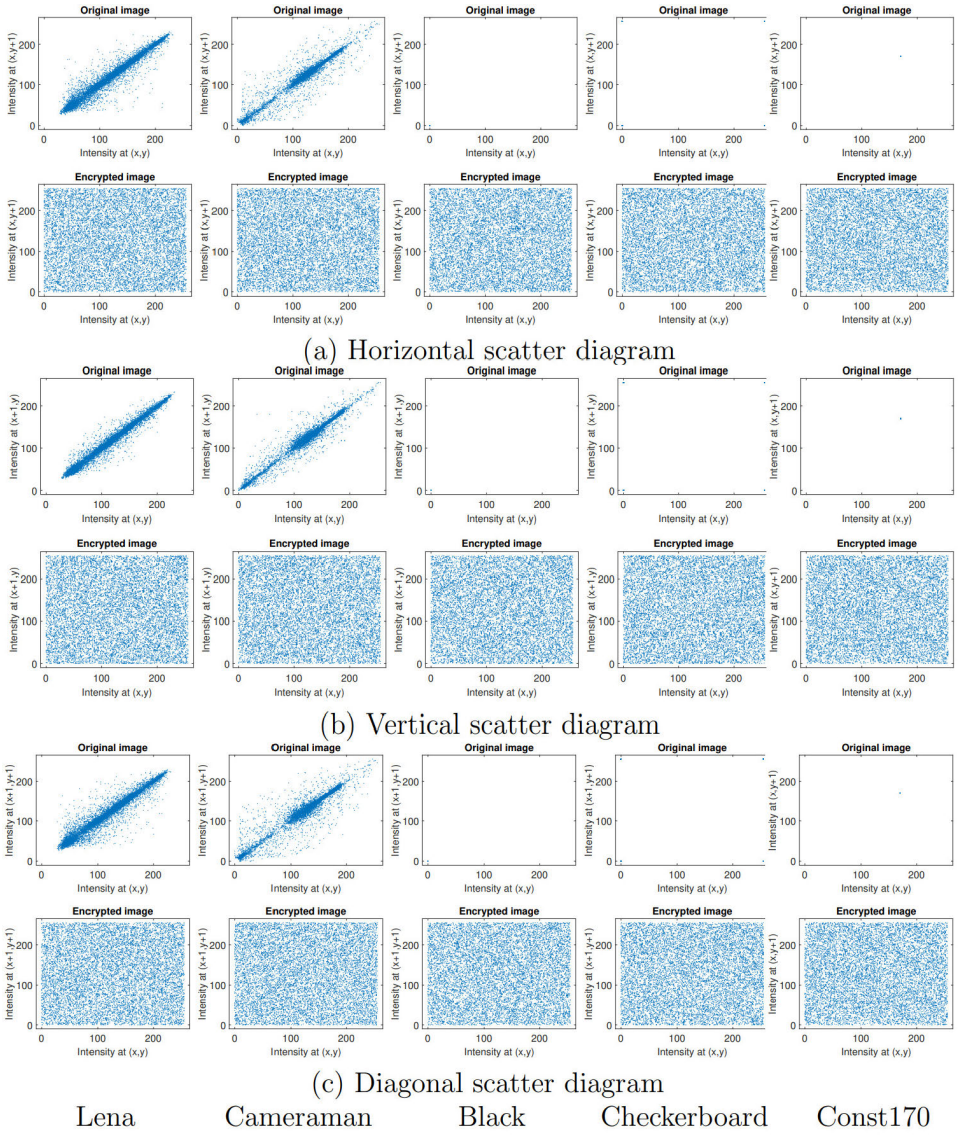

FIGURE 6. Scatter diagrams of some original and corresponding encrypted images.

encryption keys (maybe a single bit difference) are used. In this testing part, we have derived the second key (for the encryption) from the first by complementing a particular bit. Two parameters, NPCR (Number of Pixel Change Rate) and UACI (Unified Average Changing Intensity) are used to measure the deviation between two cipher images. NPCR and UACI can be formulated in Eq. (22).

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

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

where $D(i,j)$ defined as

$$D(i,j) = \begin{cases} 1, \text{ if } C_1(i,j) \neq C_2(i,j) \\ 0, \text{ otherwise} \end{cases} \quad (22)$$

where C_1 and C_2 are two cipher images of the same size $M \times N$ obtained from the same plain image using two different keys. The ideal NCPR and UACI are 100% and 33.33%, respectively [23]. In this experiment, the key size is 256, and we consider five cases, as given in Table 12, to analyze the key sensitivity of the proposed method. The NPCR and UACI under the above test cases and their average values for the test images are given in Tables 13 and 14.

From these two tables, we note that the computed values are very close to the ideal value, and we may conclude that the proposed image encryption method is highly key-sensitive. The comparison with SoA methods is given in Table 15, where the NPCR value is reported. From this table,

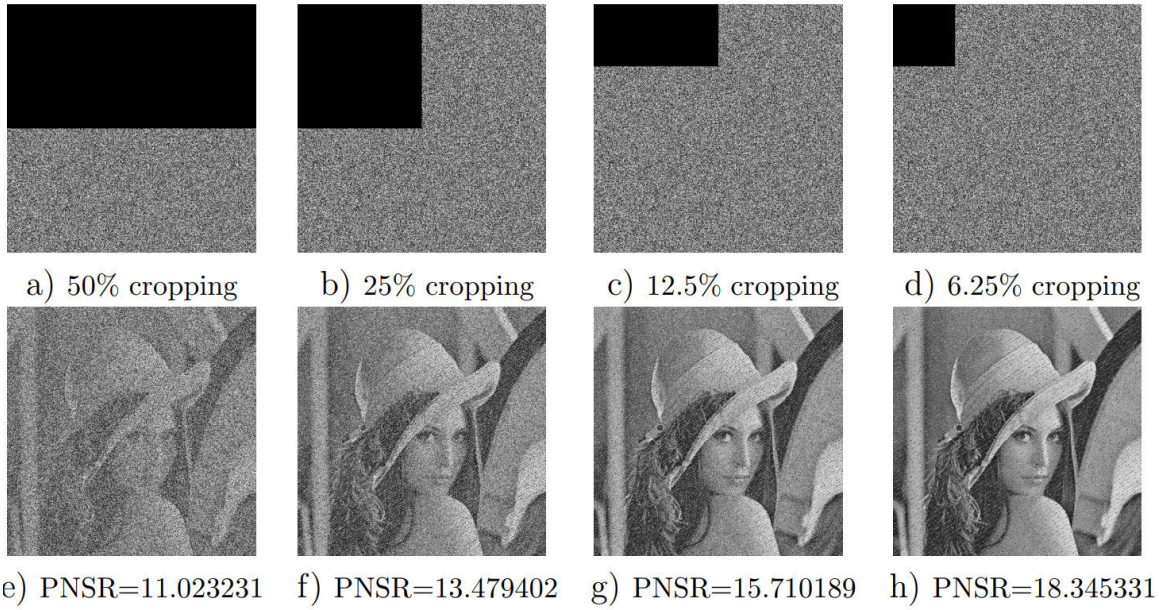

FIGURE 7. Deciphered 'Lena cipher images' with different degrees of cropping.

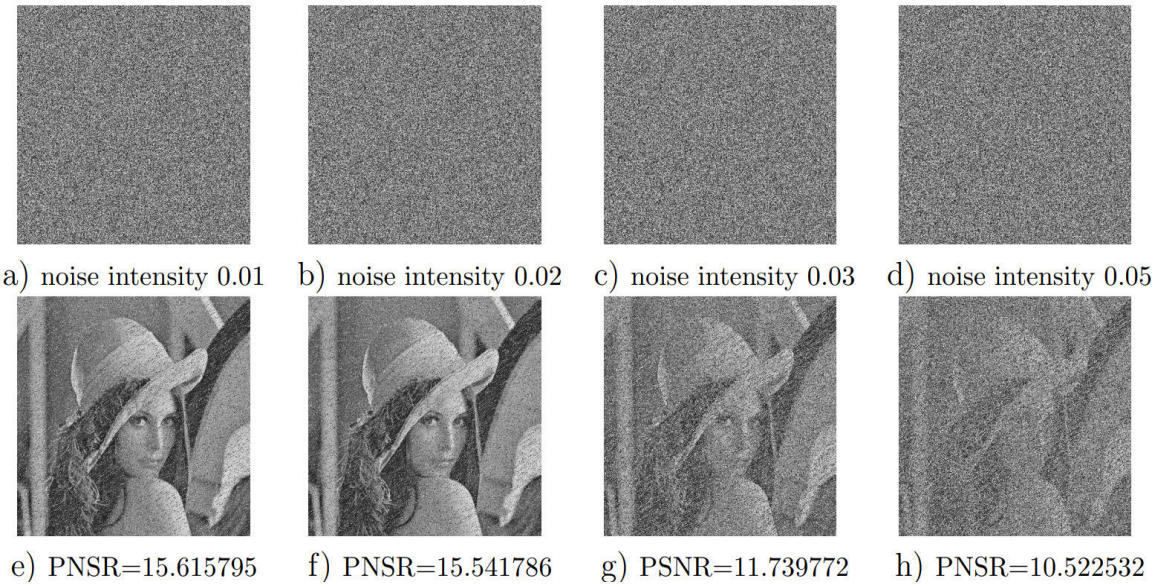

FIGURE 8. Cipher images with different degrees of Salt & Pepper noise (a - d) and corresponding decipher images (e - f).

TABLE 12. Test cases of key sensitivity analysis.

Test cases →	Case 1	Case 2	Case 3	Case 4	Case 5
Bit complemented	4^{th}	56^{th}	132^{nd}	208^{th}	256^{th}

we observe that the performance of the proposed method is similar to the first two existing methods [40], [41]. However, it is slightly less than the third method [39].

E. KEY SENSITIVITY OF THE DECRYPTION PROCESS

In this section, we study the key sensitivity of the proposed decryption process. It is already mentioned that the proposed decryption method can only produce the original image if the correct key is given (see Fig. 4(b)). To test the key sensitivity of the decryption process, we consider two cases:

1) The cipher image is decrypted by keys key_a and key'_a, where key_a is the actual key and key'_a is derived from key_a by flipping only one bit. So, the decrypted image with key_a is the original image, and the deciphered image using key'_a is a noise-like image, as shown in Fig. 4(b). Then the NPCR and UACI values between the original and noisy decipher images are computed.
2) In this scenario, the same cipher image is decrypted using key'_a and key''_a, respectively, where key''_a is obtained from key'_a by complementing a bit other than

TABLE 13. NPCR measure (%) for key sensitivity analysis.

Test condition →	Case 1	Case 2	Case 3	Case 4	Case 5	Average
Image ↓	NPCR	NPCR	NPCR	NPCR	NPCR	NPCR
Lena	99.582291	99.598694	99.608612	99.200058	99.582291	99.514389
Baboon	99.606323	99.614716	99.605942	99.605942	99.594879	99.605560
Boat	99.611664	99.606323	99.584198	99.595642	99.615479	99.602661
Cameraman	99.596024	99.605560	99.232101	99.594498	99.628067	99.531250
Peppers	99.603271	99.604797	99.598694	99.597931	99.619293	99.604797
Black	99.621582	99.621582	99.612808	99.621582	99.593353	99.614181
White	99.626923	99.602509	99.618149	99.623871	99.613190	99.616928
Checkerboard	99.615479	99.579620	99.610901	99.615479	99.214935	99.527283
Constant170	99.548340	99.615097	99.580383	99.600601	99.644470	99.597778
Constant85	99.603653	99.595642	99.622345	99.622345	99.612808	99.611359

TABLE 14. UACI measure (%) for key sensitivity analysis.

Test condition →	Case 1	Case 2	Case 3	Case 4	Case 5	Average
Image ↓	UACI	UACI	UACI	UACI	UACI	UACI
Lena	33.348443	33.442854	33.459736	33.409161	33.348443	33.401727
Baboon	33.309637	33.410598	33.424123	33.358507	33.420190	33.384611
Boat	33.414022	33.483049	33.402626	33.425547	33.522388	33.449526
Cameraman	33.539064	33.485827	33.489577	33.494475	33.530526	33.507894
Peppers	33.408230	33.468824	33.460194	33.413883	33.503502	33.450927
Black	33.450742	33.453988	33.443636	33.453988	33.546138	33.469698
White	33.515939	33.451167	33.454111	33.518425	33.511207	33.490170
Checkerboard	33.489221	33.394068	33.493416	33.489221	33.396577	33.452501
Constant170	33.550738	33.464792	33.509507	33.360810	33.497044	33.476578
Constant85	33.421457	33.477122	33.406869	33.406869	33.402857	33.423035

TABLE 15. The key sensitivity of the proposed method and SoA methods in terms of NPCR in %.

Method→ Image ↓	M2 [40]	M3 [41]	M4 [39]	FTTIE
Lena	99.60	99.5998	99.93	99.5144
Baboon	99.61	99.6101	99.93	99.6056
Boat	99.60	99.5977	99.95	99.6027
Cameraman	99.60	99.6071	99.88	99.5313
Peppers	99.62	99.5983	99.92	99.6048

the bit which was complemented during the computation of key'_a. So, key'_a and key''_a differ by a single bit, and none of these is the actual key. Both the decipher images, obtained in this case, are noisy, and then we compute the NPCR and UACI values between these two decipher images.

The test results under the above two cases are reported in Table 16. Since in computation of NPCR consider whether the pixels are the same or not so, in both cases, the NPCR value is acceptable. In UACI computation, the actual difference between two pixels is taken. In the first case, one deciphered image is the original one, i.e., not a random image, and the other image is a random one. So, in the first case, we did not achieve the expected value of UACI. In the second case, both the decipher images are noisy (i.e., random), so we achieve the expected result. From these test cases and the result given in Table 16, we may conclude that the proposed decryption process is also highly key-sensitive.

F. DIFFERENTIAL ANALYSIS

Usually, an attacker would slightly modify a plain image and encrypt the images (plain and modified) with a given

TABLE 16. Key sensitivity measure of the decryption process in %.

Test condition →	$key_a \rightarrow key'_a$		$key'_a \rightarrow key''_a (\neq key_a)$	
Image ↓	NPCR	UACI	NPCR	UACI
Lena	99.6033	28.6088	99.5903	33.4537
Baboon	99.5424	27.8631	99.6250	33.5700
Boat	99.5636	28.4531	99.5861	33.4177
Cameraman	99.6132	31.1033	99.6216	33.5664
Peppers	99.5773	29.5813	99.5937	33.5376
Black	99.6033	49.9736	99.5987	33.4442
White	99.6296	50.1252	99.6162	33.4315
Checkerboard	99.5987	49.9795	99.5983	33.3967
Constant170	99.5926	27.8675	99.6067	33.4897
Constant85	99.5964	27.8515	99.5995	33.4336

TABLE 17. Test cases of differential attacks.

Test cases →	Case 1	Case 2	Case 3	Case 4	Case 5	Case 6
Pixel modified	(1,1)	(1,512)	(512,1)	(512,512)	(256,256)	(150,210)

algorithm to obtain two cipher images. Then, an attacker tries to find the relationship between the plain image and the cipher image by comparing two encrypted images. This process is known as a 'difference attack'. A secure image encryption method should resist differential attack, i.e., due to a tiny change in the plain image, the encryption method should produce a different cipher image using the same cryptosystem. Moreover, if it happens, finding any clue about the plain image from the cipher image will be challenging. NPCR and UACI parameters are used to measure the difference between two cipher images. So, we use Eq. (22), where C_1 and C_2 are the cipher images obtained from the plain and modified

TABLE 18. NPCR value (%) due to modified plain image.

Position→	(1,1)	(1,512)	(512,1)	(512,512)	(256,256)	(150,210)	Average
Image ↓	NPCR	NPCR	NPCR	NPCR	NPCR	NPCR	NPCR
Lena	99.624252	99.600983	99.612427	99.592209	99.624252	99.626923	99.613508
Baboon	99.600220	99.589157	99.616241	99.625397	99.601364	99.596405	99.604797
Boat	99.589539	99.613190	99.600601	99.605942	99.626541	99.612808	99.608103
Cameraman	99.619293	99.593735	99.608994	99.607086	99.597168	99.601364	99.604607
Peppers	99.631882	99.616241	99.633026	99.592590	99.624252	99.610901	99.618149
Black	99.618530	99.587631	99.607468	99.617004	99.596024	99.612045	99.606450
White	99.635315	99.591827	99.607849	99.616623	99.611282	99.582672	99.607595
Checkerboard	99.607086	99.595261	99.607468	99.605942	99.614334	99.607468	99.606260
Constant170	99.607086	99.617004	99.629974	99.632645	99.605179	99.597168	99.614843
Constant85	99.603653	99.610519	99.610519	99.609756	99.606705	99.606705	99.607976

TABLE 19. UACI value (%) due to modified plain image.

Position→	(1,1)	(1,512)	(512,1)	(512,512)	(256,256)	(150,210)	Average
Image ↓	UACI	UACI	UACI	UACI	UACI	UACI	UACI
Lena	33.413755	33.454439	33.379728	33.454401	33.505837	33.458171	33.444389
Baboon	33.451004	33.440667	33.520867	33.439270	33.499757	33.462248	33.468969
Boat	33.520524	33.427105	33.468743	33.427565	33.468623	33.474045	33.464434
Cameraman	33.515205	33.484876	33.475239	33.480551	33.521425	33.505224	33.497087
Peppers	33.396327	33.423345	33.432240	33.446555	33.414833	33.464578	33.429646
Black	33.475123	33.467127	33.490922	33.446733	33.477324	33.429009	33.464373
White	33.514914	33.518723	33.495575	33.459799	33.434952	33.522321	33.491047
Checkerboard	33.462349	33.494892	33.477976	33.448968	33.473994	33.413604	33.461964
Constant170	33.465041	33.510117	33.463981	33.460575	33.432919	33.407864	33.456749
Cconstant85	33.419471	33.470540	33.420730	33.388661	33.458832	33.454155	33.435398

TABLE 20. The performance of the proposed method and SoA methods against differential attack in %.

Method→		M1	M2	M3	M4	M5	M6	M7	FTTIE
Image ↓		[70]	[40]	[41]	[39]	[63]	[6]	[75]	
Lena	NPCR	99.6379	99.61	99.5998	99.6289	99.6231	99.7979	99.6167	99.6135
	UACI	33.4617	33.27	33.4715	33.5420	33.4556	33.4338	33.4589	33.4444
Baboon	NPCR	99.6264	99.62	99.6101	99.3665	99.6162	99.8088	99.6063	99.6048
	UACI	33.4948	33.19	33.4238	33.5084	33.5117	33.3604	33.4565	33.4690
Boat	NPCR	99.6322	99.57	99.5977	99.6742				99.6081
	UACI	33.4423	33.14	33.4447	33.6392				33.4644
Cameraman	NPCR	99.6462	99.53	99.6071	99.5283	99.6250	99.8052		99.6046
	UACI	33.5464	33.08	33.4647	33.4292	33.4656	33.4173		33.4971
Peppers	NPCR	99.6338	99.62	99.5983	99.6047	99.5918	99.7873	99.6112	99.6181
	UACI	33.4852	33.20	33.4823	33.3447	33.3994	33.4353	33.4776	33.4296
Black	NPCR					99.6761	99.8051		99.6065
	UACI					33.9281	33.4059		33.4644
Checkerboard	NPCR					99.6112	99.7983		99.6063
	UACI					33.5467	33.4340		33.4620

images with the same key. In the experiment, the LSB of a particular pixel is complemented to obtain the modified image. Since the cipher image depends on the plain image, different cipher images will be obtained when we modify different pixels. To achieve a more acceptable result in this analysis, we modify different pixels, and corresponding test cases are given in Table 17. The NPCR and UACI values of the underline test cases of differential attacks are reported in Tables 18 and 19. These tables show that the test values are close to the ideal values. Table 18 shows that for each test image, the NPCR value is sometimes less than the ideal value of 99.6094% and sometimes greater than the ideal value. To achieve a more robust result, we report the average value. The average value of NPCR for each test image is almost equal to the ideal value. So, we can conclude that the proposed method can resist the differential attack. The applicability of the proposed method against the differential attack is studied in Table 20. This study shows that the proposed method performs similarly to the SoA methods.

G. CHOSEN-PLAINTEXT AND KNOWN-PLAINTEXT ATTACKS

In a chosen-plaintext attack, the attacker can temporarily access the encryption module and use typical plaintext images like 'Black', 'White', or other images to identify the information about the encryption key. The resistance against the chosen-plaintext attack ensures robustness against the known-plaintext attack. In a chosen-plaintext attack, the third party uses the information of some chosen plaintext-ciphertext pairs. However, the attacker can use any plain image other than the actual plain image of the current cipher image, which he wants to break. As the key of the encryption process depends on the plain image and the proposed method is sensitive to the plain image (i.e., robust against

the differential attack), it is not possible to find any relation between the underline cipher image and the cipher image obtained from the used plain image. Hence, the proposed method is robust against the known-plaintext and chosen-plaintext attacks.

H. CROPPING ATTACK

A good image encryption method should have the capability to restore the features of the plain image after a certain degree of the cipher image is cropped out. Here, we show the result on image 'Lena' in Fig. 7. The cipher image of 'Lena' is cropped with different levels like 50%, 25%, 12.5%, and 6.25%, and then the decipher images are obtained. Figs. 7(a)-(d) show the cropped cipher images, while Figs. 7(e)-(h) show the decipher images. The PSNR between the plain and deciphered images is also calculated. From the figures, it is easy to understand that the deciphered images are 'Lena', and the quality of the deciphered image is better when the level of cropping is lower. So, the proposed method can restore the features of a plain image to a certain extent, and therefore, the present method can resist cropping attacks.

I. NOISE ATTACK

The anti-interference ability of a cryptosystem is the ability of the system to defend against a noise attack. During the transmission of the cipher image, the image may be distorted by the transmission noise, affecting the deciphered image. To test the anti-noise ability of the proposed method, we distort the cipher image with salt-and-pepper noise and then decrypt the image. Here, we test the 'Lena' image. The noisy cipher and corresponding decipher images are shown in Fig. 8. The deciphered images clearly show that those are 'Lena' images; hence, we may conclude that the proposed the method can defend against noise attacks.

VII. CONCLUSION

This article proposed a new plaintext-based image encryption technique using the Fibonacci and Tribonacci transformations. This method obtains the encryption key from the plain image using SHA256. A 2×2 Fibonacci matrix is defined to scramble the image, and a 3×3 Tribonacci matrix is used in the diffusion phase to modify the value of the pixels. To our knowledge, this is the first time a Tribonacci Transformation has been used in image encryption. The proposed method has a high key space, and the proposed method has similar performance to SoA methods. The performance of the proposed method is equally suitable for different kinds of images (the test set includes some special images also). This paper highlights a new direction to diffuse the pixels using the Tribonacci Transformation. Since the Tribonacci matrix is 3×3, the proposed method can be extended for color image encryption.

REFERENCES

[1] C.-S. Lin, C.-C. Chen, and Y.-C. Chen, "XOR-based progressively secret image sharing," *Mathematics*, vol. 9, no. 6, p. 612, Mar. 2021.
[2] A. Paul, S. Kandar, and B. C. Dhara, "Boolean operation based lossless threshold secret image sharing," *Multimedia Tools Appl.*, vol. 81, pp. 35293–35316, Mar. 2022.
[3] L. Li, Y. Lu, L. Liu, Y. Sun, and J. Wang, "Practical secret image sharing based on the Chinese remainder theorem," *Mathematics*, vol. 10, no. 12, p. 1959, Jun. 2022.
[4] S. K. Das and B. C. Dhara, "An LSB based novel data hiding method using extended LBP," *Multimedia Tools Appl.*, vol. 77, no. 12, pp. 15321–15351, Jun. 2018.
[5] N. Subramanian, O. Elharrouss, S. Al-Maadeed, and A. Bouridane, "Image steganography: A review of the recent advances," *IEEE Access*, vol. 9, pp. 23409–23423, 2021.
[6] A. Paul, S. Kandar, and B. C. Dhara, "Image encryption using permutation generated by modified Regula–Falsi method," *Appl. Intell.*, vol. 52, pp. 10979–10998, Jan. 2022.
[7] S. Kandar, D. Chaudhuri, A. Bhattacharjee, and B. C. Dhara, "Image encryption using sequence generated by cyclic group," *J. Inf. Secur. Appl.*, vol. 44, pp. 117–129, Feb. 2019.
[8] G. Alvarez and S. Li, "Some basic cryptographic requirements for chaos-based cryptosystems," *Int. J. Bifurcation Chaos*, vol. 16, no. 8, pp. 2129–2151, Aug. 2006.
[9] C. Zhu, C. Liao, and X. Deng, "Breaking and improving an image encryption scheme based on total shuffling scheme," *Nonlinear Dyn.*, vol. 71, nos. 1–2, pp. 25–34, Jan. 2013.
[10] J.-X. Chen, Z.-L. Zhu, C. Fu, L.-B. Zhang, and Y. Zhang, "An image encryption scheme using nonlinear inter-pixel computing and swapping based permutation approach," *Commun. Nonlinear Sci. Numer. Simul.*, vol. 23, nos. 1–3, pp. 294–310, Jun. 2015.
[11] J. C. Dagadu, J.-P. Li, and E. O. Aboagye, "Medical image encryption based on hybrid chaotic DNA diffusion," *Wireless Pers. Commun.*, vol. 108, no. 1, pp. 591–612, Sep. 2019.
[12] M. Sokouti, A. Zakerolhosseini, and B. Sokouti, "Medical image encryption: An application for improved padding based GGH encryption algorithm," *Open Med. Informat. J.*, vol. 10, no. 1, pp. 11–22, Oct. 2016.
[13] A. S. Dongare, A. Alvi, and N. Tarbani, "An efficient technique for image encryption and decryption for secured multimedia application," *Int. Res. J. Eng. Technol.*, vol. 4, no. 4, pp. 3186–3190, 2017.
[14] Y. Li, H. Yu, B. Song, and J. Chen, "Image encryption based on a single-round dictionary and chaotic sequences in cloud computing," *Concurrency Comput., Pract. Exp.*, vol. 33, no. 7, pp. 1, Apr. 2021.
[15] R. K. Dwivedi, R. Kumar, and R. Buyya, "Secure healthcare monitoring sensor cloud with attribute-based elliptical curve cryptography," *Int. J. Cloud Appl. Comput.*, vol. 11, no. 3, pp. 1–18, Jul. 2021.
[16] B. Joshi, B. Joshi, A. Mishra, V. Arya, A. K. Gupta, and D. Peraković, "A comparative study of privacy-preserving homomorphic encryption techniques in cloud computing," *Int. J. Cloud Appl. Comput.*, vol. 12, no. 1, pp. 1–11, Sep. 2022.
[17] X. Li, J. Knipe, and H. Cheng, "Image compression and encryption using tree structures," *Pattern Recognit. Lett.*, vol. 18, nos. 11–13, pp. 1253–1259, Nov. 1997.
[18] H. K.-C. Chang and J.-L. Liu, "A linear quadtree compression scheme for image encryption," *Signal Process., Image Commun.*, vol. 10, no. 4, pp. 279–290, Sep. 1997.
[19] J. Wang, X. Song, and A. A. A. El-Latif, "Efficient entropic security with joint compression and encryption approach based on compressed sensing with multiple chaotic systems," *Entropy*, vol. 24, no. 7, p. 885, Jun. 2022.
[20] J. Fridrich, "Symmetric ciphers based on two-dimensional chaotic maps," *Int. J. Bifurcation Chaos*, vol. 8, no. 6, pp. 1259–1284, Jun. 1998.
[21] J.-C. Yen and J.-I. Guo, "Efficient hierarchical chaotic image encryption algorithm and its VLSI realisation," *IEE Proc., Vis., Image Signal Process.*, vol. 147, no. 2, pp. 167–175, 2000.
[22] C. Li, G. Luo, K. Qin, and C. Li, "An image encryption scheme based on chaotic tent map," *Nonlinear Dyn.*, vol. 87, no. 1, pp. 127–133, Jan. 2017.
[23] I. Hussain and M. A. Gondal, "An extended image encryption using chaotic coupled map and S-box transformation," *Nonlinear Dyn.*, vol. 76, no. 2, pp. 1355–1363, Apr. 2014.
[24] B. Wang, B. F. Zhang, and X. W. Liu, "An image encryption approach on the basis of a time delay chaotic system," *Optik*, vol. 225, Jan. 2021, Art. no. 165737.

[25] Z. Hua, F. Jin, B. Xu, and H. Huang, "2D logistic-sine-coupling map for image encryption," *Signal Process.*, vol. 149, pp. 148–161, Aug. 2018.

[26] P. Biswas, S. Kandar, and B. C. Dhara, "A novel image encryption technique using one dimensional chaotic map and circular shift technique," in *Proc. 6th Int. Conf. Softw. Comput. Appl.*, Feb. 2017, pp. 112–116.

[27] X.-Y. Wang, L. Yang, R. Liu, and A. Kadir, "A chaotic image encryption algorithm based on perceptron model," *Nonlinear Dyn.*, vol. 62, no. 3, pp. 615–621, Nov. 2010.

[28] M. A. B. Farah, A. Farah, and T. Farah, "An image encryption scheme based on a new hybrid chaotic map and optimized substitution box," *Nonlinear Dyn.*, vol. 99, no. 4, pp. 3041–3064, Mar. 2020.

[29] S. J. Sheela, K. V. Suresh, and D. Tandur, "Image encryption based on modified Henon map using hybrid chaotic shift transform," *Multimedia Tools Appl.*, vol. 77, no. 19, pp. 25223–25251, Oct. 2018.

[30] A. P. Kari, A. H. Navin, A. M. Bidgoli, and M. Mirnia, "A new image encryption scheme based on hybrid chaotic maps," *Multimedia Tools Appl.*, vol. 80, no. 2, pp. 2753–2772, Jan. 2021.

[31] N. Khalil, A. Sarhan, and M. A. M. Alshewimy, "An efficient color/grayscale image encryption scheme based on hybrid chaotic maps," *Opt. Laser Technol.*, vol. 143, Nov. 2021, Art. no. 107326.

[32] Z. Li, C. Peng, L. Li, and X. Zhu, "A novel plaintext-related image encryption scheme using hyper-chaotic system," *Nonlinear Dyn.*, vol. 94, no. 2, pp. 1319–1333, Oct. 2018.

[33] E. Hasanzadeh and M. Yaghoobi, "A novel color image encryption algorithm based on substitution box and hyper-chaotic system with fractal keys," *Multimedia Tools Appl.*, vol. 79, nos. 11–12, pp. 7279–7297, Mar. 2020.

[34] M. Kaur, D. Singh, K. Sun, and U. Rawat, "Color image encryption using non-dominated sorting genetic algorithm with local chaotic search based 5D chaotic map," *Future Gener. Comput. Syst.*, vol. 107, pp. 333–350, Jun. 2020.

[35] X. Wang, Q. Ren, and D. Jiang, "An adjustable visual image cryptosystem based on 6D hyperchaotic system and compressive sensing," *Nonlinear Dyn.*, vol. 104, no. 4, pp. 4543–4567, Jun. 2021.

[36] K. M. Hosny, S. T. Kamal, M. M. Darwish, and G. A. Papakostas, "New image encryption algorithm using hyperchaotic system and Fibonacci *Q*-matrix," *Electronics*, vol. 10, no. 9, p. 1066, Apr. 2021.

[37] X. Chai, Y. Chen, and L. Broyde, "A novel chaos-based image encryption algorithm using DNA sequence operations," *Opt. Lasers Eng.*, vol. 88, pp. 197–213, Jan. 2017.

[38] J. Chen, L. Chen, and Y. Zhou, "Cryptanalysis of a DNA-based image encryption scheme," *Inf. Sci.*, vol. 520, pp. 130–141, May 2020.

[39] H. Nematzadeh, R. Enayatifar, M. Yadollahi, M. Lee, and G. Jeong, "Binary search tree image encryption with DNA," *Optik*, vol. 202, Feb. 2020, Art. no. 163505.

[40] M. Yadollahi, R. Enayatifar, H. Nematzadeh, M. Lee, and J.-Y. Choi, "A novel image security technique based on nucleic acid concepts," *J. Inf. Secur. Appl.*, vol. 53, Aug. 2020, Art. no. 102505.

[41] A. A. Abbasi, M. Mazinani, and R. Hosseini, "Evolutionary-based image encryption using biomolecules and non-coupled map lattice," *Opt. Laser Technol.*, vol. 140, Aug. 2021, Art. no. 106974.

[42] X. Xue, D. Zhou, and C. Zhou, "New insights into the existing image encryption algorithms based on DNA coding," *PLoS ONE*, vol. 15, no. 10, Oct. 2020, Art. no. e0241184.

[43] S. Zhu and C. Zhu, "Secure image encryption algorithm based on hyper-chaos and dynamic DNA coding," *Entropy*, vol. 22, no. 7, p. 772, Jul. 2020.

[44] K. C. Jithin and S. Sankar, "Colour image encryption algorithm combining Arnold map, DNA sequence operation, and a Mandelbrot set," *J. Inf. Secur. Appl.*, vol. 50, Feb. 2020, Art. no. 102428.

[45] X. Wang and Y. Su, "Image encryption based on compressed sensing and DNA encoding," *Signal Process., Image Commun.*, vol. 95, Jul. 2021, Art. no. 116246.

[46] S. Zhang and L. Liu, "A novel image encryption algorithm based on SPWLCM and DNA coding," *Math. Comput. Simul.*, vol. 190, pp. 723–744, Dec. 2021.

[47] D. Wei and M. Jiang, "A fast image encryption algorithm based on parallel compressive sensing and DNA sequence," *Optik*, vol. 238, Jul. 2021, Art. no. 166748.

[48] S. Kumar, B. Panna, and R. K. Jha, "Medical image encryption using fractional discrete cosine transform with chaotic function," *Med. Biol. Eng. Comput.*, vol. 57, no. 11, pp. 2517–2533, Nov. 2019.

[49] A. Banu S and R. Amirtharajan, "A robust medical image encryption in dual domain: Chaos-DNA-IWT combined approach," *Med. Biol. Eng. Comput.*, vol. 58, no. 7, pp. 1445–1458, Jul. 2020.

[50] P. T. Akkasaligar and S. Biradar, "Selective medical image encryption using DNA cryptography," *Inf. Secur. J., Global Perspective*, vol. 29, no. 2, pp. 91–101, Mar. 2020.

[51] D. Ravichandran, A. Banu S, B. K. Murthy, V. Balasubramanian, S. Fathima, and R. Amirtharajan, "An efficient medical image encryption using hybrid DNA computing and chaos in transform domain," *Med. Biol. Eng. Comput.*, vol. 59, no. 3, pp. 589–605, Mar. 2021.

[52] Y. Ding, F. Tan, Z. Qin, M. Cao, K. R. Choo, and Z. Qin, "DeepKeyGen: A deep learning-based stream cipher generator for medical image encryption and decryption," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 9, pp. 4915–4929, Sep. 2022.

[53] L. Min, L. Ting, and H. Yu-Jie, "Arnold transform based image scrambling method," in *Proc. 3rd Int. Conf. Multimedia Technol.*, 2013, pp. 1309–1316.

[54] H. Zhu, C. Zhao, X. Zhang, and L. Yang, "An image encryption scheme using generalized Arnold map and affine cipher," *Optik*, vol. 125, no. 22, pp. 6672–6677, Nov. 2014.

[55] G. S. Chandel and V. Sharma, "X-OR and Arnold cipher based double phase image encryption technique," *Int. J. Emerg. Res. Manage. Technol.*, vol. 4, no. 12, pp. 175–181, 2015.

[56] M. Mishra, P. Mishra, M. C. Adhikary, and S. Kumar, "Image encryption using Fibonacci–Lucas transformation," 2012, *arXiv:1210.5912*.

[57] C. Maiti and B. C. Dhara, "Image encryption with a new Fibonacci transform," in *Proc. 5th Int. Conf. Emerg. Appl. Inf. Technol. (EAIT)*, Jan. 2018, pp. 1–4.

[58] H. Tora, E. Gokcay, M. Turan, and M. Buker, "A generalized Arnold's cat map transformation for image scrambling," *Multimedia Tools Appl.*, vol. 81, pp. 31349–31362, Apr. 2022.

[59] J.-X. Chen, Z.-L. Zhu, C. Fu, H. Yu, and L.-B. Zhang, "An efficient image encryption scheme using gray code based permutation approach," *Opt. Lasers Eng.*, vol. 67, pp. 191–204, Apr. 2015.

[60] R. K. Sinha et al., "Image encryption using modified Rubik's cube algorithm," in *Advances in Computational Intelligence: Proceedings of Second International Conference on Computational Intelligence*. Singapore: Springer, 2020.

[61] R. Vidhya and M. Brindha, "A chaos based image encryption algorithm using Rubik's cube and prime factorization process (CIERPF)," *J. King Saud Univ., Comput. Inf. Sci.*, vol. 34, no. 5, pp. 2000–2016, May 2022.

[62] A. Priya et al., "A novel multimedia encryption and decryption technique using binary tree traversal," in *Proc. 2nd Int. Conf. Microelectron., Comput. Commun. Syst. (MCCS)*. Singapore: Springer, 2019.

[63] P. Biswas, S. Kandar, and B. C. Dhara, "An image encryption scheme using sequence generated by interval bisection of polynomial function," *Multimedia Tools Appl.*, vol. 79, nos. 43–44, pp. 31715–31738, Nov. 2020.

[64] A. Paul and S. Kandar, "Simultaneous encryption of multiple images using pseudo-random sequences generated by modified Newton–Raphson technique," *Multimedia Tools Appl.*, vol. 81, no. 10, pp. 14355–14378, Apr. 2022.

[65] T. Sivakumar and P. Li, "A secure image encryption method using scan pattern and random key stream derived from laser chaos," *Opt. Laser Technol.*, vol. 111, pp. 196–204, Apr. 2019.

[66] T. Sivakumar, M. Pandi, N. S. Madasamy, and R. Bharathi, "An image encryption algorithm with Hermite chaotic polynomials and scan pattern," *J. Phys., Conf. Ser.*, vol. 1767, no. 1, Feb. 2021, Art. no. 012044.

[67] S. K. Das and B. C. Dhara, "An image encryption technique using sine curve," in *Proc. 9th Int. Conf. Adv. Pattern Recognit. (ICAPR)*, Dec. 2017, pp. 1–6.

[68] S. K. Das and B. C. Dhara, "A new image encryption method using circle," in *Proc. 8th Int. Conf. Comput., Commun. Netw. Technol. (ICCCNT)*, Jul. 2017, pp. 1–6.

[69] Z. Tang, Y. Yang, S. Xu, C. Yu, and X. Zhang, "Image encryption with double spiral scans and chaotic maps," *Secur. Commun. Netw.*, vol. 2019, pp. 1–15, Jan. 2019.

[70] R. Vidhya, M. Brindha, and N. A. Gounden, "Analysis of zig-zag scan based modified feedback convolution algorithm against differential attacks and its application to image encryption," *Appl. Intell.*, vol. 50, no. 10, pp. 3101–3124, Oct. 2020.

[71] M. Z. Spivey, "Fibonacci identities via the determinant sum property," *College Math. J.*, vol. 37, no. 4, pp. 286–289, Sep. 2006.

[72] Y. K. Panwar, B. Singh, and V. K. Gupta, "Generalized Fibonacci sequences and its properties," *Palestine J. Math.*, vol. 3, no. 1, pp. 141–147, 2014.
[73] M. Basu and M. Das, "Tribonacci matrices and a new coding theory," *Discrete Math., Algorithms Appl.*, vol. 6, no. 1, pp. 1450008(1)–1450008(17), Mar. 2014.
[74] A. G. Weber. (2006). *The USC-SIPI Image Database: Version 5*. Accessed: Oct. 2022. [Online]. Available: http://sipi.usc.edu/database/
[75] M. Li, M. Wang, H. Fan, K. An, and G. Liu, "A novel plaintext-related chaotic image encryption scheme with no additional plaintext information," *Chaos, Solitons Fractals*, vol. 158, May 2022, Art. no. 111989.
[76] R. E. Boriga, A. C. Dăscălescu, and A. V. Diaconu, "A new fast image encryption scheme based on 2D chaotic maps," *IAENG Int. J. Comput. Sci.*, vol. 41, no. 4, pp. 249–258, 2014.

CHINMAY MAITI received the B.Tech. degree (Hons.) from the University of Calcutta, in 2000, and the M.E. degree (Hons.) from the Bengal Engineering College (DU), Shibpur (currently known as IIEST), in 2002. He was a Lecturer at the College of Engineering Management, Kolaghat, Midnapore, West Bengal, since 2007, then Visiting Faculty at the Jadavpur University from 2007 to 2008, and then Assistant Professor at the College of Engineering Management, in 2008. He is the Present Faculty Member at the Department of Computer Science Engineering.

BIBHAS CHANDRA DHARA received the B.Sc. degree (Hons.) in mathematics and the B.Tech. degree in computer science and engineering from the University of Calcutta, India, in 1997 and 2000, respectively, and the M.Tech. and Ph.D. degrees in computer science from the Indian Statistical Institute, in 2002 and 2008, respectively. Currently, he is a Professor with the Department of Information Technology, Jadavpur University, India. He has published more than a 100 articles, in peer-reviewed journals and proceedings, in the field of image and video compression, pattern recognition, audio classification, deep learning, and information security. He is currently doing research in image processing, computer vision, pattern recognition, machine learning, deep learning, and information security.

SAIYED UMER received the B.Sc. degree (Hons.) in mathematics from Vidyasagar University, India, in 2005, the Master of Computer Applications degree from the West Bengal University of Technology, India, in 2008, the M.Tech. degree from the University of Kalyani, India, in 2012, and the Ph.D. degree in engineering from the Department of Information Technology, Jadavpur University, Kolkata, India, in March 2017. He is currently an Assistant Professor with the Department of Computer Science and Engineering, Aliah University, Kolkata. He was the Research Personnel with the Indian Statistical Institute (ISI), Kolkata, from November 2012 to April 2017. He has published several papers in peer-reviewed international and scientific journals in the field of biometrics, affective computing, big-data research, and computational biology. His research interests include biometrics, computer vision, machine learning, deep learning, and business data analytics.

VIJAYAN ASARI received the bachelor's degree in electronics and communication engineering from the College of Engineering Trivandrum, University of Kerala, India, in 1978, and the M.Tech. and Ph.D. degrees in electrical engineering from the Indian Institute of Technology Madras (IIT Madras), in 1984 and 1994, respectively. He is currently a Professor in electrical and computer engineering and the Ohio Research Scholars Endowed Chair of Wide Area Surveillance with the University of Dayton, Dayton, OH, USA. He is the Director of the University of Dayton Vision Laboratory (Center of Excellence for Computational Intelligence and Machine Vision). He was a Professor in electrical and computer engineering with Old Dominion University (ODU), Norfolk, VA, USA, until January 2010, where he was also the Founding Director of the Computational Intelligence and Machine Vision Laboratory (ODU Vision Laboratory). His Ph.D. dissertation was titled "Multiple-Valued Logic Neural Network Architectures and Algorithms for Prediction, Classification and Recognition of Multi-Valued Patterns" under the guidance of Dr. C. Eswaran with IIT Madras. His research interests include signal processing, image processing, computer vision, pattern recognition, machine learning, deep learning, artificial neural networks, brain-machine interface, and high-performance embedded systems.